DSLab. 03 Boolean Algebra

Lab. 03 Boolean Algebra

- Design and Verify the following circuits using Schematic and Verilog HDL
- Schematic
- Verilog HDL
  - Behavioral level modeling
  - Dataflow modeling
  - Structural level modeling
- Please write and upload the lab report (Lab03) -- Due on 2022/09/23 23:59

Operators in Verilog

Table 8.1
Verilog 2001 HDL Operators

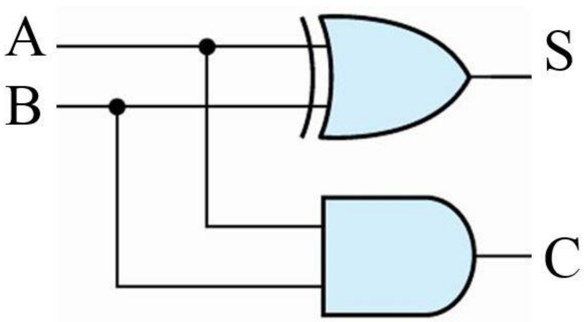
Operator Type	Symbol	Operation Performed
Arithmetic	+	addition
	–	subtraction
	*	multiplication
	/	division
	%	modulus
	**	exponentiation
Bitwise or Reduction	~	negation (complement)
	&	AND
	|	OR
	^	exclusive-OR (XOR)
Logical	!	negation
	&&	AND
	||	OR
Shift	>>	logical right shift
	<<	logical left shift
	>>>	arithmetic right shift
	<<<	arithmetic left shift
	{ , }	concatenation
Relational	>	greater than
	<	less than
	==	equality
	!=	inequality
	===	case equality
	!==	case inequality
	>=	greater than or equal
	<=	less than or equal

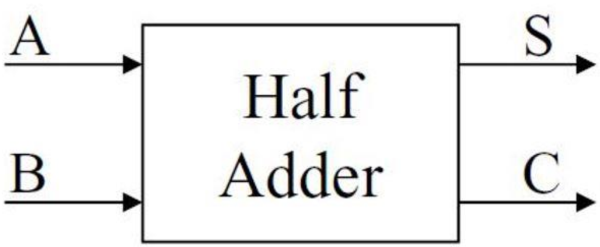
Table 8.2
Verilog Operator Precedence

+ – ! ~ & ~& | ~| ^ ~^ ^~ (unary)	Highest precedence
**	
* / %	
+ – (binary)	
<< >> <<< >>>	
< <= > >=	
== != === !==	
& (binary)	
^ ^~ ~^ (binary)	
| (binary)	
&&	
||	
?: (conditional operator)	
{ } { { } }	Lowest precedence

Use parentheses to enforce your priority

Example 1: Half Adder (Dataflow modeling)



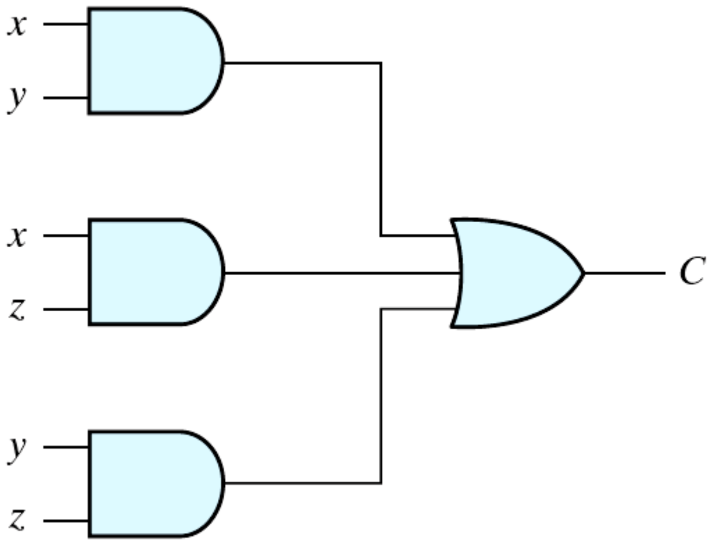


$S = A \oplus B$

$C = AB$

```
module half_adder (S, C, A, B);
    output S, C;
    input A, B;

    assign S = A ^ B;
    assign C = A & B;

endmodule
```

Example 2: Carry of Full Adder (Dataflow modeling)



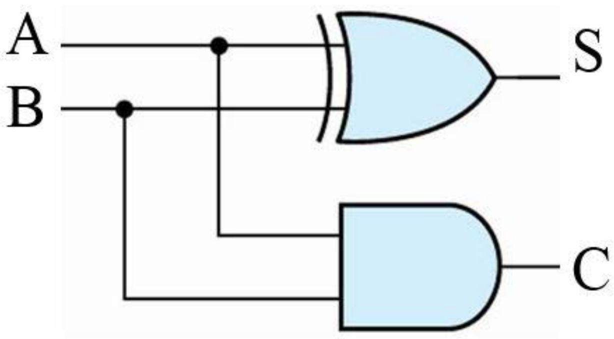
$C = xy + xz + yz$

```
module C_FA (C, x, y ,z);
    output C;
    input x, y, z;

    assign C = x & y | x & z | y & z;

endmodule
```

Verilog Structural Model (Gate Level)

- Built-in gate primitives:
  - **and, nand, nor, or, xor, xnor, buf, not, bufif0, bufif1, notif0, notif1**
- Usage:
  - nand (out, in1, in2); 2-input NAND without delay
  - and #2 (out, in1, in2, in3); 3-input AND with 2 t.u. delay
  - not #1 N1(out, in); NOT with 1 t.u. delay and instance name
  - xor X1(out, in1, in2); 2-input XOR with instance name
- Write them inside module, outside procedures

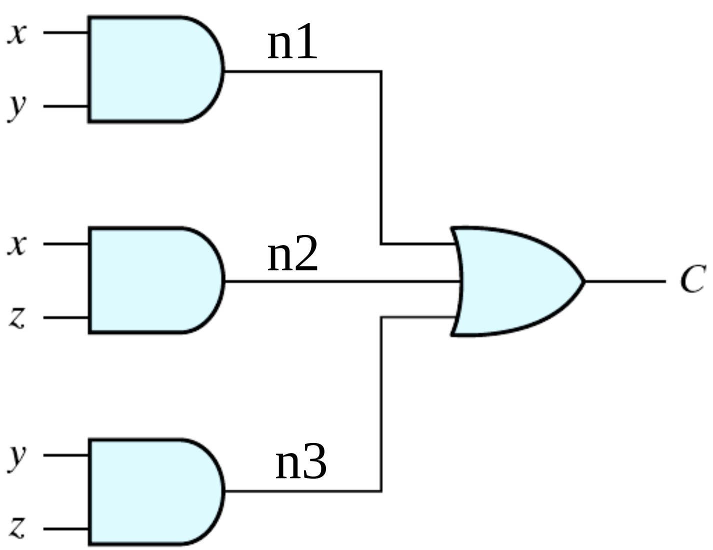
Example 3: Half Adder (Structural level modeling)



Assuming:
- XOR: 2 t.u. delay
- AND: 1 t.u. delay

```
module half_adder (S, C, A, B);
    output S, C;
    input A, B;

    xor #2 (S, A, B);
    and #1 (C, A, B);

endmodule
```

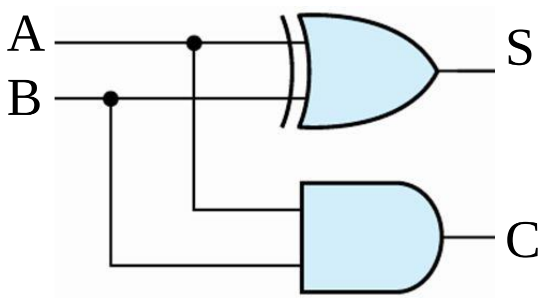
Example 4: Carry of Full Adder (Structural level modeling)



$C = xy + xz + yz$

```
module C_FA(C, x, y ,z);
    output C;
    input x, y, z;
    wire n1, n2, n3;

    and (n1, x, y);
    and (n2, x, z);
    and (n3, y, z);
    or  (C, n1, n2, n3);

endmodule
```

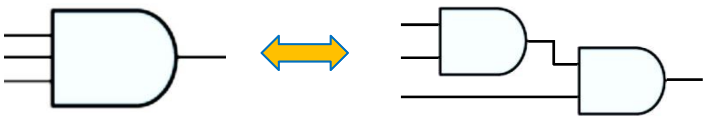
Testbench

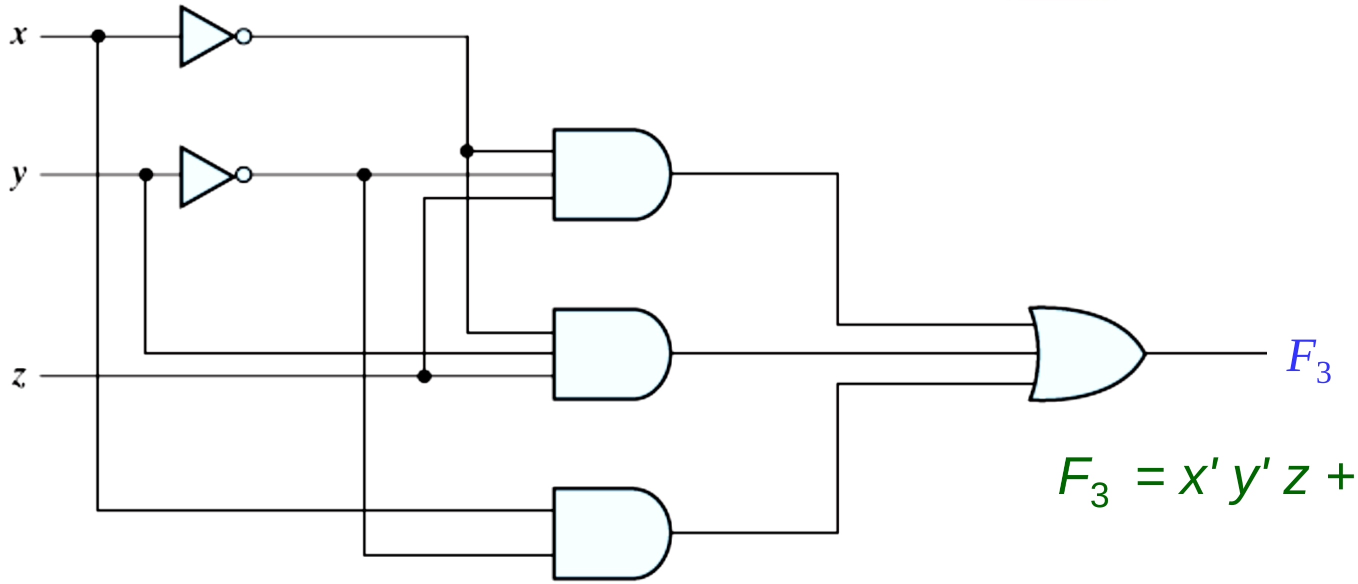



```
module t_half_adder;
  reg A;
  reg B;
  wire S;
  wire C;

  half_adder UUT ( .A(A), .B(B), .S(S), .C(C) );

  initial begin
     A=1'b0; B=1'b0;
     #10  A=1'b0; B=1'b1;
     #10  A=1'b1; B=1'b0;
     #10  A=1'b1; B=1'b1;
     #10  $finish;
  end
endmodule
```

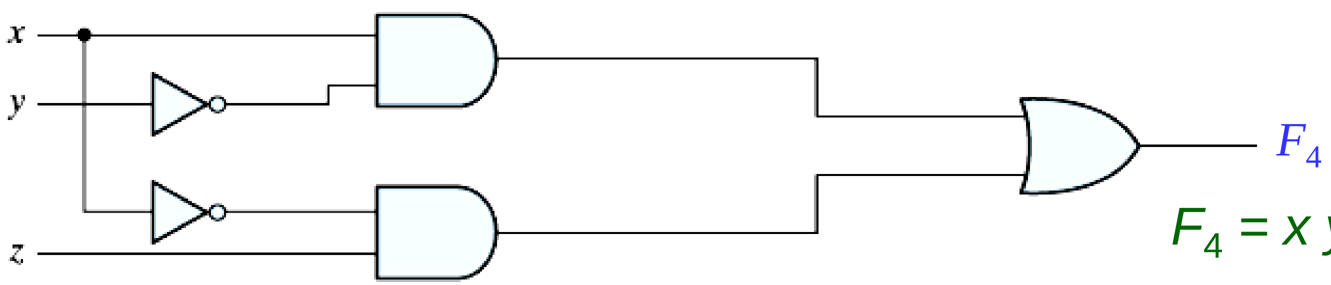
Exercise 1

- Verify $F_3 = F_4$ (using Schematic)





$$F_3 = x'\,y'\,z + x'\,y\,z + x\,y'$$



$$F_4 = x\,y' + x'\,z$$

Exercise 2

- Verify Postulate 4(b) $x + yz = (x+y)(x+z)$
  - using Dataflow modeling

Postulate 2	(a)	$x + 0 = x$	(b)	$x \cdot 1 = x$
Postulate 5	(a)	$x + x' = 1$	(b)	$x \cdot x' = 0$
Theorem 1	(a)	$x + x = x$	(b)	$x \cdot x = x$
Theorem 2	(a)	$x + 1 = 1$	(b)	$x \cdot 0 = 0$
Theorem 3, involution		$(x')' = x$		
Postulate 3, commutative	(a)	$x + y = y + x$	(b)	$xy = yx$
Theorem 4, associative	(a)	$x + (y + z) = (x + y) + z$	(b)	$x(yz) = (xy)z$
Postulate 4, distributive	(a)	$x(y + z) = xy + xz$	(b)	$x + yz = (x + y)(x + z)$
Theorem 5, DeMorgan	(a)	$(x + y)' = x'y'$	(b)	$(xy)' = x' + y'$
Theorem 6, absorption	(a)	$x + xy = x$	(b)	$x(x + y) = x$

Exercise 3

- Verify *xy + xy'z + x'yz = xy + xz + yz*
  - using Structural level modeling

$$
\begin{aligned}
\underline{xy} + xy'z + x'yz &= \underline{xy + \boxed{xyz}} + \underline{xy'z} + x'yz && \text{by absorption} && \textbf{T6(a): } x + xy = x \\
&= xy + x(y + y')z + x'yz && \text{by distributivity} && \textbf{P4(a)} \\
&= xy + x1z + x'yz && \text{by complement} \\
&= xy + xz + x'yz && \text{by identity} \\
&= xy + \boxed{xyz} + xz + \underline{x'yz} && \text{by absorption} && \textbf{T6(a)} \\
&= xy + xz + (x + x')yz && \text{by distributivity} && \textbf{P4(a)} \\
&= xy + xz + 1yz && \text{by complement} \\
&= xy + xz + yz && \text{by identity}
\end{aligned}
$$