2.3 Addition im Binärsystem und Halbaddierer

Leitidee

Jeder Taschenrechner, jedes Handy und jeder Computer kann in Bruchteilen einer Sekunde Millionen von Zahlen addieren. Dabei kennt die Elektronik nur 0 und 1, und rechnen können ihre Gatter eigentlich gar nicht: Sie verknüpfen nur Bits nach festen Regeln. Wie kann daraus ein Rechenwerk werden? Der Trick ist, Zahlen im **Binärsystem** zu schreiben, also nur mit den Ziffern 0 und 1. Dann funktioniert die Addition genau wie die schriftliche Addition aus der Grundschule. In diesem Abschnitt rechnest du zuerst selbst im Binärsystem. Danach baust du aus zwei Gattern eine Schaltung, die zwei Bits addiert: den **Halbaddierer**.

Lernziele

Nach diesem Abschnitt kannst du

- natürliche Zahlen vom Dezimalsystem ins Binärsystem umrechnen und umgekehrt,
- zwei natürliche Zahlen im Binärsystem schriftlich addieren und dabei die Überträge richtig berücksichtigen,
- den Aufbau des Halbaddierers aus einem XOR- und einem UND-Gatter skizzieren,
- die Funktionsweise des Halbaddierers mit seiner Belegungstabelle erläutern,
- einen Halbaddierer in CircuitVerse bauen und als Python-Funktion nachbilden.

2.3.1 Zahlen im Binärsystem

Im Alltag schreiben wir Zahlen im **Dezimalsystem** (Zehnersystem). Es gibt zehn Ziffern, 0 bis 9. Welchen Wert eine Ziffer hat, hängt von ihrer Stelle ab. In der Zahl 347 steht die 3 an der Hunderterstelle, die 4 an der Zehnerstelle und die 7 an der Einerstelle: $347 = 3 \cdot 100 + 4 \cdot 10 + 7 \cdot 1$. Von Stelle zu Stelle wird der Stellenwert zehnmal so groß.

Das **Binärsystem** (Zweiersystem) funktioniert genauso, nur mit zwei Ziffern: 0 und 1. Von Stelle zu Stelle wird der Stellenwert **doppelt** so groß. Von rechts nach links sind die Stellenwerte also 1, 2, 4, 8, 16, 32 und so weiter. Jede Stelle einer Binärzahl ist ein Bit.

Definition: Binärsystem

Im **Binärsystem** (Zweiersystem) werden Zahlen nur mit den Ziffern 0 und 1 geschrieben. Die Stellenwerte sind von rechts nach links 1, 2, 4, 8, 16, ...; jeder ist doppelt so groß wie der vorige.

Von binär nach dezimal

Um eine Binärzahl in eine Dezimalzahl umzurechnen, schreibst du sie in eine **Stellenwerttabelle**. Dann addierst du die Stellenwerte, unter denen eine 1 steht. Beispiel: die Binärzahl 1101.

Stellenwert	**8**	**4**	**2**	**1**
Ziffer	1	1	0	1

Unter 8, 4 und 1 steht eine 1. Also ist die Binärzahl 1101 gleich $8+4+1 = 13$. Damit man die Zahlensysteme nicht verwechselt, schreibt man manchmal das System klein dahinter: $1101_2 = 13_{10}$.

Von dezimal nach binär

Für den umgekehrten Weg füllst du die Stellenwerttabelle von links nach rechts. Beginne mit dem größten Stellenwert, der in die Zahl passt. Beispiel: die Dezimalzahl 22.

1. Die Stellenwerte sind 1, 2, 4, 8, 16, 32, … Der größte, der in 22 passt, ist 16. Schreibe unter 16 eine 1. Es bleiben $22 - 16 = 6$.
2. Passt 8 in 6? Nein. Schreibe unter 8 eine 0.
3. Passt 4 in 6? Ja. Schreibe unter 4 eine 1. Es bleiben $6 - 4 = 2$.
4. Passt 2 in 2? Ja. Schreibe unter 2 eine 1. Es bleibt 0.
5. Unter 1 kommt eine 0, denn es ist nichts mehr übrig.

Stellenwert	**16**	**8**	**4**	**2**	**1**
Ziffer	1	0	1	1	0

Also ist $22_{10} = 10110_2$. Zur Probe rechnest du zurück: $16 + 4 + 2 = 22$.

Nullen am Anfang ändern den Wert einer Zahl nicht: 0101 und 101 sind beide gleich 5. Im Folgenden schreiben wir Binärzahlen oft mit vier Stellen, also 0101 statt 101. Dann stehen beim Rechnen alle Stellen schön untereinander.

Merke

Im Binärsystem gibt es nur die Ziffern 0 und 1. Die Stellenwerte sind von rechts nach links 1, 2, 4, 8, 16, … Den Dezimalwert einer Binärzahl erhält man, indem man die Stellenwerte addiert, unter denen eine 1 steht.

Arbeitsauftrag 26: Umrechnen

Rechne um. Mache jeweils die Probe.

dezimal	**6**	**9**	**12**	**15**
binär				

binär	**0111**	**1010**	**1011**	**10000**
dezimal				

Werkzeug

Python kann Zahlen zwischen Dezimal- und Binärsystem umrechnen. Diese beiden Befehle brauchst du nur zum Nachprüfen:

- `int("1101", 2)` wandelt die Zeichenkette `"1101"` in eine Zahl um. Die `2` sagt Python, dass die Ziffern im Binärsystem gemeint sind. Ergebnis: `13`.
- `bin(13)` liefert die Binärschreibweise von 13 als Zeichenkette: `'0b1101'`. Die Vorsilbe `0b` zeigt nur an, dass eine Binärzahl folgt.

```
print(int("1101", 2))
print(bin(22))
```

AUSGABE

```
13
0b10110
```

2.3.2 Schriftliche Addition im Binärsystem

Erinnere dich an die schriftliche Addition im Dezimalsystem. Du rechnest Stelle für Stelle von rechts nach links. Wird eine Stelle 10 oder größer, schreibst du nur die Einerziffer hin und nimmst eine 1 als **Übertrag** in die nächste Stelle mit:

```
    4 7
  + 3 8
Ü   1
  -----
    8 5
```

Rechts: $7 + 8 = 15$. Du schreibst 5 und überträgst 1. Links: $4 + 3 + 1 = 8$. Die Zeile mit dem Ü enthält die Überträge. Jeder Übertrag steht genau unter der Stelle, zu der er addiert wird.

Im Binärsystem ist es genauso, nur dass schon bei 2 ein Übertrag entsteht. Denn eine Ziffer 2 gibt es nicht; 2 schreibt man binär als 10. Für zwei Bits gibt es nur vier Fälle:

Rechnung	Ergebnis (binär)	schreibe	Übertrag
$0 + 0$	0	0	0
$0 + 1$	1	1	0
$1 + 0$	1	1	0
$1 + 1$	10	0	1

Kommt ein Übertrag aus der Stelle davor dazu, können auch drei Einsen zusammenkommen: $1 + 1 + 1 = 3$, binär 11. Dann schreibst du 1 und überträgst 1.

Merke

- Die **binäre Addition** funktioniert wie die schriftliche Addition im Dezimalsystem, nur mit den Ziffern 0 und 1: $0 + 0 = 0$, $0 + 1 = 1$, $1 + 0 = 1$, $1 + 1 = 10$ (also 0 schreiben, 1 **Übertrag**).
- Mit einem Übertrag aus der Stelle davor gilt außerdem $1 + 1 + 1 = 11$ (also 1 schreiben, 1 Übertrag).

Beispiel: 0101 + 0011

Wir addieren 0101 (das ist 5) und 0011 (das ist 3):

```
      0 1 0 1     (= 5)
    + 0 0 1 1     (= 3)
Ü     1 1 1
    ---------
      1 0 0 0     (= 8)
```

Von rechts nach links:

Stelle	Rechnung	schreibe	Übertrag in die nächste Stelle
1 (ganz rechts)	$1+1=10$	0	1
2	$0+1+1=10$	0	1
3	$1+0+1=10$	0	1
4	$0+0+1=1$	1	0

Ergebnis: 1000. Probe: $1000_2 = 8$ und $5+3=8$. ✓

In diesem Beispiel entsteht in fast jeder Stelle ein Übertrag. Ab der zweiten Stelle werden deshalb **drei** Bits addiert: die beiden Ziffern der Zahlen und der Übertrag. Das wird im nächsten Abschnitt wichtig.

Stolperstein

Typische Fehler bei der Binäraddition:

- **„1 + 1 = 2“ hinschreiben.** Die Ziffer 2 gibt es im Binärsystem nicht. Richtig: 0 schreiben, 1 übertragen.
- **Übertrag vergessen.** Schreibe jeden Übertrag sofort in die Ü-Zeile, und zwar unter die nächste Stelle links.
- **Stellen verrutscht.** Ergänze kürzere Zahlen vorn mit Nullen, damit die Stellen genau untereinanderstehen.
- **Letzten Übertrag vergessen.** Entsteht ganz links noch ein Übertrag, wird das Ergebnis eine Stelle länger: $1111 + 0001 = 10000$.

Mache immer die Probe im Dezimalsystem.

Arbeitsauftrag 27: Von Hand rechnen

Berechne **von Hand** Schritt für Schritt mit Überträgen die folgenden binären Additionen. Kontrolliere anschließend mit der Probe im Dezimalsystem und mit Python (siehe Werkzeug-Kasten oben).

1. $0110 + 0101$
2. $1011 + 0111$
3. $1111 + 0001$

Deine Rechnungen:

2.3.3 Eine Schaltung, die zwei Bits addiert

Jetzt soll eine Schaltung die Arbeit übernehmen. Wir beginnen mit der einfachsten Aufgabe: der Stelle ganz rechts. Dort werden nur **zwei** Bits addiert, nennen wir sie A und B. Das Ergebnis besteht aus zwei Bits: der Ziffer, die man hinschreibt, und dem Übertrag. Man nennt sie **Summe** S und **Übertrag** C (vom englischen *carry*, „tragen“).

A	B	$A+B$ **(dezimal)**	**Übertrag** C	**Summe** S
0	0	0	0	0
0	1	1	0	1
1	0	1	0	1
1	1	2	1	0

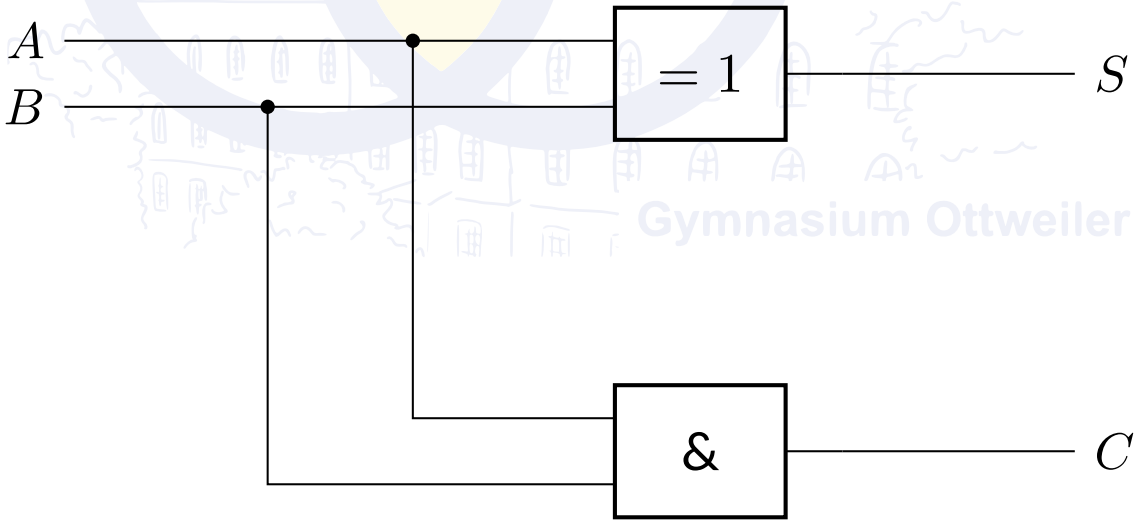
Liest man C und S zusammen als zweistellige Binärzahl, steht dort genau die Summe: 00, 01, 01, 10, also 0, 1, 1, 2.

Schau dir die beiden Ausgangsspalten genau an. Kommen sie dir bekannt vor?

- Die Spalte S ist 1, wenn A und B **unterschiedlich** sind. Das ist die Belegungstabelle des **XOR-Gatters**.
- Die Spalte C ist nur 1, wenn A **und** B beide 1 sind. Das ist die Belegungstabelle des **UND-Gatters**.

Mehr braucht man nicht. Eine Schaltung aus einem XOR-Gatter und einem UND-Gatter addiert zwei Bits. Sie heißt **Halbaddierer**.

2.3.4 Der Halbaddierer



Halbaddierer: Das XOR-Gatter liefert die Summe S, das UND-Gatter den Übertrag C

So ist der Halbaddierer aufgebaut:

- Die beiden Eingänge A und B werden **beiden** Gattern zugeführt. Deshalb verzweigen sich die Leitungen. Die Punkte markieren die Verzweigungen. Wo sich die Leitung von A und die Leitung von B nur kreuzen, ist kein Punkt.

- Das XOR-Gatter berechnet die Summe: $S = A \oplus B$.
- Das UND-Gatter berechnet den Übertrag: $C = A \wedge B$.

Die Belegungstabelle des Halbaddierers fasst das zusammen:

A	B	$S = A \oplus B$	$C = A \wedge B$
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Funktionsweise an einem Beispiel: Liegt an A und an B je eine 1 an, dann sind die Eingänge gleich. Das XOR-Gatter liefert also $S = 0$. Das UND-Gatter liefert $C = 1$, weil beide Eingänge 1 sind. Zusammen ergibt das die Binärzahl 10, also 2. Das ist richtig, denn $1 + 1 = 2$.

Definition: Halbaddierer

Ein **Halbaddierer** ist eine Schaltung mit zwei Eingängen A, B und zwei Ausgängen. Er addiert die beiden Bits und liefert die Summe $S = A \oplus B$ (XOR-Gatter) und den Übertrag $C = A \wedge B$ (UND-Gatter).

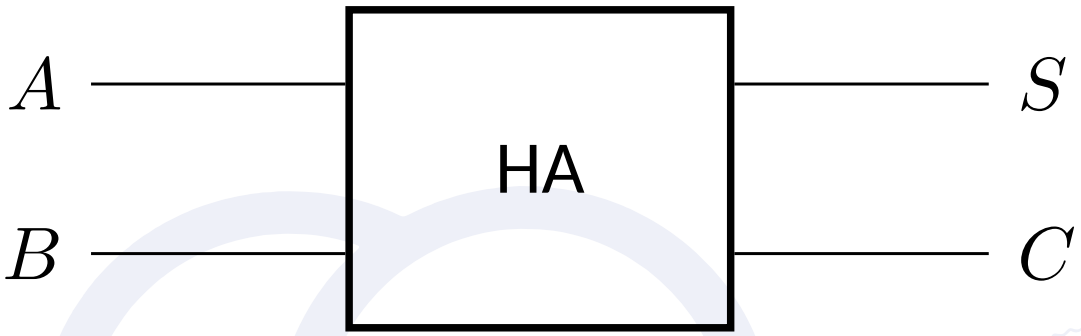
Warum heißt er „halb“? Schau noch einmal auf das Beispiel $0101 + 0011$. In der Stelle ganz rechts werden zwei Bits addiert; das schafft der Halbaddierer. Ab der zweiten Stelle muss aber zusätzlich der **Übertrag aus der Stelle davor** addiert werden. Dafür hat der Halbaddierer keinen Eingang. Er erledigt also nur die halbe Arbeit. Eine Schaltung, die auch den eingehenden Übertrag verarbeitet, lernst du im nächsten Abschnitt kennen: den Volladdierer.

Gymnasium Ottweiler

Merke

- Ein **Halbaddierer** addiert **zwei einzelne Bits** (A, B) und liefert zwei Ausgänge:
  - **Summe** $S = A \oplus B$ (XOR),
  - **Übertrag** $C = A \wedge B$ (UND; englisch *carry*).
- „Halb“-Addierer heißt er, weil er **keinen eingehenden Übertrag** aus einer vorherigen Stelle berücksichtigen kann. Das kann erst der Volladdierer.

Wenn man den Halbaddierer in größere Schaltungen einbaut, zeichnet man oft nicht jedes Gatter, sondern nur einen Kasten mit der Aufschrift HA. So ein vereinfachtes Bild heißt **Blockschaltbild**. Es zeigt nur, was hineingeht und was herauskommt.



Blockschaltbild des Halbaddierers: zwei Eingänge, zwei Ausgänge

So kannst du es sagen

So beschreibst du den Halbaddierer in Fachsprache – vom einfachen zum genauen Satz:

- *„Da gehen zwei Leitungen rein und zwei raus.“ → „Ein Halbaddierer hat zwei Eingänge und zwei Ausgänge.“*
- *„Der rechnet A plus B.“ → „Ein Halbaddierer addiert zwei Bits und liefert die Summe und den Übertrag.“*
- *„Bei 1 und 1 kommt 0 und 1 raus.“ → „Ein Halbaddierer addiert die an den beiden Eingängen anliegenden Bits und liefert deren Summe und den entstehenden Übertrag. Liegen zwei Einsen an, ist die Summe 0 und der Übertrag 1.“*

Arbeitsauftrag 28: Halbaddierer in CircuitVerse

1. Lege zwei Input-Schalter A und B an.
2. Füge ein **XOR-Gatter** hinzu (Eingänge: A, B). Sein Ausgang ist die **Summe** S.
3. Füge ein **AND-Gatter** hinzu (Eingänge: A, B). Sein Ausgang ist der **Übertrag** C.
4. Verbinde beide Gatterausgänge mit je einem Output.
5. Teste alle vier Belegungen, trage die Werte ein und vergleiche mit der Belegungstabelle oben.

A	B	S	C
0	0		
0	1		
1	0		
1	1		

Arbeitsauftrag 29: Halbaddierer in Python

Schreibe eine Funktion `halbaddierer(a, b)`, die die Summe und den Übertrag als Liste `[summe, uebertrag]` zurückgibt. Teste sie für alle vier Belegungen.

Tipp

Mit `return` gibt eine Funktion einen Wert zurück. Wenn du zwei Ergebnisse brauchst, packst du sie in eine Liste: `return [summe, uebertrag]`. An die einzelnen Werte kommst du später mit dem Index heran: `ergebnis[0]` ist die Summe, `ergebnis[1]` der Übertrag.

2.3.5 Übungen

Arbeitsauftrag 30: Halbaddierer erläutern

1. Erläutere den Aufbau und die Funktionsweise eines Halbaddierers. Gehe dabei auf die Belegung $A = 0, B = 1$ ein.
2. Begründe am Beispiel $0101 + 0011$, warum ein Halbaddierer für die zweite Stelle von rechts nicht ausreicht.

Gymnasium Ottweiler

Deine Antwort:

Arbeitsauftrag 31: Ergebnis vorhersagen

An einem Halbaddierer liegt $A = 1$ und $B = 0$ an. Gib ohne Tabelle an, welche Werte S und C haben, und begründe mit den beiden Gattern. Welche Belegung führt zu $C = 1$?

Deine Antwort:

Gymnasium Ottweiler

2.3.6 Weitere Übungen

Hier kannst du das Gelernte weiter üben. Die Punkte im Titel zeigen, wie schwer eine Aufgabe ist: ●○○ leicht, ●●○ mittel, ●●● knifflig. Fang mit den leichten an.

Arbeitsauftrag 32: Bits addieren ●○○

Bestimme für jede Rechnung, welche Ziffer man hinschreibt und welcher Übertrag entsteht.

Rechnung	Ergebnis (binär)	schreibe	Übertrag
$1 + 0$			
$1 + 1$			
$0 + 0 + 1$			
$1 + 0 + 1$			
$1 + 1 + 1$			

Arbeitsauftrag 33: Addieren ohne Übertrag ●○○

Berechne von Hand. Die Dezimalwerte stehen zur Kontrolle in Klammern.

a) $0100 + 0011$ (4 + 3)
b) $1010 + 0101$ (10 + 5)
c) $1000 + 0110$ (8 + 6)

Woran erkennst du schon vor dem Rechnen, dass kein Übertrag entsteht?

Deine Rechnungen und deine Antwort:

Gymnasium Ottweiler

Arbeitsauftrag 34: Addieren mit einem Übertrag ●○○

Berechne von Hand. Schreibe den Übertrag in die Ü-Zeile.

a) $0010 + 0010$ (2 + 2)
b) $0101 + 0001$ (5 + 1)
c) $0110 + 0100$ (6 + 4)

Deine Rechnungen:

Arbeitsauftrag 35: Lückentext zum Halbaddierer ●○○

Ergänze die Lücken (1) bis (7).

Ein Halbaddierer hat (1) Eingänge und (2) Ausgänge. Die Summe S liefert ein (3)-Gatter, den Übertrag C ein (4)-Gatter. Liegt an beiden Eingängen eine 1 an, ist $S =$ (5) und $C =$ (6). „Halb“ heißt er, weil er keinen (7) aus der Stelle davor verarbeiten kann.

Deine Lösungen:

Arbeitsauftrag 36: Addieren mit mehreren Überträgen ●●○

Berechne von Hand Schritt für Schritt. Schreibe alle Überträge in die Ü-Zeile. Prüfe dein Ergebnis anschließend mit Python.

a) $0111 + 0011$ (7 + 3)
b) $0110 + 0110$ (6 + 6)
c) $1110 + 0011$ (14 + 3)

Deine Rechnungen:

Arbeitsauftrag 37: Fehler finden ●●○

Lena und Tim haben gerechnet. Beide Ergebnisse sind falsch.

```
Lena:   0 1 0 1          Tim:   0 1 1 0
      + 0 0 0 1                + 0 1 1 1
        ---------                ---------
          0 1 0 2                  0 0 0 1
```

1. Beschreibe jeweils den Fehler.
2. Rechne beide Aufgaben richtig.
3. Tim hat in jeder Spalte dasselbe Gatter „nachgespielt“. Welches Gatter ist das?

Deine Antwort und die richtigen Rechnungen:

Arbeitsauftrag 38: Rechnungen mit Python prüfen ●●○

Schreibe eine Funktion `addiere(a_text, b_text)`, die zwei Binärzahlen als Zeichenketten bekommt. Sie soll die Summe als Binärzahl ausgeben und zur Kontrolle auch die Rechnung im Dezimalsystem. Prüfe damit deine Ergebnisse aus dem Auftrag *Addieren mit mehreren Überträgen*.

Für `addiere("0111", "0011")` soll die Ausgabe so aussehen:

```
0111 + 0011 = 0b1010 | dezimal: 7 + 3 = 10
```

Tipp

Die Befehle `int("0111", 2)` und `bin(...)` kennst du aus dem Werkzeug-Kasten im Abschnitt über das Binärsystem.

Arbeitsauftrag 39: Fehlende Ziffern ●●●

In jeder Rechnung ist eine Ziffer durch **x** ersetzt. Bestimme x und begründe, Spalte für Spalte von rechts.

a) $1\text{x}01 + 0110 = 10011$
b) $01\text{x}1 + 0101 = 1100$

Deine Begründung:

2.3.7 Das Wichtigste in Kürze

- Im **Binärsystem** gibt es nur die Ziffern 0 und 1. Die Stellenwerte sind 1, 2, 4, 8, 16, … Beispiel: $1101_2 = 8 + 4 + 1 = 13$.
- Binärzahlen werden schriftlich addiert wie Dezimalzahlen: Stelle für Stelle von rechts nach links. Es gilt $1 + 1 = 10$ (0 schreiben, 1 übertragen) und $1 + 1 + 1 = 11$.
- Die Überträge schreibt man in eine eigene Zeile. Die Probe im Dezimalsystem deckt Fehler auf.
- Ein **Halbaddierer** addiert zwei Bits: Summe $S = A \oplus B$ (XOR-Gatter), Übertrag $C = A \wedge B$ (UND-Gatter).
- Er heißt „halb“, weil er keinen Übertrag aus der Stelle davor verarbeiten kann.

Selbstcheck

- ☐ Ich kann eine Binärzahl mit der Stellenwerttabelle in eine Dezimalzahl umrechnen und umgekehrt.
- ☐ Ich kann zwei Binärzahlen schriftlich addieren und schreibe die Überträge in eine eigene Zeile.
- ☐ Ich weiß, was $1 + 1$ und $1 + 1 + 1$ im Binärsystem ergibt.
- ☐ Ich kann den Halbaddierer aus XOR- und UND-Gatter skizzieren.
- ☐ Ich kann die Belegungstabelle des Halbaddierers angeben und seine Funktionsweise erläutern.
- ☐ Ich kann erklären, warum der Halbaddierer „halb“ heißt.

Gymnasium Ottweiler