DRAFT FOR OPEN COMMUNITY REVIEW AND SUBJECT TO CHANGE

Nubit: Trustless World Computer for Bitcoin

Enabling Open Scaling Through Bitcoin-Native Technology

The Nubit Team

Sept 30, 2024

DRAFT FOR OPEN COMMUNITY REVIEW AND SUBJECT TO CHANGE

The Nubit Whitepaper

Computing has undergone a remarkable evolution over the past decades. Initially, it was characterized by proprietary applications running on individual local machines, which limited accessibility and collaboration. The 1990s and early 2000s saw the rise of open-source software, democratizing access and allowing users to run free applications on their personal computers. The subsequent shift to massive, multi-user applications hosted on proprietary cloud platforms offered unprecedented connectivity and convenience but introduced new concerns about centralization and control by dominant entities.

In recent years, the advent of blockchain technology has opened the door to decentralized networks, leading to a trustless, decentralized computing platform. Ethereum [1] played a pivotal role in this evolution by introducing the idea of a "world computer," enabling programmable smart contracts on a global scale. This innovation allowed developers to build decentralized applications (dApps) without centralized control, fostering a new era of openness and user empowerment.

However, current implementations face persistent challenges in scalability, security, and true decentralization. Issues such as network congestion, high transaction fees, and vulnerabilities in smart contract code have highlighted the limitations of existing platforms. The need for a robust, universally accessible, and genuinely trustless computing platform remains unmet, particularly as we move toward a new era of autonomous interactions between humans and machines.

Bitcoin [2], the most secure and decentralized blockchain network, is uniquely positioned as the ideal foundation for this next-generation world computer. Its Proof-of-Work consensus mechanism and extensive network of nodes provide unparalleled resistance to attacks, while its simplicity and robustness have garnered trust from users and institutions globally. Leveraging Bitcoin's unmatched security and universal accessibility, enhancing its programmability and data capabilities can unlock a transformative era—**Bitcoin 3.0**—where **Mankind-to-Machine (M2M) economies** flourish, enabling seamless automation and interaction between humans and autonomous systems.

Despite its strengths, Bitcoin's network faces significant limitations that must be addressed to achieve this vision of a trustless world computer:

- **Limited Programmability**: Bitcoin's intentionally restrictive scripting language limits its ability to support complex computations and advanced smart contract functionalities.
- **Storage Constraints**: With a design focused on security, Bitcoin offers limited on-chain storage and high costs for data-intensive applications.
- **Scalability Issues**: High transaction fees and slow confirmation times hinder the development of scalable applications.

To address these challenges, Nubit introduces rigorous **formal methods** for unparalleled automation and security guarantees, ensuring reliability and correctness in a trustless environment. By extending Bitcoin's programmability and data capabilities, Nubit enables the development of scalable and secure applications, paving the way for **Bitcoin 3.0** and unlocking its potential as the backbone of the next era in decentralized computing.

Overview of Nubit

Nubit envisions a future where Bitcoin becomes the backbone of a trustless world computer, empowering developers and users with advanced programmability and scalable data storage. Our mission is to enhance Bitcoin's capabilities while preserving its unparalleled security and decentralization, unlocking its full potential for global, trustless computing.

While Bitcoin is a highly secure and decentralized network, its architecture presents significant challenges in programmability and data storage. Bitcoin's scripting language is intentionally restrictive, offering only basic operations for validating transactions. This limitation, designed for security, prevents the development of complex dApps and smart contracts on Bitcoin. In contrast, Ethereum processes over 1 million smart contract calls daily, compared to Bitcoin's lack of Turing-complete programmability, which leaves developers constrained to simple, predefined functionalities.

In terms of storage, Bitcoin's on-chain data capacity is significantly limited. With a maximum block size of approximately 4 MB and the blockchain growing by about 60 GB per year, the network faces high fees and inefficiency for data-intensive applications. For comparison, Ethereum and other blockchains that support decentralized storage allow for more scalable solutions, but often at the cost of decentralization and security. These constraints hinder Bitcoin's potential to support modern decentralized applications and global-scale solutions.

To overcome these challenges, Nubit introduces a comprehensive infrastructure (shown in Figure 1) to transform Bitcoin into a trustless world computer:

Figure 1: An overview of Nubit

- **BitVM for Advanced Bitcoin Programmability**: A comprehensive virtual machine that enables Turing-complete programmability on Bitcoin, allowing for complex computations and smart contracts. This innovation empowers developers to build dApps directly on the Bitcoin network.
- **Bitcoin-native Data Availability (DA)**: A solution to Bitcoin's storage limitations by enabling scalable data capacities with off-chain storage and on-chain verification. This ensures data integrity while significantly reducing the burden on the blockchain.
- **Goldinals**: A trustless framework for fungible tokens on Bitcoin, leveraging zero-knowledge proofs (ZKPs) and state commitments for full on-chain verification. Goldinals enables secure and decentralized applications like tokenization, stablecoin issuance, and DAO governance while preserving Bitcoin's principles of decentralization and security.
- **Formal Verification** for Security: Applies rigorous formal methods to ensure the correctness and security of Nubit's stack. By leveraging decades of expertise in mathematical reasoning, it minimizes risks, validates program behavior, and protects against vulnerabilities, enabling a robust and trustless ecosystem.

Technical Architecture

Nubit's architecture is designed to overcome Bitcoin's inherent limitations in programmability and storage while maintaining its unmatched security and decentralization. At its core are two transformative components: the **BitVM Compiler**, which enables advanced programmability by translating high-level code into optimized Bitcoin-compatible scripts, and **Bitcoin-native Data Availability (DA)**, which scales storage by leveraging off-chain data with on-chain verification. Together, these components provide a robust foundation for building dApps and managing data-intensive workloads on the Bitcoin network, unlocking its potential as a trustless world computer.

BitVM Compiler

Turning high-level code into Bitcoin script is particularly challenging. Bitcoin's scripting language is not like typical programming languages; it's limited, not "Turing complete" (meaning it can't handle all types of computations), and has strict limits on space. This makes translating even simple programs a lot more complex than it would be on other blockchains, like Ethereum.

For example, a Groth16 Verifier, in its high-level form, only has a few hundred lines of code. But when we compile it down to Bitcoin script, it explodes in size—potentially taking up millions of lines of Bitcoin script code and several gigabytes in memory! This is a huge problem, especially considering that Bitcoin's blocks are limited to 4 MB each. Trying to fit something this large into Bitcoin's blocks is akin to trying to fit an elephant into a small car—the sheer size and complexity make it unfeasible under current network constraints. This massive size and verbosity make BitVM programs impractical without serious adjustments.

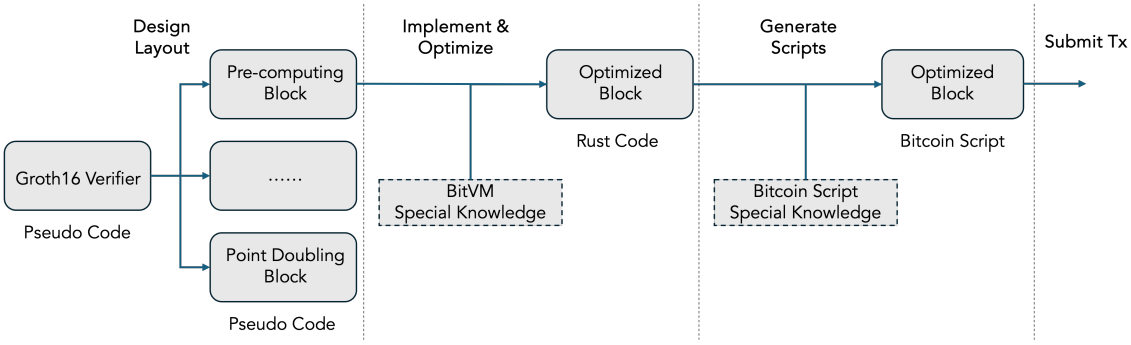

Figure 2: The staging pipeline of Nubit's BitVM compiler

To address the size and complexity challenges of running advanced computations on Bitcoin, we propose a fully automatic **staging compilation pipeline** (Figure 2). This strategy simplifies the problem by breaking it into manageable steps, optimizing both size and performance while ensuring compatibility with Bitcoin's constraints. Here's how the staging compilation process works:

- **Part I: Layout Process: Rearranging the Elephant.** The first step in staging compilation involves transforming the original program into a form that fits within Bitcoin's size limitations.
  - **Representation as a Computational Graph**: The original program, written in a Rust-like language, is represented as a computational graph (**G**), which captures its operations and data flows.
  - **Reorganization into Bitcoin-Compatible Chunks**: This graph is rearranged into an optimized version (**G'**) designed to fit within Bitcoin's constraints. The key objectives are:
    - **Reuse of Data and Subgraphs**: Identifying and consolidating redundant computations to reduce program size.
    - **Adherence to Bitcoin's 4MB Block Limit**: Breaking down the program into smaller, modular pieces that can fit within the Bitcoin block size limit, ensuring deployability.
  - **Outcome**: The layout step creates a compressed, modular representation of the program that fits Bitcoin's constraints.

- **Part II: Optimization: Trimming the Elephant.** Once the program is broken into manageable pieces, the next step focuses on making each piece as efficient as possible.
  - **Minimizing Computational Costs**: This involves simplifying computations to reduce the number of operations, drawing on cryptographic techniques and insights. For example, repetitive multiplications might be condensed or eliminated.
  - **Enhancing Efficiency**: The goal is to slim down each part of the graph, making it less resource-intensive and faster to execute.
  - **Outcome**: The optimized program is not only smaller but also performs better in terms of speed and cost.
- **Part III: Code Generation: Loading the Elephant for Transport.** The final step translates the optimized computational graph (**G'**) into actual Bitcoin-compatible script code.
  - **Conversion to Bitcoin Instructions**: Each node in the graph is traversed and converted into a sequence of low-level Bitcoin script instructions.
  - **Deployable Code**: The result is a compact, Bitcoin-compatible program ready to run on the blockchain.
  - **Outcome**: The program, once impractical in its original form, is now compact and deployable within Bitcoin's technical limitations.

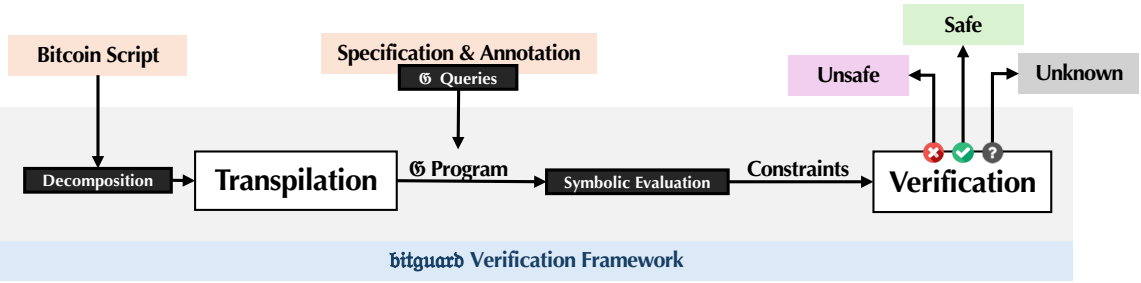

Figure 3: Push-button Verification for BitVM [3]

Part IV: Formal Verification for Security. Subtle mistakes in the compilation of BitVM can lead to catastrophic outcomes, such as the loss of funds or vulnerabilities that malicious actors can exploit. To address these risks, **formal verification** emerges as a powerful tool to validate the correctness of BitVM programs rigorously. We introduce a **formal computational model** capturing the semantics of both BitVM execution and Bitcoin Script. As shown in Figure 3, this model is built upon a **register-based domain-specific language (DSL)** that abstracts away complex stack operations while preserving the semantics of the original low-level code. The DSL facilitates reasoning about program correctness through a higher-level, verification-friendly interface.

- **Lifting to a Higher-Level DSL**: The DSL replaces intricate stack manipulations with simplified, register-based operations. This abstraction significantly enhances the clarity of the program's behavior, making verification more tractable. Complex patterns, such as unrolled loops simulating iterative computations, are represented using **loop invariants** to summarize their behavior concisely.
- **Counterexample-Guided Inductive Synthesis (CEGIS)**: To ensure equivalence between the original Bitcoin Script and the high-level DSL, the system employs a synthesis approach that iteratively generates and verifies candidate programs. This process guarantees that the higher-level representation faithfully preserves the behavior of the low-level implementation.
- **Hoare-Style Verification**: Using the DSL, the framework applies Hoare logic [4] to establish the correctness of BitVM programs. Predefined **preconditions**, **postconditions**, and **loop invariants** ensure that the program's behavior matches its intended specifications, while **verification conditions (VCs)** are systematically generated and checked for consistency using off-the-shelf SMT solvers.

Putting It All Together. Through the staging compilation process—rearranging, optimizing, and generating code—BitVM transforms computationally complex programs into streamlined, Bitcoin-compatible scripts, all powered by decades of our team's expertise in formal verification and security. This innovative approach

makes advanced computations feasible on Bitcoin, overcoming its inherent limitations while maintaining its security and decentralization.

Bitcoin-Native Data Availability

Bitcoin's limited block size restricts data-intensive applications and scalability. Nubit provides a native DA solution that offloads storage off-chain while ensuring data integrity via Bitcoin, enabling advanced applications without compromising decentralization or security. In particular, Nubit DA involves three essential components:

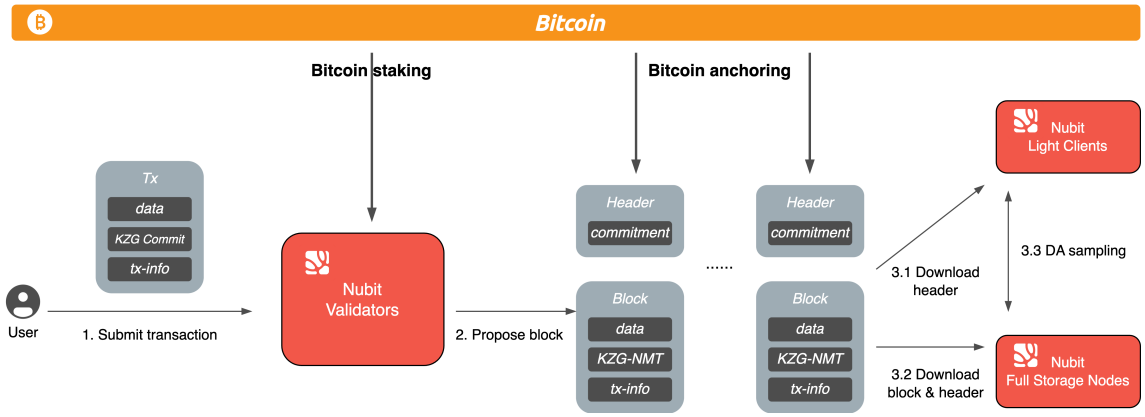

Figure 4: Essential components of Nubit DA [5]

- **Validators**: These nodes operate using a consensus algorithm rooted in NubitBFT and are tasked with proposing blocks and verifying the integrity of blocks and transactions. Nubit is proposing a new consensus mechanism designed to support an exceptionally large validator set (potentially of size 200,000) and facilitate consensus within the chain. Nubit aims to fully inherit Bitcoin security, including economic security, temper resistance, and censorship resistance. It achieves this through the implementation of Bitcoin's native staking and Bitcoin anchoring methods. Further, Nubit explores an efficient CometBFT-based consensus powered by SNARK for signature aggregation, named NubitBFT, bringing an unprecedented decentralized validator network.
- **Full Storage Nodes**: After receiving block data from validators, these nodes are entrusted with the reliable storage of all data. The integrity and availability of stored data are critical, especially given the risks of malicious activities such as data withholding or tampering. To mitigate these risks, Data Availability Sampling (DAS) requests from light clients are employed to verify data availability, ensuring the system's resilience against such threats.
- **Light Clients**: Nubit enables light clients to be part of its consensus protocol, so it is crucial to ensure the data integrity held by those light clients. This is where **Data Availability Sampling (DAS)** comes into play. DAS is a technique that allows participants, like light clients, to verify the presence of block data without needing to download the entire block.
  This is achieved by conducting multiple rounds of random sampling on small portions of the block data. Each successful sampling round increases the likelihood that the data is fully available. Once a predetermined confidence level is reached, the block data is deemed accessible.

Validators, full storage nodes, and light nodes each have specific roles in ensuring data availability through their designated protocols. In particular, by participating in Nubit DA and running a data availability node, individuals can operate either a full node or a light node.

Nubit utilizes a data availability mechanism that combines **KZG commitments** with two-dimensional Reed-Solomon erasure codes. This framework ensures the correctness of RS encoding through validity proofs based on KZG commitments rather than using traditional fraud proofs. Consequently, light nodes can retrieve necessary data through KZG opening proofs instead of Merkle proofs, significantly enhancing retrieval speeds — by up to 10X in typical scenarios and up to 100X in more extreme situations.

Use Cases and Applications

BitVM can power use cases like BTC Layer 2 and Bitcoin-native assets by enabling advanced programmability and verifiable off-chain computation, all while maintaining Bitcoin's decentralized trust model. This allows seamless extensions to Bitcoin's functionality without altering its core protocol. In particular, Goldinals introduces fungible tokens directly on Bitcoin, while BitVM-powered L2 solutions provide off-chain scalability and secure settlement. Together, they preserve Bitcoin's unmatched security and decentralization while expanding its utility for DeFi, tokenization, and advanced applications.

Goldinals: The First Native Asset Protocol on Bitcoin

Goldinals introduces a trust-minimized standard for fungible tokens on Bitcoin, resolving the limitations of earlier solutions like Colored Coins and BRC20, which relied on centralized indexers [6] for token validation. Leveraging **zero-knowledge proofs (ZKPs)** and **state commitments** via BitVM, Goldinals achieves fully on-chain verification for all operations, adhering to Bitcoin's principles of decentralization and security. This framework enables advanced token functionality directly on Bitcoin, avoiding trust assumptions while ensuring scalability and fraud resistance.

As shown in Figure 5, the protocol's operations—**deploy, mint, and transfer**—are executed using a three-phase process: **Prepare, Kickoff, and Confirm**. For instance, during a mint operation:

- **Prepare Phase:** Records parameters such as the mint amount and recipient address in Bitcoin's witness fields.
- **Kickoff Phase:** The operator submits a ZKP proving compliance with the protocol's rules, such as staying within the token's predefined supply limits.
- **Confirm Phase:** Finalizes the operation by updating token balances while incentivizing participants to challenge any fraudulent transactions.

This phased design ensures robust security while minimizing reliance on centralized infrastructure.

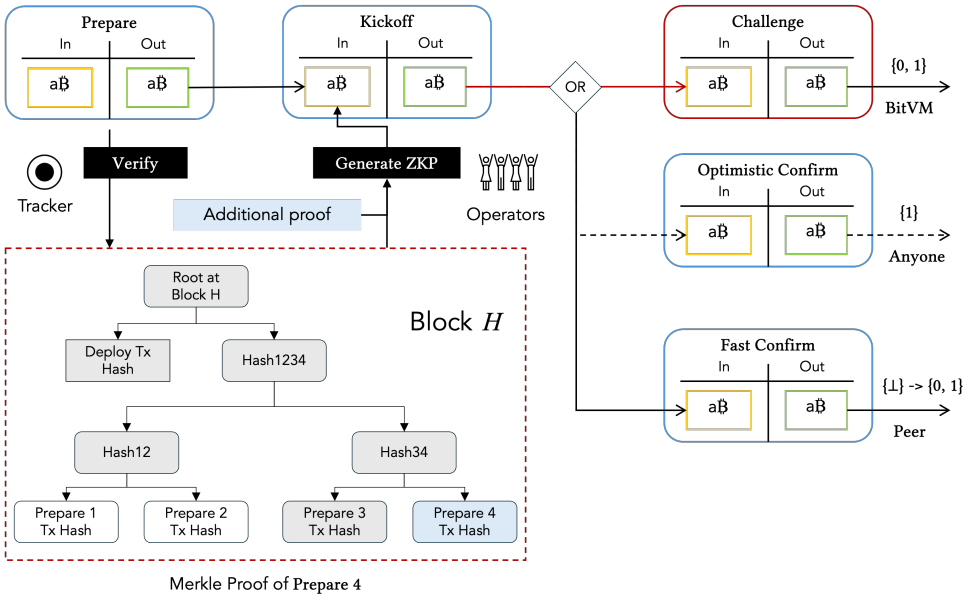

Figure 5: The Prepare, Kickoff, and Confirm process for a mint operation, illustrating the flow of data and verification.

Goldinals uses **state commitments** to minimize dependency on centralized indexers. These commitments rely on a tracker that records the state of token operations off-chain and provides state commitments for verification. Unlike traditional indexers, state commitments are lightweight, enabling scalability without

sacrificing decentralization. Future improvements, such as implementing Bitcoin covenants, could eliminate even these minimal off-chain components, allowing transaction rules to be enforced entirely on-chain, further enhancing the protocol's decentralization and trust minimization.

By extending Bitcoin's functionality with BitVM and trust-minimized mechanisms, Goldinals unlocks a diverse array of applications, including but not limited to:

- **Crowdsales:** Goldinals provides a secure and trust-minimized mechanism for conducting token crowdsales directly on Bitcoin. By setting parameters like mint price and total supply during the deploy operation, deployers can ensure transparency and fairness in token distribution. The Prepare, Kickoff, and Confirm phases prevent malicious behavior, such as token minters withdrawing funds before the sale concludes. With Goldinals, deployers are guaranteed to receive their funds securely once all token supplies are minted and validated, creating a reliable foundation for fundraising on Bitcoin. This approach eliminates trust assumptions, enabling decentralized, tamper-proof crowdfunding for new projects.
- **Stablecoin Issuance:** Goldinals facilitates decentralized stablecoin issuance by leveraging its zero-knowledge proof-based minting process. Deployers can integrate governance mechanisms, such as multi-signature approvals or proof-of-burn requirements, to ensure the stability and legitimacy of the stablecoin. For instance, stablecoins can be issued only after demonstrating that equivalent assets have been locked or burned on another chain. Goldinals' on-chain verification ensures that any minting operation with an invalid proof is nullified, maintaining transparency and preventing unauthorized inflation. This trust-minimized framework empowers the creation of stablecoins on Bitcoin, backed by robust decentralization and security.
- **DAO Governance:** Goldinals enables decentralized, tamper-proof voting for decentralized autonomous organizations (DAOs) using its fungible token framework. By utilizing tokens issued via Goldinals as voting power, DAO members can participate in governance decisions with full transparency. Each transfer of tokens or vote submission is secured through the Prepare, Kickoff, and Confirm phases, ensuring that every action is verifiable and fraud-resistant. Additionally, the use of zero-knowledge proofs can preserve voter privacy by validating votes without revealing individual choices. This makes Goldinals an ideal solution for implementing scalable, secure, and private DAO governance systems on Bitcoin.

Enabling Bitcoin Layer 2 Solutions

Nubit's BitVM introduces an innovative approach to enabling truly Bitcoin-native Layer 2 (L2) solutions by leveraging Bitcoin's robust security and decentralization. Similar to Goldinals' design, BitVM [7] utilizes an **optimistic challenge-response mechanism** and **zero-knowledge proofs (ZKPs)** to extend Bitcoin's capabilities without modifying its core protocol or relying on centralized components. This architecture ensures that Bitcoin remains the ultimate arbiter of trust while enabling scalable and flexible off-chain computation and transactions.

- **Trust-Minimized Settlement:** With BitVM, L2 transactions are anchored to Bitcoin through cryptographic proofs, ensuring that all off-chain operations can be verified and settled trustlessly on the Bitcoin mainnet. Disputes between participants are resolved using BitVM's challenge-response mechanism, where any invalid operation is challenged and nullified by publishing a ZKP on-chain. This eliminates reliance on trusted intermediaries or custodial entities, preserving Bitcoin's decentralized ethos.
- **Off-Chain Computation with On-Chain Security:** BitVM allows for complex computations to occur off-chain, drastically improving scalability while maintaining Bitcoin's security guarantees. The computations are compressed into succinct ZKPs, which are submitted to Bitcoin for validation. This design significantly reduces on-chain data requirements while ensuring that all off-chain actions adhere to the rules of the protocol.
- **Native Data Availability (DA):** Layer 2 solutions often struggle with data availability issues. BitVM addresses this by integrating Bitcoin-native data availability mechanisms, where off-chain data is stored

securely in Nubit DA and referenced on-chain through cryptographic commitments. Participants can retrieve and verify data directly, ensuring integrity without overburdening the main Bitcoin blockchain.

Conclusion

Nubit redefines Bitcoin's role in the blockchain ecosystem by transforming it into a trustless world computer, unlocking the full potential of its unparalleled security and decentralization. By introducing **BitVM** for advanced programmability and **Bitcoin-native Data Availability (DA)** for scalable data solutions, Nubit addresses Bitcoin's inherent limitations in computation and storage. These innovations pave the way for **Bitcoin 3.0**, an era where **Mankind-to-Machine (M2M) economies** flourish, enabling seamless automation and interaction between humans and machines.

With rigorous **formal verification** ensuring security and correctness, Nubit provides the foundation for robust decentralized applications that preserve Bitcoin's trust-minimized principles. From enabling trustless fungible tokens with **Goldinals** to powering truly Bitcoin-native Layer 2 solutions, Nubit catalyzes a new wave of decentralized innovation. Together, these advancements position Bitcoin not just as a store of value but as the backbone of a secure, scalable, and universally accessible computing platform.

Reference

[1] Vitalik Buterin. Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform (2014).

[2] Satoshi Nakamoto. Bitcoin: A Peer-to-Peer Electronic Cash System (2008).

[3] Hanzhi Liu, Jingyu Ke, Hongbo Wen, Luke Pearson, Robin Linus, Lukas George, Manish Bista, Hakan Karakuş, Domo, Junrui Liu, Yanju Chen, and Yu Feng. Push-Button Verification for BitVM Implementations (2024). https://eprint.iacr.org/2024/1768.

[4] Charles Antony Richard Hoare. An axiomatic basis for computer programming. *Communications of the ACM* 12.10 (1969), pp. 576–580.

[5] The Nubit Team. Nubit DA: Bitcoin-Native Data Availability Layer with Instant Finality (2024). https://www.nubit.org/nubit_orangepaper.pdf.

[6] Hongbo Wen, Hanzhi Liu, Shuyang Tang, Tianyue Li, Shuhan Cao, Domo, Yanju Chen, and Yu Feng. Stateless and Verifiable Execution Layer for Meta-Protocols on Bitcoin (2024). https://eprint.iacr.org/2024/408.

[7] Robin Linus, Lukas Aumayr, Alexei Zamyatin, Andrea Pelosi, Zeta Avarikioti, and Matteo Maffei. BitVM2: Bridging Bitcoin to Second Layers (2024). https://bitvm.org/bitvm_bridge.pdf.