Whitepaper

BitcoinBT (BTCBT)

---

1. Introduction

BitcoinBT (BTCBT) is a Proof-of-Work (PoW) based blockchain that adopts a hard fork structure inheriting the existing Bitcoin chain up to block height 903,844. The project is designed to preserve the fundamental structure of the original Proof-of-Work chain while redefining a limited set of consensus parameters.

BitcoinBT does not introduce a new genesis block. Block data prior to the defined fork height remains unchanged, and subsequent blocks proceed based on the chain state at that point. Newly defined consensus rules apply only to blocks after the fork height. This structure maintains chain continuity while clearly distinguishing the scope of consensus modifications.

The primary parameters applied after the fork are defined as follows:

- Target block interval: 300 seconds
- Difficulty adjustment algorithm: ASERT
- Maximum consensus block size: 32MB
- Retention of the SHA-256 based Proof-of-Work mechanism

The Proof-of-Work mechanism, block header structure, transaction validation model, and chain selection rule remain unchanged. Chain selection continues to follow the cumulative work principle.

BitcoinBT is not limited to a conceptual proposal. Based on the Bitcoin Core v26 codebase, a consensus branching structure has been implemented. Rules before and after the fork height are conditionally separated within the source code. The corresponding behavior has been verified within an internal execution environment.

This whitepaper is a general document intended to describe the structural design, consensus definitions, network model, economic structure, and current implementation stage of BitcoinBT. Code-level implementation details and computational procedures are addressed separately in technical documentation.

2. Background and Design Basis

BitcoinBT was developed to evaluate whether specific consensus definitions could be redefined while preserving the structural framework of an existing Proof-of-Work chain. The objective was not to introduce a new consensus model, but to determine whether limited parameter adjustments could be implemented without altering the underlying consensus architecture.

At the initial design stage, the following conditions were defined and maintained:

- The SHA-256 Proof-of-Work mechanism remains unchanged.
- The block header structure and transaction format remain unchanged.
- The existing UTXO state model remains unchanged.
- The cumulative work-based chain selection rule is retained.
- The genesis block of the original chain is not modified.

These conditions preserve structural continuity of the chain.

The fork activation height was defined as block 903,844. Block data up to this height is inherited without modification. Blocks after this height operate under redefined consensus parameters. Consensus rules are applied based on block height: blocks prior to the fork follow the original rules, while blocks after the fork follow the redefined parameters.

At the consensus level, modifications are limited to the following elements:

- Definition of the target block interval
- Difficulty adjustment method (application of ASERT)
- Definition of the maximum consensus block size
- Block height–based consensus branching logic

Core structural components outside this scope remain unchanged.

This approach does not involve structural redesign. Instead, it redefines specific consensus parameters while preserving the existing chain framework. Post-fork rules are implemented through conditional separation from the original validation path at the code level.

Implementation included direct modification and compilation of the Bitcoin Core v26 source code. Within an internal execution environment, post-fork chain behavior was verified to follow the defined branching structure.

3. Design Principles and Scope of Consensus Changes

BitcoinBT is designed on the premise of preserving the structural framework of the existing Proof-of-Work chain. Consensus changes are limited to the redefinition of specific parameters and do not constitute a comprehensive redesign of the core chain architecture.

3.1 Preservation of Continuity

BitcoinBT inherits the block history of the existing chain up to the defined fork height. The validation criteria for blocks prior to the fork remain unchanged. Redefined consensus parameters apply only to blocks following the fork height.

3.2 Limitation of Consensus Modifications

Modifications at the consensus level are limited to the following elements:

- Definition of the target block interval
- Replacement of the difficulty adjustment algorithm (application of ASERT)
- Definition of the maximum consensus block size

The following core elements remain unchanged:

- The SHA-256 Proof-of-Work mechanism
- Block header structure
- Transaction structure
- The UTXO state model
- The cumulative work-based chain selection rule

3.3 Block Height–Based Branching Structure

Consensus branching is applied based on a predefined block height. Blocks prior to the fork follow the original rules, while blocks after the fork operate under the redefined consensus conditions and calculation paths.

This branching structure does not remove the original validation logic at the code level. Instead, it conditionally selects the appropriate consensus definition for blocks beyond the specified height.

3.4 Deterministic Operation

Nodes running identical software derive identical validation results for the same block input. Consensus application is determined by block height and does not depend on external signaling or manual intervention.

4. Block Interval and Difficulty Adjustment

BitcoinBT defines a target block interval of 300 seconds. This value represents the defined average block interval at the consensus level.

Actual block generation time varies according to the total network hash rate. The difficulty adjustment mechanism compensates for such variations so that the long-term average interval is aligned with the defined target.

4.1 Target Block Interval

The target block interval is set to 300 seconds. This alters the time-based progression rate while maintaining a block-height–based issuance structure.

4.2 Difficulty Adjustment Mechanism

Difficulty adjustment follows the ASERT algorithm. ASERT references a defined anchor block and adjusts the target value based on observed time deviation.

Difficulty is calculated on a per-block basis and is not reset using fixed adjustment periods. This structure is intended to mitigate long-term accumulation of block time deviations.

4.3 Scope of Application

The ASERT-based difficulty calculation applies only to blocks following the defined fork height. Within this range, the target value is adjusted continuously according to observed deviation from the defined interval.

4.4 Verification of Operation

Within an internal execution environment, the following aspects were examined:

- Whether difficulty calculations are applied on a per-block basis
- Whether difficulty adjusts in response to changes in block production rate
- Stability under boundary calculation conditions

These procedures were conducted to confirm that the defined difficulty adjustment mechanism operates at the consensus level as specified.

5. Economic Structure and Emission Model

The issuance of new units in BitcoinBT is defined by its block reward mechanism. Issuance amounts are calculated through consensus code paths and produce identical computation results across nodes.

5.1 Block Reward–Based Issuance

New units are issued as block rewards upon successful block generation. The issuance amount is determined by block height and does not depend on external inputs or discretionary adjustments.

Reward calculation is performed as part of the consensus logic.

5.2 Halving Structure

The block reward is reduced by half every 210,000 blocks. The halving mechanism is applied based on block height rather than elapsed time.

Since the target block interval is defined as 300 seconds, an identical block-count–based halving structure results in a shorter time-based interval compared to chains with longer block intervals.

5.3 Total Supply Structure

Block rewards decrease in successive stages according to a geometric reduction model. As a result, the total supply converges toward a finite maximum.

The maximum supply defined at the consensus level is 21,000,000 units.

Only block rewards generated after the fork height are considered part of the BitcoinBT issuance.

5.4 Mining Reward Composition

Miners receive the following components:

- Block reward
- Transaction fees

Rewards are recognized only for blocks that pass consensus validation. Modification of the reward structure requires changes to consensus code and deployment of updated software.

6. Network Architecture and Participation Model

BitcoinBT is implemented as a peer-to-peer network protocol. Participation occurs by running the node software, through which nodes receive block and transaction data from connected peers and validate that data according to consensus rules.

6.1 Scope of Node Functions

Nodes perform the following functions:

- Receive and validate transactions
- Receive and validate blocks
- Relay valid data to connected peers
- Synchronize chain data

Validation is performed independently by each node. Blocks or transactions that fail validation are not incorporated into the local chain state and are not treated as valid network data.

6.2 Mining Participation Structure

Block generation in BitcoinBT is performed through SHA-256–based Proof-of-Work computation. Miners participate according to the following procedure:

1. Select valid transactions and construct a candidate block.
2. Build the block header and compute hashes against the defined difficulty target.
3. If a hash meeting the target requirement is found, relay the block to connected peers.

4. Receiving nodes validate the block and accept and connect it to the active chain if it satisfies consensus rules.

A block is incorporated into the chain only after full validation.

6.3 Chain Selection Rule

Nodes select the chain with the most cumulative work as the valid chain. This process follows consensus rules and does not rely on manual intervention or centralized approval.

6.4 Verification Scope During Testing

In an isolated multi-node testing environment, the following behaviors were examined:

- Connectivity and synchronization between nodes
- Block propagation and validation flow after reception
- Rejection of blocks violating consensus rules
- Chain state updates during block connection

These procedures were performed to confirm that network participation and validation flow operate in accordance with the defined consensus rules.

7. Development Structure and Change Control

BitcoinBT is implemented as a fork based on the Bitcoin Core v26 codebase.
Modifications are limited to consensus-related parameters and their associated validation logic.

7.1 Scope of Modifications

Changes are limited to the following areas:

- Chain parameter definitions
- Difficulty adjustment computation path
- Target block interval parameter
- Consensus maximum block size limit

- Height-based consensus branching logic

The underlying block validation structure and transaction verification flow remain consistent with the original architecture.

7.2 Activation Mechanism

Consensus-related changes are activated at a defined block height. Legacy and updated validation paths coexist within the codebase and are selected through height-based conditional logic.

Evaluation under legacy or updated rules is determined exclusively by block height and chain state.

7.3 Separation of Consensus and Policy

Consensus rules determine block validity and chain acceptance.

Policy rules govern transaction relay and local mempool behavior but do not alter consensus validation of blocks.

The implemented changes are confined to consensus parameters. Policy-layer mechanisms were not redefined.

7.4 Deterministic Consensus Evaluation

Consensus evaluation depends solely on block data and current chain state. Given identical inputs and software versions, nodes produce identical validation outcomes.

8. Security Model and Network Stability

BitcoinBT follows a Proof-of-Work–based consensus model. Block validity is determined by consensus rules, and the active chain is selected according to cumulative work.

Security assumptions follow the distribution of computational work and the cumulative work selection rule.

8.1 Proof-of-Work Chain Structure

Blocks are generated using SHA-256 hashing. Only blocks containing a hash value below the defined difficulty target are considered valid. This condition is applied consistently

across nodes.

As additional blocks are appended, the cumulative work securing earlier blocks increases. Block validity is independently evaluated by each node.

8.2 Structural Impact of a 300-Second Block Interval

The target block interval is defined as 300 seconds, resulting in more blocks per unit time compared to longer interval configurations.

This configuration affects:

- The relative impact of propagation latency
- The frequency of temporary forks and reorganization events
- The responsiveness of difficulty recalculation

Consensus evaluation continues to rely on cumulative work and difficulty target computation.

8.3 Difficulty Adjustment and Long-Term Behavior

Difficulty is recalculated at each block using the ASERT algorithm. When time deviations occur, the difficulty target adjusts according to the defined computation method.

Under varying hash rate conditions, this mechanism adjusts toward the defined target interval over time.

Internal observation included:

- Difficulty response under varying block production speeds
- Boundary-condition behavior
- Consistency of computation results across nodes

8.4 Block Size Limit and Validation Cost

The consensus-level maximum serialized block size is defined as 32MB. Validation cost scales with transaction count and script verification complexity within a block.

Blocks exceeding the consensus size limit are invalid.

8.5 Scope of Internal Observation

In multi-node testing environments, the following behaviors were observed:

- Block generation and propagation
- Rejection of blocks violating consensus rules
- Chain selection during reorganization
- Synchronization behavior across nodes

These observations were conducted to confirm that consensus evaluation and validation logic operate consistently under network conditions.

9. Current Status and Project Phase

BitcoinBT is in an implementation and validation stage focused on verification that defined consensus parameters are applied to chain operation according to the codebase.

This stage aims to verify that consensus-related modifications are reflected in the intended validation and computation paths.

9.1 Scope of Consensus Application Verification

For blocks beyond the defined fork height, the following elements were evaluated within testing conditions:

- Application of the defined target block interval parameter
- Application of the ASERT-based difficulty adjustment path
- Enforcement of the consensus maximum block size limit
- Operation of signature verification logic included in consensus

During block validation, consensus parameters were evaluated to confirm their presence within the applicable computation paths.

9.2 Block Generation and Chain Progress Evaluation

Within testing environments, block generation behavior was evaluated, including assessment of:

- Reward calculation path
- Increment of chain height

- Reflection of difficulty computation results

Only blocks satisfying consensus validation were accepted and connected to the active chain.

9.3 Structure of Code-Level Application

Consensus-related changes are activated through height-based conditional branching. Legacy and modified computation paths coexist and are selected through deterministic conditional logic.

Consensus parameters are defined as constants or height-based conditions within the source code and are not altered dynamically during execution.

9.4 Scope of the Current Phase

The current phase includes:

- Implementation of consensus parameter changes
- Evaluation of block generation behavior
- Synchronization testing in internal multi-node environments

Assessment under large-scale or heterogeneous public network conditions remains outside the scope of this document.

10. Scope of This Document and Closing Statement

This document provides a structured overview of the BitcoinBT architecture, defined consensus parameters, and the scope of implemented modifications. The described scope is limited to consensus parameter changes and observations conducted within internal execution environments.

BitcoinBT references the existing chain history up to the defined fork height and activates updated consensus parameters beyond that height. This document outlines those parameter definitions and their activation scope. Detailed function-level implementation and source-code computation processes are not included.

The scope of this document includes:

- Height-based consensus transition structure

- Definition of the target block interval
- ASERT-based difficulty adjustment structure
- Definition of the consensus maximum serialized block size
- Block generation and internal evaluation procedures
- Network synchronization and chain selection flow

This document is not intended for investment analysis, market valuation, price forecasting, or external environmental assessment. It does not provide legal or financial interpretation.

Consensus parameters are defined as constants and conditional logic within the source code. Nodes operating identical software evaluate identical block inputs and produce deterministic validation results.

Internal evaluation included:

- Block generation behavior
- Application of the difficulty computation path
- Rejection of blocks exceeding defined consensus limits
- Synchronization behavior in multi-node testing environments

These activities were conducted to verify application of the defined consensus computation paths. Evaluation under large-scale public network conditions and long-term operational environments remains outside the scope of this document.

This document documents the defined modification scope and the observed implementation status of BitcoinBT.