**Software Requirements Specification**

**for**

QNTU

**Version 1.0**

**Prepared by LongHG**

**IBL**

**03-15-2018**

**Table of Contents**

[**Introduction**](#_l5dio95tavbm) **4**

[**Purpose**](#_3znysh7) **4**

[**Product Scope**](#_tw4kw6a1n1tl) **4**

[**References**](#_hwlvj3k98pll) **4**

[**Overall Description**](#_8necbvgbvprz) **4**

[**Smart Contract Characteristics And Features**](#_vu16khivetg) **4**

[**User Characteristics And Features**](#_x7zxzu9rgtuj) **9**

[**Operating Environment**](#_sesi6du63n3c) **9**

[**Design and Implementation Constraints**](#_9b7mkzczuves) **9**

[**QNTU Functions Description**](#_tb1wpql2x2kf) **10**

[**transferOwnership(address \_newOwner)**](#_rcaltsg0pyh8) **10**

[**changeTokenInformation(string \_name, string \_symbol)**](#_r7ddobm5h604) **10**

[**transfer(address \_to, uint \_value)**](#_hwhm30ltiboz) **10**

[**transferFrom(address \_from, address \_to, uint \_value)**](#_g2uqunzhc8gs) **11**

[**approve(address \_spender, uint \_value)**](#_cxinlldup4rl) **11**

[**increaseApproval(address \_spender, uint \_addedValue)**](#_e13t99nux891) **12**

[**decreaseApproval(address \_spender, uint \_subtractedValue)**](#_rlayr0y8tkhi) **12**

[**allowance(address \_owner, address \_spender)**](#_vwno6lhuzj7k) **13**

[**freezeTransfer()**](#_4ba8ojytmrp) **13**

[**unfreezeTransfer()**](#_sl6ur9mlsruq) **13**

[**freezeUpgrade()**](#_6xacb0jogbev) **13**

[**unfreezeUpgrade()**](#_keivs1ft3pnm) **14**

[**changeUpgradeMaster(address \_newMaster)**](#_gvozjx49w09a) **14**

[**changeUpgradeAgent(address \_newAgent)**](#_m3y6c7g5xce6) **14**

[**upgrade()**](#_sdhjbygck9dz) **15**

[**forceUpgrade(address[] \_holders)**](#_5ualzcpacv2r) **15**

[**transferToContract(address \_to, uint \_value)**](#_uwpq7a9u1frv) **16**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# **Introduction**

## **Purpose**

*This document is for designing and developing QNTU. A ERC20 standard token smart contract on Ethereum.*

## **Product Scope**

*QNTU is the utility token of the Quanta ecosystem that the various participants will be able to put up as collateral. QNTU forms the basis of an internal econonomy that ensures the security of the network.*

## **References**

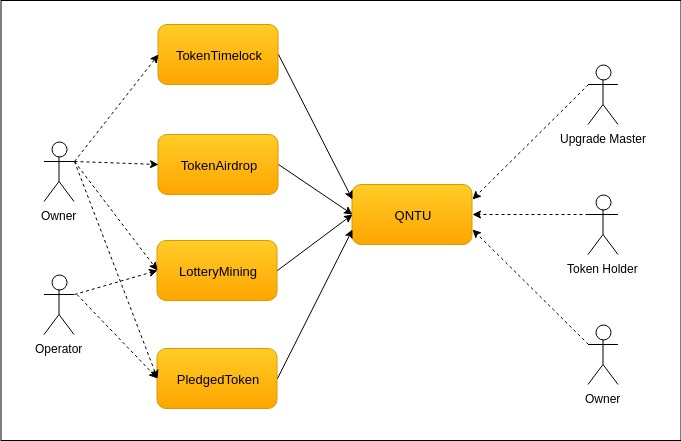
*Quanta White Paper: https://www.quantaplc.im/Quanta.pdf*

# **Overall Description**

## Smart Contract Characteristics And Features

* *QNTU*

*This smart contract is main smart contract in QNTU. It initializes and manages tokens. In addition, it also allow to token holder transfer their token to other person.*



## User Characteristics And Features

* *Owner*

*Owner operate important functions in QNTU smart contracts. Owner is MultiSig smart contract.*

* *Token Holder*

*Token Holder is ETH address or smart contract that hold QNTU token.*

* *Upgrade Master*

*Upgrade Master perform to force token holders upgrade their tokens to new smart contract. Upgrade Master is ETH address.*

* *Operator*

*Operator operate less important functions in QNTU smart contracts. Operator is ETH address.*

## **Operating Environment**

*QNTU runs on Ethereum blockchain platform.*

## **Design and Implementation Constraints**

*Gas cost is important issue in smart contract design.*

# QNTU Functions Description

## transferOwnership(address \_newOwner)

* Description  
  - Allows the current owner to transfer control of the smart contract to a new owner.

* Flow  
  - Change to new owner.  
  - Emit event OwnershipTransferred(address indexed previousOwner, address indexed newOwner).
* Check #1: Sender is not owner  
  - Throw exception.

## changeTokenInformation(string \_name, string \_symbol)

* Description  
  - Change token information.

* Flow  
  - Change full name and symbol of token.  
  - Emit event ChangeTokenInformation(string name, string symbol).
* Check #1: Sender is not owner  
  - Throw exception.

## transfer(address \_to, uint \_value)

* Description  
  - Holder transfer their own tokens to another address.

* Flow  
  - Add to balance of receiver.  
  - Subtract from balance of sender.  
  - Emit event Transfer(address indexed from, address indexed to, uint value).
* Check #1: This function has been frozen  
  - Throw exception.

* Check #2: Address of receiver is 0x0  
  - Throw exception.

* Check #3: \_value is equal 0  
  - Throw exception.

* Check #4: Balance of sender is not enough  
  - Throw exception.

## transferFrom(address \_from, address \_to, uint \_value)

* Description  
  - The person who are authorised can transfer tokens from the authorised address to another address.

* Flow  
  - Add to balance of receiver.  
  - Subtract from balance of the authorised person.  
  - Decrease the amount of tokens that the person who are authorised can transfer.  
  - Emit event Transfer(address indexed from, address indexed to, uint value).
* Check #1: This function has been frozen  
  - Throw exception.

* Check #2: Address of receiver is 0x0  
  - Throw exception.

* Check #3: \_value is equal 0  
  - Throw exception.
* Check #4: Balance of \_from is not enough  
  - Throw exception.
* Check #5: Token amount which want to use is greater than authorised amount  
  - Throw exception.

## approve(address \_spender, uint \_value)

* Description  
  - Token holder authorise another address to own a specific amount of their tokens.

* Flow  
  - Authorise another address to own a specific amount of tokens.  
  - Emit event Approval(address indexed owner, address indexed spender, uint value).
* Check #1: This function has been frozen  
  - Throw exception.

## increaseApproval(address \_spender, uint \_addedValue)

* Description  
  - Token holder increase the amount of tokens that the person who are authorised can transfer.

* Flow  
  - Increase the amount of tokens that the person who are authorised can transfer.  
  - Emit event Approval(address indexed owner, address indexed spender, uint value).
* Check #1: This function has been frozen  
  - Throw exception.
* Check #2: \_addedValue is equal 0  
  - Throw exception.

## decreaseApproval(address \_spender, uint \_subtractedValue)

* Description  
  - Token holder decrease the amount of tokens that the person who are authorised can transfer.

* Flow  
  - Decrease the amount of tokens that the person who are authorised can transfer.  
  - Emit event Approval(address indexed owner, address indexed spender, uint value).
* Check #1: This function has been frozen  
  - Throw exception.
* Check #2: \_subtractedValue is equal 0  
  - Throw exception.

## allowance(address \_owner, address \_spender)

* Description  
  - Get the remaining amount of tokens that the person who are authorised can transfer.

## freezeTransfer()

* Description  
  - Disallow to transfer token from an address to other address.

* Flow  
  - Update isTradable to false.  
  - Emit event FreezeTransfer().
* Check #1: Sender is not owner  
  - Throw exception.

## unfreezeTransfer()

* Description  
  - Allow to transfer token from an address to other address.

* Flow  
  - Update isTradable to true.  
  - Emit event UnfreezeTransfer().
* Check #1: Sender is not owner  
  - Throw exception.

## freezeUpgrade()

* Description  
  - Disallow to upgrade token to new smart contract.

* Flow  
  - Update isUpgradable to false.  
  - Emit event FreezeUpgrade().
* Check #1: Sender is not owner  
  - Throw exception.

## unfreezeUpgrade()

* Description  
  - Allow to upgrade token to new smart contract.

* Flow  
  - Update isUpgradable to true.  
  - Emit event UnfreezeUpgrade().
* Check #1: Sender is not owner  
  - Throw exception.

## changeUpgradeMaster(address \_newMaster)

* Description  
  - Change the upgrade master who perform to upgrade token to new smart contract.

* Flow  
  - Change to new upgrade master.  
  - Emit event ChangeUpgradeMaster(address newMaster).
* Check #1: Sender is not owner  
  - Throw exception.

* Check #2: Address of new upgrade master is 0x0  
  - Throw exception.

## changeUpgradeAgent(address \_newAgent)

* Description  
  - Change the new address of upgrade agent (new QNTU smart contract that tokens will be upgraded to it).

* Flow  
  - Change to new upgrade agent.  
  - Emit event ChangeUpgradeAgent(address newAgent).
* Check #1: Sender is not owner  
  - Throw exception.

* Check #2: Token is upgrading  
  - Throw exception.
* Check #3: \_newAgent is not address of UpgradeAgent smart contract  
  - Throw exception.

## upgrade()

* Description  
  - Token holder upgrade their tokens to a new smart contract.

* Flow  
  - Copy balance of holder from current smart contract to new smart contract.  
  - Reset balance of holder in current smart contract to 0.  
  - Subtract from total supply in current smart contract.  
  - Add to total token upgraded in current smart contract.  
  - Emit event Upgrade(address indexed from, address indexed to, uint value).
* Check #1: QNTU smart contract is not allowed to upgrade  
  - Throw exception.

* Check #2: Balance of holder is equal 0  
  - Throw exception.

## forceUpgrade(address[] \_holders)

* Description  
  - Upgrade master force to upgrade token of holders to a new smart contract.

* Flow  
  - Loop each holder to perform upgrade:
  + Copy balance of holder from current smart contract to new smart contract.
  + Reset balance of holder in current smart contract to 0.
  + Subtract from total supply in current smart contract.
  + Add to total token upgraded in current smart contract.
  + Emit event Upgrade(address indexed from, address indexed to, uint value).
* Check #1: Sender is not upgrade master  
  - Throw exception.

* Check #2: QNTU smart contract is not allowed to upgrade  
  - Throw exception.

* Check #3: Balance of holder in list is equal 0  
  - Exclude that holder and upgrade next holder.

## transferToContract(address \_to, uint \_value)

* Description  
  - Holder transfer their own tokens to specified ReceivingContract smart contract.

* Flow  
  - Add to balance of receiver.  
  - Subtract from balance of sender.

- Call ReceivingContract.tokenFallback function to store address and value of sender.  
- Emit event Transfer(address indexed from, address indexed to, uint value).

* Check #1: This function has been frozen  
  - Throw exception.

* Check #2: \_value is equal 0  
  - Throw exception.
* Check #3: Balance of sender is not enough  
  - Throw exception.

* Check #4: \_to is not address of ReceivingContract smart contract  
  - Throw exception.