جامعة بيرزيت

BIRZEIT UNIVERSITY

ANSWER BOOKLET

Student: Digital Number 8

Course: Department: Number:

Division: Instructor:

Date: Day Month Year

For Instructor's Use

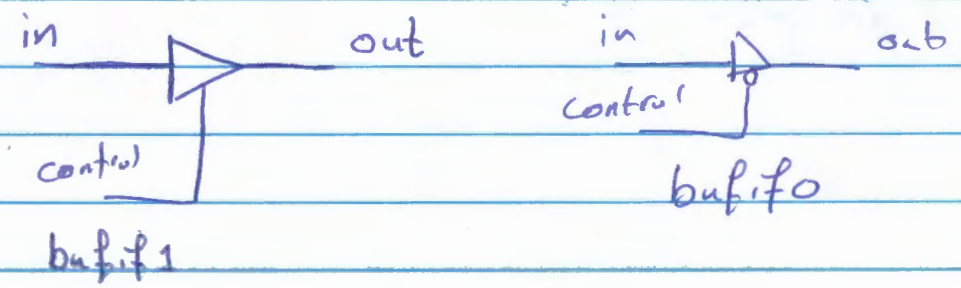
Question	Grade
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
Total	

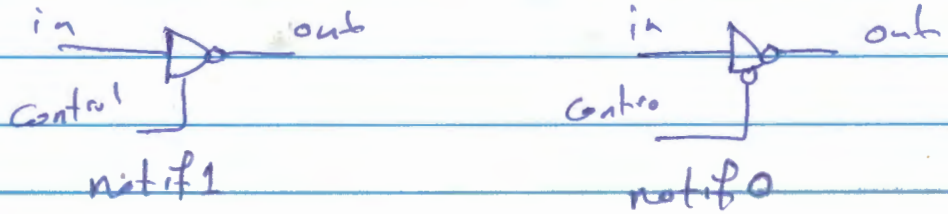
Binary adder

⇒ gate level HDL Ex 4.2

⇒ data flow HDL Ex 4.4

3 state Gates

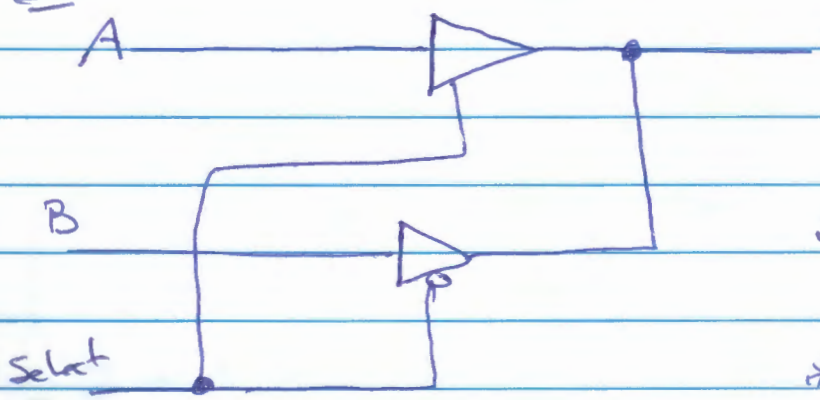




e.g bufif1 (out, A, control); ⇒ out = A if c=1

notif1 (Y, B, enable); y = B' if enable=0

Ex.



if select = 1

out = A

if select = 0

out = B

* mux (tri) in gate level page 152

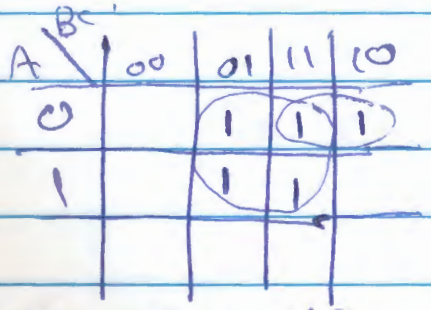
* mux with data flow HDL example 4-6

* comparator with data flows
HDL Example 4-5

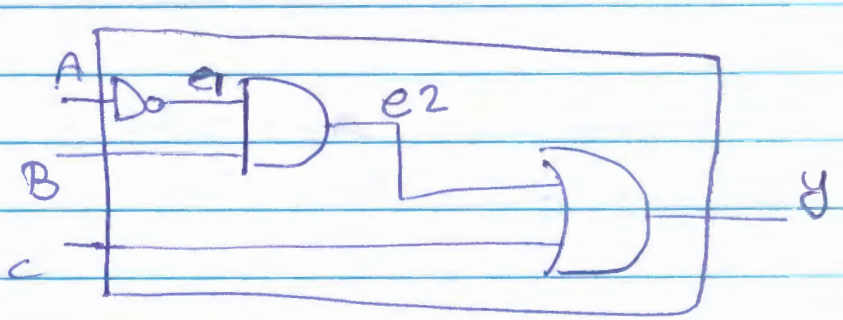
Example: Design a logic circuit with 3 inputs A, B, and C and one output that indicates whether the input number is prime or not.
Write an gate level HDL to describe this circuit.
Write a test bench that tests the circuit for all possible input combinations and generates a log file of the results.

Solution

A	B	C	P
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1



$P = C + A'B$



```
module prime(A,B,C,y);
  input A, B, C;
  output y;
  wire e1,e2;
  not g1(e1, A);
  and g2(e2,e1,B);
  or g3(y,e2,c);
endmodule

module test
    reg [2:0] D;
    wire out;
    prime pr1(D[2],D[1], D[0],out);
    initial
      begin
        D = 3'b000;
      repeat(8)
      #10 D = D + 1'b1
      end
  initial
    $monitor("ABC=%b y=%b", D, out);
  endmodule
```

(*) S = A+B

⇒ assign S = A+B; continuous assignment
data flow

always @ (A or B)
S = A+B; procedural assignment
behavioural description

(*) Verilog HDL operators

1) Arithmetic +, −, *, /, %

2) Logic (bitwise or reduction operation)
~, &, |, ^

(3) Logical : ! && ||

e.g A = 101 B = 011 C = 000

A & B = 001

A && B = (1) && (1) = 1

A && C = (1) && (0) = 0

(4) shift >> shift right
<< " left
{ , } concatenation
eg { C, S } = X + Y + Z

(5) Relational >, <, ==, !=, >=, <=

eg R = 11010
R = R>>1 → R =

⊛ Loop Statements

- Verilog HDL has four types of loops that allow procedural statements to be executed repeatedly: repeat, forever, while, for.

- All looping statements must appear inside an initial or always block.

Ex:

```
integer count;
initial
  begin
    count = 0;
    while (count < 100) // forever // repeat(20)
    #5  count = count + 2
  end
```

Ex:

```
for (i=0; i<8; i=i+1)
    procedural statement
```

HDL Example 8-1

Logic Synthesis
- ASIC, PLD and FPGA

assign for combinational circuits

assign y = A | B (gate circuit)
" y = A + B (full adder)
" y = A - B (" " with Xor gates)
" y = s ? I1 : I0 (2x1 mux)

- always : for sequential or combinational circuits

e.g always @ (I1 or I0 or s)
if (s) y = I1 ;
else y = I0 ;
(2x1 mux)

- case statement (large mux).