



School: Campus:

Academic Year: Subject Name: Subject Code:

Semester: Program: Branch: Specialization:

Date:

Applied and Action Learning

(Learning by Doing and Discovery)

Name of the Experiment : Dive into Ethereum – Clients and EVM

Objective/Aim:

To study and understand **Ethereum Clients** and the **Ethereum Virtual Machine (EVM)** – their roles, architecture, and how they execute smart contracts in the Ethereum blockchain ecosystem.

Apparatus/Software Used:

- **Ethereum Test Network (Sepolia)**
- **Remix IDE** (for writing and deploying smart contracts)
- **Metamask** (for wallet and transaction management)
- **Etherscan / Block Explorer** (to view contract deployment and execution details)

Theory:

Ethereum is a decentralized, open-source blockchain platform that supports **smart contracts**—self-executing code that runs on the blockchain without intermediaries.

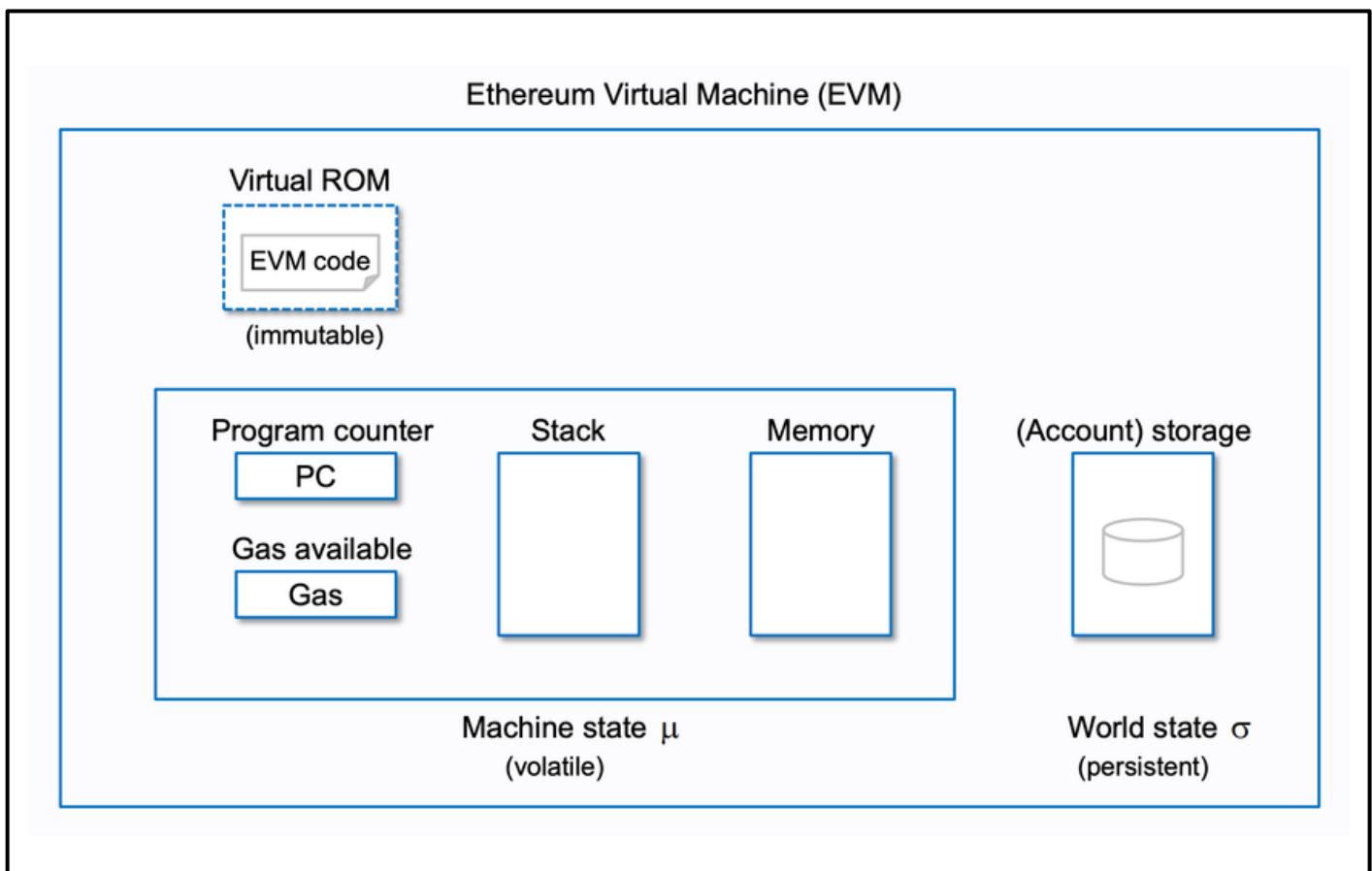
Ethereum Virtual Machine (EVM)

The **EVM** is the execution environment for Ethereum smart contracts.

It is a **stack-based virtual machine** that executes bytecode (compiled from Solidity or Vyper code). Each node runs the EVM locally to verify transactions and update the blockchain state.

EVM Components:

1. **Stack:** Holds temporary values for computation.
2. **Memory:** Temporary, per-transaction storage.
3. **Storage:** Persistent storage linked to a contract address.
4. **Program Counter:** Tracks instruction execution.
5. **Gas:** Measures computation cost and prevents infinite loops.



Procedure:

1. Open remix IDE and create a file eg. EVM_Test.sol and compile it.

```
✓ Compiled ▾ | 🔎 🔍 ⚡ Home | EVM_Test.sol X
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.0;
3
4 contract StorageDemo {
5     uint public number;
6
7     function setNumber(uint _num) public { 22492 gas
8         number = _num;
9     }
10
11    function getNumber() public view returns (uint) { 2453 gas
12        return number;
13    }
14 }
```

2. Connect Metamask to Remix (choose Sepolia testnet).

3. Deploy the contract using a test account.

The screenshot shows the 'DEPLOY & RUN TRANSACTIONS' interface in Remix. On the left, under 'ENVIRONMENT', 'Injected Provider - MetaMask' is selected. Under 'ACCOUNT', 'Sepolia (1155111) network' is chosen, and the account '0x42E...1b8D1 (0.50027031239)' is highlighted. Below it, there's a 'Create Smart Account' button. Under 'GAS LIMIT', 'Estimated Gas' is selected. Under 'VALUE', '0 Wei' is entered. In the 'CONTRACT' section, 'StorageDemo - contracts/EVM_Test.s' is selected, and 'evm version: prague' is checked. There's also a 'Verify Contract on Explorers' checkbox and a 'Deploy & Verify' button. On the right, a modal window titled 'Deploy a contract' asks 'This site wants you to deploy a contract'. It shows 'Estimated changes' as 'No changes', 'Network' as 'Sepolia', 'Request from' as 'remix.ethereum.org', 'Network fee' as '0.0001 SepoliaETH', and 'Speed' as 'Market ~12 sec'. At the bottom of the modal are 'Cancel' and 'Confirm' buttons.

The screenshot shows the Remix Debugger interface. It displays a transaction log entry: '[block:9552567 txIndex:10] from: 0x42e...1b8d1 to: StorageDemo.(constructor) value: 0 wei data: 0x608...e0033 logs: 0 hash: 0x1eb...810a6'. Below this, it says 'Verification process started...' and 'Verifying with Sourcify...'. There are 'view on Etherscan' and 'view on Blockscout' links at the top. A 'Debug' button is visible on the right.

4. After deployment, view the transaction hash on Etherscan.

The screenshot shows the Etherscan transaction details page for the deployed contract. Key information includes:

- TRANSACTION ACTION:** Call `0x60806040` Method by `0x42Ec11Bc...9cC91b8D1`
- [This is a Sepolia Testnet transaction only]**
- Transaction Hash:** `0x06c00848236722a355e5e9e6ad74a5168289586c367643f2a3377fbfce1616b1`
- Status:** Success
- Block:** `9552567` 3 Block Confirmations
- Timestamp:** 43 secs ago (Nov-03-2025 03:46:00 PM UTC)
- From:** `0x42Ec11BcdF103cCcAa71be8E655488d9cC91b8D1`
- To:** `0x77c13e65f3c40fa7ea2e594e07c89c7d007ae0da` Created
- Value:** 0 ETH
- Transaction Fee:** 0.000199426501595412 ETH
- Gas Price:** 1.500000012 Gwei (0.000000001500000012 ETH)

5. Observe:

- Gas used
- Bytecode generated
- Execution steps in Remix Debugger

Call the `setNumber()` and `getNumber()` functions to observe state changes.

Review the **EVM bytecode** generated in the compilation details.

Observation:

Applied and Action Learning

Through this experiment, we understood:

- The **role of Ethereum clients** in maintaining the decentralized network.
- How the **EVM** ensures deterministic and secure execution of smart contracts.
- That **smart contracts** are converted into low-level bytecode and executed identically across all nodes.

ASSESSMENT

Rubrics	Full Mark	Marks Obtained	Remarks
Concept	10		
Planning and Execution/ Practical Simulation/ Programming	10		
Result and Interpretation	10		
Record of Applied and Action Learning	10		
Viva	10		
Total	50		

Signature of the Student:

Name :

Signature of the Faculty:

Regn. No. :

Page No.....

*As applicable according to the experiment.
Two sheets per experiment (10-20) to be used.