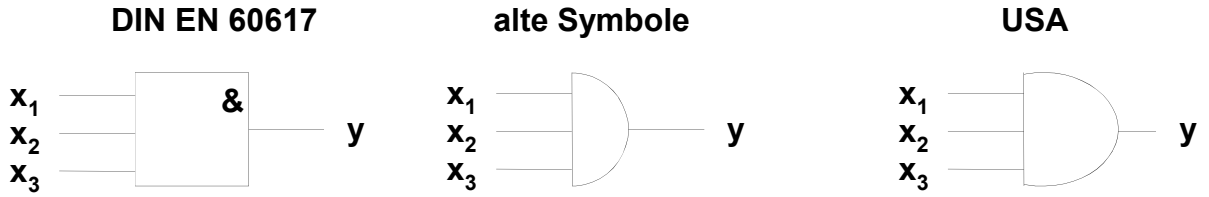
Beginn 10:00

Foliensatz zum Download

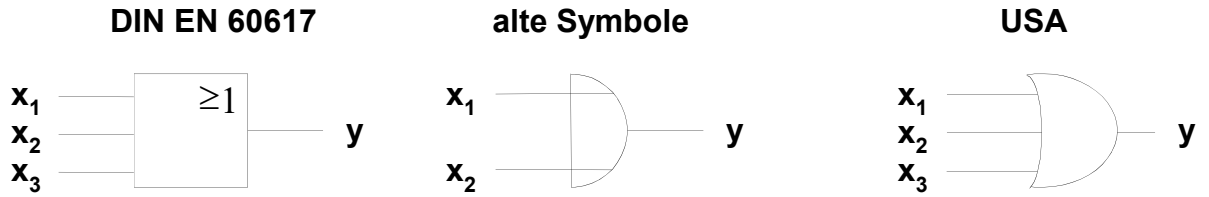
http://dodo.fb06.fh-muenchen.de/hermann/digital/mfm_digital_v3.pdf

Grundgatter

- **UND (AND)**

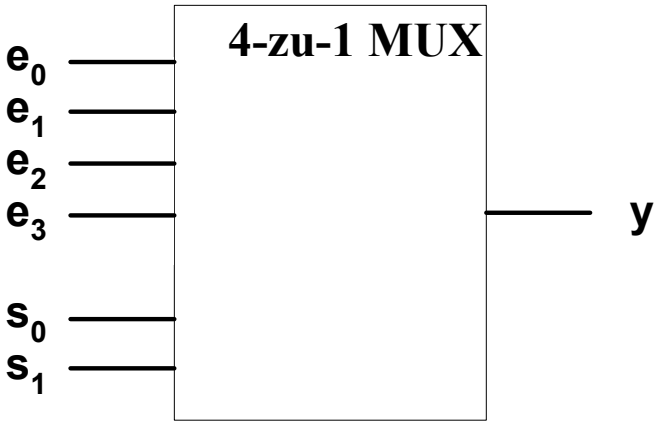


- **ODER (OR)**



Multiplexer

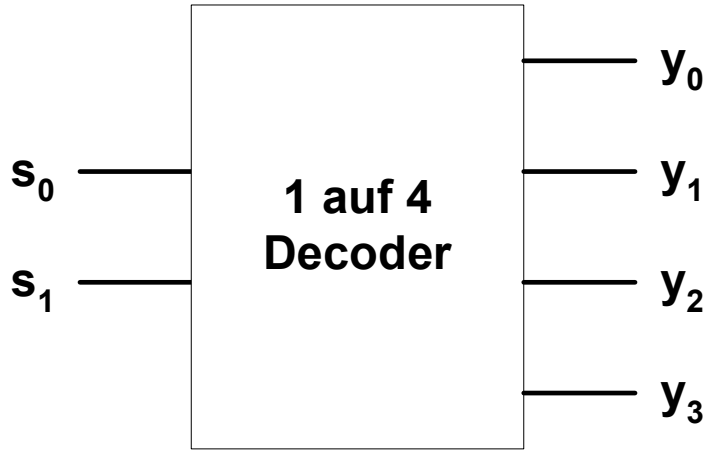
- **Aufgabe: Durchschalten eines von n Eingängen auf einen einzigen Ausgang**



s1	s0	e3	e2	e1	e0	Y
0	0					
0	0					
0	1					
0	1					

Demultiplexer/Dekoder

- **Aufgabe: Aktivieren genau eines Ausgangs von n Ausgängen**



s1	s0	y0	y1	y2	y3
0	0				
0	1				
1	0				
1	1				

Codes

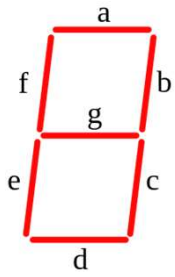
- **Abbildung eines Zeichensatzes auf einen anderen Zeichensatz**
  - ASCII-Code: Ziffern und Buchstaben mit 7 Bit
  - Dualzahl: Zahlen im Dualsystem
  - int. Vorwahl: internationale Telefonvorwahl

Hexadezimale Darstellung 41

Code	Bedeutung
ASCII	A
Dualzahl	65 (dezimal)
int. Vorwahl	Schweiz

Codes, Wahrheitstabellen, Funktionen

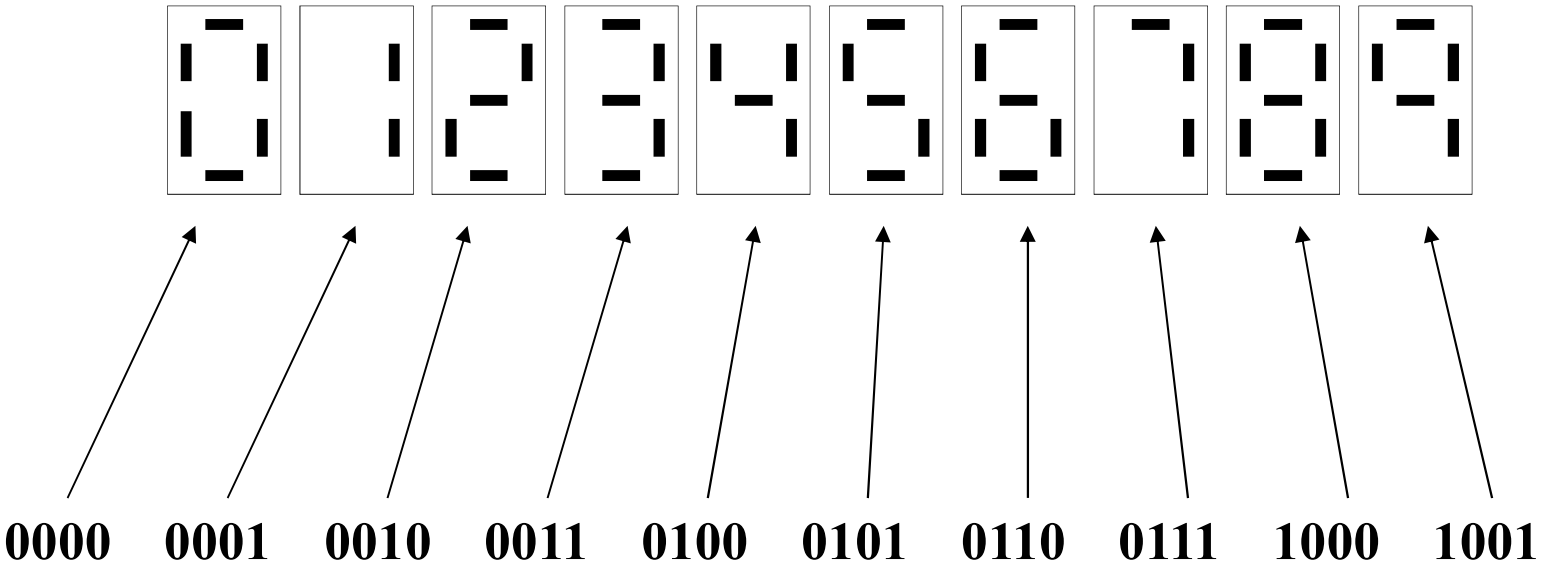
Beispiel: Siebensegment-Dekoder



- Betrachtung als Code: Abbildung von vierstelligen Dualzahlen auf siebenstellige Siebensegmentanzeigemuster
- Betrachtung als Wahrheitstabelle: Auflistung aller zehn Wertemuster und Angabe der resultierenden Wertemuster für die zugehörigen Segmente der Anzeige
- Betrachtung als Funktion: Aufstellen von sieben Einzelfunktionen (je eine für ein Segment der Anzeige) in Abhängigkeit von vier Variablen

Siebensegmentdekoder (Code)

Zeichenvorrat des Codes Siebensegment



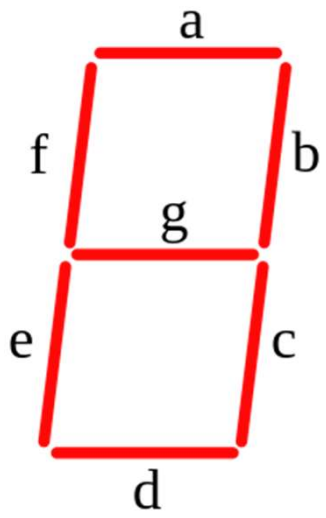
Zeichenvorrat des Codes Dualzahl

Siebensegmentdekoder (Tabelle)

Eingänge

$$Z = \sum_{i=0}^{3} x_i \times 2^i$$

Ausgänge



Eingänge				Ausgänge						
x_3	x_2	x_1	x_0	a	b	c	d	e	f	g
0	0	0	0							
0	0	0	1							
0	0	1	0							
0	0	1	1							
0	1	0	0							
0	1	0	1							
0	1	1	0							
0	1	1	1							
1	0	0	0							
1	0	0	1							

Siebensegmentdekoder (Funktionen)

Ableitung der Einzelfunktionen aus der Tabelle

$$a = ...$$

$$b = ...$$

$$c = ...$$

$$d = ...$$

$$e = \bar{x}_3\bar{x}_2\bar{x}_1\bar{x}_0 + \bar{x}_3\bar{x}_2x_1\bar{x}_0 + \bar{x}_3x_2x_1\bar{x}_0 + x_3\bar{x}_2\bar{x}_1\bar{x}_0$$

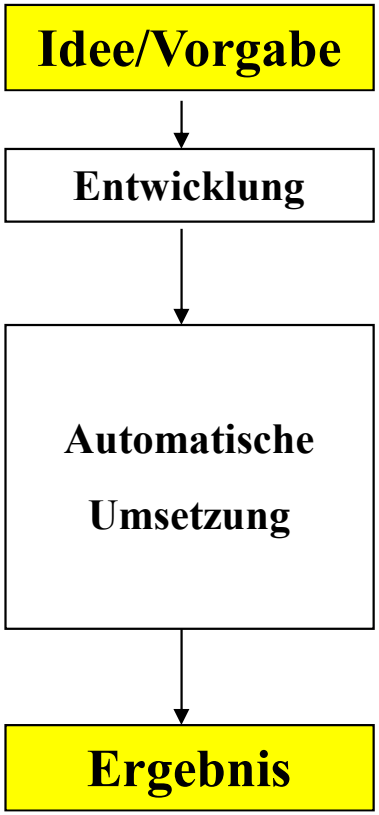
$$f = ...$$

$$g = ...$$

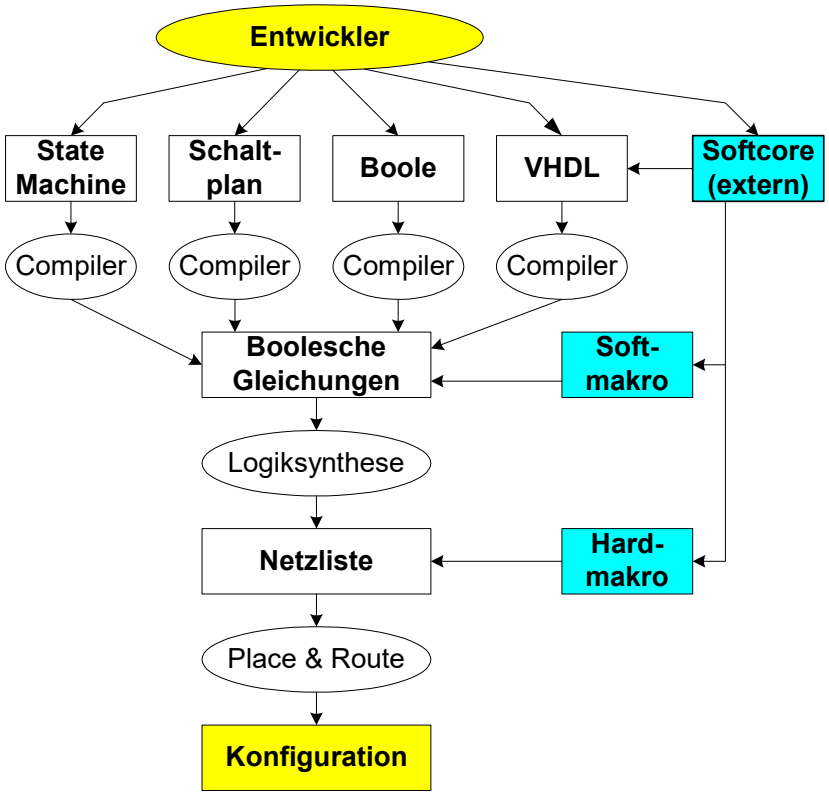
Entwurfswerkzeuge

- **Früher: Zeichenbrett, Lötkolben, Steckbrett, Oszilloskop**
- **Heute:**
  - **Entwurfswerkzeuge** für
    - Schaltpläne (mit Gatterbibliotheken und Konsistenzprüfung)
    - Funktionsgleichungen (inklusive automatischer Minimierung)
    - Verhaltensbeschreibungen (VHDL, Verilog)
  - **Compiler** für
    - Flächen- bzw. Laufzeitoptimale Umsetzung in beliebige Technologien
  - **Simulatoren** für
    - Funktionstest am Rechner
  - **Emulatoren** für
    - Funktionstest in Hardware

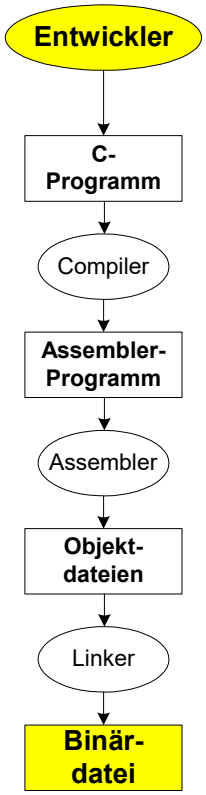
Entwurfsprozess



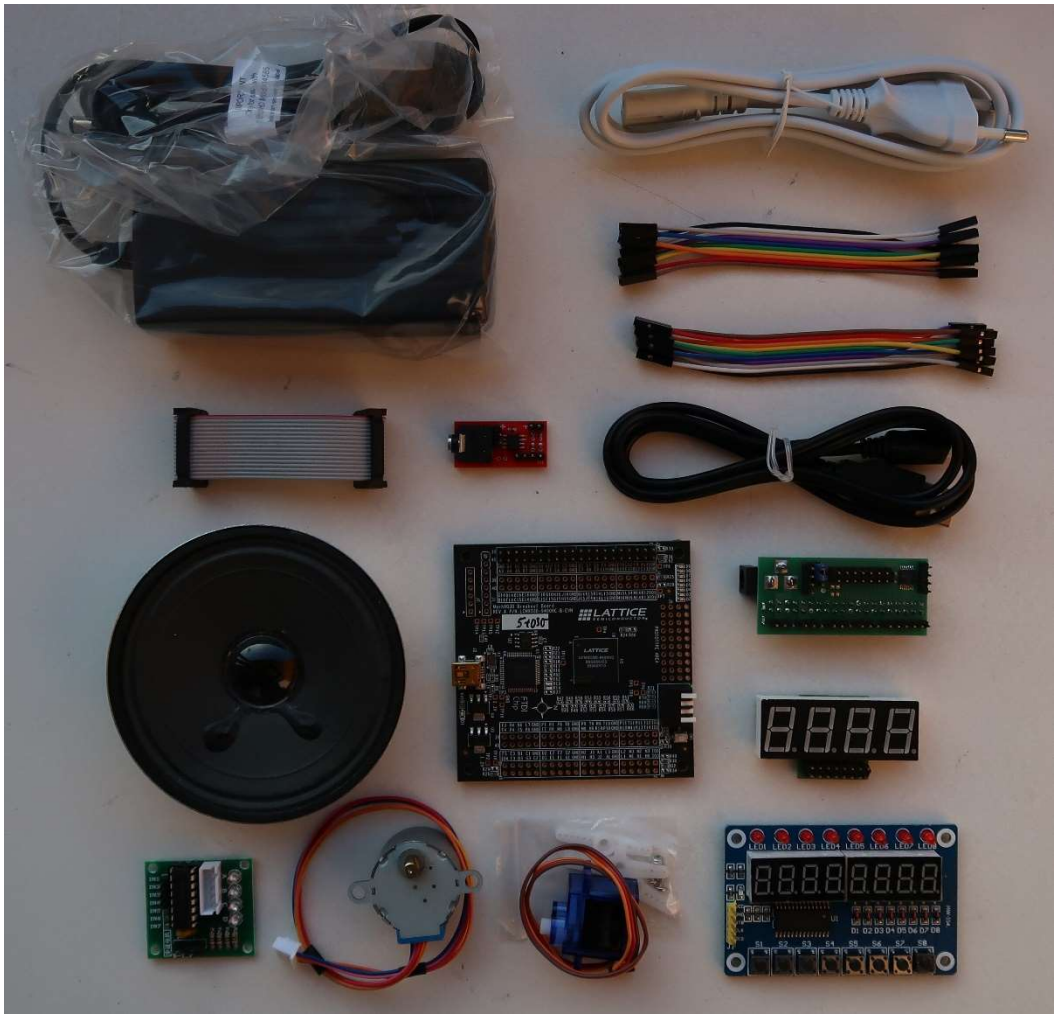
Entwurfsablauf FPGA



Vergleich: Software



Kit im WS20/21



1. Ausgabetermin

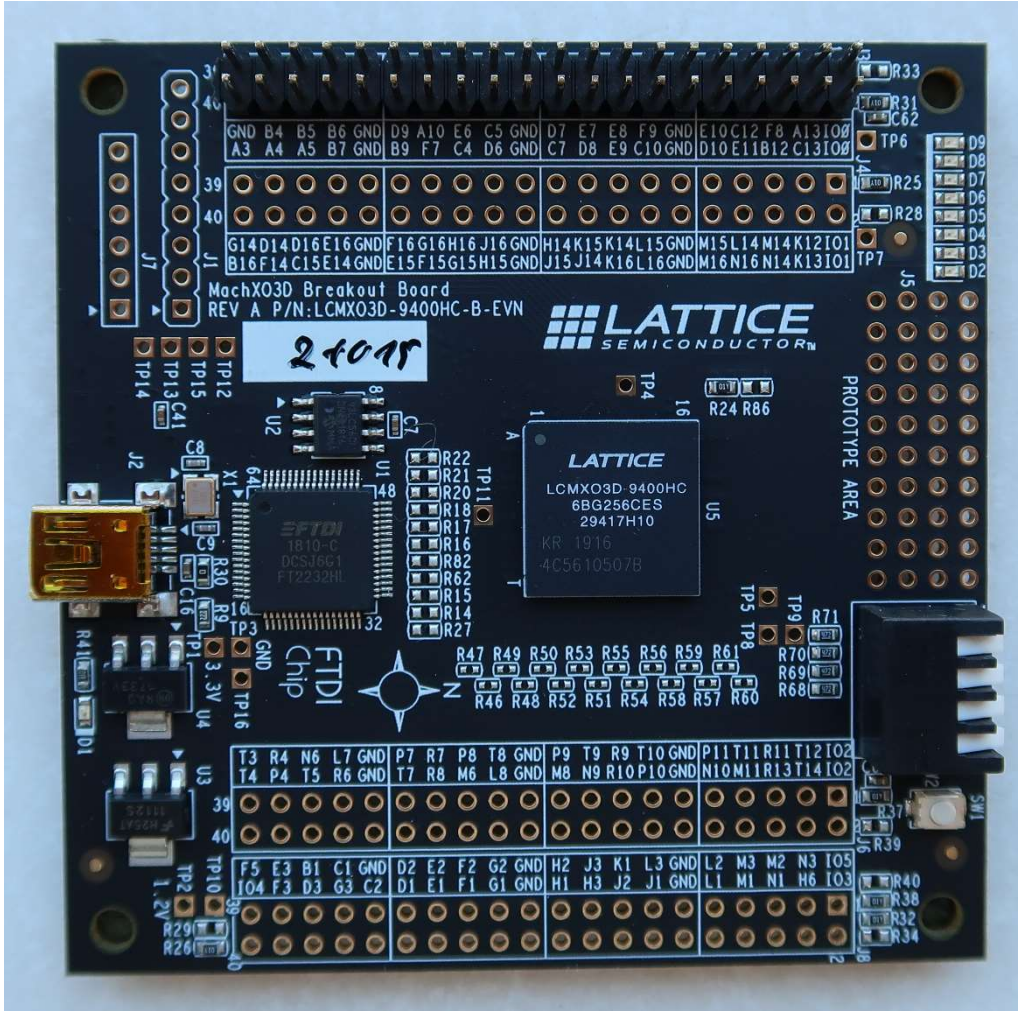
Freitag, 0730-1130
Raum D205

Aktuell 40, daher
besser vorher E-Mail

Pfand 50€

Folgetermin (10)
vor. Ende Oktober

Baseboard


MachXO3D Breakout Board
REV A P/N:LCMXO3D-9400HC-B-EVN
LATTICE
SEMICONDUCTOR
LATTICE
LCMXO3D-9400HC
6BG256CES
29417H10
PROTOTYPE AREA
FTDI
Chip

VHDL

- **HDL: Hardware Description Language**
  - Programmiersprachenähnliche Syntax
  - keine Programmiersprache (im Sinne von: Befehle werden in einer gegebenen Reihenfolge abgearbeitet)

- Ursprünglich für **Simulationen** gedacht, jedoch heute neben Verilog Standardsprache für **Schaltungsentwurf**
  - Nur eine Teilmenge des Sprachumfangs ist synthetisierbar, d.h., kann in eine Schaltung „gegossen“ werden

Allgemeine Angaben

- **Formatfreie Sprache**
  - Einrückungen und Zeilenumbrüche belanglos
- **Keine Unterscheidung Groß-/Kleinschreibung**
- **Namen**
  - müssen mit A-Z beginnen
  - danach auch _ und 0-9 zulässig
- **Kommentar**
  - Immer Zeilenendkommentar, wird mit -- eingeleitet
- **Häufiger Abschluss eines Blocks mit end**

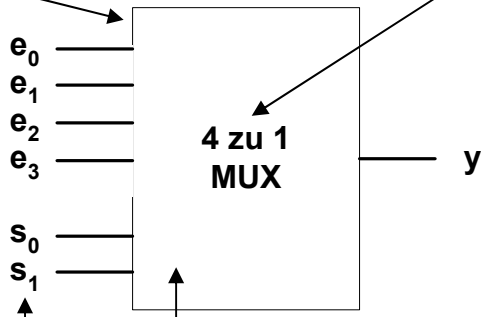
VHDL – Grundlagen

entity

beschreibt die Schnittstelle des Funktionsblocks

architecture

beschreibt die Funktion im Inneren der entity



signal

benennt Verbindungsleitungen
- im Inneren
- nach aussen

VHDL - Dateiaufbau

```
library ieee;
use ieee.std_logic_1164.all;


entity MUX_4_to_1 is
port
(
)
end MUX_4_to_1;


architecture VERHALTEN of MUX_4_to_1 is

begin


end VERHALTEN;
```

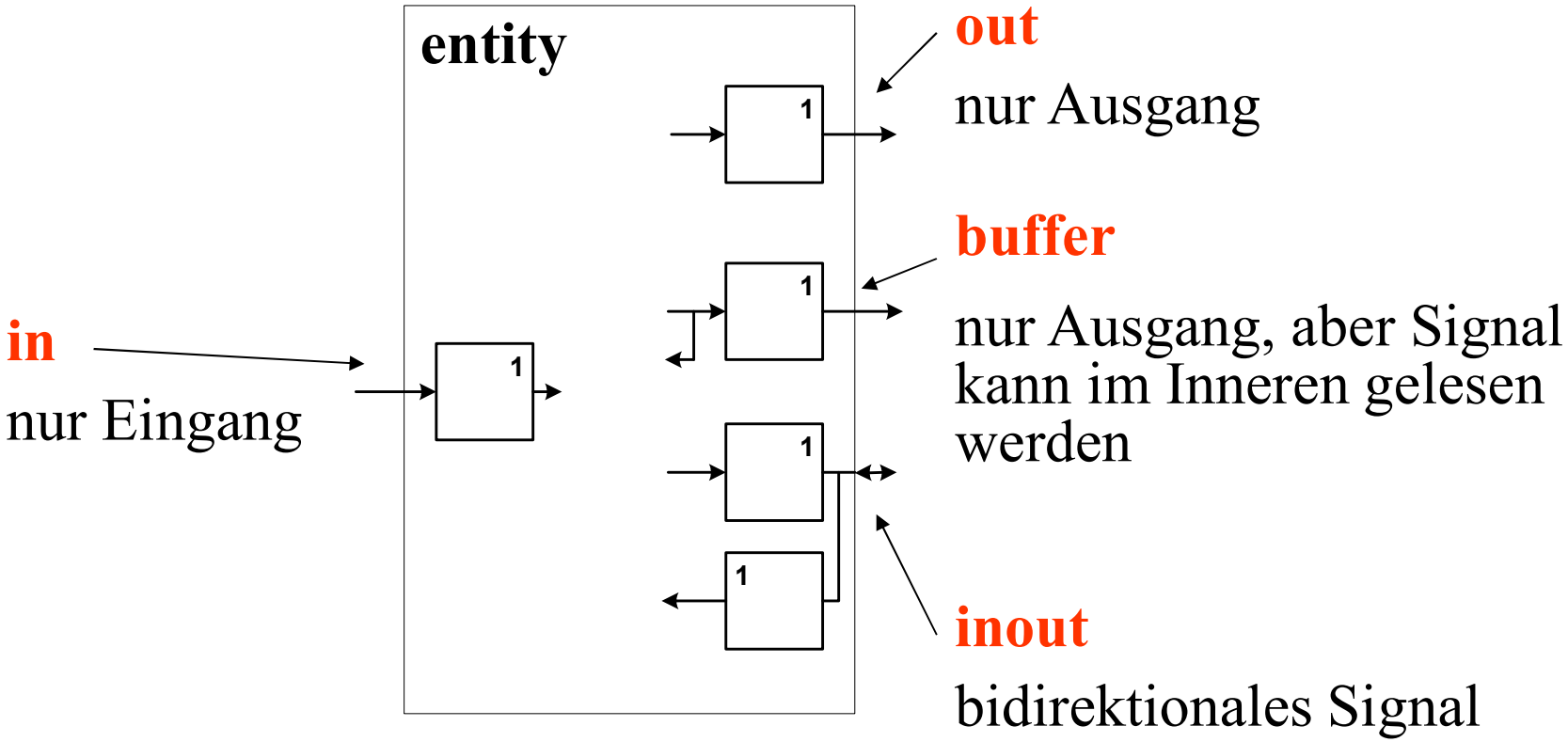
Angabe der verwendeten Definitionen

Beschreibung der Schnittstelle des Multiplexers

Beschreibung des Verhaltens des Multiplexers

VHDL - Ports

- **Richtungen an der entity können sein:**



VHDL – log. Verknüpfungen

- **Zuweisungen an Signale** erlauben die üblichen logischen Verknüpfungen
  - Y <= x1 **and** x2;
  - Z <= **not** x1;

- **Priorität**
  - **not** hat Vorrang
  - **and, or, xor, nand, nor, xor** sind gleichberechtigt
    - Priorisierung durch Klammerung y <= (x1 **and** x2) **or** x3;

VHDL – Typologie

- **Signale haben immer einen Typ**
- **Grundlegende Typen für den Entwurf:**
  - boolean (true, false)
  - bit (0,1)
  - std_logic (0,1,-,Z,U,X,W,L,H)
  - integer, natural, positive (Zahlen)

- **Typkonversionen sind möglich, müssen aber explizit angegeben werden**

VHDL – boolean und bit

- **boolean**
  - kann die Werte **true** und **false** annehmen
  - wird bei Bedingungen benötigt (if *ausdruck* then ...) wobei *ausdruck* vom Typ boolean sein **muss**
  - Bsp: signal v: boolean;
v <= true;
- **bit**
  - kann die Werte **'0'** und **'1'** annehmen
  - Bsp.: signal y: bit;
y <= '1';

VHDL - std_logic

- **std_logic wird bei der Simulation und der Synthese verwendet**
  - Standardtyp in VHDL
  - hat 9 Werte (0,1,-,Z,X,U,H,L,W), von denen 0,1 und Z für die Synthese am wichtigsten sind
  - ist nicht einfach eine Obermenge von bit

VHDL – std_logic

- **Synthese und Simulation**

  1, 0: entsprechend Bitwerten 0,1

  Z: hochohmig (tristate)

- **Simulation**

  X: unbestimmt, schwerer Konflikt (0 gegen 1)

  W: unbestimmt, ungefährlicher Konflikt (L gegen H)

  U: nicht initialisierter Wert bei der Simulation

  H, L: schwache 1, schwache 0

- **don‘t care: kaum unterstützt**

Architecture – Dateiaufbau