ICO
Smart Contract
for TON network

Dec, 2019

1.0 Destination.

This smart contract is intended for use in the TON blockchain network and allows you to conduct an Initial Coin Offering (ICO) fundraising event for various projects. The collection is carried out in the cryptocurrency Gram. If the collection was successful and the minimum amount for the project was collected, tokens will be transferred to users' wallets. If the fundraising failed, users will be able to get their funds back.

2.0 Key features of a smart contract:

2.1 SoftCup.

Smart contract allows you to set the minimum amount of funds required for the implementation of the project. If softcup is not collected, users will be able to return their funds back.

2.2. Accrual of bounty steaks on users' wallets with their subsequent exchange for tokens.

The initiator of the ICO can take the help of enthusiasts before the start of the ICO, and in order to somehow interest them, he can offer bounty steaks for a certain job - write an article, post a video, etc.

2.3 Smart contract management.

The smart contract is managed using special commands sent from the owner’s wallet specified in the owner_wallet parameter at the stage of setting up the smart contract. The following commands are implemented:

Command	*Parameters*	*Action*
enroll_bounty	*wallet address, stake*	Credits the specified number of steaks to the user's balance in the smart contract
finalize_ico		Completes the ICO. This command send tokens to users' wallets, incl. bounty steak owners. After that, all collected funds are transferred to the target wallet specified in the dest_wallet parameter when setting up a smart contract.

3.0 Principle of work:

3.1 Setting up a smart contract.

Before uploading a smart contract to the TON network, it must be initialized with the appropriate settings. To do this, use .fif scripts in the "fift" folder. The settings indicate such parameters as the start and end dates of the ICO, the SoftCup size, the address of the target wallet, where all the funds collected will be transferred, as well as the address of the owner of the smart contract. From this wallet, smart contract management actions will be performed, such as enrolling a bounty and finalization an ICO. The ICO finalization procedure transfers tokens to investors' addresses, then transfers all collected funds to the wallet specified in the dest_wallet parameter. Please note that at this point it is desirable that the dest_wallet wallet be initialized. A multi-signature wallet is preferred.

3.2. The basic principle of work.

After uploading the smart contract to the TON network, it waiting the start date of the ICO. Until this point, any incoming transactions will be discarded.
When the start date for the ICO is coming, the smart contract enters the first stage of the sale. Each stage has its own discounts and a limited number of tokens for sale are offered. The closer to the end of the sale, the lower the discount.
Here's what the ICO stage table might look like:

Stage	*Tokens amount*	*Token price in ng*
1	500000	30000
2	500000	40000
3	500000	60000
4	500000	70000

Upon purchase of all tokens allocated for a particular stage, the smart contract automatically proceeds to the next stage. It allow to complete the ICO ahead of schedule.

3.2.1 Behavior of a smart contract between stages.

When the payment amount exceeds the number of tokens available for purchase at this stage, the smart contract can act in two ways, depending on the any_stage parameter transmitted in the message. If any_stage = 1, then the smart contract will go to the next stage and buy tokens for the remaining funds at the price of a new stage. This allow complete the ICO ahead of schedule. If parameter any_stage = 0, then the smart contract will purchase the tokens of this stage only, and unspent funds will be returned to the buyer. By default, any_stage = 1, which means that you can make a simple transfer grams to the address of the smart contract, without

any message. If not the entire amount has been spent, the remaining grams will be returned to the buyer's account.

This smart contract allows the owner to accrue bounty steaks to various wallets. After a successful ICO (SoftCup collected), the corresponding number of tokens will be transferred to these wallets. The number of tokens due for each steak will be known at the end of the ICO, when the total number of tokens sold is known. The total number of bounty tokens is defined as a percentage of the total number of tokens sold and is indicated in the parameters when settings up a smart contract. Note that the bounty_percent parameter is a percentage times 100. That is, if it is necessary to allocate 12.33% for bounty from the number of tokens sold, then in the bounty_percent parameter you need to specify as 1233 when setting up a smart contract.

Upon completion of all stages, or reaching the end date of the ICO, the smart contract goes to the final part.

In the final part, if Soft Cup was collected, then all purchased tokens are sent to investor wallets and the funds collected during the ICO is transferred to the wallet specified in the dest_wallet parameter.

If SoftCup was not collected by the end date of the ICO, then all investors can return their funds back. To do this, they just need to send a small amount of grams, say 1 gram, from the same wallet to a smart contract. A smart contract will return the investor’s funds and this 1 gram, minus gas fees.

4.0 Calling smart contract methods.

A smart contract allows you to call certain methods to get the current status of a smart contract by any user. To do this, enter the runmethod command in the lite-client console program, then specify the address of the smart contract, the name of the method, and, if necessary, parameters. The following methods are supported:

Method	*Parameters*	*Description*
current_stage	-	Returns information about the current stage in the form [number of tokens, price in nanograms] For example, [500000 1500]
start_end_date	-	ICO start and end dates in unixtimestamp format. For example, [1576972800 1579651200]
can_buy	-	Returns -1 if the purchase of tokens is possible and 0 if the ICO is not active.

balance_of	wallet hash	Displays the number of tokens purchased and the amount of funds spent in nanograms. For example, [833333 999999500] In this case, 833333 tokens were purchased, of which 500,000 at the price of 1000 ng, and 333333 at the price of 1500 ng. The buyer transfered 1,000,000,000 ng. The smart contract completed the purchase of tokens for this amount and returned remaining 500 ng back to the investor.
bounty_stake	wallet hash	Displays the number of bounty steaks for given wallet. Upon completion of the ICO, these steaks will be converted to tokens and transferred to the user's wallet.
get_settings	-	Returns the current settings. Parameters: [token_id, [softcup, collected_funds, tokens_sold] [owner_wallet, dest_wallet] bounty_percent] For example, [239 [3000000000 3500000000 2000000] [CS {Cell {00438009fbc3d615ce8c1398a8ab27ed3fe0f847d869768848e878e605dc1627a7b28d90} bits: 0..267; refs: 0..0} CS {Cell {0043800db20d7944cbeebe0a85fb64f664086403063b2b89d9fac26e8a435c28c2ccc9d0} bits: 0..267; refs: 0..0}] 3333]
get_flags	-	Returns the flags can_buy, is_finished, [date_start, date_end]

You can convert the date to unixtimestamp and vice versa, for example, on the site - https://www.unixtimestamp.com
To call the balance_of and bounty_stake methods, you need to specify a wallet hash as a parameter. Use the addr_to_h_addr.fif script in the "fift" folder to get the wallet hash.

5.0 Technical Details:

Upon receipt of payment, this smart contract remembers the address from which the payment was made, the amount and number of tokens purchased. Tokens are transferred to investor wallets at the end of the ICO, if Soft Cup was collected. Tokens are transferred by the owner of the smart contract with a special transaction with the finalize_ico command. This transaction also transfers all the grams collected by the smart contract to the multi-signature wallet specified in the dest_wallet parameter.

For this smart contract to function, a special transaction to masterchain require, which will to register a new cryptocurrency / token (config param 7) with a specific currency_id, create the necessary number of tokens and send them to the address of this smart contract.