VM-specific v1.3.0 opcodes simulation (verbatim)

NOTES:

- changed META - it can be used for MSIZE simulation
- setting ergs per pubdata is done by separate opcode now (not part of `near_call`)
- incrementing TX counter is done by separate opcode now (not part of `far_call`)

Our VM has some opcodes that are not expressible in Solidity, but we can simulate them on the Yul compiler level by using "verbatim_*" instruction.

For some simulations below we assume that there exist a hidden global pseudo-variable called `ACTIVE_PTR` for manipulations, since one can not easily load pointer value into Solidity's variable.

Simulated opcode	Verbatim signature	Arg 1	Arg 2	Arg 3	Arg 4
`to_l1(is_first, in0, in1`	`verbatim_3i_0o("to_l1", ...)`	if_first (bool)	in0 (u256)	in1 (u256)	
`code_source`	`verbatim_0i_1o("code_source", ...)`				
`precompile(in0, ergs_to_burn, out0)`	`verbatim_2i_1o("precompile", ...)`	in0 (u256)	ergs_to_burn (u32)		
`meta`	`verbatim_0i_1o("meta", ...)`				
`mimic_call(to, abi_data, implicit r3 = who to mimic)`	`verbatim_3i_1o("mimic_call", ...)`	who_to_call	who_to_mimic	abi_data	
`system_mimic_call(to, abi_data, implicit r3, r4, r5 = who to mimic)`	`verbatim_7i_1o("system_mimic_call", ...)`	who_to_call	who_to_mimic	abi_data	value_
`mimic_call_byref`	`verbatim_2i_1o("mimic_call_byref", ...)`	who_to_call	who_to_mimic		
`system_mimic_call_byref`	`verbatim_6i_1o("system_mimic_call_byref", ...)`	who_to_call	who_to_mimic	value_to_put_into_r3	value_
`raw_call`	`verbatim_4i_1o("raw[_<type>]_call", ...)` type = '' | static | delegate	who_to_call	abi_data (CAN be with "to system = true")	output_offset	output

`raw_call_byref`	`verbatim_3i_1o("raw[_<type>]_call_byref", ...)` type = ‘’ | static | delegate	who_to_call	output_offset	output_length	
`system_call`	`verbatim_6i_1o("system[_<type>]_call", ...)` type = ‘’ | static | delegate	who_to_call	abi_data (MUST have “to system” set)	value_to_put_into_r3	value_
`system_call_byref`	`verbatim_5o_1o("system[_<type>]_call_byref", ...)` type = ‘’ | static | delegate	who_to_call	value_to_put_into_r3	value_to_put_into_r4	value_
`set_context_u128`	`verbatim_1i_0o("set_context_u128", ...)`	value			
`set_pubdata_price`	`verbatim_1i_0o("set_pubdata_price", ...)`	price			
`increment_tx_counter`	`verbatim_0i_0o("increment_tx_counter", ...)`				
`event_initialize`	`verbatim_2i_0o("event_initialize", ...)`	in0 (u256)	in1 (u256)		
`event_write`	`verbatim_2i_0o("event_write", ...)`	in0 (u256)	in1 (u256)		
`load_calldata_into_active_ptr`	`verbatim_0i_0o("calldata_ptr_to_active", ...)`				
`load_returndata_into_active_ptr`	`verbatim_0i_0o("return_data_ptr_to_active", ...)`				
`ptr_add_into_active`	`verbatim_1i_0o("active_ptr_add_assign", ...)`	offset			
`ptr_shrink_into_active`	`verbatim_1i_0o("active_ptr_shrink_assign", ...)`	offset			
`ptr_pack_into_active`	`verbatim_1i_0o("active_ptr_pack_assign", ...)`	data			
`multiplication_high`	`verbatim_2i_1o("mul_high", ...)`	operand_1	operand_2		
`get_global`	`verbatim_0i_1o("get_global::<name>", ...)` (`<name>` from the table below)	index			
`throw`	`verbatim_i0_o0("throw", ...)`				

List of globals (zero-enumerated in the order below for purposes of `get_global`):

- `ptr_calldata` - one passed in `r1` on `far_call` to the callee (save in very first instructions on entry)
- `call_flags` - one passed in `r2` on `far_call` to the callee (save in very first instructions on entry)
- `extra_abi_data_{N}` - ones passed in `r3-r12` on `far_call` to the callee (save in very first instructions on entry), `0 <= N <= 9`
- `ptr_return_data` - one passed in `r1` on return from `far_call` back to the caller (save in very first instruction in the corresponding branch!)

Requirements for calling system contracts

By default, all system contracts at addresses 0x80XX require that the call was done via system call (i.e. `call_flags&2 != 0` .

Exceptions:

- BOOTLOADER_FORMAL address as the users need to be able to send money there.

Meaning of ABI params:

- MSG_VALUE_SIMULATOR: `extra_abi_data_1 = value || whether_the_call_is_system`, where || denotes the concatenation, value should occupy first 128 bits, while `whether_the_call_is_system` is a 1-bit flag that denotes whether the call should be a system call.

  `extra_abi_data_2` is the address of the callee.
- No meaning for the rest