Contents

[EVM (Ethereum Virtual Machine) 2](#_Toc98441357)

[1. Introduction of EVM 2](#_Toc98441358)

[2. So how do you make a program that execute on all of the computers? 2](#_Toc98441359)

[3. Why do we have EVM in the middle on the process? 3](#_Toc98441360)

[4. How does EVM execute instructions? 3](#_Toc98441361)

# EVM (Ethereum Virtual Machine)

## Introduction of EVM

Just like Bitcoin, ETH is a network of computers

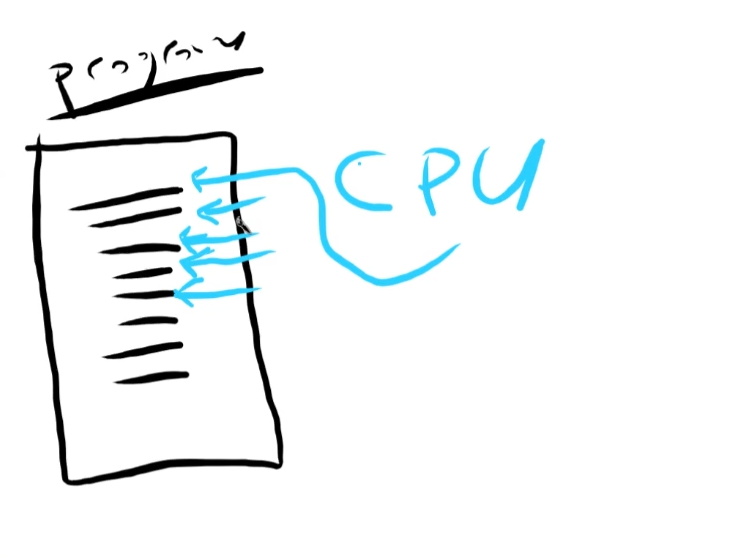
The miners have proof of work (POW) in order to create the blocks.

Ethereum adds another level, and we need to understand this and that is EVM

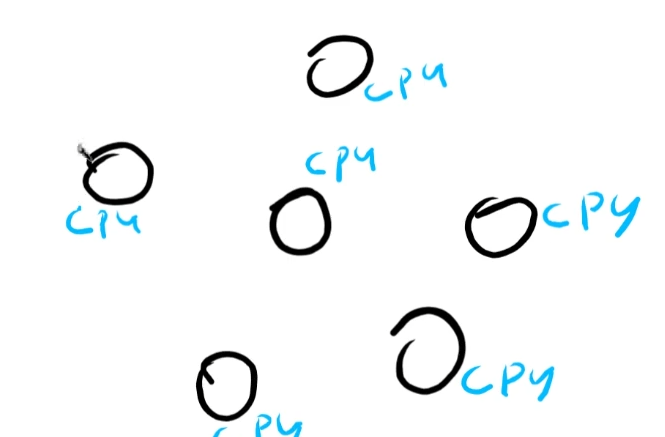
And EVM makes it possible to execute codes and networks.

Remember how Smart contract is a small of codes that can be deployed in ETH network and then you can have unstoppable, transparent application that you can handle money

Normally, a program that is running on your computer has a code written in it, normally CPU would execute these instructions such as remove, add or else. A single CPU would execute the following the instructions.



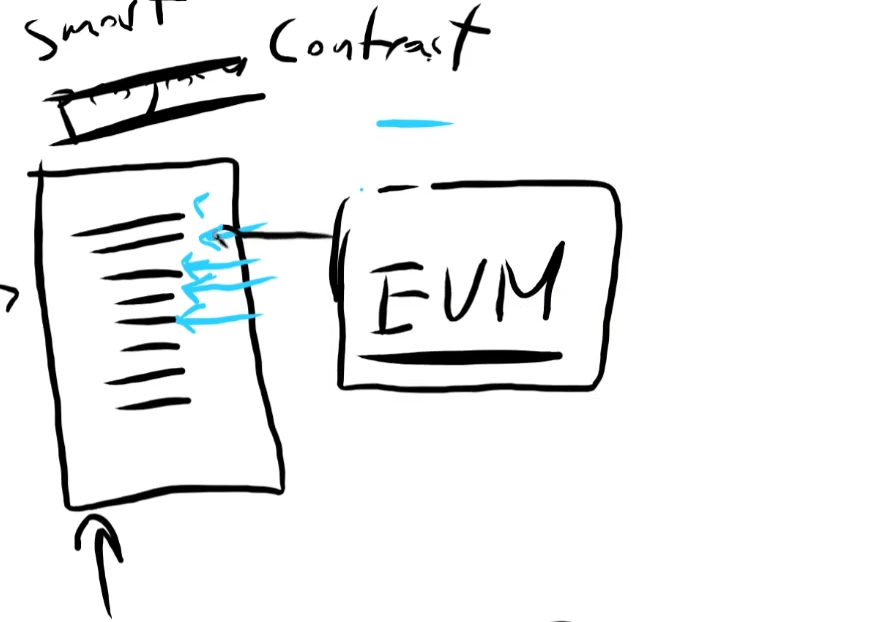
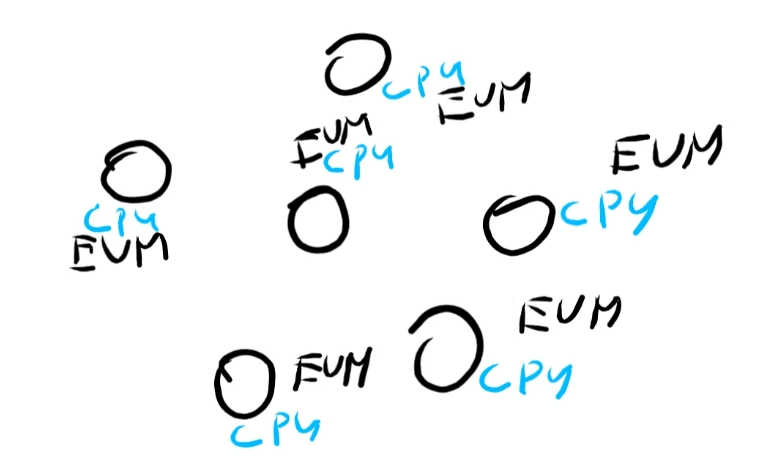
In ETH, the smart contract is normally the same as the program script or instruction mentioned above, but instead ETH is a network of computers so there is no single CPU.



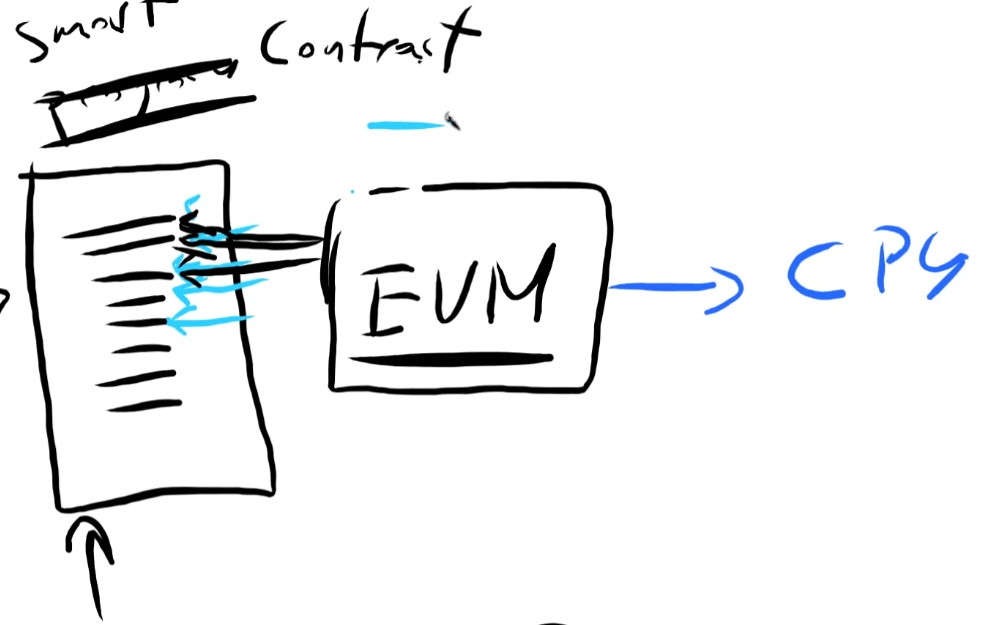
## So how do you make a program that execute on all of the computers?

EVM, ETH virtual machine is a piece of code that all the computers are running. So all the machines/computers are running EVM application, and EVM is running on top of hardware of the existing hardware on each node ETH network so EVM will use CPUs therefore will use all other resources that nodes have.

The good thing about EVM is that it can also read the smart contracts and execute them



So when EVM comes in to play, the CPU is no longer directly executing scripts from program. The CPU is executing indirectly because what is executing it first and foremost is the EVM. **EVM will look at all the instructions** and then it will do all the calculations. **It will basically tell CPU of computer what to do** and it will **end up with results at the end.**



All the other EVMs (nodes), all of the different nodes need to do it. They need to execute all of the instructions in particular smart contracts they need to reach same results and this is how we can have consensus(agreement) results on the entire network. If they have reached the same result then everyone is consensus of what happened in specific contract.

For example if I have a kid and have made an agreement with giving out money if he/she passed an exam, I can lock in my money in smart contract and then when it is time to pay out all of the nodes in ETH network will check so SMART CONTRACT can actually pay and approve this current condition. So they will run this instruction. So, all the nodes have to execute the same piece of codes and then they will reach all of the same conclusion.

So this is how EVM works it is an application that is running on all of the nodes and it is executing on smart contracts and gives instructions to CPU and this is how it works, CPU does not directly run the smart program.

## Why do we have EVM in the middle on the process?

It is mainly for security reasons, because I can deploy any smart contracts in the ETH network.

The Smart Contract may have messages in the instructions to track you down, show images you hold and something you don’t want to share. You do not want smart contract directly linked to your CPU because it will execute anything according to scripts. Smart contract directly linked to CPU is dangerous so we want to incapsulate the smart contracts in the EVM so the EVM will ensure that nothing malicious things will happen. EVM will ensure that the smart contract will not break rules and take over your computer. EVM has the access of your computer.

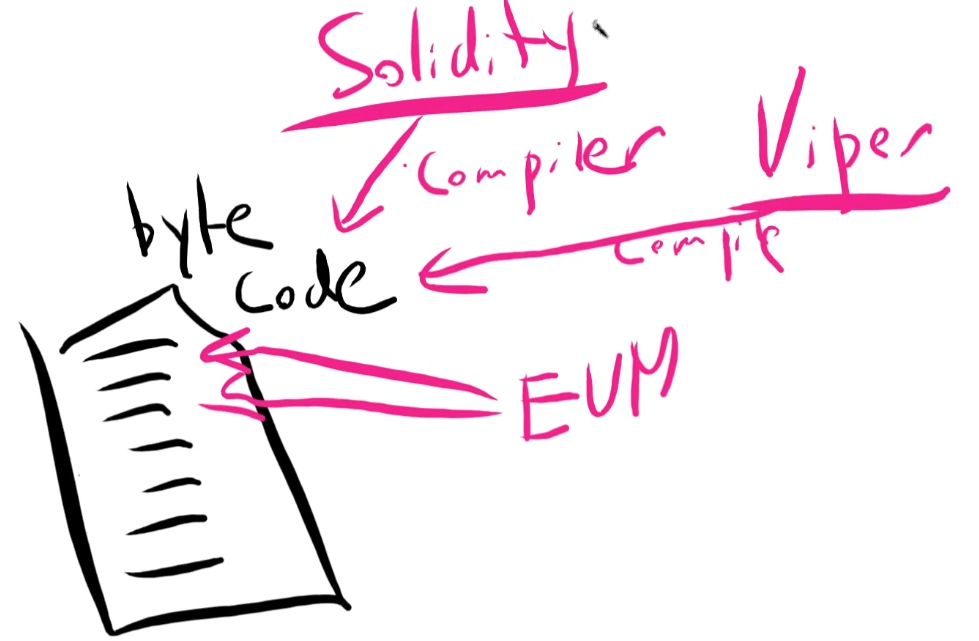
## How does EVM execute instructions?

The smart code is written in byte codes, Solidity. Solidity is like JAVASCRIPT, compiles it down byte code and then EVM will read all the instructions.

Currently the ETH is working on new program language called VIPER and this will compile the same way as solidity

Solidity -> compile -> byte codes -> EVM

VIPER -> compile -> byte codes -> EVM



Important note is that no matter what programming language you choose, the EVM will read the same, only reads the byte codes.

How did this ETH start?

Vitalik founder of ETH wanted to create an idea on top of BTC, so BTC can be programmable and have lots of features. BTC also has some programmable ability called Script, you can do very simple smart contract you can do on bitcoin 2 but however it is very limited.

ETH has solidity and turing complete (TC) which means you can program anything you want.

BTC didn’t want flexibility like ETH have due to potential spamming of loops so therefore we should treat btc as an internet gold or digital money.

So therefore, Vitalik had ETH that has the TC ability and opportunity for developers to program anything they want.

1. What does the EVM do?  
2. What is the format of the instructions that the EVM can read?  
3. What is the relationship between the programming language Solidity and Bytecode?  
4. Why can’t Bitcoin run the same type of complex programs like Ethereum can?  
4. What is a Turing Complete programming language?