

Opcode and host I/O pricing

This reference provides the latest gas and ink costs for specific WASM opcodes and host I/Os when using Stylus. For a conceptual introduction to Stylus gas and ink, see [Gas and ink \(Stylus\)](#).

ALPHA RELEASE, PUBLIC PREVIEW DOCS Stylus is currently tagged as alpha release. The code has not been audited, and should not be used in production scenarios. This documentation is currently in [public preview](#).

To provide feedback, click the Request an update button at the top of this document [join the Arbitrum Discord](#), or reach out to our team directly by completing [this form](#).

Opcode costs

The Stylus VM charges for WASM opcodes according to the following table, which was determined via a conservative statistical analysis and is expected to change as Stylus matures. Prices may fluctuate across upgrades as our analysis evolves and optimizations are made.

Hex Opcode	Ink	Gas	Notes
0x00	Unreachable	1	0.0001
0x01	Nop	1	0.0001
0x02	Block	1	0.0001
0x03	Loop	1	0.0001
0x04	If	2400	0.24
0x05	Else	1	0.0001
0x0b	End	1	0.0001
0x0c	Br	2400	0.24
0x0d	BrIf	2400	0.24
0x0e	BrTable	2400 + 325x	0.24 + 0.0325x
Cost varies with table size	0x0f	Return	1
0.0001	0x10	Call	13750
1.375	0x11	CallIndirect	13610 + 650x
1.361 + 0.065x	Cost varies with no. of args	0x1a	Drop
1	0.0001	0x1b	Select
4000	0.4	Large optimization opportunity	0x20
LocalGet	200	0.02	0x21
LocalSet	375	0.0375	0x22
LocalTee	200	0.02	0x23
GlobalGet	300	0.03	0x24
GlobalSet	990	0.099	0x28
I32Load	2200	0.22	0x29
I64Load	2750	0.275	0x2c
I32Load8S	2200	0.22	0x2d
I32Load8U	2200	0.22	0x2e
I32Load16S	2200	0.22	0x2f
I32Load16U	2200	0.22	0x30
I64Load8S	2750	0.275	0x31
I64Load8U	2750	0.275	0x32
I64Load16S	2750	0.275	0x33
I64Load16U	2750	0.275	0x34
I64Load32S	2750	0.275	0x35
I64Load32U	2750	0.275	0x36
I32Store	2400	0.24	0x37
I64Store	3100	0.31	0x3a
I32Store8	2400	0.24	0x3b
I32Store16	2400	0.24	0x3c
I64Store8	3100	0.31	0x3d
I64Store16	3100	0.31	0x3e
I64Store32	3100	0.31	0x3f
MemorySize	13500	1.35	0x40
MemoryGrow	1	0.0001	0x41
I32Const	1	0.0001	0x42
I64Const	1	0.0001	0x45
I32Eqz	570	0.057	0x46
I32Eq	570	0.057	0x47
I32Ne	570	0.057	0x48
I32LtS	570	0.057	0x49
I32LtU	570	0.057	0x4a
I32GtS	570	0.057	0x4b
I32GtU	570	0.057	0x4c
I32LeS	570	0.057	0x4d
I32LeU	570	0.057	0x4e
I32GeS	570	0.057	0x4f
I32GeU	570	0.057	0x50
I64Eqz	760	0.076	0x51
I64Eq	760	0.076	0x52
I64Ne	760	0.076	0x53
I64LtS	760	0.076	0x54
I64LtU	760	0.076	0x55
I64GtS	760	0.076	0x56
I64GtU	760	0.076	0x57
I64LeS	760	0.076	0x58
I64LeU	760	0.076	0x59
I64GeS	760	0.076	0x5a
I64GeU	760	0.076	0x67
I32Clz	750	0.075	0x68
I32Ctz	750	0.075	0x69
I32Popcnt	500	0.05	0x6a
I32Add	200	0.02	0x6b
I32Sub	200	0.02	0x6c
I32Mul	550	0.055	0x6d
I32DivS	2500	0.25	0x6e
I32DivU	2500	0.25	0x6f
I32RemS	2500	0.25	0x70
I32RemU	2500	0.25	0x71
I32And	200	0.02	0x72
I32Or	200	0.02	0x73
I32Xor	200	0.02	0x74
I32Shl	200	0.02	0x75
I32ShrS	200	0.02	0x76
I32ShrU	200	0.02	0x77
I32Rotl	200	0.02	0x78
I32Rotr	200	0.02	0x79
I64Clz	750	0.075	0x7a
I64Ctz	750	0.075	0x7b
I64Popcnt	750	0.075	0x7c
I64Add	200	0.02	0x7d
I64Sub	200	0.02	0x7e
I64Mul	550	0.055	0x7f
I64DivS	2900	0.29	0x80
I64DivU	2900	0.29	0x81
I64RemS	2900	0.29	0x82
I64RemU	2900	0.29	0x83
I64And	200	0.02	0x84
I64Or	200	0.02	0x85
I64Xor	200	0.02	0x86
I64Shl	200	0.02	0x87
I64ShrS	200	0.02	0x88
I64ShrU	200	0.02	0x89
I64Rotl	200	0.02	0x8a
I64Rotr	200	0.02	0xa7
I32Wrapl64	200	0.02	0xac
I64ExtendI32S	200	0.02	0xad
I64ExtendI32U	200	0.02	0xc0
I32Extend8S	200	0.02	0xc1
I32Extend16S	200	0.02	0xc2
I64Extend8S	200	0.02	0xc3
I64Extend16S	200	0.02	0xc4
I64Extend32S	200	0.02	0xfc0a
MemoryCopy	3100 + 125x	0.31 + 0.0125x	Cost varies with no. of bytes
MemoryFill	3100 + 125x	0.31 + 0.0125x	Cost varies with no. of bytes

Host I/O costs

Certain operations require suspending WASM execution so that the Stylus VM can perform tasks natively in the host. This costs about 1.25 gas to do. Though we'll publish a full specification later, the following table details the costs of simple operations that run in the host.

Note that the values in this table were determined via a conservative statistical analysis and are expected to change as Stylus matures. Prices may fluctuate across upgrades as our analysis evolves and optimizations are made.

Host I/O	Ink	Gas	Notes
read_args	12513 + 18287b	1.2513 + 1.8287b	b = bytes after first 32
write_result	12513 + 40423b		b = bytes after first 32
keccak	281040 + 41920w	28.104 + 4.192w	Due to a pricing mistake, keccak will soon be ~8 gas cheaper!
w = EVM words	block_basefee	22137	2.2137
block_coinbase	22137	2.2137	
block_gas_limit	12513	1.2513	
block_number	12513	1.2513	
block_timestmap	12513	1.2513	
chain_id	12513	1.2513	
contract_address	22137	2.2137	
evm_gas_left	12513	1.2513	
evm_ink_left	12513	1.2513	
msg_reentrant	12513	1.2513	
msg_sender	22137	2.2137	
msg_value	22137	2.2137	
return_data_size	12513	1.2513	
tx_ink_price	12513	1.2513	
tx_gas_price	22137	2.2137	
tx_origin	22137	2.2137	
console_log_text	0	0	debug-only
console_log	0	0	debug-only
console_tee	0	0	debug-only
null_host	0	0	debug-only

For some opcodes, the value in the table may be larger than its EVM equivalent. In the vast majority of real-world cases, the Stylus VM will still perform better. For example, getting the rawmsg::value costs 0.2137 more gas in Stylus than it does in the EVM. But the moment one takes that value and does anything with it, Stylus becomes much cheaper. Recall that a single EVMMADD instruction costs 3 gas, during which 150 WASMMADD instructions may run.

See also

- [Gas and ink \(Stylus\)](#)

- : A conceptual introduction to the "gas" and "ink" primitives[Edit this page](#) Last updated on Mar 19, 2024 [Previous Gas and ink in Stylus](#) [Next Stylus Rust SDK overview](#)