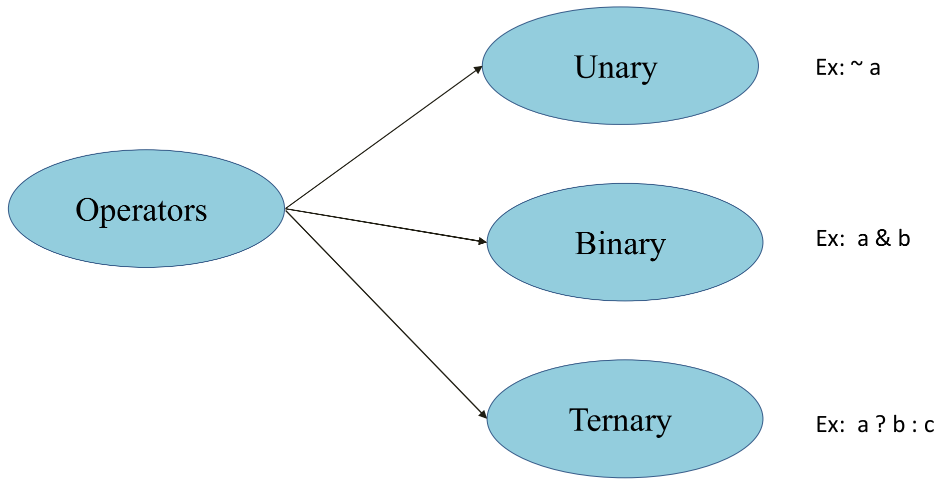
Indian Institute of Information Technology Vadodara

CS/IT 429

Operators

Dr. Yash Agrawal
Visiting Faculty, IIIT Vadodara
Associate Professor, DA-IICT Gandhinagar

Introduction



Operators

- Arithmetic
- Bitwise
- Logical
- Reduction
- Relational and Equality
- Shift
- Concatenate and Replication
- Conditional

Operators

Arithmetic Operator

- Addition (+)
- Subtraction (-)
- Multiplication (*)
- Division (/)
- Modulus (%)
- Power/Exponent (**)

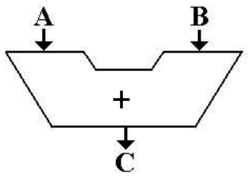
If any input to arithmetic operator is x, it leads to output as x

Operators

Arithmetic Operator

Example

1. To Check: Whether the output is even or odd
   - A and B can be considered as 3 bits



2. Check: $(a + b)^3 = a^3 + b^3 + 3a^2b + 3ab^2$
   - Use Only Arithmetic Operators
   - a and b can be considered as 2 bits

Operators

Bitwise Operator

- Unary NOT (~)
- Binary AND (&)
- Binary OR (|)
- Binary XOR (^)
- Binary XNOR (~^ or ^~)

- Bitwise operator performs bit-by-bit operation on two operands.

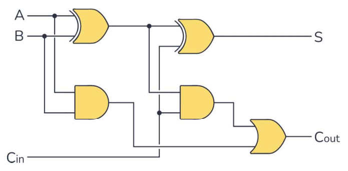
- z is treated as x in bitwise operation.

Operators

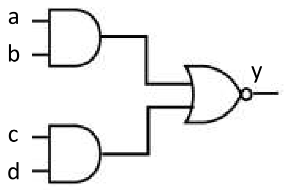
Bitwise Operator

Example

Implement Full Adder using bitwise operator
- Inputs can be considered as single bit



Implement using Bit-wise Operator
- Inputs can be considered as single bit



Operators

Logical Operator

- Logical NOT (!)
- Logical AND (&&)
- Logical OR (||)

- Only one-bit output.

Operators

Logical Operator

Example

1) If a = 5`b 1 1 0 1 1
   b = 4`b 0 0 1 0

If a && b	=	1 && 1 = 1
a || b	=	1 || 1 = 1
! a	=	not (logical 1) = 0
! b	=	not (logical 1) = 0

2) If a = 5`b 1 1 0 1 1
   b = 4`b 0 0 0 0

If a && b	=	1 && 0 = 0
a || b	=	1 || 0 = 1
! a	=	not (logical 1) = 0
! b	=	not (logical 0) = 1

Operators

Logical Operator

Example

3) If a = 4`d 3
   b = 5`d 0

If a && b	=	1 && 0 = 0
a || b	=	1 || 0 = 1
! a	=	not (logical 1) = 0
! b	=	not (logical 0) = 1

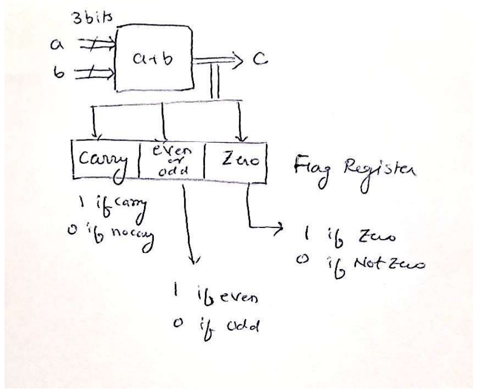
4) If a = 2`b 0 x
   b = 2`b 1 0

If a && b	=	x && 1 = x
a || b	=	x || 1 = 1
! a	=	not (logical x) = x
! b	=	not (logical 1) = 0

Operators

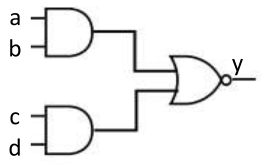
Example

Write the HDL code for the circuit defined



Operators

Example



```
module aoi1 (y, a, b, c, d);

input   [3 : 0]  a, b, c, d;
output [3 : 0]  y;

assign    y = ~ ( ( a & b)  | ( c & d) );

endmodule
```

```
module aoi2 (y, a, b, c, d);

input   [3 : 0]  a, b, c, d;
output [3 : 0]  y;

assign    y = ! ( ( a && b)  | ( c && d) );

endmodule
```

a = 4`b0001
b = 4`b0000
c = 4`b1110
d = 4`b1001

⇩

Identify the output values…!

Operators

Reduction Operator

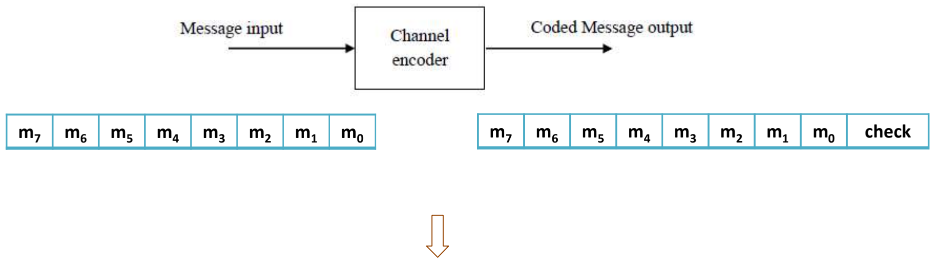
- Reduction AND (&)
- Reduction OR (|)
- Reduction NAND (~ &)
- Reduction NOR (~ |)
- Reduction XOR (^)
- Reduction XNOR (~ ^ or ^ ~)

- Reduction operators take one operand and returns a single bit.

Operators

Example

Single parity check bit


Message input
Channel encoder
Coded Message output
m_7 m_6 m_5 m_4 m_3 m_2 m_1 m_0
m_7 m_6 m_5 m_4 m_3 m_2 m_1 m_0 check

Implement the functionality defined above…

Operators

Bitwise, Logical and Reduction Operators

Operators		
Bitwise	Logical	Reduction
Not (~)	NOT (!)	
AND (&)	AND (&&)	AND (&)
OR (|)	OR (| |)	OR (|)
EXOR (^)		EXOR (^)
EXNOR (~^ or ^~)		EXNOR (~^ or ^~)
		NAND (~&)
		NOR (~|)

Operators

Relational and Equality Operator

Relational Operator

- Less than (<)
- Less than or equal to (<=)
- Greater than (>)
- Greater than or equal to (>=)

- If z or x, return x value.

Operators

Relational Operator

Example

```
1) If a =  4
      b =  3
      c =  4`b1010
      d =  4`b1101
      e =  4`b1xxx

    a <= b     = 0
    a >  b     = 1
    c >= d     = 0
    c <  d     = x
```

Operators

Relational Operator

Example

Find whether a is greater than or equal to b using Verilog HDL

- Inputs – ‘a’ and ‘b’ can be considered as 3 bits

⇩

Implement the functionality defined above…

Operators

Relational and Equality Operators

Equality Operator

- Equality (= =) // Possible logical values are 0, 1, x
- Inequality (! =) // Possible logical values are 0, 1, x
- Case Equality (= = =) // Possible logical values are 0, 1
- Case Inequality (! = =) // Possible logical values are 0, 1

Operators

Equality Operator

Example

Implement a system to compare two numbers

- Inputs – ‘a’ and ‘b’ can be considered as 8 bits

```
module compare2 (a, b, check1, check2);

input   [7:0] a, b;
output check1, check2;

assign    check1 = a = = b;
assign    check2 = a = = = b;

endmodule
```

⇨ Identify the output values…!

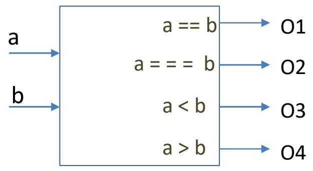
Operators

Equality Operator

Example

Implement a system to compare two numbers

- Inputs – ‘a’ and ‘b’ can be considered as 8 bits



⇩

Implement the functionality defined above…

Operators

Shift Operator

- Shift Left (<<)
- Shift Right (>>)
- Arithmetic Shift Left (<<<)
- Arithmetic Shift Right (>>>)

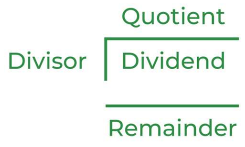
Operators

Shift Operator

Example

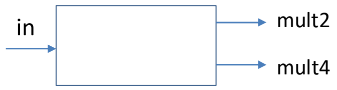
Implement divide by 4 Operation

- Input – ‘dividend’ be considered as 4 bits



Implement Multiply by 2 and Multiply by 4 Operations

- Input – ‘in’ be considered as 3 bits



Operators

Concatenation and Replication Operators

- Concatenation ({ })
- Replication ({ { } })

Operators

Concatenation Operator

Example

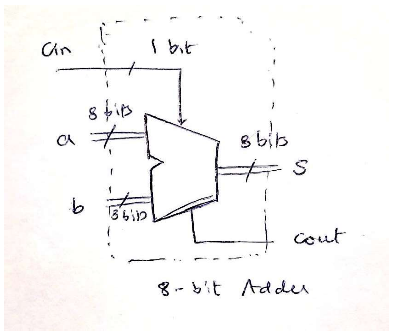
{1`b1, {2 {1`b0} }   = 3’b100

If a = 1`b1	Y = {b, c}	Y = 4`b0010
b = 2`b00	Y = {a, b, c, d, 3`b001}	Y = 11`b10010100001
c = 2`b10	Y = {a, b[0], c[1]}	Y = 3`b101
d = 3`b100		

Operators

Example

Write the HDL code for the circuit defined



⇩

Implement the functionality defined above…

Operators

Replication Operator

Example

{1`b1, {2 {1`b0} }   = 3’b100

If a = 1`b1	Y = { 4 { a } }	Y = 4`b1111
b = 2`b00	Y = { 4 { a }, 2 { b } }	Y = 8`b11110000
c = 2`b10	Y = { 4 { a }, 2 { b }, c }	Y = 10`b1111000010
d = 3`b100		

Operators

Conditional Operator

- Conditional (? :)

Syntax

Output = Conditional Exp ? True Exp : False Exp;

- If the result is x (ambiguous), then both true_exp and false_exp are evaluated bit-by-bit to return for each bit position as
  - x if the bits are different and
  - the value of bits if they are same.

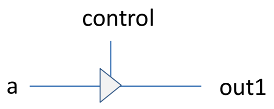
Operators

Conditional Operator

Example

To model functionality of Tristate Buffer

- Input – ‘a’ can be considered as 8 bits



⇩

Implement the functionality defined above…

Operators

Example

```
module logic_condn (a, b, control, out1);
input   [7 : 0 ] a, b;
input   control;
output [7 : 0] out1;
assign  out1  = control ? a : b;
endmodule
```

a = 8`b1101_1001
b = 8`b1000_0111

⇩

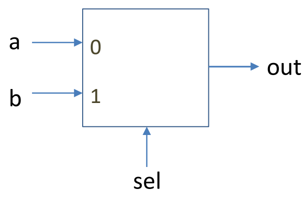
Identify the output values…!

Operators

Example

Implement 2:1 Mux

- Inputs – ‘a’ and ‘b’ can be considered as single bit

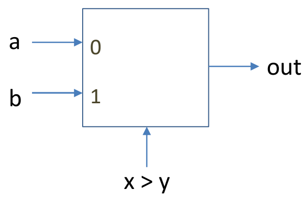


Implement 2:1 Mux

- Inputs – ‘a’ and ‘b’ to be considered as 8 bits

Implement 2:1 Mux

- Inputs – ‘a’ and ‘b’ can be considered as 8 bits
- ‘x’ and ‘y’ can be considered as 3 bits



Operators

Example

Find Greatest number amongst 3 numbers: a, b, c

Implement 4:1 Mux using Conditional Operator

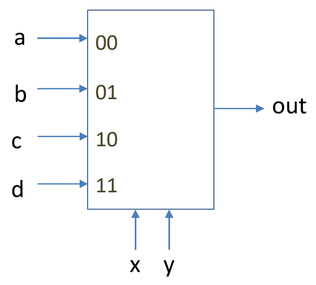
- Inputs – ‘a’, ‘b’, ‘c’ and ‘d’ can be considered as single bit

Implement 4:1 Mux using Bitwise Operator

- Inputs – ‘a’, ‘b’, ‘c’ and ‘d’ can be considered as single bit

Implement 4:1 Mux using 2:1 Mux Instantiation

- Inputs – ‘a’, ‘b’, ‘c’ and ‘d’ can be considered as single bit



I am available/approachable at

email: yash_agrawal@iiitvadodara.ac.in
yash_agrawal@daiict.ac.in