Tut 14 (Tutor) 2nd Edition

Monday, 4. February 2019 20:59

0. ORGANISATORISCHES

⚠ 3. Block-Test

- Zeit bis So. 10.02.19 23^{59}

⚠ Klausur

27.02.19 (Mi) 08^{00}, Anmeldung 10.02. – 20.02 über MOSES && QISPOS ‼

- Seit dem 04.02. (Mo) ist die Probeklausur verfügbar
- → Bearbeiten bis nächste Woche (empfohlen), Q&A nächste Woche. Mehr auf Freitagsrunde

1. DIE ALU

- Letzte Woche: Schaltungen konstruiert, um Aufgaben & Funktionen zu erfüllen – z.B. Primzahlerkennung
- Computerprozessor setzt sich aus Logikgatter zusammen
- Rechenwerk, Steuerwerk, Speicher…
- Rechenwerk (ALU) führt Berechnungen aus, u.a. Addition

1.1. Halbaddierer

- Wie sieht so ein Addierwerk aus ?
- Addieren eines Bits

ALU: Halbaddierer

① 1 Bit Addition

0	0	1	1
+0	+1	+0	+1
0	1	1	10
			↓↓ C S

② WT

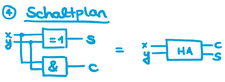
x	y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

③ Log. Ausdrücke

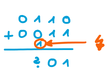
$S = x \oplus y$

$C = x \cdot y$

④ Schaltplan



- Ein Halbaddierer setzt sich einfach aus einem XOR & AND zusammen ‼
- Mehrere Bits addieren



- Funktioniert nicht, weil wir nur zwei Eingangsbits haben
- Doof: wie kann das Carry in berücksichtigt werden ?

1.2 Der Volladdierer

- Zielsetzung durch Wertetabelle

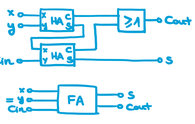
ALU: Volladdierer

- Ziel: Carry in berücksichtigen ‼

Wertetabelle

Cin	x	y	S	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

… lässt sich aus 2 HA und 1 OR bauen ‼



- Zusätzlicher Eintrag für den alten Übertrag (Carry-In)
- Wird aus zwei HA und einem OR-Gatter zusammengesetzt
- Logische Ausdrücke:
  - → $S = (x \oplus y) \oplus Cin$
  - → $Cout = x \cdot y + (Cin \cdot (x \oplus y))$
- Nun können wir zwei 4 Bit zahlen bspw. durch Kaskadierung von 4 Volladdierer addieren
- Bild: Was ist Cin0 ?
- → =0V, auf GND gelegt, da ich kein Carry In ‼
- Ende des reg. Stoffes

1.3 ALU-Entwurf