Combinational Logic Modules

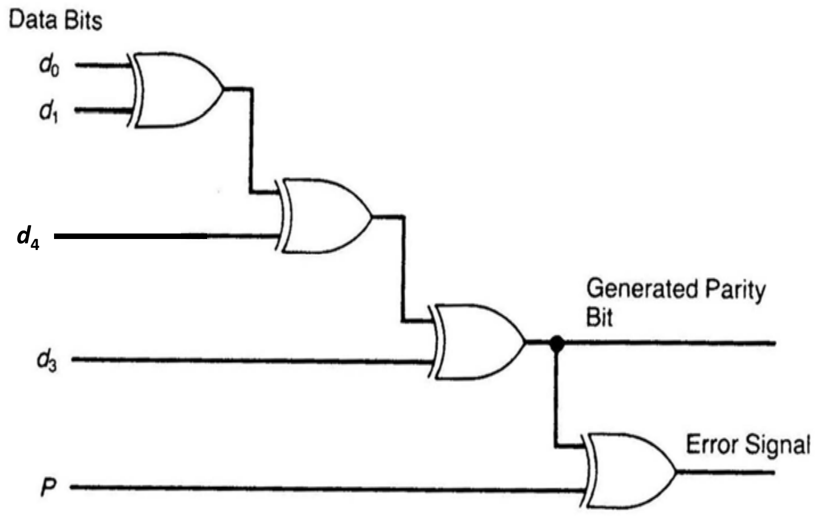
Source:
Digital System Designs and Practices Using Verilog HDL and FPGAs
@ 2008, John Wiley
Partial from Digital IC Design, By Pei-Yin Chen

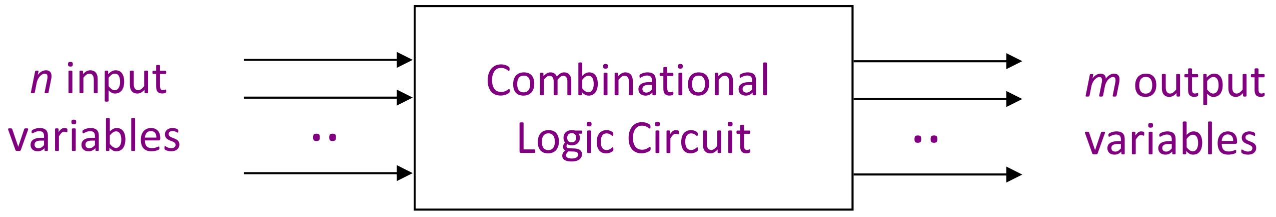
Outline

- Combinational Circuit vs. Sequential Circuit
- Parameter
- Basic Combinational Logic Modules
- Options for Modeling Combinational Logic

Combinational Circuit

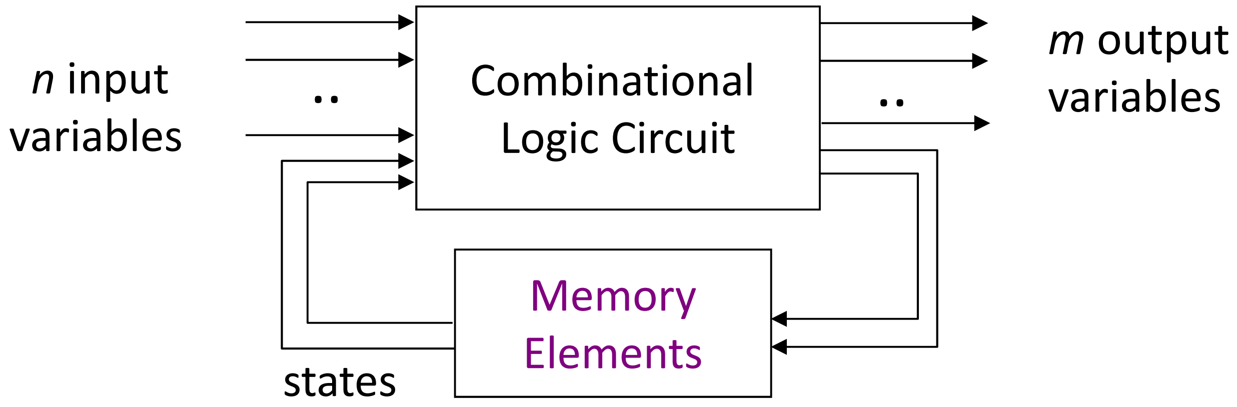
- A combinational circuit: Outputs at any time are determined directly from the present combination of inputs without regard to previous inputs.





Sequential Circuit

A sequential circuit is a system whose outputs at any time are determined from the present combination of inputs and the previous states.



- Sequential components contain memory elements

  Ex: Ring counter that starts the answering machine after 4 rings

Parameter

- Parameter declaration

  parameter identifier = constant_expression ,

- You can use a parameter anywhere that you can use a literal.

```
module mod(ina, inb, out);
……

parameter m1=8;
……

wire [m1:0] w1;
……

endmodule
```

w1 can be set as a (n+1)-bit wire if we change m1 to n

(i.e., m1=10 $\Rightarrow$ w1 becomes a 11-bit wire

m1=4 $\Rightarrow$ w1 becomes a 5-bit wire)

Parameterized Design (1/2)

```
module test (a, b, c);
parameter width = 8;
input  [width - 1 : 0] a, b;
output [width - 1 : 0] c;

assign c = a & b;

endmodule
```

```
module PARAM(A, B, C);
input  [3:0] A, B;
output [3:0] C;
wire f;

or        o1(f, A, B);
test#(4) u1(A, f, C);

endmodule
```

Override the value of width when the test module is instantiated

Save the file as PARAM.v and compile (synthesis) it

➡ the width value become 4

Parameterized Design (2/2)

```
module test_2(A, B, C, D);
parameter width  = 8;
parameter height = 8;
parameter length = 8;

input  [width - 1 : 0]  A;
input  [height - 1 : 0] B;
input  [length- 1 : 0]  C;
output [width : 0]      D;

assign D = A + B + C;

endmodule
```

```
module PARAM_1(A, B, C, D);
input  [4 : 0] A;
input  [3 : 0] B;
input  [3 : 0] C;
output [5 : 0] D;

test_2#(5, 4, 4) u1(A, B, C, D);

endmodule
```

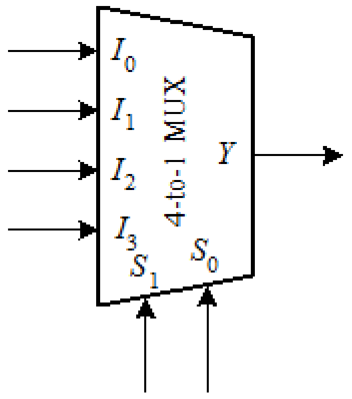
Override those values of many parameters when the test_2 module is instantiated (width = 5; height = 4; length = 4)

Basic Combinational Logic Modules

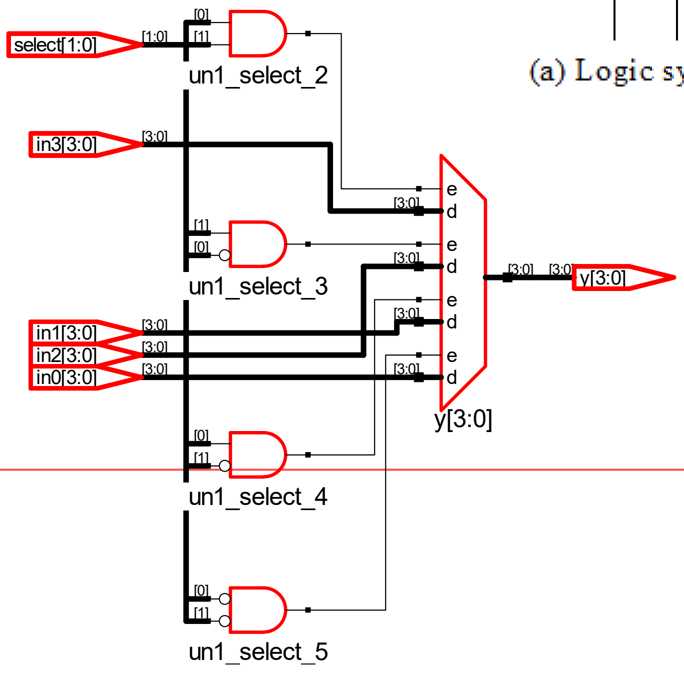
- Commonly used combinational logic modules:
  - Multiplexer
  - Decoder
  - Encoder
  - Comparator
  - Adder
  - Subtracter
  - Multiplier
  - Arithmetic Logic Unit(ALU)

An *n*-bit 4-to-1 Multiplexer Example

```
// an N-bit 4-to-1 multiplexer using conditional operator.
module mux_nbit_4to1(select, in3, in2, in1, in0, y);
parameter N = 4; // define the width of 4-to-1 multiplexer
input [1:0] select;
input [N-1:0] in3, in2, in1, in0;
output [N-1:0] y;
// the body of the N-bit 4-to-1 multiplexer
assign y = select[1] ?
                (select[0] ? in3 : in2) :
                (select[0] ? in1 : in0) ;
endmodule
```

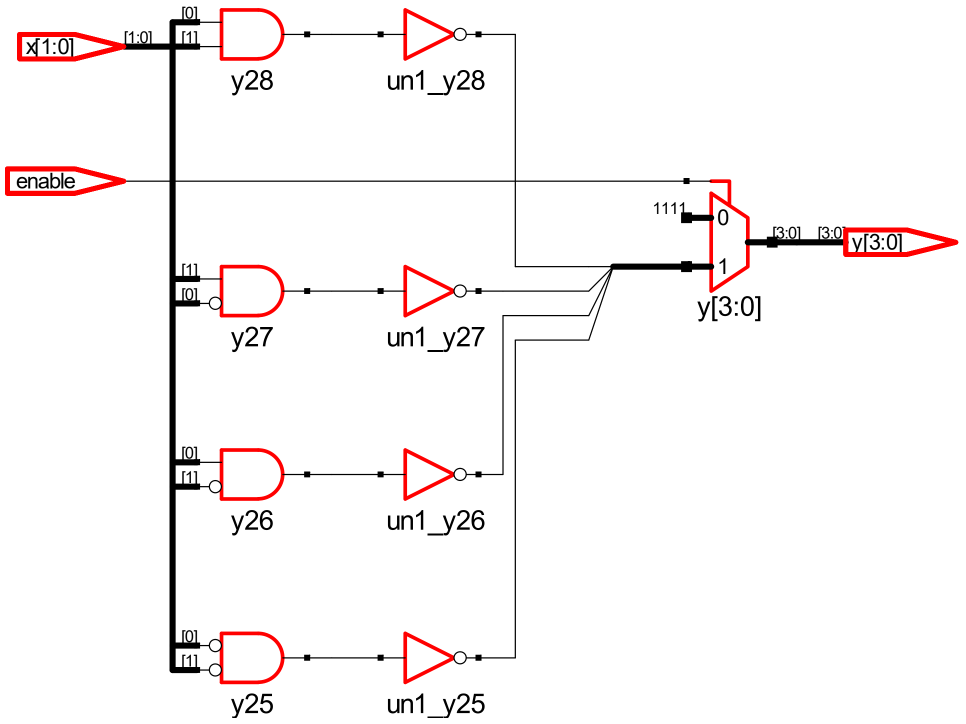


(a) Logic symbol



A 2-to-4 Decoder Example

```
// a 2-to-4 decoder with active low output
module decoder_2to4_low(x,enable,y);
input  [1:0] x;
input  enable;
output reg [3:0] y;
// the body of the 2-to-4 decoder
always @(x or enable)
    if (enable) y = 4'b1111; else
        case (x)
            2'b00 : y = 4'b1110;
            2'b01 : y = 4'b1101;
            2'b10 : y = 4'b1011;
            2'b11 : y = 4'b0111;
           default : y = 4'b1111;
        endcase
endmodule
```



Options for Modeling Combinational Logic

- Options for modeling combinational logic:
  - Verilog HDL primitives
    - Ex: and (a, b, c)
  - Continuous assignment
    - Ex: assign out = i1& i2;
  - Behavioral statement
    - always statement