Kapitel 3 – Kombinatorische Logik

1. Kombinatorische Schaltkreise
2. Normalformen, zweistufige Synthese
3. Berechnung eines Minimalpolynoms
4. Arithmetische Schaltungen
5. **Anwendung: ALU von ReTI**

Albert-Ludwigs-Universität Freiburg

UNI FREIBURG

Dr. Tobias Schubert, Dr. Ralf Wimmer

Professur für Rechnerarchitektur

WS 2016/17

Anwendung: ALU von ReTI

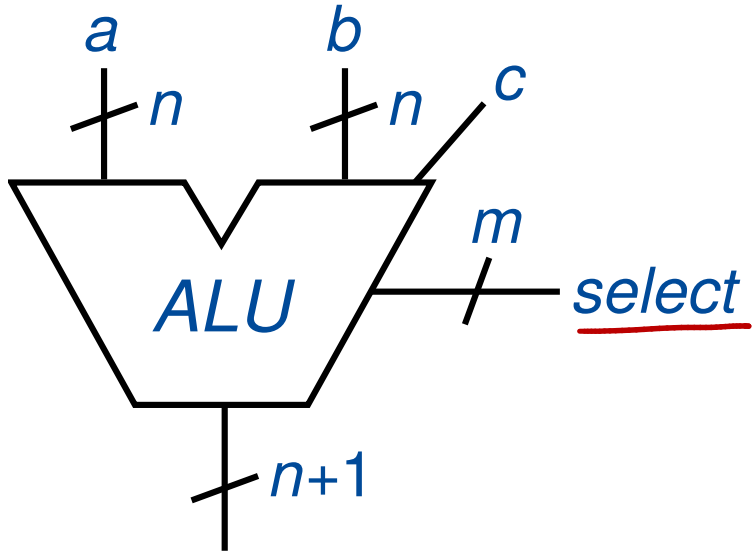
- Die ALU (Arithmetic Logic Unit, arithmetisch-logische Einheit) dient der Berechnung von arithmetischen und logischen Operationen.
- Sie wird von den Compute-Befehlen verwendet und übernimmt weitere Aufgaben, z.B. Berechnung von Speicheradressen oder Erhöhung des *PC*.
- Erinnerung: Der Befehlssatz von ReTI hat die folgenden Compute-Befehle (s. nächste Folie).

Compute-Befehle: Kodierung

Typ	MI	F	Befehl	Wirkung	
0 0	0	0 1 0	SUBI i	$[ACC] := [ACC] - [i]$	$\langle PC \rangle := \langle PC \rangle + 1$
		0 1 1	ADDI i	$[ACC] := [ACC] + [i]$	$\langle PC \rangle := \langle PC \rangle + 1$
		1 0 0	OPLUSI i	$ACC := ACC \oplus 0^8 i$	$\langle PC \rangle := \langle PC \rangle + 1$
		1 0 1	ORI i	$ACC := ACC \vee 0^8 i$	$\langle PC \rangle := \langle PC \rangle + 1$
		1 1 0	ANDI i	$ACC := ACC \wedge 0^8 i$	$\langle PC \rangle := \langle PC \rangle + 1$
0 0	1	0 1 0	SUB i	$[ACC] := [ACC] - [M(\langle i \rangle)]$	$\langle PC \rangle := \langle PC \rangle + 1$
		0 1 1	ADD i	$[ACC] := [ACC] + [M(\langle i \rangle)]$	$\langle PC \rangle := \langle PC \rangle + 1$
		1 0 0	OPLUS i	$ACC := ACC \oplus M(\langle i \rangle)$	$\langle PC \rangle := \langle PC \rangle + 1$
		1 0 1	OR i	$ACC := ACC \vee M(\langle i \rangle)$	$\langle PC \rangle := \langle PC \rangle + 1$
		1 1 0	AND i	$ACC := ACC \wedge M(\langle i \rangle)$	$\langle PC \rangle := \langle PC \rangle + 1$

Spezifikation der ALU für ReTI

- Eine n-Bit-ALU mit:
  - Zwei n-Bit-Operanden a, b, Eingangscarry c,
    - ReTI: $n = 32$.
  - Einem m-Bit select-Eingang, der ausgewählt, welche Funktion ausgeführt wird,
    - Hier: 8 Funktionen (s. nächste Folie), daher $m = 3$ Bits.
  - Einem $(n+1)$-Bit-Ausgang.
    - $n = 32$.



- Insgesamt 68 Ein- und 33 Ausgänge.

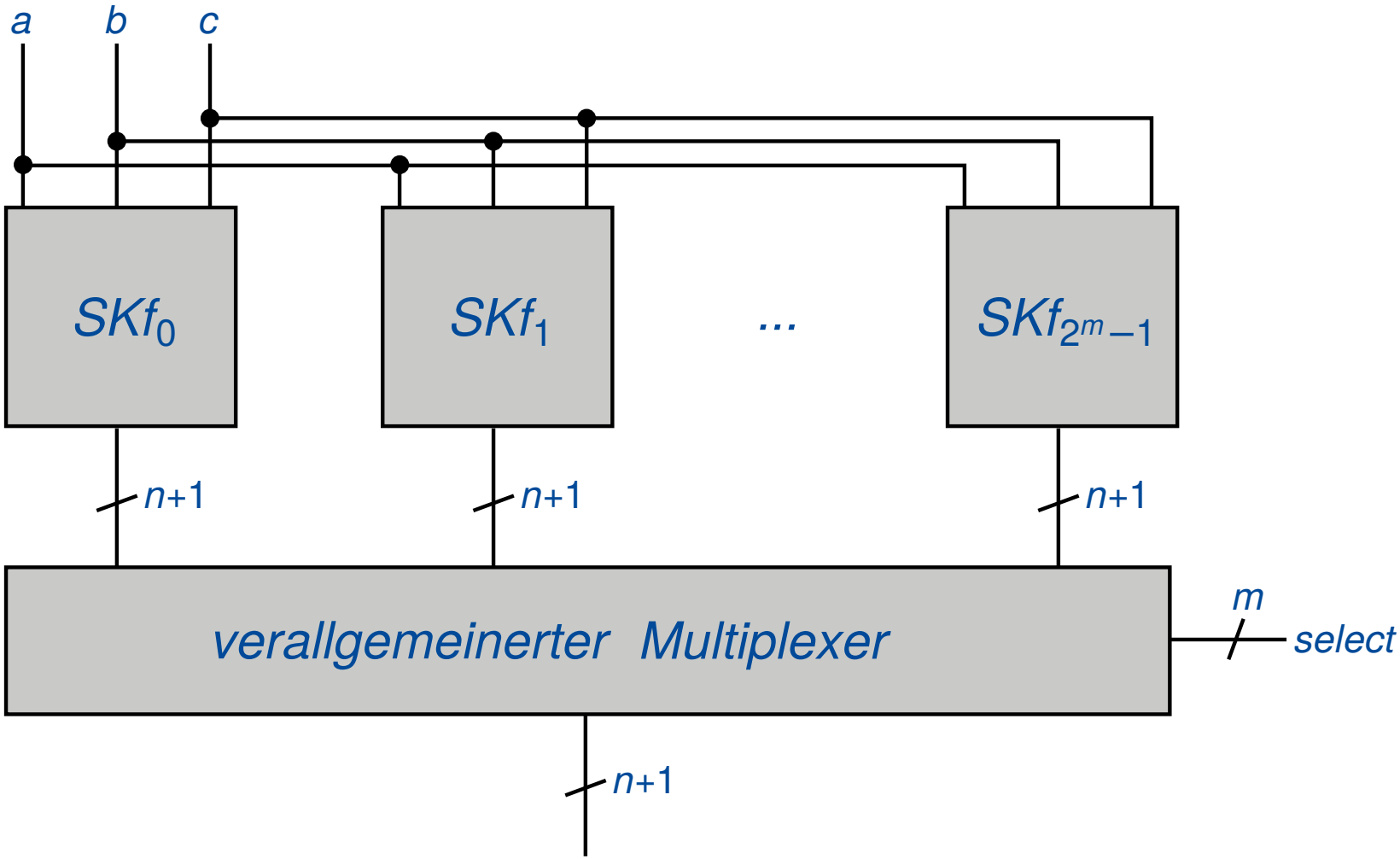
Select-Eingang bei ReTI-ALU

Funktionsnummer			ALU-Funktion
s_2	s_1	s_0	
0	0	0	$0 \ldots 0$
0	0	1	$[b] - [a]$
0	1	0	$[a] - [b]$
0	1	1	$[a] + [b] + c$
1	0	0	$a \oplus b = (a_{n-1} \oplus b_{n-1}, \ldots, a_0 \oplus b_0)$
1	0	1	$a \vee b = (a_{n-1} \vee b_{n-1}, \ldots, a_0 \vee b_0)$
1	1	0	$a \wedge b = (a_{n-1} \wedge b_{n-1}, \ldots, a_0 \wedge b_0)$
1	1	1	$1 \ldots 1$

Mögliche Realisierungen der ALU (1/2)

- **Option 1**: Realisiere Funktionen $f_0, \ldots, f_{2^m-1}$ getrennt durch SK_{f_i} für f_i, dann Auswahl durch einen verallgemeinerten Multiplexer.

Realisierung durch einen verallgemeinerten Multiplexer

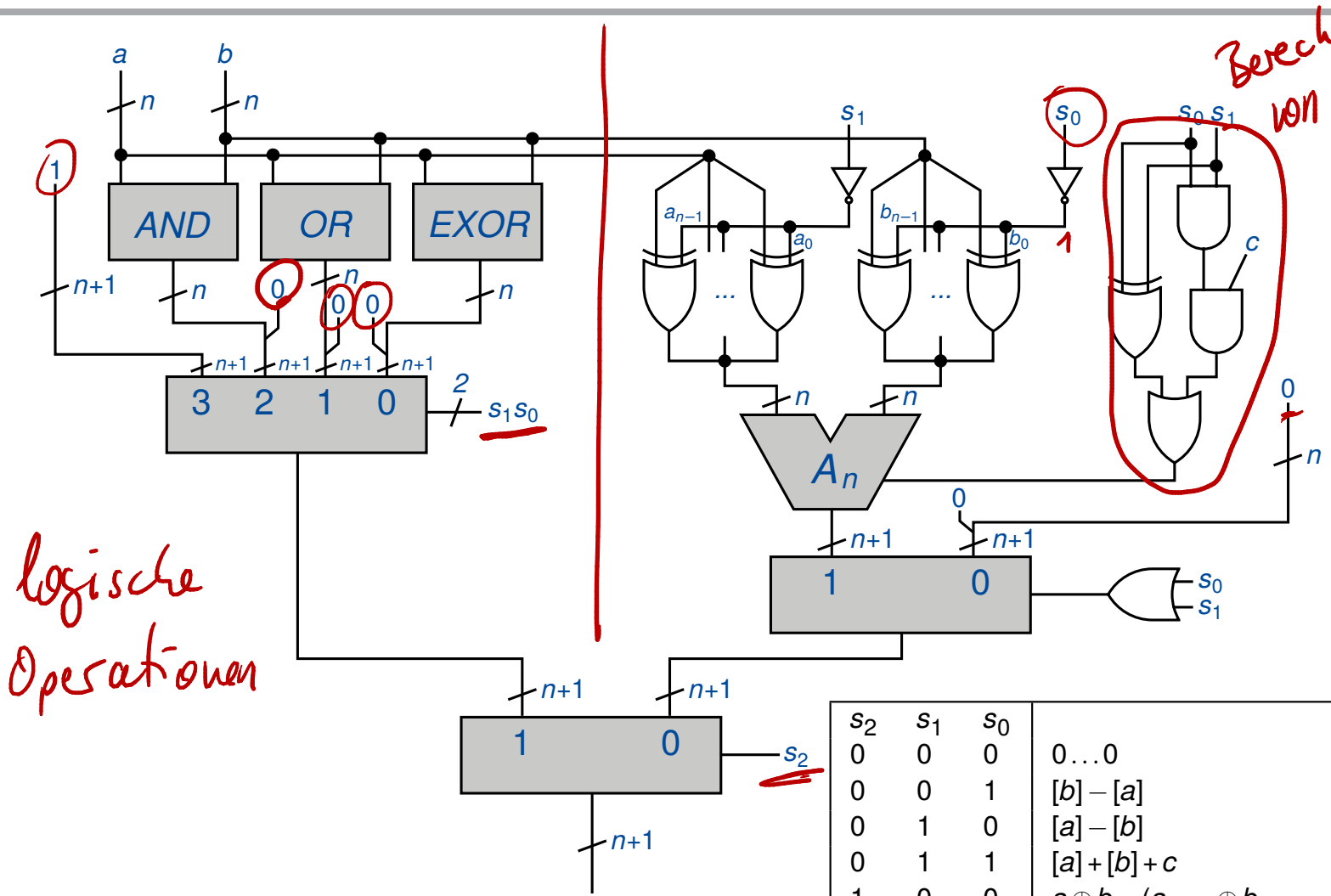


Mögliche Realisierungen der ALU (2/2)

- **Option 1**: Realisiere Funktionen $f_0, \ldots, f_{2^m-1}$ getrennt durch SK_{f_i} für f_i, dann Auswahl durch einen Verallgemeinerten Multiplexer.

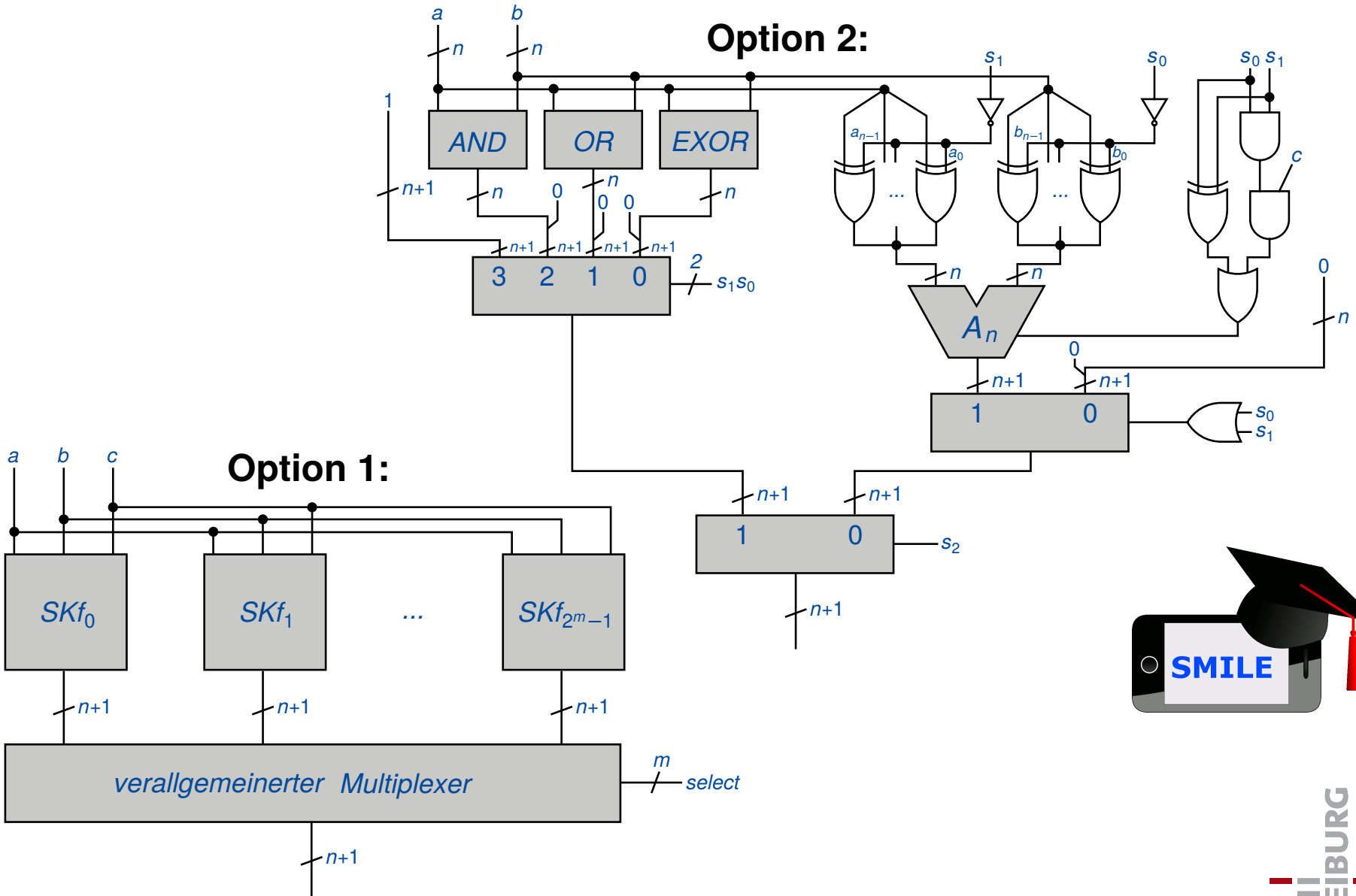
- **Option 2**: Gemeinsame Behandlung ähnlicher Funktionen.
  - Komplexer, aber effizienter.

Schaltrealisierung der ALU (1/2)



s_2	s_1	s_0	
0	0	0	0...0
0	0	1	$[b]-[a]$
0	1	0	$[a]-[b]$
0	1	1	$[a]+[b]+c$
1	0	0	$a \oplus b = (a_{n-1} \oplus b_{n-1}, \ldots, a_0 \oplus b_0)$
1	0	1	$a \vee b = (a_{n-1} \vee b_{n-1}, \ldots, a_0 \vee b_0)$
1	1	0	$a \wedge b = (a_{n-1} \wedge b_{n-1}, \ldots, a_0 \wedge b_0)$
1	1	1	1...1

Schaltrealisierung der ALU (2/2)



SMILE - Vergleich Option 1 mit Option 2

Wie verhalten sich Kosten und Tiefe der beiden Realisierungsoptionen zueinander?

a. $(\mathit{Tiefe}(\mathit{Option}1) \leq \mathit{Tiefe}(\mathit{Option}2))$ und $(\mathit{Kosten}(\mathit{Option}1) \leq \mathit{Kosten}(\mathit{Option}2))$

b. $(\mathit{Tiefe}(\mathit{Option}1) \geq \mathit{Tiefe}(\mathit{Option}2))$ und $(\mathit{Kosten}(\mathit{Option}1) \leq \mathit{Kosten}(\mathit{Option}2))$

c. $(\mathit{Tiefe}(\mathit{Option}1) \leq \mathit{Tiefe}(\mathit{Option}2))$ und $(\mathit{Kosten}(\mathit{Option}1) \geq \mathit{Kosten}(\mathit{Option}2))$

d. keine der obigen

Zusammenfassung Kombinatorische Logik

- Kombinatorische Schaltkreise setzen boolesche Funktionen um.
- PLAs sind zweistufig, mehrstufige Schaltungen bestehen aus Gattern und diese aus Transistoren.
- Minimierung von PLAs mit Verfahren von Quine-McCluskey und Lösen des Überdeckungsproblems.
- Statt Minimierung allgemeiner mehrstufiger Schaltkreise wurde eine Klasse (Addierer für Binär- und Zweierkomplementzahlen) betrachtet und ihre Integration in der ALU von ReTI diskutiert.