Stellalpha

High-Fidelity Non-Custodial Execution Layer for Solana

Whitepaper

Version 1.3
May 2026

www.stellalpha.xyz

Contents

Abstract

Stellalpha is an intent-based, non-custodial capital allocation layer on Solana. Moving beyond naive wallet mirroring, Stellalpha replicates "trade intent" by automatically adjusting execution based on relative portfolio weights and curated strategy models.

Built on a low-latency dual-ingestion engine (Yellowstone gRPC + Helius) and a custom Rust parser, the protocol provides the secure infrastructure needed for stablecoin capital to follow high-signal on-chain traders without surrendering custody. User funds are secured in Program-Derived Addresses (PDAs) governed by an Anchor-based smart contract, ensuring that users retain full ownership and control of their assets while delegating only specific execution permissions.

By synthesizing curated trader signals with proprietary execution models, Stellalpha enables institutional-grade capital allocation to be executed with sub-second latency, providing a professional and transparent gateway to high-performance Solana strategies.

1 Introduction

Decentralized finance (DeFi) has entered an era where speed and precision are the primary determinants of alpha. However, for most users, capturing high-performance on-chain strategies remains a manual and risky endeavor. Traditional "copy-trading" has historically relied on naive wallet mirroring or custodial key-sharing, both of which are fundamentally incompatible with secure institutional-grade capital management.

Stellalpha represents a paradigm shift in automated capital allocation. It is an intent-based, non-custodial layer built on Solana that moves beyond simple mirroring to replicate the *strategic intent* of proven performers.

By analyzing the relative conviction and portfolio weighting of "Star Traders," the protocol enables stablecoin capital to flow into high-signal opportunities with the security of on-chain vault enforcement.

Stellalpha bridges the gap between sophisticated trading and secure capital management through:

- **Intent-Based Replication**: Automatically adjusting execution based on relative portfolio weights rather than raw token amounts.
- **Non-Custodial Sovereignty**: Users retain full control over their funds in segregated Program-Derived Addresses (PDAs).
- **High-Fidelity Infrastructure**: A dual-ingestion engine (Yellowstone gRPC + Helius) combined with a custom Rust parser for sub-second execution latency.
- **Strategy Synthesis**: A curated registry of proprietary execution models designed to match specific trader styles and risk profiles.

Protocol Ecosystem:

Execution Layer: https://github.com/akm2006/stellalpha

Vault Infrastructure: https://github.com/akm2006/stellalpha_vault

2 Problem Statement

1. **Manual Monitoring is Impractical**: Cryptocurrency markets operate 24/7 with high volatility. For retail traders, manually tracking and mirroring the trades of professional "alpha" wallets is physically impossible without automated tools, leading to missed opportunities and suboptimal entry prices.
2. **Custodial Risk in Centralized Platforms**: Traditional copy-trading platforms (e.g., eToro, Bybit) require users to deposit funds into centralized exchange wallets. This exposes users to counterparty risks, insolvency events (e.g., FTX), and potential withdrawal freezes.
3. **Key-Sharing Risks in Botting**: Existing on-chain automation tools often require users to export their private keys or API keys to third-party servers. This creates a massive security vulnerability, where a single server breach could result in total loss of user funds.
4. **High Costs on EVM Chains**: Ethereum-based solutions suffer from high gas fees and slow finality. This makes high-frequency copy-trading prohibitively expensive for smaller portfolios, as transaction costs erode potential profits.
5. **Lack of Trustless Standards**: There is currently no standardized, trustless protocol for on-chain copy trading that guarantees execution transparency without relying on opaque off-chain logic or custodial bridges.
6. **Convergent Mirroring and Liquidity Crushing**: Naive wallet mirroring protocols typically copy raw token amounts or fixed percentages without regard for the follower's relative liquidity or the trader's total conviction. When multiple followers attempt to mirror a single high-signal trade simultaneously, they create an artificial demand spike that "crushes" available liquidity, resulting in massive slippage for the followers and eroding the original strategy's alpha.

3 Mathematical Execution Models

Stellalpha's core innovation is the decoupling of the *signal* from the *execution size*. While legacy systems use a static multiplier, Stellalpha utilizes five distinct mathematical models to synthesize the leader's intent into an optimized follower execution.

3.1 Model 1: Balanced Hybrid (Protocol Baseline)

The Balanced Hybrid model is designed for maximum strategy preservation with a strict safety envelope. It calculates the execution size by taking the leader's relative conviction and applying it within a user-defined liquid cash limit.

$$\text{Size}_{\text{USD}} = \min\left(\text{Limit}_{\text{USD}}, \text{Limit}_{\text{USD}} \times \frac{\text{LeaderTradeValue}_{\text{USD}}}{\text{LeaderAvailableCash}_{\text{USD}}}\right)$$

Best For: High-volatility meme-cycles and long-tail alpha where capital preservation is paramount.

3.2 Model 2: Trader Ratio (Direct Conviction)

This model directly replicates the leader's conviction signal. It scales the trade size based on the ratio of the follower's balance to the leader's balance at the moment of execution.

$$\text{Size}_{\text{Follower}} = \text{Size}_{\text{Leader}} \times \frac{\text{Balance}_{\text{Follower}}}{\text{Balance}_{\text{Leader}}}$$

Best For: Mirroring institutional-grade traders or portfolios with similar liquidity profiles.

3.3 Model 3: Fixed % of Free Cash (Compounding)

An dynamic model that allocates a consistent percentage of the follower's current available liquid capital to every buy signal.

$$\text{Size}_{\text{USD}} = \text{FreeCash}_{\text{Follower}} \times \text{ConfiguredPercentage}$$

Best For: High-frequency traders where maintaining a consistent presence in every trade is the primary edge.

3.4 Model 4: Fixed % of Starting Funds (Stable)

A conservative model that uses a percentage of the follower's initial starting capital as the base for every execution, regardless of current PnL.

$$\text{Size}_{\text{USD}} = \text{InitialCapital}_{\text{Follower}} \times \text{ConfiguredPercentage}$$

Best For: Stable participation and risk-parity strategies where drawdowns should not decay the entry size.

3.5 Model 5: Trader Buy % With Cap (Risk-Bounded)

Replicates the leader's relative buy size but enforces a hard USD cap on any single entry to prevent over-exposure to illiquid assets.

$$\text{Size}_{\text{USD}} = \min\left(\text{HardCap}_{\text{USD}}, \text{LeaderTradeValue}_{\text{USD}} \times \text{ScalingFactor}\right)$$

Best For: Following traders with aggressive conviction who may occasionally size into positions that exceed the follower's risk tolerance.

4 Protocol / Architecture Overview

Stellalpha is architected on the Solana blockchain using the Anchor Framework, optimized for sub-second execution and secure non-custodial capital management.

4.1 Non-Custodial Vault Enforcement

The foundation of the protocol is the User Vault system. Unlike custodial platforms, Stellalpha utilizes Program Derived Addresses (PDAs) to ensure that user capital never leaves the cryptographic control of the owner.

$$\text{UserVault} = \text{seeds}[b\text{"user\_vault\_v1"}, \text{user\_pubkey}]$$

This architecture guarantees:

- **Sovereignty**: The vault is program-owned but user-governed. Only the user can authorize withdrawals or re-allocate capital between strategy models.
- **Security**: Funds are secured within the Solana Token Program's safety guarantees. The protocol can only execute trades through user-initialized vaults with predefined constraints.
- **Constraint Enforcement**: The smart contract enforces strict execution rules, preventing unauthorized transfers or deviations from the selected copy model.

4.2 Low-Latency Dual-Ingestion Engine

Execution speed is critical for minimizing slippage and maximizing replication accuracy. Stellalpha employs a high-performance dual-ingestion system to monitor the Star Trader network.

- **Yellowstone gRPC**: Utilized for ultra-low latency, real-time transaction streaming directly from the validator network. This ensures that trade signals are captured within milliseconds of their inclusion in a block.
- **Helius Webhooks**: Provides a robust secondary ingestion layer, ensuring redundant coverage and reliable transaction confirmation data.
- **Custom Rust Parser**: A high-performance off-chain processing engine built in Rust that decodes raw Solana transaction data. It identifies complex trade intents (multi-hop swaps, liquidity provision) and converts them into actionable signal packets.

4.3 Intent-Based Strategy Synthesis

Stellalpha moves beyond naive mirroring by synthesizing the *strategic intent* of a trader. The protocol calculates the relative portfolio weight of every signal to determine the optimal allocation for the follower.

- **Conviction Weighting**: Buy orders are sized based on the ratio of the trade value to the leader's available liquid equity.

$$\text{ExecutionRatio} = \frac{\text{LeaderTradeValue}_{\text{USD}}}{\text{LeaderTotalBuyingPower}_{\text{USD}}}$$

- **Dynamic Normalization**: Execution is automatically adjusted based on the follower's specific risk settings and vault balance.

4.4 The Strategy Registry

Users can select from a curated list of proprietary execution models, each designed for specific trader profiles:

1. **Balanced Hybrid**: Protocol baseline that sets a safety envelope for capital while maintaining high conviction signals.
2. **Trader Ratio**: Direct replication of the leader's conviction signals based on liquid balance.
3. **Fixed % Free Cash**: Optimized for compounding and high-frequency liquidity preservation.

4.5 Jupiter CPI Execution Engine

The protocol utilizes Jupiter's Cross-Program Invocation (CPI) interface for optimal routing and settlement.

1. **Route Computation**: The ingestion engine calculates the optimal execution path via the Jupiter API to minimize price impact.
2. **CPI Execution**: The Stellalpha program invokes the Jupiter CPI to settle the swap. The `TraderState` PDA signs for the transaction via `invoke_signed`, ensuring non-custodial integrity throughout the entire swap lifecycle.

5 Execution Safety & Risk Mitigation

Solana's high-throughput environment requires rigorous on-chain and off-chain safety checks to ensure that follower execution matches the strategic intent without exposure to extreme market volatility or malicious actors.

5.1 Programmatic Slippage Guardrails

The Stellalpha smart contract enforces deterministic slippage checks for every swap. Before invoking the Jupiter CPI, the program validates that the resulting token amount meets the minimum threshold defined by the follower.

$$\text{MinOut} = \text{ExpectedOut} \times (1 - \text{SlippageTolerance})$$

If the market moves against the trade during the sub-second ingestion-to-execution window, the transaction reverts on-chain, protecting the user's capital from suboptimal entry prices.

5.2 Anti-MEV Strategy (Jito Integration)

Followers are particularly vulnerable to sandwich attacks and front-running in a public copy-trading environment. To mitigate this, Stellalpha utilizes Jito Bundles and Tip-Cap logic.

- **Private Bundle Submission**: Strategic execution packets are submitted as atomic bundles to Jito validators, bypassing the public mempool.
- **Dynamic Priority Fees**: The execution engine automatically calculates and attaches the optimal Jito Tip to ensure high inclusion rates even during periods of heavy network congestion.

5.3 Execution Latency Benchmarks

The dual-ingestion engine is calibrated for sub-second precision. By utilizing Yellowstone gRPC, Stellalpha captures signals at the moment of block inclusion.

Pipeline Stage	**Mechanism**	**Target Latency**
Signal Capture	Yellowstone gRPC	¡ 50ms
Intent Decoding	Rust Custom Parser	¡ 100ms
Route Optimization	Jupiter SDK	¡ 200ms
On-Chain Settlement	Solana RPC / Jito	¡ 400ms
Total Pipeline		**¡ 750ms**

Table 1: System Latency Benchmarks

5.4 Vault Integrity & PDA Security

Stellalpha utilizes a "Restricted Signature" model. While the protocol is authorized to sign for swaps via the user's initialized 'TraderState' PDA, it cannot sign for withdrawals or transfers to external wallets. Only the original user wallet (the PDA seed) has the authority to reclaim capital, ensuring that a protocol-level breach cannot result in the theft of user funds.

6 Simulation & Functional Validation

To ensure users can verify the integrity of the protocol before committing capital, Stellalpha provides a dedicated functional validation layer: the Demo Vault.

6.1 The Demo Vault Architecture

The Demo Vault is a complete mirroring of the production execution environment, but it operates with virtualized capital. Every demo vault is pre-funded with **$1,000.00 virtual USD**, which can be allocated across curated Star Traders.

- **Logic Parity**: The same Rust parser and dual-ingestion engine drive both real and demo executions.
- **Synthesized PnL**: Results are calculated in real-time based on live market price marks (via Jupiter API), providing a high-fidelity representation of expected performance.
- **Risk-Free Onboarding**: Users can test complex combinations of copy models and trader conviction signals without gas fees or financial risk.

6.2 Curation Criteria for Star Traders

Stellalpha's performance is tied to the quality of its signal sources. The protocol utilizes a multi-metric filtering system to whitelist traders for the "Star Trader" registry.

- **Strategic Consistency**: Traders must demonstrate consistent strategic intent (e.g., focus on specific sectors or token classes) rather than random noise.
- **Max Drawdown**: Only wallets with historical drawdowns within acceptable protocol limits are whitelisted.
- **Liquidity Compliance**: Trader conviction must align with available DEX liquidity to prevent the "Convergent Mirroring" failure identified in Section 3.

7 Tokenomics

7.1 Economic Model

The native Stellalpha token is designed to align protocol growth with secure, high-performance capital allocation. The token economy features a fixed-supply model to ensure long-term value preservation and governance integrity.

7.2 Protocol Utility & Governance

- **Staking & Performance Yield**: Stakeholders receive a 30% share of protocol performance fees generated from profitable copy-trading vaults.

- **Strategic Governance**: Token holders govern the Strategy Registry, including the whitelisting of Star Traders and the optimization of core execution models.
- **Fee Normalization**: Holding specific token thresholds provides dynamic discounts on execution and relay fees.
- **Priority Signal Access**: Tokens grant priority access to capped, high-signal Star Trader vaults during periods of high demand.

8 Roadmap

The Stellalpha development cycle is focused on achieving maximum execution stability while maintaining non-custodial sovereignty.

8.1 Development Phases

- **Phase 01: Demo Environment (Completed)**: Full virtual-capital simulation proving copy models, sub-second latency, and intent replication logic without financial risk.
- **Phase 02: Strategy Synthesis (Active)**: Integration of the curated Star Trader registry with proprietary execution models and risk-optimized sizing parameters.
- **Phase 03: Non-custodial Pilot (Upcoming)**: Direct live execution of curated signals through secure, user-controlled vaults and precise allocation limits.
- **Phase 04: Protocol Expansion (Research)**: Deep infrastructure integration for complex DeFi interactions and the launch of a decentralized logic registry for multi-chain replication.

9 Competitive Analysis

Stellalpha's positioning as an intent-based execution layer distinguishes it from legacy copy-trading platforms. The following analysis summarizes the technical moats provided by the protocol:

Feature	Stellalpha	Centralized Platforms	API / TG Bots
Custody	Non-Custodial (PDA)	Custodial (Exchange)	Exchange / Shared Keys
Security Model	PDA Vault Enforcement	Trust-Based	Private Key Risk
Signal Type	**Intent-Based**	Naive Mirroring	Raw Mirroring
Execution	On-Chain (Jupiter)	Opaque Off-Chain	Semi-Opaque API
Transparency	100% Verified	Low (Opaque)	Variable
Latency	**Sub-800ms**	High (API Overhead)	Variable (Polling)

Table 2: Technical comparison of Stellalpha with legacy alternatives

Stellalpha's primary competitive advantage lies in the synthesis of *strategic intent*. While legacy systems simply copy raw token amounts, Stellalpha normalizes signals based on relative conviction and portfolio weighting, providing institutional-grade risk management that is otherwise unavailable in the retail market.

10 Team & Advisors

Aakash Mandal (Founder, Developer): Founder & builder working on blockchain, automation, and decentralized systems. Lead developer of Stellalpha, focused on non-custodial trading infrastructure.

Manobendra Mandal (Co-founder, Developer): Co-founder & developer with strong Web3 and full-stack experience. Hackathon-driven builder focused on smart contracts, dApps, and scalable systems.

11 Legal / Compliance Notes

This document is provided for informational purposes only and does not constitute a solicitation for investment or financial advice. The Stellalpha protocol operates in a non-custodial manner, and users are solely responsible for their capital allocation and risk management within the Solana ecosystem.