Please find some design patterns (below) which use modulo arithmetic & exponentiation to access and store multiple values inside a single uint256 slot.

Pseudo code 1 - read one or more digits from a uint256 variable

((_value % (10 ** _position)) - (_value % (10 ** (_position - _size)))) / (10 ** (_position - _size));

Solidity

This has been implemented in Solidity as a pure function

contract uintTool { function getInt(uint256 _value, uint256 _position, uint256 _size) public pure returns(uint256){ uint256 a = ((_value % (10 ** _position)) - (_value % (10 ** (_position - _size)))) / (10 ** (_position - _size)); return a; } }

Implementation

It has also been deployed on the Ropsten test network at the following contract address if you would like to interact with it.

0x33539788196abf0155e74b80f4d0916e020226f7

Usage

The above getInt function will return a value of 1234567

if passed the following arguments

_value 99999991234567999999999999999999999999999999999999999999999999999999999999999999 _position 70 _size 7

Thank you

Thank you for having a super quick chat with me about this at EDCON2019@vbuterin, I went ahead and wrote this getInt function in Vyper straight after our quick chat. I appreciate that this is something that you have already thought of/tried; I would like to refine the general idea and formalize it as an Informational EIP.

For more information, and many more examples, please seethis GitHub repository