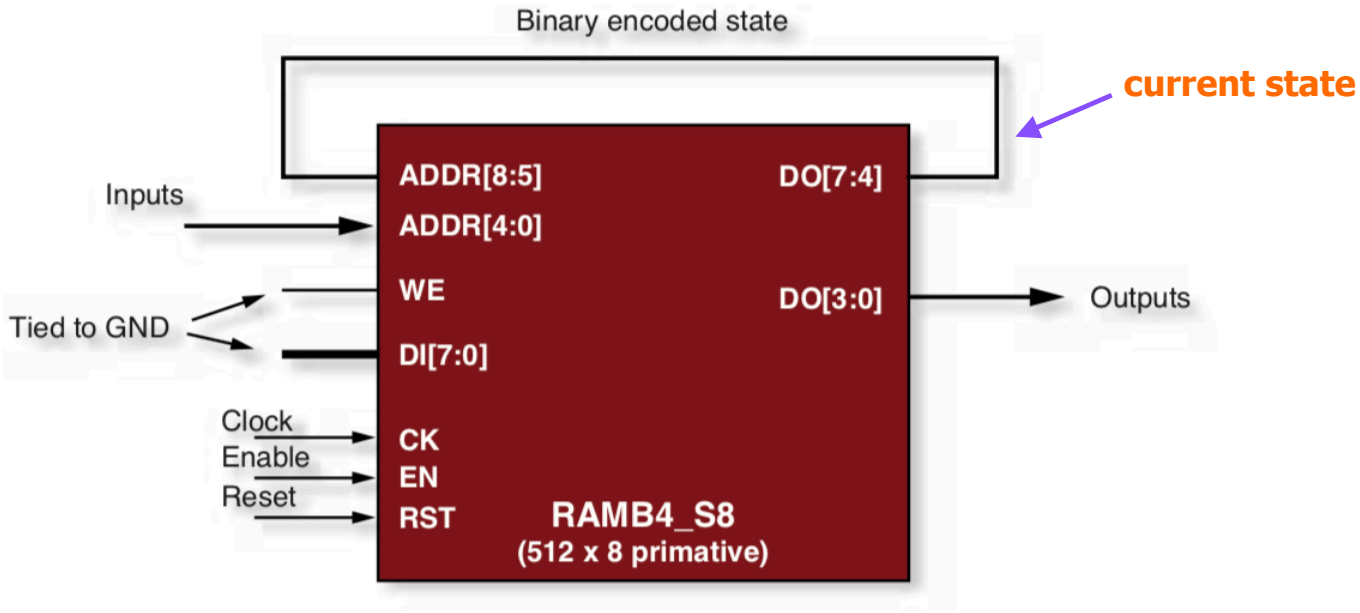
Finite State Machines in Block RAM

The idea of using initialized BRAM as ROM suggests FSMs can be implemented in BRAM as well. The idea is shown below:



Using fully synchronous RAM blocks for FSM implementation

Here a BRAM has a 9-bit address bus and an 8-bit data output. 4-bits of the output are a binary encoding of the current state and the remaining 4 outputs are FSM outputs. The EN can be used to activate/deactivate the FSM and the RST to put it into its initial state

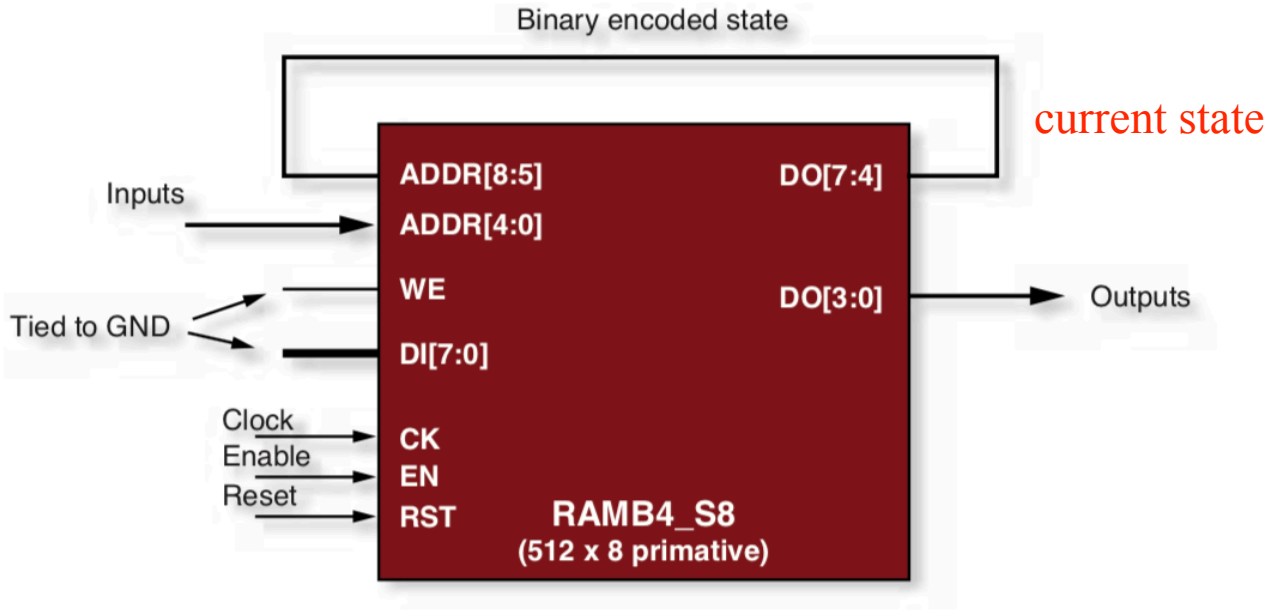
The 9-bit address bus is partitioned into two parts. The 4 MSBs fed back is the current state and the 5 LSBs are FSM inputs.

Thus, the FSM implemented can have up to 16 states, 5 inputs and 4 outputs.

As an example, consider a Moore FSM is currently in state 2 (4'b0010) and the next 5-bit input set is 5'b01001. Then the 9-bit address is 001001001 = 049H. Suppose

mem[001001001] = mem[049] = D5H.

Then the next state would be DH = 1101 = 13 and the corresponding output would 0101. (No transition or output change until the next clock.)



Using fully synchronous RAM blocks for FSM implementation