CERTIK

Audit Report

Produced by CertiK

for MONETA DIGITAL

June 18, 2021

CertiK Audit Report For Moneta Digital

MMXN

Request Date: 2021-06-18
Revision Date: 2021-06-18
Platform Name: Ethereum

CERTIK

Contents

Disclaimer

This report is subject to the terms and conditions (including without limitation, description of services, confidentiality, disclaimer and limitation of liability) set forth in the Verification Services Agreement between CertiK and Moneta Digital (the “Company”), or the scope of services/verification, and terms and conditions provided to the Company in connection with the verification (collectively, the “Agreement”). This report provided in connection with the Services set forth in the Agreement shall be used by the Company only to the extent permitted under the terms and conditions set forth in the Agreement. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes without CertiK’s prior written consent.

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 6.2B in assets.

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

Executive Summary

This report has been prepared for Moneta Digital to discover issues and vulnerabilities in the source code of their mmxn smart contracts. A comprehensive examination has been performed, utilizing CertiK's Formal Verification Platform, Static Analysis, and Manual Review techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors.
- Assessing the codebase to ensure compliance with current best practices 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

CertiK categorizes issues into three buckets based on overall risk levels:

Critical

Code implementation does not match specification, which could result in the loss of funds for contract owner or users.

Medium

Code implementation does not match the specification under certain conditions, which could affect the security standard by loss of access control.

Low

Code implementation does not follow best practices, or uses suboptimal design patterns, which could lead to security vulnerabilities further down the line.

Testing Summary

PASS

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

Jun 18, 2021

Type of Issues

CertiK's smart label engine applied 100% formal verification coverage on the source code. Our team of engineers has scanned the source code using proprietary static analysis tools and code-review methodologies. The following technical issues were found:

Title	Description	Issues	SWC ID
Integer Overflow/ Underflow	An overflow/underflow occurs when an arithmetic operation reaches the maximum or minimum size of a type.	0	SWC-101
Function Incorrectness	Function implementation does not meet specification, leading to intentional or unintentional vulnerabilities.	0	
Buffer Overflow	An attacker can 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 miners to some degree.	0	SWC-116
Insecure Compiler Version	Using a 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	Using block attributes to generate random numbers is unreliable, as they can be influenced by miners 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

Title	Description	Issues	SWC ID
Delegatecall to Untrusted Callee	Calling untrusted contracts is very dangerous, so the target and arguments provided must be sanitized.	0	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, meaning a malicious user can 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.	0	SWC-111
Unused Variables	Unused variables reduce code quality	0	SWC-131

Vulnerability Details

Critical

No issue found.

Medium

No issue found.

Low

No issue found.

Review Notes

Source Code SHA-256 Checksum

- **[mmxn].sol**
- **(https://etherscan.io/address/0x95c2e7cbc7ae370e28160bd04297c53f96d092b4)**
  `91b0dcf712f24b19b427e3ad075960c4d44d3c1478bcac0ab1a3b9e285e1637c`

Summary

CertiK worked closely with Moneta Digital to audit the design and implementation of its soon-to-be released smart contract. To ensure comprehensive protection, the source code was 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 best practices.

Our client Moneta Digital has demonstrated their professional and knowledgeable understanding of the project Moneta MMXN stablecoin, by having 1) a production ready repository with high-quality source code; 2) unit tests covering the majority of its business scenarios; 3) accessible, clean, and accurate readme documents for intentions, functionalities, and responsibilities of the smart contracts.

Overall, we found Moneta Digital's 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, continually improve the codebase, and perform additional tests before the mainnet release.

To bridge the trust gap between administrator and users, administrator needs to express a sincere attitude with the consideration of the administrator team's anonymousness. The administrator has the responsibility to notify users with the following capability of the administrator:
. Initial minter and pauser is the contract deployer.
. Minter has privilege to add a minter.
. Minter has privilege to mint tokens to any account.
. Pauser has privilege to add a pauser.
. Pauser has privilege to pause user's transaction.
. Pauser has privilege to unpause user's transaction.

Static Analysis Results

INSECURE_COMPILER_VERSION

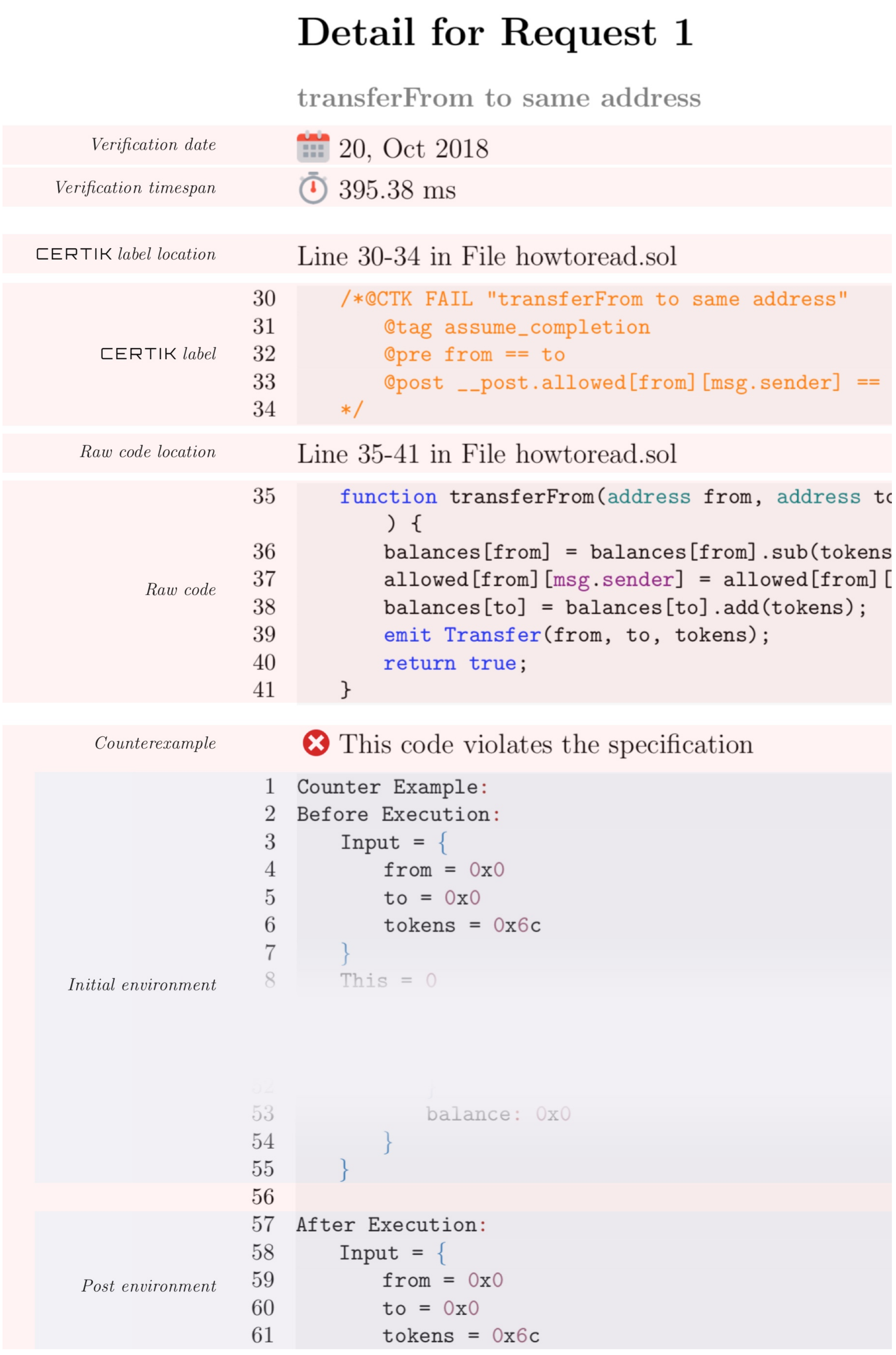
Line 5 in File mmxn.sol

```
pragma solidity ^0.4.25;
```

! Version to compile has the following bug:
0.4.25: EmptyByteArrayCopy, DynamicArrayCleanup, ImplicitConstructorCallvalueCheck, TupleAssignmentMultiStackSlotComponents, MemoryArrayCreationOverflow, privateCanBeOverridden, SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, DynamicConstructorArgumentsClippedABIV2, UninitializedFunctionPointerInConstructor_0.4.x, IncorrectEventSignatureInLibraries_0.4.x, ABIEncoderV2PackedStorage_0.4.x
0.4.26: EmptyByteArrayCopy, DynamicArrayCleanup, ImplicitConstructorCallvalueCheck, TupleAssignmentMultiStackSlotComponents, MemoryArrayCreationOverflow, privateCanBeOverridden, SignedArrayStorageCopy, ABIEncoderV2StorageArrayWithMultiSlotElement, 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 t
        ) {
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

```
Counter Example:
Before Execution:
    Input = {
        from = 0x0
        to = 0x0
        tokens = 0x6c
    }
    This = 0

            balance: 0x0
        }
    }

```

Post environment

```
After Execution:
    Input = {
        from = 0x0
        to = 0x0
        tokens = 0x6c
```

Formal Verification Request 1

If method completes, integer overflow would not happen.

📅 18, Jun 2021
⏱ 27.3 ms

Line 41 in File mmxn.sol

```
    //@CTK NO_OVERFLOW
```

Line 50-62 in File mmxn.sol

```
    function mul(uint256 a, uint256 b) internal pure returns (uint256) {
        // Gas optimization: this is cheaper than requiring 'a' not being
↪   zero, but the
        // benefit is lost if 'b' is also tested.
        // See:
↪   https://github.com/OpenZeppelin/openzeppelin-solidity/pull/522
        if (a == 0) {
            return 0;
        }

        uint256 c = a * b;
        require(c / a == b);

        return c;
    }
```

✅ The code meets the specification.

Formal Verification Request 2

SafeMath_mul

📅 18, Jun 2021
⏱ 108.02 ms

Line 42-49 in File mmxn.sol

```
    /*@CTK SafeMath_mul
        @tag assume_completion
        @tag spec
        @tag is_pure
        @post a==0 -> __return == 0
        @post a!=0 -> ((a>0) && (a*b/a != b)) == (__reverted)
        @post !__reverted -> __return == a * b
    */
```

Line 50-62 in File mmxn.sol

```
    function mul(uint256 a, uint256 b) internal pure returns (uint256) {
        // Gas optimization: this is cheaper than requiring 'a' not being
↪   zero, but the
        // benefit is lost if 'b' is also tested.
        // See:
↪   https://github.com/OpenZeppelin/openzeppelin-solidity/pull/522
        if (a == 0) {
            return 0;
        }

        uint256 c = a * b;
        require(c / a == b);

        return c;
    }
```

✓ The code meets the specification.

Formal Verification Request 3

If method completes, integer overflow would not happen.

18, Jun 2021
13.55 ms

Line 67 in File mmxn.sol

```
    //@CTK NO_OVERFLOW
```

Line 75-82 in File mmxn.sol

```
    function div(uint256 a, uint256 b) internal pure returns (uint256) {
        // Solidity only automatically asserts when dividing by 0
        require(b > 0);
        uint256 c = a / b;
        // assert(a == b * c + a % b); // There is no case in which this
↪   doesn't hold

        return c;
    }
```

✓ The code meets the specification.

Formal Verification Request 4

SafeMath_div

18, Jun 2021
2.12 ms

Line 68-74 in File mmxn.sol

```
    /*@CTK SafeMath_div
        @tag assume_completion
        @tag spec
        @tag is_pure
        @post b > 0
        @post __return == a / b
    */
```

Line 75-82 in File mmxn.sol

```
    function div(uint256 a, uint256 b) internal pure returns (uint256) {
        // Solidity only automatically asserts when dividing by 0
        require(b > 0);
        uint256 c = a / b;
        // assert(a == b * c + a % b); // There is no case in which this
↪   doesn't hold

        return c;
    }
```

✓ The code meets the specification.

Formal Verification Request 5

If method completes, integer overflow would not happen.

📅 18, Jun 2021
⏱ 14.32 ms

Line 87 in File mmxn.sol

```
    //@CTK NO_OVERFLOW
```

Line 95-100 in File mmxn.sol

```
    function sub(uint256 a, uint256 b) internal pure returns (uint256) {
        require(b <= a);
        uint256 c = a - b;

        return c;
    }
```

✓ The code meets the specification.

Formal Verification Request 6

SafeMath_sub

📅 18, Jun 2021
⏱ 2.09 ms

Line 88-94 in File mmxn.sol

```
    /*@CTK SafeMath_sub
        @tag assume_completion
        @tag spec
        @tag is_pure
        @post b <= a
        @post __return == a - b
    */
```

Line 95-100 in File mmxn.sol

```
    function sub(uint256 a, uint256 b) internal pure returns (uint256) {
        require(b <= a);
        uint256 c = a - b;

        return c;
    }
```

✓ The code meets the specification.

Formal Verification Request 7

If method completes, integer overflow would not happen.

📅 18, Jun 2021
⏱ 14.52 ms

Line 105 in File mmxn.sol

```
    //@CTK NO_OVERFLOW
```

Line 113-118 in File mmxn.sol

```
    function add(uint256 a, uint256 b) internal pure returns (uint256) {
        uint256 c = a + b;
        require(c >= a);

        return c;
    }
```

✓ The code meets the specification.

Formal Verification Request 8

SafeMath_add

📅 18, Jun 2021
⏱ 1.7 ms

Line 106-112 in File mmxn.sol

```
    /*@CTK SafeMath_add
        @tag assume_completion
        @tag spec
        @tag is_pure
        @post (a+b < a) == (__reverted)
        @post !__reverted -> __return == a + b
    */
```

Line 113-118 in File mmxn.sol

```
    function add(uint256 a, uint256 b) internal pure returns (uint256) {
        uint256 c = a + b;
        require(c >= a);

        return c;
    }
```

✓ The code meets the specification.

Formal Verification Request 9

If method completes, integer overflow would not happen.

18, Jun 2021
13.53 ms

Line 124 in File mmxn.sol

```
    //@CTK NO_OVERFLOW
```

Line 132-135 in File mmxn.sol

```
    function mod(uint256 a, uint256 b) internal pure returns (uint256) {
        require(b != 0);
        return a % b;
    }
```

✓ The code meets the specification.

Formal Verification Request 10

SafeMath_mod

18, Jun 2021
1.77 ms

Line 125-131 in File mmxn.sol

```
    /*@CTK SafeMath_mod
        @tag assume_completion
        @tag spec
        @tag is_pure
        @post b != 0
        @post __return == a % b
    */
```

Line 132-135 in File mmxn.sol

```
    function mod(uint256 a, uint256 b) internal pure returns (uint256) {
        require(b != 0);
        return a % b;
    }
```

✓ The code meets the specification.

Formal Verification Request 11

If method completes, integer overflow would not happen.

18, Jun 2021
5.49 ms

Line 164 in File mmxn.sol

```
    //@CTK NO_OVERFLOW
```

Line 170-172 in File mmxn.sol

```
    function totalSupply() public view returns (uint256) {
        return _totalSupply;
    }
```

✓ The code meets the specification.

Formal Verification Request 12

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

18, Jun 2021
1.6 ms

Line 165 in File mmxn.sol

```
    //@CTK NO_BUF_OVERFLOW
```

Line 170-172 in File mmxn.sol

```
    function totalSupply() public view returns (uint256) {
        return _totalSupply;
    }
```

✓ The code meets the specification.

Formal Verification Request 13

Method will not encounter an assertion failure.

📅 18, Jun 2021
⏱ 1.83 ms

Line 166 in File mmxn.sol

```
    //@CTK NO_ASF
```

Line 170-172 in File mmxn.sol

```
    function totalSupply() public view returns (uint256) {
        return _totalSupply;
    }
```

✓ The code meets the specification.

Formal Verification Request 14

ERC20_totalSupply

📅 18, Jun 2021
⏱ 0.99 ms

Line 167-169 in File mmxn.sol

```
    /*@CTK ERC20_totalSupply
        @post __return == _totalSupply
    */
```

Line 170-172 in File mmxn.sol

```
    function totalSupply() public view returns (uint256) {
        return _totalSupply;
    }
```

✓ The code meets the specification.

Formal Verification Request 15

If method completes, integer overflow would not happen.

📅 18, Jun 2021
⏱ 4.59 ms

Line 179 in File mmxn.sol

```
    //@CTK NO_OVERFLOW
```

Line 185-187 in File mmxn.sol

```
    function balanceOf(address owner) public view returns (uint256) {
        return _balances[owner];
    }
```

✓ The code meets the specification.

Formal Verification Request 16

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

📅 18, Jun 2021
⏱ 0.93 ms

Line 180 in File mmxn.sol

```
    //@CTK NO_BUF_OVERFLOW
```

Line 185-187 in File mmxn.sol

```
    function balanceOf(address owner) public view returns (uint256) {
        return _balances[owner];
    }
```

✓ The code meets the specification.

Formal Verification Request 17

Method will not encounter an assertion failure.

📅 18, Jun 2021
⏱ 1.07 ms

Line 181 in File mmxn.sol

```
    //@CTK NO_ASF
```

Line 185-187 in File mmxn.sol

```
    function balanceOf(address owner) public view returns (uint256) {
        return _balances[owner];
    }
```

✓ The code meets the specification.

Formal Verification Request 18

ERC20__balanceOf

📅 18, Jun 2021
⏱ 1.04 ms

Line 182-184 in File mmxn.sol

```
    /*@CTK ERC20_balanceOf
        @post __return == _balances[owner]
    */
```

Line 185-187 in File mmxn.sol

```
    function balanceOf(address owner) public view returns (uint256) {
        return _balances[owner];
    }
```

✓ The code meets the specification.

Formal Verification Request 19

If method completes, integer overflow would not happen.

📅 18, Jun 2021
⏱ 6.92 ms

Line 195 in File mmxn.sol

```
    //@CTK NO_OVERFLOW
```

Line 201-203 in File mmxn.sol

```
    function allowance(address owner, address spender) public view returns
 ↪  (uint256) {
        return _allowed[owner][spender];
    }
```

✅ The code meets the specification.

Formal Verification Request 20

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

📅 18, Jun 2021
⏱ 1.45 ms

Line 196 in File mmxn.sol

```
    //@CTK NO_BUF_OVERFLOW
```

Line 201-203 in File mmxn.sol

```
    function allowance(address owner, address spender) public view returns
 ↪  (uint256) {
        return _allowed[owner][spender];
    }
```

✅ The code meets the specification.

Formal Verification Request 21

Method will not encounter an assertion failure.

📅 18, Jun 2021
⏱ 0.81 ms

Line 197 in File mmxn.sol

```
    //@CTK NO_ASF
```

Line 201-203 in File mmxn.sol

```
    function allowance(address owner, address spender) public view returns
 ↪  (uint256) {
        return _allowed[owner][spender];
    }
```

✅ The code meets the specification.

Formal Verification Request 22

ERC20_allowance

18, Jun 2021
1.14 ms

Line 198-200 in File mmxn.sol

```
    /*@CTK ERC20_allowance
        @post __return == _allowed[owner][spender]
    */
```

Line 201-203 in File mmxn.sol

```
    function allowance(address owner, address spender) public view returns
↪   (uint256) {
        return _allowed[owner][spender];
    }
```

✓ The code meets the specification.

Formal Verification Request 23

ERC20_transfer

18, Jun 2021
100.41 ms

Line 210-216 in File mmxn.sol

```
    /*@CTK "ERC20_transfer"
      @tag assume_completion
      @post to != address(0)
      @post to != msg.sender -> __post._balances[msg.sender] ==
↪   _balances[msg.sender] - value
      @post to != msg.sender -> __post._balances[to] == _balances[to] +
↪   value
      @post to == msg.sender -> __post._balances[msg.sender] ==
↪   _balances[msg.sender]
     */
```

Line 217-220 in File mmxn.sol

```
    function transfer(address to, uint256 value) public returns (bool) {
        _transfer(msg.sender, to, value);
        return true;
    }
```

✓ The code meets the specification.

Formal Verification Request 24

If method completes, integer overflow would not happen.

📅 18, Jun 2021
⏱ 16.04 ms

Line 231 in File mmxn.sol

```
    //@CTK NO_OVERFLOW
```

Line 239-245 in File mmxn.sol

```
    function approve(address spender, uint256 value) public returns (bool) {
        require(spender != address(0));

        _allowed[msg.sender][spender] = value;
        emit Approval(msg.sender, spender, value);
        return true;
    }
```

✓ The code meets the specification.

Formal Verification Request 25

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

📅 18, Jun 2021
⏱ 0.93 ms

Line 232 in File mmxn.sol

```
    //@CTK NO_BUF_OVERFLOW
```

Line 239-245 in File mmxn.sol

```
    function approve(address spender, uint256 value) public returns (bool) {
        require(spender != address(0));

        _allowed[msg.sender][spender] = value;
        emit Approval(msg.sender, spender, value);
        return true;
    }
```

✓ The code meets the specification.

Formal Verification Request 26

Method will not encounter an assertion failure.

📅 18, Jun 2021
⏱ 1.03 ms

Line 233 in File mmxn.sol

```
    //@CTK NO_ASF
```

Line 239-245 in File mmxn.sol

```
    function approve(address spender, uint256 value) public returns (bool) {
        require(spender != address(0));

        _allowed[msg.sender][spender] = value;
        emit Approval(msg.sender, spender, value);
        return true;
    }
```

✓ The code meets the specification.

Formal Verification Request 27

ERC20_approve

📅 18, Jun 2021
⏱ 2.42 ms

Line 234-238 in File mmxn.sol

```
    /*@CTK ERC20_approve
        @tag assume_completion
        @post spender != address(0)
        @post __post._allowed[msg.sender][spender] == value
    */
```

Line 239-245 in File mmxn.sol

```
    function approve(address spender, uint256 value) public returns (bool) {
        require(spender != address(0));

        _allowed[msg.sender][spender] = value;
        emit Approval(msg.sender, spender, value);
        return true;
    }
```

✓ The code meets the specification.

Formal Verification Request 28

ERC20_transferFrom

📅 18, Jun 2021
⏱ 133.18 ms

Line 255-262 in File mmxn.sol

```
    /*@CTK "ERC20_transferFrom"
      @tag assume_completion
      @post __post._allowed[from][msg.sender] == _allowed[from][msg.sender]
↪  - value
      @post (to == address(0)) == (__reverted)
      @post (!__reverted && to != from) -> (__post._balances[from] ==
↪  _balances[from] - value)
      @post (!__reverted && to != from) -> (__post._balances[to] ==
↪  _balances[to] + value)
      @post (!__reverted && to == from) -> (__post._balances[from] ==
↪  _balances[from])
     */
```

Line 263-268 in File mmxn.sol

```
    function transferFrom(address from, address to, uint256 value) public
↪  returns (bool) {
        _allowed[from][msg.sender] = _allowed[from][msg.sender].sub(value);
        _transfer(from, to, value);
        emit Approval(from, msg.sender, _allowed[from][msg.sender]);
        return true;
    }
```

✓ The code meets the specification.

Formal Verification Request 29

ERC20__increaseAllowance

18, Jun 2021
24.28 ms

Line 280-284 in File mmxn.sol

```
    /*@CTK ERC20_increaseAllowance
        @tag assume_completion
        @post spender != address(0)
        @post __post._allowed[msg.sender][spender] ==
↪  _allowed[msg.sender][spender] + addedValue
    */
```

Line 285-291 in File mmxn.sol

```
    function increaseAllowance(address spender, uint256 addedValue) public
↪  returns (bool) {
        require(spender != address(0));

        _allowed[msg.sender][spender] =
↪  _allowed[msg.sender][spender].add(addedValue);
        emit Approval(msg.sender, spender, _allowed[msg.sender][spender]);
        return true;
    }
```

✓ The code meets the specification.

Formal Verification Request 30

ERC20_decreaseAllowance

📅 18, Jun 2021
⏱ 25.9 ms

Line 303-307 in File mmxn.sol

```
    /*@CTK ERC20_decreaseAllowance
        @tag assume_completion
        @post spender != address(0)
        @post __post._allowed[msg.sender][spender] ==
↪   _allowed[msg.sender][spender] - subtractedValue
    */
```

Line 308-314 in File mmxn.sol

```
    function decreaseAllowance(address spender, uint256 subtractedValue)
↪   public returns (bool) {
        require(spender != address(0));

        _allowed[msg.sender][spender] =
↪   _allowed[msg.sender][spender].sub(subtractedValue);
        emit Approval(msg.sender, spender, _allowed[msg.sender][spender]);
        return true;
    }
```

✓ The code meets the specification.

Formal Verification Request 31

ERC20___transfer

📅 18, Jun 2021
⏱ 28.99 ms

Line 322-328 in File mmxn.sol

```
    /*@CTK "ERC20__transfer"
      @tag assume_completion
      @post to != address(0)
      @post to != from -> __post._balances[from] == _balances[from] - value
      @post to != from -> __post._balances[to] == _balances[to] + value
      @post to == from -> __post._balances[from] == _balances[from]
     */
```

Line 329-335 in File mmxn.sol

```
    function _transfer(address from, address to, uint256 value) internal {
        require(to != address(0));

        _balances[from] = _balances[from].sub(value);
        _balances[to] = _balances[to].add(value);
        emit Transfer(from, to, value);
    }
```

✓ The code meets the specification.

Formal Verification Request 32

ERC20___mint

18, Jun 2021
36.61 ms

Line 344-349 in File mmxn.sol

```
    /*@CTK "ERC20__mint"
      @tag assume_completion
      @post account != address(0)
      @post __post._totalSupply == _totalSupply + value
      @post __post._balances[account] == _balances[account] + value
     */
```

Line 350-356 in File mmxn.sol

```
    function _mint(address account, uint256 value) internal {
        require(account != address(0));

        _totalSupply = _totalSupply.add(value);
        _balances[account] = _balances[account].add(value);
        emit Transfer(address(0), account, value);
    }
```

✓ The code meets the specification.

Formal Verification Request 33

ERC20___burn

18, Jun 2021
40.17 ms

Line 364-369 in File mmxn.sol

```
    /*@CTK "ERC20__burn"
      @tag assume_completion
      @post account != address(0)
      @post __post._totalSupply == _totalSupply - value
      @post __post._balances[account] == _balances[account] - value
     */
```

Line 370-376 in File mmxn.sol

```
    function _burn(address account, uint256 value) internal {
        require(account != address(0));

        _totalSupply = _totalSupply.sub(value);
        _balances[account] = _balances[account].sub(value);
        emit Transfer(account, address(0), value);
    }
```

✅ The code meets the specification.

Formal Verification Request 34

ERC20___burnFrom

📅 18, Jun 2021
⏱ 114.21 ms

Line 386-392 in File mmxn.sol

```
    /*@CTK "ERC20__burnFrom"
      @tag assume_completion
      @post __post._allowed[account][msg.sender] ==
    ↪ _allowed[account][msg.sender] - value
      @post (account == address(0)) == __reverted
      @post !__reverted -> __post._totalSupply == _totalSupply - value
      @post !__reverted -> __post._balances[account] == _balances[account] -
    ↪ value
     */
```

Line 393-397 in File mmxn.sol

```
    function _burnFrom(address account, uint256 value) internal {
        _allowed[account][msg.sender] =
    ↪ _allowed[account][msg.sender].sub(value);
        _burn(account, value);
        emit Approval(account, msg.sender, _allowed[account][msg.sender]);
    }
```

✅ The code meets the specification.

Formal Verification Request 35

Roles_add

📅 18, Jun 2021
⏱ 45.52 ms

Line 414-419 in File mmxn.sol

```
    /*@CTK "Roles_add"
      @tag assume_completion
      @post account != address(0)
      @post role.bearer[account] == false
      @post role__post.bearer[account] == true
     */
```

Line 420-425 in File mmxn.sol

```
    function add(Role storage role, address account) internal {
        require(account != address(0));
        require(!has(role, account));

        role.bearer[account] = true;
    }
```

✓ The code meets the specification.

Formal Verification Request 36

Roles_remove

18, Jun 2021
40.49 ms

Line 430-435 in File mmxn.sol

```
    /*@CTK "Roles_remove"
      @tag assume_completion
      @post account != address(0)
      @post role.bearer[account] == true
      @post role__post.bearer[account] == false
     */
```

Line 436-441 in File mmxn.sol

```
    function remove(Role storage role, address account) internal {
        require(account != address(0));
        require(has(role, account));

        role.bearer[account] = false;
    }
```

✓ The code meets the specification.

Formal Verification Request 37

Roles_has

18, Jun 2021
2.94 ms

Line 447-451 in File mmxn.sol

```
    /*@CTK "Roles_has"
      @tag assume_completion
      @post account != address(0)
      @post __return == role.bearer[account]
     */
```

Line 452-455 in File mmxn.sol

```
    function has(Role storage role, address account) internal view returns
↪   (bool) {
        require(account != address(0));
        return role.bearer[account];
    }
```

✓ The code meets the specification.

Formal Verification Request 38

MinterRole_constructor

18, Jun 2021
109.48 ms

Line 468-472 in File mmxn.sol

```
    /*@CTK "MinterRole_constructor"
      @tag assume_completion
      @post _minters.bearer[msg.sender] == false
      @post __post._minters.bearer[msg.sender] == true
     */
```

Line 473-475 in File mmxn.sol

```
    constructor () internal {
        _addMinter(msg.sender);
    }
```

✓ The code meets the specification.

Formal Verification Request 39

MinterRole_isMinter

18, Jun 2021
32.89 ms

Line 482-486 in File mmxn.sol

```
    /*@CTK "MinterRole_isMinter"
      @tag assume_completion
      @post account != address(0)
      @post __return == _minters.bearer[account]
     */
```

Line 487-489 in File mmxn.sol

```
    function isMinter(address account) public view returns (bool) {
        return _minters.has(account);
    }
```

✓ The code meets the specification.

Formal Verification Request 40

MinterRole_addMinter

📅 18, Jun 2021
⏱ 97.75 ms

Line 491-497 in File mmxn.sol

```
    /*@CTK "MinterRole_addMinter"
      @tag assume_completion
      @post _minters.bearer[msg.sender] == true
      @post account != address(0)
      @post _minters.bearer[account] == false
      @post __post._minters.bearer[account] == true
     */
```

Line 498-500 in File mmxn.sol

```
    function addMinter(address account) public onlyMinter {
        _addMinter(account);
    }
```

✓ The code meets the specification.

Formal Verification Request 41

MinterRole_renounceMinter

📅 18, Jun 2021
⏱ 109.27 ms

Line 502-506 in File mmxn.sol

```
    /*@CTK "MinterRole_renounceMinter"
      @tag assume_completion
      @post _minters.bearer[msg.sender] == true
      @post __post._minters.bearer[msg.sender] == false
     */
```

Line 507-509 in File mmxn.sol

```
    function renounceMinter() public {
        _removeMinter(msg.sender);
    }
```

✓ The code meets the specification.

Formal Verification Request 42

MinterRole___addMinter

📅 18, Jun 2021
⏱ 3.81 ms

Line 511-516 in File mmxn.sol

```
    /*@CTK "MinterRole__addMinter"
      @tag assume_completion
      @post account != address(0)
      @post _minters.bearer[account] == false
      @post __post._minters.bearer[account] == true
     */
```

Line 517-520 in File mmxn.sol

```
    function _addMinter(address account) internal {
        _minters.add(account);
        emit MinterAdded(account);
    }
```

✓ The code meets the specification.

Formal Verification Request 43

MinterRole___removeMinter

📅 18, Jun 2021
⏱ 3.76 ms

Line 522-527 in File mmxn.sol

```
    /*@CTK "MinterRole__removeMinter"
      @tag assume_completion
      @post account != address(0)
      @post _minters.bearer[account] == true
      @post __post._minters.bearer[account] == false
     */
```

Line 528-531 in File mmxn.sol

```
    function _removeMinter(address account) internal {
        _minters.remove(account);
        emit MinterRemoved(account);
    }
```

✓ The code meets the specification.

Formal Verification Request 44

ERC20Mintable_mint

📅 18, Jun 2021
⏱ 175.54 ms

Line 547-553 in File mmxn.sol

```
    /*@CTK "ERC20Mintable_mint"
      @tag assume_completion
      @post _minters.bearer[msg.sender] == true
      @post to != address(0)
      @post __post._totalSupply == _totalSupply + value
      @post __post._balances[to] == _balances[to] + value
    */
```

Line 554-557 in File mmxn.sol

```
    function mint(address to, uint256 value) public onlyMinter returns (bool)
    ↪  {
        _mint(to, value);
        return true;
    }
```

✅ The code meets the specification.

Formal Verification Request 45

ERC20Burnable___burn

📅 18, Jun 2021
⏱ 76.01 ms

Line 571-575 in File mmxn.sol

```
    /*@CTK "ERC20Burnable__burn"
      @tag assume_completion
      @post __post._totalSupply == _totalSupply - value
      @post __post._balances[msg.sender] == _balances[msg.sender] - value
    */
```

Line 576-578 in File mmxn.sol

```
    function burn(uint256 value) public {
        _burn(msg.sender, value);
    }
```

✅ The code meets the specification.

Formal Verification Request 46

ERC20Burnable___burnFrom

📅 18, Jun 2021
⏱ 174.03 ms

Line 585-591 in File mmxn.sol

```
    /*@CTK "ERC20Burnable__burnFrom"
      @tag assume_completion
      @post __post._allowed[from][msg.sender] == _allowed[from][msg.sender]
↪   - value
      @post (from == address(0)) == __reverted
      @post !__reverted -> __post._totalSupply == _totalSupply - value
      @post !__reverted -> __post._balances[from] == _balances[from] - value
     */
```

Line 592-594 in File mmxn.sol

```
    function burnFrom(address from, uint256 value) public {
        _burnFrom(from, value);
    }
```

✅ The code meets the specification.

Formal Verification Request 47

PauserRole_constructor

📅 18, Jun 2021
⏱ 95.72 ms

Line 607-611 in File mmxn.sol

```
    /*@CTK "PauserRole_constructor"
      @tag assume_completion
      @post _pausers.bearer[msg.sender] == false
      @post __post._pausers.bearer[msg.sender] == true
     */
```

Line 612-614 in File mmxn.sol

```
    constructor () internal {
        _addPauser(msg.sender);
    }
```

✅ The code meets the specification.

Formal Verification Request 48

PauserRole_isPauser

📅 18, Jun 2021
⏱ 29.68 ms

Line 621-625 in File mmxn.sol

```
    /*@CTK "PauserRole_isPauser"
      @tag assume_completion
      @post account != address(0)
      @post __return == _pausers.bearer[account]
     */
```

Line 626-628 in File mmxn.sol

```
    function isPauser(address account) public view returns (bool) {
        return _pausers.has(account);
    }
```

✅ The code meets the specification.

Formal Verification Request 49

PauserRole_addPauser

📅 18, Jun 2021
⏱ 95.07 ms

Line 630-636 in File mmxn.sol

```
    /*@CTK "PauserRole_addPauser"
      @tag assume_completion
      @post _pausers.bearer[msg.sender] == true
      @post account != address(0)
      @post _pausers.bearer[account] == false
      @post __post._pausers.bearer[account] == true
     */
```

Line 637-639 in File mmxn.sol

```
    function addPauser(address account) public onlyPauser {
        _addPauser(account);
    }
```

✅ The code meets the specification.

Formal Verification Request 50

PauserRole_renouncePauser

📅 18, Jun 2021
⏱ 100.12 ms

Line 641-645 in File mmxn.sol

```
    /*@CTK "PauserRole_renouncePauser"
      @tag assume_completion
      @post _pausers.bearer[msg.sender] == true
      @post __post._pausers.bearer[msg.sender] == false
     */
```

Line 646-648 in File mmxn.sol

```
    function renouncePauser() public {
        _removePauser(msg.sender);
    }
```

✓ The code meets the specification.

Formal Verification Request 51

PauserRole___addPauser

📅 18, Jun 2021
⏱ 3.76 ms

Line 650-655 in File mmxn.sol

```
    /*@CTK "PauserRole__addPauser"
      @tag assume_completion
      @post account != address(0)
      @post _pausers.bearer[account] == false
      @post __post._pausers.bearer[account] == true
     */
```

Line 656-659 in File mmxn.sol

```
    function _addPauser(address account) internal {
        _pausers.add(account);
        emit PauserAdded(account);
    }
```

✓ The code meets the specification.

Formal Verification Request 52

PauserRole___removePauser

📅 18, Jun 2021
⏱ 4.06 ms

Line 661-666 in File mmxn.sol

```
    /*@CTK "PauserRole__removePauser"
      @tag assume_completion
      @post account != address(0)
      @post _pausers.bearer[account] == true
      @post __post._pausers.bearer[account] == false
     */
```

Line 667-670 in File mmxn.sol

```
    function _removePauser(address account) internal {
        _pausers.remove(account);
        emit PauserRemoved(account);
    }
```

✓ The code meets the specification.

Formal Verification Request 53

Pausable__constructor

18, Jun 2021
7.36 ms

Line 685-687 in File mmxn.sol

```
    /*@CTK "Pausable_constructor"
      @post __post._paused == false
     */
```

Line 688-690 in File mmxn.sol

```
    constructor () internal {
        _paused = false;
    }
```

✓ The code meets the specification.

Formal Verification Request 54

Pausable__paused

18, Jun 2021
6.16 ms

Line 695-697 in File mmxn.sol

```
    /*@CTK "Pausable_paused"
      @post __return == _paused
     */
```

Line 698-700 in File mmxn.sol

```
    function paused() public view returns (bool) {
        return _paused;
    }
```

✓ The code meets the specification.

Formal Verification Request 55

Pausable_pause

📅 18, Jun 2021
⏱ 78.88 ms

Line 721-726 in File mmxn.sol

```
    /*@CTK "Pausable_pause"
        @tag assume_completion
        @post _pausers.bearer[msg.sender] == true
        @post _paused == false
        @post __post._paused == true
     */
```

Line 727-730 in File mmxn.sol

```
    function pause() public onlyPauser whenNotPaused {
        _paused = true;
        emit Paused(msg.sender);
    }
```

✓ The code meets the specification.

Formal Verification Request 56

Pausable_unpause

📅 18, Jun 2021
⏱ 53.41 ms

Line 735-740 in File mmxn.sol

```
    /*@CTK "Pausable_unpause"
        @tag assume_completion
        @post _pausers.bearer[msg.sender] == true
        @post _paused == true
        @post __post._paused == false
     */
```

Line 741-744 in File mmxn.sol

```
    function unpause() public onlyPauser whenPaused {
        _paused = false;
        emit Unpaused(msg.sender);
    }
```

✓ The code meets the specification.

Formal Verification Request 57

ERC20Pausable__transfer

📅 18, Jun 2021
⏱ 197.08 ms

Line 754-761 in File mmxn.sol

```
    /*@CTK "ERC20Pausable_transfer"
      @tag assume_completion
      @post _paused == false
      @post to != address(0)
      @post to != msg.sender -> __post._balances[msg.sender] ==
↪   _balances[msg.sender] - value
      @post to != msg.sender -> __post._balances[to] == _balances[to] +
↪   value
      @post to == msg.sender -> __post._balances[msg.sender] ==
↪   _balances[msg.sender]
     */
```

Line 762-764 in File mmxn.sol

```
    function transfer(address to, uint256 value) public whenNotPaused returns
↪   (bool) {
        return super.transfer(to, value);
    }
```

✅ The code meets the specification.

Formal Verification Request 58

ERC20Pausable__transferFrom

📅 18, Jun 2021
⏱ 205.05 ms

Line 766-774 in File mmxn.sol

```
    /*@CTK "ERC20Pausable_transferFrom"
      @tag assume_completion
      @post _paused == false
      @post __post._allowed[from][msg.sender] == _allowed[from][msg.sender]
↪   - value
      @post (to == address(0)) == (__reverted)
      @post (!__reverted && to != from) -> (__post._balances[from] ==
↪   _balances[from] - value)
      @post (!__reverted && to != from) -> (__post._balances[to] ==
↪   _balances[to] + value)
      @post (!__reverted && to == from) -> (__post._balances[from] ==
↪   _balances[from])
     */
```

Line 775-777 in File mmxn.sol

```
    function transferFrom(address from, address to, uint256 value) public
    ↪ whenNotPaused returns (bool) {
        return super.transferFrom(from, to, value);
    }
```

✓ The code meets the specification.

Formal Verification Request 59

ERC20Pausable_approve

18, Jun 2021
52.3 ms

Line 779-784 in File mmxn.sol

```
    /*@CTK ERC20Pausable_approve
        @tag assume_completion
        @post _paused == false
        @post spender != address(0)
        @post __post._allowed[msg.sender][spender] == value
    */
```

Line 785-787 in File mmxn.sol

```
    function approve(address spender, uint256 value) public whenNotPaused
    ↪ returns (bool) {
        return super.approve(spender, value);
    }
```

✓ The code meets the specification.

Formal Verification Request 60

ERC20Pausable_increaseAllowance

18, Jun 2021
75.27 ms

Line 789-794 in File mmxn.sol

```
    /*@CTK ERC20Pausable_increaseAllowance
        @tag assume_completion
        @post _paused == false
        @post spender != address(0)
        @post __post._allowed[msg.sender][spender] ==
    ↪ _allowed[msg.sender][spender] + addedValue
    */
```

Line 795-797 in File mmxn.sol

```
    function increaseAllowance(address spender, uint addedValue) public
↪   whenNotPaused returns (bool success) {
        return super.increaseAllowance(spender, addedValue);
    }
```

✓ The code meets the specification.

Formal Verification Request 61

ERC20Pausable_decreaseAllowance

18, Jun 2021
73.71 ms

Line 799-804 in File mmxn.sol

```
    /*@CTK ERC20Pausable_decreaseAllowance
        @tag assume_completion
        @post _paused == false
        @post spender != address(0)
        @post __post._allowed[msg.sender][spender] ==
↪   _allowed[msg.sender][spender] - subtractedValue
    */
```

Line 805-807 in File mmxn.sol

```
    function decreaseAllowance(address spender, uint subtractedValue) public
↪   whenNotPaused returns (bool success) {
        return super.decreaseAllowance(spender, subtractedValue);
    }
```

✓ The code meets the specification.

Formal Verification Request 62

ERC20Detailed_constructor

18, Jun 2021
9.76 ms

Line 823-828 in File mmxn.sol

```
    /*@CTK ERC20Detailed_constructor
        @tag assume_completion
        @post __post._name == name
        @post __post._symbol == symbol
        @post __post._decimals == decimals
    */
```

Line 829-833 in File mmxn.sol

```
    constructor (string memory name, string memory symbol, uint8 decimals)
↪   public {
        _name = name;
        _symbol = symbol;
        _decimals = decimals;
    }
```

✓ The code meets the specification.

Formal Verification Request 63

ERC20Detailed__name

18, Jun 2021
6.28 ms

Line 838-841 in File mmxn.sol

```
     /*@CTK ERC20Detailed_name
        @tag assume_completion
        @post __return == _name
    */
```

Line 842-844 in File mmxn.sol

```
    function name() public view returns (string memory) {
        return _name;
    }
```

✓ The code meets the specification.

Formal Verification Request 64

ERC20Detailed__symbol

18, Jun 2021
4.9 ms

Line 849-852 in File mmxn.sol

```
    /*@CTK ERC20Detailed_symbol
        @tag assume_completion
        @post __return == _symbol
    */
```

Line 853-855 in File mmxn.sol

```
    function symbol() public view returns (string memory) {
        return _symbol;
    }
```

✓ The code meets the specification.

Formal Verification Request 65

ERC20Detailed_decimals

18, Jun 2021
5.44 ms

Line 860-863 in File mmxn.sol

```
    /*@CTK ERC20Detailed_decimals
        @tag assume_completion
        @post __return == _decimals
    */
```

Line 864-866 in File mmxn.sol

```
    function decimals() public view returns (uint8) {
        return _decimals;
    }
```

✓ The code meets the specification.

Source Code with CertiK Labels

mmxn.sol

```
/**
 *Submitted for verification at Etherscan.io on 2021-06-16
*/

pragma solidity ^0.4.25;

// File: openzeppelin-solidity/contracts/token/ERC20/IERC20.sol

/**
 * @title ERC20 interface
 * @dev see https://github.com/ethereum/EIPs/issues/20
 */
interface IERC20 {
    function transfer(address to, uint256 value) external returns (bool);

    function approve(address spender, uint256 value) external returns (bool);

    function transferFrom(address from, address to, uint256 value) external
 ↪  returns (bool);

    function totalSupply() external view returns (uint256);

    function balanceOf(address who) external view returns (uint256);

    function allowance(address owner, address spender) external view returns
 ↪  (uint256);

    event Transfer(address indexed from, address indexed to, uint256 value);

    event Approval(address indexed owner, address indexed spender, uint256
 ↪  value);
}

// File: openzeppelin-solidity/contracts/math/SafeMath.sol

/**
 * @title SafeMath
 * @dev Unsigned math operations with safety checks that revert on error
 */
library SafeMath {
    /**
    * @dev Multiplies two unsigned integers, reverts on overflow.
    */
    //@CTK NO_OVERFLOW
    /*@CTK SafeMath_mul
        @tag assume_completion
        @tag spec
        @tag is_pure
        @post a==0 -> __return == 0
        @post a!=0 -> ((a>0) && (a*b/a != b)) == (__reverted)
        @post !__reverted -> __return == a * b
    */
    function mul(uint256 a, uint256 b) internal pure returns (uint256) {
        // Gas optimization: this is cheaper than requiring 'a' not being
↪   zero, but the
        // benefit is lost if 'b' is also tested.
        // See:
↪   https://github.com/OpenZeppelin/openzeppelin-solidity/pull/522
        if (a == 0) {
            return 0;
        }

        uint256 c = a * b;
        require(c / a == b);

        return c;
    }

    /**
    * @dev Integer division of two unsigned integers truncating the
↪   quotient, reverts on division by zero.
    */
    //@CTK NO_OVERFLOW
    /*@CTK SafeMath_div
        @tag assume_completion
        @tag spec
        @tag is_pure
        @post b > 0
        @post __return == a / b
    */
    function div(uint256 a, uint256 b) internal pure returns (uint256) {
        // Solidity only automatically asserts when dividing by 0
        require(b > 0);
        uint256 c = a / b;
        // assert(a == b * c + a % b); // There is no case in which this
↪   doesn't hold

        return c;
    }

    /**
    * @dev Subtracts two unsigned integers, reverts on overflow (i.e. if
↪   subtrahend is greater than minuend).
    */
    //@CTK NO_OVERFLOW
    /*@CTK SafeMath_sub
        @tag assume_completion
        @tag spec
        @tag is_pure
        @post b <= a
        @post __return == a - b
    */
    function sub(uint256 a, uint256 b) internal pure returns (uint256) {
        require(b <= a);
        uint256 c = a - b;

        return c;
    }

    /**
    * @dev Adds two unsigned integers, reverts on overflow.
    */
    //@CTK NO_OVERFLOW
    /*@CTK SafeMath_add
        @tag assume_completion
        @tag spec
        @tag is_pure
        @post (a+b < a) == (__reverted)
        @post !__reverted -> __return == a + b
    */
    function add(uint256 a, uint256 b) internal pure returns (uint256) {
        uint256 c = a + b;
        require(c >= a);

        return c;
    }

    /**
    * @dev Divides two unsigned integers and returns the remainder (unsigned
↪   integer modulo),
    * reverts when dividing by zero.
    */
    //@CTK NO_OVERFLOW
    /*@CTK SafeMath_mod
        @tag assume_completion
        @tag spec
        @tag is_pure
        @post b != 0
        @post __return == a % b
    */
    function mod(uint256 a, uint256 b) internal pure returns (uint256) {
        require(b != 0);
        return a % b;
    }
}

// File: openzeppelin-solidity/contracts/token/ERC20/ERC20.sol

/**
 * @title Standard ERC20 token
 *
 * @dev Implementation of the basic standard token.
 * https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20.md
 * Originally based on code by FirstBlood:
 *
↪   https://github.com/Firstbloodio/token/blob/master/smart_contract/FirstBloodToken.s
 *
 * This implementation emits additional Approval events, allowing
↪   applications to reconstruct the allowance status for
 * all accounts just by listening to said events. Note that this isn't
↪   required by the specification, and other
 * compliant implementations may not do it.
 */
contract ERC20 is IERC20 {
    using SafeMath for uint256;

    mapping (address => uint256) private _balances;

    mapping (address => mapping (address => uint256)) private _allowed;

    uint256 private _totalSupply;

    /**
    * @dev Total number of tokens in existence
    */
    //@CTK NO_OVERFLOW
    //@CTK NO_BUF_OVERFLOW
    //@CTK NO_ASF
    /*@CTK ERC20_totalSupply
        @post __return == _totalSupply
    */
    function totalSupply() public view returns (uint256) {
        return _totalSupply;
    }

    /**
    * @dev Gets the balance of the specified address.
    * @param owner The address to query the balance of.
    * @return An uint256 representing the amount owned by the passed
↪   address.
    */
    //@CTK NO_OVERFLOW
    //@CTK NO_BUF_OVERFLOW
    //@CTK NO_ASF
    /*@CTK ERC20_balanceOf
        @post __return == _balances[owner]
    */
    function balanceOf(address owner) public view returns (uint256) {
        return _balances[owner];
    }

    /**
     * @dev Function to check the amount of tokens that an owner allowed to
↪   a spender.
     * @param owner address The address which owns the funds.
     * @param spender address The address which will spend the funds.
     * @return A uint256 specifying the amount of tokens still available for
↪   the spender.
     */
    //@CTK NO_OVERFLOW
    //@CTK NO_BUF_OVERFLOW
    //@CTK NO_ASF
    /*@CTK ERC20_allowance
        @post __return == _allowed[owner][spender]
    */
    function allowance(address owner, address spender) public view returns
↪   (uint256) {
        return _allowed[owner][spender];
    }

    /**
    * @dev Transfer token for a specified address
    * @param to The address to transfer to.
    * @param value The amount to be transferred.
    */
    /*@CTK "ERC20_transfer"
      @tag assume_completion
      @post to != address(0)
      @post to != msg.sender -> __post._balances[msg.sender] ==
↪   _balances[msg.sender] - value
      @post to != msg.sender -> __post._balances[to] == _balances[to] +
↪   value
      @post to == msg.sender -> __post._balances[msg.sender] ==
↪   _balances[msg.sender]
     */
    function transfer(address to, uint256 value) public returns (bool) {
        _transfer(msg.sender, to, value);
        return true;
    }

    /**
     * @dev Approve the passed address to spend the specified amount of
    ↪ tokens on behalf of msg.sender.
     * Beware that changing an allowance with this method brings the risk
    ↪ that someone may use both the old
     * and the new allowance by unfortunate transaction ordering. One
    ↪ possible solution to mitigate this
     * race condition is to first reduce the spender's allowance to 0 and
    ↪ set the desired value afterwards:
     * https://github.com/ethereum/EIPs/issues/20#issuecomment-263524729
     * @param spender The address which will spend the funds.
     * @param value The amount of tokens to be spent.
     */
    //@CTK NO_OVERFLOW
    //@CTK NO_BUF_OVERFLOW
    //@CTK NO_ASF
    /*@CTK ERC20_approve
        @tag assume_completion
        @post spender != address(0)
        @post __post._allowed[msg.sender][spender] == value
    */
    function approve(address spender, uint256 value) public returns (bool) {
        require(spender != address(0));

        _allowed[msg.sender][spender] = value;
        emit Approval(msg.sender, spender, value);
        return true;
    }

    /**
     * @dev Transfer tokens from one address to another.
     * Note that while this function emits an Approval event, this is not
    ↪ required as per the specification,
     * and other compliant implementations may not emit the event.
     * @param from address The address which you want to send tokens from
     * @param to address The address which you want to transfer to
     * @param value uint256 the amount of tokens to be transferred
     */
    /*@CTK "ERC20_transferFrom"
      @tag assume_completion
      @post __post._allowed[from][msg.sender] == _allowed[from][msg.sender]
    ↪ - value
      @post (to == address(0)) == (__reverted)
      @post (!__reverted && to != from) -> (__post._balances[from] ==
↪ _balances[from] - value)
      @post (!__reverted && to != from) -> (__post._balances[to] ==
↪ _balances[to] + value)
      @post (!__reverted && to == from) -> (__post._balances[from] ==
↪ _balances[from])
     */
    function transferFrom(address from, address to, uint256 value) public
↪ returns (bool) {
        _allowed[from][msg.sender] = _allowed[from][msg.sender].sub(value);
        _transfer(from, to, value);
        emit Approval(from, msg.sender, _allowed[from][msg.sender]);
        return true;
    }

    /**
     * @dev Increase the amount of tokens that an owner allowed to a
↪ spender.
     * approve should be called when allowed_[_spender] == 0. To increment
     * allowed value is better to use this function to avoid 2 calls (and
↪ wait until
     * the first transaction is mined)
     * From MonolithDAO Token.sol
     * Emits an Approval event.
     * @param spender The address which will spend the funds.
     * @param addedValue The amount of tokens to increase the allowance by.
     */
    /*@CTK ERC20_increaseAllowance
        @tag assume_completion
        @post spender != address(0)
        @post __post._allowed[msg.sender][spender] ==
↪ _allowed[msg.sender][spender] + addedValue
    */
    function increaseAllowance(address spender, uint256 addedValue) public
↪ returns (bool) {
        require(spender != address(0));

        _allowed[msg.sender][spender] =
↪ _allowed[msg.sender][spender].add(addedValue);
        emit Approval(msg.sender, spender, _allowed[msg.sender][spender]);
        return true;
    }

    /**
     * @dev Decrease the amount of tokens that an owner allowed to a
↪ spender.
     * approve should be called when allowed_[_spender] == 0. To decrement
     * allowed value is better to use this function to avoid 2 calls (and
↪   wait until
     * the first transaction is mined)
     * From MonolithDAO Token.sol
     * Emits an Approval event.
     * @param spender The address which will spend the funds.
     * @param subtractedValue The amount of tokens to decrease the allowance
↪   by.
     */
    /*@CTK ERC20_decreaseAllowance
        @tag assume_completion
        @post spender != address(0)
        @post __post._allowed[msg.sender][spender] ==
↪   _allowed[msg.sender][spender] - subtractedValue
    */
    function decreaseAllowance(address spender, uint256 subtractedValue)
↪   public returns (bool) {
        require(spender != address(0));

        _allowed[msg.sender][spender] =
↪   _allowed[msg.sender][spender].sub(subtractedValue);
        emit Approval(msg.sender, spender, _allowed[msg.sender][spender]);
        return true;
    }

    /**
    * @dev Transfer token for a specified addresses
    * @param from The address to transfer from.
    * @param to The address to transfer to.
    * @param value The amount to be transferred.
    */
    /*@CTK "ERC20__transfer"
      @tag assume_completion
      @post to != address(0)
      @post to != from -> __post._balances[from] == _balances[from] - value
      @post to != from -> __post._balances[to] == _balances[to] + value
      @post to == from -> __post._balances[from] == _balances[from]
     */
    function _transfer(address from, address to, uint256 value) internal {
        require(to != address(0));

        _balances[from] = _balances[from].sub(value);
        _balances[to] = _balances[to].add(value);
        emit Transfer(from, to, value);
    }

    /**
```

```
     * @dev Internal function that mints an amount of the token and assigns
↪   it to
     * an account. This encapsulates the modification of balances such that
↪   the
     * proper events are emitted.
     * @param account The account that will receive the created tokens.
     * @param value The amount that will be created.
     */
    /*@CTK "ERC20__mint"
      @tag assume_completion
      @post account != address(0)
      @post __post._totalSupply == _totalSupply + value
      @post __post._balances[account] == _balances[account] + value
     */
    function _mint(address account, uint256 value) internal {
        require(account != address(0));

        _totalSupply = _totalSupply.add(value);
        _balances[account] = _balances[account].add(value);
        emit Transfer(address(0), account, value);
    }

    /**
     * @dev Internal function that burns an amount of the token of a given
     * account.
     * @param account The account whose tokens will be burnt.
     * @param value The amount that will be burnt.
     */
    /*@CTK "ERC20__burn"
      @tag assume_completion
      @post account != address(0)
      @post __post._totalSupply == _totalSupply - value
      @post __post._balances[account] == _balances[account] - value
     */
    function _burn(address account, uint256 value) internal {
        require(account != address(0));

        _totalSupply = _totalSupply.sub(value);
        _balances[account] = _balances[account].sub(value);
        emit Transfer(account, address(0), value);
    }

    /**
     * @dev Internal function that burns an amount of the token of a given
     * account, deducting from the sender's allowance for said account. Uses
↪   the
     * internal burn function.
     * Emits an Approval event (reflecting the reduced allowance).
     * @param account The account whose tokens will be burnt.
     * @param value The amount that will be burnt.
     */
    /*@CTK "ERC20__burnFrom"
      @tag assume_completion
      @post __post._allowed[account][msg.sender] ==
↪   _allowed[account][msg.sender] - value
      @post (account == address(0)) == __reverted
      @post !__reverted -> __post._totalSupply == _totalSupply - value
      @post !__reverted -> __post._balances[account] == _balances[account] -
↪   value
     */
    function _burnFrom(address account, uint256 value) internal {
        _allowed[account][msg.sender] =
↪   _allowed[account][msg.sender].sub(value);
        _burn(account, value);
        emit Approval(account, msg.sender, _allowed[account][msg.sender]);
    }
}

// File: openzeppelin-solidity/contracts/access/Roles.sol

/**
 * @title Roles
 * @dev Library for managing addresses assigned to a Role.
 */
library Roles {
    struct Role {
        mapping (address => bool) bearer;
    }

    /**
     * @dev give an account access to this role
     */
    /*@CTK "Roles_add"
      @tag assume_completion
      @post account != address(0)
      @post role.bearer[account] == false
      @post role__post.bearer[account] == true
     */
    function add(Role storage role, address account) internal {
        require(account != address(0));
        require(!has(role, account));

        role.bearer[account] = true;
    }

    /**
```

```
     * @dev remove an account's access to this role
     */
    /*@CTK "Roles_remove"
      @tag assume_completion
      @post account != address(0)
      @post role.bearer[account] == true
      @post role__post.bearer[account] == false
     */
    function remove(Role storage role, address account) internal {
        require(account != address(0));
        require(has(role, account));

        role.bearer[account] = false;
    }

    /**
     * @dev check if an account has this role
     * @return bool
     */
    /*@CTK "Roles_has"
      @tag assume_completion
      @post account != address(0)
      @post __return == role.bearer[account]
     */
    function has(Role storage role, address account) internal view returns
 ↪  (bool) {
        require(account != address(0));
        return role.bearer[account];
    }
}

// File: openzeppelin-solidity/contracts/access/roles/MinterRole.sol

contract MinterRole {
    using Roles for Roles.Role;

    event MinterAdded(address indexed account);
    event MinterRemoved(address indexed account);

    Roles.Role private _minters;

    /*@CTK "MinterRole_constructor"
      @tag assume_completion
      @post _minters.bearer[msg.sender] == false
      @post __post._minters.bearer[msg.sender] == true
     */
    constructor () internal {
        _addMinter(msg.sender);
    }

    modifier onlyMinter() {
        require(isMinter(msg.sender));
        _;
    }

    /*@CTK "MinterRole_isMinter"
      @tag assume_completion
      @post account != address(0)
      @post __return == _minters.bearer[account]
     */
    function isMinter(address account) public view returns (bool) {
        return _minters.has(account);
    }

    /*@CTK "MinterRole_addMinter"
      @tag assume_completion
      @post _minters.bearer[msg.sender] == true
      @post account != address(0)
      @post _minters.bearer[account] == false
      @post __post._minters.bearer[account] == true
     */
    function addMinter(address account) public onlyMinter {
        _addMinter(account);
    }

    /*@CTK "MinterRole_renounceMinter"
      @tag assume_completion
      @post _minters.bearer[msg.sender] == true
      @post __post._minters.bearer[msg.sender] == false
     */
    function renounceMinter() public {
        _removeMinter(msg.sender);
    }

    /*@CTK "MinterRole__addMinter"
      @tag assume_completion
      @post account != address(0)
      @post _minters.bearer[account] == false
      @post __post._minters.bearer[account] == true
     */
    function _addMinter(address account) internal {
        _minters.add(account);
        emit MinterAdded(account);
    }

    /*@CTK "MinterRole__removeMinter"
```

```
      @tag assume_completion
      @post account != address(0)
      @post _minters.bearer[account] == true
      @post __post._minters.bearer[account] == false
     */
    function _removeMinter(address account) internal {
        _minters.remove(account);
        emit MinterRemoved(account);
    }
}

// File: openzeppelin-solidity/contracts/token/ERC20/ERC20Mintable.sol

/**
 * @title ERC20Mintable
 * @dev ERC20 minting logic
 */
contract ERC20Mintable is ERC20, MinterRole {
    /**
     * @dev Function to mint tokens
     * @param to The address that will receive the minted tokens.
     * @param value The amount of tokens to mint.
     * @return A boolean that indicates if the operation was successful.
     */
    /*@CTK "ERC20Mintable_mint"
      @tag assume_completion
      @post _minters.bearer[msg.sender] == true
      @post to != address(0)
      @post __post._totalSupply == _totalSupply + value
      @post __post._balances[to] == _balances[to] + value
    */
    function mint(address to, uint256 value) public onlyMinter returns (bool)
 ↪  {
        _mint(to, value);
        return true;
    }
}

// File: openzeppelin-solidity/contracts/token/ERC20/ERC20Burnable.sol

/**
 * @title Burnable Token
 * @dev Token that can be irreversibly burned (destroyed).
 */
contract ERC20Burnable is ERC20 {
    /**
     * @dev Burns a specific amount of tokens.
     * @param value The amount of token to be burned.
     */
    /*@CTK "ERC20Burnable__burn"
      @tag assume_completion
      @post __post._totalSupply == _totalSupply - value
      @post __post._balances[msg.sender] == _balances[msg.sender] - value
    */
    function burn(uint256 value) public {
        _burn(msg.sender, value);
    }

    /**
     * @dev Burns a specific amount of tokens from the target address and
 ↪  decrements allowance
     * @param from address The address which you want to send tokens from
     * @param value uint256 The amount of token to be burned
     */
    /*@CTK "ERC20Burnable__burnFrom"
      @tag assume_completion
      @post __post._allowed[from][msg.sender] == _allowed[from][msg.sender]
 ↪  - value
      @post (from == address(0)) == __reverted
      @post !__reverted -> __post._totalSupply == _totalSupply - value
      @post !__reverted -> __post._balances[from] == _balances[from] - value
     */
    function burnFrom(address from, uint256 value) public {
        _burnFrom(from, value);
    }
}

// File: openzeppelin-solidity/contracts/access/roles/PauserRole.sol

contract PauserRole {
    using Roles for Roles.Role;

    event PauserAdded(address indexed account);
    event PauserRemoved(address indexed account);

    Roles.Role private _pausers;

    /*@CTK "PauserRole_constructor"
      @tag assume_completion
      @post _pausers.bearer[msg.sender] == false
      @post __post._pausers.bearer[msg.sender] == true
     */
    constructor () internal {
        _addPauser(msg.sender);
    }

    modifier onlyPauser() {
        require(isPauser(msg.sender));
        _;
    }

    /*@CTK "PauserRole_isPauser"
      @tag assume_completion
      @post account != address(0)
      @post __return == _pausers.bearer[account]
     */
    function isPauser(address account) public view returns (bool) {
        return _pausers.has(account);
    }

    /*@CTK "PauserRole_addPauser"
      @tag assume_completion
      @post _pausers.bearer[msg.sender] == true
      @post account != address(0)
      @post _pausers.bearer[account] == false
      @post __post._pausers.bearer[account] == true
     */
    function addPauser(address account) public onlyPauser {
        _addPauser(account);
    }

    /*@CTK "PauserRole_renouncePauser"
      @tag assume_completion
      @post _pausers.bearer[msg.sender] == true
      @post __post._pausers.bearer[msg.sender] == false
     */
    function renouncePauser() public {
        _removePauser(msg.sender);
    }

    /*@CTK "PauserRole__addPauser"
      @tag assume_completion
      @post account != address(0)
      @post _pausers.bearer[account] == false
      @post __post._pausers.bearer[account] == true
     */
    function _addPauser(address account) internal {
        _pausers.add(account);
        emit PauserAdded(account);
    }

    /*@CTK "PauserRole__removePauser"
      @tag assume_completion
      @post account != address(0)
      @post _pausers.bearer[account] == true
      @post __post._pausers.bearer[account] == false
     */
    function _removePauser(address account) internal {
        _pausers.remove(account);
        emit PauserRemoved(account);
    }
}

// File: openzeppelin-solidity/contracts/lifecycle/Pausable.sol

/**
 * @title Pausable
 * @dev Base contract which allows children to implement an emergency stop
 ↪  mechanism.
 */
contract Pausable is PauserRole {
    event Paused(address account);
    event Unpaused(address account);

    bool private _paused;

    /*@CTK "Pausable_constructor"
      @post __post._paused == false
     */
    constructor () internal {
        _paused = false;
    }

    /**
     * @return true if the contract is paused, false otherwise.
     */
    /*@CTK "Pausable_paused"
      @post __return == _paused
     */
    function paused() public view returns (bool) {
        return _paused;
    }

    /**
     * @dev Modifier to make a function callable only when the contract is
 ↪  not paused.
     */
    modifier whenNotPaused() {
        require(!_paused);
        _;
    }

    /**
     * @dev Modifier to make a function callable only when the contract is
↪   paused.
     */
    modifier whenPaused() {
        require(_paused);
        _;
    }

    /**
     * @dev called by the owner to pause, triggers stopped state
     */
    /*@CTK "Pausable_pause"
        @tag assume_completion
        @post _pausers.bearer[msg.sender] == true
        @post _paused == false
        @post __post._paused == true
     */
    function pause() public onlyPauser whenNotPaused {
        _paused = true;
        emit Paused(msg.sender);
    }

    /**
     * @dev called by the owner to unpause, returns to normal state
     */
    /*@CTK "Pausable_unpause"
        @tag assume_completion
        @post _pausers.bearer[msg.sender] == true
        @post _paused == true
        @post __post._paused == false
     */
    function unpause() public onlyPauser whenPaused {
        _paused = false;
        emit Unpaused(msg.sender);
    }
}

// File: openzeppelin-solidity/contracts/token/ERC20/ERC20Pausable.sol

/**
 * @title Pausable token
 * @dev ERC20 modified with pausable transfers.
 **/
contract ERC20Pausable is ERC20, Pausable {
    /*@CTK "ERC20Pausable_transfer"
      @tag assume_completion
      @post _paused == false
      @post to != address(0)
      @post to != msg.sender -> __post._balances[msg.sender] ==
↪  _balances[msg.sender] - value
      @post to != msg.sender -> __post._balances[to] == _balances[to] +
↪  value
      @post to == msg.sender -> __post._balances[msg.sender] ==
↪  _balances[msg.sender]
     */
    function transfer(address to, uint256 value) public whenNotPaused returns
↪  (bool) {
        return super.transfer(to, value);
    }

    /*@CTK "ERC20Pausable_transferFrom"
      @tag assume_completion
      @post _paused == false
      @post __post._allowed[from][msg.sender] == _allowed[from][msg.sender]
↪  - value
      @post (to == address(0)) == (__reverted)
      @post (!__reverted && to != from) -> (__post._balances[from] ==
↪  _balances[from] - value)
      @post (!__reverted && to != from) -> (__post._balances[to] ==
↪  _balances[to] + value)
      @post (!__reverted && to == from) -> (__post._balances[from] ==
↪  _balances[from])
     */
    function transferFrom(address from, address to, uint256 value) public
↪  whenNotPaused returns (bool) {
        return super.transferFrom(from, to, value);
    }

    /*@CTK ERC20Pausable_approve
        @tag assume_completion
        @post _paused == false
        @post spender != address(0)
        @post __post._allowed[msg.sender][spender] == value
    */
    function approve(address spender, uint256 value) public whenNotPaused
↪  returns (bool) {
        return super.approve(spender, value);
    }

    /*@CTK ERC20Pausable_increaseAllowance
        @tag assume_completion
        @post _paused == false
        @post spender != address(0)
        @post __post._allowed[msg.sender][spender] ==
↪  _allowed[msg.sender][spender] + addedValue
```

```
    */
    function increaseAllowance(address spender, uint addedValue) public
↪   whenNotPaused returns (bool success) {
        return super.increaseAllowance(spender, addedValue);
    }

    /*@CTK ERC20Pausable_decreaseAllowance
        @tag assume_completion
        @post _paused == false
        @post spender != address(0)
        @post __post._allowed[msg.sender][spender] ==
↪   _allowed[msg.sender][spender] - subtractedValue
    */
    function decreaseAllowance(address spender, uint subtractedValue) public
↪   whenNotPaused returns (bool success) {
        return super.decreaseAllowance(spender, subtractedValue);
    }
}

// File: openzeppelin-solidity/contracts/token/ERC20/ERC20Detailed.sol

/**
 * @title ERC20Detailed token
 * @dev The decimals are only for visualization purposes.
 * All the operations are done using the smallest and indivisible token
↪   unit,
 * just as on Ethereum all the operations are done in wei.
 */
contract ERC20Detailed is IERC20 {
    string private _name;
    string private _symbol;
    uint8 private _decimals;

    /*@CTK ERC20Detailed_constructor
        @tag assume_completion
        @post __post._name == name
        @post __post._symbol == symbol
        @post __post._decimals == decimals
    */
    constructor (string memory name, string memory symbol, uint8 decimals)
↪   public {
        _name = name;
        _symbol = symbol;
        _decimals = decimals;
    }

    /**
     * @return the name of the token.
     */
     /*@CTK ERC20Detailed_name
        @tag assume_completion
        @post __return == _name
    */
    function name() public view returns (string memory) {
        return _name;
    }

    /**
     * @return the symbol of the token.
     */
    /*@CTK ERC20Detailed_symbol
        @tag assume_completion
        @post __return == _symbol
    */
    function symbol() public view returns (string memory) {
        return _symbol;
    }

    /**
     * @return the number of decimals of the token.
     */
    /*@CTK ERC20Detailed_decimals
        @tag assume_completion
        @post __return == _decimals
    */
    function decimals() public view returns (uint8) {
        return _decimals;
    }
}

/**
 * WARNING
 * This contract is a draft for a stable coin prototype which is being
↪  designed to work in conjunction with a payment gateway.
 * At present this contract is in Beta and must not be used in production or
↪  when there is real value at stake.
 * Use this contract at your own risk.
 */

//Import contracts
//This file has been marked old because dynamically linking to these
↪  contracts was causing issues at times when OpenZeppelin updated their
↪  code base and GitHub repository file structure etc.

//Create MonetaToken contract
```

```
contract MonetaToken is ERC20Mintable, ERC20Burnable, ERC20Pausable,
↪   ERC20Detailed {
    constructor() public
    ERC20Mintable()
    ERC20Burnable()
    ERC20Pausable()
    ERC20Detailed("Moneta Stablecoin", "MMXN", 6) {}
}
```