Introduction to Bitcoins

- **Introducing Bitcoins:** Bitcoin definition, Digitalkeys and addresses: Private keys in Bitcoin, Public keys in Bitcoin, Addresses in Bitcoin, Transactions: Transaction lifecycle, Transaction data structure, Types of transaction, transaction verification. Blockchain: structure of a block, structure of block header, genesis block. Mining: Task of miners, mining rewards, Proof of work, mining algorithm, hash rate, mining systems, mining pools.
- **Bitcoin Network and Payments:** The bitcoin network, wallets: Non deterministic wallets, deterministic wallet, Hierarchical deterministic wallet, brain wallet, paper wallet, hardware wallet, online wallet, mobile wallet. Bitcoin payments, Innovation in Bitcoin: Bitcoin Improvement Proposals, advanced protocols, segregated witness, bitcoin cash, bitcoin unlimited, bitcoin gold, bitcoin investment-buying and selling bitcoins**Textbook1:**Ch 5, Ch 6

Bitcoin definition

- Bitcoin can be defined in various ways; it's a protocol, a digital currency, and a platform.
- It is a combination of peer-to-peer network, protocols, software that facilitate the creation and usage of the digital currency named bitcoin.
- Decentralization of currency was made possible for the first time with the invention of bitcoin.
- Double spending problem was solved in an elegant and ingenious way in bitcoin.
- Double spending problem arises when, for example, a user sends coins to two different users at the same time and they are verified independently as valid transactions.

Components of Bitcoin

Bitcoin is composed of these elements:

- Digital keys
- Addresses
- Transactions
- Blockchain
- Miners
- The Bitcoin network
- Wallets (client software)

Digital Keys and addresses

- On the Bitcoin network, possession of bitcoins and transfer of value via transactions is reliant upon private keys, public keys, and addresses.
- Elliptic Curve Cryptography (ECC) is used to generate public and private key pairs in the Bitcoin network.

Private keys in Bitcoin

- Private keys are required to be kept safe and normally resides only on the owner's side.
- Private keys are used to digitally sign the transactions proving the ownership of the bitcoins.
- Private keys are fundamentally 256-bit numbers randomly chosen in the range specified by the secp256k1 ECDSA curve recommendation.
- Private keys are usually encoded using Wallet Import Format (WIF) in order to make them easier to copy and use.
- WIF can be converted into a private key and vice versa. The private key encoded using mini private key format is also sometimes called minikey.

A Casascius physical bitcoin's security hologram paper with minikey and QR code

Public keys in Bitcoin

- Public keys exist on the blockchain and all network participants can see it.
- Public keys are derived from private keys due to their special mathematical relationship with the private keys.
- Once a transaction signed with the private key is broadcasted on the Bitcoin network, public keys are used by the nodes to verify that the transaction has indeed been signed with the corresponding private key. This process of verification proves the ownership of the bitcoin.
- This makes use of ECDSA to ensure that funds remain secure and can only be spent by the legitimate owner.
- Initially, Bitcoin client used uncompressed keys, but starting from Bitcoin core client 0.6, compressed keys are used as standard. This resulted in almost 50% reduction of space used to store public keys in the blockchain.

Addresses in Bitcoin

- A bitcoin address is created by taking the corresponding public key of a private key and hashing it twice, first with the SHA-256 algorithm and then with RIPEMD-160. The resultant 160-bit hash is then prefixed with a version number and finally encoded with a Base58Check encoding scheme.
- A typical bitcoin address looks like a string shown here: 1ANAguGG8bikEv2fYsTBnRUmx7QUcK58wt
- This is also commonly encoded in a QR code for easy distribution.

Transaction

- Transactions are at the core of the bitcoin ecosystem.
- Transactions can be as simple as just sending some bitcoins to a bitcoin address, or it can be quite complex depending on the requirements.
- Each transaction is composed of at least one input and output.
- Inputs can be thought of as coins being spent that have been created in a previous transaction and outputs as coins being created.
- If a transaction is minting new coins, then there is no input and therefore no signature is needed.
- If a transaction is to send coins to some other user (a bitcoin address), then it needs to be signed by the sender with their private key and a reference is also required to the previous transaction in order to show the origin of the coins.

The Transaction Life Cycle

These are the steps:

- A user/sender sends a transaction using wallet software or some other interface.
- The wallet software signs the transaction using the sender's private key.
- The transaction is broadcasted to the Bitcoin network using a flooding algorithm.
- Mining nodes (miners) who are listening for the transactions verify and include this transaction in the next block to be mined.
- Just before the transaction are placed in the block they are placed in a special memory buffer called transaction pool.

- Mining starts, which is a process by which the blockchain is secured and new coins are generated as a reward for the miners who spend appropriate computational resources.
- Once a miner solves the PoW problem it broadcasts the newly mined block to the network.
- The nodes verify the block and propagate the block further, and confirmations start to generate.
- Finally, the confirmations start to appear in the receiver's wallet and after approximately three confirmations, the transaction is considered finalized and confirmed.
- The key idea behind waiting for six confirmations is that the probability of double spending is virtually eliminated after three confirmations.

The Transaction Fee

- Transaction fees are charged by the miners.
- The fee charged is dependent upon the size and weight of the transaction.
  - Size: The amount of data the transaction represents, measured in bytes.
  - Weight:It is a unit of measurement used to compare transaction sizes.
- The fees are used as an incentive for miners to encourage them to include a user transaction in the block the miners are creating.
- All transactions end up in the memory pool, from where miners pick up transactions based on their priority to include them in the proposed block.
- Fees are not fixed by the Bitcoin protocol and are not mandatory; even a transaction with no fee will be processed due course but may take a very long time. This is however no longer practical due to the high volume of transactions and competing investors on the Bitcoin network, therefore it is advisable to provide a fee always.
- The time for transaction confirmation usually ranges from 10 minutes to over 12 hours in some cases.
- Transaction time is dependent on transaction fees and network activity. If the network is very busy then naturally transactions will take longer to process and if you pay a higher fee then your transaction is more likely to be picked by miners first due to additional incentive of the higher fee.

The Transaction pools

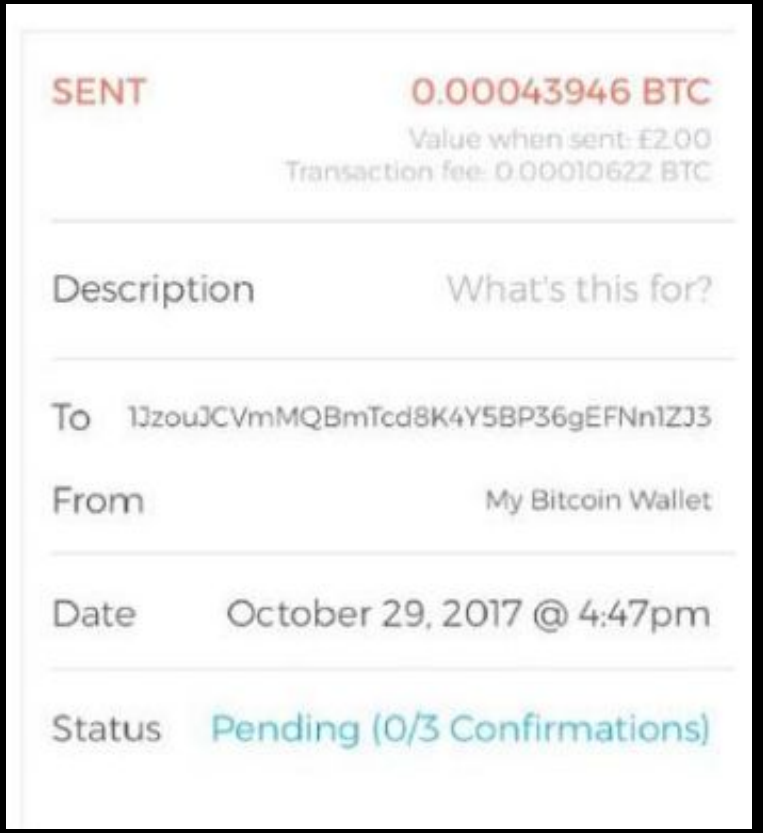
- Also known as memory pools, these pools are basically created in local memory by nodes in order to maintain a temporary list of transactions that are not yet confirmed in a block.
- Transactions are included in a block after passing verification and based on their priority.

Sending a payment to someone

- First, either the payment is requested from a user by sending his Bitcoin address to the sender via email or some other means such as SMS, chat applications or in fact any appropriate communication mechanism.
- The sender can also initiate a transfer to send money to another user.
- In both cases, the address of beneficiary is required.

- The sender either enters the receiver's address or scans the QR code that has the Bitcoin address, amount and optional description encoded in it.
- The wallet application recognizes this QR code and decodes it.
- In the wallet application of the sender, this transaction is constructed by following some rules and broadcasted to the Bitcoin network.
- This transaction is digitally signed using the private key of the sender before broadcasting it.

- Fee in Bitcoin network ensures that your transaction will be included by miners in the block.
- a Recently the Bitcoin fees were so high that even for smaller transactions a high amount of fee was charged. This was due to the fact that miners are free to choose which transactions they pick to verify and add in a block, and they select the ones with higher fees.
- The higher the transaction fee the more chances are that your transaction will be picked up at priority and included in the block. This task is performed by the miners.



- At this stage, the transaction has been constructed, signed and sent out to the Bitcoin network. This transaction will be picked up by miners to be verified and included in the block.
- Also note that in the preceding screenshot, confirmation is pending for this transaction. These confirmations will start to appear as soon as the transaction is verified, included in the block, and mined.
- Also, the appropriate fee will be deducted from the original value to be transferred and will be paid to the miner who has included it in the block for mining.

Transaction Data structure

Field	Size	Description
Version Number	4 bytes	Used to specify rules to be used by the miners and nodes for transaction processing.
Input counter	1-9 bytes	The number (positive integer) of inputs included in the transaction.
List of inputs	Variable	Each input is composed of several fields, including Previous Tx hash, Previous Txout-index, Txin-script length, Txin-script, and optional sequence number. The first transaction in a block is also called a Coinbase transaction. It specifies one or more transaction inputs.
Output counter	1-9 bytes	A positive integer representing the number of outputs.
List of outputs	Variable	Outputs included in the transaction.
Lock time	4 bytes	This field defines the earliest time when a transaction becomes valid.

- **1. Version Number (4 bytes)**
- Specifies the rules or format version used for this transaction.
- It helps miners and nodes know how to interpret the transaction.
- **2. Input Counter (1–9 bytes)**
- Indicates how many **inputs** this transaction has.
- An input means where the Bitcoin is coming from (i.e., referencing a previous transaction output).
- Since a person can combine multiple past outputs to make one payment, the counter tells the network how many such references exist.
- **3. List of Inputs (Variable size)**
- Contains the actual input details.
- Each input includes:
  - **Previous Tx hash** → the transaction ID where the Bitcoin originally came from.
  - **Txout-index** → the specific output number in that previous transaction.
  - **Txin-script length** and **Txin-script** → a script (often a digital signature + public key) proving ownership of the Bitcoin.
  - **Sequence number** (optional) → used for advanced features like transaction replacement.
- Special case: The first transaction in a block (called the **coinbase transaction**) does not reference earlier outputs but instead generates new Bitcoins as a reward for the miner.
- **4. Output Counter (1–9 bytes)**
- A positive integer showing how many **outputs** the transaction creates.
- Outputs specify where the Bitcoins are going.
- **5. List of Outputs (Variable size)**
- Contains all outputs of the transaction.
- Each output specifies:
  - The amount of Bitcoin being sent.
  - A locking script (scriptPubKey) that defines the conditions under which the Bitcoin can be spent in the future (usually requiring the recipient's private key to unlock).
- **6. Lock Time (4 bytes)**
- Defines the earliest time or block number when the transaction becomes valid.
- If lock time = 0 → transaction is valid immediately.
- If set to a future block height or timestamp, the transaction cannot be confirmed until that time.

Transaction Inputs data structure

Field	Size	Description
Transaction hash	32 bytes	This is the hash of the previous transaction with UTXO.
Output index	4 bytes	This is the previous transactions output index, that is, UTXO to be spent.
Script length	1-9 bytes	This is the size of the unlocking script.
Unlocking script	Variable	Input script (ScriptSig) which satisfies the requirements of the locking script.
Sequence number	4 bytes	Usually disabled or contains lock time. Disabled is represented by '0xFFFFFFFF'.

- **1. Transaction Hash (32 bytes)**
- This is the **hash (ID)** of the previous transaction that created the UTXO.
- It points back to where the coins originally came from.
-

- **2. Output Index (4 bytes)**
- Each transaction can have multiple outputs (e.g., 2 BTC to Alice, 1 BTC to Bob).
- The **index** specifies *which output* from the previous transaction is being spent.

- **3. Script Length (1–9 bytes)**
- Indicates the length (in bytes) of the unlocking script that follows.
- Since scripts can vary in size, this field tells nodes how much data to read for the unlocking script.

4. Unlocking Script (Variable)

- Also called **ScriptSig**.
- This is the "proof" that you own the coins being spent.
- Typically includes:
  - A digital signature (signed with the sender's private key).
  - The sender's public key.
- This script must satisfy the conditions set in the locking script (scriptPubKey) of the previous output.

- **5. Sequence Number (4 bytes)**
- Originally meant for updating transactions before they're confirmed (replace-by-fee).
- Commonly set to its maximum value (0xFFFFFFFF), meaning disabled.

Transaction Output data structure

- Outputs have three fields, and they contain instructions for sending bitcoins.
- The first field contains the amount of Satoshis
- the second field contains the size of the locking script.
- Finally, the third field contains a locking script that holds the conditions that need to be met in order for the output to be spent.

Field	Size	Description
Value	8 bytes	Total number in positive integers of Satoshis to be transferred
Script size	1-9 bytes	Size of the locking script
Locking script	Variable	Output script (ScriptPubKey)

Types of Transaction

- Pay to Public Key Hash (P2PKH):
  - P2PKH is the most commonly used transaction type and is used to send transactions to the bitcoin addresses.
- Pay to Script Hash (P2SH):
  - P2SH is used in order to send transactions to a script hash (that is, the addresses starting with 3) and was standardized in BIP16. In addition to passing the script, the redeem script is also evaluated and must be valid.
- MultiSig (Pay to MultiSig):
  - M-of-N MultiSig transaction script is a complex type of script where it is possible to construct a script that required multiple signatures to be valid in order to redeem a transaction. Various complex transactions such as escrow and deposits can be built using this script.
- Pay to Pubkey:
  - This script is a very simple script that is commonly used in coinbase transactions. It is now obsolete and was used in an old version of bitcoin. The public key is stored within the script in this case, and the unlocking script is required to sign the transaction with the private key.
- Null data/OP_RETURN:
  - This script is used to store arbitrary data on the blockchain for a fee. The limit of the message is 40 bytes. The output of this script is unredeemable because OP_RETURN will fail the validation. This is used to store contracts.

Transaction verification

- 1. Check the syntax and ensure that the syntax and data structure of the transaction conforms to the rules provided by the protocol.
- 2. Verify that no transaction inputs and outputs are empty.
- 3. Check whether the size in bytes is less than the maximum block size.
- 4. The output value must be in the allowed money range (0 to 21 million BTC).
- 5. All inputs must have a specified previous output, except for coinbase transactions, which should not be relayed.
- 6. Verify that nLockTime must not exceed 31-bits. (nLockTime specifies the time before which transaction will not be included in the block.)
- 7. For a transaction to be valid, it should not be less than 100 bytes.
- 8. The number of signature operations in a standard transaction should be less than or not more than two.
- 9. Reject nonstandard transactions; The isStandard() checks specify that only standard transactions are allowed.

- 10. A transaction is rejected if there is already a matching transaction in the pool or in a block in the main branch.
- 11. The transaction will be rejected if the referenced output for each input exists in any other transaction in the pool.
- 12. For each input, there must exist a referenced output unspent transaction.
- 13. For each input, if the referenced output transaction is the coinbase, it must have at least 100 confirmations; otherwise, the transaction will be rejected.
- 14. For each input, if the referenced output does not exist or has been spent already, the transaction will be rejected.
- 15. Using the referenced output transactions to get input values, verify that each input value, as well as the sum, is in the allowed range of 0-21 million BTC. Reject the transaction if the sum of input values is less than the sum of output values.
- 16. Reject the transaction if the transaction fee would be too low to get into an empty block.
- 17. Each input unlocking script must have corresponding valid output scripts.

Bitcoin: Structure of a block

Field	Size	Description
Block size	4 bytes	This is the size of the block.
Block header	80 bytes	This includes fields from the block header described in the next section.
Transaction counter	Variable	This field contains the total number of transactions in the block, including the coinbase transaction. Size ranges from 1-9 bytes
Transactions	Variable	All transactions in the block.

- **1. Block size (4 bytes)**
- This tells us **how large the block is** in bytes.
- Maximum block size in Bitcoin ≈ 1 MB Example: If the block size is 800 KB, this field records that number.

- **2. Block header (80 bytes)**
- A fixed-size part of every block.
- Contains **metadata** about the block, not the transactions themselves.
- Includes fields like:
  - Version
  - Previous block hash
  - Merkle root (summary of all transactions)
  - Timestamp
  - Difficulty target
  - Nonce (used in mining)
- Think of it like a **summary page** for the whole block.

- **3. Transaction counter (Variable, 1–9 bytes)**
- Records **how many transactions are inside the block**.
- This includes the **coinbase transaction**
- Example: If there are 2,000 transactions in this block, this field will store that number.

- **4. Transactions (Variable)**
- This is the **actual list of all transactions** inside the block.
- Each transaction has its own structure (inputs, outputs, scripts, etc.).
- Size is variable because different blocks can have different numbers of transactions.

Bitcoin: Structure of the block header

Field	Size	Description
Version	4 bytes	The block version number that dictates the block validation rules to follow.
Previous block's header hash	32 bytes	This is a double SHA-256 hash of the previous block's header.
Merkle root hash	32 bytes	This is a double SHA-256 hash of the Merkle tree of all transactions included in the block.
Timestamp	4 bytes	This field contains the approximate creation time of the block in the Unix epoch time format. More precisely, this is the time when the miner has started hashing the header. (The time from the miner's point of view.)
Difficulty target	4 bytes	This is the current difficulty target of the network/block.
Nonce	4 bytes	This is an arbitrary number that miners change repeatedly to produce a hash that is lower than the difficulty target.

- Version
- Indicates which set of block validation rules to follow.
- Helps in upgrading Bitcoin protocol

- Previous Block Hash
- A 256-bit hash of the previous block's header.
- Ensures blocks are linked together → creates the blockchain.

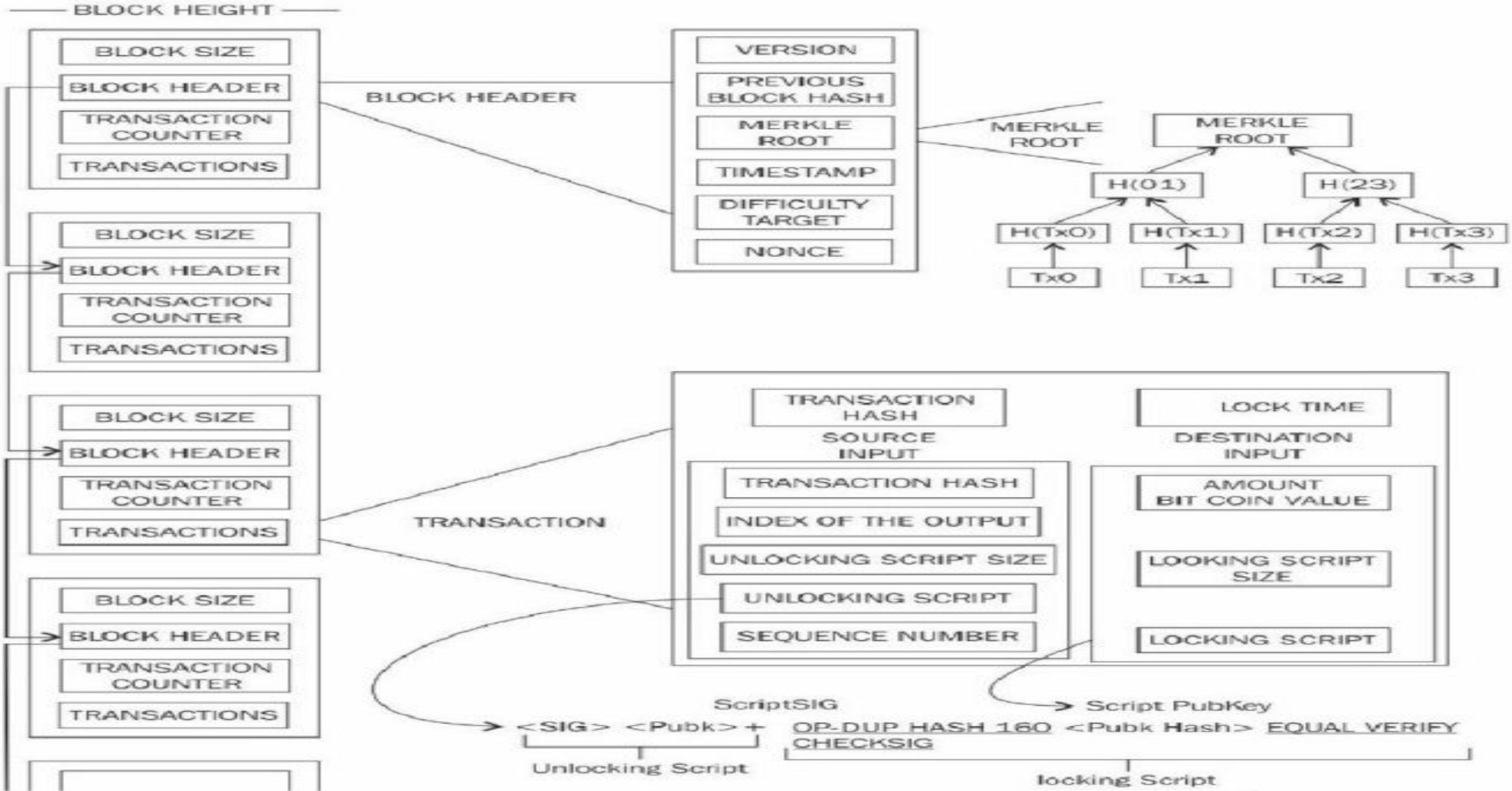
- Merkle Root
- A single hash summarizing all transactions in the block.
- Built using a Merkle tree, where every transaction hash is combined until only one root hash remains.

- Timestamp
- Approximate creation time of the block (in UNIX epoch time).
- Used to keep blocks in chronological order

- Difficulty Target (nBits)
- Encodes the current mining difficulty.
- Defines how hard it is to find a valid block hash (hash must be lower than this target).

- Nonce
- A 32-bit number miners adjust when hashing the block header.
- Miners keep changing the nonce to try and find a block hash that meets the difficulty target.

Bitcoin: Visualization of Blockchain



- On the left-hand side blocks are shown starting from top to bottom.
- Each block contains transactions and block headers which are further magnified on the right-hand side.
- On the top, first, block header is expanded to show various elements within the block header.
- Then on the right-hand side the Merkle root element of the block header is shown in magnified view which shows that how Merkle root is calculated.
  - Merkle root is a hash of all of the nodes of a Merkle tree. Merkle trees are widely used to validate the large data structures securely and efficiently. Merkle root in a blockchain is present in the block header section of a block, which is the hash of all transactions in a block.
  - verifying only the Merkle root is required to verify all transactions present in the Merkle tree instead of verifying all transactions one by one.
- Further down transactions are also magnified to show the structure of a transaction and the elements that it contains. Also, note that transactions are then further elaborated by showing that what locking and unlocking scripts look like.

Genesis Block

- Bitcoin provides protection against double spending by enforcing strict rules on transaction verification and via mining. Transactions and blocks are added in the blockchain only after strict rule checking explained earlier in the Transaction verification.
- Block height is the number of blocks before a particular block in the blockchain.
- Each block contains one or more transactions, out of which the first transaction is a coinbase transaction.
- Stale blocks are created . A stale block (sometimes called an “orphan block”) happens when two miners solve a valid block at nearly the same time.
  - Both blocks are valid, but only one can stay in the main chain.
  - The other block becomes stale because the network eventually builds on one version of the chain and abandons the other.
- This is an undesirable situation but can be addressed by the Bitcoin network only by accepting the longest chain. In this case, the smaller chain will be considered orphaned.
- Forks(split in the chain) in blockchain can also occur with the introduction of changes in the Bitcoin protocol. In case of a soft fork, a client which chooses not to upgrade to the latest version supporting the updated protocol will still be able to work and operate normally.

Mining

- Mining is a process by which new blocks are added to the blockchain.
- Blocks contain transactions that are validated via the mining process by mining nodes on the Bitcoin network.
- Blocks, once mined and verified are added to the blockchain which keeps the blockchain growing.
- This process is resource-intensive due to the requirements of PoW where miners compete in order to find a number which is less than the difficulty target of the network.
- This difficulty in finding the correct value (also called sometimes the mathematical puzzle) is there to ensure that the required resources have been spent by miners before a new proposed block can be accepted.

Tasks of miners

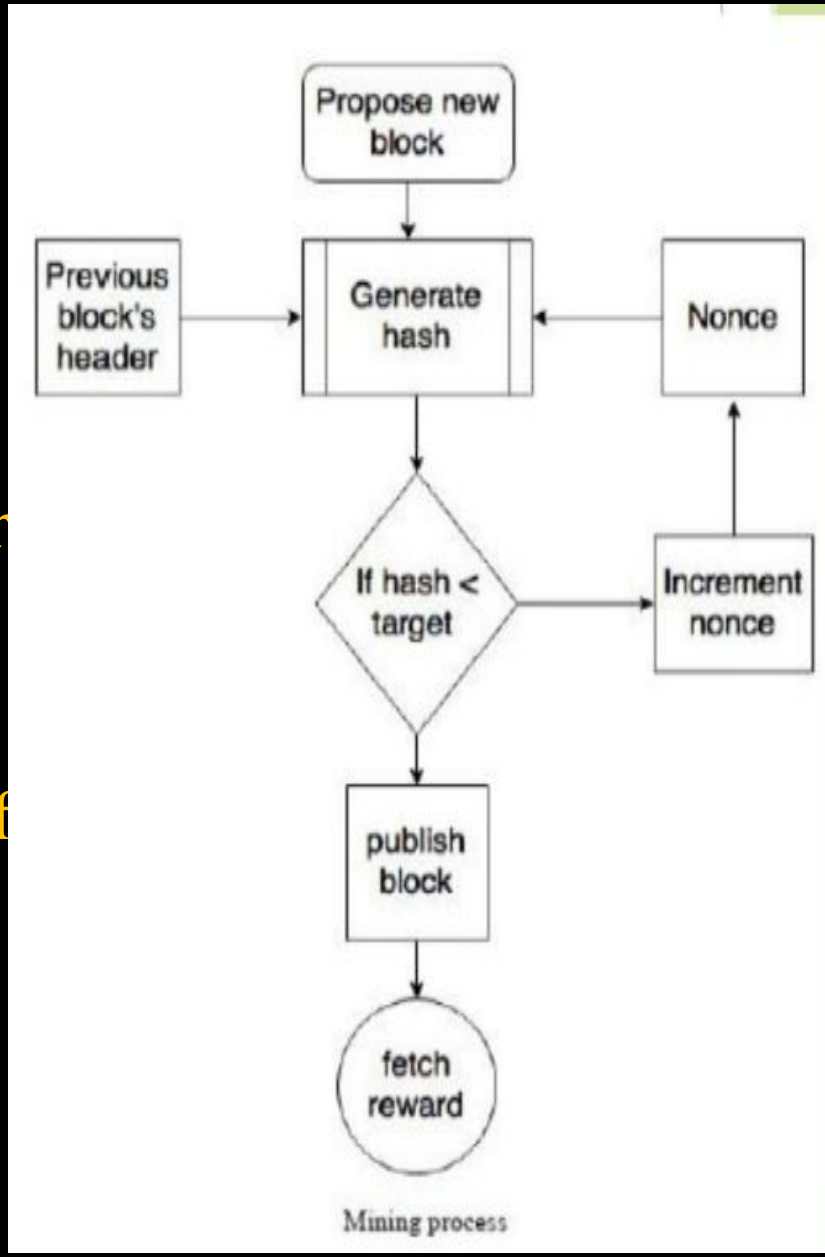
- Synching up with the network: Once a new node joins the bitcoin network, it downloads the blockchain by requesting historical blocks from other nodes. This is mentioned here in the context of the bitcoin miner;
- Transaction validation: Transactions broadcasted on the network are validated by full nodes by verifying and validating signatures and outputs.
- Block validation: Miners and full nodes can start validating blocks received by them by evaluating the against certain rules. This includes the verification of each transaction in the block along with verification of the nonce value.
- Create a new block: Miners propose a new block by combining transactions broadcasted on the network after validating them.
- Perform Proof of Work: This task is the core of the mining process and this is where miners find a valid block by solving a computational puzzle. The block header contains a 32-bit nonce field and miners are required repeatedly vary the nonce until the resultant hash is less than a predetermined target.
- Fetch reward: Once a node solves the hash puzzle (PoW), it immediately broadcasts the results, and other nodes verify it and accept the block. There is a slight chance that the newly minted block will not be accepted by other miners on the network due to a clash with another block found at roughly the same time, but once accepted, the miner is rewarded with 12.5 bitcoins and any associated transaction fees.

Mining rewards, Proof of work

- When Bitcoin started in 2009 the mining reward used to be 50 bitcoins. After every 210,000 blocks, the block reward halves.
- In November 2012 it halved down to 25 bitcoins.
- It was reduced to 12.5 BTC per block since July 2016.
- Currently it is 6.25 BTC.
- This is a proof that enough computational resources have been spent in order to build a valid block.
- PoW is based on the idea that a random node is selected every time to create a new block. nodes compete with each other in order to be selected in proportion to their computing capacity.
- The following equation sums up the PoW requirement in bitcoin:
- H(N|P_hash || Tx || Tx ||... Tx)
- Where N is a nonce, P_hash is a hash of the previous block, Tx represents transactions in the block, and Target is the target network difficulty value. This means that the hash of the previously mentioned concatenated fields should be less than the target hash value.

Mining algorithm

- The previous block's header is retrieved from the bitcoin network.
- Assemble a set of transactions broadcasted on the network into a block to be proposed
- Compute the double hash of the previous block's header combined with a nonce and the newly proposed block using the SHA-256 algorithm.
- Check if the resultant hash is lower than the current difficulty level (target) then PoW is solved. As a result of successful PoW the discovered block is broadcasted to the network and miners fetch the reward.
- If the resultant hash is not less than the current difficulty level (target), then repeat the process after incrementing the nonce.
- H(N|| P_hash || Tx || Tx ||... Tx) < Target



Mining process

Hash rate, Mining systems

- The hashing rate basically represents the rate of calculating hashes per second. In other words, this is the speed at which miners in the Bitcoin network are calculating hashes to find a block.
- In recent days, the Bitcoin network miners are computing more than 24,000,000,000,000,000,000 hashes per second.
- Mining Systems
- CPU:
  - CPU mining was the first type of mining available in the original bitcoin client.
  - Users could even use laptop or desktop computers to mine bitcoins.
  - CPU mining is no longer profitable and now more advanced mining methods such as ASIC-based mining is used.
  - CPU mining only lasted for around just over a year since the introduction of Bitcoin and soon other methods were explored and tried by the miners.

- GPU:
  - Due to the increased difficulty of the bitcoin network and the general tendency of finding faster methods to mine, miners started to use GPUs or graphics cards available in PCs to perform mining.
  - GPUs support faster and parallelized calculations that are usually programmed using the OpenCL language. This turned out to be a faster option as compared to CPUs. Users also used techniques such as overclocking to gain maximum benefit of the GPU power.
  - Also, the possibility of using multiple graphics cards increased the popularity of graphics cards' usage for bitcoin mining.
  - GPU mining, however, has some limitations, such as overheating and the requirement for specialized motherboards and extra hardware to house multiple graphics cards.
- FPGA:
  - Field Programmable Gate Array (FPGA) is basically an integrated circuit that can be programmed to perform specific operations.
  - FPGAs are usually programmed in Hardware Description Languages (HDLs), such as Verilog and VHDL.
  - Double SHA-256 quickly became an attractive programming task for FPGA programmers and several open source projects started too.
  - FPGA offered much better performance as compared to GPUs; however, issues such as accessibility, programming difficulty, and the requirement for specialized knowledge to program and configure FPGAs resulted in a short life of the FPGA era for bitcoin mining.

- Application Specific Integrated Circuit (ASIC):
  - It was designed to perform the SHA-256 operation. These special chips were sold by various manufacturers and offered a very high hashing rate.
  - This worked for some time, but due to the quickly increasing mining difficulty level, single-unit ASICs are no longer profitable.
- Mining Pools:
  - A mining pool forms when group of miners work together to mine a block.
  - The pool manager receives the coinbase transaction if the block is successfully mined, which is then responsible for distributing the reward to the group of miners who invested resources to mine the block.
  - There are various models that a mining pool manager can use to pay to the miners, such as the Pay Per Share (PPS) model and the proportional model.
  - In the PPS model, the mining pool manager pays a flat fee to all miners who participated in the mining exercise, whereas in the proportional model, the share is calculated based on the amount of computing resources spent to solve the hash puzzle.

Bitcoin network and payments

- The Bitcoin network is a peer-to-peer network where nodes exchange transactions and blocks.
- There are two main types of nodes, full nodes and SPV nodes.
  - Full nodes, as the name implies, are implementations of Bitcoin core clients performing the wallet, miner, full blockchain storage, and network routing functions. However, it is not necessary to perform all these functions.
  - Simple Payment Verification (SPV) nodes or lightweight clients perform only wallet and network routing functionality.
- There are nodes such as pool protocol servers: Nodes that only compute hashes use the stratum protocol to submit their solutions to the mining pool.
- Some nodes perform only mining functions and are called mining nodes.

- The most commonly used protocol messages and their explanation are listed as follows
  - version: This is the first message that a node sends out to the network, advertising its version and block count. The remote node then replies with the same information and the connection is then established.
  - verack: This is the response of the version message accepting the connection request.
  - inv: This is used by nodes to advertise their knowledge of blocks and transactions.
  - getdata: This is a response to inv, requesting a single block or transaction identified by its hash.
  - getblocks: This returns an inv packet containing the list of all blocks starting after the last known hash or 500 blocks.
  - getheaders: This is used to request block headers in a specified range.
  - tx: This is used to send a transaction as a response to the getdata protocol message.