Name	Description
High level constructs	
$\boldsymbol{\sigma}_t$	World-state at time t.
T	An Ethereum transaction
T_0, T_1, ...	Individual transactions within a block
B	A block: $B \equiv (..., (T_0, T_1, ...))$
Υ	The Ethereum state transition function: $\boldsymbol{\sigma}_{t+1} \equiv \Upsilon(\boldsymbol{\sigma}_t, T)$
Ω	The block-finalisation state transition function (pays out the mining reward).
Π	The block-level state-transition function: $\Pi(\boldsymbol{\sigma}, B) \equiv \Omega(B, \Upsilon(\Upsilon(\boldsymbol{\sigma}, T_0), T_1)...)$
$\boldsymbol{\mu}$	Machine-state tuple, $(g, pc, \mathbf{m}, i, \mathbf{s})$, which are gas, program counter, memory, memory size, stack.
World state	
$\boldsymbol{\sigma}[a]_n$	The nonce of account a.
$\boldsymbol{\sigma}[a]_b$	The balance of account a.
$\boldsymbol{\sigma}[a]_s$	A 256-bit hash of the root node of a Merkle Patricia tree that encodes the storage contents of account a. Note that $\texttt{TRIE}\big(L_I^*(\boldsymbol{\sigma}[a]_\mathbf{s})\big) \equiv \boldsymbol{\sigma}[a]_s$
$\boldsymbol{\sigma}[a]_c$	The hash of the EVM code of account a.
Machine state	
$\boldsymbol{\mu}_g$	The gas available.
$\boldsymbol{\mu}_{pc}$	The program counter.
$\boldsymbol{\mu}_\mathbf{m}$	The memory contents.
$\boldsymbol{\mu}_i$	The number of memory words allocated.
$\boldsymbol{\mu}_\mathbf{s}$	The stack.
$\boldsymbol{\mu}_\mathbf{s}[n]$	Item at stack depth n.
Substate	
A	A Transaction substate during execution: $\equiv (A_\mathbf{s}, A_\mathbf{l}, A_r)$.
$A_\mathbf{s}$	The self-destruct set.
$A_\mathbf{l}$	The log series.
A_r	The gas refund balance. Can partially offset execution costs.
A^0	The empty substate: $A^0 \equiv (\varnothing, (), 0)$.
Execution environment	
I_a	The address of the account which owns the code that is executing.
I_o	The sender address of the transaction that originated this execution.
I_p	The price of gas in the transaction that originated this execution.
$I_\mathbf{d}$	The byte array that is the input data to this execution; if the execution agent is a transaction, this would be the transaction data.
I_s	The address of the account which caused the code to be executing; if the execution agent is a transaction, this would be the transaction sender.
I_v	The value, in Wei, passed to this account as part of the same procedure as execution; if the execution agent is a transaction, this would be the transaction value.
$I_\mathbf{b}$	The byte array that is the machine code to be executed.
I_H	The block header of the present block.
I_e	The depth of the present message-call or contract-creation (i.e. the number of CALLs or CREATEs being executed at present).
Blocks	
B	A block: $B \equiv (B_H, B_\mathbf{T}, B_\mathbf{U})$.
B_H	The block's header.
$B_\mathbf{T}$	The block's transactions.
$B_\mathbf{U}$	Headers of ommer/uncle blocks of this block.
$B_\mathbf{R}$	Transaction receipts.

Name	Description
Block header	
H_p	**parentHash**: The Keccak 256-bit hash of the parent block's header, in its entirety.
H_o	**ommersHash** The Keccak 256-bit hash of the ommers list portion of this block.
H_c	**beneficiary** The 160-bit address to which all fees collected from the successful mining of this block be transferred.
H_r	**stateRoot** The Keccak 256-bit hash of the root node of the state trie, after all transactions are executed and finalisations applied.
H_t	**transactionsRoot** The Keccak 256-bit hash of the root node of the trie structure populated with each transaction in the transactions list portion of the block.
H_e	**receiptsRoot** The Keccak 256-bit hash of the root node of the trie structure populated with the receipts of each transaction in the transactions list portion of the block.
H_b	**logsBloom** The Bloom filter composed from indexable information (logger address and log topics) contained in each log entry from the receipt of each transaction in the transactions list.
H_d	**difficulty** A scalar value corresponding to the difficulty level of this block.
H_i	**number** A scalar value equal to the number of ancestor blocks. The genesis block has a number of zero.
H_l	**gasLimit** A scalar value equal to the current limit of gas expenditure per block.
H_g	**gasUsed** A scalar value equal to the total gas used in transactions in this block.
H_s	**timestamp** A scalar value equal to the reasonable output of Unix's time() at this block's inception.
H_x	**extraData** An arbitrary byte array containing data relevant to this block. This must be 32 bytes or fewer.
H_m	**mixHash** A 256-bit hash which proves combined with the nonce that a sufficient amount of computation has been carried out on this block.
H_n	**nonce** A 64-bit hash which proves combined with the mix-hash that a sufficient amount of computation has been carried out on this block.
$V(H)$	The block header validity function.
Transactions	
T_n	Transaction nonce.
T_p	Gas price for the transaction.
T_g	The maximum gas for a transaction.
T_t	The "to" address for the transaction.
T_v	The value to be transferred by the transaction.
T_w, T_r, T_s	The v, r, s values of the transaction signature.
$T_{\mathbf{i}}$	EVM-code for account initialisation (i.e. contract deployment).
$T_{\mathbf{d}}$	Input data of a message call.
$S(T)$	The sender address of a transaction.
Transaction Receipt	
R	A transaction receipt: $R \equiv (R_{\boldsymbol{\sigma}}, R_u, R_b, R_{\mathbf{l}})$
$R_{\boldsymbol{\sigma}}$	The post-transaction state.
R_u	The cumulative gas used so far in the block.
R_b	The bloom filter composed from the information in the transaction logs.
$R_{\mathbf{l}}$	The log entries created by the transaction, $(O_0, O_1, ...)$.
O	A log entry: $O \equiv (O_a, (O_{\mathbf{t}0}, O_{\mathbf{t}1}, ...), O_{\mathbf{d}})$.
O_a	The logger's address.
$O_{\mathbf{t}}$	A 32-byte log topic.
$O_{\mathbf{d}}$	The log data for this entry.
Misc functions	
$\ell(\mathbf{x})$	The last item in sequence $\mathbf{x}$: $\ell(\mathbf{x}) \equiv \mathbf{x}[\lVert\mathbf{x}\rVert - 1]$
$M(s, f, l)$	Memory expansion function. s is the current top of memory; f is the start of writing; l is the number of bytes to be written.
$L(n)$	The "all but one 64th" function: $L(n) \equiv n - \lfloor n/64 \rfloor$.
$L_I\big((k, v)\big)$	Representation of key–value pairs in the trie: $L_I\big((k, v)\big) \equiv \big(\mathtt{KEC}(k), \mathtt{RLP}(v)\big)$

Name	Description
$S(T)$	Sender function—recovers the sender address from the transaction: $S(T) \equiv \mathcal{B}_{96..255}\big(\texttt{KEC}\big(\texttt{ECDSARECOVER}(h(T), T_w, T_r, T_s)\big)\big)$.
Conventions	
$\square$	A placeholder in the following; the 'input' value.
$\square'$	A modified and utilisable value.
$\square^*$, $\square^{**}$	Intermediate values.
f^*	An element-wise version of a function f that maps between sequences.
Todo	
$\mathbf{b}$	The code associated with an account. $\texttt{KEC}(\mathbf{b}) = \boldsymbol{\sigma}[a]_c$.
L_R	Eqn 19
L_S	Eqn 19
L_T	Eqn 13
$V(H)$	Block header validity Eqn 48.
Ξ	Code execution function.
etc.	