COMPUTER SYSTEMS SECURITY

Chain of Blocks: basics

Bitcoin: A Peer-to-Peer Electronic Cash System

Goal

- Build «*... an electronic payment system based on cryptographic proof instead of trust, allowing any two willing parties to transact directly with each other without the need for a trusted third party*.» (Nakamoto, 2008)

Definition

- «*We define an electronic coin as a chain of digital signatures*.» (Nakamoto, 2008)
- type of **currency**
  - that is money, recognized value!
  - Bitcoin symbol: BTC
  - current value (as of March 2026): 1 BTC ≅ 60 300 EUR
  - 1 BTC = 100,000,000 satoshis[1]

1 also known as *sats*

Components

- Transactions
- Blocks (of Transactions)
- Chain (of Blocks)
- Nodes
- Network (of Nodes)
- Users

Users

- management of (their)
  - crytographic keys and digital money
- how: digital wallets
  - private (user controls everything) - software or hardware
  - custodial (control via broker - or bank!) - they have your private keys!
  - run their own node (even for minting money!)

Network

- interconnection of peer machines (Nodes), of course, running over the Internet
- running Bitcoin open-source software
- exchanging messages (e.g. with Blocks) in a peer-to-peer, delivered on a best effort basis

Node

- any computer running Bitcoin software that participates in the Bitcoin network
- its role, in general:
  - enforcing of Bitcoin's rules
  - maintaining a copy of the Chain of Blocks (*Blockchain*) or part of it
  - validating Transactions and Blocks
  - relaying data to other nodes
- can be of different types (having varying capabilities)
  - Full nodes - copy of whole Bitcoin chain
  - Lightweight nodes - mostly, verification of Blocks
  - Mining nodes - as full nodes plus Mint facility!

Chain (of Blocks)[1]

- «*A chain of blocks with each block referencing the block that preceded it*.» (Developer-gloss, 2020)
- the Bitcoin digital bank: *just* a **global ledger** of the transactions!
- dynamic linear structure whose size does not cease to grow[2], supported by other structures (important, for instance, for efficiency)
- mantained by Nodes

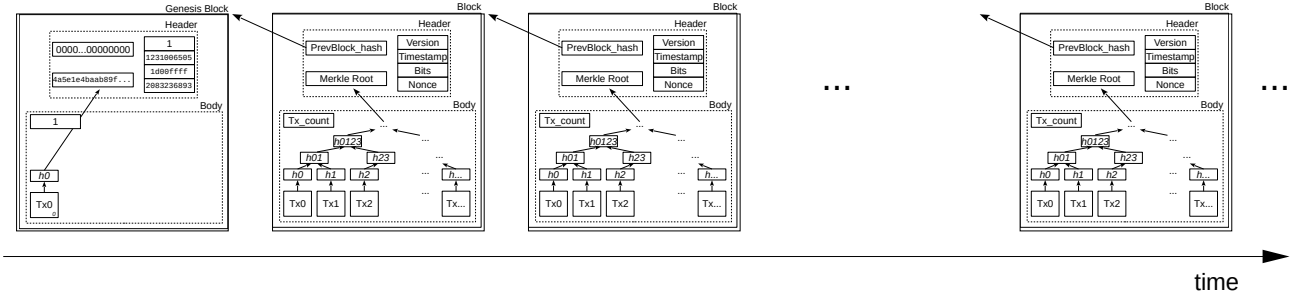


Fig. Bitcoin's Chain of Blocks.

1 usually abreviated to (the overused) *blockchain*
2 as of March 2026, the Bitcoin blockchain has a number of blocks (block height) of, approximately, 942,000, taking about 726 GiB.

Block

- «*One or more transactions prefaced by a block header and protected by proof of work. Blocks are the data stored on the block chain*.» (Developer-gloss, 2020)
- «*As miners construct a new block, they add unverified transactions from ... [a] pool to the new block and then attempt to prove the validity of that new block, with the mining algorithm (Proof-of-Work)*.» (Antonopoulos, 2014)
- So, a block is meant to contain a group of Transactions, each representing the movement of the digital money (from "hand to hand")

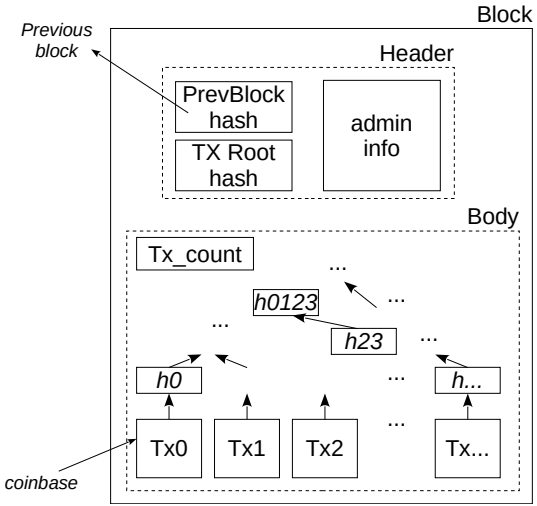


Fig. Basic block. Tx0, first transaction of block, is called *coinbase* transaction and is where new money appears, mined and owned by the builder of the block.

Transaction

- «*Transactions are the most important part of the bitcoin system. Everything else in bitcoin is designed to ensure that transactions can be created, propagated on the network, validated, and finally added to the global ledger of transactions (the blockchain).*» (Antonopoulos, 2014)
- operation by means of which Bitcoin's money changes hands[1]:
  - atom of Bitcoin system's structure, but
  - cannot be "standalone": must be connected to other transactions

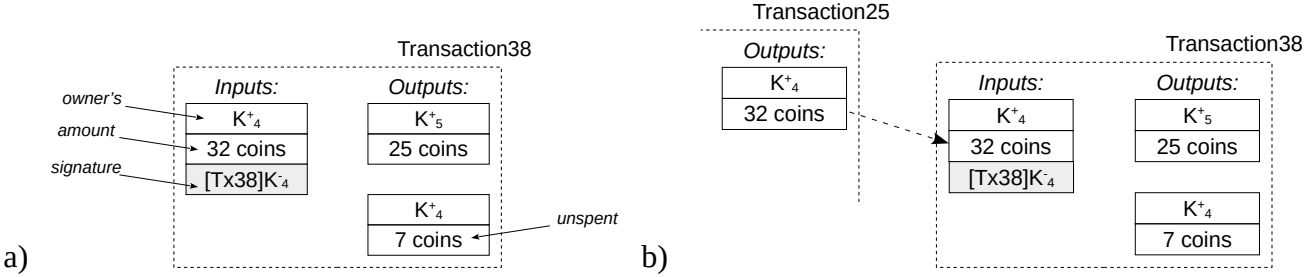


Fig. Basic transaction pictured very simplified-A.
a) main data with a single input owner; b) dashed arrow shows where money is coming from!

1 owners!

...Components: Transaction...

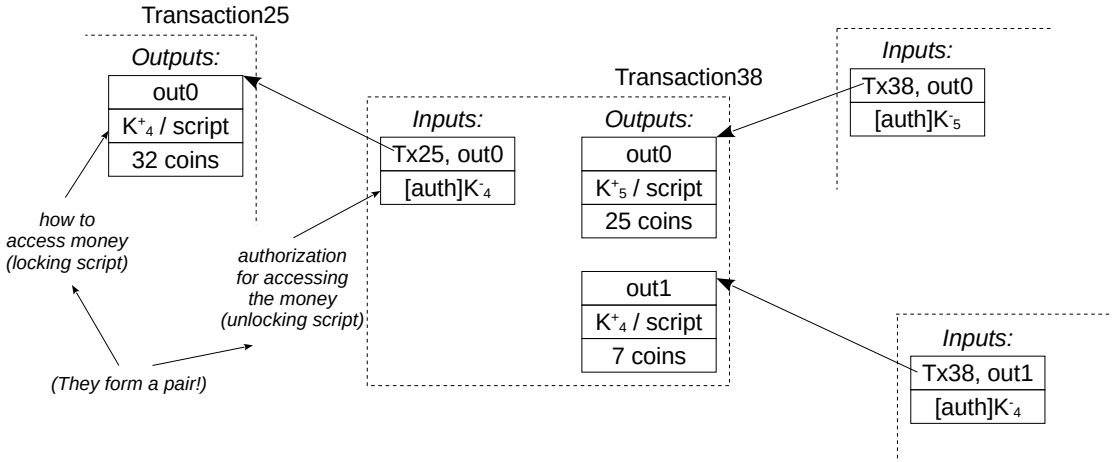
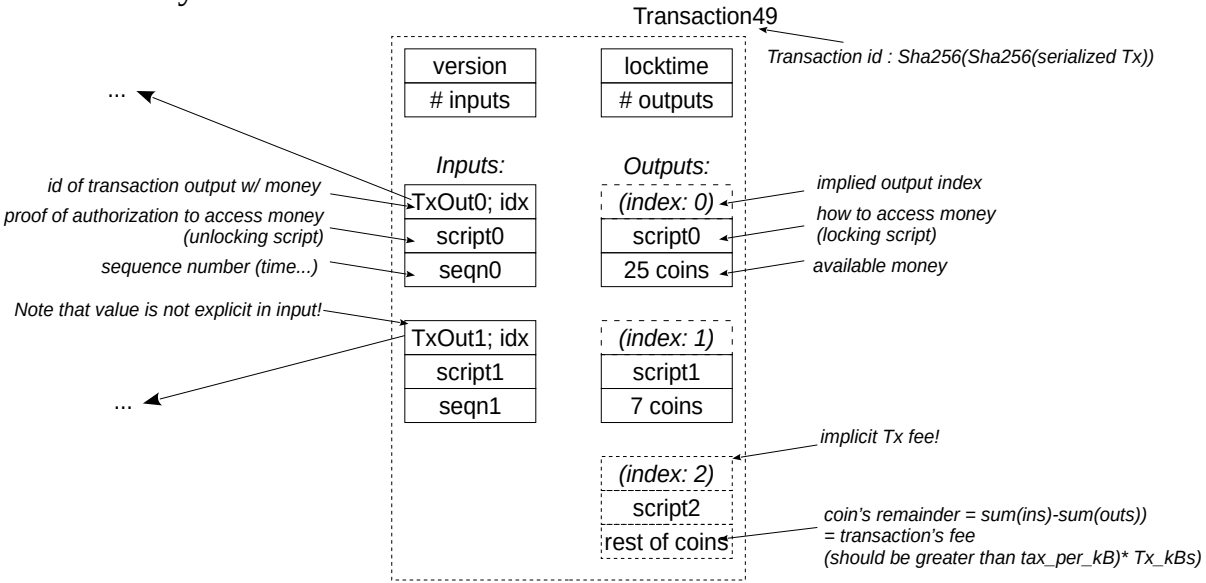


Fig. Transaction pictured a bit more realistically.

Transaction

Structure

- more realistically:



Types

- coinbase
  - 1st transaction in a block
  - always created by a miner
  - no Input, one Output
- general - identified by their address types (inputs and outputs) -> see below
  - Pay-to-Public-Key (P2PK)
  - Pay To Public Key Hash (P2PKH)
  - Pay To Script Hash (P2SH)
  - ...

Signing & Scripting

Signing

- «*Digital signatures are applied to messages, which in the case of bitcoin, are the transactions themselves. The signature implies a commitment by the signer to specific transaction data. In the simplest form, the signature applies to the entire transaction, thereby committing all the inputs, outputs, and other transaction fields. However, a signature can commit to only a subset of the data in a transaction...*» (Antonopoulos, 2014)
- procedure (see previous FIG of Transaction structure):
  - select inputs and outputs to be signed (sighash type)
  - make a copy of the transaction
  - for all inputs, clear scripts xcept 1
  - insert scriptPubKey of available unspent money (UTXO)
  - serialize the transaction into bytes
  - append sighash type
  - double SHA-256 hash of serialized bytes
  - sign hash with private key of owner of input money

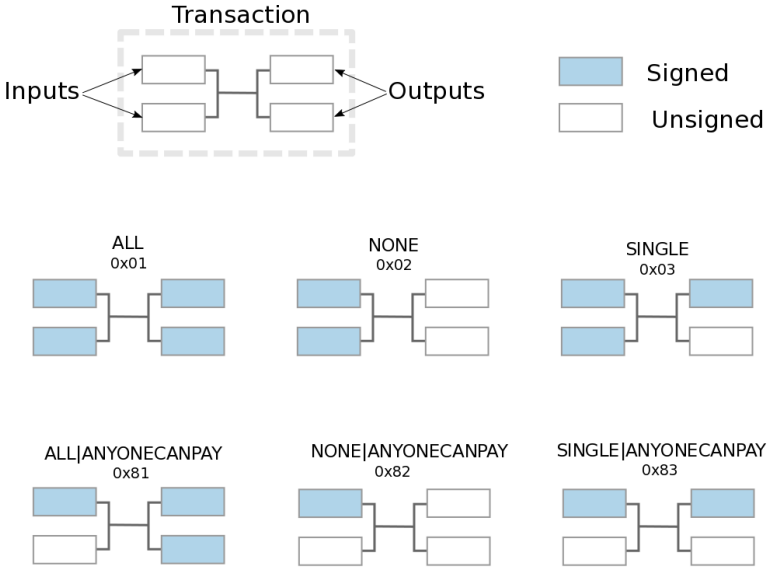
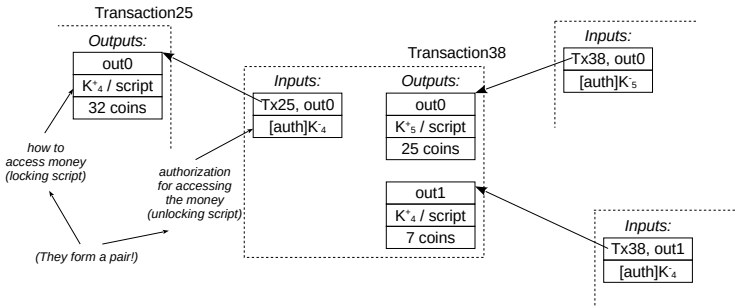


Fig. SIGHASH flag and SIGHASH_ANYONECANPAY modifier combinations on the signing of a transaction (exemplified in a 2-input, 2-output transaction). (in Antonopoulos, 2014)

Scripting

- Scripts are what gives flexibility (power?...) to bitcoin transactions
- each transaction needs both of two types of scripts (they make a "pair"):
  - *Output Script* (*Pubkey Script*, *scriptPubKey*, *Lock Script*)
    - define the conditions for spending money; live in Outputs
  - *Input Script* (*Signature Script*, *scriptSig*, *Unlock Script*)
    - try to) satisfy of the conditions for spending money; live in Inputs
- for spending some money (UTXO) on current TX, the script's pair is:
  - scriptPubKey of previous TX
  - scriptSig of current TX

  ---> see a previous picture:



Types

- Public Key or Pay-to-Public-Key (P2PK)
  - initial form of payment
  - normal
    - input contains recipient's signature
    - output contains recipient's pubkey
  - (bare) multisig
    - input contains *m* (up to *n*) signatures (of different recipients)
    - output contains *n* (greater or equal to *m*) signatures (of different recipients)

--> Figs next page

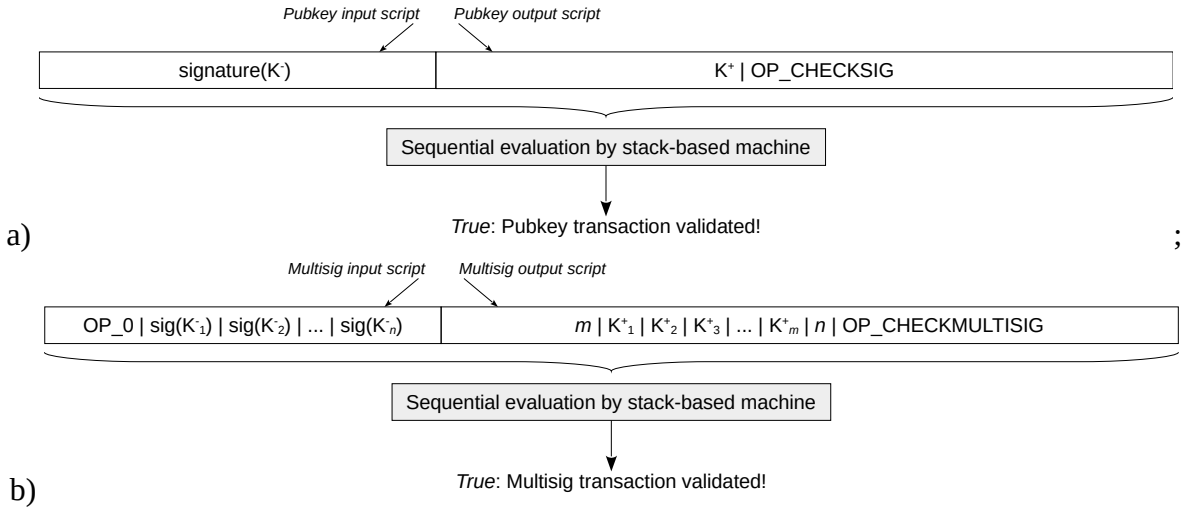


Fig. Pubkey type of scripts: a) normal (simple); b) multisig.
Note: $[TX]_{user}$ means *Signature of TX by user* (with K^-_{user}, of course).

...Scripting: types (cont.)...

- Pay To Public Key Hash (P2PKH)
  - enhanced form of payment:
    - more resistance to attacks and privacy
    - shorter addresses (recipient's of money transfer)
    - built-in error-detection (via checksums)
  - normal
    - input contains recipient’s pubkey and a recipient's signature of TX (including previous output’s scriptPubKey)
    - output contains recipient’s pubkey hash
      - Hex: `76a914 <20-byte pubKeyHash> 88ac`

--> *Figs next page*

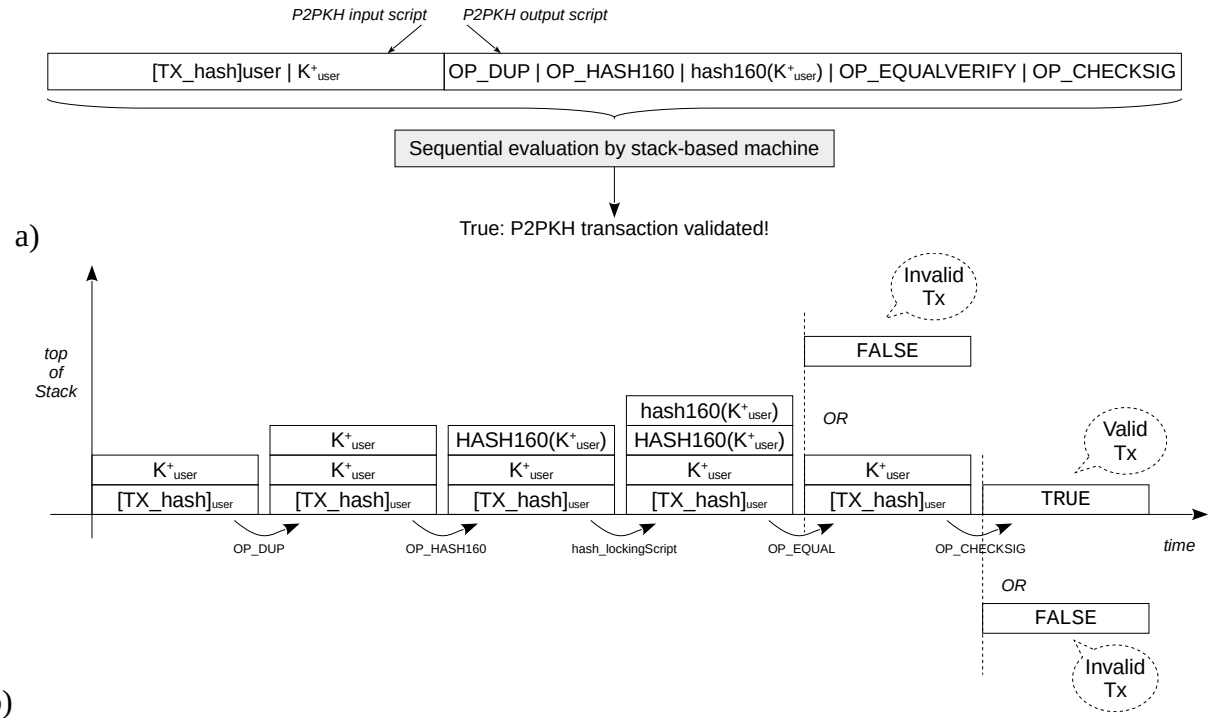


Fig. Pay To Public Key Hash (P2PKH) type of scripts: a) scripts structure; b) Sequential Evaluation of Input Script by stack-based machine.

...Scripting: types (cont.)...

- Pay To Script Hash (P2SH)
  - normal
    - input contains recipient's *Redeem Script* and a signature
    - output contains recipient's *Redeem Script* hash
  - multisig pubkey script (most used)
  - allows storing of textual data on the blockchain (up to 1.5kb of text data)

  --> *Figs next page*

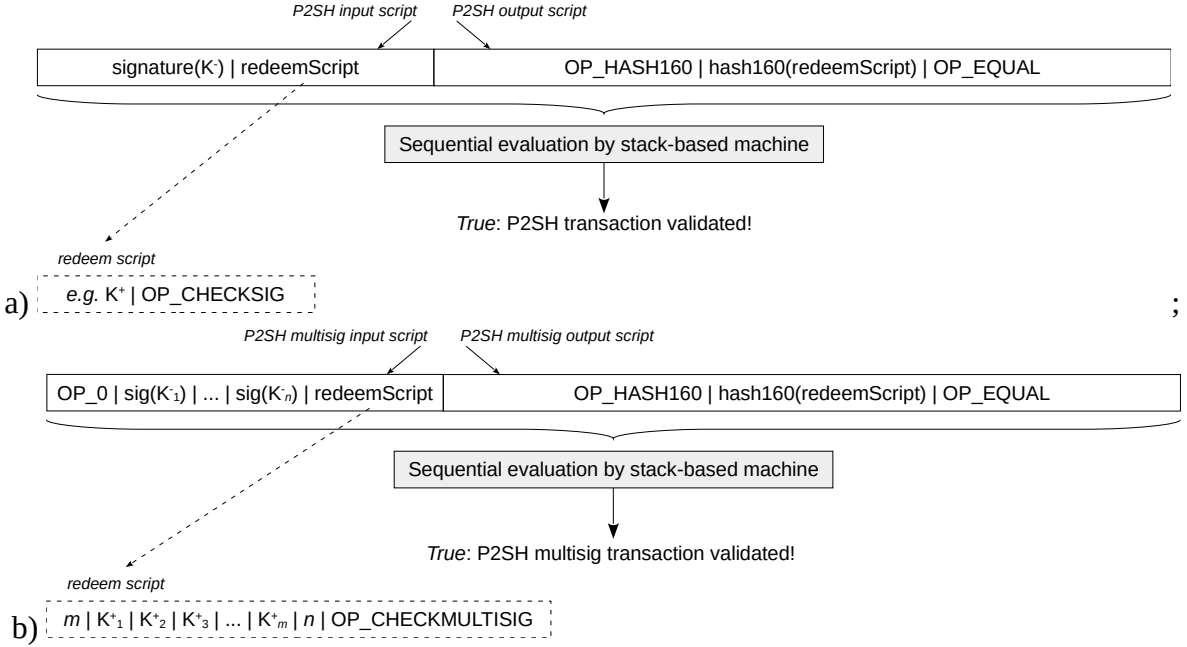


Fig. Pay To Script Hash (P2SH) type of scripts: a) normal (simple); b) multisig.

- Null Data (OP_RETURN or Data carrier)
  - «*adds arbitrary data to a provably unspendable pubkey script that full nodes don't have to store in their UTXO database*» (Antonopoulos, 2014)
  - there is no input (unlocking) script, as there is no money to spend!
  - output script has arbitrary data with limited size (tens of bytes, up to ~10kB); value is zero (any other value is lost!)
  - allows storing of textual data on the blockchain!!!

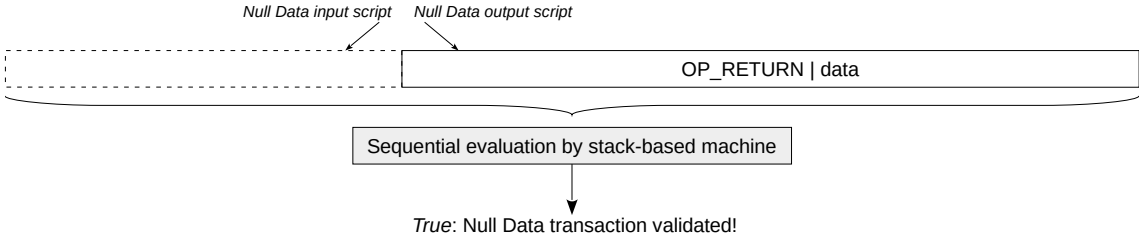


Fig. Null Data type of script. Data is of limited size.

Block

Structure

- Block Header (80 bytes)
  - version - different version, different Block validation rules[1]
  - previous_block_hash - linking of chain of blocks!
  - Merkle Root - represents all transactions in Block
  - Timestamp - Unix creation time
  - Bits (4 bytes) - encoded difficulty *target* [2]
    - bits = [exponent (1 byte)][coefficient (3 bytes)]
    - target = C × 256^(E - 3)
    - mining: valid block_hash ≤ target (Proof-of-work!)
    - adjusts every 2016 blocks (~2 weeks)

--> *Fig next page*

1 examples: v1- Original Bitcoin rules; v2 - P2SH + coinbase height; v3 - Strict DER signatures; ...
2 sets how hard is mining - the smaller, the more difficult!

- Nonce - changed by miners to try different hashes for Proof-of-work
  - See Mining below
- block_hash =
  SHA256 (SHA256 (block_header))
- Body
  - Transaction count
  - TX0 (coinbase), TX1, TX2...
  - hashes... in Merkle tree technique

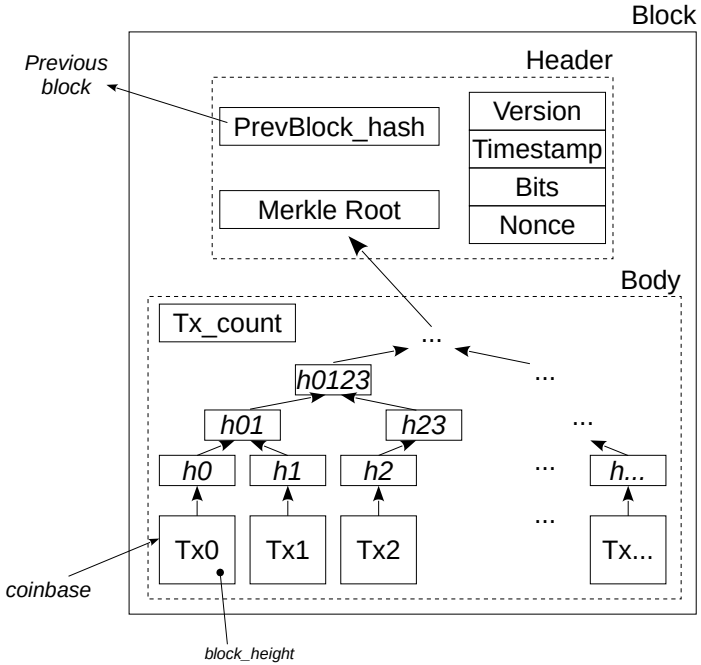


Fig. Basic block. Tx0, first transaction of block, is called *coinbase* and is where new money appears, mined and owned by the builder of the block; *blockheight* is number of block in chain of blocks!.

Mining

- building a Block to be connected to the Chain of Blocks
  - ... and collect the *subsidy* (reward for mining)
- Process:
  - assemble a candidate block
    - set transactions
    - compute Merkle root
    - set version, previous block, timestamp, bits
  - Hash the header with the current nonce:
    - `block_hash = SHA256(SHA256(block_header))`
  - Check if block_hash ≤ target(bits)
    - Yes → block found
    - No → change nonce[1] → repeat Process

1 If necessary, miners can also adjust *coinbase's extra data* and *Timestamp*.

Chain (of Blocks)

Structure

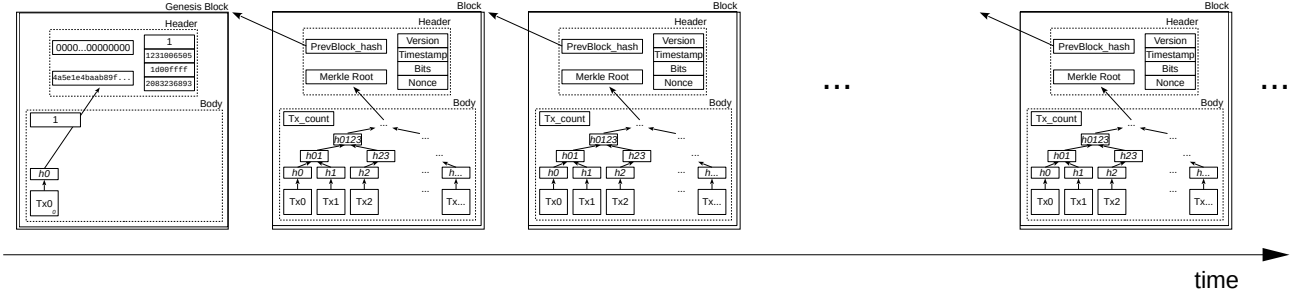


Fig. Bitcoin's Chain of Blocks.

Consensus Rules

- *The laws of the Bitcoin system!*
- enforced by every node: determine validity of blocks and transactions
- define the decentralized shared agreement
- Types:
  - Transaction rules
  - Block rules
  - Monetary rules
  - Chain rules

Consensus rules' types:

- Transaction rules
  - inputs must exist in the UTXO set (previous TX)
  - signatures must be valid
  - no double spending
  - scripts must evaluate to true
  - total outputs ≤ total inputs
- Block rules
  - block hash must satisfy difficulty
    - difficulty (Bits-target) must be same as value related to the calculated times of previous 2,016 blocks!
  - block size / weight limits respected
  - Merkle root must match transactions
  - first transaction must be Coinbase
  - block reward must be correct

Consensus rules' types (cont.):

- Monetary rules
  - block subsidy follows halving schedule
  - total supply capped (~21 million BTC)
  - no creation of coins outside Coinbase
- Chain rules
  - each block must reference a valid previous block
  - nodes follow the chain with the most *cumulative work*

Chain (temporary) fork

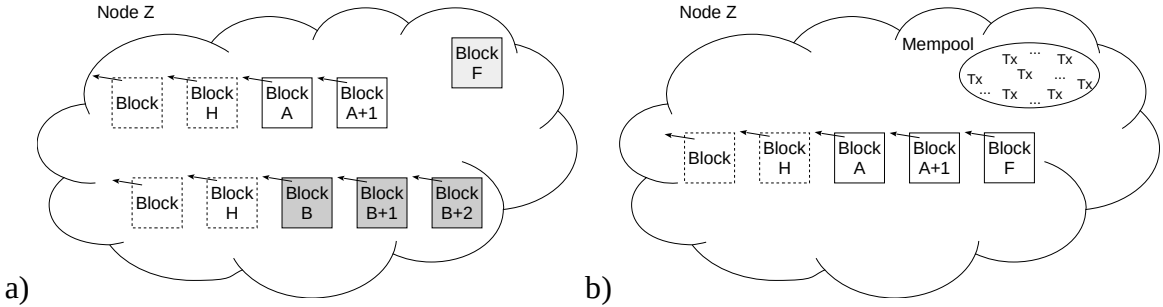


Fig. Bitcoin's Chain temporary fork at Node Z:
a) conflict situation;
b) final tie-breaker chain, e.g. as cumulative work of chain with A-series blocks is greater than chain's with B-series blocks. Transactions fromB-series blocks are kept in Mempool to be added to future block.

Major details

Hard limit on Bitcoin total amount of money: ~21 million BTC

...

Cumulative work defines chain choosing

...

10-minute block interval between new block formation

...

Addresses & encoding

---> *next page*

Addresses & encoding

- Bitcoin address: the recipient of a money transfer!
- recipient can be:
  - an entity (represented by a public key) or
  - a script (set of rules, eventually relating to one or more entities).
- address: convolved fingerprint (hash) of a recipient's public key or a (redeem) script, both written in a *Base58Check* format that is human-readable
  - see FIG
  - HASH160 (m) = RIPEMD-160(SHA256(m))
  - Base58Check (m) = Base58 encode (version | m | checksum (version | m))
    - checksum (version | m) = 1st 4 B of SHA256(SHA256(version | m)))
    - Base58 encode(): `123456789ABCDEFGHJKLMNPQRSTUVWXYZabcdefghijkmnopqrstuvwxyz`[1]

1 Note: no 0 (zero) , O (capital o) , I (capital i), l (lowercase L) -- avoids visual confusion

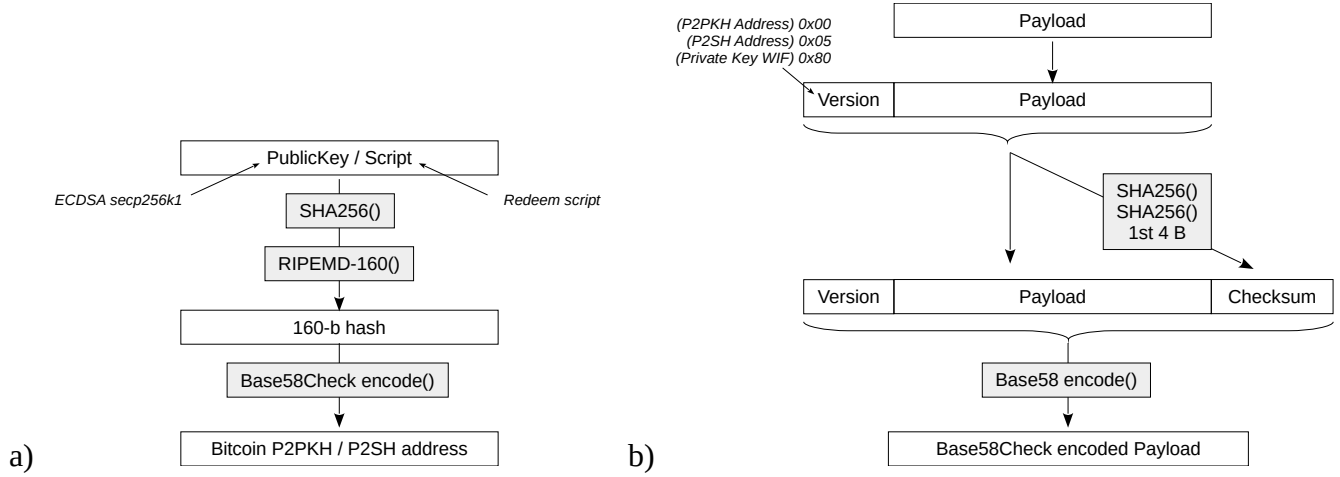


Fig. Bitcoin address building:
a) general construction;
b) detailing of Base58Check encoding (reversible operation). The payload can be a hash, such as the shown in a) or something else.

Pointers...

- "**Bitcoin: A Peer-to-Peer Electronic Cash System**", 2008 – S. Nakamoto
  - https://bitcoin.org/bitcoin.pdf
- "**Bitcoin**", 2026 – Wikipedia
  - https://en.wikipedia.org/wiki/Bitcoin
- The "**Bitcoin Developer's Glossary**", 2020, Bitcoin Project
  - https://developer.bitcoin.org/glossary.html
- The "**Bitcoin Developer's Reference**", 2020, Bitcoin Project
  - https://developer.bitcoin.org/reference/
- "**Mastering Bitcoin**", 2nd Ed., 2017 – Andreas M. Antonopoulos
  - https://www.oreilly.com/library/view/mastering-bitcoin/9781491902639,
  - https://github.com/bitcoinbook/bitcoinbook