

Rechnertechnik

Prof. Dr. rer. nat. Peter Gerwinski

29. März 2022

Rechnertechnik

1 Einführung

1.1 Was ist Rechnertechnik?

1.2 Was ist ein Computer?

2 Vom Schaltkreis zum Computer

2.1 Logik-Schaltkreise

2.2 Binärdarstellung von Zahlen

2.3 Vom Logik-Schaltkreis zum Addierer

2.4 Negative Zahlen

2.5 Vom Addierer zum Computer

...

3 Architekturmerkmale von Prozessoren

4 Der CPU-Stack

...

2.4 Negative Zahlen

Speicher ist begrenzt!

→ feste Anzahl von Bits

8-Bit-Zahlen ohne Vorzeichen: `uint8_t`

→ Zahlenwerte von `0x00` bis `0xff` = 0 bis 255

2.4 Negative Zahlen

Speicher ist begrenzt!

→ feste Anzahl von Bits

8-Bit-Zahlen ohne Vorzeichen: `uint8_t`

→ Zahlenwerte von `0x00` bis `0xff` = 0 bis 255

→ $255 + 1 = 0$

2.4 Negative Zahlen

Speicher ist begrenzt!

→ feste Anzahl von Bits

8-Bit-Zahlen ohne Vorzeichen: `uint8_t`

→ Zahlenwerte von `0x00` bis `0xff` = 0 bis 255

→ $255 + 1 = 0$

8-Bit-Zahlen mit Vorzeichen: `int8_t`

`0xff` = 255 ist die „natürliche“ Schreibweise für -1 .

2.4 Negative Zahlen

Speicher ist begrenzt!

→ feste Anzahl von Bits

8-Bit-Zahlen ohne Vorzeichen: `uint8_t`

→ Zahlenwerte von `0x00` bis `0xff` = 0 bis 255

→ $255 + 1 = 0$

8-Bit-Zahlen mit Vorzeichen: `int8_t`

`0xff` = 255 ist die „natürliche“ Schreibweise für -1 .

→ Zweierkomplement

2.4 Negative Zahlen

Speicher ist begrenzt!

→ feste Anzahl von Bits

8-Bit-Zahlen ohne Vorzeichen: `uint8_t`

→ Zahlenwerte von `0x00` bis `0xff` = 0 bis 255

→ $255 + 1 = 0$

8-Bit-Zahlen mit Vorzeichen: `int8_t`

`0xff` = 255 ist die „natürliche“ Schreibweise für -1 .

→ Zweierkomplement

Oberstes Bit = 1: negativ

Oberstes Bit = 0: positiv

→ $127 + 1 = -128$

2.4 Negative Zahlen

Speicher ist begrenzt!

→ feste Anzahl von Bits

16-Bit-Zahlen ohne Vorzeichen: `uint16_t`

→ Zahlenwerte von `0x0000` bis `0xffff` = 0 bis 65535

→ $65535 + 1 = 0$

`uint8_t`

0 bis 255

$255 + 1 = 0$

16-Bit-Zahlen mit Vorzeichen: `int16_t`

`0xffff` = 65535 ist die „natürliche“ Schreibweise für -1 .

→ Zweierkomplement

`int8_t`

`0xff` = 255 = -1

Oberstes Bit = 1: negativ

Oberstes Bit = 0: positiv

→ $32767 + 1 = -32768$

Literatur: <http://xkcd.com/571/>

2.4 Negative Zahlen

Frage: Für welche Zahl steht der Speicherinhalt

a3	90
----	----

 (hexadezimal)?

2.4 Negative Zahlen

Frage: Für welche Zahl steht der Speicherinhalt

a3	90
----	----

 (hexadezimal)?

Antwort: Das kommt darauf an. ;—)

2.4 Negative Zahlen

Frage: Für welche Zahl steht der Speicherinhalt

a3	90
----	----

 (hexadezimal)?

Antwort: Das kommt darauf an. ;—)

Little-Endian:

als <code>int8_t</code> :	−93	(nur erstes Byte)
als <code>uint8_t</code> :	163	(nur erstes Byte)
als <code>int16_t</code> :	−28509	
als <code>uint16_t</code> :	37027	
<code>int32_t</code> oder größer:	37027	(zusätzliche Bytes mit Nullen aufgefüllt)

2.4 Negative Zahlen

Frage: Für welche Zahl steht der Speicherinhalt

a3	90
----	----

 (hexadezimal)?

Antwort: Das kommt darauf an. ;—)

Little-Endian:

als <code>int8_t</code> :	−93	(nur erstes Byte)
als <code>uint8_t</code> :	163	(nur erstes Byte)
als <code>int16_t</code> :	−28509	
als <code>uint16_t</code> :	37027	
<code>int32_t</code> oder größer:	37027	(zusätzliche Bytes mit Nullen aufgefüllt)

Big-Endian:

als <code>int8_t</code> :	−93	(nur erstes Byte)
als <code>uint8_t</code> :	163	(nur erstes Byte)
als <code>int16_t</code> :	−23664	
als <code>uint16_t</code> :	41872	
als <code>int32_t</code> :	−1550843904	(zusätzliche Bytes mit Nullen aufgefüllt)
als <code>uint32_t</code> :	2744123392	
als <code>int64_t</code> :	−6660823848880963584	
als <code>uint64_t</code> :	11785920224828588032	

2.4 Negative Zahlen

Aufbau einer Schaltung zum Bilden des Zweierkomplements:

Sie sind dran.

→ Siehe: [../20220322/2er-komplement-*.png](#)

2.5 Vom Addierer zum Computer

Wir können jetzt addieren und subtrahieren.

Wie bauen wir daraus einen Turing-vollständigen Computer?

2.5 Vom Addierer zum Computer

Wir können jetzt addieren und subtrahieren.

Wie bauen wir daraus einen Turing-vollständigen Computer?

- Arithmetisch-logische Einheit (ALU)
- Speicher
- Takt

2.5 Vom Addierer zum Computer

1-Bit-Multiplizierer

A	B	Q
0	0	
0	1	
1	0	
1	1	

2.5 Vom Addierer zum Computer

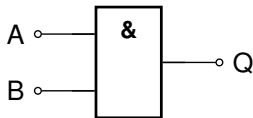
1-Bit-Multiplizierer

A	B	Q
0	0	0
0	1	0
1	0	0
1	1	1

2.5 Vom Addierer zum Computer

1-Bit-Multiplizierer = Und-Verknüpfung

A	B	Q
0	0	0
0	1	0
1	0	0
1	1	1



2.5 Vom Addierer zum Computer

1-Bit-Multiplizierer = Und-Verknüpfung

n-Bit-Multiplizierer

2.5 Vom Addierer zum Computer

1-Bit-Multiplizierer = Und-Verknüpfung

n-Bit-Multiplizierer: „schriftlich“ multiplizieren

2.5 Vom Addierer zum Computer

1-Bit-Multiplizierer = Und-Verknüpfung

n-Bit-Multiplizierer: „schriftlich“ multiplizieren

Beispiel: $13 \cdot 5$

$$\begin{array}{r} 1101 \cdot 101 \\ \hline 1101 \\ 0 \\ 1101 \\ \hline 1111 \\ 100001 \end{array}$$

2.5 Vom Addierer zum Computer

1-Bit-Multiplizierer = Und-Verknüpfung

n-Bit-Multiplizierer: „schriftlich“ multiplizieren

Beispiel: $13 \cdot 5$

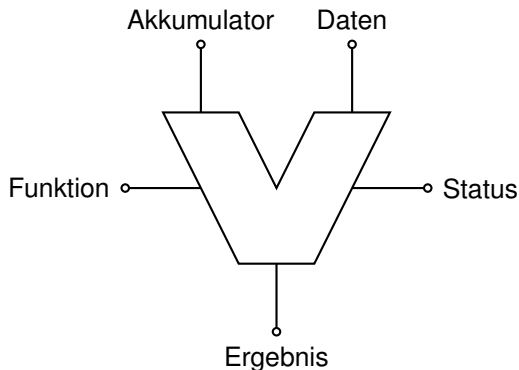
$$\begin{array}{r} 1101 \cdot 101 \\ \hline 1101 \\ 0 \\ 1101 \\ \hline 1111 \\ 100001 \end{array}$$

Multiplizier-Schaltkreis
für zwei 2-Bit-Zahlen:

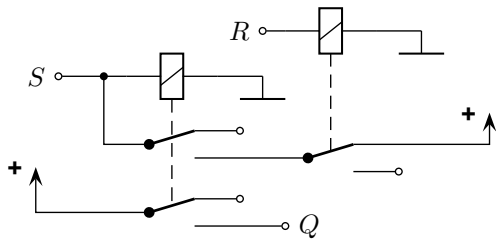
Sie sind dran.

2.5 Vom Addierer zum Computer

Schaltkreis, der wahlweise eine von mehreren Verknüpfungen durchführt:
arithmetisch-logische Einheit – arithmetic logic unit (ALU)

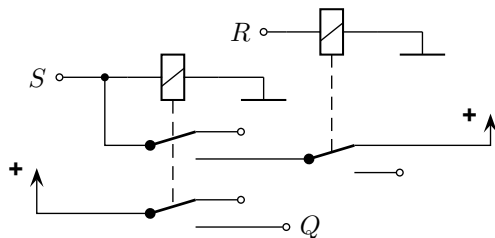


Siehe z. B.: <https://en.wikipedia.org/wiki/File:74181aluschematic.png>



2.5 Vom Addierer zum Computer

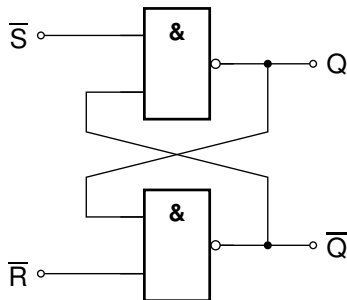
Information speichern



Selbsthalterschaltung
1-Bit-Speicherzelle

2.5 Vom Addierer zum Computer

Information speichern



Bistabile Kippstufe – Bistabiler Multivibrator – Flip-Flop
1-Bit-Speicherzelle

2.5 Vom Addierer zum Computer

Information speichern

Kondensator

1-Bit-Speicherzelle



2.5 Vom Addierer zum Computer

Information speichern

Kondensator

dynamische 1-Bit-Speicherzelle

→ benötigt *Refresh*-Schaltung



2.5 Vom Addierer zum Computer

Information speichern

Kondensator

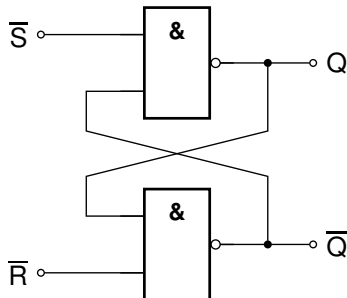
dynamische 1-Bit-Speicherzelle

→ benötigt *Refresh*-Schaltung



Flip-Flop

statische 1-Bit-Speicherzelle



2.5 Vom Addierer zum Computer

Computer

- Rechenwerk (ALU)
- Speicher

2.5 Vom Addierer zum Computer

Computer

- Rechenwerk (ALU)
- Speicher: Register, adressierbarer Hauptspeicher

2.5 Vom Addierer zum Computer

Computer

- Rechenwerk (ALU)
- Speicher: Register, adressierbarer Hauptspeicher
- Takt

2.5 Vom Addierer zum Computer

Computer

- Rechenwerk (ALU)
- Speicher: Register, adressierbarer Hauptspeicher
- Takt: Befehle abarbeiten

2.5 Vom Addierer zum Computer

Computer

- Rechenwerk (ALU)
- Speicher: Register, adressierbarer Hauptspeicher
- Takt: Befehle abarbeiten
- Peripherie: Kommunikation mit der Außenwelt

2.5 Vom Addierer zum Computer

Computer

- Rechenwerk (ALU)
- Speicher: Register, adressierbarer Hauptspeicher
- Takt: Befehle abarbeiten
- Peripherie: Kommunikation mit der Außenwelt

—→ in Maschinensprache programmierbar

Rechnertechnik

1 Einführung

2 Vom Schaltkreis zum Computer

2.1 Logik-Schaltkreise

2.2 Binärdarstellung von Zahlen

2.3 Vom Logik-Schaltkreis zum Addierer

2.4 Negative Zahlen

2.5 Vom Addierer zum Computer

...

3 Architekturmerkmale von Prozessoren

4 Der CPU-Stack

...