



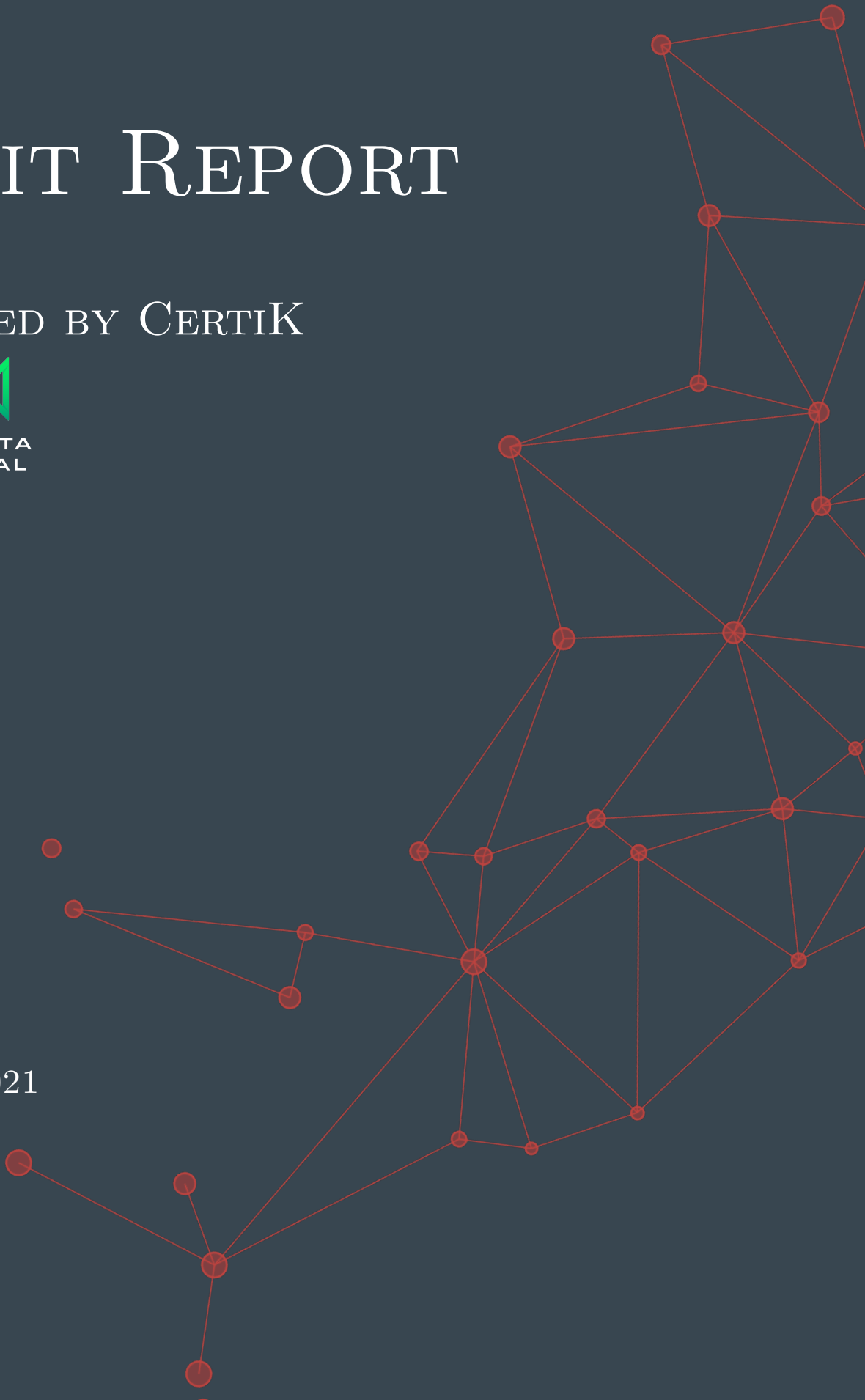
CERTIK

AUDIT REPORT

PRODUCED BY CERTIK

FOR 
MONETA
DIGITAL

JUNE 18, 2021



CERTIK AUDIT REPORT FOR MONETA DIGITAL



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



Contents

Disclaimer	1
About CertiK	2
Executive Summary	3
Vulnerability Classification	3
Testing Summary	4
Audit Score	4
Type of Issues	4
Vulnerability Details	5
Review Notes	6
Static Analysis Results	7
Formal Verification Results	8
How to read	8
Source Code with CertiK Labels	40

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. msg.sender instead.	Use 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

```
5 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 to
) {
	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
	52	}
	53	balance: 0x0
	54	}
	55	}
Post environment	56	
	57	After Execution:
	58	Input = {
	59	from = 0x0
	60	to = 0x0
	61	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

41 *//@CTK NO_OVERFLOW*

Line 50-62 in File mmxn.sol

```

50     function mul(uint256 a, uint256 b) internal pure returns (uint256) {
51         // Gas optimization: this is cheaper than requiring 'a' not being
↪      zero, but the
52         // benefit is lost if 'b' is also tested.
53         // See:
↪      https://github.com/OpenZeppelin/openzeppelin-solidity/pull/522
54         if (a == 0) {
55             return 0;
56         }
57
58         uint256 c = a * b;
59         require(c / a == b);
60
61         return c;
62     }

```

✅ The code meets the specification.

Formal Verification Request 2

SafeMath_mul

📅 18, Jun 2021

🕒 108.02 ms

Line 42-49 in File mmxn.sol

```

42     /*@CTK SafeMath_mul
43         @tag assume_completion
44         @tag spec
45         @tag is_pure
46         @post a==0 -> __return == 0
47         @post a!=0 -> ((a>0) && (a*b/a != b)) == (__reverted)
48         @post !__reverted -> __return == a * b
49     */

```

Line 50-62 in File mmxn.sol

```

50     function mul(uint256 a, uint256 b) internal pure returns (uint256) {
51         // Gas optimization: this is cheaper than requiring 'a' not being
↪ zero, but the
52         // benefit is lost if 'b' is also tested.
53         // See:
↪ https://github.com/OpenZeppelin/openzeppelin-solidity/pull/522
54         if (a == 0) {
55             return 0;
56         }
57
58         uint256 c = a * b;
59         require(c / a == b);
60
61         return c;
62     }

```

✓ 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

```

67     //OCTK NO_OVERFLOW

```

Line 75-82 in File mmxn.sol

```

75     function div(uint256 a, uint256 b) internal pure returns (uint256) {
76         // Solidity only automatically asserts when dividing by 0
77         require(b > 0);
78         uint256 c = a / b;
79         // assert(a == b * c + a % b); // There is no case in which this
↪ doesn't hold
80
81         return c;
82     }

```

✓ The code meets the specification.

Formal Verification Request 4

SafeMath_div

📅 18, Jun 2021

🕒 2.12 ms

Line 68-74 in File mmxn.sol

```
68  /*@CTK SafeMath_div
69      @tag assume_completion
70      @tag spec
71      @tag is_pure
72      @post b > 0
73      @post __return == a / b
74  */
```

Line 75-82 in File mmxn.sol

```
75  function div(uint256 a, uint256 b) internal pure returns (uint256) {
76      // Solidity only automatically asserts when dividing by 0
77      require(b > 0);
78      uint256 c = a / b;
79      // assert(a == b * c + a % b); // There is no case in which this
↪  doesn't hold
80
81      return c;
82  }
```

✓ 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

```
87  //@CTK NO_OVERFLOW
```

Line 95-100 in File mmxn.sol

```
95  function sub(uint256 a, uint256 b) internal pure returns (uint256) {
96      require(b <= a);
97      uint256 c = a - b;
98
99      return c;
100 }
```

✓ The code meets the specification.

Formal Verification Request 6

SafeMath_sub

📅 18, Jun 2021

🕒 2.09 ms

Line 88-94 in File mmxn.sol

```
88  /*@CTK SafeMath_sub
89      @tag assume_completion
90      @tag spec
91      @tag is_pure
92      @post b <= a
93      @post __return == a - b
94  */
```

Line 95-100 in File mmxn.sol

```
95  function sub(uint256 a, uint256 b) internal pure returns (uint256) {
96      require(b <= a);
97      uint256 c = a - b;
98
99      return c;
100 }
```

✓ 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

```
105  //@CTK NO_OVERFLOW
```

Line 113-118 in File mmxn.sol

```
113  function add(uint256 a, uint256 b) internal pure returns (uint256) {
114      uint256 c = a + b;
115      require(c >= a);
116
117      return c;
118  }
```

✓ The code meets the specification.

Formal Verification Request 8

SafeMath_add



18, Jun 2021



1.7 ms

Line 106-112 in File mmxn.sol

```
106  /*@CTK SafeMath_add
107      @tag assume_completion
108      @tag spec
109      @tag is_pure
110      @post (a+b < a) == (__reverted)
111      @post !__reverted -> __return == a + b
112  */
```

Line 113-118 in File mmxn.sol

```
113  function add(uint256 a, uint256 b) internal pure returns (uint256) {
114      uint256 c = a + b;
115      require(c >= a);
116
117      return c;
118  }
```

✓ 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

```
124  //@CTK NO_OVERFLOW
```

Line 132-135 in File mmxn.sol

```
132  function mod(uint256 a, uint256 b) internal pure returns (uint256) {
133      require(b != 0);
134      return a % b;
135  }
```

✓ The code meets the specification.

Formal Verification Request 10

SafeMath_mod



18, Jun 2021



1.77 ms

Line 125-131 in File mmxn.sol

```
125  /*@CTK SafeMath_mod
126      @tag assume_completion
127      @tag spec
128      @tag is_pure
```



```
129     @post b != 0
130     @post __return == a % b
131 */
```

Line 132-135 in File mmxn.sol

```
132     function mod(uint256 a, uint256 b) internal pure returns (uint256) {
133         require(b != 0);
134         return a % b;
135     }
```

✓ 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

```
164 // @CTK NO_OVERFLOW
```

Line 170-172 in File mmxn.sol

```
170     function totalSupply() public view returns (uint256) {
171         return _totalSupply;
172     }
```

✓ 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

```
165 // @CTK NO_BUF_OVERFLOW
```

Line 170-172 in File mmxn.sol

```
170     function totalSupply() public view returns (uint256) {
171         return _totalSupply;
172     }
```

✓ 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

```
166 // @CTK NO_ASF
```

Line 170-172 in File mmxn.sol

```
170 function totalSupply() public view returns (uint256) {  
171     return _totalSupply;  
172 }
```

✓ The code meets the specification.

Formal Verification Request 14

ERC20_totalSupply

18, Jun 2021

0.99 ms

Line 167-169 in File mmxn.sol

```
167 /* @CTK ERC20_totalSupply  
168     @post __return == _totalSupply  
169 */
```

Line 170-172 in File mmxn.sol

```
170 function totalSupply() public view returns (uint256) {  
171     return _totalSupply;  
172 }
```

✓ 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

```
179 // @CTK NO_OVERFLOW
```

Line 185-187 in File mmxn.sol

```
185 function balanceOf(address owner) public view returns (uint256) {  
186     return _balances[owner];  
187 }
```

✓ 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

```
180 // @CTK NO_BUF_OVERFLOW
```

Line 185-187 in File mmxn.sol

```
185 function balanceOf(address owner) public view returns (uint256) {  
186     return _balances[owner];  
187 }
```

✓ 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

```
181 // @CTK NO_ASF
```

Line 185-187 in File mmxn.sol

```
185 function balanceOf(address owner) public view returns (uint256) {  
186     return _balances[owner];  
187 }
```

✓ The code meets the specification.

Formal Verification Request 18

ERC20_balanceOf

18, Jun 2021

1.04 ms

Line 182-184 in File mmxn.sol

```
182 /* @CTK ERC20_balanceOf  
183     @post __return == _balances[owner]  
184 */
```

Line 185-187 in File mmxn.sol

```
185 function balanceOf(address owner) public view returns (uint256) {  
186     return _balances[owner];  
187 }
```

✓ 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

195 `//@CTK NO_OVERFLOW`

Line 201-203 in File mmxn.sol

```
201 function allowance(address owner, address spender) public view returns  
↪ (uint256) {  
202     return _allowed[owner][spender];  
203 }
```

✓ 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

196 `//@CTK NO_BUF_OVERFLOW`

Line 201-203 in File mmxn.sol

```
201 function allowance(address owner, address spender) public view returns  
↪ (uint256) {  
202     return _allowed[owner][spender];  
203 }
```

✓ 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

197 `//@CTK NO_ASF`

Line 201-203 in File mmxn.sol

```
201 function allowance(address owner, address spender) public view returns  
↪ (uint256) {  
202     return _allowed[owner][spender];  
203 }
```

✓ The code meets the specification.

Formal Verification Request 22

ERC20_allowance

18, Jun 2021

1.14 ms

Line 198-200 in File mmxn.sol

```
198  /*@CTK ERC20_allowance
199      @post __return == _allowed[owner][spender]
200  */
```

Line 201-203 in File mmxn.sol

```
201  function allowance(address owner, address spender) public view returns
↪  (uint256) {
202      return _allowed[owner][spender];
203  }
```

✓ The code meets the specification.

Formal Verification Request 23

ERC20_transfer

18, Jun 2021

100.41 ms

Line 210-216 in File mmxn.sol

```
210  /*@CTK "ERC20_transfer"
211      @tag assume_completion
212      @post to != address(0)
213      @post to != msg.sender -> __post._balances[msg.sender] ==
↪  _balances[msg.sender] - value
214      @post to != msg.sender -> __post._balances[to] == _balances[to] +
↪  value
215      @post to == msg.sender -> __post._balances[msg.sender] ==
↪  _balances[msg.sender]
216  */
```

Line 217-220 in File mmxn.sol

```
217  function transfer(address to, uint256 value) public returns (bool) {
218      _transfer(msg.sender, to, value);
219      return true;
220  }
```

✓ 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

231 *//@CTK NO_OVERFLOW*

Line 239-245 in File mmxn.sol

```
239     function approve(address spender, uint256 value) public returns (bool) {  
240         require(spender != address(0));  
241  
242         _allowed[msg.sender][spender] = value;  
243         emit Approval(msg.sender, spender, value);  
244         return true;  
245     }
```

✓ 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

232 *//@CTK NO_BUF_OVERFLOW*

Line 239-245 in File mmxn.sol

```
239     function approve(address spender, uint256 value) public returns (bool) {  
240         require(spender != address(0));  
241  
242         _allowed[msg.sender][spender] = value;  
243         emit Approval(msg.sender, spender, value);  
244         return true;  
245     }
```

✓ 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

233

//@CTK NO_ASF

Line 239-245 in File mmxn.sol

```
239     function approve(address spender, uint256 value) public returns (bool) {
240         require(spender != address(0));
241
242         _allowed[msg.sender][spender] = value;
243         emit Approval(msg.sender, spender, value);
244         return true;
245     }
```

 The code meets the specification.

Formal Verification Request 27

ERC20_approve



18, Jun 2021



2.42 ms

Line 234-238 in File mmxn.sol

```
234     /*@CTK ERC20_approve
235         @tag assume_completion
236         @post spender != address(0)
237         @post __post._allowed[msg.sender][spender] == value
238     */
```

Line 239-245 in File mmxn.sol

```
239     function approve(address spender, uint256 value) public returns (bool) {
240         require(spender != address(0));
241
242         _allowed[msg.sender][spender] = value;
243         emit Approval(msg.sender, spender, value);
244         return true;
245     }
```

 The code meets the specification.

Formal Verification Request 28

ERC20_transferFrom



18, Jun 2021



133.18 ms

Line 255-262 in File mmxn.sol

```

255     /*@CTK "ERC20_transferFrom"
256         @tag assume_completion
257         @post __post._allowed[from][msg.sender] == _allowed[from][msg.sender]
↪ - value
258         @post (to == address(0)) == (__reverted)
259         @post (!__reverted && to != from) -> (__post._balances[from] ==
↪ _balances[from] - value)
260         @post (!__reverted && to != from) -> (__post._balances[to] ==
↪ _balances[to] + value)
261         @post (!__reverted && to == from) -> (__post._balances[from] ==
↪ _balances[from])
262     */

```

Line 263-268 in File mmxn.sol

```

263     function transferFrom(address from, address to, uint256 value) public
↪ returns (bool) {
264         _allowed[from][msg.sender] = _allowed[from][msg.sender].sub(value);
265         _transfer(from, to, value);
266         emit Approval(from, msg.sender, _allowed[from][msg.sender]);
267         return true;
268     }

```

✓ The code meets the specification.

Formal Verification Request 29

ERC20__increaseAllowance



18, Jun 2021



24.28 ms

Line 280-284 in File mmxn.sol

```

280     /*@CTK ERC20__increaseAllowance
281         @tag assume_completion
282         @post spender != address(0)
283         @post __post._allowed[msg.sender][spender] ==
↪ _allowed[msg.sender][spender] + addedValue
284     */

```

Line 285-291 in File mmxn.sol

```

285     function increaseAllowance(address spender, uint256 addedValue) public
↪ returns (bool) {
286         require(spender != address(0));
287
288         _allowed[msg.sender][spender] =
↪ _allowed[msg.sender][spender].add(addedValue);
289         emit Approval(msg.sender, spender, _allowed[msg.sender][spender]);
290         return true;
291     }

```

✓ The code meets the specification.

Formal Verification Request 30

ERC20_decreaseAllowance

18, Jun 2021

25.9 ms

Line 303-307 in File mmxn.sol

```

303  /*@CTK ERC20_decreaseAllowance
304      @tag assume_completion
305      @post spender != address(0)
306      @post __post._allowed[msg.sender][spender] ==
↪   _allowed[msg.sender][spender] - subtractedValue
307  */

```

Line 308-314 in File mmxn.sol

```

308  function decreaseAllowance(address spender, uint256 subtractedValue)
↪  public returns (bool) {
309      require(spender != address(0));
310
311      _allowed[msg.sender][spender] =
↪  _allowed[msg.sender][spender].sub(subtractedValue);
312      emit Approval(msg.sender, spender, _allowed[msg.sender][spender]);
313      return true;
314  }

```

✓ The code meets the specification.

Formal Verification Request 31

ERC20__transfer

18, Jun 2021

28.99 ms

Line 322-328 in File mmxn.sol

```

322  /*@CTK "ERC20__transfer"
323      @tag assume_completion
324      @post to != address(0)
325      @post to != from -> __post._balances[from] == _balances[from] - value
326      @post to != from -> __post._balances[to] == _balances[to] + value
327      @post to == from -> __post._balances[from] == _balances[from]
328  */

```

Line 329-335 in File mmxn.sol

```

329  function _transfer(address from, address to, uint256 value) internal {
330      require(to != address(0));
331

```

```
332     _balances[from] = _balances[from].sub(value);
333     _balances[to] = _balances[to].add(value);
334     emit Transfer(from, to, value);
335 }
```

✓ The code meets the specification.

Formal Verification Request 32

ERC20__mint

📅 18, Jun 2021

🕒 36.61 ms

Line 344-349 in File mmxn.sol

```
344  /*@CTK "ERC20__mint"
345     @tag assume_completion
346     @post account != address(0)
347     @post __post._totalSupply == _totalSupply + value
348     @post __post._balances[account] == _balances[account] + value
349  */
```

Line 350-356 in File mmxn.sol

```
350  function _mint(address account, uint256 value) internal {
351      require(account != address(0));
352
353      _totalSupply = _totalSupply.add(value);
354      _balances[account] = _balances[account].add(value);
355      emit Transfer(address(0), account, value);
356  }
```

✓ The code meets the specification.

Formal Verification Request 33

ERC20__burn

📅 18, Jun 2021

🕒 40.17 ms

Line 364-369 in File mmxn.sol

```
364  /*@CTK "ERC20__burn"
365     @tag assume_completion
366     @post account != address(0)
367     @post __post._totalSupply == _totalSupply - value
368     @post __post._balances[account] == _balances[account] - value
369  */
```

Line 370-376 in File mmxn.sol

```

370     function _burn(address account, uint256 value) internal {
371         require(account != address(0));
372
373         _totalSupply = _totalSupply.sub(value);
374         _balances[account] = _balances[account].sub(value);
375         emit Transfer(account, address(0), value);
376     }

```

✓ The code meets the specification.

Formal Verification Request 34

ERC20__burnFrom

📅 18, Jun 2021

🕒 114.21 ms

Line 386-392 in File mmxn.sol

```

386     /*@CTK "ERC20__burnFrom"
387         @tag assume_completion
388         @post __post._allowed[account][msg.sender] ==
↪     _allowed[account][msg.sender] - value
389         @post (account == address(0)) == __reverted
390         @post !__reverted -> __post._totalSupply == _totalSupply - value
391         @post !__reverted -> __post._balances[account] == _balances[account] -
↪     value
392     */

```

Line 393-397 in File mmxn.sol

```

393     function _burnFrom(address account, uint256 value) internal {
394         _allowed[account][msg.sender] =
↪     _allowed[account][msg.sender].sub(value);
395         _burn(account, value);
396         emit Approval(account, msg.sender, _allowed[account][msg.sender]);
397     }

```

✓ The code meets the specification.

Formal Verification Request 35

Roles_add

📅 18, Jun 2021

🕒 45.52 ms

Line 414-419 in File mmxn.sol

```

414     /*@CTK "Roles_add"
415         @tag assume_completion

```

```
416     @post account != address(0)
417     @post role.bearer[account] == false
418     @post role__post.bearer[account] == true
419     */
```

Line 420-425 in File mmxn.sol

```
420     function add(Role storage role, address account) internal {
421         require(account != address(0));
422         require(!has(role, account));
423
424         role.bearer[account] = true;
425     }
```

✓ The code meets the specification.

Formal Verification Request 36

Roles_remove



18, Jun 2021



40.49 ms

Line 430-435 in File mmxn.sol

```
430     /*@CTK "Roles_remove"
431     @tag assume_completion
432     @post account != address(0)
433     @post role.bearer[account] == true
434     @post role__post.bearer[account] == false
435     */
```

Line 436-441 in File mmxn.sol

```
436     function remove(Role storage role, address account) internal {
437         require(account != address(0));
438         require(has(role, account));
439
440         role.bearer[account] = false;
441     }
```

✓ The code meets the specification.

Formal Verification Request 37

Roles_has



18, Jun 2021



2.94 ms

Line 447-451 in File mmxn.sol

```
447  /*@CTK "Roles_has"
448      @tag assume_completion
449      @post account != address(0)
450      @post __return == role.bearer[account]
451  */
```

Line 452-455 in File mmxn.sol

```
452  function has(Role storage role, address account) internal view returns
↪  (bool) {
453      require(account != address(0));
454      return role.bearer[account];
455  }
```

✓ The code meets the specification.

Formal Verification Request 38

MinterRole_constructor

📅 18, Jun 2021

🕒 109.48 ms

Line 468-472 in File mmxn.sol

```
468  /*@CTK "MinterRole_constructor"
469      @tag assume_completion
470      @post _minters.bearer[msg.sender] == false
471      @post __post._minters.bearer[msg.sender] == true
472  */
```

Line 473-475 in File mmxn.sol

```
473  constructor () internal {
474      _addMinter(msg.sender);
475  }
```

✓ The code meets the specification.

Formal Verification Request 39

MinterRole_isMinter

📅 18, Jun 2021

🕒 32.89 ms

Line 482-486 in File mmxn.sol

```
482  /*@CTK "MinterRole_isMinter"
483      @tag assume_completion
484      @post account != address(0)
485      @post __return == _minters.bearer[account]
486  */
```

Line 487-489 in File mmxn.sol

```
487     function isMinter(address account) public view returns (bool) {  
488         return _minters.has(account);  
489     }
```

✓ The code meets the specification.

Formal Verification Request 40

MinterRole_addMinter



18, Jun 2021



97.75 ms

Line 491-497 in File mmxn.sol

```
491     /*@CTK "MinterRole_addMinter"  
492         @tag assume_completion  
493         @post _minters.bearer[msg.sender] == true  
494         @post account != address(0)  
495         @post _minters.bearer[account] == false  
496         @post __post._minters.bearer[account] == true  
497     */
```

Line 498-500 in File mmxn.sol

```
498     function addMinter(address account) public onlyMinter {  
499         _addMinter(account);  
500     }
```

✓ The code meets the specification.

Formal Verification Request 41

MinterRole_renounceMinter



18, Jun 2021



109.27 ms

Line 502-506 in File mmxn.sol

```
502     /*@CTK "MinterRole_renounceMinter"  
503         @tag assume_completion  
504         @post _minters.bearer[msg.sender] == true  
505         @post __post._minters.bearer[msg.sender] == false  
506     */
```

Line 507-509 in File mmxn.sol

```
507     function renounceMinter() public {  
508         _removeMinter(msg.sender);  
509     }
```

✓ The code meets the specification.

Formal Verification Request 42

MinterRole__addMinter

18, Jun 2021

3.81 ms

Line 511-516 in File mmxn.sol

```
511    /*@CTK "MinterRole__addMinter"
512       @tag assume_completion
513       @post account != address(0)
514       @post _minters.bearer[account] == false
515       @post __post._minters.bearer[account] == true
516    */
```

Line 517-520 in File mmxn.sol

```
517    function _addMinter(address account) internal {
518        _minters.add(account);
519        emit MinterAdded(account);
520    }
```

✓ The code meets the specification.

Formal Verification Request 43

MinterRole__removeMinter

18, Jun 2021

3.76 ms

Line 522-527 in File mmxn.sol

```
522    /*@CTK "MinterRole__removeMinter"
523       @tag assume_completion
524       @post account != address(0)
525       @post _minters.bearer[account] == true
526       @post __post._minters.bearer[account] == false
527    */
```

Line 528-531 in File mmxn.sol

```
528    function _removeMinter(address account) internal {
529        _minters.remove(account);
530        emit MinterRemoved(account);
531    }
```

✓ The code meets the specification.

Formal Verification Request 44

ERC20Mintable__mint

18, Jun 2021

175.54 ms

Line 547-553 in File mmxn.sol

```
547  /*@CTK "ERC20Mintable__mint"
548      @tag assume_completion
549      @post _minters.bearer[msg.sender] == true
550      @post to != address(0)
551      @post __post._totalSupply == _totalSupply + value
552      @post __post._balances[to] == _balances[to] + value
553  */
```

Line 554-557 in File mmxn.sol

```
554  function mint(address to, uint256 value) public onlyMinter returns (bool)
↪  {
555      _mint(to, value);
556      return true;
557  }
```

✓ The code meets the specification.

Formal Verification Request 45

ERC20Burnable__burn

18, Jun 2021

76.01 ms

Line 571-575 in File mmxn.sol

```
571  /*@CTK "ERC20Burnable__burn"
572      @tag assume_completion
573      @post __post._totalSupply == _totalSupply - value
574      @post __post._balances[msg.sender] == _balances[msg.sender] - value
575  */
```

Line 576-578 in File mmxn.sol

```
576  function burn(uint256 value) public {
577      _burn(msg.sender, value);
578  }
```

✓ The code meets the specification.

Formal Verification Request 46

ERC20Burnable__burnFrom

18, Jun 2021

174.03 ms

Line 585-591 in File mmxn.sol

```
585      /*@CTK "ERC20Burnable__burnFrom"
586         @tag assume_completion
587         @post __post._allowed[from][msg.sender] == _allowed[from][msg.sender]
↪      - value
588         @post (from == address(0)) == __reverted
589         @post !__reverted -> __post._totalSupply == _totalSupply - value
590         @post !__reverted -> __post._balances[from] == _balances[from] - value
591      */
```

Line 592-594 in File mmxn.sol

```
592      function burnFrom(address from, uint256 value) public {
593          _burnFrom(from, value);
594      }
```

✓ The code meets the specification.

Formal Verification Request 47

PauserRole_constructor

18, Jun 2021

95.72 ms

Line 607-611 in File mmxn.sol

```
607      /*@CTK "PauserRole_constructor"
608         @tag assume_completion
609         @post _pausers.bearer[msg.sender] == false
610         @post __post._pausers.bearer[msg.sender] == true
611      */
```

Line 612-614 in File mmxn.sol

```
612      constructor () internal {
613          _addPauser(msg.sender);
614      }
```

✓ The code meets the specification.

Formal Verification Request 48

PauserRole_isPauser

18, Jun 2021

29.68 ms

Line 621-625 in File mmxn.sol

```
621  /*@CTK "PauserRole_isPauser"
622     @tag assume_completion
623     @post account != address(0)
624     @post __return == _pausers.bearer[account]
625  */
```

Line 626-628 in File mmxn.sol

```
626  function isPauser(address account) public view returns (bool) {
627      return _pausers.has(account);
628  }
```

✓ The code meets the specification.

Formal Verification Request 49

PauserRole_addPauser

18, Jun 2021

95.07 ms

Line 630-636 in File mmxn.sol

```
630  /*@CTK "PauserRole_addPauser"
631     @tag assume_completion
632     @post _pausers.bearer[msg.sender] == true
633     @post account != address(0)
634     @post _pausers.bearer[account] == false
635     @post __post._pausers.bearer[account] == true
636  */
```

Line 637-639 in File mmxn.sol

```
637  function addPauser(address account) public onlyPauser {
638      _addPauser(account);
639  }
```

✓ The code meets the specification.

Formal Verification Request 50

PauserRole_renouncePauser

18, Jun 2021

100.12 ms

Line 641-645 in File mmxn.sol

```
641    /*@CTK "PauserRole_renouncePauser"  
642       @tag assume_completion  
643       @post _pausers.bearer[msg.sender] == true  
644       @post __post._pausers.bearer[msg.sender] == false  
645    */
```

Line 646-648 in File mmxn.sol

```
646    function renouncePauser() public {  
647        _removePauser(msg.sender);  
648    }
```

✓ The code meets the specification.

Formal Verification Request 51

PauserRole__addPauser



18, Jun 2021



3.76 ms

Line 650-655 in File mmxn.sol

```
650    /*@CTK "PauserRole__addPauser"  
651       @tag assume_completion  
652       @post account != address(0)  
653       @post _pausers.bearer[account] == false  
654       @post __post._pausers.bearer[account] == true  
655    */
```

Line 656-659 in File mmxn.sol

```
656    function _addPauser(address account) internal {  
657        _pausers.add(account);  
658        emit PauserAdded(account);  
659    }
```

✓ The code meets the specification.

Formal Verification Request 52

PauserRole__removePauser



18, Jun 2021



4.06 ms

Line 661-666 in File mmxn.sol

```
661  /*@CTK "PauserRole__removePauser"
662      @tag assume_completion
663      @post account != address(0)
664      @post _pausers.bearer[account] == true
665      @post __post._pausers.bearer[account] == false
666  */
```

Line 667-670 in File mmxn.sol

```
667  function _removePauser(address account) internal {
668      _pausers.remove(account);
669      emit PauserRemoved(account);
670  }
```

✓ The code meets the specification.

Formal Verification Request 53

Pausable__constructor



18, Jun 2021



7.36 ms

Line 685-687 in File mmxn.sol

```
685  /*@CTK "Pausable_constructor"
686      @post __post._paused == false
687  */
```

Line 688-690 in File mmxn.sol

```
688  constructor () internal {
689      _paused = false;
690  }
```

✓ The code meets the specification.

Formal Verification Request 54

Pausable__paused



18, Jun 2021



6.16 ms

Line 695-697 in File mmxn.sol

```
695  /*@CTK "Pausable_paused"
696      @post __return == _paused
697  */
```

Line 698-700 in File mmxn.sol

```
698  function paused() public view returns (bool) {
699      return _paused;
700  }
```

✓ The code meets the specification.

Formal Verification Request 55

Pausable_pause

18, Jun 2021

78.88 ms

Line 721-726 in File mmxn.sol

```
721      /*@CTK "Pausable_pause"
722         @tag assume_completion
723         @post _pausers.bearer[msg.sender] == true
724         @post _paused == false
725         @post __post._paused == true
726     */
```

Line 727-730 in File mmxn.sol

```
727     function pause() public onlyPauser whenNotPaused {
728         _paused = true;
729         emit Paused(msg.sender);
730     }
```

✓ The code meets the specification.

Formal Verification Request 56

Pausable_unpause

18, Jun 2021

53.41 ms

Line 735-740 in File mmxn.sol

```
735      /*@CTK "Pausable_unpause"
736         @tag assume_completion
737         @post _pausers.bearer[msg.sender] == true
738         @post _paused == true
739         @post __post._paused == false
740     */
```

Line 741-744 in File mmxn.sol

```
741     function unpause() public onlyPauser whenPaused {
742         _paused = false;
743         emit Unpaused(msg.sender);
744     }
```

✓ The code meets the specification.

Formal Verification Request 57

ERC20Pausable_transfer

18, Jun 2021

197.08 ms

Line 754-761 in File mmxn.sol

```

754  /*@CTK "ERC20Pausable_transfer"
755      @tag assume_completion
756      @post _paused == false
757      @post to != address(0)
758      @post to != msg.sender -> __post._balances[msg.sender] ==
↪  _balances[msg.sender] - value
759      @post to != msg.sender -> __post._balances[to] == _balances[to] +
↪  value
760      @post to == msg.sender -> __post._balances[msg.sender] ==
↪  _balances[msg.sender]
761  */

```

Line 762-764 in File mmxn.sol

```

762  function transfer(address to, uint256 value) public whenNotPaused returns
↪  (bool) {
763      return super.transfer(to, value);
764  }

```

✓ The code meets the specification.

Formal Verification Request 58

ERC20Pausable_transferFrom

18, Jun 2021

205.05 ms

Line 766-774 in File mmxn.sol

```

766  /*@CTK "ERC20Pausable_transferFrom"
767      @tag assume_completion
768      @post _paused == false
769      @post __post._allowed[from][msg.sender] == _allowed[from][msg.sender]
↪  - value
770      @post (to == address(0)) == (__reverted)
771      @post (!__reverted && to != from) -> (__post._balances[from] ==
↪  _balances[from] - value)
772      @post (!__reverted && to != from) -> (__post._balances[to] ==
↪  _balances[to] + value)
773      @post (!__reverted && to == from) -> (__post._balances[from] ==
↪  _balances[from])
774  */

```

Line 775-777 in File mmxn.sol

```
775     function transferFrom(address from, address to, uint256 value) public
↪    whenNotPaused returns (bool) {
776         return super.transferFrom(from, to, value);
777     }
```

✓ The code meets the specification.

Formal Verification Request 59

ERC20Pausable_approve

📅 18, Jun 2021

🕒 52.3 ms

Line 779-784 in File mmxn.sol

```
779     /*@CTK ERC20Pausable_approve
780         @tag assume_completion
781         @post _paused == false
782         @post spender != address(0)
783         @post __post._allowed[msg.sender][spender] == value
784     */
```

Line 785-787 in File mmxn.sol

```
785     function approve(address spender, uint256 value) public whenNotPaused
↪    returns (bool) {
786         return super.approve(spender, value);
787     }
```

✓ The code meets the specification.

Formal Verification Request 60

ERC20Pausable_increaseAllowance

📅 18, Jun 2021

🕒 75.27 ms

Line 789-794 in File mmxn.sol

```
789     /*@CTK ERC20Pausable_increaseAllowance
790         @tag assume_completion
791         @post _paused == false
792         @post spender != address(0)
793         @post __post._allowed[msg.sender][spender] ==
↪    _allowed[msg.sender][spender] + addedValue
794     */
```

Line 795-797 in File mmxn.sol

```

795     function increaseAllowance(address spender, uint addedValue) public
↪    whenNotPaused returns (bool success) {
796         return super.increaseAllowance(spender, addedValue);
797     }

```

✓ The code meets the specification.

Formal Verification Request 61

ERC20Pausable_decreaseAllowance

📅 18, Jun 2021

🕒 73.71 ms

Line 799-804 in File mmxn.sol

```

799     /*@CTK ERC20Pausable_decreaseAllowance
800         @tag assume_completion
801         @post _paused == false
802         @post spender != address(0)
803         @post __post._allowed[msg.sender][spender] ==
↪    _allowed[msg.sender][spender] - subtractedValue
804     */

```

Line 805-807 in File mmxn.sol

```

805     function decreaseAllowance(address spender, uint subtractedValue) public
↪    whenNotPaused returns (bool success) {
806         return super.decreaseAllowance(spender, subtractedValue);
807     }

```

✓ The code meets the specification.

Formal Verification Request 62

ERC20Detailed_constructor

📅 18, Jun 2021

🕒 9.76 ms

Line 823-828 in File mmxn.sol

```

823     /*@CTK ERC20Detailed_constructor
824         @tag assume_completion
825         @post __post._name == name
826         @post __post._symbol == symbol
827         @post __post._decimals == decimals
828     */

```

Line 829-833 in File mmxn.sol


```
829     constructor (string memory name, string memory symbol, uint8 decimals)
↪   public {
830         _name = name;
831         _symbol = symbol;
832         _decimals = decimals;
833     }
```

✓ The code meets the specification.

Formal Verification Request 63

ERC20Detailed_name



18, Jun 2021



6.28 ms

Line 838-841 in File mmxn.sol

```
838     /*@CTK ERC20Detailed_name
839         @tag assume_completion
840         @post __return == _name
841     */
```

Line 842-844 in File mmxn.sol

```
842     function name() public view returns (string memory) {
843         return _name;
844     }
```

✓ The code meets the specification.

Formal Verification Request 64

ERC20Detailed_symbol



18, Jun 2021



4.9 ms

Line 849-852 in File mmxn.sol

```
849     /*@CTK ERC20Detailed_symbol
850         @tag assume_completion
851         @post __return == _symbol
852     */
```

Line 853-855 in File mmxn.sol

```
853     function symbol() public view returns (string memory) {
854         return _symbol;
855     }
```

✓ The code meets the specification.

Formal Verification Request 65

ERC20Detailed_decimals



18, Jun 2021



5.44 ms

Line 860-863 in File mmxn.sol

```
860      /*@CTK ERC20Detailed_decimals
861         @tag assume_completion
862         @post __return == _decimals
863      */
```

Line 864-866 in File mmxn.sol

```
864      function decimals() public view returns (uint8) {
865          return _decimals;
866      }
```



The code meets the specification.

Source Code with CertiK Labels

mmxn.sol

```

1  /**
2   *Submitted for verification at Etherscan.io on 2021-06-16
3   */
4
5  pragma solidity ^0.4.25;
6
7  // File: openzeppelin-solidity/contracts/token/ERC20/IERC20.sol
8
9  /**
10   * @title ERC20 interface
11   * @dev see https://github.com/ethereum/EIPs/issues/20
12   */
13  interface IERC20 {
14      function transfer(address to, uint256 value) external returns (bool);
15
16      function approve(address spender, uint256 value) external returns (bool);
17
18      function transferFrom(address from, address to, uint256 value) external
19      ↪ returns (bool);
20
21      function totalSupply() external view returns (uint256);
22
23      function balanceOf(address who) external view returns (uint256);
24
25      function allowance(address owner, address spender) external view returns
26      ↪ (uint256);
27
28      event Transfer(address indexed from, address indexed to, uint256 value);
29
30      event Approval(address indexed owner, address indexed spender, uint256
31      ↪ value);
32  }
33
34  // File: openzeppelin-solidity/contracts/math/SafeMath.sol
35
36  /**
37   * @title SafeMath
38   * @dev Unsigned math operations with safety checks that revert on error
39   */
40
41  library SafeMath {
42      /**
43       * @dev Multiplies two unsigned integers, reverts on overflow.
44       */
45      //@CTK NO_OVERFLOW

```

```

42  /*@CTK SafeMath_mul
43      @tag assume_completion
44      @tag spec
45      @tag is_pure
46      @post a==0 -> __return == 0
47      @post a!=0 -> ((a>0) && (a*b/a != b)) == (__reverted)
48      @post !__reverted -> __return == a * b
49  */
50  function mul(uint256 a, uint256 b) internal pure returns (uint256) {
51      // Gas optimization: this is cheaper than requiring 'a' not being
↪ zero, but the
52      // benefit is lost if 'b' is also tested.
53      // See:
↪ https://github.com/OpenZeppelin/openzeppelin-solidity/pull/522
54      if (a == 0) {
55          return 0;
56      }
57
58      uint256 c = a * b;
59      require(c / a == b);
60
61      return c;
62  }
63
64  /**
65   * @dev Integer division of two unsigned integers truncating the
↪ quotient, reverts on division by zero.
66  */
67  //@CTK NO_OVERFLOW
68  /*@CTK SafeMath_div
69      @tag assume_completion
70      @tag spec
71      @tag is_pure
72      @post b > 0
73      @post __return == a / b
74  */
75  function div(uint256 a, uint256 b) internal pure returns (uint256) {
76      // Solidity only automatically asserts when dividing by 0
77      require(b > 0);
78      uint256 c = a / b;
79      // assert(a == b * c + a % b); // There is no case in which this
↪ doesn't hold
80
81      return c;
82  }
83
84  /**

```

```

85      * @dev Subtracts two unsigned integers, reverts on overflow (i.e. if
↪      subtrahend is greater than minuend).
86      */
87      //@CTK NO_OVERFLOW
88      /*@CTK SafeMath_sub
89          @tag assume_completion
90          @tag spec
91          @tag is_pure
92          @post b <= a
93          @post __return == a - b
94      */
95      function sub(uint256 a, uint256 b) internal pure returns (uint256) {
96          require(b <= a);
97          uint256 c = a - b;
98
99          return c;
100     }
101
102     /**
103     * @dev Adds two unsigned integers, reverts on overflow.
104     */
105     //@CTK NO_OVERFLOW
106     /*@CTK SafeMath_add
107         @tag assume_completion
108         @tag spec
109         @tag is_pure
110         @post (a+b < a) == (__reverted)
111         @post !__reverted -> __return == a + b
112     */
113     function add(uint256 a, uint256 b) internal pure returns (uint256) {
114         uint256 c = a + b;
115         require(c >= a);
116
117         return c;
118     }
119
120     /**
121     * @dev Divides two unsigned integers and returns the remainder (unsigned
↪     integer modulo),
122     * reverts when dividing by zero.
123     */
124     //@CTK NO_OVERFLOW
125     /*@CTK SafeMath_mod
126         @tag assume_completion
127         @tag spec
128         @tag is_pure
129         @post b != 0
130         @post __return == a % b

```

```

131     */
132     function mod(uint256 a, uint256 b) internal pure returns (uint256) {
133         require(b != 0);
134         return a % b;
135     }
136 }
137
138 // File: openzeppelin-solidity/contracts/token/ERC20/ERC20.sol
139
140 /**
141  * @title Standard ERC20 token
142  *
143  * @dev Implementation of the basic standard token.
144  * https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20.md
145  * Originally based on code by FirstBlood:
146  * https://github.com/Firstbloodio/token/blob/master/smart_contract/FirstBloodToken.sol
147  *
148  * This implementation emits additional Approval events, allowing
149  * applications to reconstruct the allowance status for
150  * all accounts just by listening to said events. Note that this isn't
151  * required by the specification, and other
152  * compliant implementations may not do it.
153  */
154 contract ERC20 is IERC20 {
155     using SafeMath for uint256;
156
157     mapping (address => uint256) private _balances;
158
159     mapping (address => mapping (address => uint256)) private _allowed;
160
161     uint256 private _totalSupply;
162
163     /**
164      * @dev Total number of tokens in existence
165      */
166     // @CTK NO_OVERFLOW
167     // @CTK NO_BUF_OVERFLOW
168     // @CTK NO_ASF
169     /* @CTK ERC20_totalSupply
170      * @post __return == _totalSupply
171      */
172     function totalSupply() public view returns (uint256) {
173         return _totalSupply;
174     }
175
176     /**
177      * @dev Gets the balance of the specified address.

```

```

176     * @param owner The address to query the balance of.
177     * @return An uint256 representing the amount owned by the passed
↪    address.
178     */
179     //@CTK NO_OVERFLOW
180     //@CTK NO_BUF_OVERFLOW
181     //@CTK NO_ASF
182     /*@CTK ERC20_balanceOf
183         @post __return == _balances[owner]
184     */
185     function balanceOf(address owner) public view returns (uint256) {
186         return _balances[owner];
187     }
188
189     /**
190     * @dev Function to check the amount of tokens that an owner allowed to
↪    a spender.
191     * @param owner address The address which owns the funds.
192     * @param spender address The address which will spend the funds.
193     * @return A uint256 specifying the amount of tokens still available for
↪    the spender.
194     */
195     //@CTK NO_OVERFLOW
196     //@CTK NO_BUF_OVERFLOW
197     //@CTK NO_ASF
198     /*@CTK ERC20_allowance
199         @post __return == _allowed[owner][spender]
200     */
201     function allowance(address owner, address spender) public view returns
↪    (uint256) {
202         return _allowed[owner][spender];
203     }
204
205     /**
206     * @dev Transfer token for a specified address
207     * @param to The address to transfer to.
208     * @param value The amount to be transferred.
209     */
210     /*@CTK "ERC20_transfer"
211         @tag assume_completion
212         @post to != address(0)
213         @post to != msg.sender -> __post._balances[msg.sender] ==
↪    _balances[msg.sender] - value
214         @post to != msg.sender -> __post._balances[to] == _balances[to] +
↪    value
215         @post to == msg.sender -> __post._balances[msg.sender] ==
↪    _balances[msg.sender]
216     */

```

```

217     function transfer(address to, uint256 value) public returns (bool) {
218         _transfer(msg.sender, to, value);
219         return true;
220     }
221
222     /**
223      * @dev Approve the passed address to spend the specified amount of
224      ↪ tokens on behalf of msg.sender.
225      * Beware that changing an allowance with this method brings the risk
226      ↪ that someone may use both the old
227      ↪ * and the new allowance by unfortunate transaction ordering. One
228      ↪ possible solution to mitigate this
229      ↪ * race condition is to first reduce the spender's allowance to 0 and
230      ↪ set the desired value afterwards:
231      ↪ * https://github.com/ethereum/EIPs/issues/20#issuecomment-263524729
232      ↪ * @param spender The address which will spend the funds.
233      ↪ * @param value The amount of tokens to be spent.
234      */
235     // @CTK NO_OVERFLOW
236     // @CTK NO_BUF_OVERFLOW
237     // @CTK NO_ASF
238     /* @CTK ERC20_approve
239      @tag assume_completion
240      @post spender != address(0)
241      @post __post._allowed[msg.sender][spender] == value
242      */
243     function approve(address spender, uint256 value) public returns (bool) {
244         require(spender != address(0));
245
246         _allowed[msg.sender][spender] = value;
247         emit Approval(msg.sender, spender, value);
248         return true;
249     }
250
251     /**
252      * @dev Transfer tokens from one address to another.
253      * Note that while this function emits an Approval event, this is not
254      ↪ required as per the specification,
255      ↪ * and other compliant implementations may not emit the event.
256      ↪ * @param from address The address which you want to send tokens from
257      ↪ * @param to address The address which you want to transfer to
258      ↪ * @param value uint256 the amount of tokens to be transferred
259      */
260     /* @CTK "ERC20_transferFrom"
261      @tag assume_completion
262      @post __post._allowed[from][msg.sender] == _allowed[from][msg.sender]
263      ↪ - value
264      @post (to == address(0)) == (__reverted)

```



```

259     @post (!__reverted && to != from) -> (__post._balances[from] ==
↪   _balances[from] - value)
260     @post (!__reverted && to != from) -> (__post._balances[to] ==
↪   _balances[to] + value)
261     @post (!__reverted && to == from) -> (__post._balances[from] ==
↪   _balances[from])
262     */
263     function transferFrom(address from, address to, uint256 value) public
↪   returns (bool) {
264         _allowed[from][msg.sender] = _allowed[from][msg.sender].sub(value);
265         _transfer(from, to, value);
266         emit Approval(from, msg.sender, _allowed[from][msg.sender]);
267         return true;
268     }
269
270     /**
271     * @dev Increase the amount of tokens that an owner allowed to a
↪   spender.
272     * approve should be called when allowed_[spender] == 0. To increment
273     * allowed value is better to use this function to avoid 2 calls (and
↪   wait until
274     * the first transaction is mined)
275     * From MonolithDAO Token.sol
276     * Emits an Approval event.
277     * @param spender The address which will spend the funds.
278     * @param addedValue The amount of tokens to increase the allowance by.
279     */
280     /*@CTK ERC20_increaseAllowance
281     @tag assume_completion
282     @post spender != address(0)
283     @post __post._allowed[msg.sender][spender] ==
↪   _allowed[msg.sender][spender] + addedValue
284     */
285     function increaseAllowance(address spender, uint256 addedValue) public
↪   returns (bool) {
286         require(spender != address(0));
287
288         _allowed[msg.sender][spender] =
↪   _allowed[msg.sender][spender].add(addedValue);
289         emit Approval(msg.sender, spender, _allowed[msg.sender][spender]);
290         return true;
291     }
292
293     /**
294     * @dev Decrease the amount of tokens that an owner allowed to a
↪   spender.
295     * approve should be called when allowed_[spender] == 0. To decrement

```

```

296     * allowed value is better to use this function to avoid 2 calls (and
↪    wait until
297     * the first transaction is mined)
298     * From MonolithDAO Token.sol
299     * Emits an Approval event.
300     * @param spender The address which will spend the funds.
301     * @param subtractedValue The amount of tokens to decrease the allowance
↪    by.
302     */
303     /*@CTK ERC20_decreaseAllowance
304         @tag assume_completion
305         @post spender != address(0)
306         @post __post._allowed[msg.sender][spender] ==
↪    _allowed[msg.sender][spender] - subtractedValue
307     */
308     function decreaseAllowance(address spender, uint256 subtractedValue)
↪    public returns (bool) {
309         require(spender != address(0));
310
311         _allowed[msg.sender][spender] =
↪    _allowed[msg.sender][spender].sub(subtractedValue);
312         emit Approval(msg.sender, spender, _allowed[msg.sender][spender]);
313         return true;
314     }
315
316     /**
317     * @dev Transfer token for a specified addresses
318     * @param from The address to transfer from.
319     * @param to The address to transfer to.
320     * @param value The amount to be transferred.
321     */
322     /*@CTK "ERC20__transfer"
323         @tag assume_completion
324         @post to != address(0)
325         @post to != from -> __post._balances[from] == _balances[from] - value
326         @post to != from -> __post._balances[to] == _balances[to] + value
327         @post to == from -> __post._balances[from] == _balances[from]
328     */
329     function _transfer(address from, address to, uint256 value) internal {
330         require(to != address(0));
331
332         _balances[from] = _balances[from].sub(value);
333         _balances[to] = _balances[to].add(value);
334         emit Transfer(from, to, value);
335     }
336
337     /**

```

```

338     * @dev Internal function that mints an amount of the token and assigns
↪ it to
339     * an account. This encapsulates the modification of balances such that
↪ the
340     * proper events are emitted.
341     * @param account The account that will receive the created tokens.
342     * @param value The amount that will be created.
343     */
344     /*@CTK "ERC20__mint"
345     @tag assume_completion
346     @post account != address(0)
347     @post __post._totalSupply == _totalSupply + value
348     @post __post._balances[account] == _balances[account] + value
349     */
350     function _mint(address account, uint256 value) internal {
351         require(account != address(0));
352
353         _totalSupply = _totalSupply.add(value);
354         _balances[account] = _balances[account].add(value);
355         emit Transfer(address(0), account, value);
356     }
357
358     /**
359     * @dev Internal function that burns an amount of the token of a given
360     * account.
361     * @param account The account whose tokens will be burnt.
362     * @param value The amount that will be burnt.
363     */
364     /*@CTK "ERC20__burn"
365     @tag assume_completion
366     @post account != address(0)
367     @post __post._totalSupply == _totalSupply - value
368     @post __post._balances[account] == _balances[account] - value
369     */
370     function _burn(address account, uint256 value) internal {
371         require(account != address(0));
372
373         _totalSupply = _totalSupply.sub(value);
374         _balances[account] = _balances[account].sub(value);
375         emit Transfer(account, address(0), value);
376     }
377
378     /**
379     * @dev Internal function that burns an amount of the token of a given
380     * account, deducting from the sender's allowance for said account. Uses
↪ the
381     * internal burn function.
382     * Emits an Approval event (reflecting the reduced allowance).

```

```

383     * @param account The account whose tokens will be burnt.
384     * @param value The amount that will be burnt.
385     */
386     /*@CTK "ERC20__burnFrom"
387     @tag assume_completion
388     @post __post._allowed[account][msg.sender] ==
↪ _allowed[account][msg.sender] - value
389     @post (account == address(0)) == __reverted
390     @post !__reverted -> __post._totalSupply == _totalSupply - value
391     @post !__reverted -> __post._balances[account] == _balances[account] -
↪ value
392     */
393     function _burnFrom(address account, uint256 value) internal {
394         _allowed[account][msg.sender] =
↪ _allowed[account][msg.sender].sub(value);
395         _burn(account, value);
396         emit Approval(account, msg.sender, _allowed[account][msg.sender]);
397     }
398 }
399
400 // File: openzeppelin-solidity/contracts/access/Roles.sol
401
402 /**
403  * @title Roles
404  * @dev Library for managing addresses assigned to a Role.
405  */
406 library Roles {
407     struct Role {
408         mapping (address => bool) bearer;
409     }
410
411     /**
412     * @dev give an account access to this role
413     */
414     /*@CTK "Roles_add"
415     @tag assume_completion
416     @post account != address(0)
417     @post role.bearer[account] == false
418     @post role__post.bearer[account] == true
419     */
420     function add(Role storage role, address account) internal {
421         require(account != address(0));
422         require(!has(role, account));
423
424         role.bearer[account] = true;
425     }
426
427     /**

```

```

428     * @dev remove an account's access to this role
429     */
430     /*@CTK "Roles_remove"
431     @tag assume_completion
432     @post account != address(0)
433     @post role.bearer[account] == true
434     @post role__post.bearer[account] == false
435     */
436     function remove(Role storage role, address account) internal {
437         require(account != address(0));
438         require(has(role, account));
439
440         role.bearer[account] = false;
441     }
442
443     /**
444     * @dev check if an account has this role
445     * @return bool
446     */
447     /*@CTK "Roles_has"
448     @tag assume_completion
449     @post account != address(0)
450     @post __return == role.bearer[account]
451     */
452     function has(Role storage role, address account) internal view returns
↪     (bool) {
453         require(account != address(0));
454         return role.bearer[account];
455     }
456 }
457
458 // File: openzeppelin-solidity/contracts/access/roles/MinterRole.sol
459
460 contract MinterRole {
461     using Roles for Roles.Role;
462
463     event MinterAdded(address indexed account);
464     event MinterRemoved(address indexed account);
465
466     Roles.Role private _minters;
467
468     /*@CTK "MinterRole_constructor"
469     @tag assume_completion
470     @post _minters.bearer[msg.sender] == false
471     @post __post._minters.bearer[msg.sender] == true
472     */
473     constructor () internal {
474         _addMinter(msg.sender);

```

```

475 }
476
477 modifier onlyMinter() {
478     require(isMinter(msg.sender));
479     _;
480 }
481
482 /*@CTK "MinterRole_isMinter"
483     @tag assume_completion
484     @post account != address(0)
485     @post __return == _minters.bearer[account]
486 */
487 function isMinter(address account) public view returns (bool) {
488     return _minters.has(account);
489 }
490
491 /*@CTK "MinterRole_addMinter"
492     @tag assume_completion
493     @post _minters.bearer[msg.sender] == true
494     @post account != address(0)
495     @post _minters.bearer[account] == false
496     @post __post._minters.bearer[account] == true
497 */
498 function addMinter(address account) public onlyMinter {
499     _addMinter(account);
500 }
501
502 /*@CTK "MinterRole_renounceMinter"
503     @tag assume_completion
504     @post _minters.bearer[msg.sender] == true
505     @post __post._minters.bearer[msg.sender] == false
506 */
507 function renounceMinter() public {
508     _removeMinter(msg.sender);
509 }
510
511 /*@CTK "MinterRole__addMinter"
512     @tag assume_completion
513     @post account != address(0)
514     @post _minters.bearer[account] == false
515     @post __post._minters.bearer[account] == true
516 */
517 function _addMinter(address account) internal {
518     _minters.add(account);
519     emit MinterAdded(account);
520 }
521
522 /*@CTK "MinterRole__removeMinter"

```

```

523     @tag assume_completion
524     @post account != address(0)
525     @post _minters.bearer[account] == true
526     @post __post._minters.bearer[account] == false
527     */
528     function _removeMinter(address account) internal {
529         _minters.remove(account);
530         emit MinterRemoved(account);
531     }
532 }
533
534 // File: openzeppelin-solidity/contracts/token/ERC20/ERC20Mintable.sol
535
536 /**
537  * @title ERC20Mintable
538  * @dev ERC20 minting logic
539  */
540 contract ERC20Mintable is ERC20, MinterRole {
541     /**
542      * @dev Function to mint tokens
543      * @param to The address that will receive the minted tokens.
544      * @param value The amount of tokens to mint.
545      * @return A boolean that indicates if the operation was successful.
546      */
547     /*@CTK "ERC20Mintable_mint"
548      @tag assume_completion
549      @post _minters.bearer[msg.sender] == true
550      @post to != address(0)
551      @post __post._totalSupply == _totalSupply + value
552      @post __post._balances[to] == _balances[to] + value
553      */
554     function mint(address to, uint256 value) public onlyMinter returns (bool)
555     ↪ {
556         _mint(to, value);
557         return true;
558     }
559 }
560 // File: openzeppelin-solidity/contracts/token/ERC20/ERC20Burnable.sol
561
562 /**
563  * @title Burnable Token
564  * @dev Token that can be irreversibly burned (destroyed).
565  */
566 contract ERC20Burnable is ERC20 {
567     /**
568      * @dev Burns a specific amount of tokens.
569      * @param value The amount of token to be burned.

```

```

570     */
571     /*@CTK "ERC20Burnable__burn"
572         @tag assume_completion
573         @post __post._totalSupply == _totalSupply - value
574         @post __post._balances[msg.sender] == _balances[msg.sender] - value
575     */
576     function burn(uint256 value) public {
577         _burn(msg.sender, value);
578     }
579
580     /**
581     ↪     * @dev Burns a specific amount of tokens from the target address and
582         decrements allowance
583     ↪     * @param from address The address which you want to send tokens from
584     ↪     * @param value uint256 The amount of token to be burned
585     */
586     /*@CTK "ERC20Burnable__burnFrom"
587         @tag assume_completion
588         @post __post._allowed[from][msg.sender] == _allowed[from][msg.sender]
589     ↪     - value
590         @post (from == address(0)) == __reverted
591         @post !__reverted -> __post._totalSupply == _totalSupply - value
592         @post !__reverted -> __post._balances[from] == _balances[from] - value
593     */
594     function burnFrom(address from, uint256 value) public {
595         _burnFrom(from, value);
596     }
597 }
598
599 // File: openzeppelin-solidity/contracts/access/roles/PauserRole.sol
600
601 contract PauserRole {
602     using Roles for Roles.Role;
603
604     event PauserAdded(address indexed account);
605     event PauserRemoved(address indexed account);
606
607     Roles.Role private _pausers;
608
609     /*@CTK "PauserRole_constructor"
610         @tag assume_completion
611         @post _pausers.bearer[msg.sender] == false
612         @post __post._pausers.bearer[msg.sender] == true
613     */
614     constructor () internal {
615         _addPauser(msg.sender);
616     }

```



```

616     modifier onlyPauser() {
617         require(isPauser(msg.sender));
618         _;
619     }
620
621     /*@CTK "PauserRole_isPauser"
622         @tag assume_completion
623         @post account != address(0)
624         @post __return == _pausers.bearer[account]
625     */
626     function isPauser(address account) public view returns (bool) {
627         return _pausers.has(account);
628     }
629
630     /*@CTK "PauserRole_addPauser"
631         @tag assume_completion
632         @post _pausers.bearer[msg.sender] == true
633         @post account != address(0)
634         @post _pausers.bearer[account] == false
635         @post __post._pausers.bearer[account] == true
636     */
637     function addPauser(address account) public onlyPauser {
638         _addPauser(account);
639     }
640
641     /*@CTK "PauserRole_renouncePauser"
642         @tag assume_completion
643         @post _pausers.bearer[msg.sender] == true
644         @post __post._pausers.bearer[msg.sender] == false
645     */
646     function renouncePauser() public {
647         _removePauser(msg.sender);
648     }
649
650     /*@CTK "PauserRole__addPauser"
651         @tag assume_completion
652         @post account != address(0)
653         @post _pausers.bearer[account] == false
654         @post __post._pausers.bearer[account] == true
655     */
656     function _addPauser(address account) internal {
657         _pausers.add(account);
658         emit PauserAdded(account);
659     }
660
661     /*@CTK "PauserRole__removePauser"
662         @tag assume_completion
663         @post account != address(0)

```

```

664     @post _pausers.bearer[account] == true
665     @post __post._pausers.bearer[account] == false
666     */
667     function _removePauser(address account) internal {
668         _pausers.remove(account);
669         emit PauserRemoved(account);
670     }
671 }
672
673 // File: openzeppelin-solidity/contracts/lifecycle/Pausable.sol
674
675 /**
676  * @title Pausable
677  * @dev Base contract which allows children to implement an emergency stop
678  * ↪ mechanism.
679  */
680 contract Pausable is PauserRole {
681     event Paused(address account);
682     event Unpaused(address account);
683
684     bool private _paused;
685
686     /*@CTK "Pausable_constructor"
687     @post __post._paused == false
688     */
689     constructor () internal {
690         _paused = false;
691     }
692
693     /**
694     * @return true if the contract is paused, false otherwise.
695     */
696     /*@CTK "Pausable_paused"
697     @post __return == _paused
698     */
699     function paused() public view returns (bool) {
700         return _paused;
701     }
702
703     /**
704     * @dev Modifier to make a function callable only when the contract is
705     * ↪ not paused.
706     */
707     modifier whenNotPaused() {
708         require(!_paused);
709         _;
710     }

```

```

710  /**
711  * @dev Modifier to make a function callable only when the contract is
↪   paused.
712  */
713  modifier whenPaused() {
714      require(_paused);
715      _;
716  }
717
718  /**
719  * @dev called by the owner to pause, triggers stopped state
720  */
721  /*@CTK "Pausable_pause"
722     @tag assume_completion
723     @post _pausers.bearer[msg.sender] == true
724     @post _paused == false
725     @post __post._paused == true
726  */
727  function pause() public onlyPauser whenNotPaused {
728      _paused = true;
729      emit Paused(msg.sender);
730  }
731
732  /**
733  * @dev called by the owner to unpause, returns to normal state
734  */
735  /*@CTK "Pausable_unpause"
736     @tag assume_completion
737     @post _pausers.bearer[msg.sender] == true
738     @post _paused == true
739     @post __post._paused == false
740  */
741  function unpause() public onlyPauser whenPaused {
742      _paused = false;
743      emit Unpaused(msg.sender);
744  }
745  }
746
747  // File: openzeppelin-solidity/contracts/token/ERC20/ERC20Pausable.sol
748
749  /**
750  * @title Pausable token
751  * @dev ERC20 modified with pausable transfers.
752  */
753  contract ERC20Pausable is ERC20, Pausable {
754      /*@CTK "ERC20Pausable_transfer"
755         @tag assume_completion
756         @post _paused == false

```

```

757     @post to != address(0)
758     @post to != msg.sender -> __post._balances[msg.sender] ==
↪   _balances[msg.sender] - value
759     @post to != msg.sender -> __post._balances[to] == _balances[to] +
↪   value
760     @post to == msg.sender -> __post._balances[msg.sender] ==
↪   _balances[msg.sender]
761     */
762     function transfer(address to, uint256 value) public whenNotPaused returns
↪   (bool) {
763         return super.transfer(to, value);
764     }
765
766     /*@CTK "ERC20Pausable_transferFrom"
767     @tag assume_completion
768     @post _paused == false
769     @post __post._allowed[from][msg.sender] == _allowed[from][msg.sender]
↪   - value
770     @post (to == address(0)) == (__reverted)
771     @post (!__reverted && to != from) -> (__post._balances[from] ==
↪   _balances[from] - value)
772     @post (!__reverted && to != from) -> (__post._balances[to] ==
↪   _balances[to] + value)
773     @post (!__reverted && to == from) -> (__post._balances[from] ==
↪   _balances[from])
774     */
775     function transferFrom(address from, address to, uint256 value) public
↪   whenNotPaused returns (bool) {
776         return super.transferFrom(from, to, value);
777     }
778
779     /*@CTK ERC20Pausable_approve
780     @tag assume_completion
781     @post _paused == false
782     @post spender != address(0)
783     @post __post._allowed[msg.sender][spender] == value
784     */
785     function approve(address spender, uint256 value) public whenNotPaused
↪   returns (bool) {
786         return super.approve(spender, value);
787     }
788
789     /*@CTK ERC20Pausable_increaseAllowance
790     @tag assume_completion
791     @post _paused == false
792     @post spender != address(0)
793     @post __post._allowed[msg.sender][spender] ==
↪   _allowed[msg.sender][spender] + addedValue

```

```

794     */
795     function increaseAllowance(address spender, uint addedValue) public
↪ whenNotPaused returns (bool success) {
796         return super.increaseAllowance(spender, addedValue);
797     }
798
799     /*@CTK ERC20Pausable_decreaseAllowance
800         @tag assume_completion
801         @post _paused == false
802         @post spender != address(0)
803         @post __post._allowed[msg.sender][spender] ==
↪ _allowed[msg.sender][spender] - subtractedValue
804     */
805     function decreaseAllowance(address spender, uint subtractedValue) public
↪ whenNotPaused returns (bool success) {
806         return super.decreaseAllowance(spender, subtractedValue);
807     }
808 }
809
810 // File: openzeppelin-solidity/contracts/token/ERC20/ERC20Detailed.sol
811
812 /**
813  * @title ERC20Detailed token
814  * @dev The decimals are only for visualization purposes.
815  * All the operations are done using the smallest and indivisible token
↪ unit,
816  * just as on Ethereum all the operations are done in wei.
817  */
818 contract ERC20Detailed is IERC20 {
819     string private _name;
820     string private _symbol;
821     uint8 private _decimals;
822
823     /*@CTK ERC20Detailed_constructor
824         @tag assume_completion
825         @post __post._name == name
826         @post __post._symbol == symbol
827         @post __post._decimals == decimals
828     */
829     constructor (string memory name, string memory symbol, uint8 decimals)
↪ public {
830         _name = name;
831         _symbol = symbol;
832         _decimals = decimals;
833     }
834
835     /**
836     * @return the name of the token.

```

```

837     */
838     /*@CTK ERC20Detailed_name
839         @tag assume_completion
840         @post __return == _name
841     */
842     function name() public view returns (string memory) {
843         return _name;
844     }
845
846     /**
847      * @return the symbol of the token.
848     */
849     /*@CTK ERC20Detailed_symbol
850         @tag assume_completion
851         @post __return == _symbol
852     */
853     function symbol() public view returns (string memory) {
854         return _symbol;
855     }
856
857     /**
858      * @return the number of decimals of the token.
859     */
860     /*@CTK ERC20Detailed_decimals
861         @tag assume_completion
862         @post __return == _decimals
863     */
864     function decimals() public view returns (uint8) {
865         return _decimals;
866     }
867 }
868
869 /**
870  * WARNING
871  * This contract is a draft for a stable coin prototype which is being
872  ↪ designed to work in conjunction with a payment gateway.
873  * At present this contract is in Beta and must not be used in production or
874  ↪ when there is real value at stake.
875  * Use this contract at your own risk.
876  */
877
878 //Import contracts
879 //This file has been marked old because dynamically linking to these
880 ↪ contracts was causing issues at times when OpenZeppelin updated their
881 ↪ code base and GitHub repository file structure etc.
882
883 //Create MonetaToken contract

```

```
880 contract MonetaToken is ERC20Mintable, ERC20Burnable, ERC20Pausable,  
    ↪ ERC20Detailed {  
881     constructor() public  
882     ERC20Mintable()  
883     ERC20Burnable()  
884     ERC20Pausable()  
885     ERC20Detailed("Moneta Stablecoin", "MMXN", 6) {}  
886 }
```

