CertiK Audit Report For Stasis

Request Date: 2019-06-12
Revision Date: 2019-06-18
Platform Name: Ethereum

Contents

Disclaimer

About CertiK

CertiK is a technology-led blockchain security company founded by Computer Science professors from Yale University and Columbia University built to prove the security and correctness of smart contracts and blockchain protocols.

CertiK, in partnership with grants from IBM and the Ethereum Foundation, has developed a proprietary Formal Verification technology to apply rigorous and complete mathematical reasoning against code. This process ensures algorithms, protocols, and business functionalities are secured and working as intended across all platforms.

CertiK differs from traditional testing approaches by employing Formal Verification to mathematically prove blockchain ecosystem and smart contracts are hacker-resistant and bug-free. CertiK uses this industry-leading technology together with standardized test suites, static analysis and expert manual review to create a full-stack solution for our partners across the blockchain world to secure 1.4B in assets.

For more information: https://certik.org/

Exective Summary

This report has been prepared as product of the Smart Contract Audit request by Stasis. This audit was conducted to discover issues and vulnerabilities in the source code of Stasis's Smart Contracts. Utilizing CertiK's Formal Verification Platform, Static Analysis and Manual Review, a comprehensive examination has been performed. The auditing process pays special attention to the following considerations.

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessment of the codebase for best practice and industry standards.
- Ensuring contract logic meets the specifications and intentions of the client.
- Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders.
- Thorough line by line manual review of the entire codebase by industry experts.

Vulnerability Classification

For every issues found, CertiK categorizes them into 3 buckets based on its risk level:

Critical

The code implementation does not match the specification, or it could result in loss of funds for contract owner or users.

Medium

The code implementation does not match the specification at certain condition, or it could affect the security standard by lost of access control.

Low

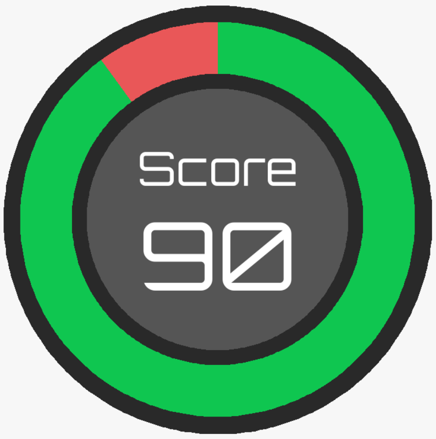
The code implementation is not a best practice, or use a suboptimal design pattern, which may lead to security vulnerability, but no concern found yet.

Testing Summary

PASS

CERTIK *believes this smart contract passes security qualifications to be listed on digital asset exchanges.*

Jun 18, 2019



Type of Issues

CertiK smart label engine applied 100% coveraged formal verification labels on the source code, and scanned the code using our proprietary static analysis and formal verification engine to detect the follow type of issues.

Title	Description	Issues	SWC ID
Integer Overflow and Underflow	An overflow/underflow happens when an arithmetic operation reaches the maximum or minimum size of a type.	1	SWC-101
Function incorrectness	Function implementation does not meet the specification, leading to intentional or unintentional vulnerabilities.	0	
Buffer Overflow	An attacker is able to write to arbitrary storage locations of a contract if array of out bound happens	0	SWC-124
Reentrancy	A malicious contract can call back into the calling contract before the first invocation of the function is finished.	0	SWC-107
Transaction Order Dependence	A race condition vulnerability occurs when code depends on the order of the transactions submitted to it.	0	SWC-114
Timestamp Dependence	Timestamp can be influenced by minors to some degree.	0	SWC-116
Insecure Compiler Version	Using an fixed outdated compiler version or floating pragma can be problematic, if there are publicly disclosed bugs and issues that affect the current compiler version used.	1	SWC-102 SWC-103
Insecure Randomness	Block attributes are insecure to generate random numbers, as they can be influenced by minors to some degree.	0	SWC-120

"tx.origin" for authorization	tx.origin should not be used for authorization. Use msg.sender instead.	0	SWC-115
Delegatecall to Untrusted Callee	Calling into untrusted contracts is very dangerous, the target and arguments provided must be sanitized.	1	SWC-112
State Variable Default Visibility	Labeling the visibility explicitly makes it easier to catch incorrect assumptions about who can access the variable.	0	SWC-108
Function Default Visibility	Functions are public by default. A malicious user is able to make unauthorized or unintended state changes if a developer forgot to set the visibility.	0	SWC-100
Uninitialized variables	Uninitialized local storage variables can point to other unexpected storage variables in the contract.	0	SWC-109
Assertion Failure	The assert() function is meant to assert invariants. Properly functioning code should never reach a failing assert statement.	0	SWC-110
Deprecated Solidity Features	Several functions and operators in Solidity are deprecated and should not be used as best practice.	0	SWC-111
Unused variables	Unused variables reduce code quality	0	

Vulnerability Details

Critical

No issue found.

Medium

No issue found.

Low

Missing address checks in transfer functions.

(Note: The violations in the formal verification section of the report are for internal evaluation and are not indication of security issue in the client contracts. However we recommend replacing `assert` with `require` in `SafeMath` as did in OpenZeppelin's latest `SafeMath` library.)

Manual Review Notes

Source Code SHA-256 Checksum

- **EURSToken.sol**
  `953ed4a23199ca0e00bb69bf1a605e96c9fed50a3d279aa1126efef958d12131`

Summary

CertiK was chosen by Stasis to audit the design and implementation of its EURS smart contract. To ensure comprehensive protection,the source code has been analyzed by the proprietary CertiK formal verification engine and manually reviewed by our smart contract experts and engineers. That end-to-end process ensures proof of stability as well as a hands-on, engineering-focused process to close potential loopholes and recommend design changes in accordance with the best practices in the space.

Overall we found the smart contracts to follow good practices. With the final update of source code and delivery of the audit report, we conclude that the contract is structurally sound and not vulnerable to any classically known anti-patterns or security issues. The audit report itself is not necessarily a guarantee of correctness or trustworthiness, and we always recommend to seek multiple opinions, keep improving the codebase, and more test coverage and sandbox deployments before the mainnet release.

Recommendations

SafeMath:

1. (INFO) Recommend avoiding use of magic number in `safeAdd()` & `safeMul()`. This can be achieved by performing the math operation first and then assert the result.
   - Example for `safeAdd()`: `z = x + y; assert(z >= x);`
   - Example for `safeMul()`: `z = x * y; assert(z / x == b);`
2. (INFO) Recommend removing `return` statements from function `safeAdd()`, `safeSub()` , `safeMul()`, as the return value `uint256 z` has been specified in the functions already.
3. (INFO) Recommend complementing `SafeDiv()` to be consistent with the `SafeMath` library for future contract update.

AbstractToken:

1. (IMPORTANT) Missing address check in function `transfer (address _to, uint256 _value)` and `function transferFrom (address _from, address _to, uint256 _value)`, which may lead to possible value loss. Same for `EURSToken`, where the address check is missing in `function approve (address _spender, uint256 _value)`.
2. (INFO) Recommend using `require` statement at line 197, 198, 218, 220, 225 with error message provided.

EURSToken

1. (IMPORTANT) When the contract is initialized the `EURSToken()` has no checking for the `_feeCollector` address.
2. (IMPORTANT) Missing address check in `function setOwner`, `function setFeeCollector`, `function setDelegate`, `function approve`, and transfer functions.
3. (INFO) Recommend checking whether the `_from` address recovered from `ecrecover()` is zero address.
4. (INFO) `function transfer`: line 443/445 use `require` statement instead of `if` statement. Same for function `transferFrom`, `createTokens`, `burnTokens`, `freezeTransfers`, `unfreezeTransfers`.
5. (INFO) Recommend emitting error logs to transfer condition failure.
6. (INFO) Recommend complementing error messages to current `require()` statements.
7. (INFO) Recommend adding a deadline timestamp field besides `nonce` to the signature for `delegatedTransfer`.
8. (INFO) Inconsistent use of `SafeMath` in the `delegatedTransfer()` for the increment of nounce `nonces [_from] = _nonce + 1`, which may lead to integer overflow.
9. (INFO) Recommend following EIP-712 for the implementation of data signing and delegate transfer. Reference: EIP-712.
10. (INFO) The EURS contract employs various roles and features such as `Owner`, `Delegator`, `FeeCollector` to meet its business requirements. We recommend:
    - Providing PAUSING functionality to be able to assist in legal investigation and auditing, as well as emergencies.
    - Splitting the responsibility into single aspects to minimize negative effects under special circumstances. For example:
        - `SupplyController`: `createToken` & `burnToken` opertions
        - `RegularCompliance`: `freeze` & `unfreeze` operations
        - `Owner`: Administrative operations such as role transfer
11. (INFO) Recommend extracting the following checking logic in transfer functions as separate internal function/modifier to improve readability and testability:
    - `isTransferBlocked(address _from, address _to): (addressFlags [_from] | addressFlags [_to])\& BLACK_LIST_FLAG == BLACK_LIST_FLAG)`
    - `getTransferFee(address _from, address _to, uint256 _value): (addressFlags [_from ] | addressFlags [_to])\& ZERO_FEE_FLAG == ZERO_FEE_FLAG ? 0 : calculateFee (_value)`
12. (INFO) Recommend use `onlyOwner` modifier in place of `require (msg.sender == owner )` and provide error message for `require()`.
13. (IMPORTANT) Recommend using pull model instead of push model for the transfer of ownership & delegated role to reduce the risk of manual mistake.

```
address indexed owner;
address indexed proposedOwner;
function proposeNewOwner(address newOwner) isOwner public {
  require(newOwner != address(0), ...);
  proposedOwner = newOwner;
 // emit LogOwnerTransferProposed ...
}
function claimOwnership() public {
 require(msg.sender == proposedOwner, ...);
 owner = proposedOwner;
 proposedOwner = address(0);
 // emit LogOwnerTransferred ...
}
address indexed delegate;
address indexed proposedDelegate;
function proposeNewDelegate(address newDelegate) isOwner public {
 require(newDelegate != address(0), ...);
 require(newDelegate != delegate, ...);
 proposedDelegate = newDelegate;
 // emit LogDelegateRoleTransferProposed ...
}
function claimDelegate() public {
 require(msg.sender == proposedDelegate, ...);
 delegate = proposedDelegate;
 proposedDelegate = address(0);
 // emit LogDelegateRoleTransferred ...
}
```

Static Analysis Results

INSECURE_COMPILER_VERSION

Line 10 in File EURSToken.sol

```
pragma solidity ^0.4.20;
```

❗ Version to compile has the following bug: 0.4.20: DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x, ExpExponentCleanup, EventStructWrongData, NestedArrayFunctionCallDecoder 0.4.21: DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x, ExpExponentCleanup, EventStructWrongData, NestedArrayFunctionCallDecoder 0.4.22: DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x, ExpExponentCleanup, EventStructWrongData, OneOfTwoConstructorsSkipped 0.4.23: DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x, ExpExponentCleanup, EventStructWrongData 0.4.24: DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x, ExpExponentCleanup, EventStructWrongData 0.4.25: DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x 0.4.26: DynamicConstructorArgumentsClippedABIV2

Formal Verification Results

How to read

Detail for Request 1

transferFrom to same address

Verification date	20, Oct 2018
Verification timespan	395.38 ms

CERTIK *label location*: Line 30-34 in File howtoread.sol

CERTIK *label*

```
30  /*@CTK FAIL "transferFrom to same address"
31      @tag assume_completion
32      @pre from == to
33      @post __post.allowed[from][msg.sender] ==
34  */
```

Raw code location: Line 35-41 in File howtoread.sol

Raw code

```
35  function transferFrom(address from, address tc
        ) {
36      balances[from] = balances[from].sub(tokens
37      allowed[from][msg.sender] = allowed[from][
38      balances[to] = balances[to].add(tokens);
39      emit Transfer(from, to, tokens);
40      return true;
41  }
```

Counterexample: ✗ This code violates the specification

Initial environment

```
 1  Counter Example:
 2  Before Execution:
 3      Input = {
 4          from = 0x0
 5          to = 0x0
 6          tokens = 0x6c
 7      }
 8      This = 0

53              balance: 0x0
54          }
55      }
56
```

Post environment

```
57  After Execution:
58      Input = {
59          from = 0x0
60          to = 0x0
61          tokens = 0x6c
```

Formal Verification Request 1

Method will not encounter an assertion failure.

📅 18, Jun 2019
⏱ 15.76 ms

Line 26 in File EURSToken.sol

```
//@CTK FAIL NO_ASF
```

Line 35-40 in File EURSToken.sol

```
function safeAdd (uint256 x, uint256 y)
pure internal
returns (uint256 z) {
  assert (x <= MAX_UINT256 - y);
  return x + y;
}
```

❌ This code violates the specification.

```
Counter Example:
Before Execution:
    Input = {
        x = 3
        y = 254
    }
    This = 0
    Internal = {
        __has_assertion_failure = false
        __has_buf_overflow = false
        __has_overflow = false
        __has_returned = false
        __reverted = false
        msg = {
          "gas": 0,
          "sender": 0,
          "value": 0
        }
    }
    Other = {
        block = {
          "number": 0,
          "timestamp": 0
        }
        z = 0
    }
    Address_Map = [
      {
        "key": "ALL_OTHERS",
        "value": {
          "contract_name": "SafeMath",
          "balance": 0,
          "contract": {
            "MAX_UINT256": 0
          }
        }
      }
    ]
```

```

Function invocation is reverted.
```

Formal Verification Request 2

SafeMath add

📅 18, Jun 2019
⏱ 5.78 ms

Line 27-34 in File EURSToken.sol

```
/*@CTK "SafeMath add"
  @tag spec
  @pre (MAX_UINT256) == (0xFF)
  @post (x > (MAX_UINT256 - y)) == __reverted
  @post (!__reverted) -> z == (x + y)
  @post (!__reverted) -> !__has_overflow
  @post !(__has_buf_overflow)
*/
```

Line 35-40 in File EURSToken.sol

```
function safeAdd (uint256 x, uint256 y)
pure internal
returns (uint256 z) {
  assert (x <= MAX_UINT256 - y);
  return x + y;
}
```

✓ The code meets the specification.

Formal Verification Request 3

Method will not encounter an assertion failure.

📅 18, Jun 2019
⏱ 15.21 ms

Line 49 in File EURSToken.sol

```
//@CTK FAIL NO_ASF
```

Line 57-62 in File EURSToken.sol

```
function safeSub (uint256 x, uint256 y)
pure internal
returns (uint256 z) {
  assert (x >= y);
  return x - y;
}
```

✗ This code violates the specification.

```
Counter Example:
Before Execution:
    Input = {
        x = 0
        y = 1
    }
    This = 0
    Internal = {
        __has_assertion_failure = false
        __has_buf_overflow = false
        __has_overflow = false
        __has_returned = false
        __reverted = false
        msg = {
          "gas": 0,
          "sender": 0,
          "value": 0
        }
    }
    Other = {
        block = {
          "number": 0,
          "timestamp": 0
        }
        z = 0
    }
    Address_Map = [
      {
        "key": "ALL_OTHERS",
        "value": {
          "contract_name": "SafeMath",
          "balance": 0,
          "contract": {
            "MAX_UINT256": 0
          }
        }
      }
    ]

Function invocation is reverted.
```

Formal Verification Request 4

SafeMath sub

📅 18, Jun 2019
⏱ 1.35 ms

Line 50-56 in File EURSToken.sol

```
  /*@CTK "SafeMath sub"
    @tag spec
    @post (x < y) == (__reverted)
    @post (!__reverted) -> (z == (x - y))
    @post (!__reverted) -> (!__has_overflow)
    @post !(__has_buf_overflow)
  */
```

Line 57-62 in File EURSToken.sol

```
  function safeSub (uint256 x, uint256 y)
  pure internal
```

```
returns (uint256 z) {
  assert (x >= y);
  return x - y;
}
```

✓ The code meets the specification.

Formal Verification Request 5

Method will not encounter an assertion failure.

18, Jun 2019
21.91 ms

Line 71 in File EURSToken.sol

```
//@CTK FAIL NO_ASF
```

Line 81-87 in File EURSToken.sol

```
function safeMul (uint256 x, uint256 y)
pure internal
returns (uint256 z) {
  if (y == 0) return 0; // Prevent division by zero at the next line
  assert (x <= MAX_UINT256 / y);
  return x * y;
}
```

✗ This code violates the specification.

```
Counter Example:
Before Execution:
    Input = {
        x = 255
        y = 1
    }
    This = 0
    Internal = {
        __has_assertion_failure = false
        __has_buf_overflow = false
        __has_overflow = false
        __has_returned = false
        __reverted = false
        msg = {
          "gas": 0,
          "sender": 0,
          "value": 0
        }
    }
    Other = {
        block = {
          "number": 0,
          "timestamp": 0
        }
        z = 0
    }
    Address_Map = [
      {
        "key": "ALL_OTHERS",
        "value": {
          "contract_name": "SafeMath",
          "balance": 0,
          "contract": {
            "MAX_UINT256": 254
          }
        }
      }
    ]

Function invocation is reverted.
```

Formal Verification Request 6

SafeMath mul

📅 18, Jun 2019
⏱ 17.92 ms

Line 72-80 in File EURSToken.sol

```
  /*@CTK "SafeMath mul"
    @tag spec
    @pre (MAX_UINT256) == (0xFF)
    @post (y == 0) -> (z == 0)
    @post (x > (MAX_UINT256 / y)) == (__reverted)
    @post (!__reverted) -> (z == (x * y))
    @post (!__reverted) -> (!__has_overflow)
    @post !(__has_buf_overflow)
  */
```

Line 81-87 in File EURSToken.sol

```
  function safeMul (uint256 x, uint256 y)
  pure internal
  returns (uint256 z) {
    if (y == 0) return 0; // Prevent division by zero at the next line
    assert (x <= MAX_UINT256 / y);
    return x * y;
  }
```

✅ The code meets the specification.

Formal Verification Request 7

balanceOf

📅 18, Jun 2019
⏱ 5.87 ms

Line 205-209 in File EURSToken.sol

```
  /*@CTK "balanceOf"
    @tag spec
    @tag assume_completion
    @post balance == accounts [_owner]
  */
```

Line 210-212 in File EURSToken.sol

```
function balanceOf (address _owner) public view returns (uint256 balance) {
  return accounts [_owner];
}
```

✓ The code meets the specification.

Formal Verification Request 8

If method completes, integer overflow would not happen.

18, Jun 2019
98.15 ms

Line 221 in File EURSToken.sol

```
//@CTK FAIL NO_OVERFLOW
```

Line 238-248 in File EURSToken.sol

```
function transfer (address _to, uint256 _value)
public payable returns (bool success) {
  uint256 fromBalance = accounts [msg.sender];
  if (fromBalance < _value) return false;
  if (_value > 0 && msg.sender != _to) {
    accounts [msg.sender] = safeSub (fromBalance, _value);
    accounts [_to] = safeAdd (accounts [_to], _value);
  }
  Transfer (msg.sender, _to, _value);
  return true;
}
```

✗ This code violates the specification.

```
Counter Example:
Before Execution:
    Input = {
        _to = 1
        _value = 128
    }
    This = 0
    Internal = {
        __has_assertion_failure = false
        __has_buf_overflow = false
        __has_overflow = false
        __has_returned = false
        __reverted = false
        msg = {
          "gas": 0,
          "sender": 0,
          "value": 0
        }
    }
    Other = {
        block = {
          "number": 0,
          "timestamp": 0
        }
        success = false
    }
    Address_Map = [
      {
        "key": 0,
        "value": {
          "contract_name": "AbstractToken",
          "balance": 0,
          "contract": {
            "accounts": [
              {
                "key": 0,
                "value": 128
              },
              {
                "key": 1,
                "value": 0
              },
              {
                "key": "ALL_OTHERS",
                "value": 1
              }
            ],
            "allowances": [
              {
                "key": "ALL_OTHERS",
                "value": [
                  {
                    "key": "ALL_OTHERS",
                    "value": 0
                  }
                ]
              }
            ],
            "MAX_UINT256": 0
          }
        }
      },
      {
        "key": "ALL_OTHERS",
        "value": "EmptyAddress"
      }
    ]

After Execution:
    Input = {
        _to = 1
        _value = 128
    }
    This = 0
    Internal = {
        __has_assertion_failure = false
        __has_buf_overflow = false
        __has_overflow = true
        __has_returned = true
        __reverted = false
        msg = {
          "gas": 0,
          "sender": 0,
          "value": 0
        }
    }
    Other = {
        block = {
          "number": 0,
          "timestamp": 0
        }
        fromBalance = 128
        success = true
    }
    Address_Map = [
      {
        "key": 0,
        "value": {
          "contract_name": "AbstractToken",
          "balance": 0,
          "contract": {
            "accounts": [
              {
                "key": 0,
                "value": 0
              },
              {
                "key": 1,
                "value": 128
              },
              {
                "key": "ALL_OTHERS",
                "value": 1
              }
            ],
            "allowances": [
              {
                "key": "ALL_OTHERS",
                "value": [
                  {
                    "key": "ALL_OTHERS",
                    "value": 0
                  }
                ]
              }
            ],
            "MAX_UINT256": 0
          }
        }
      },
      {
        "key": "ALL_OTHERS",
        "value": "EmptyAddress"
      }
    ]
```

Formal Verification Request 9

Buffer overflow / array index out of bound would never happen.

📅 18, Jun 2019
⏱ 1.29 ms

Line 222 in File EURSToken.sol

```
//@CTK NO_BUF_OVERFLOW
```

Line 238-248 in File EURSToken.sol

```
function transfer (address _to, uint256 _value)
public payable returns (bool success) {
  uint256 fromBalance = accounts [msg.sender];
  if (fromBalance < _value) return false;
  if (_value > 0 && msg.sender != _to) {
    accounts [msg.sender] = safeSub (fromBalance, _value);
    accounts [_to] = safeAdd (accounts [_to], _value);
  }
  Transfer (msg.sender, _to, _value);
  return true;
}
```

✓ The code meets the specification.

Formal Verification Request 10

Method will not encounter an assertion failure.

📅 18, Jun 2019
⏱ 14.78 ms

Line 223 in File EURSToken.sol

```
//@CTK FAIL NO_ASF
```

Line 238-248 in File EURSToken.sol

```
function transfer (address _to, uint256 _value)
public payable returns (bool success) {
  uint256 fromBalance = accounts [msg.sender];
  if (fromBalance < _value) return false;
  if (_value > 0 && msg.sender != _to) {
    accounts [msg.sender] = safeSub (fromBalance, _value);
    accounts [_to] = safeAdd (accounts [_to], _value);
  }
  Transfer (msg.sender, _to, _value);
  return true;
}
```

✗ This code violates the specification.

```
Counter Example:
Before Execution:
    Input = {
        _to = 4
        _value = 97
    }
    This = 0
    Internal = {
        __has_assertion_failure = false
        __has_buf_overflow = false
        __has_overflow = false
        __has_returned = false
        __reverted = false
        msg = {
          "gas": 0,
          "sender": 0,
          "value": 0
        }
    }
    Other = {
        block = {
          "number": 0,
          "timestamp": 0
        }
        success = false
    }
    Address_Map = [
      {
        "key": 0,
        "value": {
          "contract_name": "AbstractToken",
          "balance": 0,
          "contract": {
            "accounts": [
              {
                "key": 0,
                "value": 225
              },
              {
                "key": 4,
                "value": 64
              },
              {
                "key": "ALL_OTHERS",
                "value": 0
              }
            ],
            "allowances": [
              {
                "key": "ALL_OTHERS",
                "value": [
                  {
                    "key": "ALL_OTHERS",
                    "value": 4
                  }
                ]
              }
            ],
            "MAX_UINT256": 128
          }
        }
      },
      {
```

```
      "key": "ALL_OTHERS",
      "value": "EmptyAddress"
    }
  ]

Function invocation is reverted.
```

Formal Verification Request 11

transfer error

📅 18, Jun 2019
⏱ 110.33 ms

Line 224-230 in File EURSToken.sol

```
/*@CTK "transfer error"
  @tag spec
  @tag assume_completion
  @post (accounts[msg.sender] < _value) -> (success == false)
  @post (accounts[msg.sender] >= _value && msg.sender != _to) -> (__post.accounts[
      msg.sender] == accounts[msg.sender] - _value) && (__post.accounts[_to] ==
      accounts[_to] + _value)
  @post (accounts[msg.sender] < _value || _value == 0 || msg.sender == _to) -> (
      __post.accounts[msg.sender] == accounts[msg.sender]) && (__post.accounts[_to]
      == accounts[_to])
*/
```

Line 238-248 in File EURSToken.sol

```
function transfer (address _to, uint256 _value)
public payable returns (bool success) {
  uint256 fromBalance = accounts [msg.sender];
  if (fromBalance < _value) return false;
  if (_value > 0 && msg.sender != _to) {
    accounts [msg.sender] = safeSub (fromBalance, _value);
    accounts [_to] = safeAdd (accounts [_to], _value);
  }
  Transfer (msg.sender, _to, _value);
  return true;
}
```

✓ The code meets the specification.

Formal Verification Request 12

transfer

📅 18, Jun 2019
⏱ 110.99 ms

Line 231-237 in File EURSToken.sol

```
/*@CTK "transfer"
  @tag spec
  @tag assume_completion
  @post (accounts[msg.sender] < _value) -> (success == false)
  @post (accounts[msg.sender] >= _value && msg.sender != _to) -> (__post.accounts[
      msg.sender] == accounts[msg.sender] - _value) && (__post.accounts[_to] ==
      accounts[_to] + _value)
  @post (accounts[msg.sender] < _value || _value == 0 || msg.sender == _to) -> (
      __post.accounts[msg.sender] == accounts[msg.sender]) && (__post.accounts[_to]
      == accounts[_to])
*/
```

Line 238-248 in File EURSToken.sol

```
function transfer (address _to, uint256 _value)
public payable returns (bool success) {
  uint256 fromBalance = accounts [msg.sender];
  if (fromBalance < _value) return false;
  if (_value > 0 && msg.sender != _to) {
    accounts [msg.sender] = safeSub (fromBalance, _value);
    accounts [_to] = safeAdd (accounts [_to], _value);
  }
  Transfer (msg.sender, _to, _value);
  return true;
}
```

✓ The code meets the specification.

Formal Verification Request 13

If method completes, integer overflow would not happen.

18, Jun 2019
128.59 ms

Line 259 in File EURSToken.sol

```
//@CTK FAIL NO_OVERFLOW
```

Line 272-288 in File EURSToken.sol

```
function transferFrom (address _from, address _to, uint256 _value)
public payable returns (bool success) {
  uint256 spenderAllowance = allowances [_from][msg.sender];
  if (spenderAllowance < _value) return false;
  uint256 fromBalance = accounts [_from];
  if (fromBalance < _value) return false;

  allowances [_from][msg.sender] =
    safeSub (spenderAllowance, _value);

  if (_value > 0 && _from != _to) {
    accounts [_from] = safeSub (fromBalance, _value);
    accounts [_to] = safeAdd (accounts [_to], _value);
  }
  Transfer (_from, _to, _value);
  return true;
}
```

✗ This code violates the specification.

```
Counter Example:
Before Execution:
    Input = {
        _from = 128
        _to = 0
        _value = 1
    }
    This = 0
    Internal = {
        __has_assertion_failure = false
        __has_buf_overflow = false
        __has_overflow = false
        __has_returned = false
        __reverted = false
        msg = {
          "gas": 0,
          "sender": 0,
          "value": 0
        }
    }
    Other = {
        block = {
          "number": 0,
          "timestamp": 0
        }
        success = false
    }
    Address_Map = [
      {
        "key": 0,
        "value": {
          "contract_name": "AbstractToken",
          "balance": 0,
          "contract": {
            "accounts": [
              {
                "key": 0,
                "value": 0
              },
              {
                "key": 64,
                "value": 16
              },
              {
                "key": 32,
                "value": 2
              },
              {
                "key": "ALL_OTHERS",
                "value": 128
              }
            ],
            "allowances": [
              {
                "key": 128,
                "value": [
                  {
                    "key": 0,
                    "value": 128
                  },
                  {
                    "key": "ALL_OTHERS",
                    "value": 0
                  }
                ]
              },
              {
                "key": "ALL_OTHERS",
                "value": [
                  {
                    "key": "ALL_OTHERS",
                    "value": 128
                  }
                ]
              }
            ],
            "MAX_UINT256": 0
          }
        }
      },
      {
        "key": "ALL_OTHERS",
        "value": "EmptyAddress"
      }
    ]

After Execution:
    Input = {
        _from = 128
        _to = 0
        _value = 1
    }
    This = 0
    Internal = {
        __has_assertion_failure = false
        __has_buf_overflow = false
        __has_overflow = true
        __has_returned = true
        __reverted = false
        msg = {
          "gas": 0,
          "sender": 0,
          "value": 0
        }
    }
    Other = {
        block = {
          "number": 0,
          "timestamp": 0
        }
        fromBalance = 128
        spenderAllowance = 128
        success = true
    }
    Address_Map = [
      {
        "key": 0,
        "value": {
          "contract_name": "AbstractToken",
          "balance": 0,
          "contract": {
              "accounts": [
                {
                  "key": 128,
                  "value": 127
                },
                {
                  "key": 0,
                  "value": 1
                },
                {
                  "key": 64,
                  "value": 16
                },
                {
                  "key": 32,
                  "value": 2
                },
                {
                  "key": "ALL_OTHERS",
                  "value": 128
                }
              ],
              "allowances": [
                {
                  "key": 128,
                  "value": [
                    {
                      "key": 0,
                      "value": 127
                    },
                    {
                      "key": "ALL_OTHERS",
                      "value": 0
                    }
                  ]
                },
                {
                  "key": "ALL_OTHERS",
                  "value": [
                    {
                      "key": "ALL_OTHERS",
                      "value": 128
                    }
                  ]
                }
              ],
              "MAX_UINT256": 0
          }
        }
      },
      {
        "key": "ALL_OTHERS",
        "value": "EmptyAddress"
```

```
175       }
176     ]
```

Formal Verification Request 14

Buffer overflow / array index out of bound would never happen.

18, Jun 2019
13.86 ms

Line 260 in File EURSToken.sol

```
260   //@CTK NO_BUF_OVERFLOW
```

Line 272-288 in File EURSToken.sol

```
272   function transferFrom (address _from, address _to, uint256 _value)
273   public payable returns (bool success) {
274     uint256 spenderAllowance = allowances [_from][msg.sender];
275     if (spenderAllowance < _value) return false;
276     uint256 fromBalance = accounts [_from];
277     if (fromBalance < _value) return false;
278
279     allowances [_from][msg.sender] =
280       safeSub (spenderAllowance, _value);
281
282     if (_value > 0 && _from != _to) {
283       accounts [_from] = safeSub (fromBalance, _value);
284       accounts [_to] = safeAdd (accounts [_to], _value);
285     }
286     Transfer (_from, _to, _value);
287     return true;
288   }
```

✓ The code meets the specification.

Formal Verification Request 15

Method will not encounter an assertion failure.

18, Jun 2019
43.37 ms

Line 261 in File EURSToken.sol

```
261   //@CTK FAIL NO_ASF
```

Line 272-288 in File EURSToken.sol

```
272   function transferFrom (address _from, address _to, uint256 _value)
273   public payable returns (bool success) {
274     uint256 spenderAllowance = allowances [_from][msg.sender];
275     if (spenderAllowance < _value) return false;
276     uint256 fromBalance = accounts [_from];
277     if (fromBalance < _value) return false;
278
279     allowances [_from][msg.sender] =
      safeSub (spenderAllowance, _value);

    if (_value > 0 && _from != _to) {
      accounts [_from] = safeSub (fromBalance, _value);
      accounts [_to] = safeAdd (accounts [_to], _value);
    }
    Transfer (_from, _to, _value);
    return true;
  }
```

❌ This code violates the specification.

```
Counter Example:
Before Execution:
    Input = {
        _from = 0
        _to = 64
        _value = 1
    }
    This = 0
    Internal = {
        __has_assertion_failure = false
        __has_buf_overflow = false
        __has_overflow = false
        __has_returned = false
        __reverted = false
        msg = {
          "gas": 0,
          "sender": 0,
          "value": 0
        }
    }
    Other = {
        block = {
          "number": 0,
          "timestamp": 0
        }
        success = false
    }
    Address_Map = [
      {
        "key": 0,
        "value": {
          "contract_name": "AbstractToken",
          "balance": 0,
          "contract": {
            "accounts": [
              {
                "key": 64,
                "value": 1
              },
              {
                "key": 128,
                "value": 32
              },
              {
                "key": 4,
                "value": 0
              },
              {
                "key": 8,
                "value": 0
              },
              {
                "key": "ALL_OTHERS",
                "value": 128
              }
            ],
            "allowances": [
              {
                "key": 0,
                "value": [
                  {
                    "key": 128,
                    "value": 0
                  },
                  {
                    "key": 0,
                    "value": 128
                  },
                  {
                    "key": 4,
                    "value": 128
                  },
                  {
                    "key": 8,
                    "value": 1
                  },
                  {
                    "key": "ALL_OTHERS",
                    "value": 64
                  }
                ]
              },
              {
                "key": "ALL_OTHERS",
                "value": [
                  {
                    "key": "ALL_OTHERS",
                    "value": 128
                  }
                ]
              }
            ],
            "MAX_UINT256": 1
          }
        }
      },
      {
        "key": "ALL_OTHERS",
        "value": "EmptyAddress"
      }
    ]

Function invocation is reverted.
```

Formal Verification Request 16

transferFrom

📅 18, Jun 2019
⏱ 182.42 ms

Line 262-271 in File EURSToken.sol

```
/*@CTK "transferFrom"
  @tag spec
  @tag assume_completion
  @post (allowances[_from][msg.sender] < _value) -> (success == false)
  @post (accounts [_from] < _value) -> (success == false)
  @post (allowances[_from][msg.sender] >= _value && accounts [_from] >= _value) -> (
      __post.allowances[_from][msg.sender] == allowances[_from][msg.sender] - _value
      )
  @post (allowances[_from][msg.sender] < _value || accounts [_from] < _value) -> (
      __post.allowances[_from][msg.sender] == allowances[_from][msg.sender])
  @post (allowances[_from][msg.sender] >= _value && accounts [_from] >= _value &&
      _from != _to) -> (__post.accounts[_from] == accounts [_from] - _value) && (
      __post.accounts[_to] == accounts [_to] + _value)
  @post (allowances[_from][msg.sender] < _value || accounts [_from] < _value ||
      _from == _to) -> (__post.accounts[_from] == accounts [_from]) && (__post.
      accounts[_to] == accounts [_to])
*/
```

Line 272-288 in File EURSToken.sol

```
function transferFrom (address _from, address _to, uint256 _value)
public payable returns (bool success) {
  uint256 spenderAllowance = allowances [_from][msg.sender];
  if (spenderAllowance < _value) return false;
  uint256 fromBalance = accounts [_from];
  if (fromBalance < _value) return false;

  allowances [_from][msg.sender] =
    safeSub (spenderAllowance, _value);

  if (_value > 0 && _from != _to) {
    accounts [_from] = safeSub (fromBalance, _value);
    accounts [_to] = safeAdd (accounts [_to], _value);
  }
  Transfer (_from, _to, _value);
  return true;
}
```

✅ The code meets the specification.

Formal Verification Request 17

If method completes, integer overflow would not happen.

📅 18, Jun 2019
⏱ 7.75 ms

Line 298 in File EURSToken.sol

```
//@CTK NO_OVERFLOW
```

Line 306-312 in File EURSToken.sol

```
function approve (address _spender, uint256 _value)
public payable returns (bool success) {
  allowances [msg.sender][_spender] = _value;
  Approval (msg.sender, _spender, _value);

  return true;
}
```

✓ The code meets the specification.

Formal Verification Request 18

Buffer overflow / array index out of bound would never happen.

18, Jun 2019
0.3 ms

Line 299 in File EURSToken.sol

```
//@CTK NO_BUF_OVERFLOW
```

Line 306-312 in File EURSToken.sol

```
function approve (address _spender, uint256 _value)
public payable returns (bool success) {
  allowances [msg.sender][_spender] = _value;
  Approval (msg.sender, _spender, _value);

  return true;
}
```

✓ The code meets the specification.

Formal Verification Request 19

Method will not encounter an assertion failure.

18, Jun 2019
0.34 ms

Line 300 in File EURSToken.sol

```
//@CTK NO_ASF
```

Line 306-312 in File EURSToken.sol

```
function approve (address _spender, uint256 _value)
public payable returns (bool success) {
  allowances [msg.sender][_spender] = _value;
  Approval (msg.sender, _spender, _value);

  return true;
}
```

✓ The code meets the specification.

Formal Verification Request 20

approve

📅 18, Jun 2019
⏱ 1.23 ms

Line 301-305 in File EURSToken.sol

```
/*@CTK "approve"
  @tag spec
  @tag assume_completion
  @post (__post.allowances[msg.sender][_spender]) == (_value)
*/
```

Line 306-312 in File EURSToken.sol

```
function approve (address _spender, uint256 _value)
public payable returns (bool success) {
  allowances [msg.sender][_spender] = _value;
  Approval (msg.sender, _spender, _value);

  return true;
}
```

✅ The code meets the specification.

Formal Verification Request 21

If method completes, integer overflow would not happen.

📅 18, Jun 2019
⏱ 4.57 ms

Line 325 in File EURSToken.sol

```
//@CTK NO_OVERFLOW
```

Line 333-336 in File EURSToken.sol

```
function allowance (address _owner, address _spender)
public view returns (uint256 remaining) {
  return allowances [_owner][_spender];
}
```

✅ The code meets the specification.

Formal Verification Request 22

Buffer overflow / array index out of bound would never happen.

📅 18, Jun 2019
⏱ 0.28 ms

Line 326 in File EURSToken.sol

```
//@CTK NO_BUF_OVERFLOW
```

Line 333-336 in File EURSToken.sol

```
function allowance (address _owner, address _spender)
public view returns (uint256 remaining) {
  return allowances [_owner][_spender];
}
```

✓ The code meets the specification.

Formal Verification Request 23

Method will not encounter an assertion failure.

📅 18, Jun 2019
⏱ 0.28 ms

Line 327 in File EURSToken.sol

```
//@CTK NO_ASF
```

Line 333-336 in File EURSToken.sol

```
function allowance (address _owner, address _spender)
public view returns (uint256 remaining) {
  return allowances [_owner][_spender];
}
```

✓ The code meets the specification.

Formal Verification Request 24

allowance

📅 18, Jun 2019
⏱ 0.32 ms

Line 328-332 in File EURSToken.sol

```
/*@CTK "allowance"
  @tag spec
  @tag assume_completion
  @post (remaining) == (allowances[_owner][_spender])
*/
```

Line 333-336 in File EURSToken.sol

```
function allowance (address _owner, address _spender)
public view returns (uint256 remaining) {
  return allowances [_owner][_spender];
}
```

✓ The code meets the specification.

Formal Verification Request 25

If method completes, integer overflow would not happen.

📅 18, Jun 2019
⏱ 32.8 ms

Line 449 in File EURSToken.sol

```
//@CTK NO_OVERFLOW
```

Line 461-469 in File EURSToken.sol

```
function EURSToken (address _feeCollector) public {
  fixedFee = DEFAULT_FEE;
  minVariableFee = 0;
  maxVariableFee = 0;
  variableFeeNumerator = 0;

  owner = msg.sender;
  feeCollector = _feeCollector;
}
```

✅ The code meets the specification.

Formal Verification Request 26

Buffer overflow / array index out of bound would never happen.

📅 18, Jun 2019
⏱ 0.41 ms

Line 450 in File EURSToken.sol

```
//@CTK NO_BUF_OVERFLOW
```

Line 461-469 in File EURSToken.sol

```
function EURSToken (address _feeCollector) public {
  fixedFee = DEFAULT_FEE;
  minVariableFee = 0;
  maxVariableFee = 0;
  variableFeeNumerator = 0;

  owner = msg.sender;
  feeCollector = _feeCollector;
}
```

✅ The code meets the specification.

Formal Verification Request 27

Method will not encounter an assertion failure.

📅 18, Jun 2019
⏱ 0.41 ms

Line 451 in File EURSToken.sol

```
//@CTK NO_ASF
```

Line 461-469 in File EURSToken.sol

```
function EURSToken (address _feeCollector) public {
  fixedFee = DEFAULT_FEE;
  minVariableFee = 0;
  maxVariableFee = 0;
  variableFeeNumerator = 0;

  owner = msg.sender;
  feeCollector = _feeCollector;
}
```

✓ The code meets the specification.

Formal Verification Request 28

EURSToken

18, Jun 2019
1.27 ms

Line 452-460 in File EURSToken.sol

```
/*@CTK "EURSToken"
  @tag assume_completion
  @post __post.fixedFee == DEFAULT_FEE
  @post __post.minVariableFee == 0
  @post __post.maxVariableFee == 0
  @post __post.variableFeeNumerator == 0
  @post __post.owner == msg.sender
  @post __post.feeCollector == _feeCollector
 */
```

Line 461-469 in File EURSToken.sol

```
function EURSToken (address _feeCollector) public {
  fixedFee = DEFAULT_FEE;
  minVariableFee = 0;
  maxVariableFee = 0;
  variableFeeNumerator = 0;

  owner = msg.sender;
  feeCollector = _feeCollector;
}
```

✓ The code meets the specification.

Formal Verification Request 29

EURSToken totalSupply

18, Jun 2019
4.43 ms

Line 510-513 in File EURSToken.sol

```
/*@CTK "EURSToken totalSupply"
  @tag assume_completion
  @post __return == tokensCount
*/
```

Line 514-516 in File EURSToken.sol

```
function totalSupply () public /*>IGNORE delegatable IGNORE<*/ view returns (uint256
    ) {
  return tokensCount;
}
```

✓ The code meets the specification.

Formal Verification Request 30

EURSToken balanceOf

📅 18, Jun 2019
⏱ 20.79 ms

Line 525-528 in File EURSToken.sol

```
/*@CTK "EURSToken balanceOf"
  @tag assume_completion
  @post balance == accounts[_owner]
*/
```

Line 529-532 in File EURSToken.sol

```
function balanceOf (address _owner)
  public /*>IGNORE delegatable IGNORE<*/ view returns (uint256 balance) {
  return AbstractToken.balanceOf (_owner);
}
```

✓ The code meets the specification.

Formal Verification Request 31

If method completes, integer overflow would not happen.

📅 18, Jun 2019
⏱ 36.13 ms

Line 876 in File EURSToken.sol

```
//@CTK NO_OVERFLOW
```

Line 884-887 in File EURSToken.sol

```
function approve (address _spender, uint256 _value)
public /*>IGNORE delegatable IGNORE<*/ payable returns (bool success) {
  return AbstractToken.approve (_spender, _value);
}
```

✓ The code meets the specification.

Formal Verification Request 32

Buffer overflow / array index out of bound would never happen.

📅 18, Jun 2019
⏱ 0.41 ms

Line 877 in File EURSToken.sol

```
//@CTK NO_BUF_OVERFLOW
```

Line 884-887 in File EURSToken.sol

```
function approve (address _spender, uint256 _value)
public /*>IGNORE delegatable IGNORE<*/ payable returns (bool success) {
  return AbstractToken.approve (_spender, _value);
}
```

✓ The code meets the specification.

Formal Verification Request 33

Method will not encounter an assertion failure.

📅 18, Jun 2019
⏱ 0.42 ms

Line 878 in File EURSToken.sol

```
//@CTK NO_ASF
```

Line 884-887 in File EURSToken.sol

```
function approve (address _spender, uint256 _value)
public /*>IGNORE delegatable IGNORE<*/ payable returns (bool success) {
  return AbstractToken.approve (_spender, _value);
}
```

✓ The code meets the specification.

Formal Verification Request 34

EURSToken approve

📅 18, Jun 2019
⏱ 1.81 ms

Line 879-883 in File EURSToken.sol

```
/*@CTK "EURSToken approve"
  @tag spec
  @tag assume_completion
  @post (__post.allowances[msg.sender][_spender]) == (_value)
*/
```

Line 884-887 in File EURSToken.sol

```
function approve (address _spender, uint256 _value)
public /*>IGNORE delegatable IGNORE<*/ payable returns (bool success) {
  return AbstractToken.approve (_spender, _value);
}
```

✓ The code meets the specification.

Formal Verification Request 35

If method completes, integer overflow would not happen.

18, Jun 2019
23.58 ms

Line 900 in File EURSToken.sol

```
//@CTK NO_OVERFLOW
```

Line 908-911 in File EURSToken.sol

```
function allowance (address _owner, address _spender)
public /*>IGNORE delegatable IGNORE<*/ view returns (uint256 remaining) {
  return AbstractToken.allowance (_owner, _spender);
}
```

✓ The code meets the specification.

Formal Verification Request 36

Buffer overflow / array index out of bound would never happen.

18, Jun 2019
0.43 ms

Line 901 in File EURSToken.sol

```
//@CTK NO_BUF_OVERFLOW
```

Line 908-911 in File EURSToken.sol

```
function allowance (address _owner, address _spender)
public /*>IGNORE delegatable IGNORE<*/ view returns (uint256 remaining) {
  return AbstractToken.allowance (_owner, _spender);
}
```

✓ The code meets the specification.

Formal Verification Request 37

Method will not encounter an assertion failure.

18, Jun 2019
0.42 ms

Line 902 in File EURSToken.sol

```
//@CTK NO_ASF
```

Line 908-911 in File EURSToken.sol

```
function allowance (address _owner, address _spender)
public /*>IGNORE delegatable IGNORE<*/ view returns (uint256 remaining) {
  return AbstractToken.allowance (_owner, _spender);
}
```

✓ The code meets the specification.

Formal Verification Request 38

EURSToken allowance

📅 18, Jun 2019
⏱ 0.43 ms

Line 903-907 in File EURSToken.sol

```
/*@CTK "EURSToken allowance"
  @tag spec
  @tag assume_completion
  @post (remaining) == (allowances[_owner][_spender])
*/
```

Line 908-911 in File EURSToken.sol

```
function allowance (address _owner, address _spender)
public /*>IGNORE delegatable IGNORE<*/ view returns (uint256 remaining) {
  return AbstractToken.allowance (_owner, _spender);
}
```

✓ The code meets the specification.

Formal Verification Request 39

If method completes, integer overflow would not happen.

📅 18, Jun 2019
⏱ 222.47 ms

Line 1149 in File EURSToken.sol

```
//@CTK FAIL NO_OVERFLOW
```

Line 1159-1173 in File EURSToken.sol

```
function createTokens (uint256 _value)
public /*>IGNORE delegatable IGNORE<*/ payable returns (bool) {
  require (msg.sender == owner);

  if (_value > 0) {
    if (_value <= safeSub (MAX_TOKENS_COUNT, tokensCount)) {
      accounts [msg.sender] = safeAdd (accounts [msg.sender], _value);
      tokensCount = safeAdd (tokensCount, _value);

      Transfer (address (0), msg.sender, _value);
```

```

        return true;
      } else return false;
    } else return true;
  }
```

❌ This code violates the specification.

```
Counter Example:
Before Execution:
    Input = {
        _value = 66
    }
    This = 0
    Internal = {
        __has_assertion_failure = false
        __has_buf_overflow = false
        __has_overflow = false
        __has_returned = false
        __reverted = false
        msg = {
          "gas": 0,
          "sender": 0,
          "value": 0
        }
    }
    Other = {
        __return = false
        block = {
          "number": 0,
          "timestamp": 0
        }
    }
    Address_Map = [
      {
        "key": 0,
        "value": {
          "contract_name": "EURSToken",
          "balance": 0,
          "contract": {
            "FEE_DENOMINATOR": 0,
            "MAX_FEE_NUMERATOR": 0,
            "MIN_FEE_NUMERATIOR": 0,
            "MAX_TOKENS_COUNT": 135,
            "DEFAULT_FEE": 0,
            "BLACK_LIST_FLAG": 0,
            "ZERO_FEE_FLAG": 0,
            "owner": 0,
            "feeCollector": 0,
            "tokensCount": 7,
            "frozen": false,
            "nonces": [
              {
                "key": 0,
                "value": 4
              },
              {
                "key": 32,
                "value": 16
        },
        {
          "key": 8,
          "value": 128
        },
        {
          "key": "ALL_OTHERS",
          "value": 190
        }
      ],
      "fixedFee": 0,
      "minVariableFee": 0,
      "maxVariableFee": 0,
      "variableFeeNumerator": 0,
      "addressFlags": [
        {
          "key": 0,
          "value": 0
        },
        {
          "key": 16,
          "value": 8
        },
        {
          "key": "ALL_OTHERS",
          "value": 190
        }
      ],
      "delegate": 0,
      "accounts": [
        {
          "key": 16,
          "value": 0
        },
        {
          "key": 0,
          "value": 111
        },
        {
          "key": 32,
          "value": 0
        },
        {
          "key": 8,
          "value": 0
        },
        {
          "key": "ALL_OTHERS",
          "value": 190
        }
      ],
      "allowances": [
        {
          "key": "ALL_OTHERS",
          "value": [
            {
              "key": "ALL_OTHERS",
              "value": 190
```

```
                  }
                ]
              }
            ],
            "MAX_UINT256": 32
          }
        }
      },
      {
        "key": "ALL_OTHERS",
        "value": "EmptyAddress"
      }
    ]

After Execution:
    Input = {
        _value = 66
    }
    This = 0
    Internal = {
        __has_assertion_failure = false
        __has_buf_overflow = false
        __has_overflow = true
        __has_returned = true
        __reverted = false
        msg = {
          "gas": 0,
          "sender": 0,
          "value": 0
        }
    }
    Other = {
        __return = true
        block = {
          "number": 0,
          "timestamp": 0
        }
    }
    Address_Map = [
      {
        "key": 0,
        "value": {
          "contract_name": "EURSToken",
          "balance": 0,
          "contract": {
            "FEE_DENOMINATOR": 0,
            "MAX_FEE_NUMERATOR": 0,
            "MIN_FEE_NUMERATIOR": 0,
            "MAX_TOKENS_COUNT": 135,
            "DEFAULT_FEE": 0,
            "BLACK_LIST_FLAG": 0,
            "ZERO_FEE_FLAG": 0,
            "owner": 0,
            "feeCollector": 0,
            "tokensCount": 73,
            "frozen": false,
            "nonces": [
              {
```

```
            "key": 0,
            "value": 4
          },
          {
            "key": 32,
            "value": 16
          },
          {
            "key": 8,
            "value": 128
          },
          {
            "key": "ALL_OTHERS",
            "value": 190
          }
        ],
        "fixedFee": 0,
        "minVariableFee": 0,
        "maxVariableFee": 0,
        "variableFeeNumerator": 0,
        "addressFlags": [
          {
            "key": 0,
            "value": 0
          },
          {
            "key": 16,
            "value": 8
          },
          {
            "key": "ALL_OTHERS",
            "value": 190
          }
        ],
        "delegate": 0,
        "accounts": [
          {
            "key": 32,
            "value": 0
          },
          {
            "key": 0,
            "value": 177
          },
          {
            "key": 16,
            "value": 0
          },
          {
            "key": 8,
            "value": 0
          },
          {
            "key": "ALL_OTHERS",
            "value": 190
          }
        ],
        "allowances": [
              {
                "key": "ALL_OTHERS",
                "value": [
                  {
                    "key": "ALL_OTHERS",
                    "value": 190
                  }
                ]
              }
            ],
            "MAX_UINT256": 32
          }
        }
      },
      {
        "key": "ALL_OTHERS",
        "value": "EmptyAddress"
      }
    ]
```

Formal Verification Request 40

Buffer overflow / array index out of bound would never happen.

18, Jun 2019
3.52 ms

Line 1150 in File EURSToken.sol

```
1150  //@CTK NO_BUF_OVERFLOW
```

Line 1159-1173 in File EURSToken.sol

```
1159  function createTokens (uint256 _value)
1160  public /*>IGNORE delegatable IGNORE<*/ payable returns (bool) {
1161    require (msg.sender == owner);
1162
1163    if (_value > 0) {
1164      if (_value <= safeSub (MAX_TOKENS_COUNT, tokensCount)) {
1165        accounts [msg.sender] = safeAdd (accounts [msg.sender], _value);
1166        tokensCount = safeAdd (tokensCount, _value);
1167
1168        Transfer (address (0), msg.sender, _value);
1169
1170        return true;
1171      } else return false;
1172    } else return true;
1173  }
```

✓ The code meets the specification.

Formal Verification Request 41

Method will not encounter an assertion failure.

18, Jun 2019
352.92 ms

Line 1151 in File EURSToken.sol

```
//@CTK FAIL NO_ASF
```

Line 1159-1173 in File EURSToken.sol

```
function createTokens (uint256 _value)
public /*>IGNORE delegatable IGNORE<*/ payable returns (bool) {
  require (msg.sender == owner);

  if (_value > 0) {
    if (_value <= safeSub (MAX_TOKENS_COUNT, tokensCount)) {
      accounts [msg.sender] = safeAdd (accounts [msg.sender], _value);
      tokensCount = safeAdd (tokensCount, _value);

      Transfer (address (0), msg.sender, _value);

      return true;
    } else return false;
  } else return true;
}
```

❌ This code violates the specification.

```
Counter Example:
Before Execution:
    Input = {
        _value = 6
    }
    This = 0
    Internal = {
        __has_assertion_failure = false
        __has_buf_overflow = false
        __has_overflow = false
        __has_returned = false
        __reverted = false
        msg = {
          "gas": 0,
          "sender": 0,
          "value": 0
        }
    }
    Other = {
        __return = false
        block = {
          "number": 0,
          "timestamp": 0
        }
    }
    Address_Map = [
      {
        "key": 0,
        "value": {
          "contract_name": "EURSToken",
          "balance": 0,
          "contract": {
            "FEE_DENOMINATOR": 0,
            "MAX_FEE_NUMERATOR": 0,
            "MIN_FEE_NUMERATIOR": 0,
            "MAX_TOKENS_COUNT": 144,
            "DEFAULT_FEE": 0,
            "BLACK_LIST_FLAG": 0,
            "ZERO_FEE_FLAG": 0,
            "owner": 0,
            "feeCollector": 0,
            "tokensCount": 196,
            "frozen": false,
            "nonces": [
              {
                "key": 16,
                "value": 8
              },
              {
                "key": 0,
                "value": 16
              },
              {
                "key": 1,
                "value": 64
              },
              {
                "key": 129,
                "value": 0
              },
              {
                "key": 2,
                "value": 0
              },
              {
                "key": 4,
                "value": 64
              },
              {
                "key": "ALL_OTHERS",
                "value": 196
              }
            ],
            "fixedFee": 0,
            "minVariableFee": 0,
            "maxVariableFee": 0,
            "variableFeeNumerator": 0,
            "addressFlags": [
              {
                "key": 0,
                "value": 0
              },
              {
                "key": 128,
                "value": 0
              },
              {
                "key": 64,
                "value": 0
              },
              {
                "key": "ALL_OTHERS",
                "value": 32
            }
          ],
          "delegate": 0,
          "accounts": [
            {
              "key": 32,
              "value": 0
            },
            {
              "key": 0,
              "value": 226
            },
            {
              "key": 129,
              "value": 4
            },
            {
              "key": 2,
              "value": 64
            },
            {
              "key": 4,
              "value": 0
            },
            {
              "key": 16,
              "value": 1
            },
            {
              "key": 6,
              "value": 128
            },
            {
              "key": 8,
              "value": 32
            },
            {
              "key": 128,
              "value": 32
            },
            {
              "key": "ALL_OTHERS",
              "value": 196
            }
          ],
          "allowances": [
            {
              "key": 0,
              "value": [
                {
                  "key": 16,
                  "value": 64
                },
                {
                  "key": 0,
                  "value": 0
                },
                {
                  "key": 128,
                  "value": 32
                },
                {
                  "key": "ALL_OTHERS",
                  "value": 196
                }
              ]
            },
            {
              "key": 64,
              "value": [
                {
                  "key": 0,
                  "value": 0
                },
                {
                  "key": "ALL_OTHERS",
                  "value": 64
                }
              ]
            },
            {
              "key": "ALL_OTHERS",
              "value": [
                {
                  "key": "ALL_OTHERS",
                  "value": 32
                }
              ]
            }
          ],
          "MAX_UINT256": 64
        }
      }
    },
    {
      "key": "ALL_OTHERS",
      "value": "EmptyAddress"
    }
  ]

Function invocation is reverted.
```

Formal Verification Request 42

EURSToken createTokens

📅 18, Jun 2019
⏱ 222.89 ms

Line 1152-1158 in File EURSToken.sol

```
  /*@CTK "EURSToken createTokens"
    @tag assume_completion
    @pre msg.sender == owner
    @post ((_value > 0) && (_value > MAX_TOKENS_COUNT - tokensCount)) -> !__return
    @post ((_value == 0) || (_value <= MAX_TOKENS_COUNT - tokensCount)) -> __return
    @post ((_value == 0) && (_value <= MAX_TOKENS_COUNT - tokensCount)) -> (__post.
        accounts[msg.sender] == accounts[msg.sender] + _value) && (__post.tokensCount
        == tokensCount + _value)
  */
```

Line 1159-1173 in File EURSToken.sol

```
function createTokens (uint256 _value)
public /*>IGNORE delegatable IGNORE<*/ payable returns (bool) {
  require (msg.sender == owner);

  if (_value > 0) {
    if (_value <= safeSub (MAX_TOKENS_COUNT, tokensCount)) {
      accounts [msg.sender] = safeAdd (accounts [msg.sender], _value);
      tokensCount = safeAdd (tokensCount, _value);

      Transfer (address (0), msg.sender, _value);

      return true;
    } else return false;
  } else return true;
}
```

✓ The code meets the specification.

Formal Verification Request 43

If method completes, integer overflow would not happen.

📅 18, Jun 2019
⏱ 104.99 ms

Line 1180 in File EURSToken.sol

```
//@CTK NO_OVERFLOW
```

Line 1190-1204 in File EURSToken.sol

```
function burnTokens (uint256 _value)
public /*>IGNORE delegatable IGNORE<*/ payable returns (bool) {
  require (msg.sender == owner);

  if (_value > 0) {
    if (_value <= accounts [msg.sender]) {
      accounts [msg.sender] = safeSub (accounts [msg.sender], _value);
      tokensCount = safeSub (tokensCount, _value);

      Transfer (msg.sender, address (0), _value);

      return true;
    } else return false;
  } else return true;
}
```

✓ The code meets the specification.

Formal Verification Request 44

Buffer overflow / array index out of bound would never happen.

18, Jun 2019
7.81 ms

Line 1181 in File EURSToken.sol

```
//@CTK NO_BUF_OVERFLOW
```

Line 1190-1204 in File EURSToken.sol

```
function burnTokens (uint256 _value)
public /*>IGNORE delegatable IGNORE<*/ payable returns (bool) {
  require (msg.sender == owner);

  if (_value > 0) {
    if (_value <= accounts [msg.sender]) {
      accounts [msg.sender] = safeSub (accounts [msg.sender], _value);
      tokensCount = safeSub (tokensCount, _value);

      Transfer (msg.sender, address (0), _value);

      return true;
    } else return false;
  } else return true;
}
```

✓ The code meets the specification.

Formal Verification Request 45

Method will not encounter an assertion failure.

18, Jun 2019
49.84 ms

Line 1182 in File EURSToken.sol

```
//@CTK FAIL NO_ASF
```

Line 1190-1204 in File EURSToken.sol

```
function burnTokens (uint256 _value)
public /*>IGNORE delegatable IGNORE<*/ payable returns (bool) {
  require (msg.sender == owner);

  if (_value > 0) {
    if (_value <= accounts [msg.sender]) {
      accounts [msg.sender] = safeSub (accounts [msg.sender], _value);
      tokensCount = safeSub (tokensCount, _value);

      Transfer (msg.sender, address (0), _value);

      return true;
    } else return false;
  } else return true;
}
```

❌ This code violates the specification.

```
Counter Example:
Before Execution:
    Input = {
        _value = 32
    }
    This = 0
    Internal = {
        __has_assertion_failure = false
        __has_buf_overflow = false
        __has_overflow = false
        __has_returned = false
        __reverted = false
        msg = {
          "gas": 0,
          "sender": 0,
          "value": 0
        }
    }
    Other = {
        __return = false
        block = {
          "number": 0,
          "timestamp": 0
        }
    }
    Address_Map = [
      {
        "key": 0,
        "value": {
          "contract_name": "EURSToken",
          "balance": 0,
          "contract": {
            "FEE_DENOMINATOR": 0,
            "MAX_FEE_NUMERATOR": 0,
            "MIN_FEE_NUMERATIOR": 0,
            "MAX_TOKENS_COUNT": 0,
            "DEFAULT_FEE": 0,
            "BLACK_LIST_FLAG": 0,
            "ZERO_FEE_FLAG": 0,
            "owner": 0,
            "feeCollector": 0,
            "tokensCount": 0,
            "frozen": false,
            "nonces": [
              {
                "key": 4,
                "value": 16
              },
              {
                "key": 0,
                "value": 32
              },
              {
                "key": 2,
                "value": 8
              },
              {
                "key": "ALL_OTHERS",
                "value": 128
              }
            ],
            "fixedFee": 0,
            "minVariableFee": 0,
            "maxVariableFee": 0,
            "variableFeeNumerator": 0,
            "addressFlags": [
              {
                "key": 4,
                "value": 0
              },
              {
                "key": 2,
                "value": 64
              },
              {
                "key": 64,
                "value": 0
              },
              {
                "key": "ALL_OTHERS",
                "value": 128
              }
            ],
            "delegate": 0,
            "accounts": [
              {
                "key": 2,
                "value": 0
              },
              {
                "key": 64,
                "value": 16
              },
              {
                "key": "ALL_OTHERS",
                "value": 128
              }
            ],
            "allowances": [
              {
                "key": "ALL_OTHERS",
                "value": [
                  {
                    "key": "ALL_OTHERS",
                    "value": 128
                  }
                ]
              }
            ],
            "MAX_UINT256": 0
          }
        }
      },
      {
        "key": "ALL_OTHERS",
        "value": "EmptyAddress"
      }
    ]

Function invocation is reverted.
```

Formal Verification Request 46

EURSToken burnTokens

📅 18, Jun 2019
⏱ 285.3 ms

Line 1183-1189 in File EURSToken.sol

```
/*@CTK "EURSToken burnTokens"
  @tag assume_completion
  @pre msg.sender == owner
  @post ((_value > 0) && (_value > accounts [msg.sender])) -> !__return
  @post ((_value == 0) || (_value <= accounts [msg.sender])) -> __return
  @post ((_value == 0) || (_value <= accounts [msg.sender])) -> (__post.accounts[msg
      .sender] == accounts[msg.sender] - _value) && (__post.tokensCount ==
      tokensCount - _value)
*/
```

Line 1190-1204 in File EURSToken.sol

```
function burnTokens (uint256 _value)
public /*>IGNORE delegatable IGNORE<*/ payable returns (bool) {
  require (msg.sender == owner);

  if (_value > 0) {
    if (_value <= accounts [msg.sender]) {
      accounts [msg.sender] = safeSub (accounts [msg.sender], _value);
      tokensCount = safeSub (tokensCount, _value);

      Transfer (msg.sender, address (0), _value);

      return true;
    } else return false;
  } else return true;
}
```

✓ The code meets the specification.

Formal Verification Request 47

If method completes, integer overflow would not happen.

📅 18, Jun 2019
⏱ 22.63 ms

Line 1209 in File EURSToken.sol

```
//@CTK NO_OVERFLOW
```

Line 1217-1225 in File EURSToken.sol

```
1217  function freezeTransfers () public /*>IGNORE delegatable IGNORE<*/ payable {
1218    require (msg.sender == owner);
1219
1220    if (!frozen) {
1221      frozen = true;
1222
1223      Freeze ();
1224    }
1225  }
```

✅ The code meets the specification.

Formal Verification Request 48

Buffer overflow / array index out of bound would never happen.

📅 18, Jun 2019
⏱️ 0.51 ms

Line 1210 in File EURSToken.sol

```
1210  //@CTK NO_BUF_OVERFLOW
```

Line 1217-1225 in File EURSToken.sol

```
1217  function freezeTransfers () public /*>IGNORE delegatable IGNORE<*/ payable {
1218    require (msg.sender == owner);
1219
1220    if (!frozen) {
1221      frozen = true;
1222
1223      Freeze ();
1224    }
1225  }
```

✅ The code meets the specification.

Formal Verification Request 49

Method will not encounter an assertion failure.

📅 18, Jun 2019
⏱️ 0.5 ms

Line 1211 in File EURSToken.sol

```
1211  //@CTK NO_ASF
```

Line 1217-1225 in File EURSToken.sol

```
1217  function freezeTransfers () public /*>IGNORE delegatable IGNORE<*/ payable {
1218    require (msg.sender == owner);
1219
1220    if (!frozen) {
1221      frozen = true;
1222
1223      Freeze ();
```

```
    }
  }
```

✓ The code meets the specification.

Formal Verification Request 50

freezeTransfers

📅 18, Jun 2019
⏱ 10.73 ms

Line 1212-1216 in File EURSToken.sol

```
/*@CTK "freezeTransfers"
  @tag assume_completion
  @pre msg.sender == owner
  @post __post.frozen
 */
```

Line 1217-1225 in File EURSToken.sol

```
function freezeTransfers () public /*>IGNORE delegatable IGNORE<*/ payable {
  require (msg.sender == owner);

  if (!frozen) {
    frozen = true;

    Freeze ();
  }
}
```

✓ The code meets the specification.

Formal Verification Request 51

If method completes, integer overflow would not happen.

📅 18, Jun 2019
⏱ 21.61 ms

Line 1230 in File EURSToken.sol

```
//@CTK NO_OVERFLOW
```

Line 1238-1246 in File EURSToken.sol

```
function unfreezeTransfers () public /*>IGNORE delegatable IGNORE<*/ payable {
  require (msg.sender == owner);

  if (frozen) {
    frozen = false;

    Unfreeze ();
  }
}
```

✓ The code meets the specification.

Formal Verification Request 52

Buffer overflow / array index out of bound would never happen.

📅 18, Jun 2019
⏱ 0.5 ms

Line 1231 in File EURSToken.sol

```
//@CTK NO_BUF_OVERFLOW
```

Line 1238-1246 in File EURSToken.sol

```
function unfreezeTransfers () public /*>IGNORE delegatable IGNORE<*/ payable {
  require (msg.sender == owner);

  if (frozen) {
    frozen = false;

    Unfreeze ();
  }
}
```

✅ The code meets the specification.

Formal Verification Request 53

Method will not encounter an assertion failure.

📅 18, Jun 2019
⏱ 0.5 ms

Line 1232 in File EURSToken.sol

```
//@CTK NO_ASF
```

Line 1238-1246 in File EURSToken.sol

```
function unfreezeTransfers () public /*>IGNORE delegatable IGNORE<*/ payable {
  require (msg.sender == owner);

  if (frozen) {
    frozen = false;

    Unfreeze ();
  }
}
```

✅ The code meets the specification.

Formal Verification Request 54

unfreezeTransfers

📅 18, Jun 2019
⏱ 10.37 ms

Line 1233-1237 in File EURSToken.sol

```
/*@CTK "unfreezeTransfers"
  @tag assume_completion
  @pre msg.sender == owner
  @post __post.frozen == false
 */
```

Line 1238-1246 in File EURSToken.sol

```
function unfreezeTransfers () public /*>IGNORE delegatable IGNORE<*/ payable {
  require (msg.sender == owner);

  if (frozen) {
    frozen = false;

    Unfreeze ();
  }
}
```

✓ The code meets the specification.

Formal Verification Request 55

If method completes, integer overflow would not happen.

📅 18, Jun 2019
⏱ 17.97 ms

Line 1253 in File EURSToken.sol

```
//@CTK NO_OVERFLOW
```

Line 1261-1265 in File EURSToken.sol

```
function setOwner (address _newOwner) public {
  require (msg.sender == owner);

  owner = _newOwner;
}
```

✓ The code meets the specification.

Formal Verification Request 56

Buffer overflow / array index out of bound would never happen.

📅 18, Jun 2019
⏱ 0.47 ms

Line 1254 in File EURSToken.sol

```
//@CTK NO_BUF_OVERFLOW
```

Line 1261-1265 in File EURSToken.sol

```
function setOwner (address _newOwner) public {
  require (msg.sender == owner);

  owner = _newOwner;
}
```

✓ The code meets the specification.

Formal Verification Request 57

Method will not encounter an assertion failure.

18, Jun 2019
0.43 ms

Line 1255 in File EURSToken.sol

```
//@CTK NO_ASF
```

Line 1261-1265 in File EURSToken.sol

```
function setOwner (address _newOwner) public {
  require (msg.sender == owner);

  owner = _newOwner;
}
```

✓ The code meets the specification.

Formal Verification Request 58

unfreezeTransfers

18, Jun 2019
2.62 ms

Line 1256-1260 in File EURSToken.sol

```
/*@CTK "unfreezeTransfers"
  @tag assume_completion
  @post msg.sender == owner
  @post __post.owner == _newOwner
 */
```

Line 1261-1265 in File EURSToken.sol

```
function setOwner (address _newOwner) public {
  require (msg.sender == owner);

  owner = _newOwner;
}
```

✓ The code meets the specification.

Formal Verification Request 59

If method completes, integer overflow would not happen.

18, Jun 2019
16.58 ms

Line 1272 in File EURSToken.sol

```
1272  //@CTK NO_OVERFLOW
```

Line 1280-1285 in File EURSToken.sol

```
1280  function setFeeCollector (address _newFeeCollector)
1281  public /*>IGNORE delegatable IGNORE<*/ payable {
1282    require (msg.sender == owner);
1283
1284    feeCollector = _newFeeCollector;
1285  }
```

The code meets the specification.

Formal Verification Request 60

Buffer overflow / array index out of bound would never happen.

18, Jun 2019
0.42 ms

Line 1273 in File EURSToken.sol

```
1273  //@CTK NO_BUF_OVERFLOW
```

Line 1280-1285 in File EURSToken.sol

```
1280  function setFeeCollector (address _newFeeCollector)
1281  public /*>IGNORE delegatable IGNORE<*/ payable {
1282    require (msg.sender == owner);
1283
1284    feeCollector = _newFeeCollector;
1285  }
```

The code meets the specification.

Formal Verification Request 61

Method will not encounter an assertion failure.

18, Jun 2019
0.42 ms

Line 1274 in File EURSToken.sol

```
1274  //@CTK NO_ASF
```

Line 1280-1285 in File EURSToken.sol

```
1280  function setFeeCollector (address _newFeeCollector)
1281  public /*>IGNORE delegatable IGNORE<*/ payable {
1282    require (msg.sender == owner);
1283
1284    feeCollector = _newFeeCollector;
1285  }
```

The code meets the specification.

Formal Verification Request 62

setFeeCollector

📅 18, Jun 2019
⏱ 2.69 ms

Line 1275-1279 in File EURSToken.sol

```
/*@CTK "setFeeCollector"
  @tag assume_completion
  @pre msg.sender == owner
  @post (__post.feeCollector) == (_newFeeCollector)
*/
```

Line 1280-1285 in File EURSToken.sol

```
function setFeeCollector (address _newFeeCollector)
public /*>IGNORE delegatable IGNORE<*/ payable {
  require (msg.sender == owner);

  feeCollector = _newFeeCollector;
}
```

✅ The code meets the specification.

Formal Verification Request 63

If method completes, integer overflow would not happen.

📅 18, Jun 2019
⏱ 5.55 ms

Line 1294 in File EURSToken.sol

```
//@CTK NO_OVERFLOW
```

Line 1301-1303 in File EURSToken.sol

```
function nonce (address _owner) public view /*>IGNORE delegatable IGNORE<*/ returns
    (uint256) {
  return nonces [_owner];
}
```

✅ The code meets the specification.

Formal Verification Request 64

Buffer overflow / array index out of bound would never happen.

📅 18, Jun 2019
⏱ 0.35 ms

Line 1295 in File EURSToken.sol

```
//@CTK NO_BUF_OVERFLOW
```

Line 1301-1303 in File EURSToken.sol

```
function nonce (address _owner) public view /*>IGNORE delegatable IGNORE<*/ returns
    (uint256) {
  return nonces [_owner];
}
```

✓ The code meets the specification.

Formal Verification Request 65

Method will not encounter an assertion failure.

📅 18, Jun 2019
⏱ 0.32 ms

Line 1296 in File EURSToken.sol

```
//@CTK NO_ASF
```

Line 1301-1303 in File EURSToken.sol

```
function nonce (address _owner) public view /*>IGNORE delegatable IGNORE<*/ returns
    (uint256) {
  return nonces [_owner];
}
```

✓ The code meets the specification.

Formal Verification Request 66

nonce

📅 18, Jun 2019
⏱ 0.33 ms

Line 1297-1300 in File EURSToken.sol

```
/*@CTK nonce
  @tag assume_completion
  @post __return == nonces [_owner]
*/
```

Line 1301-1303 in File EURSToken.sol

```
function nonce (address _owner) public view /*>IGNORE delegatable IGNORE<*/ returns
    (uint256) {
  return nonces [_owner];
}
```

✓ The code meets the specification.

Formal Verification Request 67

If method completes, integer overflow would not happen.

📅 18, Jun 2019
⏱ 59.45 ms

Line 1313 in File EURSToken.sol

```
//@CTK NO_OVERFLOW
```

Line 1326-1343 in File EURSToken.sol

```
function setFeeParameters (
  uint256 _fixedFee,
  uint256 _minVariableFee,
  uint256 _maxVariableFee,
  uint256 _variableFeeNumerator) public /*>IGNORE delegatable IGNORE<*/ payable {
  require (msg.sender == owner);

  require (_minVariableFee <= _maxVariableFee);
  require (_variableFeeNumerator <= MAX_FEE_NUMERATOR);

  fixedFee = _fixedFee;
  minVariableFee = _minVariableFee;
  maxVariableFee = _maxVariableFee;
  variableFeeNumerator = _variableFeeNumerator;

  FeeChange (
    _fixedFee, _minVariableFee, _maxVariableFee, _variableFeeNumerator);
}
```

✓ The code meets the specification.

Formal Verification Request 68

Buffer overflow / array index out of bound would never happen.

18, Jun 2019
35.88 ms

Line 1314 in File EURSToken.sol

```
//@CTK NO_BUF_OVERFLOW
```

Line 1326-1343 in File EURSToken.sol

```
function setFeeParameters (
  uint256 _fixedFee,
  uint256 _minVariableFee,
  uint256 _maxVariableFee,
  uint256 _variableFeeNumerator) public /*>IGNORE delegatable IGNORE<*/ payable {
  require (msg.sender == owner);

  require (_minVariableFee <= _maxVariableFee);
  require (_variableFeeNumerator <= MAX_FEE_NUMERATOR);

  fixedFee = _fixedFee;
  minVariableFee = _minVariableFee;
  maxVariableFee = _maxVariableFee;
  variableFeeNumerator = _variableFeeNumerator;

  FeeChange (
    _fixedFee, _minVariableFee, _maxVariableFee, _variableFeeNumerator);
}
```

✓ The code meets the specification.

Formal Verification Request 69

Method will not encounter an assertion failure.

18, Jun 2019
35.97 ms

Line 1315 in File EURSToken.sol

```
//@CTK NO_ASF
```

Line 1326-1343 in File EURSToken.sol

```
function setFeeParameters (
  uint256 _fixedFee,
  uint256 _minVariableFee,
  uint256 _maxVariableFee,
  uint256 _variableFeeNumerator) public /*>IGNORE delegatable IGNORE<*/ payable {
  require (msg.sender == owner);

  require (_minVariableFee <= _maxVariableFee);
  require (_variableFeeNumerator <= MAX_FEE_NUMERATOR);

  fixedFee = _fixedFee;
  minVariableFee = _minVariableFee;
  maxVariableFee = _maxVariableFee;
  variableFeeNumerator = _variableFeeNumerator;

  FeeChange (
    _fixedFee, _minVariableFee, _maxVariableFee, _variableFeeNumerator);
}
```

✓ The code meets the specification.

Formal Verification Request 70

setFeeCollector

18, Jun 2019
5.88 ms

Line 1316-1325 in File EURSToken.sol

```
/*@CTK setFeeCollector
  @tag assume_completion
  @pre msg.sender == owner
  @pre _minVariableFee <= _maxVariableFee
  @pre _variableFeeNumerator <= MAX_FEE_NUMERATOR
  @post (__post.fixedFee) == (_fixedFee)
  @post (__post.minVariableFee) == (_minVariableFee)
  @post (__post.maxVariableFee) == (_maxVariableFee)
  @post (__post.variableFeeNumerator) == (_variableFeeNumerator)
*/
```

Line 1326-1343 in File EURSToken.sol

```
function setFeeParameters (
  uint256 _fixedFee,
  uint256 _minVariableFee,
  uint256 _maxVariableFee,
  uint256 _variableFeeNumerator) public /*>IGNORE delegatable IGNORE<*/ payable {
  require (msg.sender == owner);

  require (_minVariableFee <= _maxVariableFee);
  require (_variableFeeNumerator <= MAX_FEE_NUMERATOR);

  fixedFee = _fixedFee;
  minVariableFee = _minVariableFee;
  maxVariableFee = _maxVariableFee;
  variableFeeNumerator = _variableFeeNumerator;

  FeeChange (
    _fixedFee, _minVariableFee, _maxVariableFee, _variableFeeNumerator);
}
```

✓ The code meets the specification.

Formal Verification Request 71

If method completes, integer overflow would not happen.

18, Jun 2019
6.85 ms

Line 1350 in File EURSToken.sol

```
//@CTK NO_OVERFLOW
```

Line 1360-1369 in File EURSToken.sol

```
function getFeeParameters () public /*>IGNORE delegatable IGNORE<*/ view returns (
  uint256 _fixedFee,
  uint256 _minVariableFee,
  uint256 _maxVariableFee,
  uint256 _variableFeeNumnerator) {
  _fixedFee = fixedFee;
  _minVariableFee = minVariableFee;
  _maxVariableFee = maxVariableFee;
  _variableFeeNumnerator = variableFeeNumerator;
}
```

✓ The code meets the specification.

Formal Verification Request 72

Buffer overflow / array index out of bound would never happen.

18, Jun 2019
0.37 ms

Line 1351 in File EURSToken.sol

```
//@CTK NO_BUF_OVERFLOW
```

Line 1360-1369 in File EURSToken.sol

```
function getFeeParameters () public /*>IGNORE delegatable IGNORE<*/ view returns (
  uint256 _fixedFee,
  uint256 _minVariableFee,
  uint256 _maxVariableFee,
  uint256 _variableFeeNumnerator) {
  _fixedFee = fixedFee;
  _minVariableFee = minVariableFee;
  _maxVariableFee = maxVariableFee;
  _variableFeeNumnerator = variableFeeNumerator;
}
```

✓ The code meets the specification.

Formal Verification Request 73

Method will not encounter an assertion failure.

18, Jun 2019
0.36 ms

Line 1352 in File EURSToken.sol

```
//@CTK NO_ASF
```

Line 1360-1369 in File EURSToken.sol

```
function getFeeParameters () public /*>IGNORE delegatable IGNORE<*/ view returns (
  uint256 _fixedFee,
  uint256 _minVariableFee,
  uint256 _maxVariableFee,
  uint256 _variableFeeNumnerator) {
  _fixedFee = fixedFee;
  _minVariableFee = minVariableFee;
  _maxVariableFee = maxVariableFee;
  _variableFeeNumnerator = variableFeeNumerator;
}
```

✓ The code meets the specification.

Formal Verification Request 74

getFeeParameters

18, Jun 2019
0.38 ms

Line 1353-1359 in File EURSToken.sol

```
/*@CTK getFeeParameters
  @tag assume_completion
  @post _fixedFee == fixedFee
  @post _minVariableFee == minVariableFee
  @post _maxVariableFee == maxVariableFee
  @post _variableFeeNumnerator == variableFeeNumerator
*/
```

Line 1360-1369 in File EURSToken.sol

```
function getFeeParameters () public /*>IGNORE delegatable IGNORE<*/ view returns (
  uint256 _fixedFee,
  uint256 _minVariableFee,
  uint256 _maxVariableFee,
  uint256 _variableFeeNumnerator) {
  _fixedFee = fixedFee;
  _minVariableFee = minVariableFee;
  _maxVariableFee = maxVariableFee;
  _variableFeeNumnerator = variableFeeNumerator;
}
```

✓ The code meets the specification.

Formal Verification Request 75

If method completes, integer overflow would not happen.

18, Jun 2019
121.09 ms

Line 1377 in File EURSToken.sol

```
//@CTK FAIL NO_OVERFLOW
```

Line 1388-1396 in File EURSToken.sol

```
function calculateFee (uint256 _amount)
  public /*>IGNORE delegatable IGNORE<*/ view returns (uint256 _fee) {
  require (_amount <= MAX_TOKENS_COUNT);

  _fee = safeMul (_amount, variableFeeNumerator) / FEE_DENOMINATOR;
  if (_fee < minVariableFee) _fee = minVariableFee;
  if (_fee > maxVariableFee) _fee = maxVariableFee;
  _fee = safeAdd (_fee, fixedFee);
}
```

✗ This code violates the specification.

```
Counter Example:
Before Execution:
    Input = {
        _amount = 10
    }
    This = 0
    Internal = {
        __has_assertion_failure = false
        __has_buf_overflow = false
        __has_overflow = false
        __has_returned = false
        __reverted = false
        msg = {
          "gas": 0,
          "sender": 0,
          "value": 0
        }
    }
    Other = {
        _fee = 0
        block = {
          "number": 0,
          "timestamp": 0
        }
    }
    Address_Map = [
      {
        "key": "ALL_OTHERS",
        "value": {
          "contract_name": "EURSToken",
          "balance": 0,
          "contract": {
            "FEE_DENOMINATOR": 176,
            "MAX_FEE_NUMERATOR": 0,
            "MIN_FEE_NUMERATIOR": 0,
            "MAX_TOKENS_COUNT": 132,
            "DEFAULT_FEE": 0,
            "BLACK_LIST_FLAG": 0,
            "ZERO_FEE_FLAG": 0,
            "owner": 0,
            "feeCollector": 0,
            "tokensCount": 0,
            "frozen": false,
            "nonces": [
              {
                "key": 0,
                "value": 0
              },
              {
                "key": "ALL_OTHERS",
                "value": 16
              }
            ],
            "fixedFee": 242,
            "minVariableFee": 1,
            "maxVariableFee": 0,
            "variableFeeNumerator": 15,
            "addressFlags": [
              {
                "key": "ALL_OTHERS",
                "value": 0
              }
            ],
            "delegate": 0,
            "accounts": [
              {
                "key": 4,
                "value": 8
              },
              {
                "key": "ALL_OTHERS",
                "value": 16
              }
            ],
            "allowances": [
              {
                "key": "ALL_OTHERS",
                "value": [
                  {
                    "key": 0,
                    "value": 0
                  },
                  {
                    "key": "ALL_OTHERS",
                    "value": 16
                  }
                ]
              }
            ],
            "MAX_UINT256": 240
          }
        }
      }
    ]

After Execution:
    Input = {
        _amount = 10
    }
    This = 0
    Internal = {
        __has_assertion_failure = false
        __has_buf_overflow = false
        __has_overflow = true
        __has_returned = false
        __reverted = false
        msg = {
          "gas": 0,
          "sender": 0,
          "value": 0
        }
    }
    Other = {
        _fee = 242
        block = {
          "number": 0,
          "timestamp": 0
        }
    }
    Address_Map = [
      {
        "key": "ALL_OTHERS",
        "value": {
          "contract_name": "EURSToken",
          "balance": 0,
          "contract": {
            "FEE_DENOMINATOR": 176,
            "MAX_FEE_NUMERATOR": 0,
            "MIN_FEE_NUMERATIOR": 0,
            "MAX_TOKENS_COUNT": 132,
            "DEFAULT_FEE": 0,
            "BLACK_LIST_FLAG": 0,
            "ZERO_FEE_FLAG": 0,
            "owner": 0,
            "feeCollector": 0,
            "tokensCount": 0,
            "frozen": false,
            "nonces": [
              {
                "key": 0,
                "value": 0
              },
              {
                "key": "ALL_OTHERS",
                "value": 16
              }
            ],
            "fixedFee": 242,
            "minVariableFee": 1,
            "maxVariableFee": 0,
            "variableFeeNumerator": 15,
            "addressFlags": [
              {
                "key": "ALL_OTHERS",
                "value": 0
              }
            ],
            "delegate": 0,
            "accounts": [
              {
                "key": 4,
                "value": 8
              },
              {
                "key": "ALL_OTHERS",
                "value": 16
              }
            ],
            "allowances": [
              {
                "key": "ALL_OTHERS",
                "value": [
                  {
                    "key": 0,
                    "value": 0
                  },
                  {
                    "key": "ALL_OTHERS",
                    "value": 16
                  }
                ]
              }
            ],
            "MAX_UINT256": 240
          }
        }
      }
    ]
```

Formal Verification Request 76

Buffer overflow / array index out of bound would never happen.

📅 18, Jun 2019
⏱ 8.36 ms

Line 1378 in File EURSToken.sol

```
//@CTK NO_BUF_OVERFLOW
```

Line 1388-1396 in File EURSToken.sol

```
function calculateFee (uint256 _amount)
  public /*>IGNORE delegatable IGNORE<*/ view returns (uint256 _fee) {
  require (_amount <= MAX_TOKENS_COUNT);

  _fee = safeMul (_amount, variableFeeNumerator) / FEE_DENOMINATOR;
  if (_fee < minVariableFee) _fee = minVariableFee;
  if (_fee > maxVariableFee) _fee = maxVariableFee;
  _fee = safeAdd (_fee, fixedFee);
}
```

✓ The code meets the specification.

Formal Verification Request 77

Method will not encounter an assertion failure.

📅 18, Jun 2019
⏱ 38.54 ms

Line 1379 in File EURSToken.sol

```
//@CTK FAIL NO_ASF
```

Line 1388-1396 in File EURSToken.sol

```
function calculateFee (uint256 _amount)
  public /*>IGNORE delegatable IGNORE<*/ view returns (uint256 _fee) {
  require (_amount <= MAX_TOKENS_COUNT);

  _fee = safeMul (_amount, variableFeeNumerator) / FEE_DENOMINATOR;
  if (_fee < minVariableFee) _fee = minVariableFee;
  if (_fee > maxVariableFee) _fee = maxVariableFee;
  _fee = safeAdd (_fee, fixedFee);
}
```

✗ This code violates the specification.

```
Counter Example:
Before Execution:
    Input = {
        _amount = 0
    }
    This = 0
    Internal = {
        __has_assertion_failure = false
        __has_buf_overflow = false
        __has_overflow = false
        __has_returned = false
        __reverted = false
        msg = {
          "gas": 0,
          "sender": 0,
          "value": 0
        }
    }
    Other = {
        _fee = 0
        block = {
          "number": 0,
          "timestamp": 0
        }
    }
    Address_Map = [
      {
        "key": "ALL_OTHERS",
        "value": {
          "contract_name": "EURSToken",
          "balance": 0,
          "contract": {
            "FEE_DENOMINATOR": 14,
            "MAX_FEE_NUMERATOR": 0,
            "MIN_FEE_NUMERATIOR": 0,
            "MAX_TOKENS_COUNT": 0,
            "DEFAULT_FEE": 0,
            "BLACK_LIST_FLAG": 0,
            "ZERO_FEE_FLAG": 0,
            "owner": 0,
            "feeCollector": 0,
            "tokensCount": 0,
            "frozen": false,
            "nonces": [
              {
                "key": "ALL_OTHERS",
                "value": 0
              }
            ],
            "fixedFee": 0,
            "minVariableFee": 32,
            "maxVariableFee": 128,
            "variableFeeNumerator": 5,
            "addressFlags": [
              {
                "key": 0,
                "value": 2
              },
              {
                "key": "ALL_OTHERS",
                "value": 64
              }
            ],
            "delegate": 0,
            "accounts": [
              {
                "key": 0,
                "value": 0
              },
              {
                "key": "ALL_OTHERS",
                "value": 4
              }
            ],
            "allowances": [
              {
                "key": "ALL_OTHERS",
                "value": [
                  {
                    "key": "ALL_OTHERS",
                    "value": 0
                  }
                ]
              }
            ],
            "MAX_UINT256": 0
          }
        }
      }
    ]

  Function invocation is reverted.
```

Formal Verification Request 78

calculateFee

📅 18, Jun 2019
⏱ 2703.68 ms

Line 1380-1387 in File EURSToken.sol

```
  /*@CTK calculateFee
    @tag assume_completion
    @pre _amount <= MAX_TOKENS_COUNT
    @pre maxVariableFee > minVariableFee
    @post (_amount * variableFeeNumerator) / FEE_DENOMINATOR > maxVariableFee -> _fee
        == fixedFee + maxVariableFee
    @post (_amount * variableFeeNumerator) / FEE_DENOMINATOR < minVariableFee -> _fee
        == fixedFee + minVariableFee
    @post ((_amount * variableFeeNumerator) / FEE_DENOMINATOR >= minVariableFee && (
        _amount * variableFeeNumerator) / FEE_DENOMINATOR <= maxVariableFee) -> _fee
        == fixedFee + (_amount * variableFeeNumerator) / FEE_DENOMINATOR
  */
```

Line 1388-1396 in File EURSToken.sol

```
  function calculateFee (uint256 _amount)
    public /*>IGNORE delegatable IGNORE<*/ view returns (uint256 _fee) {
    require (_amount <= MAX_TOKENS_COUNT);

    _fee = safeMul (_amount, variableFeeNumerator) / FEE_DENOMINATOR;
    if (_fee < minVariableFee) _fee = minVariableFee;
    if (_fee > maxVariableFee) _fee = maxVariableFee;
    _fee = safeAdd (_fee, fixedFee);
```

```
}
```

✓ The code meets the specification.

Formal Verification Request 79

If method completes, integer overflow would not happen.

18, Jun 2019
15.43 ms

Line 1404 in File EURSToken.sol

```
//@CTK NO_OVERFLOW
```

Line 1412-1417 in File EURSToken.sol

```
function setFlags (address _address, uint256 _flags)
public /*>IGNORE delegatable IGNORE<*/ payable {
  require (msg.sender == owner);

  addressFlags [_address] = _flags;
}
```

✓ The code meets the specification.

Formal Verification Request 80

Buffer overflow / array index out of bound would never happen.

18, Jun 2019
0.45 ms

Line 1405 in File EURSToken.sol

```
//@CTK NO_BUF_OVERFLOW
```

Line 1412-1417 in File EURSToken.sol

```
function setFlags (address _address, uint256 _flags)
public /*>IGNORE delegatable IGNORE<*/ payable {
  require (msg.sender == owner);

  addressFlags [_address] = _flags;
}
```

✓ The code meets the specification.

Formal Verification Request 81

Method will not encounter an assertion failure.

18, Jun 2019
0.4 ms

Line 1406 in File EURSToken.sol

```
//@CTK NO_ASF
```

Line 1412-1417 in File EURSToken.sol

```
function setFlags (address _address, uint256 _flags)
public /*>IGNORE delegatable IGNORE<*/ payable {
  require (msg.sender == owner);

  addressFlags [_address] = _flags;
}
```

✓ The code meets the specification.

Formal Verification Request 82

setFlags

18, Jun 2019
2.85 ms

Line 1407-1411 in File EURSToken.sol

```
/*@CTK setFlags
  @tag assume_completion
  @pre msg.sender == owner
  @post __post.addressFlags [_address] == _flags
 */
```

Line 1412-1417 in File EURSToken.sol

```
function setFlags (address _address, uint256 _flags)
public /*>IGNORE delegatable IGNORE<*/ payable {
  require (msg.sender == owner);

  addressFlags [_address] = _flags;
}
```

✓ The code meets the specification.

Formal Verification Request 83

If method completes, integer overflow would not happen.

18, Jun 2019
19.24 ms

Line 1437 in File EURSToken.sol

```
//@CTK NO_OVERFLOW
```

Line 1445-1452 in File EURSToken.sol

```
function setDelegate (address _delegate) public {
  require (msg.sender == owner);

  if (delegate != _delegate) {
    delegate = _delegate;
    Delegation (delegate);
  }
}
```

✓ The code meets the specification.

Formal Verification Request 84

Buffer overflow / array index out of bound would never happen.

18, Jun 2019
0.47 ms

Line 1438 in File EURSToken.sol

```
//@CTK NO_BUF_OVERFLOW
```

Line 1445-1452 in File EURSToken.sol

```
function setDelegate (address _delegate) public {
  require (msg.sender == owner);

  if (delegate != _delegate) {
    delegate = _delegate;
    Delegation (delegate);
  }
}
```

✓ The code meets the specification.

Formal Verification Request 85

Method will not encounter an assertion failure.

18, Jun 2019
0.45 ms

Line 1439 in File EURSToken.sol

```
//@CTK NO_ASF
```

Line 1445-1452 in File EURSToken.sol

```
function setDelegate (address _delegate) public {
  require (msg.sender == owner);

  if (delegate != _delegate) {
    delegate = _delegate;
    Delegation (delegate);
  }
}
```

✓ The code meets the specification.

Formal Verification Request 86

setDelegate

18, Jun 2019
6.14 ms

Line 1440-1444 in File EURSToken.sol

```
/*@CTK "setDelegate"
  @tag assume_completion
  @pre msg.sender == owner
  @post __post.delegate == _delegate
 */
```

Line 1445-1452 in File EURSToken.sol

```
function setDelegate (address _delegate) public {
  require (msg.sender == owner);

  if (delegate != _delegate) {
    delegate = _delegate;
    Delegation (delegate);
  }
}
```

✓ The code meets the specification.

Source Code with CertiK Labels

File EURSToken.sol

```
/**
 *Submitted for verification at Etherscan.io on 2018-07-03
*/

/**
 * EURS Token Smart Contract: EIP-20 compatible token smart contract that
 * manages EURS tokens.
 */

pragma solidity ^0.4.20;

/**
 * Provides methods to safely add, subtract and multiply uint256 numbers.
 */
contract SafeMath {
  uint256 constant private MAX_UINT256 =
    0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF;

  /**
   * Add two uint256 values, throw in case of overflow.
   *
   * @param x first value to add
   * @param y second value to add
   * @return x + y
   */
  //@CTK FAIL NO_ASF
  /*@CTK "SafeMath add"
    @tag spec
    @pre (MAX_UINT256) == (0xFF)
    @post (x > (MAX_UINT256 - y)) == __reverted
    @post (!__reverted) -> z == (x + y)
    @post (!__reverted) -> !__has_overflow
    @post !(__has_buf_overflow)
  */
  function safeAdd (uint256 x, uint256 y)
  pure internal
  returns (uint256 z) {
    assert (x <= MAX_UINT256 - y);
    return x + y;
  }

  /**
   * Subtract one uint256 value from another, throw in case of underflow.
   *
   * @param x value to subtract from
   * @param y value to subtract
   * @return x - y
   */
  //@CTK FAIL NO_ASF
  /*@CTK "SafeMath sub"
    @tag spec
    @post (x < y) == (__reverted)
    @post (!__reverted) -> (z == (x - y))
    @post (!__reverted) -> (!__has_overflow)
    @post !(__has_buf_overflow)
  */
  function safeSub (uint256 x, uint256 y)
  pure internal
  returns (uint256 z) {
    assert (x >= y);
    return x - y;
  }

  /**
   * Multiply two uint256 values, throw in case of overflow.
   *
   * @param x first value to multiply
   * @param y second value to multiply
   * @return x * y
   */
  //@CTK FAIL NO_ASF
  /*@CTK "SafeMath mul"
    @tag spec
    @pre (MAX_UINT256) == (0xFF)
    @post (y == 0) -> (z == 0)
    @post (x > (MAX_UINT256 / y)) == (__reverted)
    @post (!__reverted) -> (z == (x * y))
    @post (!__reverted) -> (!__has_overflow)
    @post !(__has_buf_overflow)
  */
  function safeMul (uint256 x, uint256 y)
  pure internal
  returns (uint256 z) {
    if (y == 0) return 0; // Prevent division by zero at the next line
    assert (x <= MAX_UINT256 / y);
    return x * y;
  }
}
/*
 * EIP-20 Standard Token Smart Contract Interface.
 * Copyright (c) 2018 by STSS (Malta) Limited.
 * Contact: <tech@stasis.net>

 * ERC-20 standard token interface, as defined
 * <a href="https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20.md">here</a>.
 */
contract Token {
  /**
   * Get total number of tokens in circulation.
   *
   * @return total number of tokens in circulation
   */
  function totalSupply () public view returns (uint256 supply);

  /**
   * Get number of tokens currently belonging to given owner.
   *
   * @param _owner address to get number of tokens currently belonging to the
   *        owner of
   * @return number of tokens currently belonging to the owner of given address
   */
  function balanceOf (address _owner) public view returns (uint256 balance);
```

```

/**
 * Transfer given number of tokens from message sender to given recipient.
 *
 * @param _to address to transfer tokens to the owner of
 * @param _value number of tokens to transfer to the owner of given address
 * @return true if tokens were transferred successfully, false otherwise
 */
function transfer (address _to, uint256 _value)
public payable returns (bool success);

/**
 * Transfer given number of tokens from given owner to given recipient.
 *
 * @param _from address to transfer tokens from the owner of
 * @param _to address to transfer tokens to the owner of
 * @param _value number of tokens to transfer from given owner to given
 *        recipient
 * @return true if tokens were transferred successfully, false otherwise
 */
function transferFrom (address _from, address _to, uint256 _value)
public payable returns (bool success);

/**
 * Allow given spender to transfer given number of tokens from message sender.
 *
 * @param _spender address to allow the owner of to transfer tokens from
 *        message sender
 * @param _value number of tokens to allow to transfer
 * @return true if token transfer was successfully approved, false otherwise
 */
function approve (address _spender, uint256 _value)
public payable returns (bool success);

/**
 * Tell how many tokens given spender is currently allowed to transfer from
 * given owner.
 *
 * @param _owner address to get number of tokens allowed to be transferred
 *        from the owner of
 * @param _spender address to get number of tokens allowed to be transferred
 *        by the owner of
 * @return number of tokens given spender is currently allowed to transfer
 *         from given owner
 */
function allowance (address _owner, address _spender)
public view returns (uint256 remaining);

/**
 * Logged when tokens were transferred from one owner to another.
 *
 * @param _from address of the owner, tokens were transferred from
 * @param _to address of the owner, tokens were transferred to
 * @param _value number of tokens transferred
 */
event Transfer (address indexed _from, address indexed _to, uint256 _value);

/**
```

```
   * Logged when owner approved his tokens to be transferred by some spender.
   *
   * @param _owner owner who approved his tokens to be transferred
   * @param _spender spender who were allowed to transfer the tokens belonging
   *        to the owner
   * @param _value number of tokens belonging to the owner, approved to be
   *        transferred by the spender
   */
  event Approval (
    address indexed _owner, address indexed _spender, uint256 _value);
}
/*
 * Abstract Token Smart Contract.
 * Copyright (c) 2018 by STSS (Malta) Limited.
 * Contact: <tech@stasis.net>

 * Abstract Token Smart Contract that could be used as a base contract for
 * ERC-20 token contracts.
 */
contract AbstractToken is Token, SafeMath {
  /**
   * Create new Abstract Token contract.
   */
  function AbstractToken () public {
    // Do nothing
  }

  /**
   * Get number of tokens currently belonging to given owner.
   *
   * @param _owner address to get number of tokens currently belonging to the
   *        owner of
   * @return number of tokens currently belonging to the owner of given address
   */
  /*@CTK "balanceOf"
    @tag spec
    @tag assume_completion
    @post balance == accounts [_owner]
  */
  function balanceOf (address _owner) public view returns (uint256 balance) {
    return accounts [_owner];
  }

  /**
   * Transfer given number of tokens from message sender to given recipient.
   *
   * @param _to address to transfer tokens to the owner of
   * @param _value number of tokens to transfer to the owner of given address
   * @return true if tokens were transferred successfully, false otherwise
   */
  //@CTK FAIL NO_OVERFLOW
  //@CTK NO_BUF_OVERFLOW
  //@CTK FAIL NO_ASF
  /*@CTK "transfer error"
    @tag spec
    @tag assume_completion
    @post (accounts[msg.sender] < _value) -> (success == false)
    @post (accounts[msg.sender] >= _value && msg.sender != _to) -> (__post.accounts[
        msg.sender] == accounts[msg.sender] - _value) && (__post.accounts[_to] ==
        accounts[_to] + _value)
    @post (accounts[msg.sender] < _value || _value == 0 || msg.sender == _to) -> (
        __post.accounts[msg.sender] == accounts[msg.sender]) && (__post.accounts[_to]
        == accounts[_to])
  */
  /*@CTK "transfer"
    @tag spec
    @tag assume_completion
    @post (accounts[msg.sender] < _value) -> (success == false)
    @post (accounts[msg.sender] >= _value && msg.sender != _to) -> (__post.accounts[
        msg.sender] == accounts[msg.sender] - _value) && (__post.accounts[_to] ==
        accounts[_to] + _value)
    @post (accounts[msg.sender] < _value || _value == 0 || msg.sender == _to) -> (
        __post.accounts[msg.sender] == accounts[msg.sender]) && (__post.accounts[_to]
        == accounts[_to])
  */
  function transfer (address _to, uint256 _value)
  public payable returns (bool success) {
    uint256 fromBalance = accounts [msg.sender];
    if (fromBalance < _value) return false;
    if (_value > 0 && msg.sender != _to) {
      accounts [msg.sender] = safeSub (fromBalance, _value);
      accounts [_to] = safeAdd (accounts [_to], _value);
    }
    Transfer (msg.sender, _to, _value);
    return true;
  }

  /**
   * Transfer given number of tokens from given owner to given recipient.
   *
   * @param _from address to transfer tokens from the owner of
   * @param _to address to transfer tokens to the owner of
   * @param _value number of tokens to transfer from given owner to given
   *        recipient
   * @return true if tokens were transferred successfully, false otherwise
   */
  //@CTK FAIL NO_OVERFLOW
  //@CTK NO_BUF_OVERFLOW
  //@CTK FAIL NO_ASF
  /*@CTK "transferFrom"
    @tag spec
    @tag assume_completion
    @post (allowances[_from][msg.sender] < _value) -> (success == false)
    @post (accounts [_from] < _value) -> (success == false)
    @post (allowances[_from][msg.sender] >= _value && accounts [_from] >= _value) -> (
        __post.allowances[_from][msg.sender] == allowances[_from][msg.sender] - _value
        )
    @post (allowances[_from][msg.sender] < _value || accounts [_from] < _value) -> (
        __post.allowances[_from][msg.sender] == allowances[_from][msg.sender])
    @post (allowances[_from][msg.sender] >= _value && accounts [_from] >= _value &&
        _from != _to) -> (__post.accounts[_from] == accounts [_from] - _value) && (
        __post.accounts[_to] == accounts [_to] + _value)
    @post (allowances[_from][msg.sender] < _value || accounts [_from] < _value ||
        _from == _to) -> (__post.accounts[_from] == accounts [_from]) && (__post.
        accounts[_to] == accounts [_to])
```

```
*/
function transferFrom (address _from, address _to, uint256 _value)
public payable returns (bool success) {
  uint256 spenderAllowance = allowances [_from][msg.sender];
  if (spenderAllowance < _value) return false;
  uint256 fromBalance = accounts [_from];
  if (fromBalance < _value) return false;

  allowances [_from][msg.sender] =
    safeSub (spenderAllowance, _value);

  if (_value > 0 && _from != _to) {
    accounts [_from] = safeSub (fromBalance, _value);
    accounts [_to] = safeAdd (accounts [_to], _value);
  }
  Transfer (_from, _to, _value);
  return true;
}

/**
 * Allow given spender to transfer given number of tokens from message sender.
 *
 * @param _spender address to allow the owner of to transfer tokens from
 *        message sender
 * @param _value number of tokens to allow to transfer
 * @return true if token transfer was successfully approved, false otherwise
 */
//@CTK NO_OVERFLOW
//@CTK NO_BUF_OVERFLOW
//@CTK NO_ASF
/*@CTK "approve"
  @tag spec
  @tag assume_completion
  @post (__post.allowances[msg.sender][_spender]) == (_value)
*/
function approve (address _spender, uint256 _value)
public payable returns (bool success) {
  allowances [msg.sender][_spender] = _value;
  Approval (msg.sender, _spender, _value);

  return true;
}

/**
 * Tell how many tokens given spender is currently allowed to transfer from
 * given owner.
 *
 * @param _owner address to get number of tokens allowed to be transferred
 *        from the owner of
 * @param _spender address to get number of tokens allowed to be transferred
 *        by the owner of
 * @return number of tokens given spender is currently allowed to transfer
 *         from given owner
 */
//@CTK NO_OVERFLOW
//@CTK NO_BUF_OVERFLOW
//@CTK NO_ASF
/*@CTK "allowance"
```

```
    @tag spec
    @tag assume_completion
    @post (remaining) == (allowances[_owner][_spender])
  */
  function allowance (address _owner, address _spender)
  public view returns (uint256 remaining) {
    return allowances [_owner][_spender];
  }

  /**
   * Mapping from addresses of token holders to the numbers of tokens belonging
   * to these token holders.
   */
  mapping (address => uint256) internal accounts;

  /**
   * Mapping from addresses of token holders to the mapping of addresses of
   * spenders to the allowances set by these token holders to these spenders.
   */
  mapping (address => mapping (address => uint256)) internal allowances;
}

/*
 * EURS Token Smart Contract.
 * Copyright (c) 2018 by STSS (Malta) Limited.
 * Contact: <tech@stasis.net>
 */

contract EURSToken is AbstractToken {
  /**
   * Fee denominator (0.001%).
   */
  uint256 constant internal FEE_DENOMINATOR = 100000;

  /**
   * Maximum fee numerator (100%).
   */
  uint256 constant internal MAX_FEE_NUMERATOR = FEE_DENOMINATOR;

  /**
   * Minimum fee numerator (0%).
   */
  uint256 constant internal MIN_FEE_NUMERATIOR = 0;

  /**
   * Maximum allowed number of tokens in circulation.
   */
  uint256 constant internal MAX_TOKENS_COUNT =
    0xffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff /
    MAX_FEE_NUMERATOR;

  /**
   * Default transfer fee.
   */
  uint256 constant internal DEFAULT_FEE = 5e2;

  /**
   * Address flag that marks black listed addresses.
   */
  uint256 constant internal BLACK_LIST_FLAG = 0x01;

  /**
   * Address flag that marks zero fee addresses.
   */
  uint256 constant internal ZERO_FEE_FLAG = 0x02;

  modifier delegatable {
    if (delegate == address (0)) {
      require (msg.value == 0); // Non payable if not delegated
      _;
    } else {
      //@CTK NO_OVERFLOW
      //@CTK NO_BUF_OVERFLOW
      //@CTK NO_ASF
      /*@CTK "assembly call status"
        @post !(__reverted) -> delegate == delegate__pre
        @post !(__reverted) -> owner == owner__pre
       */
      assembly {
        // Save owner
        let oldOwner := sload (owner_slot)

        // Save delegate
        let oldDelegate := sload (delegate_slot)

        // Solidity stores address of the beginning of free memory at 0x40
        let buffer := mload (0x40)

        // Copy message call data into buffer
        calldatacopy (buffer, 0, calldatasize)

        // Lets call our delegate
        let result := delegatecall (gas, oldDelegate, buffer, calldatasize, buffer, 0)

        // Check, whether owner was changed
        switch eq (oldOwner, sload (owner_slot))
        case 1 {} // Owner was not changed, fine
        default {revert (0, 0) } // Owner was changed, revert!

        // Check, whether delegate was changed
        switch eq (oldDelegate, sload (delegate_slot))
        case 1 {} // Delegate was not changed, fine
        default {revert (0, 0) } // Delegate was changed, revert!

        // Copy returned value into buffer
        returndatacopy (buffer, 0, returndatasize)

        // Check call status
        switch result
        case 0 { revert (buffer, returndatasize) } // Call failed, revert!
        default { return (buffer, returndatasize) } // Call succeeded, return
      }
    }
  }

  /**
```

```
   * Create EURS Token smart contract with message sender as an owner.
   *
   * @param _feeCollector address fees are sent to
   */
  //@CTK NO_OVERFLOW
  //@CTK NO_BUF_OVERFLOW
  //@CTK NO_ASF
  /*@CTK "EURSToken"
    @tag assume_completion
    @post __post.fixedFee == DEFAULT_FEE
    @post __post.minVariableFee == 0
    @post __post.maxVariableFee == 0
    @post __post.variableFeeNumerator == 0
    @post __post.owner == msg.sender
    @post __post.feeCollector == _feeCollector
   */
  function EURSToken (address _feeCollector) public {
    fixedFee = DEFAULT_FEE;
    minVariableFee = 0;
    maxVariableFee = 0;
    variableFeeNumerator = 0;

    owner = msg.sender;
    feeCollector = _feeCollector;
  }

  /**
   * Delegate unrecognized functions.
   */
  function () public payable {
    revert (); // Revert if not delegated
  }

  /**
   * Get name of the token.
   *
   * @return name of the token
   */
  function name () public delegatable view returns (string) {
    return "STASIS EURS Token";
  }

  /**
   * Get symbol of the token.
   *
   * @return symbol of the token
   */
  function symbol () public delegatable view returns (string) {
    return "EURS";
  }

  /**
   * Get number of decimals for the token.
   *
   * @return number of decimals for the token
   */
  function decimals () public delegatable view returns (uint8) {
    return 2;
```

```
}

/**
 * Get total number of tokens in circulation.
 *
 * @return total number of tokens in circulation
 */
/*@CTK "EURSToken totalSupply"
  @tag assume_completion
  @post __return == tokensCount
*/
function totalSupply () public /*>IGNORE delegatable IGNORE<*/ view returns (uint256
    ) {
  return tokensCount;
}

/**
 * Get number of tokens currently belonging to given owner.
 *
 * @param _owner address to get number of tokens currently belonging to the
 *        owner of
 * @return number of tokens currently belonging to the owner of given address
 */
/*@CTK "EURSToken balanceOf"
  @tag assume_completion
  @post balance == accounts[_owner]
*/
function balanceOf (address _owner)
  public /*>IGNORE delegatable IGNORE<*/ view returns (uint256 balance) {
  return AbstractToken.balanceOf (_owner);
}

/**
 * Transfer given number of tokens from message sender to given recipient.
 *
 * @param _to address to transfer tokens to the owner of
 * @param _value number of tokens to transfer to the owner of given address
 * @return true if tokens were transferred successfully, false otherwise
 */
/*@CTK "transfer frozen"
  @tag spec
  @tag assume_completion
  @pre frozen
  @post (__post.accounts[msg.sender] == accounts[msg.sender])
  @post (__post.accounts[_to] == accounts[_to])
  @post (__post.accounts[feeCollector] == accounts[feeCollector])
  @post !__return
*/
/*@CTK "transfer blacklisted"
  @tag spec
  @tag assume_completion
  @pre !frozen
  @pre (addressFlags[msg.sender] == BLACK_LIST_FLAG) || (addressFlags [_to] ==
      BLACK_LIST_FLAG)
  @post (__post.accounts[msg.sender] == accounts[msg.sender])
  @post (__post.accounts[_to] == accounts[_to])
  @post (__post.accounts[feeCollector] == accounts[feeCollector])
  @post !__return
*/
/*@CTK "transfer zero-feed succeed"
  @tag spec
  @tag assume_completion
  @pre !frozen
  @pre (addressFlags[msg.sender] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
      BLACK_LIST_FLAG)
  @pre (addressFlags[msg.sender] == ZERO_FEE_FLAG) || (addressFlags [_to] ==
      ZERO_FEE_FLAG)
  @pre (_value <= accounts[msg.sender])
  @post (__post.accounts[msg.sender] == accounts[msg.sender] - _value)
  @post (__post.accounts[_to] == accounts[_to] + _value)
  @post (__post.accounts[feeCollector] == accounts[feeCollector])
  @post __return
*/
/*@CTK "transfer zero-feed fail"
  @tag spec
  @tag assume_completion
  @pre !frozen
  @pre (addressFlags[msg.sender] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
      BLACK_LIST_FLAG)
  @pre (addressFlags[msg.sender] == ZERO_FEE_FLAG) || (addressFlags [_to] ==
      ZERO_FEE_FLAG)
  @pre (_value > accounts[msg.sender])
  @post (__post.accounts[msg.sender] == accounts[msg.sender])
  @post (__post.accounts[_to] == accounts[_to])
  @post (__post.accounts[feeCollector] == accounts[feeCollector])
  @post __return
*/
/*@CTK "transfer maxVariableFee succeed"
  @tag spec
  @tag assume_completion
  @pre !frozen
  @pre (addressFlags[msg.sender] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
      BLACK_LIST_FLAG)
  @pre (addressFlags[msg.sender] != ZERO_FEE_FLAG) && (addressFlags [_to] !=
      ZERO_FEE_FLAG)
  @pre maxVariableFee > minVariableFee
  @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR > maxVariableFee
  @pre (_value + fixedFee + maxVariableFee) <= accounts[msg.sender]
  @post __post.accounts[msg.sender] == accounts[msg.sender] - (_value + fixedFee +
      maxVariableFee)
  @post __post.accounts[_to] == accounts[_to] + _value
  @post __post.accounts[feeCollector] == accounts[feeCollector] + (fixedFee +
      maxVariableFee)
  @post __return
*/
/*#CTK "transfer maxVariableFee fail"
  @tag spec
  @tag assume_completion
  @pre !frozen
  @pre (addressFlags[msg.sender] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
      BLACK_LIST_FLAG)
  @pre (addressFlags[msg.sender] != ZERO_FEE_FLAG) && (addressFlags [_to] !=
      ZERO_FEE_FLAG)
  @pre maxVariableFee > minVariableFee
  @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR > maxVariableFee
  @pre (_value + fixedFee + maxVariableFee) > accounts[msg.sender]
  @post __post.accounts[msg.sender] == accounts[msg.sender]
  @post __post.accounts[_to] == accounts[_to]
  @post __post.accounts[feeCollector] == accounts[feeCollector]
  @post !__return
*/
/*@CTK "transfer minVariableFee succeed"
  @tag spec
  @tag assume_completion
  @pre !frozen
  @pre (addressFlags[msg.sender] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
      BLACK_LIST_FLAG)
  @pre (addressFlags[msg.sender] != ZERO_FEE_FLAG) && (addressFlags [_to] !=
      ZERO_FEE_FLAG)
  @pre maxVariableFee > minVariableFee
  @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR < minVariableFee
  @pre (_value + fixedFee + minVariableFee) <= accounts[msg.sender]
  @post __post.accounts[msg.sender] == accounts[msg.sender] - (_value + fixedFee +
      minVariableFee)
  @post __post.accounts[_to] == accounts[_to] + _value
  @post __post.accounts[feeCollector] == accounts[feeCollector] + (fixedFee +
      minVariableFee)
  @post __return
*/
/*@CTK "transfer minVariableFee fail"
  @tag spec
  @tag assume_completion
  @pre !frozen
  @pre (addressFlags[msg.sender] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
      BLACK_LIST_FLAG)
  @pre (addressFlags[msg.sender] != ZERO_FEE_FLAG) && (addressFlags [_to] !=
      ZERO_FEE_FLAG)
  @pre maxVariableFee > minVariableFee
  @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR < minVariableFee
  @pre (_value + fixedFee + minVariableFee) > accounts[msg.sender]
  @post __post.accounts[msg.sender] == accounts[msg.sender]
  @post __post.accounts[_to] == accounts[_to]
  @post __post.accounts[feeCollector] == accounts[feeCollector]
  @post !__return
*/
/*@CTK "transfer normal variableFee succeed"
  @tag spec
  @tag assume_completion
  @pre !frozen
  @pre (addressFlags[msg.sender] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
      BLACK_LIST_FLAG)
  @pre (addressFlags[msg.sender] != ZERO_FEE_FLAG) && (addressFlags [_to] !=
      ZERO_FEE_FLAG)
  @pre maxVariableFee > minVariableFee
  @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR >= minVariableFee
  @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR <= maxVariableFee
  @pre (_value + fixedFee + (_value * variableFeeNumerator) / FEE_DENOMINATOR) <=
      accounts[msg.sender]
  @post __post.accounts[msg.sender] == accounts[msg.sender] - (_value + fixedFee + (
      _value * variableFeeNumerator) / FEE_DENOMINATOR)
  @post __post.accounts[_to] == accounts[_to] + _value
  @post __post.accounts[feeCollector] == accounts[feeCollector] + (fixedFee + (
      _value * variableFeeNumerator) / FEE_DENOMINATOR)
  @post __return
```

```
*/
/*@CTK "transfer normal variableFee fail"
  @tag spec
  @tag assume_completion
  @pre !frozen
  @pre (addressFlags[msg.sender] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
      BLACK_LIST_FLAG)
  @pre (addressFlags[msg.sender] != ZERO_FEE_FLAG) && (addressFlags [_to] !=
      ZERO_FEE_FLAG)
  @pre maxVariableFee > minVariableFee
  @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR >= minVariableFee
  @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR <= maxVariableFee
  @pre (_value + fixedFee + (_value * variableFeeNumerator) / FEE_DENOMINATOR) >
      accounts[msg.sender]
  @post __post.accounts[msg.sender] == accounts[msg.sender]
  @post __post.accounts[_to] == accounts[_to]
  @post __post.accounts[feeCollector] == accounts[feeCollector]
  @post !__return
*/
function transfer (address _to, uint256 _value)
public delegatable payable returns (bool) {
  if (frozen) return false;
  else if (
    (addressFlags [msg.sender] | addressFlags [_to]) & BLACK_LIST_FLAG ==
    BLACK_LIST_FLAG)
    return false;
  else {
    uint256 fee =
      (addressFlags [msg.sender] | addressFlags [_to]) & ZERO_FEE_FLAG ==
          ZERO_FEE_FLAG ?
        0 :
        calculateFee (_value);

    if (_value <= accounts [msg.sender] &&
        fee <= safeSub (accounts [msg.sender], _value)) {
      require (AbstractToken.transfer (_to, _value));
      require (AbstractToken.transfer (feeCollector, fee));
      return true;
    } else return false;
  }
}

/**
 * Transfer given number of tokens from given owner to given recipient.
 *
 * @param _from address to transfer tokens from the owner of
 * @param _to address to transfer tokens to the owner of
 * @param _value number of tokens to transfer from given owner to given
 *        recipient
 * @return true if tokens were transferred successfully, false otherwise
 */
/*@CTK "transferFrom frozen"
  @tag spec
  @tag assume_completion
  @pre frozen
  @post (__post.accounts[_from] == accounts[_from])
  @post (__post.accounts[_to] == accounts[_to])
  @post (__post.accounts[feeCollector] == accounts[feeCollector])
  @post (__post.allowances[_from][msg.sender] == allowances[_from][msg.sender])
  @post !__return
*/
/*@CTK "transferFrom blacklisted"
  @tag spec
  @tag assume_completion
  @pre !frozen
  @pre (addressFlags[_from] == BLACK_LIST_FLAG) || (addressFlags[_to] ==
      BLACK_LIST_FLAG)
  @post (__post.accounts[_from] == accounts[_from])
  @post (__post.accounts[_to] == accounts[_to])
  @post (__post.accounts[feeCollector] == accounts[feeCollector])
  @post (__post.allowances[_from][msg.sender] == allowances[_from][msg.sender])
  @post !__return
*/
/*@CTK "transferFrom zero-feed succeed"
  @tag spec
  @tag assume_completion
  @pre !frozen
  @pre (addressFlags[_from] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
      BLACK_LIST_FLAG)
  @pre (addressFlags[_from] == ZERO_FEE_FLAG) || (addressFlags [_to] ==
      ZERO_FEE_FLAG)
  @pre (_value <= accounts[_from]) && (_value <= allowances [_from][msg.sender])
  @post (__post.accounts[_from] == accounts[_from] - _value)
  @post (__post.accounts[_to] == accounts[_to] + _value)
  @post (__post.accounts[feeCollector] == accounts[feeCollector])
  @post (__post.allowances[_from][msg.sender] == allowances[_from][msg.sender] -
      _value)
  @post __return
*/
/*@CTK "transferFrom zero-feed fail"
  @tag spec
  @tag assume_completion
  @pre !frozen
  @pre (addressFlags[_from] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
      BLACK_LIST_FLAG)
  @pre (addressFlags[_from] == ZERO_FEE_FLAG) || (addressFlags [_to] ==
      ZERO_FEE_FLAG)
  @pre (_value > accounts[_from]) || (_value > allowances [_from][msg.sender])
  @post (__post.accounts[_from] == accounts[_from])
  @post (__post.accounts[_to] == accounts[_to])
  @post (__post.accounts[feeCollector] == accounts[feeCollector])
  @post (__post.allowances[_from][msg.sender] == allowances[_from][msg.sender])
  @post __return
*/
/*@CTK "transferFrom maxVariableFee succeed"
  @tag spec
  @tag assume_completion
  @pre !frozen
  @pre (addressFlags[_from] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
      BLACK_LIST_FLAG)
  @pre (addressFlags[_from] != ZERO_FEE_FLAG) && (addressFlags [_to] !=
      ZERO_FEE_FLAG)
  @pre maxVariableFee > minVariableFee
  @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR > maxVariableFee
  @pre (_value + fixedFee + maxVariableFee) <= accounts[_from] && (_value + fixedFee
       + maxVariableFee) <= allowances[_from][msg.sender]
```

```
    @post __post.accounts[_from] == accounts[_from] - (_value + fixedFee +
        maxVariableFee)
    @post __post.accounts[_to] == accounts[_to] + _value
    @post __post.accounts[feeCollector] == accounts[feeCollector] + (fixedFee +
        maxVariableFee)
    @post __post.allowances[_from][msg.sender] == allowances[_from][msg.sender] - (
        _value + fixedFee + maxVariableFee)
    @post __return
  */
  /*@CTK "transferFrom maxVariableFee fail"
    @tag spec
    @tag assume_completion
    @pre !frozen
    @pre (addressFlags[_from] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
        BLACK_LIST_FLAG)
    @pre (addressFlags[_from] != ZERO_FEE_FLAG) && (addressFlags [_to] !=
        ZERO_FEE_FLAG)
    @pre maxVariableFee > minVariableFee
    @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR > maxVariableFee
    @pre (_value + fixedFee + maxVariableFee) > accounts[_from] || (_value + fixedFee
        + maxVariableFee) > allowances [_from][msg.sender]
    @post __post.accounts[_from] == accounts[_from]
    @post __post.accounts[_to] == accounts[_to]
    @post __post.accounts[feeCollector] == accounts[feeCollector]
    @post __post.allowances[_from][msg.sender] == allowances[_from][msg.sender]
    @post !__return
  */
  /*@CTK "transferFrom minVariableFee succeed"
    @tag spec
    @tag assume_completion
    @pre !frozen
    @pre (addressFlags[_from] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
        BLACK_LIST_FLAG)
    @pre (addressFlags[_from] != ZERO_FEE_FLAG) && (addressFlags [_to] !=
        ZERO_FEE_FLAG)
    @pre maxVariableFee > minVariableFee
    @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR < minVariableFee
    @pre (_value + fixedFee + minVariableFee) <= accounts[_from] && (_value + fixedFee
         + minVariableFee) <= allowances [_from][msg.sender]
    @post __post.accounts[_from] == accounts[_from] - (_value + fixedFee +
        minVariableFee)
    @post __post.accounts[_to] == accounts[_to] + _value
    @post __post.accounts[feeCollector] == accounts[feeCollector] + (fixedFee +
        minVariableFee)
    @post __post.allowances[_from][msg.sender] == allowances[_from][msg.sender] - (
        _value + fixedFee + minVariableFee)
    @post __return
  */
  /*@CTK "transferFrom minVariableFee fail"
    @tag spec
    @tag assume_completion
    @pre !frozen
    @pre (addressFlags[_from] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
        BLACK_LIST_FLAG)
    @pre (addressFlags[_from] != ZERO_FEE_FLAG) && (addressFlags [_to] !=
        ZERO_FEE_FLAG)
    @pre maxVariableFee > minVariableFee
    @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR < minVariableFee
    @pre (_value + fixedFee + minVariableFee) > accounts[_from] || (_value + fixedFee
        + minVariableFee) > allowances [_from][msg.sender]
    @post __post.accounts[_from] == accounts[_from]
    @post __post.accounts[_to] == accounts[_to]
    @post __post.accounts[feeCollector] == accounts[feeCollector]
    @post __post.allowances[_from][msg.sender] == allowances[_from][msg.sender]
    @post !__return
  */
  /*@CTK "transferFrom normal variableFee succeed"
    @tag spec
    @tag assume_completion
    @pre !frozen
    @pre (addressFlags[_from] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
        BLACK_LIST_FLAG)
    @pre (addressFlags[_from] != ZERO_FEE_FLAG) && (addressFlags [_to] !=
        ZERO_FEE_FLAG)
    @pre maxVariableFee > minVariableFee
    @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR >= minVariableFee
    @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR <= maxVariableFee
    @pre (_value + fixedFee + (_value * variableFeeNumerator) / FEE_DENOMINATOR) <=
        accounts[_from] && (_value + fixedFee + (_value * variableFeeNumerator) /
        FEE_DENOMINATOR) <= allowances [_from][msg.sender]
    @post __post.accounts[_from] == accounts[_from] - (_value + fixedFee + (_value *
        variableFeeNumerator) / FEE_DENOMINATOR)
    @post __post.accounts[_to] == accounts[_to] + _value
    @post __post.accounts[feeCollector] == accounts[feeCollector] + (fixedFee + (
        _value * variableFeeNumerator) / FEE_DENOMINATOR)
    @post __post.allowances[_from][msg.sender] == allowances[_from][msg.sender] - (
        _value + fixedFee + (_value * variableFeeNumerator) / FEE_DENOMINATOR)
    @post __return
  */
  /*@CTK "transferFrom normal variableFee fail"
    @tag spec
    @tag assume_completion
    @pre !frozen
    @pre (addressFlags[_from] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
        BLACK_LIST_FLAG)
    @pre (addressFlags[_from] != ZERO_FEE_FLAG) && (addressFlags [_to] !=
        ZERO_FEE_FLAG)
    @pre maxVariableFee > minVariableFee
    @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR >= minVariableFee
    @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR <= maxVariableFee
    @pre (_value + fixedFee + (_value * variableFeeNumerator) / FEE_DENOMINATOR) >
        accounts[_from] || (_value + fixedFee + (_value * variableFeeNumerator) /
        FEE_DENOMINATOR) > allowances [_from][msg.sender]
    @post __post.accounts[_from] == accounts[_from]
    @post __post.accounts[_to] == accounts[_to]
    @post __post.accounts[feeCollector] == accounts[feeCollector]
    @post __post.allowances[_from][msg.sender] == allowances[_from][msg.sender]
    @post !__return
  */
  function transferFrom (address _from, address _to, uint256 _value)
  public /*>IGNORE delegatable IGNORE<*/ payable returns (bool) {
    if (frozen) return false;
    else if (
      (addressFlags [_from] | addressFlags [_to]) & BLACK_LIST_FLAG ==
      BLACK_LIST_FLAG)
      return false;
    else {
      uint256 fee =
        (addressFlags [_from] | addressFlags [_to]) & ZERO_FEE_FLAG == ZERO_FEE_FLAG ?
          0 :
          calculateFee (_value);

      if (_value <= allowances [_from][msg.sender] &&
          fee <= safeSub (allowances [_from][msg.sender], _value) &&
          _value <= accounts [_from] &&
          fee <= safeSub (accounts [_from], _value)) {
        require (AbstractToken.transferFrom (_from, _to, _value));
        require (AbstractToken.transferFrom (_from, feeCollector, fee));
        return true;
      } else return false;
    }
  }

  /**
   * Allow given spender to transfer given number of tokens from message sender.
   *
   * @param _spender address to allow the owner of to transfer tokens from
   *        message sender
   * @param _value number of tokens to allow to transfer
   * @return true if token transfer was successfully approved, false otherwise
   */
  //@CTK NO_OVERFLOW
  //@CTK NO_BUF_OVERFLOW
  //@CTK NO_ASF
  /*@CTK "EURSToken approve"
    @tag spec
    @tag assume_completion
    @post (__post.allowances[msg.sender][_spender]) == (_value)
  */
  function approve (address _spender, uint256 _value)
  public /*>IGNORE delegatable IGNORE<*/ payable returns (bool success) {
    return AbstractToken.approve (_spender, _value);
  }

  /**
   * Tell how many tokens given spender is currently allowed to transfer from
   * given owner.
   *
   * @param _owner address to get number of tokens allowed to be transferred
   *        from the owner of
   * @param _spender address to get number of tokens allowed to be transferred
   *        by the owner of
   * @return number of tokens given spender is currently allowed to transfer
   *         from given owner
   */
  //@CTK NO_OVERFLOW
  //@CTK NO_BUF_OVERFLOW
  //@CTK NO_ASF
  /*@CTK "EURSToken allowance"
    @tag spec
    @tag assume_completion
    @post (remaining) == (allowances[_owner][_spender])
  */
  function allowance (address _owner, address _spender)
```

```
public /*>IGNORE delegatable IGNORE<*/ view returns (uint256 remaining) {
  return AbstractToken.allowance (_owner, _spender);
}

/**
 * Transfer given number of token from the signed defined by digital signature
 * to given recipient.
 *
 * @param _to address to transfer token to the owner of
 * @param _value number of tokens to transfer
 * @param _fee number of tokens to give to message sender
 * @param _nonce nonce of the transfer
 * @param _v parameter V of digital signature
 * @param _r parameter R of digital signature
 * @param _s parameter S of digital signature
 */
/*@CTK "delegatedTransfer frozen"
  @tag spec
  @tag assume_completion
  @pre frozen
  @post (__post.accounts[_from] == accounts[_from])
  @post (__post.accounts[_to] == accounts[_to])
  @post (__post.accounts[feeCollector] == accounts[feeCollector])
  @post (__post.accounts[msg.sender] == accounts[msg.sender])
  @post __post._nonce[_from] == nonces[_from]
  @post !__return
*/
/*@CTK "delegatedTransfer signature reused"
  @tag spec
  @tag assume_completion
  @pre !frozen
  @pre _nonce != nonces [_from]
  @pre (addressFlags[_from] == BLACK_LIST_FLAG) || (addressFlags[_to] ==
      BLACK_LIST_FLAG)
  @post (__post.accounts[_from] == accounts[_from])
  @post (__post.accounts[_to] == accounts[_to])
  @post (__post.accounts[feeCollector] == accounts[feeCollector])
  @post (__post.accounts[msg.sender] == accounts[msg.sender])
  @post __post._nonce[_from] == nonces[_from]
  @post !__return
*/
/*@CTK "delegatedTransfer blacklisted"
  @tag spec
  @tag assume_completion
  @pre !frozen
  @pre _nonce == nonces [_from]
  @pre (addressFlags[_from] == BLACK_LIST_FLAG) || (addressFlags[_to] ==
      BLACK_LIST_FLAG)
  @post (__post.accounts[_from] == accounts[_from])
  @post (__post.accounts[_to] == accounts[_to])
  @post (__post.accounts[feeCollector] == accounts[feeCollector])
  @post (__post.accounts[msg.sender] == accounts[msg.sender])
  @post __post._nonce[_from] == nonces[_from]
  @post !__return
*/
/*@CTK "delegatedTransfer zero-feed succeed"
  @tag spec
  @tag assume_completion
```

```
    @pre !frozen
    @pre _nonce == nonces [_from]
    @pre (addressFlags[_from] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
        BLACK_LIST_FLAG)
    @pre (addressFlags[_from] == ZERO_FEE_FLAG) || (addressFlags [_to] ==
        ZERO_FEE_FLAG)
    @pre (_value + _fee <= accounts[_from])
    @post (__post.accounts[_from] == accounts[_from] - _value - _fee)
    @post (__post.accounts[_to] == accounts[_to] + _value + _fee)
    @post (__post.accounts[feeCollector] == accounts[feeCollector])
    @post (__post.accounts[msg.sender] == accounts[msg.sender] + _fee)
    @post __post._nonce[_from] == nonces[_from] + 1
    @post __return
  */
  /*@CTK "delegatedTransfer zero-feed fail"
    @tag spec
    @tag assume_completion
    @pre !frozen
    @pre _nonce == nonces [_from]
    @pre (addressFlags[_from] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
        BLACK_LIST_FLAG)
    @pre (addressFlags[_from] == ZERO_FEE_FLAG) || (addressFlags [_to] ==
        ZERO_FEE_FLAG)
    @pre (_value + _fee > accounts[_from])
    @post (__post.accounts[_from] == accounts[_from])
    @post (__post.accounts[_to] == accounts[_to])
    @post (__post.accounts[feeCollector] == accounts[feeCollector])
    @post (__post.accounts[msg.sender] == accounts[msg.sender])
    @post __post._nonce[_from] == nonces[_from]
    @post __return
  */
  /*@CTK "delegatedTransfer maxVariableFee succeed"
    @tag spec
    @tag assume_completion
    @pre !frozen
    @pre _nonce == nonces [_from]
    @pre (addressFlags[_from] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
        BLACK_LIST_FLAG)
    @pre (addressFlags[_from] != ZERO_FEE_FLAG) && (addressFlags [_to] !=
        ZERO_FEE_FLAG)
    @pre maxVariableFee > minVariableFee
    @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR > maxVariableFee
    @pre (_value + fixedFee + maxVariableFee + _fee) <= accounts[_from]
    @post __post.accounts[_from] == accounts[_from] - (_value + fixedFee +
        maxVariableFee + _fee)
    @post __post.accounts[_to] == accounts[_to] + _value
    @post __post.accounts[feeCollector] == accounts[feeCollector] + (fixedFee +
        maxVariableFee)
    @post (__post.accounts[msg.sender] == accounts[msg.sender] + _fee)
    @post __post._nonce[_from] == nonces[_from] + 1
    @post __return
  */
  /*@CTK "delegatedTransfer maxVariableFee fail"
    @tag spec
    @tag assume_completion
    @pre !frozen
    @pre _nonce == nonces [_from]
    @pre (addressFlags[_from] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
        BLACK_LIST_FLAG)
    @pre (addressFlags[_from] != ZERO_FEE_FLAG) && (addressFlags [_to] !=
        ZERO_FEE_FLAG)
    @pre maxVariableFee > minVariableFee
    @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR > maxVariableFee
    @pre (_value + fixedFee + maxVariableFee + _fee) > accounts[_from]
    @post __post.accounts[_from] == accounts[_from]
    @post __post.accounts[_to] == accounts[_to]
    @post __post.accounts[feeCollector] == accounts[feeCollector]
    @post (__post.accounts[msg.sender] == accounts[msg.sender])
    @post __post._nonce[_from] == nonces[_from]
    @post !__return
  */
  /*@CTK "delegatedTransfer minVariableFee succeed"
    @tag spec
    @tag assume_completion
    @pre !frozen
    @pre _nonce == nonces [_from]
    @pre (addressFlags[_from] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
        BLACK_LIST_FLAG)
    @pre (addressFlags[_from] != ZERO_FEE_FLAG) && (addressFlags [_to] !=
        ZERO_FEE_FLAG)
    @pre maxVariableFee > minVariableFee
    @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR < minVariableFee
    @pre (_value + fixedFee + minVariableFee + _fee) <= accounts[_from]
    @post __post.accounts[_from] == accounts[_from] - (_value + fixedFee +
        minVariableFee + _fee)
    @post __post.accounts[_to] == accounts[_to] + _value
    @post __post.accounts[feeCollector] == accounts[feeCollector] + (fixedFee +
        minVariableFee)
    @post (__post.accounts[msg.sender] == accounts[msg.sender] + _fee)
    @post __post._nonce[_from] == nonces[_from] + 1
    @post __return
  */
  /*@CTK "delegatedTransfer minVariableFee fail"
    @tag spec
    @tag assume_completion
    @pre !frozen
    @pre _nonce == nonces [_from]
    @pre (addressFlags[_from] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
        BLACK_LIST_FLAG)
    @pre (addressFlags[_from] != ZERO_FEE_FLAG) && (addressFlags [_to] !=
        ZERO_FEE_FLAG)
    @pre maxVariableFee > minVariableFee
    @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR < minVariableFee
    @pre (_value + fixedFee + minVariableFee + _fee) > accounts[_from]
    @post __post.accounts[_from] == accounts[_from]
    @post __post.accounts[_to] == accounts[_to]
    @post __post.accounts[feeCollector] == accounts[feeCollector]
    @post (__post.accounts[msg.sender] == accounts[msg.sender])
    @post __post._nonce[_from] == nonces[_from]
    @post !__return
  */
  /*@CTK "delegatedTransfer normal variableFee succeed"
    @tag spec
    @tag assume_completion
    @pre !frozen
    @pre _nonce == nonces [_from]
    @pre (addressFlags[_from] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
        BLACK_LIST_FLAG)
    @pre (addressFlags[_from] != ZERO_FEE_FLAG) && (addressFlags [_to] !=
        ZERO_FEE_FLAG)
    @pre maxVariableFee > minVariableFee
    @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR >= minVariableFee
    @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR <= maxVariableFee
    @pre (_value + fixedFee + (_value * variableFeeNumerator) / FEE_DENOMINATOR + _fee
        ) <= accounts[_from]
    @post __post.accounts[_from] == accounts[_from] - (_value + fixedFee + (_value *
        variableFeeNumerator) / FEE_DENOMINATOR + _fee)
    @post __post.accounts[_to] == accounts[_to] + _value
    @post __post.accounts[feeCollector] == accounts[feeCollector] + (fixedFee + (
        _value * variableFeeNumerator) / FEE_DENOMINATOR)
    @post (__post.accounts[msg.sender] == accounts[msg.sender] + _fee)
    @post __post._nonce[_from] == nonces[_from] + 1
    @post __return
  */
  /*@CTK "delegatedTransfer normal variableFee fail"
    @tag spec
    @tag assume_completion
    @pre !frozen
    @pre _nonce == nonces [_from]
    @pre (addressFlags[_from] != BLACK_LIST_FLAG) && (addressFlags [_to] !=
        BLACK_LIST_FLAG)
    @pre (addressFlags[_from] != ZERO_FEE_FLAG) && (addressFlags [_to] !=
        ZERO_FEE_FLAG)
    @pre maxVariableFee > minVariableFee
    @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR >= minVariableFee
    @pre (_value * variableFeeNumerator) / FEE_DENOMINATOR <= maxVariableFee
    @pre (_value + fixedFee + (_value * variableFeeNumerator) / FEE_DENOMINATOR + _fee
        ) > accounts[_from]
    @post __post.accounts[_from] == accounts[_from]
    @post __post.accounts[_to] == accounts[_to]
    @post __post.accounts[feeCollector] == accounts[feeCollector]
    @post (__post.accounts[msg.sender] == accounts[msg.sender])
    @post __post._nonce[_from] == nonces[_from]
    @post !__return
  */
  function delegatedTransfer (
    address _to, uint256 _value, uint256 _fee,
    uint256 _nonce, uint8 _v, bytes32 _r, bytes32 _s)
  public delegatable payable returns (bool) {
    if (frozen) return false;
    else {
      address _from = ecrecover (
        keccak256 (
          thisAddress (), messageSenderAddress (), _to, _value, _fee, _nonce),
        _v, _r, _s);

      if (_nonce != nonces [_from]) return false;

      if (
        (addressFlags [_from] | addressFlags [_to]) & BLACK_LIST_FLAG ==
        BLACK_LIST_FLAG)
        return false;

      uint256 fee =
        (addressFlags [_from] | addressFlags [_to]) & ZERO_FEE_FLAG == ZERO_FEE_FLAG ?
          0 :
          calculateFee (_value);

      uint256 balance = accounts [_from];
      if (_value > balance) return false;
      balance = safeSub (balance, _value);
      if (fee > balance) return false;
      balance = safeSub (balance, fee);
      if (_fee > balance) return false;
      balance = safeSub (balance, _fee);

      nonces [_from] = _nonce + 1;

      accounts [_from] = balance;
      accounts [_to] = safeAdd (accounts [_to], _value);
      accounts [feeCollector] = safeAdd (accounts [feeCollector], fee);
      accounts [msg.sender] = safeAdd (accounts [msg.sender], _fee);

      Transfer (_from, _to, _value);
      Transfer (_from, feeCollector, fee);
      Transfer (_from, msg.sender, _fee);

      return true;
    }
  }

  /**
   * Create tokens.
   *
   * @param _value number of tokens to be created.
   */
  //@CTK FAIL NO_OVERFLOW
  //@CTK NO_BUF_OVERFLOW
  //@CTK FAIL NO_ASF
  /*@CTK "EURSToken createTokens"
    @tag assume_completion
    @pre msg.sender == owner
    @post ((_value > 0) && (_value > MAX_TOKENS_COUNT - tokensCount)) -> !__return
    @post ((_value == 0) || (_value <= MAX_TOKENS_COUNT - tokensCount)) -> __return
    @post ((_value == 0) && (_value <= MAX_TOKENS_COUNT - tokensCount)) -> (__post.
        accounts[msg.sender] == accounts[msg.sender] + _value) && (__post.tokensCount
        == tokensCount + _value)
  */
  function createTokens (uint256 _value)
  public /*>IGNORE delegatable IGNORE<*/ payable returns (bool) {
    require (msg.sender == owner);

    if (_value > 0) {
      if (_value <= safeSub (MAX_TOKENS_COUNT, tokensCount)) {
        accounts [msg.sender] = safeAdd (accounts [msg.sender], _value);
        tokensCount = safeAdd (tokensCount, _value);

        Transfer (address (0), msg.sender, _value);

        return true;
      } else return false;
    } else return true;
  }

  /**
   * Burn tokens.
   *
   * @param _value number of tokens to burn
   */
  //@CTK NO_OVERFLOW
  //@CTK NO_BUF_OVERFLOW
  //@CTK FAIL NO_ASF
  /*@CTK "EURSToken burnTokens"
    @tag assume_completion
    @pre msg.sender == owner
    @post ((_value > 0) && (_value > accounts [msg.sender])) -> !__return
    @post ((_value == 0) || (_value <= accounts [msg.sender])) -> __return
    @post ((_value == 0) || (_value <= accounts [msg.sender])) -> (__post.accounts[msg
        .sender] == accounts[msg.sender] - _value) && (__post.tokensCount ==
        tokensCount - _value)
  */
  function burnTokens (uint256 _value)
  public /*>IGNORE delegatable IGNORE<*/ payable returns (bool) {
    require (msg.sender == owner);

    if (_value > 0) {
      if (_value <= accounts [msg.sender]) {
        accounts [msg.sender] = safeSub (accounts [msg.sender], _value);
        tokensCount = safeSub (tokensCount, _value);

        Transfer (msg.sender, address (0), _value);

        return true;
      } else return false;
    } else return true;
  }

  /**
   * Freeze token transfers.
   */
  //@CTK NO_OVERFLOW
  //@CTK NO_BUF_OVERFLOW
  //@CTK NO_ASF
  /*@CTK "freezeTransfers"
    @tag assume_completion
    @pre msg.sender == owner
    @post __post.frozen
   */
  function freezeTransfers () public /*>IGNORE delegatable IGNORE<*/ payable {
    require (msg.sender == owner);

    if (!frozen) {
      frozen = true;

      Freeze ();
    }
  }

  /**
```

```
   * Unfreeze token transfers.
   */
  //@CTK NO_OVERFLOW
  //@CTK NO_BUF_OVERFLOW
  //@CTK NO_ASF
  /*@CTK "unfreezeTransfers"
    @tag assume_completion
    @pre msg.sender == owner
    @post __post.frozen == false
   */
  function unfreezeTransfers () public /*>IGNORE delegatable IGNORE<*/ payable {
    require (msg.sender == owner);

    if (frozen) {
      frozen = false;

      Unfreeze ();
    }
  }

  /**
   * Set smart contract owner.
   *
   * @param _newOwner address of the new owner
   */
  //@CTK NO_OVERFLOW
  //@CTK NO_BUF_OVERFLOW
  //@CTK NO_ASF
  /*@CTK "unfreezeTransfers"
    @tag assume_completion
    @post msg.sender == owner
    @post __post.owner == _newOwner
   */
  function setOwner (address _newOwner) public {
    require (msg.sender == owner);

    owner = _newOwner;
  }

  /**
   * Set fee collector.
   *
   * @param _newFeeCollector address of the new fee collector
   */
  //@CTK NO_OVERFLOW
  //@CTK NO_BUF_OVERFLOW
  //@CTK NO_ASF
  /*@CTK "setFeeCollector"
    @tag assume_completion
    @pre msg.sender == owner
    @post (__post.feeCollector) == (_newFeeCollector)
  */
  function setFeeCollector (address _newFeeCollector)
  public /*>IGNORE delegatable IGNORE<*/ payable {
    require (msg.sender == owner);

    feeCollector = _newFeeCollector;
  }
```

```

/**
 * Get current nonce for token holder with given address, i.e. nonce this
 * token holder should use for next delegated transfer.
 *
 * @param _owner address of the token holder to get nonce for
 * @return current nonce for token holder with give address
 */
//@CTK NO_OVERFLOW
//@CTK NO_BUF_OVERFLOW
//@CTK NO_ASF
/*@CTK nonce
  @tag assume_completion
  @post __return == nonces [_owner]
*/
function nonce (address _owner) public view /*>IGNORE delegatable IGNORE<*/ returns
    (uint256) {
  return nonces [_owner];
}

/**
 * Set fee parameters.
 *
 * @param _fixedFee fixed fee in token units
 * @param _minVariableFee minimum variable fee in token units
 * @param _maxVariableFee maximum variable fee in token units
 * @param _variableFeeNumerator variable fee numerator
 */
//@CTK NO_OVERFLOW
//@CTK NO_BUF_OVERFLOW
//@CTK NO_ASF
/*@CTK setFeeCollector
  @tag assume_completion
  @pre msg.sender == owner
  @pre _minVariableFee <= _maxVariableFee
  @pre _variableFeeNumerator <= MAX_FEE_NUMERATOR
  @post (__post.fixedFee) == (_fixedFee)
  @post (__post.minVariableFee) == (_minVariableFee)
  @post (__post.maxVariableFee) == (_maxVariableFee)
  @post (__post.variableFeeNumerator) == (_variableFeeNumerator)
*/
function setFeeParameters (
  uint256 _fixedFee,
  uint256 _minVariableFee,
  uint256 _maxVariableFee,
  uint256 _variableFeeNumerator) public /*>IGNORE delegatable IGNORE<*/ payable {
  require (msg.sender == owner);

  require (_minVariableFee <= _maxVariableFee);
  require (_variableFeeNumerator <= MAX_FEE_NUMERATOR);

  fixedFee = _fixedFee;
  minVariableFee = _minVariableFee;
  maxVariableFee = _maxVariableFee;
  variableFeeNumerator = _variableFeeNumerator;

  FeeChange (
    _fixedFee, _minVariableFee, _maxVariableFee, _variableFeeNumerator);
}

/**
 * Get fee parameters.
 *
 * @return fee parameters
 */
//@CTK NO_OVERFLOW
//@CTK NO_BUF_OVERFLOW
//@CTK NO_ASF
/*@CTK getFeeParameters
  @tag assume_completion
  @post _fixedFee == fixedFee
  @post _minVariableFee == minVariableFee
  @post _maxVariableFee == maxVariableFee
  @post _variableFeeNumnerator == variableFeeNumerator
*/
function getFeeParameters () public /*>IGNORE delegatable IGNORE<*/ view returns (
  uint256 _fixedFee,
  uint256 _minVariableFee,
  uint256 _maxVariableFee,
  uint256 _variableFeeNumnerator) {
  _fixedFee = fixedFee;
  _minVariableFee = minVariableFee;
  _maxVariableFee = maxVariableFee;
  _variableFeeNumnerator = variableFeeNumerator;
}

/**
 * Calculate fee for transfer of given number of tokens.
 *
 * @param _amount transfer amount to calculate fee for
 * @return fee for transfer of given amount
 */
//@CTK FAIL NO_OVERFLOW
//@CTK NO_BUF_OVERFLOW
//@CTK FAIL NO_ASF
/*@CTK calculateFee
  @tag assume_completion
  @pre _amount <= MAX_TOKENS_COUNT
  @pre maxVariableFee > minVariableFee
  @post (_amount * variableFeeNumerator) / FEE_DENOMINATOR > maxVariableFee -> _fee
      == fixedFee + maxVariableFee
  @post (_amount * variableFeeNumerator) / FEE_DENOMINATOR < minVariableFee -> _fee
      == fixedFee + minVariableFee
  @post ((_amount * variableFeeNumerator) / FEE_DENOMINATOR >= minVariableFee && (
      _amount * variableFeeNumerator) / FEE_DENOMINATOR <= maxVariableFee) -> _fee
      == fixedFee + (_amount * variableFeeNumerator) / FEE_DENOMINATOR
*/
function calculateFee (uint256 _amount)
  public /*>IGNORE delegatable IGNORE<*/ view returns (uint256 _fee) {
  require (_amount <= MAX_TOKENS_COUNT);

  _fee = safeMul (_amount, variableFeeNumerator) / FEE_DENOMINATOR;
  if (_fee < minVariableFee) _fee = minVariableFee;
  if (_fee > maxVariableFee) _fee = maxVariableFee;
  _fee = safeAdd (_fee, fixedFee);
}
```

```

  /**
   * Set flags for given address.
   *
   * @param _address address to set flags for
   * @param _flags flags to set
   */
  //@CTK NO_OVERFLOW
  //@CTK NO_BUF_OVERFLOW
  //@CTK NO_ASF
  /*@CTK setFlags
    @tag assume_completion
    @pre msg.sender == owner
    @post __post.addressFlags [_address] == _flags
   */
  function setFlags (address _address, uint256 _flags)
  public /*>IGNORE delegatable IGNORE<*/ payable {
    require (msg.sender == owner);

    addressFlags [_address] = _flags;
  }

  /**
   * Get flags for given address.
   *
   * @param _address address to get flags for
   * @return flags for given address
   */
  function flags (address _address) public delegatable view returns (uint256) {
    return addressFlags [_address];
  }

  /**
   * Set address of smart contract to delegate execution of delegatable methods
   * to.
   *
   * @param _delegate address of smart contract to delegate execution of
   * delegatable methods to, or zero to not delegate delegatable methods
   * execution.
   */
  //@CTK NO_OVERFLOW
  //@CTK NO_BUF_OVERFLOW
  //@CTK NO_ASF
  /*@CTK "setDelegate"
    @tag assume_completion
    @pre msg.sender == owner
    @post __post.delegate == _delegate
   */
  function setDelegate (address _delegate) public {
    require (msg.sender == owner);

    if (delegate != _delegate) {
      delegate = _delegate;
      Delegation (delegate);
    }
  }

  /**
```

```
   * Get address of this smart contract.
   *
   * @return address of this smart contract
   */
  function thisAddress () internal view returns (address) {
    return this;
  }

  /**
   * Get address of message sender.
   *
   * @return address of this smart contract
   */
  function messageSenderAddress () internal view returns (address) {
    return msg.sender;
  }

  /**
   * Owner of the smart contract.
   */
  address internal owner;

  /**
   * Address where fees are sent to.
   */
  address internal feeCollector;

  /**
   * Number of tokens in circulation.
   */
  uint256 internal tokensCount;

  /**
   * Whether token transfers are currently frozen.
   */
  bool internal frozen;

  /**
   * Mapping from sender's address to the next delegated transfer nonce.
   */
  mapping (address => uint256) internal nonces;

  /**
   * Fixed fee amount in token units.
   */
  uint256 internal fixedFee;

  /**
   * Minimum variable fee in token units.
   */
  uint256 internal minVariableFee;

  /**
   * Maximum variable fee in token units.
   */
  uint256 internal maxVariableFee;

  /**
```

```
   * Variable fee numerator.
   */
  uint256 internal variableFeeNumerator;

  /**
   * Maps address to its flags.
   */
  mapping (address => uint256) internal addressFlags;

  /**
   * Address of smart contract to delegate execution of delegatable methods to,
   * or zero to not delegate delegatable methods execution.
   */
  address internal delegate;

  /**
   * Logged when token transfers were frozen.
   */
  event Freeze ();

  /**
   * Logged when token transfers were unfrozen.
   */
  event Unfreeze ();

  /**
   * Logged when fee parameters were changed.
   *
   * @param fixedFee fixed fee in token units
   * @param minVariableFee minimum variable fee in token units
   * @param maxVariableFee maximum variable fee in token units
   * @param variableFeeNumerator variable fee numerator
   */
  event FeeChange (
    uint256 fixedFee,
    uint256 minVariableFee,
    uint256 maxVariableFee,
    uint256 variableFeeNumerator);

  /**
   * Logged when address of smart contract execution of delegatable methods is
   * delegated to was changed.
   *
   * @param delegate new address of smart contract execution of delegatable
   * methods is delegated to or zero if execution of delegatable methods is
   * oot delegated.
   */
  event Delegation (address delegate);
}
```