Einführung in die Programmierung

Tutorium 1 ('Binärzahlen-Tutorium')

Wintersemester 2025/2026

Arbeitsgruppe Systemsoftware
Angewandte Informatik 12

technische universität
dortmund

Arbeitsgruppe
Systemsoftware

- **Einfache Datentypen** enthalten genau einen Wert: `int`, `float`, `char`, …
- **Zusammengesetzte Datentypen** fassen mehrere Werte zusammen: Arrays, Structs, …

```
printf("Geben Sie Datum und Uhrzeit (TT.MM.JJJJ; SS:MM) ein:\n");
scanf("%u.%u.%u; %u:%u", &tag, &monat, &jahr, &stunde, &minute);
```

$$\mathrm{int}(\vec{b}) = \sum_{i=0}^{n-1} 2^i b_i$$

$$\text{int}(\vec{b}) = \sum_{i=0}^{n-1} 2^i b_i$$

$$\sum a_n 2^n = a_0 \cdot 2^0 + a_1 \cdot 2^1 + a_2 \cdot 2^2 + a_3 \cdot 2^3 + \cdots$$

$$\text{int}(\vec{b}) = \sum_{i=0}^{n-1} 2^i b_i$$

$$\begin{aligned}\sum a_n 2^n &= a_0 \cdot 2^0 + a_1 \cdot 2^1 + a_2 \cdot 2^2 + a_3 \cdot 2^3 + \cdots \\ &= a_0 \cdot 1 + a_1 \cdot 2 + a_2 \cdot 4 + a_3 \cdot 8 + \cdots\end{aligned}$$

$$\text{int}(\vec{b}) = \sum_{i=0}^{n-1} 2^i b_i$$

$$\begin{aligned}\sum a_n 2^n &= a_0 \cdot 2^0 + a_1 \cdot 2^1 + a_2 \cdot 2^2 + a_3 \cdot 2^3 + \cdots \\ &= a_0 \cdot 1 + a_1 \cdot 2 + a_2 \cdot 4 + a_3 \cdot 8 + \cdots \\ &= 1 \cdot 1 + 0 \cdot 2 + 1 \cdot 4 + 1 \cdot 8 + \cdots\end{aligned}$$

$$\text{int}(\vec{b}) = \sum_{i=0}^{n-1} 2^i b_i$$

$$\begin{aligned}
\sum a_n 2^n &= a_0 \cdot 2^0 + a_1 \cdot 2^1 + a_2 \cdot 2^2 + a_3 \cdot 2^3 + \cdots \\
&= a_0 \cdot 1 + a_1 \cdot 2 + a_2 \cdot 4 + a_3 \cdot 8 + \cdots \\
&= 1 \cdot 1 + 0 \cdot 2 + 1 \cdot 4 + 1 \cdot 8 + \cdots \\
&= 1 + 0 + 4 + 8
\end{aligned}$$

$$\text{int}(\vec{b}) = \sum_{i=0}^{n-1} 2^i b_i$$

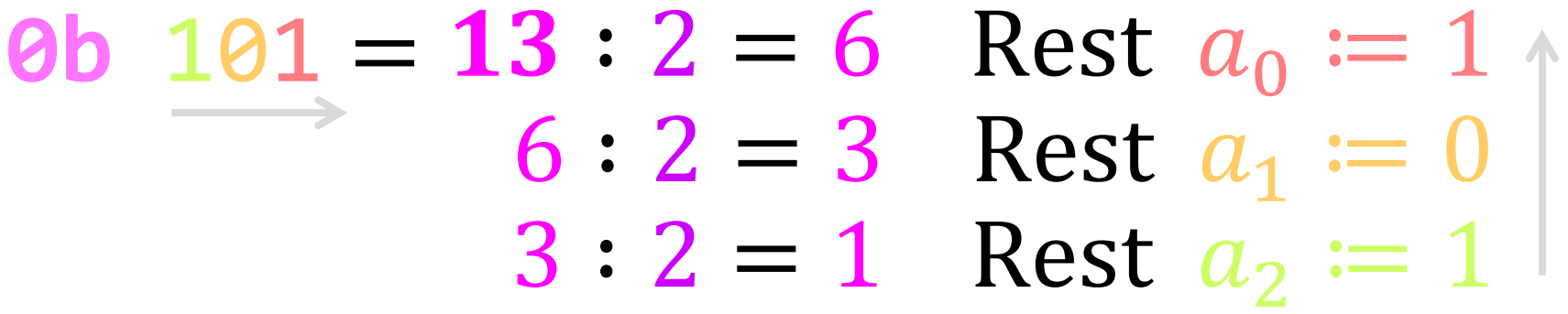
$$\begin{aligned}
\sum a_n 2^n &= a_0 \cdot 2^0 + a_1 \cdot 2^1 + a_2 \cdot 2^2 + a_3 \cdot 2^3 + \cdots \\
&= a_0 \cdot 1 + a_1 \cdot 2 + a_2 \cdot 4 + a_3 \cdot 8 + \cdots \\
&= 1 \cdot 1 + 0 \cdot 2 + 1 \cdot 4 + 1 \cdot 8 + \cdots \\
&= 1 + 0 + 4 + 8 = 13
\end{aligned}$$

Dezimalzahl in Binärzahl umwandeln I

```
13 = 0b ????
```

$$\texttt{0b} \quad 1 = \mathbf{13} : 2 = 6 \quad \text{Rest } a_0 := 1$$

$$\texttt{0b}\ \underrightarrow{\texttt{01}} = \mathbf{13} : 2 = 6 \quad \text{Rest } a_0 := 1 \uparrow$$
$$6 : 2 = 3 \quad \text{Rest } a_1 := 0$$


0b 101 = 13 : 2 = 6 Rest $a_0 := 1$
6 : 2 = 3 Rest $a_1 := 0$
3 : 2 = 1 Rest $a_2 := 1$

$$\texttt{0b1101} = 13 : 2 = 6 \quad \text{Rest } a_0 := 1$$
$$6 : 2 = 3 \quad \text{Rest } a_1 := 0$$
$$3 : 2 = 1 \quad \text{Rest } a_2 := 1$$
$$1 : 2 = 0 \quad \text{Rest } a_3 := 1$$

Die Reste a_i werden **Bits** genannt.

827 = 0b ??????????????

$$\mathbf{827} : 2 = 413 \quad \text{Rest } 1$$

$$\mathbf{827} : 2 = 413 \quad \text{Rest } 1$$
$$413 : 2 = 206 \quad \text{Rest } 1$$

$$\begin{aligned} \mathbf{827} : 2 &= 413 \quad \text{Rest } 1 \\ 413 : 2 &= 206 \quad \text{Rest } 1 \\ 206 : 2 &= 103 \quad \text{Rest } 0 \end{aligned}$$

$$\begin{aligned} \mathbf{827} : 2 &= 413 \quad \text{Rest } 1 \\ 413 : 2 &= 206 \quad \text{Rest } 1 \\ 206 : 2 &= 103 \quad \text{Rest } 0 \\ 103 : 2 &= \ 51 \quad \text{Rest } 1 \end{aligned}$$

$$\begin{aligned}
\mathbf{827} : 2 &= 413 \quad \text{Rest } 1 \\
413 : 2 &= 206 \quad \text{Rest } 1 \\
206 : 2 &= 103 \quad \text{Rest } 0 \\
103 : 2 &= 51 \quad \text{Rest } 1 \\
51 : 2 &= 25 \quad \text{Rest } 1
\end{aligned}$$

$$\begin{aligned}
\mathbf{827} : 2 &= 413 \quad \text{Rest } 1 \\
413 : 2 &= 206 \quad \text{Rest } 1 \\
206 : 2 &= 103 \quad \text{Rest } 0 \\
103 : 2 &= 51 \quad \text{Rest } 1 \\
51 : 2 &= 25 \quad \text{Rest } 1 \\
25 : 2 &= 12 \quad \text{Rest } 1
\end{aligned}$$

$$\begin{aligned}
\mathbf{827} : 2 &= 413 \quad \text{Rest } 1 \\
413 : 2 &= 206 \quad \text{Rest } 1 \\
206 : 2 &= 103 \quad \text{Rest } 0 \\
103 : 2 &= 51 \quad \text{Rest } 1 \\
51 : 2 &= 25 \quad \text{Rest } 1 \\
25 : 2 &= 12 \quad \text{Rest } 1 \\
12 : 2 &= 6 \quad \text{Rest } 0
\end{aligned}$$

$$
\begin{aligned}
\mathbf{827} : 2 &= 413 \quad \text{Rest } 1 \\
413 : 2 &= 206 \quad \text{Rest } 1 \\
206 : 2 &= 103 \quad \text{Rest } 0 \\
103 : 2 &= 51 \quad \text{Rest } 1 \\
51 : 2 &= 25 \quad \text{Rest } 1 \\
25 : 2 &= 12 \quad \text{Rest } 1 \\
12 : 2 &= 6 \quad \text{Rest } 0 \\
6 : 2 &= 3 \quad \text{Rest } 0
\end{aligned}
$$

$$
\begin{aligned}
827 : 2 &= 413 \quad \text{Rest } 1 \\
413 : 2 &= 206 \quad \text{Rest } 1 \\
206 : 2 &= 103 \quad \text{Rest } 0 \\
103 : 2 &= 51 \quad \text{Rest } 1 \\
51 : 2 &= 25 \quad \text{Rest } 1 \\
25 : 2 &= 12 \quad \text{Rest } 1 \\
12 : 2 &= 6 \quad \text{Rest } 0 \\
6 : 2 &= 3 \quad \text{Rest } 0 \\
3 : 2 &= 1 \quad \text{Rest } 1
\end{aligned}
$$

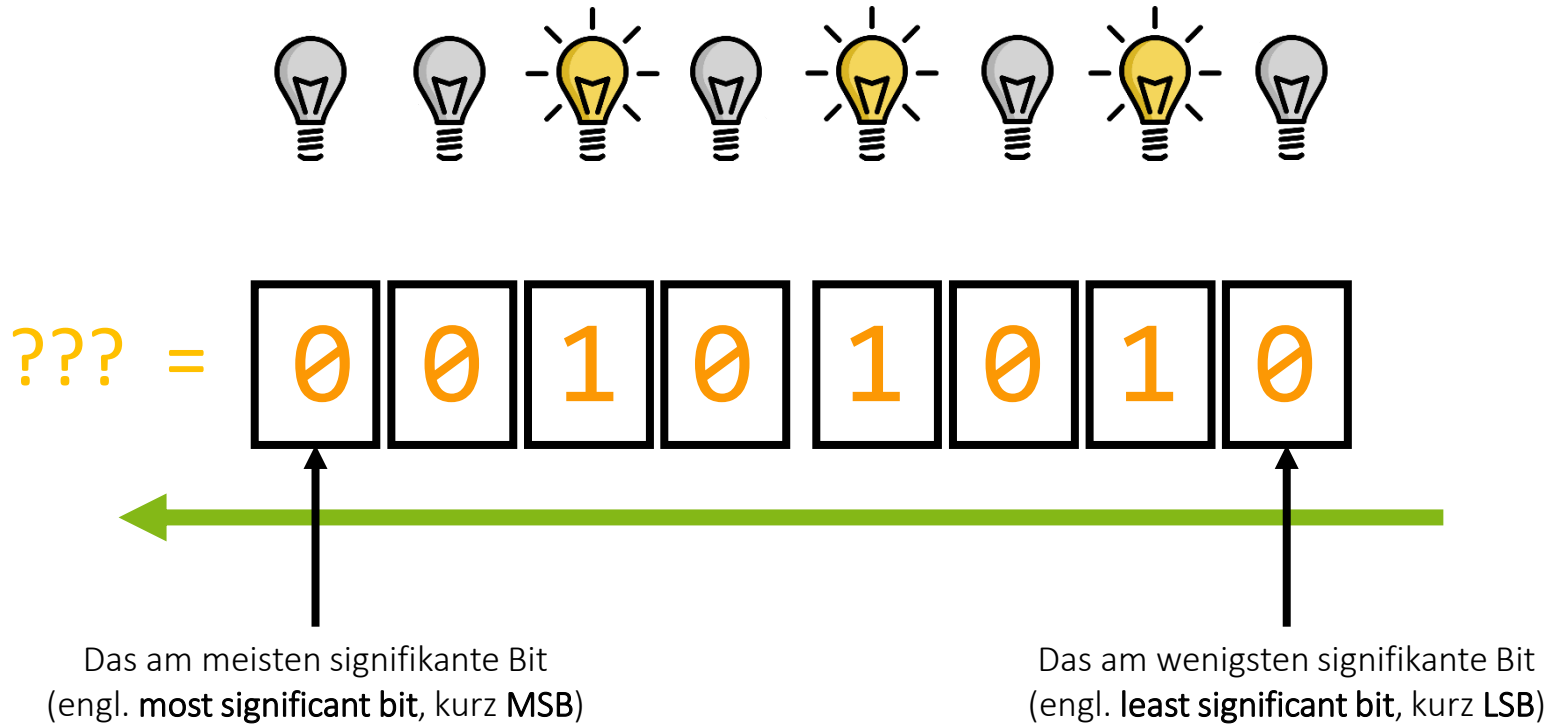
$$
\begin{aligned}
\mathbf{827} : 2 &= 413 \quad \text{Rest } 1 \\
413 : 2 &= 206 \quad \text{Rest } 1 \\
206 : 2 &= 103 \quad \text{Rest } 0 \\
103 : 2 &= 51 \quad \text{Rest } 1 \\
51 : 2 &= 25 \quad \text{Rest } 1 \\
25 : 2 &= 12 \quad \text{Rest } 1 \\
12 : 2 &= 6 \quad \text{Rest } 0 \\
6 : 2 &= 3 \quad \text{Rest } 0 \\
3 : 2 &= 1 \quad \text{Rest } 1 \\
1 : 2 &= \mathbf{0} \quad \text{Rest } 1
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{827} : 2 &= 413 \quad \text{Rest } 1 \\
413 : 2 &= 206 \quad \text{Rest } 1 \\
206 : 2 &= 103 \quad \text{Rest } 0 \\
103 : 2 &= 51 \quad \text{Rest } 1 \\
51 : 2 &= 25 \quad \text{Rest } 1 \\
25 : 2 &= 12 \quad \text{Rest } 1 \\
12 : 2 &= 6 \quad \text{Rest } 0 \\
6 : 2 &= 3 \quad \text{Rest } 0 \\
3 : 2 &= 1 \quad \text{Rest } 1 \\
1 : 2 &= \mathbf{0} \quad \text{Rest } 1
\end{aligned}
$$

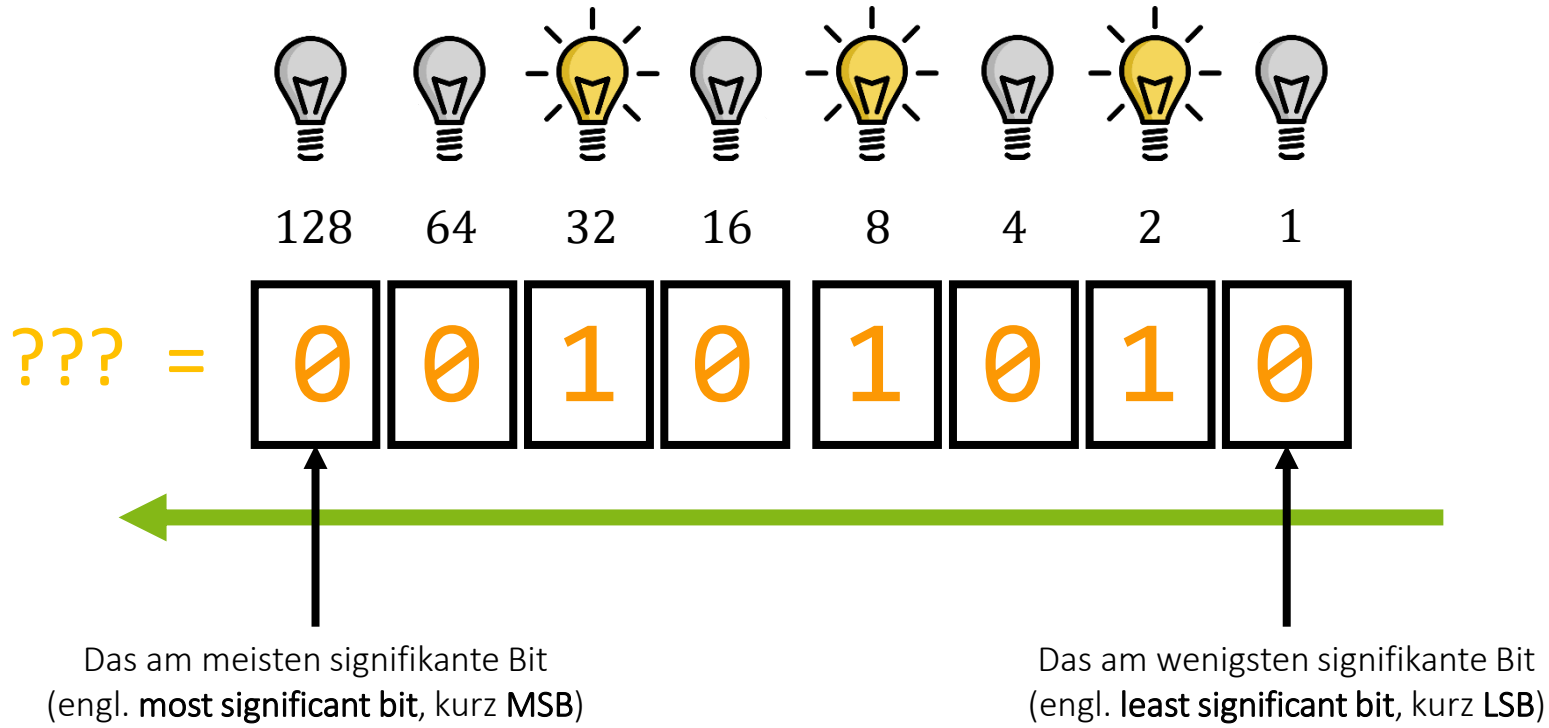
$$\texttt{0b}\ 0000\ 0011\ 0011\ 1011$$

??? = 0 0 1 0 1 0 1 0

Binärzahl in Dezimalzahlen umwandeln I



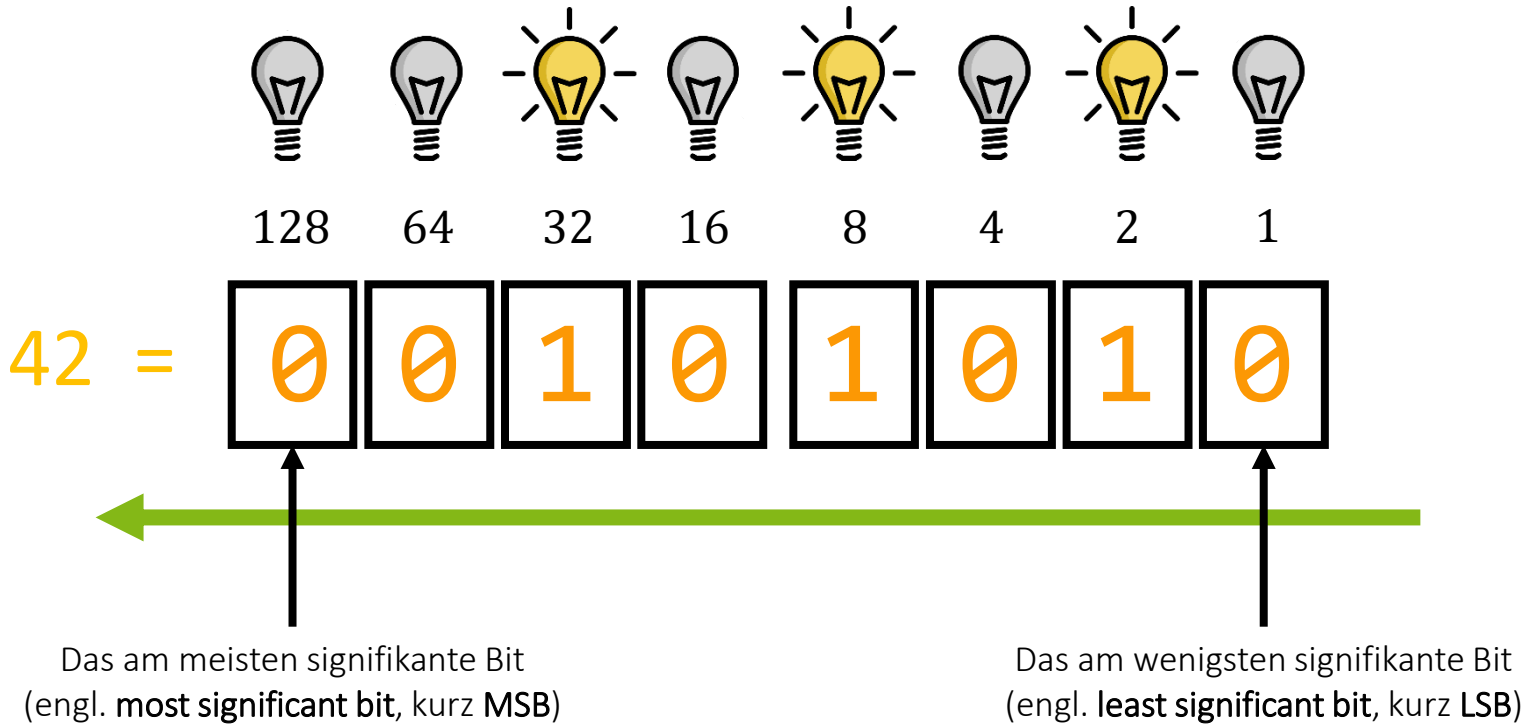
Binärzahl in Dezimalzahlen umwandeln I

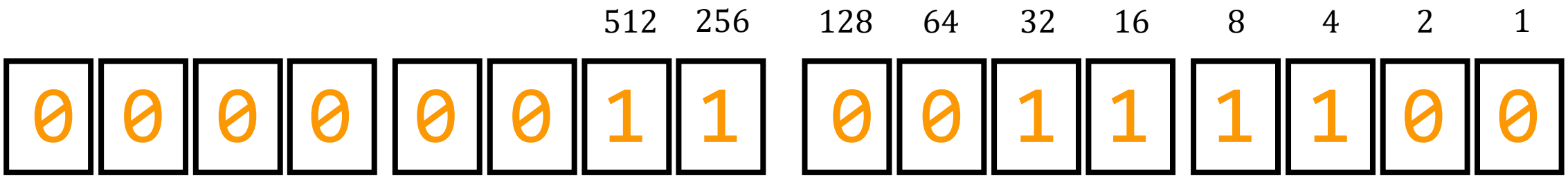


Das am meisten signifikante Bit
(engl. **most significant bit**, kurz **MSB**)

Das am wenigsten signifikante Bit
(engl. **least significant bit**, kurz **LSB**)

Binärzahl in Dezimalzahlen umwandeln I



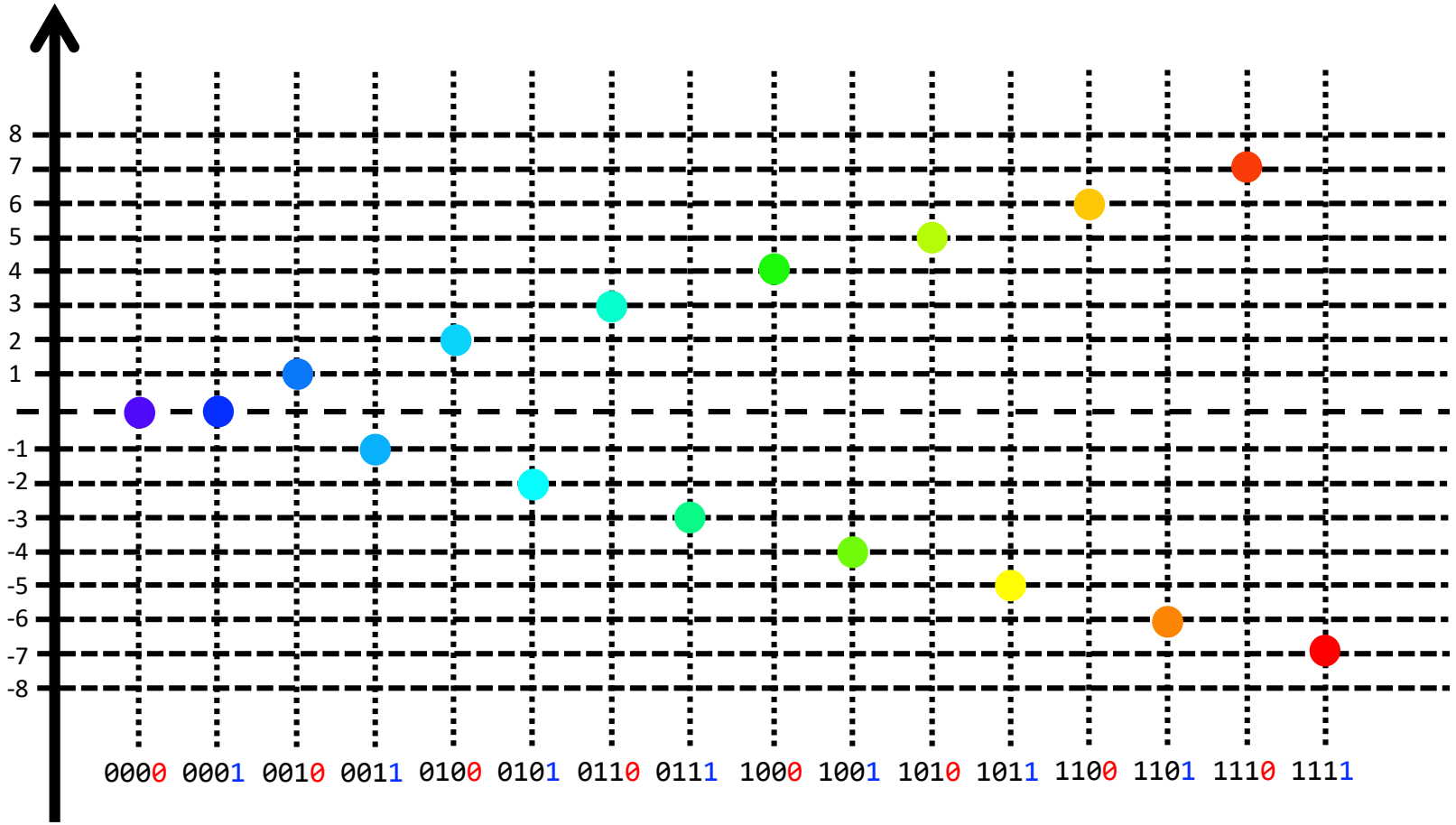


= 828

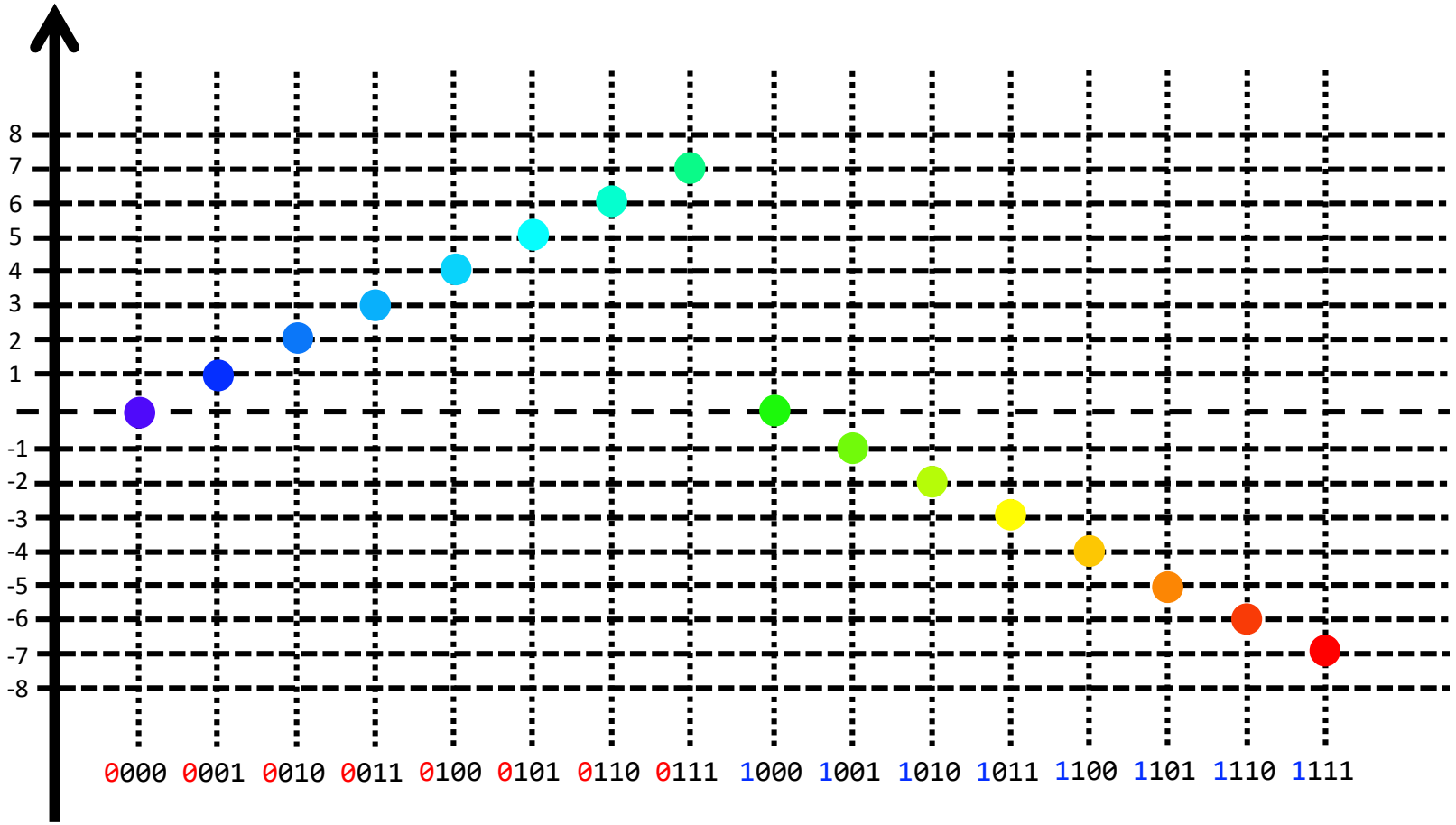
- Die ganzen Zahlen $\mathbb{Z}$ umfassen neben den natürlichen Zahlen $\mathbb{N}_0$ (mit Null) auch negative Zahlen $\mathbb{Z}_-$
- Um negative Zahlen darstellen zu können, wird ein weiteres sogenanntes **Vorzeichenbit** benötigt

$$\mathbb{Z} = \mathbb{N}_0 \cup \mathbb{Z}_- = \{0,1,-1,2,-2,3,-3,\dots\}$$

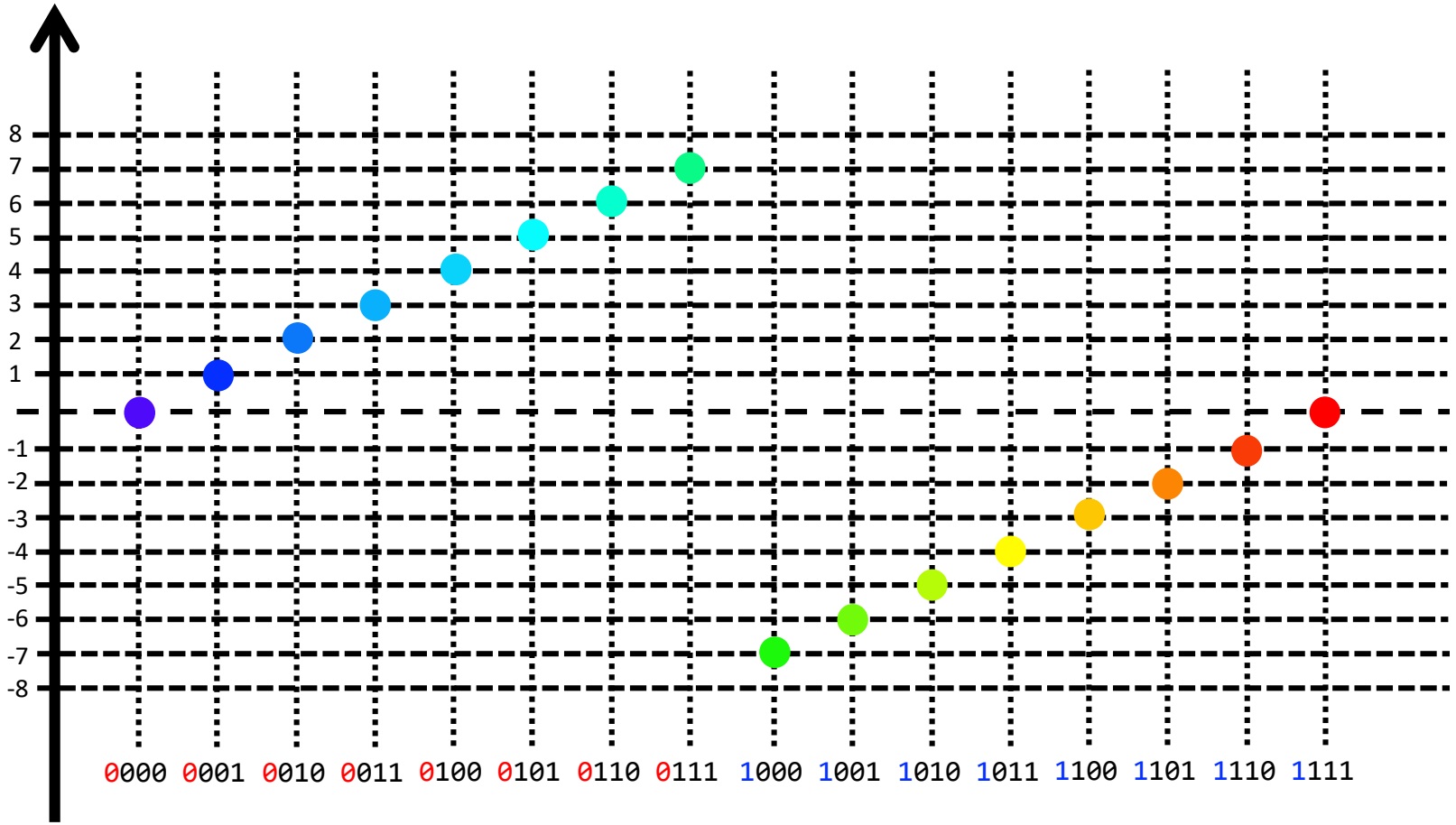
Vorzeichenbitdarstellung



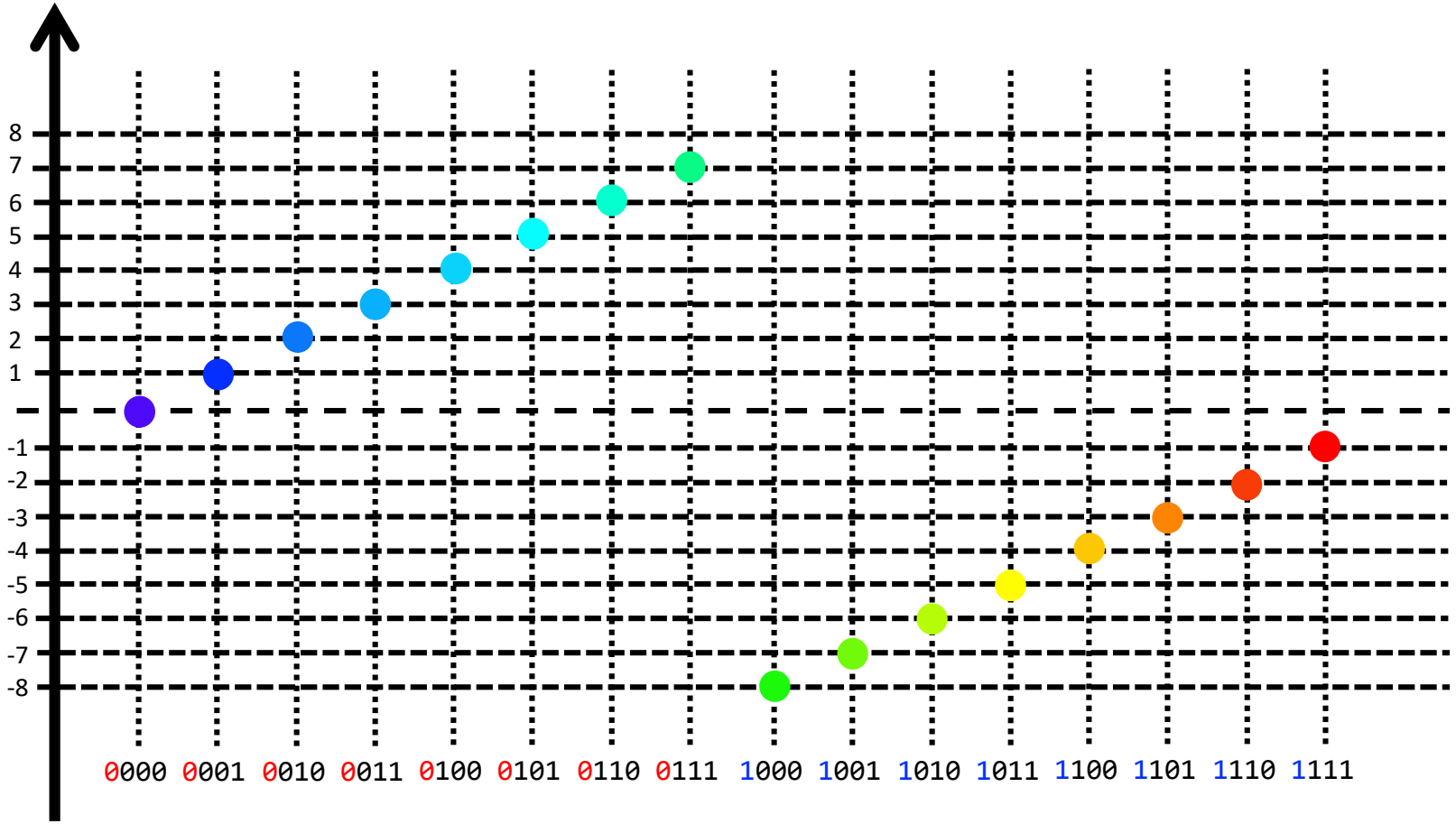
Vorzeichenbitdarstellung mit fester Breite



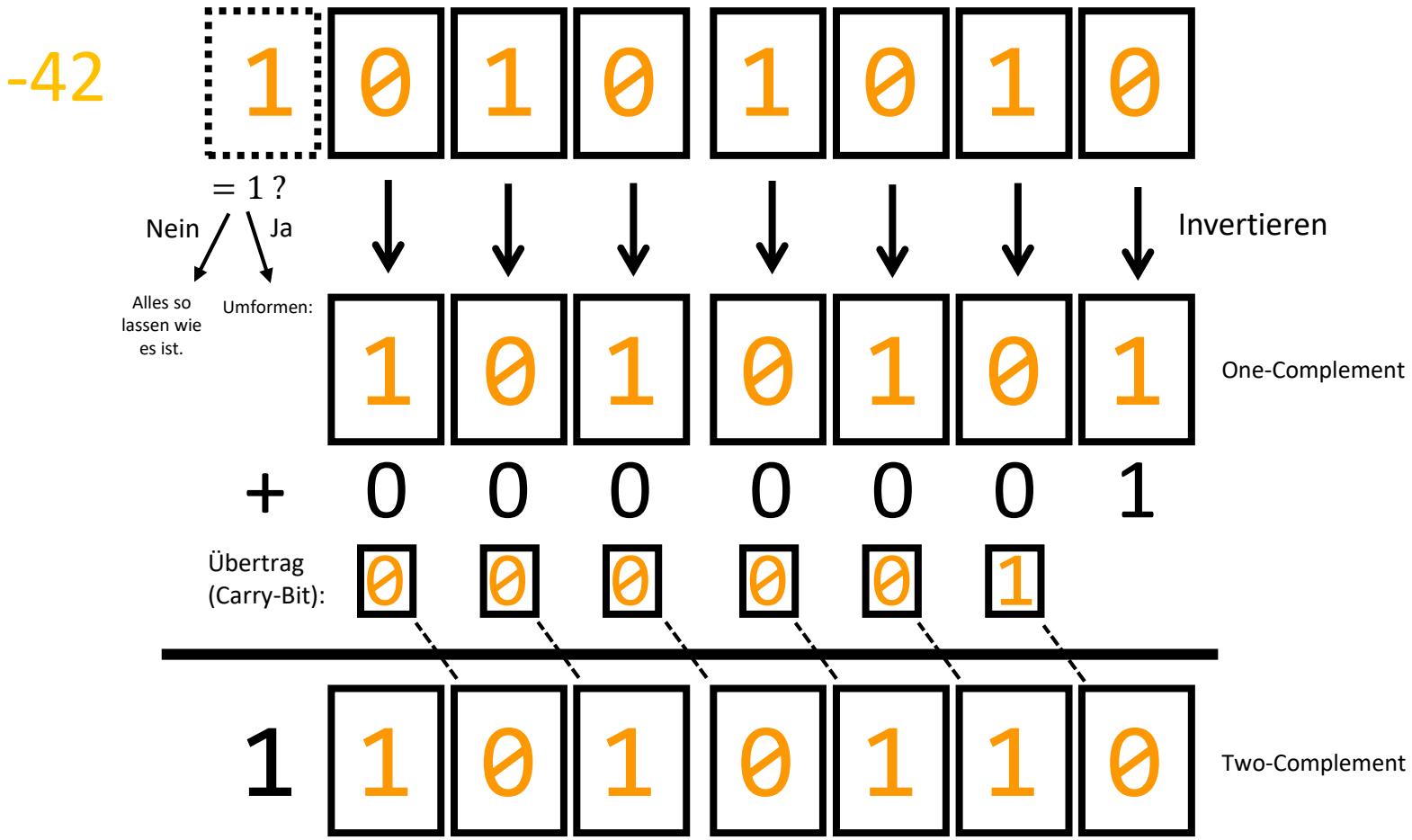
Einerkomplementendarstellung



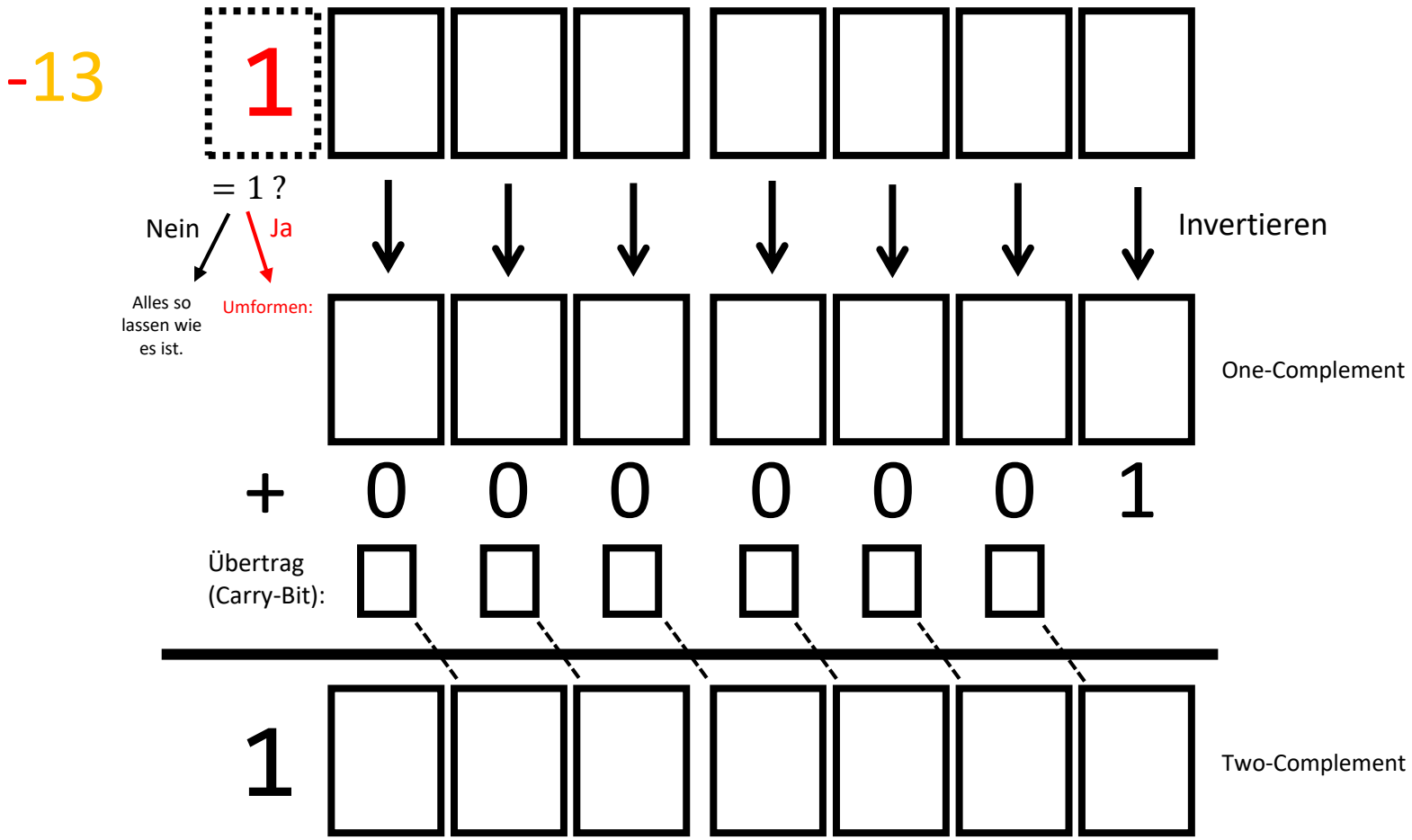
Zweierkomplementendarstellung



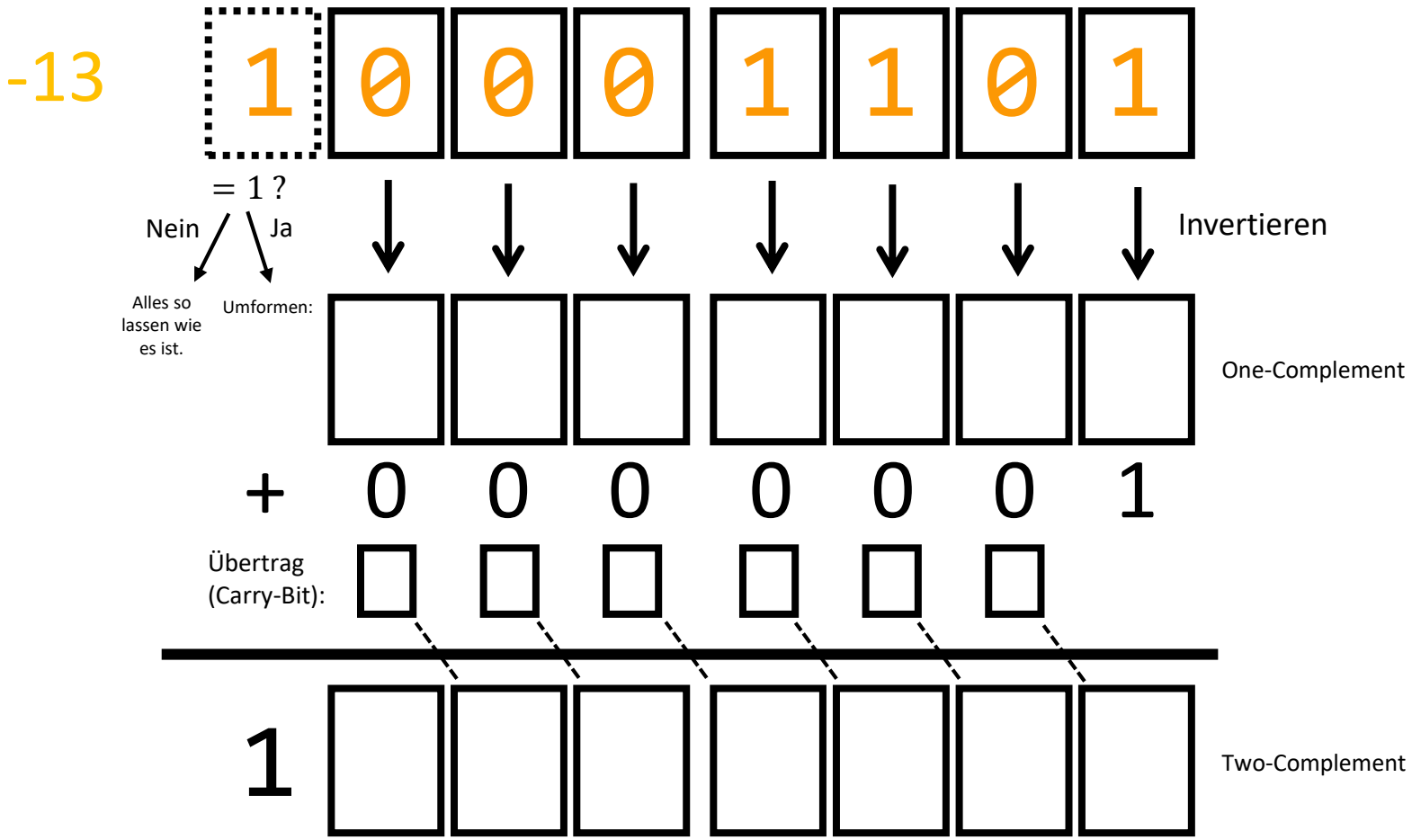
Beispiel einer Zweierkomplementendarstellung



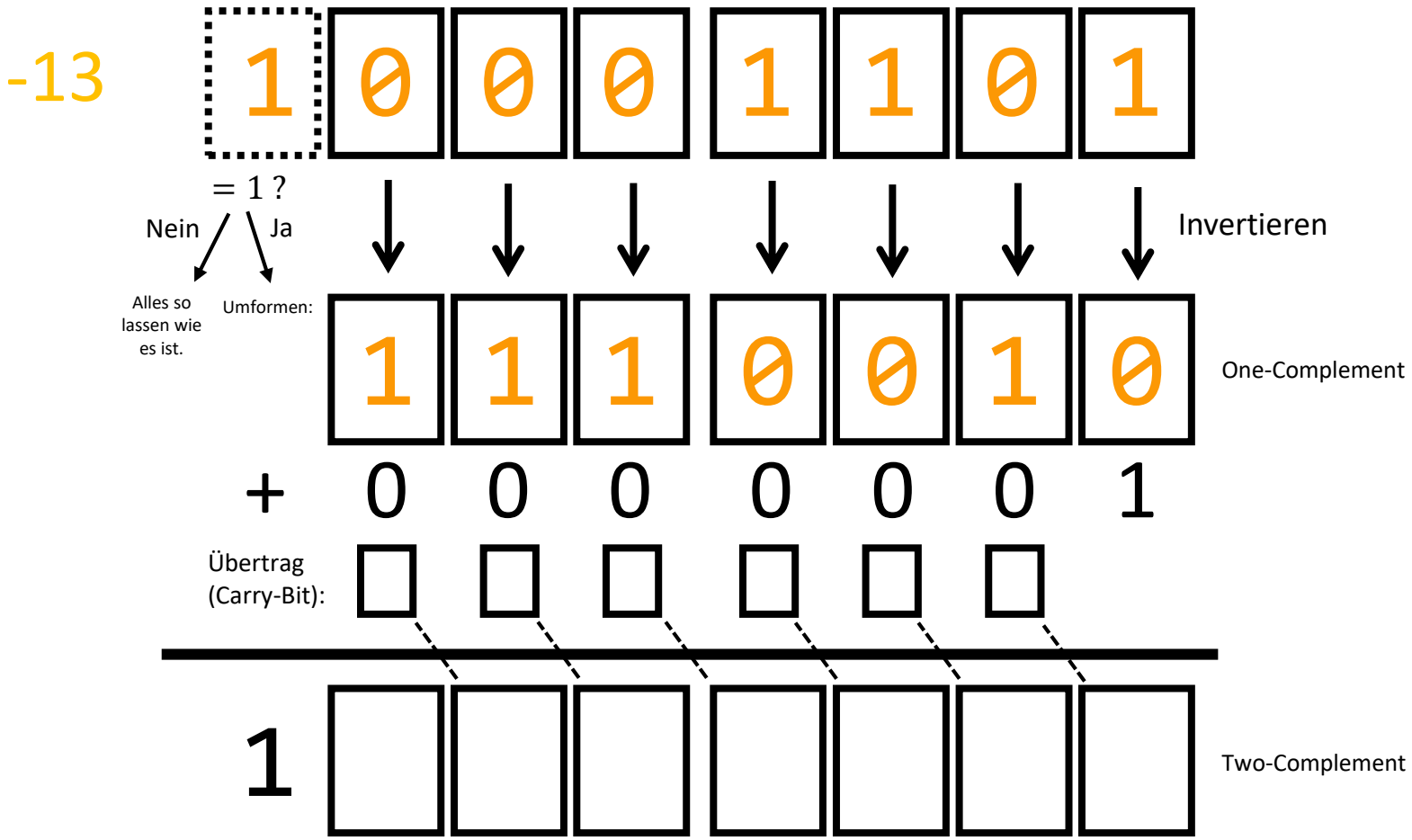
Beispiel einer Zweierkomplementendarstellung



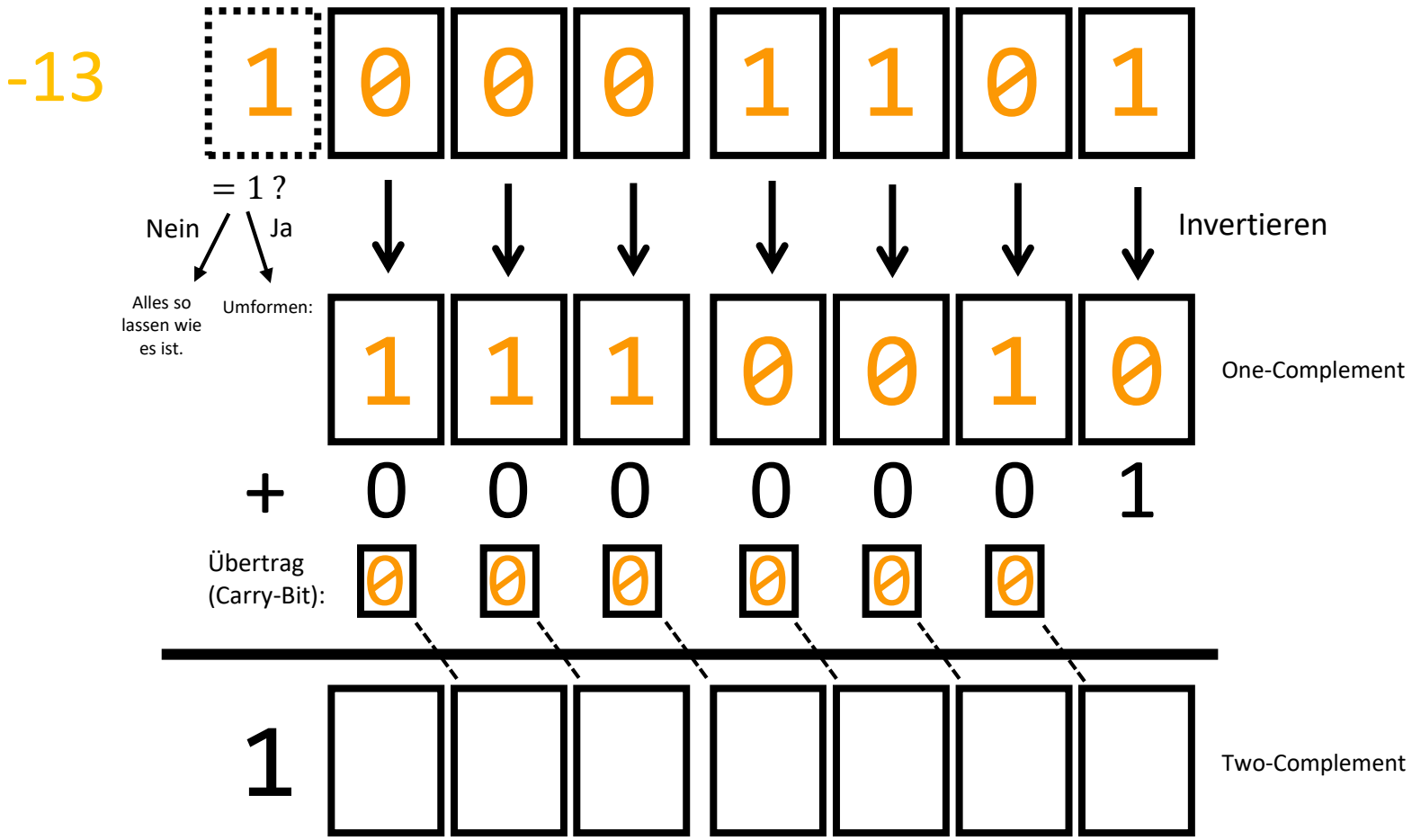
Beispiel einer Zweierkomplementendarstellung



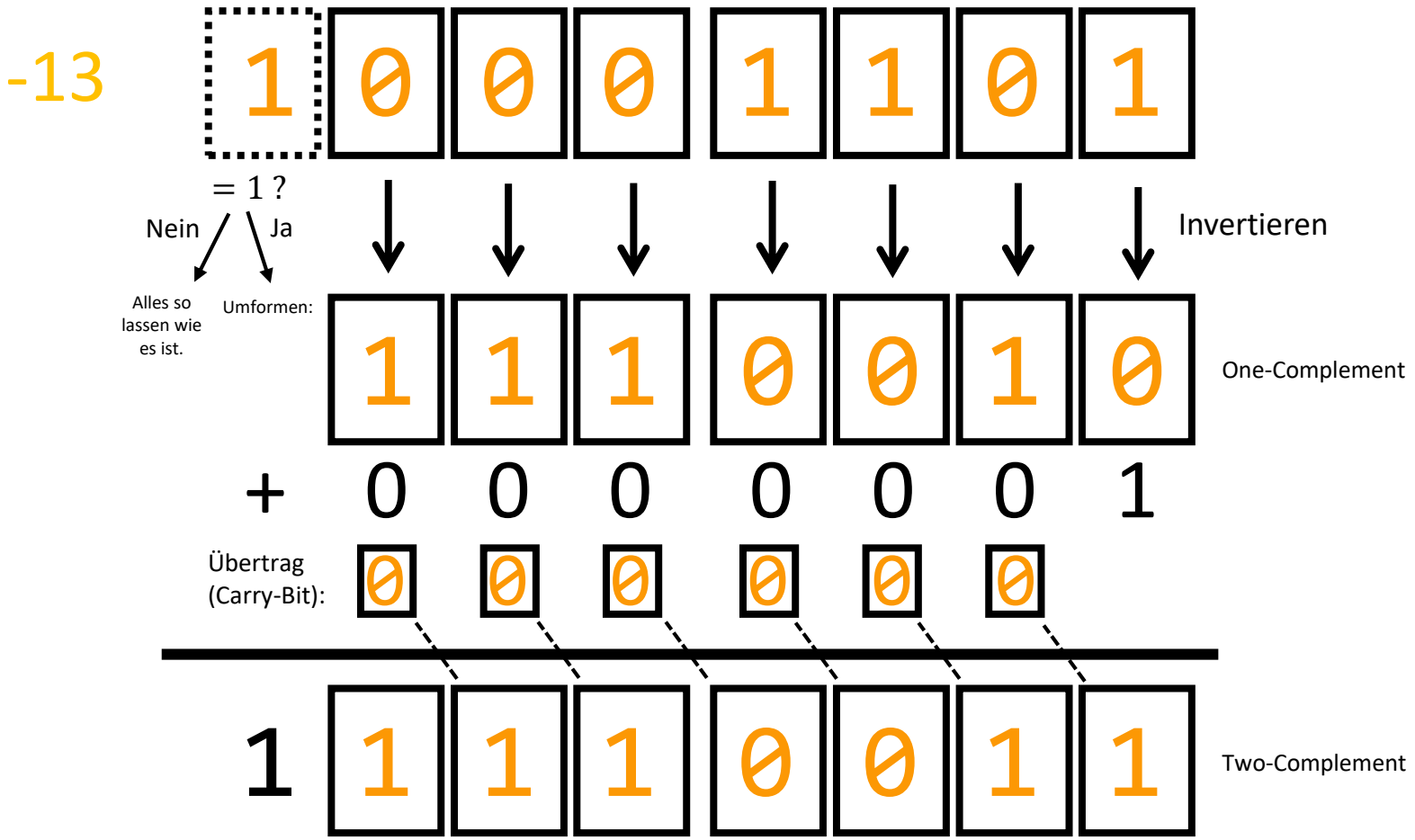
Beispiel einer Zweierkomplementendarstellung



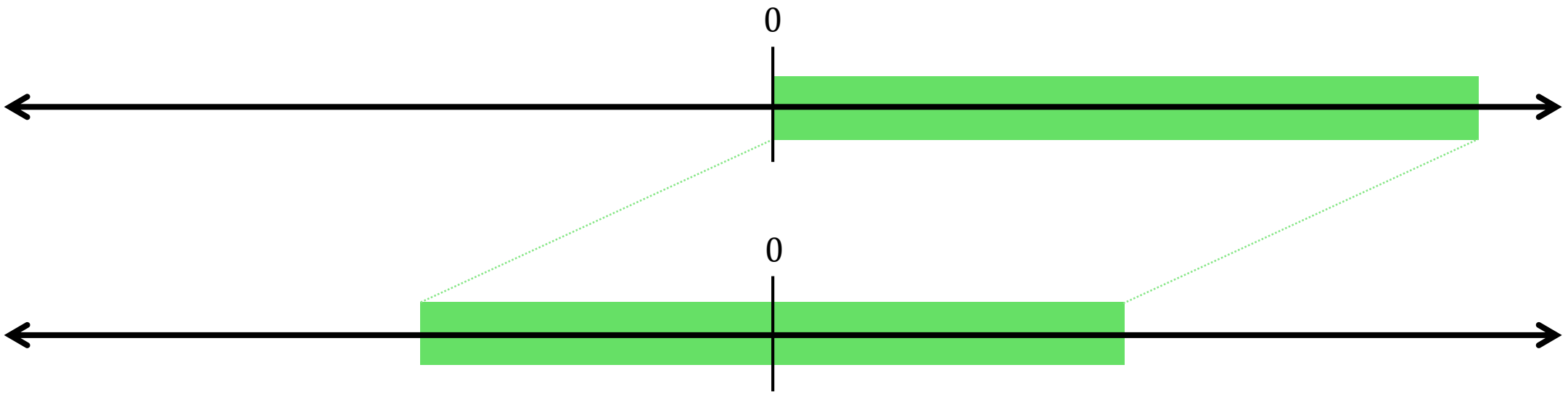
Beispiel einer Zweierkomplementendarstellung



Beispiel einer Zweierkomplementendarstellung

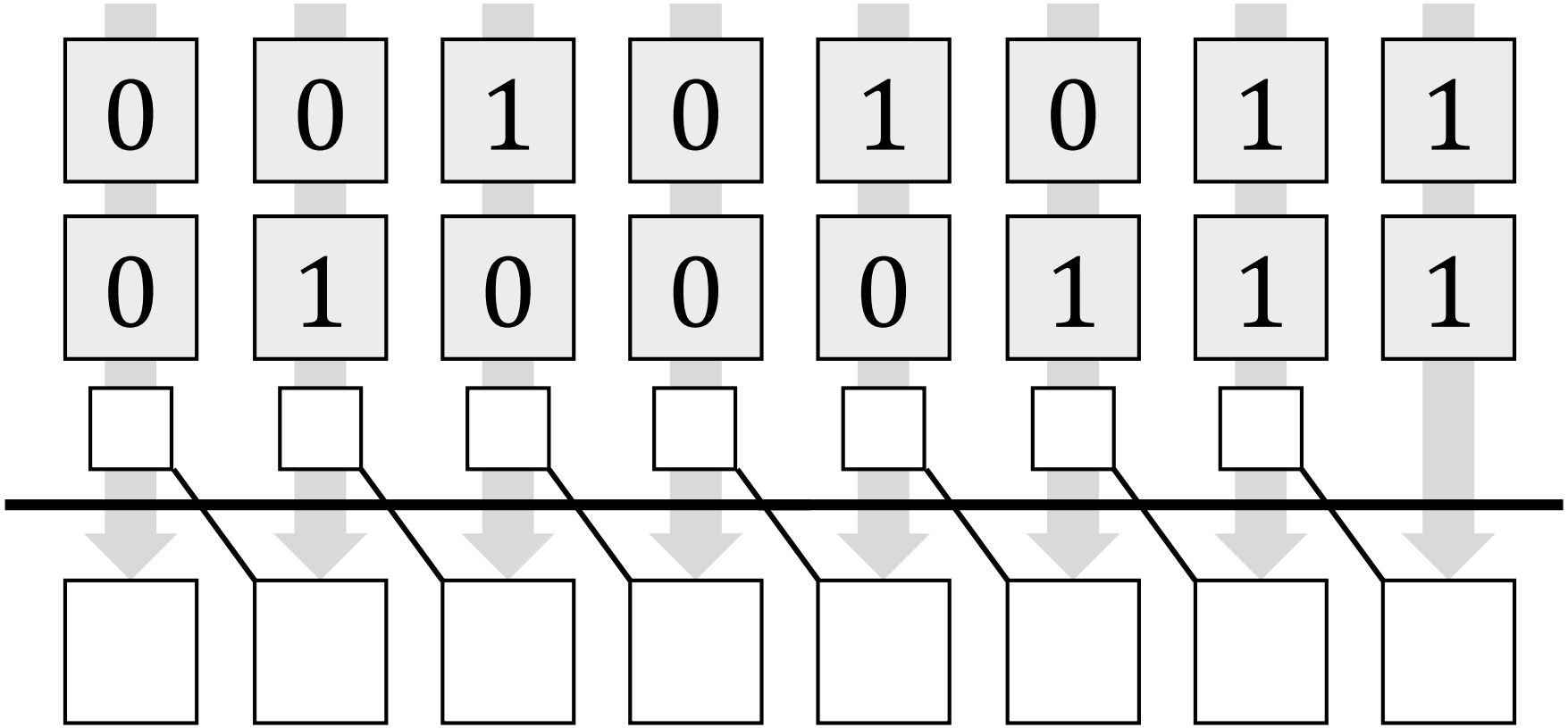


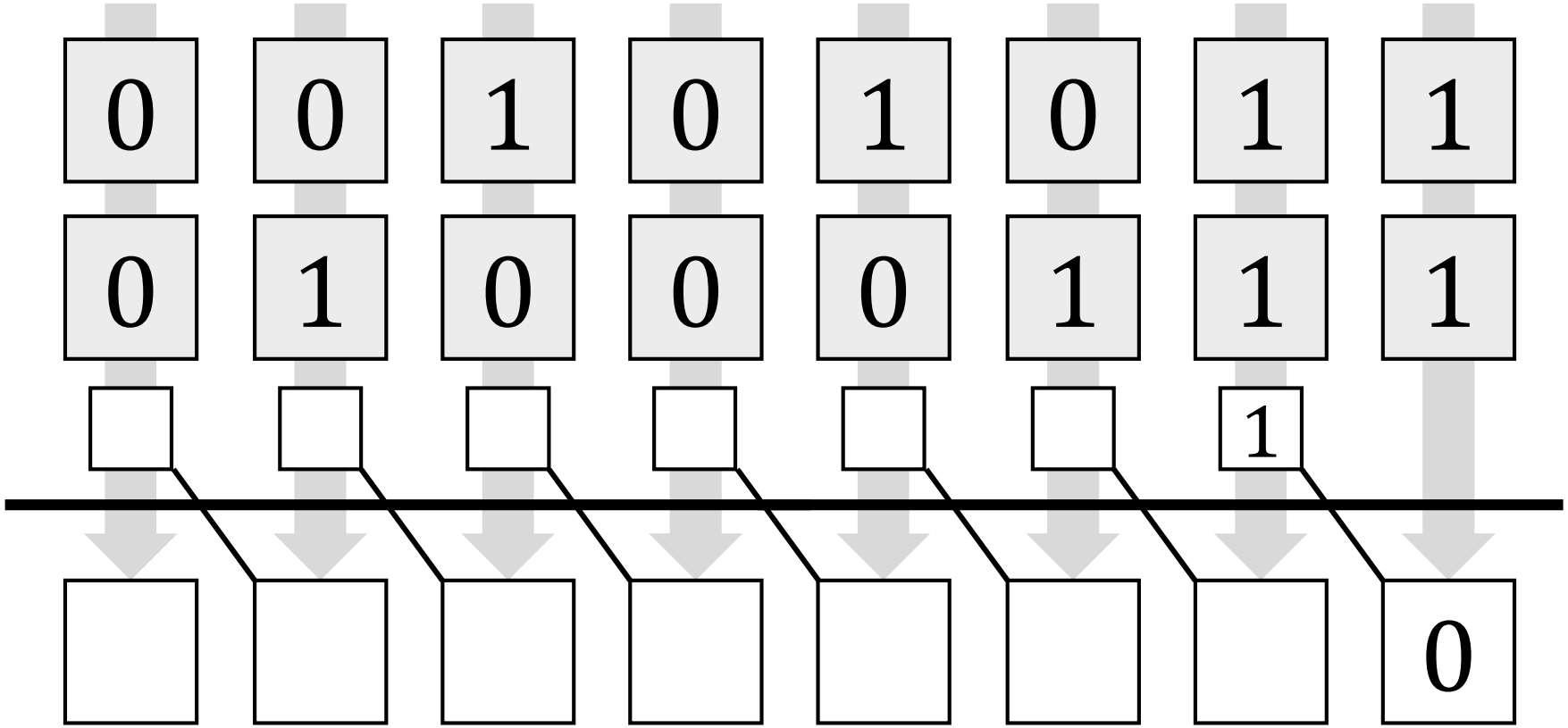
Wertebereich



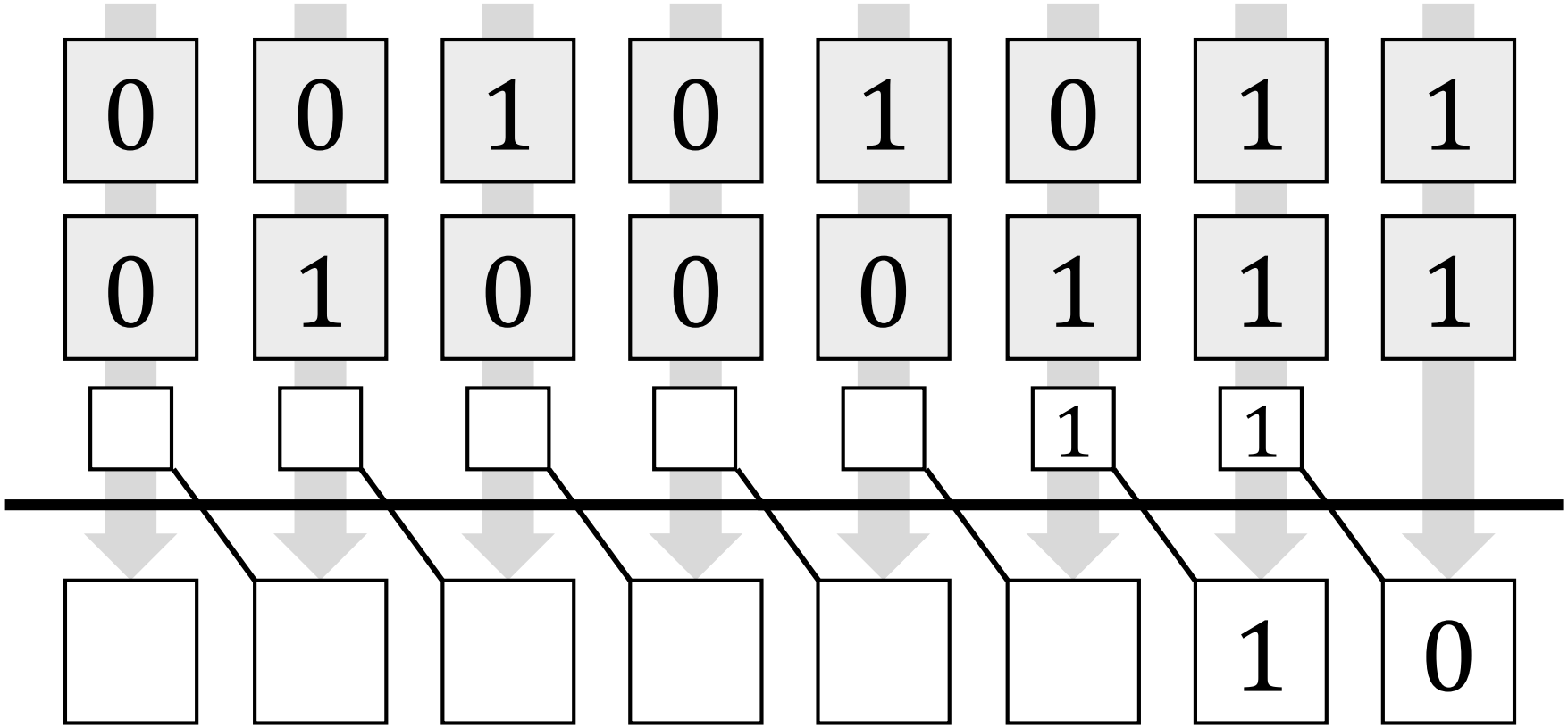
$$\begin{array}{rr} & 0 \\ + & 0 \\ \hline = & 0 \end{array} \qquad \begin{array}{rr} & 1 \\ + & 0 \\ \hline = & 1 \end{array} \qquad \begin{array}{rr} & 0 \\ + & 1 \\ \hline = & 1 \end{array} \qquad \begin{array}{rr} & 1 \\ + & 1 \\ \hline = & 0 \end{array}$$

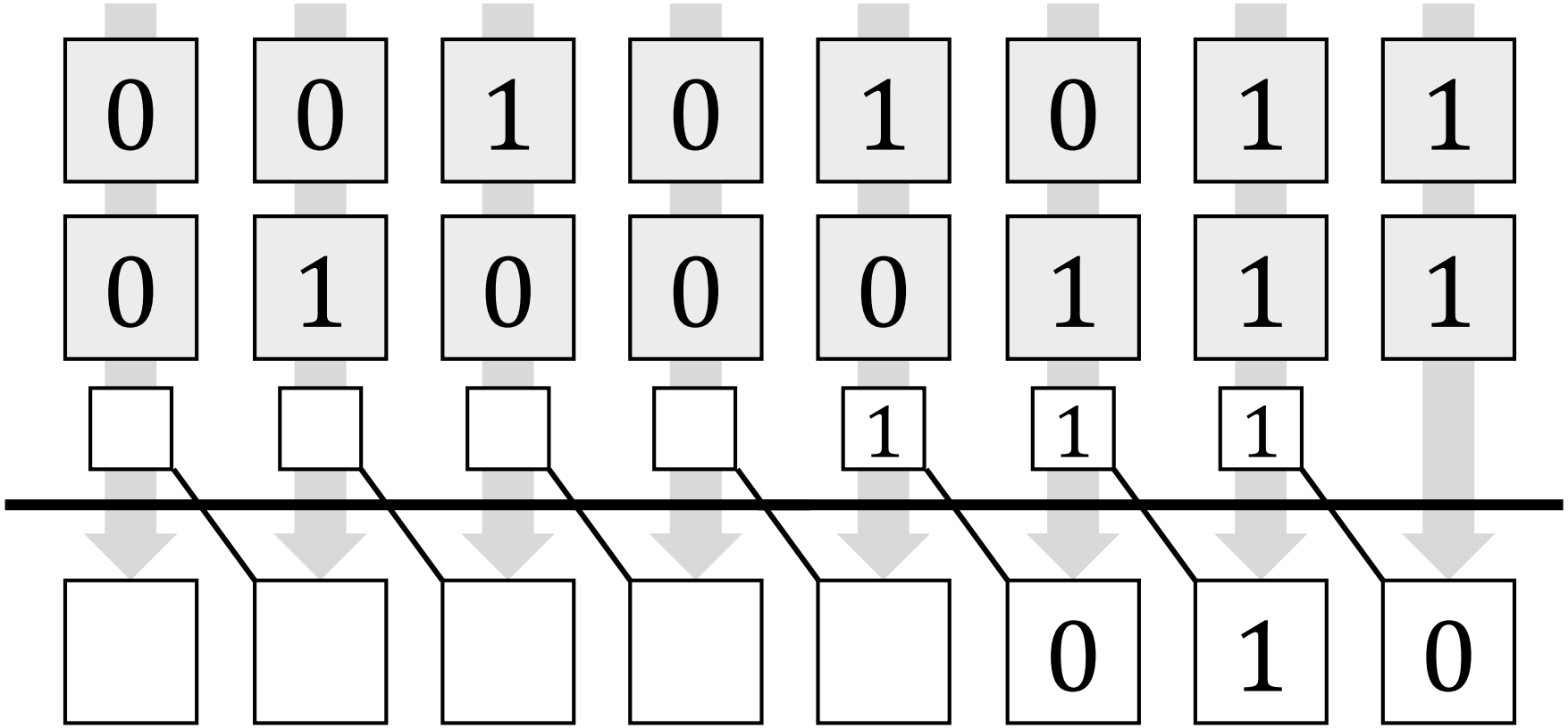
X	Y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

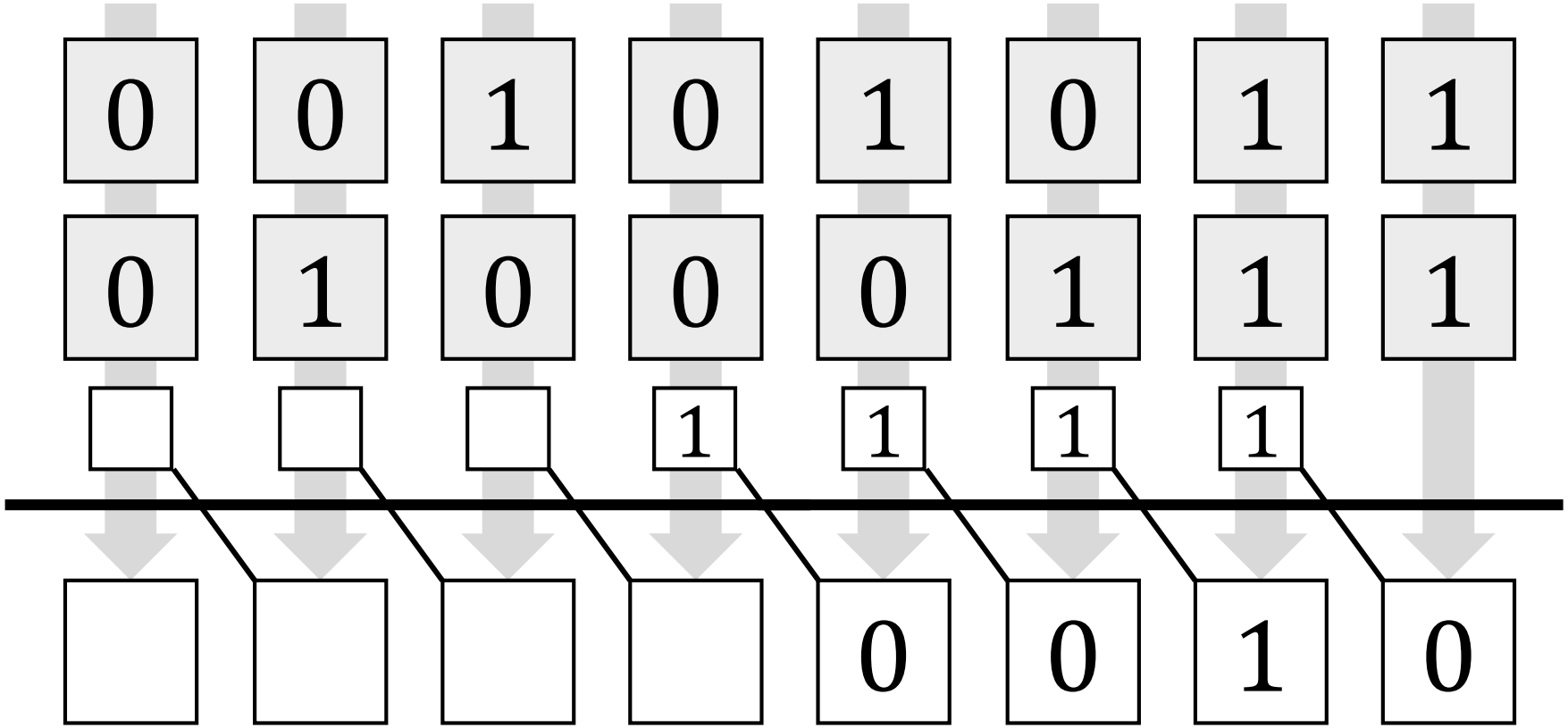

0 0 1 0 1 0 1 1
0 1 0 0 0 1 1 1

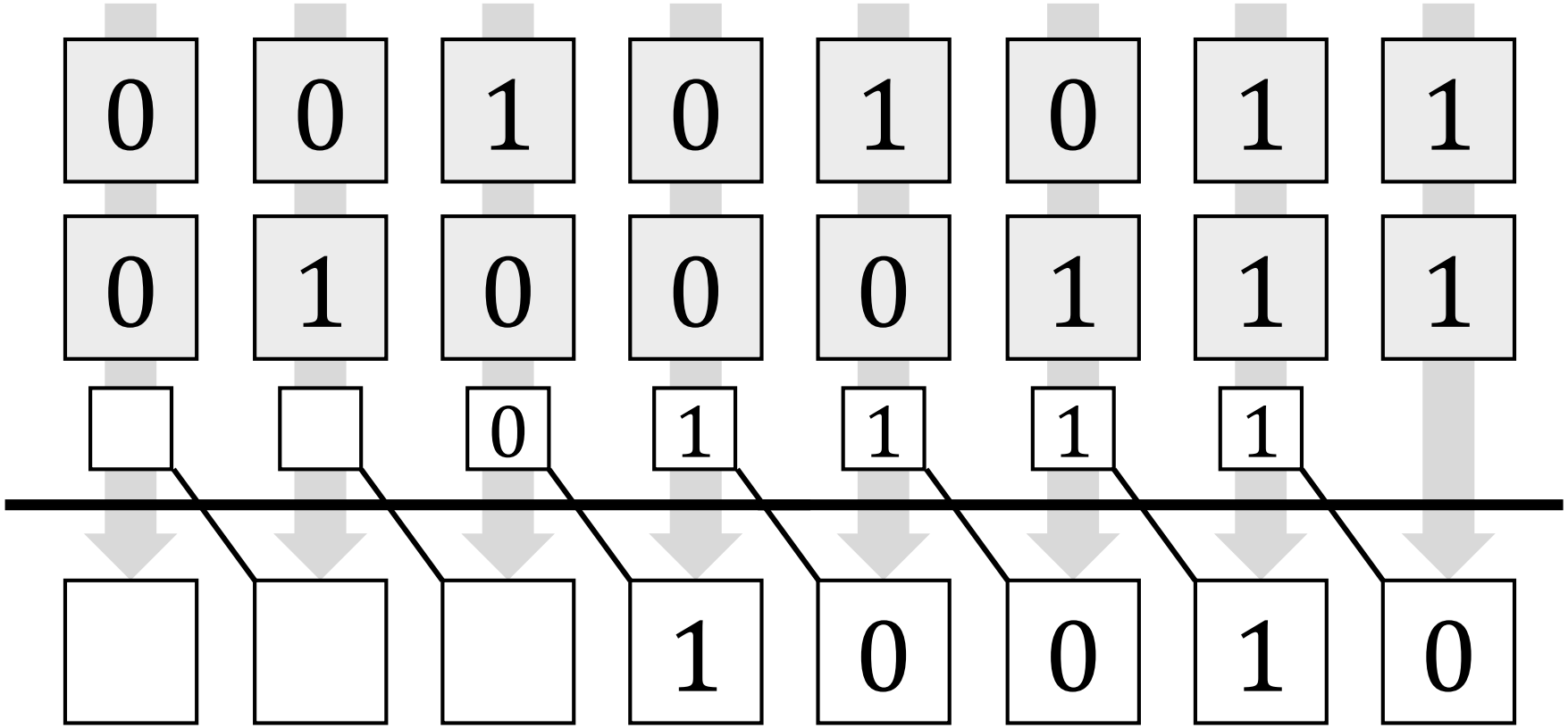

0 0 1 0 1 0 1 1
0 1 0 0 0 1 1 1
1
0

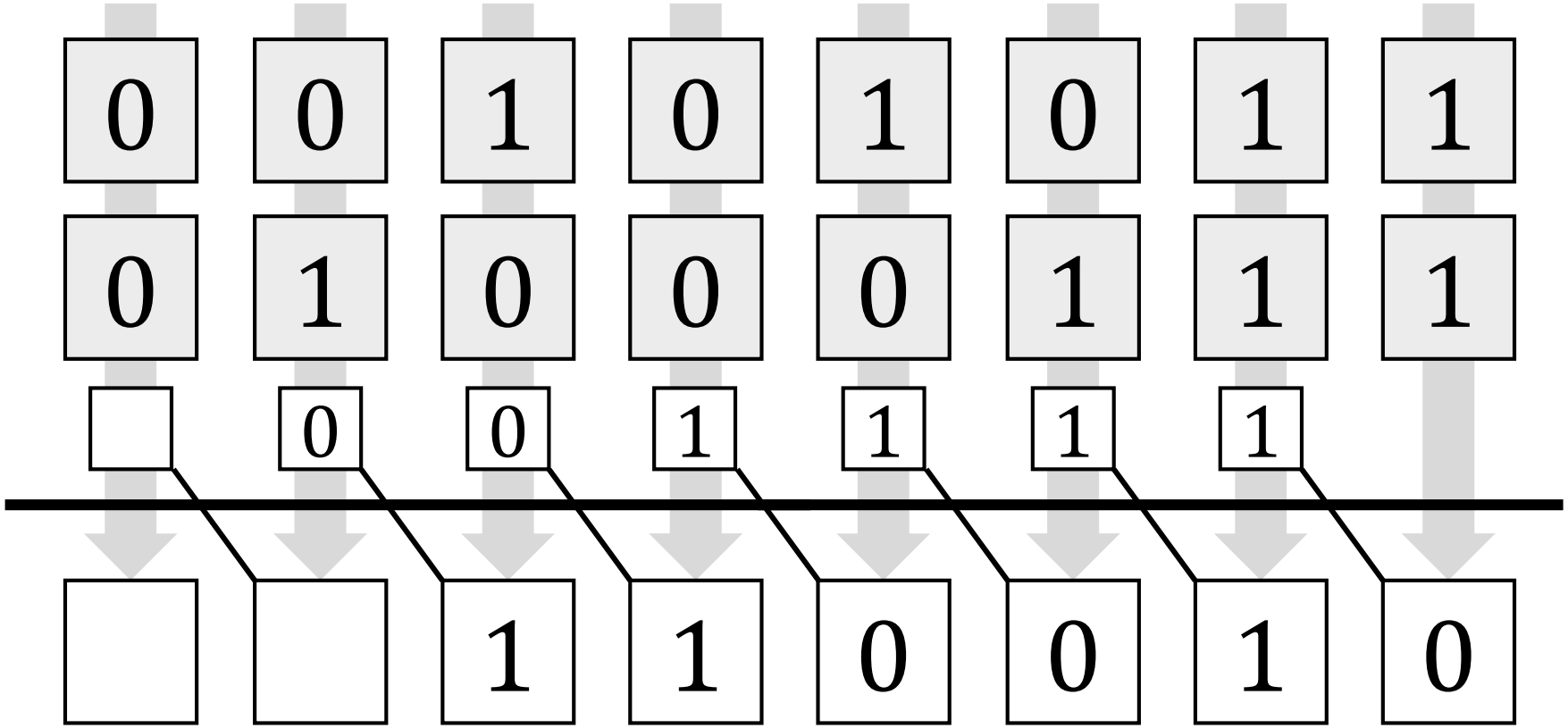
Aufgabe 2c



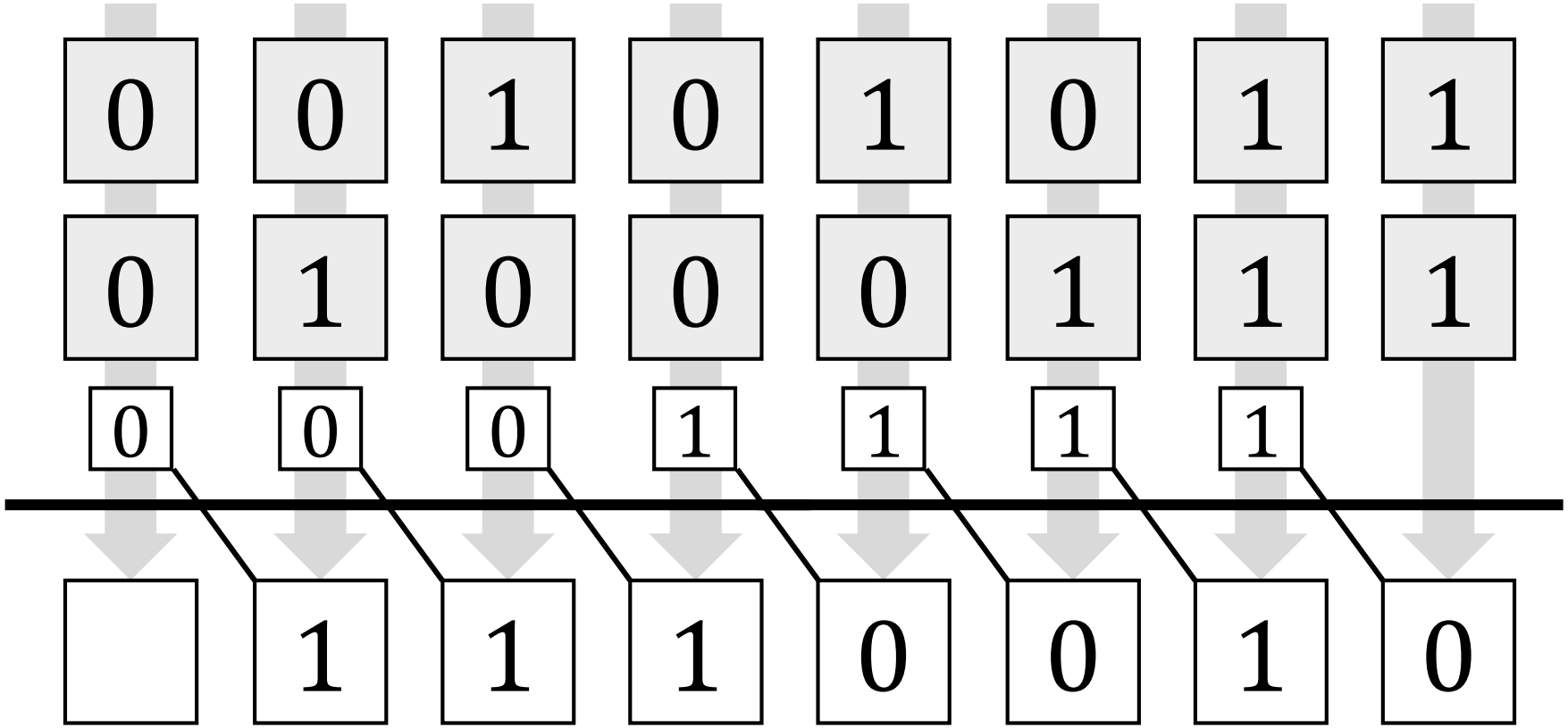

0 0 1 0 1 0 1 1
0 1 0 0 0 1 1 1
1 1 1
0 1 0

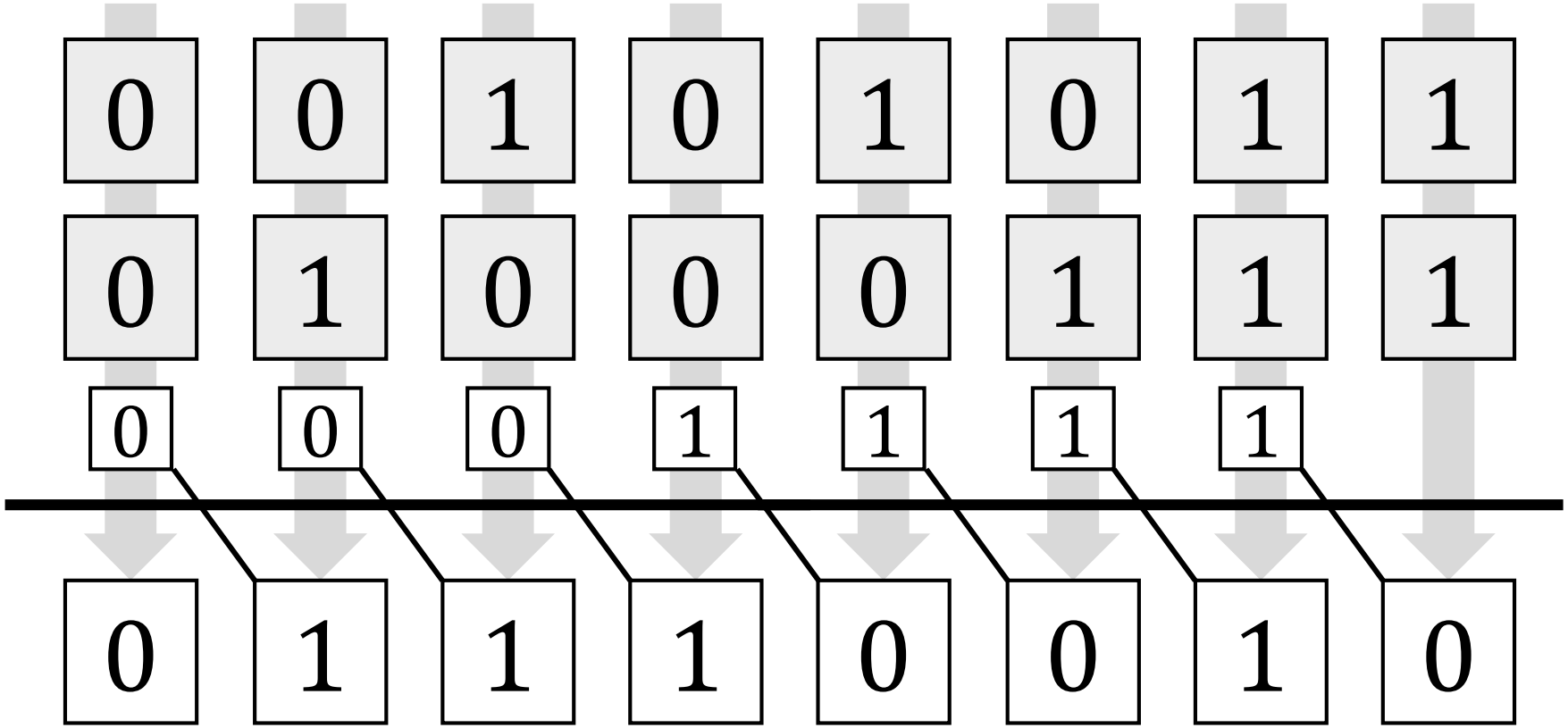

0 0 1 0 1 0 1 1
0 1 0 0 0 1 1 1
1 1 1 1
0 0 1 0


0 0 1 0 1 0 1 1
0 1 0 0 0 1 1 1
0 1 1 1 1
1 0 0 1 0


0 0 1 0 1 0 1 1
0 1 0 0 0 1 1 1
0 0 1 1 1 1
1 1 0 0 1 0

Aufgabe 2c




0 0 1 0 1 0 1 1
0 1 0 0 0 1 1 1
0 0 0 1 1 1 1
0 1 1 1 0 0 1 0

- Arithmetisch entspricht das Verschieben nach links einer Multiplikation mit 2
- … gdw. Zahl kleiner als 2^{n-1} ist
- Das Verschieben nach rechts entspricht arithmetisch einer ganzzahligen Division mit 2
- … gdw. Zahl teilbar durch 2 ist

```
w <<= 1
```

```
w >>= 1
```

- Arithmetisch entspricht das Verschieben nach links einer Multiplikation mit 2^k
- … gdw. Zahl kleiner als 2^{n-k} ist
- Das Verschieben nach rechts entspricht arithmetisch einer ganzzahligen Division mit 2^k
- … gdw. Zahl teilbar durch 2^k ist

```
w <<= k
```

```
w >>= k
```

Wann ist eine Binärzahl durch 2^k teilbar?

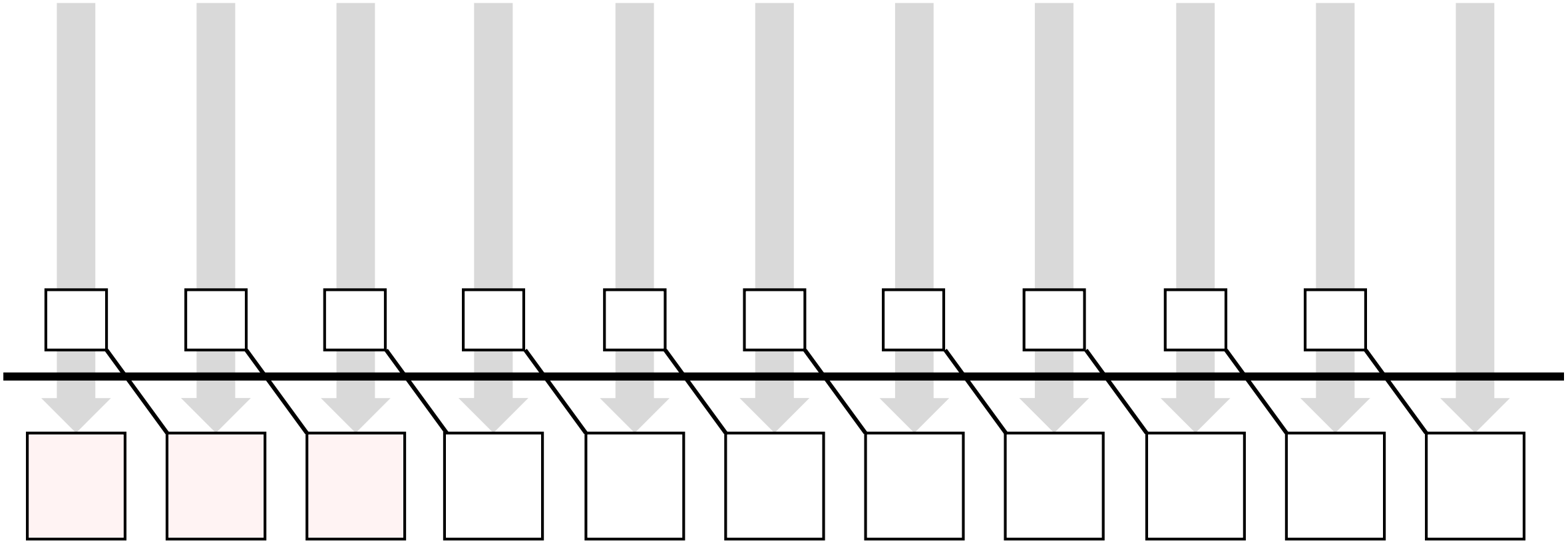
Wann ist eine Binärzahl durch 2^k teilbar?

- Teilbarkeit gilt genau dann, wenn die k am wenigsten signifikanten Bits gleich Null sind
- Beispielsweise ist die Binärzahl immer dann durch 4 teilbar, wenn die letzten beiden Bits gleich Null sind

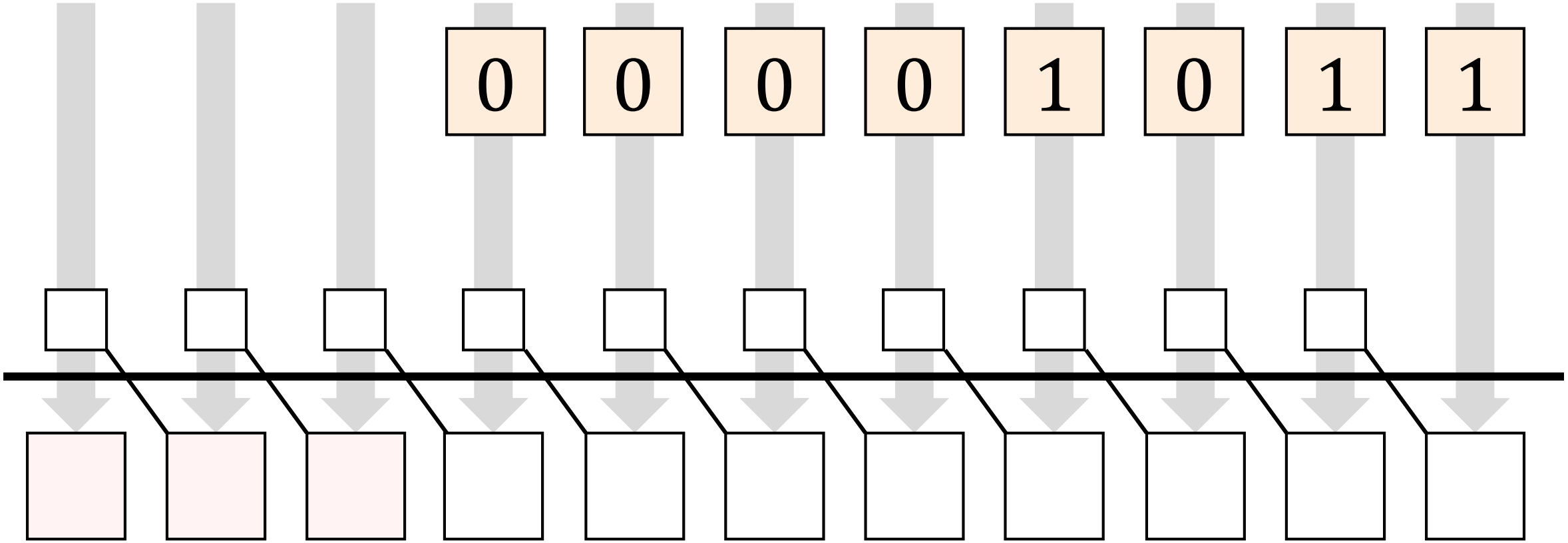
$$a \cdot b = \left[\sum_{j=0}^{n-1} a_k 2^k\right] \cdot \left[\sum_{k=0}^{n-1} b_k 2^k\right] = \sum_{j=0}^{n-1} \left[a_k 2^k \sum_{k=0}^{n-1} b_k 2^k\right] = \sum_{k=0}^{n-1} [a_k (b<<k)]$$

```
mul(a, b)                          result := 0
1  for (i := 0; i < n; ++i) :
2     if (b &- 1) : result += a
3     b >>= 1
4     a <<= 1
5  return a
```

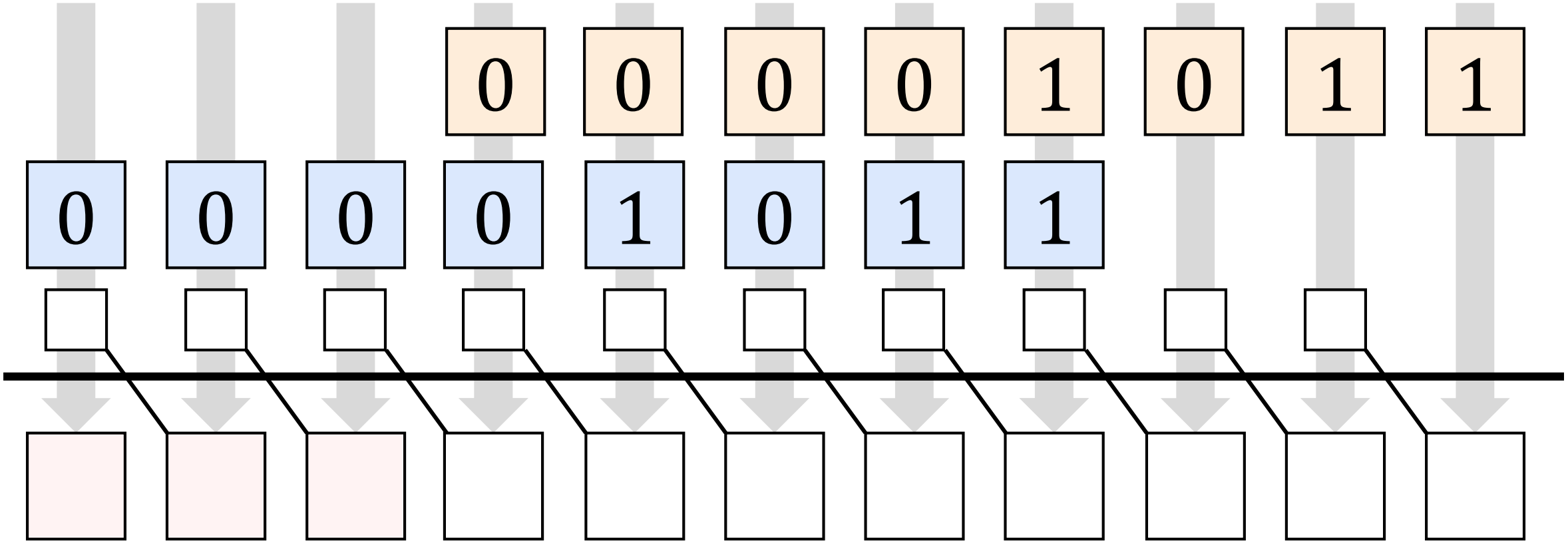
$$00001011 \cdot 00001001$$



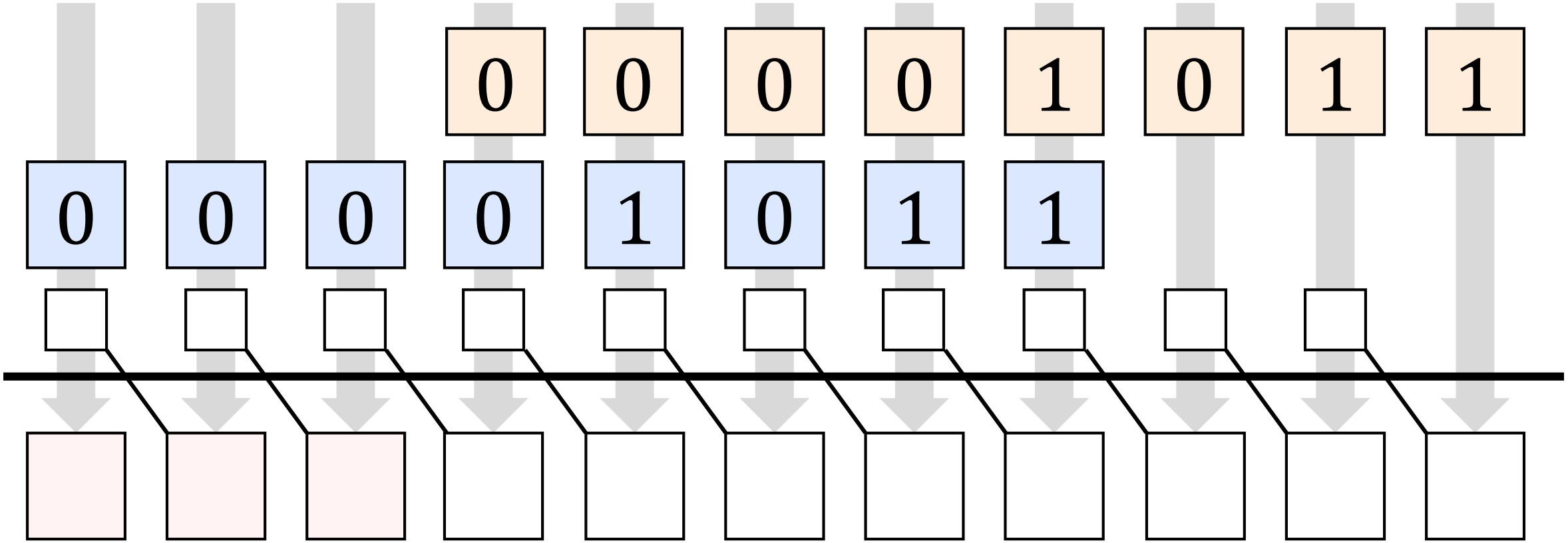
$$00001011 \cdot 00001001$$



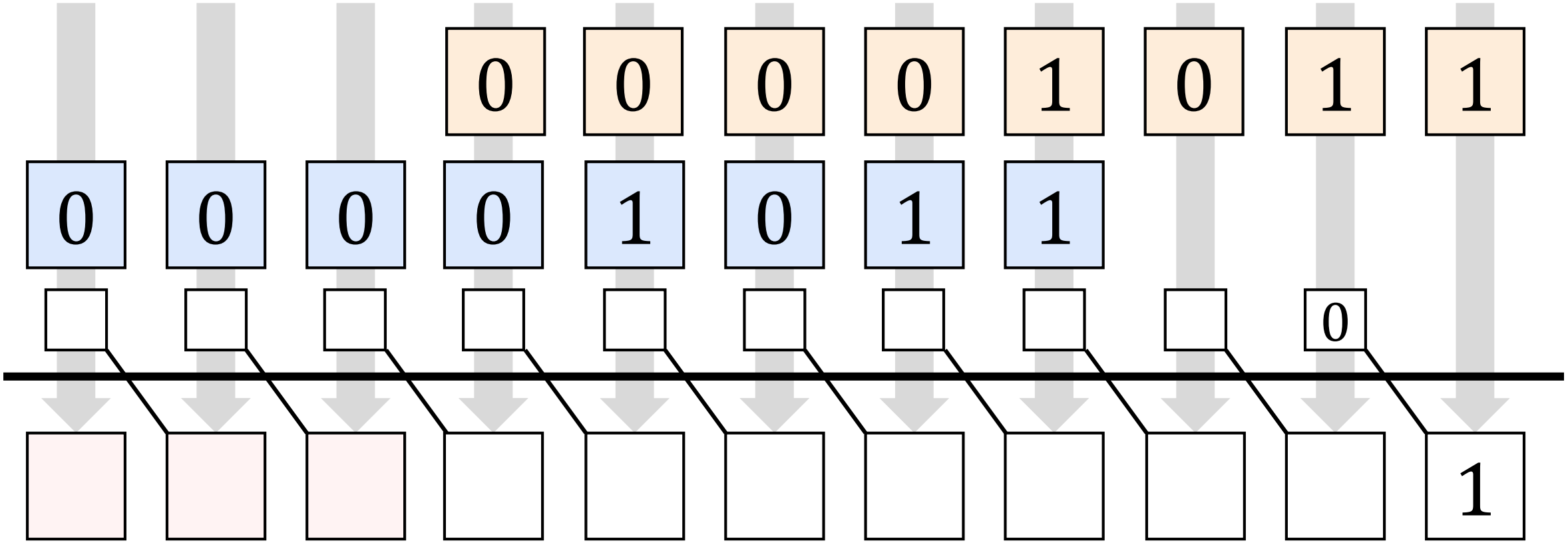
$$00001011 \cdot 00001001$$



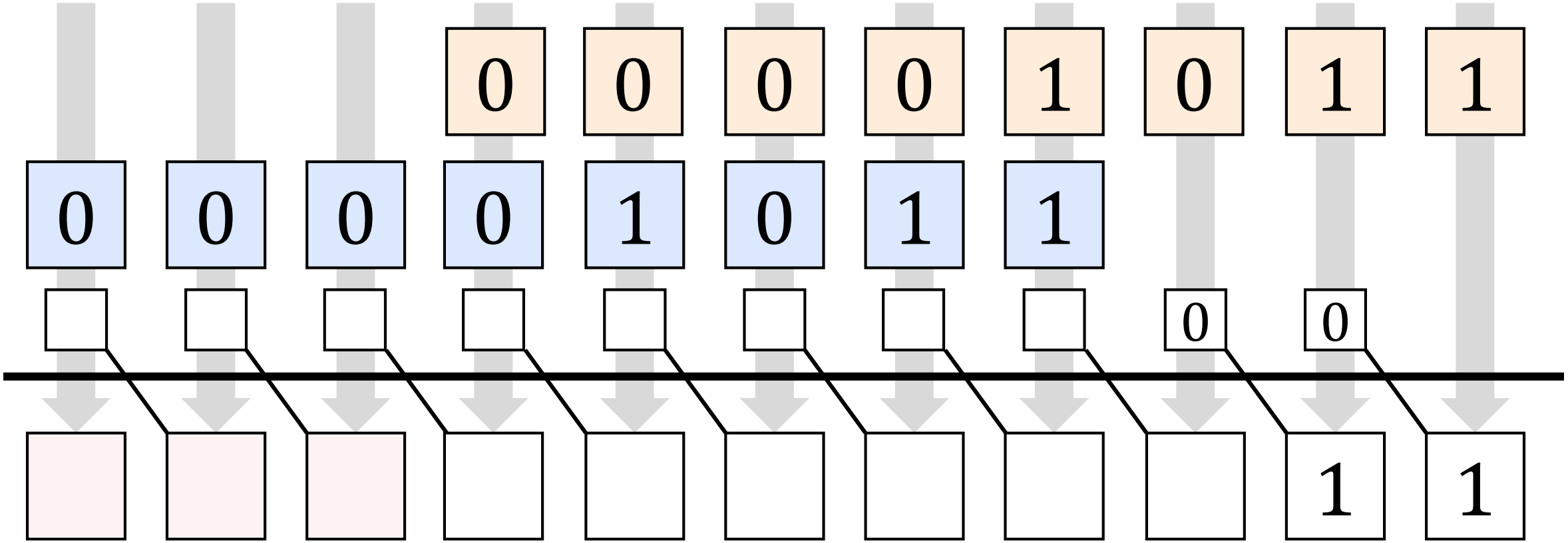
$$00001011 \cdot 00001001$$



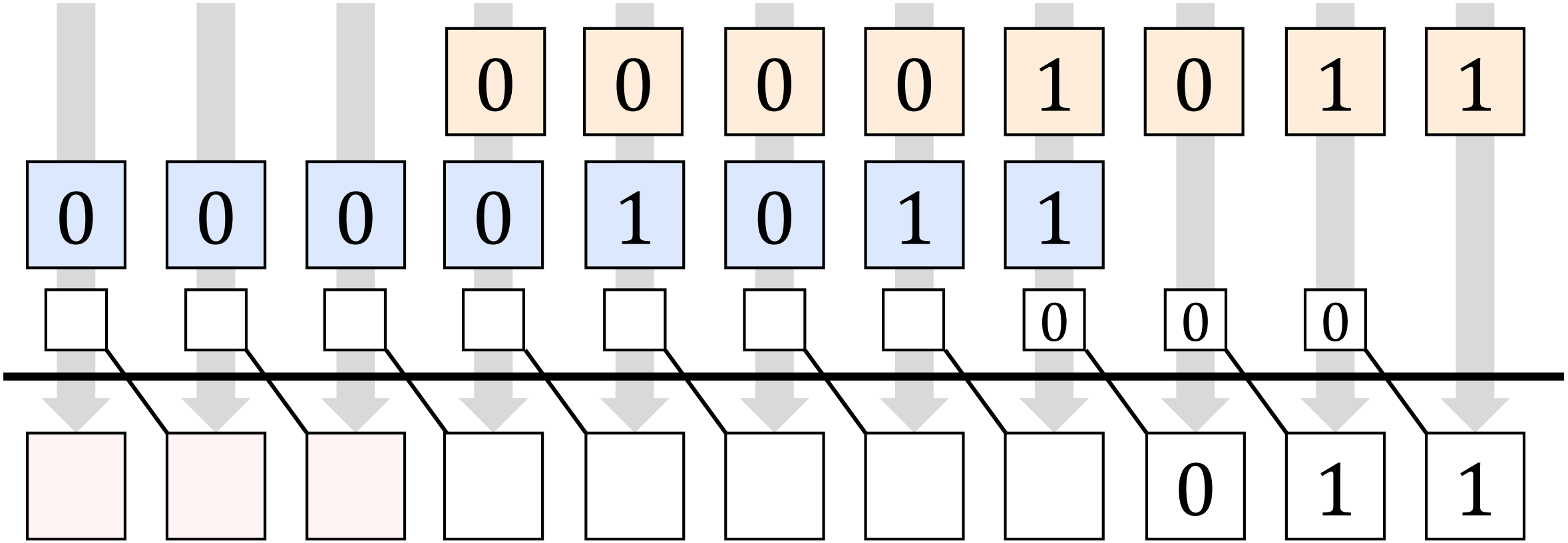
$$00001011 \cdot 00001001$$



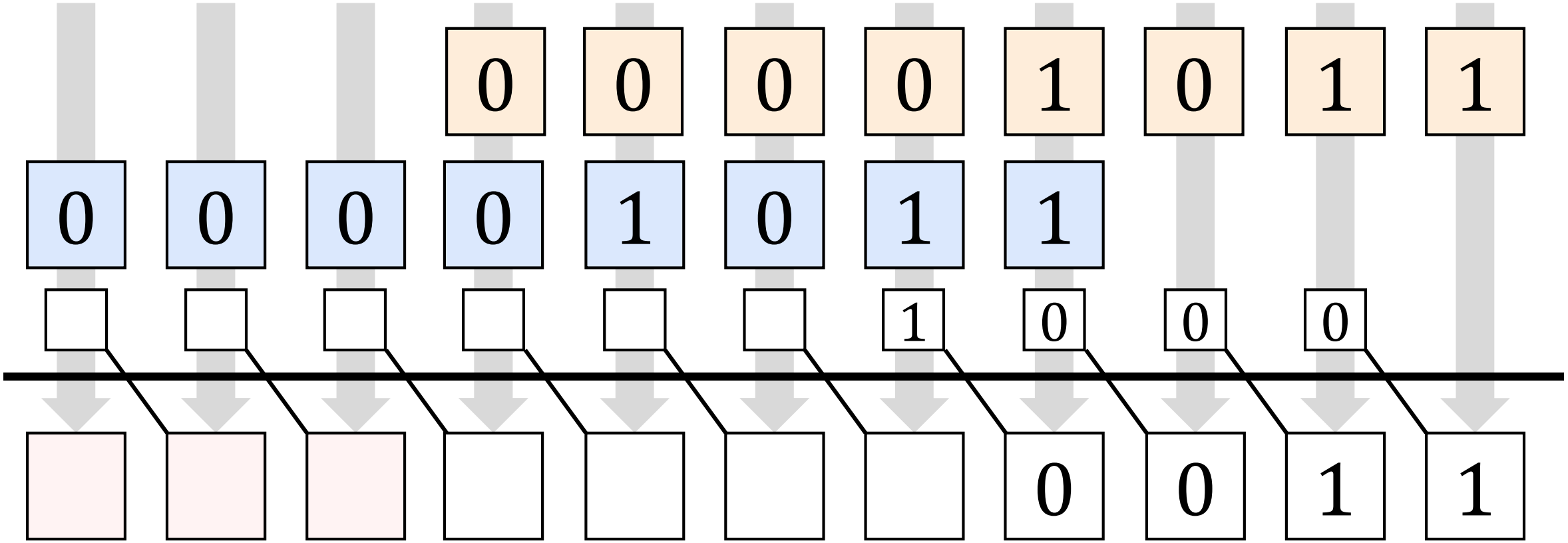
$$00001011 \cdot 00001001$$



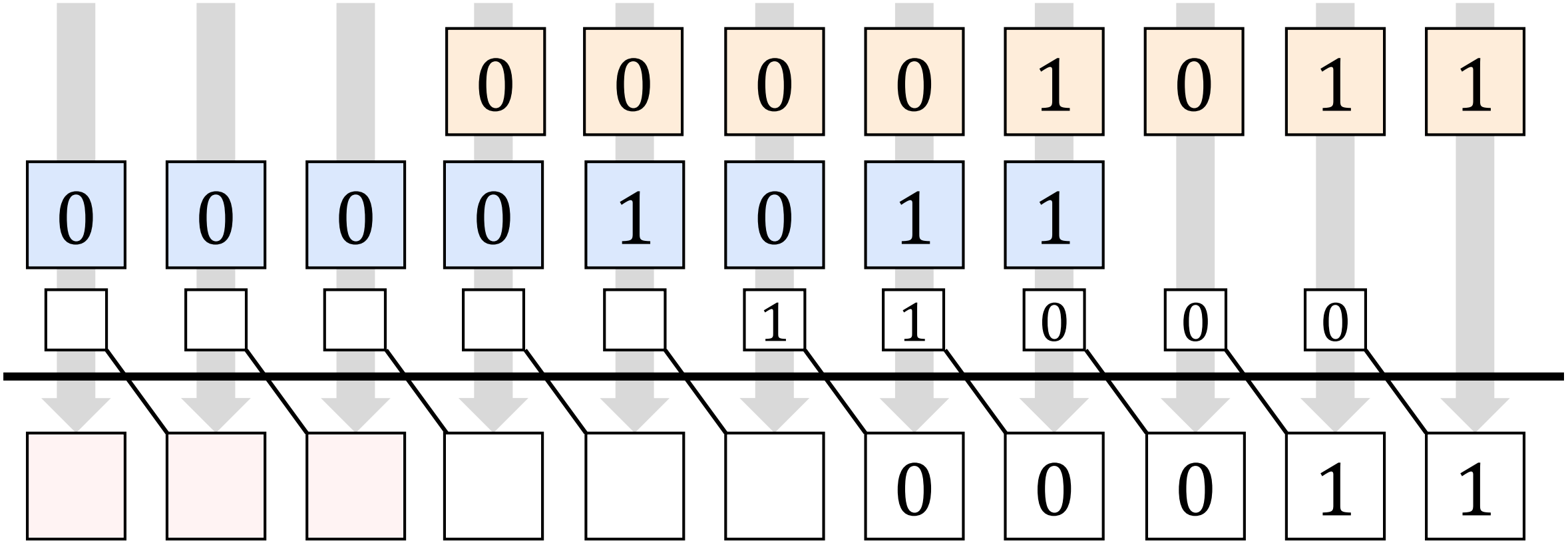
$00001011 \cdot 00001001$



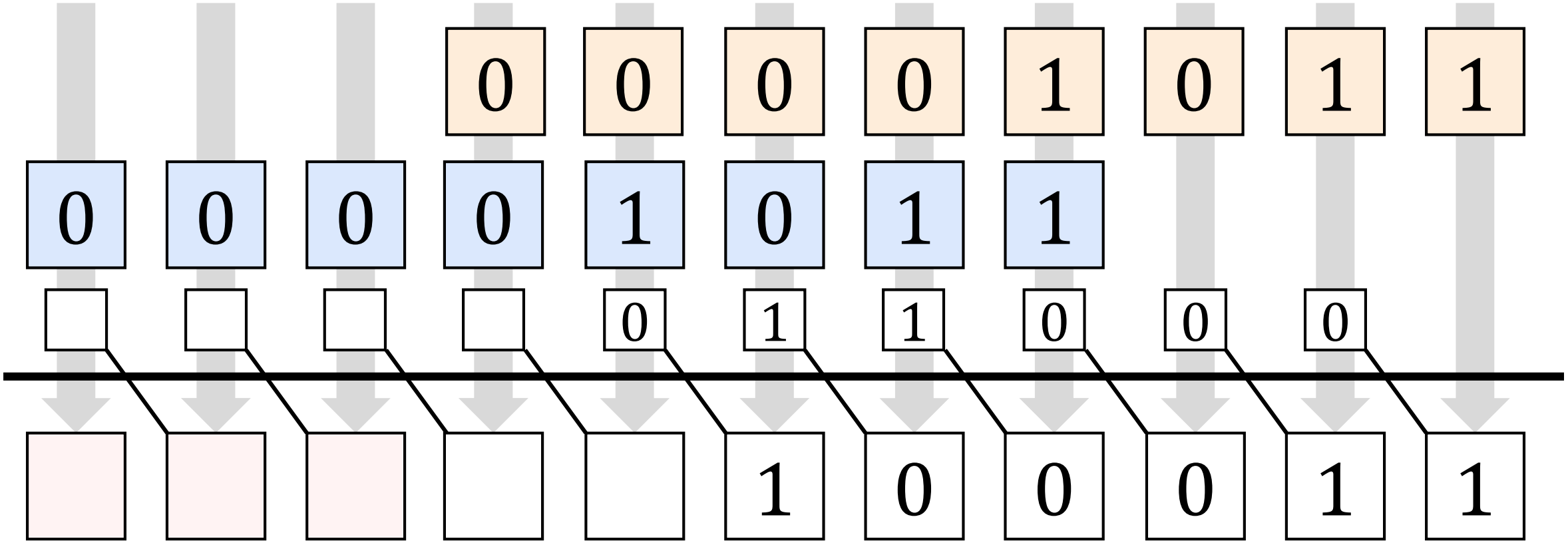
$$00001011 \cdot 00001001$$



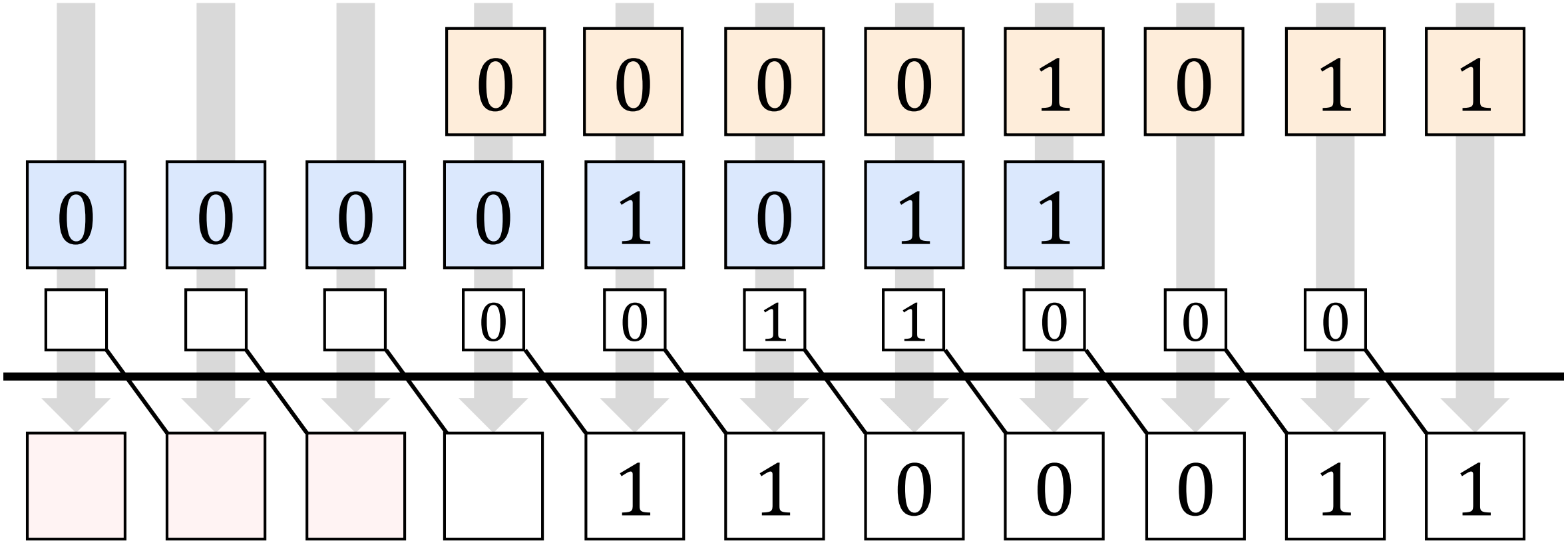
$$00001011 \cdot 00001001$$



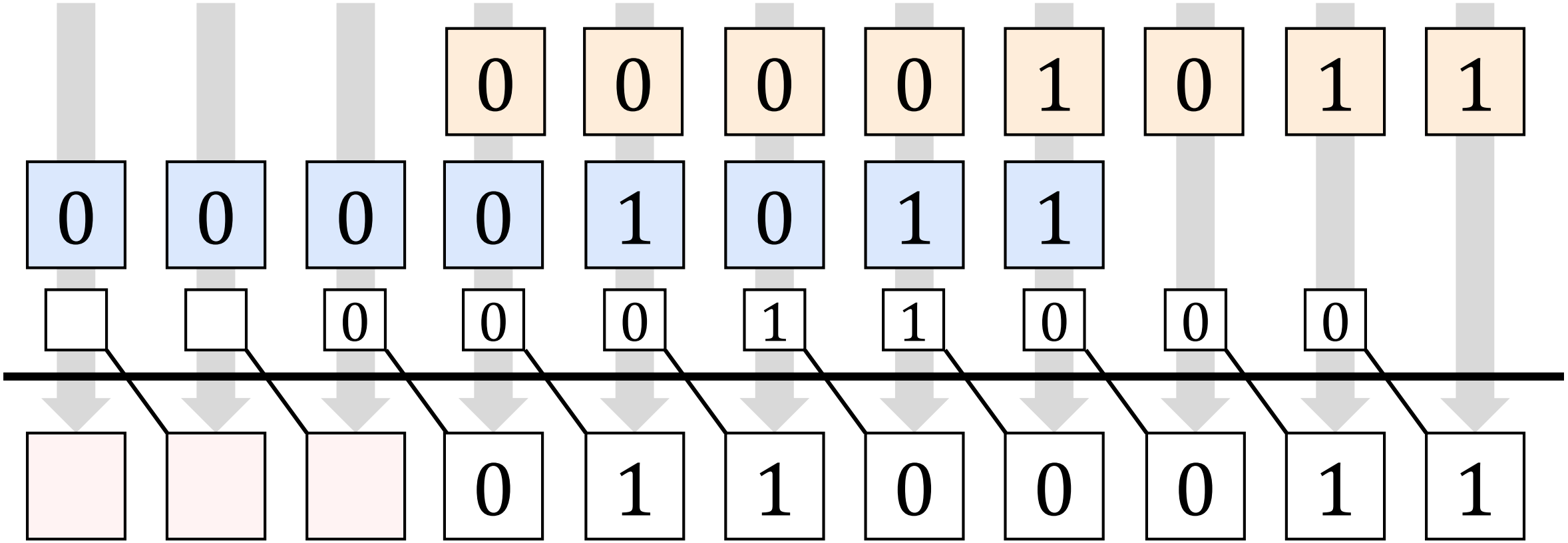
$00001011 \cdot 00001001$



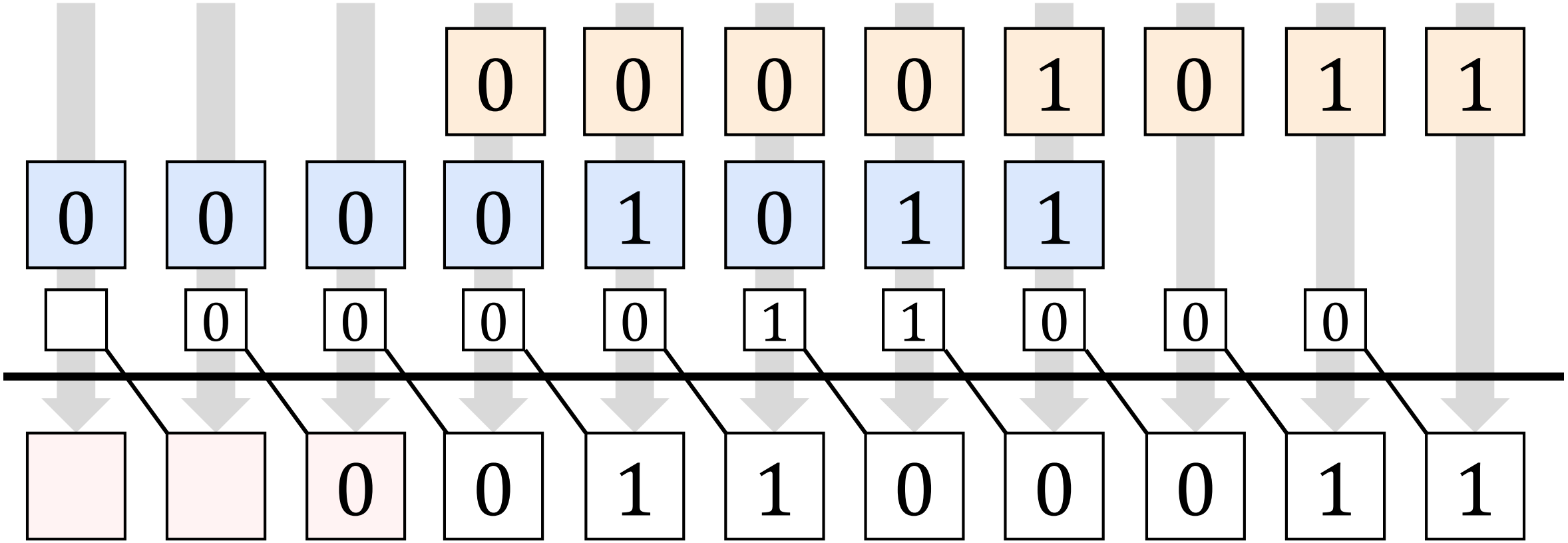
$00001011 \cdot 00001001$



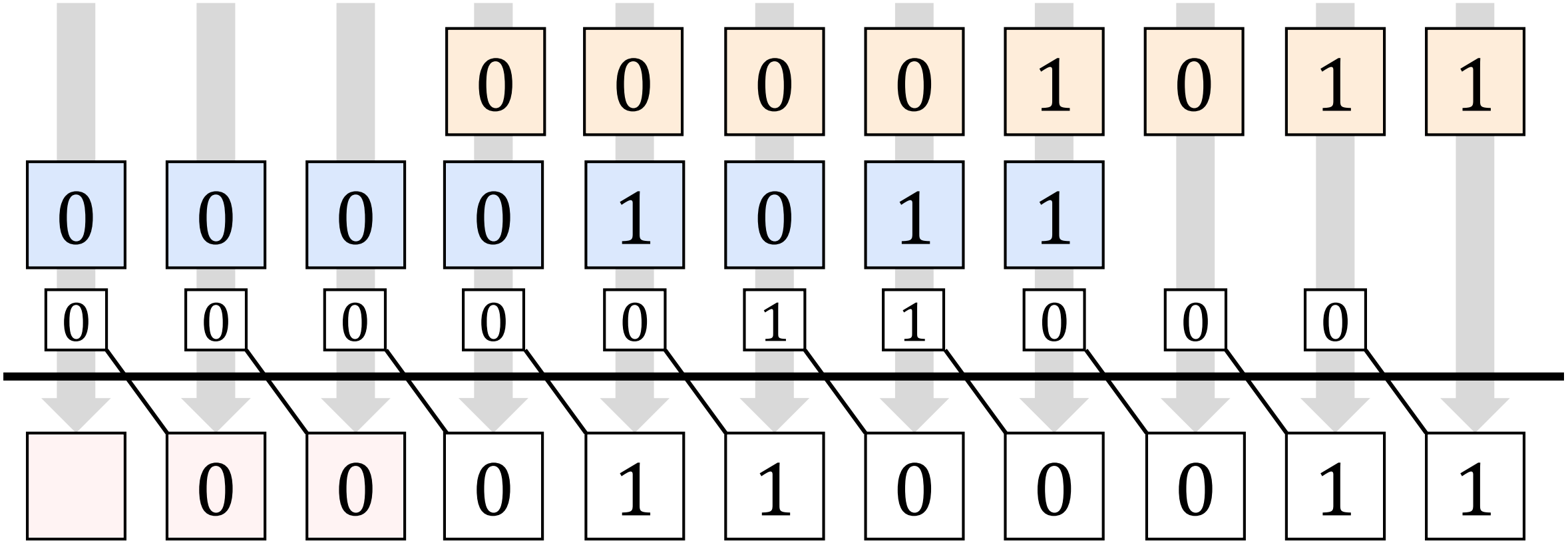
$$00001011 \cdot 00001001$$



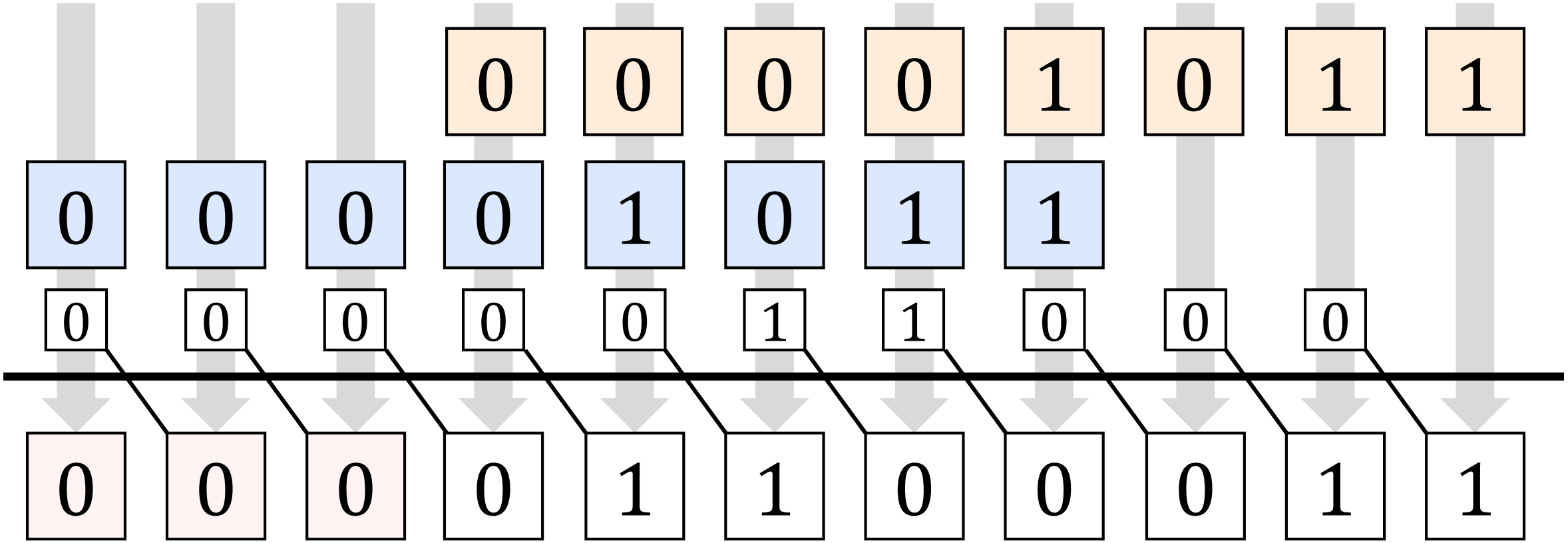
$00001011 \cdot 00001001$



$$00001011 \cdot 00001001$$



$00001011 \cdot 00001001$



```
div(a, b)                          q := 0, r := a
1  for (i := n − 1; i ≥ 0; --i) :
2    if (r ≥ (b << i)) :
3      r := -(b << i) + r
4      q := (1 << i) | q
5  return (q, r)
```

Auch die Division kann als Add-Shift-Algorithmus umgesetzt werden.

$$10101 \,/\, 11 =$$

$$10101 \,/\, 11 = 1$$
$$-1100$$

$$
\begin{array}{l}
10101 \,/\, 11 = 1 \\
\quad -11\,00 \\
\hline
\quad =10\,01
\end{array}
$$

$$
\begin{array}{r}
10101 \,/\, 11 = 11 \\
-1100 \\
\hline
=1001 \\
-110
\end{array}
$$

$$
\begin{array}{rl}
10101 / 11 = 11 & \\
-1100 & \\
\hline
=1001 & \\
-110 & \\
\hline
=11 &
\end{array}
$$

$$
\begin{array}{rl}
10101 / 11 = 111 & \\
-1100 & \\
\hline
=1001 & \\
-110 & \\
\hline
=11 & \\
-11 &
\end{array}
$$

$$
\begin{array}{rl}
10101 / 11 = 111 & \\
-1100 & \\
\hline
=1001 & \\
-110 & \\
\hline
=11 & \\
-11 = 0 &
\end{array}
$$

Divisionsalgorithmus (Beispiel II)

$$11111100 \,/\, 1100 =$$

Divisionsalgorithmus (Beispiel II)

$$11111100 \,/\, 1100 = 1$$

$$\text{-}1100\,0000$$

$$11111100 \,/\, 1100 = 1$$

$$\underline{-1100}0000$$

$$=11\,1100$$

Divisionsalgorithmus (Beispiel II)

$$11111100 \,/\, 1100 = 101$$

$$\begin{array}{r} -1100\,0000 \\ \hline =11\,1100 \\ -1100\,00 \end{array}$$

Divisionsalgorithmus (Beispiel II)

$$11111100 \,/\, 1100 = 101$$

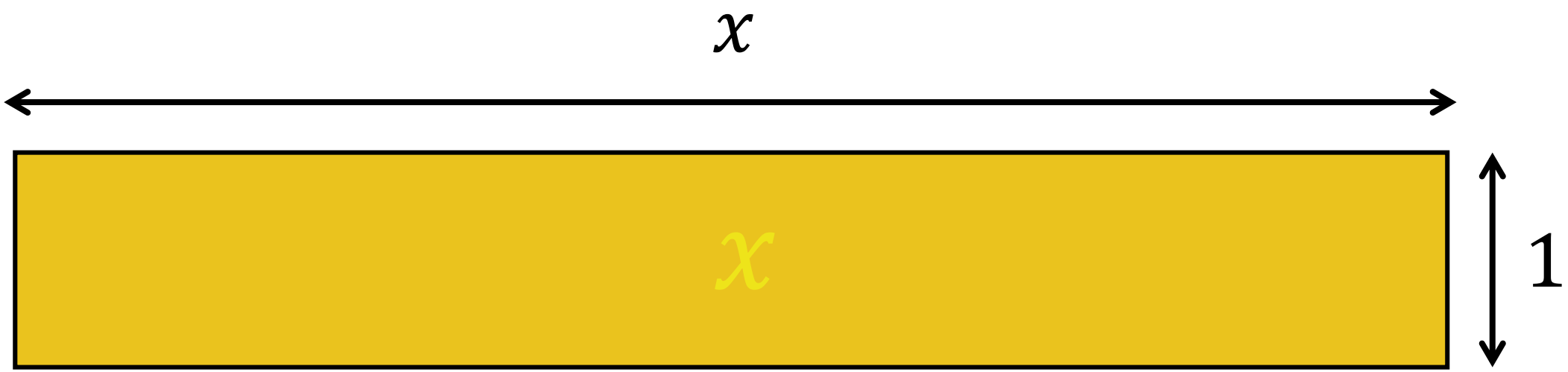
$$\begin{array}{r} -1100\,0000 \\ \hline =11\,1100 \\ -1100\,00 \\ \hline =11\,00 \end{array}$$

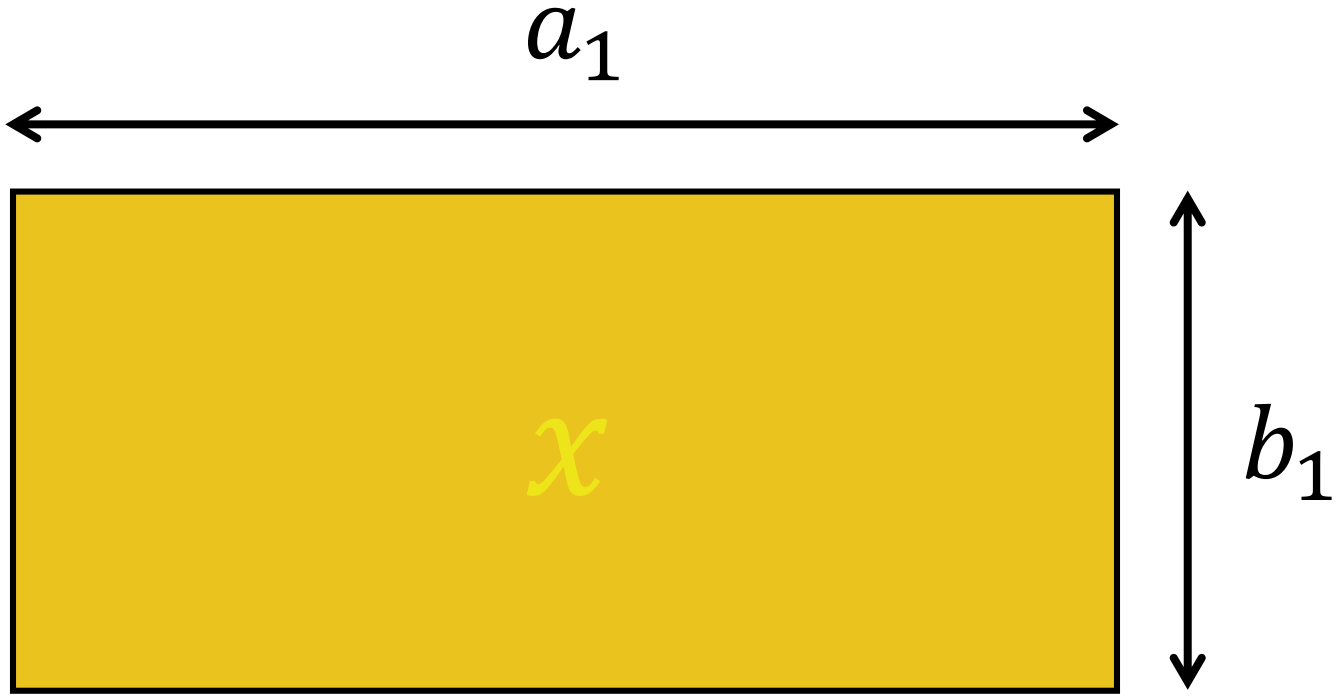
$$11111100 \,/\, 1100 = 10101$$

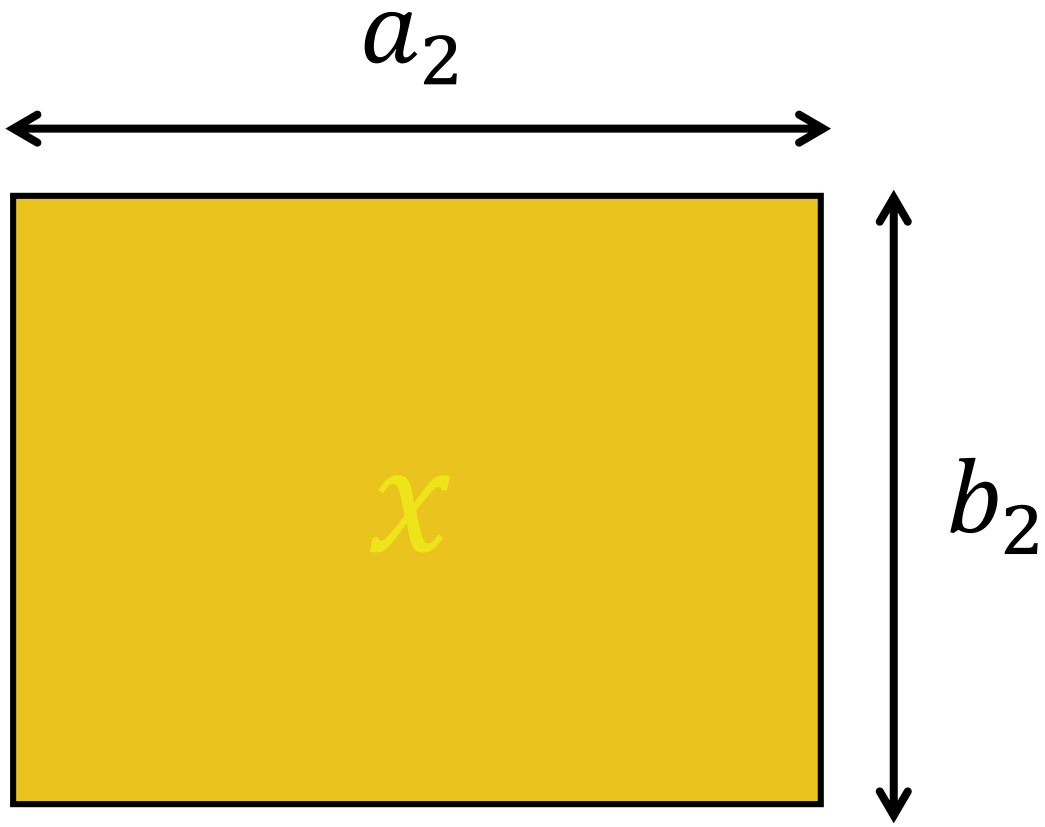
$$\begin{array}{r} -1100\,0000 \\ \hline =11\,1100 \\ -1100\,00 \\ \hline =11\,00 \\ -1100 \end{array}$$

$$11111100 \,/\, 1100 = 10101$$

$$\begin{array}{r} -1100\,0000 \\ \hline =\;\;11\,1100 \\ \quad -1100\,00 \\ \hline \quad\;\; =\;\;11\,00 \\ \quad\quad -1100 \\ \hline \quad\quad\;\; =\;\;0 \end{array}$$


x
x
1


a_1
x
b_1


a_2
x
b_2

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char* argv[]) {
    // Testwert
    double x = 9.0;

    if (argc > 1) {
        double eingegebeneZahl = atof(argv[1]); // atof konvertiert String in double
        if (eingegebeneZahl > 0) {
            x = eingegebeneZahl;
        }
    }

    // Startwert Heron-Verfahren
    double a = (x + 1) / 2.0;

    // Array zum Speichern der vier Iterationen
    double iteration[4];

    // Heron-Verfahren in 4 Iterationen
    for (int i = 0; i < 4; ++i) {
        a = 0.5 * (a + x / a);
        iteration[i] = a; // Speichern der aktuellen Iteration im Array
    }

    // Ergebnis und Iterationen ausgeben
    printf("Quadratwurzel von %f (nach 4 Iterationen): %f\n", x, a);
    printf("Zwischenwerte bei jeder Iteration:\n");
    for (int i = 0; i < 4; ++i) {
        printf("Iteration %d: %f\n", i + 1, iteration[i]);
    }

    return 0;
}
```

```
for (int i = 0; i < 4; ++i) {
    a = 0.5 * (a + x / a);
    iteration[i] = a;
}
```

Die Aufgabe kann auch ohne Kontrollstrukturen gelöst werden.
Mit den Schleifen und Verzweigungen werden wir uns im nächsten Tutorium befassen.