

L13: Ethereum Architecture

Module B: Ethereum & Smart Contracts

Blockchain & Cryptocurrency Course

December 2025

What Is the World Computer?

[COMIC: Developer announces “I’m building on the world computer!” while sitting at an actual laptop with 8GB RAM and a spinning fan. The Ethereum logo floats above looking majestic while the laptop displays “Low Memory Warning.”]

Frame the Concept:

- The “world computer” executes code across thousands of nodes simultaneously
- Yet each node is just an ordinary machine running the same deterministic bytecode

Ethereum: Where your code runs everywhere, but somehow still costs more than AWS

By the end of this lesson, you will be able to:

- Explain the Ethereum Virtual Machine (EVM) architecture
- Distinguish between Externally Owned Accounts (EOA) and Contract Accounts
- Describe Ethereum as a state machine and explain the world state concept
- Explain the Merkle Patricia Trie data structure
- Compare Ethereum's account model to Bitcoin's UTXO model

Building on L12: Lab: Block Explorer

The Problem: How do we make money programmable?

The Challenge

How do we create a blockchain system that can execute arbitrary code with value at stake, where code execution is deterministic and verifiable across all nodes?

Why It Matters

- Bitcoin's scripting language only allows simple operations (send, verify signatures)
- Complex financial logic (lending, options, DAOs) requires Turing-complete computation
- Vitalik's 2013 whitepaper proposed "world computer"; Ethereum launched 2015; The DAO hack 2016 revealed execution risks

What We Need

- System design principles
- A virtual machine where all nodes execute code identically and reach consensus on state changes

The Cryptoeconomics Question

Creating systems with desired properties

Today's lesson: How Ethereum Architecture addresses this challenge

Continued

How Is Ethereum Different from Bitcoin?

Key Characteristics:

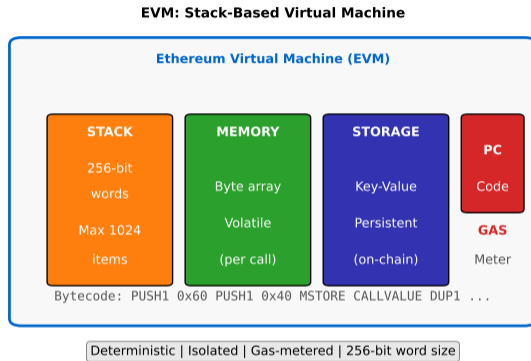
- Decentralized computation platform
- Turing-complete blockchain
- Smart contract support
- Native cryptocurrency: Ether (ETH)
- Launched July 30, 2015

Beyond Bitcoin:

- Bitcoin: Digital gold, value transfer
- Ethereum: World computer, programmable logic
- Enables decentralized applications (dApps)
- Foundation for DeFi, NFTs, DAOs

→ Problem: How do we make money programmable? — What is Ethereum? Ethereum provides a global computer where code handles value, not just records it

How Does the EVM Execute Smart Contracts?



The EVM is a quasi-Turing complete (can compute anything, but gas limits prevent infinite loops), stack-based (operations push/pop values from a last-in-first-out stack) virtual machine with gas metering.

What Are the Technical Details of the EVM?

The EVM is a quasi-Turing complete state machine:

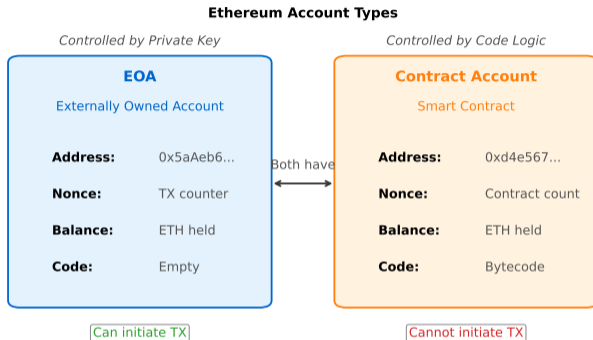
- Stack-based virtual machine (256-bit word size)
- Executes bytecode compiled from Solidity, Vyper
- Deterministic: same input always produces same output
- Gas metering prevents infinite loops
- Isolated: no network, filesystem, or process access

EVM Components:

- **Stack:** LIFO, max 1024 items, 256-bit each
- **Memory:** Byte array, volatile (per call), linear cost
- **Storage:** Key-value, persistent (on-chain), expensive
- **Program Counter:** Current bytecode position
- **Gas Counter:** Remaining computation budget

Key point: The EVM is a quasi-Turing complete state machine

What Are the Two Types of Ethereum Accounts?



*EOAs are controlled by private keys; contracts are controlled by code. | → Problem: How do we make money programmable? — Account Types: EOA vs Contract
Contracts ARE the programmable money – code that controls value automatically*

What Components Make Up Account State?

Every account (EOA or Contract) has four fields:

① **Nonce:**

- EOA: Counter of transactions sent
- Contract: Counter of contracts created
- Prevents replay attacks

② **Balance:**

- Amount of Wei (10^{-18} ETH) owned
- 1 ETH = 1,000,000,000,000,000 Wei

③ **StorageRoot:**

- Hash of account's storage trie root
- Empty for EOAs

④ **CodeHash:**

- Hash of EVM bytecode
- Empty for EOAs, immutable for contracts

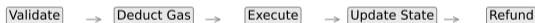
Key point: Every account (EOA or Contract) has four fields

How Does Ethereum Track State Changes?

Ethereum: Deterministic State Machine

State Transition Function

$$S_{t+1} = Y(S_t, T)$$



Transaction Execution Flow

State transitions are deterministic: given S_t and T , S_{t+1} is uniquely determined.

What Is the World State?

The World State is a mapping between addresses and account states:

- Maps 160-bit addresses to account states
- Stored as a Merkle Patricia Trie (from L02b: tree combining hash verification with key-value storage)
- Root hash included in every block header
- Allows efficient verification of account state
- Enables light clients to verify data without full state

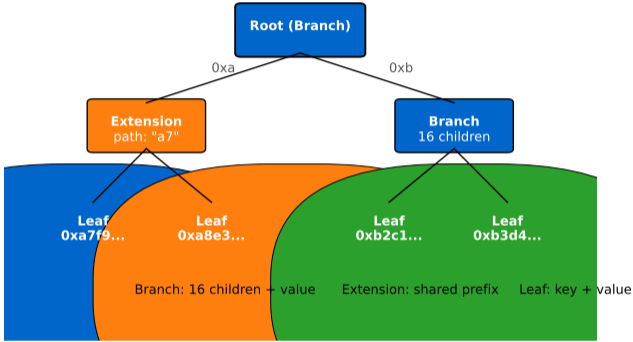
State Root:

- 256-bit hash representing entire world state
- Changes with every block
- Uniquely identifies state at a specific block height

Key point: The World State is a mapping between addresses and account states

How Does Ethereum Efficiently Store State?

Merkle Patricia Trie: Node Types



MPT combines Merkle verification, Patricia path compression, and radix optimization. | → Problem: How do we make money programmable? — Merkle Patricia Trie (MPT) MPT enables efficient verification of any account state – essential for light clients

Recall Our Problem

How do we make money programmable?

What We've Learned So Far

- The EVM executes bytecode deterministically across all nodes
- Two account types: EOAs (humans with keys) control contracts (autonomous code)
- Together they enable programmable money: code that controls value without intermediaries

Still to Address

- How do we verify state efficiently? (Merkle Patricia Trie) What trade-offs vs Bitcoin?
- How would YOU design a system to make money programmable?

Think About

- Based on what you've seen, how would *you* solve this problem?
- What trade-offs do you expect?

Pause and reflect: How does what we've learned so far address "How do we make money programmable?"?

What Properties Does the MPT Have?

Combines three data structures:

- 1 **Merkle Tree:** Cryptographic verification via hashes
- 2 **Patricia Trie:** Efficient key-value storage with shared prefixes
- 3 **Radix Trie:** Optimized path compression

Key Properties:

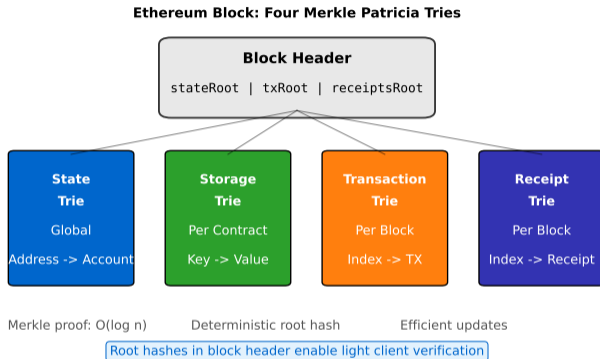
- Deterministic: Same key-value pairs → same root hash
- Efficient verification: $O(\log n)$ proof size
- Efficient updates: Only modified paths need rehashing

Node Types:

- **Branch:** 16 children (hex 0-F) + optional value
- **Extension:** Shared path prefix + next node pointer
- **Leaf:** Remaining key path + value

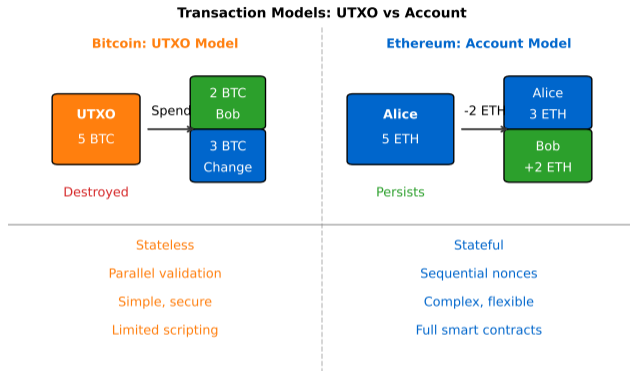
Key point: Combines three data structures

What Are the Four MPT Tries in Ethereum?



Block header contains root hashes of all four tries for verification.

Why Did Ethereum Choose Accounts Instead of UTXOs?



Account model enables smart contracts but requires nonce tracking.

What Are the Trade-offs of the Account Model?

Advantages:

- 1 **Simplicity:** Intuitive balance model, easier wallets
- 2 **Space Efficiency:** No UTXO tracking overhead
- 3 **Fungibility:** All Ether equivalent, no dust
- 4 **Smart Contracts:** Natural fit for persistent storage

Challenges:

- 1 **Replay Prevention:** Requires nonce tracking
- 2 **State Growth:** All accounts stored indefinitely
- 3 **Privacy:** All activity linked to single address
- 4 **Parallelization:** Sequential nonces limit parallel validation

→ Problem: How do we make money programmable? — Account Model: Trade-offs Account model enables smart contracts but sacrifices privacy and parallelization

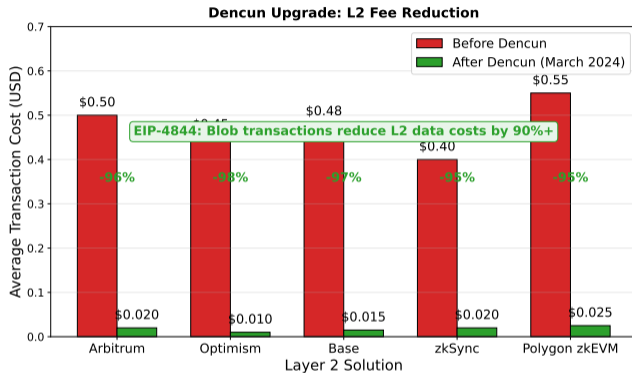
What Is Ethereum's Block Structure?

Ethereum block header contains:

- **parentHash:** Hash of parent block
- **beneficiary:** Address receiving block reward
- **stateRoot:** Root hash of state trie
- **transactionsRoot:** Root hash of transaction trie
- **receiptsRoot:** Root hash of receipt trie
- **logsBloom:** Bloom filter for efficient log lookup
- **number:** Block height
- **gasLimit / gasUsed:** Gas constraints
- **timestamp:** Unix timestamp
- **extraData:** Arbitrary data (max 32 bytes)

Key point: Ethereum block header contains

How Did the Dencun Upgrade Reduce L2 Costs?



EIP-4844 blob transactions reduce L2 data costs by 90%+.

What Is Proto-Danksharding in the Dencun Upgrade?

Major Network Upgrade (March 13, 2024):

- **EIP-4844:** Introduces “blob” transactions for Layer 2 data
- **Problem:** L2 rollups pay expensive calldata for data availability
- **Solution:** New data type (blobs) with separate, cheaper gas market

How Blobs Work:

- Blobs: 128 KB data chunks, stored for ~18 days
- Separate “blob gas” market (independent of execution gas)
- L2s post transaction batches as blobs instead of calldata

Impact:

- L2 transaction fees reduced by 90%+ (\$0.50 → \$0.01)
- Arbitrum, Optimism, Base adopted immediately
- Paves way for full Danksharding (future upgrade)

Key point: Major Network Upgrade (March 13, 2024)

The Original Problem

How do we make money programmable?

How Ethereum Architecture Solves It

- **EVM executes bytecode:** All nodes run identical stack-based VM, ensuring deterministic state transitions
- Persistent storage (vs UTXO) enables complex contract logic and state management
- Cryptographic verification of all account states at every block

Remaining Limitations

- Gas fees make computation expensive; state bloat from permanent storage
- Sequential transaction processing (nonce ordering) limits parallelization and throughput

Open Questions

- How to scale smart contract execution (10,000+ TPS) while maintaining decentralization and security?
- Risk: State growth, MEV exploitation, smart contract bugs with irreversible financial consequences

Ethereum Architecture solves programmable money but introduces scalability and cost trade-offs

Incentive Structure

- Creating systems with desired properties
- Mechanism design for target behaviors
- Complexity vs simplicity

Economic Security

- Attack cost must exceed potential gain
- Honest behavior = Nash equilibrium

Cryptoeconomic security: Honest behavior must be the Nash equilibrium

Key Economic Question

Who Pays, Who Earns?

Complexity vs simplicity

Design Principle

Attack Cost $>$ Potential Gain

Alternatives Considered

- 1 Architecture patterns, upgrade mechanisms
- 2 Alternative design patterns

Trade-offs Made

- Every design optimizes some properties
- ... at the expense of others

Design Questions

- What would YOU change?
- What's optimized? What's sacrificed?
- Are there other approaches?

Key Insight

No Perfect Solution

All blockchain designs involve trade-offs between decentralization, security, and scalability.

Every design is a trade-off. Understanding alternatives reveals the "why" behind choices.

Failure Modes

Critical Failure Mode

- **What breaks:** Unintended consequences, edge cases
- **Why it happens:** Economic incentives misaligned

Root Cause

- Assumption violated
- Incentive structure broken
- External shock

Historical Context

- Multiple real-world failures documented
- Patterns repeating across protocols

Early Warning Signs

- ! Unusual economic behavior
- ! Incentive misalignment
- ! Centralization drift

Prediction: What could cause this to fail? How would you detect it early?

Why Is the EVM Like a Universal Calculator?

[COMIC: The EVM depicted as a giant vintage calculator with a tiny screen showing “256-bit math.” Thousands of identical calculators around the world all display the same result. One calculator in the corner shows a different number and is being ejected from the network with a “CONSENSUS FAILED” stamp.]

Key Insight:

- The EVM's power lies in determinism: every node computes the same result, every time
- Disagreement means you are wrong—consensus is ruthless but reliable

10,000 calculators, one answer: that is the magic of deterministic computation

- 1 **EVM:** Deterministic, stack-based VM enabling Turing-complete smart contracts
- 2 **Two Account Types:** EOAs (user-controlled) and Contracts (code-controlled)
- 3 **State Machine:** Ethereum transitions between states via transactions
- 4 **World State:** Mapping of addresses to accounts, stored as MPT
- 5 **MPT:** Efficient cryptographic data structure for state verification
- 6 **Account Model:** Simpler than UTXO but with privacy/parallelization trade-offs

Next Lesson: L14 – Gas Mechanics

Key point: EVM

- ① Why is the EVM only “quasi”-Turing complete?
- ② What security benefits does the nonce provide?
- ③ How does MPT enable light client verification?
- ④ When might Bitcoin's UTXO model be preferable?
- ⑤ How does state growth affect node operators?

Key point: Discussion Questions

Continued

Coming up next:

- Understanding gas as a computational unit
- Gas price, gas limit, and transaction cost calculation
- EIP-1559: Base fee and priority fee mechanism
- Gas costs for different EVM operations
- Optimization techniques for reducing gas consumption

Preparation:

- Review basic Ethereum transaction structure
- Familiarize yourself with Wei/Gwei/ETH units
- Browse Etherscan to see gas usage in real transactions

Key point: Coming up next

Quiz Questions (1/4)

Q1. What is the EVM's word size?

- A) 128 bits B) 160 bits C) 256 bits D) 512 bits

Answer: C – The EVM uses 256-bit words for stack operations.

Q2. Which component makes the EVM “quasi-Turing complete” rather than fully Turing complete?

- A) Stack limit B) Gas metering C) Memory size D) Bytecode length

Answer: B – Gas metering prevents infinite loops, limiting true Turing completeness.

Q3. What is the maximum stack depth in the EVM?

- A) 256 items B) 512 items C) 1024 items D) 2048 items

Answer: C – The EVM stack can hold a maximum of 1024 items.

Q4. Which account type is controlled by private keys?

- A) Contract accounts B) Both types C) Neither type D) Externally Owned Accounts (EOA)

Answer: D – EOAs are controlled by private keys; contracts are controlled by code.

Q5. How many Wei are in 1 ETH?

- A) 10^9 B) 10^{12} C) 10^{18} D) 10^{21}

Answer: C – 1 ETH = 1,000,000,000,000,000,000 Wei (10^{18}).

Q6. What does the nonce field track for an EOA?

- A) Balance B) Transaction count sent C) Gas used D) Contract deployments

Answer: B – For EOAs, nonce counts transactions sent to prevent replay attacks.

Q7. Which field is empty for Externally Owned Accounts but populated for Contract Accounts?

- A) Nonce B) Balance C) CodeHash D) All fields are populated

Answer: C – EOAs have no code, so CodeHash is empty; contracts store bytecode.

Q8. What data structure is used to store the Ethereum world state?

- A) Hash table B) Binary tree C) Merkle Patricia Trie D) Bloom filter

Answer: C – The world state uses a Merkle Patricia Trie for efficient verification.

Q9. How many tries does an Ethereum block header reference?

- A) One B) Two C) Three D) Four

Answer: D – State, transaction, receipt, and storage tries (four total).

Q10. What is the state root?

- A) First block hash B) 256-bit hash of entire world state C) Genesis state D) Merkle tree root

Answer: B – The state root is a 256-bit hash representing the entire world state.

Q11. Which node type in MPT has 16 children?

- A) Leaf B) Extension C) Branch D) Root

Answer: C – Branch nodes have 16 children (hex 0-F) plus optional value.

Q12. What is the primary advantage of Ethereum's account model over Bitcoin's UTXO model?

- A) Better privacy B) Easier parallelization C) Simpler for smart contracts D) Smaller state size

Answer: C – Account model is more natural for persistent contract storage.

Q13. What is Ethereum often called to distinguish it from Bitcoin?

- A) Digital silver B) World computer C) Smart ledger D) Crypto platform

Answer: B – Ethereum is called a “world computer” for programmable logic.

Q14. When was Ethereum mainnet launched?

- A) January 3, 2009 B) July 30, 2015 C) August 1, 2017 D) December 1, 2020

Answer: B – Ethereum launched July 30, 2015 (Bitcoin was 2009).

Q15. Which EVM memory type is persistent and stored on-chain?

- A) Stack B) Memory C) Storage D) Cache

Answer: C – Storage is persistent on-chain (expensive); Memory is volatile per call.

Quiz

Q16. What does EIP-4844 (Dencun upgrade) introduce?

- A) Proof of Stake B) Blob transactions C) Smart contracts D) Sharding

Answer: B – EIP-4844 introduces blob transactions for cheaper L2 data availability.

Q17. By approximately how much did Dencun reduce Layer 2 transaction fees?

- A) 30% B) 50% C) 70% D) 90%+

Answer: D – L2 fees dropped 90%+ (e.g., \$0.50 to \$0.01) after Dencun.

Q18. What is the address size in Ethereum?

- A) 128 bits B) 160 bits C) 256 bits D) 512 bits

Answer: B – Ethereum addresses are 160 bits (40 hex characters).

Q19. Which field in the account state prevents replay attacks?

- A) Balance B) Nonce C) CodeHash D) StorageRoot

Answer: B – The nonce ensures each transaction is unique and sequential.

Q20. What challenge does Ethereum's account model face compared to UTXO?

- A) Complexity B) Privacy limitations C) Higher storage D) No smart contracts

Answer: B – All activity is linked to a single address, reducing privacy vs UTXO.