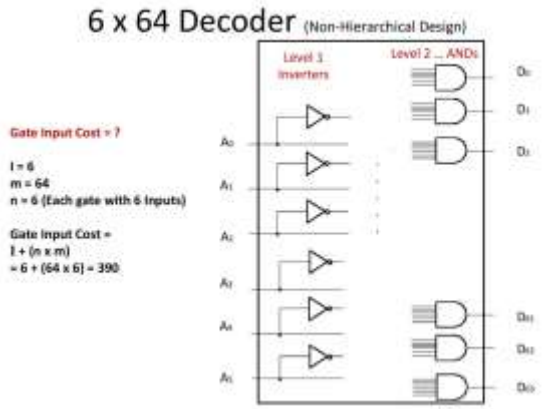
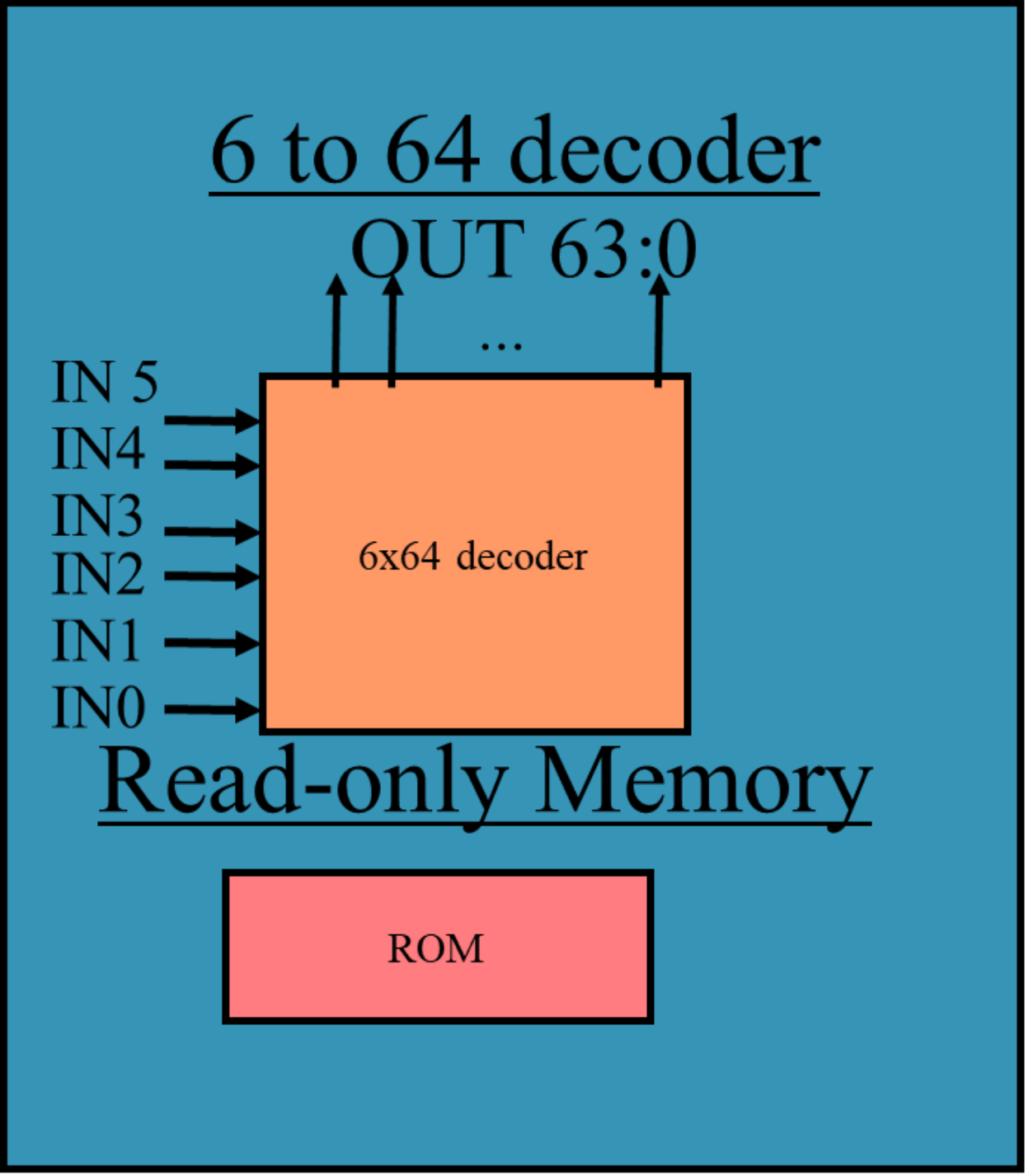
6x64 decoder

By: Steven Yeung
Instructor: Dr. Paul West

Why is the decoder important?



- The decoder is a device that takes a combination of binary lines.
- This in turn will turn on a singular output signal in a 2x4, 3x8, 4x16, 5x16, 6x64 decoder. This is depicted by the number 1. The singular number 1 represents the wire that is on. The other 0s are wires that are off.
- The first six lines depict will depict one of the sixty-four lines are going to activate.
- This can be significant because they are important logic blocks for many digital applications
  - For example. A soda machine. When a person selects an item, the item they choose will be dispensed. Why? All because of a decoder. With a decoder, the machine will know which wire to turn on and which wires to keep off to dispense the correct item.



IN	OUT
6543210	6543210
0000000	0000001
0000001	0000010
0000010	0000100
0000011	0001000

etc.

ROM just stores preset data in each location.
Give address to access data

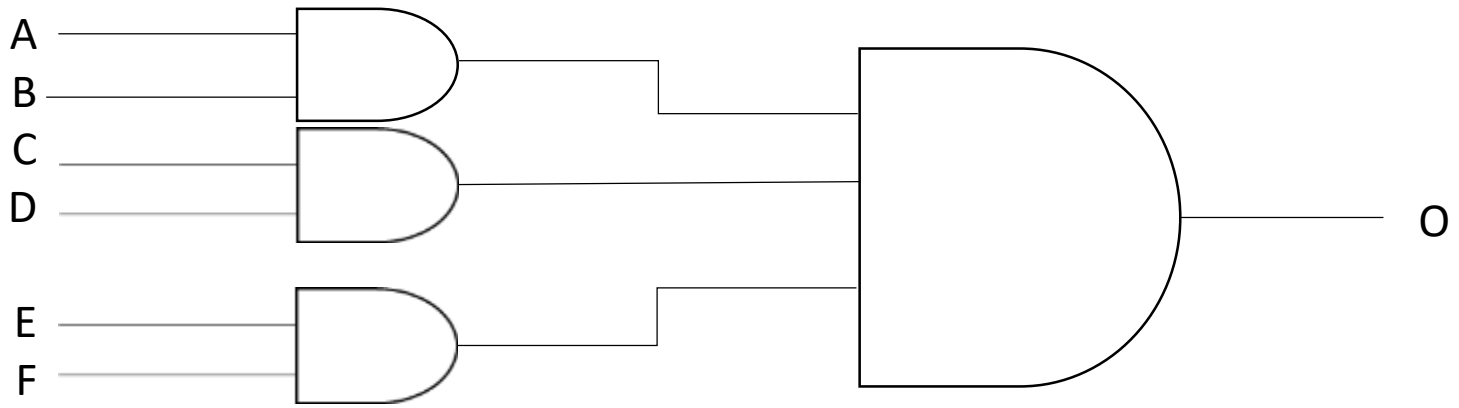
Instructions that are supported

- It can do the ADD instruction
- It can do the AND instruction
- It can do the MOV instruction
- It can do the CMP instruction
- It can do the JA instruction
- It can do the JALR instruction
- It can do the CMOV instruction

Results

```
111100 -> 00010000000000000000000000000000000000000000000000000000000000000000
000000 -> 00000000000000000000000000000000000000000000000000000000000000000001
000100 -> 00000000000000000000000000000000000000000000000000000000000000010000
011000 -> 00000000000000000000000000000000000000010000000000000000000000000000
001011 -> 00000000000000000000000000000000000000000000000000000100000000000000
001100 -> 00000000000000000000000000000000000000000000000000000100000000000000
111100 -> 00010000000000000000000000000000000000000000000000000000000000000000
001111 -> 00000000000000000000000000000000000000000000000000000100000000000000
```

Delay Calculation



- The diagram above was used to implement a 6x64 decoder.
- Since it was 6x64, it is implied that there are 6 inputs with 3 and gates.
- The 3 and gates have a total of 2 gate delays because they in parallel.
- The output gate has 2 gate delays because it is a separate and gate that is outputting the 64-bit result.
- The 6x64 decoder has a 4 gate delays in every round of output

Problems that were encountered

- Used the structural mode of coding the decoder. An issue that popped up was finding a diagram to code from.
- Another issue was finding out that there were several not gates that acted as inverters for the 6x64 decoder. These not gates serve as signal “blockers”. Figuring out where to place them during implementation was a challenge.
- There was a pattern like binary because your setup had to account for any of the sixty-four outputs.

Implementation (Test cases!)

```
// Test code uses behavioral, only use behavioral for tests (unless othewaise directed.)
module simple_test();

reg [5:0]I;

wire [63:0]O;

// By convention we can the device DUT = Device Under Test
decoder6x64 DUT(I, O);

initial
begin
    /*I <= 64'b0000000000000000000000000000000000000000000000000000000000000100;
    #7 // We MUST wait some time for the circuit to saturate!
    $display("%b -> %b", I, O); */

    I <= 6'b000000;
    #7 // We MUST wait some time for the circuit to saturate!
    $display("%b -> %b", I, O);


    I <= 6'b000100;
    #7 // We MUST wait some time for the circuit to saturate!
    $display("%b -> %b", I, O);

    I <= 6'b011000;
    #7 // We MUST wait some time for the circuit to saturate!
    $display("%b -> %b", I, O);


    I <= 6'b001011;
    #7 // We MUST wait some time for the circuit to saturate!
    $display("%b -> %b", I, O);


    I <= 6'b001100;
    #7 // We MUST wait some time for the circuit to saturate!
    $display("%b -> %b", I, O);

    I <= 6'b111100;
    #7 // We MUST wait some time for the circuit to saturate!
    $display("%b -> %b", I, O);

    I <= 6'b001111;
    #7 // We MUST wait some time for the circuit to saturate!
    $display("%b -> %b", I, O);

```