Module sui::groth16

Represents an elliptic curve construction to be used in the verifier. Currently we support BLS12-381 and BN254. This should be given as the first parameter to prepare_verifying_key or verify_groth16_proof .

A PreparedVerifyingKey consisting of four components in serialized form.

A PublicProofInputs wrapper around its serialized bytes.

A ProofPoints wrapper around the serialized form of three proof points.

Return the Curve value indicating that the BLS12-381 construction should be used in a given function.

Return the Curve value indicating that the BN254 construction should be used in a given function.

Creates a PreparedVerifyingKey from bytes.

Returns bytes of the four components of the PreparedVerifyingKey .

Creates a PublicProofInputs wrapper from bytes. The bytes parameter should be a concatenation of a number of 32 bytes scalar field elements to be used as public inputs in little-endian format to a circuit.

Creates a Groth16 ProofPoints from bytes.

@param curve: What elliptic curve construction to use. See bls12381 and bn254 . @param verifying_key: An Arkworks canonical compressed serialization of a verifying key.

Returns four vectors of bytes representing the four components of a prepared verifying key. This step computes one pairing e(P, Q), and binds the verification to one particular proof statement. This can be used as inputs for the verify_groth16_proof function.

Native functions that flattens the inputs into an array and passes to the Rust native function. May abort with EInvalidVerifyingKey or EInvalidCurve .

@param curve: What elliptic curve construction to use. See the bls12381 and bn254 functions. @param prepared_verifying_key: Consists of four vectors of bytes representing the four components of a prepared verifying key. @param public_proof_inputs: Represent inputs that are public. @param proof_points: Represent three proof points.

Returns a boolean indicating whether the proof is valid.

Native functions that flattens the inputs into arrays of vectors and passed to the Rust native function. May abort with EInvalidCurve or ETooManyPublicInputs .

Struct

Represents an elliptic curve construction to be used in the verifier. Currently we support BLS12-381 and BN254. This should be given as the first parameter to prepare_verifying_key or verify_groth16_proof .

```bash

```

A PreparedVerifyingKey consisting of four components in serialized form.

```bash

```

A PublicProofInputs wrapper around its serialized bytes.

```bash

```

A ProofPoints wrapper around the serialized form of three proof points.

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

Return the Curve value indicating that the BLS12-381 construction should be used in a given function.

```bash
```

```bash
```

Return the Curve value indicating that the BN254 construction should be used in a given function.

```bash
```

```bash
```

Creates a PreparedVerifyingKey from bytes.

```bash
```

```bash
```

Returns bytes of the four components of the PreparedVerifyingKey .

```bash
```

```bash
```

Creates a PublicProofInputs wrapper from bytes. The bytes parameter should be a concatenation of a number of 32 bytes scalar field elements to be used as public inputs in little-endian format to a circuit.

```bash
```

```bash
```

Creates a Groth16 ProofPoints from bytes.

```bash
```

```bash
```

@param curve: What elliptic curve construction to use. See bls12381 and bn254 . @param verifying_key: An Arkworks canonical compressed serialization of a verifying key.

Returns four vectors of bytes representing the four components of a prepared verifying key. This step computes one pairing e(P, Q), and binds the verification to one particular proof statement. This can be used as inputs for the verify_groth16_proof function.

```bash
```

```bash
```

Native functions that flattens the inputs into an array and passes to the Rust native function. May abort with EInvalidVerifyingKey or EInvalidCurve .

```bash
```

```bash
```

@param curve: What elliptic curve construction to use. See the bls12381 and bn254 functions. @param prepared_verifying_key: Consists of four vectors of bytes representing the four components of a prepared verifying key. @param public_proof_inputs: Represent inputs that are public. @param proof_points: Represent three proof points.

Returns a boolean indicating whether the proof is valid.

```bash
```

```bash
```

Native functions that flattens the inputs into arrays of vectors and passed to the Rust native function. May abort with EInvalidCurve or ETooManyPublicInputs .

```bash
```

```bash
```

Struct

A PreparedVerifyingKey consisting of four components in serialized form.

```bash
```

A PublicProofInputs wrapper around its serialized bytes.

```bash
```

A ProofPoints wrapper around the serialized form of three proof points.

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

Return the Curve value indicating that the BLS12-381 construction should be used in a given function.

```bash
```

```bash
```

Return the Curve value indicating that the BN254 construction should be used in a given function.

```bash
```

```bash
```

Creates a PreparedVerifyingKey from bytes.

```bash
```

```bash
```

Returns bytes of the four components of the PreparedVerifyingKey .

```bash

```

```bash

```

Creates a PublicProofInputs wrapper from bytes. The bytes parameter should be a concatenation of a number of 32 bytes scalar field elements to be used as public inputs in little-endian format to a circuit.

```bash

```

```bash

```

Creates a Groth16 ProofPoints from bytes.

```bash

```

```bash

```

@param curve: What elliptic curve construction to use. See bls12381 and bn254 . @param verifying_key: An Arkworks canonical compressed serialization of a verifying key.

Returns four vectors of bytes representing the four components of a prepared verifying key. This step computes one pairing e(P, Q), and binds the verification to one particular proof statement. This can be used as inputs for the verify_groth16_proof function.

```bash

```

```bash

```

Native functions that flattens the inputs into an array and passes to the Rust native function. May abort with EInvalidVerifyingKey or EInvalidCurve .

```bash

```

```bash

```

@param curve: What elliptic curve construction to use. See the bls12381 and bn254 functions. @param prepared_verifying_key: Consists of four vectors of bytes representing the four components of a prepared verifying key. @param public_proof_inputs: Represent inputs that are public. @param proof_points: Represent three proof points.

Returns a boolean indicating whether the proof is valid.

```bash

```

```bash
```

Native functions that flattens the inputs into arrays of vectors and passed to the Rust native function. May abort with EInvalidCurve or ETooManyPublicInputs .

```bash
```

```bash
```

Struct

A PublicProofInputs wrapper around its serialized bytes.

```bash
```

A ProofPoints wrapper around the serialized form of three proof points.

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

Return the Curve value indicating that the BLS12-381 construction should be used in a given function.

```bash
```

```bash
```

Return the Curve value indicating that the BN254 construction should be used in a given function.

```bash
```

```bash
```

Creates a PreparedVerifyingKey from bytes.

```bash
```

```bash
```

Returns bytes of the four components of the PreparedVerifyingKey .

```bash
```

```bash
```

Creates a PublicProofInputs wrapper from bytes. The bytes parameter should be a concatenation of a number of 32 bytes scalar field elements to be used as public inputs in little-endian format to a circuit.

```bash
```

```bash
```

Creates a Groth16 ProofPoints from bytes.

```bash
```

```bash
```

@param curve: What elliptic curve construction to use. See bls12381 and bn254 . @param verifying_key: An Arkworks canonical compressed serialization of a verifying key.

Returns four vectors of bytes representing the four components of a prepared verifying key. This step computes one pairing e(P, Q), and binds the verification to one particular proof statement. This can be used as inputs for the verify_groth16_proof function.

```bash
```

```bash
```

Native functions that flattens the inputs into an array and passes to the Rust native function. May abort with EInvalidVerifyingKey or EInvalidCurve .

```bash
```

```bash
```

@param curve: What elliptic curve construction to use. See the bls12381 and bn254 functions. @param prepared_verifying_key: Consists of four vectors of bytes representing the four components of a prepared verifying key. @param public_proof_inputs: Represent inputs that are public. @param proof_points: Represent three proof points.

Returns a boolean indicating whether the proof is valid.

```bash
```

```bash
```

Native functions that flattens the inputs into arrays of vectors and passed to the Rust native function. May abort with EInvalidCurve or ETooManyPublicInputs .

```bash
```

```bash
```

Struct

A ProofPoints wrapper around the serialized form of three proof points.

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

Return the Curve value indicating that the BLS12-381 construction should be used in a given function.

```bash
```

```bash
```

Return the Curve value indicating that the BN254 construction should be used in a given function.

```bash
```

```bash

```

Creates a PreparedVerifyingKey from bytes.

```bash

```

```bash

```

Returns bytes of the four components of the PreparedVerifyingKey .

```bash

```

```bash

```

Creates a PublicProofInputs wrapper from bytes. The bytes parameter should be a concatenation of a number of 32 bytes scalar field elements to be used as public inputs in little-endian format to a circuit.

```bash

```

```bash

```

Creates a Groth16 ProofPoints from bytes.

```bash

```

```bash

```

@param curve: What elliptic curve construction to use. See bls12381 and bn254 . @param verifying_key: An Arkworks canonical compressed serialization of a verifying key.

Returns four vectors of bytes representing the four components of a prepared verifying key. This step computes one pairing e(P, Q), and binds the verification to one particular proof statement. This can be used as inputs for the verify_groth16_proof function.

```bash

```

```bash

```

Native functions that flattens the inputs into an array and passes to the Rust native function. May abort with EInvalidVerifyingKey or EInvalidCurve .

```bash

```

```bash
```

@param curve: What elliptic curve construction to use. See the bls12381 and bn254 functions. @param prepared_verifying_key: Consists of four vectors of bytes representing the four components of a prepared verifying key. @param public_proof_inputs: Represent inputs that are public. @param proof_points: Represent three proof points.

Returns a boolean indicating whether the proof is valid.

```bash
```

```bash
```

Native functions that flattens the inputs into arrays of vectors and passed to the Rust native function. May abort with EInvalidCurve or ETooManyPublicInputs .

```bash
```

```bash
```

Constants

```bash
```

```bash
```

```bash
```

```bash
```

```bash
```

Return the Curve value indicating that the BLS12-381 construction should be used in a given function.

```bash
```

```bash
```

Return the Curve value indicating that the BN254 construction should be used in a given function.

```bash
```

```bash
```

Creates a PreparedVerifyingKey from bytes.

```bash
```

```bash
```

Returns bytes of the four components of the PreparedVerifyingKey .

```bash
```

```bash
```

Creates a PublicProofInputs wrapper from bytes. The bytes parameter should be a concatenation of a number of 32 bytes scalar field elements to be used as public inputs in little-endian format to a circuit.

```bash
```

```bash
```

Creates a Groth16 ProofPoints from bytes.

```bash
```

```bash
```

@param curve: What elliptic curve construction to use. See bls12381 and bn254 . @param verifying_key: An Arkworks canonical compressed serialization of a verifying key.

Returns four vectors of bytes representing the four components of a prepared verifying key. This step computes one pairing e(P, Q), and binds the verification to one particular proof statement. This can be used as inputs for the verify_groth16_proof function.

```bash
```

```bash
```

Native functions that flattens the inputs into an array and passes to the Rust native function. May abort with EInvalidVerifyingKey or EInvalidCurve .

```bash
```

```bash
```

@param curve: What elliptic curve construction to use. See the bls12381 and bn254 functions. @param prepared_verifying_key: Consists of four vectors of bytes representing the four components of a prepared verifying key. @param public_proof_inputs: Represent inputs that are public. @param proof_points: Represent three proof points.

Returns a boolean indicating whether the proof is valid.

```bash
```

```bash
```

Native functions that flattens the inputs into arrays of vectors and passed to the Rust native function. May abort with EInvalidCurve or ETooManyPublicInputs .

```bash
```

```bash
```

Function

Return the Curve value indicating that the BLS12-381 construction should be used in a given function.

```bash
```

```bash
```

Return the Curve value indicating that the BN254 construction should be used in a given function.

```bash
```

```bash
```

Creates a PreparedVerifyingKey from bytes.

```bash
```

```bash
```

Returns bytes of the four components of the PreparedVerifyingKey .

```bash
```

```bash
```

Creates a PublicProofInputs wrapper from bytes. The bytes parameter should be a concatenation of a number of 32 bytes scalar field elements to be used as public inputs in little-endian format to a circuit.

```bash
```

```bash
```

Creates a Groth16 ProofPoints from bytes.

```bash
```

```bash
```

@param curve: What elliptic curve construction to use. See bls12381 and bn254 . @param verifying_key: An Arkworks canonical compressed serialization of a verifying key.

Returns four vectors of bytes representing the four components of a prepared verifying key. This step computes one pairing e(P, Q), and binds the verification to one particular proof statement. This can be used as inputs for the verify_groth16_proof function.

```bash
```

```bash
```

Native functions that flattens the inputs into an array and passes to the Rust native function. May abort with EInvalidVerifyingKey or EInvalidCurve .

```bash
```

```bash
```

@param curve: What elliptic curve construction to use. See the bls12381 and bn254 functions. @param prepared_verifying_key: Consists of four vectors of bytes representing the four components of a prepared verifying key. @param public_proof_inputs: Represent inputs that are public. @param proof_points: Represent three proof points.

Returns a boolean indicating whether the proof is valid.

```bash
```

```bash
```

Native functions that flattens the inputs into arrays of vectors and passed to the Rust native function. May abort with EInvalidCurve or ETooManyPublicInputs .

```bash
```

```bash
```

Function

Return the Curve value indicating that the BN254 construction should be used in a given function.

```bash
```

```bash
```

Creates a PreparedVerifyingKey from bytes.

```bash
```

```bash
```

Returns bytes of the four components of the PreparedVerifyingKey .

```bash
```

```bash
```

Creates a PublicProofInputs wrapper from bytes. The bytes parameter should be a concatenation of a number of 32 bytes scalar field elements to be used as public inputs in little-endian format to a circuit.

```bash
```

```bash
```

Creates a Groth16 ProofPoints from bytes.

```bash
```

```bash
```

@param curve: What elliptic curve construction to use. See bls12381 and bn254 . @param verifying_key: An Arkworks canonical compressed serialization of a verifying key.

Returns four vectors of bytes representing the four components of a prepared verifying key. This step computes one pairing e(P, Q), and binds the verification to one particular proof statement. This can be used as inputs for the verify_groth16_proof function.

```bash
```

```bash

```

Native functions that flattens the inputs into an array and passes to the Rust native function. May abort with EInvalidVerifyingKey or EInvalidCurve .

```bash

```

```bash

```

@param curve: What elliptic curve construction to use. See the bls12381 and bn254 functions. @param prepared_verifying_key: Consists of four vectors of bytes representing the four components of a prepared verifying key. @param public_proof_inputs: Represent inputs that are public. @param proof_points: Represent three proof points.

Returns a boolean indicating whether the proof is valid.

```bash

```

```bash

```

Native functions that flattens the inputs into arrays of vectors and passed to the Rust native function. May abort with EInvalidCurve or ETooManyPublicInputs .

```bash

```

```bash

```

Function

Creates a PreparedVerifyingKey from bytes.

```bash

```

```bash

```

Returns bytes of the four components of the PreparedVerifyingKey .

```bash

```

```bash

```

Creates a PublicProofInputs wrapper from bytes. The bytes parameter should be a concatenation of a number of 32 bytes scalar field elements to be used as public inputs in little-endian format to a circuit.

```bash
```

```bash
```

Creates a Groth16 ProofPoints from bytes.

```bash
```

```bash
```

@param curve: What elliptic curve construction to use. See bls12381 and bn254 . @param verifying_key: An Arkworks canonical compressed serialization of a verifying key.

Returns four vectors of bytes representing the four components of a prepared verifying key. This step computes one pairing e(P, Q), and binds the verification to one particular proof statement. This can be used as inputs for the verify_groth16_proof function.

```bash
```

```bash
```

Native functions that flattens the inputs into an array and passes to the Rust native function. May abort with EInvalidVerifyingKey or EInvalidCurve .

```bash
```

```bash
```

@param curve: What elliptic curve construction to use. See the bls12381 and bn254 functions. @param prepared_verifying_key: Consists of four vectors of bytes representing the four components of a prepared verifying key. @param public_proof_inputs: Represent inputs that are public. @param proof_points: Represent three proof points.

Returns a boolean indicating whether the proof is valid.

```bash
```

```bash
```

Native functions that flattens the inputs into arrays of vectors and passed to the Rust native function. May abort with EInvalidCurve or ETooManyPublicInputs .

```bash
```

```bash
```

Function

Returns bytes of the four components of the PreparedVerifyingKey .

```bash

```

```bash

```

Creates a PublicProofInputs wrapper from bytes. The bytes parameter should be a concatenation of a number of 32 bytes scalar field elements to be used as public inputs in little-endian format to a circuit.

```bash

```

```bash

```

Creates a Groth16 ProofPoints from bytes.

```bash

```

```bash

```

@param curve: What elliptic curve construction to use. See bls12381 and bn254 . @param verifying_key: An Arkworks canonical compressed serialization of a verifying key.

Returns four vectors of bytes representing the four components of a prepared verifying key. This step computes one pairing e(P, Q), and binds the verification to one particular proof statement. This can be used as inputs for the verify_groth16_proof function.

```bash

```

```bash

```

Native functions that flattens the inputs into an array and passes to the Rust native function. May abort with EInvalidVerifyingKey or EInvalidCurve .

```bash

```

```bash

```

@param curve: What elliptic curve construction to use. See the bls12381 and bn254 functions. @param prepared_verifying_key: Consists of four vectors of bytes representing the four components of a prepared verifying key. @param public_proof_inputs: Represent inputs that are public. @param proof_points: Represent three proof points.

Returns a boolean indicating whether the proof is valid.

```bash
```

```bash

```

Native functions that flattens the inputs into arrays of vectors and passed to the Rust native function. May abort with EInvalidCurve or ETooManyPublicInputs .

```bash

```

```bash

```

Function

Creates a PublicProofInputs wrapper from bytes. The bytes parameter should be a concatenation of a number of 32 bytes scalar field elements to be used as public inputs in little-endian format to a circuit.

```bash

```

```bash

```

Creates a Groth16 ProofPoints from bytes.

```bash

```

```bash

```

@param curve: What elliptic curve construction to use. See bls12381 and bn254 . @param verifying_key: An Arkworks canonical compressed serialization of a verifying key.

Returns four vectors of bytes representing the four components of a prepared verifying key. This step computes one pairing e(P, Q), and binds the verification to one particular proof statement. This can be used as inputs for the verify_groth16_proof function.

```bash

```

```bash

```

Native functions that flattens the inputs into an array and passes to the Rust native function. May abort with EInvalidVerifyingKey or EInvalidCurve .

```bash

```

```bash

```

@param curve: What elliptic curve construction to use. See the bls12381 and bn254 functions. @param prepared_verifying_key:

Consists of four vectors of bytes representing the four components of a prepared verifying key. @param public_proof_inputs: Represent inputs that are public. @param proof_points: Represent three proof points.

Returns a boolean indicating whether the proof is valid.

```bash
```

```bash
```

Native functions that flattens the inputs into arrays of vectors and passed to the Rust native function. May abort with EInvalidCurve or ETooManyPublicInputs .

```bash
```

```bash
```

Function

Creates a Groth16 ProofPoints from bytes.

```bash
```

```bash
```

@param curve: What elliptic curve construction to use. See bls12381 and bn254 . @param verifying_key: An Arkworks canonical compressed serialization of a verifying key.

Returns four vectors of bytes representing the four components of a prepared verifying key. This step computes one pairing e(P, Q), and binds the verification to one particular proof statement. This can be used as inputs for the verify_groth16_proof function.

```bash
```

```bash
```

Native functions that flattens the inputs into an array and passes to the Rust native function. May abort with EInvalidVerifyingKey or EInvalidCurve .

```bash
```

```bash
```

@param curve: What elliptic curve construction to use. See the bls12381 and bn254 functions. @param prepared_verifying_key: Consists of four vectors of bytes representing the four components of a prepared verifying key. @param public_proof_inputs: Represent inputs that are public. @param proof_points: Represent three proof points.

Returns a boolean indicating whether the proof is valid.

```bash
```

```bash
```

Native functions that flattens the inputs into arrays of vectors and passed to the Rust native function. May abort with EInvalidCurve or ETooManyPublicInputs .

```bash
```

```bash
```

Function

@param curve: What elliptic curve construction to use. See bls12381 and bn254 . @param verifying_key: An Arkworks canonical compressed serialization of a verifying key.

Returns four vectors of bytes representing the four components of a prepared verifying key. This step computes one pairing e(P, Q), and binds the verification to one particular proof statement. This can be used as inputs for the verify_groth16_proof function.

```bash
```

```bash
```

Native functions that flattens the inputs into an array and passes to the Rust native function. May abort with EInvalidVerifyingKey or EInvalidCurve .

```bash
```

```bash
```

@param curve: What elliptic curve construction to use. See the bls12381 and bn254 functions. @param prepared_verifying_key: Consists of four vectors of bytes representing the four components of a prepared verifying key. @param public_proof_inputs: Represent inputs that are public. @param proof_points: Represent three proof points.

Returns a boolean indicating whether the proof is valid.

```bash
```

```bash
```

Native functions that flattens the inputs into arrays of vectors and passed to the Rust native function. May abort with EInvalidCurve or ETooManyPublicInputs .

```bash
```

```bash
```

Function

Native functions that flattens the inputs into an array and passes to the Rust native function. May abort with EInvalidVerifyingKey or EInvalidCurve .

```bash
```

```bash
```

@param curve: What elliptic curve construction to use. See the bls12381 and bn254 functions. @param prepared_verifying_key: Consists of four vectors of bytes representing the four components of a prepared verifying key. @param public_proof_inputs: Represent inputs that are public. @param proof_points: Represent three proof points.

Returns a boolean indicating whether the proof is valid.

```bash
```

```bash
```

Native functions that flattens the inputs into arrays of vectors and passed to the Rust native function. May abort with EInvalidCurve or ETooManyPublicInputs .

```bash
```

```bash
```

Function

@param curve: What elliptic curve construction to use. See the bls12381 and bn254 functions. @param prepared_verifying_key: Consists of four vectors of bytes representing the four components of a prepared verifying key. @param public_proof_inputs: Represent inputs that are public. @param proof_points: Represent three proof points.

Returns a boolean indicating whether the proof is valid.

```bash
```

```bash
```

Native functions that flattens the inputs into arrays of vectors and passed to the Rust native function. May abort with EInvalidCurve or ETooManyPublicInputs .

```bash
```

```bash
```

Function

Native functions that flattens the inputs into arrays of vectors and passed to the Rust native function. May abort with EInvalidCurve or ETooManyPublicInputs .

```bash
```

```bash
```