# Einführung in die Informatik | Übung 2

## Aufgabe: Speichern und Rechnen

a) Eine Binary Cell kann genau ein Bit speichern. Ein Byte sind 8 Bit und ein Gigabyte sind 10^9 Bytes, also 8*10^9 Bit. Dementsprechend bräuchte es für 8 Gigabyte Arbeitsspeicher 8*8*10^9 Bit und damit **64000000000 Binary Cells**. Da in jeder Binary Cell ein RS- Flip- Flop verbaut ist, braucht es auch **64000000000 RS- Flip- Flops.**

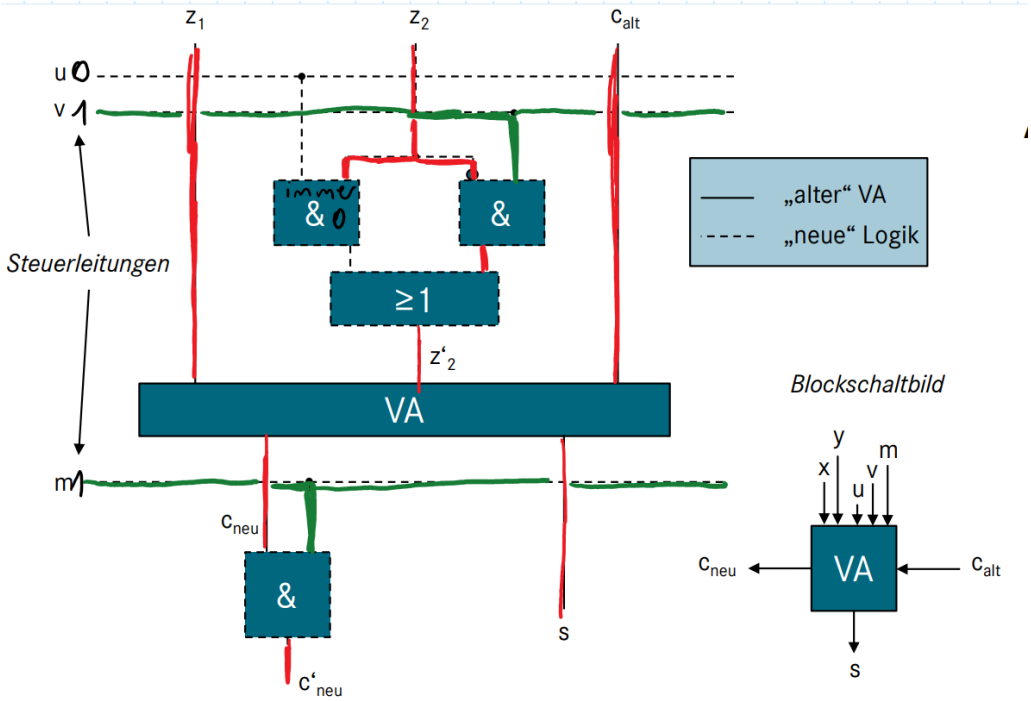
b) Der **Demultiplexer** ermöglicht es, jede Zeile des Tabellenspeichers einzeln anzusprechen, ohne, dass für jede eine eigene Adressleitung von Prozessor zum Speicher führt. Durch zwei (oder bei größeren Registern mehr) select- Signale entstehen verschiedene Kombinationen aus „0“ und „1“, die dabei helfen, die verschiedenen Register anzusprechen.
Wenn es darum geht, alle Zeilen des Registers auf einmal anzusprechen, ist der Multiplexer das nötige Bauteil. Hier reicht eine „1“ bei einer der verschiedenen Kombinationen, damit alle Register angesprochen werden.

c) Um den Arbeitsspeicher einer 8GB RAM- Spielekonsole ansprechen zu können, muss der Adressbus zu diesem Speicher mindestens **30 Leitungen** haben.
Die Zahl ergibt sich daraus, dass diese Spielekonsole 8*8*10^9 Bit, also 6,4*10^10 Bit speichern kann. Der RAM hat eine Wortbreite von 64 Bit, dies bedeutet, dass der Speicher (6,4*10^10)/64 = 1000000000 = 10^9 Zeilen benötigt. Da im Speicher Demultiplexer verwendet werden, braucht es nicht zu jeder Zeile eine eigene Leitung. Die Anzahl der Select- Leitungen für eine bestimmte Zahlenanzahl kann mit 2^x beschrieben werden, da ja alle Kombinationen nur aus „0“ und „1“ bestehen können. X ist hier die Anzahl der Leitungen und wird demensprechend gesucht.
2^x = 10^9 -> x = 29,897. Da man keine „halben“ Leitungen haben kann, muss auf 30 aufgerundet werden.

Zum Rechenwerk:

a) Um zwei beliebige 64Bit- Zahlen miteinander addieren zu können, braucht es **127 Halbaddierer**, da für die Addition jedes Bits- außer des Ersten- ein Volladdierer benötigt wird, um mögliche Überträge berücksichtigen zu können. Nur für das erste Bit reicht ein Halbaddierer, da hier, bei der ersten Addition, noch nicht die Möglichkeit eines Übertrages besteht. Da ein Volladdierer aus zwei Halbaddierern besteht, werden 63 * 2 + 1 = 127 Halbaddierer benötigt.

b) Eine Binärzahl kann in einer ALU von einer anderen subtrahiert werden, indem in der ALU erst die subtrahierende Zahl in ihr Komplement umgewandelt wird und das Komplement anschließend mit der zu subtrahierenden Zahl im VA unter Rücksichtnahme der Überträge addiert wird. Die **Steuerleitungen m und v müssen hierfür mit einer „1“ belegt sein, während u mit einer „0“ belegt sein muss**:




- Wenn bei z2 eine bestimmte Zahl eingeht, „fragt“ die ALU an der ersten Schaltung zuerst, ob addiert oder subtrahiert werden soll. Dies ist durch eine verschiedene Belegung der Steuerleitungen u und v möglich. Steht u auf „1“ und v auf 0 wird addiert. Steht v auf „1“ und u auf „0“ steht fest, dass subtrahiert werden soll:
  - U ist auf „0“ also wird die Schaltung, durch die addiert wird, nie geöffnet- es wird verhindert, dass Z2 zweimal in den Volladdierer eingeht
  - V ist auf „1“, also gehen alle Eingänge von z2 den Weg durch die AND- Schaltung mit der v verbunden ist. Dabei werden sie durch die Negierung vor der AND- Schaltung in ihr Komplement (bisher ohne die Addition von 1) umgewandelt.
- Z1 und das Komplement von z2 werden nun zusammen mit Calt- das bei der Subtraktion den Startwert „1“ besitzt, damit das vollständige Komplement von z2 gebildet werden kann- an den Volladdierer weitergeleitet.
- Volladdierer addiert z1 und das Komplement von z2 und rechnet damit z1 – z2.
- Dadurch, dass m auf „1“ gesetzt ist, wird ein Übertrag von Cneu auf C’neu möglich und das Ergebnis der Subtraktion der beiden Zahlen kann gesetzt werden.

## Aufgabe: Optimierung einer Schaltung

a)

| a | b | c | d | y | Lampe |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 1 | An |
| 0 | 0 | 0 | 1 | 0 | Aus |
| 0 | 0 | 1 | 0 | 1 | An |
| 0 | 0 | 1 | 1 | 0 | Aus |
| 0 | 1 | 0 | 0 | 1 | An |
| 0 | 1 | 1 | 0 | 1 | An |
| 0 | 1 | 0 | 1 | 0 | Aus |
| 0 | 1 | 1 | 1 | 1 | An |
| 1 | 0 | 0 | 0 | 1 | An |
| 1 | 1 | 0 | 0 | 1 | An |
| 1 | 0 | 1 | 0 | 1 | An |
| 1 | 0 | 0 | 1 | 0 | Aus |
| 1 | 0 | 1 | 1 | 1 | An |
| 1 | 1 | 1 | 0 | 1 | An |
| 1 | 1 | 0 | 1 | 0 | Aus |
| 1 | 1 | 1 | 1 | 1 | An |

b)

$$y = (a \wedge c) \vee (\neg a \wedge b \wedge c) \vee (\neg a \wedge c \wedge \neg d) \vee \neg(c \vee d)$$

c)