BitSlowly: A CPU Based Electronic Cash System

Anonymous
slw2qdc6xqca@gmail.com

Abstract mentation written entirely in Python. Instead of pursuing maximum transaction throughput or complex protocol designs, **BitSlowly** focuses on implementing the essential **BitSlowly** is an experimental cryptocurrency developed to investigate peer-to-peer digital cash on Android devices. The system emphasizes simplicity, transparency, and verifiable computation through a custom blockchain imple components of a decentralized cryptocurrency, including block mining, transaction verification, wallet management, UTXO processing, and peer-to-peer networking. The native currency unit, **Lowly**, reflects the project's philosophy that secure and decentralized value can emerge through gradual participation and computational work rather than centralized control. The primary objective of **BitSlowly** is to provide a lightweight and understandable blockchain platform suitable for experimentation, education, and further research on mobile hardware. Although the system incorporates core blockchain mechanisms such as Proof-of-Work mining, digital signatures based on the Elliptic Curve Digital Signature Algorithm (ECDSA), Merkle tree verification, and distributed ledger synchronization, it is intended as a research prototype rather than a production-ready financial infrastructure. The project demonstrates that the fundamental principles of decentralized digital currency can be implemented and evaluated within the computational constraints of Android devices while maintaining transparency, reproducibility, and ease of study.

1. Introduction

Since the introduction of Bitcoin in 2008, blockchain technology has enabled decentralized digital currency through cryptographic hashing, digital signatures, distributed consensus, and Proof-of-Work (PoW). While modern blockchain platforms have become increasingly complex, their implementation is often difficult to study and deploy on resource-constrained devices. **BitSlowly** is an experimental cryptocurrency developed to provide a lightweight and understandable blockchain implementation for Android devices using Python. The system implements fundamental blockchain components, including UTXO-based transactions, ECDSA signature verification, Merkle tree validation, peer-to-peer networking, blockchain synchronization, and CPU-based Proof-of-Work mining. Rather than competing with existing cryptocurrencies, Bitslowly is intended as a research and educational platform that demonstrates the practical implementation of decentralized digital cash on mobile hardware.

2. System Architecture

BitSlowly is designed as a lightweight peer-to-peer cryptocurrency consisting of four primary components: Wallet, Miner, Node, and Blockchain Storage. Users create and sign transactions using the wallet application. Transactions are propagated to network nodes, validated, and stored temporarily in the mempool. Miners collect valid transactions, generate candidate blocks through the proposed CPU-based Proof-of-Work algorithm, and broadcast newly mined blocks to other nodes. Each node independently verifies the block before appending it to the local blockchain and updating the UTXO database. This modular architecture enables decentralized transaction processing while remaining lightweight enough to operate on Android devices.

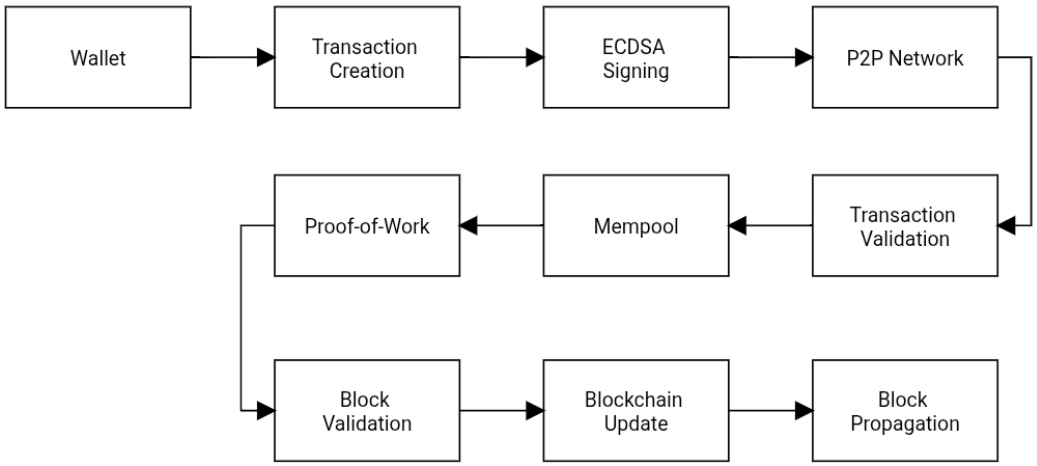

Figure 1. Overall architecture and transaction processing workflow of the BitSlowly blockchain.

Figure 1 illustrates the overall architecture of **BitSlowly**. A transaction is created by the wallet and digitally signed using ECDSA before being propagated through the peer-to-peer network. Each receiving node validates the transaction and stores valid transactions in the mempool. Miners select transactions from the mempool to construct candidate blocks and perform CPU-based Proof-of-Work mining. Once a valid block is found, all nodes verify the block, update their local blockchain and UTXO database, and finally propagate the new block throughout the network.

3. Blockchain Design

BitSlowly implements a lightweight blockchain based on the Unspent Transaction Output (UTXO) model. Each block contains a block header and a list of validated transactions. The block header includes the block height, timestamp, mining difficulty, Merkle root, previous block hash, nonce, and the resulting block hash. Transactions are organized into a Merkle tree, allowing every node to verify data integrity efficiently while minimizing computational overhead. Each transaction is identified by a double SHA-256 transaction identifier (TXID), while digital signatures are verified using the Elliptic Curve Digital Signature Algorithm (ECDSA) over the secp256k1 curve. The blockchain is stored as a sequential chain of blocks, where each block references the hash of its predecessor, making any modification computationally detectable. This structure provides immutability, transaction traceability, and efficient verification while remaining suitable for execution on Android devices with limited computational resources.

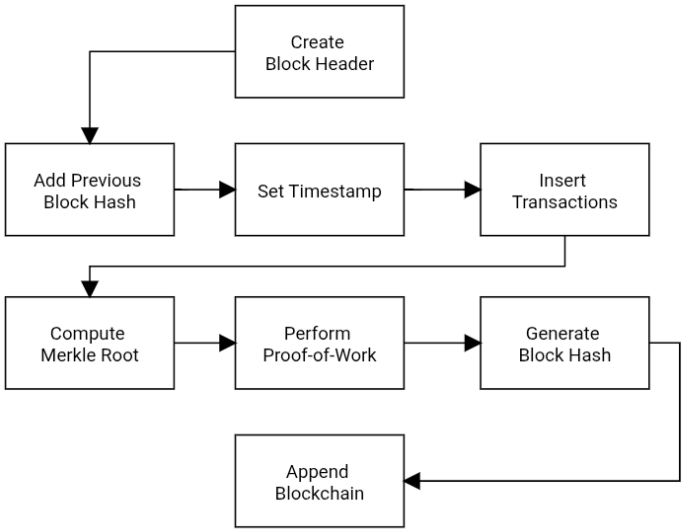

Figure 2. Blockchain block construction process in BitSlowly

Figure 2 illustrates the process of constructing a new block in **BitSlowly**. A miner first creates the block header by including the previous block hash and the current timestamp. Valid transactions collected from the mempool are inserted into the candidate block, after which the Merkle root is computed to represent all transactions. The miner then performs the Proof-of-Work algorithm by searching for a valid nonce that satisfies the current difficulty target. Once a valid block hash is generated, the block is appended to the local blockchain and becomes ready for propagation to other nodes.

4. CPU Based Proof-of-Work Algorithm

BitSlowly employs a CPU-based Proof-of-Work (PoW) consensus mechanism in which miners compete to discover a valid nonce that satisfies the current network difficulty target. Unlike GPU- or ASIC-oriented mining systems, the proposed approach is designed for execution on general-purpose processors commonly available in Android devices. For each candidate block, the block header is serialized and hashed using a double SHA-256 function to generate an initial seed. The mining algorithm repeatedly modifies the nonce value and computes a candidate block hash through the CPU hashing algorithm until the resulting hash is numerically lower than the target derived from the current difficulty. Once a valid hash is obtained, the mined block is considered valid and is propagated to peer nodes for independent verification. This approach maintains deterministic verification while remaining lightweight enough for experimental deployment on mobile hardware. The mining process is parallelized using multiple CPU cores through Python multiprocessing, allowing each worker to search different nonce ranges simultaneously and improving mining efficiency without specialized hardware.

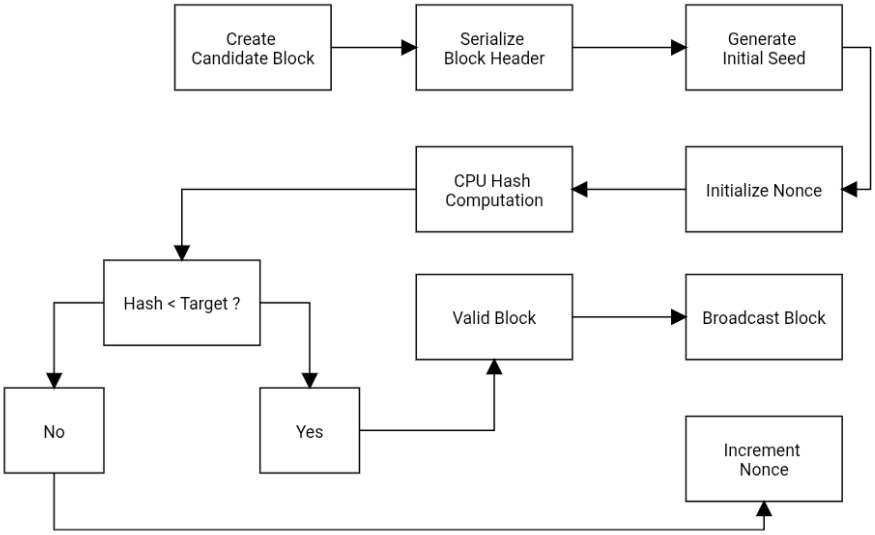

Figure 3. CPU based Proof-of-Work mining algorithm BitSlowly

Figure 3 illustrates the CPU-based Proof-of-Work mining process in **BitSlowly**. A candidate block is created, its header is serialized, and block hashes are repeatedly computed using different nonce values until the current difficulty target is satisfied. Once a valid hash is found, the block is broadcast to the network.

5. Transaction Validation

BitSlowly validates every transaction before it is accepted into the mempool. Each transaction is verified by checking the transaction format, digital signature, and the availability of unspent transaction outputs (UTXOs). Invalid, duplicate, or already spent inputs are rejected to prevent double-spending attacks. Only transactions that satisfy all validation rules are stored in the mempool and become eligible for inclusion in newly mined blocks

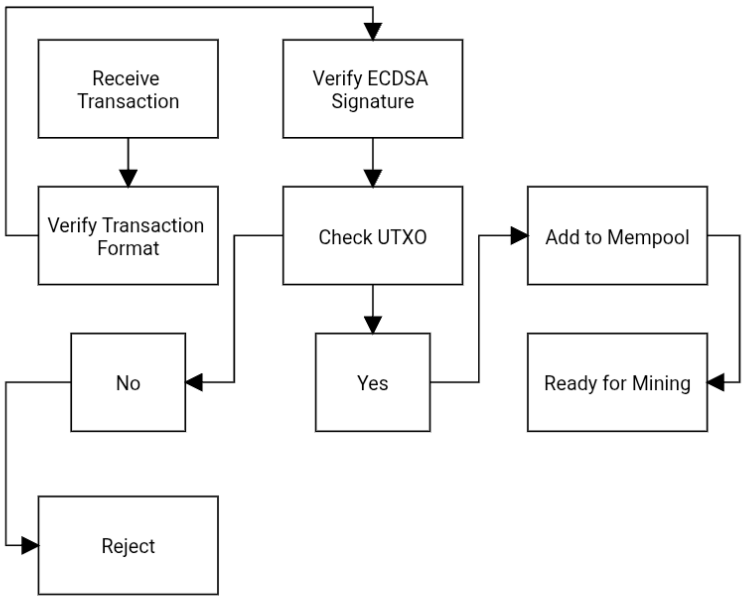

Figure 4. Transaction validation process in BitSlowly

Figure 4 illustrates the transaction validation process in **BitSlowly**. Each received transaction is first checked for its structure and required fields before the ECDSA digital signature is verified. The node then validates the referenced UTXOs to ensure that the inputs have not been spent previously. Transactions that fail any verification step are immediately rejected and are not stored in the mempool. Valid transactions are accepted into the mempool and remain available for selection by miners during block construction. This validation process helps maintain transaction integrity and prevents double-spending within the blockchain network.

6. Network Communication

BitSlowly uses a lightweight peer-to-peer communication model to synchronize blockchain data and propagate transactions across the network. Each node communicates through HTTP-based APIs to exchange transactions, blocks, and network information. Newly received transactions are validated before entering the mempool, while newly mined blocks are independently verified before being appended to the local blockchain. After successful verification, the updated blockchain is propagated to neighboring nodes to maintain network consistency. This decentralized communication mechanism enables every participant to independently verify data and maintain a synchronized copy of the blockchain without relying on a trusted central authority.

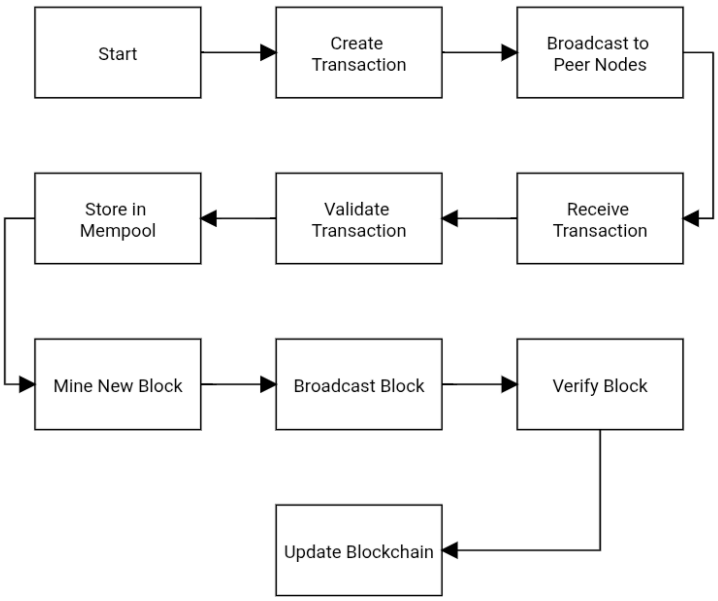

Figure 5. Peer-to-peer communication process in BitSlowly

illustrates the peer-to-peer communication process in **BitSlowly**. A transaction is first created and broadcast to connected peer nodes, where it is independently validated before being stored in the mempool. Miners then select valid transactions to construct a candidate block and perform Proof-of-Work. Once a valid block is mined, it is broadcast to the network for verification and blockchain synchronization. This process ensures that all participating nodes maintain a consistent and up-to-date blockchain ledger.

7. Experimental Results

The **BitSlowly** prototype was implemented in Python and evaluated on an Android smartphone using the Termux environment. The implementation includes wallet generation, UTXO-based transaction processing, ECDSA signature verification, Merkle tree construction, CPU-based Proof-of-Work mining, and peer-to-peer communication through HTTP APIs. During testing, the blockchain successfully generated new blocks, validated transactions, synchronized data between nodes, and maintained blockchain consistency throughout the mining process. The results demonstrate that the proposed architecture can operate correctly on resource-constrained mobile hardware while preserving the fundamental properties of decentralized digital cash.

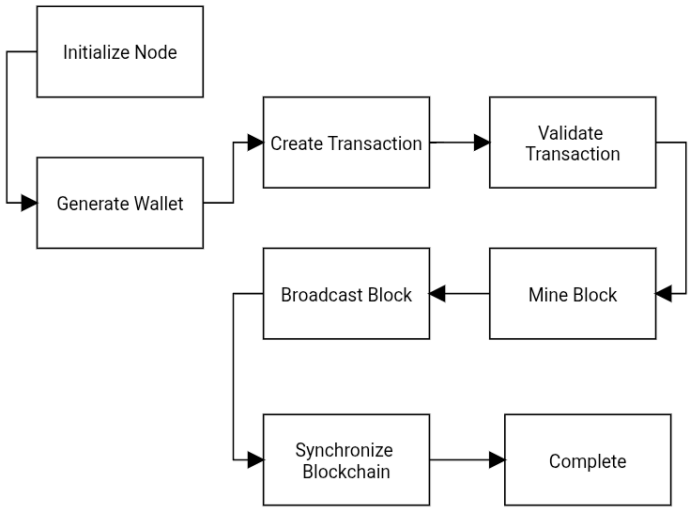

Figure 6. Experimental workflow of the BitSlowly implementation

Figure 6 illustrates the experimental workflow used to evaluate the **BitSlowly** prototype. A node initializes the blockchain, generates a wallet, creates and validates transactions, performs CPU-based mining, broadcasts newly mined blocks, and synchronizes the blockchain with peer nodes. The successful execution of each stage confirms the correctness of the proposed implementation.

8. Discussion

.The experimental results demonstrate that **BitSlowly** successfully implements the fundamental components of a decentralized cryptocurrency on Android devices. The proposed architecture integrates wallet generation, transaction validation, UTXO management, ECDSA digital signatures, Merkle tree verification, CPU-based Proof-of-Work mining, and peer-to-peer communication into a lightweight blockchain system. The implementation confirms that blockchain operations can be executed correctly on resource-constrained mobile hardware while preserving data integrity and decentralized verification. Although the prototype is not intended for production deployment, it provides a practical platform for studying blockchain architecture, cryptographic protocols, and distributed consensus. The modular design also allows individual components to be extended independently for future research. Future work may include dynamic difficulty adjustment, optimized networking, enhanced peer discovery, improved synchronization efficiency, and comprehensive performance evaluation under larger network conditions.

9. Conclusion

BitSlowly presents a lightweight blockchain implementation designed for experimentation and education on Android devices. The proposed system integrates wallet generation, UTXO-based transaction processing, ECDSA signature verification, Merkle tree construction, CPU-based Proof-of-Work mining, and peer-to-peer networking into a complete cryptocurrency prototype. Experimental evaluation demonstrates that the proposed architecture is capable of generating blocks, validating transactions, synchronizing blockchain data, and maintaining ledger consistency across participating nodes. The results indicate that the essential principles of decentralized digital cash can be implemented efficiently using Python without requiring specialized hardware. Consequently, **BitSlowly** serves as a practical reference implementation for blockchain education and experimental research. Future development will focus on scalability improvements, stronger network security, optimized mining performance, and support for larger decentralized blockchain environments.

References

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

[2] A. M. Antonopoulos, Mastering Bitcoin, 2nd ed. Sebastopol, CA, USA: O'Reilly Media, 2017.

[3] W. Stallings, Cryptography and Network Security: Principles and Practice, 8th ed. Pearson, 2020.

[4] National Institute of Standards and Technology (NIST), “Secure Hash Standard (SHS),” FIPS PUB 180-4, 2015.

[5] National Institute of Standards and Technology (NIST), “Digital Signature Standard (DSS),” FIPS PUB 186-5, 2023.

[6] Python Software Foundation, Python 3 Documentation. Available: https://docs.python.org/3/

[7] P. Wuille, “secp256k1 Elliptic Curve Cryptography Library,” GitHub, 2024.