# BitBlend: A Non-Disruptive Solution to the Bitcoin 2106 Timestamp Overflow

10 January 2024

**Abstract:**

The Bitcoin network faces a significant technical challenge as it approaches the year 2106, when the 32-bit field for block timestamps will reach its maximum value. This paper introduces "BitBlend," a solution designed to address this impending overflow without necessitating a coordinated hard fork or consensus change. Similar to the idea by Pieter Wuille [2], BitBlend proposes a novel reinterpretation of the existing 32-bit time field, extending its functionality by representing only the last 32 bits of the full timestamp. The solution incorporates an innovative overflow detection and correction method, named the BitBlend procedure, which seamlessly integrates into the current system. Key to BitBlend is maintaining the original 32-bit format for external communication while employing a 64-bit internal representation for block times. This dual approach ensures backward compatibility and network continuity, allowing nodes to gradually adopt the update without synchronization. Additionally, BitBlend addresses the implications for time locks, advocating for the natural expiration of absolute time locks post-2106 and the continued use of block height and relative time locks. This solution prioritizes minimal alterations to Bitcoin's core components, striving to preserve its foundational principles while ensuring its longevity and functionality. The paper seeks feedback from the Bitcoin development community on the feasibility and potential integration of the BitBlend solution into the Bitcoin protocol.

**The Problem:**

In 2106, the 32-bit field used for block timestamps in Bitcoin will reach its maximum value. This limitation poses a challenge for maintaining the network's operation, as Bitcoin's consensus rules rely on these timestamps for validating block times, calculating the difficulty adjustment, and verifying time locked transactions.

**Proposed Solution:**

The proposed solution uses a reinterpretation of the 32-bit time field to avoid changes in block structure or consensus rules. The solution allows for a gradual non-synchronous update of node software. Nodes can update at any point before 2106, ensuring network integrity without necessitating a coordinated, simultaneous hard fork. Because non-upgraded nodes are not capable of accepting additional blocks beyond the 32-bit time maximum, the overflow event in 2106 will not result in a chain split with two separate continuations between upgraded and non-upgraded nodes. The BitBlend solution is as follows:

- **Reinterpreting the 32-bit Field:** Continue to use the existing 32-bit field in the block header but reinterpret it. Instead of representing the total time since 1970, it would represent only the last 32 bits of the full timestamp.

- **Compare with Median of Last Eleven Blocks:** When evaluating a proposed block, the 32-bit time header is zero-padded into a 64-bit variable and compared with the median past time (MPT) of the last eleven blocks. This comparison is already a part of the consensus rules [1] and can be extended to detect a 32-bit overflow.

- **Overflow Detection and Correction via BitBlend:** If the new time is less than half of the MPT, it's assumed that an overflow has occurred due to the 32-bit limit. To correct for this overflow, the following operation is performed:

    - The first 32 bits of the MPT are blended with the last 32 bits of the new time.
    - Then 2^32 is added to this blended result to account for the overflow.

  This process effectively extends the timestamp into the 64-bit space while maintaining continuity with past times. It corrects for the 2106 overflow as well as each future 32-bit overflow occurring every 136 years. The future proofing without the need to store enumerated epochs makes BitBlend an improvement over similar solutions offered by Pieter Wuille [2] and Yanmaani [3].

- **External Use of 32-bit Block Times:** All forms of communication between nodes retain the 32-bit representation of block times. In order to maintain consensus with non-upgraded nodes, the same 32-bit block time serialization format is used for block header propagation, initial block downloads, and all other external node communications.After 2106 the 32-bit field only captures the last 32 bits of the block time.

- **Internal Use of 64-bit Block Times:** The handling of block times internally within a node is carried out in 64-bit representation. After the BitBlend procedure converts a new 32-bit block time, the 64-bit representation is consistently employed for various time-related operations, including block time comparisons and difficulty adjustment calculations That surprising little effort is needed to maintain the consensus elements of the difficulty adjustment and block time comparison after 2106 was pointed out by Nate Eldredge [4] and is due to the minimal reliance of bitcoin on Earth time. A path towards further reducing Earth time reliance is offered by The Time Lock Implication section below.

- **Storage of Historial Block Times:** Nodes have two potential approaches to store past block times: they could maintain the original 32-bit storage structure, applying the BitBlend process to derive 64-bit times as needed, or they could opt to store all past block times directly in a 64-bit format. In the latter case, when transmitting block times over the network, these 64-bit values would be condensed back to their last 32 bits to comply with network protocol standards.

- **Block Template Creation:** When miners or pool operators assemble a block header for a new block, they incorporate only the final 32 bits of the block creation time into the block header's time field. This approach aligns with the existing consensus rules regarding the block header format. The BitBlend mechanism ensures that the block creation time is corrected to its full 64-bit value when read in by network nodes.

- **Unchanged Script Interpreter:** While bitcoin transaction scripts contain 32-bit variables that are validated against block time, the BitBlend solution proposes leaving the bitcoin script interpreter unchanged. Block height based locks and relative time locks will function normally after 2106 as they do not require 64-bit representation. Absolute time locks will naturally expire in 2106 as it is not possible to use the 32-bit nLocktime field to set a locktime beyond 2106. See the Time Lock Implications section below for further details on continued locktime functionality after 2106 and an argument for not extending absolute lock times in seconds after they naturally expire.

- **Continuation Post-2106:** Current node implementations do not allow for a block to be accepted after the maximum 32-bit block time is reached in 2106. At any point before 2106, nodes can gradually update to the BitBlend solution at their convenience as they remain synchronized with non-upgraded nodes during the upgrade. In the situation where both upgraded and non-upgraded nodes coexist in the network at the overflow event in 2106, the non-upgraded nodes will cease to function, unable to process blocks with timestamps exceeding the 32-bit maximum. Conversely, upgraded nodes will continue to operate, interpreting and validating timestamps correctly. Consequently, the occurrence in 2106 will not lead to a split in the blockchain with two separate continuations among the upgraded and non-upgraded nodes; instead, only the path followed by the upgraded nodes will persist.

**Time Lock Implications:**

The BitBlend solution effectively preserves the functionality of block height locks and relative time locks beyond 2106. The absolute and relative block height locks of nLocktime, OP_CHECKLOCKTIMEVERIFY (CLTV), nSequence, and OP_CHECKSEQUENCEVERIFY (CSV) are unaffected by the 32-bit timestamp overflow. For relative time locks, the BitBlend upgrade facilitates the processing of nSequence in a 64-bit context during preliminary transaction validation, as depicted in Figure 1. The solution proposes the natural phasing out of absolute time locks after they reach their 32-bit limits in 2106, which is a non-disruptive change to the current usage of these features, and decreases bitcoin’s reliance on Earth time. Extending absolute time locks to 64 bits is shown to offer very little benefit compared to the complexity of overhauling the 32-bit script evaluation engine, as most of the niche cases where absolute time locks are significantly different than block height locks can be satisfied by relative time locks. Specific implications for each type of time lock are as follows:

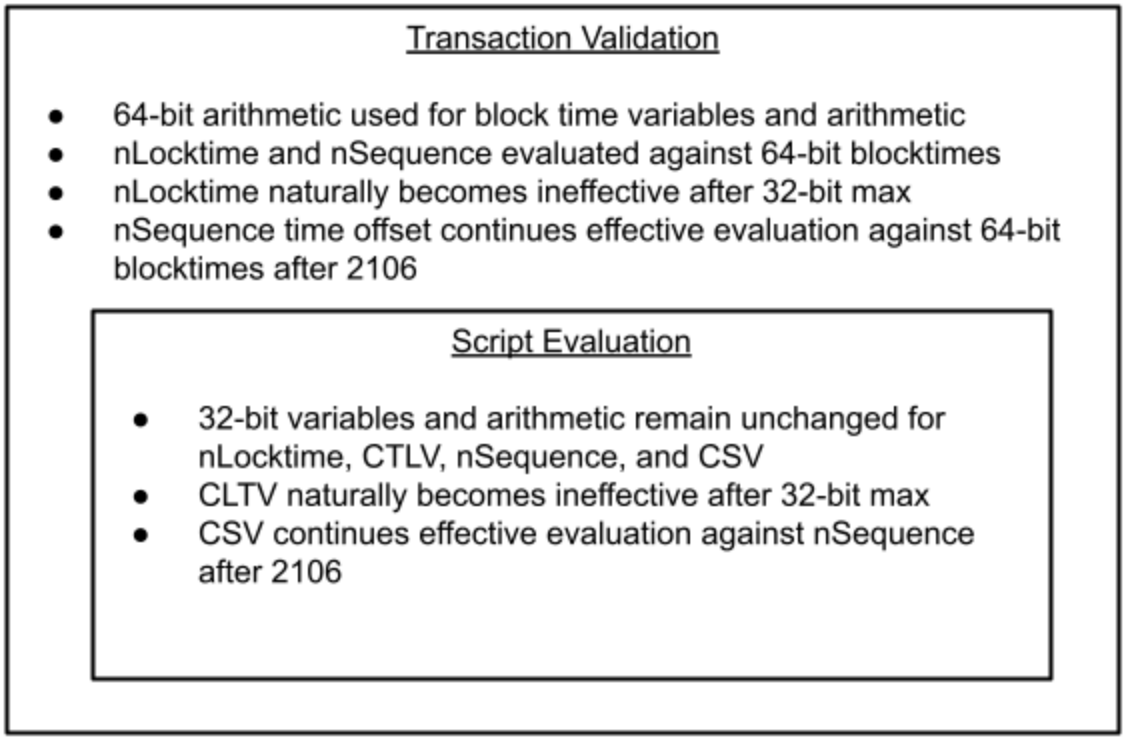


Figure 1: Proposed usage of 64-bit block times in preliminary transaction validation while leaving 32-bit script evaluation unchanged.

- **Persistence of Absolute Block Height Locks:** The functionality of nLocktime and CLTV for absolute block height-based restrictions remains consistent after 2106. If an nLocktime or CLTV value in a transaction is set above zero and is less than 500 million, it continues to signify a constraint on the earliest absolute block height at which the transaction (or output) can be included (or spent) in a block.[5] This aspect of nLocktime and CLTV is not impacted by the overflow of the 32-bit timestamp field in 2106. Consequently, no adjustments are required for this feature until the blockchain reaches a block height of 500 million, a milestone projected to occur in approximately 9,500 years.

- **Persistence of Relative Block Height Locks:** The functionality of nSequence and CSV for relative block height-based restrictions remains consistent after 2106. If the type flags are disabled (set to zero) in bits 32 and 23 of an nSequence or CSV field, it signifies to specify the earliest relative block height at which the input or output is unlocked.[6] This aspect of nSequence and CSV is not impacted by the overflow of the 32-bit timestamp field in 2106. Additionally, because the block heights are 16-bit maximum relative offset to past block heights, they are not constrained by the 500 million block height limitation of nLocktime.

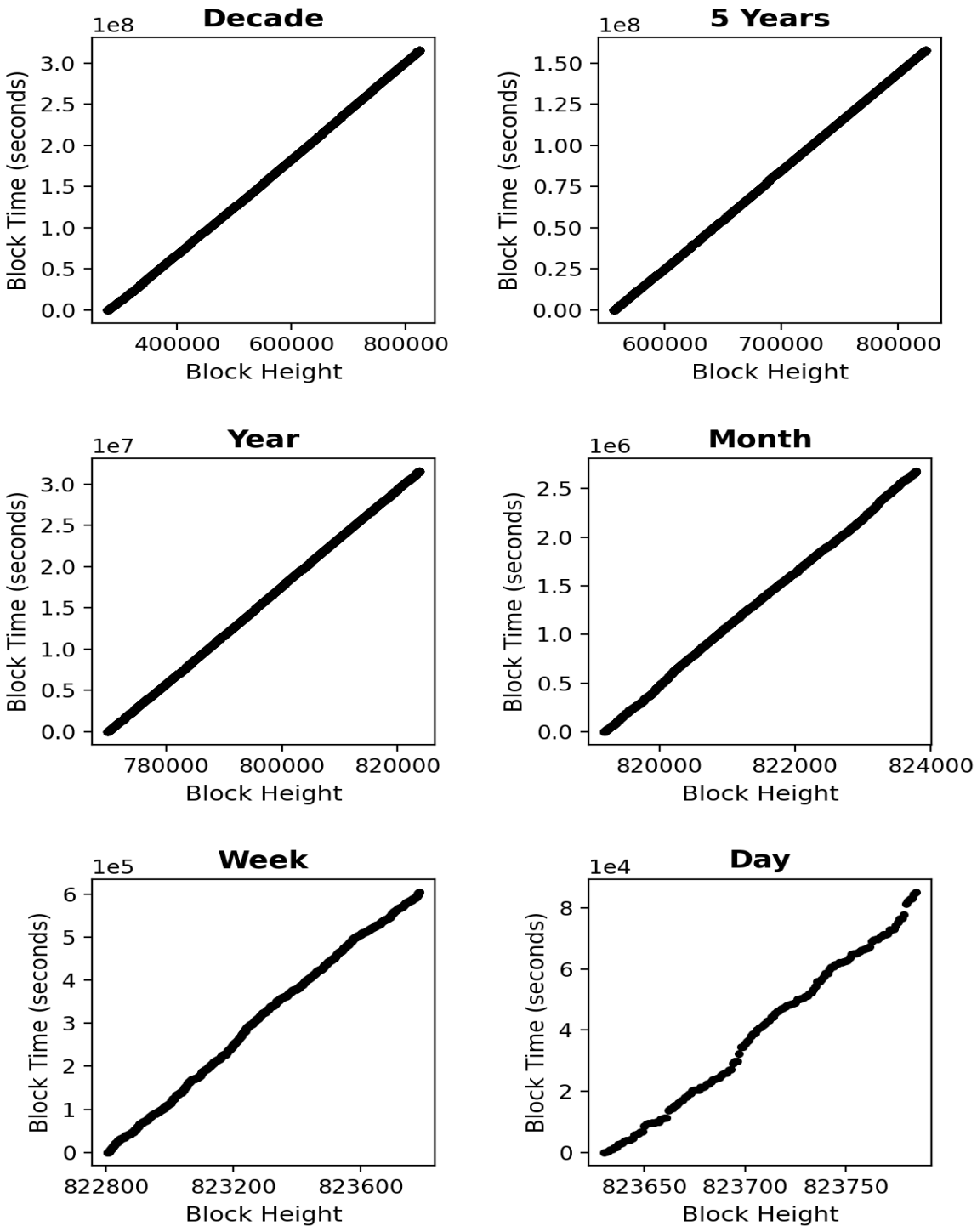


Figure 2: The heights and times of bitcoin blocks show that setting locktimes via block heights instead of block times makes very little difference on timescales from a decade to a day. Block times in seconds (y-axis) are offset from the start of the time period (decade, 5-years, year, month, week, and day) ending on the first block of 1 January 2024 (height 823786).

- **Persistence of Relative Time Locks:** The BitBlend upgrade ensures that nodes maintain the relative time lock functionality of nSequence and CSV after the 2106 event. While the script evaluation engine in bitcoin is confined to 32-bit arithmetic inputs, the validation of nSequence relative time values is conducted in a preliminary part of the transaction validation before script evaluation. This allows for the handling of nSequence time values in a 64-bit context, akin to how difficulty adjustments can be transitioned to 64-bit without necessitating a consensus change. By integrating the 16-bit peak relative time extension of nSequence with the 64-bit timestamps of past blocks, this aspect of transaction validation is safeguarded well beyond the 500 million block height limit of nLocktime. The script evaluation component of transaction validation persistently utilizes 32-bit arithmetic, maintaining the 16-bit maximum time offset values of nSequence and CSV without alteration.

- **Natural Expiration of Absolute Time Locks:** The BitBlend solution advocates for the phasing out of absolute time locks in nLocktime and CTLV after they reach their maximum 32-bit values in 2106. This will not cause disruption to their current operation as it is not currently possible to set a time lock beyond 2106 using these fields. Using the BitBlend solution, the restriction of nLocktime being greater than or equal to the MPT during transaction validation will never be satisfied after the MPT surpasses the 32-bit maximum. If a transaction does not pass the nLocktime check, the CTLV value is not processed as the transaction is already invalid. Thus no disruption or census break will occur by allowing this functionality to naturally expire.

- **Argument Against Expanding Absolute LockTimes to 64 bits:** In order to extend absolute time locks in seconds beyond 2106, nLocktime and CTLV would need to obtain 64-bit precision during script evaluation. While the BitBlend solution for 64-bit block times requires a manageable increase in complexity in order to save bitcoin from halting entirely in 2106, extending nLocktime and CTLV to 64-bits would require a dramatic overhaul of the script evaluation engine. Such an undertaking, if even possible without a coordinated hard fork, would aim to preserve a minor addition to the advanced features of bitcoin, and one that can already be fulfilled via block height and relative time locks except for the smallest of niche cases. Figure 2 shows that block height locks could be substituted for absolute time locks in a large majority of cases as block heights are almost perfectly aligned with block times over timescales ranging from decades to days. For precision on the scale of hours or less, relative time locks, constrained to a 2^16-bit limit (where one bit represents 512 seconds), offer a practical solution, especially for transactions using inputs confirmed within the past year. Therefore, the extensive changes needed for 64-bit absolute time locks may not be justified, considering the availability of effective alternatives and the niche nature of the requirements they serve.

**Conclusion:**

The BitBlend solution offers a pragmatic approach to the Bitcoin 2106 Timestamp Overflow challenge, emphasizing minimal disruption to the network's existing operations. By reinterpreting the 32-bit time field and introducing the BitBlend method for overflow correction, the proposal adeptly extends Bitcoin's timestamp functionality without necessitating a simultaneous hard fork or consensus change. It facilitates a gradual, non-coordinated update process for nodes, ensuring network continuity past 2106. The solution's handling of time locks, particularly its approach towards the natural expiration of absolute time locks and the preservation of block height and relative time locks, aligns with Bitcoin's operational framework. This approach avoids over-complication, focusing on maintaining the core features of Bitcoin in a manner that is both efficient and consistent with the network's foundational principles.

**Support:** To support the development and testing of the BitBlend solution please send donations to:

bc1qr92gh7j46f0ymdgtru0yx59fvjfs8t0zfhkv5a5uda65cgl52mnsrwnck3

**References:**


1. https://github.com/bitcoin/bitcoin/blob/master/src/validation.cpp
2. https://bitcoin.stackexchange.com/questions/106783/will-a-hard-fork-be-required-to-change-timestamp-fields
3. https://lists.linuxfoundation.org/pipermail/bitcoin-dev/2020-September/018172.html
4. https://bitcoin.stackexchange.com/questions/79182/january-19th-2038-rip-unix-timestamps/79185#79185
5. https://github.com/bitcoin/bips/blob/master/bip-0065.mediawiki
6. https://github.com/bitcoin/bips/blob/master/bip-0112.mediawiki