Precompiles reference

ArbOS provides L2-specific precompiles with methods smart contracts can call the same way they can solidity functions. This reference page exhaustively documents the specific calls ArbOS makes available through precompiles. For a more conceptual description of what precompiles are and how they work, please refer to theprecompiles conceptual page .

This reference page is divided into two sections. The first one lists all precompiles in a summary table with links to the reference of the specific precompile, along with the address where they live, their purpose and links to the go implementation and solidity interface. The second one details the methods available in each precompile with links to the specific implementation.

General information of precompiles

This section is divided into two tables. We first list precompiles we expect users to most often use, and then the rest of precompiles. However, both tables display the same information: name and purpose of the precompile, address, and links to the solidity interface and the go implementation.

Common precompiles

Precompile Address Solidity interface Go implementation PurposeArbAggregator 0x6d Interface Implementation Configuring transaction aggregation ArbGasInfo 0x6c Interface Implementation Info about gas pricing ArbRetryableTx 0x6e Interface Implementation Managing retryables ArbSys 0x64 Interface Implementation System-level functionality

Other precompiles

Precompile Address Solidity interface Go implementation PurposeArbAddressTable 0x66 Interface Implementation Supporting compression of addresses ArbBLS - - - Disabled (Former registry of BLS public keys) ArbDebug 0xff Interface Implementation Testing tools ArbFunctionTable 0x68 Interface Implementation No longer usedArbInfo 0x65 Interface Implementation Info about accounts ArbOwner 0x70 Interface Implementation Chain administration, callable only by chain owner ArbOwnerPublic 0x6b Interface Implementation Info about chain owners ArbosTest 0x69 Interface Implementation No longer used ArbStatistics 0x6f Interface Implementation Info about the pre-Nitro state

Precompiles reference

ArbAddressTable

ArbAddressTable (Interface |Implementation) provides the ability to create short-hands for commonly used accounts.

Precompile address:0x0000000000000000000000000000000000000066

Method Solidity interface Go implementation Description addressExists(address addr)Interface Implementation AddressExists checks if an address exists in the table compress(address addr) Interface Implementation Compress and returns the bytes that represent the address decompress(bytes calldata buf, uint256 offset) Interface Implementation Decompress the compressed bytes at the given offset with those of the corresponding account lookup(address addr) Interface Implementation Lookup the index of an address in the table lookupIndex(uint256 index)Interface Implementation LookupIndex for an address in the table by index register(address addr) Interface Implementation Register adds an account to the table, shrinking its compressed representation size() Interface Implementation Size gets the number of addresses in the table

ArbAggregator

ArbAggregator (Interface |Implementation) provides aggregators and their users methods for configuring how they participate in L1 aggregation. Arbitrum One's default aggregator is the Sequencer, which a user will prefer unlessSetPreferredAggregator is invoked to change it.

Compression ratios are measured in basis points. Methods that are checkmarked are access-controlled and will revert if not called by the aggregator, its fee collector, or a chain owner.

Precompile address:0x000000000000000000000000000000000000006D

Method Solidity interface Go implementation Description ⚠getPreferredAggregator(address addr)Interface Implementation Deprecated: Do not use this method. ⚠getDefaultAggregator() Interface Implementation Deprecated: Do not use this method. getBatchPosters() Interface Implementation GetBatchPosters gets the addresses of all current batch posters addBatchPoster(address newBatchPoster) Interface Implementation Adds newBatchPoster as a batch poster getFeeCollector(address batchPoster) Interface Implementation GetFeeCollector gets a batch poster's fee collector setFeeCollector(address batchPoster, address newFeeCollector) Interface Implementation SetFeeCollector sets a batch poster's fee collector (caller must be the batch poster, its fee collector, or an owner) ⚠getTxBaseFee(address aggregator)

Interface Implementation Deprecated: returns 0 ⚠setTxBaseFee(address aggregator, uint256 feeInL1Gas) Interface Implementation Deprecated: does nothing Note: methods marked with ⚠ are deprecated and their use is not supported.

ArbBLS

Disabled This precompile has been disabled. It previously provided a registry of BLS public keys for accounts.

ArbDebug

ArbDebug (Interface |Implementation) provides mechanisms useful for testing. The methods ofArbDebug are only available for chains with theAllowDebugPrecompiles chain parameter set. Otherwise, calls to this precompile will revert.

Precompile address:0x00000000000000000000000000000000000000ff

Method Solidity interface Go implementation Description becomeChainOwner()Interface Implementation Caller becomes a chain owner events(bool flag, bytes32 value) Interface Implementation Emit events with values based on the args provided eventsView() Interface Implementation Tries (and fails) to emit logs in a view context customRevert(uint64 number)Interface Implementation Throws a custom error legacyError() Interface Implementation Throws a hardcoded error Event Solidity interface Go implementation Description Basic Interface Implementation Emitted inEvents for testing Mixed Interface Implementation Emitted inEvents for testing Store Interface Implementation Never emitted (used for testing log sizes)

ArbFunctionTable

ArbFunctionTable (Interface |Implementation) provides aggregators the ability to manage function tables, to enable one form of transaction compression. The Nitro aggregator implementation does not use these, so these methods have been stubbed and their effects disabled. They are kept for backwards compatibility.

Precompile address:0x0000000000000000000000000000000000000068

Method Solidity interface Go implementation Description upload(bytes calldata buf)Interface Implementation Upload does nothing size(address addr) Interface Implementation Size returns the empty table's size, which is 0 get(address addr, uint256 index) Interface Implementation Get reverts since the table is empty

ArbGasInfo

ArbGasInfo (Interface |Implementation) provides insight into the cost of using the chain. These methods have been adjusted to account for Nitro's heavy use of calldata compression. Of note to end-users, we no longer make a distinction between non-zero and zero-valued calldata bytes.

Precompile address:0x000000000000000000000000000000000000006C

Method Solidity interface Go implementation Description getPricesInWeiWithAggregator(address aggregator)Interface Implementation GetPricesInWeiWithAggregator gets prices in wei when using the provided aggregator getPricesInWei() Interface Implementation GetPricesInWei gets prices in wei when using the caller's preferred aggregator getPricesInArbGasWithAggregator(address aggregator) Interface Implementation GetPricesInArbGasWithAggregator gets prices in ArbGas when using the provided aggregator getPricesInArbGas() Interface Implementation GetPricesInArbGas gets prices in ArbGas when using the caller's preferred aggregator getGasAccountingParams() Interface Implementation GetGasAccountingParams gets the rollup's speed limit, pool size, and tx gas limit getMinimumGasPrice() Interface Implementation GetMinimumGasPrice gets the minimum gas price needed for a transaction to succeed getL1BaseFeeEstimate() Interface Implementation GetL1BaseFeeEstimate gets the current estimate of the L1 basefee getL1BaseFeeEstimateInertia() Interface Implementation GetL1BaseFeeEstimateInertia gets how slowly ArbOS updates its estimate of the L1 basefee getL1RewardRate() Interface Implementation GetL1RewardRate gets the L1 pricer reward rate getL1RewardRecipient() Interface Implementation GetL1RewardRecipient gets the L1 pricer reward recipient getL1GasPriceEstimate() Interface Implementation GetL1GasPriceEstimate gets the current estimate of the L1 basefee getCurrentTxL1GasFees() Interface Implementation GetCurrentTxL1GasFees gets the fee paid to the aggregator for posting this tx getGasBacklog() Interface Implementation GetGasBacklog gets the backlogged amount of gas burnt in excess of the speed limit getPricingInertia() Interface Implementation GetPricingInertia gets the L2 basefee in response to backlogged gas getGasBacklogTolerance() Interface Implementation GetGasBacklogTolerance gets the forgivable amount of backlogged gas ArbOS will ignore when raising the basefee getL1PricingSurplus() Interface Implementation Returns the surplus of funds for L1 batch posting payments (may be negative) getPerBatchGasCharge() Interface Implementation Returns the base charge (in L1 gas) attributed to each data batch in the calldata pricer getAmortizedCostCapBips() Interface Implementation Returns the cost amortization cap in basis points getL1FeesAvailable() Interface Implementation Returns the available funds from L1 fees getL1PricingEquilibrationUnits() Interface Implementation getLastL1PricingUpdateTime() Interface Implementation getL1PricingFundsDueForRewards() Interface Implementation getL1PricingUnitsSinceUpdate() Interface Implementation getLastL1PricingSurplus() Interface Implementation

ArbInfo

ArbInfo (Interface |Implementation) provides the ability to lookup basic info about accounts and contracts.

Precompile address:0x0000000000000000000000000000000000000065

Method Solidity interface Go implementation Description getBalance(address account)Interface Implementation GetBalance retrieves an account's balance getCode(address account) Interface Implementation GetCode retrieves a contract's deployed code

ArbosTest

ArbosTest (Interface |Implementation) provides a method of burning arbitrary amounts of gas, which exists for historical reasons. In Classic,ArbosTest had additional methods only the zero address could call. These have been removed since users don't use them and calls to missing methods revert.

Precompile address:0x0000000000000000000000000000000000000069

Method Solidity interface Go implementation Description burnArbGas(uint256 gasAmount)Interface Implementation BurnArbGas unproductively burns the amount of L2 ArbGas

ArbOwner

ArbOwner (Interface |Implementation) provides owners with tools for managing the rollup. Calls by non-owners will always revert.

Most of Arbitrum Classic's owner methods have been removed since they no longer make sense in Nitro:

- What were once chain parameters are now parts of ArbOS's state, and those that remain are set at genesis.
- ArbOS upgrades happen with the rest of the system rather than being independent
- Exemptions to address aliasing are no longer offered. Exemptions were intended to support backward compatibility for contracts deployed before aliasing was introduced, but no exemptions were ever requested.

Precompile address:0x0000000000000000000000000000000000000070

Method Solidity interface Go implementation Description addChainOwner(address newOwner)Interface Implementation AddChainOwner adds account as a chain owner removeChainOwner(address ownerToRemove) Interface Implementation RemoveChainOwner removes account from the list of chain owners isChainOwner(address addr) Interface Implementation IsChainOwner checks if the account is a chain owner getAllChainOwners() Interface Implementation GetAllChainOwners retrieves the list of chain owners setL1BaseFeeEstimateInertia(uint64 inertia) Interface Implementation SetL1BaseFeeEstimateInertia sets how slowly ArbOS updates its estimate of the L1 basefee setL2BaseFee(uint256 priceInWei) Interface Implementation SetL2BaseFee sets the L2 gas price directly, bypassing the pool calculus setMinimumL2BaseFee(uint256 priceInWei) Interface Implementation SetMinimumL2BaseFee sets the minimum base fee needed for a transaction to succeed setSpeedLimit(uint64 limit) Interface Implementation SetSpeedLimit sets the computational speed limit for the chain setMaxTxGasLimit(uint64 limit) Interface Implementation SetMaxTxGasLimit sets the maximum size a tx (and block) can be setL2GasPricingInertia(uint64 sec) Interface Implementation SetL2GasPricingInertia sets the L2 gas pricing inertia setL2GasBacklogTolerance(uint64 sec) Interface Implementation SetL2GasBacklogTolerance sets the L2 gas backlog tolerance getNetworkFeeAccount() Interface Implementation GetNetworkFeeAccount gets the network fee collector getInfraFeeAccount() Interface Implementation GetInfraFeeAccount gets the infrastructure fee collector setNetworkFeeAccount(address newNetworkFeeAccount) Interface Implementation SetNetworkFeeAccount sets the network fee collector to the new network fee account setInfraFeeAccount(address newInfraFeeAccount) Interface Implementation SetInfraFeeAccount sets the infra fee collector to the new network fee account scheduleArbOSUpgrade(uint64 newVersion, uint64 timestamp) Interface Implementation ScheduleArbOSUpgrade to the requested version at the requested timestamp setL1PricingEquilibrationUnits(uint256 equilibrationUnits) Interface Implementation Sets equilibration units parameter for L1 price adjustment algorithm setL1PricingInertia(uint64 inertia) Interface Implementation Sets inertia parameter for L1 price adjustment algorithm setL1PricingRewardRecipient(address recipient) Interface Implementation Sets reward recipient address for L1 price adjustment algorithm setL1PricingRewardRate(uint64 weiPerUnit) Interface Implementation Sets reward amount for L1 price adjustment algorithm, in wei per unit setL1PricePerUnit(uint256 pricePerUnit) Interface Implementation Set how much ArbOS charges per L1 gas spent on transaction data. setPerBatchGasCharge(int64 cost) Interface Implementation Sets the base charge (in L1 gas) attributed to each data batch in the calldata pricer setBrotliCompressionLevel(uint64 level) Interface Implementation Sets the Brotli compression level used for fast compression (Available in ArbOS version 12 with default level as 1) setAmortizedCostCapBips(uint64 cap) Interface Implementation Sets the cost amortization cap in basis points releaseL1PricerSurplusFunds(uint256 maxWeiToRelease) Interface Implementation Releases surplus funds from L1PricerFundsPoolAddress for use setChainConfig(string calldata chainConfig) Interface Implementation Sets serialized chain config in ArbOS state Event Solidity interface Go implementation Description OwnerActs Interface Implementation Emitted when a successful call is made to this precompile

ArbOwnerPublic

ArbOwnerPublic (Interface |Implementation) provides non-owners with info about the current chain owners.

Precompile address:0x000000000000000000000000000000000000006b

Method Solidity interface Go implementation Description isChainOwner(address addr)Interface Implementation IsChainOwner checks if the user is a chain owner rectifyChainOwner(address ownerToRectify) Interface Implementation RectifyChainOwner checks if the account is a chain owner (Available in ArbOS version 11) getAllChainOwners() Interface Implementation GetAllChainOwners retrieves the list of chain owners getNetworkFeeAccount()Interface Implementation GetNetworkFeeAccount gets the network fee collector getInfraFeeAccount() Interface Implementation GetInfraFeeAccount gets the infrastructure fee collector getBrotliCompressionLevel() Interface Implementation GetBrotliCompressionLevel gets the current brotli compression level used for fast compression getScheduledUpgrade() Interface Implementation Returns (0, 0, nil) if no ArbOS upgrade is scheduled. Event Solidity interface Go implementation Description ChainOwnerRectified Interface Implementation Emitted when verifying a chain owner

ArbRetryableTx

ArbRetryableTx (Interface |Implementation) provides methods for managing retryables. The model has been adjusted for Nitro, most notably in terms of how retry transactions are scheduled. For more information on retryables, please seethe retryable documentation .

Precompile address:0x000000000000000000000000000000000000006E

Method Solidity interface Go implementation Description redeem(bytes32 ticketId)Interface Implementation Redeem schedules an attempt to redeem the retryable, donating all of the call's gas to the redeem attempt getLifetime() Interface Implementation GetLifetime gets the default lifetime period a retryable has at creation getTimeout(bytes32 ticketId)Interface Implementation GetTimeout gets the timestamp for when ticket will expire keepalive(bytes32 ticketId)Interface Implementation Keepalive adds one lifetime period to the ticket's expiry getBeneficiary(bytes32 ticketId)Interface Implementation GetBeneficiary gets the beneficiary of the ticket cancel(bytes32 ticketId)Interface Implementation Cancel the ticket and refund its callvalue to its beneficiary getCurrentRedeemer() Interface Implementation Gets the redeemer of the current retryable redeem attempt submitRetryable() Interface Implementation Do not call. This method represents a retryable submission to aid explorers. Calling it will always revert. Event Solidity interface Go implementation Description TicketCreated Interface Implementation Emitted when creating a retryable LifetimeExtendedInterface Implementation Emitted when extending a retryable's expiry date RedeemScheduled Interface Implementation Emitted when scheduling a retryable Canceled Interface Implementation Emitted when cancelling a retryable RedeemedInterface Implementation DEPRECATED in favour of newRedeemScheduled event after the nitro upgrade.

ArbStatistics

ArbStatistics (Interface |Implementation) provides statistics about the chain as of just before the Nitro upgrade. In Arbitrum Classic, this was how a user would get info such as the total number of accounts, but there are better ways to get that info in Nitro.

Precompile address:0x000000000000000000000000000000000000006F

Method Solidity interface Go implementation Description getStats()Interface Implementation GetStats returns the current block number and some statistics about the rollup's pre-Nitro state

ArbSys

ArbSys (Interface |Implementation) provides system-level functionality for interacting with L1 and understanding the call stack.

Precompile address:0x0000000000000000000000000000000000000064

Method Solidity interface Go implementation Description arbBlockNumber()Interface Implementation ArbBlockNumber gets the current L2 block number arbBlockHash(uint256 arbBlockNum) Interface Implementation ArbBlockHash gets the L2 block hash, if sufficiently recent arbChainID() Interface Implementation ArbChainID gets the rollup's unique chain identifier arbOSVersion() Interface Implementation ArbOSVersion gets the current ArbOS version getStorageGasAvailable()Interface Implementation GetStorageGasAvailable returns 0 since Nitro has no concept of storage gas isTopLevelCall()Interface Implementation IsTopLevelCall checks if the call is top-level (deprecated) mapL1SenderContractAddressToL2Alias(address sender, address unused) Interface Implementation MapL1SenderContractAddressToL2Alias gets the contract's L2 alias wasMyCallersAddressAliased() Interface Implementation WasMyCallersAddressAliased checks if the caller's caller was aliased myCallersAddressWithoutAliasing() Interface Implementation MyCallersAddressWithoutAliasing gets the caller's caller without any potential aliasing withdrawEth(address destination) Interface Implementation WithdrawEth send paid eth to the destination on L1 sendTxToL1(address destination, bytes calldata data) Interface Implementation SendTxToL1 sends a transaction to L1, adding it to the outbox sendMerkleTreeState() Interface Implementation SendMerkleTreeState gets the root, size, and partials of the outbox Merkle tree state (caller must be the 0 address) Event Solidity interface Go implementation Description L2ToL1Tx Interface Implementation Logs a send transaction from L2 to L1, including data for outbox proving L2ToL1Transaction Interface Implementation DEPRECATED in favour of the newL2ToL1Tx event above after the nitro upgrade SendMerkleUpdate Interface Implementation Logs a new merkle branch needed for constructing outbox proofs Edit this page Last updatedonMar 22, 2024 Previous Precompiles overview Next NodeInterface overview