

In this lecture we take a deeper dive into VHDL signal assignments and explore language features crucial for performing simulations requiring more complex signal timings. Understanding their semantics will also give us more insight into the event-based VHDL simulation in general.

HWMod  
WS25

Signal Assignn.

Introduction

Assign. Statements

Waveforms

Examples

Concurrent Assignn.

## Hardware Modeling [VU] (191.011) – WS25 – Signal Assignments

Florian Huemer & Sebastian Wiedemann & Dylan Baumann

WS 2025/26

Signal assignments were one of the first topics we covered in this course. We learned that there are two types of them, depending on whether they are used in a process or as a stand-alone concurrent assignment. We also intensively covered how signal assignments are fundamentally different from variable assignments. However, so far we have only seen and used very basic assignments. In most cases some target signal was assigned a constant value or some other signal. Beyond that we have only used conditional assignments, as a compact way to express if-ELSE-if statements.

## Introduction

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Concurrent Assign.

- Recap
  - Two types
    - Assignment statements in processes
    - Concurrent signal assignments
  - Fundamentally different from variable assignments
  - Only covered very basic value assignments

In this lecture, we will take a more systematic and comprehensive look at the signal assignment syntax supported by VHDL. We will learn that for simulations signal assignments can be equipped with delay information and that it is even possible to describe complex waveform patterns with them. Exploring the semantics of these language features and the way they are handled during simulation, will also give us insight into the event-based VHDL simulation.

## Introduction

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Concurrent Assign.

- Recap
  - Two types
    - Assignment statements in processes
    - Concurrent signal assignments
  - Fundamentally different from variable assignments
  - Only covered very basic value assignments
- This lecture
  - More systematic look at the signal assignment syntax
  - For simulations signal assignments
    - can be with equipped with timing information
    - can describe complex waveforms
  - Gain some insight into the simulator

- └ Signal Assignments
  - └ Assign. Statements
    - └ **Signal Assignment Statements (for Processes)**

- Signal assignments in processes
  - Concurrent signals assignments have a similar syntax
  - Covered separately at the end of the lecture

Let's start with discussing signal assignment statements in processes. To the extent to which we cover VHDL in this course you can consider concurrent signal assignments to use the same syntax. While there are some differences, they go beyond the scope of this introductory course. We will come back to concurrent assignments at the end of the lecture.

## Signal Assignment Statements (for Processes)

- **Signal assignments in processes**
  - Concurrent signals assignments have a similar syntax
  - Covered separately at the end of the lecture

HWMoD  
WS25

Signal Assign.  
Introduction  
**Assign. Statements**  
Simple Assign.  
Conc. Assign.  
Waveforms  
Examples  
Concurrent Assign.

# Signal Assignments

## Assign. Statements

### Signal Assignment Statements (for Processes)

- Signal assignments in processes
  - Concurrent signal assignments have a similar syntax
  - Covered separately at the end of the lecture
- Signal assignment statement syntax ◊

```
signal_assignment_statement ::=  
[ label : ] simple_signal_assignment  
| [ label : ] conditional_signal_assignment  
| [ label : ] selected_signal_assignment
```

Consulting the VHDL reference reveals that there are three general classes of assignment statements, which are referred to as simple, conditional and selected assignments.

## Signal Assignment Statements (for Processes)

### ■ Signal assignments in processes

- Concurrent signals assignments have a similar syntax
- Covered separately at the end of the lecture

### ■ Signal assignment statement syntax ◊

```
signal_assignment_statement ::=  
    [ label : ] simple_signal_assignment  
    | [ label : ] conditional_signal_assignment  
    | [ label : ] selected_signal_assignment
```

# Signal Assignments

## Assign. Statements

### Signal Assignment Statements (for Processes)

- Signal assignments in processes
  - Concurrent signal assignments have a similar syntax
  - Covered separately at the end of the lecture
- Signal assignment statement syntax ◊

```
signal_assignment_statement ::=  
[ label : ] simple_signal_assignment  
| [ label : ] conditional_signal_assignment  
| [ label : ] selected_signal_assignment
```
- Optional label

Note that, as all statements in VHDL, assignments can be equipped with an optional label.

## Signal Assignment Statements (for Processes)

### ■ Signal assignments in processes

- Concurrent signals assignments have a similar syntax
- Covered separately at the end of the lecture

### ■ Signal assignment statement syntax ◊

```
signal_assignment_statement ::=
```

```
[ label : ] simple_signal_assignment  
| [ label : ] conditional_signal_assignment  
| [ label : ] selected_signal_assignment
```

### ■ Optional label

# Signal Assignments

## Assign. Statements

### Signal Assignment Statements (for Processes)

- Signal assignments in processes
  - Concurrent signal assignments have a similar syntax
  - Covered separately at the end of the lecture
- Signal assignment statement syntax  $\diamond$ 

```
signal_assignment_statement ::=  
[ label : ] simple_signal_assignment  
| [ label : ] conditional_signal_assignment  
| [ label : ] selected_signal_assignment
```
- Optional label
- Examples

```
simple : x <= a xor b;  
conditional : y <= a when c else b when d else c;  
x <= ('1', others=>'0'); --no label
```

These example assignments show how this can look like in practice. You should be familiar with the semantics of these statements by now.

## Signal Assignment Statements (for Processes)

### ■ Signal assignments in processes

- Concurrent signals assignments have a similar syntax
- Covered separately at the end of the lecture

### ■ Signal assignment statement syntax $\diamond$

```
signal_assignment_statement ::=  
    [ label : ] simple_signal_assignment  
    | [ label : ] conditional_signal_assignment  
    | [ label : ] selected_signal_assignment
```

### ■ Optional label

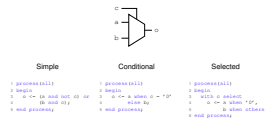
### ■ Examples

```
simple : x <= a xor b;  
conditional : y <= a when c else b when d else c;  
z <= ('1', others=>'0'); --no label
```

# Signal Assignments

## Assign. Statements

### Signal Assignment Statements - Multiplexer Example

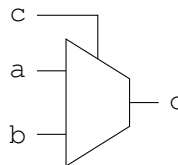


Before we dig deeper into the syntax specification, let's have a quick look at the three assignment classes in action. Here we see a simple two-to-one multiplexer, with the data inputs A and B the control input C and the output O.

## Signal Assignment Statements - Multiplexer Example

HWMMod  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Simple Assign.  
Cond. Assign.  
Waveforms  
Examples  
Concurrent Assign.



Simple

```

1 process(all)
2 begin
3   o <= (a and not c) or
4       (b and c);
5 end process;

```

Conditional

```

1 process(all)
2 begin
3   o <= a when c = '0'
4       else b;
5 end process;

```

Selected

```

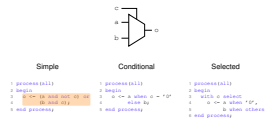
1 process(all)
2 begin
3   with c select
4     o <= a when '0',
5         b when others;
6 end process;

```

# Signal Assignments

## Assign. Statements

### Signal Assignment Statements - Multiplexer Example

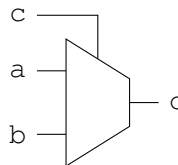


Using a simple assignment, the functionality of the multiplexer is directly described by a Boolean expression.

## Signal Assignment Statements - Multiplexer Example

HWMMod  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Simple Assign.  
Cond. Assign.  
Waveforms  
Examples  
Concurrent Assign.



Simple

```

1 process(all)
2 begin
3   o <= (a and not c) or
4     (b and c);
5 end process;

```

Conditional

```

1 process(all)
2 begin
3   o <= a when c = '0'
4     else b;
5 end process;

```

Selected

```

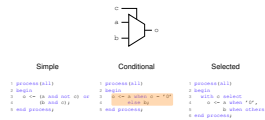
1 process(all)
2 begin
3   with c select
4     o <= a when '0',
5       b when others;
6 end process;

```

# Signal Assignments

## Assign. Statements

### Signal Assignment Statements - Multiplexer Example

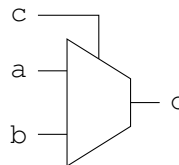


The conditional assignment allows to express the same behavior in a slightly more readable way, as we can simply list the different cases.

## Signal Assignment Statements - Multiplexer Example

HWMMod  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Simple Assign.  
Cond. Assign.  
Waveforms  
Examples  
Concurrent Assign.



Simple

```
1 process (all)
2 begin
3   o <= (a and not c) or
4     (b and c);
5 end process;
```

Conditional

```
1 process (all)
2 begin
3   o <= a when c = '0'
4     else b;
5 end process;
```

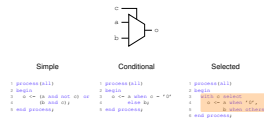
Selected

```
1 process (all)
2 begin
3   with c select
4     o <= a when '0',
5         b when others;
6 end process;
```

# Signal Assignments

## Assign. Statements

### Signal Assignment Statements - Multiplexer Example

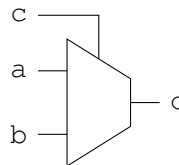


The last possibility is a selected assignment. Where you can think of a conditional assignment as an if-ELSE-if construct, embedded into an assignment statement, the selected variant embeds a case statement. For the sake of brevity we don't go into further details on this assignment class. However, we wanted to include it such that you have seen it once and that you know it exists.

## Signal Assignment Statements - Multiplexer Example

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Simple Assign.  
Cond. Assign.  
Waveforms  
Examples  
Concurrent Assign.



Simple

```

1 process (all)
2 begin
3   o <= (a and not c) or
4     (b and c);
5 end process;

```

Conditional

```

1 process (all)
2 begin
3   o <= a when c = '0'
4     else b;
5 end process;

```

Selected

```

1 process (all)
2 begin
3   with c select
4     o <= a when '0',
5     b when others;
6 end process;

```

Now, let us turn to the simple signal assignment and see what the VHDL reference manual has to say about it. According to the manual such an assignment can either be a simple waveform assignment or a force or release assignment.

## Simple Signal Assignments

### ■ Simple signal assignment syntax

```
simple_signal_assignment ::=
  simple_waveform_assignment
  | simple_force_assignment
  | simple_release_assignment
simple_waveform_assignment ::=
  target <= [ delay_mechanism ] waveform ;
```

Force assignments can be used to forcefully drive a signal to a certain value, overriding other drivers on that signal. The release assignment is used to disable a previous force assignment. However, these concepts go beyond the scope of this lecture and we, thus, don't go into further detail.

## Simple Signal Assignments

### ■ Simple signal assignment syntax

```
simple_signal_assignment ::=
    simple_waveform_assignment
    | simple_force_assignment
    | simple_release_assignment
simple_waveform_assignment ::=
    target <= [ delay_mechanism ] waveform ;
```

### ■ Force/release assignments

- Can be used to override signal drivers
- Not covered in this course

What we are interested in, is the simple waveform assignment.

## Simple Signal Assignments

### ■ Simple signal assignment syntax

```
simple_signal_assignment ::=
    simple_waveform_assignment
    | simple_force_assignment
    | simple_release_assignment
simple_waveform_assignment ::=
    target <= [ delay_mechanism ] waveform ;
```

### ■ Force/release assignments

- Can be used to override signal drivers
- Not covered in this course

### ■ Simple waveform assignments

As we already know, a signal assignment needs a target, which must obviously be a signal object.

## Simple Signal Assignments

### ■ Simple signal assignment syntax

```
simple_signal_assignment ::=
  simple_waveform_assignment
  | simple_force_assignment
  | simple_release_assignment
simple_waveform_assignment ::=
  target <= [ delay_mechanism ] waveform ;
```

### ■ Force/release assignments

- Can be used to override signal drivers
- Not covered in this course

### ■ Simple waveform assignments

- Target signal

- Simple signal assignment syntax
  - `simple_signal_assignment ::=`
  - `simple_waveform_assignment`
  - `| simple_force_assignment`
  - `| simple_release_assignment`
- Simple waveform assignment syntax
  - `simple_waveform_assignment ::=`
  - `target <= [ delay_mechanism ] waveform ;`
- Force/release assignments
  - Can be used to override signal drivers
  - Not covered in this course
- Simple waveform assignments
  - Target signal
  - Optional delay mechanism (covered in separate lecture)

After the assignment arrow symbol follows the optional specification of the delay mechanism. Here, the delay model that should be applied to the assignment can be defined. However, as this is a separate topic on its own, there exists a dedicated lecture on it. For this lecture, we simply don't use this language feature.

## Simple Signal Assignments

### ■ Simple signal assignment syntax

```
simple_signal_assignment ::=
    simple_waveform_assignment
    | simple_force_assignment
    | simple_release_assignment

simple_waveform_assignment ::=
    target <= [ delay_mechanism ] waveform ;
```

### ■ Force/release assignments

- Can be used to override signal drivers
- Not covered in this course

### ■ Simple waveform assignments

- Target signal
- Optional delay mechanism (covered in separate lecture)

Then, the actual waveform, that should be assigned to the target signal, is specified. From your experience with VHDL so far, you can probably infer that in the most basic case, a waveform can just be a VHDL expression. However, as we will see shortly, in the general case waveforms are a list of value expressions and associated delay values specifying the exact times at which an assignment shall take effect. Thus, waveforms allow it to specify multiple subsequent signal transitions.

## Simple Signal Assignments

### ■ Simple signal assignment syntax

```

simple_signal_assignment ::=
  simple_waveform_assignment
  | simple_force_assignment
  | simple_release_assignment ;
simple_waveform_assignment ::=
  target <= [ delay_mechanism ] waveform ;
  
```

### ■ Force/release assignments

- Can be used to override signal drivers
- Not covered in this course

### ■ Simple waveform assignments

- Target signal
- Optional delay mechanism (covered in separate lecture)
- Waveform (a series of expressions with timing information)

Before we look at the exact syntax specification for waveforms, let us have a quick look at the conditional signal assignment statement.

## Conditional Assignments

### ■ Conditional signal assignment syntax

```
conditional_signal_assignment ::=
    target <= [ delay_mechanism ] cond_waveforms ;

cond_waveforms ::=
    waveform when condition
    { else waveform when condition }
    [ else waveform ]
```

```
■ Conditional signal assignment syntax  
conditional_signal_assignment ::=  
    target <= [ delay_mechanism ] cond_waveforms ;  
  
cond_waveforms ::=  
    waveform when condition  
    { else waveform when condition }  
    [ else waveform ]  
  
■ Target  
■ Optional delay mechanism
```

As with the simple signal assignment we need a target and can specify an optional delay mechanism.

## Conditional Assignments

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Simple Assign.  
**Cond. Assign.**  
Waveforms  
Examples  
Concurrent Assign.

### ■ Conditional signal assignment syntax

```
conditional_signal_assignment ::=
```

```
target <= [ delay_mechanism ] cond_waveforms ;
```

```
cond_waveforms ::=
```

```
    waveform when condition
```

```
    { else waveform when condition }
```

```
    [ else waveform ]
```

### ■ Target

### ■ Optional delay mechanism

```
■ Conditional signal assignment syntax
conditional_signal_assignment ::=
  target <= [ delay_mechanism ] cond_waveforms ;

cond_waveforms ::=
  waveform when condition
  { else waveform when condition }
  [ else waveform ]

■ Target
■ Optional delay mechanism
■ Each of the expressions that are assignment are waveforms
```

You can see that in the conditional waveforms block, each of the values that are assigned in a particular case are waveforms.

## Conditional Assignments

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Simple Assign.  
Cond. Assign.  
Waveforms  
Examples  
Concurrent Assign.

### ■ Conditional signal assignment syntax

```
conditional_signal_assignment ::=
  target <= [ delay_mechanism ] cond_waveforms ;

cond_waveforms ::=
  waveform when condition
  { else waveform when condition }
  [ else waveform ]
```

- Target
- Optional delay mechanism
- Each of the expressions that are assignment are waveforms

At this point we can finally discuss the waveform itself, which is the actual core topic of this lecture. As you can see from the syntax specification, there are two basic cases of what a waveform can be.

## Waveforms

HWMod  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
**Waveforms**  
Examples  
Concurrent Assign.

### ■ Waveform syntax

```
waveform ::=
  waveform_element { , waveform_element }
  | unaffected
```

The first case is a comma-separated list of one or more waveform elements, which will be discussed in detail on the next slide.

## Waveforms

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
**Waveforms**  
Examples  
Concurrent Assign.

### ■ Waveform syntax

```
waveform ::=
```

```
  waveform_element { , waveform_element }
```

```
  | unaffected
```

### ■ Comma-separated list of waveform elements

The other case is the keyword "unaffected", which is basically used to indicate a no-operation.

## Waveforms

HWMod  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
**Waveforms**  
Examples  
Concurrent Assign.

- Waveform syntax

```
waveform ::=
    waveform_element { , waveform_element }
    | unaffected
```

- Comma-separated list of waveform elements
- Special case: `unaffected`

```
■ Waveform syntax
  waveform ::=
    waveform_element { , waveform_element }
    | unaffected
■ Comma-separated list of waveform elements
■ Special case: unaffected
  ■ s <= unaffected; ⇔ null;
```

Assigning `unaffected` to a signal is equivalent to executing a `null` statement. Hence, as you can see, such direct assignment to `unaffected` are largely pointless.

## Waveforms

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Concurrent Assign.

### ■ Waveform syntax

```
waveform ::=
    waveform_element { , waveform_element }
    | unaffected
```

### ■ Comma-separated list of waveform elements

### ■ Special case: `unaffected`

■ `s <= unaffected;` ⇔ `null;`

```

■ Waveform syntax
  waveform :=
    waveform_element { , waveform_element }
    | unaffected
■ Comma-separated list of waveform elements
■ Special case: unaffected
  ■ s <= unaffected; ⇔ null;
  ■ s <= unaffected when c else '0'; ⇔ if not c then
                                         s <= '0';
                                         end if;

```

However, when employed in a conditional assignment, they can be used to disable one branch of the when-ELSE-when construct.

## Waveforms

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Concurrent Assign.

### ■ Waveform syntax

```

waveform ::=
    waveform_element { , waveform_element }
    | unaffected

```

### ■ Comma-separated list of waveform elements

### ■ Special case: unaffected

```

■ s <= unaffected; ⇔ null;
■ s <= unaffected when c else '0'; ⇔ if not c then
                                         s <= '0';
                                         end if;

```

For the waveform element itself we again have two possibilities.

## Waveforms (cont'd)

### ■ Waveform element syntax

```
waveform_element ::=
    value_expression [ after time_expression ]
| null [ after time_expression ]
```

The first, and arguably more important one, is given by a value expression followed by an optional `after` clause containing a time expression. The value expression can be any VHDL expression that evaluates to the type of the target signal of the assignment.

## Waveforms (cont'd)

- Waveform element syntax

```
waveform_element ::=
  value_expression [ after time_expression ]
  | null [ after time_expression ]
```

- *Value expression* evaluating to the type of the assignment target

- Waveform element syntax

```
waveform_element ::=  
  value_expression [after time_expression ]  
  | null [after time_expression ]
```
- Value expression evaluating to the type of the assignment target
- Optional time expression
  - Implicit default: `after 0 ns`
  - Must not evaluate to a negative `time` value
  - Must not decrease in succeeding waveform elements

The optional time expression introduced with the `after` keyword, specifies when the assignment of the value produced by the associated value expression should take effect. At this point you might already be able to see how multiple waveform elements can be used to describe complex waveforms. In any case, we will look at several examples over the remainder of this lecture to show you how this works. In the case of a value expressions without an `after` clause the VHDL standard defines an implicit `after` with a time of 0 nanoseconds. Another important point is that the time expression must not evaluate to a negative value, since it is not possible to specify events that happened in the past. Moreover, it must be ensured that the time expression values of consecutive waveform elements increase.

## Waveforms (cont'd)

### ■ Waveform element syntax

```
waveform_element ::=  
  value_expression [ after time_expression ]  
  | null [ after time_expression ]
```

### ■ Value expression evaluating to the type of the assignment target

### ■ Optional *time expression*

- Implicit default: `after 0 ns`
- Must not evaluate to a negative `time` value
- Must not decrease in succeeding waveform elements

- Waveform element syntax

```
waveform_element ::=
  value_expression [ after time_expression ]
  | null [ after time_expression ]
```
- Value expression evaluating to the type of the assignment target
- Optional time expression
  - Implicit default: `after 0 ns`
  - Must not evaluate to a negative `time` value
  - Must not decrease in succeeding waveform elements
- `null` waveform elements
  - Used to turn-off drivers to a signal
  - Not needed or covered in this course

Finally, a waveform element can also be `null`. This special case is used to completely turn off the driver to a signal. However, since this feature goes beyond the scope of this course, we will not cover it in any more detail. The only thing you should note is that an assignment to `unaffected` is not the same as a `null` assignment.

## Waveforms (cont'd)

### ■ Waveform element syntax

```
waveform_element ::=
  value_expression [ after time_expression ]
  | null [ after time_expression ]
```

### ■ *Value expression* evaluating to the type of the assignment target

### ■ Optional *time expression*

- Implicit default: `after 0 ns`
- Must not evaluate to a negative `time` value
- Must not decrease in succeeding waveform elements

### ■ `null` waveform elements

- Used to turn-off drivers to a signal
- Not needed or covered in this course

This slides shows a few examples of how waveforms can look like in simple as well as conditional assignments. You may want to convince yourself that these code snippets indeed adhere to the presented grammar rules.

## Waveforms (cont'd)

### ■ Simple waveform assignment examples

- `s <= t;`
- `s <= '0' after 1 ns;`
- `s <= not s after 1 ns;`
- `s <= '1' after 1 ns, '0' after 2 ns, '1' after 3 ns;`

### ■ Conditional signals assignment examples

- `s <= '1' after 1 ns when c = '1'`  
`'0' after 2 ns when c = '0'`  
`else unaffected;`
- `s <= 42 after 1 ns, 1 after 42 ns when c = '1'`  
`else t;`

# Signal Assignments

## Examples

### Example 1

```
1 architecture arch of sim01 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     s <= '1';
6   end process;
7 end arch;
8 end architecture;
```

Now that we have sufficiently covered the syntax of signal assignments and waveforms, let us turn to their semantics. We will do that by the means of concrete examples, for which we will show how they are evaluated by the simulator on a step-by-step basis. By the end of the lecture you should have acquired a "feeling" for the semantics of signal assignments, which you can deepen and extend with the quizzes in TUWEL. Before we go to more complex examples, let us first work through a basic assignment without any `after` clauses in order to establish some notation and fundamental principles. From your knowledge about VHDL it should be completely clear what happens in this architecture. The signal `s` is set to high at simulation time zero and stays at this value. However, how does the simulation actually arrive at this conclusion and what internal data structures are needed for that?

## Example 1

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
Example 4  
Example 5  
Concurrent Assign.

```
1 architecture arch of sim01 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1';
7       wait;
8     end process;
9 end architecture;
```

# Signal Assignments

## Examples

### Example 1

```

1 architecture arch of sim1 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     s <= '1';
6     wait;
7   end process;
8 end architecture;

```

current simulation time: 0 ns  
 scheduled processes:  
 active processes:  
 completed processes:  
 signal value scheduled events

Internally the simulator not only stores the current value of every signal in a design but also keeps a list of events scheduled for the future. These can be seen in the table on the right side of the slide. Whenever a signal assignment statement is executed, only this list is changed and **not** the actual value. Variables, on the other hand, don't have event lists as their value is always assigned immediately.

## Example 1

HWMoD  
WS25

- Signal Assign.
- Introduction
- Assign. Statements
- Waveforms
- Examples
- Example 1**
- Example 2
- Example 3
- Example 4
- Example 5
- Concurrent Assign.

```

1 architecture arch of sim01 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1';
7       wait;
8     end process;
9 end architecture;

```



current simulation time: 0 ns  
 scheduled processes:  
 active processes:  
 completed processes:

signal	value	scheduled events
s	--	--

# Signal Assignments

## Examples

### Example 1

```
1 architecture arch of sim1 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     s <= '1';
6     wait;
7   end process;
8 end architecture;
```

current simulation time: 0 ns  
scheduled processes: p0  
active processes:  
completed processes:  
signal value scheduled events  
s '0' --

During initialization of the simulator, all the initial values of all signals in a design are derived from their declarations. Hence, in this example `s` is set to the value zero. Moreover, all processes in the design are scheduled for execution.

## Example 1

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
Example 4  
Example 5  
Concurrent Assign.

```
1 architecture arch of sim01 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1';
7       wait;
8     end process;
9 end architecture;
```

s | |

current simulation time: 0 ns  
scheduled processes: p0  
active processes:  
completed processes:

signal	value	scheduled events
s	'0'	--

# Signal Assignments

## Examples

### Example 1

```
1 architecture arch of sim1 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     s <= '1';
6     wait;
7   end process;
8 end architecture;
```

current simulation time: 0 ns  
scheduled processes:  
active processes: p0  
completed processes:  
signal value scheduled events  
s '0' --

Then the simulation starts to run the processes. As there is only one in our simple example, p0 is executed.

## Example 1

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
Example 4  
Example 5  
Concurrent Assign.

```
1 architecture arch of sim01 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1';
7       wait;
8     end process;
9 end architecture;
```

s | |

current simulation time: 0 ns  
scheduled processes:  
active processes: p0  
completed processes:

signal	value	scheduled events
s	'0'	--

# Signal Assignments

## Examples

### Example 1

```
1 architecture arch of sim1 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1';
7       wait;
8     end process;
9 end architecture;
```

current simulation time: 0 ns  
scheduled processes:  
active processes: p0  
completed processes:

signal	value	scheduled events
s	'0'	'1' @ 0 ns

When the simulator reaches the assignment statement it adds the value one to the event queue of the signal `s` and schedules it to run at simulation time 0 nanoseconds.

## Example 1

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
Example 4  
Example 5  
Concurrent Assign.

```
1 architecture arch of sim01 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1';
7       wait;
8     end process;
9 end architecture;
```

s | |

current simulation time: 0 ns  
scheduled processes:  
active processes: p0  
completed processes:

signal	value	scheduled events
s	'0'	'1' @ 0 ns

# Signal Assignments

## Examples

### Example 1

```
1 architecture arch of sim1 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     s <= '1';
6     wait;
7   end process;
8 end architecture;
```

current simulation time: 0 ns  
scheduled processes:  
active processes:  
completed processes: p0

signal	value	scheduled events
s	'0'	'1' @ 0 ns

Next, since the simulation ran into an unconditional `wait` statement it marks the process as completed and will never execute it again. Afterwards, the simulation goes through all the event lists associated with the process and checks whether there are pending assignments.

## Example 1

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
Example 4  
Example 5  
Concurrent Assign.

```
1 architecture arch of sim01 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1';
7       wait;
8     end process;
9 end architecture;
```

s | |

current simulation time: 0 ns  
scheduled processes:  
active processes:  
completed processes: p0

signal	value	scheduled events
s	'0'	'1' @ 0 ns

# Signal Assignments

## Examples

### Example 1

```

1 architecture arch of sim1 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     s <= '1';
6     wait;
7     end process;
8 end architecture;

```

current simulation time: 0 ns  
 scheduled processes:  
 active processes:  
 completed processes: p0

---

signal value scheduled events  
 s '1' --

As `s` is scheduled to transition to one at 0 nanoseconds, the assignment is popped from the event queue and executed. Note that the simulation time is still 0 nanoseconds so far, as none of the encountered statements did consume any "physical" time.

## Example 1

HWMoD  
WS25

Signal Assign.  
 Introduction  
 Assign. Statements  
 Waveforms  
 Examples  
**Example 1**  
 Example 2  
 Example 3  
 Example 4  
 Example 5  
 Concurrent Assign.

```

1 architecture arch of sim01 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1';
7       wait;
8     end process;
9 end architecture;
```



current simulation time: 0 ns  
 scheduled processes:  
 active processes:  
 completed processes: p0

signal	value	scheduled events
s	'1'	--

# Signal Assignments

## Examples

### Example 1

```

1 architecture arch of sim01 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     s <= '1';
6     wait;
7   end process;
8 end architecture;

```

current simulation time: 0 ns  
 scheduled processes:  
 active processes:  
 completed processes: p0

signal	value	scheduled events
s	'1'	--

Simulation End  
 No scheduled events or processes → no further signal transitions possible.

The simulator now recognizes that there are neither scheduled events for any signal, nor any scheduled processes to run. Thus, it concludes that no further signal transitions are possible. Hence, the simulation time does not need to be advanced anymore.

## Example 1

HWMoD  
WS25

Signal Assign.  
 Introduction  
 Assign. Statements  
 Waveforms  
 Examples  
**Example 1**  
 Example 2  
 Example 3  
 Example 4  
 Example 5  
 Concurrent Assign.

```

1 architecture arch of sim01 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1';
7       wait;
8     end process;
9 end architecture;
```



current simulation time: 0 ns  
 scheduled processes:  
 active processes:  
 completed processes: p0

signal	value	scheduled events
s	'1'	--

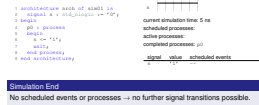
### Simulation End

No scheduled events or processes → no further signal transitions possible.

# Signal Assignments

## Examples

### Example 1



If we would still forcefully simulate beyond this point we would see in the timing diagram that, indeed, no further signal transitions happen. Before we continue to the next example, we want to point out that the explanations in this lecture use a slight simplification. The simulation actually maintains an event list for every driver of a signal. Hence, if there are two processes writing to the same signal there are also two event lists.

## Example 1

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
Example 4  
Example 5  
Concurrent Assign.

```

1 architecture arch of sim01 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1';
7       wait;
8     end process;
9 end architecture;
```



current simulation time: 5 ns  
scheduled processes: --  
active processes: p0  
completed processes: p0

signal	value	scheduled events
s	'1'	--

### Simulation End

No scheduled events or processes → no further signal transitions possible.

# Signal Assignments

## Examples

### Example 2

```
1 architecture arch of sim02 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 1 ns;
7     end
8   end process;
9 end architecture;
```

Let's now look at a slightly more interesting example. Here we have an architecture that looks very similar to the one from the previous slide. The only difference is that the waveform now uses an `after` clause. The semantic of this construct is not hard to guess. The signal assignment should not happen at simulation time 0 but after 1 nanosecond. Hence, we would expect a timing diagram where `s` is low for 1 nanosecond, then transitions to high and stays there.

## Example 2

HWMoD  
WS25

Signal Assign.

Introduction

Assign. Statements

Waveforms

Examples

Example 1

**Example 2**

Example 3

Example 4

Example 5

Concurrent Assign.

```
1 architecture arch of sim02 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 1 ns;
7     wait;
8   end process;
9 end architecture;
```

# Signal Assignments

## Examples

### Example 2

```
1 architecture arch of sim2 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     s <= '1' after 1 ns;
6   wait;
7   end process;
8 end architecture;
```

current simulation time: 0 ns  
scheduled processes: p0  
active processes:  
completed processes:  
signal value scheduled events

The initial state of the simulator is the same as for the previous example.

## Example 2

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
Example 4  
Example 5  
Concurrent Assign.

```
1 architecture arch of sim02 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 1 ns;
7       wait;
8     end process;
9 end architecture;
```



current simulation time: 0 ns  
scheduled processes: p0  
active processes:  
completed processes:

signal	value	scheduled events
s	'0'	--

# Signal Assignments

## Examples

### Example 2

```

1 architecture arch of sim02 is
2   signal s : std_ulogic := '0';
3   begin
4     p0 : process
5       begin
6         s <= '1' after 1 ns;
7       end process;
8   end architecture;

```

current simulation time: 0 ns  
 scheduled processes:  
 active processes: p0  
 completed processes:

signal	value	scheduled events
s	'0'	'1' @ 1 ns

As before, the simulator then executes the single process p0. When reaching the assignment statement, the simulator schedules an appropriate event. However, compared to the previous example, the simulator now schedules this event not for 0 but rather for 1 nanosecond simulation time. The reason is of course the 1 nanosecond `after` clause.

## Example 2

HWMoD  
WS25

Signal Assign.  
 Introduction  
 Assign. Statements  
 Waveforms  
 Examples  
 Example 1  
**Example 2**  
 Example 3  
 Example 4  
 Example 5  
 Concurrent Assign.

```

1 architecture arch of sim02 is
2   signal s : std_ulogic := '0';
3   begin
4     p0 : process
5       begin
6         s <= '1' after 1 ns;
7       wait;
8     end process;
9 end architecture;

```



current simulation time: 0 ns  
 scheduled processes:  
 active processes: p0  
 completed processes:

signal	value	scheduled events
s	'0'	'1' @ 1 ns

# Signal Assignments

## Examples

### Example 2

```

1 architecture arch of sim02 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     s <= '1' after 1 ns;
6   end process;
7   wait;
8 end architecture;

```

current simulation time: 0 ns  
 scheduled processes:  
 active processes:  
 completed processes: p0

signal	value	scheduled events
s	'0'	'1' @ 1 ns

At the `wait` statement, the process is marked as complete but no actual value update to the signal `s` takes place.

## Example 2

HWMoD  
WS25

- Signal Assign.
- Introduction
- Assign. Statements
- Waveforms
- Examples
  - Example 1
  - Example 2**
  - Example 3
  - Example 4
  - Example 5
- Concurrent Assign.

```

1 architecture arch of sim02 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 1 ns;
7       wait;
8     end process;
9 end architecture;

```



current simulation time: 0 ns  
 scheduled processes:  
 active processes:  
 completed processes: p0

signal	value	scheduled events
s	'0'	'1' @ 1 ns

```

1 architecture arch of sim02 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     s <= '1' after 1 ns;
6   wait;
7   end process;
8 end architecture;
  
```

current simulation time: 1 ns  
 scheduled processes:  
 active processes:  
 completed processes: p0

signal	value	scheduled events
s	'0'	'1' @ 1 ns

The simulator then goes through all the scheduled events and advances the simulation time to the point in time of the next event.

## Example 2

HWMoD  
 WS25

Signal Assign.  
 Introduction  
 Assign. Statements  
 Waveforms  
 Examples  
 Example 1  
**Example 2**  
 Example 3  
 Example 4  
 Example 5  
 Concurrent Assign.

```

1 architecture arch of sim02 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 1 ns;
7       wait;
8     end process;
9 end architecture;
  
```



current simulation time: 1 ns  
 scheduled processes:  
 active processes:  
 completed processes: p0

signal	value	scheduled events
s	'0'	'1' @ 1 ns

# Signal Assignments

## Examples

### Example 2

```
1 architecture arch of sim02 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     s <= '1' after 1 ns;
6   wait
7   end process;
8 end architecture;
```

current simulation time: 1 ns  
scheduled processes:  
active processes:  
completed processes: p0

signal	value	scheduled events
s	'1'	--

Then the value assignment is executed.

## Example 2

HWMoD  
WS25

- Signal Assign.
- Introduction
- Assign. Statements
- Waveforms
- Examples
- Example 1
- Example 2**
- Example 3
- Example 4
- Example 5
- Concurrent Assign.

```
1 architecture arch of sim02 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 1 ns;
7       wait;
8     end process;
9 end architecture;
```



current simulation time: 1 ns  
scheduled processes:  
active processes:  
completed processes: p0

signal	value	scheduled events
s	'1'	--

# Signal Assignments

## Examples

### Example 2

```
1 architecture arch of sim02 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     s <= '1' after 1 ns;
6   wait;
7   end process;
8 end architecture;
```



current simulation time: 5 ns  
scheduled processes:  
active processes:  
completed processes: p0

signal	value	scheduled events
s	'1'	--

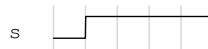
Again, there are no further events or scheduled processes and hence no more signal changes. The simulation thus terminates, and we can go to the next example.

## Example 2

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
Example 4  
Example 5  
Concurrent Assign.

```
1 architecture arch of sim02 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 1 ns;
7       wait;
8     end process;
9 end architecture;
```



current simulation time: 5 ns  
scheduled processes:  
active processes:  
completed processes: p0

signal	value	scheduled events
s	'1'	--

# Signal Assignments

## Examples

### Example 3

```
1 architecture arch of sim3 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 1 ns;
7       s <= '0' after 2 ns;
8     end process;
9 end architecture;
```

Again, the code is very similar, but this time the signal `s` should transition back to 0 after 2 nanoseconds.

## Example 3

HWMoD  
WS25

Signal Assign.

Introduction

Assign. Statements

Waveforms

Examples

Example 1

Example 2

**Example 3**

Example 4

Example 5

Concurrent Assign.

```
1 architecture arch of sim03 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 1 ns,
7         '0' after 2 ns;
8     wait;
9   end process;
10 end architecture;
```

# Signal Assignments

## Examples

### Example 3

```

1 architecture arch of sim3 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     s <= '1' after 1 ns;
6     wait
7     '0' after 2 ns;
8   end process;
9 end architecture;

```

current simulation time: 0 ns  
scheduled processes: p0  
active processes:  
completed processes:  
signal value scheduled events

The initialization is again the same as in the previous examples.

## Example 3

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
Example 4  
Example 5  
Concurrent Assign.

```

1 architecture arch of sim03 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 1 ns,
7         '0' after 2 ns;
8       wait;
9     end process;
10 end architecture;

```



current simulation time: 0 ns  
scheduled processes: p0  
active processes:  
completed processes:

signal	value	scheduled events
s	'0'	--

# Signal Assignments

## Examples

### Example 3

```

1 architecture arch of sim3 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     wait;
6     s <= '1' after 1 ns;
7     wait;
8     s <= '0' after 2 ns;
9   end process;
10 end architecture;

```

current simulation time: 0 ns  
scheduled processes:  
active processes: p0  
completed processes:

signal	value	scheduled events
s	'0'	'1' @ 1 ns, '0' @ 2 ns

Executing the assignment statement now schedules 2 events for the signal s.

## Example 3

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
**Example 3**  
Example 4  
Example 5  
Concurrent Assign.

```

1 architecture arch of sim03 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 1 ns,
7         '0' after 2 ns;
8     wait;
9   end process;
10 end architecture;

```



current simulation time: 0 ns  
scheduled processes:  
active processes: p0  
completed processes:

signal	value	scheduled events
s	'0'	'1' @ 1 ns, '0' @ 2 ns

# Signal Assignments

## Examples

### Example 3

```

1 architecture arch of sim03 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     wait
6     s <= '1' after 1 ns;
7     s <= '0' after 2 ns;
8   end process;
9 end architecture;

```

current simulation time: 0 ns  
scheduled processes:  
active processes:  
completed processes: p0

signal	value	scheduled events
s	'0'	'1' @ 1 ns, '0' @ 2 ns

At the `wait` statement the simulator marks the process as complete and

## Example 3

HWMoD  
WS25

### Signal Assign.

- Introduction
- Assign. Statements
- Waveforms
- Examples
- Example 1
- Example 2
- Example 3**
- Example 4
- Example 5
- Concurrent Assign.

```

1 architecture arch of sim03 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 1 ns;
7       s <= '0' after 2 ns;
8     wait;
9   end process;
10 end architecture;

```



current simulation time: 0 ns  
scheduled processes:  
active processes:  
completed processes: p0

signal	value	scheduled events
s	'0'	'1' @ 1 ns, '0' @ 2 ns

# Signal Assignments

## Examples

### Example 3

```

1 architecture arch of sim03 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     wait
6     s <= '1' after 1 ns;
7     s <= '0' after 2 ns;
8   end wait;
9   wait
10  end process;
11 end architecture;

```

current simulation time: 1 ns  
 scheduled processes:  
 active processes:  
 completed processes: p0  
 signal value scheduled events

then advances the simulation time to the next scheduled event, which is at 1 nanosecond.

## Example 3

HWMoD  
WS25

Signal Assignn.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
Example 4  
Example 5  
Concurrent Assignn.

```

1 architecture arch of sim03 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 1 ns,
7         '0' after 2 ns;
8     wait;
9   end process;
10 end architecture;

```



current simulation time: 1 ns  
 scheduled processes:  
 active processes:  
 completed processes: p0

signal	value	scheduled events
s	'0'	'1' @ 1 ns, '0' @ 2 ns

# Signal Assignments

## Examples

### Example 3

```
1 architecture arch of sim3 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     wait
6     s <= '1' after 1 ns;
7     s <= '0' after 2 ns;
8   wait;
9   end process;
10 end architecture;
```

current simulation time: 1 ns  
scheduled processes:  
active processes:  
completed processes: p0

signal	value	scheduled events
s	'1'	'0' @ 2 ns

After executing the assignment the simulation again looks for the next scheduled event,

## Example 3

HWMoD  
WS25

Signal Assign.

Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
Example 4  
Example 5  
Concurrent Assign.

```
1 architecture arch of sim03 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 1 ns;
7       '0' after 2 ns;
8     wait;
9     end process;
10 end architecture;
```

s 

current simulation time: 1 ns  
scheduled processes:  
active processes:  
completed processes: p0

signal	value	scheduled events
s	'1'	'0' @ 2 ns

```

1 architecture arch of sim3 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     wait
6     s <= '1' after 1 ns;
7     wait
8     s <= '0' after 2 ns;
9   end process;
10 end architecture;

```

current simulation time: 2 ns  
scheduled processes:  
active processes:  
completed processes: p0

signal	value	scheduled events
s	'0'	--

advances the simulation time accordingly and executes it.

## Example 3

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
**Example 3**  
Example 4  
Example 5  
Concurrent Assign.

```

1 architecture arch of sim03 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 1 ns;
7       '0' after 2 ns;
8       wait;
9     end process;
10 end architecture;

```



current simulation time: 2 ns  
scheduled processes:  
active processes:  
completed processes: p0

signal	value	scheduled events
s	'0'	--

# Signal Assignments

## Examples

### Example 3

```

1 architecture arch of sim03 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     wait
6     s <= '1' after 1 ns;
7     s <= '0' after 2 ns;
8   wait;
9   end process;
10 end architecture;

```



```

current simulation time: 5 ns
scheduled processes:
active processes:
completed processes: p0
signal      value  scheduled events
s           '0'   --

```

Since, there are no scheduled events in the event queue, the simulation ends here.

## Example 3

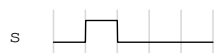
HWMoD  
WS25

- Signal Assign.
- Introduction
- Assign. Statements
- Waveforms
- Examples
- Example 1
- Example 2
- Example 3**
- Example 4
- Example 5
- Concurrent Assign.

```

1 architecture arch of sim03 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 1 ns;
7       '0' after 2 ns;
8     wait;
9     end process;
10 end architecture;

```



current simulation time: 5 ns  
 scheduled processes:  
 active processes:  
 completed processes: p0

signal	value	scheduled events
s	'0'	--

# Signal Assignments

## Examples

### Example 4

```
1 architecture arch of sim4 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns;
7       t <= '0' after 4 ns;
8       t <= '1' after 6 ns;
9     end process;
10  p1 : process
11    begin
12      t <= s after 1 ns;
13    end process;
14  end process;
15 end architecture;
```

In this example, we want to show how multiple processes are handled and how the sensitivity list is used to schedule new process executions.

## Example 4

HWMod  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
Example 4  
Example 5  
Concurrent Assign.

```
1 architecture arch of sim04 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns;
9     wait;
10    end process;
11
12   p1 : process(all)
13     begin
14       t <= s after 1 ns;
15     end process;
16 end architecture;
```

# Signal Assignments

## Examples

### Example 4

```

1 architecture arch of sim4 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns;
7       t <= '1' after 4 ns;
8       wait
9     end process;
10  p1 : process(all)
11    begin
12      t <= s after 1 ns;
13    end process;
14 end architecture;

```

signal	value	scheduled events
s	'1'	--
t	'0'	--

In the beginning the signals *s* and *t* are set to 0, and both processes are scheduled for execution.

## Example 4

HWMoD  
WS25

- Signal Assign.
- Introduction
- Assign. Statements
- Waveforms
- Examples
- Example 1
- Example 2
- Example 3
- Example 4**
- Example 5
- Concurrent Assign.

```

1 architecture arch of sim04 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns;
9       wait;
10    end process;
11
12   p1 : process(all)
13     begin
14       t <= s after 1 ns;
15     end process;
16 end architecture;

```



current simulation time: 0 ns  
 scheduled processes: p0, p1  
 active processes:  
 completed processes:

signal	value	scheduled events
s	'0'	--
t	'0'	--

# Signal Assignments

## Examples

### Example 4

```

1 architecture arch of sim4 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     wait;
6     s <= '1' after 2 ns;
7     t <= '0' after 4 ns;
8     s <= '1' after 6 ns;
9   end process;
10
11   p1 : process(all)
12     t <= s after 1 ns;
13   end process;
14 end architecture;

```

current simulation time: 0 ns  
 scheduled processes: p1  
 active processes: p0  
 completed processes:

signal	value	scheduled events
s	'0'	'1' @ 2 ns, '0' @ 4 ns, '1' @ 6 ns
t	'0'	--

Let's say process p0 is run first. Executing the assignment statement schedules three events for the signal s.

## Example 4

HWMoD  
WS25

Signal Assignn.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
**Example 4**  
Example 5  
Concurrent Assignn.

```

1 architecture arch of sim04 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns;
9     wait;
10    end process;
11
12    p1 : process(all)
13      begin
14        t <= s after 1 ns;
15      end process;
16 end architecture;

```



current simulation time: 0 ns  
 scheduled processes: p1  
 active processes: p0  
 completed processes:

signal	value	scheduled events
s	'0'	'1' @ 2 ns, '0' @ 4 ns, '1' @ 6 ns
t	'0'	--

# Signal Assignments

## Examples

### Example 4

```

1 architecture arch of sim4 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns;
7       t <= '1' after 4 ns;
8       wait
9     end process;
10
11  p1 : process(all)
12    begin
13      t <= s after 1 ns;
14    end process;
15  end process;
16 end architecture;

```

signal	value	scheduled events
s	'0'	'1' @ 2 ns,
s	'0'	'0' @ 4 ns,
t	'0'	'1' @ 6 ns

Then the unconditional `wait` statement terminates the process `p0`.

## Example 4

HWMoD  
WS25

- Signal Assign.
- Introduction
- Assign. Statements
- Waveforms
- Examples
- Example 1
- Example 2
- Example 3
- Example 4**
- Example 5
- Concurrent Assign.

```

1 architecture arch of sim04 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns;
9       wait;
10    end process;
11
12   p1 : process(all)
13     begin
14       t <= s after 1 ns;
15     end process;
16 end architecture;

```



current simulation time: 0 ns  
 scheduled processes: p1  
 active processes:  
 completed processes: p0

signal	value	scheduled events
s	'0'	'1' @ 2 ns, '0' @ 4 ns,
s	'0'	'1' @ 6 ns
t	'0'	--

# Signal Assignments

## Examples

### Example 4

```

1 architecture arch of sim4 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     wait;
6     s <= '1' after 2 ns;
7     t <= '1' after 4 ns;
8     t <= '1' after 6 ns;
9   end process;
10  wait;
11  p1 : process(all)
12  begin
13    t <= s after 1 ns;
14  end process;
15  end process;
16 end architecture;

```

signal	value	scheduled events
s	'0'	'1' @ 2 ns, '0' @ 4 ns,
t	'0'	'1' @ 6 ns
		'0' @ 1 ns

Since, there are still processes scheduled, the simulator does not yet advance the simulation time, but executes p1 first. The single assignment statement in p1 specifies that the signal t shall get the current value of s after 1 nanosecond. Hence, an appropriate event is scheduled for simulation time 1 nanosecond and the process execution is completed.

## Example 4

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
Example 4  
Example 5  
Concurrent Assign.

```

1 architecture arch of sim04 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns;
9     wait;
10    end process;
11
12    p1 : process(all)
13    begin
14      t <= s after 1 ns;
15    end process;
16 end architecture;

```



current simulation time: 0 ns  
scheduled processes:  
active processes: p1  
completed processes: p0

signal	value	scheduled events
s	'0'	'1' @ 2 ns, '0' @ 4 ns,
t	'0'	'1' @ 6 ns
		'0' @ 1 ns

# Signal Assignments

## Examples

### Example 4

```

1 architecture arch of sim4 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     wait;
6     s <= '1' after 2 ns;
7     t <= '1' after 4 ns;
8     wait;
9     s <= '0' after 4 ns;
10    wait;
11  end process;
12  p1 : process(all)
13    begin
14      t <= s after 1 ns;
15    end process;
16 end architecture;

```

Next, the simulation time advances to the next event and the associated assignment is executed. However, since the value of signal `t` is already 0, nothing changes.

## Example 4

- HWMoD
- WS25
- Signal Assign.
- Introduction
- Assign. Statements
- Waveforms
- Examples
- Example 1
- Example 2
- Example 3
- Example 4**
- Example 5
- Concurrent Assign.

```

1 architecture arch of sim04 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns;
9       wait;
10    end process;
11
12   p1 : process(all)
13     begin
14       t <= s after 1 ns;
15     end process;
16 end architecture;

```



current simulation time: 1 ns  
 scheduled processes:  
 active processes:  
 completed processes: p0, p1

signal	value	scheduled events
s	'0'	'1' @ 2 ns, '0' @ 4 ns,
		'1' @ 6 ns
t	'0'	--

# Signal Assignments

## Examples

### Example 4

```

1 architecture arch of sim4 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     wait
6     s <= '1' after 2 ns;
7     ...
8     t <= s after 1 ns;
9   end process;
10  p1 : process(all)
11    t <= s after 1 ns;
12  end process;
13 end architecture;

```

signal	value	scheduled events
s	'1'	'0' @ 4 ns, '1' @ 6 ns
t	'0'	--

The next scheduled event is at 2 nanoseconds and sets the signal `s` to high. Notice that since `s` is in the sensitivity list of process `p1`, and it changed its value, the process needs to be rescheduled for execution.

## Example 4

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
**Example 4**  
Example 5  
Concurrent Assign.

```

1 architecture arch of sim04 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns;
9     wait;
10    end process;
11
12    p1 : process(all)
13      begin
14        t <= s after 1 ns;
15      end process;
16 end architecture;

```



current simulation time: 2 ns  
scheduled processes: p1  
active processes:  
completed processes: p0

signal	value	scheduled events
s	'1'	'0' @ 4 ns, '1' @ 6 ns
t	'0'	--

# Signal Assignments

## Examples

### Example 4

```

1 architecture arch of sim4 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns;
7       '0' after 4 ns;
8       '1' after 6 ns;
9     end process;
10
11   wait;
12
13   p1 : process(all)
14     begin
15       t <= s after 1 ns;
16     end process;
17 end architecture;

```

signal	value	scheduled events
s	'1'	'0' @ 4 ns, '1' @ 6 ns
t	'0'	'1' @ 3 ns

Thus, in the next step p1 is executed again, which in turn schedules an event for the signal t at simulation time 3 nanoseconds.

## Example 4

HWMoD  
WS25

- Signal Assign.
- Introduction
- Assign. Statements
- Waveforms
- Examples
- Example 1
- Example 2
- Example 3
- Example 4**
- Example 5
- Concurrent Assign.

```

1 architecture arch of sim04 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns;
9     wait;
10    end process;
11
12   p1 : process(all)
13     begin
14       t <= s after 1 ns;
15     end process;
16 end architecture;

```



current simulation time: 2 ns  
 scheduled processes:  
 active processes: p1  
 completed processes: p0

signal	value	scheduled events
s	'1'	'0' @ 4 ns, '1' @ 6 ns
t	'0'	'1' @ 3 ns

# Signal Assignments

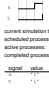
## Examples

### Example 4

```

1 architecture arch of sim4 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     wait;
6     s <= '1' after 2 ns;
7     t <= '1' after 4 ns;
8     wait;
9     s <= '0' after 4 ns;
10    wait;
11  end process;
12 p1 : process(all)
13   t <= s after 1 ns;
14 end process;
15 end process;
16 end architecture;

```



After advancing the simulation time this assignment is actually executed and `t` is set to 1.

## Example 4

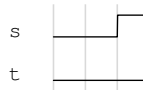
HWMoD  
WS25

- Signal Assign.
- Introduction
- Assign. Statements
- Waveforms
- Examples
- Example 1
- Example 2
- Example 3
- Example 4**
- Example 5
- Concurrent Assign.

```

1 architecture arch of sim04 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns;
9       wait;
10    end process;
11
12   p1 : process(all)
13     begin
14       t <= s after 1 ns;
15     end process;
16 end architecture;

```



current simulation time: 3 ns  
 scheduled processes:  
 active processes:  
 completed processes: p0, p1

signal	value	scheduled events
s	'1'	'0' @ 4 ns, '1' @ 6 ns
t	'1'	--

# Signal Assignments

## Examples

### Example 4

```

1 architecture arch of sim4 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns;
7       t <= '1' after 4 ns;
8       wait;
9     end process;
10
11  p1 : process(all)
12    begin
13      t <= s after 1 ns;
14    end process;
15
16 end architecture;

```



The next few steps should be quite obvious, as they are just a repetition of the events that we already showed. We therefore only show them as animation steps.

## Example 4

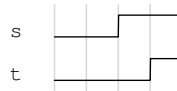
HWMoD  
WS25

Signal Assignn.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
Example 4  
Example 5  
Concurrent Assignn.

```

1 architecture arch of sim04 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns;
9       wait;
10    end process;
11
12   p1 : process(all)
13     begin
14       t <= s after 1 ns;
15     end process;
16 end architecture;

```



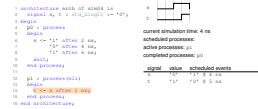
current simulation time: 4 ns  
scheduled processes: p1  
active processes:  
completed processes: p0

signal	value	scheduled events
s	'0'	'1' @ 6 ns
t	'1'	--

# Signal Assignments

## Examples

### Example 4



## Example 4

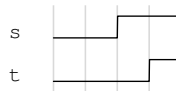
HWMoD  
WS25

Signal Assignn.  
 Introduction  
 Assign. Statements  
 Waveforms  
 Examples  
 Example 1  
 Example 2  
 Example 3  
**Example 4**  
 Example 5  
 Concurrent Assignn.

```

1 architecture arch of sim04 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns;
9     wait;
10    end process;
11
12    p1 : process(all)
13    begin
14      t <= s after 1 ns;
15    end process;
16 end architecture;

```



current simulation time: 4 ns  
 scheduled processes:  
 active processes: p1  
 completed processes: p0

signal	value	scheduled events
s	'0'	'1' @ 2 ns
t	'1'	'0' @ 5 ns

# Signal Assignments

## Examples

### Example 4



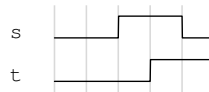
## Example 4

HWMoD  
WS25

Signal Assignn.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
**Example 4**  
Example 5  
Concurrent Assignn.

```

1 architecture arch of sim04 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns;
9       wait;
10    end process;
11
12   p1 : process(all)
13     begin
14       t <= s after 1 ns;
15     end process;
16 end architecture;
```



current simulation time: 5 ns  
 scheduled processes:  
 active processes:  
 completed processes: p0, p1

signal	value	scheduled events
s	'0'	'1' @ 6 ns
t	'0'	--

# Signal Assignments

## Examples

### Example 4



## Example 4

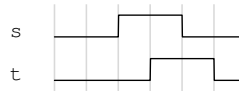
HWMoD  
WS25

Signal Assignn.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
**Example 4**  
Example 5  
Concurrent Assignn.

```

1 architecture arch of sim04 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns;
9     wait;
10    end process;
11
12   p1 : process(all)
13     begin
14       t <= s after 1 ns;
15     end process;
16 end architecture;

```



current simulation time: 6 ns  
scheduled processes: p1  
active processes:  
completed processes: p0

signal	value	scheduled events
s	'1'	--
t	'0'	--

# Signal Assignments

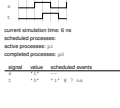
## Examples

### Example 4

```

1 architecture arch of sim4 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns;
7       t <= '1' after 4 ns;
8       wait for 6 ns;
9     end process;
10
11    p1 : process(all)
12    begin
13      t <= s after 1 ns;
14    end process;
15  end architecture;

```



## Example 4

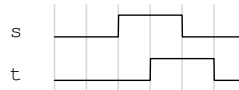
HWMod  
WS25

Signal Assignn.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
**Example 4**  
Example 5  
Concurrent Assignn.

```

1 architecture arch of sim04 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns;
9     wait;
10    end process;
11
12    p1 : process(all)
13    begin
14      t <= s after 1 ns;
15    end process;
16 end architecture;

```



current simulation time: 6 ns  
scheduled processes:  
active processes: p1  
completed processes: p0

signal	value	scheduled events
s	'1'	--
t	'0'	'1' @ 7 ns

# Signal Assignments

## Examples

### Example 4



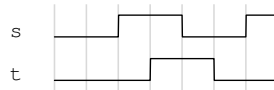
## Example 4

HWMoD  
WS25

Signal Assignn.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
**Example 4**  
Example 5  
Concurrent Assignn.

```

1 architecture arch of sim04 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns;
9       wait;
10    end process;
11
12   p1 : process(all)
13     begin
14       t <= s after 1 ns;
15     end process;
16 end architecture;
```



current simulation time: 7 ns  
scheduled processes:  
active processes:  
completed processes: p0, p1

signal	value	scheduled events
s	'1'	--
t	'1'	--

# Signal Assignments

## Examples

### Example 4



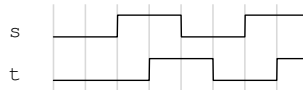
## Example 4

HWMoD  
WS25

Signal Assignn.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
**Example 4**  
Example 5  
Concurrent Assignn.

```

1 architecture arch of sim04 is
2   signal s, t : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns;
9       wait;
10    end process;
11
12   p1 : process(all)
13     begin
14       t <= s after 1 ns;
15     end process;
16 end architecture;
```



current simulation time: 8 ns  
scheduled processes:  
active processes:  
completed processes: p0, p1

signal	value	scheduled events
s	'1'	--
t	'1'	--

# Signal Assignments

## Examples

### Example 5

```
1 architecture arch of sim05 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5   begin
6     s <= '1' after 2 ns;
7     s <= '0' after 4 ns;
8     s <= '1' after 6 ns;
9     s <= '0' after 8 ns;
10    wait for 3 ns;
11    s <= '0' after 2 ns;
12  end;
13 end process;
14 end architecture;
```

Finally, in our last example we demonstrate what happens when a value is assigned to a signal that already has scheduled events in its event queue.

## Example 5

HWMoD  
WS25

### Signal Assign.

- Introduction
- Assign. Statements
- Waveforms
- Examples
  - Example 1
  - Example 2
  - Example 3
  - Example 4
  - Example 5**
  - Concurrent Assign.

```
1 architecture arch of sim05 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5   begin
6     s <= '1' after 2 ns,
7       '0' after 4 ns,
8       '1' after 6 ns,
9       '0' after 8 ns;
10    wait for 3 ns;
11    s <= '0' after 2 ns;
12  wait;
13  end process;
14 end architecture;
```

# Signal Assignments

## Examples

### Example 5

```

1 architecture arch of sim5 is
2   signal s : std_ulogic := '0';
3   process
4     begin
5       s <= '1' after 2 ns;
6       s <= '0' after 4 ns;
7       s <= '1' after 6 ns;
8       s <= '0' after 8 ns;
9       wait for 3 ns;
10      s <= '0' after 2 ns;
11      wait;
12    end process;
13 end architecture;

```

current simulation time: 0 ns  
 scheduled processes: p0  
 active processes:  
 completed processes:  
 signal: s value: '0' scheduled events: --

As before, the signal `s` is initialized to low, and the single process `p0` is scheduled for execution.

## Example 5

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
Example 4  
Example 5  
Concurrent Assign.

```

1 architecture arch of sim05 is
2   signal s : std_ulogic := '0';
3   begin
4     p0 : process
5       begin
6         s <= '1' after 2 ns,
7           '0' after 4 ns,
8           '1' after 6 ns,
9           '0' after 8 ns;
10        wait for 3 ns;
11        s <= '0' after 2 ns;
12        wait;
13      end process;
14 end architecture;

```



current simulation time: 0 ns  
 scheduled processes: p0  
 active processes:  
 completed processes:

signal	value	scheduled events
s	'0'	--

# Signal Assignments

## Examples

### Example 5

```

1 architecture arch of sim5 is
2   signal A : std_ulogic := '0';
3   begin
4     p0 : process
5     begin
6       wait for 2 ns;
7       A <= '1';
8       wait for 4 ns;
9       A <= '0';
10      wait for 6 ns;
11      A <= '1';
12      wait for 8 ns;
13      A <= '0';
14    end process;
15 end architecture;

```

current simulation time: 0 ns  
 scheduled processes:  
 active processes: p0  
 completed processes:

signal	value	scheduled events
A	'1'	@ 2 ns, '0' @ 4 ns, '1' @ 6 ns, '0' @ 8 ns

Next, the process becomes active and the simulator reaches the first statement, highlighted on the slide. The waveform that is assigned to `s` in this example describes two 2 nanosecond pulses. The first pulse is scheduled at 2 nanoseconds of simulation time, while the second one is scheduled for 6 nanoseconds. Hence, the simulator schedules four separate events and continues.

## Example 5

HWMoD  
WS25

Signal Assign.  
 Introduction  
 Assign. Statements  
 Waveforms  
 Examples  
 Example 1  
 Example 2  
 Example 3  
 Example 4  
**Example 5**  
 Concurrent Assign.

```

1 architecture arch of sim05 is
2   signal s : std_ulogic := '0';
3   begin
4     p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns,
9         '0' after 8 ns;
10      wait for 3 ns;
11      s <= '0' after 2 ns;
12      wait;
13    end process;
14 end architecture;

```



current simulation time: 0 ns  
 scheduled processes:  
 active processes: p0  
 completed processes:

signal	value	scheduled events
s	'0'	'1' @ 2 ns, '0' @ 4 ns, '1' @ 6 ns, '0' @ 8 ns

# Signal Assignments

## Examples

### Example 5

```

1 architecture arch of sim5 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     s <= '1' after 2 ns;
6     s <= '0' after 4 ns;
7     s <= '1' after 6 ns;
8     s <= '0' after 8 ns;
9     wait for 3 ns;
10    s <= '0' after 2 ns;
11  wait;
12  end process;
13 end architecture;

```

current simulation time: 0 ns  
 scheduled processes: p0 @ 3 ns  
 active processes:  
 completed processes:  
 signal value scheduled events  
 s '0' '1' @ 2 ns, '0' @ 4 ns,  
 '1' @ 6 ns, '0' @ 8 ns

Now the simulator encounters the `wait-for` statement, which suspends the process until three nanoseconds of simulation time have passed.

## Example 5

HWMoD  
WS25

Signal Assignn.  
 Introduction  
 Assign. Statements  
 Waveforms  
 Examples  
 Example 1  
 Example 2  
 Example 3  
 Example 4  
 Example 5  
 Concurrent Assignn.

```

1 architecture arch of sim05 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns,
9         '0' after 8 ns;
10      wait for 3 ns;
11      s <= '0' after 2 ns;
12    wait;
13  end process;
14 end architecture;

```



current simulation time: 0 ns  
 scheduled processes: p0 @ 3 ns  
 active processes:  
 completed processes:

signal	value	scheduled events
s	'0'	'1' @ 2 ns, '0' @ 4 ns, '1' @ 6 ns, '0' @ 8 ns

# Signal Assignments

## Examples

### Example 5

```

1 architecture arch of sim5 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     s <= '1' after 2 ns;
6     s <= '0' after 4 ns;
7     s <= '1' after 6 ns;
8     s <= '0' after 8 ns;
9     wait for 3 ns;
10    s <= '0' after 2 ns;
11    wait;
12  end process;
13 end architecture;

```

current simulation time: 0 ns  
 scheduled processes: p0 @ 3 ns  
 active processes:  
 completed processes:  
 signal value scheduled events  
 s '0' '1' @ 2 ns, '0' @ 4 ns,  
 '1' @ 6 ns, '0' @ 8 ns

Since there are no processes to execute, the simulator goes through the event queue. The next scheduled event is the assignment of the value one to *s* at 2 nanoseconds.

## Example 5

HWMoD  
WS25

Signal Assignn.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
Example 4  
Example 5  
Concurrent Assignn.

```

1 architecture arch of sim05 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns,
9         '0' after 8 ns;
10      wait for 3 ns;
11      s <= '0' after 2 ns;
12      wait;
13    end process;
14 end architecture;

```



current simulation time: 0 ns  
 scheduled processes: p0 @ 3 ns  
 active processes:  
 completed processes:

signal	value	scheduled events
s	'0'	'1' @ 2 ns, '0' @ 4 ns, '1' @ 6 ns, '0' @ 8 ns

# Signal Assignments

## Examples

### Example 5

```

1 architecture arch of sim5 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     s <= '1' after 2 ns;
6     s <= '0' after 4 ns;
7     s <= '1' after 6 ns;
8     s <= '0' after 8 ns;
9     wait for 3 ns;
10    s <= '0' after 2 ns;
11    wait;
12  end process;
13 end architecture;

```

current simulation time: 2 ns  
 scheduled processes: p0  
 active processes:  
 completed processes:  
 signal value scheduled events  
 s '1' @ 2 ns, '0' @ 4 ns

Hence, the simulation time is advanced accordingly, and the assignment is executed. Note that the process still remains suspended.

## Example 5

HWMoD  
WS25

Signal Assign.  
 Introduction  
 Assign. Statements  
 Waveforms  
 Examples  
 Example 1  
 Example 2  
 Example 3  
 Example 4  
**Example 5**  
 Concurrent Assign.

```

1 architecture arch of sim05 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns,
9         '0' after 8 ns;
10      wait for 3 ns;
11      s <= '0' after 2 ns;
12      wait;
13    end process;
14 end architecture;

```



current simulation time: 2 ns  
 scheduled processes: p0  
 active processes:  
 completed processes:

signal	value	scheduled events
s	'1'	'0' @ 4 ns, '1' @ 6 ns
		'0' @ 8 ns

# Signal Assignments

## Examples

### Example 5

```

1 architecture arch of sim5 is
2   signal A : std_ulogic := '0';
3 begin
4   p0 : process
5     '0' after 2 ns,
6     '1' after 4 ns,
7     '1' after 6 ns,
8     '0' after 8 ns;
9   wait for 3 ns;
10  s <= '0' after 2 ns;
11  wait;
12 end process;
13 end architecture;

```

current simulation time: 3 ns  
 scheduled processes:  
 active processes: p0  
 completed processes:  
 signal: value scheduled events  
 s '1' @ 2 ns, '0' @ 4 ns

Now, the next thing on the agenda is waking up process p0 at 3 nanoseconds. Hence, the simulation time is advanced accordingly.

## Example 5

HWMoD  
WS25

Signal Assign.  
 Introduction  
 Assign. Statements  
 Waveforms  
 Examples  
 Example 1  
 Example 2  
 Example 3  
 Example 4  
**Example 5**  
 Concurrent Assign.

```

1 architecture arch of sim05 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns,
9         '0' after 8 ns;
10    wait for 3 ns;
11    s <= '0' after 2 ns;
12    wait;
13  end process;
14 end architecture;

```



current simulation time: 3 ns  
 scheduled processes:  
 active processes: p0  
 completed processes:

signal	value	scheduled events
s	'1'	'0' @ 4 ns, '1' @ 6 ns
		'0' @ 8 ns

# Signal Assignments

## Examples

### Example 5

```

1 architecture arch of sim5 is
2   signal A : std_ulogic := '0';
3 begin
4   p0 : process
5     s <= '1' after 2 ns;
6     s <= '0' after 4 ns;
7     s <= '1' after 6 ns;
8     s <= '0' after 8 ns;
9     wait for 3 ns;
10    s <= '0' after 2 ns;
11    wait;
12  end process;
13 end architecture;

```

current simulation time: 3 ns  
 scheduled processes:  
 active processes: p0  
 completed processes:  
 signal value scheduled events  
 s '1' @ 2 ns, '0' @ 4 ns

The next statement in the process is another assignment to `s`. The simulator will schedule this assignment which takes place at 5 nanoseconds. However, now something particularly noteworthy happens. Every time a new assignment to a signal happens in the same process, all old transactions that are projected to occur at or after the time at which the earliest new transaction is projected to occur are deleted. Note that the latter part of the last sentence is a direct quote from the VHDL reference.

## Example 5

HWMoD  
WS25

Signal Assignn.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
Example 4  
Example 5  
Concurrent Assignn.

```

1 architecture arch of sim05 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6     s <= '1' after 2 ns,
7     '0' after 4 ns,
8     '1' after 6 ns,
9     '0' after 8 ns;
10    wait for 3 ns;
11    s <= '0' after 2 ns;
12    wait;
13  end process;
14 end architecture;

```



current simulation time: 3 ns  
 scheduled processes:  
 active processes: p0  
 completed processes:

signal	value	scheduled events
s	'1'	'0' @ 4 ns, '1' @ 6 ns
		'0' @ 8 ns

# Signal Assignments

## Examples

### Example 5

```

1 architecture arch of sim5 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     s <= '1' after 2 ns;
6     s <= '0' after 4 ns;
7     s <= '1' after 6 ns;
8     s <= '0' after 8 ns;
9     wait for 3 ns;
10    s <= '0' after 2 ns;
11    wait;
12  end process;
13 end architecture;

```

current simulation time: 3 ns  
 scheduled processes:  
 active processes: p0  
 completed processes:  
 signal: s  
 scheduled events: '1' @ 2 ns, '0' @ 4 ns

For our concrete example this means that every event at or after 5 nanoseconds is deleted from the event queue. Thus, the transition to zero at 4 nanoseconds is kept, while every thing else is replaced by a new event at 5 nanoseconds setting *s* to zero.

## Example 5

HWMoD  
WS25

Signal Assignn.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
Example 4  
Example 5  
Concurrent Assignn.

```

1 architecture arch of sim05 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6     s <= '1' after 2 ns,
7       '0' after 4 ns,
8       '1' after 6 ns,
9       '0' after 8 ns;
10    wait for 3 ns;
11    s <= '0' after 2 ns;
12    wait;
13  end process;
14 end architecture;

```



current simulation time: 3 ns  
 scheduled processes:  
 active processes: p0  
 completed processes:

signal	value	scheduled events
s	'1'	'0' @ 4 ns, '0' @ 5 ns

# Signal Assignments

## Examples

### Example 5

```

1 architecture arch of sim5 is
2   signal A : std_ulogic := '0';
3 begin
4   p0 : process
5     wait for 1 ns;
6     A <= '1' after 2 ns;
7     wait for 4 ns;
8     A <= '0' after 2 ns;
9     wait for 3 ns;
10    A <= '0' after 2 ns;
11  end process;
12 end architecture;

```

current simulation time: 3 ns  
 scheduled processes:  
 active processes:  
 completed processes: p0  
 signal: value: scheduled events

Finally, the unconditional `wait` statement is reached which completes the process. As in previous examples, the simulator now goes through its list of scheduled events and advances the simulation time to the one of the next event.

## Example 5

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
Example 4  
Example 5  
Concurrent Assign.

```

1 architecture arch of sim05 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns,
9         '0' after 8 ns;
10    wait for 3 ns;
11    s <= '0' after 2 ns;
12    wait;
13  end process;
14 end architecture;

```



current simulation time: 3 ns  
 scheduled processes:  
 active processes:  
 completed processes: p0

signal	value	scheduled events
s	'1'	'0' @ 4 ns, '0' @ 5 ns



As a result the simulation time is set to 4 nanoseconds, and s is set to zero.

## Example 5

HWMoD  
 WS25

Signal Assignn.  
 Introduction  
 Assign. Statements  
 Waveforms  
 Examples  
 Example 1  
 Example 2  
 Example 3  
 Example 4  
 Example 5  
 Concurrent Assignn.

```

1 architecture arch of sim05 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns,
9         '0' after 8 ns;
10      wait for 3 ns;
11      s <= '0' after 2 ns;
12      wait;
13    end process;
14 end architecture;
```



current simulation time: 4 ns  
 scheduled processes:  
 active processes:  
 completed processes: p0

signal	value	scheduled events
s	'0'	'0' @ 5 ns

# Signal Assignments

## Examples

### Example 5



The last event happens at 5 nanoseconds, which also sets `s` to zero. However, since `s` is already zero, nothing changes.

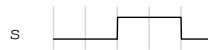
## Example 5

HWMoD  
WS25

Signal Assignn.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
Example 4  
Example 5  
Concurrent Assignn.

```

1 architecture arch of sim05 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns,
9         '0' after 8 ns;
10    wait for 3 ns;
11    s <= '0' after 2 ns;
12    wait;
13  end process;
14 end architecture;
```



current simulation time: 5 ns  
scheduled processes:  
active processes:  
completed processes: p0

signal	value	scheduled events
s	'0'	--

# Signal Assignments

## Examples

### Example 5



Since there are no more scheduled events or processes the simulation terminates. Hence, further advancing the simulation time does not lead to any more signal changes.

## Example 5

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Example 1  
Example 2  
Example 3  
Example 4  
Example 5  
Concurrent Assign.

```

1 architecture arch of sim05 is
2   signal s : std_ulogic := '0';
3 begin
4   p0 : process
5     begin
6       s <= '1' after 2 ns,
7         '0' after 4 ns,
8         '1' after 6 ns,
9         '0' after 8 ns;
10      wait for 3 ns;
11      s <= '0' after 2 ns;
12      wait;
13    end process;
14 end architecture;
```



current simulation time: 10 ns  
scheduled processes:  
active processes:  
completed processes: p0

signal	value	scheduled events
s	'0'	--

- └ Signal Assignments
  - └ Concurrent Assign.
    - └ **Concurrent Signal Assignments**

■ Largely use the same syntax as assignment statements in processes

Before we close this lecture, let us also briefly cover concurrent signal assignments. As already mentioned in the beginning of the lecture, for the scope of this course, you can consider concurrent assignments and assignments in processes as using the same syntax rules. However, the question remains, how they should be interpreted in simulation.

## Concurrent Signal Assignments

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Concurrent Assign.

- Largely use the same syntax as assignment statements in processes

The VHDL reference manual states that for any concurrent signal assignment statement, there is an equivalent process statement with the same meaning. It then goes on, presenting a detailed transformation algorithm that is guaranteed to preserve the meaning of the original assignment.

## Concurrent Signal Assignments

206

HWMoD  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Concurrent Assign.

- Largely use the same syntax as assignment statements in processes
- VHDL Reference

*“For any concurrent signal assignment statement, there is an equivalent process statement with the same meaning.”*

However, to the extent we covered concurrent assignments in this course you can simply think of them as being embedded in a process with an `all` sensitivity list. Note that, if there are only constants on the right-hand-side of the assignment then the process is only executed once.

## Concurrent Signal Assignments

206

HWMod  
WS25

Signal Assign.  
Introduction  
Assign. Statements  
Waveforms  
Examples  
Concurrent Assign.

- Largely use the same syntax as assignment statements in processes
- VHDL Reference  
*“For any concurrent signal assignment statement, there is an equivalent process statement with the same meaning.”*
- Think of them as embedded in a process with an `all` sensitivity list

```
s <= t after 1 ns;  ⇔  process (all)
                       begin
                           s <= t after 1 ns;
                       end if;
```

Thank you for listening! We recommend you to immediately take the self-check test in TUWEL, to see if you understood the material presented in this lecture.

HWMod  
WS25

Signal Assign.

Introduction

Assign. Statements

Waveforms

Examples

Concurrent Assign.

# Lecture Complete!