FiiiCOIN

A versatile, scalable and energy efficient blockchain technology

Yellow Paper

Authors

Sylvester Lee
CTO & Founder

John Liu
Solutions Architect

Abstract

FiiiCoin is a transaction network specifically designed for mobile devices mining purpose only. The blockchain technology enable all mobile devices participate in maintaining the blockchain network while leaving it idle and charging battery instead of relying on expensive and powerful computer hardware running 24/7 to do the mining work. The main objective is to create a least effort way and promote re-using the existing available resources (mobile devices) together to take part in maintaining the blockchain. Even the non-IT people or non-crypto fans can easily learn and involve in mining as long as they at least have a smartphone.

In fact, smartphone or any other mobile devices are not suitable to be setup as a blockchain node. The reasons are the computing power is weak, storage capacity is limited to keep a full blockchain and battery powered device are not meant to be 24/7 running all time, and it requires high network bandwidth to synchronize blocks data. If the mobile device runs the node using 4G network, it will finish up all the bandwidth in no time.

FiiiCoin is built using an enhanced version of Proof-of-Capacity consensus algorithm – Delegated Proof-of-Capacity (DPoC) to achieve the mobile mining capability while keeping the mobile device from being a full node and perform extensive block synchronization work. FiiiCoin is developed using an in-house built customizable blockchain technology - FiiiChain. FiiiChain provide standard blockchain modules with the “plug-and-play” capability for developers to modify the blockchain characteristic base on the given business requirement.

P2P Network

FiiiCoin network is based on P2P (Peer-to-Peer) network architecture which constantly connect to multiple nodes within the same network to synchronize data. P2P network means each computer in the same network is identical, same privilege and authorization to communicate with each other in flat topology. In addition to peer-to-peer protocols, FiiiCoin network also include additional custom gateway protocol to provide routing service to enhance the efficiency of P2P network connectivity, so that it can help newly added nodes to the network can quickly access to other nodes.

Routing Protocol

The FiiiCoin network contains multiple master-node servers that are distributed around the world, providing the blockchain data synchronization service to every participating nodes, and they run continuously non-stop to help other nodes to discover other nodes. FiiiCoin has a global tracker service hosted to seed all the active nodes at tracker.fiii.io, which work similarly to BitTorrent. The seeder provides all the IP addresses of all active nodes to the node which just join the FiiiCoin network. All the active nodes periodically provide heartbeat data to the tracker to indicate they are still active in the network.

Node Distribution Strategy

Each FiiiCoin node is a collection of functions for routing, blockchain databases, mining, and wallet services. Each node participates in the whole network routing function, participates in verifying and propagating the transaction and the block information, discovers and maintains the connection with the P2P nodes.

When a new node is online, all the node IP lists are obtained from the tracker and then attempt to connect to these nodes.

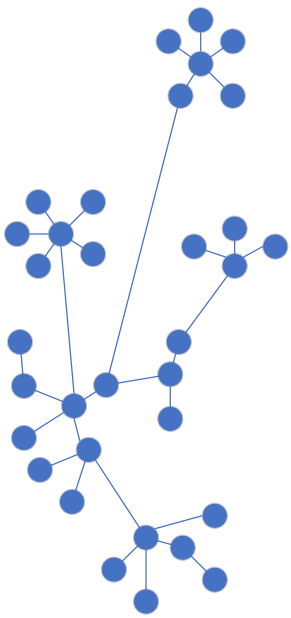
Whenever a node establishes a connection to another node for the first time, the first thing to do is to get the full list of IP addresses which the other node currently connecting, download the IP list to own node address database.

When a node is passively connected by another node, the passive node requests the latest IP list from the active node to synchronize its own node address database.

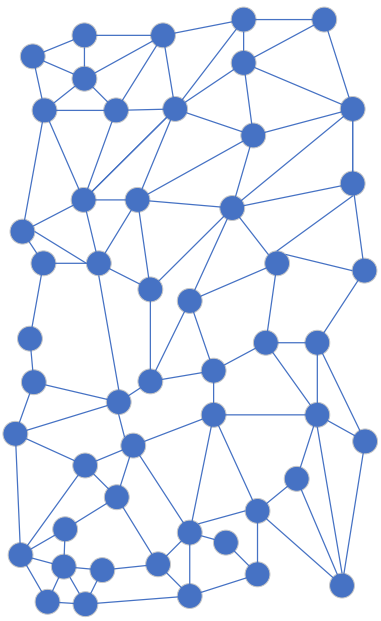
Each node will randomly establish connection to 100 active nodes base on the IP list from the own node address database. In order for each node to maintain a fast network speed, reducing the chances of data packet loss, avoid network congestion, the strategy applied is to select 100 nodes that to be connected, 50% are less than 100ms response time, 30% for less than 300ms response time, 10% for less than 500ms response time, and lastly 10% for greater than 500ms response time. The network latency can be used to assume the distance between nodes. The lower the latency, the nearer the nodes are, while the higher latency mean the node is far away from the connecting node.

If an active connection gets disconnected for some reason, the node will automatically attempt to reconnect. If the node is unable to be connected after 3 attempts, the node will be removed from the node address database since it is an offline node.

The strategy is to make sure the nodes are connected in real distributed way across the globe, preventing nodes being overly packed in one single location if the nodes are connecting to nodes which has the fast connection only.



Unhealthy Distribution



Healthy Distribution

Accounts

FiiiCoin wallet contains only keys, not cryptocurrency. Each user has a wallet that contains multiple keys. The wallet contains only the keychain with the private/public key pair. The user uses the key to sign a transaction to prove the ownership of the Unspent Transaction Output (UTXO). The FiiiCoin amount that stored in the wallet is based on the total balance of UTXO set in the blockchain that user own.

Private Key & Public Key

FiiiCoin contains a series of asymmetric key pairs, and each asymmetric key pair includes a private key and a public key. The private key is a random number, and the public key is obtained by single encryption via the Elliptic Curve Cryptography (ECC) algorithm. The user produces a digital signature using the private key to sign a transaction, and then using the public key to apply data fingerprint. The application of this asymmetric cryptography allows anyone to verify the signature of each transaction, while ensuring that only the owner with the correct private key can produce a valid signature.

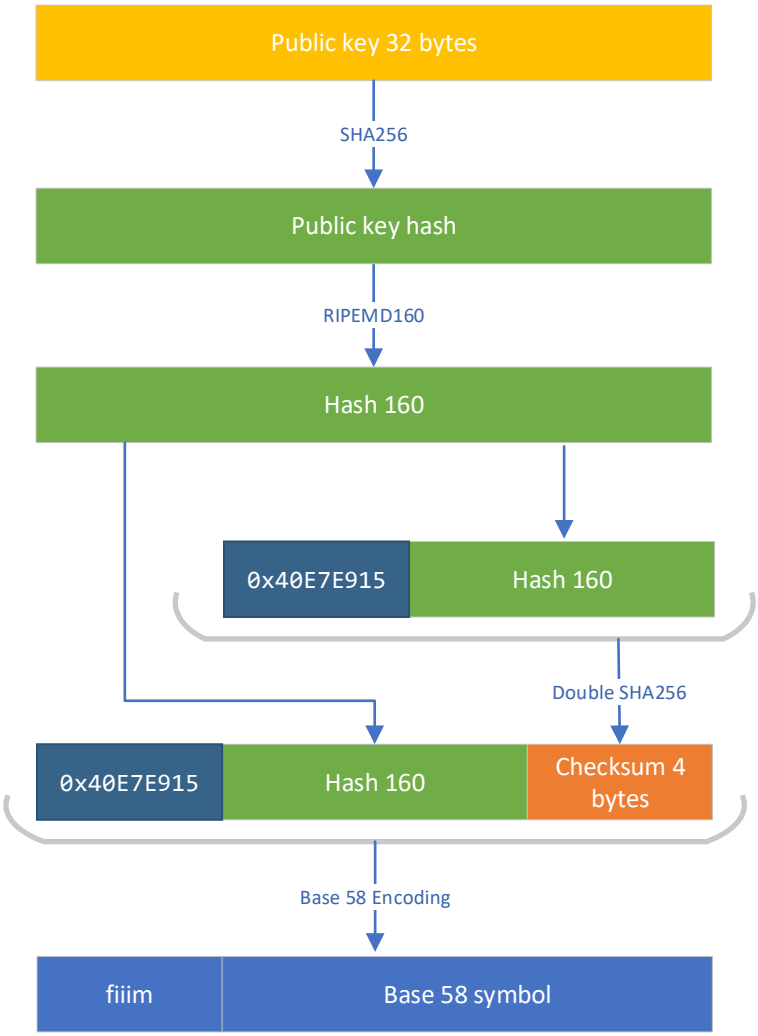
The private key is a 32 bytes long random number, but its generation process is ensured that the results are unpredictable and unique. Hence, FiiiCoin public/private key pair is generated using Ed25519[1] high-speed and high-security signature algorithm which is derived from EdDSA (Edwards-curve Digital Signature Algorithm). The benefits [2] of Ed25519 are:

1. Fast single-signature verification
2. Even faster batch verification
3. Fast signing capability
4. Fast key generation
5. High security level
6. Fool proof session keys
7. Collision resilience
8. No secret array indices
9. No secret branch conditions
10. Small signatures
11. Small keys

The generated private key is a byte stream (bytes) of length 256 bits, which is stored in the keys.dat file in the Wallet directory.

Address

FiiiCoin address is a string of characters with combination of numbers and letters but excluded the uppercase letter "O", uppercase letter "I", lowercase letter "l", and the number "0" are never used to prevent visual ambiguity. The address is generated by hashing the public key with SHA256, following by perform RACE integrity Primitives Message Evaluation (RIPEMD160) public key hash, then lastly encode the final hash with Base58 encoding to become an address which is unique, recognizable and verifiable.



fiiimkDjC3adfPkseGazFFesdawE241daLdn

Address generation process flow

For mainnet, the address prefix is fiiim
For testnet, the address prefix is fiiit

Transactions

FiiiCoin is a huge public ledger that keeping all the transaction records into blocks which is similarly to Bitcoin [3] – the blockchain technology originator. Transactions are the core data which to be distributed to all the participants in the network.

Input & Output

The FiiiCoin transaction consists of a set of inputs and outputs, whereby the output is the fundamental of the blockchain transaction. A transaction must have at least one output or more, but it can have no input. The output is inseparable, stored in the blocks, recognizable by the entire blockchain network. All the spendable FiiiCoin is actually Unspent Transaction Outputs (UTXO). The wallet stores the collection of UTXO (known as UTXO Set) in a wallet.dat file, from there calculating and displaying the total amount of FiiiCoin which is usable.

When someone send a FiiiCoin to a person, the wallet forms a new transaction that the sender UTXO become input, while the recipient will receive as output. Each UTXO can only be consumed once. UTXO behaviour is similar to a bank note. When transferring an amount of FiiiCoin to a recipient, it may involve combination use of multiple UTXOs to achieve same or greater than the transfer amount to be set as transaction inputs, the excess output amount will be returned back with a new UTXO or add up into an existing UTXO if requested.

Mining Fee

FiiiCoin mining fee can be as low as 0.00000001 FIII per KB. FiiiCoin is powered by Delegate Proof of Capacity [4] (DPoC) consensus algorithm whereby only the dedicated super nodes responsible in mining and forming blocks, hence the mining is less competition and the mining fee is low. Refer to the consensus algorithm section below for more DPoC explanation.

Transaction Data Structure

The transaction data structure is as follow:

```
{
  "version": 1,
  "locktime": 0,
  "vin": [{
            "txid":"7957a35fe64f80d234d76d83a2a8f1a0d…",
            "vout": 0,
            "sequence": 4294967295
          }],
  "vout": [{
            "value": 0.01500000,
            "scriptPubKey": "OP_DUP OP_HASH160 ab68025513c3db… OP_EQUALVERIFY OP_CHECKSIG"
           },
           {
            "value": 0.08450000,
            "scriptPubKey": "OP_DUP OP_HASH160 7f9b1a7fb68d60… OP_EQUALVERIFY OP_CHECKSIG",
          }]
}
```

version: Transaction version

locktime: The time for the UTXO to be spendable

vin: Inputs

- **txid**: Current transaction ID
- **vout**: The sequence order of the output
- **Sequence:** Transaction time but must be greater or equal to the transaction lock time.
- **scriptSig**: Unlock script with digital signature and public key. Refer to transaction script section for more detail.

vout: Outputs

- **value**: Transfer amount
- **scriptPubKey**: Unlock script with digital signature and public key. Refer to transaction script section for more detail.

Transaction Script

The FiiiCoin transaction verification engine relies on two types of scripts to authenticate: Locking scripts and unlocking scripts. Locking scripts often contain a series of instructions and a public key hash (obtained from the recipient wallet address). An unlock script typically contains a user's public key and a digital signature generated by the user's private key.

Scripts are typically used in the following two scenarios:

Validating UTXO

After the node receives a new block, it read the locking script instructions, iterates through all of the transactions and attempting to match its own public key and digital signature for unlock verification. If matching is successful, the node claims the output ownership, then added the output to node UTXO set, and then updates the wallet balance.

Transaction Verification

While the miner prepares the block data or during the node receives a new block, it validates all the inputs in the transaction by extracting the digital signature and public key, and then trying to match the inputs with the outputs from other transactions, and then unlock the output using its own digital signature and a public key. If the unlock is successful, it indicates that the transaction input is valid, otherwise, the transaction is considered illegal or tampered.

Transaction Confirmation

After a transaction is generated, it is first entered into the Transaction Pool (or known as Memory Pool) and then broadcast the transactions to all the connecting nodes in the entire blockchain network.

The miners first verify the genuine of the transactions (as explained in the section Transaction Verification above), and then prioritize the transactions base on the transaction fee (mining fee / the byte length of the transaction) and writing the transactions into a block, then undergo the mining process. If the block being successfully mined, block being broadcasted to all the connecting nodes.

A transaction is considered confirmed only after the transaction in the block has 8 behind. Once the transaction is confirmed, the transaction outputs will be unlocked base on the unlock script and then UTXO set and wallet balance will be updated.

Blockchain

Blockchain is a series of data that linked between blocks base on individual block hash ID, and each block contain the data structure of a public ledger that book keeping transactions between accounts. It can be stored as flat files or a simple database. FiiiCoin uses the SQLite database to store metadata. The blocks are chained from old to new sequence order, and each block points to the previous block. Blockchain are often treated as a vertical stack, the first block (known as Genesis Block) will be at the very bottom of the stack, and each block is stacked above of the previous block. The "height" is to indicate the distance of the block from the first block.

Blocks

Block is a ledger that contain transaction records. FiiiCoin block size is limited to 12MB containing all the transactions data size add-up. The following are the block data structure:

Attribute	Data Type	Nullable	Remark
ID	String	No	Block ID hash value
Height	Long	No	Block height
Version	Integer	No	Block version
Timestamp	Datetime	No	Block creation time
PreviousBlockID	String	No	Previous block hash
NextBlockID	String	Yes	Next block hash
Nonce	Long	No	Mining nonce value for this block hash
Bits	Decimal	No	Locking time
Difficulty	Decimal	No	Difficult of mining this block
Size	Long	No	Block size

$$tps = \frac{block\ size\ \times\ 1024^2 \div tx\ size}{time}$$

The block generation time is 1 minute (60 seconds). Since each transaction size is around 300 bytes, the coinbase transaction size is about 250 bytes. With 12MB block size limit, it can store 41,943 transactions per block. Hence, theocratically it can support around 700 transactions per second.

Each block will be generating 250 FiiiCoin coinbase transaction to the miner. The coninbase reward will be halved every 5 million blocks. The following are the coin supply rules:

Stage	Coin Supply	Block Reward	Blocks
Genesis	2,500,000,000.00000000	0.00000000	0
1	1,250,000,000.00000000	250.00000000	5,000,000
2	625,000,000.00000000	125.00000000	10,000,000
3	312,500,000.00000000	62.50000000	15,000,000
4	156,250,000.00000000	31.25000000	20,000,000
5	78,125,000.00000000	15.62500000	25,000,000
6	39,062,500.00000000	7.81250000	30,000,000
7	19,531,250.00000000	3.90625000	35,000,000
8	9,765,625.00000000	1.95312500	40,000,000
9	4,882,812.50000000	0.97656250	45,000,000
10	2,441,406.25000000	0.48828125	50,000,000
11	1,220,703.12500000	0.24414063	55,000,000
12	610,351.56250000	0.12207031	60,000,000
13	305,175.78125000	0.06103516	65,000,000
14	152,587.89062500	0.03051758	70,000,000
15	76,293.94531250	0.01525879	75,000,000
16	38,146.97265625	0.00762939	80,000,000
17	19,073.48632813	0.00381470	85,000,000
18	9,536.74316406	0.00190735	90,000,000
19	4,768.37158203	0.00095367	95,000,000

20	2,384.18579102	0.00047684	100,000,000
21	1,192.09289551	0.00023842	105,000,000
22	596.04644775	0.00011921	110,000,000
23	298.02322388	0.00005960	115,000,000
24	149.01161194	0.00002980	120,000,000
25	74.50580597	0.00001490	125,000,000
26	37.25290298	0.00000745	130,000,000
27	18.62645149	0.00000373	135,000,000
28	9.31322575	0.00000186	140,000,000
29	4.65661287	0.00000093	145,000,000
30	2.32830644	0.00000047	150,000,000
31	1.16415322	0.00000023	155,000,000
32	0.58207661	0.00000012	160,000,000
33	0.29103830	0.00000006	165,000,000
34	0.14551915	0.00000003	170,000,000
35	0.07275958	0.00000001	175,000,000
36	0.03637979	0.00000001	180,000,000
37	0.01818989	0.00000000	185,000,000
Total	4,999,999,999.92724000		

Total FiiiCoin maximum supply is 4,999,999,999.92724000. Base on the coin allocation for the FiiiLab marketing purpose, and considering the blockchain economy, the genesis block has been hardcoded to mint 2.5 billion of FiiiCoin into a designated account. Remaining 2.5 billion of FiiiCoin to be open for mining.

When the total of 185,000,000 blocks have been mined, there will longer be any coinbase reward. The miner can only earn from the transactions fee.

Genesis Block

The first block in the blockchain is called the "Genesis Block", which is the common ancestor of all the blocks within the block chain, meaning that if you backtrack from any of the block, ultimately will go back to the genesis block. The height of the Genesis block is 0, and its previous block hash value is always 0000000000000000000000000000000000000000000000000000000000000000.

Byzantine Fault Tolerance

FiiiCoin nodes contain a *Block Pool* that temporary keep unconfirmed blocks, then perform block validation and identify the block which has forged signature base on the transaction script to prevent double spending. The transaction block need to be recognized in at least 2/3 of the entire blockchain network then only considered as the genuine chain.

FiiiCoin nodes periodically synchronize blocks at the same time validating the chain with all the peers, and capable of rejecting false blocks with unrecognized digital signature, making those blocks become orphan blocks.

The rule is simple:

1. The longest chain
2. The highest accumulative difficulty chain
3. Recognizable in at least 2/3 of the entire blockchain network

There may be a case whereby the network latency delays the block synchronization causing the block with greater height has no parent. The node cannot reject it yet but have to temporarily keep it in the *Block Pool*, until there is a same height block already chained into the main chain, the block will be rejected permanently.

Delegate Proof of Capacity

Proof of Capacity [5] (PoC) or also known as Proof of Space (PoSpace) is a consensus algorithm originated from Burstcoin [6] which is somehow similar to Proof of Work [7] (PoW). However, PoC uses storage instead of computing power to perform proof of work. PoC is energy efficient for mining and the concept is suitable for mining with mobile device which has weak CPU and memory.

Due to the consideration of the balances of security, decentralization and performance of the blockchain, FiiiCoin form a new delegated PoC consensus that limiting certain participation in the blockchain network which aim to solve the following problem:

1. Limiting mining access to prevent 51% attack
2. Improve performance
3. Better crowd control

Consensus Formation

Base on the implementation of Delegate PoC, there will be 3 types of node in order to form the consensus.

Super Node

The super node is the main delegated node that responsible in generating new block, verification and data synchronization. The roles of the super node are as follow:

1. Identify and delegate mining access.
2. Issue proof of capacity mining work.
3. Validate mining work.
4. Construct blocks.
5. Synchronize & broadcast blocks.

The super node will be setup with a mining pool to cater for managing the miner delegation and coinbase reward allocation.

Mining Node

Each mining node is registered on the super node and allocates a desired storage space for storing the pre-computed hash data beforehand to get mining work from the super node.

Mining node need to obtain the access token from the super node by providing digital signature and account ID. The mining access can only support single session, using the same account ID and digital signature to get work from the super node will refresh the access token, eventually kick the previous connecting session.

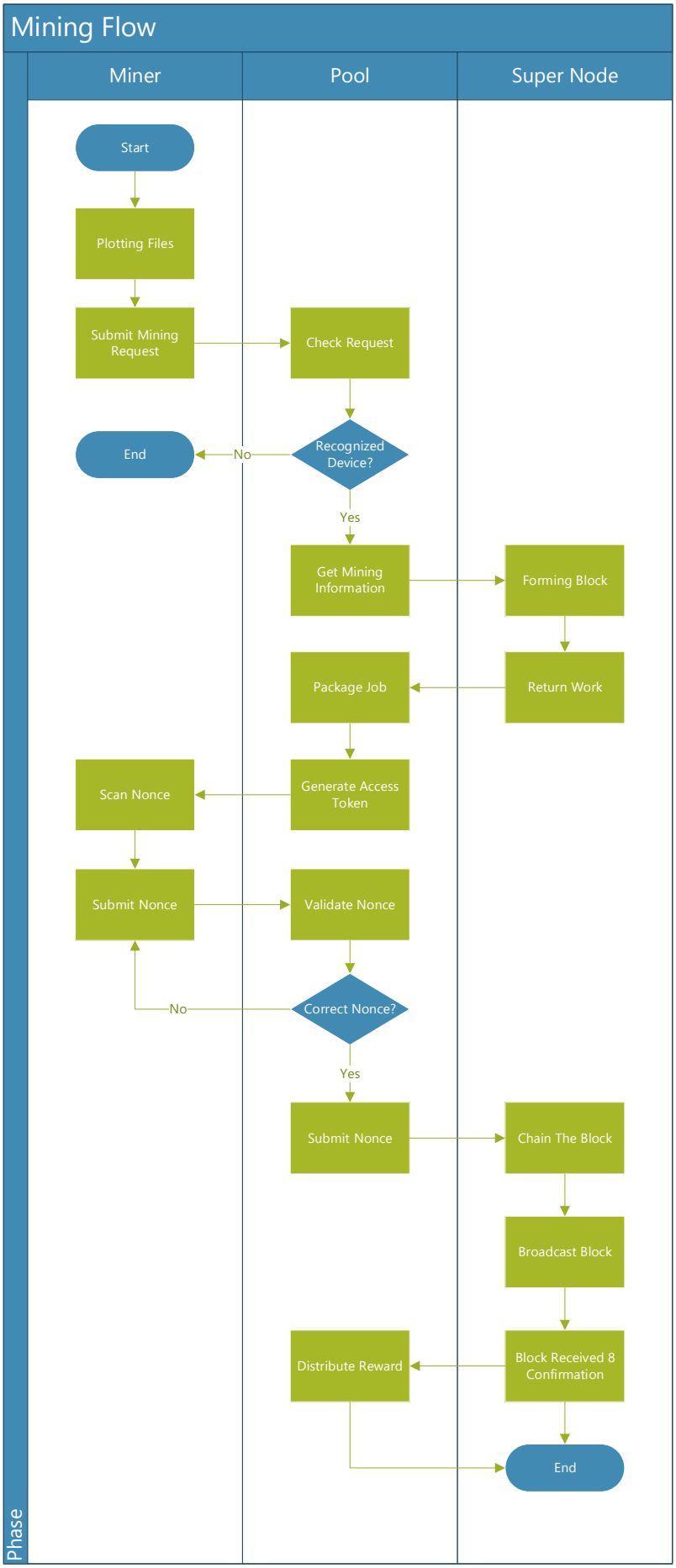
Wallet Node

Wallet node is the basic blockchain node that download blocks data, validate blocks, submit transaction, check account balance and providing routing services to the peers.

Mining Process Flow

The mining process is as follows:

1. The miner submits the following information to the super node to request for mining access:
    a. FiiiPOS Serial Number
    b. Device ID
    c. Account ID
    d. Wallet address
2. The super node validates the device and approve the mining access request.
3. The super node return access token to the miner.
4. The miner proceeds to request job with the access token from the super node
5. Once the miner completed the job and then repeat step 4.
6. The miner receives the coinbase reward if the nonce is found successfully.



Plotting

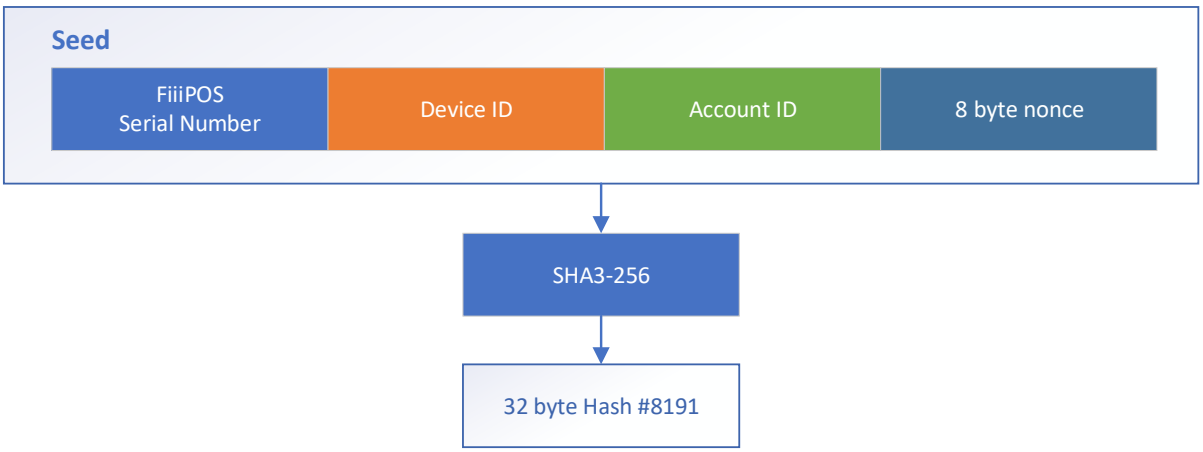
Prior to mining, file plotting is the first mandatory step to compute all the hashes and nonce and then save them into storage space.

Nonce is an integer range from 0 to 2^{64}. Hashes are generated base on nonce. Each nonce generates 256KB of unique hash data. The amount of allocated storage space determines how many nonce and hashes can be stored.

All the generated nonce and hashes can be grouped into a scoop, and each scoop has 4,096 slots. Each scoop is 64 bytes large and it is formed base on two 32 bytes hash value. Each scoop has an ID from 0 ~ 4095.

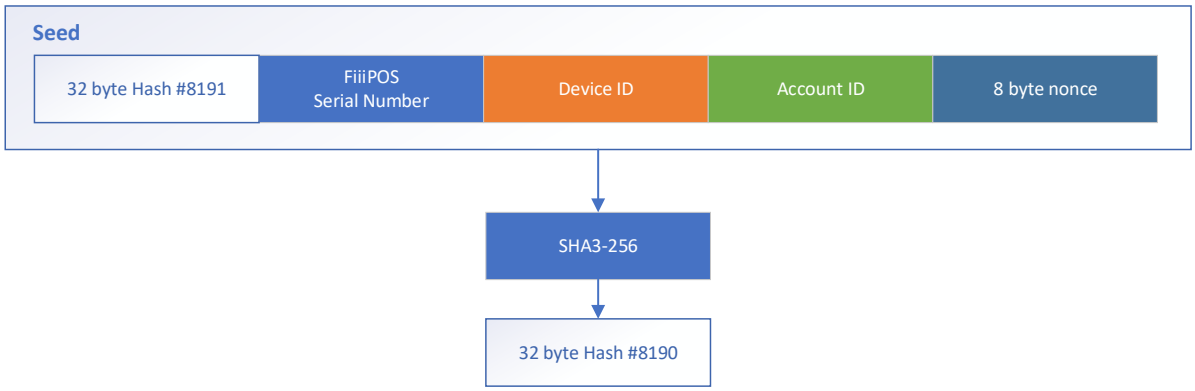
FiiiCoin uses SHA3-256 to create mining hash values. Nonce is the base value parameter given to the SHA3-256 hash function to generate a unique hash value. The reason using SHA3 is to make the hash value difficult to compute and making it difficult in building Application-Specific Integrated Circuit [8] (ASIC) mining machine for FiiiCoin. Also, since the hashes need to be pre-computed only once during the file plotting, it become an advantage to use the SHA3 slow hash which is not suitable in real time mining.

Before generating hash value with nonce, it prepares seed data first, then only perform SHA3-256 hash, then save it into storage. The seed data consists of FiiiPOS serial number, device ID, account ID, wallet address and 8 bytes long nonce. The hash value will be stored in the scoop with a hash label.



As mentioned above, each scoop slot has 64 bytes of storage which can keep two 32 bytes of hash values, hence one scoop can keep 8192 hashes. So, the first hash will be labelled as #8191 starting form the back.

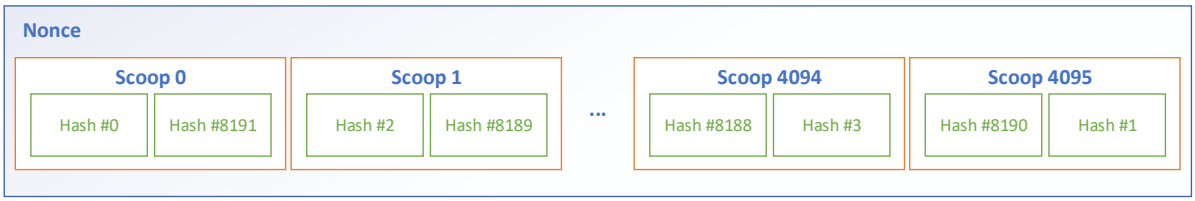
Next, form a new seed with #8191 hash, then hash again to get #8190 hash.



According to the previous step, the length of the seed data exceeds 4096 bytes when the calculation exceeds 128 times. When the seed length is greater than 4096 byte, the seed used in the subsequent step calculation takes only the first 4096 byte. In other words, the seed of the SHA3-256 operation does not exceed 4096 bytes.

Base on the above steps, repeat the hash function continuously until 8,192 hash value generated, the last hash value will be hash #0, also known as the final hash.

The following is how the scoop and hashes being stored in the plot file:



As the DPoC mining consensus rule, the super node will first determine which scoop to find for nonce of current block hash. In order to enhance the efficiency of the mining node, the system will automatically optimize the data storage format for easy searching correct plot file.

The plot file name format is:

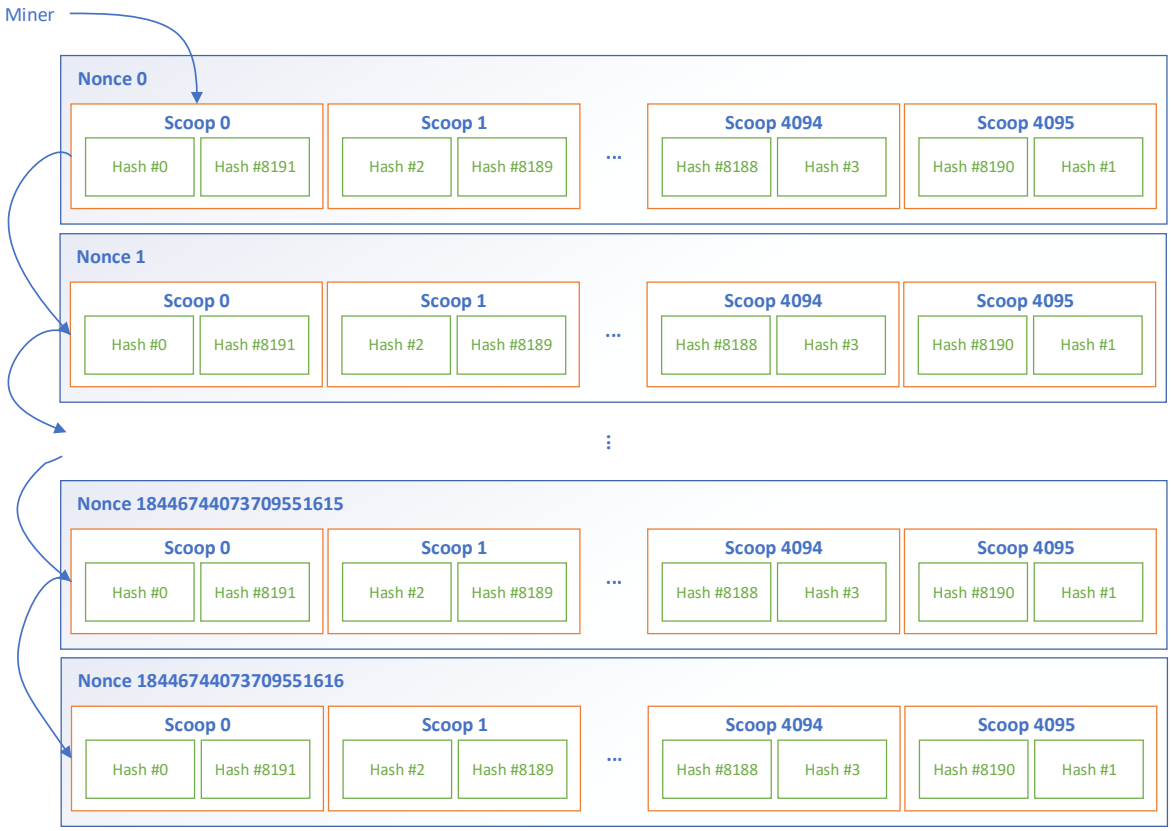
DeviceID_AccountID_WalletAddress_ScoopNumber_StartingNonceNumber_NumberOfNonce

With the above mechanism, the mining process force the miner to read the plot file from beginning till the end, there will be no cheating, though in Burstcoin had an optimized plot file [9] way to reduce the scanning time, but FiiiCoin will remain the unoptimized plot file structure.

Mining

Once the plot files have been created, miner connect to the Super Node to get mining information. The Super Node will forge a block by taking transactions from the Memory Pool. Then, base on the last 8 blocks generation time, calculate the *BaseTarget* value as the deadline in order to maintain the 60 seconds block generation time, then follow by the scoop number.

The miner starts scanning from the nonce 0 file until the last file, then look for the scoop and hash value from within.

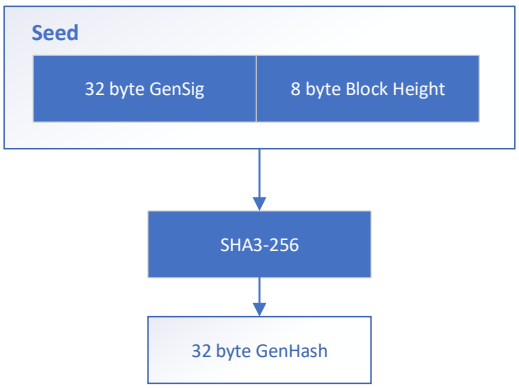


The rule to determine the block is mined successful is the nonce value for the corresponding hash value has the derived *Target* value in the specific scoop is lower or equal than the *BaseTarget*.

The following is the process of determining the scoop and target:

1. Generate the block signature (*GenSig*) by hashing the previous block signature and current block generator address with SHA3-256.

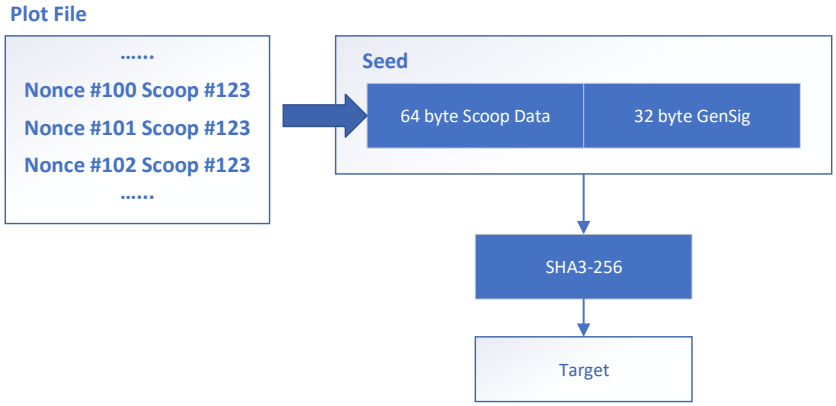
2. Obtain the *GenHash* by hashing the *GenSig* and *Block Height*.



3. Calculate the scoop number by modulo 4096 of *GenHash*.

4. Miner search for the nonce in the specific scoop number only

5. The 64 bytes scoop data and 32 bytes GenSig will be hashed to get the *Target*.



6. Compare the *Target* and *BaseTarget*, if Target is lesser or equal to *BaseTarget*, it means the nonce is correct, Super Node will chain the block and broadcast to inform all the connecting peers.

Mining Difficulty

In order to maintain the block generation time to average 60 seconds, the mining difficulty will be automatically adjusted every 24 blocks. When the difficulty is high, the *BaseTarget* value will be low as the deadline is short. When no miner is able to complete the mining within the deadline, the miner obtained lowest target value will be the winner.

$$D = O \times \left(\frac{\sum t}{60}\right)$$

The new difficulty D is calculated based on previous block old difficulty (O) and sum of last 24 blocks time (t) divide by 60 seconds.

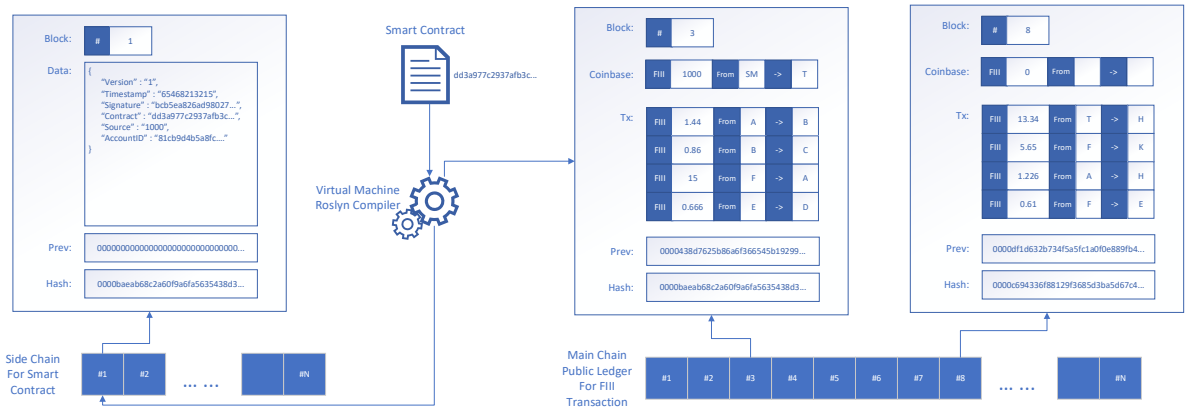
Smart Contract

Smart contract is a pre-programmed script that to be executable when certain criteria or situation is met. The contract is not amendable and has its own digital signature.

FiiiCoin is also equipped with smart contract capability. It relies on Microsoft Roslyn [10] compiler to perform the smart contract script execution. Hence, the smart contract can be coded in C# or VB.NET. Roslyn compiler is supported in Windows, Linux and MacOS platform.

Smart Contract script will be digitally signed by the contract creator. The signed contract has a unique hash as fingerprint and given a unique address that to be kept in a side chain.

The smart contract script will be kept in the *Contract Node*.



The main chain will be remained as public ledger to track transactions between accounts. The additional side chain is specifically to keep smart contract. The reason doing so is to maintain the cleanliness of the blockchain and clearly differentiate the aspect of purpose.

The contract is in *.cs for C# or *.vb for VB.NET file format and the script content follow the FiiiCoin smart contract template.

```
using System;
using FiiiLang;

namespace FiiiCoin.Contract
{
    class ExchangeContract
    {
        public void Deposit(string address, decimal amount)
        {
            //some coding
        }

        public void Convert(decimal amount, DateTime date)
        {
            //some coding
        }

        public void Withdraw()
        {
            //some coding
        }
    }
}
```

Signing Contract

After the contract has been created, the creator uses its own account private key to sign the contract. The contract will be uploaded to the Contract Node for compilation. The Contract Node has the collection of all smart contracts. The virtual machine run in the Contract Node is responsible in executing the smart contract when the contract defined condition is met.

Contract Bug

There is no developer that does not create bug despite the script has been fully tested. Since the contract is kept in the Contract Node and not in the blockchain, the contract can be amended and then re-signing and reupload to the Contract Node.

The new contract signature will be updated in the side chain. There will be a *PrevContractHash* reference that act as a contract chain that telling the Contract Node there is a newer version of the contract.

Contract Attributes

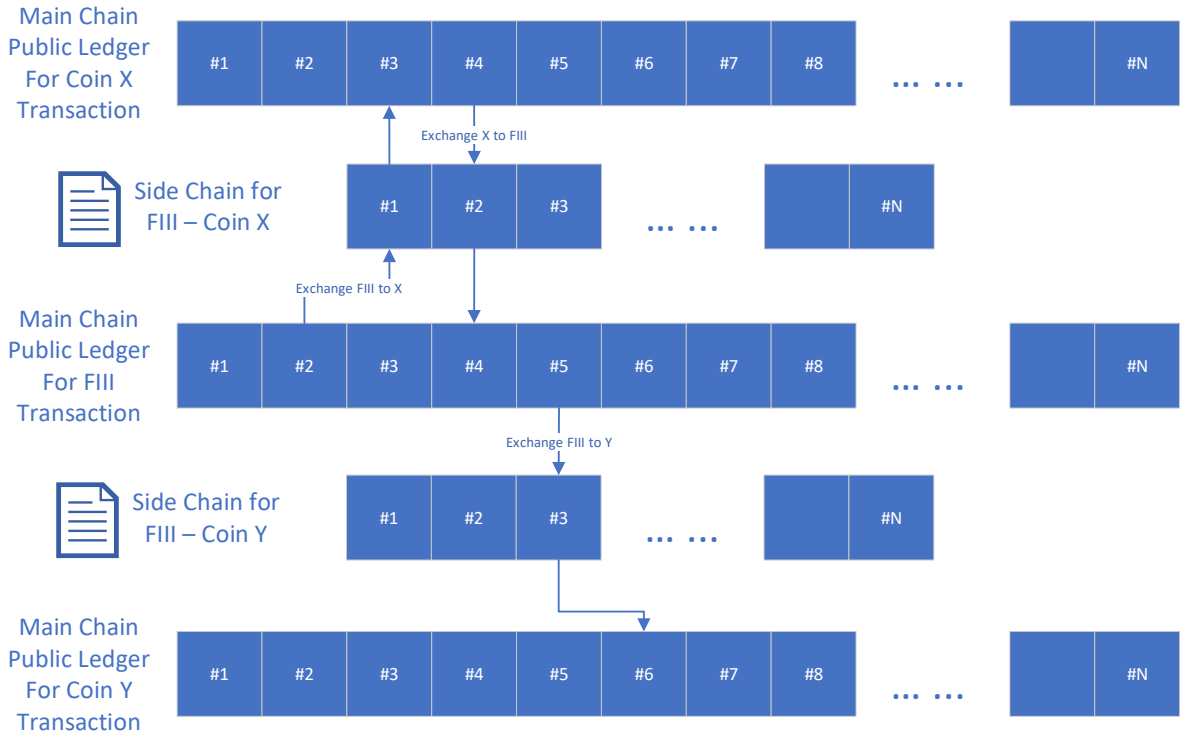
The following are the contract attributes that store in the side chain:

Attribute	Data Type	Nullable	Remark
ID	String	No	Contract ID hash value
Name	String	No	Contract name
Version	Integer	No	Contract version
Language	Integer	No	1 = C#, 2 = VB.NET
Timestamp	Datetime	No	Contract upload time
PreviousContractID	String	No	Previous contract hash
NextContractID	String	Yes	Next contract hash
CreatorSign	String	No	Contract creator signature
AccountID	String	No	Contract creator account ID

FiiiCoin smart contract is not meant for public use and it is not a blockchain operating system platform like Ethereum or EOS. FiiiLab use a different approach for Platform as a Service (PaaS), which is to provide a Blockchain as a Service (BaaS) by creating an online platform that generate blockchain source code template based on user requirement to customize. The new blockchain from other project can link to FiiiCoin blockchain to support cross chain communication.

Multichain

FiiiCoin leverage on smart contract to perform multichain and cross-chain communication, as ultimately, the payment and crypto exchange solution will go fully decentralized. The smart contract in FiiiCoin is used to track the exchange activity of other crypto.



All the crypto-to-crypto exchange must go through a smart contract to book keeping the transaction. The third party crypto that uses FiiiChain technology will have a ready API that capable of direct communicate with FiiiCoin smart contract to perform automated exchange.

Acknowledgement

The authors would like to thank all the Fiii community, enthusiast, fans, evangelist with many useful feedbacks that bring success to the technical implementation. Also, would like to express our gratitude to Burstcoin for the original idea of Proof of Capacity consensus, Bitcoin for the P2P payment and blockchain idea, Ethereum for the smart contract, and EOS for the delegation approach in risk mitigation.

The content of yellow paper is subject to update in the future in case any changes in the technical implementation.

References

[1] Ed25519 high-speed high-security signatures - https://ed25519.cr.yp.to/ed25519-20110926.pdf

[2] The features of Ed25519 - https://ed25519.cr.yp.to/index.html

[3] Bitcoin: A peer-to-peer electronic cash system - https://bitcoin.org/bitcoin.pdf

[4] Delegate Proof of Capacity - https://sylvester-lee.blogspot.com/2018/07/delegate-proof-of-capacity.html

[5] Proof of Capacity - https://en.wikipedia.org/wiki/Proof-of-space

[6] Burstcoin - https://burstwiki.org/wiki/Whitepaper:Burst

[7] Proof of Work - https://en.wikipedia.org/wiki/Proof-of-work_system

[8] Application-Specific Integrated Circuit - https://en.wikipedia.org/wiki/Application-specific_integrated_circuit

[9] Burstcoin Optimized Plot File - https://burstwiki.org/wiki/Technical_information_to_create_plot_files

[10] Microsoft Roslyn Compiler - https://github.com/dotnet/roslyn

Change Log

Version 0.1 Initial draft of the yellow paper