# Digitaltechnik

## Wintersemester 2025/2026

## Vorlesung 4

TECHNISCHE UNIVERSITÄT DARMSTADT

ENCRYPTO
CRYPTOGRAPHY AND PRIVACY ENGINEERING

# Inhalt

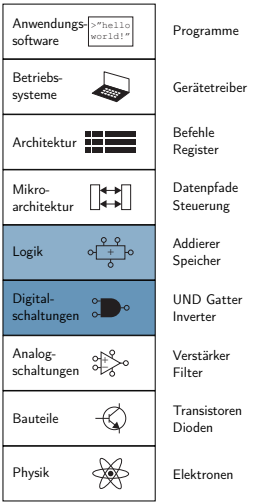

1 Boole'sche Gleichungen

2 Kombinatorische Logik

3 SystemVerilog für kombinatorische Logik

4 SystemVerilog Modulhierarchie


Harris 2016
Kap. 4.1 - 4.3, 2.2



ENCRYPTO
CRYPTOGRAPHY AND
PRIVACY ENGINEERING

# Abgabefrist für Hausaufgabe A zu Vorlesungen 01 und 02 **diese** Woche Freitag 23:59!

Wöchentliches Moodle-Quiz nicht vergessen!

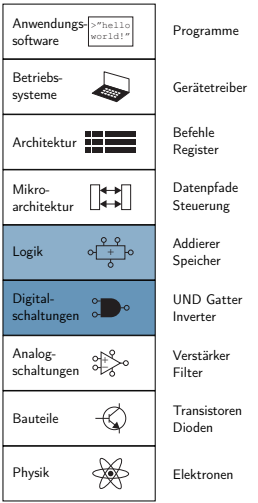
# Agenda


1 Boole'sche Gleichungen

2 Kombinatorische Logik

3 SystemVerilog für kombinatorische Logik

4 SystemVerilog Modulhierarchie

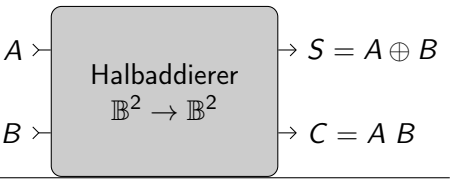



- beschreiben Ausgänge einer kombinatorischen Schaltung als (boole'sche) Funktion der Eingänge

$\Rightarrow$ Spezifikation des funktionalen Verhaltens (ohne zeitliche Information)

- unter Verwendung elementarer boole'scher Operatoren (sortiert nach *Operatorpräzedenz*):
  - NOT: $\overline{A}$
  - AND: $A\ B = A \cdot B$
  - XOR: $A \oplus B$
  - OR: $A + B$

- Beispiel: Halbaddierer (später in VL07)
  - $S = F_1 : (A, B) \in \mathbb{B}^2 \mapsto \mathbb{B}$
  - $C = F_2 : (A, B) \in \mathbb{B}^2 \mapsto \mathbb{B}$



**Komplement:** boole'sche Variable mit einem Balken (invertiert)
$\overline{A}, \overline{B}, \overline{C}$

**Literal:** Variable oder ihr Komplement
$A, \overline{A}, B, \overline{B}, C, \overline{C}$

**Implikant:** Produkt von Literalen
$ABC, A\overline{C}, BC$

**Minterm:** Produkt (UND, Konjunktion) über alle Eingangsvariablen
$ABC, AB\overline{C}, \overline{A}BC$

**Maxterm:** Summe (ODER, Disjunktion) über alle Eingangsvariablen
$(A + \overline{B} + \overline{C}), (A + B + \overline{C}), (\overline{A} + \overline{B} + \overline{C})$

# Minterm

- Produkt (Implikant), das jede Eingangsvariable *genau einmal* enthält
- entspricht einer Zeile in Wahrheitswertetabelle
- jeder Minterm wird für *genau eine* Eingangskombination ***wahr*** (unabhängig von Ergebnisspalte)

| A | B | Y | Minterm |
|---|---|---|---|
| 0 | 0 | 0 | $m_0 = \overline{A}\ \overline{B}$ |
| 0 | 1 | 1 | $m_1 = \overline{A}\ B$ |
| 1 | 0 | 1 | $m_2 = A\ \overline{B}$ |
| 1 | 1 | 0 | $m_3 = A\ B$ |

- Summe aller Minterme, für welche die Funktion *wahr* ist

$\Rightarrow$ jede boole'sche Funktion hat *genau eine* DNF (abgesehen von Kommutation)

- Im Beispiel: $Y = m_1 + m_2 = \overline{A}\ B + A\ \overline{B}$

$\Rightarrow$ $A \oplus B$ nur kompakte Schreibweise für $\overline{A}\ B + A\ \overline{B}$

| A | B | Y | Minterm |
|---|---|---|---|
| 0 | 0 | 0 | $m_0 = \overline{A}\ \overline{B}$ |
| 0 | 1 | 1 | $m_1 = \overline{A}\ B$ |
| 1 | 0 | 1 | $m_2 = A\ \overline{B}$ |
| 1 | 1 | 0 | $m_3 = A\ B$ |

# Maxterm

- Summe, welche jede Eingangsvariable *genau einmal* enthält
- entspricht einer Zeile in Wahrheitswertetabelle
- jeder Maxterm wird für *genau eine* Eingangskombination ***falsch*** (unabhängig von Ergebnisspalte)

| A | B | Y | Maxterm |
|---|---|---|---|
| 0 | 0 | 0 | $M_0 = A + B$ |
| 0 | 1 | 1 | $M_1 = A + \overline{B}$ |
| 1 | 0 | 1 | $M_2 = \overline{A} + B$ |
| 1 | 1 | 0 | $M_3 = \overline{A} + \overline{B}$ |

- Produkt aller Maxterme, für welche die Funktion *falsch* ist

$\Rightarrow$ jede boole'sche Funktion hat *genau eine* KNF (abgesehen von Kommutation)

- Im Beispiel: $Y = M_0\ M_3 = (A + B)\ (\overline{A} + \overline{B})$

$\Rightarrow$ $A \oplus B$ nur kompakte Schreibweise für $(A + B)\ (\overline{A} + \overline{B})$

| A | B | Y | Maxterm |
|---|---|---|---|
| 0 | 0 | 0 | $M_0 = A + B$ |
| 0 | 1 | 1 | $M_1 = A + \overline{B}$ |
| 1 | 0 | 1 | $M_2 = \overline{A} + B$ |
| 1 | 1 | 0 | $M_3 = \overline{A} + \overline{B}$ |

ENCRYPTO
CRYPTOGRAPHY AND
PRIVACY ENGINEERING

# Quiz' 2 - Electric Boogaloo

(Hat nichts mit dem Moodle-Quiz zu tun)

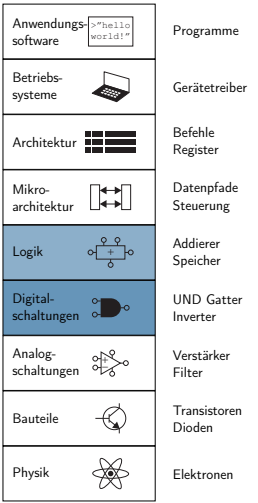
# Agenda


1 Boole'sche Gleichungen

2 Kombinatorische Logik

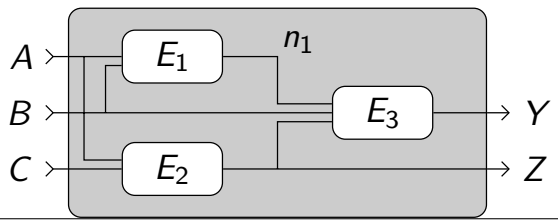
3 SystemVerilog für kombinatorische Logik

4 SystemVerilog Modulhierarchie

- Eingänge
- Ausgänge
- Spezifikation des Funktionalen Verhaltens = realisierte (boole'sche) Funktion
- Spezifikation des Zeitverhaltens

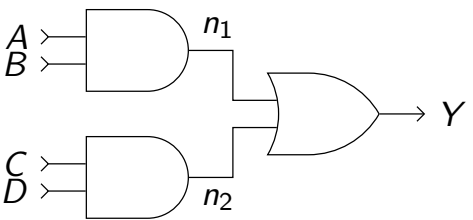
- Verbindungsknoten
  - Eingangs-Terminale: $A, B, C$
  - Ausgangs-Terminale: $Y, Z$
  - Interne Knoten: $n_1$
- Schaltungselemente
  - $E_1, E_2, E_3$
  - jedes selbst eine Schaltung $\rightarrow$ Hierarchie



- kombinatorische Logik („Schaltnetz“)
  - Ausgänge hängen nur von *aktuellen* Eingangswerten ab
- sequentielle Logik („Schaltwerk“, in späteren Vorlesungen)
  - Ausgänge hängen von aktuellen Eingangswerten und **internem** Zustand ab

  $\Rightarrow$ Ausgänge indirekt abhängig von *vorherigen* Eingangswerten

- jedes Schaltungselement ist selbst kombinatorisch
- jeder Verbindungsknoten ist
  - Eingang in die Schaltung, oder
  - an *genau ein* Ausgangsterminal („Treiber“) eines Schaltungselements angeschlossen
- jeder Pfad durch die Schaltung besucht jeden Verbindungsknoten maximal einmal (*zyklenfrei*)



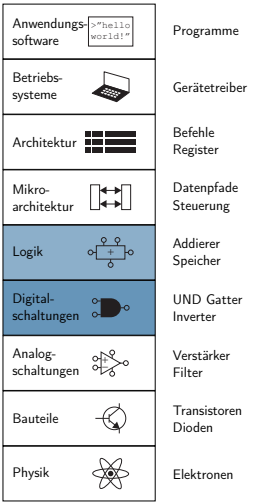
# Agenda


1 Boole'sche Gleichungen

2 Kombinatorische Logik

3 SystemVerilog für kombinatorische Logik

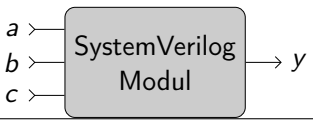
4 SystemVerilog Modulhierarchie

# SystemVerilog Module

- Ein *Modul* beschreibt wie eine Aufgabe (Berechnung) durchgeführt wird
  - Ähnlich einer *Funktion* in Programmiersprachen
- Modulbeschreibung:
  - Eingänge
  - Ausgänge
  - (Parameter)
- zwei Arten von Modul-Beschreibungen:
  - Strukturbeschreibung: Wie ist die Schaltung aus (Sub-)Modulen aufgebaut?
  - Verhaltensbeschreibung: Was tut die Schaltung?

$\Rightarrow$ strukturelle Modul-Hierarchie mit Verhaltensbeschreibung auf unterster Ebene



# Beispiel für Verhaltensbeschreibung



example.sv

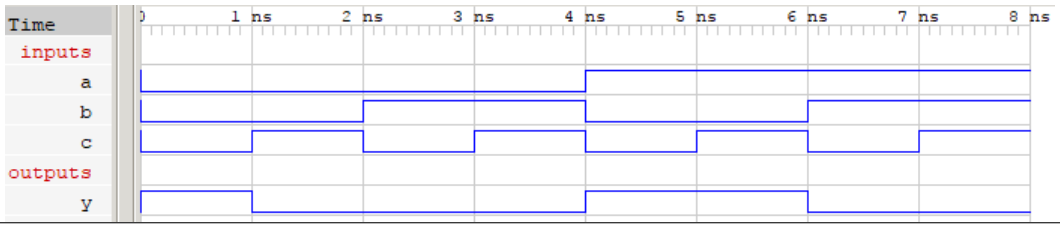
```
1  module example(input logic a, b, c, output logic y);
2
3    assign y = ~a & ~b & ~c | a & ~b & ~c | a & ~b & c;
4
5  endmodule
```

- `module` Beginn der Schnittstellenbeschreibung
- `example` Modulname
- `input, output` Port-Richtung
- `logic` Port-Datentyp
- `a,b,c,y` Port-Namen
- `assign` (kombinatorische) Signalzuweisung
- `~,&,|` (kombinatorische) Operatoren (NOT, AND, OR)
- `endmodule` Ende der Schnittstellenbeschreibung

example.sv

```
1  module example(input logic a, b, c, output logic y);
2
3    assign y = ~a & ~b & ~c | a & ~b & ~c | a & ~b & c;
4
5  endmodule
```

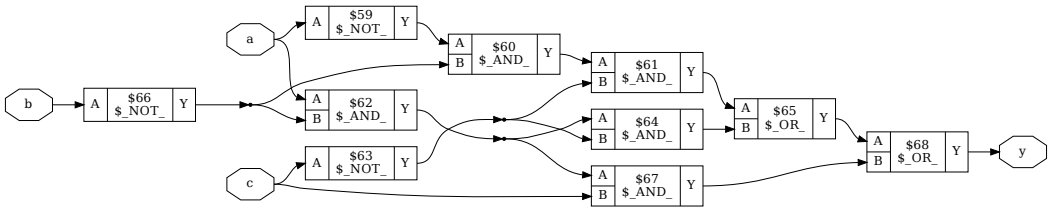
Plot mit Open-Source Tools Icarus Verilog + GTKWave



example.sv

```
1  module example(input logic a, b, c, output logic y);
2
3    assign y = ~a & ~b & ~c | a & ~b & ~c | a & ~b & c;
4
5  endmodule
```

Plot mit Open-Source Tools YoSyS + GraphViz



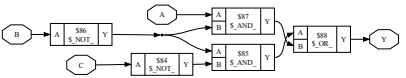
# Synthese vs. Simulation

simple.sv

```
1 module simple(input  logic A, B, C,
2                output logic Y);
3
4   assign Y = ~B & ~C | A & ~B;
5
6 endmodule
```

Synthese

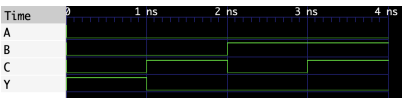
z.B. ./synth.sh simple




Simulation

z.B. ./sim.sh simple

```
VCD info: dumpfile simple_selftest_tb.vcd opened for output.
VCD warning: $dumpvars: Package ($unit) is not dumpable with VCD.
000 ok.
001 ok.
010 ok.
011 ok.
FINISHED simple_selftest_tb
simple_selftest_tb.sv:28: $finish called at 400 (10ps)
```




Testbench

z.B. simple_tb.sv

z.B. GTKWave

- Unterscheidet Groß- und Kleinschreibung
  - bspw. `reset` $\neq$ `Reset`
- Bezeichner für Modul- und Signalnamen dürfen nicht mit Ziffern anfangen
  - bspw. `2mux` ungültig
- Anzahl von Leerzeichen, Leerzeilen und Tabulatoren irrelevant
- Kommentare:
  - `// Kommentar bis zum Ende der Zeile`
  - `/* Kommentar über`
    `mehrere Zeilen */`

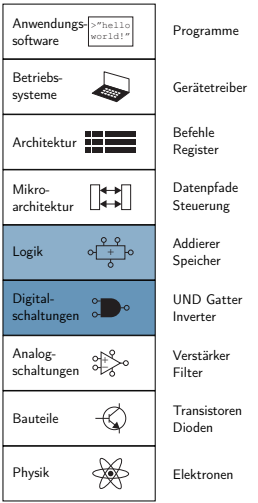
# Agenda


1 Boole'sche Gleichungen

2 Kombinatorische Logik

3 SystemVerilog für kombinatorische Logik

4 SystemVerilog Modulhierarchie

and3.sv

```
1 module and3(input logic a, b, c, output logic y);
2   assign y = a & b & c;
3 endmodule
```

inv.sv

```
1 module inv(input logic a, output logic y);
2   assign y = ~a;
3 endmodule
```

nand3.sv

```
1 module nand3 (input logic d, e, f, output logic w);
2   logic s;                  //internes Signal für Modulverbindung
3   and3 andgate(d, e, f, s);//Instanz von and3 namens andgate
4   inv  inverter(s, w);      //Instanz von inv namens inverter
5 endmodule
```

nand3.sv

```
1 module nand3 (input logic d, e, f, output logic w);
2   logic s;                  //internes Signal für Modulverbindung
3   and3 andgate(d, e, f, s);//Instanz von and3 namens andgate
4   inv  inverter(s, w);      //Instanz von inv namens inverter
5 endmodule
```

nand3_named.sv

```
1 module nand3_named(input logic d, e, f, output logic w);
2   logic s;
3   and3 andgate(.a(d), .b(e), .c(f), .y(s));
4   inv  inverter(.a(s), .y(w));
5 endmodule
```

- 10 bis 100 ports pro Modul nicht unüblich

$\Rightarrow$ absolute Portzuweisung per Namen übersichtlicher (selbstdokumentierend)

# Zusammenfassung und Ausblick

ENCRYPTO
CRYPTOGRAPHY AND
PRIVACY ENGINEERING

1. Boole'sche Gleichungen
2. Kombinatorische Logik
3. SystemVerilog für kombinatorische Logik
4. SystemVerilog Modulhierarchie

nächste Vorlesung beinhaltet

- Boole'sche Algebra
- Umformung von Schaltungen: Bubble Pushing
- Logikrealisierung mit Basisgattern

**Hausaufgabe A zu Vorlesungen 01 und 02 muss bis diese Woche Freitag 23:59 abgegeben werden. Wöchentliches Moodle-Quiz nicht vergessen!**