

Verilog HDL 연산자

Kyung-Wook Shin
kwshin@kumoh.ac.kr

School of Electronic Eng.,
Kumoh National Institute of Technology

Verilog HDL

자료형과 연산자

K. W. SHIN

2.2 Verilog의 연산자

2

표 2.8 Verilog HDL의 연산자

연산자	기능	연산자	기능
{}, {{}}	결합, 반복	^	비트 exclusive or (비트 xor)
+, -, *, /, **	산술	^~ 또는 ~^	비트 등가 (비트 xnor)
%	나머지	&	축약 and
>, >=, <, <=	관계	~&	축약 nand
!	논리 부정		축약 or
&&	논리 and	~	축약 nor
	논리 or	^	축약 xor
==	논리 등가	^~ 또는 ~^	축약 xnor
!=	논리 부등	<<	논리 왼쪽 시프트
===	case 등가	>>	논리 오른쪽 시프트
!==	case 부등	<<<	산술 왼쪽 시프트
~	비트 부정	>>>	산술 오른쪽 시프트
&	비트 and	? :	조건
	비트 inclusive or	or	Event or

Verilog HDL

자료형과 연산자

K. W. SHIN

2.2 Verilog의 연산자

3

표 2.9 실수형 수식에 사용될 수 있는 연산자


연산자	기능	연산자	기능
+, -, *, /, **	산술		논리 or
+, -	부호	==	논리 등가
>, >=, <, <=	관계	!=	논리 부등
!	논리 부정	? :	조건
&&	논리 and	or	Event or

❖ 비트, 축약, 시프트 연산자는 실수형 수식에 사용될 수 없음.

2.2 Verilog의 연산자

4

표 2.10 Verilog 연산자의 우선순위

+, -, !, ~ (단항)	Highest priority
**	
*, /, %	
+, - (이항)	
<<, >>, <<<, >>>	
<, <=, >, >=	
==, !=, ==~, !=~	
&, ~&	
^, ^~, ~^	
, ~	
&&	
? : (조건연산자)	
	Lowest priority

2.2.1 산술 연산자

5

- ❖ 피연산자의 비트에 x (unknown)나 z (high-impedance)가 포함된 경우에는 전체 결과 값은 x가 됨
- ❖ 나누기와 나머지 연산자에서 두 번째 피연산자가 0인 경우, 결과값은 x가 됨
- ❖ 나머지 연산자의 결과 값은 첫번째 피연산자의 부호를 따름
- ❖ 거듭제곱 연산자에서 다음의 경우에는 결과 값이 정의되지 않음
 - 첫번째 피연산자가 0이고 두 번째 피연산자가 양수가 아닌 경우
 - 첫번째 피연산자가 음수이고 두 번째 피연산자가 정수 값이 아닌 경우

표 2.11 산술 연산자

기 호	기 능
+	더하기
-	빼기
*	곱하기
/	나누기(몫)
%	나머지(modulo)
**	거듭제곱(power)

2.2.1 산술 연산자

6

표 2.12 나머지 연산자의 연산 결과 예

수 식	결과값	설 명
10 % 3	1	10을 3으로 나눈 나머지는 1
11 % 3	2	11을 3으로 나눈 나머지는 2
12 % 3	0	12를 3으로 나눈 나머지는 0
-10 % 3	-1	결과 값은 첫번째 피연산자의 부호를 따름
11 % -3	2	결과 값은 첫번째 피연산자의 부호를 따름

2.2.2 관계 연산자

7

- ❖ 결과 값: 1비트의 참(1) 또는 거짓(0)
- ❖ 산술 연산자보다 낮은 우선 순위를 가짐
- ❖ 피연산자의 비트에 x (unknown)나 z(high-impedance)가 포함된 경우에는 결과 값은 1비트의 x가 됨
- ❖ 두 피연산자의 비트 수가 다른 경우에는, 비트 수가 작은 피연산자의 MSB 쪽에 0이 채워져 비트 수가 큰 피연산자에 맞추어진 후, 관계를 판단함
- ❖ 피연산자 중 하나가 실수형이면 다른 피연산자가 실수형으로 변환된 후, 비교됨

표 2.14 관계 연산자

관계 연산자 식	의 미
$a < b$	a가 b보다 작다
$a > b$	a가 b보다 크다
$a \leq b$	a가 b보다 작거나 같다
$a \geq b$	a가 b보다 크거나 같다

Verilog HDL

자료형과 연산자

K. W. SHIN

2.2.2 관계 연산자

8

예 2.2.3 관계 연산자 수식

```
// A = 9, B = 4
// D = 4'b1001, E = 4'b1100, F = 4'b1xxx

A <= B    // 결과 값은 거짓 (0)
A > B     // 결과 값은 참 (1)
E >= D    // 결과 값은 참 (1)
E < F     // 결과 값은 x
```

예 2.2.4

```
① a < b-1      // ①과 ②는 결과가 동일
② a < (b-1)
③ b-1 < a      // ③과 ④는 결과가 다를 수 있음
④ b-1 < a
```

Verilog HDL

자료형과 연산자

K. W. SHIN

2.2.3 등가 연산자

9

- ❖ 결과 값: **1비트의 참(1) 또는 거짓(0)**
- ❖ 피연산자의 **비트끼리 비교**
- ❖ 관계 연산자 보다 낮은 우선순위를 가짐
- ❖ 두 피연산자의 비트 수가 다른 경우에는, 비트 수가 작은 피연산자의 **MSB** 쪽에 **0**이 채워져 비트 수가 큰 피연산자에 맞추어진 후, 등가를 판단함
- ❖ **case equality**와 **case inequality** 연산자(==, !=)는 EDA 툴에서 논리합성이 지원되지 않을 수 있음

표 2.15 등가 연산자

관계 연산자 식	의 미
<code>a === b</code>	a와 b는 같다. (x와 z가 포함된 일치를 판단)
<code>a !== b</code>	a와 b는 같지 않다. (x와 z가 포함된 불일치를 판단)
<code>a == b</code>	a와 b는 같다. (결과가 x가 될 수 있음)
<code>a != b</code>	a와 b는 같지 않다. (결과가 x가 될 수 있음)

Verilog HDL

자료형과 연산자

K. W. SHIN

2.2.3 등가 연산자

10

예 2.2.5 등가 연산자의 예

```
// A = 9, B = 4
// D = 4'b1001, E = 4'b1100
// F = 4'b1xxz, G = 4'b1xxz, H = 4'b1xxx

A === B    // 결과 값은 거짓 (0)
D != E     // 결과 값은 참 (1)
D == F     // 결과 값은 x
F === G    // 결과 값은 참 (1)
F === H    // 결과 값은 거짓 (0)
G !== H    // 결과 값은 참 (1)
```

Verilog HDL

자료형과 연산자

K. W. SHIN

2.2.4 논리 연산자

11

- ❖ 결과값은 1비트의 참(1) 또는 거짓(0)
- ❖ 참 또는 거짓의 판단이 모호한 경우에는 결과값은 x

논리 연산자 식	의 미
a && b	a와 b의 논리 AND
a b	a와 b의 논리 OR
!a	a의 부정 (NOT a)

예 2.2.6 논리 연산자의 예

```
// A = 3, B = 0, C = 2'b0x, D = 2'b10인 경우에,  
A && B // 결과 값은 0  
A || B // 결과 값은 1  
!A     // 결과 값은 0  
!B     // 결과 값은 1  
C && D // 결과 값은 x
```

2.2.4 논리 연산자

12

예 2.2.7

```
// alpha = 237, beta=0인 경우에,  
  
regA = alpha && beta; // regA에는 0이 할당된다.  
regB = alpha || beta; // regB에는 1이 할당된다.  
  
a < size-1 && b != c && index != last_one  
(a < size-1) && (b != c) && (index != last_one) // recommended  
  
if(!reset) // if(reset == 1'b0)과 동가임
```

2.2.5 비트 연산자

13

- ❖ 피연산자의 해당 비트들에 대한 연산을 수행
 - 피연산자의 비트 수만큼의 결과를 출력함
- ❖ 피연산자의 비트 수가 같지 않으면, 비트 수가 작은 피연산자의 MSB 위치에 0이 채워진 후, 연산됨
- ❖ (a ~^ b)는 허용되나, (a ~& b) 또는 (a ~| b)는 허용되지 않음
 - ~(a & b), ~(a | b)로 표현해야 함

표 2.16 비트 and 연산자

&	0	1	x	z
0	0	0	0	0
1	0	1	x	x
x	0	x	x	x
z	0	x	x	x

표 2.17 비트 or 연산자

	0	1	x	z
0	0	1	x	x
1	1	1	1	1
x	x	1	x	x
z	x	1	x	x

Verilog HDL

자료형과 연산자

K. W. SHIN

2.2.5 비트 연산자

14

표 2.18 비트 xnor 연산자

~^	0	1	x	z
0	1	0	x	x
1	0	1	x	x
x	x	x	x	x
z	x	x	x	x

표 2.19 비트 xor 연산자

^	0	1	x	z
0	0	1	x	x
1	1	0	x	x
x	x	x	x	x
z	x	x	x	x

표 2.20 비트 부정 연산자

~	0
0	1
1	0
x	x
z	x

예 2.2.8 비트 연산자의 예

```
// D = 4'b1001, E = 4'b1101, F = 4'b10x1
~D          // 결과 값은 4'b0110
D & E       // 결과 값은 4'b1001
D | E       // 결과 값은 4'b1101
D ^ E       // 결과 값은 4'b0100
D ~^ E      // 결과 값은 4'b1011
D & F       // 결과 값은 4'b1001
```

Verilog HDL

자료형과 연산자

K. W. SHIN

2.2.6 축약(Reduction) 연산자

15

- ❖ 단항 연산자
- ❖ 피연산자의 단위 비트들에 적용되어 1비트의 결과값을 생성

```
reg[7:0] cnt;  
assign parity = ^cnt;  
assign parity = cnt[7]^cnt[6]^cnt[5]^cnt[4]^cnt[3]^cnt[2]^cnt[1]^cnt[0];
```

표 2.21 축약 and 연산자

&	0	1	x	z
0	0	0	0	0
1	0	1	x	x
x	0	x	x	x
z	0	x	x	x

축약 nand 연산자

~&	0	1	x	z
0	1	1	1	1
1	1	0	x	x
x	1	x	x	x
z	1	x	x	x

Verilog HDL

자료형과 연산자

K. W. SHIN

2.2.6 축약 연산자

16

표 2.22 축약 or 연산자

	0	1	x	z
0	0	1	x	x
1	1	1	1	1
x	x	1	x	x
z	x	1	x	x

축약 nor 연산자

~	0	1	x	z
0	1	0	x	x
1	0	0	0	0
x	x	0	x	x
z	x	0	x	x

표 2.23 축약 xor 연산자

^	0	1	x	z
0	0	1	x	x
1	1	0	x	x
x	x	x	x	x
z	x	x	x	x

축약 xnor 연산자

~^	0	1	x	z
0	1	0	x	x
1	0	1	x	x
x	x	x	x	x
z	x	x	x	x

Verilog HDL

자료형과 연산자

K. W. SHIN

2.2.6 축약 연산자

17

예 2.2.9 축약 연산자의 연산 결과

연산자 피연산자	연산 결과						설 명
	&	~&		~	^	~^	
4'b0000	0	1	0	1	0	1	모든 비트가 0인 경우
4'b1111	1	0	1	0	0	1	모든 비트가 1인 경우
4'b0110	0	1	1	0	0	1	1의 개수가 짝수인 경우
4'b1000	0	1	1	0	1	0	1의 개수가 홀수인 경우

2.2.6 축약 연산자

18

예 2.2.10 연산자를 이용한 4입력 NAND 게이트 모델링



[그림 2.7] 4입력 NAND 게이트

```
module nand4_op2(a, y);
  input [3:0] a;
  output y;

  assign y = ~&a;
endmodule
```

축약 nand 연산자 사용

```
module nand4_op1(a, y);
  input [3:0] a;
  output y;

  assign y = ~(a[0] & a[1] & a[2] & a[3]);
endmodule
```

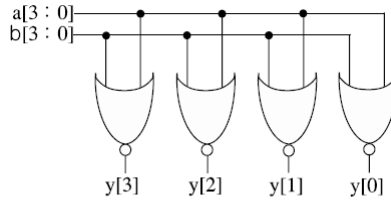
비트 연산자 사용

[코드 2.3] 연산자를 이용한 4입력 NAND 게이트 모델링

2.2.6 축약 연산자

19

예 2.2.11



[그림 2.8] 4비트 2입력 NOR 게이트 회로

```
module nor_op(a, b, y);
    input [3:0] a, b;
    output [3:0] y;

    assign y = ~(a | b);

endmodule
```

[코드 2.4] 연산자를 이용한 4비트 2입력 NOR 게이트 모델링

Verilog HDL

자료형과 연산자

K. W. SHIN

2.2.7 시프트 연산자

20

□ 논리 시프트 연산자 (<<, >>)

- ❖ << : 우측 피연산자 값만큼 좌측 피연산자를 **왼쪽으로 시프트** 후, 비어 있는 비트에 **0을 채움**
- ❖ >> : 우측 피연산자 값만큼 우측 피연산자를 **오른쪽으로 시프트** 후, 비어 있는 비트에 **0을 채움**

□ 산술 시프트 연산자 (>>>, <<<)

- ❖ <<< : 우측 피연산자 값만큼 좌측 피연산자를 **왼쪽으로 시프트** 후, 비어 있는 비트에 **0을 채움**
- ❖ >>> : 우측 피연산자 값만큼 우측 피연산자를 **오른쪽으로 시프트** 후, 비어 있는 비트에 **좌측 피연산자의 MSB를 채움**
 - 자료형에 **signed** 속성이 추가되어야 함

□ 우측 피연산자

- ❖ x 또는 z가 포함된 경우, 시프트 연산의 결과 값은 x
- ❖ 항상 **unsigned** 수

Verilog HDL

자료형과 연산자

K. W. SHIN

2.2.7 시프트 연산자

21

예 2.2.12

```
// A = 4'b1100
B = A >> 1 // 오른쪽으로 1비트 시프트, 결과 값은 B=4'b0110
C = A << 1 // 왼쪽으로 1비트 시프트, 결과 값은 B=4'b1000
D = A << 2 // 왼쪽으로 2비트 시프트, 결과 값은 B=4'b0000
```

예 2.2.13

[코드 2.5] 왼쪽 논리시프트 연산자의 예

```
module shift;
    reg [3:0] start, result;

    initial begin
        start = 1;
        result =(start << 2); // 결과 값은 0100
    end
endmodule
```

2.2.7 시프트 연산자

22

예 2.2.13

[코드 2.6] 오른쪽 산술시프트 연산자의 예

```
module ashift;
    reg signed [3:0] start, result;
    initial begin
        start = 4'b1000;
        result =(start >>> 2); // 결과 값은 1110
    end
endmodule
```

2.2.7 시프트 연산자

23

```
module shift_op ();
  wire signed [7:0] data1 = 8'b1010_1100;
  wire      [7:0] data2 = 8'b1000_1111;
  reg        [7:0] out1, out2;

  always @(data1, data2) begin
    out1 = data1 >>> 2; // out1 : 1110_1011
    out2 = data2 >>> 2; // out2 : 0010_0011
    #100 $stop;
  end
endmodule
```

Verilog HDL

자료형과 연산자

K. W. SHIN

2.2.8 조건 연산자

24

□ 조건 연산자

- ❖ **expression1**이 참(1, 즉 0, x 또는 z가 아닌 값)으로 평가되면 **expression2**의 값이 좌변의 변수에 할당
- ❖ **expression1**이 거짓(0)으로 평가되면 **expression3**의 값이 좌변의 변수에 결과 값으로 할당
- ❖ **expression1**이 x 또는 z이면(즉, 참 또는 거짓을 판단할 수 없는 모호성이 존재하는 경우), **expression2**와 **expression3**을 함께 평가하여 비트 단위로 비교된 값이 좌변의 변수에 할당
 - **expression3**이 real 형 값이 아니면 결과 값은 비트 단위로 비교되어 결정되며, real 형 값인 경우에는 결과 값은 0이 됨

```
conditional_expression ::= expression1 ? expression2 : expression3
```

Verilog HDL

자료형과 연산자

K. W. SHIN

2.2.8 조건 연산자

25

표 2.24 조건에 애매성이 존재하는 경우의 조건 연산자의 결과 값 결정

?:	0	1	x	z
0	0	x	x	x
1	x	1	x	x
x	x	x	x	x
z	x	x	x	x

예 2.2.14 조건 연산자를 이용한 3상태 버퍼

```
wire [15:0] busa, data;
assign busa = drive_bus ? data : 16'bz;
```

Verilog HDL

자료형과 연산자

K. W. SHIN

2.2.9 결합 및 반복 연산자

26

□ 결합 연산자

- ❖ 중괄호 { }에 의해 묶인 두 개 이상의 표현이 갖는 비트들을 결합
 - 결합되는 피연산자들은 각각의 크기를 결정할 수 있어야 결합이 가능함
 - **unsized** 상수는 결합 연산자로 결합시킬 수 없음
- ❖ 대입문의 좌측 또는 우측에 사용 가능
- ❖ 비트 폭이 일치하지 않는 변수의 연산이나 대입이 허용됨
 - 우변의 비트 폭이 작은 경우, 우변의 **MSB**에 0을 붙여 연산 됨
 - 좌변의 비트 폭이 우변 보다 작을 경우, **MSB**는 누락되어 저장

예 2.2.15

```
wire [15:0] addr_bus;
Wire [7:0] addr_h, addr_l
assign addr_bus = {addr_h, addr_l};
```

Verilog HDL

자료형과 연산자

K. W. SHIN

2.2.9 결합 및 반복 연산자

27

예 2.2.16

```
wire [3:0] a, b, sum;
wire carry;
assign {carry, sum} = a + b; // 4비트 데이터의 덧셈은 5비트 결과
// 좌변이 5 비트이므로, 우변의 a+b는 MSB에 0을 붙인 5비트로 연산 됨
```

Verilog HDL

자료형과 연산자

K. W. SHIN

2.2.9 결합 및 반복 연산자

28

□ 반복 연산자

- ❖ {a(b)}의 형태로 표현하여 b를 a회 반복
 - 반복 횟수 a는 0, x, z가 아닌 상수이어야 함

예 2.2.17

```
{4{w}} // {w, w, w, w}와 동일한 표현임.
a[31:0] = {1'b1, {0{1'b0}}}; // 우변이 {1'b1}가 되므로 잘못된 표현임
a[31:0] = {1'b1, {1'bz{1'b0}}}; // 우변이 {1'b1}가 되므로 잘못된 표현임
a[31:0] = {1'b1, {1'bx{1'b0}}}; // 우변이 {1'b1}가 되므로 잘못된 표현임
```

예 2.2.18

```
{b, {3{a, b}}} // {b, a, b, a, b, a, b}와 동일함
```

Verilog HDL

자료형과 연산자

K. W. SHIN