CertiK Verification Report For Celer

Request Date: 2019-03-11
Revision Date: 2019-03-12

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 Celer(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.

PASS

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

Mar 12, 2019

Summary

This audit report summarises the smart contract verification service requested by Celer. The goal of this security audit is to guarantee that the audited smart contracts are robust enough to avoid any potential security loopholes.
The result of this report is only a reflection of the source code that was determined in this scope, and of the source code at the time of the audit.

Type of Issues

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

Title	Description	Issues	SWC ID
Integer Overflow and Underflow	An overflow/underflow happens when an arithmetic operation reaches the maximum or minimum size of a type.	0	SWC-101
Function incorrectness	Function implementation does not meet the specification, leading to intentional or unintentional vulnerabilities.	0	
Buffer Overflow	An attacker is able to write to arbitrary storage locations of a contract if array of out bound happens	0	SWC-124
Reentrancy	A malicious contract can call back into the calling contract before the first invocation of the function is finished.	0	SWC-107
Transaction Order Dependence	A race condition vulnerability occurs when code depends on the order of the transactions submitted to it.	0	SWC-114
Timestamp Dependence	Timestamp can be influenced by minors to some degree.	0	SWC-116

Insecure Compiler Version	Using an fixed outdated compiler version or floating pragma can be problematic, if there are publicly disclosed bugs and issues that affect the current compiler version used.	0	SWC-102 SWC-103
Insecure Randomness	Block attributes are insecure to generate random numbers, as they can be influenced by minors to some degree.	0	SWC-120
"tx.origin" for authorization	tx.origin should not be used for authorization. Use msg.sender instead.	0	SWC-115
Delegatecall to Untrusted Callee	Calling into untrusted contracts is very dangerous, the target and arguments provided must be sanitized.	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. A malicious user is able to make unauthorized or unintended state changes if a developer forgot to set the visibility.	0	SWC-100
Uninitialized variables	Uninitialized local storage variables can point to other unexpected storage variables in the contract.	0	SWC-109
Assertion Failure	The assert() function is meant to assert invariants. Properly functioning code should never reach a failing assert statement.	0	SWC-110
Deprecated Solidity Features	Several functions and operators in Solidity are deprecated and should not be used as best practice.	0	SWC-111
Unused variables	Unused variables reduce code quality	0	

Vulnerability Details

Critical

No issue found.

Medium

No issue found.

Low

No issue found.

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

- Critical: The code implementation does not match the specification, or it could result in loss of funds for contract owner or users.
- Medium: The code implementation does not match the specification at certain condition, or it could affect the security standard by lost of access control.

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

Source Code with CertiK Labels

File CelerToken.sol

```
1  pragma solidity ^0.4.24;
2
3
4  /**
5   * @title SafeMath
6   * @dev Math operations with safety checks that throw on error
7   */
8  library SafeMath {
9
10   /**
11   * @dev Multiplies two numbers, throws on overflow.
12   */
13   /*@CTK SafeMath_mul
14     @tag spec
15     @post __reverted == __has_assertion_failure
16     @post __has_assertion_failure == __has_overflow
17     @post __reverted == false -> c == a * b
18     @post a == 0 -> c == 0
19     @post msg == msg__post
20     @post (a > 0 && (a * b / a != b)) == __has_assertion_failure
21     @post __addr_map == __addr_map__post
22   */
23   function mul(uint256 a, uint256 b) internal pure returns (uint256 c) {
24     // Gas optimization: this is cheaper than asserting 'a' not being zero, but the
25     // benefit is lost if 'b' is also tested.
26     // See: https://github.com/OpenZeppelin/openzeppelin-solidity/pull/522
27     if (a == 0) {
28       return 0;
29     }
30
31     c = a * b;
32     assert(c / a == b);
33     return c;
34   }
35
36   /**
37   * @dev Integer division of two numbers, truncating the quotient.
38   */
39   /*@CTK SafeMath_div
40     @tag spec
41     @post __reverted == __has_assertion_failure
42     @post b == 0 -> __reverted == true // solidity throws on 0.
43     @post __has_overflow == true -> __has_assertion_failure == true
44     @post __reverted == false -> __return == a / b
45     @post msg == msg__post
46     @post __addr_map == __addr_map__post
47   */
48   function div(uint256 a, uint256 b) internal pure returns (uint256) {
49     // assert(b > 0); // Solidity automatically throws when dividing by 0
50     // uint256 c = a / b;
51     // assert(a == b * c + a % b); // There is no case in which this doesn't hold
52     return a / b;
53   }
54
```

```
55    /**
56    * @dev Subtracts two numbers, throws on overflow (i.e. if subtrahend is greater than
          minuend).
57    */
58    /*@CTK SafeMath_sub
59      @tag spec
60      @post __reverted == __has_assertion_failure
61      @post __has_overflow == true -> __has_assertion_failure == true
62      @post __reverted == false -> __return == a - b
63      @post msg == msg__post
64      @post (a < b) == __has_assertion_failure
65      @post __addr_map == __addr_map__post
66    */
67    function sub(uint256 a, uint256 b) internal pure returns (uint256) {
68      assert(b <= a);
69      return a - b;
70    }
71
72    /**
73    * @dev Adds two numbers, throws on overflow.
74    */
75    /*@CTK SafeMath_add
76      @tag spec
77      @post __reverted == __has_assertion_failure
78      @post __has_assertion_failure == __has_overflow
79      @post __reverted == false -> c == a + b
80      @post msg == msg__post
81      @post (a + b < a) == __has_assertion_failure
82      @post __addr_map == __addr_map__post
83    */
84    function add(uint256 a, uint256 b) internal pure returns (uint256 c) {
85      c = a + b;
86      assert(c >= a);
87      return c;
88    }
89  }
90
91
92  /**
93   * @title ERC20Basic
94   * @dev Simpler version of ERC20 interface
95   * See https://github.com/ethereum/EIPs/issues/179
96   */
97  contract ERC20Basic {
98    function totalSupply() public view returns (uint256);
99    function balanceOf(address who) public view returns (uint256);
100   function transfer(address to, uint256 value) public returns (bool);
101   event Transfer(address indexed from, address indexed to, uint256 value);
102 }
103
104
105 /**
106  * @title ERC20 interface
107  * @dev see https://github.com/ethereum/EIPs/issues/20
108  */
109 contract ERC20 is ERC20Basic {
110   function allowance(address owner, address spender)
111     public view returns (uint256);
```

```
112
113   function transferFrom(address from, address to, uint256 value)
114     public returns (bool);
115
116   function approve(address spender, uint256 value) public returns (bool);
117   event Approval(
118     address indexed owner,
119     address indexed spender,
120     uint256 value
121   );
122 }
123
124
125 /**
126  * @title Basic token
127  * @dev Basic version of StandardToken, with no allowances.
128  */
129 contract BasicToken is ERC20Basic {
130   using SafeMath for uint256;
131
132   mapping(address => uint256) balances;
133
134   uint256 totalSupply_;
135
136   /**
137   * @dev Total number of tokens in existence
138   */
139   //@CTK NO_OVERFLOW
140   //@CTK NO_ASF
141   //@CTK NO_BUF_OVERFLOW
142   /*@CTK token_total_supply
143     @post __return == this.totalSupply_
144   */
145   function totalSupply() public view returns (uint256) {
146     return totalSupply_;
147   }
148
149   /**
150   * @dev Transfer token for a specified address
151   * @param _to The address to transfer to.
152   * @param _value The amount to be transferred.
153   */
154   //@CTK NO_OVERFLOW
155   /*@CTK "transfer success case"
156     @tag assume_completion
157     @pre _to != address(0)
158     @pre balances[msg.sender] >= _value
159     @post _to != msg.sender -> __post.balances[msg.sender] == balances[msg.sender] -
          _value
160     @post _to != msg.sender -> __post.balances[_to] == balances[_to] + _value
161     @post __return == true
162   */
163   /*@CTK "transfer reverted case"
164     @pre _to == address(0) \/ balances[msg.sender] < _value
165     @post __reverted == true
166   */
167   function transfer(address _to, uint256 _value) public returns (bool) {
168     require(_to != address(0));
```

```
169     require(_value <= balances[msg.sender]);
170
171     balances[msg.sender] = balances[msg.sender].sub(_value);
172     balances[_to] = balances[_to].add(_value);
173     emit Transfer(msg.sender, _to, _value);
174     return true;
175   }
176
177   /**
178   * @dev Gets the balance of the specified address.
179   * @param _owner The address to query the the balance of.
180   * @return An uint256 representing the amount owned by the passed address.
181   */
182   /*@CTK "transfer success case"
183     @post __return == this.balances[_owner]
184   */
185   function balanceOf(address _owner) public view returns (uint256) {
186     return balances[_owner];
187   }
188
189 }
190
191
192 /**
193  * @title Ownable
194  * @dev The Ownable contract has an owner address, and provides basic authorization
       control
195  * functions, this simplifies the implementation of "user permissions".
196  */
197 contract Ownable {
198   address public owner;
199
200
201   event OwnershipRenounced(address indexed previousOwner);
202   event OwnershipTransferred(
203     address indexed previousOwner,
204     address indexed newOwner
205   );
206
207
208   /**
209    * @dev The Ownable constructor sets the original `owner` of the contract to the
         sender
210    * account.
211    */
212   /*@CTK "Ownable constructor"
213     @post post(this).owner == msg.sender
214   */
215   constructor() public {
216     owner = msg.sender;
217   }
218
219   /**
220    * @dev Throws if called by any account other than the owner.
221    */
222   /*@CTK "onlyOwner"
223     @post msg.sender != owner -> __reverted
224   */
```

```
225   modifier onlyOwner() {
226     require(msg.sender == owner);
227     _;
228   }
229
230   /**
231    * @dev Allows the current owner to relinquish control of the contract.
232    * @notice Renouncing to ownership will leave the contract without an owner.
233    * It will not be possible to call the functions with the `onlyOwner`
234    * modifier anymore.
235    */
236   /*@CTK "renounceOwnership"
237     @tag assume_completion
238     @post post(this).owner == address(0)
239   */
240   function renounceOwnership() public onlyOwner {
241     emit OwnershipRenounced(owner);
242     owner = address(0);
243   }
244
245   /**
246    * @dev Allows the current owner to transfer control of the contract to a newOwner.
247    * @param _newOwner The address to transfer ownership to.
248    */
249   /*@CTK "transferOwnership"
250     @tag assume_completion
251     @post post(this).owner == _newOwner
252   */
253   function transferOwnership(address _newOwner) public onlyOwner {
254     _transferOwnership(_newOwner);
255   }
256
257   /**
258    * @dev Transfers control of the contract to a newOwner.
259    * @param _newOwner The address to transfer ownership to.
260    */
261   /*@CTK "_transferOwnership success case"
262     @pre _newOwner != address(0)
263     @post post(this).owner == _newOwner
264   */
265   /*@CTK "_transferOwnership reverted case"
266     @pre _newOwner == address(0)
267     @post __reverted
268   */
269   function _transferOwnership(address _newOwner) internal {
270     require(_newOwner != address(0));
271     emit OwnershipTransferred(owner, _newOwner);
272     owner = _newOwner;
273   }
274 }
275
276
277 /**
278  * @title Pausable
279  * @dev Base contract which allows children to implement an emergency stop mechanism.
280  */
281 contract Pausable is Ownable {
282   event Pause();
```

```
283   event Unpause();
284
285   bool public paused = false;
286
287
288   /**
289    * @dev Modifier to make a function callable only when the contract is not paused.
290    */
291   /*@CTK "whenNotPaused"
292     @post this.paused -> __reverted
293   */
294   modifier whenNotPaused() {
295     require(!paused);
296     _;
297   }
298
299   /**
300    * @dev Modifier to make a function callable only when the contract is paused.
301    */
302   /*@CTK "whenPaused"
303     @post !this.paused -> __reverted
304   */
305   modifier whenPaused() {
306     require(paused);
307     _;
308   }
309
310   /**
311    * @dev called by the owner to pause, triggers stopped state
312    */
313   /*@CTK "pause"
314     @tag assume_completion
315     @post post(this).paused == true
316   */
317   function pause() onlyOwner whenNotPaused public {
318     paused = true;
319     emit Pause();
320   }
321
322   /**
323    * @dev called by the owner to unpause, returns to normal state
324    */
325   /*@CTK "unpause"
326     @tag assume_completion
327     @post post(this).paused == false
328   */
329   function unpause() onlyOwner whenPaused public {
330     paused = false;
331     emit Unpause();
332   }
333 }
334
335
336 /**
337  * @title Standard ERC20 token
338  *
339  * @dev Implementation of the basic standard token.
340  * https://github.com/ethereum/EIPs/issues/20
```

```
341  * Based on code by FirstBlood: https://github.com/Firstbloodio/token/blob/master/
         smart_contract/FirstBloodToken.sol
342  */
343 contract StandardToken is ERC20, BasicToken {
344
345   mapping (address => mapping (address => uint256)) internal allowed;
346
347
348   /**
349    * @dev Transfer tokens from one address to another
350    * @param _from address The address which you want to send tokens from
351    * @param _to address The address which you want to transfer to
352    * @param _value uint256 the amount of tokens to be transferred
353    */
354   /*@CTK "transferFrom success"
355     @tag assume_completion
356     @pre _to != address(0)
357     @pre _value <= balances[_from]
358     @pre _value <= allowed[_from][msg.sender]
359     @post _from != _to -> __post.balances[_from] == balances[_from] - _value
360     @post _from != _to -> __post.balances[_to] == balances[_to] + _value
361     @post __post.allowed[_from][msg.sender] == allowed[_from][msg.sender] - _value
362     @post __return == true
363   */
364   /*@CTK "transferFrom failure case 1: no enough balance"
365     @pre balances[_from] < _value
366     @post __reverted
367   */
368   /*@CTK "transferFrom failure case 2: no enough allowance"
369     @pre allowed[_from][msg.sender] < _value
370     @post __reverted
371   */
372   /*@CTK "transferFrom failure case 3: _to is 0"
373     @pre _to == address(0)
374     @post __reverted
375   */
376   function transferFrom(
377     address _from,
378     address _to,
379     uint256 _value
380   )
381     public
382     returns (bool)
383   {
384     require(_to != address(0));
385     require(_value <= balances[_from]);
386     require(_value <= allowed[_from][msg.sender]);
387
388     balances[_from] = balances[_from].sub(_value);
389     balances[_to] = balances[_to].add(_value);
390     allowed[_from][msg.sender] = allowed[_from][msg.sender].sub(_value);
391     emit Transfer(_from, _to, _value);
392     return true;
393   }
394
395   /**
396    * @dev Approve the passed address to spend the specified amount of tokens on behalf
         of msg.sender.
```

```
397   * Beware that changing an allowance with this method brings the risk that someone
          may use both the old
398   * and the new allowance by unfortunate transaction ordering. One possible solution
          to mitigate this
399   * race condition is to first reduce the spender's allowance to 0 and set the
          desired value afterwards:
400   * https://github.com/ethereum/EIPs/issues/20#issuecomment-263524729
401   * @param _spender The address which will spend the funds.
402   * @param _value The amount of tokens to be spent.
403   */
404
405  //@CTK NO_OVERFLOW
406  //@CTK NO_ASF
407  /*@CTK "approve transfer allowance"
408    @post post(this).allowed[msg.sender][_spender] == _value
409    @post __return == true
410  */
411  function approve(address _spender, uint256 _value) public returns (bool) {
412    allowed[msg.sender][_spender] = _value;
413    emit Approval(msg.sender, _spender, _value);
414    return true;
415  }
416
417  /**
418   * @dev Function to check the amount of tokens that an owner allowed to a spender.
419   * @param _owner address The address which owns the funds.
420   * @param _spender address The address which will spend the funds.
421   * @return A uint256 specifying the amount of tokens still available for the spender
          .
422   */
423  /*@CTK "get the allowance"
424    @post __return == allowed[_owner][_spender]
425    @post this == post(this)
426  */
427  function allowance(
428    address _owner,
429    address _spender
430   )
431    public
432    view
433    returns (uint256)
434  {
435    return allowed[_owner][_spender];
436  }
437
438  /**
439   * @dev Increase the amount of tokens that an owner allowed to a spender.
440   * approve should be called when allowed[_spender] == 0. To increment
441   * allowed value is better to use this function to avoid 2 calls (and wait until
442   * the first transaction is mined)
443   * From MonolithDAO Token.sol
444   * @param _spender The address which will spend the funds.
445   * @param _addedValue The amount of tokens to increase the allowance by.
446   */
447  /*@CTK "increaseApproval ok"
448    @tag assume_completion
449    @post post(this).allowed[msg.sender][_spender] == this.allowed[msg.sender][
          _spender] + _addedValue
```

```
450     @post __return == true
451   */
452   function increaseApproval(
453     address _spender,
454     uint256 _addedValue
455   )
456     public
457     returns (bool)
458   {
459     allowed[msg.sender][_spender] = (
460       allowed[msg.sender][_spender].add(_addedValue));
461     emit Approval(msg.sender, _spender, allowed[msg.sender][_spender]);
462     return true;
463   }
464
465   /**
466    * @dev Decrease the amount of tokens that an owner allowed to a spender.
467    * approve should be called when allowed[_spender] == 0. To decrement
468    * allowed value is better to use this function to avoid 2 calls (and wait until
469    * the first transaction is mined)
470    * From MonolithDAO Token.sol
471    * @param _spender The address which will spend the funds.
472    * @param _subtractedValue The amount of tokens to decrease the allowance by.
473    */
474   /*@CTK "decreaseApproval case if"
475     @tag assume_completion
476     @pre _subtractedValue > allowed[msg.sender][_spender]
477     @post post(this).allowed[msg.sender][_spender] == 0
478     @post __return == true
479   */
480   /*@CTK "decreaseApproval case else"
481     @tag assume_completion
482     @pre _subtractedValue <= allowed[msg.sender][_spender]
483     @post post(this).allowed[msg.sender][_spender] == this.allowed[msg.sender][
          _spender] - _subtractedValue
484     @post __return == true
485   */
486   function decreaseApproval(
487     address _spender,
488     uint256 _subtractedValue
489   )
490     public
491     returns (bool)
492   {
493     uint256 oldValue = allowed[msg.sender][_spender];
494     if (_subtractedValue > oldValue) {
495       allowed[msg.sender][_spender] = 0;
496     } else {
497       allowed[msg.sender][_spender] = oldValue.sub(_subtractedValue);
498     }
499     emit Approval(msg.sender, _spender, allowed[msg.sender][_spender]);
500     return true;
501   }
502
503 }
504
505
506 /**
```

```
507  * @title SuccinctWhitelist
508  * @dev The SuccinctWhitelist contract has a whitelist of addresses, and provides
         basic authorization control functions.
509  * Note: this is a succinct, straightforward and easy to understand implementation of
         openzeppelin-solidity's Whitelisted,
510  * but with full functionalities and APIs of openzeppelin-solidity's Whitelisted
         without inheriting RBAC.
511  */
512 contract SuccinctWhitelist is Ownable {
513   mapping(address => bool) public whitelisted;
514
515   event WhitelistAdded(address indexed operator);
516   event WhitelistRemoved(address indexed operator);
517
518   /**
519    * @dev Throws if operator is not whitelisted.
520    * @param _operator address
521    */
522   /*@CTK "onlyIfWhitelisted"
523     @post !this.whitelisted[_operator] -> __reverted
524   */
525   modifier onlyIfWhitelisted(address _operator) {
526     require(whitelisted[_operator]);
527     _;
528   }
529
530   /**
531    * @dev add an address to the whitelist
532    * @param _operator address
533    * @return true if the address was added to the whitelist,
534    * or was already in the whitelist
535    */
536   /*@CTK "addAddressToWhitelist"
537     @tag assume_completion
538     @post post(this).whitelisted[_operator] == true
539     @post __return == true
540   */
541   function addAddressToWhitelist(address _operator)
542     onlyOwner
543     public
544     returns (bool)
545   {
546     whitelisted[_operator] = true;
547     emit WhitelistAdded(_operator);
548     return true;
549   }
550
551   /**
552    * @dev getter to determine if address is in whitelist
553    */
554   /*@CTK "whitelist happy case"
555     @post __return == this.whitelisted[_operator]
556     @post post(this) == this
557   */
558   function whitelist(address _operator)
559     public
560     view
561     returns (bool)
```

```
562   {
563     bool result = whitelisted[_operator];
564     return result;
565   }
566
567   /**
568    * @dev add addresses to the whitelist
569    * @param _operators addresses
570    * @return true if all addresses was added to the whitelist,
571    * or were already in the whitelist
572    */
573   /*@CTK "addAddressesToWhitelist happy case"
574     @tag assume_completion
575     @post forall i: uint. (i >= 0 /\ i < _operators.length) -> post(this).whitelisted[
          _operators[i]] == true
576     @post __return == true
577   */
578   function addAddressesToWhitelist(address[] _operators)
579     onlyOwner
580     public
581     returns (bool)
582   {
583     /*@CTK addAddressesToWhitelist_forloop
584       @inv i <= _operators.length
585       @inv _operators == _operators__pre
586       @inv forall j: uint. (j >= 0 /\ j < i) -> this.whitelisted[_operators[j]] ==
            true
587       @post i == _operators.length
588       @post !__should_return
589     */
590     for (uint256 i = 0; i < _operators.length; i++) {
591       require(addAddressToWhitelist(_operators[i]));
592     }
593     return true;
594   }
595
596   /**
597    * @dev remove an address from the whitelist
598    * @param _operator address
599    * @return true if the address was removed from the whitelist,
600    * or the address wasn't in the whitelist in the first place
601    */
602   /*@CTK "removeAddressFromWhitelist happy case"
603     @tag assume_completion
604     @post post(this).whitelisted[_operator] == false
605     @post __return == true
606   */
607   function removeAddressFromWhitelist(address _operator)
608     onlyOwner
609     public
610     returns (bool)
611   {
612     whitelisted[_operator] = false;
613     emit WhitelistRemoved(_operator);
614     return true;
615   }
616
617   /**
```

```
618     * @dev remove addresses from the whitelist
619     * @param _operators addresses
620     * @return true if all addresses were removed from the whitelist,
621     * or weren't in the whitelist in the first place
622     */
623    /*@CTK "removeAddressesFromWhitelist happy case"
624      @tag assume_completion
625      @post forall i: uint. (i >= 0 /\ i < _operators.length) -> post(this).whitelisted[
           _operators[i]] == false
626      @post __return == true
627    */
628    function removeAddressesFromWhitelist(address[] _operators)
629      onlyOwner
630      public
631      returns (bool)
632    {
633      /*@CTK removeAddressesFromWhitelist_forLoop
634        @inv i <= _operators.length
635        @inv _operators == _operators__pre
636        @inv forall j: uint. (j >= 0 /\ j < i) -> this.whitelisted[_operators[j]] ==
             false
637        @post i == _operators.length
638        @post !__should_return
639      */
640      for (uint256 i = 0; i < _operators.length; i++) {
641        require(removeAddressFromWhitelist(_operators[i]));
642      }
643      return true;
644    }
645
646  }
647
648
649  /**
650   * @title Pausable token
651   * @dev StandardToken modified with pausable transfers.
652   **/
653  contract PausableToken is StandardToken, Pausable {
654
655    /*@CTK "PausableToken transfer success case"
656      @tag assume_completion
657      @post _to != msg.sender -> __post.balances[msg.sender] == balances[msg.sender] -
           _value
658      @post _to != msg.sender -> __post.balances[_to] == balances[_to] + _value
659      @post __return == true
660    */
661    function transfer(
662      address _to,
663      uint256 _value
664    )
665      public
666      whenNotPaused
667      returns (bool)
668    {
669      return super.transfer(_to, _value);
670    }
671
672    /*@CTK "PausableToken transferFrom success"
```

```
673     @tag assume_completion
674     @post _from != _to -> __post.balances[_from] == balances[_from] - _value
675     @post _from != _to -> __post.balances[_to] == balances[_to] + _value
676     @post __post.allowed[_from][msg.sender] == allowed[_from][msg.sender] - _value
677     @post __return == true
678   */
679   function transferFrom(
680     address _from,
681     address _to,
682     uint256 _value
683   )
684     public
685     whenNotPaused
686     returns (bool)
687   {
688     return super.transferFrom(_from, _to, _value);
689   }
690
691   /*@CTK "PausableToken approve transfer allowance"
692     @pre paused == false
693     @post post(this).allowed[msg.sender][_spender] == _value
694     @post __return == true
695   */
696   function approve(
697     address _spender,
698     uint256 _value
699   )
700     public
701     whenNotPaused
702     returns (bool)
703   {
704     return super.approve(_spender, _value);
705   }
706   /*@CTK "increaseApproval ok"
707     @tag assume_completion
708     @post post(this).allowed[msg.sender][_spender] == this.allowed[msg.sender][
          _spender] + _addedValue
709   */
710   function increaseApproval(
711     address _spender,
712     uint _addedValue
713   )
714     public
715     whenNotPaused
716     returns (bool success)
717   {
718     return super.increaseApproval(_spender, _addedValue);
719   }
720
721   /*@CTK "decreaseApproval case if"
722     @tag assume_completion
723     @pre _subtractedValue > allowed[msg.sender][_spender]
724     @post post(this).allowed[msg.sender][_spender] == 0
725   */
726   /*@CTK "decreaseApproval case else"
727     @tag assume_completion
728     @pre _subtractedValue <= allowed[msg.sender][_spender]
```

```
729     @post post(this).allowed[msg.sender][_spender] == this.allowed[msg.sender][
            _spender] - _subtractedValue
730   */
731   function decreaseApproval(
732     address _spender,
733     uint _subtractedValue
734   )
735     public
736     whenNotPaused
737     returns (bool success)
738   {
739     return super.decreaseApproval(_spender, _subtractedValue);
740   }
741 }
742
743
744 /**
745  * @title CelerToken
746  * @dev Celer Network's token contract.
747  */
748 contract CelerToken is PausableToken, SuccinctWhitelist {
749   string public constant name = "CelerToken";
750   string public constant symbol = "CELR";
751   uint256 public constant decimals = 18;
752
753   // 10 billion tokens with 18 decimals
754   uint256 public constant INITIAL_SUPPLY = 1e28;
755
756   // Indicate whether token transferability is opened to everyone
757   bool public transferOpened = false;
758
759   /*@CTK onlyValidReceiver
760     @pre this.transferOpened == false /\ whitelisted[msg.sender] == false /\ msg.
          sender != owner
761     @post __reverted
762   */
763   modifier onlyIfTransferable() {
764     require(transferOpened || whitelisted[msg.sender] || msg.sender == owner);
765     _;
766   }
767
768   /*@CTK onlyValidReceiver
769     @post _to == address(this) -> __reverted
770   */
771   modifier onlyValidReceiver(address _to) {
772     require(_to != address(this));
773     _;
774   }
775
776
777   /*@CTK "CelerToken constructor"
778     @post post(this).totalSupply_ == INITIAL_SUPPLY
779     @post post(this).balances[msg.sender] == INITIAL_SUPPLY
780   */
781   constructor() public {
782     totalSupply_ = INITIAL_SUPPLY;
783     balances[msg.sender] = INITIAL_SUPPLY;
784   }
```

```
785
786  /**
787   * @dev Extend parent behavior requiring transfer
788   * to respect transferability and receiver's validity.
789   */
790  /*@CTK "CelerToken transfer success case"
791    @tag assume_completion
792    @post _to != msg.sender -> __post.balances[msg.sender] == balances[msg.sender] -
          _value
793    @post _to != msg.sender -> __post.balances[_to] == balances[_to] + _value
794    @post __return == true
795  */
796  function transfer(
797    address _to,
798    uint256 _value
799  )
800    public
801    onlyIfTransferable
802    onlyValidReceiver(_to)
803    returns (bool)
804  {
805    return super.transfer(_to, _value);
806  }
807
808  /**
809   * @dev Extend parent behavior requiring transferFrom
810   * to respect transferability and receiver's validity.
811   */
812  /*@CTK "transferFrom transferFrom success"
813    @tag assume_completion
814    @post _from != _to -> __post.balances[_from] == balances[_from] - _value
815    @post _from != _to -> __post.balances[_to] == balances[_to] + _value
816    @post __post.allowed[_from][msg.sender] == allowed[_from][msg.sender] - _value
817    @post __return == true
818  */
819  function transferFrom(
820    address _from,
821    address _to,
822    uint256 _value
823  )
824    public
825    onlyIfTransferable
826    onlyValidReceiver(_to)
827    returns (bool)
828  {
829    return super.transferFrom(_from, _to, _value);
830  }
831
832  /**
833   * @dev Open token transferability.
834   */
835
836  /*@CTK "openTransfer ok"
837    @tag assume_completion
838    @post post(this).transferOpened == true
839  */
840  function openTransfer() external onlyOwner {
841    transferOpened = true;
```

```
842   }
843 }
```

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

Post environment

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

Static Analysis Request

INSECURE_COMPILER_VERSION

Line 1 in File CelerToken.sol

```
1 pragma solidity ^0.4.24;
```

Only these compiler versions are safe to compile your code: 0.4.25

Formal Verification Request 1

SafeMath_mul

12, Mar 2019
366.28 ms

Line 13-22 in File CelerToken.sol

```
13  /*@CTK SafeMath_mul
14    @tag spec
15    @post __reverted == __has_assertion_failure
16    @post __has_assertion_failure == __has_overflow
17    @post __reverted == false -> c == a * b
18    @post a == 0 -> c == 0
19    @post msg == msg__post
20    @post (a > 0 && (a * b / a != b)) == __has_assertion_failure
21    @post __addr_map == __addr_map__post
22  */
```

Line 23-34 in File CelerToken.sol

```
23  function mul(uint256 a, uint256 b) internal pure returns (uint256 c) {
24    // Gas optimization: this is cheaper than asserting 'a' not being zero, but the
25    // benefit is lost if 'b' is also tested.
26    // See: https://github.com/OpenZeppelin/openzeppelin-solidity/pull/522
27    if (a == 0) {
28      return 0;
29    }
30
31    c = a * b;
32    assert(c / a == b);
33    return c;
34  }
```

The code meets the specification

Formal Verification Request 2

SafeMath_div

12, Mar 2019
7.86 ms

Line 39-47 in File CelerToken.sol

```
39  /*@CTK SafeMath_div
40    @tag spec
41    @post __reverted == __has_assertion_failure
42    @post b == 0 -> __reverted == true // solidity throws on 0.
43    @post __has_overflow == true -> __has_assertion_failure == true
44    @post __reverted == false -> __return == a / b
45    @post msg == msg__post
46    @post __addr_map == __addr_map__post
47  */
```

Line 48-53 in File CelerToken.sol

```
48  function div(uint256 a, uint256 b) internal pure returns (uint256) {
49    // assert(b > 0); // Solidity automatically throws when dividing by 0
50    // uint256 c = a / b;
51    // assert(a == b * c + a % b); // There is no case in which this doesn't hold
52    return a / b;
53  }
```

The code meets the specification

Formal Verification Request 3

SafeMath_sub

12, Mar 2019
14.08 ms

Line 58-66 in File CelerToken.sol

```
58  /*@CTK SafeMath_sub
59    @tag spec
60    @post __reverted == __has_assertion_failure
61    @post __has_overflow == true -> __has_assertion_failure == true
62    @post __reverted == false -> __return == a - b
63    @post msg == msg__post
64    @post (a < b) == __has_assertion_failure
65    @post __addr_map == __addr_map__post
66  */
```

Line 67-70 in File CelerToken.sol

```
67  function sub(uint256 a, uint256 b) internal pure returns (uint256) {
68    assert(b <= a);
69    return a - b;
70  }
```

The code meets the specification

Formal Verification Request 4

SafeMath_add

12, Mar 2019
17.71 ms

Line 75-83 in File CelerToken.sol

```
75  /*@CTK SafeMath_add
76    @tag spec
77    @post __reverted == __has_assertion_failure
78    @post __has_assertion_failure == __has_overflow
79    @post __reverted == false -> c == a + b
80    @post msg == msg__post
81    @post (a + b < a) == __has_assertion_failure
82    @post __addr_map == __addr_map__post
83  */
```

Line 84-88 in File CelerToken.sol

```
84  function add(uint256 a, uint256 b) internal pure returns (uint256 c) {
85    c = a + b;
86    assert(c >= a);
87    return c;
88  }
```

The code meets the specification

Formal Verification Request 5

If method completes, integer overflow would not happen.

12, Mar 2019
6.12 ms

Line 139 in File CelerToken.sol

```
139  //@CTK NO_OVERFLOW
```

Line 145-147 in File CelerToken.sol

```
145  function totalSupply() public view returns (uint256) {
146    return totalSupply_;
147  }
```

The code meets the specification

Formal Verification Request 6

Method will not encounter an assertion failure.

12, Mar 2019
0.55 ms

Line 140 in File CelerToken.sol

```
140  //@CTK NO_ASF
```

Line 145-147 in File CelerToken.sol

```
145  function totalSupply() public view returns (uint256) {
146    return totalSupply_;
147  }
```

The code meets the specification

Formal Verification Request 7

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

12, Mar 2019
0.54 ms

Line 141 in File CelerToken.sol

```
141  //@CTK NO_BUF_OVERFLOW
```

Line 145-147 in File CelerToken.sol

```
145  function totalSupply() public view returns (uint256) {
146    return totalSupply_;
147  }
```

The code meets the specification

Formal Verification Request 8

token_total_supply

12, Mar 2019
0.51 ms

Line 142-144 in File CelerToken.sol

```
142  /*@CTK token_total_supply
143    @post __return == this.totalSupply_
144  */
```

Line 145-147 in File CelerToken.sol

```
145  function totalSupply() public view returns (uint256) {
146    return totalSupply_;
147  }
```

The code meets the specification

Formal Verification Request 9

If method completes, integer overflow would not happen.

12, Mar 2019
54.25 ms

Line 154 in File CelerToken.sol

```
154  //@CTK NO_OVERFLOW
```

Line 167-175 in File CelerToken.sol

```
167  function transfer(address _to, uint256 _value) public returns (bool) {
168    require(_to != address(0));
169    require(_value <= balances[msg.sender]);
170
171    balances[msg.sender] = balances[msg.sender].sub(_value);
172    balances[_to] = balances[_to].add(_value);
173    emit Transfer(msg.sender, _to, _value);
174    return true;
175  }
```

The code meets the specification

Formal Verification Request 10

transfer success case

12, Mar 2019
90.94 ms

Line 155-162 in File CelerToken.sol

```
155  /*@CTK "transfer success case"
156    @tag assume_completion
157    @pre _to != address(0)
158    @pre balances[msg.sender] >= _value
159    @post _to != msg.sender -> __post.balances[msg.sender] == balances[msg.sender] -
           _value
160    @post _to != msg.sender -> __post.balances[_to] == balances[_to] + _value
161    @post __return == true
162  */
```

Line 167-175 in File CelerToken.sol

```
167  function transfer(address _to, uint256 _value) public returns (bool) {
168    require(_to != address(0));
169    require(_value <= balances[msg.sender]);
170
171    balances[msg.sender] = balances[msg.sender].sub(_value);
172    balances[_to] = balances[_to].add(_value);
173    emit Transfer(msg.sender, _to, _value);
174    return true;
175  }
```

The code meets the specification

Formal Verification Request 11

transfer reverted case

12, Mar 2019
10.75 ms

Line 163-166 in File CelerToken.sol

```
163  /*@CTK "transfer reverted case"
164    @pre _to == address(0) \/ balances[msg.sender] < _value
165    @post __reverted == true
166  */
```

Line 167-175 in File CelerToken.sol

```
167  function transfer(address _to, uint256 _value) public returns (bool) {
168    require(_to != address(0));
169    require(_value <= balances[msg.sender]);
170
171    balances[msg.sender] = balances[msg.sender].sub(_value);
172    balances[_to] = balances[_to].add(_value);
173    emit Transfer(msg.sender, _to, _value);
174    return true;
175  }
```

✅ The code meets the specification

Formal Verification Request 12

transfer success case

📅 12, Mar 2019
⏱ 5.84 ms

Line 182-184 in File CelerToken.sol

```
182  /*@CTK "transfer success case"
183    @post __return == this.balances[_owner]
184  */
```

Line 185-187 in File CelerToken.sol

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

✅ The code meets the specification

Formal Verification Request 13

Ownable constructor

📅 12, Mar 2019
⏱ 5.01 ms

Line 212-214 in File CelerToken.sol

```
212  /*@CTK "Ownable constructor"
213    @post post(this).owner == msg.sender
214  */
```

Line 215-217 in File CelerToken.sol

```
215  constructor() public {
216    owner = msg.sender;
217  }
```

✅ The code meets the specification

Formal Verification Request 14

renounceOwnership

📅 12, Mar 2019
⏱ 15.66 ms

Line 236-239 in File CelerToken.sol

```
236  /*@CTK "renounceOwnership"
237    @tag assume_completion
238    @post post(this).owner == address(0)
239  */
```

Line 240-243 in File CelerToken.sol

```
240  function renounceOwnership() public onlyOwner {
241    emit OwnershipRenounced(owner);
242    owner = address(0);
243  }
```

The code meets the specification

Formal Verification Request 15

onlyOwner_renounceOwnership

12, Mar 2019
0.96 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

Line 240-243 in File CelerToken.sol

```
240  function renounceOwnership() public onlyOwner {
241    emit OwnershipRenounced(owner);
242    owner = address(0);
243  }
```

The code meets the specification

Formal Verification Request 16

transferOwnership

12, Mar 2019
51.36 ms

Line 249-252 in File CelerToken.sol

```
249  /*@CTK "transferOwnership"
250    @tag assume_completion
251    @post post(this).owner == _newOwner
252  */
```

Line 253-255 in File CelerToken.sol

```
253  function transferOwnership(address _newOwner) public onlyOwner {
254    _transferOwnership(_newOwner);
255  }
```

The code meets the specification

Formal Verification Request 17

onlyOwner_transferOwnership

12, Mar 2019
1.64 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

Line 253-255 in File CelerToken.sol

```
253  function transferOwnership(address _newOwner) public onlyOwner {
254    _transferOwnership(_newOwner);
255  }
```

The code meets the specification

Formal Verification Request 18

_transferOwnership success case

12, Mar 2019
1.55 ms

Line 261-264 in File CelerToken.sol

```
261  /*@CTK "_transferOwnership success case"
262    @pre _newOwner != address(0)
263    @post post(this).owner == _newOwner
264  */
```

Line 269-273 in File CelerToken.sol

```
269  function _transferOwnership(address _newOwner) internal {
270    require(_newOwner != address(0));
271    emit OwnershipTransferred(owner, _newOwner);
272    owner = _newOwner;
273  }
```

The code meets the specification

Formal Verification Request 19

_transferOwnership reverted case

12, Mar 2019
0.51 ms

Line 265-268 in File CelerToken.sol

```
265  /*@CTK "_transferOwnership reverted case"
266    @pre _newOwner == address(0)
267    @post __reverted
268  */
```

Line 269-273 in File CelerToken.sol

```
269  function _transferOwnership(address _newOwner) internal {
270    require(_newOwner != address(0));
271    emit OwnershipTransferred(owner, _newOwner);
272    owner = _newOwner;
273  }
```

✅ The code meets the specification

Formal Verification Request 20

pause

📅 12, Mar 2019
⏱ 26.71 ms

Line 313-316 in File CelerToken.sol

```
313  /*@CTK "pause"
314    @tag assume_completion
315    @post post(this).paused == true
316  */
```

Line 317-320 in File CelerToken.sol

```
317  function pause() onlyOwner whenNotPaused public {
318    paused = true;
319    emit Pause();
320  }
```

✅ The code meets the specification

Formal Verification Request 21

onlyOwner_pause

📅 12, Mar 2019
⏱ 3.15 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

Line 317-320 in File CelerToken.sol

```
317  function pause() onlyOwner whenNotPaused public {
318    paused = true;
319    emit Pause();
320  }
```

The code meets the specification

Formal Verification Request 22

whenNotPaused_pause

12, Mar 2019
1.23 ms

Line 291-293 in File CelerToken.sol

```
291  /*@CTK "whenNotPaused"
292    @post this.paused -> __reverted
293  */
```

Line 317-320 in File CelerToken.sol

```
317  function pause() onlyOwner whenNotPaused public {
318    paused = true;
319    emit Pause();
320  }
```

The code meets the specification

Formal Verification Request 23

unpause

12, Mar 2019
26.76 ms

Line 325-328 in File CelerToken.sol

```
325  /*@CTK "unpause"
326    @tag assume_completion
327    @post post(this).paused == false
328  */
```

Line 329-332 in File CelerToken.sol

```
329  function unpause() onlyOwner whenPaused public {
330    paused = false;
331    emit Unpause();
332  }
```

The code meets the specification

Formal Verification Request 24

onlyOwner_unpause

12, Mar 2019
1.75 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

Line 329-332 in File CelerToken.sol

```
329  function unpause() onlyOwner whenPaused public {
330    paused = false;
331    emit Unpause();
332  }
```

✅ The code meets the specification

Formal Verification Request 25

whenPaused_unpause

📅 12, Mar 2019
⏱ 1.06 ms

Line 302-304 in File CelerToken.sol

```
302  /*@CTK "whenPaused"
303    @post !this.paused -> __reverted
304  */
```

Line 329-332 in File CelerToken.sol

```
329  function unpause() onlyOwner whenPaused public {
330    paused = false;
331    emit Unpause();
332  }
```

✅ The code meets the specification

Formal Verification Request 26

onlyOwner_Ownable__renounceOwnership

📅 12, Mar 2019
⏱ 17.22 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) Line 240-243 in File CelerToken.sol

```
240  No Snippet Available
```

✅ The code meets the specification

Formal Verification Request 27

onlyOwner_renounceOwnership

12, Mar 2019
16.18 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) Line 240-243 in File CelerToken.sol

```
240 No Snippet Available
```

The code meets the specification

Formal Verification Request 28

onlyOwner_Ownable__transferOwnership

12, Mar 2019
49.77 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) Line 253-255 in File CelerToken.sol

```
253 No Snippet Available
```

The code meets the specification

Formal Verification Request 29

onlyOwner_transferOwnership

12, Mar 2019
36.59 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) Line 253-255 in File CelerToken.sol

```
253 No Snippet Available
```

The code meets the specification

Formal Verification Request 30

transferFrom success

12, Mar 2019
256.37 ms

Line 354-363 in File CelerToken.sol

```
354  /*@CTK "transferFrom success"
355    @tag assume_completion
356    @pre _to != address(0)
357    @pre _value <= balances[_from]
358    @pre _value <= allowed[_from][msg.sender]
359    @post _from != _to -> __post.balances[_from] == balances[_from] - _value
360    @post _from != _to -> __post.balances[_to] == balances[_to] + _value
361    @post __post.allowed[_from][msg.sender] == allowed[_from][msg.sender] - _value
362    @post __return == true
363  */
```

Line 376-393 in File CelerToken.sol

```
376  function transferFrom(
377    address _from,
378    address _to,
379    uint256 _value
380  )
381    public
382    returns (bool)
383  {
384    require(_to != address(0));
385    require(_value <= balances[_from]);
386    require(_value <= allowed[_from][msg.sender]);
387
388    balances[_from] = balances[_from].sub(_value);
389    balances[_to] = balances[_to].add(_value);
390    allowed[_from][msg.sender] = allowed[_from][msg.sender].sub(_value);
391    emit Transfer(_from, _to, _value);
392    return true;
393  }
```

The code meets the specification

Formal Verification Request 31

transferFrom failure case 1: no enough balance

12, Mar 2019
15.58 ms

Line 364-367 in File CelerToken.sol

```
364  /*@CTK "transferFrom failure case 1: no enough balance"
365    @pre balances[_from] < _value
366    @post __reverted
367  */
```

Line 376-393 in File CelerToken.sol

```
376  function transferFrom(
377    address _from,
378    address _to,
379    uint256 _value
380  )
381    public
382    returns (bool)
383  {
384    require(_to != address(0));
385    require(_value <= balances[_from]);
386    require(_value <= allowed[_from][msg.sender]);
387
388    balances[_from] = balances[_from].sub(_value);
389    balances[_to] = balances[_to].add(_value);
390    allowed[_from][msg.sender] = allowed[_from][msg.sender].sub(_value);
391    emit Transfer(_from, _to, _value);
392    return true;
393  }
```

✅ The code meets the specification

Formal Verification Request 32

transferFrom failure case 2: no enough allowance

📅 12, Mar 2019
⏱ 40.75 ms

Line 368-371 in File CelerToken.sol

```
368  /*@CTK "transferFrom failure case 2: no enough allowance"
369    @pre allowed[_from][msg.sender] < _value
370    @post __reverted
371  */
```

Line 376-393 in File CelerToken.sol

```
376  function transferFrom(
377    address _from,
378    address _to,
379    uint256 _value
380  )
381    public
382    returns (bool)
383  {
384    require(_to != address(0));
385    require(_value <= balances[_from]);
386    require(_value <= allowed[_from][msg.sender]);
387
388    balances[_from] = balances[_from].sub(_value);
389    balances[_to] = balances[_to].add(_value);
390    allowed[_from][msg.sender] = allowed[_from][msg.sender].sub(_value);
391    emit Transfer(_from, _to, _value);
392    return true;
393  }
```

The code meets the specification

Formal Verification Request 33

transferFrom failure case 3: _to is 0

12, Mar 2019
0.98 ms

Line 372-375 in File CelerToken.sol

```
372  /*@CTK "transferFrom failure case 3: _to is 0"
373    @pre _to == address(0)
374    @post __reverted
375  */
```

Line 376-393 in File CelerToken.sol

```
376  function transferFrom(
377    address _from,
378    address _to,
379    uint256 _value
380  )
381    public
382    returns (bool)
383  {
384    require(_to != address(0));
385    require(_value <= balances[_from]);
386    require(_value <= allowed[_from][msg.sender]);
387
388    balances[_from] = balances[_from].sub(_value);
389    balances[_to] = balances[_to].add(_value);
390    allowed[_from][msg.sender] = allowed[_from][msg.sender].sub(_value);
391    emit Transfer(_from, _to, _value);
392    return true;
393  }
```

The code meets the specification

Formal Verification Request 34

If method completes, integer overflow would not happen.

12, Mar 2019
9.81 ms

Line 405 in File CelerToken.sol

```
405  //@CTK NO_OVERFLOW
```

Line 411-415 in File CelerToken.sol

```
411  function approve(address _spender, uint256 _value) public returns (bool) {
412    allowed[msg.sender][_spender] = _value;
413    emit Approval(msg.sender, _spender, _value);
```

```
414     return true;
415   }
```

The code meets the specification

Formal Verification Request 35

Method will not encounter an assertion failure.

12, Mar 2019
0.46 ms

Line 406 in File CelerToken.sol

```
406   //@CTK NO_ASF
```

Line 411-415 in File CelerToken.sol

```
411   function approve(address _spender, uint256 _value) public returns (bool) {
412     allowed[msg.sender][_spender] = _value;
413     emit Approval(msg.sender, _spender, _value);
414     return true;
415   }
```

The code meets the specification

Formal Verification Request 36

approve transfer allowance

12, Mar 2019
1.5 ms

Line 407-410 in File CelerToken.sol

```
407   /*@CTK "approve transfer allowance"
408     @post post(this).allowed[msg.sender][_spender] == _value
409     @post __return == true
410   */
```

Line 411-415 in File CelerToken.sol

```
411   function approve(address _spender, uint256 _value) public returns (bool) {
412     allowed[msg.sender][_spender] = _value;
413     emit Approval(msg.sender, _spender, _value);
414     return true;
415   }
```

The code meets the specification

Formal Verification Request 37

get the allowance

12, Mar 2019
6.34 ms

Line 423-426 in File CelerToken.sol

```
423  /*@CTK "get the allowance"
424    @post __return == allowed[_owner][_spender]
425    @post this == post(this)
426  */
```

Line 427-436 in File CelerToken.sol

```
427  function allowance(
428    address _owner,
429    address _spender
430   )
431    public
432    view
433    returns (uint256)
434  {
435    return allowed[_owner][_spender];
436  }
```

The code meets the specification

Formal Verification Request 38

increaseApproval ok

12, Mar 2019
19.96 ms

Line 447-451 in File CelerToken.sol

```
447  /*@CTK "increaseApproval ok"
448    @tag assume_completion
449    @post post(this).allowed[msg.sender][_spender] == this.allowed[msg.sender][
           _spender] + _addedValue
450    @post __return == true
451  */
```

Line 452-463 in File CelerToken.sol

```
452  function increaseApproval(
453    address _spender,
454    uint256 _addedValue
455  )
456    public
457    returns (bool)
458  {
459    allowed[msg.sender][_spender] = (
460      allowed[msg.sender][_spender].add(_addedValue));
461    emit Approval(msg.sender, _spender, allowed[msg.sender][_spender]);
```

```
462     return true;
463   }
```

The code meets the specification

Formal Verification Request 39

decreaseApproval case if

12, Mar 2019
32.55 ms

Line 474-479 in File CelerToken.sol

```
474   /*@CTK "decreaseApproval case if"
475     @tag assume_completion
476     @pre _subtractedValue > allowed[msg.sender][_spender]
477     @post post(this).allowed[msg.sender][_spender] == 0
478     @post __return == true
479   */
```

Line 486-501 in File CelerToken.sol

```
486   function decreaseApproval(
487     address _spender,
488     uint256 _subtractedValue
489   )
490     public
491     returns (bool)
492   {
493     uint256 oldValue = allowed[msg.sender][_spender];
494     if (_subtractedValue > oldValue) {
495       allowed[msg.sender][_spender] = 0;
496     } else {
497       allowed[msg.sender][_spender] = oldValue.sub(_subtractedValue);
498     }
499     emit Approval(msg.sender, _spender, allowed[msg.sender][_spender]);
500     return true;
501   }
```

The code meets the specification

Formal Verification Request 40

decreaseApproval case else

12, Mar 2019
2.72 ms

Line 480-485 in File CelerToken.sol

```
480   /*@CTK "decreaseApproval case else"
481     @tag assume_completion
482     @pre _subtractedValue <= allowed[msg.sender][_spender]
```

```
483     @post post(this).allowed[msg.sender][_spender] == this.allowed[msg.sender][
            _spender] - _subtractedValue
484     @post __return == true
485   */
```

Line 486-501 in File CelerToken.sol

```
486   function decreaseApproval(
487     address _spender,
488     uint256 _subtractedValue
489   )
490     public
491     returns (bool)
492   {
493     uint256 oldValue = allowed[msg.sender][_spender];
494     if (_subtractedValue > oldValue) {
495       allowed[msg.sender][_spender] = 0;
496     } else {
497       allowed[msg.sender][_spender] = oldValue.sub(_subtractedValue);
498     }
499     emit Approval(msg.sender, _spender, allowed[msg.sender][_spender]);
500     return true;
501   }
```

The code meets the specification

Formal Verification Request 41

addAddressToWhitelist

12, Mar 2019
19.6 ms

Line 536-540 in File CelerToken.sol

```
536   /*@CTK "addAddressToWhitelist"
537     @tag assume_completion
538     @post post(this).whitelisted[_operator] == true
539     @post __return == true
540   */
```

Line 541-549 in File CelerToken.sol

```
541   function addAddressToWhitelist(address _operator)
542     onlyOwner
543     public
544     returns (bool)
545   {
546     whitelisted[_operator] = true;
547     emit WhitelistAdded(_operator);
548     return true;
549   }
```

The code meets the specification

Formal Verification Request 42

onlyOwner_addAddressToWhitelist

12, Mar 2019
1.08 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

Line 541-549 in File CelerToken.sol

```
541  function addAddressToWhitelist(address _operator)
542    onlyOwner
543    public
544    returns (bool)
545  {
546    whitelisted[_operator] = true;
547    emit WhitelistAdded(_operator);
548    return true;
549  }
```

The code meets the specification

Formal Verification Request 43

whitelist happy case

12, Mar 2019
7.11 ms

Line 554-557 in File CelerToken.sol

```
554  /*@CTK "whitelist happy case"
555    @post __return == this.whitelisted[_operator]
556    @post post(this) == this
557  */
```

Line 558-565 in File CelerToken.sol

```
558  function whitelist(address _operator)
559    public
560    view
561    returns (bool)
562  {
563    bool result = whitelisted[_operator];
564    return result;
565  }
```

The code meets the specification

Formal Verification Request 44

addAddressesToWhitelist happy case

12, Mar 2019
26.7 ms

Line 573-577 in File CelerToken.sol

```
573  /*@CTK "addAddressesToWhitelist happy case"
574    @tag assume_completion
575    @post forall i: uint. (i >= 0 /\ i < _operators.length) -> post(this).whitelisted[
           _operators[i]] == true
576    @post __return == true
577  */
```

Line 578-594 in File CelerToken.sol

```
578  function addAddressesToWhitelist(address[] _operators)
579    onlyOwner
580    public
581    returns (bool)
582  {
583    /*@CTK addAddressesToWhitelist_forloop
584      @inv i <= _operators.length
585      @inv _operators == _operators__pre
586      @inv forall j: uint. (j >= 0 /\ j < i) -> this.whitelisted[_operators[j]] ==
             true
587      @post i == _operators.length
588      @post !__should_return
589    */
590    for (uint256 i = 0; i < _operators.length; i++) {
591      require(addAddressToWhitelist(_operators[i]));
592    }
593    return true;
594  }
```

The code meets the specification

Formal Verification Request 45

onlyOwner_addAddressesToWhitelist

12, Mar 2019
1.31 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

Line 578-594 in File CelerToken.sol

```
578  function addAddressesToWhitelist(address[] _operators)
579    onlyOwner
580    public
```

```
581     returns (bool)
582   {
583     /*@CTK addAddressesToWhitelist_forloop
584       @inv i <= _operators.length
585       @inv _operators == _operators__pre
586       @inv forall j: uint. (j >= 0 /\ j < i) -> this.whitelisted[_operators[j]] ==
              true
587       @post i == _operators.length
588       @post !__should_return
589     */
590     for (uint256 i = 0; i < _operators.length; i++) {
591       require(addAddressToWhitelist(_operators[i]));
592     }
593     return true;
594   }
```

The code meets the specification

Formal Verification Request 46

removeAddressFromWhitelist happy case

12, Mar 2019
19.08 ms

Line 602-606 in File CelerToken.sol

```
602   /*@CTK "removeAddressFromWhitelist happy case"
603     @tag assume_completion
604     @post post(this).whitelisted[_operator] == false
605     @post __return == true
606   */
```

Line 607-615 in File CelerToken.sol

```
607   function removeAddressFromWhitelist(address _operator)
608     onlyOwner
609     public
610     returns (bool)
611   {
612     whitelisted[_operator] = false;
613     emit WhitelistRemoved(_operator);
614     return true;
615   }
```

The code meets the specification

Formal Verification Request 47

onlyOwner_removeAddressFromWhitelist

12, Mar 2019
0.99 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

Line 607-615 in File CelerToken.sol

```
607  function removeAddressFromWhitelist(address _operator)
608    onlyOwner
609    public
610    returns (bool)
611  {
612    whitelisted[_operator] = false;
613    emit WhitelistRemoved(_operator);
614    return true;
615  }
```

The code meets the specification

Formal Verification Request 48

removeAddressesFromWhitelist happy case

12, Mar 2019
26.47 ms

Line 623-627 in File CelerToken.sol

```
623  /*@CTK "removeAddressesFromWhitelist happy case"
624    @tag assume_completion
625    @post forall i: uint. (i >= 0 /\ i < _operators.length) -> post(this).whitelisted[
           _operators[i]] == false
626    @post __return == true
627  */
```

Line 628-644 in File CelerToken.sol

```
628  function removeAddressesFromWhitelist(address[] _operators)
629    onlyOwner
630    public
631    returns (bool)
632  {
633    /*@CTK removeAddressesFromWhitelist_forLoop
634      @inv i <= _operators.length
635      @inv _operators == _operators__pre
636      @inv forall j: uint. (j >= 0 /\ j < i) -> this.whitelisted[_operators[j]] ==
             false
637      @post i == _operators.length
638      @post !__should_return
639    */
640    for (uint256 i = 0; i < _operators.length; i++) {
641      require(removeAddressFromWhitelist(_operators[i]));
642    }
643    return true;
644  }
```

The code meets the specification

Formal Verification Request 49

onlyOwner_removeAddressesFromWhitelist

12, Mar 2019
1.54 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

Line 628-644 in File CelerToken.sol

```
628  function removeAddressesFromWhitelist(address[] _operators)
629    onlyOwner
630    public
631    returns (bool)
632  {
633    /*@CTK removeAddressesFromWhitelist_forLoop
634      @inv i <= _operators.length
635      @inv _operators == _operators__pre
636      @inv forall j: uint. (j >= 0 /\ j < i) -> this.whitelisted[_operators[j]] ==
             false
637      @post i == _operators.length
638      @post !__should_return
639    */
640    for (uint256 i = 0; i < _operators.length; i++) {
641      require(removeAddressFromWhitelist(_operators[i]));
642    }
643    return true;
644  }
```

The code meets the specification

Formal Verification Request 50

onlyOwner_Ownable__renounceOwnership

12, Mar 2019
15.71 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) Line 240-243 in File CelerToken.sol

```
240  No Snippet Available
```

The code meets the specification

Formal Verification Request 51

onlyOwner_renounceOwnership

12, Mar 2019
15.14 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) Line 240-243 in File CelerToken.sol

```
240 No Snippet Available
```

The code meets the specification

Formal Verification Request 52

onlyOwner_Ownable__transferOwnership

12, Mar 2019
49.93 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) Line 253-255 in File CelerToken.sol

```
253 No Snippet Available
```

The code meets the specification

Formal Verification Request 53

onlyOwner_transferOwnership

12, Mar 2019
37.22 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) Line 253-255 in File CelerToken.sol

```
253 No Snippet Available
```

The code meets the specification

Formal Verification Request 54

addAddressesToWhitelist_forloop__Generated

12, Mar 2019
172.63 ms

(Loop) Line 583-589 in File CelerToken.sol

```
583 No Snippet Available
```

(Loop) Line 583-592 in File CelerToken.sol

```
583 No Snippet Available
```

The code meets the specification

Formal Verification Request 55

removeAddressesFromWhitelist_forLoop__Generated

12, Mar 2019
186.5 ms

(Loop) Line 633-639 in File CelerToken.sol

```
633 No Snippet Available
```

(Loop) Line 633-642 in File CelerToken.sol

```
633 No Snippet Available
```

The code meets the specification

Formal Verification Request 56

PausableToken transfer success case

12, Mar 2019
231.65 ms

Line 655-660 in File CelerToken.sol

```
655  /*@CTK "PausableToken transfer success case"
656    @tag assume_completion
657    @post _to != msg.sender -> __post.balances[msg.sender] == balances[msg.sender] -
           _value
658    @post _to != msg.sender -> __post.balances[_to] == balances[_to] + _value
659    @post __return == true
660  */
```

Line 661-670 in File CelerToken.sol

```
661  function transfer(
662    address _to,
663    uint256 _value
664  )
665    public
666    whenNotPaused
667    returns (bool)
668  {
669    return super.transfer(_to, _value);
670  }
```

The code meets the specification

Formal Verification Request 57

whenNotPaused_transfer

12, Mar 2019
1.7 ms

Line 291-293 in File CelerToken.sol

```
291  /*@CTK "whenNotPaused"
292    @post this.paused -> __reverted
293  */
```

Line 661-670 in File CelerToken.sol

```
661  function transfer(
662    address _to,
663    uint256 _value
664  )
665    public
666    whenNotPaused
667    returns (bool)
668  {
669    return super.transfer(_to, _value);
670  }
```

The code meets the specification

Formal Verification Request 58

PausableToken transferFrom success

12, Mar 2019
433.71 ms

Line 672-678 in File CelerToken.sol

```
672  /*@CTK "PausableToken transferFrom success"
673    @tag assume_completion
674    @post _from != _to -> __post.balances[_from] == balances[_from] - _value
675    @post _from != _to -> __post.balances[_to] == balances[_to] + _value
676    @post __post.allowed[_from][msg.sender] == allowed[_from][msg.sender] - _value
```

```
677     @post __return == true
678   */
```

Line 679-689 in File CelerToken.sol

```
679   function transferFrom(
680     address _from,
681     address _to,
682     uint256 _value
683   )
684     public
685     whenNotPaused
686     returns (bool)
687   {
688     return super.transferFrom(_from, _to, _value);
689   }
```

✅ The code meets the specification

Formal Verification Request 59

whenNotPaused_transferFrom

📅 12, Mar 2019
⏱ 1.85 ms

Line 291-293 in File CelerToken.sol

```
291   /*@CTK "whenNotPaused"
292     @post this.paused -> __reverted
293   */
```

Line 679-689 in File CelerToken.sol

```
679   function transferFrom(
680     address _from,
681     address _to,
682     uint256 _value
683   )
684     public
685     whenNotPaused
686     returns (bool)
687   {
688     return super.transferFrom(_from, _to, _value);
689   }
```

✅ The code meets the specification

Formal Verification Request 60

PausableToken approve transfer allowance

📅 12, Mar 2019
⏱ 47.92 ms

Line 691-695 in File CelerToken.sol

```
691  /*@CTK "PausableToken approve transfer allowance"
692    @pre paused == false
693    @post post(this).allowed[msg.sender][_spender] == _value
694    @post __return == true
695  */
```

Line 696-705 in File CelerToken.sol

```
696  function approve(
697    address _spender,
698    uint256 _value
699  )
700    public
701    whenNotPaused
702    returns (bool)
703  {
704    return super.approve(_spender, _value);
705  }
```

The code meets the specification

Formal Verification Request 61

whenNotPaused_approve

12, Mar 2019
1.33 ms

Line 291-293 in File CelerToken.sol

```
291  /*@CTK "whenNotPaused"
292    @post this.paused -> __reverted
293  */
```

Line 696-705 in File CelerToken.sol

```
696  function approve(
697    address _spender,
698    uint256 _value
699  )
700    public
701    whenNotPaused
702    returns (bool)
703  {
704    return super.approve(_spender, _value);
705  }
```

The code meets the specification

Formal Verification Request 62

increaseApproval ok

12, Mar 2019
71.16 ms

Line 706-709 in File CelerToken.sol

```
706  /*@CTK "increaseApproval ok"
707    @tag assume_completion
708    @post post(this).allowed[msg.sender][_spender] == this.allowed[msg.sender][
           _spender] + _addedValue
709  */
```

Line 710-719 in File CelerToken.sol

```
710  function increaseApproval(
711    address _spender,
712    uint _addedValue
713  )
714    public
715    whenNotPaused
716    returns (bool success)
717  {
718    return super.increaseApproval(_spender, _addedValue);
719  }
```

The code meets the specification

Formal Verification Request 63

whenNotPaused_increaseApproval

12, Mar 2019
1.29 ms

Line 291-293 in File CelerToken.sol

```
291  /*@CTK "whenNotPaused"
292    @post this.paused -> __reverted
293  */
```

Line 710-719 in File CelerToken.sol

```
710  function increaseApproval(
711    address _spender,
712    uint _addedValue
713  )
714    public
715    whenNotPaused
716    returns (bool success)
717  {
718    return super.increaseApproval(_spender, _addedValue);
719  }
```

The code meets the specification

Formal Verification Request 64

decreaseApproval case if

12, Mar 2019

⏱ 102.65 ms

Line 721-725 in File CelerToken.sol

```
721  /*@CTK "decreaseApproval case if"
722    @tag assume_completion
723    @pre _subtractedValue > allowed[msg.sender][_spender]
724    @post post(this).allowed[msg.sender][_spender] == 0
725  */
```

Line 731-740 in File CelerToken.sol

```
731  function decreaseApproval(
732    address _spender,
733    uint _subtractedValue
734  )
735    public
736    whenNotPaused
737    returns (bool success)
738  {
739    return super.decreaseApproval(_spender, _subtractedValue);
740  }
```

✅ The code meets the specification

Formal Verification Request 65

decreaseApproval case else

📅 12, Mar 2019
⏱ 43.69 ms

Line 726-730 in File CelerToken.sol

```
726  /*@CTK "decreaseApproval case else"
727    @tag assume_completion
728    @pre _subtractedValue <= allowed[msg.sender][_spender]
729    @post post(this).allowed[msg.sender][_spender] == this.allowed[msg.sender][
           _spender] - _subtractedValue
730  */
```

Line 731-740 in File CelerToken.sol

```
731  function decreaseApproval(
732    address _spender,
733    uint _subtractedValue
734  )
735    public
736    whenNotPaused
737    returns (bool success)
738  {
739    return super.decreaseApproval(_spender, _subtractedValue);
740  }
```

✅ The code meets the specification

Formal Verification Request 66

whenNotPaused_decreaseApproval

12, Mar 2019
1.94 ms

Line 291-293 in File CelerToken.sol

```
291  /*@CTK "whenNotPaused"
292    @post this.paused -> __reverted
293    */
```

Line 731-740 in File CelerToken.sol

```
731  function decreaseApproval(
732    address _spender,
733    uint _subtractedValue
734  )
735    public
736    whenNotPaused
737    returns (bool success)
738  {
739    return super.decreaseApproval(_spender, _subtractedValue);
740  }
```

The code meets the specification

Formal Verification Request 67

onlyOwner_Pausable__pause

12, Mar 2019
27.04 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224    */
```

(Inheritance) Line 317-320 in File CelerToken.sol

```
317  No Snippet Available
```

The code meets the specification

Formal Verification Request 68

whenNotPaused_Pausable__pause

12, Mar 2019
1.1 ms

Line 291-293 in File CelerToken.sol

```
291  /*@CTK "whenNotPaused"
292    @post this.paused -> __reverted
293  */
```

(Inheritance) Line 317-320 in File CelerToken.sol

```
317 No Snippet Available
```

The code meets the specification

Formal Verification Request 69

onlyOwner_pause

12, Mar 2019
28.01 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) Line 317-320 in File CelerToken.sol

```
317 No Snippet Available
```

The code meets the specification

Formal Verification Request 70

whenNotPaused_pause

12, Mar 2019
1.28 ms

Line 291-293 in File CelerToken.sol

```
291  /*@CTK "whenNotPaused"
292    @post this.paused -> __reverted
293  */
```

(Inheritance) Line 317-320 in File CelerToken.sol

```
317 No Snippet Available
```

The code meets the specification

Formal Verification Request 71

onlyOwner_Pausable__unpause

12, Mar 2019
25.33 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) Line 329-332 in File CelerToken.sol

```
329 No Snippet Available
```

✅ The code meets the specification

Formal Verification Request 72

whenPaused_Pausable__unpause

📅 12, Mar 2019
⏱ 1.08 ms

Line 302-304 in File CelerToken.sol

```
302  /*@CTK "whenPaused"
303    @post !this.paused -> __reverted
304  */
```

(Inheritance) Line 329-332 in File CelerToken.sol

```
329 No Snippet Available
```

✅ The code meets the specification

Formal Verification Request 73

onlyOwner_unpause

📅 12, Mar 2019
⏱ 25.75 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) Line 329-332 in File CelerToken.sol

```
329 No Snippet Available
```

✅ The code meets the specification

Formal Verification Request 74

whenPaused_unpause

📅 12, Mar 2019
⏱ 1.23 ms

Line 302-304 in File CelerToken.sol

```
302  /*@CTK "whenPaused"
303    @post !this.paused -> __reverted
304  */
```

(Inheritance) Line 329-332 in File CelerToken.sol

```
329 No Snippet Available
```

✅ The code meets the specification

Formal Verification Request 75

onlyOwner_Ownable__renounceOwnership

📅 12, Mar 2019
⏱ 17.82 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) (Inheritance) Line 240-243 in File CelerToken.sol

```
240 No Snippet Available
```

✅ The code meets the specification

Formal Verification Request 76

onlyOwner_Pausable__renounceOwnership

📅 12, Mar 2019
⏱ 15.94 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) (Inheritance) Line 240-243 in File CelerToken.sol

```
240 No Snippet Available
```

✅ The code meets the specification

Formal Verification Request 77

onlyOwner_renounceOwnership

📅 12, Mar 2019
⏱ 15.99 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) (Inheritance) Line 240-243 in File CelerToken.sol

```
240 No Snippet Available
```

The code meets the specification

Formal Verification Request 78

onlyOwner_Ownable__transferOwnership

12, Mar 2019
51.84 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) (Inheritance) Line 253-255 in File CelerToken.sol

```
253 No Snippet Available
```

The code meets the specification

Formal Verification Request 79

onlyOwner_Pausable__transferOwnership

12, Mar 2019
37.15 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) (Inheritance) Line 253-255 in File CelerToken.sol

```
253 No Snippet Available
```

The code meets the specification

Formal Verification Request 80

onlyOwner_transferOwnership

12, Mar 2019
38.03 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) (Inheritance) Line 253-255 in File CelerToken.sol

```
253 No Snippet Available
```

The code meets the specification

Formal Verification Request 81

CelerToken constructor

12, Mar 2019
14.71 ms

Line 777-780 in File CelerToken.sol

```
777  /*@CTK "CelerToken constructor"
778    @post post(this).totalSupply_ == INITIAL_SUPPLY
779    @post post(this).balances[msg.sender] == INITIAL_SUPPLY
780  */
```

Line 781-784 in File CelerToken.sol

```
781  constructor() public {
782    totalSupply_ = INITIAL_SUPPLY;
783    balances[msg.sender] = INITIAL_SUPPLY;
784  }
```

The code meets the specification

Formal Verification Request 82

CelerToken transfer success case

12, Mar 2019
407.03 ms

Line 790-795 in File CelerToken.sol

```
790  /*@CTK "CelerToken transfer success case"
791    @tag assume_completion
792    @post _to != msg.sender -> __post.balances[msg.sender] == balances[msg.sender] -
           _value
793    @post _to != msg.sender -> __post.balances[_to] == balances[_to] + _value
794    @post __return == true
795  */
```

Line 796-806 in File CelerToken.sol

```
796  function transfer(
797    address _to,
798    uint256 _value
799  )
```

```
800     public
801     onlyIfTransferable
802     onlyValidReceiver(_to)
803     returns (bool)
804   {
805     return super.transfer(_to, _value);
806   }
```

The code meets the specification

Formal Verification Request 83

onlyValidReceiver_transfer

12, Mar 2019
7.21 ms

Line 759-762 in File CelerToken.sol

```
759   /*@CTK onlyValidReceiver
760     @pre this.transferOpened == false /\ whitelisted[msg.sender] == false /\ msg.
          sender != owner
761     @post __reverted
762   */
```

Line 796-806 in File CelerToken.sol

```
796   function transfer(
797     address _to,
798     uint256 _value
799   )
800     public
801     onlyIfTransferable
802     onlyValidReceiver(_to)
803     returns (bool)
804   {
805     return super.transfer(_to, _value);
806   }
```

The code meets the specification

Formal Verification Request 84

onlyValidReceiver_transfer

12, Mar 2019
7.21 ms

Line 768-770 in File CelerToken.sol

```
768   /*@CTK onlyValidReceiver
769     @post _to == address(this) -> __reverted
770   */
```

Line 796-806 in File CelerToken.sol

```
796  function transfer(
797    address _to,
798    uint256 _value
799  )
800    public
801    onlyIfTransferable
802    onlyValidReceiver(_to)
803    returns (bool)
804  {
805    return super.transfer(_to, _value);
806  }
```

The code meets the specification

Formal Verification Request 85

transferFrom transferFrom success

12, Mar 2019
643.95 ms

Line 812-818 in File CelerToken.sol

```
812  /*@CTK "transferFrom transferFrom success"
813    @tag assume_completion
814    @post _from != _to -> __post.balances[_from] == balances[_from] - _value
815    @post _from != _to -> __post.balances[_to] == balances[_to] + _value
816    @post __post.allowed[_from][msg.sender] == allowed[_from][msg.sender] - _value
817    @post __return == true
818  */
```

Line 819-830 in File CelerToken.sol

```
819  function transferFrom(
820    address _from,
821    address _to,
822    uint256 _value
823  )
824    public
825    onlyIfTransferable
826    onlyValidReceiver(_to)
827    returns (bool)
828  {
829    return super.transferFrom(_from, _to, _value);
830  }
```

The code meets the specification

Formal Verification Request 86

onlyValidReceiver_transferFrom

12, Mar 2019
9.04 ms

Line 759-762 in File CelerToken.sol

```
759  /*@CTK onlyValidReceiver
760    @pre this.transferOpened == false /\ whitelisted[msg.sender] == false /\ msg.
           sender != owner
761    @post __reverted
762  */
```

Line 819-830 in File CelerToken.sol

```
819  function transferFrom(
820    address _from,
821    address _to,
822    uint256 _value
823  )
824    public
825    onlyIfTransferable
826    onlyValidReceiver(_to)
827    returns (bool)
828  {
829    return super.transferFrom(_from, _to, _value);
830  }
```

The code meets the specification

Formal Verification Request 87

onlyValidReceiver_transferFrom

12, Mar 2019
9.04 ms

Line 768-770 in File CelerToken.sol

```
768  /*@CTK onlyValidReceiver
769    @post _to == address(this) -> __reverted
770  */
```

Line 819-830 in File CelerToken.sol

```
819  function transferFrom(
820    address _from,
821    address _to,
822    uint256 _value
823  )
824    public
825    onlyIfTransferable
826    onlyValidReceiver(_to)
827    returns (bool)
828  {
829    return super.transferFrom(_from, _to, _value);
830  }
```

The code meets the specification

Formal Verification Request 88

openTransfer ok

12, Mar 2019
18.62 ms

Line 836-839 in File CelerToken.sol

```
836  /*@CTK "openTransfer ok"
837    @tag assume_completion
838    @post post(this).transferOpened == true
839  */
```

Line 840-842 in File CelerToken.sol

```
840  function openTransfer() external onlyOwner {
841    transferOpened = true;
842  }
```

The code meets the specification

Formal Verification Request 89

onlyOwner_openTransfer

12, Mar 2019
1.02 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

Line 840-842 in File CelerToken.sol

```
840  function openTransfer() external onlyOwner {
841    transferOpened = true;
842  }
```

The code meets the specification

Formal Verification Request 90

onlyOwner_SuccinctWhitelist__addAddressToWhitelist

12, Mar 2019
22.6 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) Line 541-549 in File CelerToken.sol

```
541 No Snippet Available
```

✅ The code meets the specification

Formal Verification Request 91

onlyOwner_addAddressToWhitelist

📅 12, Mar 2019
⏱ 20.63 ms

Line 222-224 in File CelerToken.sol

```
222 /*@CTK "onlyOwner"
223   @post msg.sender != owner -> __reverted
224 */
```

(Inheritance) Line 541-549 in File CelerToken.sol

```
541 No Snippet Available
```

✅ The code meets the specification

Formal Verification Request 92

onlyOwner_SuccinctWhitelist__addAddressesToWhitelist

📅 12, Mar 2019
⏱ 24.53 ms

Line 222-224 in File CelerToken.sol

```
222 /*@CTK "onlyOwner"
223   @post msg.sender != owner -> __reverted
224 */
```

(Inheritance) Line 578-594 in File CelerToken.sol

```
578 No Snippet Available
```

✅ The code meets the specification

Formal Verification Request 93

onlyOwner_addAddressesToWhitelist

📅 12, Mar 2019
⏱ 23.1 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) Line 578-594 in File CelerToken.sol

```
578 No Snippet Available
```

The code meets the specification

Formal Verification Request 94

onlyOwner_SuccinctWhitelist__removeAddressFromWhitelist

12, Mar 2019
21.8 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) Line 607-615 in File CelerToken.sol

```
607 No Snippet Available
```

The code meets the specification

Formal Verification Request 95

onlyOwner_removeAddressFromWhitelist

12, Mar 2019
20.81 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) Line 607-615 in File CelerToken.sol

```
607 No Snippet Available
```

The code meets the specification

Formal Verification Request 96

onlyOwner_SuccinctWhitelist__removeAddressesFromWhitelist

12, Mar 2019
22.95 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) Line 628-644 in File CelerToken.sol

```
628 No Snippet Available
```

The code meets the specification

Formal Verification Request 97

onlyOwner_removeAddressesFromWhitelist

12, Mar 2019
22.54 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) Line 628-644 in File CelerToken.sol

```
628 No Snippet Available
```

The code meets the specification

Formal Verification Request 98

onlyOwner_Ownable__renounceOwnership

12, Mar 2019
18.56 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) (Inheritance) Line 240-243 in File CelerToken.sol

```
240 No Snippet Available
```

The code meets the specification

Formal Verification Request 99

onlyOwner_SuccinctWhitelist__renounceOwnership

12, Mar 2019
17.19 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) (Inheritance) Line 240-243 in File CelerToken.sol

```
240 No Snippet Available
```

✅ The code meets the specification

Formal Verification Request 100

onlyOwner_renounceOwnership

📅 12, Mar 2019
⏱ 18.46 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) (Inheritance) Line 240-243 in File CelerToken.sol

```
240 No Snippet Available
```

✅ The code meets the specification

Formal Verification Request 101

onlyOwner_Ownable__transferOwnership

📅 12, Mar 2019
⏱ 57.21 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) (Inheritance) Line 253-255 in File CelerToken.sol

```
253 No Snippet Available
```

✅ The code meets the specification

Formal Verification Request 102

onlyOwner_SuccinctWhitelist__transferOwnership

📅 12, Mar 2019
⏱ 85.8 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) (Inheritance) Line 253-255 in File CelerToken.sol

```
253 No Snippet Available
```

The code meets the specification

Formal Verification Request 103

onlyOwner_transferOwnership

12, Mar 2019
41.61 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) (Inheritance) Line 253-255 in File CelerToken.sol

```
253 No Snippet Available
```

The code meets the specification

Formal Verification Request 104

whenNotPaused_PausableToken__transfer

12, Mar 2019
1.66 ms

Line 291-293 in File CelerToken.sol

```
291  /*@CTK "whenNotPaused"
292    @post this.paused -> __reverted
293  */
```

(Inheritance) Line 661-670 in File CelerToken.sol

```
661 No Snippet Available
```

The code meets the specification

Formal Verification Request 105

whenNotPaused_PausableToken__transferFrom

12, Mar 2019
1.98 ms

Line 291-293 in File CelerToken.sol

```
291  /*@CTK "whenNotPaused"
292    @post this.paused -> __reverted
293  */
```

(Inheritance) Line 679-689 in File CelerToken.sol

```
679 No Snippet Available
```

The code meets the specification

Formal Verification Request 106

whenNotPaused_PausableToken__approve

12, Mar 2019
52.55 ms

Line 291-293 in File CelerToken.sol

```
291  /*@CTK "whenNotPaused"
292    @post this.paused -> __reverted
293  */
```

(Inheritance) Line 696-705 in File CelerToken.sol

```
696 No Snippet Available
```

The code meets the specification

Formal Verification Request 107

whenNotPaused_approve

12, Mar 2019
40.13 ms

Line 291-293 in File CelerToken.sol

```
291  /*@CTK "whenNotPaused"
292    @post this.paused -> __reverted
293  */
```

(Inheritance) Line 696-705 in File CelerToken.sol

```
696 No Snippet Available
```

The code meets the specification

Formal Verification Request 108

whenNotPaused_PausableToken__increaseApproval

12, Mar 2019
72.21 ms

Line 291-293 in File CelerToken.sol

```
291  /*@CTK "whenNotPaused"
292    @post this.paused -> __reverted
293  */
```

(Inheritance) Line 710-719 in File CelerToken.sol

```
710 No Snippet Available
```

The code meets the specification

Formal Verification Request 109

whenNotPaused_increaseApproval

12, Mar 2019
50.94 ms

Line 291-293 in File CelerToken.sol

```
291  /*@CTK "whenNotPaused"
292    @post this.paused -> __reverted
293  */
```

(Inheritance) Line 710-719 in File CelerToken.sol

```
710 No Snippet Available
```

The code meets the specification

Formal Verification Request 110

whenNotPaused_PausableToken__decreaseApproval

12, Mar 2019
98.82 ms

Line 291-293 in File CelerToken.sol

```
291  /*@CTK "whenNotPaused"
292    @post this.paused -> __reverted
293  */
```

(Inheritance) Line 731-740 in File CelerToken.sol

```
731 No Snippet Available
```

The code meets the specification

Formal Verification Request 111

whenNotPaused_decreaseApproval

12, Mar 2019
66.23 ms

Line 291-293 in File CelerToken.sol

```
291  /*@CTK "whenNotPaused"
292    @post this.paused -> __reverted
293  */
```

(Inheritance) Line 731-740 in File CelerToken.sol

```
731 No Snippet Available
```

✅ The code meets the specification

Formal Verification Request 112

onlyOwner_Pausable__pause

📅 12, Mar 2019
⏱ 31.23 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) (Inheritance) Line 317-320 in File CelerToken.sol

```
317 No Snippet Available
```

✅ The code meets the specification

Formal Verification Request 113

whenNotPaused_Pausable__pause

📅 12, Mar 2019
⏱ 1.13 ms

Line 291-293 in File CelerToken.sol

```
291  /*@CTK "whenNotPaused"
292    @post this.paused -> __reverted
293  */
```

(Inheritance) (Inheritance) Line 317-320 in File CelerToken.sol

```
317 No Snippet Available
```

✅ The code meets the specification

Formal Verification Request 114

onlyOwner_PausableToken__pause

📅 12, Mar 2019
⏱ 32.82 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) (Inheritance) Line 317-320 in File CelerToken.sol

```
317 No Snippet Available
```

✅ The code meets the specification

Formal Verification Request 115

whenNotPaused_PausableToken__pause

📅 12, Mar 2019
⏱ 1.18 ms

Line 291-293 in File CelerToken.sol

```
291  /*@CTK "whenNotPaused"
292    @post this.paused -> __reverted
293  */
```

(Inheritance) (Inheritance) Line 317-320 in File CelerToken.sol

```
317 No Snippet Available
```

✅ The code meets the specification

Formal Verification Request 116

onlyOwner_pause

📅 12, Mar 2019
⏱ 31.68 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) (Inheritance) Line 317-320 in File CelerToken.sol

```
317 No Snippet Available
```

✅ The code meets the specification

Formal Verification Request 117

whenNotPaused_pause

📅 12, Mar 2019
⏱ 1.24 ms

Line 291-293 in File CelerToken.sol

```
291  /*@CTK "whenNotPaused"
292    @post this.paused -> __reverted
293  */
```

(Inheritance) (Inheritance) Line 317-320 in File CelerToken.sol

```
317 No Snippet Available
```

The code meets the specification

Formal Verification Request 118

onlyOwner_Pausable__unpause

12, Mar 2019
30.42 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) (Inheritance) Line 329-332 in File CelerToken.sol

```
329 No Snippet Available
```

The code meets the specification

Formal Verification Request 119

whenPaused_Pausable__unpause

12, Mar 2019
1.26 ms

Line 302-304 in File CelerToken.sol

```
302  /*@CTK "whenPaused"
303    @post !this.paused -> __reverted
304  */
```

(Inheritance) (Inheritance) Line 329-332 in File CelerToken.sol

```
329 No Snippet Available
```

The code meets the specification

Formal Verification Request 120

onlyOwner_PausableToken__unpause

12, Mar 2019
28.81 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) (Inheritance) Line 329-332 in File CelerToken.sol

```
329 No Snippet Available
```

The code meets the specification

Formal Verification Request 121

whenPaused_PausableToken__unpause

12, Mar 2019
1.31 ms

Line 302-304 in File CelerToken.sol

```
302  /*@CTK "whenPaused"
303    @post !this.paused -> __reverted
304  */
```

(Inheritance) (Inheritance) Line 329-332 in File CelerToken.sol

```
329 No Snippet Available
```

The code meets the specification

Formal Verification Request 122

onlyOwner_unpause

12, Mar 2019
28.74 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) (Inheritance) Line 329-332 in File CelerToken.sol

```
329 No Snippet Available
```

The code meets the specification

Formal Verification Request 123

whenPaused_unpause

12, Mar 2019
1.12 ms

Line 302-304 in File CelerToken.sol

```
302  /*@CTK "whenPaused"
303    @post !this.paused -> __reverted
304  */
```

(Inheritance) (Inheritance) Line 329-332 in File CelerToken.sol

```
329 No Snippet Available
```

The code meets the specification

Formal Verification Request 124

onlyOwner_Pausable__renounceOwnership

12, Mar 2019
17.45 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) (Inheritance) (Inheritance) Line 240-243 in File CelerToken.sol

```
240 No Snippet Available
```

The code meets the specification

Formal Verification Request 125

onlyOwner_PausableToken__renounceOwnership

12, Mar 2019
17.66 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) (Inheritance) (Inheritance) Line 240-243 in File CelerToken.sol

```
240 No Snippet Available
```

The code meets the specification

Formal Verification Request 126

onlyOwner_Pausable__transferOwnership

12, Mar 2019
41.88 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) (Inheritance) (Inheritance) Line 253-255 in File CelerToken.sol

```
253 No Snippet Available
```

✅ The code meets the specification

Formal Verification Request 127

onlyOwner_PausableToken__transferOwnership

📅 12, Mar 2019
⏱ 40.2 ms

Line 222-224 in File CelerToken.sol

```
222  /*@CTK "onlyOwner"
223    @post msg.sender != owner -> __reverted
224  */
```

(Inheritance) (Inheritance) (Inheritance) Line 253-255 in File CelerToken.sol

```
253 No Snippet Available
```

✅ The code meets the specification

Formal Verification Request 128

addAddressesToWhitelist_forloop__Generated

📅 12, Mar 2019
⏱ 190.69 ms

(Loop) (Inheritance) Line 583-589 in File CelerToken.sol

```
583 No Snippet Available
```

(Loop) (Inheritance) Line 583-592 in File CelerToken.sol

```
583 No Snippet Available
```

✅ The code meets the specification

Formal Verification Request 129

removeAddressesFromWhitelist_forLoop__Generated

📅 12, Mar 2019
⏱ 336.49 ms

(Loop) (Inheritance) Line 633-639 in File CelerToken.sol

```
633 No Snippet Available
```

(Loop) (Inheritance) Line 633-642 in File CelerToken.sol

```
633 No Snippet Available
```

The code meets the specification