



# **Bitcoin Asset sidechain**

white paper

bitcoinasset.org  
October 2017

# Content

<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	Blockchain Consensus Models	4
1.1.1	Proof of Work	4
1.1.2	Proof of Stake	4
1.1.3	Byzantine Agreement Protocols	5
1.2	Blockchain Architecture	5
1.2.1	Account based	6
1.2.2	Unspent Transaction Output (UTXO) based	6
1.3	Assets	7
<b>2</b>	<b>Bitcoin Asset sidechain</b>	<b>7</b>
2.1	Abstract	7
2.2	Motivation	7
2.3	Consensus algorithm	8
2.4	Logical Clock Proof of Stake algorithm	10
2.4.1	Brief definition	10
2.4.2	User Deposit	10
2.4.3	Time slot and epoch	11
2.4.4	Source of random and bets	12
2.4.5	Leader selection	13
2.4.6	Economic Initiatives	14
2.4.7	Long range attack	15
2.4.8	Precomputation attack	15
2.4.9	Stake Leasing	15
2.5	Blockchain architecture	15
2.6	Scripting language	16
2.6.1	Interaction with Bitcoin blockchain	16
2.7	Native Assets	17
2.7.1	Create Asset	18
2.7.2	Initial Asset distribution	18
2.7.3	Buy Assets (Crowdsale)	19
2.7.4	Buy Assets from Bitcoin blockchain	19
2.7.5	Genesis Asset	20
2.7.6	Native Assets exchange	20
2.7.7	Exchange Assets with Bitcoin	21
2.8	Node	23
	<b>Conclusion</b>	<b>23</b>
	<b>References</b>	<b>24</b>

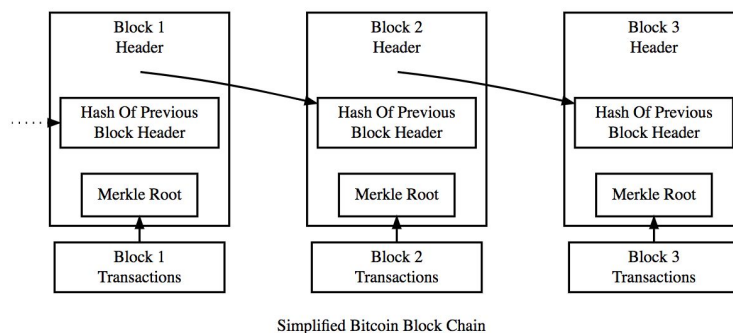
# 1 Introduction

The history of cryptocurrency begins in 2009, when Satoshi Nakamoto posted a document describing the work concept of the Bitcoin and the launch of the Bitcoin network itself [1]. The most significant and valuable invention made by Satoshi Nakamoto is a group signature in the system, where the number of participants can vary randomly. This invention may be considered as a new type of group signature *dynamic-membership multi-party signature* (DMMS).

Up to this point, all the existing digital currency systems required confidence to any central server. The server which inspired confidence had two main tasks: signing transactions and preventing double spend. Such systems are vulnerable and have a fault point in central server. Even if the system is distributed on several servers and the agreed specified number of them available for signing, the system still remains vulnerable. Such systems are poorly decentralized. Each server from a given finite set can be compromised or destroyed.

Bitcoin represents the first implementation of digital currency, completely decentralized and solving the problem of group signature of the transaction list, where the number of participants is not known in advance and can vary randomly.

The system of group signatures is implemented in the form of a consensus mechanism based on the proof of work in combination with economic incentives for participants. Economic incentives cause participants to act economically rational, as a result, a consensus is reached.



A group signature is a chain of hash function values from the block headers that contain the hash function values from the previous block header, as well as the Merkle root [2] from the list of transactions written in the block. This chain proves data integrity and protects against the substitution of transaction history.

In case of conflicting transactions (an attempt to spend the same funds twice), the transaction that first enters the blockchain is considered to be valid. In the blockchain branching is possible, both because of the delays in transmitting of information about new blocks over the network, and because of attempts to attack the transaction history in the chain. The suggested consensus algorithm is quite effective against such attacks, it has been proved in practice and now it is the key to success of Bitcoin.

After the Bitcoin rise, this technology has interested many developers, scientists and economists. A lot of research, development and experiments were carried out. The world of cryptocurrency has grown rapidly

and at the moment there are more than 240 cryptocurrencies with the capitalization of more than one million dollars.

## 1.1 Blockchain Consensus Models

A lot of research was done to find an alternative version of the consensus algorithm proposed by Satoshi Nakamoto and based on the proof of the work performed. The proof of work algorithm represents finding the hash-value of a given complexity, by brute force (mining), which requires a large amount of processing power. There is always an arm race among the participants of the system who took part in mining. The one who has more processing power - wins. In 2017 the electricity costs for the implementation of bitcoin mining are comparable to the electricity consumption of small countries. Consumption of such a huge amount of resources poses the task of finding a nicer and more energy-saving cryptographic solution. The most popular models of algorithms are considered below.

### 1.1.1 Proof of Work

The basis of the Bitcoin Consensus algorithm is the *Proof-of-Work* (PoW). In order to add a new block to the blockchain, the network participant must provide proof of the done work. The proof of work is the hash function value (32 bytes long) from the header of the new block, which satisfies the current complexity of the network (less than a certain number).

This rule can be represented by the following formula:

$$\text{hash}(B) \leq N / D,$$

where  $B$  is the block header,  $\text{hash}(B) \in \{0, \dots, N\}$ ,  $D$  is the current complexity. The larger the  $D$ -value is, the more iterations it takes to select a  $B$ , so that the inequality would be solved. The expected number of iterations equals  $D$ .

The sha256 algorithm is used as the hashing function in the Bitcoin. In the arm race with processing power, participants develop integrated circuits specialized to calculate the sha256 function - ASIC, which ultimately led to the transformation of the decentralized mining model into an oligopoly. To provide a protection against the creation of ASIC devices, alternative cryptocurrencies began to use other hashing algorithms or several algorithms at the same time. The first of the alternatives is the Scrypt algorithm. It differs from sha256 primarily in requiring in processing a large amounts of RAM. For some time, this feature was a barrier for creating ASIC devices, but this barrier was overcome and the situation took place again. This algorithm is used in Litecoin, Dogecoin, Digitalcoin. In further modifications, more complex algorithms such as X11, Ethash, Quark and others began to be used.

### 1.1.2 Proof of Stake

In the Proof-of-Stake (PoS) algorithm the right to post the next block falls randomly in accordance with the ownership percentage of shares in the system. The simple version can be expressed by the formula:

$$\text{hash}(B_n, A, T) \leq F(A) N / D,$$

where  $B_h$  is the hash value from the last block header in the blockchain,  $A$  is the address,  $T$  is the current time,  $F(A)$  is the address balance, in some implementations, along with the balance, the age of the coins is also taken into account, hash  $(B_h, A, T) \in \{0, \dots, N\}$ , and  $D$  is the current complexity.

Unlike the Proof of Work protocol, in the simplest implementation this algorithm has a number of significant disadvantages and cannot be considered safe. One of the main disadvantages is the problem of "nothing at stake." Its essence lies in the fact that the cost of creating blocks for participants equals to zero, unlike PoW, where it is impossible to create a block with no processing or energy costs. Since participants do not bear any costs creating a block, in the case of blockchain branching, rational behavior is the creation of blocks in all known chains or the creation of a new chain. If the malicious participant attempts to attack the blockchain branching, the attack will cost nothing (zero). In the absence of economic incentives to keep on only one blockchain, and in absence of punishment for creating blocks in alternative chains, the system will not be able to come to a consensus.

Various modifications have been developed to eliminate the disadvantages of simple PoS, in particular, *Delegated-Proof-of-Stake* (DPoS). In this modification, the blocks are created by delegates (certain users of the system). The duties of delegates include voting for blocks and reaching consensus. Each block should receive at least 1 vote from the delegate. Delegates are punished for malicious behavior. In some versions of the algorithm, the delegate is obliged to stand bail, and in case of violation of the rules he is fined or he loses the bail.

### 1.1.3 Byzantine Agreement Protocols

The Byzantine Agreement protocols are a class of algorithms based on a number of authorized participants who are allowed to create new blocks. Participants make decisions based on voting in the solution algorithm for the problem of *Byzantine fault tolerance* (BFT).

An example of such protocols is the HyperLedger Fabric blockchain platform, developed by the Linux Foundation [3]. This blockchain has a flexible architecture with plug-in modules of consensus models. Participants pose as consortium, where identity of each person is registered and verified on a centralized participant registration server. Currently, two consensus models are supported: *Practical Byzantine Fault Tolerance* (PBFT) and *SIEVE*.

These protocols have a weak decentralization and are mainly used as private blockchain solutions. An example of a public blockchain in this category are Ripple and Stellar.

## 1.2 Blockchain Architecture

All available cryptocurrencies can be classified according to their blockchain architecture. Now there are two types: *Account based* and *Unspent Transaction Outputs* (UTXO).

### 1.2.1 Account based

Account based architecture is a simple and intuitive approach. Reading the data sequentially from the blockchain, starting with the genesis of the block, it is possible to get the current state of the system. The

state of the entire system is a list of accounts with the corresponding balances, and the state of the accounts as well.

Advantages:

- Small transaction size
- Easy to understand
- Great flexibility in working with smart contracts that use stateful scripting language

Disadvantages:

- No parallelism. Each transaction must have a nonce, which is necessary to prevent a repetition attack.

This approach assumes that each account (object) has its own state, which changes as a result of the receipt of new transactions. Account balance is one of the properties which can be changed. More complex object models that store variable user data, data arrays, and built-in functions that can affect an object and send transactions to other objects are used executing smart contracts.

### 1.2.2 Unspent Transaction Output (UTXO) based

UTXO architecture is an approach based on transaction chains, where each transaction has one or more inputs of previous transactions and creates one or more new outputs of the specified denomination (coin). This approach was implemented in Bitcoin.

The Bitcoin blockchains are based on interconnected transaction chains. Each transaction on the input consumes the outputs (coins) of the previous transaction, and creates new, unpatched coins at the output. Transactions are packaged in blocks, which ensures their invariance and protects against the possibility of spending the same coins twice. The user's balance is a set of non-spent outputs (coins) to which he has access. Each output is an immutable object that can only be spent (destroyed).

Advantages:

- Unmodified objects / records in the blockchain provide simple and understandable analysis
- Transaction parallelism
- Simpler requirements for consensus algorithms in conflicting transactions

Disadvantages:

- The transaction size is not fixed and in most cases is bigger than in the Account based solution (larger blockchain size)
- No flexibility necessary to implement complex smart contracts dealing with states.

This approach has is better for solutions related to payment systems and solutions of electronic cash systems.

## 1.3 Assets

One of the most practical applications of cryptocurrency is the functioning as a value transfer system. Data on transactions which redistribute assets (coins) contains in the blockchain. The ability to create own assets and currencies based on blockchain technology immediately aroused great interest, which led to a rapid increase of new cryptocurrencies based on Bitcoin developments. However, the creation and maintenance of a new blockchain requires the maintenance of a team of highly skilled developers and significant resources to ensure the proper level of system security. In other words, the process of creating a separate asset blockchain is expensive. In this regard, the use of already existing public blockchains to realize the possibility of issuing digital assets will be an economically viable solution. One of the first projects implemented this idea is the Colored Coins. In this project, a meta-protocol of currencies / assets embedded in the Bitcoin transaction was implemented. The project provided an opportunity for users to issue their digital assets, transfer and exchange them within Bitcoin's blockchain.

Later separate blockchains appeared, which purpose was to create platforms for the emission and circulation of user assets. One of the first such platforms is Nxt blockchain, launched in late 2013. In 2015 the Ethereum platform [4] was launched. Its main idea is to create a system of smart contracts, where the turing-full scripting language is used, which allows to operate states of the blockchain objects. To implement the task of creating user assets, a standardized smart contract ERC20 [5] was developed, which ensures the functionality of issuing and transferring assets. The Ethereum has gained the greatest popularity among users. At the moment there are about 10 similar platforms.

## 2 Bitcoin Asset sidechain

### 2.1 Abstract

Bitcoin Asset is a sidechain asset platform related to Bitcoin's blockchain. This platform is built on the architecture of UTXO and uses the consensus Proof-of-Stake algorithm.

### 2.2 Motivation

The existing blockchain platforms have only one asset at the root. For example, in the Bitcoin blockchain the only asset is the Bitcoin itself. Platforms with creation of additional assets, are constructed so that the work with created assets is carried out through transactions of the initial asset. All operations with assets are recorded as embedded information in transaction. This approach complicates the application of assets. Let's consider some examples.

For example, consider the most popular Ethereum platform. Suppose a user has created an asset within the Ethereum platform. The intended purpose of the asset is to use it for a payment means.

- Users of the created payment system, having received assets to their addresses, face the fact that they cannot make an outgoing transaction until their address does not have enough Ethereum currency in addition to the received asset. To accomplish the operation, it is necessary to top up your address additionally with Ethereum currency.
- In order to arrange acceptance of payments of any asset, it is necessary to create unique address, which replenishment would mean the replenishment made by the customer or the balance replenishment of his account. The store generates many addresses for receiving payments from customers. After the payment is received to the created address, in order to transfer the received

assets to a certain general account of the store, each address which received payment must be replenished with Ethereum currency.

- The process of address replenishing with Ethereum currency means the payment of the for the replenishment transaction. On further asset transfer, a commission for the transfer transaction is paid. In case of receiving payments for a several one-time addresses, the total payment processing commission is doubled.
- Since a complex smart contract is executed, the commission for the assets transfer is higher than the commission for a normal native transaction, and it is necessary to pay a commission for each instruction in the contract code. If it is not sufficient funds are to pay for the execution of the smart contract instructions, but it is enough for carrying out the native Ethereum transaction, the transaction will be placed to blockchain, the commissions will be paid, but the assets transfer will not be made. The assessment of the commission in the course of the contract is not a trivial task.

With this approach to work with assets there is a constant need for a platform currency to pay commissions. The constant demand for an initial asset is a stimulus to price increase. We believe that this was a key goal designing a system. In our vision, this approach is not focused on solving applied problems for working with assets. A further evolution of the asset systems of the blockchain is a direct processing of all assets without introducing an additional level of complexity by embedding assets in the transaction.

In designing of a new blockchain, it is important to pay attention to the architecture of existing platforms. The disadvantages of using of the Account based architecture are:

- No opportunity to consolidate the balances of multiple addresses within a single transaction.
- No opportunity to make mass payments to multiple users within the same transaction.
- No transaction parallelism.

In our vision, a more appropriate approach is to use UTXO architecture as more flexible and scalable in its design. Next, we describe the key elements of the Bitcoin Asset blockchain (consensus algorithm, blockchain architecture, scripting language) and give examples of the basic assets operations.

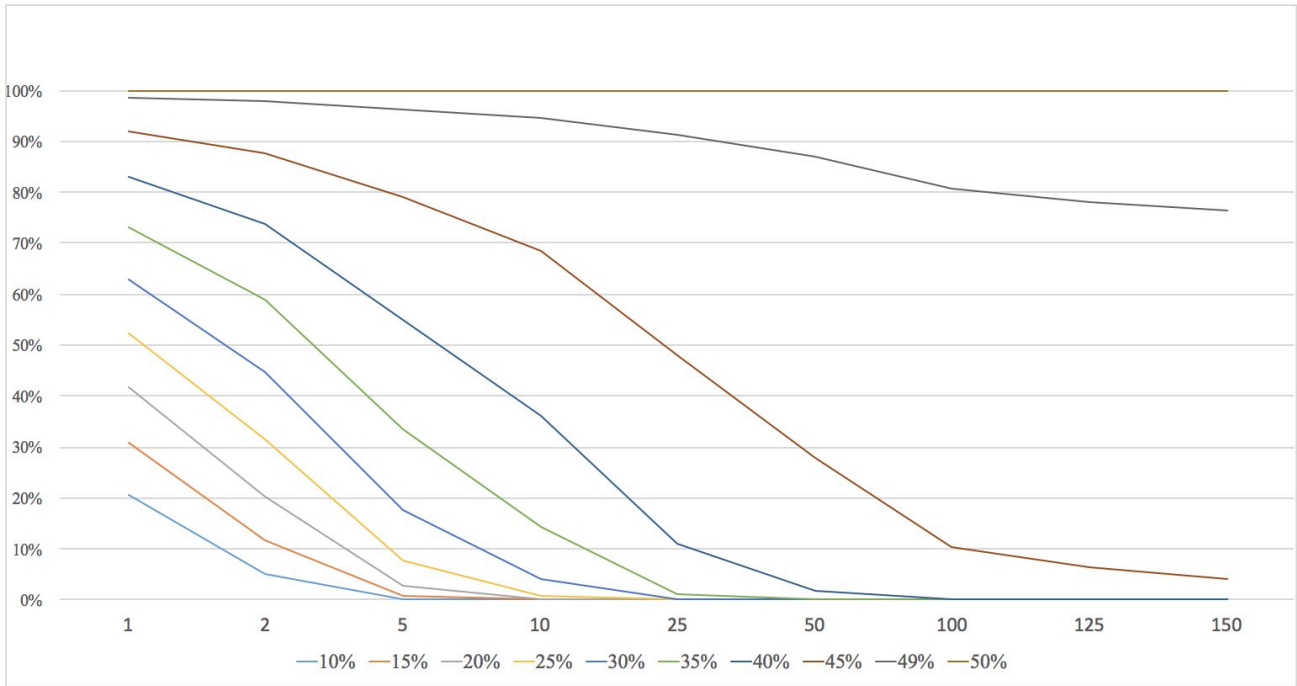
## 2.3 Consensus algorithm

As a rationale for our choice, the use of the Proof-of-Stake consensus algorithm for the Bitcoin Asset blockchain, we present two significant shortcomings of the Proof-of-Work algorithm.

PoW is extremely expensive in terms of electricity consumption. Currently, the amount of electricity consumed only for Bitcoin mining exceeds the energy consumption of small countries. Further development of the blockchain technology in this direction will lead to an energy insanity.

The use of PoW seriously increases the risks of security breach when starting a new blockchain, as well as when working with an existing one, but with a small number of participants (low hash rate), because at any time an attack, which would be carried out successfully, may occur on the blockchain.

It is all about the fact that the amount of processing power in the users' turnover is significant, the distribution of available power is extremely uneven and any owner of processing power will be able to carry out a successful attack in case if the concentrated share of processing power exceeds the critical level for a new blockchain.



Given diagram shows the probability of creating by an attacker a new chain of length N blocks. If we assume that the attacker has 35% of the processing power, then, according to calculations, the attack will be successful with a probability of 40%, even after 4-5 blocks. The total processing power of the new blockchain is not high relative to the total processing power in the turnover of miners. Due to the fact that the decentralization of PoW Mining is low and the current situation can be characterized as a developed oligopoly, there are a large number of participants with a share of processing power, who allow to conduct deliberately successful attacks on the new blockchain. Therefore, the use of PoW cannot be considered safe until the complexity of processing becomes parity to operating blockchains, which is serious security risk at the initial stage.

Bitcoin Asset sidechain uses the Proof-of-Stake algorithm as an alternative to PoW. Only the initial asset, which is created and distributed at launch of the blockchain, participates in the consensus algorithm. With a good initial assets distribution, a high level of decentralization and the safety of the consensus algorithm is achieved. At attempt to buy a significant share, the natural reaction of the market will be an increase of the asset price, which is a protective barrier, increasing the cost of the attack. As a way of initial assets distribution, pre-sale of the initial asset will be conducted. The cost of the initial asset should be initially high enough so that an attempt to buy a significant share would be very expensive. An additional argument in favor of using PoS is that, given the high competition of the transaction commission market, the cost of participation in the blocks creation tends to the cost of resources that are spent on transactions verification and blocks creation. In the PoS algorithm, the cost of block creation equals near-zero, in accordance, in this system, the cost of commissions will be low, unlike PoW, where the cost of block creation is significant.

## 2.4 Logical Clock Proof of Stake algorithm

A simple PoS algorithm cannot be considered safe, as it is vulnerable. Several modifications of the algorithm to remove all disadvantages were suggested. We conducted analysis of posted implementations: Nxt, Slasher, Bitshares, Snow White, Ouroboros. According to the results of the analysis and the current vision we have developed our own implementation of the algorithm.

The consensus algorithm must meet next key requirements:

- The cost of the attack on blockchain should not be equal zero.
- It is impossible to increase the chances of a participant to create a block by means of pre-calculation.
- The participant must be punished for non-fulfillment or violation of the protocol rules.
- Punishment for violation of the protocol must be inevitable, both in the chain of the attacker and in the legitimate chain.
- The choice of the sequence of the participant creating the block must be random, unambiguous and proportional to the share of the participant.
- The newly connected node must independently come to the current state, guided only by protocol rules.
- The protocol must be resistant to attacks with an attacker's share of up to 45%

#### 2.4.1 Brief definition

As a key concept of the algorithm for simplification of understanding it is possible to adopt a model as a game, which results are recorded. Participants of the game make security deposits before the start. After formulation of the participants list, each player makes a bet, guessing a random number. At the end of the bet, using the perfect random number generator, the winner is determined. The winner is obliged to create a block and receive a bonus and record the chronology of the game. If the winner does not create a block, he is fined.

A participant who did not create a block in his turn has a right to create alternative records of the game's chronology by writing his blocks into a parallel chronology of the results. Not in his turn, instead of blocks, he would have to record all the omissions. The chronology of the game, where the number of passes is the smallest, is considered correct. If there are a fair majority of participants, then honest players will never be able to get their results to be recognized as true.

#### 2.4.2 User Deposit

To ensure the set requirements, participants of the new blocks creation are required to make at least a *MIN (the minimum amount of qualification)* deposit. The deposit allows to implement the mechanism of punishing the participant for violating the protocol rules.

Made a deposit participants form a set:

$$U \in \{U_0, \dots, U_N\}$$

, where N is the number of participants. The distribution of the probability of obtaining a turn for the block creation depends on the amount of the deposit made.

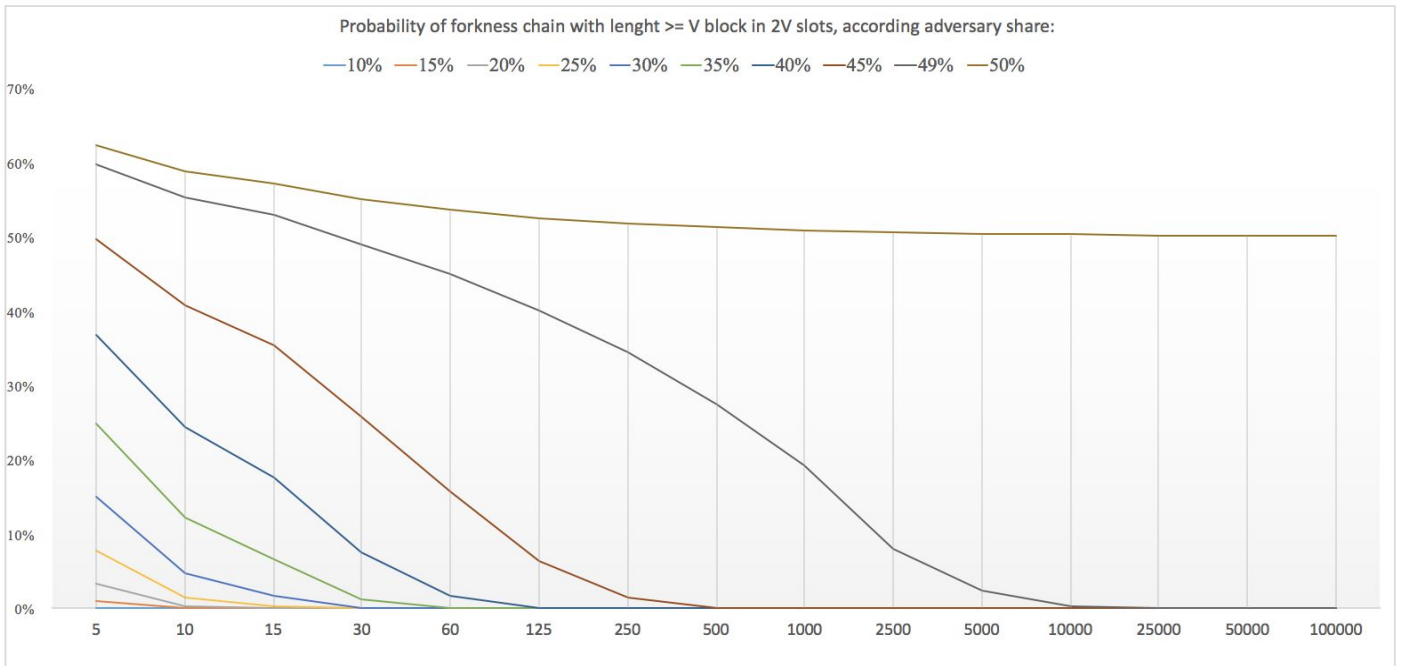
### 2.4.3 Time slot and epoch

A slot is the time for block creation. The slot time is the time needed to expand the message about the new block within the distributed network. In accordance with the tests carried out, the slot time is predetermined to 20 seconds. For participants a slot can be considered a game round.

An epoch is a time interval equal to the finite number of Z-slots. The number of Z-slots is determined by the possible value to obtain the majority of blocks for the attacker in a given number of slots subject to balanced distribution of the probabilities of block creation in accordance with the shares of the participants.

To determine the number of Z-slots we use the Bernoulli formula, since we are dealing with a binomial probability distribution [6]. Assume that the attacker has Q% of the total deposit in the epoch, but less than the majority. We denote that V is a sufficient number of blocks to prevent an attack. The number of slots we take equal to 2V according to the fact that the chain of the attacker is confronted by a chain of honest participants. We calculate the lengths of blockchains, sufficient to prevent an attack.

Adversary share	Blocks														
	5	10	15	30	60	125	250	500	1000	2500	5000	10000	25000	50000	100000
10%	0,16%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%
15%	0,99%	0,02%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%
20%	3,28%	0,26%	0,02%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%
25%	7,81%	1,39%	0,27%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%
30%	15,03%	4,80%	1,69%	0,09%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%
35%	24,85%	12,18%	6,52%	1,19%	0,05%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%
40%	36,69%	24,47%	17,54%	7,46%	1,67%	0,09%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%
45%	49,56%	40,86%	35,52%	25,76%	15,64%	6,38%	1,40%	0,08%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%
49%	59,82%	55,25%	52,85%	48,94%	44,89%	40,00%	34,36%	27,39%	19,15%	8,07%	2,33%	0,24%	0,00%	0,00%	0,00%
50%	62,30%	58,81%	57,22%	55,13%	53,63%	52,52%	51,78%	51,26%	50,89%	50,56%	50,40%	50,28%	50,18%	50,13%	50,09%



As we can see from the table of calculations and diagram above, with a 45% share of the attacking participant, the length of the chain to prevent an attack must be 500 blocks. According with the fact that the attacker is opposed by a legitimate blockchain, the length of the epoch must be  $Z = 1000$  slots.

Based on these calculations, we can determine the safe number of blocks necessary to prevent double-cost attacks, depending on the proportion of the attacker.

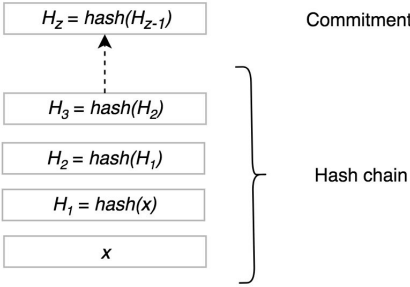
Probability of forkness chain 0,1%				
Adversary share	Blocks	Slots, max	Min, min	Min, max
10%	6	12	2,0	4,0
15%	8	16	2,7	5,3
20%	12	24	4,0	8,0
25%	18	36	6,0	12,0
30%	30	60	10,0	20,0
35%	54	108	18,0	36,0
40%	122	244	40,7	81,3

If the shares of participants are distributed in such a way that the share of each participant individually does not exceed 10%, then it is safe to accept payments with an acknowledgment awaiting for about 2-4 minutes - 6 blocks.

#### 2.4.4 Source of random and bets

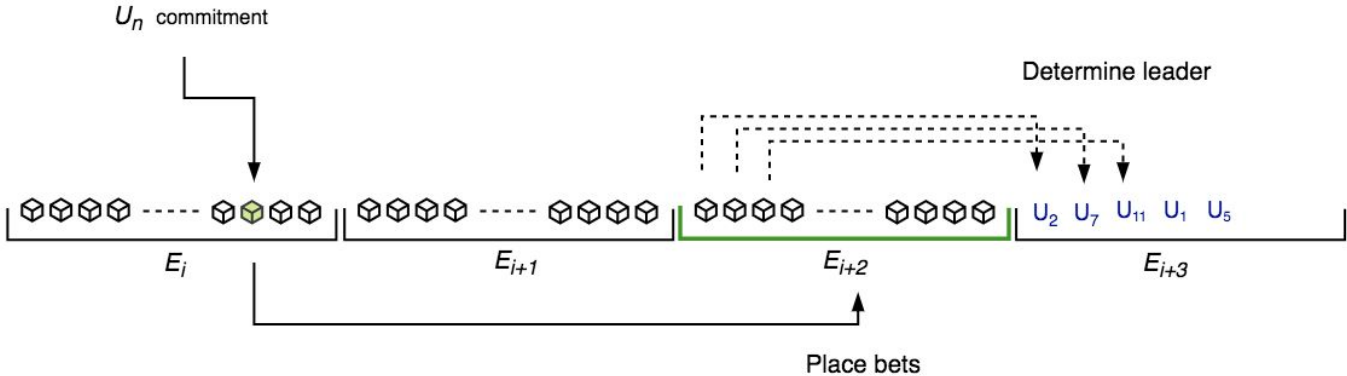
One of the important elements of the protocol is the randomness of the choice of the participant, which gets the right to create the next block. The source of randomness, in our implementation, is a chain of hash values of participant functions obtained from the random number chosen by each participant.

After making a deposit, participants should select a random X value and create a chain of hash functions values equal to the number of slots in the epoch. When the block is posted, each value in the chain will be the share of the participant.



In accordance with the properties of the hash-coding function, each value of  $H_i$  is random and there are no regularities that allow to predict the values in a given chain. In addition, the hash-coding function is a one-way function, which means that it is not possible knowing  $H_{i-1}$  to compute  $H_i$ .

The last  $H_z$  value is posted in the *Commitment* to document the evidence that the participant will not change the values in the chain in the future. In this case, you can draw an analogy of how a participant makes bets in a roulette before starting a spin.



Posted evidence in the current  $E_i$  epoch means that values can be used across a single epoch within the  $E_{i+2}$  epoch. Bet values, posted together with created block, determine the sequence of blocks creating in the next  $E_{i+3}$  epoch. In the current  $E_{i+2}$  epoch, the sequence is already predetermined by the past epoch and cannot be changed, which allows to create a mechanism for punishing of the participants for failing a block creation. Even if the attacker attempts to create an alternative reality with an alternative chain, within the current epoch, he cannot escape punishment, since he can only affect the sequence of the next epoch.

2.4.5 Leader selection

Creating a block, the participant posts a random  $H_{z-k}$  number, where  $z-k$  is the index of the last unposted value from the participant's random number chain. The rest of the participants can be sure that the posted number has not been changed from the moment of posting of the  $H_z$  certificate.

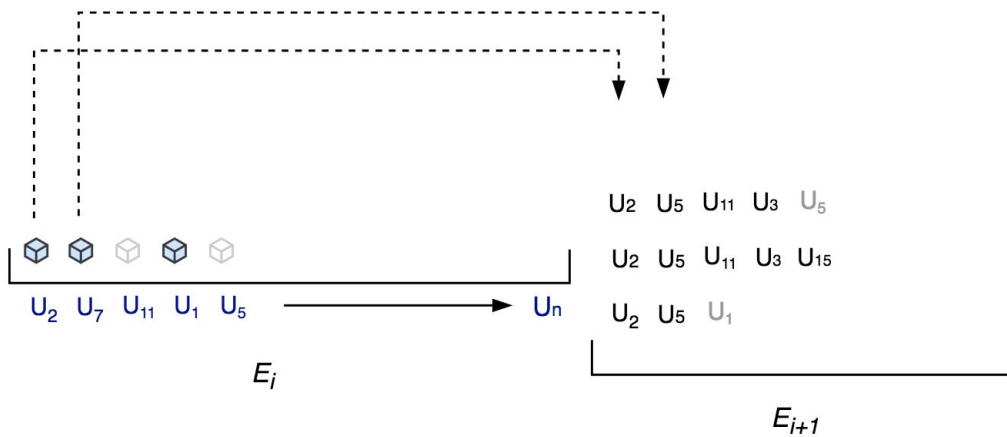
$$\text{hash}^k(H_{z-k}) = H_z$$

This random number is used to determine the participant who creates the block in the next epoch, i.e. through  $Z$  slots. The choice of participant is a kind of lottery, where the chances are distributed in accordance with the deposited shares. The right to post the next block in the slot falls to the only one participant for whom the value of this set is maximum:

$$F(U_n) M - \text{hash}(H, U_n, \text{nonce}),$$

where  $U_n$  is the address of the participant,  $F(U_n)$  is the share value of the participant's deposit,  $\text{hash}(H, U_n, \text{nonce}) \in \{0, \dots, M\}$ ,  $H$  is the value of the random number posted in the block in the previous epoch  $Z$  slots back,  $\text{nonce}$  is a non-negative integer by default equal to 0. If it is impossible to identify the participant with the maximum value, incrementing  $\text{nonce}$  to 1 to the nonce and calculate the winner again.

In the current  $E_i$  epoch, the sequence of creating blocks is already distributed and all participants know it, during the slot time the participant must post the block. If the participant does not post the block, then according to the protocol rules, an empty block is formed. In each block the  $H$ -value of the participant, which is kept secret until posting, is posted. If the block was not posted, the participant's turn is determined in accordance to the last posted random number, with 1 being added to the currently used  $\text{nonce}$ .



Since blocks may not be posted immediately for technical reasons or because of an attack, the set of blocks with the smallest number of empty ones is considered valid. At the end of the current epoch, the sequence of the next epoch stabilizes and can no longer be changed.

## 2.4.6 Economic Initiatives

Proposed implementation of the consensus algorithm is based on the following economic initiatives:

- The block was posted by the participant. Through  $n$  epochs from the current, participant's deposit increases according to the reward for the block.
- The block was not posted. If the block has not been posted, then the participant loses some of his deposits in the amount of the award for the block (by analogy with the economic initiatives in the PoW, as if the miner has purchased equipment, found the block, but did not post it, as a result suffering losses). Within the current epoch, the sequence of blocks posting is strictly fixed and

predetermined, in this regard, the absence of the block cannot be hidden from the protocol.

- A participant posted 2 blocks during his turn. In the case of the posting of several blocks with the same value of a random H-number, any other participant may receive a part of the infringer's deposit and disqualify it by posting a transaction indicating a gross violation of the protocol.

These economic initiatives are inevitable and imitate the cost of a block creating, so they solve the Nothing-at-Stake problem and a number of other attacks. Acting according to the protocol is rational behavior of participants.

### 2.4.7 Long range attack

One of the weak points of the PoS protocols is weak subjectivity. If the attacker has created the chain deep in the past, the new node, interpreting recordings of the blockchain, cannot independently determine which chain is legitimate.

To stop this type of attack, we will additionally introduce the concept of the weight of an epoch. The weight of an epoch is the share of participants' deposits of creating blocks in a given epoch, relative to the total number of assets that are currently issued. To determine the legitimacy of the chain, it is required to take and compare the total weight of  $N$  number of epochs. The chain, where the total weight is greater, is legitimate. Since the attacker cannot reach the majority for a long period of time, this strategy is correct.

### 2.4.8 Precomputation attack

The precomputation attack is a kind of attack when an attacker can affect the result of the future sequence distribution to his benefit with the help of calculations. In connection with the fact that the chains of random values are fixed by participants in the past, the precomputation attack is impossible.

### 2.4.9 Stake Leasing

In many of the existing implementations of the PoS algorithm that we have studied, we have realized the possibilities of delegating or renting a share of the participant for the minting. In our vision, this mechanism leads to the centralization of the system. Since in the process of building a decentralized system, laying the mechanisms leading to centralization contradicts the very idea of decentralization, we refused to implement such ideas. The possibility of delegating or renting a share of a participant in the long term will lead to the formation of oligopolies, which we can observe among Bitcoin miners.

## 2.5 Blockchain architecture

According to the description of Account based and UTXO based architectures, we believe that the lack of parallelism in the Account based architectures transactions is a significant drawback, which will primarily affect the scalability of the system. The UTXO approach also has advantage of mass payments within a single transaction, consolidation of outputs within a single transaction, creation of chains of interrelated transactions. Using UTXO based architecture is better for building an asset system.

## 2.6 Scripting language

Bitcoin Asset sidechain as the basis of the scripting language inherits Bitcoin Script with the introduction of a number of changes:

- the signature algorithm is changed to Schnorr signature [7]
- signatures and scripts for unlocking the outputs are transmitted according to the Segregated Witness approach [8], the Merkle root from witness hashes is recorded in block header.

$$txid = SHA256 ( [nVersion] [[inputPoint] [sequence]] [txouts] [nLockTime] )$$
$$wtxid = SHA256 ( [nVersion] [[inputPoint][signatureScript][sequence]] [txouts] [nLockTime] )$$

- the MAST analogue (Merkelized Abstract Syntax Tree) was implemented [9]
- additional opcodes to get data on the inputs and outputs of the transaction in the context of which the script is executed were added
- the opcodes that allow to obtain information about transactions in Bitcoin Asset sidechain and Bitcoin blockchain: `OP_OUTDEPTH`, `OP_OUTAMOUNT`, `OP_INDEPTH`, `OP_INTX` were added
- additional opcodes for creating and issuing assets were added

### 2.6.1 Interaction with Bitcoin blockchain

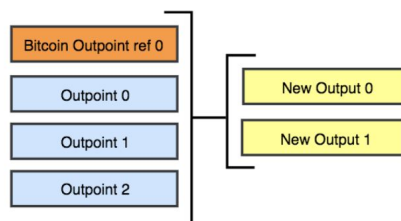
Links to the outputs from Bitcoin blockchain are used to interact with the Bitcoin, which increases the scope of the transaction scripts.

Normal transaction:

```
[Version = 1 ][inputs][outputs][nLockTime]
```

Transaction with a link to Bitcoin blockchain:

```
[Version = 2 ][bitcoinOutputReferences][inputs][outputs][nLockTime]
```

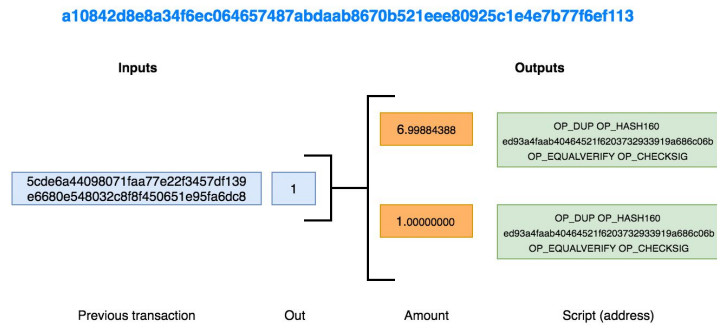


Added opcodes `OP_OUTAMOUNT`, `OP_OUTDEPTH`, `OP_INDEPTH` inside the `signatureScript` return information on output sums and the number of confirmations of the output / input in Bitcoin Asset

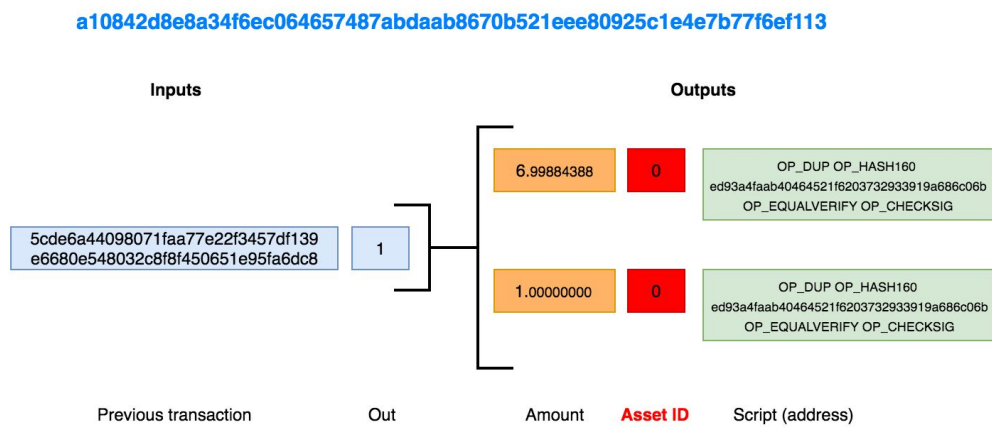
sidechain, as well as the Bitcoin blockchain in expanding of the scope of the transaction with references to Bitcoin. The opcode *OP\_INTX* returns a hash of the transaction where the data output is the input.

## 2.7 Native Assets

The key feature of the Bitcoin Asset sidechain is the use of assets in a natural way. If you look at the Bitcoin blockchain, the transaction is:



In this structure, it is implied that the output sum is the sum of the Bitcoin asset. Adding to the output a field denoting the type of asset in the blockchain, it becomes possible to operate the various assets in a natural way without any additional complicated logic.



Each asset is assigned a unique *ID* number in the blockchain, which identifies it in future.

*Identifier: 1, Name: Bitcoin, Designation: BTC is reserved for Bitcoin, as at the moment there is no implementation of a full decentralized solution for two-way connection with Bitcoin blockchain.*

The modification of this implementation in the transaction structure provides a solution of the disadvantages associated with the binding payment of commissions as the capital asset of the platform. This transaction format allows to pay commission in those assets that are used inside the transaction. If it is necessary to pay for any other asset, the flexibility of the UTXO model allows to deposit an unused coin of the required asset, thereby expanding the set of assets used within the transaction.

## 2.7.1 Create Asset

Creation of an asset in a blockchain is carried out by forming the parameters of this asset and creating a register transaction. As soon as the registering transaction enters the blockchain - the asset is initialized.

Initializing parameters of the created asset:

- *name* - the name of the asset (unique)
- *symbol* - the designation
- *id* - the unique identifier in the blockchain
- *maxTotalSupply* - the maximum possible volume of the asset issue
- *decimals* - the number of symbols after the point
- *mainChainReward* - the formula for the award at the minting
- *selling* - selling assets (crowdsale)
  - *rateList* - list of accepted assets with the exchange rate and specifying the time interval
  - *bitcoinAddress* - the address in the Bitcoin blockchain (optional)
- *initIssue* - the amount of assets distributed at initialization

Technically it is implemented with the OP\_ASSET opcode. To create an asset, it is necessary to create a P2SH address with the following redeem script:

```
<Asset data> OP_ASSET
OP_NOTIF
  OP_DUP OP_HASH160 <20> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

The OP\_ASSET opcode uses 2 parameters from the stack on the input. *SignatureScript* verifying the ownership of the address from which the asset was purchased (more in section 2.7.) and initializing the asset parameters.

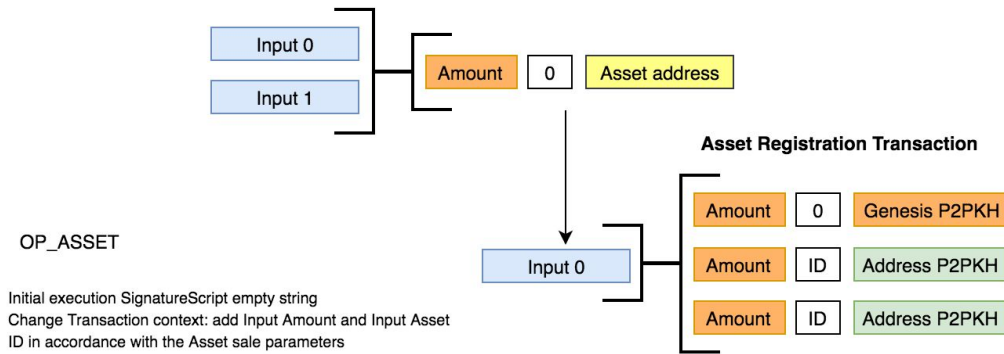
*SignatureScript* can be <sig> <pubKey> to unlock P2PKH and ...signatures... {serialized script} to unlock P2SH.

The result of execution is returned to the stack 0 or 1, and the transaction context (amount and type of asset at the input) is changed based on the asset parameters. To register the created asset in the blockchain from this address, it is necessary to make a payment to the *genesis* address. *Genesis* address is a conditional 20-byte address, where all the 20 bytes are zeros

## 2.7.2 Initial Asset distribution

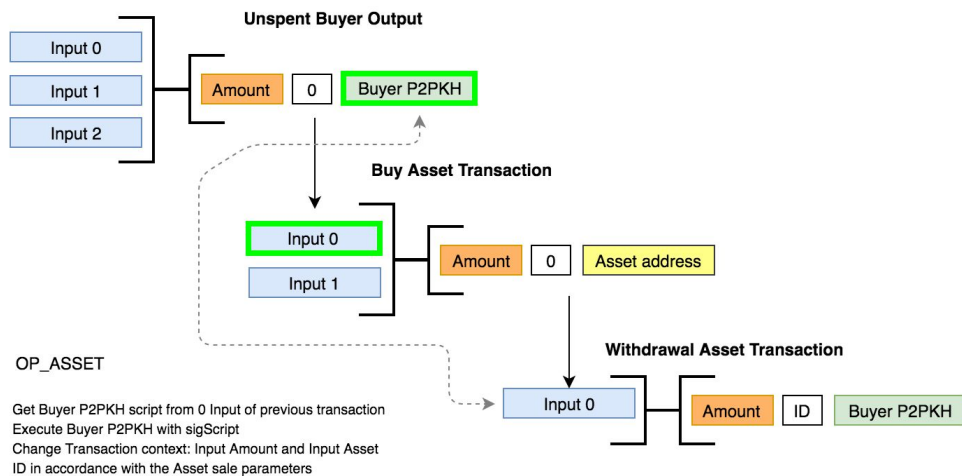
In the transaction registering the asset in the blockchain, it is possible to specify a list of addresses and amounts with the initial distribution of assets to a given set of addresses.

Initial distribution during registration:



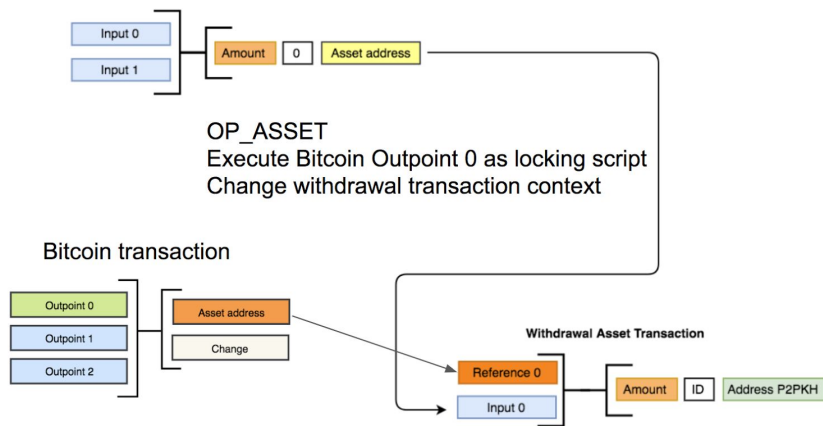
### 2.7.3 Buy Assets (Crowdsale)

Once the asset was registered and the parameter was set, that the asset can be sold, then according to the terms of the sale, users can buy assets. To buy assets users have to make a payment to the address of the created asset. To transfer the purchased asset to your address, it is necessary to create a transaction with an input pointing to this payment. The creator of the asset, in his turn, can receive paid amount to his address.



### 2.7.4 Buy Assets from Bitcoin blockchain

To buy assets using Bitcoin, it is necessary to transfer bitcoins to the address in the Bitcoin blockchain, which was registered as a Bitcoin address to purchase assets when creating an asset in the Bitcoin Asset sidechain. After attaining confirmation for bitcoin transaction, assets becomes available within the Bitcoin Asset sidechain.



A reference to the payment transaction in the Bitcoin blockchain is used to transfer the received assets. In this case, the OP\_ASSET opcode by reference receives the transaction script from transaction input 0, the outputs of which were used to pay for the assets in the Bitcoin blockchain. The fact that the reference transaction received a sufficient number of confirmations in the Bitcoin network, and that the address of the Bitcoin recipient corresponds to the one prescribed in the configuration of the asset, is verified.

After execution of this script, together with *signatureScript* (in ECDSA sek256k1 signature format), the transaction's context is changed: input 0 is ignored, the amount and type of an asset at the input to the transaction are changed to the corresponding values at purchase.

## 2.7.5 Genesis Asset

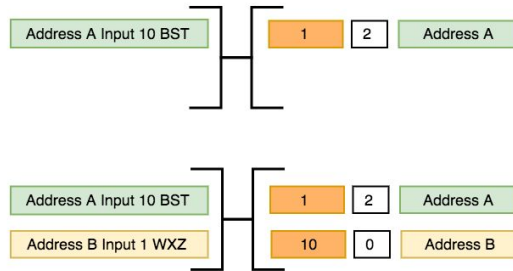
The initial asset is the asset which is created in the first place in the genesis block. This asset is basic and the system of the minting of the blocks is based on it. The participants of the minting receive a reward for creation of the block in the genesis asset.

## 2.7.6 Native Assets exchange

UTXO blockchain structure allows to exchange assets between users in the form of an atomic transaction.

Let's consider this possibility on an example. Alice owns the 10 *BST* asset and wants to exchange it for an *WXZ* asset at the rate of 1 *WXZ* = 10 *BST*. She creates a transaction which input gets an unspent 10 *BST* output and creates an output with address A and a sum of 1 *WXZ* asset. She signs this transaction with her signature, using the signature hash type *SIGHASH\_SINGLE* [10] (only part of the transaction is signed). This type of signature captures the input and corresponding output in the transaction and enables other users to add other inputs / outputs to the transaction.

Bob owns the 1 *WXZ* asset and adds his unreleased output as input, and indicates 10 *BST* as the output.



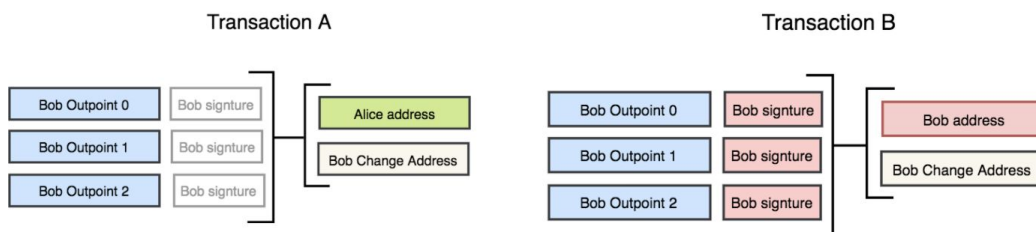
Since the transaction is atomic, Alice and Bob do not need to trust each other. As soon as the exchange transaction enters blockchain, the exchange participants receive desired assets at the same time.

### 2.7.7 Exchange Assets with Bitcoin

Bitcoin Asset sidechain has a one-way connection with the Bitcoin blockchain, which also allows to exchange assets between two blockchains using an atomic exchange which does not require the confidence of the participants. Let's consider the situation when Bob wants to exchange 10 Bitcoin (BTC) for 50 Genesis Bitcoin Asset (BST) from Alice.

To make the exchange, Bob uses unspent outputs Transaction X: 0, Transaction X: 1 and is considered an initiator of the exchange by taking the following steps:

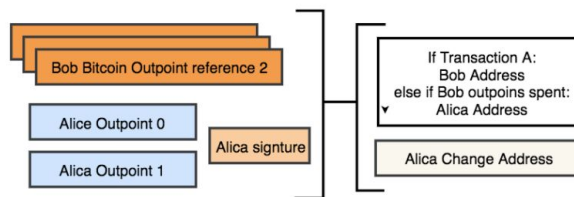
- I. Bob begins the exchange:
  - creates an unsigned Transaction A, which moves 10 BTC to the Alice's address.
  - creates a signed Transaction B that spends the same outputs from Transaction X and moves them to Bob's address.
  - informs Alice about Transaction A, Transaction B.



II. Alice checks the received transactions and takes a return move:

- Creates and sends Transaction C to Bitcoin Asset sidechain, where tokens are blocked and available to either Alice or Bob, depending on which of the transactions in the Bitcoin blockchain occurs first.

### Transaction C



This logic is implemented using the following mutually exclusive scripts:

1. If Transaction A has 6 confirmations, then the asset belongs to Bob:

```
<out Transaction A:0> OP_OUTDEPTH <5> OP_LESSTHAN
OP_IF
  OP_DUP OP_HASH160 <Bob address> OP_EQUALVERIFY OP_CHECKSIG
OP_ENDIF
```

2. If the input Bob outpoint 0 (1,2) is spent, spent transaction has 6 confirmations and if Transaction A does not have confirmations, then the asset belongs to Alice:

```
<Bob outpoint :0> OP_INDEPTH <5> OP_LESSTHAN
OP_IF
  <out Transaction A:0> OP_OUTDEPTH
  OP_NOTIF
    OP_DUP OP_HASH160 <Alisa address> OP_EQUALVERIFY
  OP_CHECKSIG
  OP_ENDIF
OP_ENDIF
```

This script is created for each Bob's coin: *Bob outpoint 0, Bob outpoint 1, Bob outpoint 2.*

- Alice waits for the confirmation of Transaction C in Bitcoin Asset and notifies Bob about this transaction.

III. Bob checks Transaction C, signs Transaction A, and sends it to Bitcoin blockchain.

As a result, Alice and Bob come to the following:

- If Bob stops contacting, Alice will be able to cancel the transaction at any time by publishing Transaction B.
- Bob can cancel the deal at any time by stopping contacting with Alice.
- If Bob sends 10 BTC not to Alice's address, 50 BST will be unlocked in Alice's favor
- If the Transaction A is confirmed, the exchange will be successful.

This type of exchange does not require the confidence between Alice and Bob, and in case of its successful execution, one transaction is published in each blockchain.

## 2.8 Node

The full Bitcoin Asset node stores its own copy of the blockchain, a copy of the Bitcoin blockchain and is involved in the process of creating of new blocks. Light Bitcoin Asset node interacts with other nodes through Simple Payment Verification Protocol - SPV [11]. The implementation of light nodes is possible in following combinations:

- Storage of transactions related solely to the user's wallet
- Storage of Bitcoin Asset sidechain only
- Storage of transactions of the selected asset in Bitcoin Asset sidechain

Implementation of the client / wallet includes the functionality of the wallet for Bitcoin Asset and Bitcoin, which simplifies the interaction between two blockchains. A client configured to work exclusively with one asset or a specific set of assets can be considered as a white label solution for these assets.

## Conclusion

- Bitcoin Asset blockchain platform is designed to create blockchain native assets.
- The architecture of UTXO blockchain and the assets storage within the blockchain as separate and complete transactions provide flexibility and simplicity in building users' deposits.
- Connection with Bitcoin blockchain allows the issuance of assets by paying with a normal Bitcoin transaction.
- Assets exchange within the platform is available as an atomic transaction.
- External assets exchange for bitcoins is available as an atomic transaction, requiring one transaction in each blockchain.

## References

- [1] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008. <https://bitcoin.org/bitcoin.pdf>
- [2] Ralph Merkle. Secrecy, authentication and public key systems/ A certified digital signature. Ph.D. dissertation, Dept. of Electrical Engineering, Stanford University, 1979.  
<http://www.merkle.com/papers/Thesis1979.pdf>
- [3] Hyperledger Whitepaper.  
<https://github.com/bellaj/Blockchain/blob/master/Hyperledger%20Whitepaper.pdf>
- [4] Gavin Wood. Ethereum: A secure decentralized transaction ledger, 2014.  
<https://gavwood.com/paper.pdf>
- [5] Fabian Vogelsteller. ERC: Token standard, 2017. <https://github.com/ethereum/eips/issues/20>
- [6] George P. Wadsworth. Introduction to Probability and Random Variables, New York, 1960.
- [7] Claus P. Schnorr. Efficient Identification and Signatures for Smart Cards. In: Brassard, G. (ed.) CRYPTO '89. LNCS, vol. 435, pp. 239–252, Springer, 1989.  
<https://pdfs.semanticscholar.org/8d69/c06d48b618a090dd19185aea7a13def894a5.pdf>
- [8] BIP: 141, Segregated Witness (Consensus layer):  
<https://github.com/bitcoin/bips/blob/master/bip-0141.mediawiki>
- [9] BIP: 114, Merkelized Abstract Syntax Tree:  
<https://github.com/bitcoin/bips/blob/master/bip-0114.mediawiki>
- [10] Signature hash types: [https://en.bitcoin.it/wiki/OP\\_CHECKSIG](https://en.bitcoin.it/wiki/OP_CHECKSIG)