2 Rechenschaltungen und andere Schaltnetze

Nachdem wir jetzt Kenntnisse über die Verbindungen von Computern untereinander gewonnen haben, wollen wir uns die „Rechner“ selbst vornehmen – und die müssen wenigstens eines können: rechnen. Die dabei gewonnenen Kenntnisse dienen dann später dazu, auch andere Schaltungen systematisch mithilfe endlicher Automaten zu entwickeln.

2.1 Einfache Addierschaltungen

Natürlich wird ein digitaler Rechner mit Dualzahlen rechnen, die z. B. addieren müssen. Im einfachsten Fall wird er zwei einstellige Dualzahlen *a* und *b* addieren, also zwei Zahlen, die nur die Werte Eins oder Null annehmen können. Eine solche Schaltung nennt man einen *Halbaddierer*.

Welche Ergebnisse liefert ein Halbaddierer?

Eingänge		Ausgänge
a	b	???
0	0	???
0	1	
1	0	
1	1	

Wir nutzen eine hilfreiche Eigenschaft von endlichen digitalen Größen aus: es gibt nur endlich viele mögliche Kombinationen. Wenn man dem Halbaddierer die beiden Zahlen auf zwei Leitungen, den *Eingängen*, zuführt, dann gibt es für jeden Eingang nur zwei mögliche Belegungen – Null oder Eins. Insgesamt erhält man damit vier mögliche Kombinationen an den Eingängen.

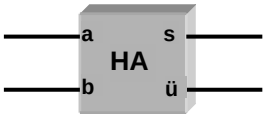
Bisher haben wir noch gar nicht definiert, was der Halbaddierer mit den Eingängen tun soll. Er soll addieren, also in den vier möglichen Fällen die Summen der Eingänge ausgeben.

Eingänge		Ausgänge
a	b	Summe
0	0	0
0	1	1
1	0	1
1	1	2

Mit den ersten drei Ergebnissen haben wir keine Probleme, denn Null und Eins sind als einstellige Dualzahlen darstellbar. Die Zwei aber erfordert zwei Dualstellen, sie wird als *10* geschrieben. Damit muss der Halbaddierer über zwei Ausgänge verfügen. Nennen wir den einen die *Summe s*, dann heißt der andere traditionell *Übertrag ü*.

Eingänge		Ausgänge	
a	b	ü	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Das Schaltsymbol für einen solchen Halbaddierer ist ein „Kasten“ mit der Aufschrift HA, an den die Ein- und Ausgange als beschriftete Leitungen angeschlossen sind:

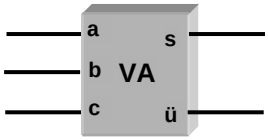


Wir wissen zwar noch nicht, wie ein solcher „Kasten“ intern aufgebaut ist, können uns aber leicht vorstellen, wie er eingesetzt werden könnte. Die Addition von 1-Bit-Dualzahlen hat nicht viel Sinn. Aus solchen einstelligen Operationen kann man aber – ähnlich wie beim „schriftlichen“ Addieren – brauchbarere Addierer zusammensetzen.

Wie addiert man schriftlich? Man addiert stellenweise und verrechnet die bei den Einzeladditionen auftretenden Überträge. Für Dualzahlen sehen einzelne Beispiele so aus:

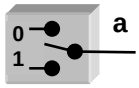
3 + 6:	0011	5 + 5:	0101	7 + 3:	0111
	+ 0110		+ 0101		+ 0011
Übertrag:	110		101		111
= 9	1001	= 10	1010	= 10	1010

Man sieht, dass für die erste Addition „ganz rechts“ ein Halbaddierer ausreicht, weil dort kein Übertrag auftritt. Ab der zweiten Stelle benötigen wir jedoch einen Addierer, der drei Dualziffern addieren kann – einen *Volladdierer VA*. Auch von diesem wollen wir erst einmal annehmen, dass es ihn gibt – ohne sein Innenleben zu kennen. Seine Wirkungsweise ist allerdings festgelegt. Er hat drei Eingänge *a*, *b* und *c* sowie zwei Ausgänge *s* und *ü*, weil sein Ergebnis höchstens eine Drei sein kann, und die lässt sich mit zwei Bits darstellen.

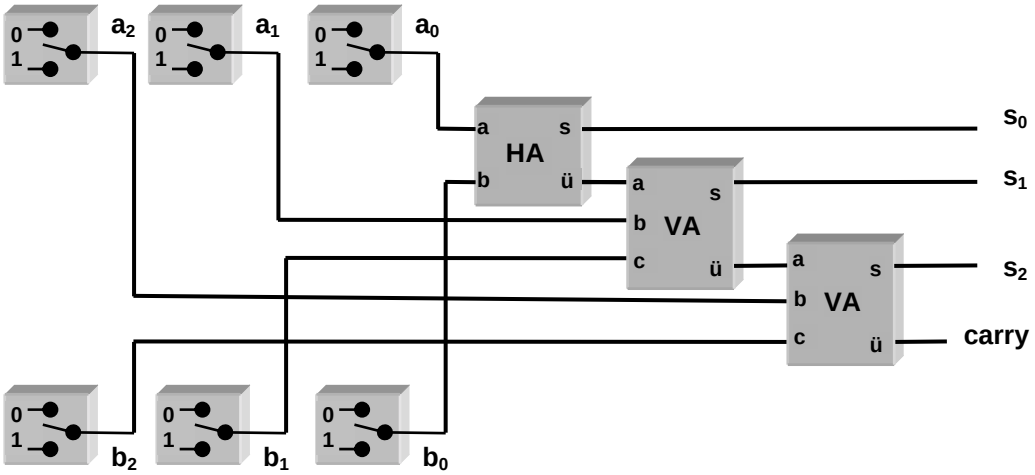


Eingänge			Ausgänge	
a	b	c	ü	s
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Mithilfe von Voll- und Halbaddierern lässt sich leicht eine Addierschaltung angeben. Wir müssen uns nur noch überlegen, wie wir die Eingänge mit unterschiedlichen Dualziffern belegen können. Vereinbaren wir z. B., dass kleine Spannungen unter 0,2 Volt die Ziffer Null (symbolisch *0*) und Spannungen zwischen 1 Volt und 5 Volt die Ziffer Eins (symbolisch *1*) darstellen – der Spannungs-Zwischenbereich sollte wegen der Eindeutigkeit dringend vermieden werden –, dann können wir als Eingabegeräte einfache Schalter verwenden, die mit entsprechenden Spannungsquellen verbunden sind. Eine solche Größe, die die beiden Werte der Dualziffern annehmen kann, bezeichnen wir unabhängig von der verwendeten Technik als *Schaltvariable*.



Eine Schaltung, die mit diesen Bauteilen zwei 3-Bit-Dualzahlen addieren kann, bezeichnet man als einen *3-Bit-Paralleladdierer*. Üblicherweise kennzeichnet man die einzelnen Bits der Dualzahlen durch ihren Index, schreibt also z. B. die 3-Bit-Dualzahl a als $(a_2a_1a_0)$. Wir addieren zuerst die niederwertigsten Bits a_0 und b_0 mithilfe eines Halbaddierers und erhalten das erste Bit s_0 der Gesamtsumme aus dessen Summenausgang. Den Übertrag addieren wir mit dem ersten Volladdierer zu den beiden mittleren Bits der Dualzahlen und erhalten das nächste Summenbit usw. Da solche Rechnungen zuletzt wieder einen Übertrag liefern, der das Ergebnis um ein Bit „verlängert“, also größer als das Zahlenformat der Summanden a und b macht (hier: aus zwei 3-Bit-Dualzahlen entsteht eine 4-Bit-Summe), kann man den letzten Übertrag als *Carry-Bit* (eben: als Übertrag) speichern und geeignet weiter verarbeiten.



Offensichtlich kann man solche Paralleladdierwerke beliebig vergrößern, z. B. auf heute übliche 32-, 64-Bit- oder noch längere Dualzahlen. Als Grundbausteine findet man sie deshalb auch noch in den modernsten Prozessoren.