Whitepaper (English)

BitcoinBT (BTCBT)

---

1. Introduction

BitcoinBT (BTCBT) is a Proof-of-Work (PoW) blockchain that follows a hard-fork design inheriting the Bitcoin chain up to block height **903,844**. The project preserves the core structure of an existing PoW chain while redefining a limited set of consensus parameters.

BitcoinBT does not create a new genesis block. All block data prior to the fork point remains unchanged, and the chain continues from the fork state. The new consensus rules apply **after** the fork point, starting from block height **903,845**. This design maintains continuity of the historical chain while clearly separating the consensus-change region.

Key definitions applied after the fork are as follows:

- Fork point (inherited history up to): **903,844**
- Activation height (new rules apply from): **903,845**
- Target block interval: **300 seconds**
- Difficulty adjustment algorithm: **ASERT**
- ASERT half-life: **172,800 seconds (2 days)**
- Consensus maximum block size: **32 MB (32,000,000 bytes)**
- Proof-of-Work: **SHA-256 (SHA-256d)**
- Halving interval: **210,000 blocks** (approximately 2 years at 300s target)
- Maximum supply cap (Production mainnet): **21,000,000 BTCBT**

On the Production mainnet, the maximum supply cap is defined as **21,000,000 BTCBT**, and a **special reward of 630,000 BTCBT** is included within this cap. The **630,000 BTCBT special reward is paid once at block height 903,850**. On the Production mainnet, the emission schedule is defined by consensus so that the special reward remains **within** the maximum supply cap. Any issuance exceeding the supply cap is treated as a consensus rule violation and the corresponding block is invalid.

BitcoinBT keeps the core PoW structure, block header format, transaction validation model, UTXO state model, and chain selection rule consistent with the inherited design. Chain selection is based on accumulated work.

BitcoinBT is implemented as a consensus-fork structure based on the Bitcoin Core v26 codebase. Consensus application is determined automatically by block height; no external approval or centralized control process exists.

This whitepaper is a technical overview document describing the design, consensus definitions, network model, economic model, and current project phase. Code-level implementation details and exact calculation methods are covered in separate technical documents, and the ultimate reference for consensus rules is the publicly released source code and the corresponding release tag deployed on the Production mainnet. A test mainnet is a separate environment for functional and performance validation, and experimental parameters may be applied there.

---

2. Background and Design Basis

BitcoinBT originated from an engineering review of whether a limited set of consensus definitions can be reconfigured while preserving the structure of an existing PoW chain. The goal was not to introduce a fundamentally new consensus model, but to confirm the feasibility of parameter changes without redesigning the core system.

The following constraints were defined from the beginning:

- Do not change the SHA-256 PoW structure.
- Do not change the block header format or transaction structure.
- Do not change the UTXO state model.
- Preserve the accumulated-work-based chain selection rule.
- Do not change the inherited genesis block.
- Preserve the same signature verification model as Bitcoin.

BitcoinBT does not introduce a separated signature scheme such as SIGHASH ForkID. Signature validity and verification follow the same model as Bitcoin. These constraints are required to maintain structural continuity.

The fork point is defined as block height **903,844**. All blocks up to that height are

inherited without modification. New consensus parameters apply after the fork, starting from **903,845**. Consensus rules are applied deterministically based on block height: pre-fork blocks follow the inherited rules, and post-fork blocks follow the redefined parameters.

Consensus-level changes are limited to:

- Target block interval definition
- Difficulty adjustment method (ASERT)
- Consensus maximum block size definition
- Block-height-based consensus branching logic

All other core structures remain unchanged, including the SHA-256 PoW method, transaction structure, UTXO model, and chain selection rule.

---

3. Design Principles and Scope of Consensus Changes

BitcoinBT is designed under the assumption that the inherited PoW chain structure is maintained. Consensus changes are restricted to a limited set of redefined parameters, and the core architecture is not redesigned.

3.1 Continuity

BitcoinBT inherits the historical blockchain up to the fork point. Validity rules for pre-fork blocks are not changed, and the defined post-fork consensus parameters apply after activation.

3.2 Minimal Consensus Changes

The modified consensus parameters are limited to:

- Target block interval definition
- Difficulty adjustment algorithm change (ASERT)
- Consensus maximum block size definition

The following components are not changed:

- SHA-256 PoW method
- Block header structure

- Transaction structure
- UTXO state model
- Accumulated-work-based chain selection rule

3.3 Block-Height-Based Branching

Consensus branching is applied based on the explicitly defined block height. Pre-fork blocks follow the inherited rules; post-fork blocks follow the redefined consensus conditions and calculation paths. This branching remains fixed at runtime and cannot be selectively disabled.

3.4 Deterministic Behavior

Nodes running the same software produce the same validation results for the same block input. Consensus application is determined by block height and does not rely on external signaling, centralized approval, or manual intervention. Consensus changes occur only through software updates adopted by network participants.

---

4. Block Interval and Difficulty Adjustment

BitcoinBT defines a target block interval of **300 seconds**. This represents the long-run average block production target in consensus. Actual block time varies with total network hashrate, and the difficulty adjustment mechanism is designed to drive the long-run average toward the target.

4.1 Target Block Interval

The target interval is set to 300 seconds. This preserves a block-count-based emission structure while changing its progression on the time axis.

4.2 Difficulty Adjustment Method

Difficulty adjustment follows the **ASERT** algorithm. ASERT adjusts the target based on time differences relative to a reference point. The ASERT half-life is defined as **172,800 seconds (2 days)**.

Difficulty is computed per block and is not reset on a fixed periodic boundary. Difficulty adjustment is not disabled, and minimum-difficulty allowance rules are not applied.

4.3 Scope of Application

ASERT difficulty computation applies to post-fork blocks under the defined fork conditions. In this region, difficulty is continuously corrected based on observed time deviation.

4.4 Validation Notes

In internal validation environments, the following were checked:

- Per-block difficulty calculation behavior
- Difficulty response under changing block production rates
- Stability at boundary conditions

---

5. Economic Structure and Emission Model

BitcoinBT's issuance is defined by block creation rewards. Issuance is computed through consensus code paths, and nodes produce identical results for the same height under the same consensus software.

5.1 Block-Reward-Based Issuance

New units are issued as block subsidies when valid blocks are created. The subsidy amount is determined by block height and is not dynamically adjusted by external factors.

5.2 Halving Structure

The subsidy halves every **210,000 blocks**. Halving is block-count-based, not time-based. Since the target block interval is 300 seconds, the same block-count schedule occurs over a shorter time horizon than a 600-second target.

5.3 Maximum Supply Structure

Subsidies decrease geometrically, causing total supply to converge to a finite value. The maximum supply cap (Production mainnet) is **21,000,000 BTCBT**. The **630,000 BTCBT special reward** is included within this cap and is issued once at block height **903,850**.

5.4 Miner Reward Composition

Miners receive:

- Block subsidy

- Transaction fees

Rewards are recognized only for blocks that pass consensus validation. Changing the reward structure requires consensus code changes and distribution of new software, which must then be adopted by the network.

---

6. Network Architecture and Participation Model

BitcoinBT uses an open P2P network architecture. Participation occurs by running node software, and nodes validate received blocks and transactions according to consensus rules.

6.1 Node Responsibilities

Nodes perform:

- Transaction reception and validation
- Block reception and validation
- Propagation of valid data
- Chain synchronization

Invalid blocks or transactions are not applied to the local chain state.

6.2 Mining Participation

Block creation occurs via SHA-256 PoW computation. Miners participate as follows:

1. Select valid transactions and assemble a candidate block.
2. Construct the block header and compute hashes against the consensus target.
3. When a hash below the target is found, broadcast the block.
4. Nodes validate the block and, if valid, connect it to the chain.

6.3 Chain Selection Rule

Nodes select the valid chain with the greatest accumulated work.

6.4 Internal Validation Scope

Internal validation environments checked:

- Multi-node connectivity and synchronization
- Block propagation and validation flow
- Rejection of consensus-invalid blocks
- Chain state update behavior during block connection

---

7. Development Structure and Change Control

BitcoinBT is implemented by modifying the Bitcoin Core v26 codebase. Changes are limited to consensus parameters and the required calculation paths.

7.1 Scope of Modifications

Key areas include:

- Chain parameter definitions
- Difficulty computation paths
- Target block interval definition
- Consensus maximum block size definition
- Block-height-based branching conditions

The baseline structure of block validation and transaction verification remains unchanged.

7.2 Application of Changes

Consensus changes are applied based on block height through explicit conditional branching. Inherited and modified calculation paths are separated by these conditions.

7.3 Consensus vs. Policy

Consensus rules determine block validity. Policy rules relate to transaction relay and mempool acceptance. BitcoinBT's modifications focus on consensus parameters and do not redesign the policy layer.

7.4 Deterministic Consensus Computation

Consensus computation is determined by block data and chain state. Identical software produces identical results for identical inputs.

---

8. Security Model and Operational Stability

BitcoinBT follows a PoW consensus model. Block validity is determined by consensus rules, and chain legitimacy is evaluated by accumulated work. Network security is shaped by hashrate distribution and competition.

8.1 PoW Chain Structure

Blocks are created via SHA-256 hashing. Only blocks that include a hash below the target are valid, and this condition is applied identically across all nodes.

8.2 Implications of a 300-Second Target

A 300-second target increases blocks per unit time. This is related to:

- Relative impact of propagation delay
- Short-term reorganization behavior under delays
- Difficulty adjustment responsiveness

8.3 Difficulty Adjustment and Long-Run Average

Difficulty is computed per block using ASERT. When time deviation occurs, the target is adjusted according to the defined formula so the long-run average approaches the target interval.

8.4 Block Size Cap and Validation Load

The consensus maximum block size is **32 MB (32,000,000 bytes)**. Blocks exceeding this cap are invalid. Validation cost increases with transaction count and script verification complexity.

8.5 Internal Validation Notes

The following were checked in test environments:

- Block creation and propagation
- Rejection of consensus-invalid blocks
- Chain selection during reorganizations
- Multi-node synchronization behavior

---

9. Current Status and Project Phase

BitcoinBT is in an implementation and internal validation phase, intended to verify that the defined consensus parameters are correctly reflected in chain behavior.

9.1 Verified Items

In post-fork regions, the following were checked:

- Application of the 300-second target interval definition
- Application of ASERT difficulty computation paths
- Application of the 32 MB consensus cap
- Application of block reward computation paths

9.2 Block Production and Chain Progress

Internal tests verified actual block production and chain height progression, including subsidy computation and difficulty results. Only blocks passing validation were connected.

9.3 Code Application Structure

Consensus changes are applied through height-based conditional branching. Consensus parameters are managed as constants and conditions in source code and are not changed dynamically at runtime.

9.4 Scope Limitation

Evaluation under large-scale external network conditions and long-term operational risk analysis is outside the scope of this document.

---

10. Scope and Closing Statement

This document is a technical overview describing BitcoinBT's consensus structure and implementation scope. The content is limited to the defined consensus-change items and the internal validation scope.

This document covers:

- Block-height-based consensus transition structure
- 300-second target block interval

- ASERT difficulty adjustment structure
- 32 MB consensus block size cap
- 21,000,000 maximum supply structure
- Scope of consensus application

This document is not intended for investment decisions, market valuation, price predictions, or external environment analysis, and it does not provide legal or financial advice. The ultimate reference for consensus rules is the publicly released source code and the corresponding release tag deployed on the Production mainnet.