

Digitaltechnik

Wintersemester 2021/2022

13. Vorlesung



TECHNISCHE
UNIVERSITÄT
DARMSTADT














Umfrage zur letzten Woche

1. Einleitung

2. Speicherfelder

3. Logikfelder

4. Zusammenfassung

Anwendungs- software		Programme
Betriebs- systeme		Gerätetreiber
Architektur		Befehle Register
Mikro- architektur		Datenpfade Steuerung
Logik		Addierer Speicher
Digital- schaltungen		UND Gatter Inverter
Analog- schaltungen		Verstärker Filter
Bauteile		Transistoren Dioden
Physik		Elektronen

Agenda



TECHNISCHE
UNIVERSITÄT
DARMSTADT

1. Einleitung

2. Speicherfelder

3. Logikfelder

4. Zusammenfassung

Anwendungs- software		Programme
Betriebs- systeme		Gerätetreiber
Architektur		Befehle Register
Mikro- architektur		Datenpfade Steuerung
Logik		Addierer Speicher
Digital- schaltungen		UND Gatter Inverter
Analog- schaltungen		Verstärker Filter
Bauteile		Transistoren Dioden
Physik		Elektronen

Überblick der heutigen Vorlesung



TECHNISCHE
UNIVERSITÄT
DARMSTADT

- ▶ Speicherfelder (Fortsetzung)
- ▶ Logikfelder



Harris 2013/2016
Kap. 5.5 - 5.6

Agenda



TECHNISCHE
UNIVERSITÄT
DARMSTADT

1. Einleitung

2. Speicherfelder

3. Logikfelder

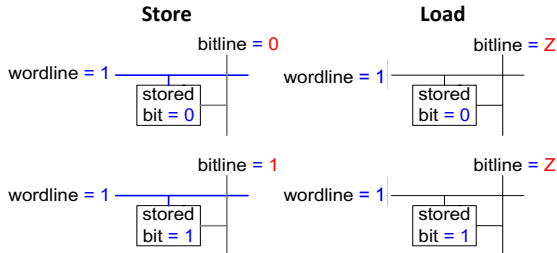
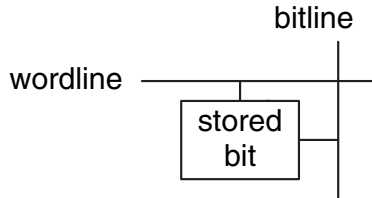
4. Zusammenfassung

Anwendungs- software		Programme
Betriebs- systeme		Gerätetreiber
Architektur		Befehle Register
Mikro- architektur		Datenpfade Steuerung
Logik		Addierer Speicher
Digital- schaltungen		UND Gatter Inverter
Analog- schaltungen		Verstärker Filter
Bauteile		Transistoren Dioden
Physik		Elektronen

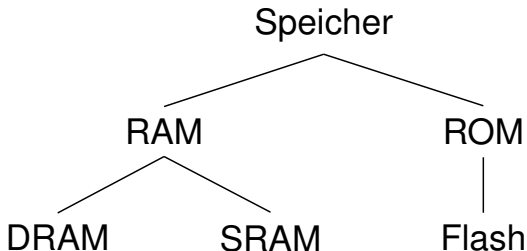
Wiederholung: Speicherfeld Bitzellen (aus VL 12)



TECHNISCHE
UNIVERSITÄT
DARMSTADT



- ▶ Direktzugriffsspeicher (*random access memory*, RAM): **flüchtig**
 - ▶ Dynamic RAM (DRAM)
 - ▶ Static RAM (SRAM)
- ▶ Festwertspeicher (*read-only memory*, ROM): **nicht flüchtig**
 - ▶ Flash





- ▶ **Flüchtig:** verliert Daten beim Ausschalten
- ▶ Schnelles Lesen und Schreiben
- ▶ Der Hauptspeicher Ihres Computers ist RAM (meist DRAM)

Historisch als Direktzugriffsspeicher bezeichnet, da auf jedes Datenwort gleich schnell / direkt zugegriffen werden kann (im Gegensatz zu sequentiellen Zugriffsspeichern wie Audiokassette oder Bandlaufwerk)

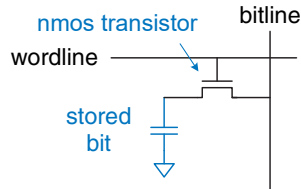
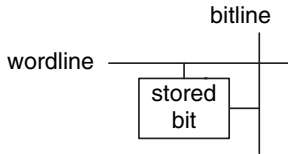
- ▶ **Nicht flüchtig:** Daten bleiben beim Ausschalten erhalten
- ▶ Schnelles Lesen, aber Schreiben ist unmöglich oder langsam
- ▶ Flash-Speicher in Digitalkameras, USB-Sticks und SSDs sind alles ROMs

Historisch als Festwertspeicher bezeichnet, da ROMs zum Zeitpunkt der Herstellung oder durch Brennen von Sicherungen geschrieben wurden. Sobald das ROM konfiguriert wurde, konnte es nicht erneut geschrieben werden. Dies gilt nicht mehr für Flash-Speicher und andere Arten von ROMs.



- ▶ **DRAM** (*dynamic random access memory*)
 - ▶ DRAM verwendet Kondensator zur Datenspeicherung
- ▶ **SRAM** (*static random access memory*)
 - ▶ verwendet Inverter mit Rückkopplung zur Datenspeicherung

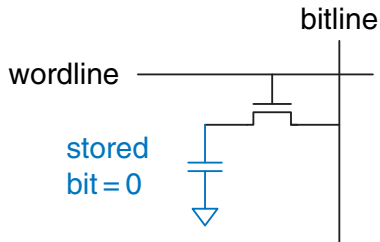
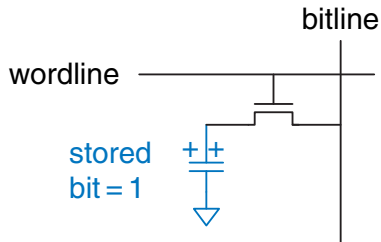
- ▶ Datenbits werden in Kondensator gespeichert
- ▶ *Dynamisch*, weil der Wert regelmäßig und nach dem Lesen aktualisiert (neu geschrieben) werden muss:
 - ▶ Ladungsverlust des Kondensators verschlechtert den Wert mit der Zeit ($1 \rightsquigarrow 0$)
 - ▶ Lesen zerstört den gespeicherten Wert



DRAM



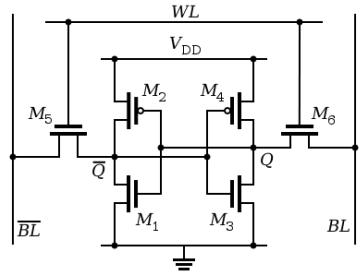
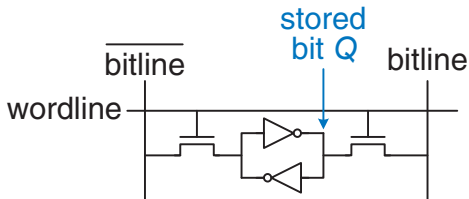
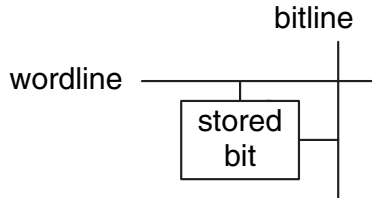
TECHNISCHE
UNIVERSITÄT
DARMSTADT



SRAM



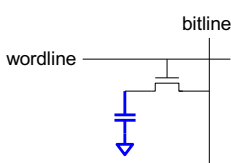
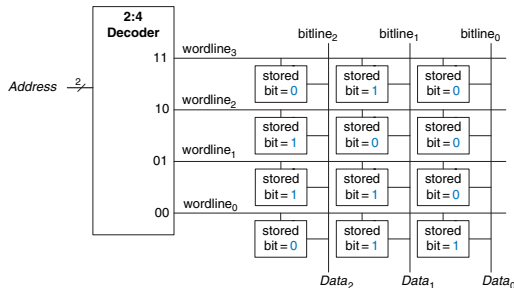
TECHNISCHE
UNIVERSITÄT
DARMSTADT



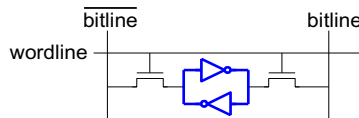
RAM Speicherfelder im Überblick



TECHNISCHE
UNIVERSITÄT
DARMSTADT



DRAM Bitzelle

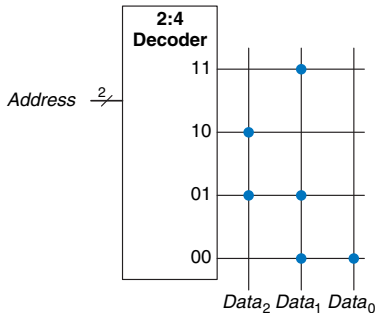


SRAM Bitzelle

ROM Punktnotation

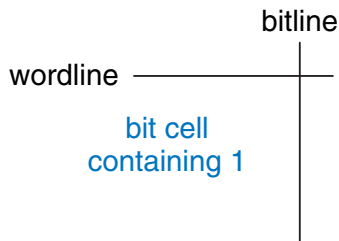
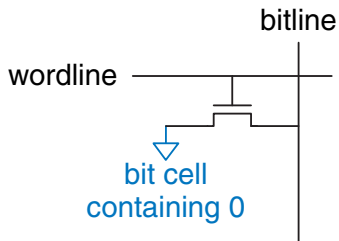


TECHNISCHE
UNIVERSITÄT
DARMSTADT



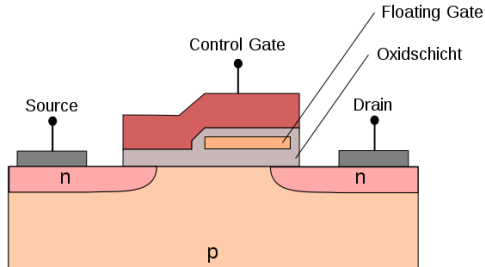
Address	Data			
11	0	1	0	depth ↑ ↓
10	1	0	0	
01	1	1	0	
00	0	1	1	
			width ↔	

Lesen: Bitline auf weak high und danach wordline auf 1 setzen. Wenn Transistor vorhanden, zieht dieser die bitline auf 0, sonst bleibt diese auf 1.



Floating gate kann durch Anlegen von hoher Spannung geladen / entladen werden.

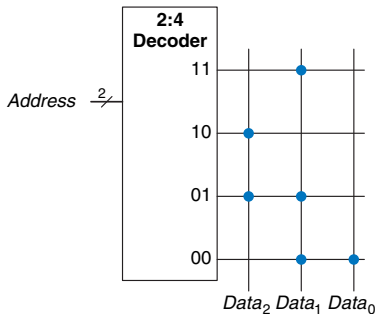
- ▶ Laden : Source = 0 V, Control Gate = Drain = 12 V
- ▶ Entladen: Source = offen, Control Gate = 0 V, Drain = 12 V



Logik via ROM



TECHNISCHE
UNIVERSITÄT
DARMSTADT



$$Address = A_1, A_0$$

$$Data_2 = A_1 \oplus A_0$$

$$Data_1 = \overline{A_1} + A_0$$

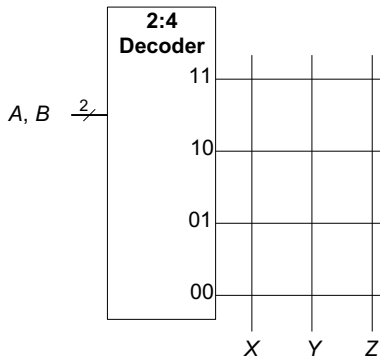
$$Data_0 = \overline{A_1} \overline{A_0}$$

Logik via ROM: Beispiel



Implementieren Sie die folgenden Logikfunktionen mit einem $2^2 \times 3$ -Bit ROM:

- ▶ $X = A B$
- ▶ $Y = A + B$
- ▶ $Z = A \bar{B}$

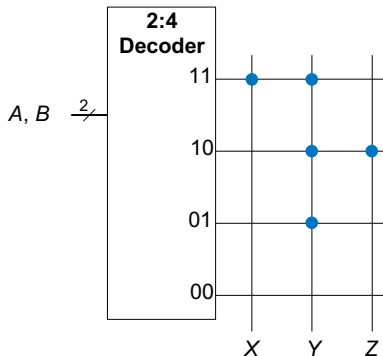


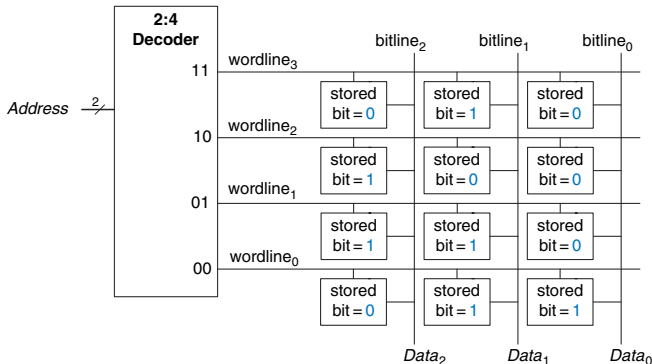
Logik via ROM: Beispiel



Implementieren Sie die folgenden Logikfunktionen mit einem $2^2 \times 3$ -Bit ROM:

- ▶ $X = A B$
- ▶ $Y = A + B$
- ▶ $Z = A \bar{B}$





$$\text{Address} = A_1, A_0$$

$$\text{Data}_2 = A_1 \oplus A_0$$

$$\text{Data}_1 = \overline{A_1} + A_0$$

$$\text{Data}_0 = \overline{A_1} \overline{A_0}$$

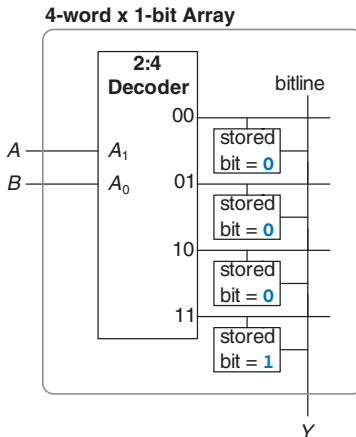
Logik via Speicherfeld: Lookup-Tabelle (LUT)



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Ausgabe bei jeder Eingangskombination (Adresse) nachschlagen

Truth Table		
A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1



SystemVerilog RAM



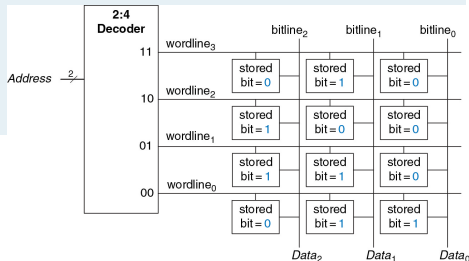
TECHNISCHE
UNIVERSITÄT
DARMSTADT

memory/ram.sv

```
1  // 2**N-word x M-bit RAM
2  module ram #(parameter N=6, M=32)
3      (input logic          clk,
4       input logic          we, // write enable
5       input logic [N-1:0]  adr,
6       input logic [M-1:0]  din,
7       output logic [M-1:0] dout);
8
9      logic [M-1:0] mem [2**N-1:0];
10
11     //write
12     always_ff @(posedge clk)
13         if (we)
14             mem [adr] <= din;
15
16     //read
17     assign dout = mem [adr];
18 endmodule
```


memory/rom.sv

```
1 // 4-word x 3-bit ROM
2 module rom(input logic [1:0]      adr,
3           output logic [2:0]      dout);
4
5     always_comb
6     case(adr)
7         2'b11: dout = 3'b010;
8         2'b10: dout = 3'b100;
9         2'b01: dout = 3'b110;
10        2'b00: dout = 3'b011;
11    endcase
12 endmodule
```





Pause & Umfrage bis hier

Agenda



TECHNISCHE
UNIVERSITÄT
DARMSTADT

1. Einleitung

2. Speicherfelder

3. Logikfelder

4. Zusammenfassung

Anwendungs- software		Programme
Betriebs- systeme		Gerätetreiber
Architektur		Befehle Register
Mikro- architektur		Datenpfade Steuerung
Logik		Addierer Speicher
Digital- schaltungen		UND Gatter Inverter
Analog- schaltungen		Verstärker Filter
Bauteile		Transistoren Dioden
Physik		Elektronen

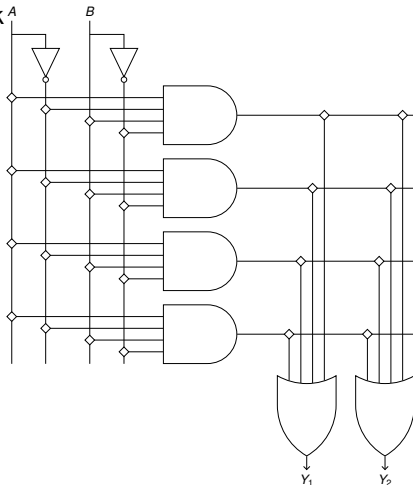
Programmierbares Logikfeld

Programmable Logic Array (PLA)



TECHNISCHE
UNIVERSITÄT
DARMSTADT

- ▶ realisiert einfache kombinatorische Logik via Sum-of-Products Form (DNF)
- ▶ zweistufige Logik mit programmierbaren Schaltern in Eingabefeld (links) und Ausgabefeld (rechts)
- ▶ Bsp.: $Y_1 = AB + \bar{A}\bar{B}$
 $Y_2 = \bar{A}B + A\bar{B}$
- ▶ Günstigere Varianten:
 - ▶ Programmable ROM:
nur Ausgabefeld programmierbar
 - ▶ Programmable Array Logic (PAL):
nur Eingabefeld programmierbar





- ▶ Anwendungsspezifische integrierte Schaltung (ASIC, application-specific integrated circuit)
 - ▶ führt *für eine Anwendung* optimierte (parallele) Datenpfade aus
 - ▶ Basisgatterschaltungen (bspw. als CMOS) durch optische/chemische Prozesse auf Silizium-Wafer realisiert
 - ⇒ zur Laufzeit nicht an neue Anwendung anpassbar

- ▶ Software-Prozessor
 - ▶ führt generische Instruktionen sequentiell aus
 - ▶ nur generische (Mikro-)Architektur in Hardware realisiert
 - ⇒ zur Laufzeit durch Austauschen der Instruktionssequenz an neue Anwendung anpassbar

- ⇒ Field Programmable Gate Array (FPGA) vereint
 - ▶ Flexibilität von Software-Prozessoren (“im Feld programmierbar”)
 - ▶ mit Performanz von ASICs (optimierte “Basisgatter-Schaltungen”)



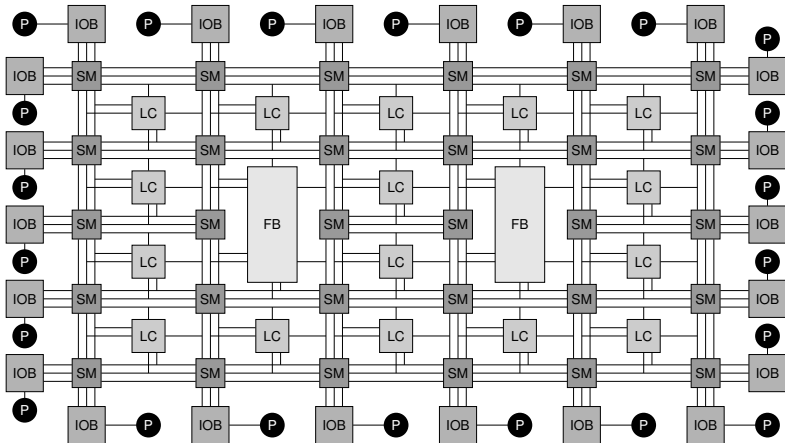
- ▶ FPGAs verwenden feingranulare (bitweise) Konfigurationsspeicher statt wortweisen Instruktionsspeichern
- ▶ Konfigurationsspeicher realisiert mit verschiedenen Speicher-Technologien:
 - ▶ volatil (bspw. SRAM): schnell beschreibbar, benötigt aber permanente Spannungsversorgung (statische Leistungsaufnahme), oder
 - ▶ nicht-volatil (bspw. Flash): aufwendiger Schreibzugriff, aber Zustand bleibt auch ohne Spannungsversorgung erhalten

Field Programmable Gate Array (FPGA)



TECHNISCHE
UNIVERSITÄT
DARMSTADT

P: Pin, IOB: I/O Block, SM: Switch Matrix, LC: Logic Cell, FB: Function Block

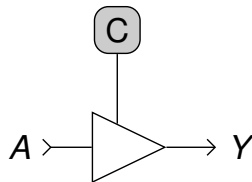
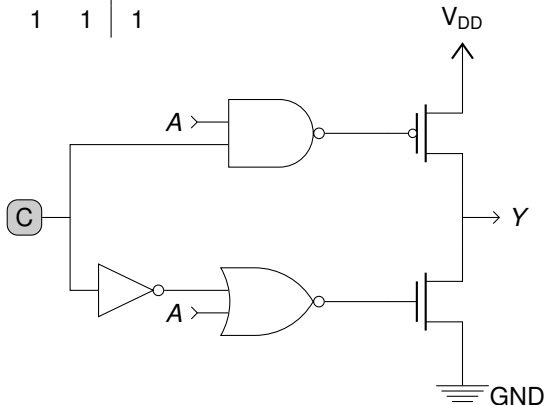


Programmierbare Schalter



TECHNISCHE
UNIVERSITÄT
DARMSTADT

C	A	Y
0	*	Z
1	0	0
1	1	1



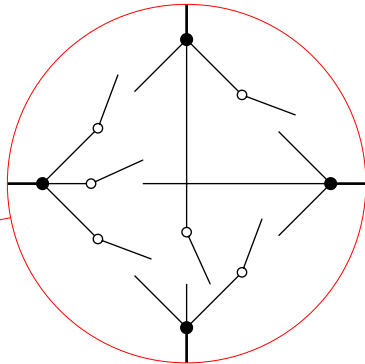
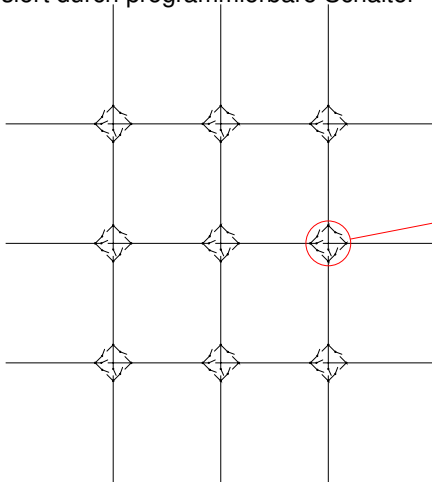
Programmierbare Leitungskreuzungen

Switch Matrix (SM)



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Realisiert durch programmierbare Schalter

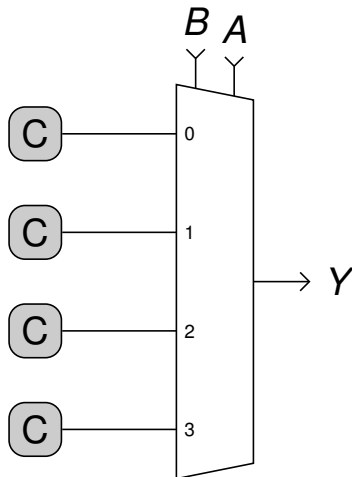


Programmierbare Tabelle Lookup Table (LUT)



TECHNISCHE
UNIVERSITÄT
DARMSTADT

- ▶ realisiert kombinatorische Logik
- ▶ 2 bis 6 Eingänge
- ▶ häufig auch aufteilbar in kleinere LUTs
- ▶ Multiplexer (MUX)



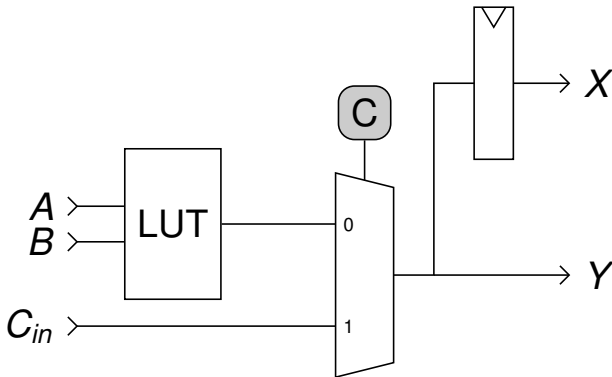
Programmierbare Logikzelle

Logic Cell (LC)



TECHNISCHE
UNIVERSITÄT
DARMSTADT

- ▶ kann als kombinatorische Logik (Y) und/oder Speicher (X) verwendet werden
- ▶ häufig auch spezielle Carry In (C_{in}) für schnelle Arithmetik

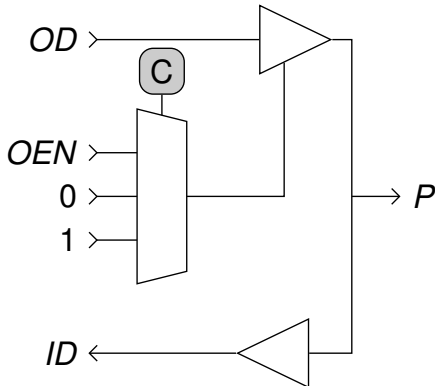


Programmierbare Ein-/Ausgänge Input-/Output Blocks (IOB)



TECHNISCHE
UNIVERSITÄT
DARMSTADT

- ▶ Ausgabetreiber kann permanent oder zur Laufzeit steuerbar (*OEN*) deaktiviert werden
- ▶ *P* wird mit physikalischen Pins verbunden
- ▶ häufig auch weitere Konfigurationsmöglichkeiten:
 - ▶ Spannungs-Level
 - ▶ maximale Stromstärke





- ▶ häufig verwendete Logikbausteine als begrenzte Ressourcen verfügbar
 - ▶ Block RAM (BRAM): kleine SRAM Speicher (wenige Kilobit)
 - ▶ Digitale Signalverarbeitung (DSP): Multiplizierer, MAC
 - ▶ Phase-Locked Loop (PLL): Taktmodifikation
 - ▶ Kommunikations-Treiber (USART, USB, Ethernet)
 - ▶ kleine Prozessoren
 - ▶ ...

Marktrelevante FPGA Hersteller



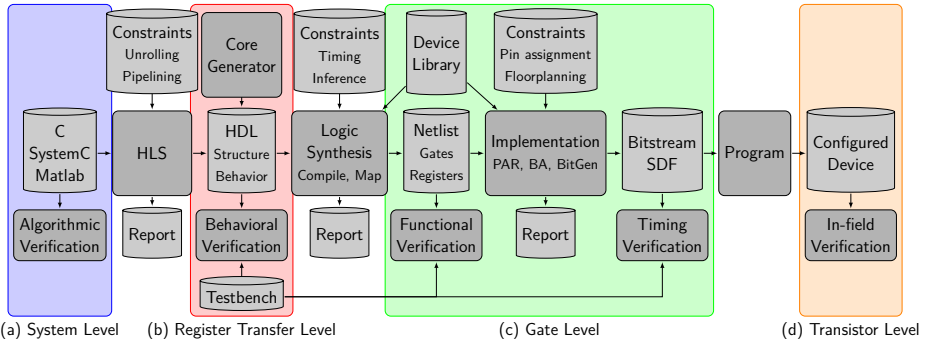
TECHNISCHE
UNIVERSITÄT
DARMSTADT

- ▶ Xilinx
 - ▶ Zynq, Virtex, Kintex
 - ▶ 7-series, UltraScale+
- ▶ Intel (hat Altera aufgekauft)
 - ▶ Cyclone, Aria, Stratix
- ▶ Microsemi
 - ▶ IGLOO, SmartFusion, PolarFire, ProAsic
- ▶ Lattice
 - ▶ iCE, Mach



Microsemi®





HLS: High-Level Synthesis, HDL: Hardware Description Language,
PAR: Place and Route, BA: Bat Algorithm, SDF: Standard Delay File

Agenda



TECHNISCHE
UNIVERSITÄT
DARMSTADT

1. Einleitung

2. Speicherfelder

3. Logikfelder

4. Zusammenfassung

Anwendungs- software		Programme
Betriebs- systeme		Gerätetreiber
Architektur		Befehle Register
Mikro- architektur		Datenpfade Steuerung
Logik		Addierer Speicher
Digital- schaltungen		UND Gatter Inverter
Analog- schaltungen		Verstärker Filter
Bauteile		Transistoren Dioden
Physik		Elektronen



- ▶ Speicherfelder
- ▶ Logikfelder

- ▶ Nächste Vorlesung behandelt
 - ▶ Abschluss Digitaltechnik und Ausblick
 - ▶ Besprechung der Evaluationsergebnisse
 - ▶ Infos zur Klausur
 - ▶ Möglichkeit zu Fragen im Plenum