

Signal & Variable

Copyright (c) 2012 Young W. Lim.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

Please send corrections (or suggestions) to youngwlim@hotmail.com.

This document was produced by using OpenOffice and Octave.

Concurrent Statement

- Block Statement
- Process Statement
- Component Statement
- Generate Statement
- Concurrent Signal Assignment
- Concurrent Assertion
- Concurrent Procedure Call

- Architecture Body
- Block Statement
- Generate Statement

- Conditional Signal Assignment
- Selected Signal Assignment

Sequential Statement

- Wait Statement
- Assertion Statement
- Report Statement
- Generate Statement
- Signal Assignment
- Variable Assignment
- Procedure Call
- If
- Case
- Loop
- Next
- Exit
- Return
- Null

- Case Statement
- If Statement
- Loop Statement
- Process Statement
- Subprogram Body

- Conditional Signal Assignment
- Selected Signal Assignment

Conditional Signal Assignment

```
Z <= A or B [ after 1 ns ] when S0 = '1' else  
      A or C [ after 2 ns ] when S1 = '1' else  
      A or D [ after 3 ns ] ;
```

```
Z <= A or B [ after 1 ns ] when S0 = '1' else  
      A or C [ after 2 ns ] ;
```

```
Z <= A or B [ after 1 ns ] when S0 = '1' ;
```

```
Z <= A or B [ after 1 ns ] ;
```

← *simple concurrent statement*

- Concurrent Signal Assignment

- Conditional Signal Assignment
- Selected Signal Assignment

Selected Signal Assignment

- Conditional Signal Assignment

```
Z <=  A or B  [ after 1 ns ]  when SEL = "00" else
      A or C  [ after 2 ns ]  when SEL = "01" else
      A or D  [ after 2 ns ]  when SEL = "10" else
      A or E  [ after 3 ns ]  when SEL = "11" else
      A or F  [ after 4 ns ]  ;
```

- Selected Signal Assignment

```
with SEL select
Z <=  A or B  [ after 1 ns ]  when "00",
      A or C  [ after 2 ns ]  when "01",
      A or D  [ after 3 ns ]  when "10",
      A or E  [ after 4 ns ]  when "11",
      A or F  [ after 5 ns ]  when others;
```

Concurrent vs Sequential

with *SEL* **select**

```
Z <= A or B [after 1 ns] when "00",  
  ● ● ●
```

```
Z <= A or B [after 1 ns] when SEL = "00" else  
  ● ● ●
```

Should be
outside process statement

```
Z <= A or B [after 1 ns] ;
```

← Simple Concurrent signal statement
outside process statement

```
process (A, B, C)
```

```
begin
```

```
  Z <= A or B [after 1 ns] ;
```

```
end
```

← Sequential signal statement
inside process statement

References

[1] <http://en.wikipedia.org/>

[2]