Ethereum

Main difference with Bitcoin

# Ethereum v 1.0

**Main differences:**

* Programs can be stored on the blockchain
* Apparition of **smart contract**: piece of code that run on the blockchain and are guaranteed to produce the same result for everyone run them
* Creation of dApps (DeFi, Gaming) without any downtime, fraud, control or interference from a third party
* Non-Fungible Tokens (**NFTs**)
* 12/14 seconds to mine a block
* Current mining reward: 2ETH + gas fees + **uncle block**

**Uncle block:** is similar to orphan block during bitcoin blockchain forks. It is valid block that isn’t included in the blockchain due to 2 miners finding the solution almost instantaneously. In the Ethereum they are 2 types of rewards for miners:

* Uncles Reward is awarded to the miner that **creates an uncle**. The value of reward depends on how old the uncle block before it is included in the blockchain: max 1.75 ETH. The older the block is the lower the reward.
* Uncle Inclusion Reward is awarded to the miner that **includes the uncle block in the blockchain**: 0.0625 ETH.

## Ether

* Ether (ETH) is the cryptocurrency used for many things on the Ethereum network.
* Ethereum allows developers to **create decentralized applications** (dapps), which all **share a pool of computing power**.
* When users want to make a transaction, they must pay ether to have their transaction recognized on the blockchain.
* Ether burn occurs in every transaction on Ethereum depending on network demand.
* **Wei** is the smallest possible amount of ether.
* **Gwei** is the human readable form.
* When the recipient address is a smart contract, this transferred ether may be used to pay for gas when the smart contract executes its code.

## Dapps

* Application built on a decentralized network that combines a smart contract and a frontend user interface.
* Has its backend code running on a decentralized peer-to-peer network, the EVM.
* A dapp can have frontend code and user interfaces written in any language to make calls to its backend. Furthermore, its frontend can get hosted on decentralized storage such as IPFS.
* Zero downtime: the network as a whole will always be able to serve clients looking to interact with the contract.
* Dapps can be harder to maintain because the code and data published to the blockchain are harder to modify.
* Performance issues.
* Network congestion: one dapp may use too many computational resources. Currently, the network can only process about 10-15 transactions per second. If transactions are being sent in faster than this, the pool of unconfirmed transactions can quickly balloon.

## Account

* Where ether is stored.
* Accounts and account balances are stored in a big table in the EVM.

## Transaction

* The formal term for a request for code execution on the EVM.
* **2 types:** message calls and contract creation.
* **Contract creation:** results in the creation of a new contract account **containing compiled smart contract** bytecode.
* **Message calls:** **execute** contract’s bytecode.
* E. g.:
  + Send X ether from my account to Alice's account.
  + Publish some smart contract code into EVM memory.
  + Execute the code of the smart contract at address X in the EVM, with arguments Y.

## Blocks

* The volume of transactions is very high, so transactions are "committed" in batches, or blocks.
* Blocks generally contain **dozens to hundreds** of transactions.

## EVM

* Stands for Ethereum Virtual Machine

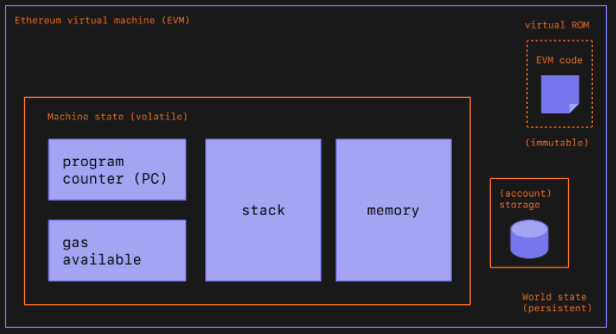


Figure 1: Diagram adapted from EVM, [source](https://ethereum.org/en/developers/docs/evm/)

* **Instead of a distributed ledger**, Ethereum is **a distributed state machine**.
* EVM behaves as a mathematical function would: **Given an input**, it produces a **deterministic output**.
* Y (S, T) = S'.
  + S: old valid state
  + T: new set of valid transactions
  + Y: the Ethereum state transition function
  + S’: new valid state
* EVM is the global virtual computer whose state every participant on the Ethereum network stores and agrees on.
* Any participant can request the execution of arbitrary code on the EVM.
* Code execution changes the state of the EVM.
* **State:** enormous data structure called a modified **Merkle Patricia Tree**. **Keeps all accounts** linked by hashes and reducible to a single root hash stored on the blockchain.
* The EVM executes as a stack machine.
* Compiled smart contract bytecode executes as a number of EVM opcodes, which perform standard stack operations like XOR, AND, ADD, SUB, etc.
* The EVM also implements a number of blockchain-specific stack operations, such as ADDRESS, BALANCE, BLOCKHASH, etc.

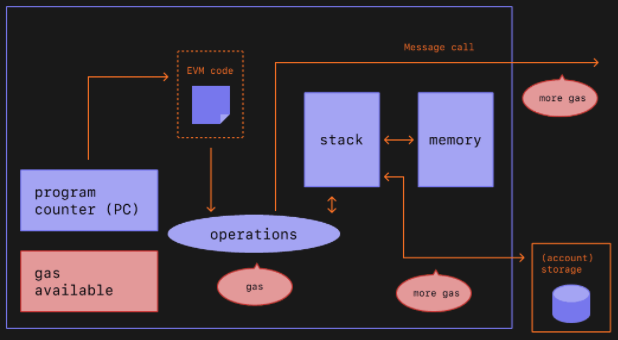


Figure 2: Diagrams adapted from Ethereum EVM, [source](https://ethereum.org/en/developers/docs/evm/).

## Smart contracts

* A **reusable** snippet of code (a **program**) which a developer publishes into EVM memory.
* Anyone can request that the smart contract code be **executed** by making a **transaction request**.
* Because developers can write **arbitrary executable applications** into the EVM (games, marketplaces, financial instruments, etc.) by publishing smart contracts, these are often also called **dapps**, or **Decentralized Apps**.
* Once smart contracts are deployed on the network you can't change them.

# Ethereum v 2.0

## Beacon chain - alive

* Upgrade from PoW to PoS
* Shipped separately from the main net

## The merge - 2022

* Main net under PoS
* Bring the ability to run smart contract on main net with PoS
* History and current state of Ethereum

## Shard chains - 2023

* Improve Ethereum’s scalability and capacity
* Spread the network’s load across 64 new chains
* Run a node by keeping requirement low
* Increase the TPS with rollups transactions