



## VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

### Einleitung

#### Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

#### Konfiguration

#### Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

#### Ausdrücke

#### Attribute

#### Literatur

1

# VHDL

D. Neuhäuser, G. Grune (W. Koch)

Dezember 2012



## Beschreibung von Rechengesystemen auf 6 Ebenen

## VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

### Einleitung

#### Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

#### Konfiguration

#### Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

#### Ausdrücke

#### Attribute

#### Literatur

2

- **Algorithmische Ebene** spezifiziert den Algorithmus zur Lösung eines Entwurfsproblems
- **PMS-Ebene** (Processor, Memory, Switch) beschreibt Rechner grob durch die Hauptelemente
- **Befehlsebene** beinhaltet die auf dem Rechner ausführbaren Befehle
- **Register-Transfer-Ebene** beschreibt Operationen zwischen Registern. Wird auch als Mikrobefehlsebene bezeichnet.
- **Logik-Ebene** beschreibt den Aufbau von Schaltwerken mit Hilfe von Gattern und Flipflops
- **Schaltkreisebene** geht auf die Realisierung durch Transistoren, Dioden, Widerstände usw. ein



## Forderungen an Entwurfssprachen

## VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

### Einleitung

#### Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

#### Konfiguration

#### Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

#### Ausdrücke

#### Attribute

#### Literatur

3

- möglichst mehrere Ebenen abdecken
- Schnittstellen zu anderen Ebenenbeschreibungen bereitstellen
- Darstellung paralleler Vorgänge
- synchrones und asynchrones Zeitverhalten wird unterstützt



## Aufgaben von Entwurfssprachen

## VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

### Einleitung

#### Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

#### Konfiguration

#### Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

#### Ausdrücke

#### Attribute

#### Literatur

4

- Bereitstellung von Entwurfshilfsmitteln
- aus logischen Entwurf wird (automatisch) Hardware-Implementierung (Erstellung des Chip-Layouts und der Verdrahtung)
- Verifikation und Simulation von Entwürfen
- Aufdecken logischer Fehler im Entwurf durch Testläufe
- Aufdecken von Fehlern im dynamischen Verhalten durch Überprüfung von Zeitbedingungen
- Ermitteln von Engpässen im Systementwurf durch Simulation
- Erleichterung des Austauschs von Entwürfen durch Formalisierung des Entwurfes und Normierung von Entwurfssprachen



## Eigenschaften von Entwurfssprachen

### VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

#### Einleitung

##### Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

##### Konfiguration

##### Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

##### Ausdrücke

##### Attribute

##### Literatur

- Prozedural: Die Ausführungsreihenfolge der Befehle ist durch die Anordnung im Programm vorgegeben
- Nicht-prozedural: Jede Anweisung wird durch einen logischen Ausdruck markiert. Ist dieser Ausdruck wahr, dann wird die Anweisung ausgeführt.
- Mischformen

5



## VHDL

### VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

#### Einleitung

##### Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

##### Konfiguration

##### Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

##### Ausdrücke

##### Attribute

##### Literatur

- **VHSIC Hardware Description Language**
- **Very High Speed Integrated Circuit**
- Anfang der 80 entwickelt vom DoD
- Hardwarebeschreibungssprache
- Standardisierung durch **IEEE VHDL87** (IEEE 1076-1987) und **VHDL93** (IEEE 1076-1993)
- Institute of **Electrical and Electronics Engineers**
- andere: Verilog, System-C

6



## VHDL vs. Programmiersprachen

### VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

#### Einleitung

##### Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

##### Konfiguration

##### Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

##### Ausdrücke

##### Attribute

##### Literatur

- Modularität
- zeitliche Denkweise
- sequenzielle und nebenläufige Anweisungen
- Signale als Verdrahtung

7



## Begriffsklärung

### VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

#### Einleitung

##### Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

##### Konfiguration

##### Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

##### Ausdrücke

##### Attribute

##### Literatur

- Entity = Schnittstellenbeschreibung
- Architektur = Modellbeschreibung
- Configuration = Bindung Architektur - Entity
- Packages = Bündelung häufig verwendeter Elemente
- Einheit = einzelnes Bauteil, welches zu keiner Schaltung gehört
- Komponente = Bauteil das zu einer Schaltung gehört
- Instanz = mehrere Bauteile des selben Typs

8



## Schlüsselwort *entity* (I)

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- Beschreibung der Schnittstellen, Informationen aus Schaltzeichen

### Syntax

```
entity entity_name is
  [generics]
  [ports]
  [declarations (types, constants, signals)]
  [definitions (funktionen, procedures)]
  [begin                               -- unueblich
    statements]
end [entity_name];
```

9



## Schlüsselwort *entity* (II)

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

### Beispiel RS-FF: Schaltbild



### Beispiel RS-FF: VHDL-entity

```
entity rs_ff is
  port (
    r, s : in  std_logic;
    q, qb : out std_logic);
end rs_ff;
```

10



## Port Deklaration

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

### Syntax

```
entity entity_name is
  [generics(Parameter_Liste);]
  [ports(Liste_der_Pins);]
end [entity] [entity_name];
```

- Deklaration von Signalen gleichen Namens/Typs für alle Architekturen
- Verwendung hängt vom Modus ab
- generics = Parameter für variablen Strukturen z.B. Busbreiten, Verzögerungszeiten

11



## Port Modi

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

### Syntax

```
entity entity_name is
  [generics(Parameter_Liste);]
  [ports(Liste_der_Pins);]
end [entity] [entity_name];
```

- Modus in: Eingang, Signal kann innerhalb der Entity und deren Architekturen nur gelesen werden
- Modus out: Ausgang, auf diese Ports darf nur geschrieben werden
- Modus inout: Lese- und Schreibzugriff, bidirektional
- Modus buffer: Ausgang, aber auch auf der rechten Seite von Zuweisungen

12



## Schlüsselwort 'architecture'

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture

Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen

Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- Beschreibung der Schaltung
  - Nebenläufiges Verhaltensmodell (Datenflussbeschreibung)
  - Sequenzielles Verhaltensmodell (Verhaltensbeschreibung)
  - Strukturmodell (Strukturbeschreibung)
  - Elemente der verschiedenen Modelle kombinierbar

### Syntax architecture

```
architecture architecture_name of entity_name is
  [arch_declarative_part]
begin
  [arch_statement_part]
end [architecture_name];
```

13



## Nebenläufiges Verhaltensmodell (I)

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture

Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen

Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- bei welcher Eingabe ergibt sich welche Ausgabe
- Datenflussbeschreibung
- nebenläufige Anweisungen
- (virtuelle) parallele Ausführung aller Anweisung

14



## Nebenläufiges Verhaltensmodell (II)

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture

Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen

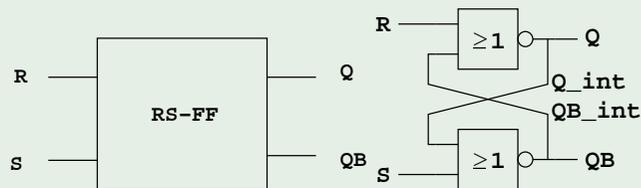
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

### Beispiel RS-FF Schaltbild



### Beispiel RS-FF VHDL-Code-Architektur

```
architecture dataflow of rs_ff is
  signal q_int,qb_int : std_logic;
begin
  q_int <= r nor qb_int;
  qb_int <= s nor q_int;
  q <= q_int;
  qb <= qb_int;
end dataflow;
```

15



## Exkurs: Testen im Simulator (I)

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture

Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen

Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

### Beispiel RS-FF VHDL-Code

```
library IEEE;
use IEEE.std_logic_1164.all;

entity rs_ff is
  port (
    r, s : in std_logic;
    q, qb : out std_logic);
end rs_ff;

architecture dataflow of rs_ff is
  signal q_int,qb_int : std_logic;
begin
  q_int <= r nor qb_int;
  qb_int <= s nor q_int;
  q <= q_int;
  qb <= qb_int;
end dataflow;
```

16



## Exkurs: Testen im Simulator (II)

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architektur

Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen

Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

17

### Beispiel RS-FF Testbench ...

```

library IEEE;
use IEEE.std_logic_1164.all;

entity E is
end E;

architecture A of E is
component rs_ff is
port (
  r, s : in std_logic;
  q, qb : out std_logic);
end component rs_ff;
signal r_tb, s_tb, q_tb, qb_tb : std_logic;
begin
  UUT : rs_ff port map (
    r => r_tb,
    s => s_tb,
    q => q_tb,
    qb => qb_tb);

```



## Exkurs: Testen im Simulator (III)

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architektur

Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen

Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

18

### Beispiel ... RS-FF Testbench

```

begin
  UUT : rs_ff port map (
    r => r_tb,
    s => s_tb,
    q => q_tb,
    qb => qb_tb);
  DATEN : process
  begin
    r_tb <= '0'; s_tb <= '0'; wait for 20 ns;
    r_tb <= '0'; s_tb <= '1'; wait for 20 ns;
    r_tb <= '0'; s_tb <= '0'; wait for 20 ns;
    r_tb <= '1'; s_tb <= '0'; wait for 20 ns;
    r_tb <= '0'; s_tb <= '0'; wait for 20 ns;
    r_tb <= '1'; s_tb <= '1'; wait for 20 ns;
    r_tb <= '0'; s_tb <= '0'; wait for 20 ns;
    assert false
      report "Ende der Testbench." severity note;
    wait;
  end process;
end A;

```



## Exkurs: Testen im Simulator (IV)

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architektur

Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen

Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

19

### Beispiel RS-FF Simulationsergebnis



## Exkurs: Synthese

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architektur

Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen

Skalare DT  
Komplexe DT  
Weitere DT

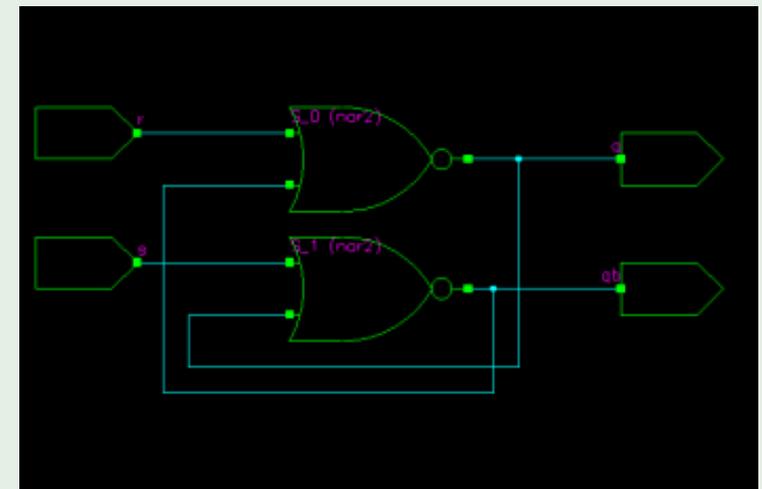
Ausdrücke

Attribute

Literatur

20

### Beispiel RS-FF Synthesergebnis





## Exkurs: GHDL

### VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

### Einleitung

### Entity

### Architecture

Nebenläufig  
Sequenziell  
Strukturell

### Konfiguration

### Packages

### Datentypen

Skalare DT  
Komplexe DT  
Weitere DT

### Ausdrücke

### Attribute

### Literatur

- Sie können alle Beispiele an Ihrem PC selbst durchspielen - mit GHDL
- Wir haben den Simulator GHDL und den Waveform-Viewer GTKWave für Windows ins CAJ gestellt
- Einfach entpacken - Howto.txt lesen - installieren - fertig!

21



## Exkurs: Simulator

### VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

### Einleitung

### Entity

### Architecture

Nebenläufig  
Sequenziell  
Strukturell

### Konfiguration

### Packages

### Datentypen

Skalare DT  
Komplexe DT  
Weitere DT

### Ausdrücke

### Attribute

### Literatur

Ein VHDL-Simulator arbeitet zyklisch 2 Phasen ab:

- Ausführungsphase  
Alle aktiven Prozesse werden (quasi parallel) abgearbeitet  
Signale werden dabei noch nicht verändert -  
die Änderungen werden (mit Zeitstempel) in einer Transaktionsliste vorgemerkt
- Aktualisierungsphase  
Signale werden entsprechend der Transaktionsliste erst jetzt verändert -  
zeitverzögerte Änderungen erfolgen in einer späteren Aktualisierungsphase

22



## nebenläufige Anweisungen

### VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

### Einleitung

### Entity

### Architecture

Nebenläufig  
Sequenziell  
Strukturell

### Konfiguration

### Packages

### Datentypen

Skalare DT  
Komplexe DT  
Weitere DT

### Ausdrücke

### Attribute

### Literatur

- einfache Signalzuweisung
- bedingte Signalzuweisung
- selektive Signalzuweisung
- Assertion
- Prozess

23



## einfache Signalzuweisung

### VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

### Einleitung

### Entity

### Architecture

Nebenläufig  
Sequenziell  
Strukturell

### Konfiguration

### Packages

### Datentypen

Skalare DT  
Komplexe DT  
Weitere DT

### Ausdrücke

### Attribute

### Literatur

- der Wert auf der rechten Seite wird berechnet und dem Signal auf der linken Seite zugewiesen
- Berechnung immer nach Signaländerung auf der rechten Seiten
- träges vs. nichtträges Modell

### Syntax

```
[label:]
signal_name <= [transport] expression [after time_expr]
{, expression after time_expr};
```

### Beispiel

```
X <= Y or Z after 5 ns, Y and Z after 10 ns;
```

24



## Träges vs. nichtträges Modell

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architektur  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

25

### Syntax

```
[label:]
signal_name <= [transport] expression [after time_expr]
               {, expression after time_expr};
```

- *inertial* (default) - träge  
reagiert nicht auf kürzere Eingangsänderungen
- *transport* - nichtträge  
d.h. reagiert auch auf kurze Eingangsänderungen



## bedingte Signalzuweisung

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architektur  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

26

- mehrere Zuweisungsalternativen
- vergleichbar mit if-then-else Struktur

### Syntax

```
[label:]
signal_name <= expression when condition else
               {expression when condition else}
               expression;
```

### Beispiel

```
X <= B when S = '1' else not B;
```



## selektive Signalzuweisung

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architektur  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

27

### Syntax

```
[label:]
with select_expression select
signal_name <= expression when value
               {, expression when value};
```

### Beispiel

```
with X select
Y <= A when "00",
     B when "01",
     C when "10";
A and B and C when "11";
```



## Schlüsselwort *assertion*

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architektur  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

28

- Ausgabe von Fehlermeldungen und Warnungen bei bestimmten Bedingungen
- beeinflusst das Verhalten des VHDL-Simulators

### Syntax

```
[assert_label:]
assert condition
[report string_expr]
[severity failure|error|warning|note];
```

### Beispiel

```
assert X = Y
report "X ist ungleich Y.";
severity note;
```



## Sequenzielles Verhaltensmodell

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- Verhaltensbeschreibung
- Grundlage ist die Prozess-Umgebung
- einfache Signalzuweisung
- Variablenzuweisung
- Assertion
- wait-Anweisung
- if-elsif-else Anweisung
- case-, null-Anweisung
- loop-, exit- und next-Anweisung

29



## Schlüsselwort 'process'

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- Prozess ist nebenläufige Anweisung: läuft fortwährend
- Umgebung für sequentiell abgearbeitete Anweisungen
- Simulationszeit bleibt stehen !

### Syntax

```
[proc_label:]
process [(sensitivity_list)]
  [proc_declarativ_part]
begin
  [sequential_statement_part]
end process [proc_label];
```

- Zwei Zustände: Ausführungsphase und Wartezustand
- Zwei Möglichkeiten, um diesen Zustandswechsel zu steuern
- Beide Möglichkeiten schliessen sich gegeneinander aus

30



## Prozess mit Sensitivitätsliste

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- Liste von Signalen
- Änderung eines Signals führt zur einmaligen Abarbeitung des Prozesses

### Syntax

```
[proc_label:]
process (sensitivity_list)
  [proc_declarativ_part]
begin
  [sequential_statement_part]
end process [proc_label];
```

31



## Prozess mit Sensitivitätsliste

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

### Beispiel RS-FF

```
architecture behaviour of rs_ff is
  signal q_int, qb_int : std_logic;
begin
  rsff:process (r,s,q_int,qb_int)
  begin
    if S = '1' or q_int = '1' then
      qb_int <= '0';
      ...
    else
      q_int <= 'X' after tpd_rnor;
    end if;
  end process rsff;
  q <= q_int;
  qb <= qb_int;
end behaviour;
```

32



## Prozess mit wait

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- Prozess wird in einer Endlosschleife abgearbeitet bis er durch wait-Anweisung unterbrochen wird
- mindestens eine wait-Anweisung in der Prozessumgebung

### Syntax

```
[proc_label:]
process
  [proc_declarativ_part]
begin
  [sequential_statements]
wait ...;
  [sequential_statements]
end process [proc_label];
```

33



## Prozess mit wait

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

### Beispiel RS-FF

```
architecture behaviour of rs_ff is
  signal q_int, qb_int : std_logic;
begin
  rsff:process
  begin
    wait on r,s,q_int,qb_int;
    if S = '1' or q_int = '1' then
      qb_int <= '0';
      ...
    else
      q_int <= 'X' after tpd_rnor;
    end if;
  end process rsff;
  q <= q_int;
  qb <= qb_int;
end behaviour;
```

34



## Schlüsselwort wait

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- nur in Prozessen ohne sensitivity list
- Steuerung der Zustandswechsel des Prozesses

### Syntax

```
wait [on signal_names]
      [until condition]
      [for time_expression];
```

- on signal\_names: der Prozess wartet solange, bis ein Signal in der Liste [signal names] sich ändert
- until condition: der Prozess wartet, bis Bedingung erfüllt
- for time\_expression: der Prozess wartet für die Zeit [time expression]
- ohne Argument: der Prozess wartet bis zum Simulationsende

35



## Einfache Signalzuweisung in Prozessen

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- Syntax ist identisch mit der nebenläufigen Signalzuweisung
- Signalzuweisung erfolgen erst am Ende des Prozesses

### Syntax (siehe nebenläufige Signalzuweisung)

```
[label:]
signal_name <= [transport] expression [after time_expr]
              {, expression after time_expr};
```

### Beispiel (siehe nebenläufige Signalzuweisung)

```
X <= Y or Z after 5 ns, Y and Z after 10 ns;
```

- keine bedingte Signalzuweisung, Realisierung mit if-elsif-else oder case Anweisungen

36



## Variablenzuweisung

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- ähnlich Signalzuweisung
- keine Verzögerungszeiten

### Syntax

```
variable_name := expression;
```

### Beispiel

```
...
process (a,b)
  variable x: std_logic;
begin
  ...
  x := a or b;
  ...
end process;
```

37



## Schlüsselwort *assertion*

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- Ausgabe von Fehlermeldungen und Warnungen
- Syntax identisch zur nebenläufigen Modellbeschreibung, nur ohne *label*

### Syntax (siehe nebenläufige Modellbeschreibung)

```
assert condition
[report string_expr]
[severity failure|error|warning|note];
```

### Beispiel (siehe nebenläufige Modellbeschreibung)

```
assert X = Y
report "X ist ungleich Y.";
severity note;
```

38



## if-elsif-else-Anweisung

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- entspricht der bedingten Verzweigung von Programmiersprachen

### Syntax

```
if condition then
  sequential_statements
{elsif condition then
  sequential_statements}
[else
  sequential_statements]
end if;
```

39



## if-elsif-else-Anweisung

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

### Beispiel RS-FF (siehe Sensitivitätsliste)

```
architecture behaviour of rs_ff is
  signal q_int, qb_int : std_logic;
begin
  rsff:process (r,s,q_int,qb_int)
  begin
    if S = '1' or q_int = '1' then
      qb_int <= '0';
      ...
    else
      q_int <= 'X' after tpd_rnor;
    end if;
  end process rsff;
  q <= q_int;
  qb <= qb_int;
end behaviour;
```

40



## Schlüsselwort *case*

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architektur  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- entspricht dem case-Konstrukt von Programmiersprachen

### Syntax

```
case expression is
  {when choices => sequential_statements}
  [when others => sequential_statements]
end case;
```

### Beispiel

```
case not X is
  when '0' => Y <= A and B;
              Z <= A or B;
  when '1' => Y <= A nand B;
              Z <= A nor B;
end case;
```

41



## Schlüsselwort *null*

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architektur  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- explizite Kennzeichnung von aktionslosen if- oder case-Fällen

### Syntax

```
null;
```

### Beispiel

```
case X is
  when '00' => Y <= A and B;
              Z <= A or B;
  when '01' => Y <= A nand B;
              Z <= A nor B;
  when others => null;
end case;
```

42



## Schlüsselwort *loop* (I)

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architektur  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- Schleifenkonstrukt analog zu Programmiersprachen
- die Laufvariable der for-Schleife muss nicht extra deklariert werden
- sie gilt als lokale Variable in der Schleife, Zuweisungen sowie externer Zugriff sind nicht möglich

### Syntax

```
[loop_label:] while condition loop|
[loop_label:] for identifier in value1 to value2 loop|
[loop_label:] loop --Endlosschleife
  sequential_statements
end loop [loop_label];
```

43



## Schlüsselwort *loop* (II)

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architektur  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

### Beispiel

```
for i in 0 to 3 loop
  Y(i) <= X(i) xor A;
  -- X,Y std_logic_vector(0 to 3); A std_logic
end loop;
```

44



## exit- und next-Anweisung

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- vorzeitiger Ausstieg aus Schleifenanweisungen
- next: Sprung zum nächsten Schleifendurchlauf
- exit: Verlassen der Schleife

### Syntax

```
next [loop_label][when condition];
exit [loop_label][when condition];
```

- *loop\_label* muss angegeben werden, wenn verschachtelte Schleifen verwendet werden

45



## Unterprogramme

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- Analog zu anderen Sprachen
- Vereinfachung, Modularität, bessere Lesbarkeit
- Prozeduren: mehrere Return-Werte mit Parameterlist
- Funktionen: ein Return-Wert, z.B. Typkonvertierung
- lokale Variablen, nur im Unterprogramm sichtbar
- Overloading-Mechanismen wie in Programmiersprachen

46



## Funktionen

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- mehrere Parameter
- gibt einen Wert zurück

### Syntax

```
function func_name (parameter_list)
return type_name is
  [variable_declaration]
  [constant_declaration]
  [type_declaration]
  [use_clause]
begin
  [sequential_statements]
return expression;
end [func_name];
```

47



## Prozeduren

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- mehrere Parameter
- Parameter Modi:
  - in: Eingabewert
  - out: Ausgabewert, nur auf der linken Seite von Zuweisungen
  - inout: Ein/Ausgabewert

### Syntax

```
procedure proc_name (parameter_list) is
  [variable_declaration]
  [constant_declaration]
  [type_declaration]
  [use_clause]
begin
  [sequential_statements]
end [proc_name];
```

48



## Strukturmodell

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

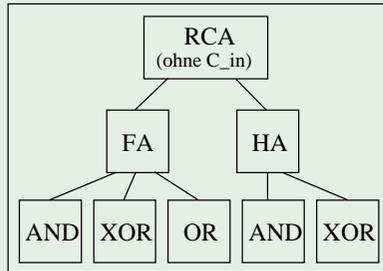
Ausdrücke

Attribute

Literatur

- Beschreibung der Komponenten einer Architektur und deren Verbindungen
- Hierarchische Beschreibung der Schaltung, Strukturbeschreibung
- Deklarieren von Komponenten
- Instanzen dieser Komponenten erzeugen

### Beispiel RCA - Strukturmodell



49



## Komponentendeklaration

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- Bekanntmachen der Komponente
- analog zur Entity

### Syntax

```

component component_name [is]
  [generic ( generic_list: type_name [:= expression]
    {; generic_list: type_name [:= expression]} );]
  [port ( signal_list: in|out|inout|buffer type_name
    {; signal_list: in|out|inout|buffer type_name} );]
end component;
  
```

### Beispiel

```

component nor2
  port (A, B: in std_logic; C: out std_logic);
end component;
  
```

50



## Komponenteninstantiierung

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- Zuordnung lokaler Signale zu Ports der Komponente

### Syntax

```

component_label: component_name
  port map (signal_mapping);
  
```

### Beispiel

```

U1: nor2 port map (S,QB,Q); -- oder ...
U2: nor2 port map (A => S, C => Q, B => QB);
  
```

- U1: Zuordnung über Reihenfolge
- U2: Zuordnung durch Namen

51



## Schlüsselwort *for* (I)

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

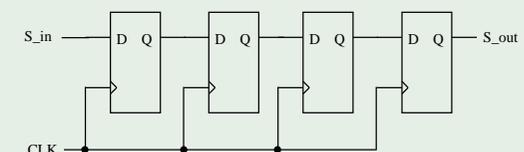
- for-Schleife
- Instantiierung von sich wiederholenden Strukturen
- Vereinfachung der Beschreibung, Verbesserung der Lesbarkeit

### Syntax

```

generate_label: for variable in range generate
  concurrent_statement
end generate [generate_label];
  
```

### Beispiel Schieberegister mittels D-FF



52



## Schlüsselwort *for* (II)

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

53

### Beispiel *for*-Schleife

```

entity SHIFT is
  port ( S_in, CLK: in std_logic;
        S_out: out std_logic);
end SHIFT;

architecture structural of SHIFT is
  component DFF
    port ( D, CLK: in std_logic; Q: out std_logic);
  end component;
  signal X: std_logic_vector (0 to 4);
begin
  X(0) <= S_in;
  SHF:for I in 0 to 3 generate
    UI: DFF port map (X(I), CLK, X(I+1));
  end generate;
  S_out <= X(4);
end structural;

```



## Schlüsselwort *if* (I)

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

54

- Auswahl bestimmter Instantiierungen abhängig von Bedingungen
- Vereinfachung der Beschreibung, Verbesserung der Lesbarkeit

### Syntax

```

generate_label: if (condition) generate
  concurrent_statement
end generate [generate_label];

```



## Schlüsselwort *if* (II)

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

55

### Beispiel *if*-Anweisung

```

...
architecture structural of SHIFT is
  component DFF
    port ( D, CLK: in std_logic; Q: out std_logic);
  end component;
  signal X: std_logic_vector (1 to 3);
begin
  SHF:for I in 0 to 3 generate
    S1: if (I = 0) generate
      U1: DFF port map (S_in, CLK, X(I+1));
    end generate;
    S2: if ((I > 0) and (I < 3)) generate
      UI: DFF port map (X(i), CLK, X(I+1));
    end generate;
    S3: if (I = 3) generate
      U3: DFF port map (x(I), CLK, S_out);
    end generate;
  end generate;
end structural;

```



## Konfiguration/Verhaltensbeschreibung

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

56

- mehrere Architekturen für selbe *entity* definierbar
- Auswahl einer Architektur
- Zuordnung Architektur zu Entity

### Syntax

```

configuration configuration_name of entity_name is
  for architecture_name
  end for;
end configuration_name;

```

### Beispiel

```

configuration cfg1_rs_ff of rs_ff is
  for behaviour
  end for;
end cfg1_rs_ff;

```



## Konfiguration/Strukturbeschreibung (I)

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- mehr Informationen für die instantiierten Komponenten notwendig
- Komponentenname nicht zwingend gleich Entityname

### Syntax

```
configuration configuration_name of entity_name is
  for architecture_name
    for label|other|all: comp_name
      use entity [lib_name.] comp_entity_name(
        comp_arch_name) |
      use configuration [lib_name.]
        comp_configuration_name
        |generic map (...)]
        [port map (...)];
    end for;
  ...
end for;
end configuration_name;
```

57



## Konfiguration/Strukturbeschreibung (II)

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

### Beispiel

```
configuration cfg_rs_ff of rs_ff is
  for structural
    for u1, u2: nor2
      use entity WORK.nor2(behaviour);
    end for;
  end for;
end cfg_rs_ff;
```

58



## Packages

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- einmalige Deklaration/Definition von häufig benötigten Datentypen, Komponenten, Funktionen, etc.
- Trennung Deklaration und Definition optional
- bei Änderung im Body muss nur dieser kompiliert werden

59



## Package Deklaration (I)

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

- Deklaration aller globalen Typen, Komponenten, Prozeduren und Funktionen

### Syntax

```
package package_name is
  [type_dec]
  [subtype_dec]
  [constant_dec]
  [deferred_constant_dec]
  [subprogram_header_dec]
  [component_dec]
end [package_name];
```

60



## Package Deklaration (II)

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architektur  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

61

### Beispiel

```
package MY_PACK is
  type COLOR is (red, green, blue, yellow);
  component NOR
    port (A,B: in std_logic; C: out std_logic);
  end component;
  constant BUS_WIDTH: natural := 8; -- constant
  constant DELAY: time; -- deferred constant
  function BIT_INVERT(V_in: std_logic_vector);
    return std_logic_vector;
end MY_PACK;
```



## Package Body (I)

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architektur  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

62

- Definition von Funktionen und Prozeduren
- Wertzuweisung an Konstanten

### Syntax

```
package body package_name is
  [deferred_constant_value]
  [subprogram_body]
end [package_name];
```



## Package Body (II)

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architektur  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

63

### Beispiel

```
package body MY_PACK is
  constant DELAY: time := 0.75 ns;
  function BIT_INVERT(V_in: std_logic_vector);
    return std_logic_vector is
  begin
    -- sequentielle Verhaltensbeschreibung
  end
end MY_PACK;
```



## Wichtige Packages

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architektur  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

64

- STANDARD boolean, bit, bit vector, character ...
- TEXTIO lesen und schreiben von Textdateien use std.textio.all
- STD\_LOGIC\_1164 normiertes neunwertiges Logiksystem von IEEE
- NUMERIC\_STD/STD\_LOGIC\_ARITH arithmetische Funktionen für die in STD\_LOGIC\_1164 deklarierten Typen



## VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture

Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

65

### Syntax std\_ulogic

```
type std_ulogic is ('U', -- nicht initialisiert
                   'X', -- stark unbekannt
                   '0', -- stark 0
                   '1', -- stark 1
                   'Z', -- hochohmig
                   'W', -- schwach unbekannt
                   'L', -- schwach 0
                   'H', -- schwach 1
                   '-'); -- don't care
```

- std\_ulogic\_vector, std\_logic\_vector, ..

### Beispiel des Einbindens von STD\_LOGIC\_1164

```
library IEEE;
use IEEE.STD_LOGIC_1164.all;
```



## VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture

Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

66

- VHDL ist stark typisiert
- Überprüfung bei der Codeanalyse
- Konvertierungsfunktionen



## VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture

Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

67

- Boolean: Werte true und false
- Integer:
  - Zahlen von  $-2^{31} - 1$  bis  $+2^{31} - 1$
  - dezimal: default, binär: 2#...#, oktal: 8#...#, hex: 16#...#
  - Untertypen: positiv (1 . . . n) und natural (0 . . . n)
- Real: Zahlen von  $-1.7014110 * 10^{38}$  bis  $+1.7014110 * 10^{38}$
- Character: Standard ASCII Zeichensatz, z.B. 'a', '8'



## VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture

Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

68

- Bit: logische Werte '1' und '0'
- Std\_Ulogic und Std\_Logic: siehe Package STD\_LOGIC\_1164
- Physikalische Literale: z.B Time mit fs, ps, ns, us, ms, sec, min, hr
- Aufzählungstypen

### Beispiel

```
type AMPEL is (rot, gelb, gruen);
```

- Untertypen: Einschränkung des Wertebereichs von definierten Typen

### Beispiel

```
subtype DIGIT is integer range 0 to 9;
```



## Komplexe Typen (I)

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

69

- Array, analog zu Programmiersprachen

### Syntax Array

```
type array_name is array ( index_range |
                          index_type |
                          index_type range index_range |
                          index_type range <>) of
                          element_type;
```

- String: Array Typ von *character*
- Bit Vector: Array Typ von *bit*



## Komplexe Typen (II)

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

70

- Std\_Logic\_Vector: Array Typ zu *std\_logic*
- Record: Datenmodell um Elemente verschiedenen Typs zusammenzufassen

### Beispiel Record

```
type TWO_DIGITS is -- Zahlen von -99 bis +99
record SIGN : bit;
      MSD : integer range 0 to 9;
      LSD : integer range 0 to 9;
end record;
```



## Weitere Typen

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

71

- Zugriffstypen (Zeiger)

### Syntax

```
type prt_typename is access type_name;
```

- Operatoren: *new* für Zuweisungen, *Deallocate* um Speicherbereiche freizugeben
- Datei Typen
  - im package *TEXTIO* vordefinierte Typen z.B. *text*, *line* und *file*
  - Prozeduren: z.B. *readline*, *read*, *write*, *writeline*
  - Funktionen: z.B. *endline*



## Ausdrücke

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

72

- logische: *and*, *or*, *nand*, *nor*, *xor*
- relationale: *=*, */=*, *<*, *<=*, *>*, *>=*
- arithmetische *+*, *-*, *\**, */*, *mod*, *rem*, *&*
- weitere: *abs*, *not*, *\*\**



## Typattribute

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

73

- Dimensionierung
  - Bereichsgrenzen für Array- und Aufzählungstypen
  - Syntax: z.B. ...'left[(n)], ...'length[(n)] ...
- Ordnung
  - zur Ermittlung von Werten, Ordnungszahlen und übergeordneten Typen bei Aufzählungstypen
  - Syntax: z.B. type'succ(value), type'pos(value)



## Signalbezogene Attribute

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

74

- Berücksichtigung von dynamischem Signalverhalten
- Auswertung von Attributen, Simulationsereignissen und Zeitpunkten während der Laufzeit der Simulation

### Syntax

```

signal 'event           -- true wenn Signaländerung
signal 'active          -- true wenn Signalzuweisung
signal 'last_event     -- Zeitdifferenz zur letzten
                       -- Signaländerung
signal 'last_value     -- Wert vor der letzten

```



## Literatur

VHDL

D. Neuhäuser,  
G. Grune  
(W. Koch)

Einleitung

Entity

Architecture  
Nebenläufig  
Sequenziell  
Strukturell

Konfiguration

Packages

Datentypen  
Skalare DT  
Komplexe DT  
Weitere DT

Ausdrücke

Attribute

Literatur

75

- VHDL - Eine Einführung; P. Molitor, J. Ritter; Pearson Studium Verlag, München, 2004; ISBN 3-8273-7047-7; INF-LB-4000-M725
- VHDL-Synthese; J. Reichardt, B. Schwarz; Oldenbourg Verlag, München, 2001; ISBN 3-486-25809-5; INF-LH-1000-R348
- VHDL-Informationsmedium der FH Köln: VHDL-easy, <http://www.nt-nv.fh-koeln.de/Labor/VhdlEasy/index.html>
- VHDL-Info; Universität Ulm, [http://mikro.e-technik.uni-ulm.de/vhdl/vhdl\\_infos.html](http://mikro.e-technik.uni-ulm.de/vhdl/vhdl_infos.html)
- VHDL-Script zur Vorlesung CT-2; G. Grune; Universität Jena, 2004