

# EVM Smart Contracts

CCTP Smart Contracts for EVM-compatible blockchains [Suggest Edits](#)

## Contract Responsibilities

- TokenMessenger
  - : Entrypoint for cross-chain USDC transfer. Routes messages to burn USDC on a source chain, and mint USDC on a destination chain.
- MessageTransmitter
  - : Generic message passing. Sends all messages on the source chain, and receives all messages on the destination chain.
- TokenMinter
  - : Responsible for minting and burning USDC. Contains chain-specific settings used by burners and minters.

\*Full contract source code is available at <https://github.com/circlefin/evm-cctp-contracts> .

## Mainnet Contract Addresses

### TokenMessenger: Mainnet

Chain [Domain](#) Address Ethereum 0 [0xbd3fa81b58ba92a82136038b25adec7066af3155](#) Avalanche 1 [0x6b25532e1060ce10cc3b0a99e5683b91bfde6982](#) OP Mainnet 2 [0x2B4069517957735bE00ceE0fadAE88a26365528f](#) Arbitrum 3 [0x19330d10D9Cc8751218eaf51E8885D058642E08A](#) Base 6 [0x1682Ae6375C4E4A97e4B583BC394c861A46D8962](#) Polygon PoS 7 [0x9daF8c91AEFAE50b9c0E69629D3F6Ca40cA3B3FE](#)

### MessageTransmitter: Mainnet

Chain [Domain](#) Address Ethereum 0 [0x0a992d191deec32afe36203ad87d7d289a738f81](#) Avalanche 1 [0x8186359af5f57fbb40c6b14a588d2a59c0c29880](#) OP Mainnet 2 [0x4d41f22c5a0e5c74090899e5a8fb597a8842b3e8](#) Arbitrum 3 [0xC30362313FBBA5cf9163F0bb16a0e01f01A896ca](#) Base 6 [0xAD09780d193884d503182aD4588450C416D6F9D4](#) Polygon PoS 7 [0xF3be9355363857F3e001be68856A2f96b4C39Ba9](#)

### TokenMinter: Mainnet

Chain [Domain](#) Address Ethereum 0 [0xc4922d64a24675e16e1586e3e3aa56c06fabe907](#) Avalanche 1 [0x420f5035fd5dc62a167e7e7f08b604335ae272b8](#) OP Mainnet 2 [0x33E76C5C31cb928dc6FE6487AB3b2C0769B1A1e3](#) Arbitrum 3 [0xE7Ed1fa7f45D05C508232aa32649D89b73b8bA48](#) Base 6 [0xe45B133ddc64bE80252b0e9c75A8E74EF280eEd6](#) Polygon PoS 7 [0x10f7835F827D6Cf035115E10c50A853d7FB2D2EC](#)

## Testnet Contract Addresses

### TokenMessenger: Testnet

Chain [Domain](#) Address Ethereum Sepolia 0 [0x9f3B8679c73C2Fef8b59B4f3444d4e156fb70AA5](#) Avalanche Fuji 1 [0xeb08f243e5d3fcff26a9e38ae5520a669f4019d0](#) OP Sepolia 2 [0x9f3B8679c73C2Fef8b59B4f3444d4e156fb70AA5](#) Arbitrum Sepolia 3 [0x9f3B8679c73C2Fef8b59B4f3444d4e156fb70AA5](#) Base Sepolia 6 [0x9f3B8679c73C2Fef8b59B4f3444d4e156fb70AA5](#) Polygon PoS Mumbai 7 [0x9f3B8679c73C2Fef8b59B4f3444d4e156fb70AA5](#)

### MessageTransmitter: Testnet

Chain [Domain](#) Address Ethereum Sepolia 0 [0x7865fAfC2db2093669d92c0F33AeEF291086BEFD](#) Avalanche Fuji 1 [0xa9fb1b3009dcb79e2fe346c16a604b8fa8ae0a79](#) OP Sepolia 2 [0x7865fAfC2db2093669d92c0F33AeEF291086BEFD](#) Arbitrum Sepolia 3 [0xaCF1ceeF35caAc005e15888dDb8A3515C41B4872](#) Base Sepolia 6 [0x7865fAfC2db2093669d92c0F33AeEF291086BEFD](#) Polygon PoS Mumbai 7 [0xe09A679F56207EF33F5b9d8fb4499Ec00792eA73](#)

### TokenMinter: Testnet

Chain [Domain](#) Address Ethereum Sepolia 0 [0xE997d7d2F6E065a9A93Fa2175E878Fb9081F1f0A](#) Avalanche Fuji 1 [0x4ed8867f9947a5fe140c9dc1c6f207f3489f501e](#) OP Sepolia 2 [0xE997d7d2F6E065a9A93Fa2175E878Fb9081F1f0A](#) Arbitrum Sepolia 3 [0xE997d7d2F6E065a9A93Fa2175E878Fb9081F1f0A](#) Base Sepolia 6 [0xE997d7d2F6E065a9A93Fa2175E878Fb9081F1f0A](#) Polygon PoS Mumbai 7

## Interface

The interface below serves as a reference for permissionless messaging functions exposed by the TokenMessenger and MessageTransmitter functions. The full ABIs are listed [here](#).

### TokenMessenger

#### depositForBurn

Deposits and burns tokens from sender to be minted on destination domain. Minted tokens will be transferred to mintRecipient.

##### Parameters

Field	Type	Description
amount	uint256	Amount of tokens to deposit and burn.
destinationDomain	uint32	Destination domain identifier.
mintRecipient	bytes32	Address of mint recipient on destination domain.
burnToken	address	Address of contract to burn deposited tokens on local domain.

#### depositForBurnWithCaller

Same as depositForBurn but with an additional parameter, destinationCaller. This parameter specifies which address has permission to call receiveMessage on the destination domain for the message.

##### Parameters

Field	Type	Description
amount	uint256	Amount of tokens to deposit and burn.
destinationDomain	uint32	Destination domain identifier.
mintRecipient	bytes32	Address of mint recipient on destination domain.
burnToken	address	Address of contract to burn deposited tokens on local domain.
destinationCaller	bytes32	Address of caller on the destination domain.

#### replaceDepositForBurn

Replace a BurnMessage to change the mint recipient and/or destination caller. Allows the sender of a previous BurnMessage (created by depositForBurn or depositForBurnWithCaller) to send a new BurnMessage to replace the original. The new BurnMessage will reuse the amount and burn token of the original, without requiring a new deposit.

This is useful in situations where the user specified an incorrect address and has no way to safely mint the previously burned USDC.

The sender of the original depositForBurn has access to call replaceDepositForBurn. The resulting mint will supersede the original mint, as long as the original mint has not confirmed yet on-chain. When using a third-party app/bridge that integrates with CCTP to burn and mint USDC, it is the choice of the app/bridge if and when to replace messages on behalf of users. When sending USDC to smart contracts, be aware of the functionality that those contracts have and their respective trust model.

##### Parameters

Field	Type	Description
originalMessage	bytes	Original message bytes (to replace).
originalAttestation	bytes	Original attestation bytes.
newDestinationCaller	bytes32	The new destination caller, which may be the same as the original destination caller, a new destination caller, or an empty destination caller, indicating that any destination caller is valid.
newMintRecipient	bytes32	The new mint recipient, which may be the same as the original mint recipient, or different.

### MessageTransmitter

#### receiveMessage

Messages with a given nonce can only be broadcast successfully once for a pair of domains. The message body of a valid message is passed to the specified recipient for further processing.

##### Parameters

Field	Type	Description
message	bytes	Message bytes.
attestation	bytes	Signed attestation of message.

#### sendMessage

Sends a message to the destination domain and recipient. Emits a MessageSent event which will be attested by Circle's

attestation service.

#### Parameters

Field Type Description destinationDomain uint32 Destination domain identifier. recipient bytes32 Address to handle message body on destination domain. messageBody bytes calldata Application-specific message to be handled by recipient.

### **sendMessageWithCaller**

Same as sendMessage but with an additional parameter, destinationCaller . This parameter specifies which address has permission to call receiveMessage on the destination domain for the message.

#### Parameters

Field Type Description destinationDomain uint32 Destination domain identifier. recipient bytes32 Address of message recipient on destination domain. destinationCaller bytes32 Address of caller on the destination domain. messageBody bytes calldata Application-specific message to be handled by recipient.

### **replaceMessage**

Replace a message with a new message body and/or destination caller. The originalAttestation must be a valid attestation of originalMessage , produced by Circle's attestation service.

#### Parameters

Field Type Description originalMessage bytes calldata Original message to replace. originalAttestation bytes calldata Attestation of originalMessage. newMessageBody bytes calldata New message body of replaced message. newDestinationCaller bytes32 The new destination caller, which may be the same as the original destination caller, a new destination caller, or an empty destination caller (bytes32(0), indicating that any destination caller is valid). Updated about 20 hours ago \* [Table of Contents](#) \* \* [Contract Responsibilities](#) \* \* [Mainnet Contract Addresses](#) \* \* \* [TokenMessenger: Mainnet](#) \* \* \* [MessageTransmitter: Mainnet](#) \* \* \* [TokenMinter: Mainnet](#) \* \* [Testnet Contract Addresses](#) \* \* \* [TokenMessenger: Testnet](#) \* \* \* [MessageTransmitter: Testnet](#) \* \* \* [TokenMinter: Testnet](#) \* \* [Interface](#) \* \* \* [TokenMessenger](#) \* \* \* [MessageTransmitter](#)