PhonexCoin (PHX): A Central Bank Digital Currency

Aditya Mishra

github.com/aam-007/PHX

Abstract. PhonexCoin (PHX) presents a novel central bank digital currency (CBDC) implementation that combines blockchain technology with active monetary policy tools. The system addresses key challenges in digital currency implementation, including price volatility, monetary policy transmission, and transaction finality. PHX introduces an algorithmic price oracle with integrated risk assessment, quantitative easing/tightening mechanisms, and controlled inflation minting. Through a dual-interface architecture (central bank console and client wallet) sharing a unified price data layer, PHX maintains a target price peg of 1 PHX = 100 USD while allowing market-driven price discovery within controlled bounds.

1. Introduction

Digital currencies have emerged as potential successors to traditional fiat systems, yet face significant challenges including price volatility [1], lack of monetary policy tools, and uncertain regulatory frameworks. Existing cryptocurrencies either embrace complete decentralization at the cost of price stability or implement centralized controls that undermine trust less operation.

PhonexCoin addresses these limitations through a hybrid approach that maintains blockchain transparency while incorporating central bank oversight. The system implements:

- A price stabilization mechanism with algorithmic risk assessment
- Active monetary policy tools (QE/QT operations)
- Controlled inflation with cooldown periods
- Real-time market analytics and crash detection
- Shared data persistence between central bank and client interfaces

This paper describes the complete technical implementation of PHX, from smart contract design to market operation algorithms.

2. Design Principles

PHX adheres to the following design principles:

2.1 Price Stability: The system maintains a target price peg of 1 PHX = 100 USD through algorithmic adjustments and central bank interventions.

2.2 Monetary Sovereignty: The central bank retains ability to conduct conventional monetary operations including quantitative easing, tightening, and controlled inflation.

2.3 Transparency: All transactions and central bank operations are recorded on-chain and visible to all participants.

2.4 Risk Management: Integrated risk assessment algorithms monitor concentration, velocity, and large transfer risks to prevent market manipulation and crashes.

2.5 Operational Efficiency: The system processes transactions with zero fees while maintaining adequate security through proof-of-work consensus on the underlying Ethereum network.

3. System Architecture

3.1 Smart Contract Layer

PHX implements two primary smart contracts:

PhonexCoin Contract: An ERC-20 compliant token with extended monetary policy functions.

```
contract PhonexCoin is ERC20, Ownable {
    address public treasury;
    uint256 public constant INFLATION_BASIS_POINTS = 250;
    uint256 public lastMintTime;

    function mintAnnualInflation() external onlyOwner;
    function emergencyMint(uint256 amount) external onlyOwner;
}
```

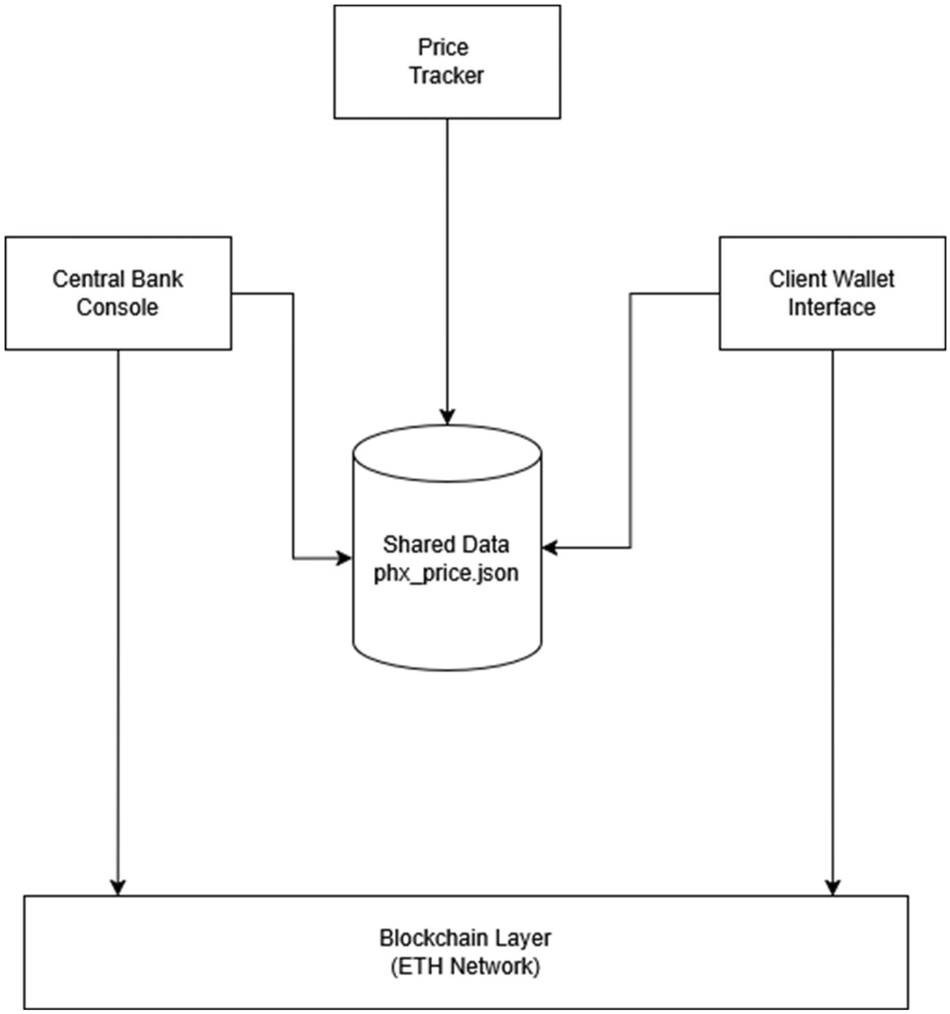
PhonexCentralBank Contract: Monetary policy execution mechanism.

```
contract PhonexCentralBank {
    PhonexCoin public phx;
    address public governor;

    function inject(address to, uint256 amount) external onlyGovernor;
    function absorb(address from, uint256 amount) external onlyGovernor;
}
```

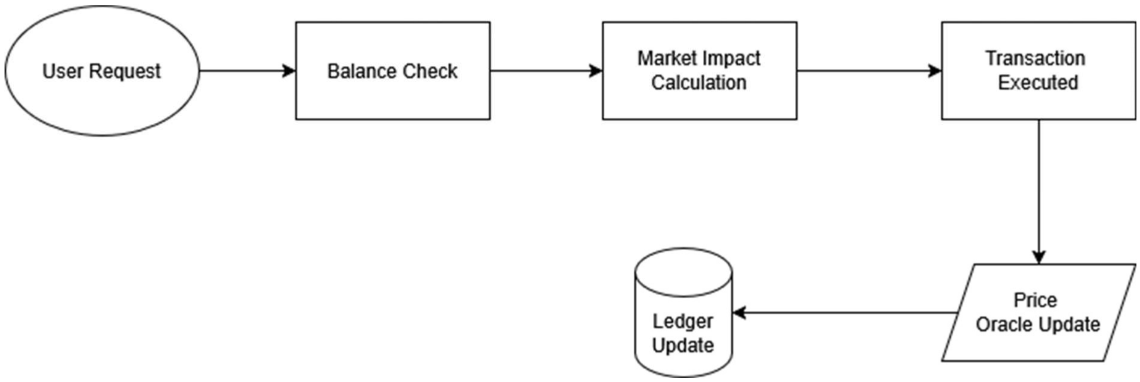
3.2 Application Layer Architecture

The system employs a dual-interface architecture:



3.3 Transaction Flow

Transactions follow standard ERC-20 transfer patterns with additional price impact assessment:



4. Cryptography and Algorithms

4.1 Price Oracle Algorithm

The PHX price oracle calculates real-time prices using multiple market factors:

Let P_t denote the price at time t, and let $P_{\text{base}} = 100$ USD be the target price.

$$P_t = P_{base} \cdot (1 + F_{volume}) \cdot (1 + F_{transactions}) \cdot (1 + F_{health}) \cdot (1 - R_{concentration}) \cdot (1 - R_{velocity}) \cdot (1 - R_{large}) \cdot (1 + M_{momentum}) \cdot (1 + \epsilon)$$

Where**:**

$$\begin{aligned}
F_{\text{volume}} &= 0.02 \cdot \log_{10}(V_{24h} + 1), \\
F_{\text{transactions}} &= 0.015 \cdot \log_{10}(N_{\text{tx}} + 1), \\
F_{\text{health}} &= 0.01 \cdot \min\left(\frac{V_{\text{total}}}{10000}, 2\right), \\
R_{\text{concentration}} &= 0.1 \cdot G, \text{(Gini coefficient)}, \\
R_{\text{velocity}} &= \min(0.1 \cdot V_{\text{velocity}}, 0.5), \\
R_{\text{large}} &= 0.15 \cdot \min\left(\frac{N_{\text{large}}}{50}, 0.3\right), \\
M_{\text{momentum}} &= \text{Price trend factor}, \\
\epsilon &\sim U(-0.04, 0.04) \text{(market noise)}.
\end{aligned}$$

4.2 Risk Assessment Algorithms

Concentration Risk (Gini Coefficient):

$$G = \frac{\sum_{i=1}^{n} \sum_{j=1}^{n} | x_i - x_j |}{2n^2\bar{x}}$$

- x_i= balance of holder i
- n= total number of holders
- $\bar{x}$= average balance

Velocity Risk

$$R_v = \min\left(0.1 \cdot \frac{V_{\text{total}}}{S_{\text{total}}}, 0.5\right)$$

- V_{total}= total transaction volume over the period
- S_{total}= total supply

Large Transfer Detection

$$R_l = 0.15 \cdot \min\left(\frac{\sum_{i=1}^{50} \mathbf{1}_{\text{amount}_i > \text{threshold}}}{50}, 0.3\right)$$

- $\mathbf{1}_{\text{condition}}$= indicator function (1 if true, 0 if false)
- Evaluates the fraction of the largest 50 transactions above a threshold

4.3 Monetary Policy Operations

Quantitative Easing (QE) Impact

$$\Delta P_{\text{QE}} = 0.1 \cdot \frac{A_{\text{QE}}}{10000}$$

- A_{QE}= amount of quantitative easing applied
- ΔP_{QE}= resulting price impact

Quantitative Tightening (QT) Impact

$$\Delta P_{\text{QT}} = -0.15 \cdot \frac{A_{\text{QT}}}{10000}$$

- A_{QT} = amount of quantitative tightening applied
- $\Delta P_{\text{Q T}}$= resulting negative price impact

Inflation Minting

$$A_{\text{inflation}} = \frac{S_{\text{total}} \cdot R_{\text{inflation}}}{10000}$$

- S_{total}= total supply of the token
- $R_{\text{inflation}} = 250$basis points $= 2.5\%$
- $A_{\text{inflation}}$= number of tokens minted due to inflation

5. Security and Privacy

5.1 Double-Spending Prevention

PhonexCoin (PHX) inherits Ethereum's consensus mechanism to achieve transaction finality. The probability of a double-spend attack succeeding is bounded by:

$$P_{\text{double}} \leq \sum_{k=0}^{z} \frac{(q\tau)^k e^{-q\tau}}{k!} [1 - (\frac{p}{q})^{z-k}]$$

where:

- p= fraction of honest mining power
- q= fraction of attacker mining power ($p + q = 1$)
- z= number of confirmations depth [2]
- τ= average block time or interval

5.2 Access Control

The system implements role-based access control:

```
modifier onlyOwner() {
    require(msg.sender == owner, "Unauthorized");
    _;
}

modifier onlyGovernor() {
    require(msg.sender == governor, "Only governor");
    _;
}
```

5.3 Privacy Considerations

While transaction details are public on-chain, the system maintains user pseudonymity. The central bank console provides additional privacy for monetary policy deliberations through local data storage.

6. Incentives and Governance

6.1 Token Economics

Initial Supply: 1,000,000 PHX minted to treasury

Inflation Rate: 2.5% annually, minted after 365-day cooldown

Emergency Mint Cap: 5% of total supply

Transaction Fees: Zero fees to encourage adoption

6.2 Governance Model

Two-tier governance structure:

- Contract Owner: Controls inflation minting and emergency operations
- Central Bank Governor: Executes QE/QT operations

6.3 Monetary Policy Framework

The system implements conventional central banking tools:

- Open Market Operations: QE injections and QT absorptions
- Inflation Targeting: Controlled 2.5% annual expansion
- Lender of Last Resort: Emergency minting facility
- Market Operations: Direct treasury transfers

7. Implementation Details

7.1 Smart Contract Implementation

PhonexCoin Contract Analysis:

The contract extends ERC-20 with monetary policy functions:

```
function mintAnnualInflation() external onlyOwner {
    require(block.timestamp >= lastMintTime + 365 days, "Inflation: too soon");
    uint256 supply = totalSupply();
    uint256 newTokens = (supply * INFLATION_BASIS_POINTS) / 10_000;
    _mint(treasury, newTokens);
    lastMintTime = block.timestamp;
}
```

The inflation mechanism uses basis points (1/100th of 1%) for precision, ensuring consistent 2.5% annual expansion regardless of supply size.

Emergency Mint Circuit Breaker:

```
function emergencyMint(uint256 amount) external onlyOwner {
    uint256 supply = totalSupply();
    require(amount <= (supply * 5) / 100, "Emergency: exceeds 5% cap");
    _mint(treasury, amount);
}
```

This implements a hard cap preventing excessive expansion during crises.

7.2 Price Oracle Implementation

The oracle implements real-time market assessment:

```
async calculateCurrentPrice() {
   const basePrice = this.basePriceUSD;

   // Calculate market factors
   const volumeFactor = Math.log10(this.volume24h + 1) * 2;
   const transactionFactor = Math.log10(this.totalTransactions + 1) * 1.5;
   const concentrationRisk = await this.calculateConcentrationRisk();

   // Apply factors to base price
   let calculatedPrice = basePrice;
   calculatedPrice *= (1 + (volumeFactor * 0.01));
   calculatedPrice *= (1 - (concentrationRisk * 0.1));

   return Math.max(calculatedPrice, basePrice * 0.3); // 70% crash protection
}
```

7.3 Data Persistence Layer

The system uses shared JSON persistence for price data:

```
class PHXPriceOracle {
   constructor(phxContract, availableAccounts) {
      this.sharedDataFile = CONFIG.SHARED_DATA_FILE;
      this.priceHistory = [];
      this.operationHistory = [];
   }

   async persistSharedData() {
      const data = {
         priceHistory: this.priceHistory,
         operationHistory: this.operationHistory,
         lastPrice: this.lastPrice,
```

```
        timestamp: new Date().toISOString()
      };
      fs.writeFileSync(this.sharedDataFile, JSON.stringify(data, null, 2));
  }
}
```

This ensures consistent price data across central bank and client interfaces.

7.4 Transaction Processing Workflow

The system processes transactions with integrated price impact assessment:

```
1. User initiates transfer
2. System checks balance and calculates market impact
3. Transaction executes on blockchain
4. Price oracle updates metrics and recalculates price
5. Transaction ledger updated with real blockchain data
6. Market analytics refreshed across both interfaces
```

8. Performance and Scalability

8.1 Transaction Throughput

PHX inherits Ethereum's transaction capacity, currently ~15-45 transactions per second. The system imposes no additional bottlenecks beyond base layer limitations.

8.2 Block Propagation Efficiency

As an ERC-20 token, PHX transactions benefit from Ethereum's block propagation optimizations including transaction gossip protocols and state pruning.

8.3 Scalability Strategy

The system employs several scaling strategies:

- Layer 2 Readiness: Contract design compatible with rollup solutions
- Batch Processing: Central bank operations can be batched for efficiency
- Data Compression: Price history uses rolling window (100 points) to manage storage

8.4 Latency Analysis

Transaction finality follows Ethereum consensus (~12-15 seconds for 3 confirmations). Price oracle updates occur synchronously with transactions, ensuring real-time market data.

9. Market Analysis and Trading Terminal

9.1 Professional Trading Interface

The PHX ecosystem includes a sophisticated trading terminal that provides real-time market analysis and technical indicators. The terminal implements professional-grade visualization and interactive analytics through a Python-based application that consumes the shared price data layer.

9.2 Technical Implementation

The trading terminal employs a multi-panel architecture with the following components:

Data Persistence Integration:

```
class ProfessionalPHXAnalyzer:
    def __init__(self, data_file="phx_price.json"):
        self.data_file = data_file
        self.price_data = None
        self.price_history = []
        self.timestamps = []
```

The terminal synchronizes with the central bank and wallet interfaces through the shared JSON data file, ensuring consistent price data across all system components.

9.3 Analytical Features

9.3.1 Real-time Statistics Engine: The terminal calculates comprehensive market statistics including:

- Current price, open, high, low values
- Volatility metrics and standard deviation
- Total return calculations
- Price momentum indicators

9.3.2 Technical Analysis Tools:

- Smooth interpolated price curves using cubic splines
- Moving averages (MA5, MA10)
- Bollinger Bands with 2-standard deviation channels

- Returns distribution analysis
- Interactive zoom and pan capabilities

9.3.3Visualization Architecture:

The system employs Matplotlib with custom dark theme styling and smooth data interpolation:

```
def smooth_data(self, x, y, smoothing_factor=300):
    x_numeric = mdates.date2num(x)
    x_smooth = np.linspace(x_numeric.min(), x_numeric.max(), smoothing_factor)
    spl = make_interp_spline(x_numeric, y, k=min(3, len(x)-1))
    y_smooth = spl(x_smooth)
    return mdates.num2date(x_smooth), y_smooth
```

9.4 Interactive Capabilities

The terminal supports advanced user interactions:

- **Scroll-wheel zoom** for detailed price examination
- **Right-click panning** for temporal navigation
- **Double-click reset** to return to default view
- **Real-time data refresh** synchronized with blockchain transactions

9.5 Risk Visualization

The system provides integrated risk assessment visualization:

- Concentration risk indicators through distribution analysis
- Volatility bands and price deviation metrics
- Crash probability indicators based on market factors
- Historical performance analytics with smooth curve rendering

9.6 Integration with PHX Ecosystem

The trading terminal completes the tri-interface architecture:

- **Central Bank Console**: Monetary policy execution
- **Client Wallet**: User transactions and balance management
- **Trading Terminal**: Market analysis and professional analytics

All components share the unified data layer (phx_price.json), ensuring consistent market data and risk assessment across the ecosystem.

The terminal serves as both a monitoring tool for central bank operations and an analytical platform for market participants, providing transparent access to the same market data and risk metrics used by monetary authorities.

10. Conclusion and Future Work

PhonexCoin demonstrates a practical implementation of a central bank digital currency with active monetary policy tools. The system successfully balances price stability objectives with market-driven price discovery through its algorithmic oracle mechanism.

<u>Key innovations include</u>:

- Unified price data layer between central bank and client interfaces
- Integrated risk assessment with crash protection mechanisms
- Transparent monetary policy execution on-chain
- Zero-fee transactions with maintained security

10.1 Limitations

Current limitations include:

- Dependence on Ethereum base layer scalability
- Limited privacy for transaction participants

10.2 Future Directions

Future work will focus on:

- Cross-chain Implementation: Multi-chain deployment for redundancy
- Enhanced Privacy: Zero-knowledge proofs for transaction privacy
- Decentralized Governance: DAO-based monetary policy voting
- Predictive Analytics: Machine learning for price forecasting
- Regulatory Compliance: Integrated KYC/AML frameworks

PhonexCoin provides a foundation for the next generation of digital currencies that combine the benefits of blockchain technology with responsible monetary policy management.

References

[1] Nakamoto, S. "Bitcoin: A Peer-to-Peer Electronic Cash System," 2008.

[2] Rosenfeld, M. "Analysis of Hashrate-Based Double Spending," 2014.

[3] Buterin, V. "Ethereum White Paper," 2014.

[4] Gervais, A. et al. "On the Security and Performance of Proof of Work Blockchains," 2016.