Übungen zur Grundlagen der Technischen Informatik

Übung 0 – Organisatorisches

Florian Frank

Friedrich-Alexander-Universität Erlangen-Nürnberg

Wintersemester 2018/19

Willkommen zu eurer GTI-Übung

Kontakt

Name: Florian Frank
5. Semester Informatik (Bachelor)
E-Mail: florian.ff.frank@fau.de
$\rightsquigarrow$ Bei Fragen, Kritik, Anregungen oder ähnlichem

Wo finde ich die Folien?

$\rightsquigarrow$ http://wwwcip.cs.fau.de/~yq53ykyr/GTI

Willkommen zu eurer GTI-Übung

Zeit und Ort

Montags

16:00 – 17:30 Uhr

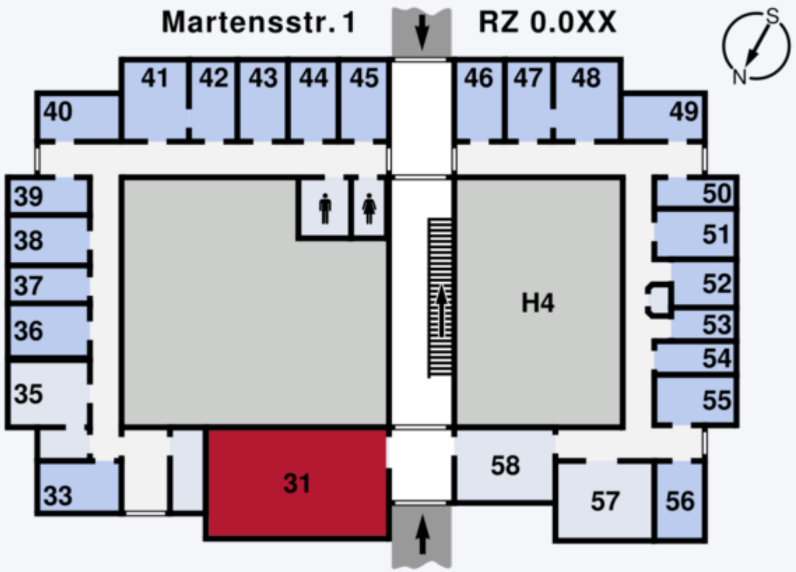
01.255-128 (1. Stock im Mathe Neubau)

Beginn der Übung kann sich um $\pm$ 5 Minuten verschieben ...

Willkommen zu eurer GTI-Übung

Zeit und Ort

Dienstags
10:15 – 11:30 Uhr
0.031-113 (rot eingefärbt)



Information und Materialien

Informationen und Materialien

Tretet dem StudOn-Kurs bei um Zugriff auf die Vorlesungsmaterialien, Übungsblätter, Musterlösungen, Praktikumsunterlagen und Ankündigungen zu erhalten.

Information und Materialien

Informationen und Materialien

Tretet dem StudOn-Kurs bei um Zugriff auf die Vorlesungsmaterialien, Übungsblätter, Musterlösungen, Praktikumsunterlagen und Ankündigungen zu erhalten.

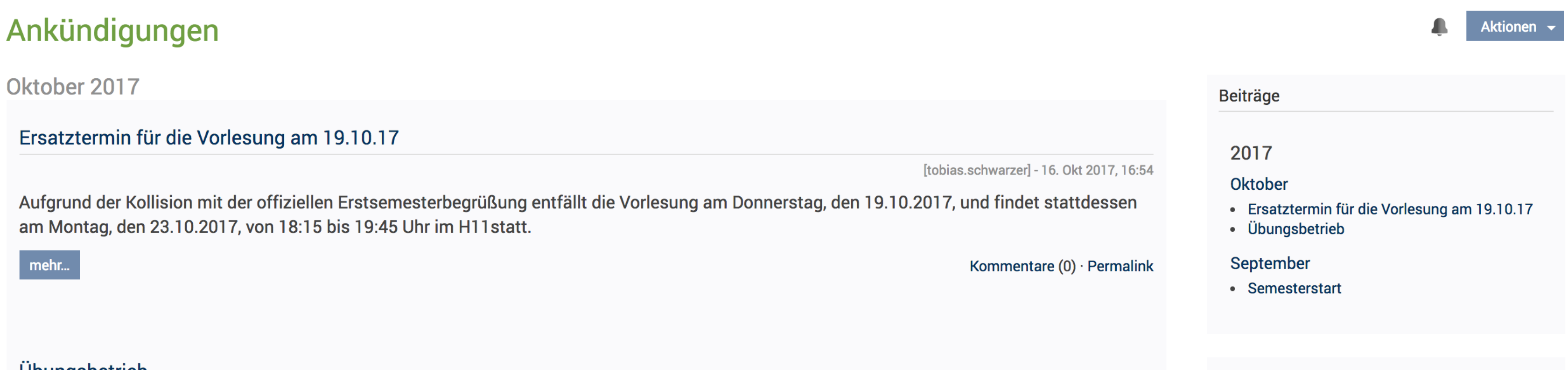
Aktiviert bitte die Ankündigkungsbenachrichtigungen im StudOn

Information und Materialien

Informationen und Materialien

Tretet dem StudOn-Kurs bei um Zugriff auf die Vorlesungsmaterialien, Übungsblätter, Musterlösungen, Praktikumsunterlagen und Ankündigungen zu erhalten.

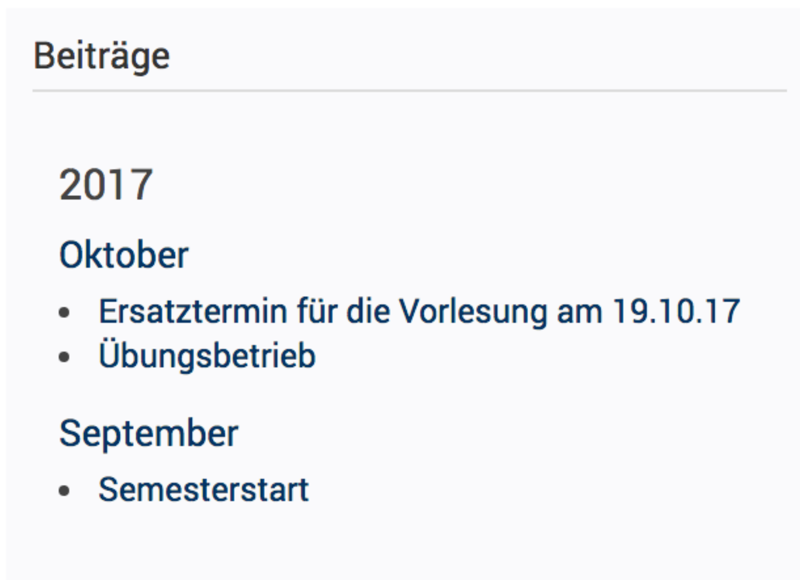
Aktiviert bitte die Ankündigungsbenachrichtigungen im StudOn



Information und Materialien

Informationen und Materialien

Tretet dem StudOn-Kurs bei um Zugriff auf die Vorlesungsmaterialien, Übungsblätter, Musterlösungen, Praktikumsunterlagen und Ankündigungen zu erhalten.
Aktiviert bitte die Ankündigungsbenachrichtigungen im StudOn



Information und Materialien

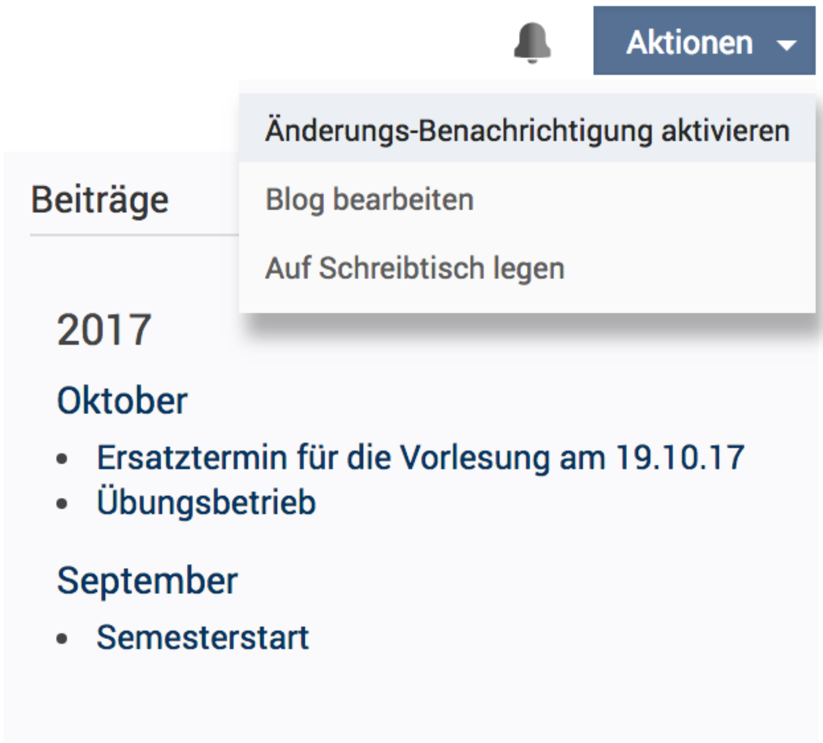
Informationen und Materialien

Tretet dem StudOn-Kurs bei um Zugriff auf die Vorlesungsmaterialien, Übungsblätter, Musterlösungen, Praktikumsunterlagen und Ankündigungen zu erhalten.
Aktiviert bitte die Ankündigkungsbenachrichtigungen im StudOn

Information und Materialien

Informationen und Materialien

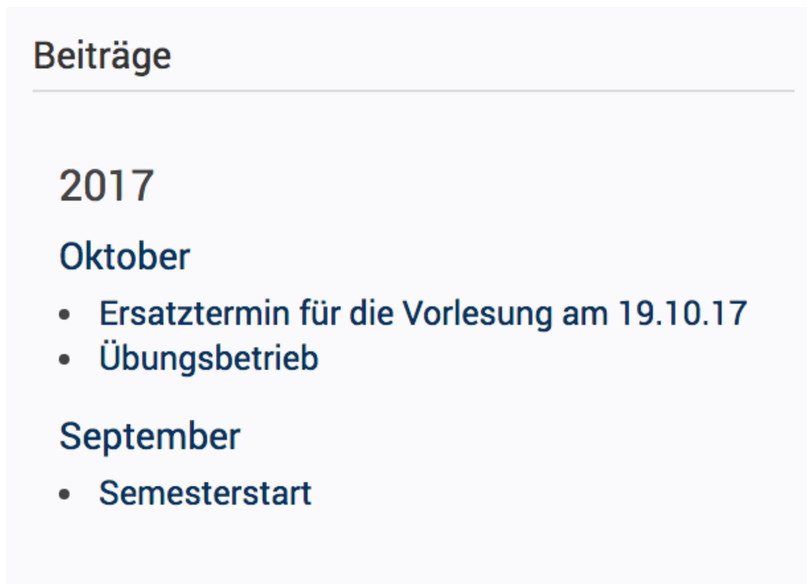
Tretet dem StudOn-Kurs bei um Zugriff auf die Vorlesungsmaterialien, Übungsblätter, Musterlösungen, Praktikumsunterlagen und Ankündigungen zu erhalten.
Aktiviert bitte die Ankündigkungsbenachrichtigungen im StudOn



Information und Materialien

Informationen und Materialien

Tretet dem StudOn-Kurs bei um Zugriff auf die Vorlesungsmaterialien, Übungsblätter, Musterlösungen, Praktikumsunterlagen und Ankündigungen zu erhalten.
Aktiviert bitte die Ankündigungsbenachrichtigungen im StudOn



Information und Materialien

Informationen und Materialien

Tretet dem StudOn-Kurs bei um Zugriff auf die Vorlesungsmaterialien, Übungsblätter, Musterlösungen, Praktikumsunterlagen und Ankündigungen zu erhalten.
Aktiviert bitte die Ankündigungsbenachrichtigungen im StudOn

- Forum der FSI Informatik:
  - https://fsi.cs.fau.de/forum/91-Grundlagen-der-Technischen-Informatik
  - Wenn ihr Fragen zum Fach habt, stellt sie hier ...

Information und Materialien

Informationen und Materialien

Tretet dem StudOn-Kurs bei um Zugriff auf die Vorlesungsmaterialien, Übungsblätter, Musterlösungen, Praktikumsunterlagen und Ankündigungen zu erhalten.
Aktiviert bitte die Ankündigkungsbenachrichtigungen im StudOn

- Forum der FSI Informatik:
  - https://fsi.cs.fau.de/forum/91-Grundlagen-der-Technischen-Informatik
  - Wenn ihr Fragen zum Fach habt, stellt sie hier ...

Kontakt

Name: Florian Frank
E-Mail: florian.ff.frank@fau.de
⇝ Bei Fragen, Kritik, Anregungen oder ähnlichem

Achtung – Wichtige Termine

Angemeldet für die Übungen?

Ihr solltet **alle** für die *aktuelle* Übung im „mein Campus“ angemeldet sein!

Achtung – Wichtige Termine

Angemeldet für die Übungen?

Ihr solltet **alle** für die *aktuelle* Übung im „mein Campus“ angemeldet sein!

1. Miniklausur

Findet am 29. November 2018 zur Vorlesungszeit statt!

Achtung – Wichtige Termine

Angemeldet für die Übungen?

Ihr solltet **alle** für die *aktuelle* Übung im „mein Campus“ angemeldet sein!

1. Miniklausur

Findet am 29. November 2018 zur Vorlesungszeit statt!

2. Miniklausur

Findet am 17. Januar 2019 zur Vorlesungszeit statt!

Übungen zur Grundlagen der Technischen Informatik

Übung 1 – Diskretisierung, Informationsgehalt und Kodierung

Florian Frank

Friedrich-Alexander-Universität Erlangen-Nürnberg

Wintersemester 2018/19

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Wichtiger Hinweis

Dieser Foliensatz enthält den Inhalt der Übung zu den „Grundlagen der Technischen Informaik“ des Wintersemesters 2018/19. Für den Inhalt dieses Foliensatzes ist der Author allein verantwortlich. Der Foliensatz ist **inoffiziell** und stellt damit **keine** Veröffentlichung des Lehrstuhls dar. Bei Unstimmigkeiten und eventuell vorhandenen Fehlern bitte ich um eine E-Mail[a].

[a] an florian.ff.frank@fau.de

Was machen wir heute?

Aufgabe 1 — Diskretisierung

Was machen wir heute?

Aufgabe 1 — Diskretisierung

Aufgabe 2 — Informationsgehalt

Was machen wir heute?

Aufgabe 1 — Diskretisierung

Aufgabe 2 — Informationsgehalt

Aufgabe 3 — Kodierung

Aufgabe 1 — Diskretisierung

Aufgabe 1 – Diskretisierung

Das Analogsignal aus Abbildung 1 mit einem Eingangsspannungsbereich von 0-10 Volt soll in ein 4-wertiges Digitalsignal umgewandelt werden. Der undefinierte Bereich zwischen zwei Digitalwerten soll ⅓ des Intervalls eines Digitalwertes betragen.

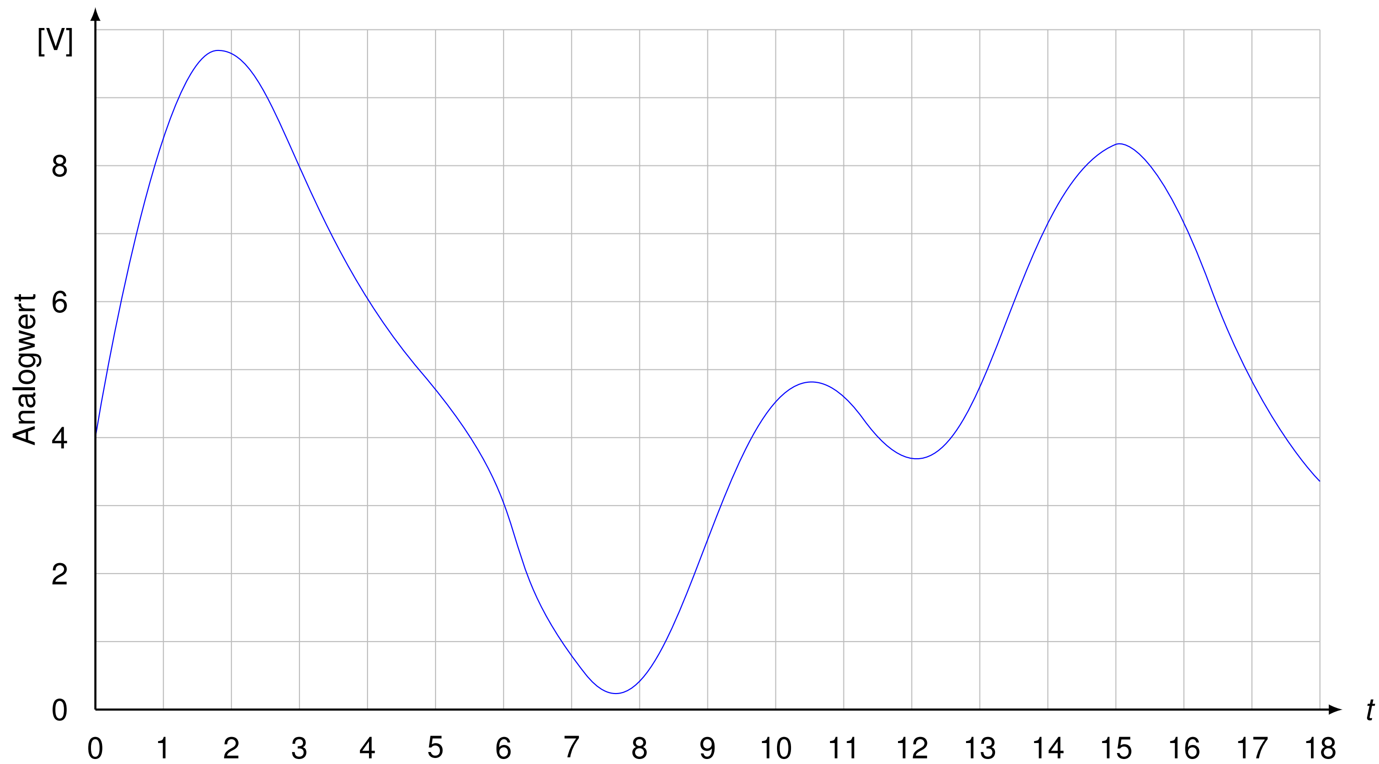


Abbildung 1: Zu konvertierendes Analogsignal

Aufgabe 1 – Diskretisierung

Das Analogsignal aus Abbildung 1 mit einem Eingangsspannungsbereich von 0-10 Volt soll in ein 4-wertiges Digitalsignal umgewandelt werden. Der undefinierte Bereich zwischen zwei Digitalwerten soll ⅓ des Intervalls eines Digitalwertes betragen.

Aufgabe 1 – Diskretisierung

Das Analogsignal aus Abbildung 1 mit einem Eingangsspannungsbereich von 0-10 Volt soll in ein 4-wertiges Digitalsignal umgewandelt werden. Der undefinierte Bereich zwischen zwei Digitalwerten soll ⅓ des Intervalls eines Digitalwertes betragen.

a) Geben Sie die Intervalle für die digitalisierten Werte an.

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein. Beim Verlassen eines Werteintervalls soll der digitalisierte Wert so lange erhalten bleiben, bis das analoge Signal in das nächste Werteintervall eintritt. Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.

Aufgabe 1 – Diskretisierung

Das Analogsignal aus Abbildung 1 mit einem Eingangsspannungsbereich von 0-10 Volt soll in ein 4-wertiges Digitalsignal umgewandelt werden. Der undefinierte Bereich zwischen zwei Digitalwerten soll ⅓ des Intervalls eines Digitalwertes betragen.

a) Geben Sie die Intervalle für die digitalisierten Werte an.

Aufgabe 1 – Diskretisierung

Das Analogsignal aus Abbildung 1 mit einem Eingangsspannungsbereich von 0-10 Volt soll in ein 4-wertiges Digitalsignal umgewandelt werden. Der undefinierte Bereich zwischen zwei Digitalwerten soll ⅓ des Intervalls eines Digitalwertes betragen.

a) Geben Sie die Intervalle für die digitalisierten Werte an.
   Lösung

Aufgabe 1 – Diskretisierung

Das Analogsignal aus Abbildung 1 mit einem Eingangsspannungsbereich von 0-10 Volt soll in ein 4-wertiges Digitalsignal umgewandelt werden. Der undefinierte Bereich zwischen zwei Digitalwerten soll ⅓ des Intervalls eines Digitalwertes betragen.

a) Geben Sie die Intervalle für die digitalisierten Werte an.

Lösung

- Wie viele Bereiche gibt es? —

Aufgabe 1 – Diskretisierung

Das Analogsignal aus Abbildung 1 mit einem Eingangsspannungsbereich von 0-10 Volt soll in ein **4-wertiges Digitalsignal** umgewandelt werden. Der undefinierte Bereich zwischen zwei Digitalwerten soll ⅓ des Intervalls eines Digitalwertes betragen.

a) Geben Sie die Intervalle für die digitalisierten Werte an.

Lösung

- □ Wie viele Bereiche gibt es? —

Aufgabe 1 – Diskretisierung

Das Analogsignal aus Abbildung 1 mit einem Eingangsspannungsbereich von 0-10 Volt soll in ein **4-wertiges Digitalsignal** umgewandelt werden. Der undefinierte Bereich zwischen zwei Digitalwerten soll **⅓ des Intervalls** eines Digitalwertes betragen.

a) Geben Sie die Intervalle für die digitalisierten Werte an.

Lösung

- Wie viele Bereiche gibt es? —

Aufgabe 1 – Diskretisierung

Das Analogsignal aus Abbildung 1 mit einem Eingangsspannungsbereich von 0-10 Volt soll in ein **4-wertiges Digitalsignal** umgewandelt werden. Der undefinierte Bereich zwischen zwei Digitalwerten soll **⅓ des Intervalls** eines Digitalwertes betragen.

a) Geben Sie die Intervalle für die digitalisierten Werte an.

Lösung

- Wie viele Bereiche gibt es? — 4 Intervalle + 3 undefinierte Bereiche $\rightarrow$ $4 + 3 \cdot \frac{1}{3} = 5$ Bereiche

Aufgabe 1 – Diskretisierung

Das Analogsignal aus Abbildung 1 mit einem Eingangsspannungsbereich von 0-10 Volt soll in ein **4-wertiges Digitalsignal** umgewandelt werden. Der undefinierte Bereich zwischen zwei Digitalwerten soll **⅓ des Intervalls** eines Digitalwertes betragen.

a) Geben Sie die Intervalle für die digitalisierten Werte an.

Lösung

- Wie viele Bereiche gibt es? — 4 Intervalle + 3 undefinierte Bereiche $\rightarrow$ $4 + 3 \cdot \frac{1}{3} = 5$ Bereiche
- Welchen Bereich gilt es zu unterteilen? —

Aufgabe 1 – Diskretisierung

Das Analogsignal aus Abbildung 1 mit einem **Eingangsspannungsbereich von 0-10 Volt** soll in ein **4-wertiges Digitalsignal** umgewandelt werden. Der undefinierte Bereich zwischen zwei Digitalwerten soll **⅓ des Intervalls** eines Digitalwertes betragen.

a) Geben Sie die Intervalle für die digitalisierten Werte an.

Lösung

- Wie viele Bereiche gibt es? — 4 Intervalle + 3 undefinierte Bereiche $\rightarrow$ $4 + 3 \cdot \frac{1}{3} = 5$ Bereiche
- Welchen Bereich gilt es zu unterteilen? — 0-10V $\Rightarrow$ $\Delta = 10V - 0V = 10V$

Aufgabe 1 – Diskretisierung

Das Analogsignal aus Abbildung 1 mit einem **Eingangsspannungsbereich von 0-10 Volt** soll in ein **4-wertiges Digitalsignal** umgewandelt werden. Der undefinierte Bereich zwischen zwei Digitalwerten soll **⅓ des Intervalls** eines Digitalwertes betragen.

a) Geben Sie die Intervalle für die digitalisierten Werte an.

Lösung

- Wie viele Bereiche gibt es? — 4 Intervalle + 3 undefinierte Bereiche $\rightarrow$ $4 + 3 \cdot \frac{1}{3} = 5$ Bereiche
- Welchen Bereich gilt es zu unterteilen? — 0-10V $\Rightarrow$ $\Delta = 10V - 0V = 10V$
- Gleichung aufstellen: $5 \cdot d_I = 10V \Rightarrow d_I = 2V \Rightarrow d_{undefiniert} = \frac{2}{3}V$

Aufgabe 1 – Diskretisierung

Das Analogsignal aus Abbildung 1 mit einem **Eingangsspannungsbereich von 0-10 Volt** soll in ein **4-wertiges Digitalsignal** umgewandelt werden. Der undefinierte Bereich zwischen zwei Digitalwerten soll **⅓ des Intervalls** eines Digitalwertes betragen.

a) Geben Sie die Intervalle für die digitalisierten Werte an.

Lösung

- Wie viele Bereiche gibt es? — 4 Intervalle + 3 undefinierte Bereiche $\rightarrow$ $4 + 3 \cdot \frac{1}{3} = 5$ Bereiche
- Welchen Bereich gilt es zu unterteilen? — 0-10V $\Rightarrow$ $\Delta = 10V - 0V = 10V$
- Gleichung aufstellen: $5 \cdot d_I = 10V \Rightarrow d_I = 2V \Rightarrow d_{undefiniert} = \frac{2}{3}V$

Intervall	Definitionsbereich
4	
3	
2	
1	$[0; 2]$

Aufgabe 1 – Diskretisierung

Das Analogsignal aus Abbildung 1 mit einem **Eingangsspannungsbereich von 0-10 Volt** soll in ein **4-wertiges Digitalsignal** umgewandelt werden. Der undefinierte Bereich zwischen zwei Digitalwerten soll **⅓ des Intervalls** eines Digitalwertes betragen.

a) Geben Sie die Intervalle für die digitalisierten Werte an.

Lösung

- □ Wie viele Bereiche gibt es? — 4 Intervalle + 3 undefinierte Bereiche $\rightarrow$ $4 + 3 \cdot \frac{1}{3} = 5$ Bereiche
- □ Welchen Bereich gilt es zu unterteilen? — 0-10V $\Rightarrow$ $\Delta = 10V - 0V = 10V$
- □ Gleichung aufstellen: $5 \cdot d_I = 10V \Rightarrow d_I = 2V \Rightarrow d_{undefiniert} = \frac{2}{3}V$

Intervall	Definitionsbereich
4	
3	
2	$[2^2/_3; 4^2/_3]$
1	$[0; 2]$

Aufgabe 1 – Diskretisierung

Das Analogsignal aus Abbildung 1 mit einem **Eingangsspannungsbereich von 0-10 Volt** soll in ein **4-wertiges Digitalsignal** umgewandelt werden. Der undefinierte Bereich zwischen zwei Digitalwerten soll **⅓ des Intervalls** eines Digitalwertes betragen.

a) Geben Sie die Intervalle für die digitalisierten Werte an.

Lösung

- Wie viele Bereiche gibt es? — 4 Intervalle + 3 undefinierte Bereiche $\rightarrow$ $4 + 3 \cdot \frac{1}{3} = 5$ Bereiche
- Welchen Bereich gilt es zu unterteilen? — 0-10V $\Rightarrow$ $\Delta = 10V - 0V = 10V$
- Gleichung aufstellen: $5 \cdot d_I = 10V \Rightarrow d_I = 2V \Rightarrow d_{undefiniert} = \frac{2}{3}V$

Intervall	Definitionsbereich
4	
3	$[5^1/_3; 7^1/_3]$
2	$[2^2/_3; 4^2/_3]$
1	$[0; 2]$

Aufgabe 1 – Diskretisierung

Das Analogsignal aus Abbildung 1 mit einem **Eingangsspannungsbereich von 0-10 Volt** soll in ein **4-wertiges Digitalsignal** umgewandelt werden. Der undefinierte Bereich zwischen zwei Digitalwerten soll **⅓ des Intervalls** eines Digitalwertes betragen.

a) Geben Sie die Intervalle für die digitalisierten Werte an.

Lösung

- Wie viele Bereiche gibt es? — 4 Intervalle + 3 undefinierte Bereiche $\rightarrow$ $4 + 3 \cdot \frac{1}{3} = 5$ Bereiche
- Welchen Bereich gilt es zu unterteilen? — 0-10V $\Rightarrow$ $\Delta = 10\text{V} - 0\text{V} = 10\text{V}$
- Gleichung aufstellen: $5 \cdot d_I = 10\text{V} \Rightarrow d_I = 2\text{V} \Rightarrow d_{undefiniert} = \frac{2}{3}\text{V}$

Intervall	Definitionsbereich
4	$[8; 10]$
3	$[5^1/_3; 7^1/_3]$
2	$[2^2/_3; 4^2/_3]$
1	$[0; 2]$

Aufgabe 1 – Diskretisierung

a) Geben Sie die Intervalle für die digitalisierten Werte an.
Lösung

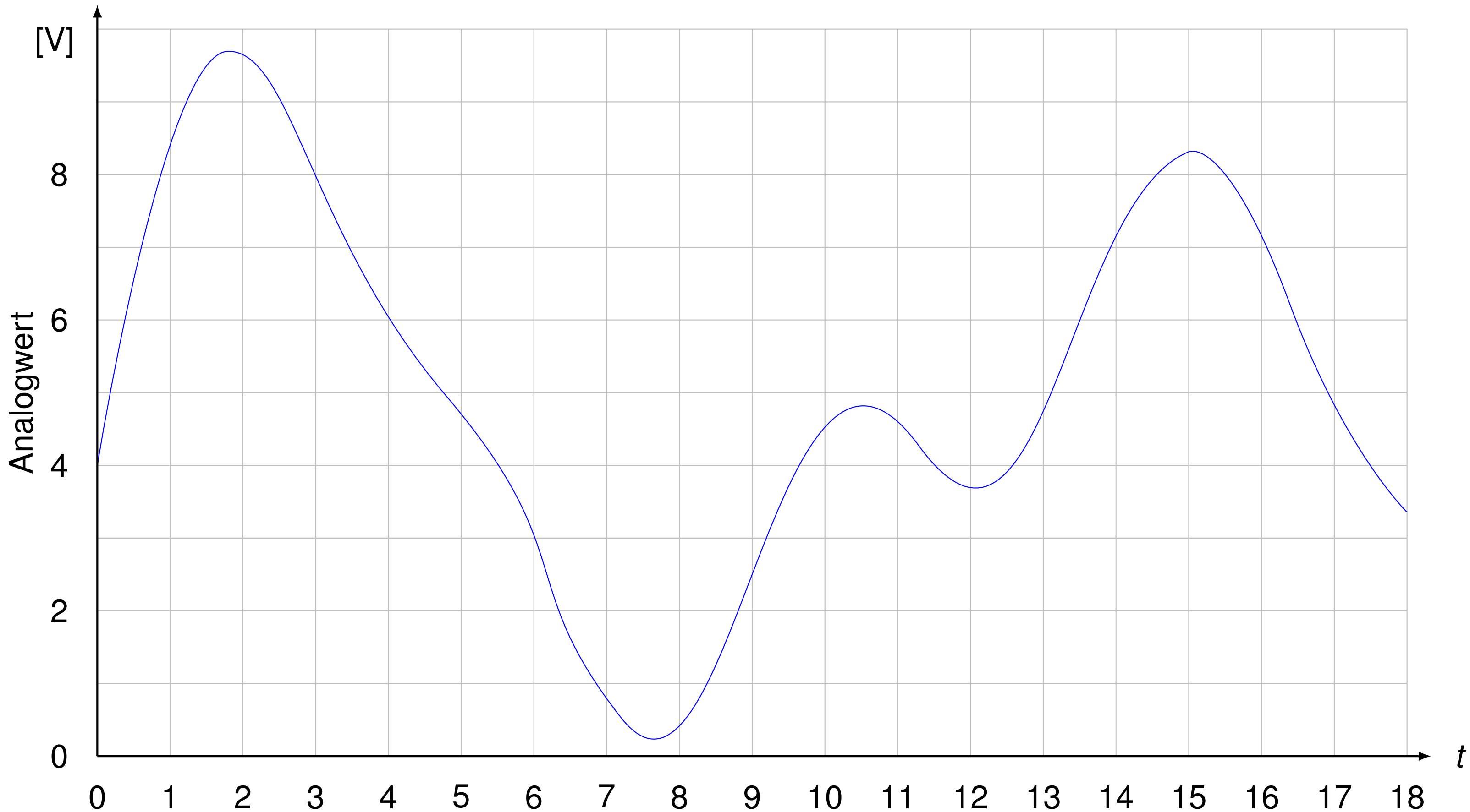


Abbildung 2: Zu diskretisierende Bereiche

Aufgabe 1 – Diskretisierung

a) Geben Sie die Intervalle für die digitalisierten Werte an.

Lösung

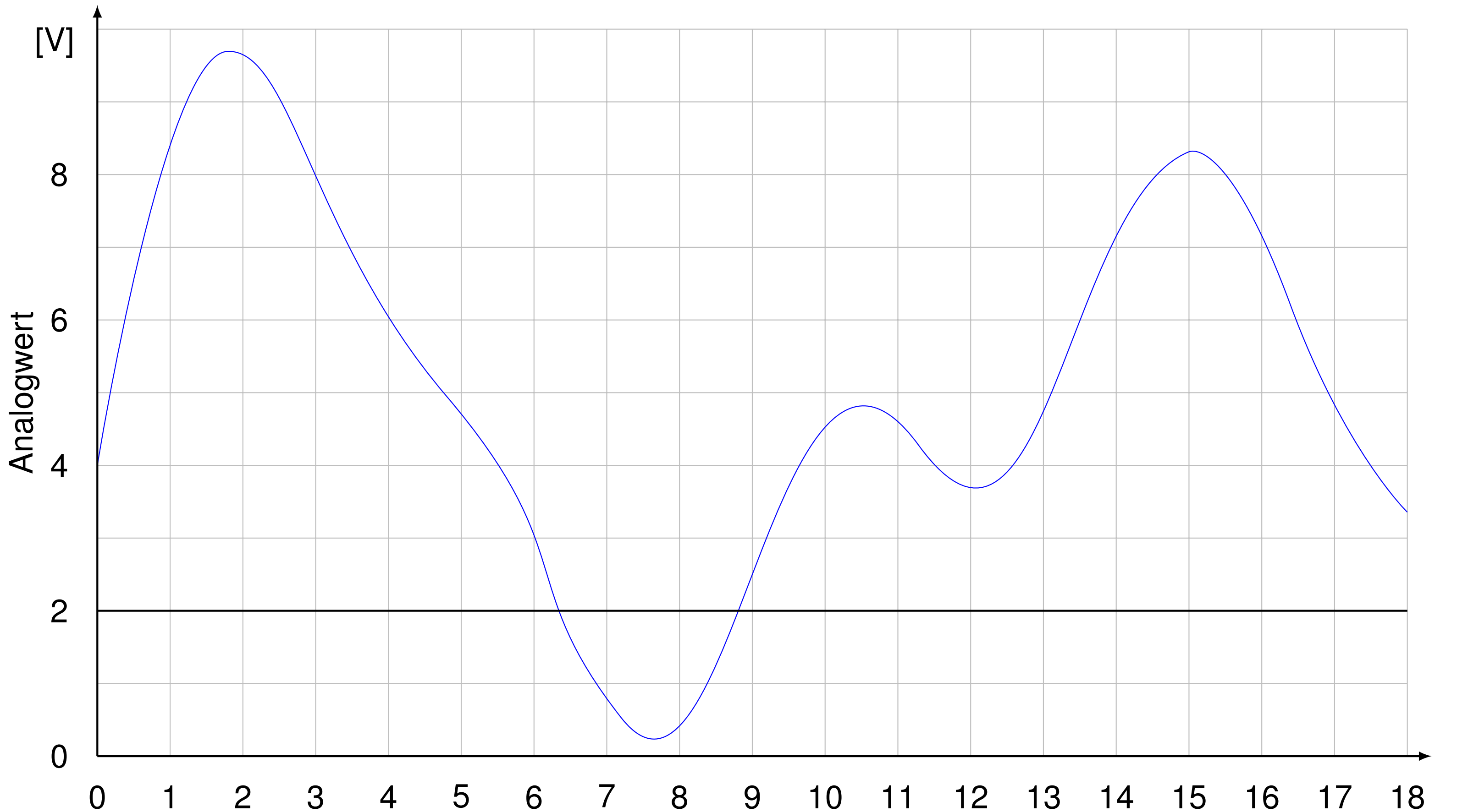


Abbildung 2: Zu diskretisierende Bereiche

Aufgabe 1 – Diskretisierung

a) Geben Sie die Intervalle für die digitalisierten Werte an.
   Lösung

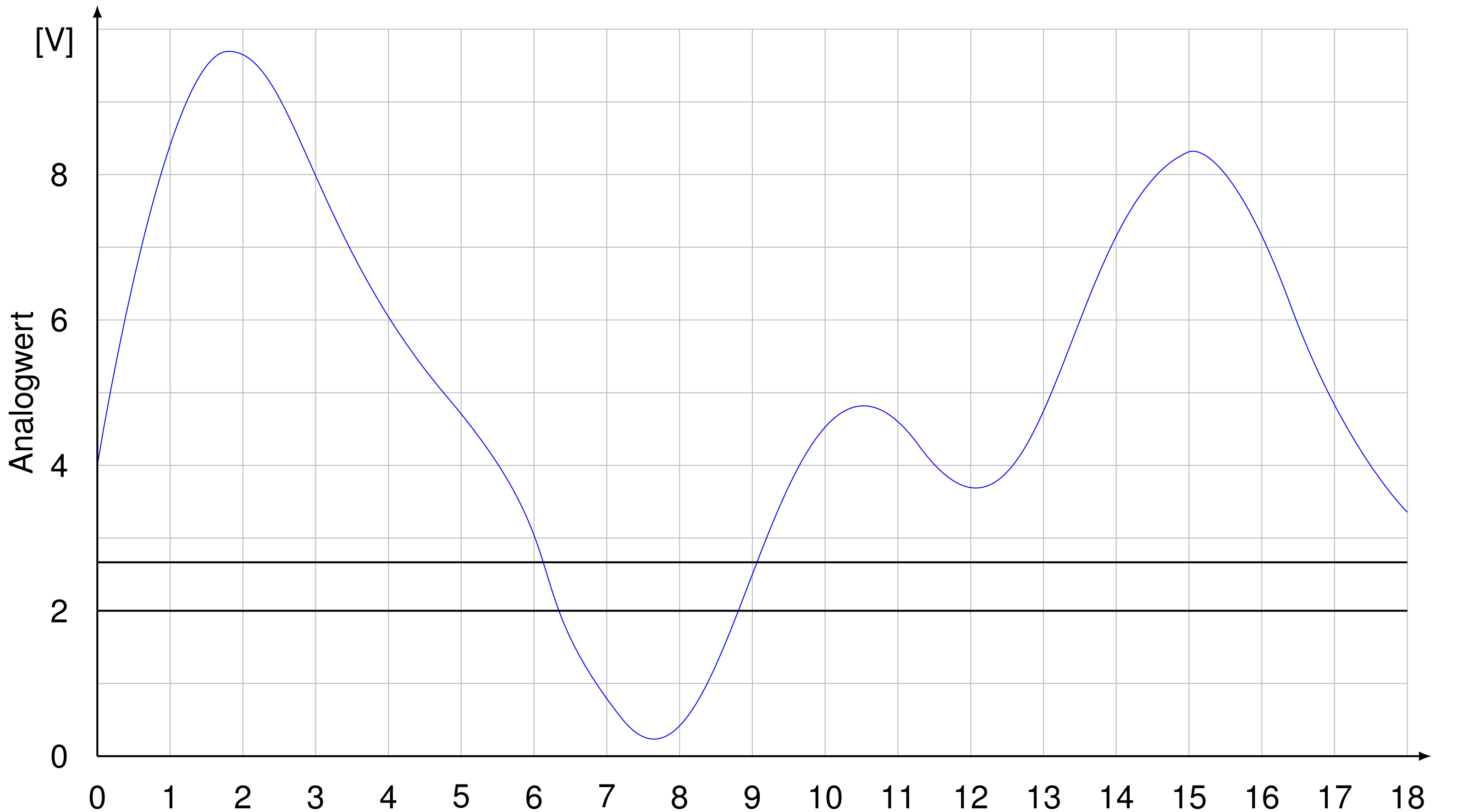


Abbildung 2: Zu diskretisierende Bereiche

Aufgabe 1 – Diskretisierung

a) Geben Sie die Intervalle für die digitalisierten Werte an.
Lösung

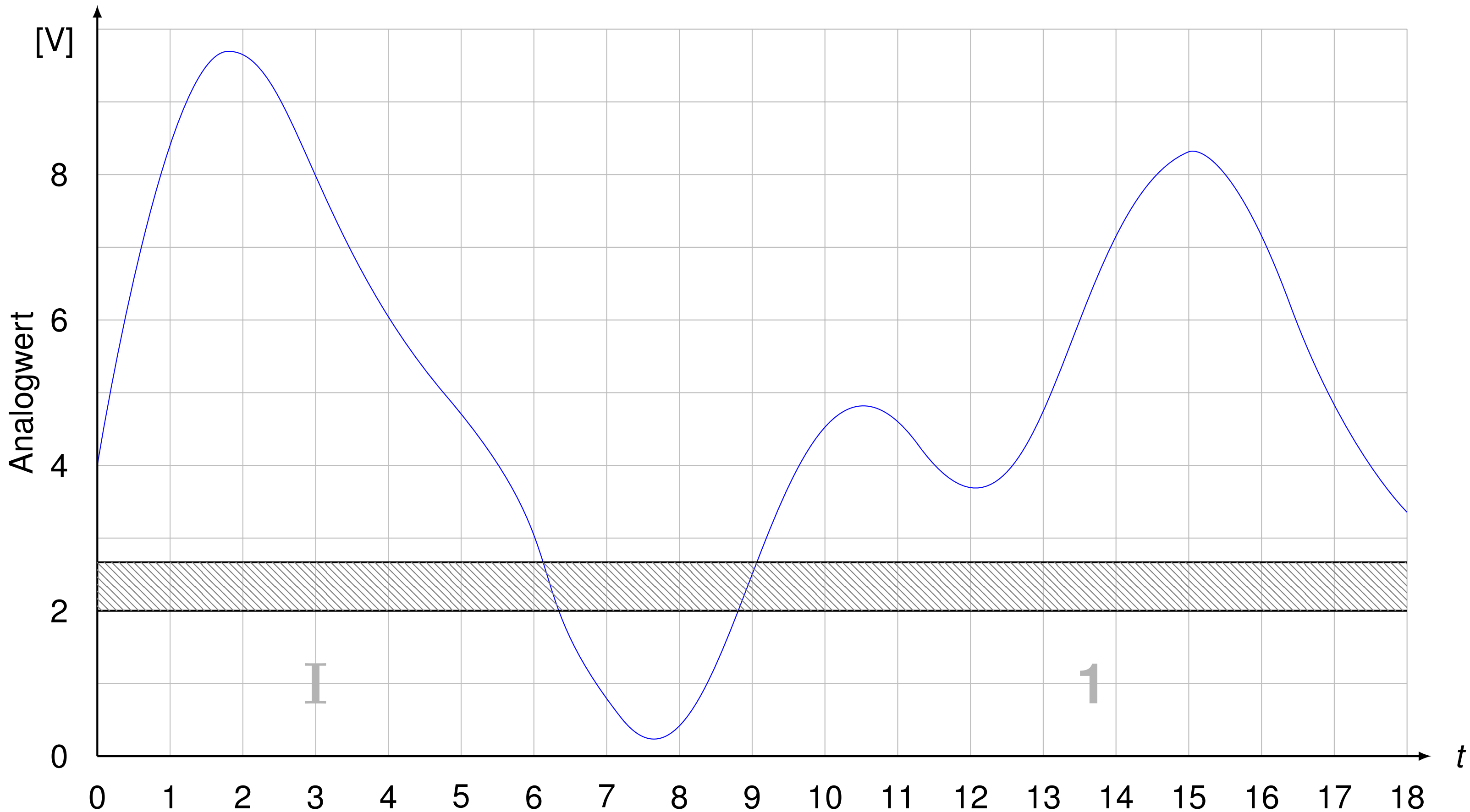


Abbildung 2: Zu diskretisierende Bereiche

Aufgabe 1 – Diskretisierung

a) Geben Sie die Intervalle für die digitalisierten Werte an.
Lösung

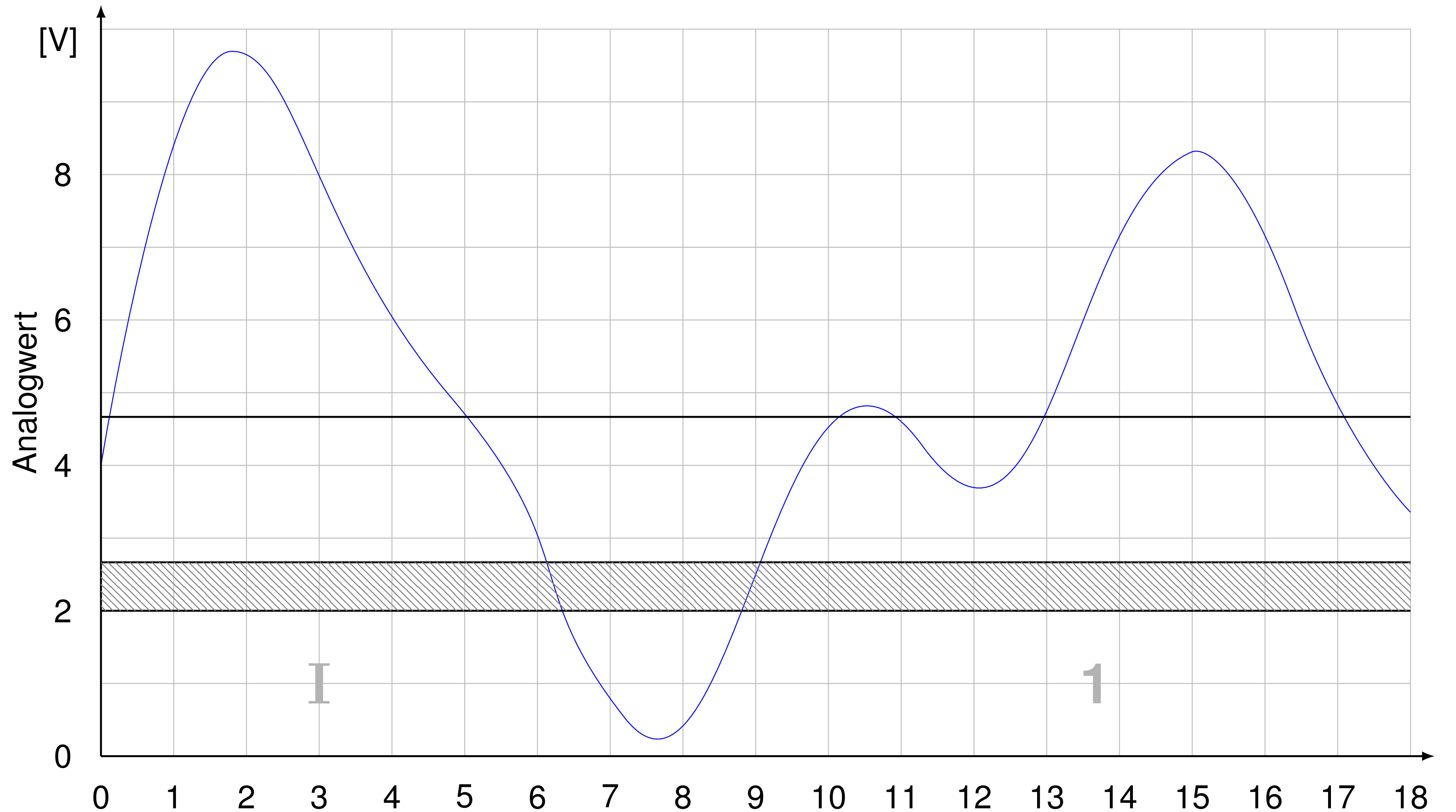


Abbildung 2: Zu diskretisierende Bereiche

Aufgabe 1 – Diskretisierung

a) Geben Sie die Intervalle für die digitalisierten Werte an.

Lösung

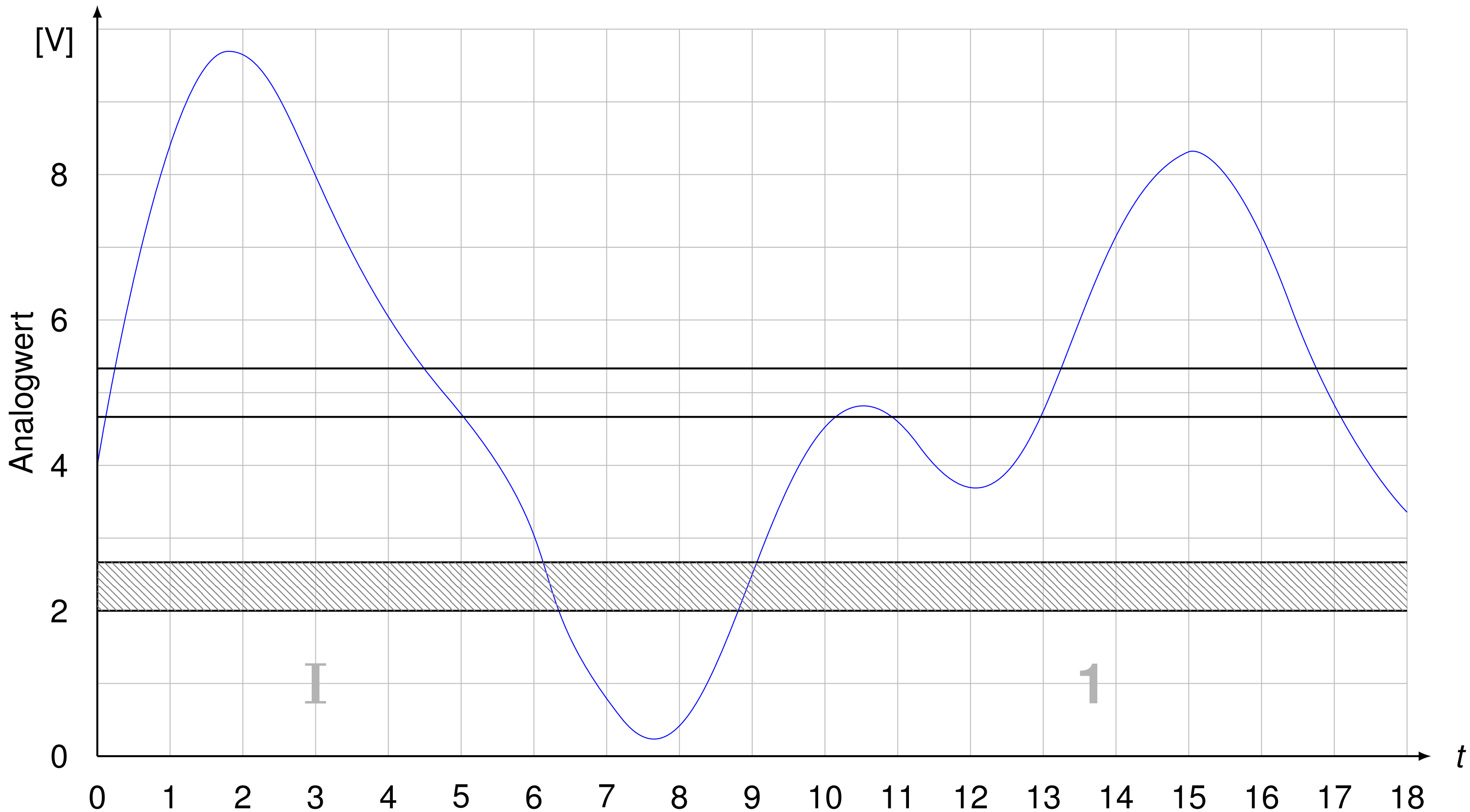


Abbildung 2: Zu diskretisierende Bereiche

Aufgabe 1 – Diskretisierung

a) Geben Sie die Intervalle für die digitalisierten Werte an.
Lösung

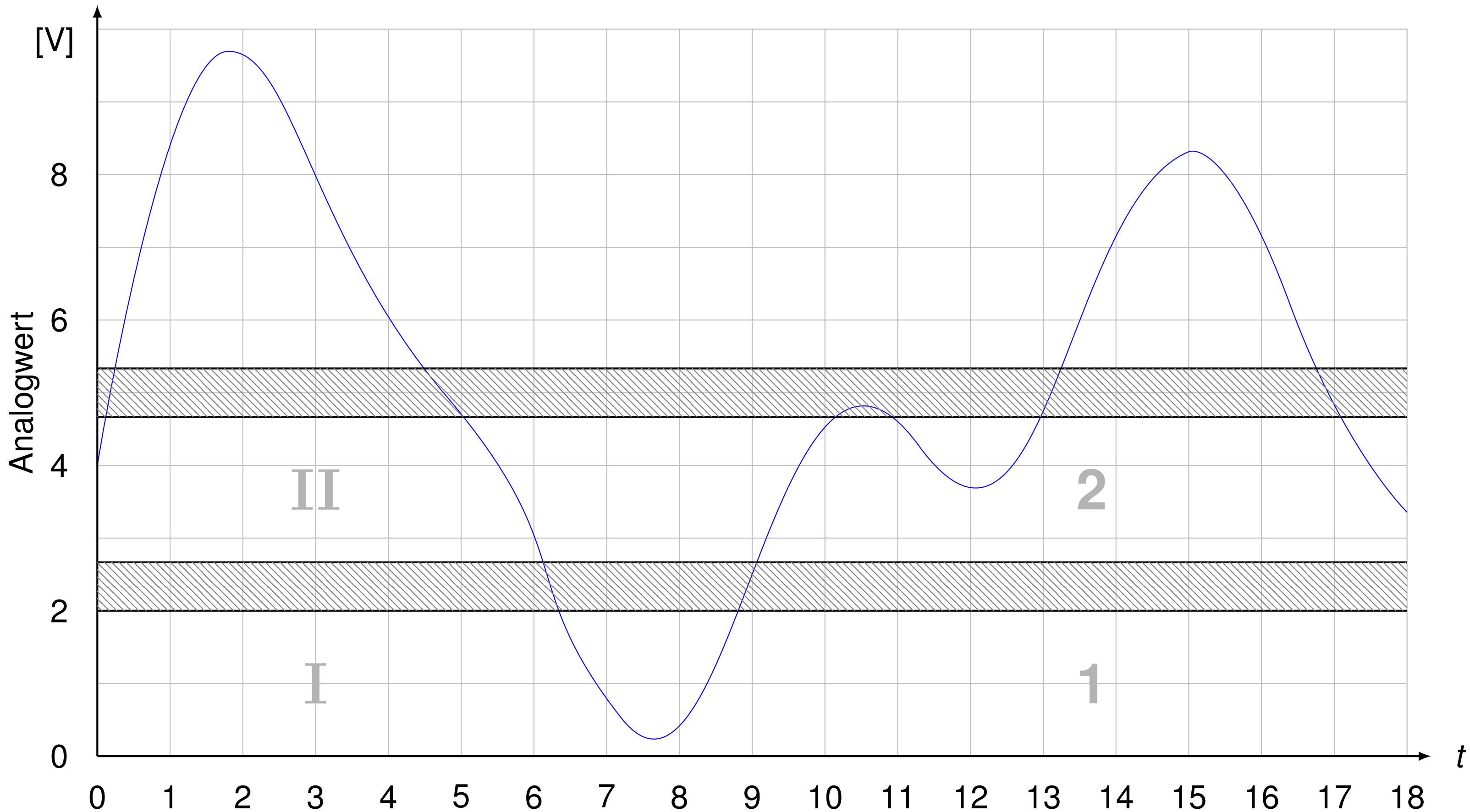


Abbildung 2: Zu diskretisierende Bereiche

Aufgabe 1 – Diskretisierung

a) Geben Sie die Intervalle für die digitalisierten Werte an.
Lösung

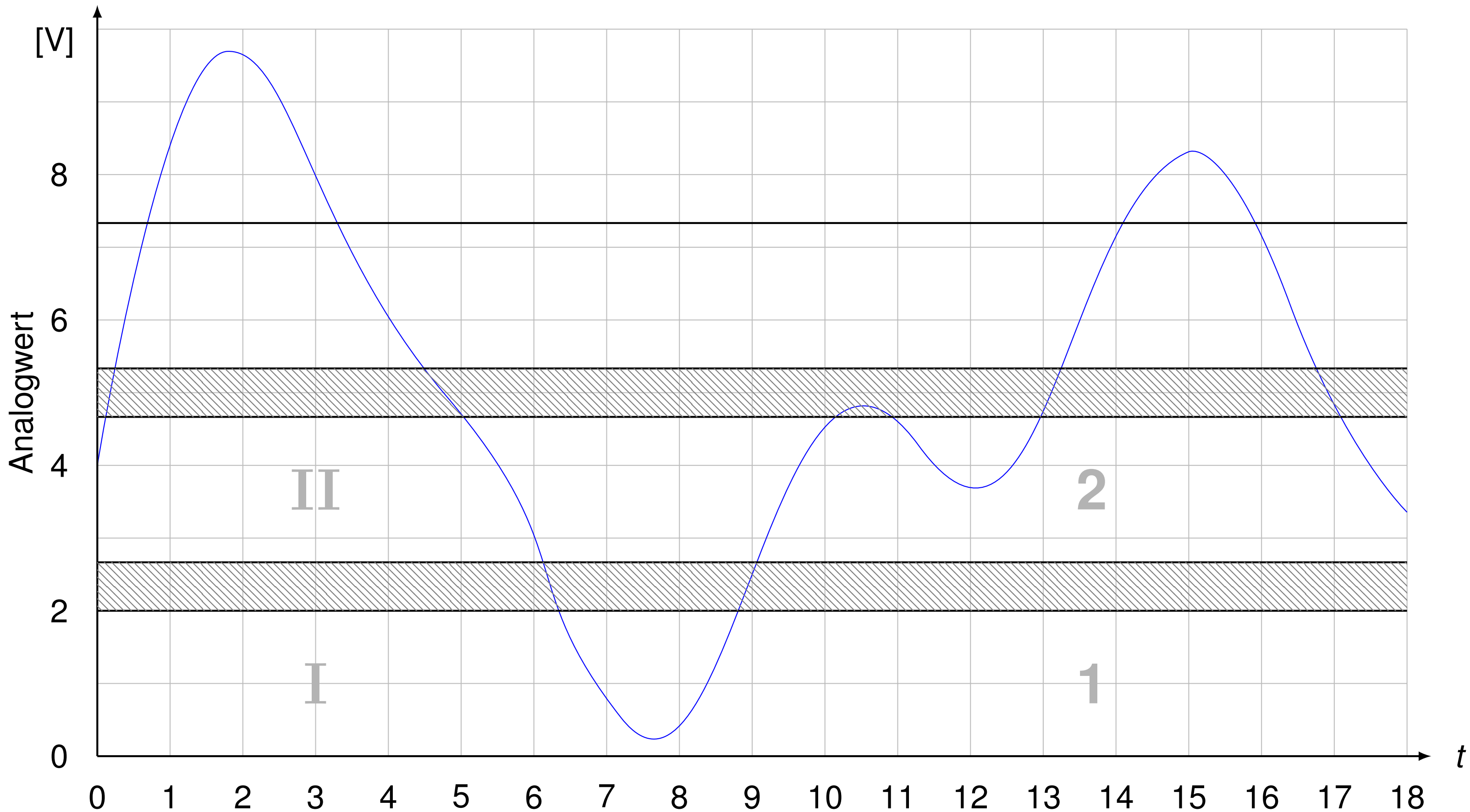


Abbildung 2: Zu diskretisierende Bereiche

Aufgabe 1 – Diskretisierung

a) Geben Sie die Intervalle für die digitalisierten Werte an.

Lösung

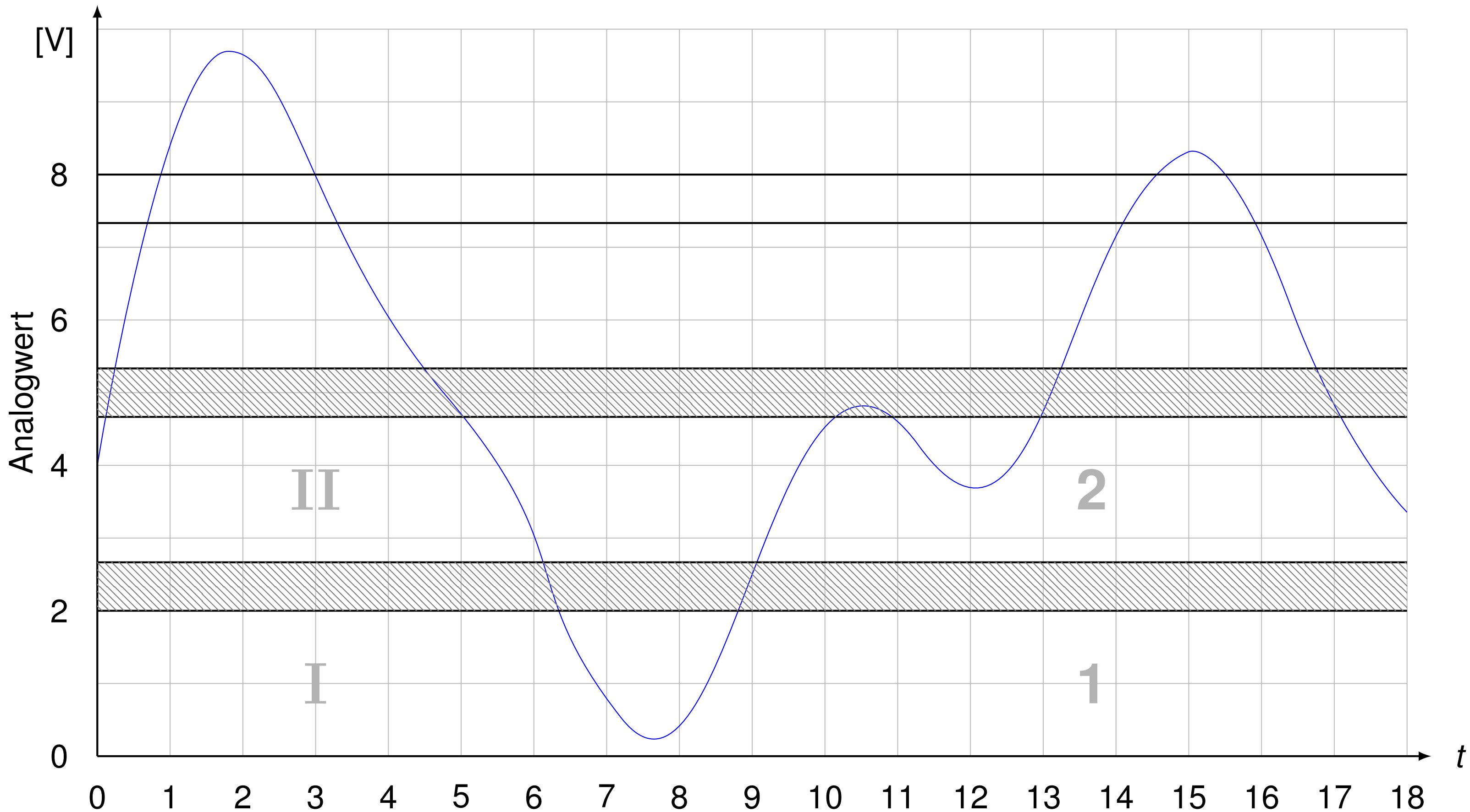


Abbildung 2: Zu diskretisierende Bereiche

Aufgabe 1 – Diskretisierung

a) Geben Sie die Intervalle für die digitalisierten Werte an.
Lösung

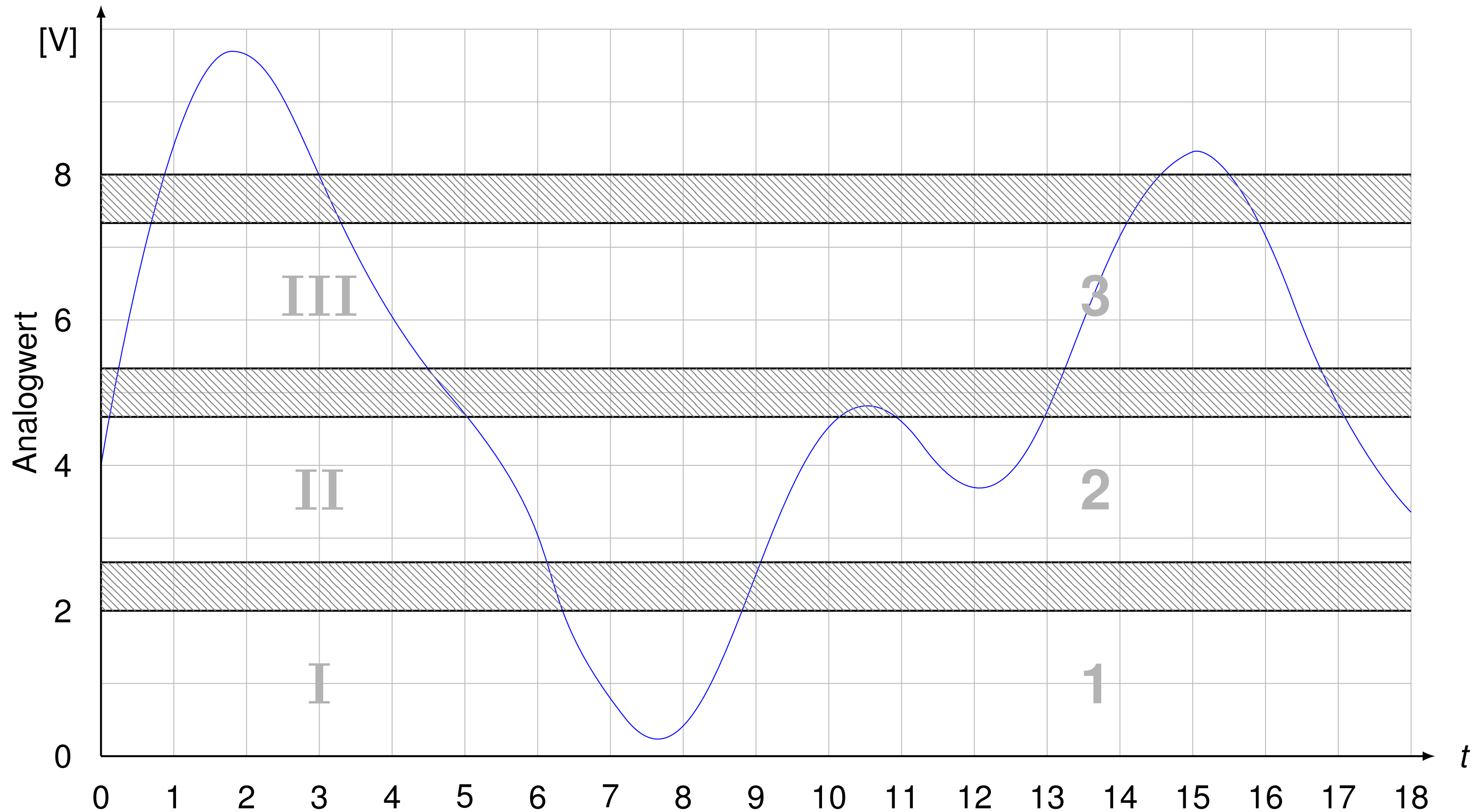


Abbildung 2: Zu diskretisierende Bereiche

Aufgabe 1 – Diskretisierung

a) Geben Sie die Intervalle für die digitalisierten Werte an.

Lösung

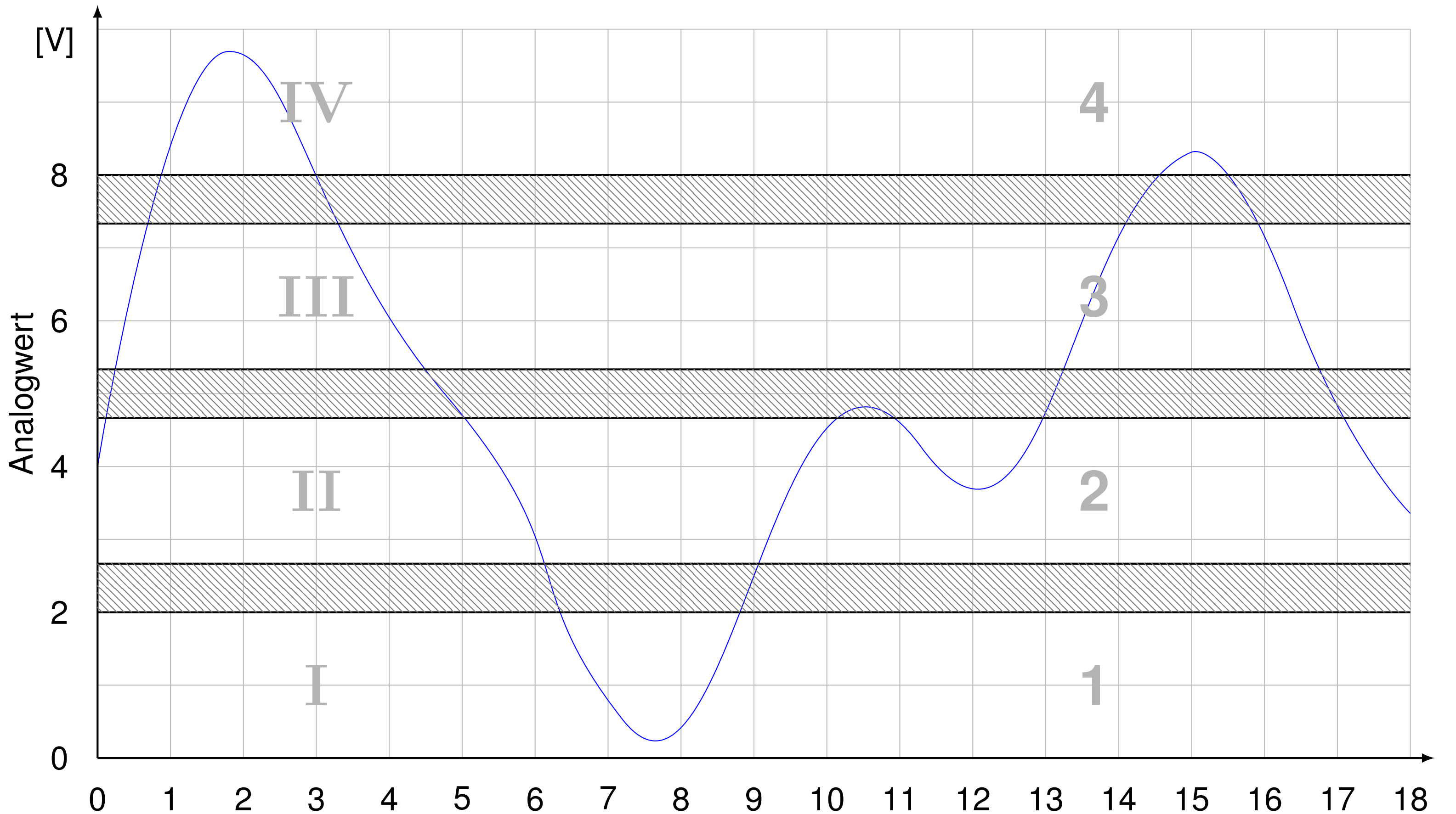


Abbildung 2: Zu diskretisierende Bereiche

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.
Beim Verlassen eines Werteintervalls soll der digitalisierte Wert so lange erhalten bleiben, bis das analoge Signal in das nächste Werteintervall eintritt.

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.
Lösung

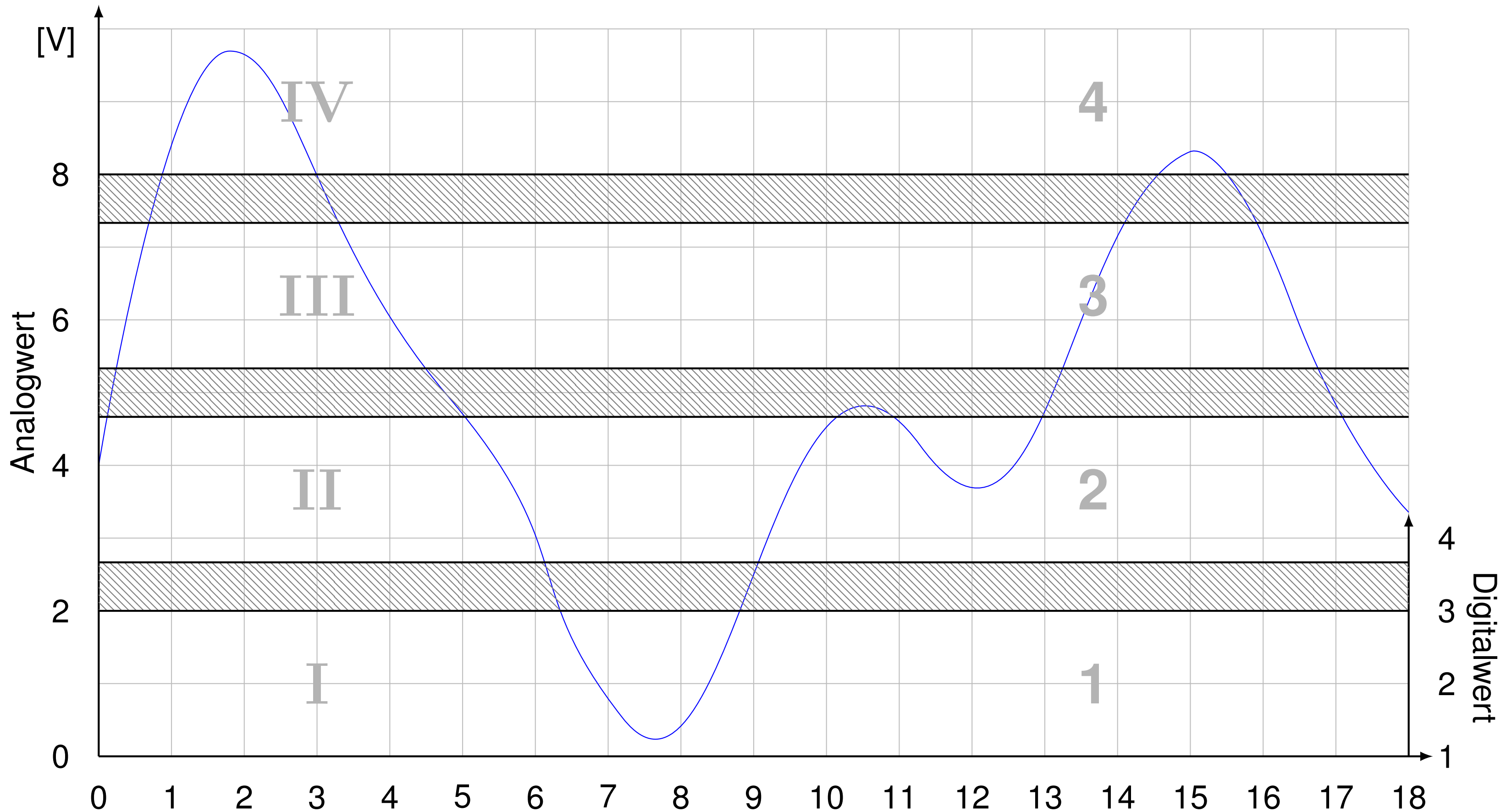


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.
Lösung

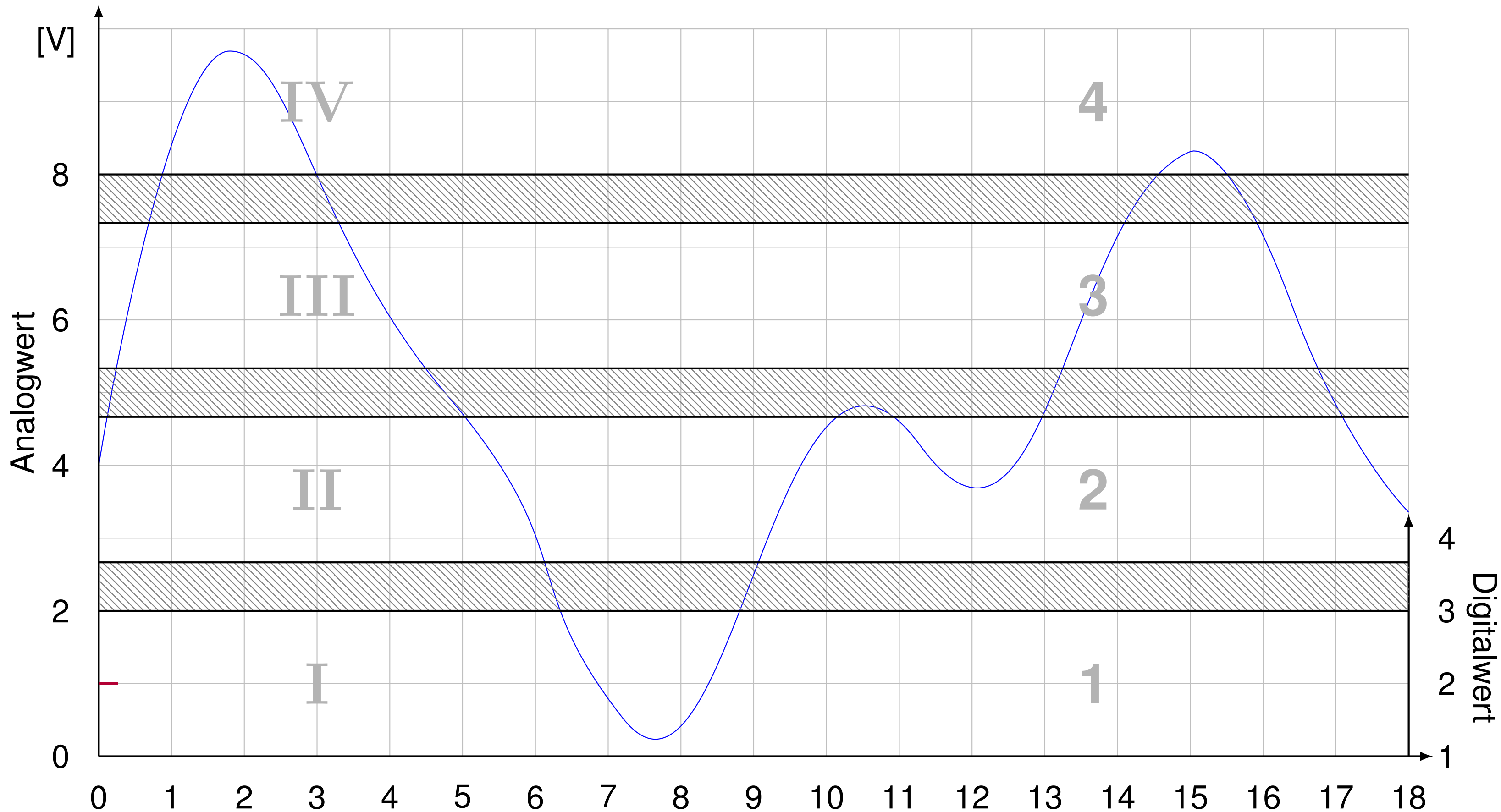


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.

Lösung

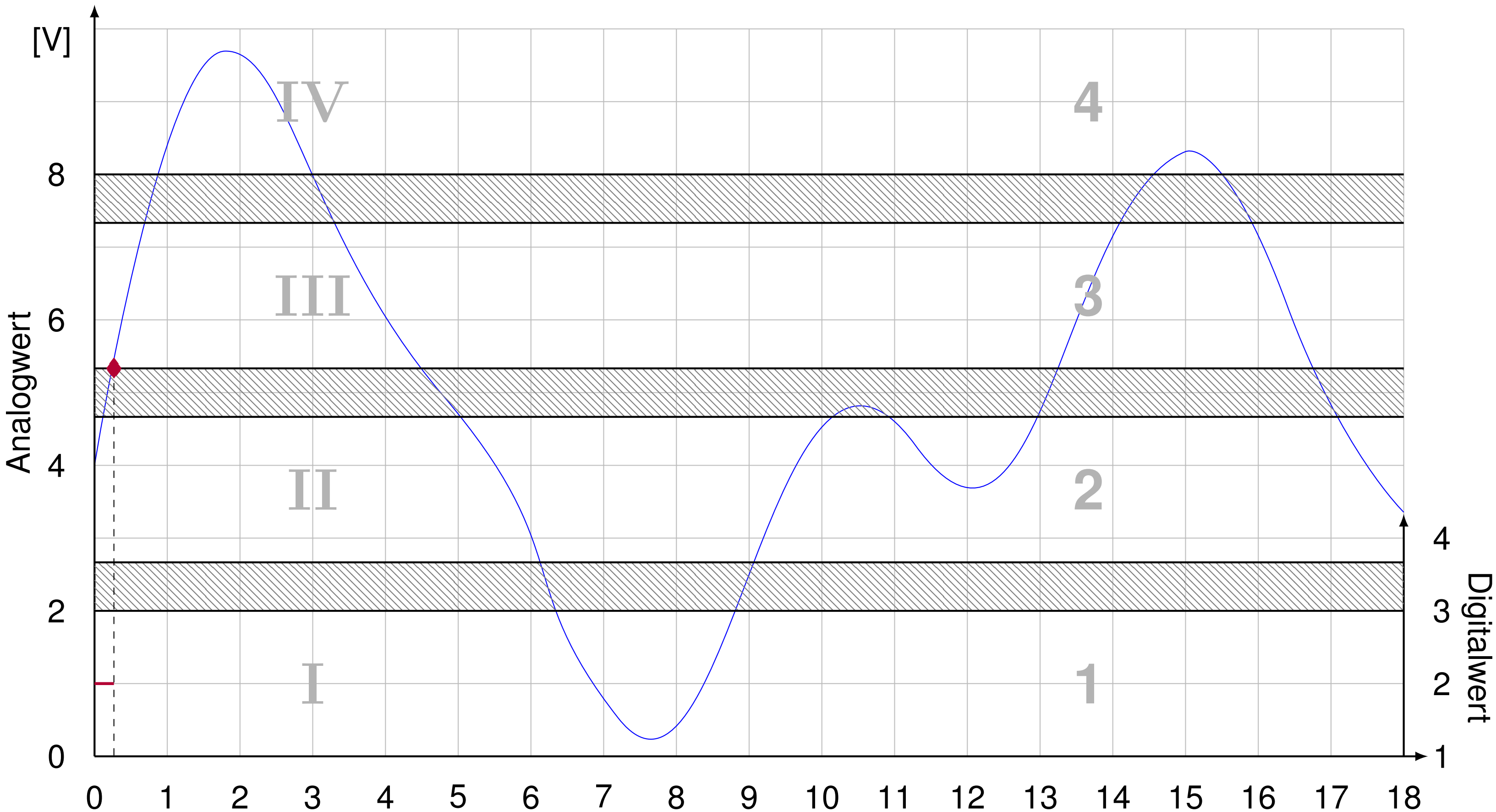


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.

Lösung

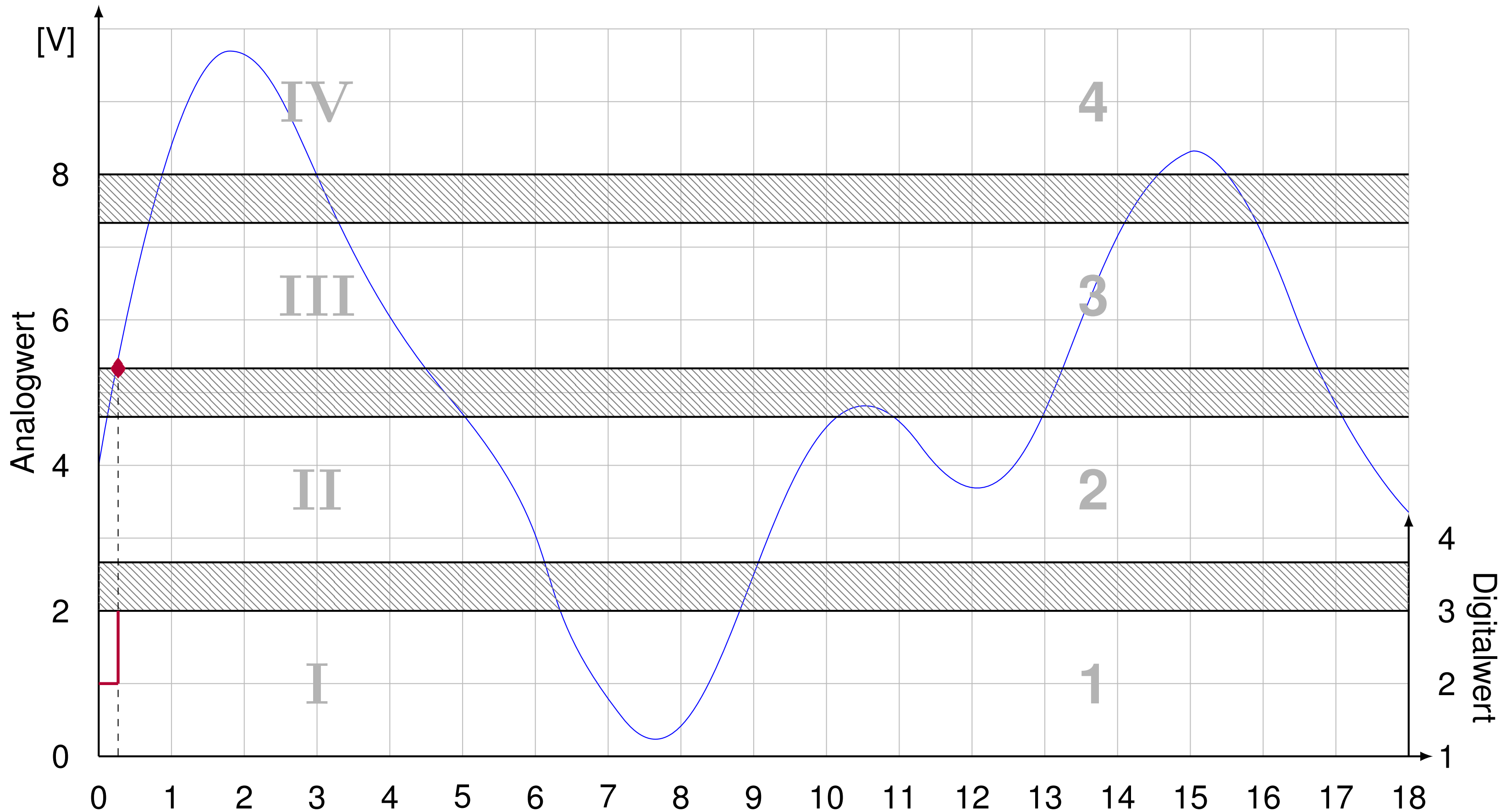


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.
Lösung

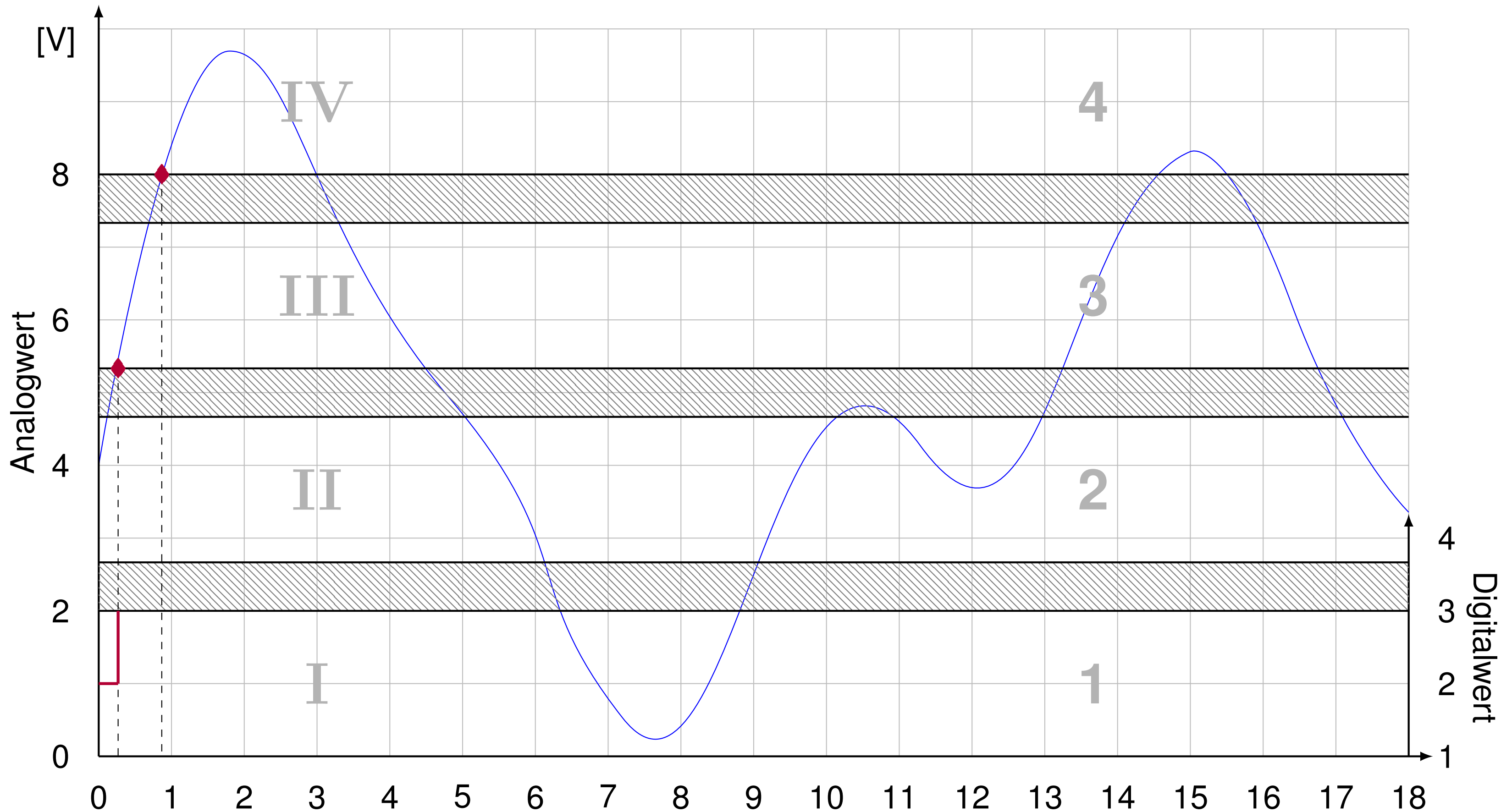


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.

Lösung

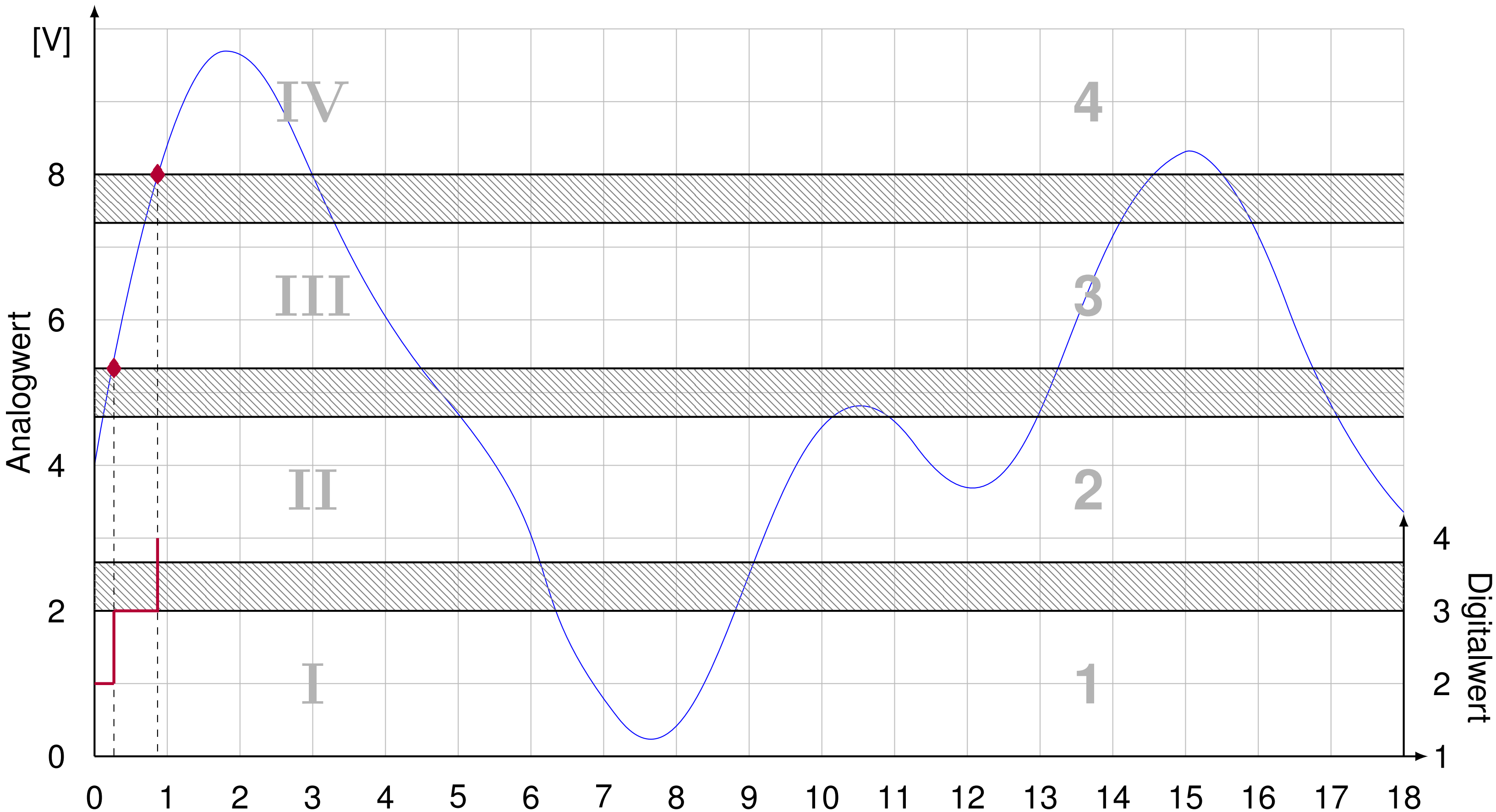


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.

Lösung

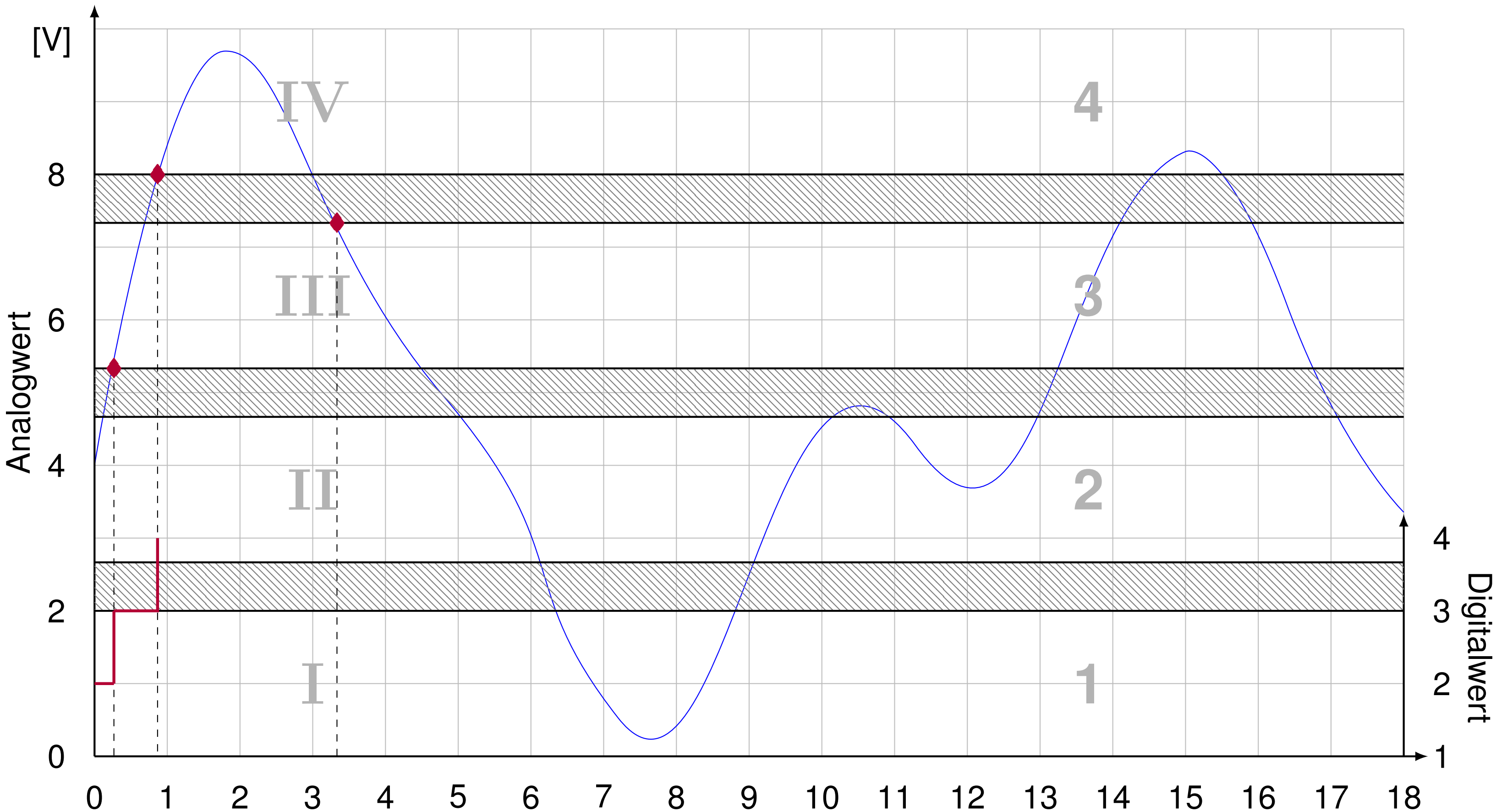


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.

Lösung

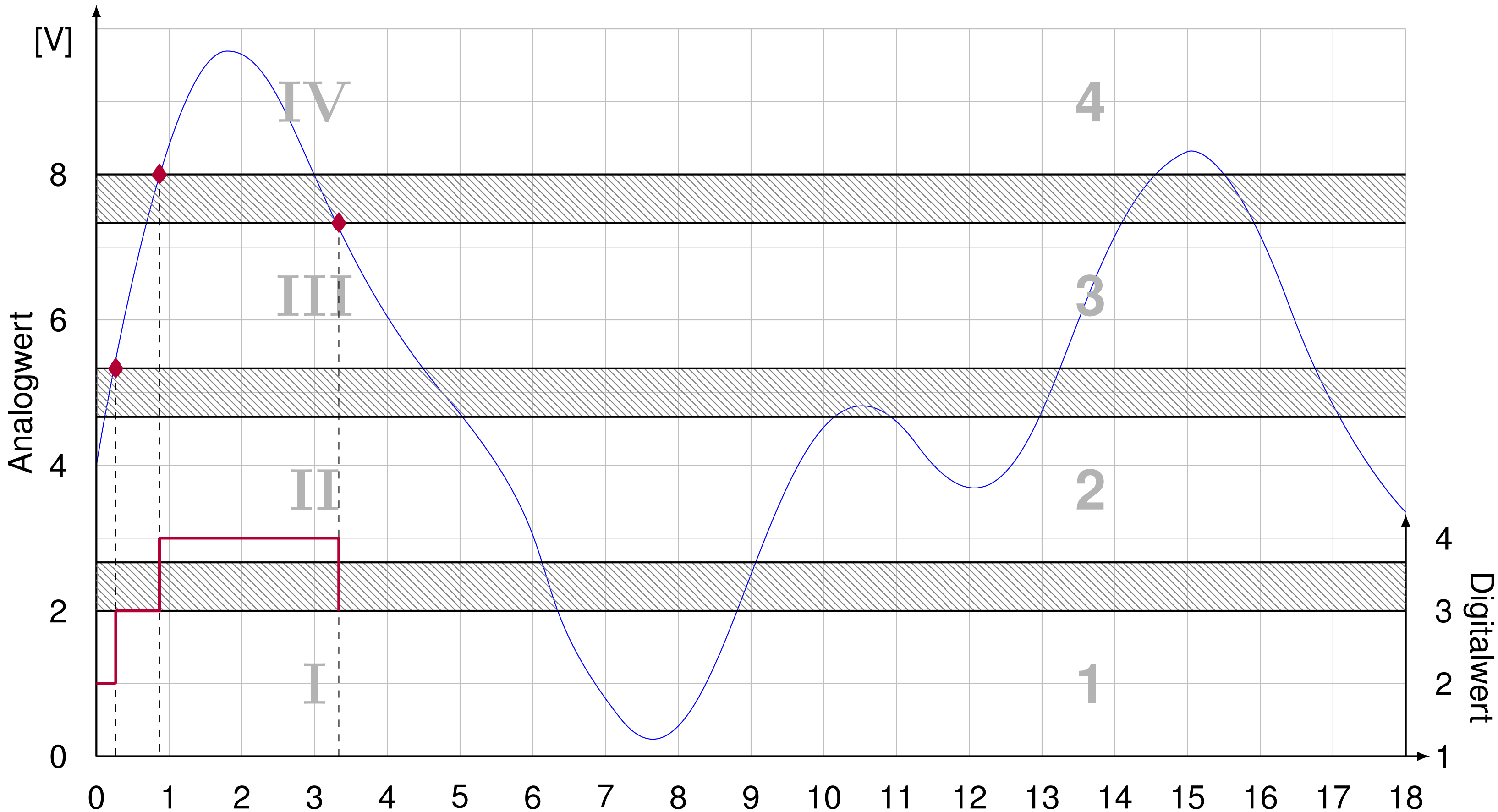


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.

Lösung

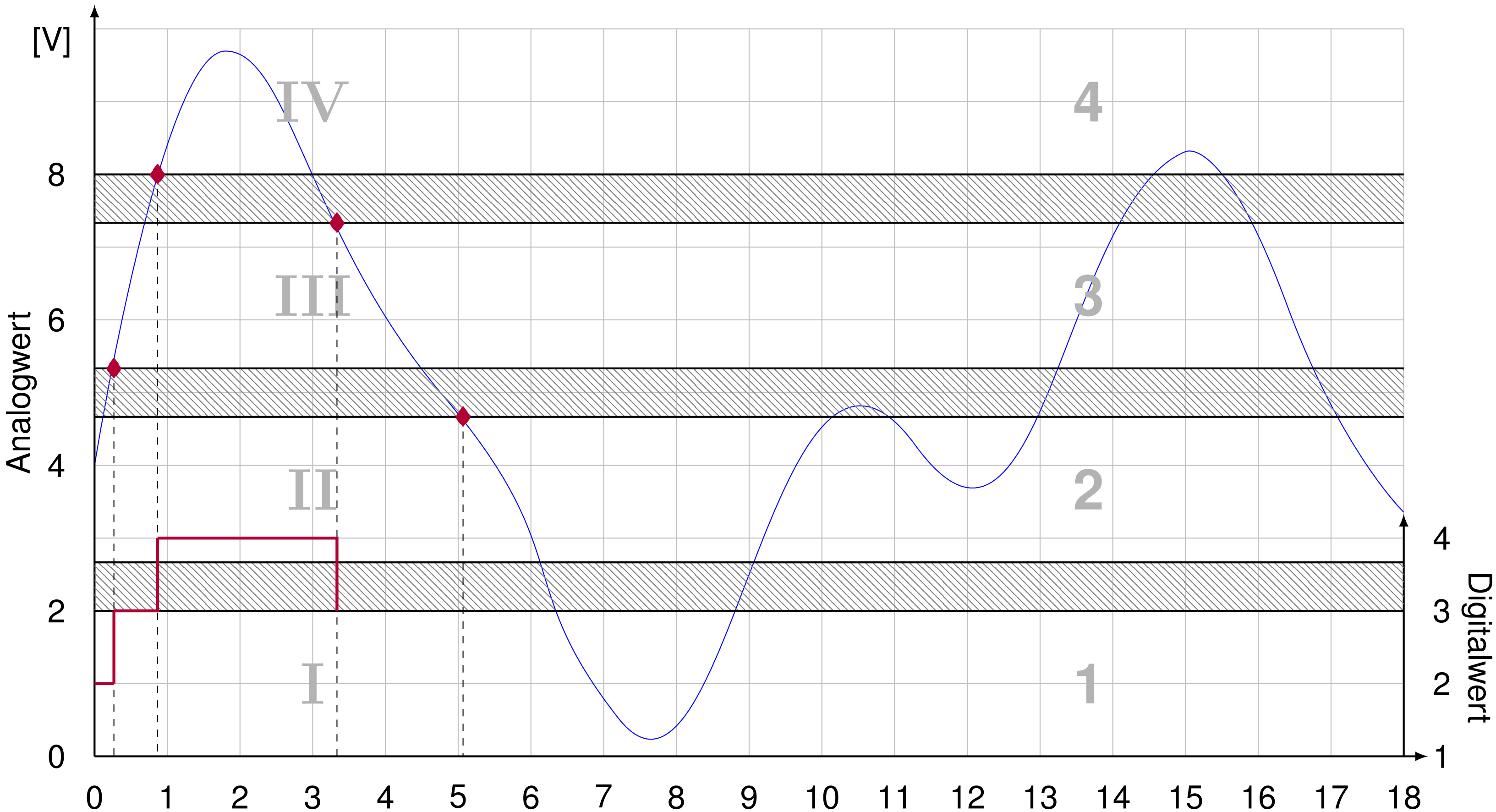


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.

Lösung

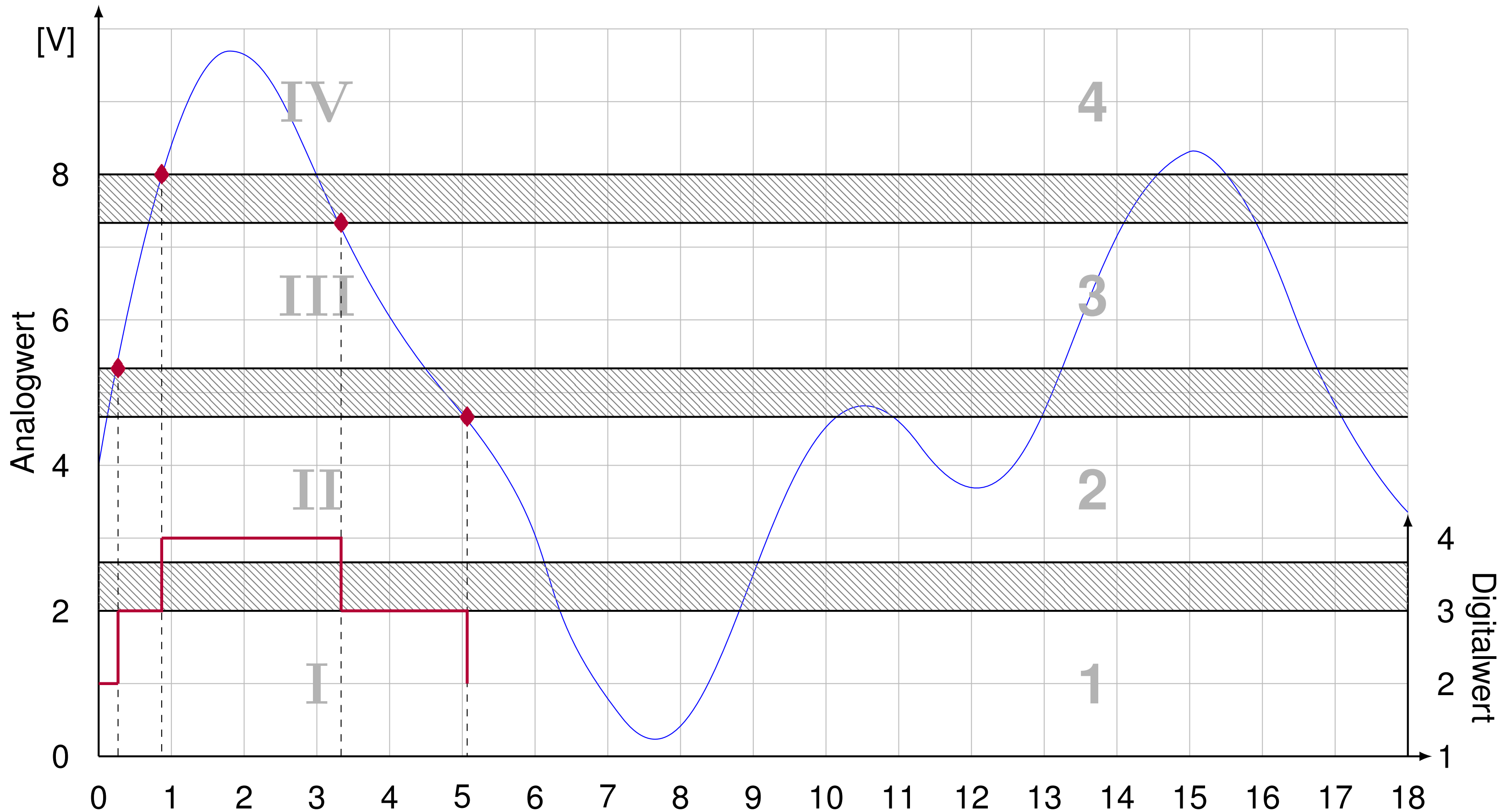


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.

Lösung

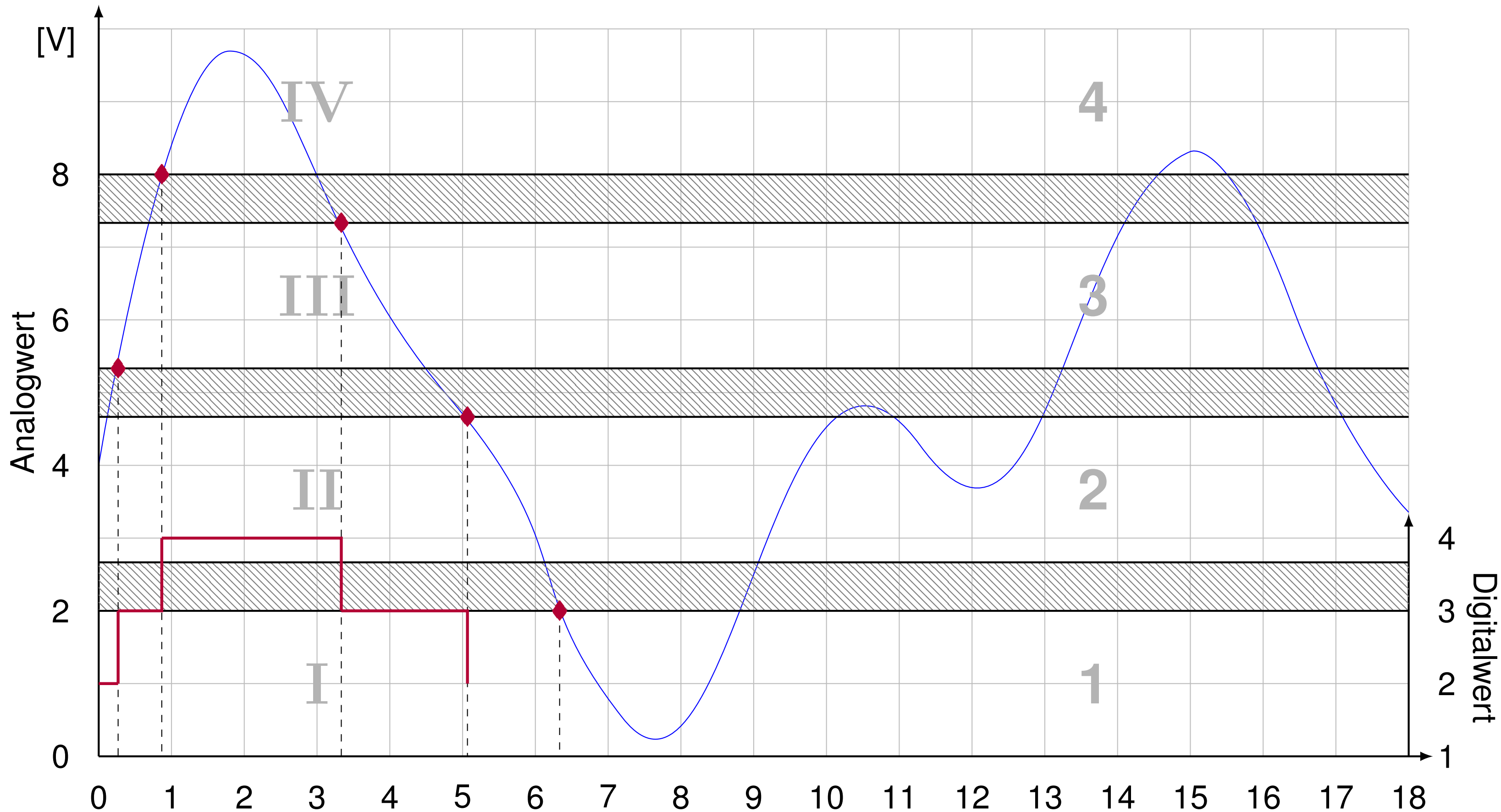


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.

Lösung

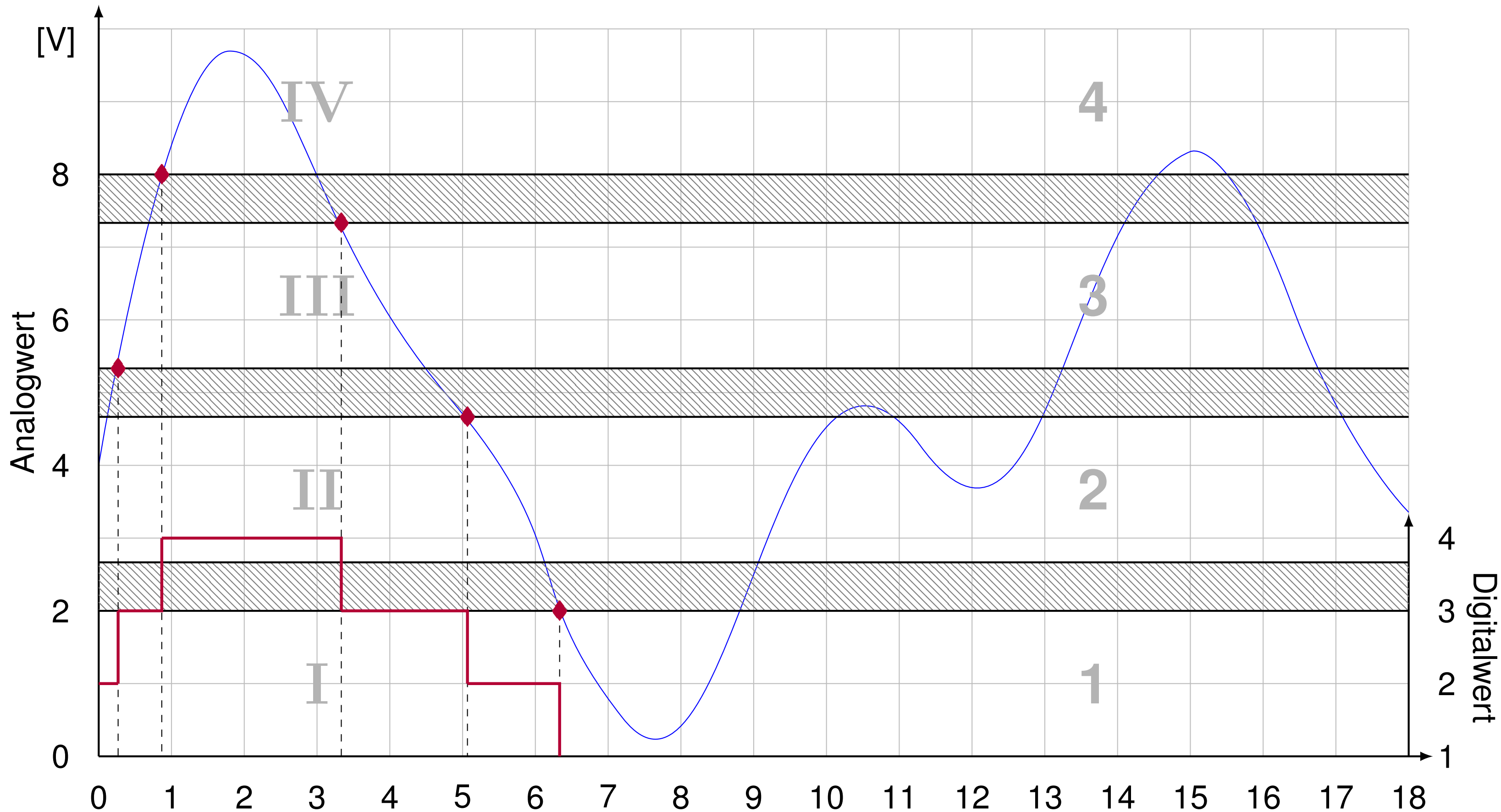


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.

Lösung

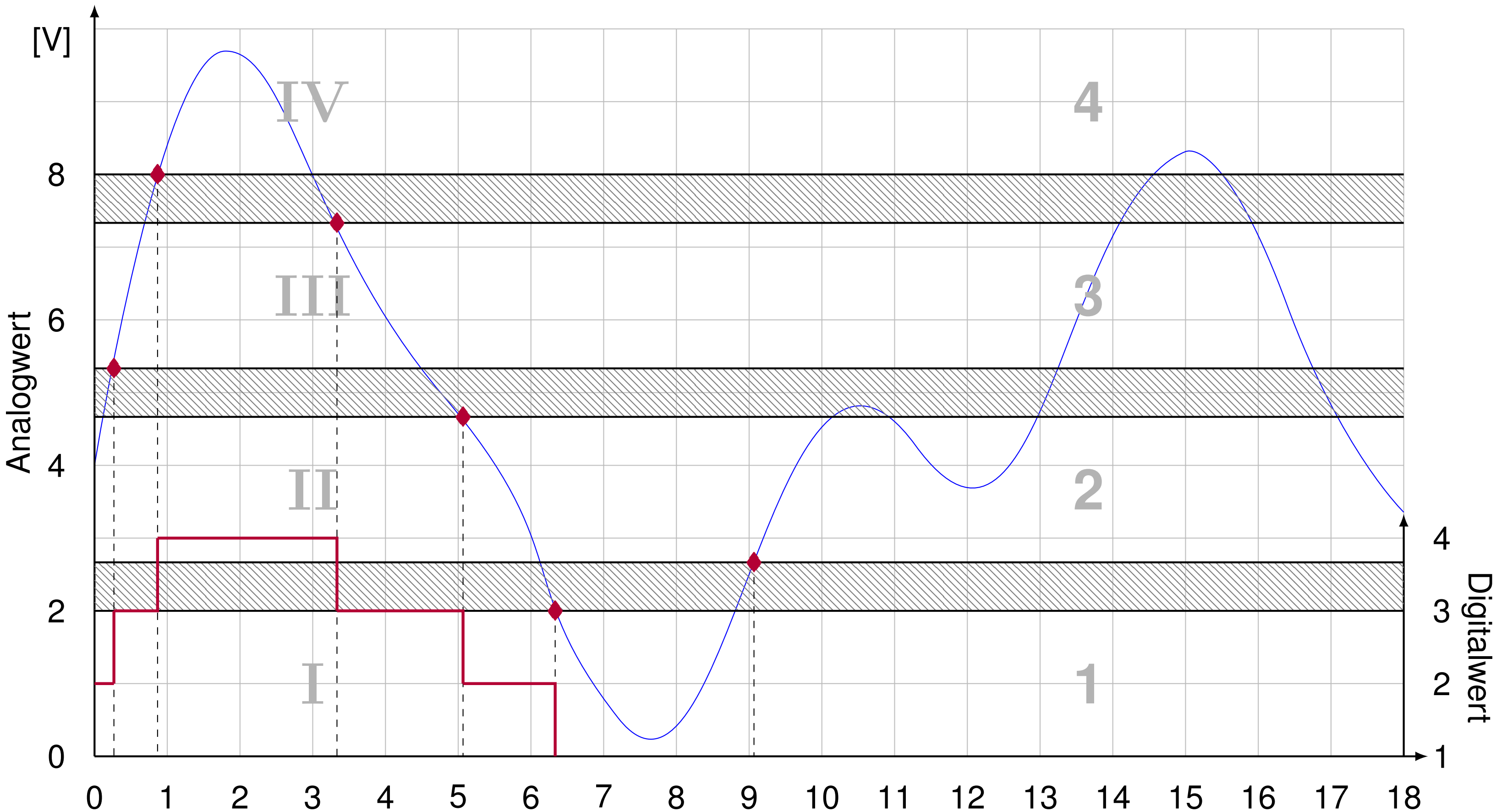


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.

Lösung

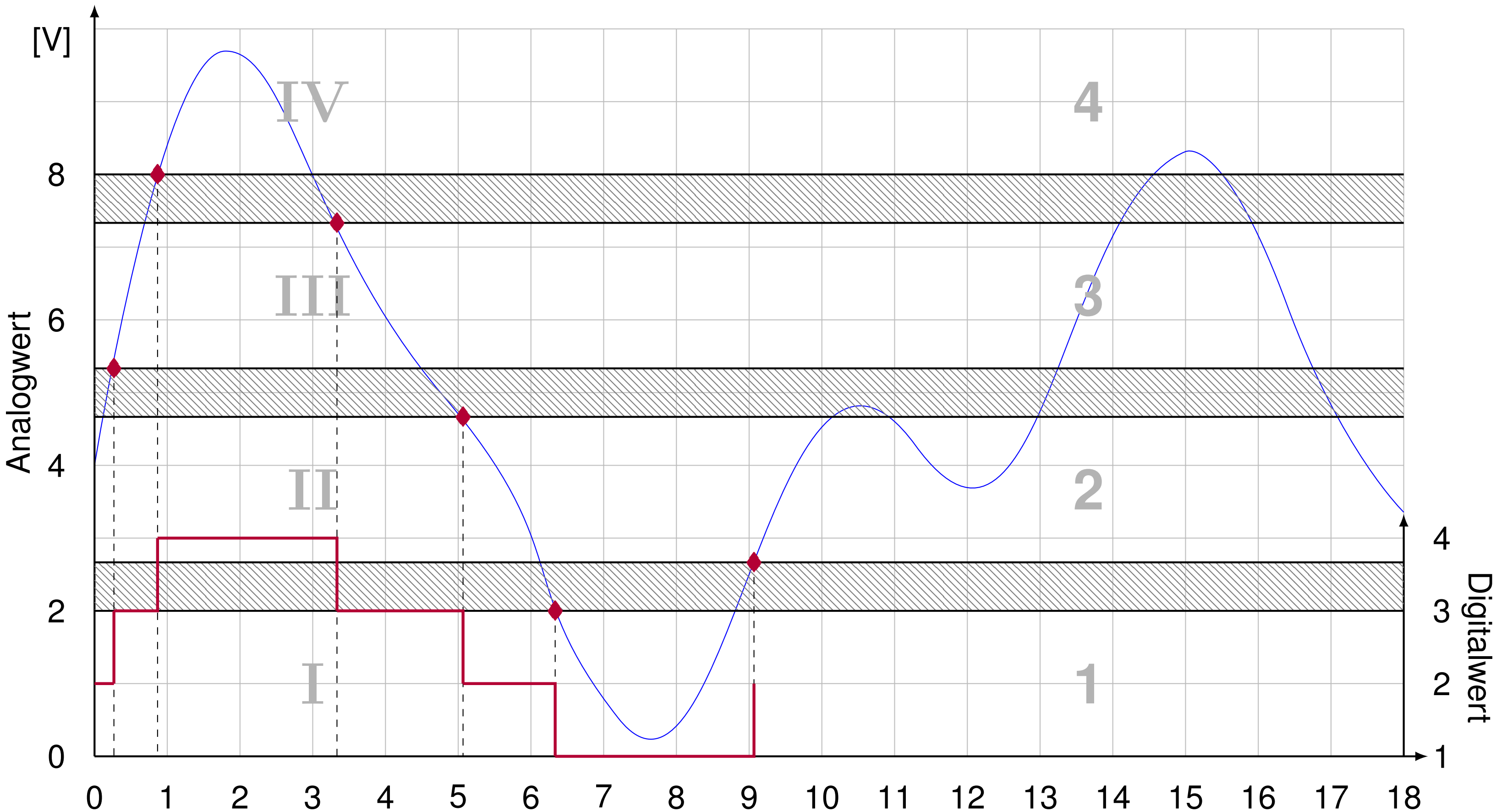


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.

Lösung

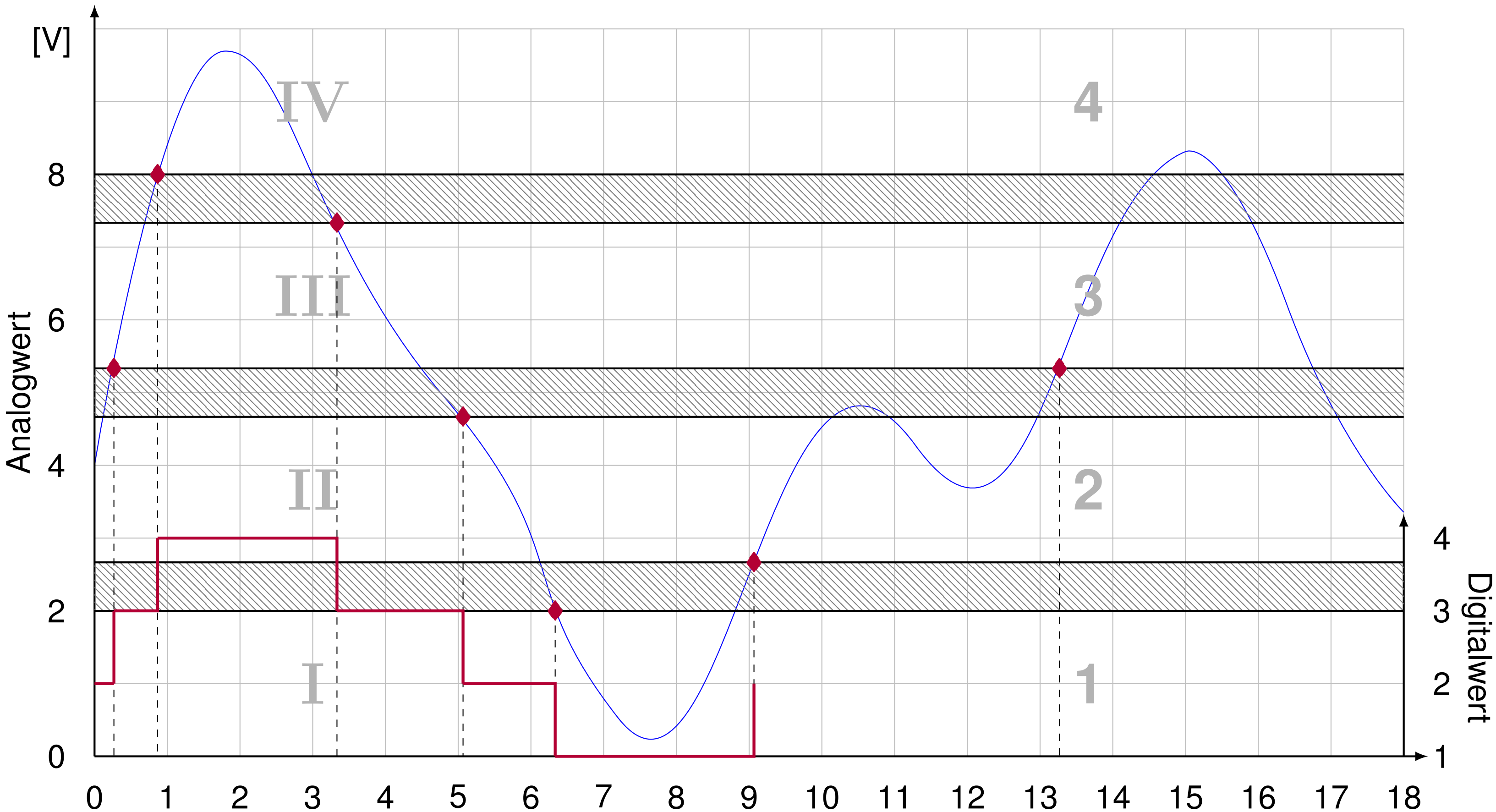


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.

Lösung

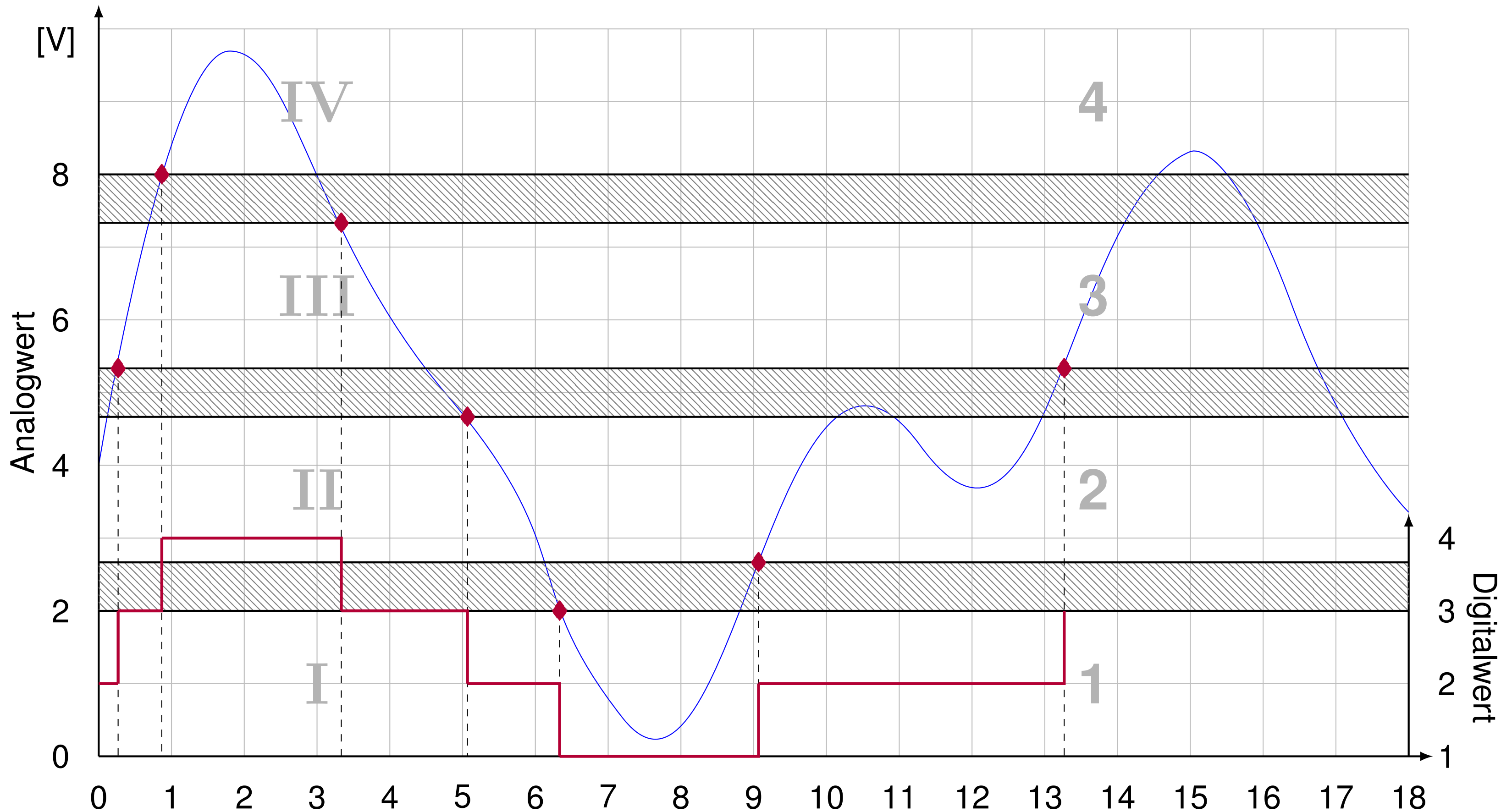


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.

Lösung

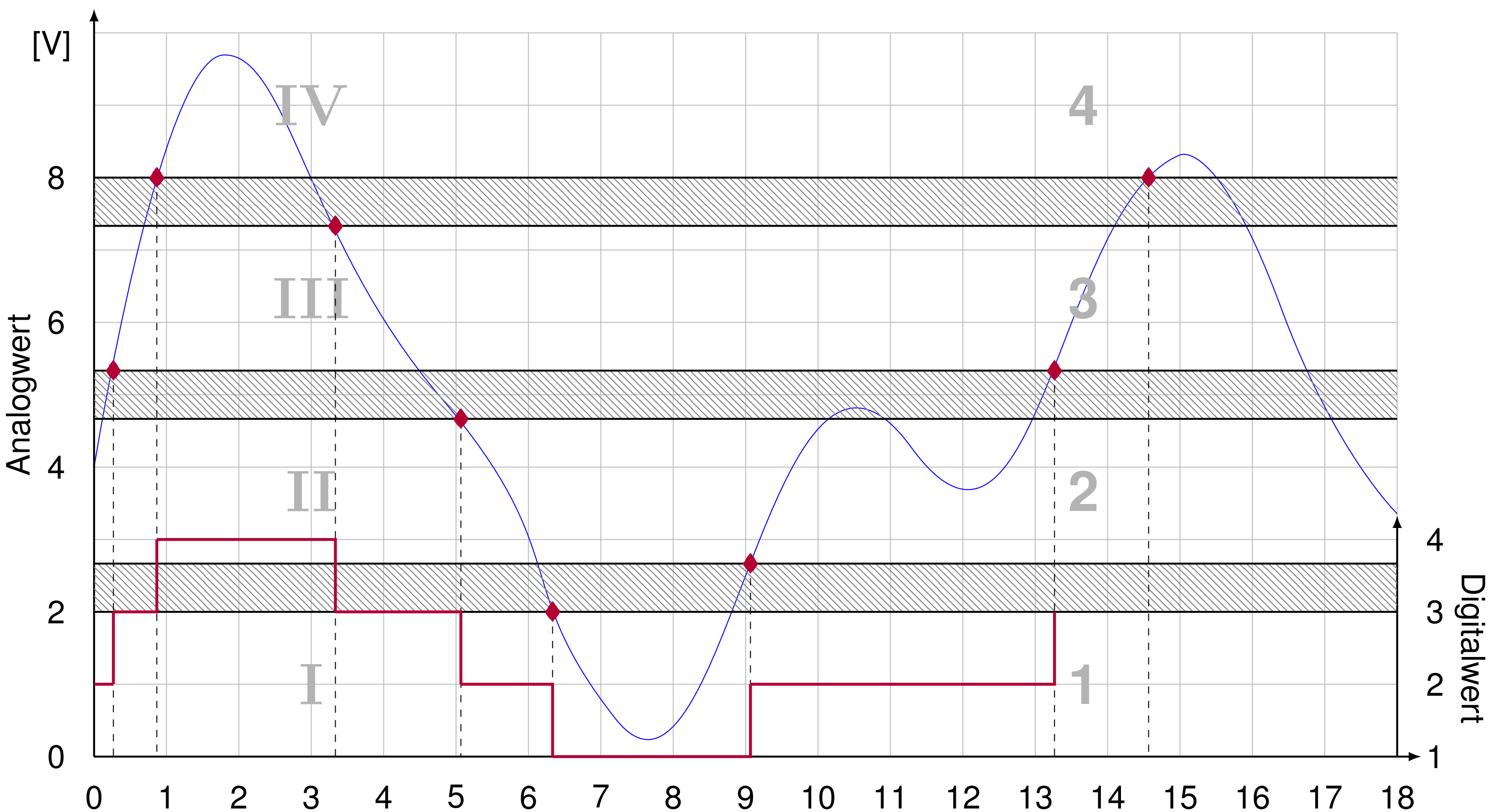


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.

Lösung

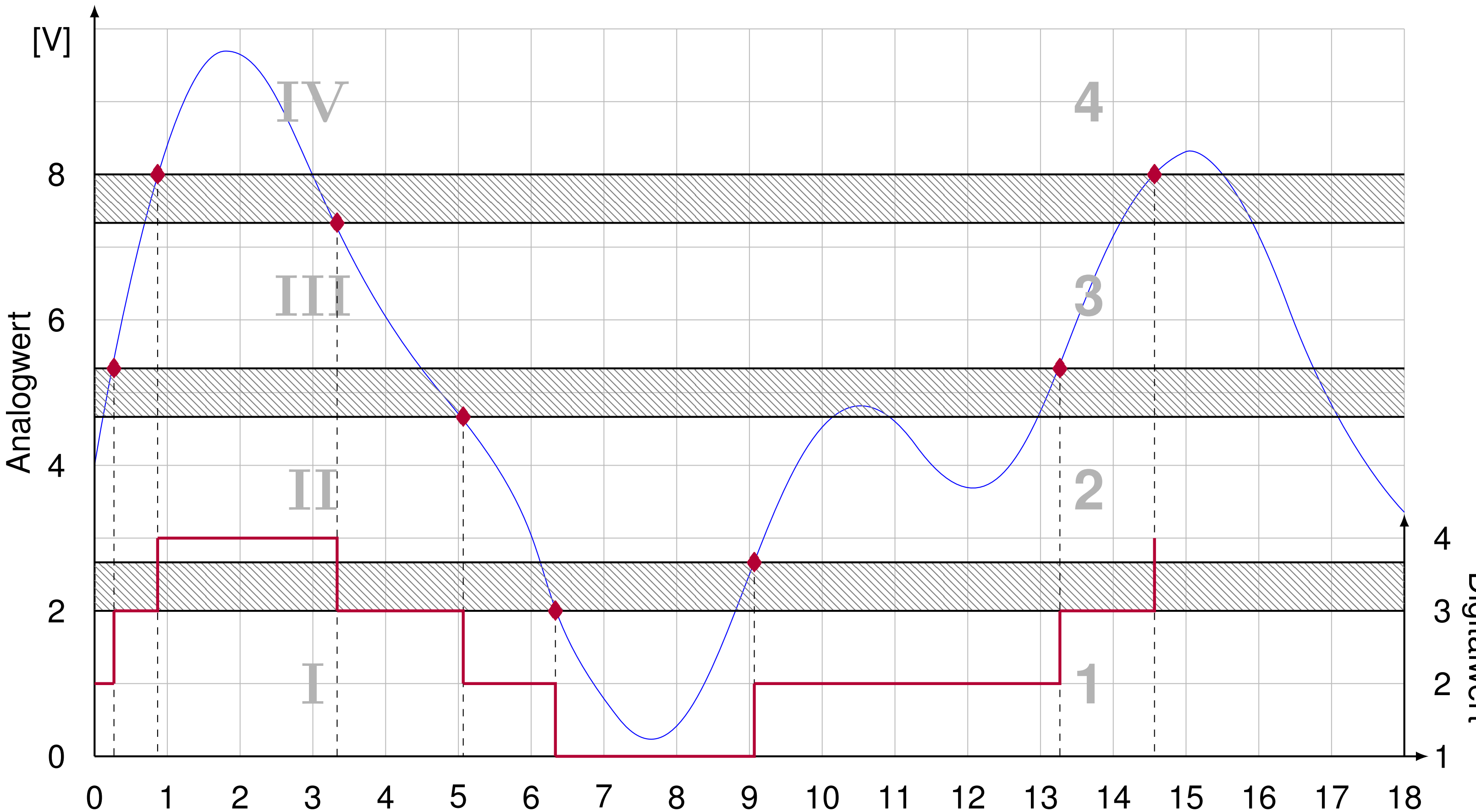


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.

Lösung

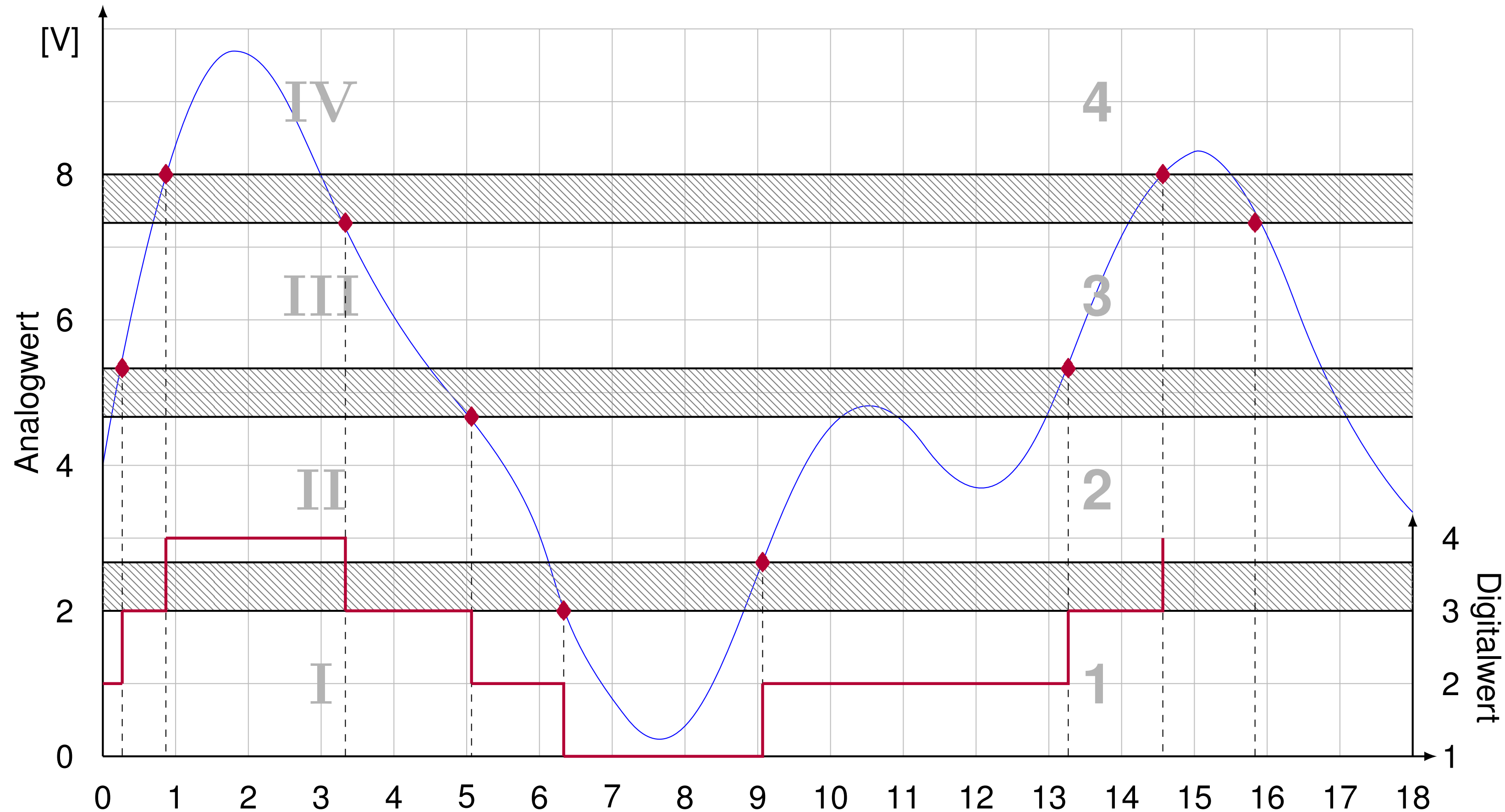


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.

Lösung

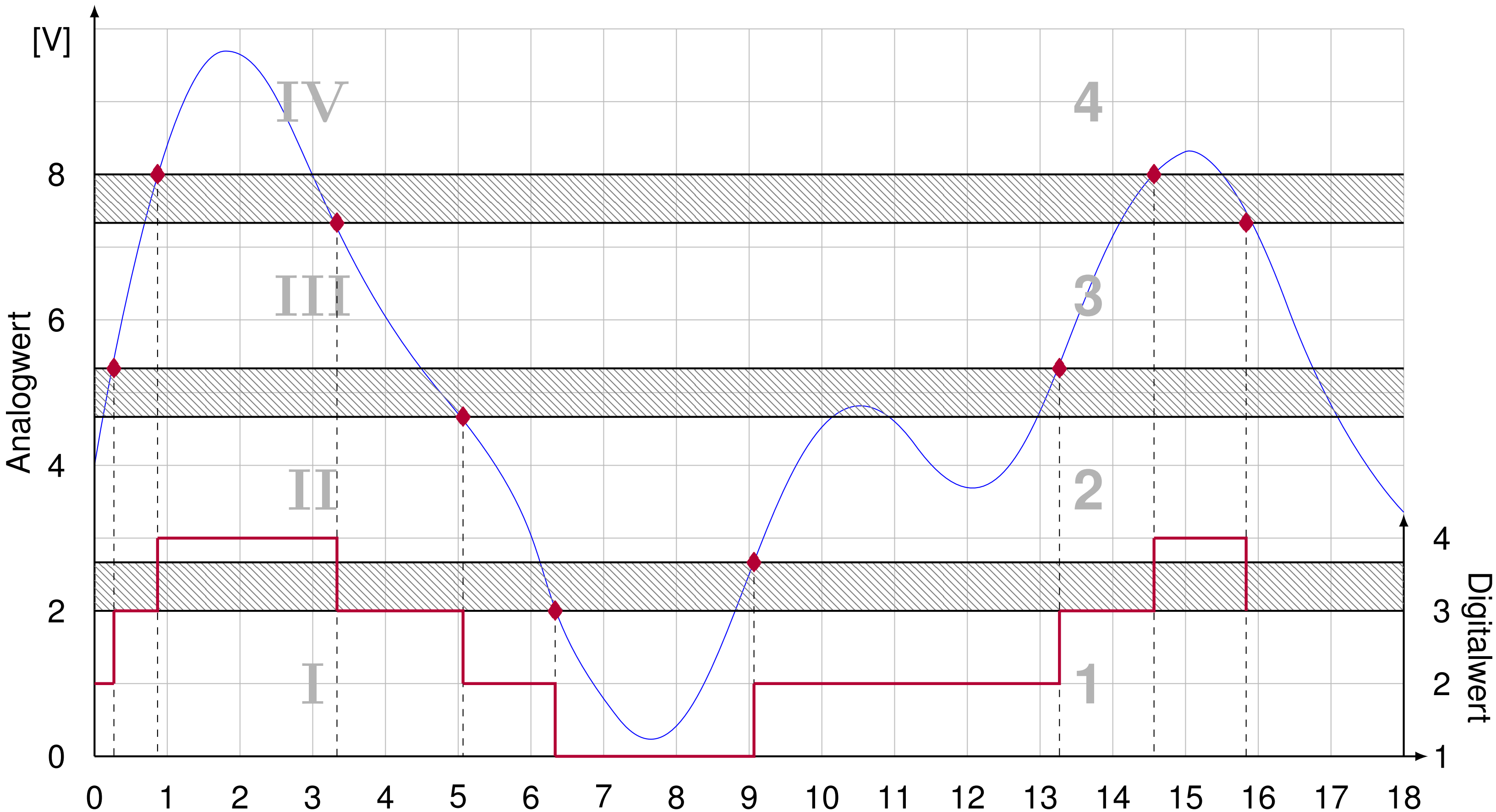


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.

Lösung

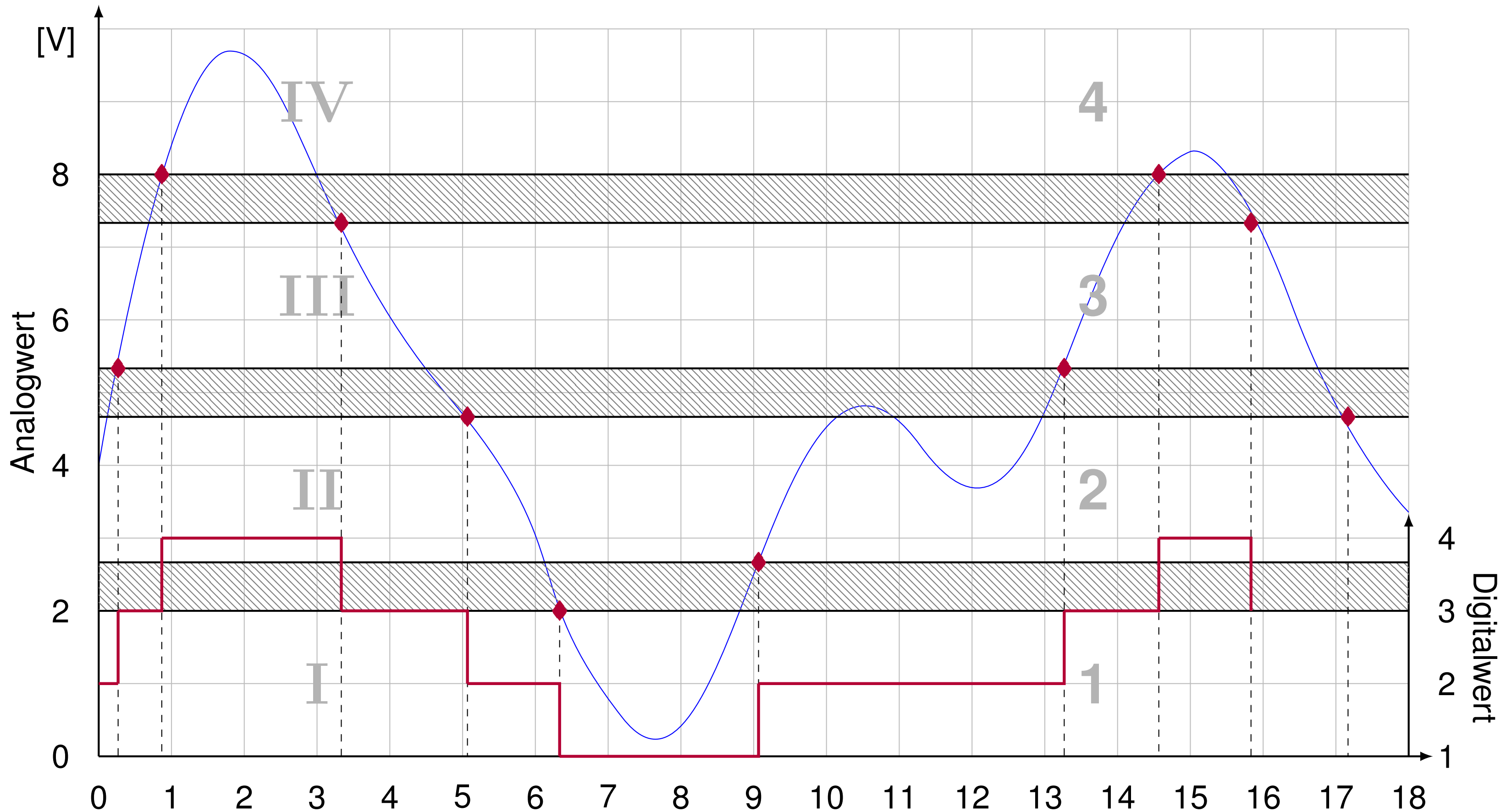


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.

Lösung

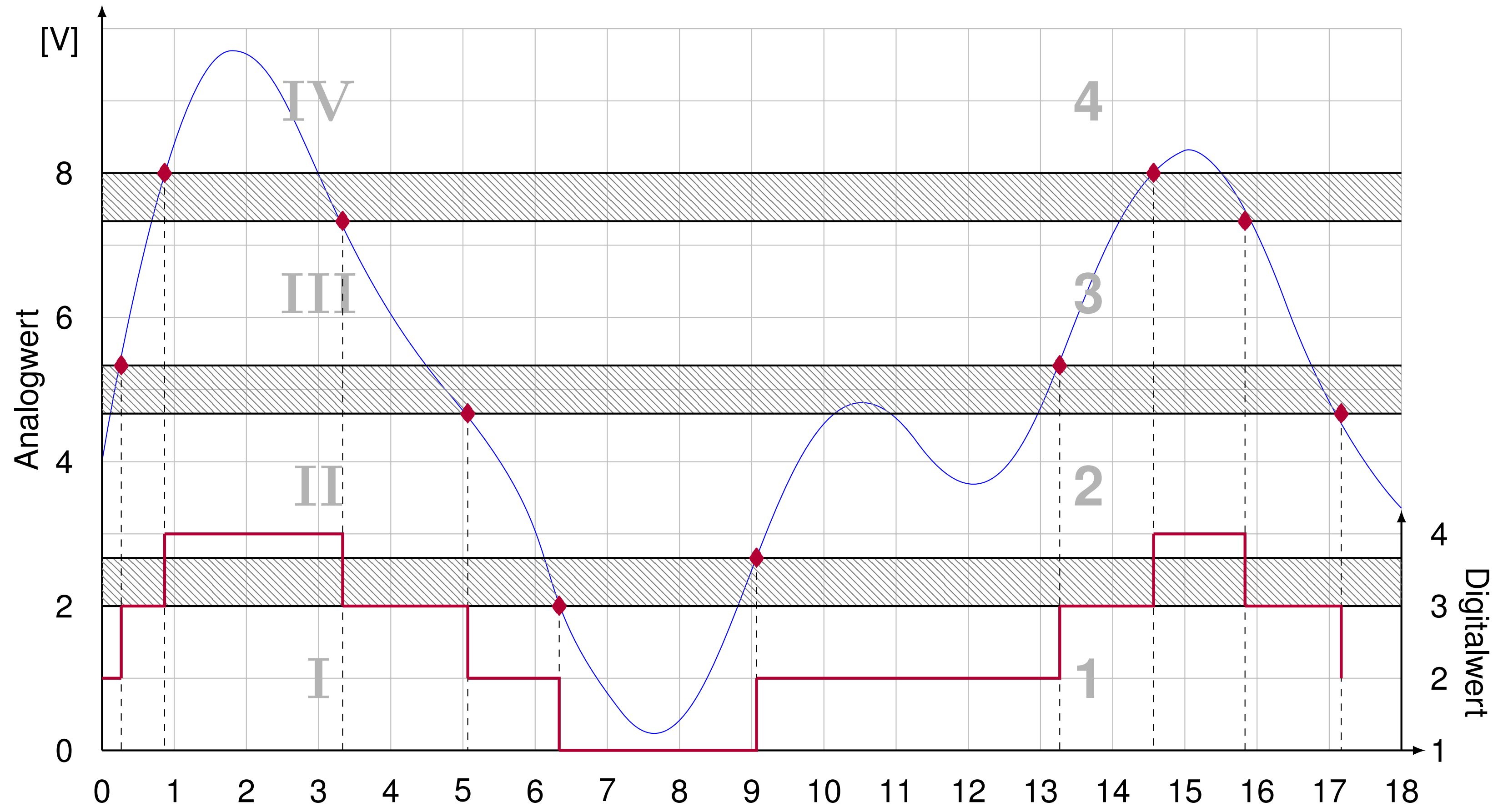


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie zuerst eine Wertdiskretisierung durch und zeichnen Sie das Ergebnis in das unten vorgegebene Diagramm ein.

Lösung

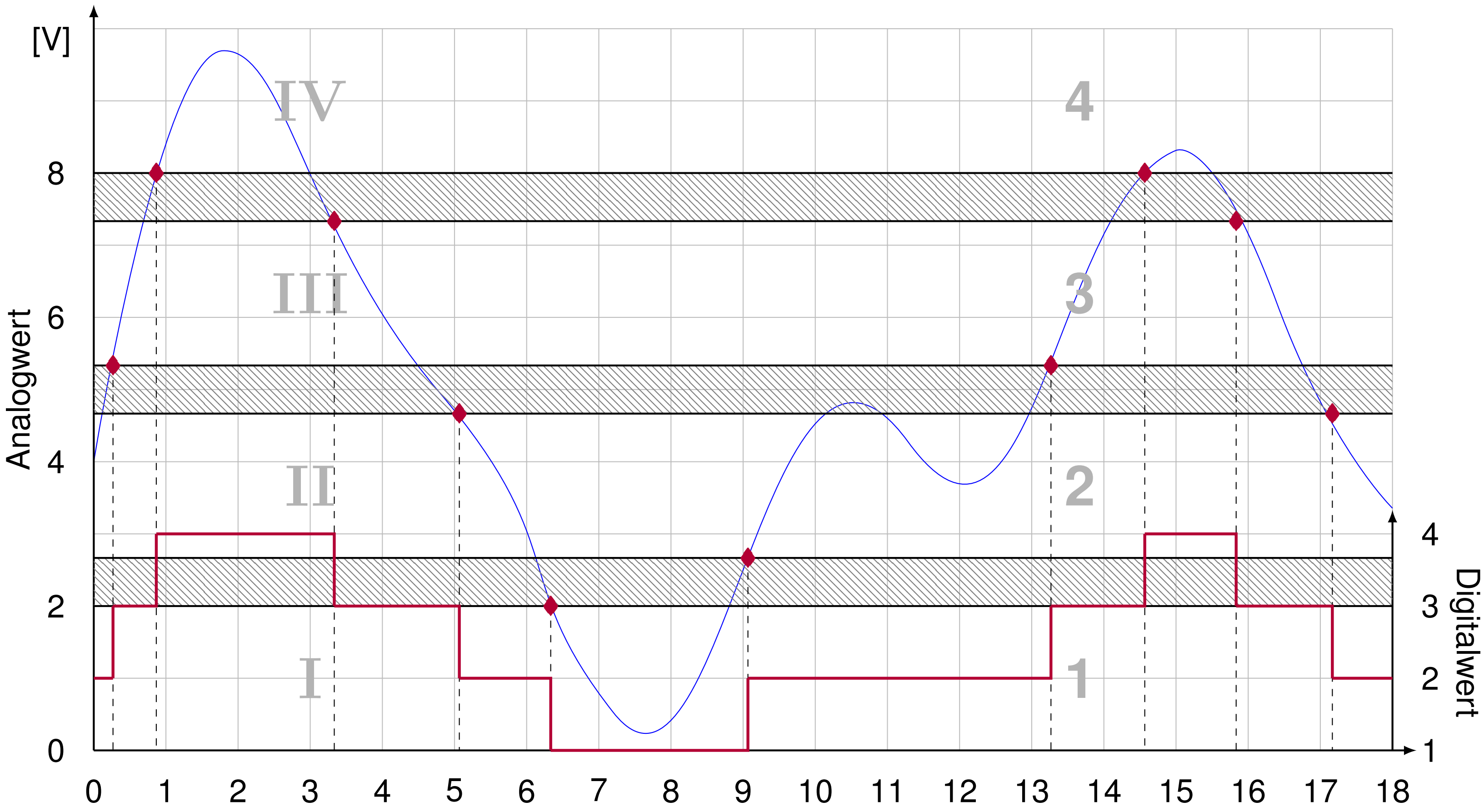


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.

Lösung

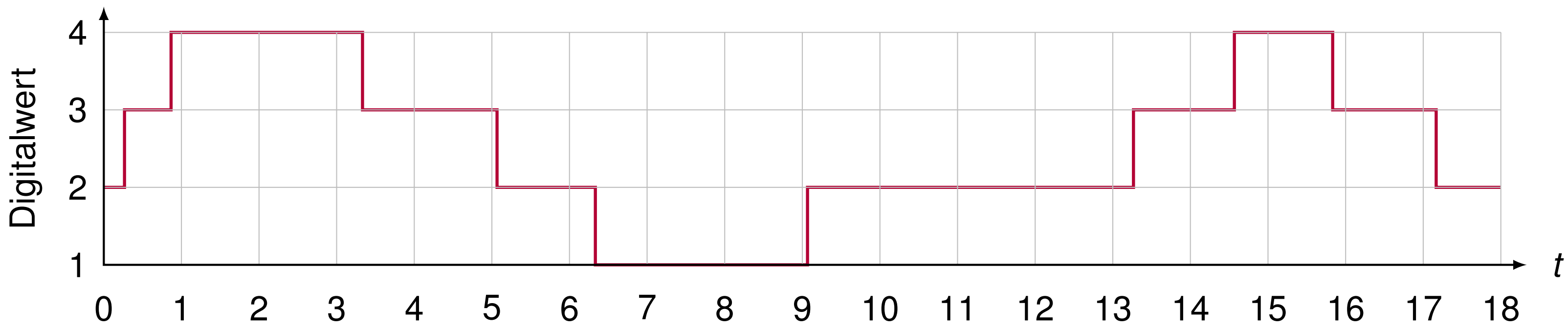


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

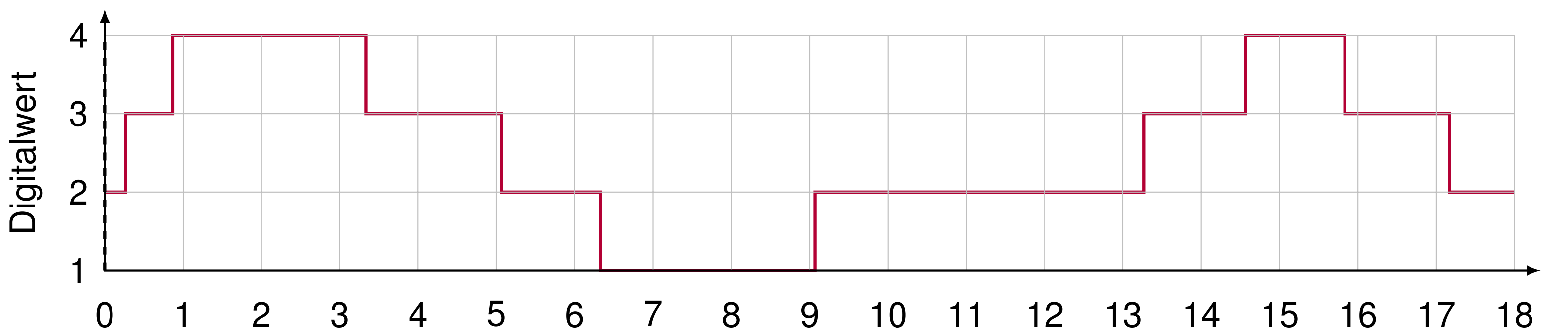


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

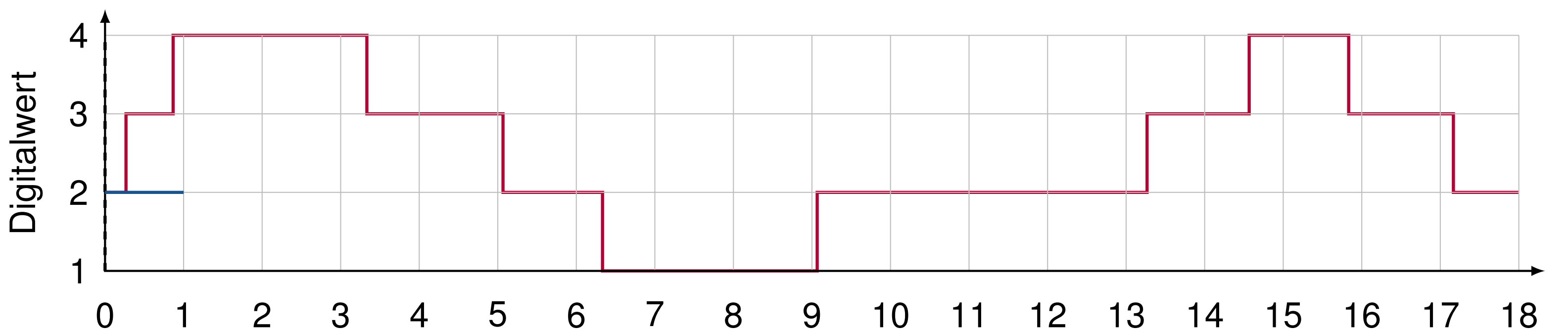


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

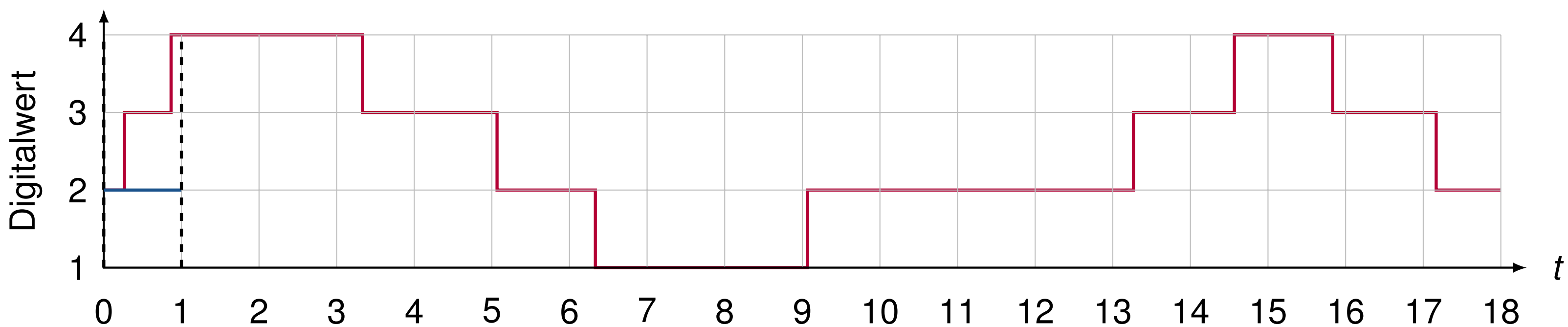


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

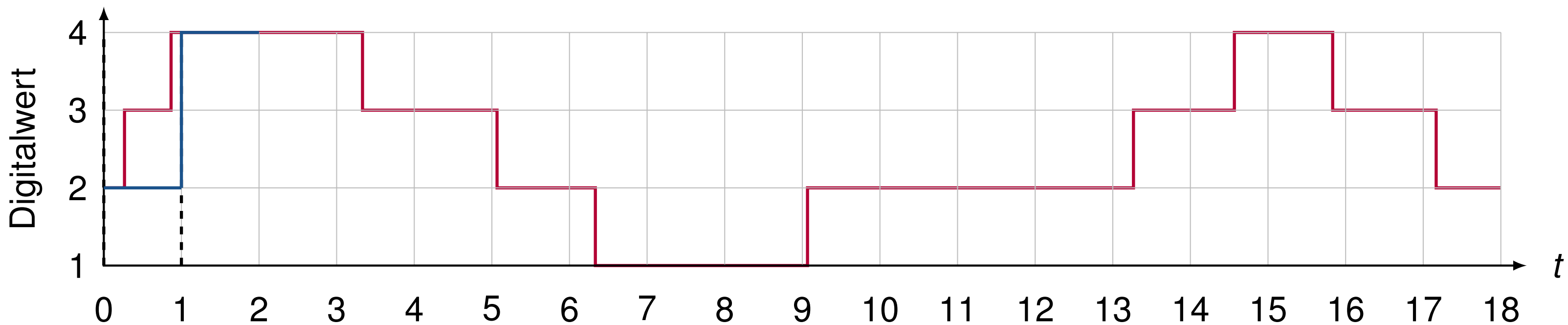


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

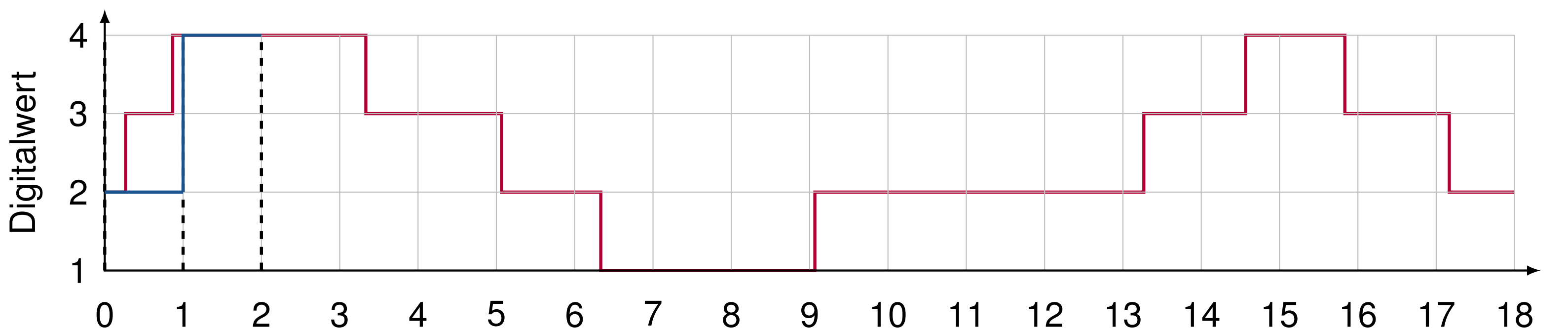


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

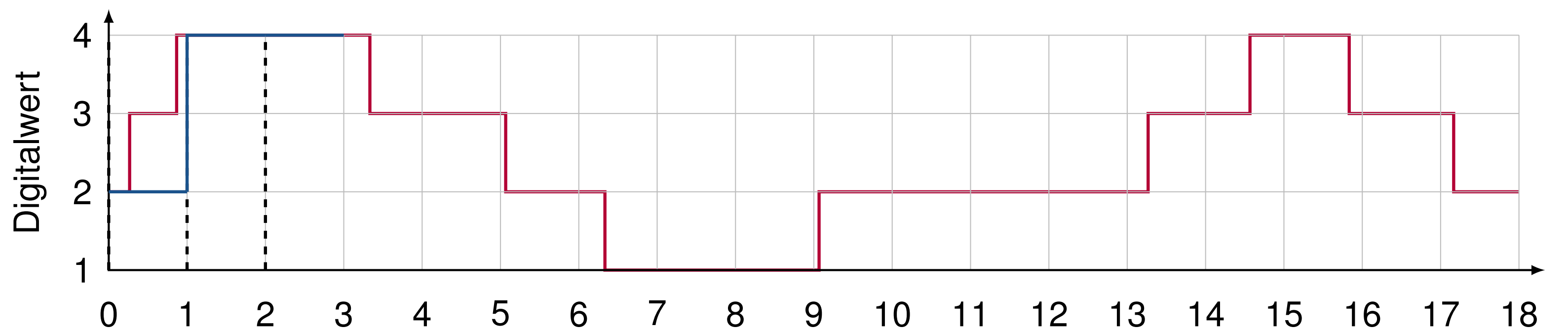


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.

Lösung

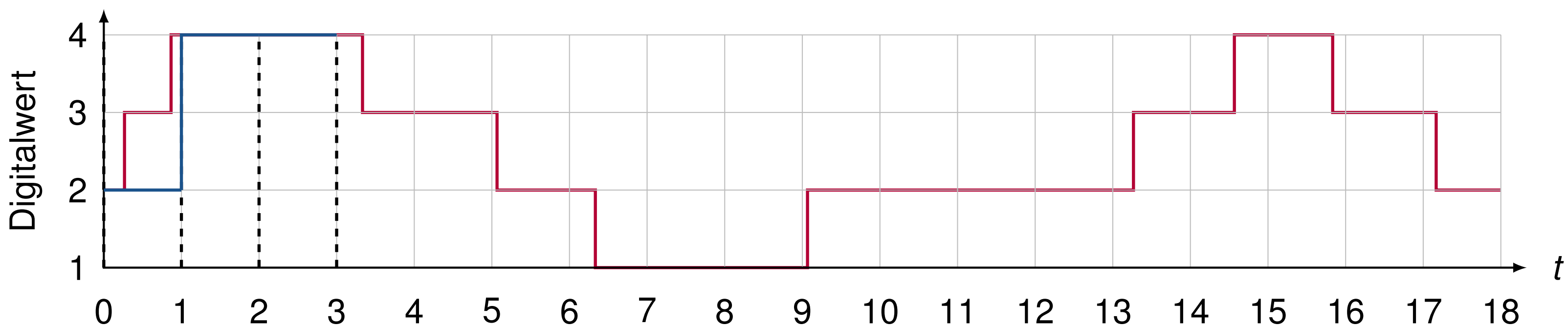


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

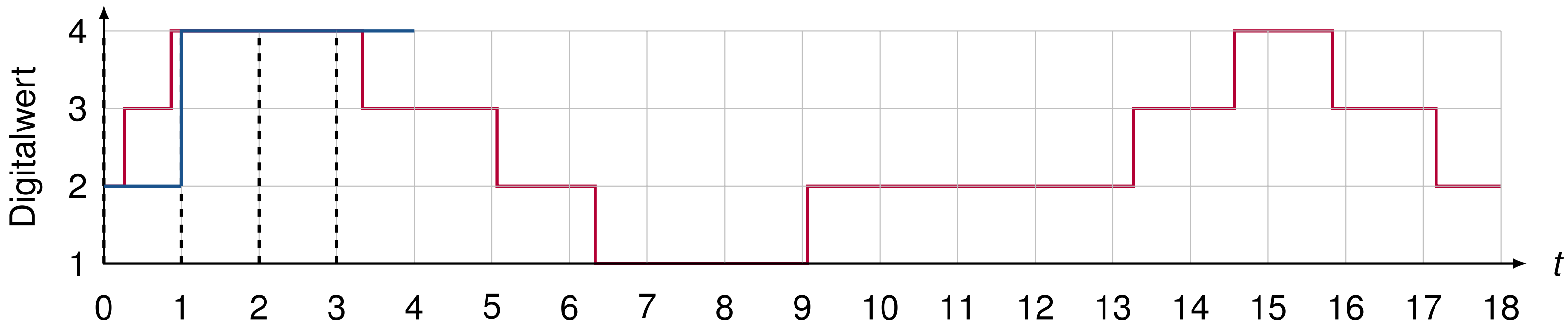


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

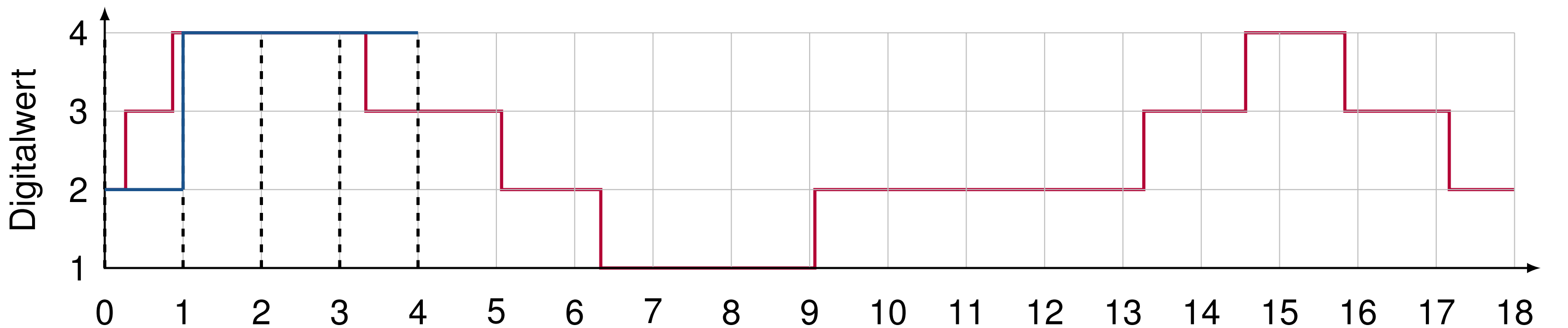


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

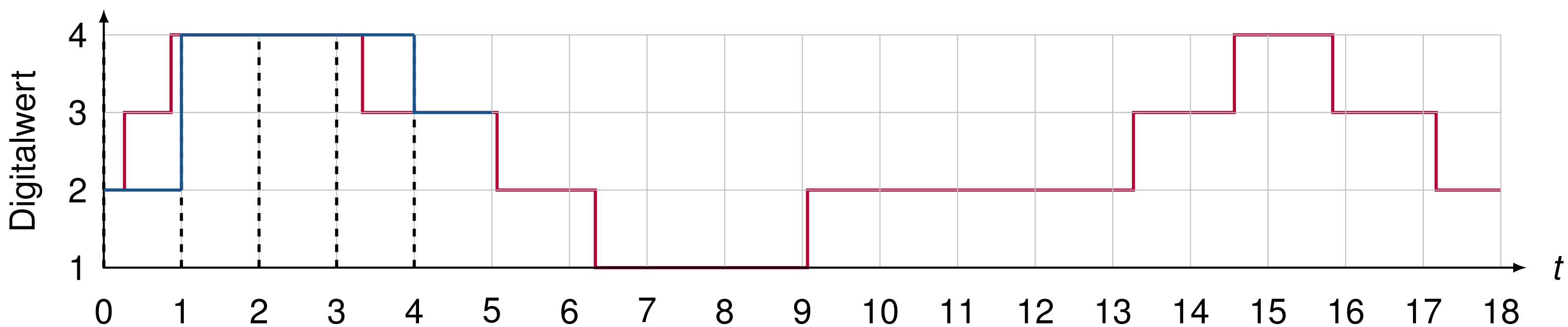


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

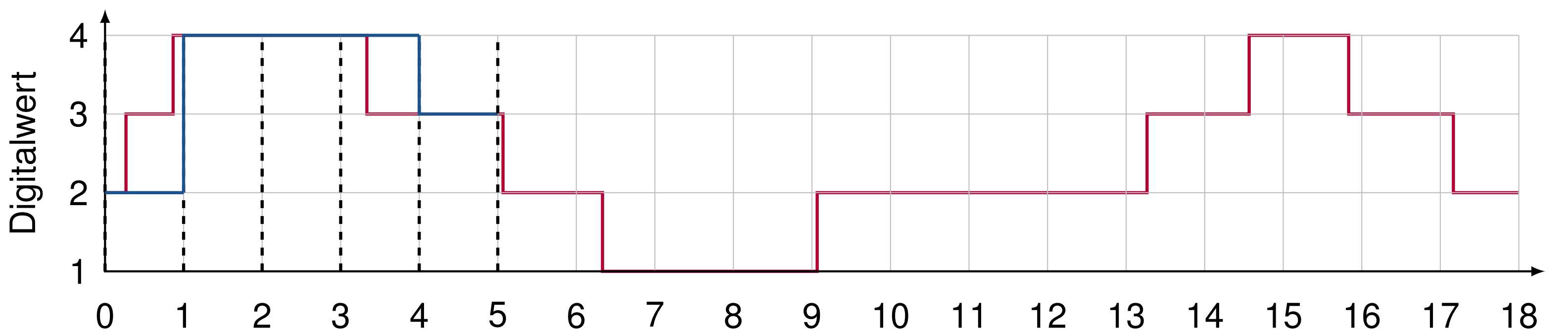


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

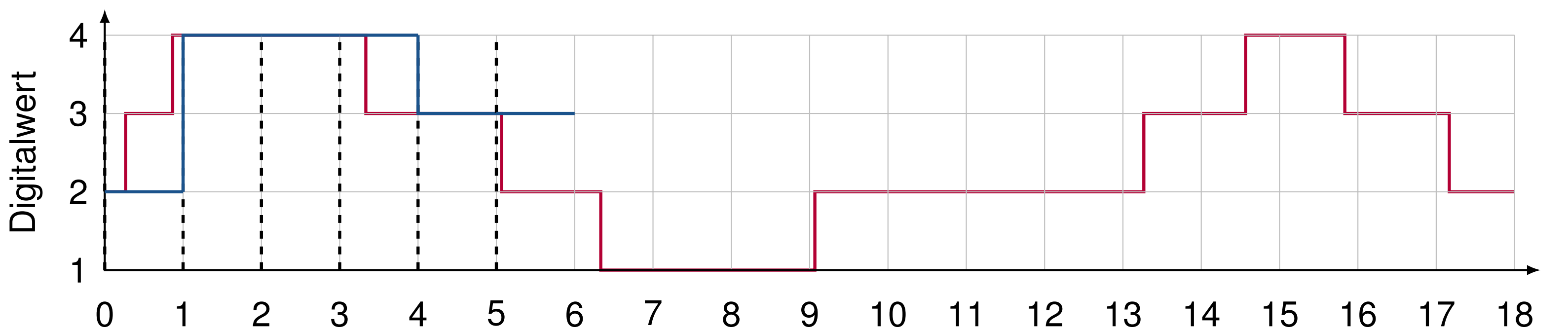


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

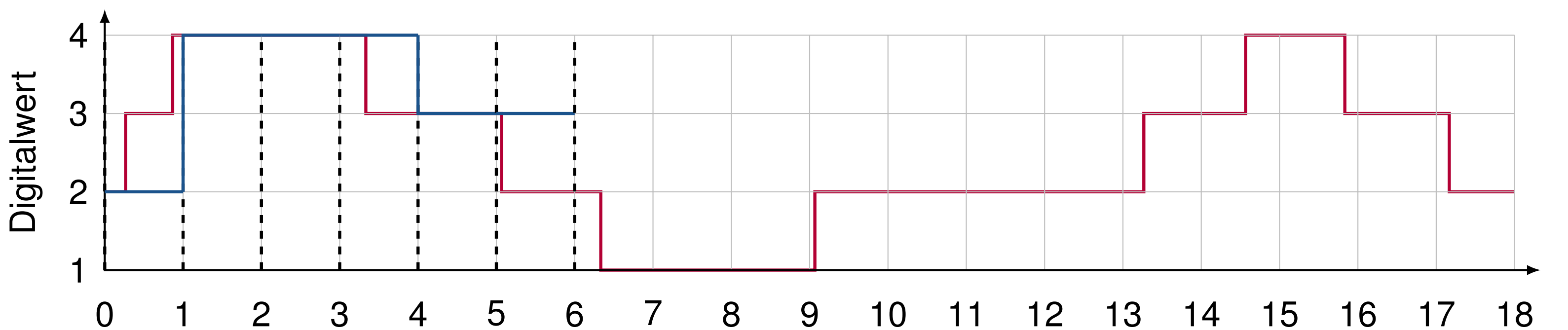


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

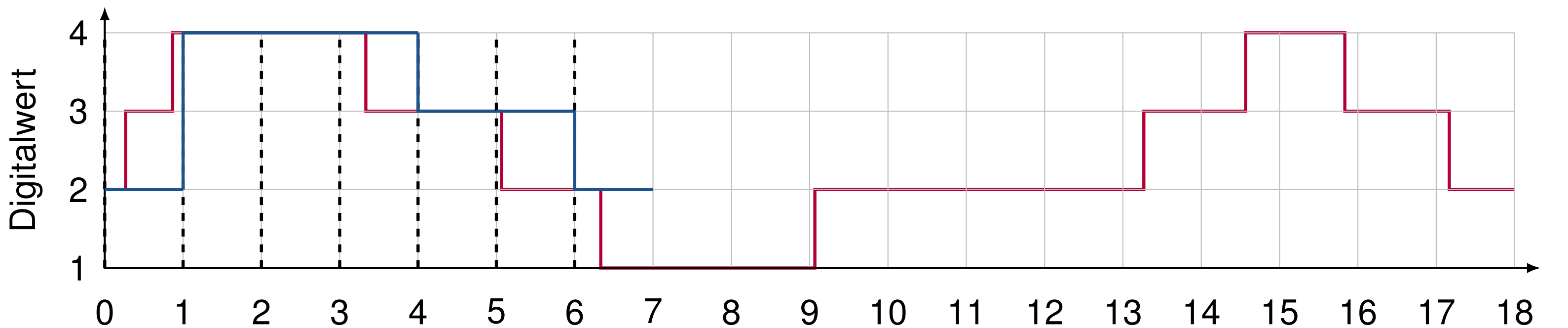


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

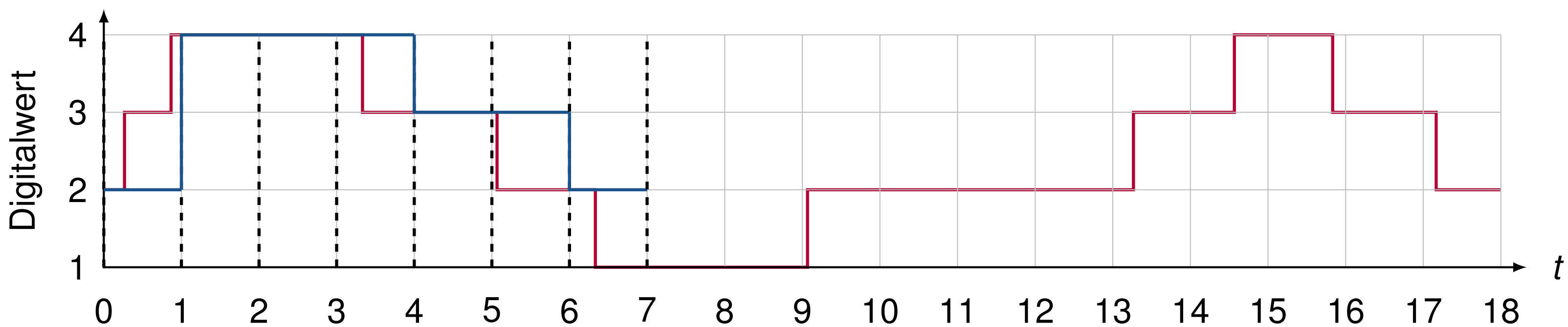


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

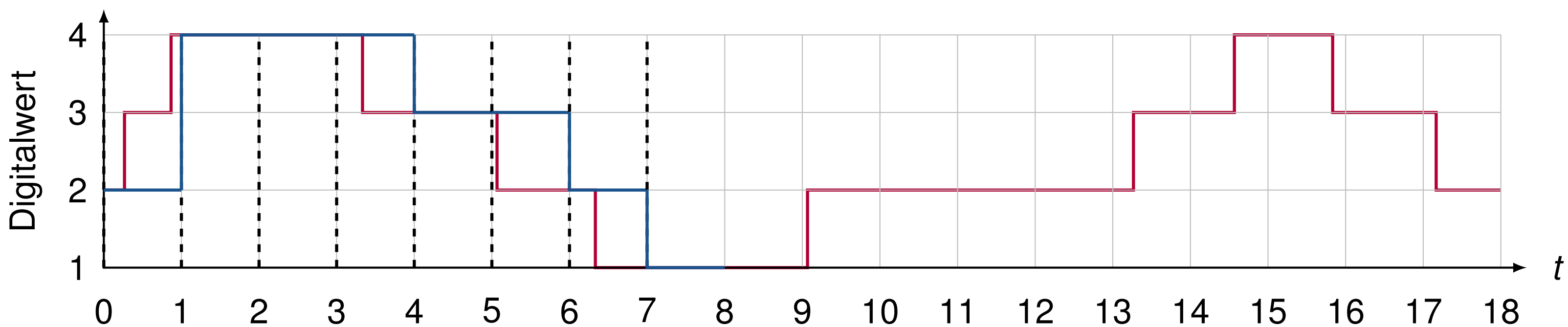


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

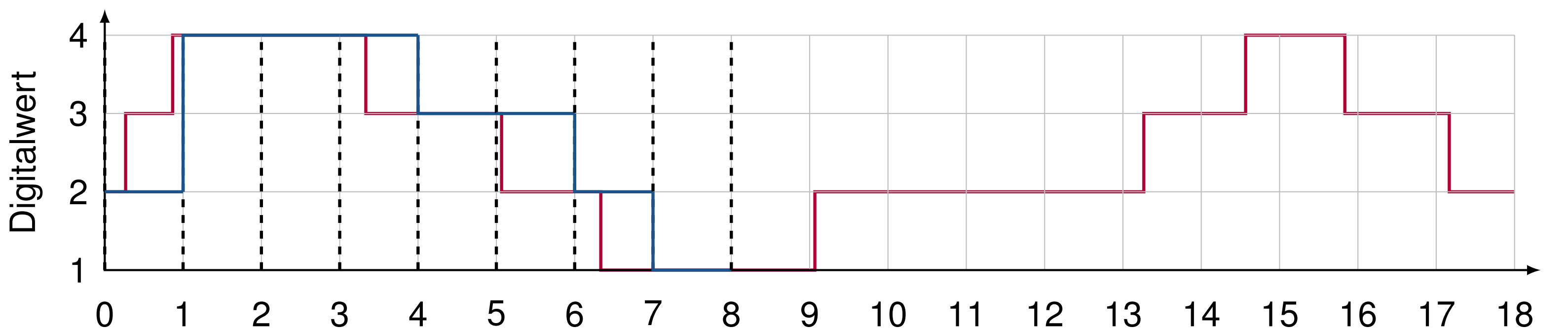


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

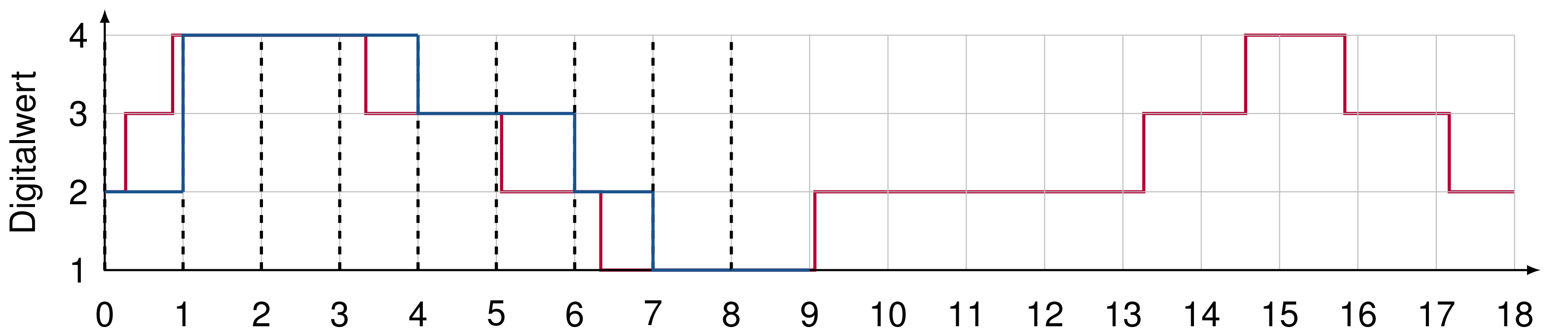


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.

Lösung

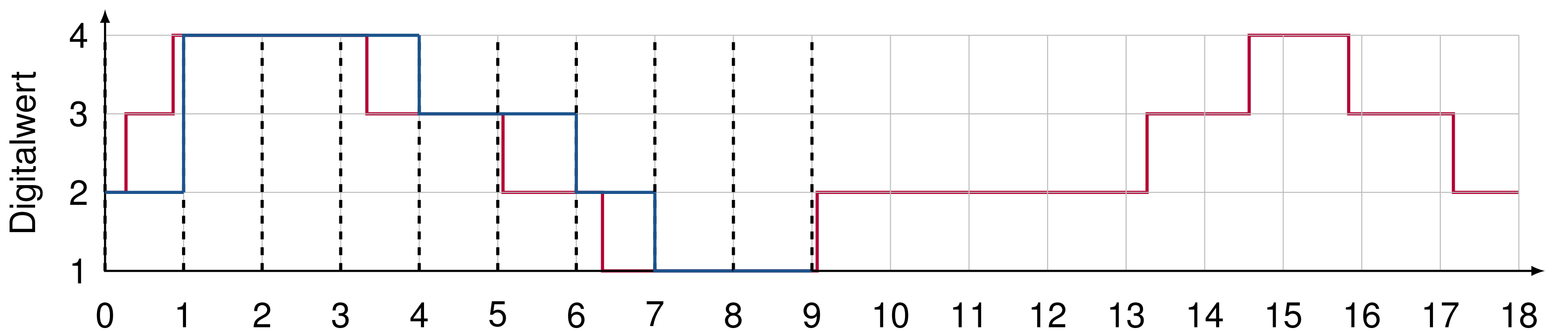


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

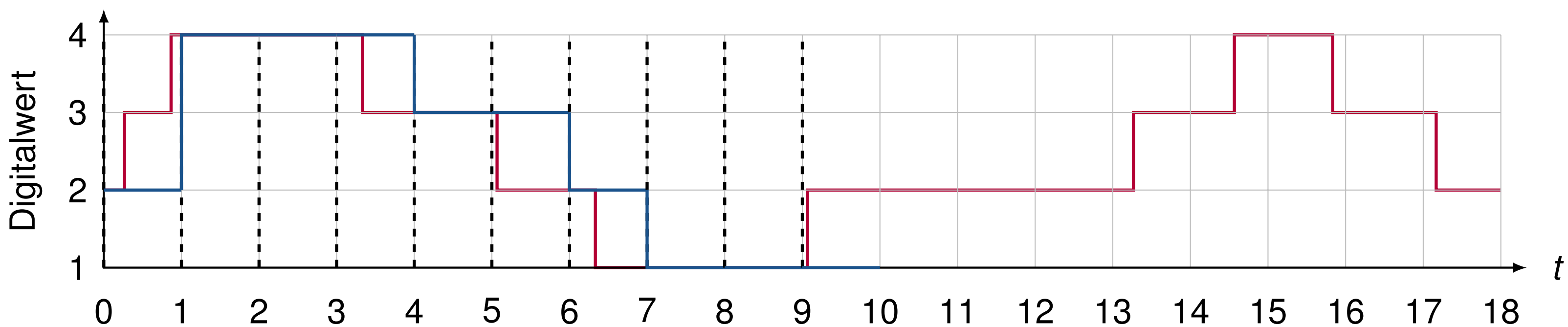


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

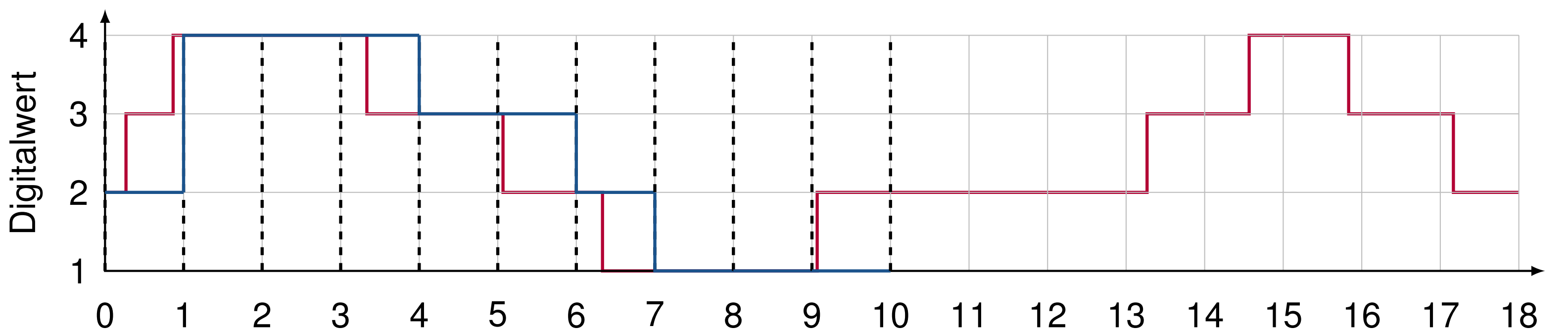


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

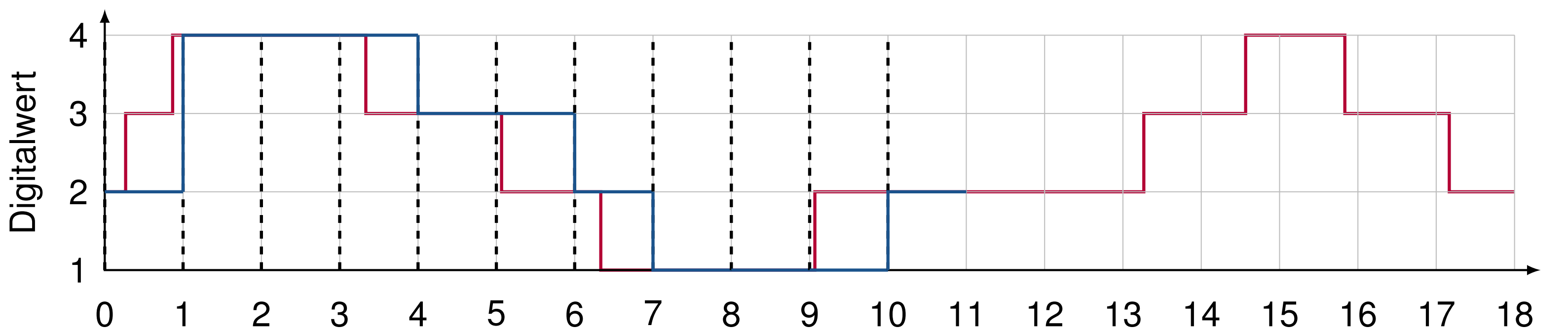


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

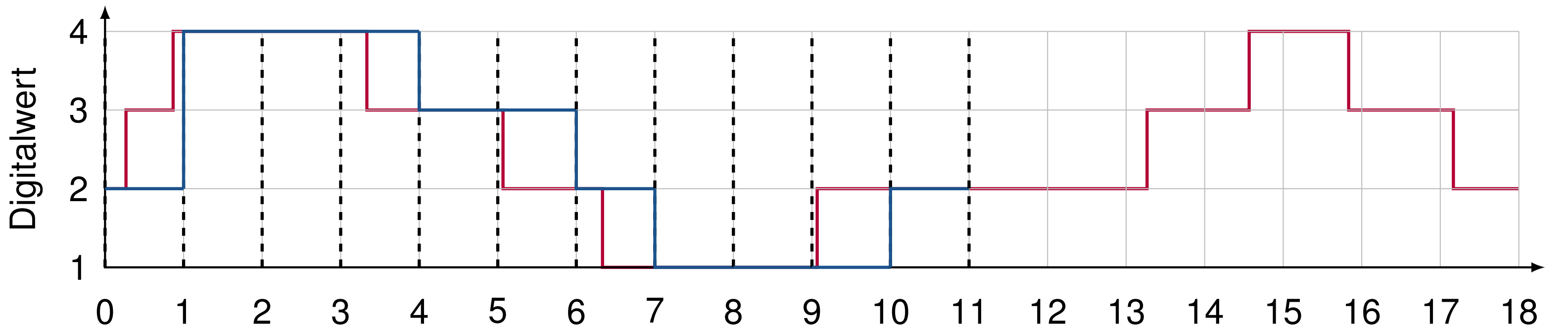


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

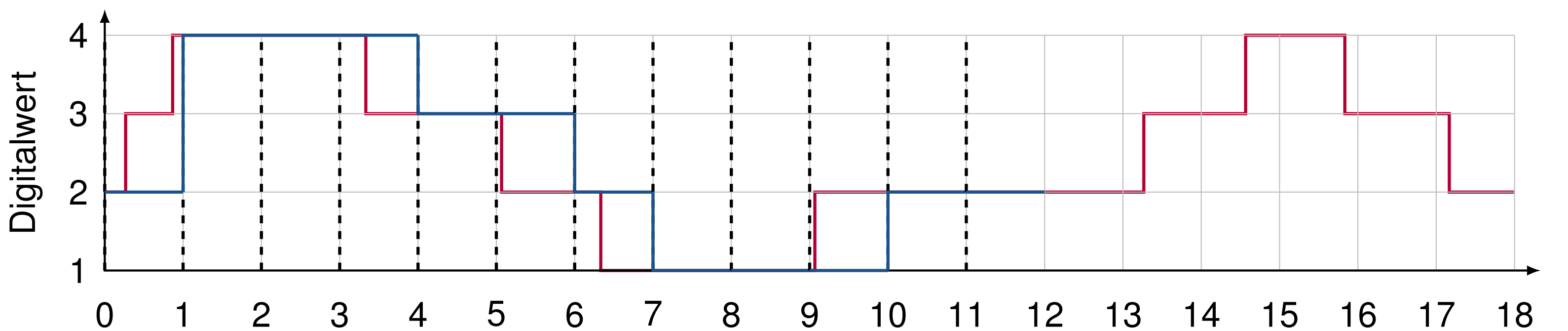


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.

Lösung

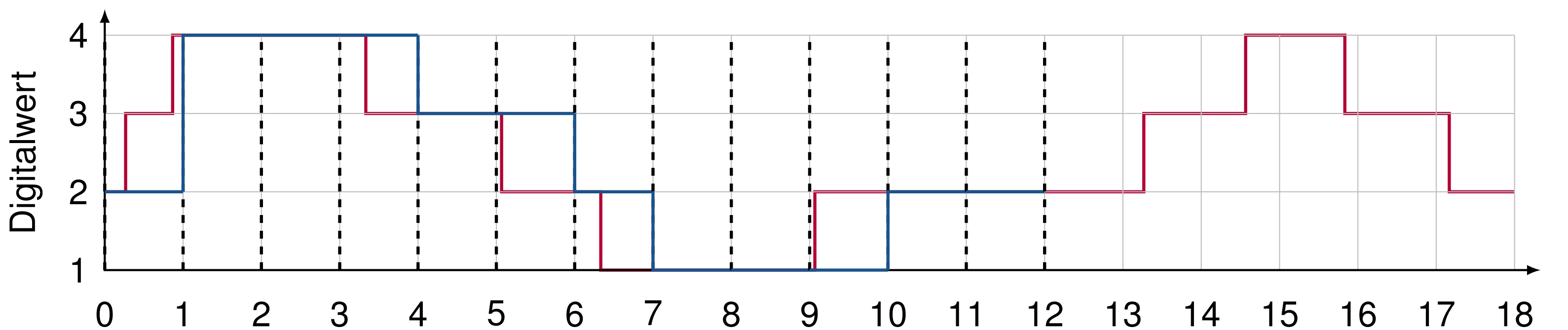


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

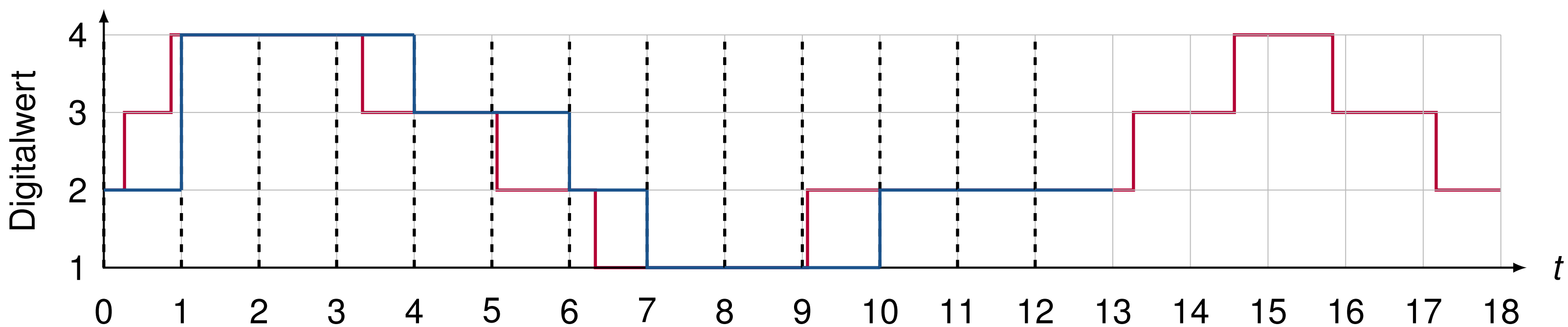


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.

Lösung

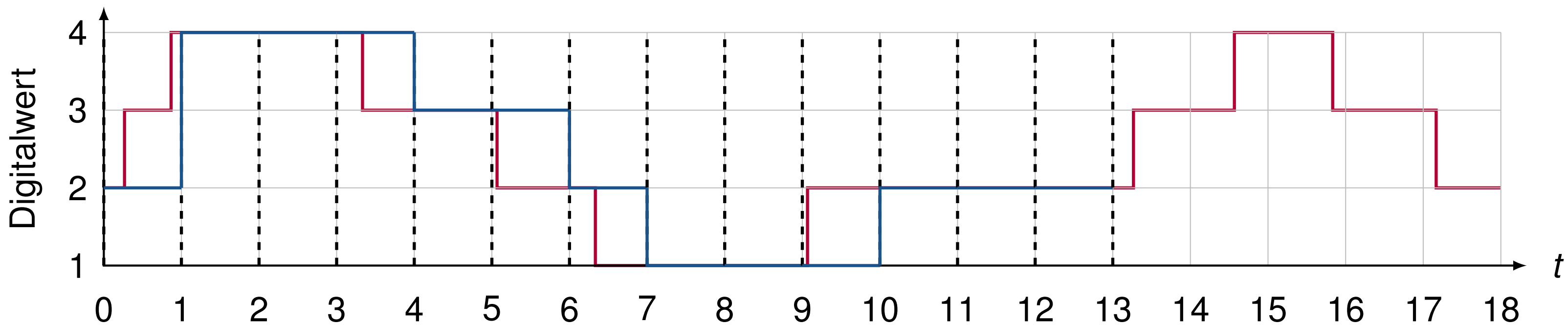


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

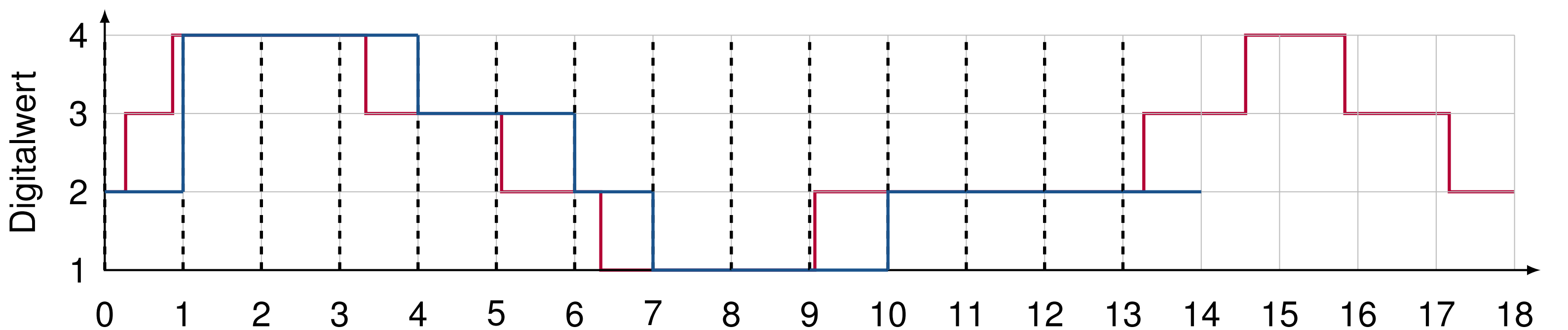


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.

Lösung

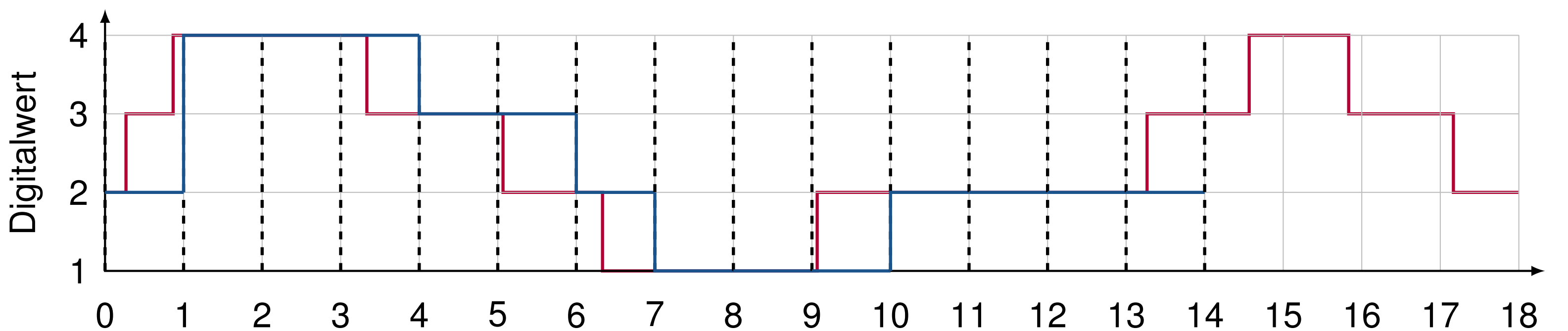


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

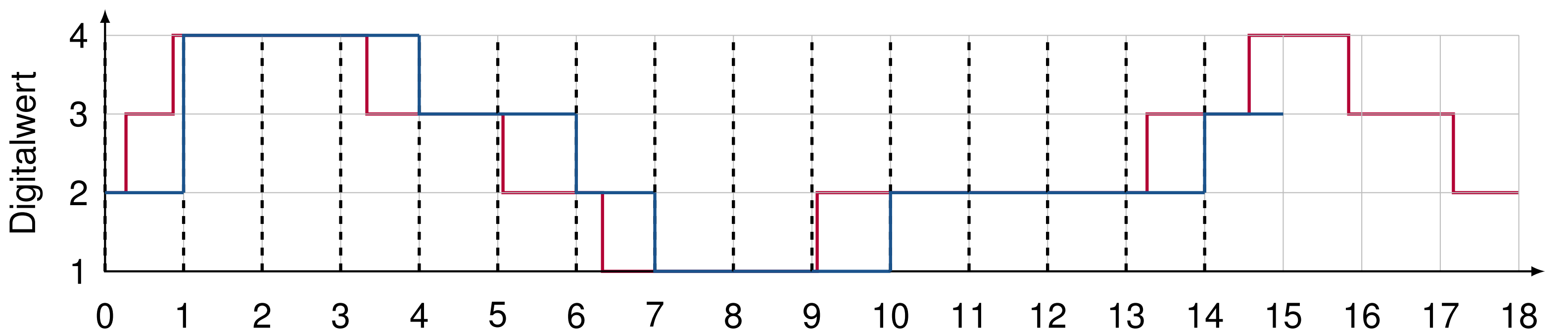


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

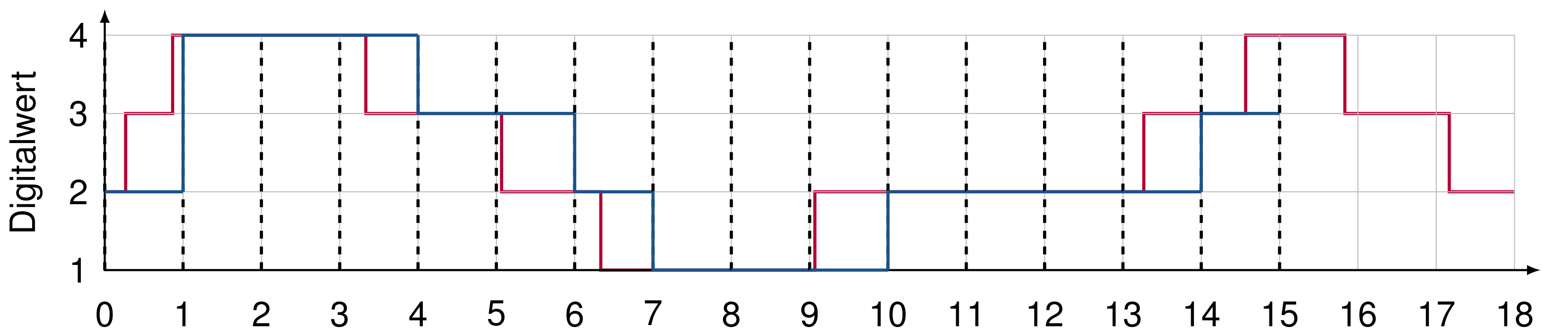


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

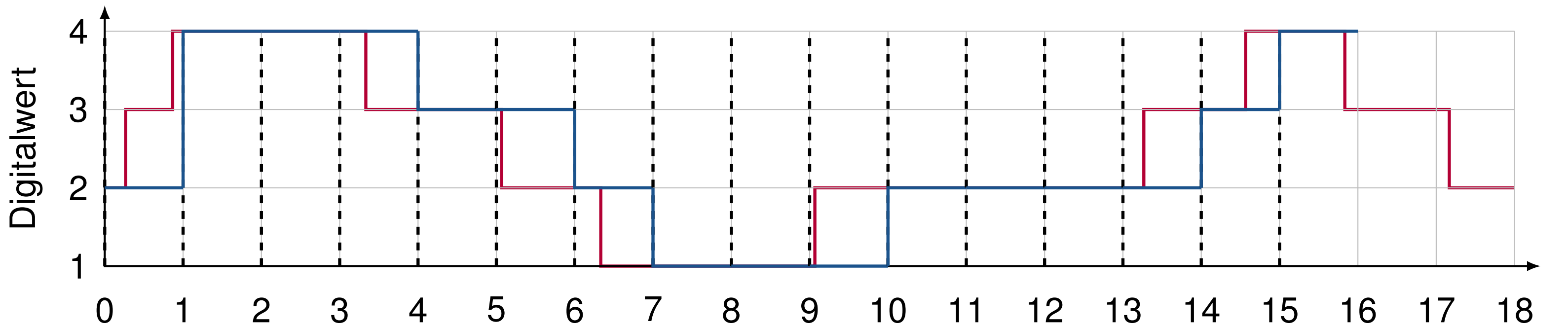


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

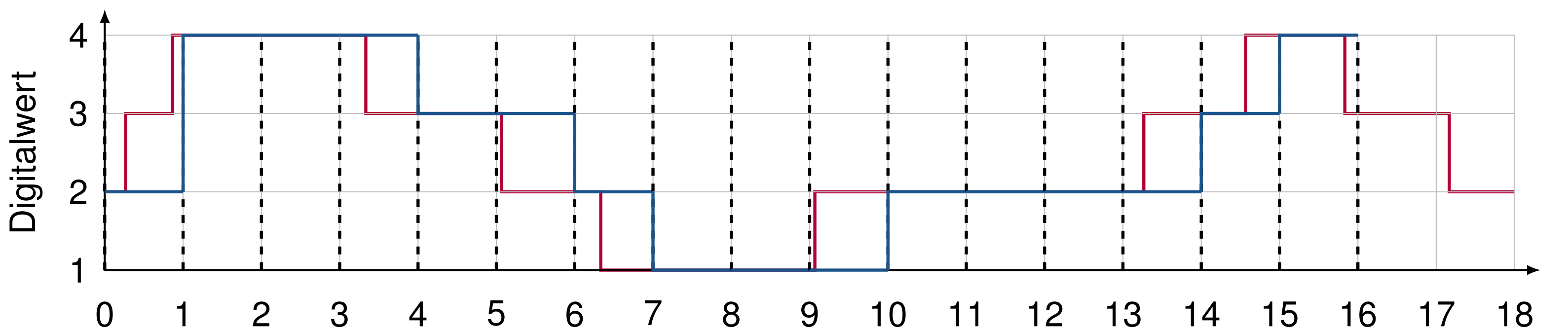


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

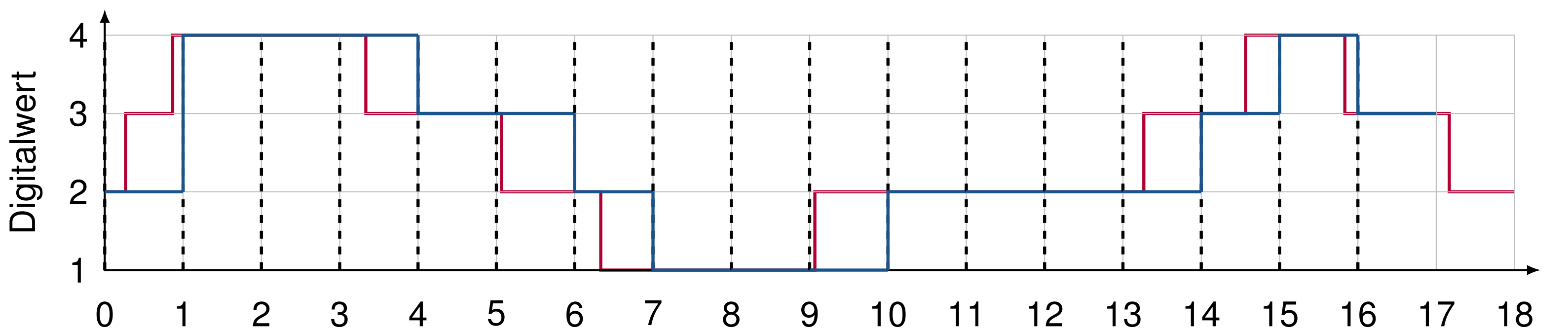


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

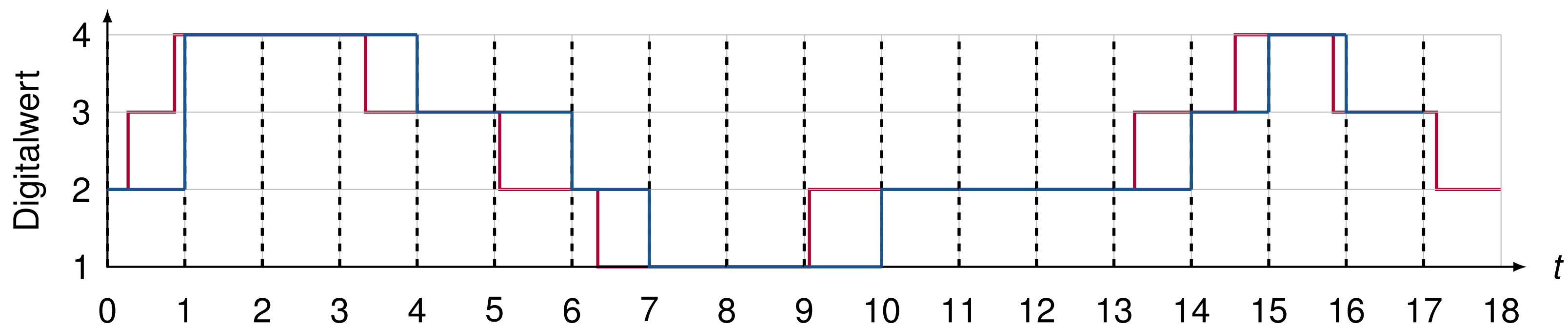


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

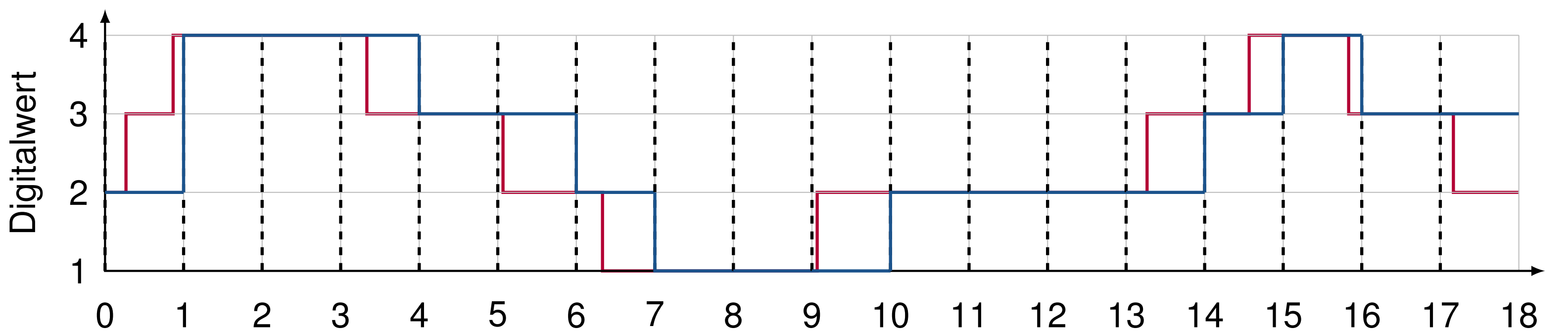


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

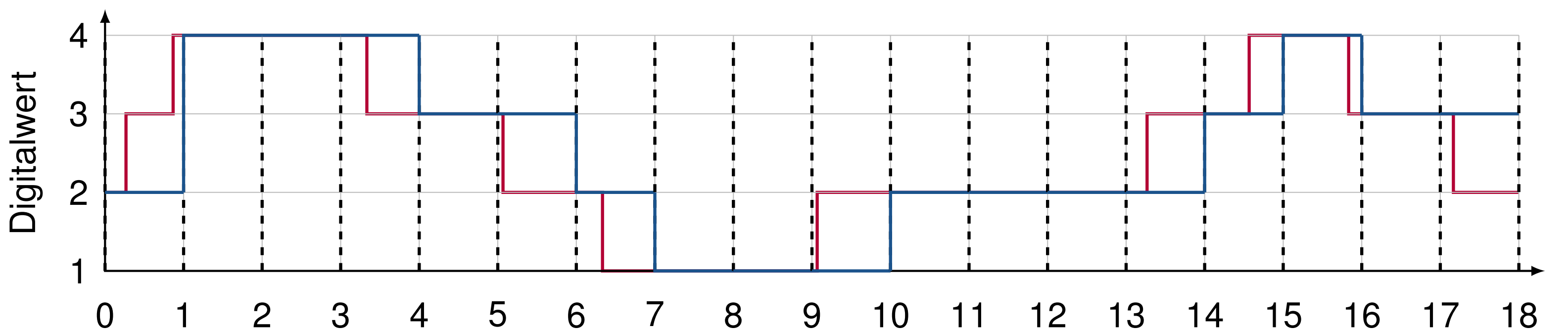


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

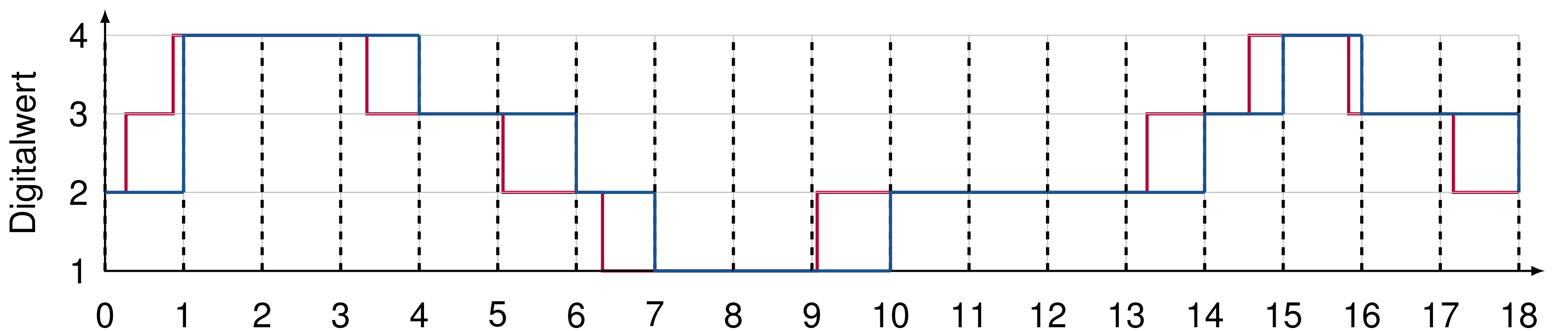


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 1 – Diskretisierung

b) Führen Sie schließlich zusätzlich eine Zeitdiskretisierung durch, wobei die Abtastzeitpunkte synchron zu den vorgegebenen Gitterlinien sein sollen.
Lösung

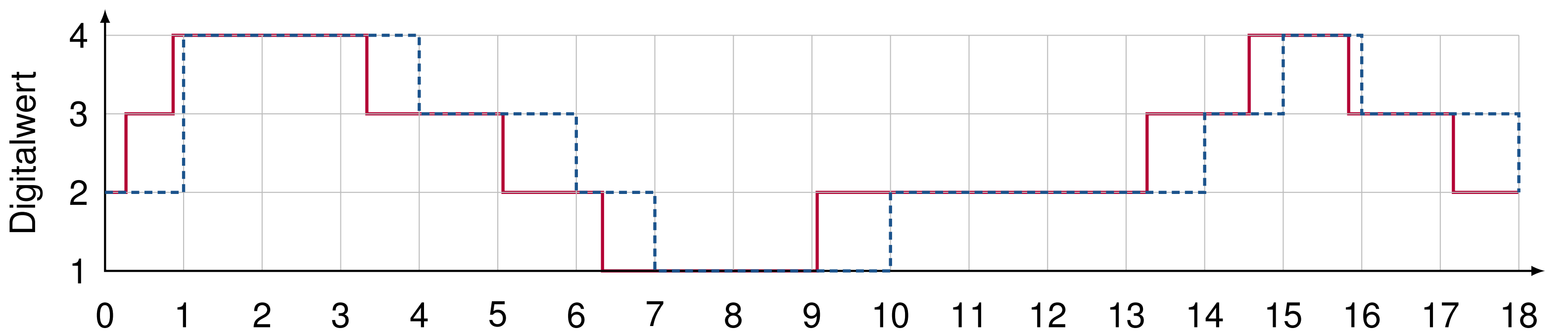


Abbildung 3: Werte- und Zeitdiskretisierung

Aufgabe 2 — Informationsgehalt

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 2 — Informationsgehalt

Für die Bevölkerung Deutschlands wird für das Jahr 2060 folgende Altersstruktur vorausgesagt:

Alter in Jahren	unter 20	20–30	30–50	50–65	über 65	Summe
Bevölkerung	10,93 M	6,42 M	15,6 M	12,32 M	22,29 M	67,56 M

(Quelle: *Bevölkerungsentwicklung Deutschlands bis 2060*, Statistisches Bundesamt 2013)

Aufgabe 2 — Informationsgehalt

Für die Bevölkerung Deutschlands wird für das Jahr 2060 folgende Altersstruktur vorausgesagt:

Alter in Jahren	unter 20	20–30	30–50	50–65	über 65	Summe
Bevölkerung	10,93 M	6,42 M	15,6 M	12,32 M	22,29 M	67,56 M

(Quelle: *Bevölkerungsentwicklung Deutschlands bis 2060*, Statistisches Bundesamt 2013)

a) Teilen Sie die Bevölkerung in zwei Gruppen Alt und Jung ein, sodass die Aussage „Herr Müller ist alt" einen Informationsgehalt von einem bit hat (ein Fehler von 0,85 M Einwohnern ist erlaubt).

Aufgabe 2 — Informationsgehalt

Für die Bevölkerung Deutschlands wird für das Jahr 2060 folgende Altersstruktur vorausgesagt:

Alter in Jahren	unter 20	20–30	30–50	50–65	über 65	Summe
Bevölkerung	10,93 M	6,42 M	15,6 M	12,32 M	22,29 M	67,56 M

(Quelle: *Bevölkerungsentwicklung Deutschlands bis 2060*, Statistisches Bundesamt 2013)

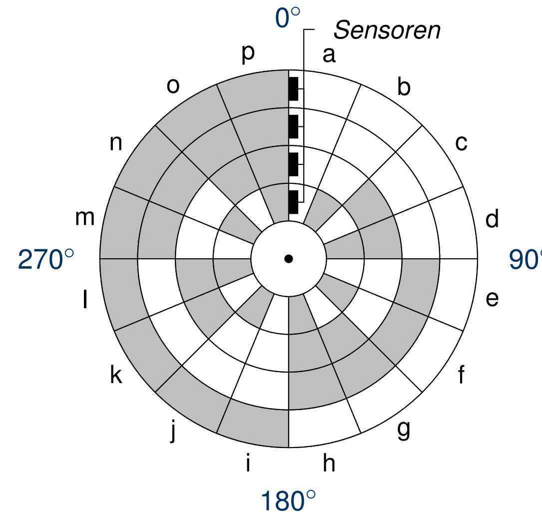
a) Teilen Sie die Bevölkerung in zwei Gruppen Alt und Jung ein, sodass die Aussage „Herr Müller ist alt" einen Informationsgehalt von einem bit hat (ein Fehler von 0,85 M Einwohnern ist erlaubt).

b) Franz ist auf der Schwelle zum 30. Lebensjahr und hält dieses für die Grenze zum Altwerden. Franz behauptet nun „Hans ist jung" und „Karl ist alt". Wie hoch ist der Informationsgehalt dieser Aussagen in bit?

Aufgabe 3 — Kodierung

Aufgabe 3 – Genauigkeit und Kodierung

Mithilfe der rechts dargestellten Drehscheibe soll ein Drehwinkel erfasst werden. Dafür ist die Scheibe in 16 Sektoren mit jeweils vier Feldern eingeteilt; vier Schleifkontakte stellen fest, ob ein Feld leitend beschichtet ist (grau dargestellt) oder nicht (weiß). Entsprechend melden sie das Signal 1 oder 0 zurück.



Intervall		Wert				Intervall		Wert			
a:	0°-22,5°	0	0	0	0	i:	180°-202,5°	1	0	0	0
b:	22,5°-45°	0	0	0	1	j:	202,5°-225°	1	0	0	1
c:	45°-67,5°	0	0	1	0	k:	225°-247,5°	1	0	1	0
d:	67,5°-90°	0	0	1	1	l:	247,5°-270°	1	0	1	1
e:	90°-112,5°	0	1	0	0	m:	270°-292,5°	1	1	0	0
f:	112,5°-135°	0	1	0	1	n:	292,5°-315°	1	1	0	1
g:	135°-157,5°	0	1	1	0	o:	315°-337,5°	1	1	1	0
h:	157,5°-180°	0	1	1	1	p:	337,5°-360°	1	1	1	1

a) Welches entscheidende Problem ergibt sich bei der angegebenen Kodierung der Intervalle bei einem realen Aufbau?

Aufgabe 3 – Kodierung, Gray-Kode

Der Gray-Kode ist ein einschrittiges, unbegrenztes Kodierungsverfahren.
Folgender Bildungsalgorithmus:

1. Basisfall: Schreibe eine 0 über eine 1
2. Spiegele jeweils den gesamten geschriebenen Block
3. Schreibe vor jedes Element des bisherigen Blockes eine 0 und vor jedes Element der Spiegelung eine 1
4. Wiederhole Schritt 2-3 solange bis die gewünschte Länge erreicht ist.

Algorithmus:

Aufgabe 3 – Kodierung, Gray-Kode

Der Gray-Kode ist ein einschrittiges, unbegrenztes Kodierungsverfahren. Folgender Bildungsalgorithmus:

1. Basisfall: Schreibe eine 0 über eine 1
2. Spiegele jeweils den gesamten geschriebenen Block
3. Schreibe vor jedes Element des bisherigen Blockes eine 0 und vor jedes Element der Spiegelung eine 1
4. Wiederhole Schritt 2-3 solange bis die gewünschte Länge erreicht ist.

Algorithmus:

0

1

Aufgabe 3 – Kodierung, Gray-Kode

Der Gray-Kode ist ein einschrittiges, unbegrenztes Kodierungsverfahren. Folgender Bildungsalgorithmus:

1. Basisfall: Schreibe eine 0 über eine 1
2. Spiegele jeweils den gesamten geschriebenen Block
3. Schreibe vor jedes Element des bisherigen Blockes eine 0 und vor jedes Element der Spiegelung eine 1
4. Wiederhole Schritt 2-3 solange bis die gewünschte Länge erreicht ist.

Algorithmus:

0
1

Aufgabe 3 – Kodierung, Gray-Kode

Der Gray-Kode ist ein einschrittiges, unbegrenztes Kodierungsverfahren. Folgender Bildungsalgorithmus:

1. Basisfall: Schreibe eine 0 über eine 1
2. Spiegele jeweils den gesamten geschriebenen Block
3. Schreibe vor jedes Element des bisherigen Blockes eine 0 und vor jedes Element der Spiegelung eine 1
4. Wiederhole Schritt 2-3 solange bis die gewünschte Länge erreicht ist.

Algorithmus:

0
1

1
0

Aufgabe 3 – Kodierung, Gray-Kode

Der Gray-Kode ist ein einschrittiges, unbegrenztes Kodierungsverfahren. Folgender Bildungsalgorithmus:

1. Basisfall: Schreibe eine 0 über eine 1
2. Spiegele jeweils den gesamten geschriebenen Block
3. Schreibe vor jedes Element des bisherigen Blockes eine 0 und vor jedes Element der Spiegelung eine 1
4. Wiederhole Schritt 2-3 solange bis die gewünschte Länge erreicht ist.

Algorithmus:

00

01

11

10

Aufgabe 3 – Kodierung, Gray-Kode

Der Gray-Kode ist ein einschrittiges, unbegrenztes Kodierungsverfahren. Folgender Bildungsalgorithmus:

1. Basisfall: Schreibe eine 0 über eine 1
2. Spiegele jeweils den gesamten geschriebenen Block
3. Schreibe vor jedes Element des bisherigen Blockes eine 0 und vor jedes Element der Spiegelung eine 1
4. Wiederhole Schritt 2-3 solange bis die gewünschte Länge erreicht ist.

Algorithmus:

00

01

11

10

Aufgabe 3 – Kodierung, Gray-Kode

Der Gray-Kode ist ein einschrittiges, unbegrenztes Kodierungsverfahren. Folgender Bildungsalgorithmus:

1. Basisfall: Schreibe eine 0 über eine 1
2. Spiegele jeweils den gesamten geschriebenen Block
3. Schreibe vor jedes Element des bisherigen Blockes eine 0 und vor jedes Element der Spiegelung eine 1
4. Wiederhole Schritt 2-3 solange bis die gewünschte Länge erreicht ist.

Algorithmus:

00
01
11
10

10
11
01
00

Aufgabe 3 – Kodierung, Gray-Kode

Der Gray-Kode ist ein einschrittiges, unbegrenztes Kodierungsverfahren. Folgender Bildungsalgorithmus:

1. Basisfall: Schreibe eine 0 über eine 1
2. Spiegele jeweils den gesamten geschriebenen Block
3. Schreibe vor jedes Element des bisherigen Blockes eine 0 und vor jedes Element der Spiegelung eine 1
4. Wiederhole Schritt 2-3 solange bis die gewünschte Länge erreicht ist.

Algorithmus:

000
001
011
010

110
111
101
100

Aufgabe 3 – Kodierung, Gray-Kode

Der Gray-Kode ist ein einschrittiges, unbegrenztes Kodierungsverfahren.
Folgender Bildungsalgorithmus:

1. Basisfall: Schreibe eine 0 über eine 1
2. Spiegele jeweils den gesamten geschriebenen Block
3. Schreibe vor jedes Element des bisherigen Blockes eine 0 und vor jedes Element der Spiegelung eine 1
4. Wiederhole Schritt 2-3 solange bis die gewünschte Länge erreicht ist.

Algorithmus:

000
001
011
010
110
111
101
100

Aufgabe 3 – Kodierung, Gray-Kode

Der Gray-Kode ist ein einschrittiges, unbegrenztes Kodierungsverfahren. Es gibt auch einen anderen Bildungsalgorithmus:

```
String[] createGray(int n) {
  String[] ergebnis = new String[n];
  for(int i = 0; i < n; i++) {
    int rShift = i >>> 1;
    int gray = rShift ^ i;
    ergebnis[i] = Integer.toBinaryString(gray);
  }
  return ergebnis;
}
```

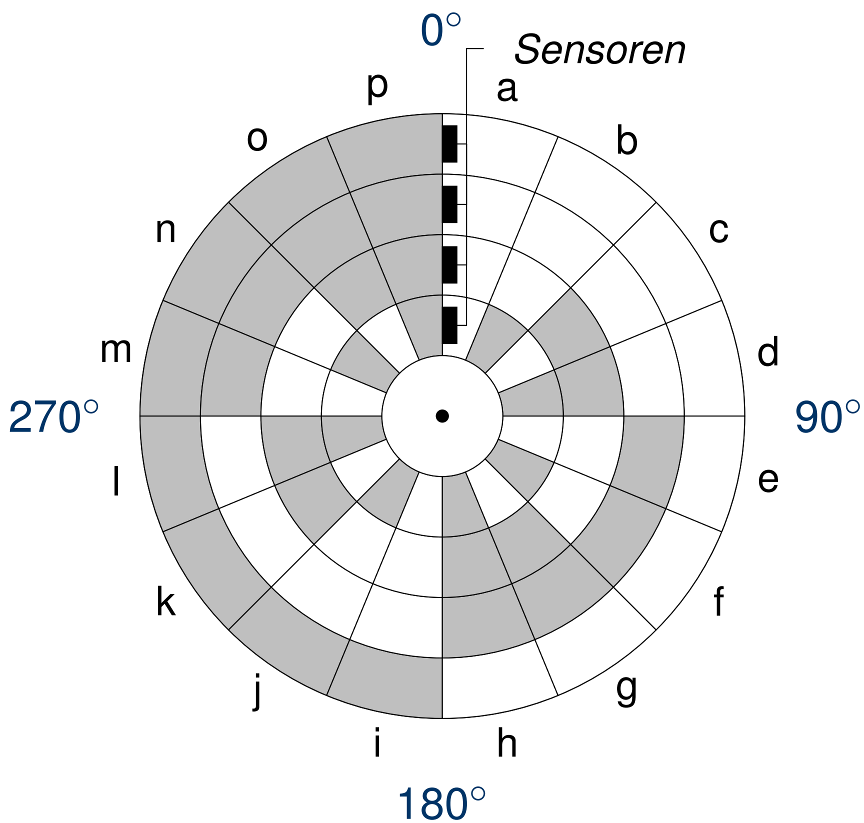
Dekodierungsalgorithmus:

1. Schreibe binär die Zahlen von 0 bis n
2. Für jede der Zahlen von 0 bis n:
   2.1 Man merke sich die binäre Zahl (i_b)
   2.2 Shifte die binäre Zahl eins nach rechts (i_{rb})
   2.3 Ver-XOR-e die beiden Zahlen i_b und i_{rb}.

```
int grayDecode(int n) {
  int number = n;
  while((n >>>= 1) != 0)
    number ^= n;
  return number;
}
```

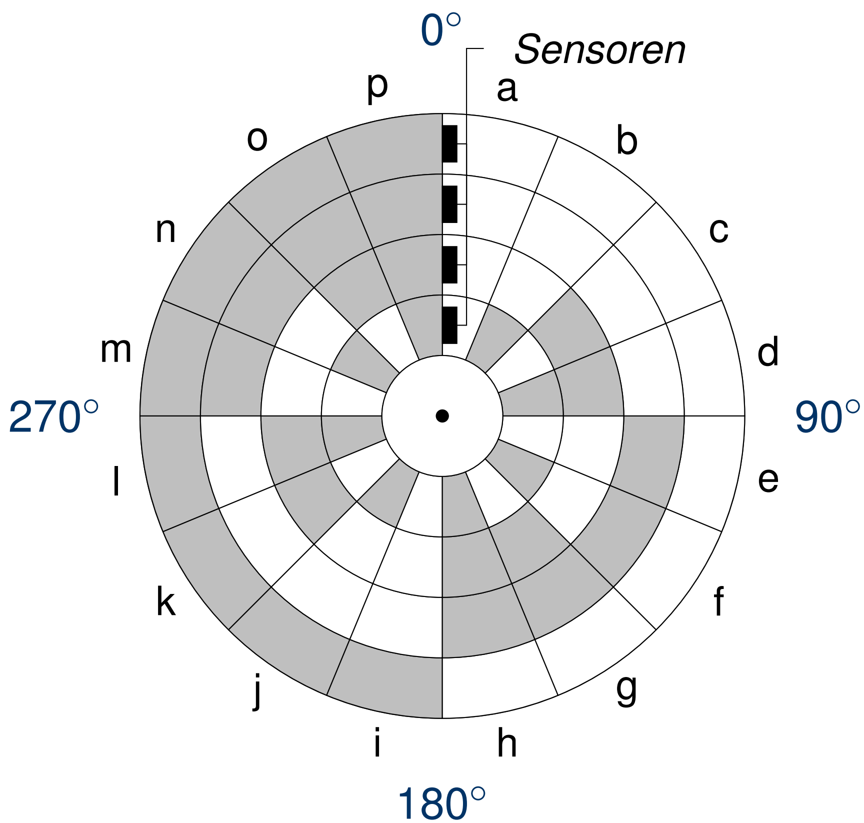
Aufgabe 3 – Genauigkeit und Kodierung

b) Entwickeln Sie eine verbesserte Kodierung. Dabei sollen die Kodierungen der Segmente *a* und *b* beibehalten und das höchstwertige Bit nicht verändert werden.

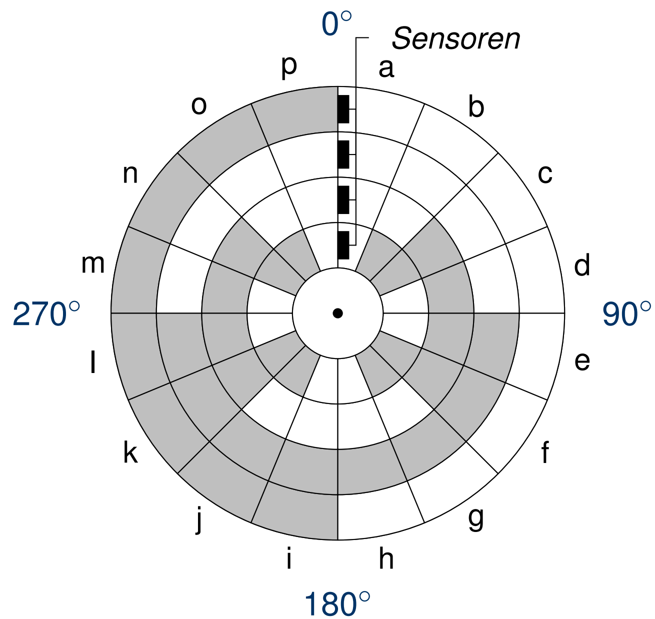


Aufgabe 3 – Genauigkeit und Kodierung

b) Entwickeln Sie eine **verbesserte** Kodierung. Dabei sollen die Kodierungen der Segmente *a* und *b* beibehalten und das höchstwertige Bit nicht verändert werden. ⇝ Gray-Code



Aufgabe 3 – Genauigkeit und Kodierung



	Intervall	Wert					Intervall	Wert			
a:	0°-22,5°	0	0	0	0	i:	180°-202,5°	1	1	0	0
b:	22,5°-45°	0	0	0	1	j:	202,5°-225°	1	1	0	1
c:	45°-67,5°	0	0	1	1	k:	225°-247,5°	1	1	1	1
d:	67,5°-90°	0	0	1	0	l:	247,5°-270°	1	1	1	0
e:	90°-112,5°	0	1	1	0	m:	270°-292,5°	1	0	1	0
f:	112,5°-135°	0	1	1	1	n:	292,5°-315°	1	0	1	1
g:	135°-157,5°	0	1	0	1	o:	315°-337,5°	1	0	0	1
h:	157,5°-180°	0	1	0	0	p:	337,5°-360°	1	0	0	0

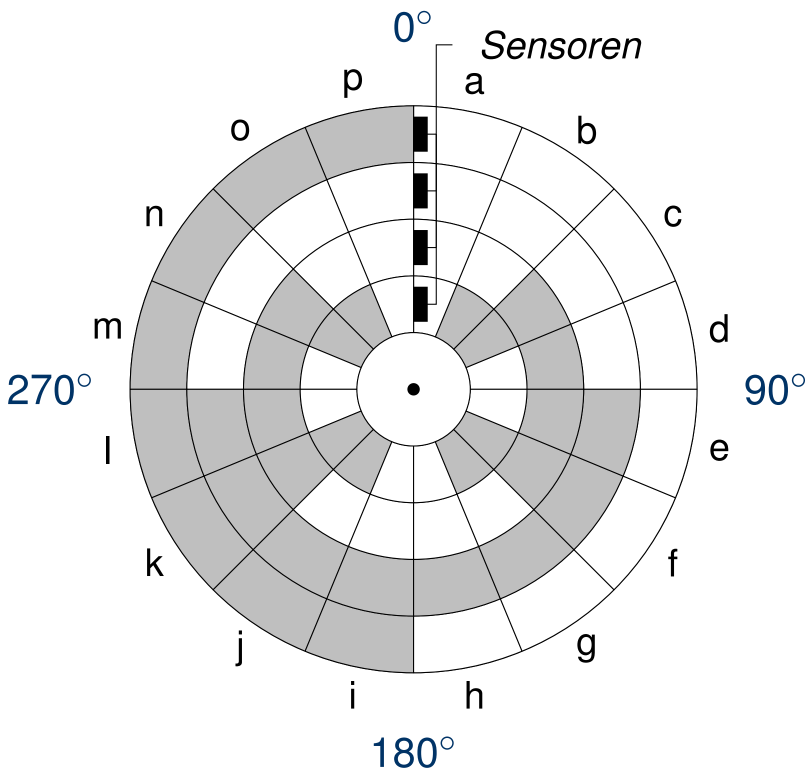
Aufgabe 3 – Genauigkeit und Kodierung

c) Wo liegen im Fall b) undefinierte Bereiche?

Aufgabe 3 – Genauigkeit und Kodierung

c) Wo liegen im Fall b) undefinierte Bereiche?

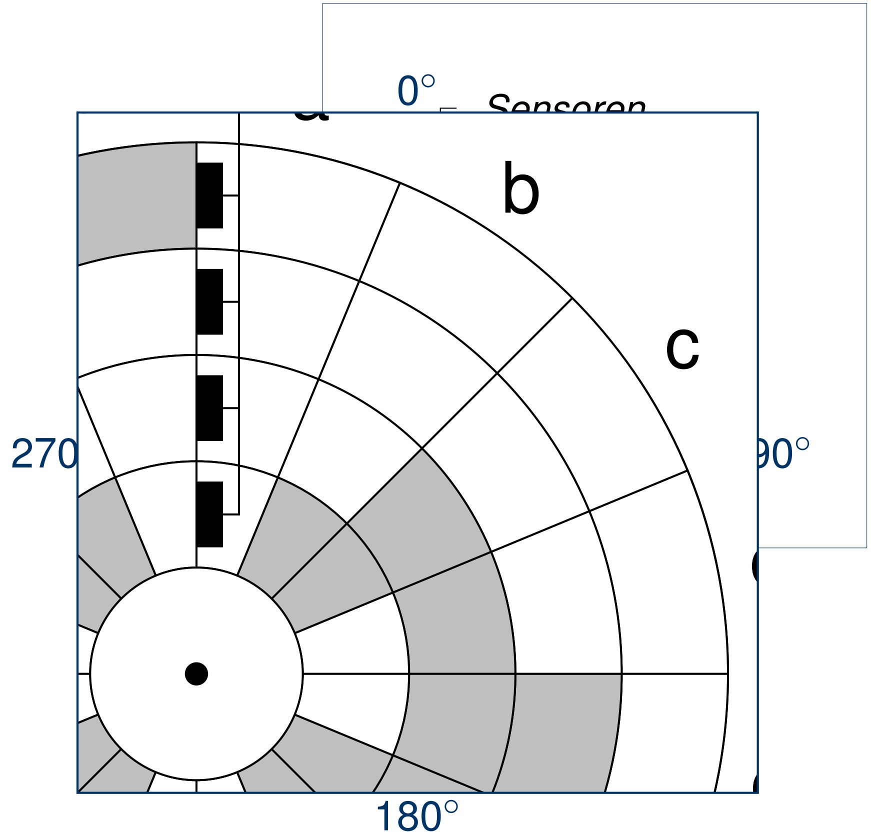
Lösung



Aufgabe 3 – Genauigkeit und Kodierung

c) Wo liegen im Fall b) undefinierte Bereiche?
Lösung

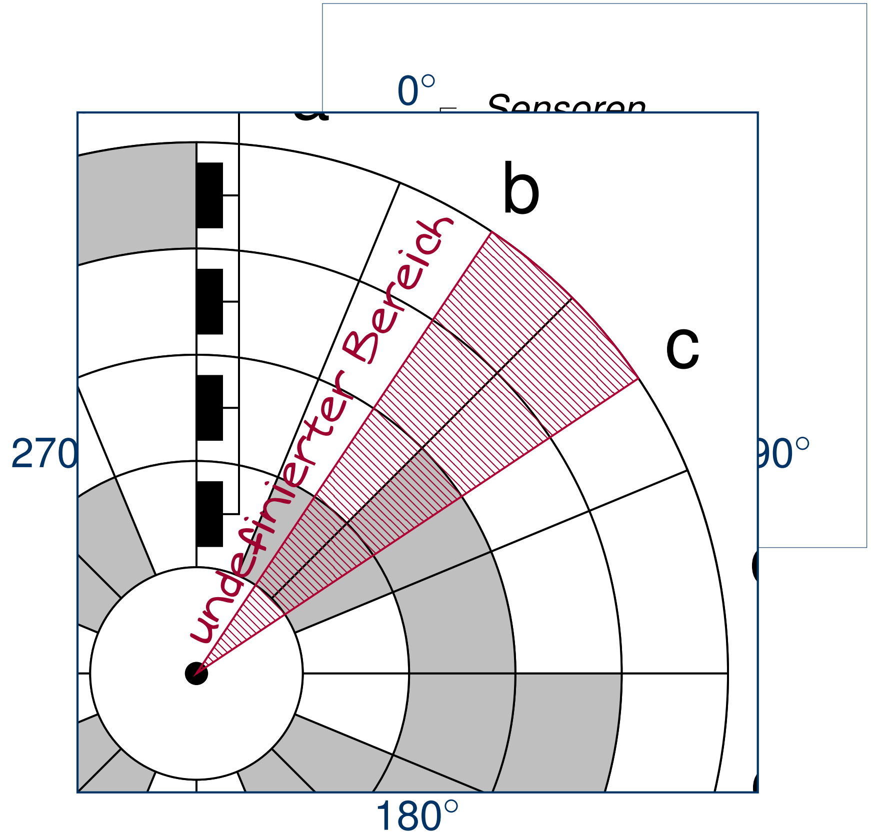
Die undefinierten Bereiche liegen genau zwischen den Intervallgrenzen.



Aufgabe 3 – Genauigkeit und Kodierung

c) Wo liegen im Fall b) undefinierte Bereiche?
Lösung

Die undefinierten Bereiche liegen genau zwischen den Intervallgrenzen.

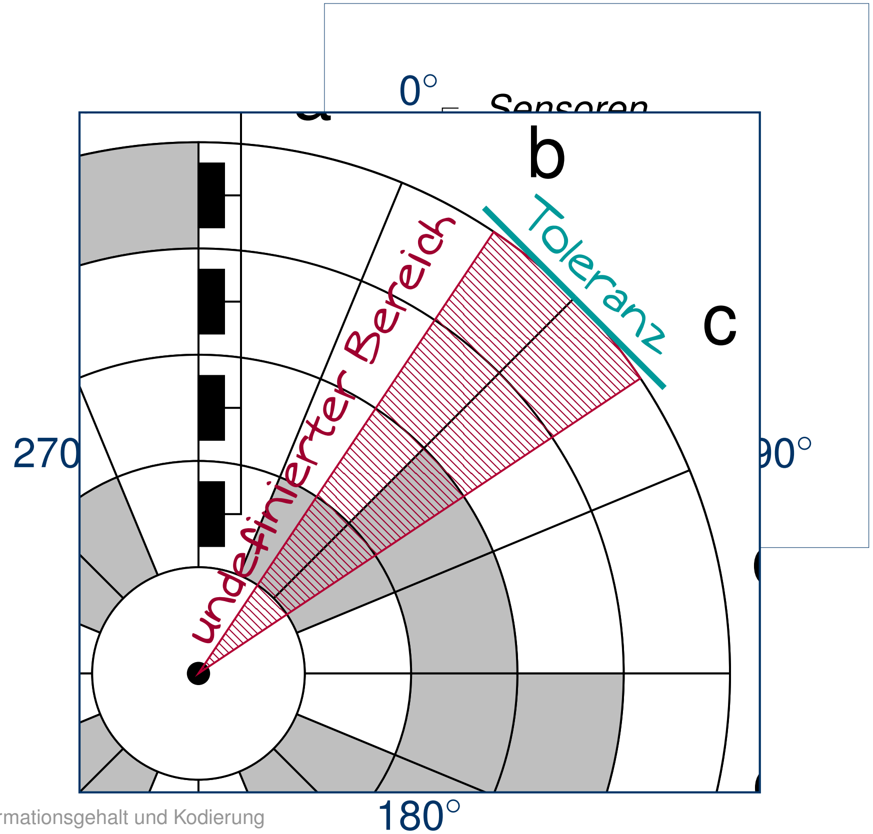


Aufgabe 3 – Genauigkeit und Kodierung

c) Wo liegen im Fall b) undefinierte Bereiche? Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

Die undefinierten Bereiche liegen genau zwischen den Intervallgrenzen.



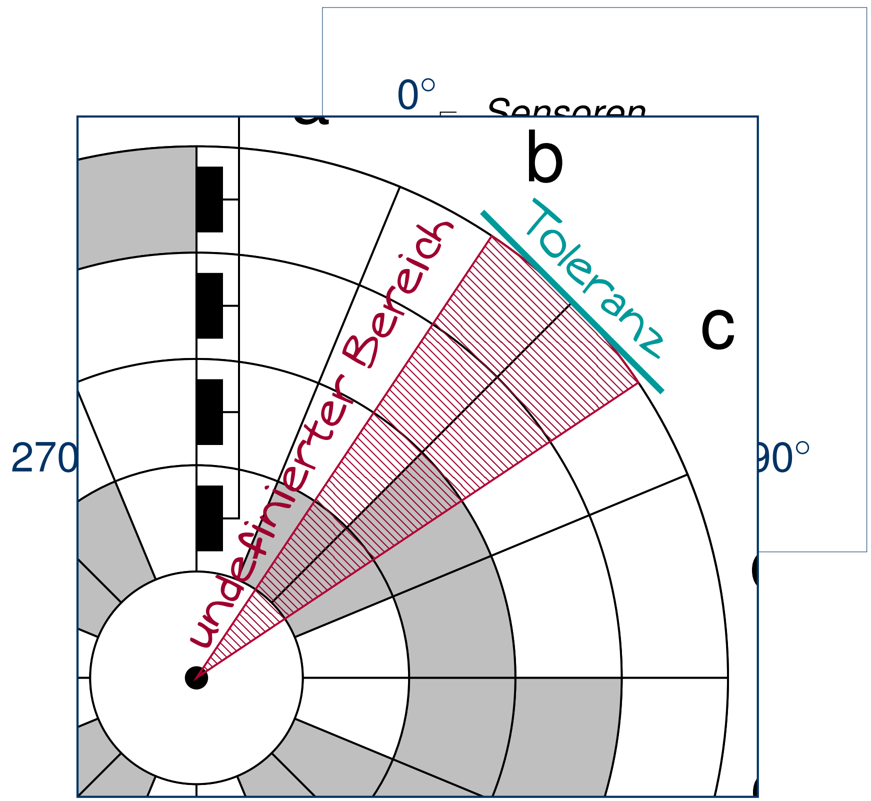
Aufgabe 3 – Genauigkeit und Kodierung

c) Wo liegen im Fall b) undefinierte Bereiche? Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

Die undefinierten Bereiche liegen genau zwischen den Intervallgrenzen.

Wir schätzen die Tangente aufgrund von kleineren Winkeln durch die Bogenlänge ab.



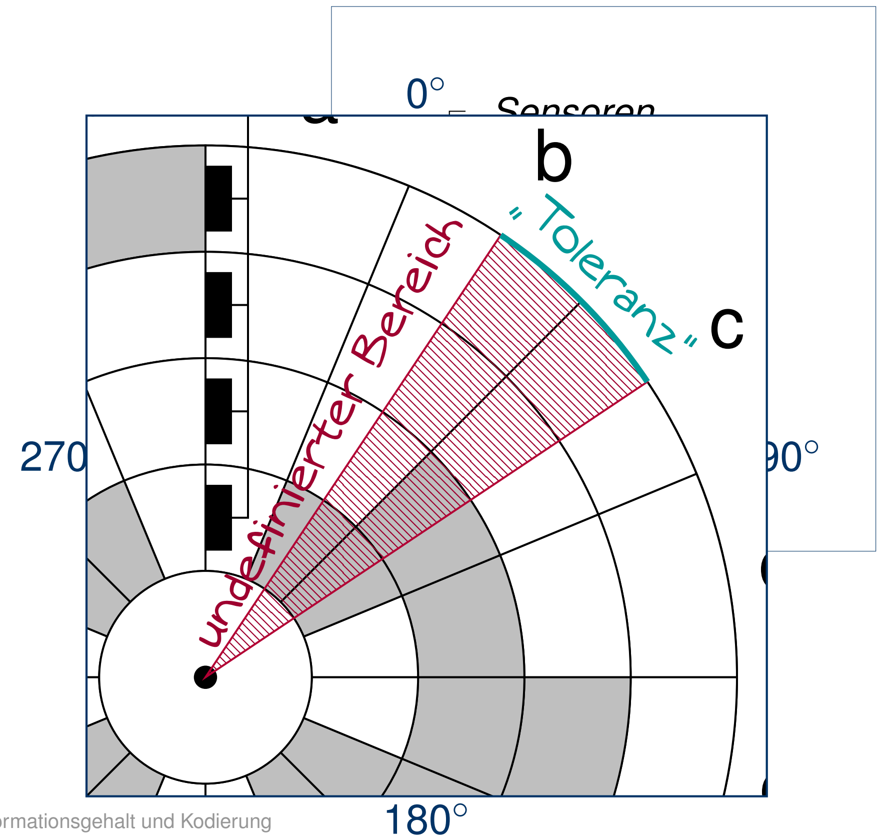
Aufgabe 3 – Genauigkeit und Kodierung

c) Wo liegen im Fall b) undefinierte Bereiche? Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

Die undefinierten Bereiche liegen genau zwischen den Intervallgrenzen.

Wir schätzen die Tangente aufgrund von kleineren Winkeln durch die Bogenlänge ab.



Aufgabe 3 – Genauigkeit und Kodierung

c) Wo liegen im Fall b) undefinierte Bereiche? Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

Wir wissen über die Bogenlänge *b* eines Kreises, dass

$$b = \frac{2\pi \cdot r \cdot \alpha}{360^\circ} \tag{1}$$

Wir sehen mit $b = \pm 0,1\text{mm}$ und aufgelöst nach α, dass:

$$\alpha \approx 360^\circ \cdot \frac{\pm 0,1\text{cm}}{2\pi \cdot \{3,4,5,6\}\text{cm}}$$

Hinweis

Die Approximation ($\approx$) kommt von der Abschätzung der Tangente durch die Bogenlänge!

Aufgabe 3 – Genauigkeit und Kodierung

c) Wo liegen im Fall b) undefinierte Bereiche? Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

$$\alpha \approx 360^\circ \cdot \frac{\pm 0,1\mathrm{cm}}{2\pi \cdot \{3,4,5,6\}\mathrm{cm}}$$

Wir setzen ein und erhalten für die verschiedenen Radien:

Radius *r*	Winkel α
3	
4	
5	
6	

Aufgabe 3 – Genauigkeit und Kodierung

c) Wo liegen im Fall b) undefinierte Bereiche? Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

$$\alpha \approx 360^\circ \cdot \frac{\pm 0,1\text{cm}}{2\pi \cdot \{3,4,5,6\}\text{cm}}$$

Wir setzen ein und erhalten für die verschiedenen Radien:

Radius r	Winkel α
3	$\alpha \approx \pm\frac{0{,}1\text{cm}}{2\pi \cdot 3\text{cm}}$
4	
5	
6	

Aufgabe 3 – Genauigkeit und Kodierung

c) Wo liegen im Fall b) undefinierte Bereiche? Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

$$\alpha \approx 360^\circ \cdot \frac{\pm 0,1\text{cm}}{2\pi \cdot \{3,4,5,6\}\text{cm}}$$

Wir setzen ein und erhalten für die verschiedenen Radien:

Radius *r*	Winkel α
3	$\pm$1,91°
4	$\alpha \approx \pm\frac{0{,}1\text{cm}}{2\pi\cdot 4\text{cm}}$
5	
6	

Aufgabe 3 – Genauigkeit und Kodierung

c) Wo liegen im Fall b) undefinierte Bereiche? Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

$$\alpha \approx 360^\circ \cdot \frac{\pm 0,1\text{cm}}{2\pi \cdot \{3,4,5,6\}\text{cm}}$$

Wir setzen ein und erhalten für die verschiedenen Radien:

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\alpha \approx \pm \frac{0{,}1\text{cm}}{2\pi \cdot 5\text{cm}}$
6	

Aufgabe 3 – Genauigkeit und Kodierung

c) Wo liegen im Fall b) undefinierte Bereiche? Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

$$\alpha \approx 360^\circ \cdot \frac{\pm 0,1\text{cm}}{2\pi \cdot \{3,4,5,6\}\text{cm}}$$

Wir setzen ein und erhalten für die verschiedenen Radien:

Radius r	Winkel α
3	$\pm$1,91°
4	$\pm$1,42°
5	$\pm$1,15°
6	$\alpha \approx \pm\frac{0{,}1\text{cm}}{2\pi \cdot 6\text{cm}}$

Aufgabe 3 – Genauigkeit und Kodierung

c) Wo liegen im Fall b) undefinierte Bereiche? Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

$$\alpha \approx 360^\circ \cdot \frac{\pm 0,1\text{cm}}{2\pi \cdot \{3,4,5,6\}\text{cm}}$$

Wir setzen ein und erhalten für die verschiedenen Radien:

Radius *r*	Winkel α
3	$\pm$1,91°
4	$\pm$1,42°
5	$\pm$1,15°
6	$\pm$0,96°

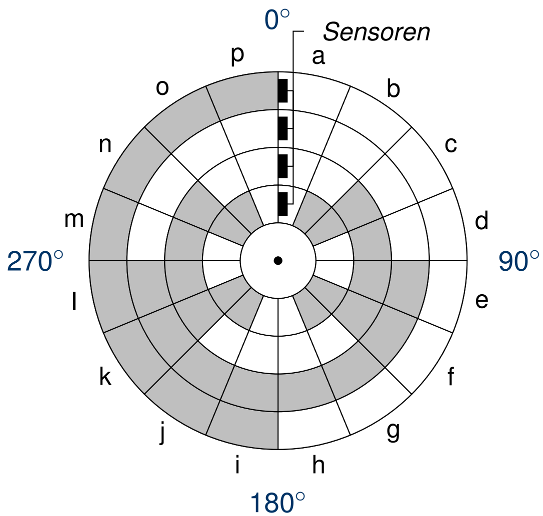
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Wir betrachten nun noch einmal die Drehscheibe und sehen, welche Sektoren sich ändern. Das Minimum des Winkels, dessen Bereich sich ändert, ist unser undefinierter Bereich.



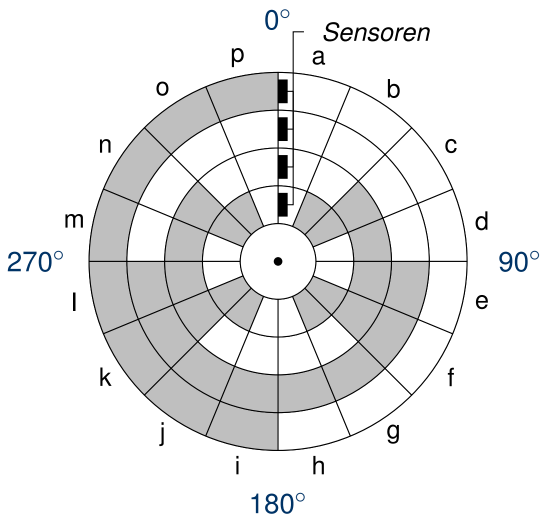
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
$a \leftrightarrow b$		$i \leftrightarrow j$	
$b \leftrightarrow c$		$j \leftrightarrow k$	
$c \leftrightarrow d$		$k \leftrightarrow l$	
$d \leftrightarrow e$		$l \leftrightarrow m$	
$e \leftrightarrow f$		$m \leftrightarrow n$	
$f \leftrightarrow g$		$n \leftrightarrow o$	
$g \leftrightarrow h$		$o \leftrightarrow p$	
$h \leftrightarrow i$		$p \leftrightarrow a$	



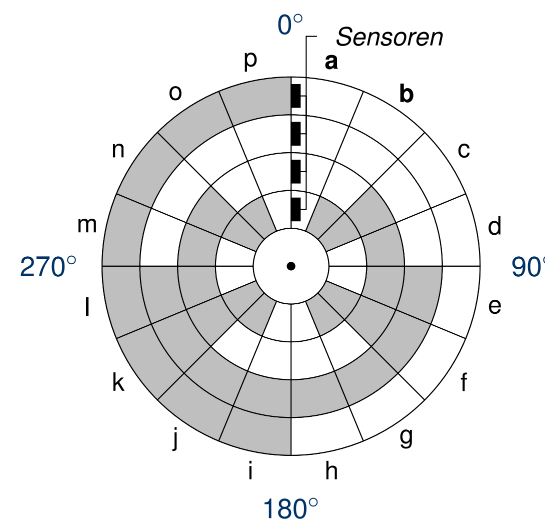
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91°$
4	$\pm 1{,}42°$
5	$\pm 1{,}15°$
6	$\pm 0{,}96°$

Übergang	α	Übergang	α
a $\leftrightarrow$ b		i $\leftrightarrow$ j	
b $\leftrightarrow$ c		j $\leftrightarrow$ k	
c $\leftrightarrow$ d		k $\leftrightarrow$ l	
d $\leftrightarrow$ e		l $\leftrightarrow$ m	
e $\leftrightarrow$ f		m $\leftrightarrow$ n	
f $\leftrightarrow$ g		n $\leftrightarrow$ o	
g $\leftrightarrow$ h		o $\leftrightarrow$ p	
h $\leftrightarrow$ i		p $\leftrightarrow$ a	



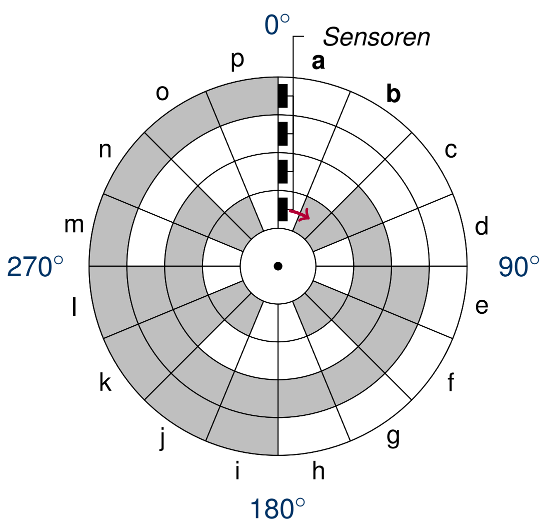
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b		i $\leftrightarrow$ j	
b $\leftrightarrow$ c		j $\leftrightarrow$ k	
c $\leftrightarrow$ d		k $\leftrightarrow$ l	
d $\leftrightarrow$ e		l $\leftrightarrow$ m	
e $\leftrightarrow$ f		m $\leftrightarrow$ n	
f $\leftrightarrow$ g		n $\leftrightarrow$ o	
g $\leftrightarrow$ h		o $\leftrightarrow$ p	
h $\leftrightarrow$ i		p $\leftrightarrow$ a	



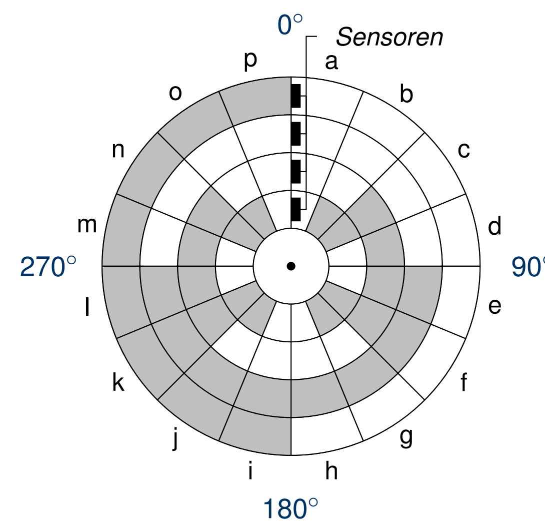
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91°$
4	$\pm 1{,}42°$
5	$\pm 1{,}15°$
6	$\pm 0{,}96°$

Übergang	α	Übergang	α
a ↔ b		i ↔ j	
b ↔ c		j ↔ k	
c ↔ d		k ↔ l	
d ↔ e		l ↔ m	
e ↔ f		m ↔ n	
f ↔ g		n ↔ o	
g ↔ h		o ↔ p	
h ↔ i		p ↔ a	



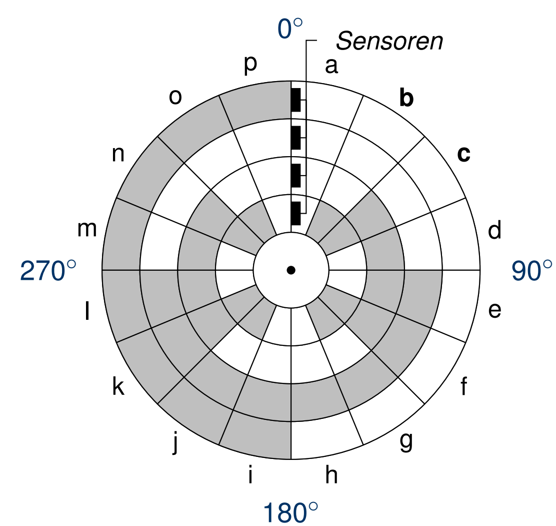
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	
b $\leftrightarrow$ c		j $\leftrightarrow$ k	
c $\leftrightarrow$ d		k $\leftrightarrow$ l	
d $\leftrightarrow$ e		l $\leftrightarrow$ m	
e $\leftrightarrow$ f		m $\leftrightarrow$ n	
f $\leftrightarrow$ g		n $\leftrightarrow$ o	
g $\leftrightarrow$ h		o $\leftrightarrow$ p	
h $\leftrightarrow$ i		p $\leftrightarrow$ a	



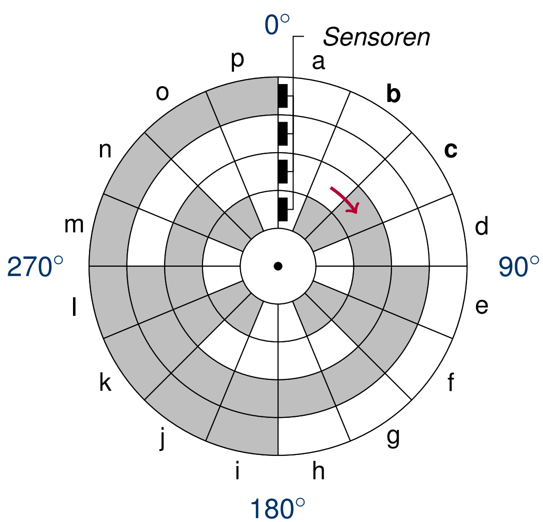
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	
b $\leftrightarrow$ c		j $\leftrightarrow$ k	
c $\leftrightarrow$ d		k $\leftrightarrow$ l	
d $\leftrightarrow$ e		l $\leftrightarrow$ m	
e $\leftrightarrow$ f		m $\leftrightarrow$ n	
f $\leftrightarrow$ g		n $\leftrightarrow$ o	
g $\leftrightarrow$ h		o $\leftrightarrow$ p	
h $\leftrightarrow$ i		p $\leftrightarrow$ a	



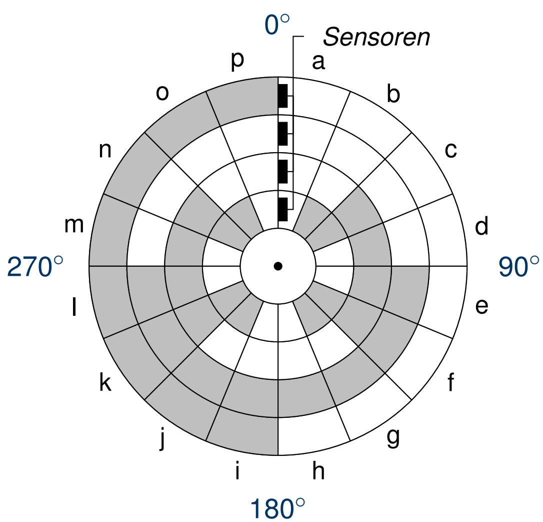
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a ↔ b	$\pm 1{,}91^\circ$	i ↔ j	
b ↔ c	$\pm 1{,}42^\circ$	j ↔ k	
c ↔ d		k ↔ l	
d ↔ e		l ↔ m	
e ↔ f		m ↔ n	
f ↔ g		n ↔ o	
g ↔ h		o ↔ p	
h ↔ i		p ↔ a	



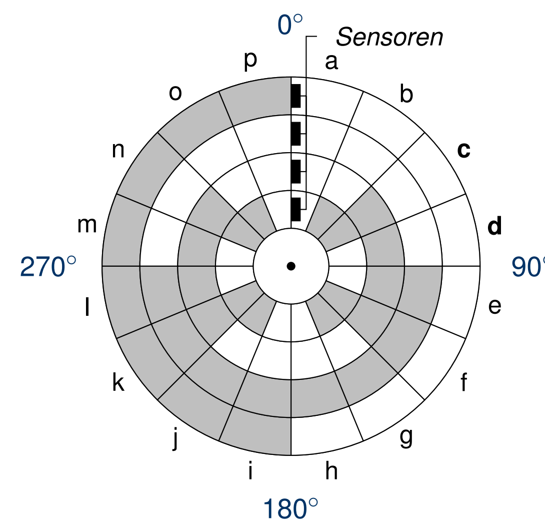
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
$a \leftrightarrow b$	$\pm 1{,}91^\circ$	$i \leftrightarrow j$	
$b \leftrightarrow c$	$\pm 1{,}42^\circ$	$j \leftrightarrow k$	
$c \leftrightarrow d$		$k \leftrightarrow l$	
$d \leftrightarrow e$		$l \leftrightarrow m$	
$e \leftrightarrow f$		$m \leftrightarrow n$	
$f \leftrightarrow g$		$n \leftrightarrow o$	
$g \leftrightarrow h$		$o \leftrightarrow p$	
$h \leftrightarrow i$		$p \leftrightarrow a$	



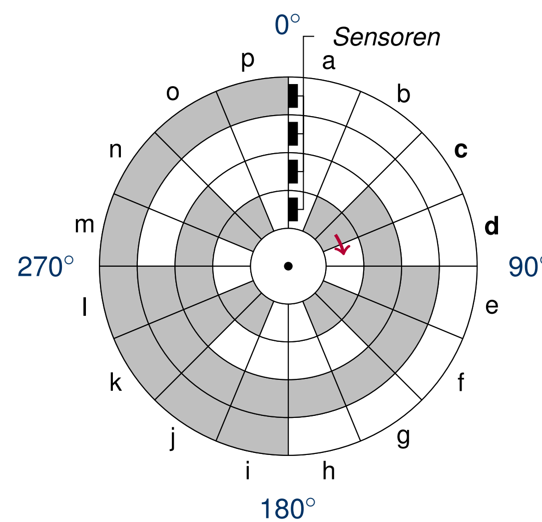
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	
c $\leftrightarrow$ d		k $\leftrightarrow$ l	
d $\leftrightarrow$ e		l $\leftrightarrow$ m	
e $\leftrightarrow$ f		m $\leftrightarrow$ n	
f $\leftrightarrow$ g		n $\leftrightarrow$ o	
g $\leftrightarrow$ h		o $\leftrightarrow$ p	
h $\leftrightarrow$ i		p $\leftrightarrow$ a	



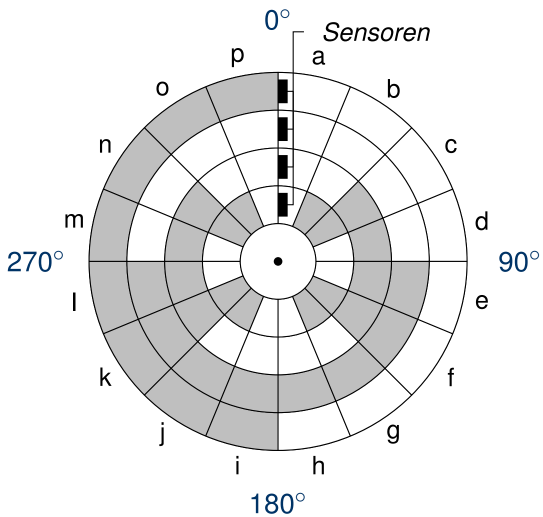
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91°$
4	$\pm 1{,}42°$
5	$\pm 1{,}15°$
6	$\pm 0{,}96°$

Übergang	α	Übergang	α
$a \leftrightarrow b$	$\pm 1{,}91°$	$i \leftrightarrow j$	
$b \leftrightarrow c$	$\pm 1{,}42°$	$j \leftrightarrow k$	
$c \leftrightarrow d$	$\pm 1{,}91°$	$k \leftrightarrow l$	
$d \leftrightarrow e$		$l \leftrightarrow m$	
$e \leftrightarrow f$		$m \leftrightarrow n$	
$f \leftrightarrow g$		$n \leftrightarrow o$	
$g \leftrightarrow h$		$o \leftrightarrow p$	
$h \leftrightarrow i$		$p \leftrightarrow a$	



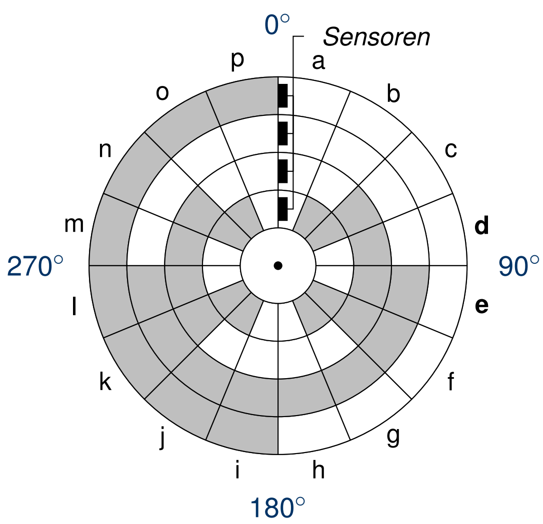
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	
d $\leftrightarrow$ e		l $\leftrightarrow$ m	
e $\leftrightarrow$ f		m $\leftrightarrow$ n	
f $\leftrightarrow$ g		n $\leftrightarrow$ o	
g $\leftrightarrow$ h		o $\leftrightarrow$ p	
h $\leftrightarrow$ i		p $\leftrightarrow$ a	



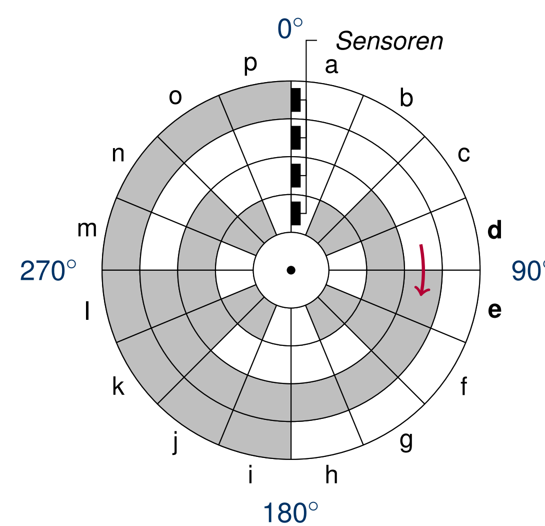
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	
d $\leftrightarrow$ e		l $\leftrightarrow$ m	
e $\leftrightarrow$ f		m $\leftrightarrow$ n	
f $\leftrightarrow$ g		n $\leftrightarrow$ o	
g $\leftrightarrow$ h		o $\leftrightarrow$ p	
h $\leftrightarrow$ i		p $\leftrightarrow$ a	



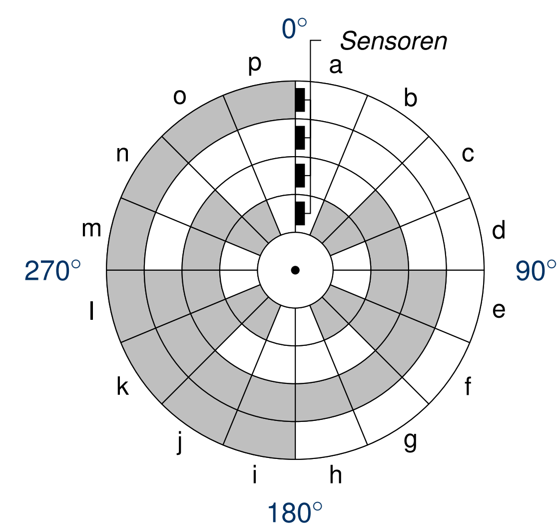
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	
d $\leftrightarrow$ e	$\pm 1{,}15^\circ$	l $\leftrightarrow$ m	
e $\leftrightarrow$ f		m $\leftrightarrow$ n	
f $\leftrightarrow$ g		n $\leftrightarrow$ o	
g $\leftrightarrow$ h		o $\leftrightarrow$ p	
h $\leftrightarrow$ i		p $\leftrightarrow$ a	



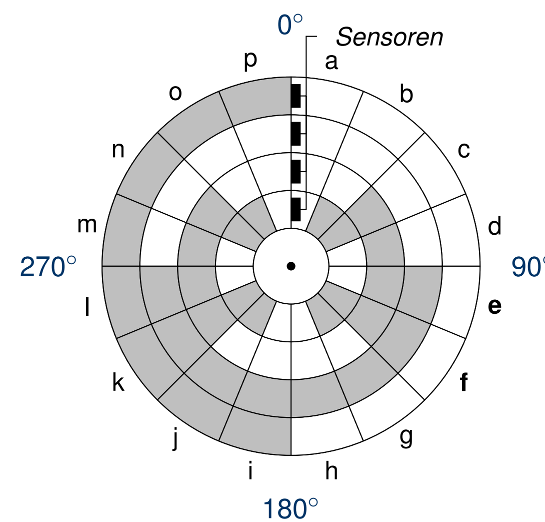
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Lösung

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	
d $\leftrightarrow$ e	$\pm 1{,}15^\circ$	l $\leftrightarrow$ m	
e $\leftrightarrow$ f		m $\leftrightarrow$ n	
f $\leftrightarrow$ g		n $\leftrightarrow$ o	
g $\leftrightarrow$ h		o $\leftrightarrow$ p	
h $\leftrightarrow$ i		p $\leftrightarrow$ a	



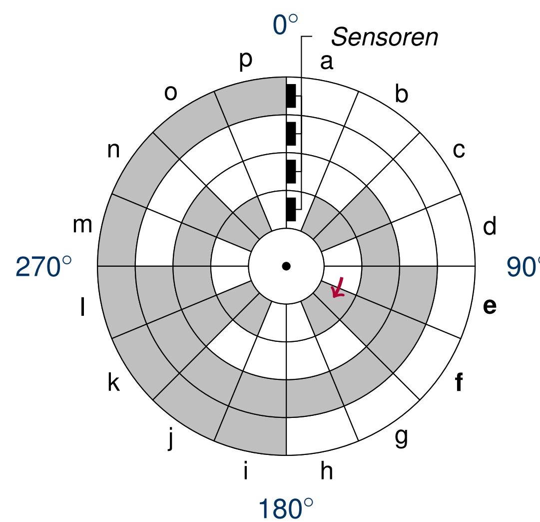
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a ↔ b	$\pm 1{,}91^\circ$	i ↔ j	
b ↔ c	$\pm 1{,}42^\circ$	j ↔ k	
c ↔ d	$\pm 1{,}91^\circ$	k ↔ l	
d ↔ e	$\pm 1{,}15^\circ$	l ↔ m	
e ↔ f		m ↔ n	
f ↔ g		n ↔ o	
g ↔ h		o ↔ p	
h ↔ i		p ↔ a	



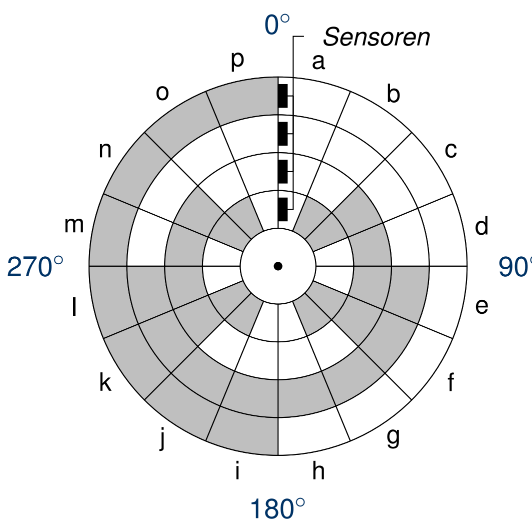
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a ↔ b	$\pm 1{,}91^\circ$	i ↔ j	
b ↔ c	$\pm 1{,}42^\circ$	j ↔ k	
c ↔ d	$\pm 1{,}91^\circ$	k ↔ l	
d ↔ e	$\pm 1{,}15^\circ$	l ↔ m	
e ↔ f	$\pm 1{,}91^\circ$	m ↔ n	
f ↔ g		n ↔ o	
g ↔ h		o ↔ p	
h ↔ i		p ↔ a	



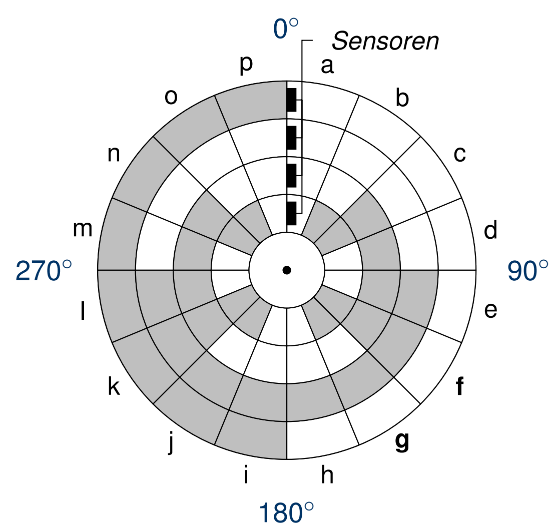
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
$a \leftrightarrow b$	$\pm 1{,}91^\circ$	$i \leftrightarrow j$	
$b \leftrightarrow c$	$\pm 1{,}42^\circ$	$j \leftrightarrow k$	
$c \leftrightarrow d$	$\pm 1{,}91^\circ$	$k \leftrightarrow l$	
$d \leftrightarrow e$	$\pm 1{,}15^\circ$	$l \leftrightarrow m$	
$e \leftrightarrow f$	$\pm 1{,}91^\circ$	$m \leftrightarrow n$	
$f \leftrightarrow g$		$n \leftrightarrow o$	
$g \leftrightarrow h$		$o \leftrightarrow p$	
$h \leftrightarrow i$		$p \leftrightarrow a$	



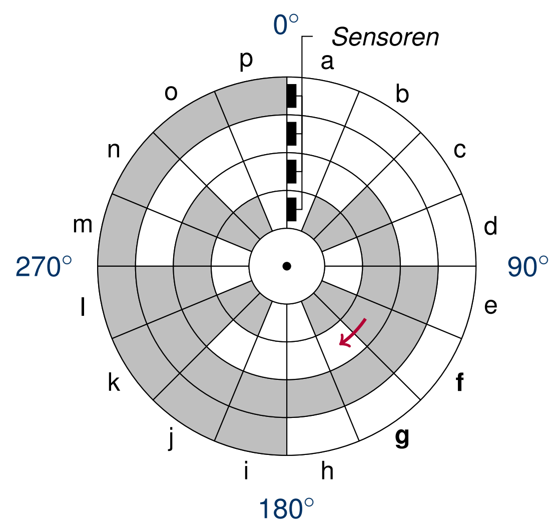
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	
d $\leftrightarrow$ e	$\pm 1{,}15^\circ$	l $\leftrightarrow$ m	
e $\leftrightarrow$ f	$\pm 1{,}91^\circ$	m $\leftrightarrow$ n	
f $\leftrightarrow$ g		n $\leftrightarrow$ o	
g $\leftrightarrow$ h		o $\leftrightarrow$ p	
h $\leftrightarrow$ i		p $\leftrightarrow$ a	



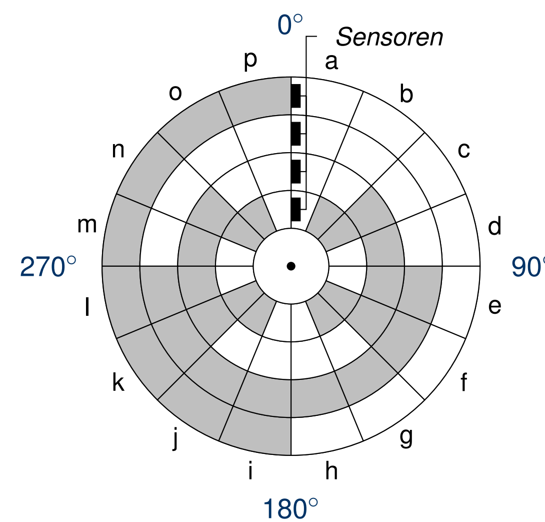
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	
d $\leftrightarrow$ e	$\pm 1{,}15^\circ$	l $\leftrightarrow$ m	
e $\leftrightarrow$ f	$\pm 1{,}91^\circ$	m $\leftrightarrow$ n	
f $\leftrightarrow$ g	$\pm 1{,}42^\circ$	n $\leftrightarrow$ o	
g $\leftrightarrow$ h		o $\leftrightarrow$ p	
h $\leftrightarrow$ i		p $\leftrightarrow$ a	



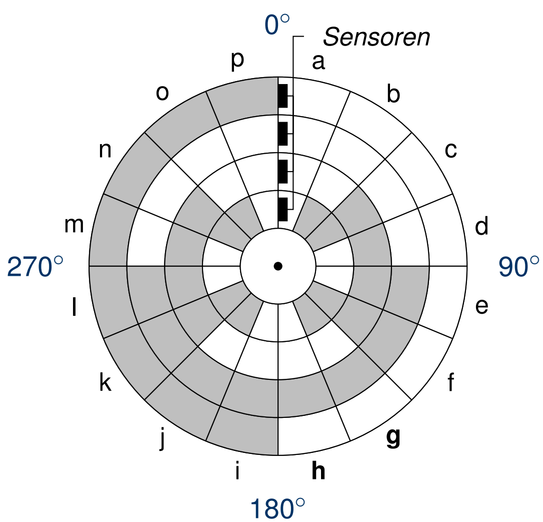
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	
d $\leftrightarrow$ e	$\pm 1{,}15^\circ$	l $\leftrightarrow$ m	
e $\leftrightarrow$ f	$\pm 1{,}91^\circ$	m $\leftrightarrow$ n	
f $\leftrightarrow$ g	$\pm 1{,}42^\circ$	n $\leftrightarrow$ o	
g $\leftrightarrow$ h		o $\leftrightarrow$ p	
h $\leftrightarrow$ i		p $\leftrightarrow$ a	



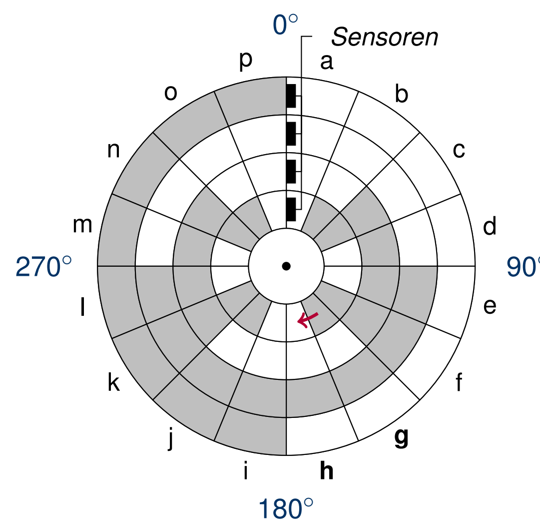
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	
d $\leftrightarrow$ e	$\pm 1{,}15^\circ$	l $\leftrightarrow$ m	
e $\leftrightarrow$ f	$\pm 1{,}91^\circ$	m $\leftrightarrow$ n	
f $\leftrightarrow$ g	$\pm 1{,}42^\circ$	n $\leftrightarrow$ o	
g $\leftrightarrow$ h		o $\leftrightarrow$ p	
h $\leftrightarrow$ i		p $\leftrightarrow$ a	



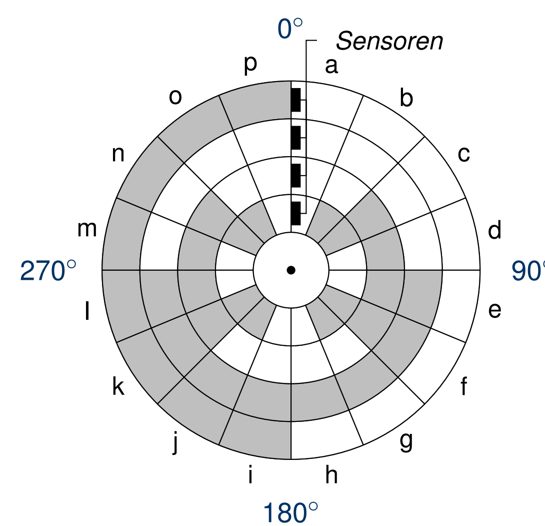
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	
d $\leftrightarrow$ e	$\pm 1{,}15^\circ$	l $\leftrightarrow$ m	
e $\leftrightarrow$ f	$\pm 1{,}91^\circ$	m $\leftrightarrow$ n	
f $\leftrightarrow$ g	$\pm 1{,}42^\circ$	n $\leftrightarrow$ o	
g $\leftrightarrow$ h	$\pm 1{,}91^\circ$	o $\leftrightarrow$ p	
h $\leftrightarrow$ i		p $\leftrightarrow$ a	



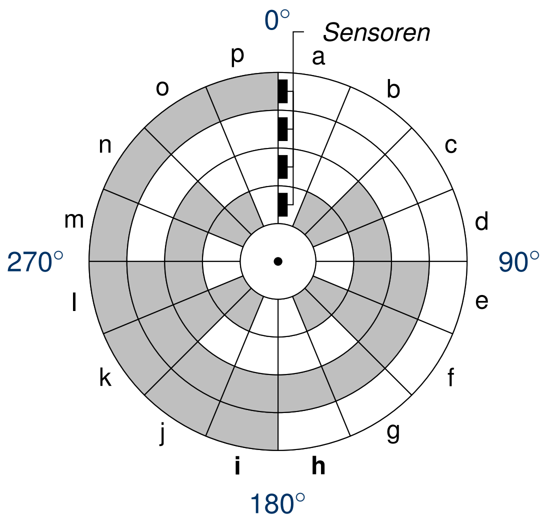
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a ↔ b	$\pm 1{,}91^\circ$	i ↔ j	
b ↔ c	$\pm 1{,}42^\circ$	j ↔ k	
c ↔ d	$\pm 1{,}91^\circ$	k ↔ l	
d ↔ e	$\pm 1{,}15^\circ$	l ↔ m	
e ↔ f	$\pm 1{,}91^\circ$	m ↔ n	
f ↔ g	$\pm 1{,}42^\circ$	n ↔ o	
g ↔ h	$\pm 1{,}91^\circ$	o ↔ p	
h ↔ i		p ↔ a	



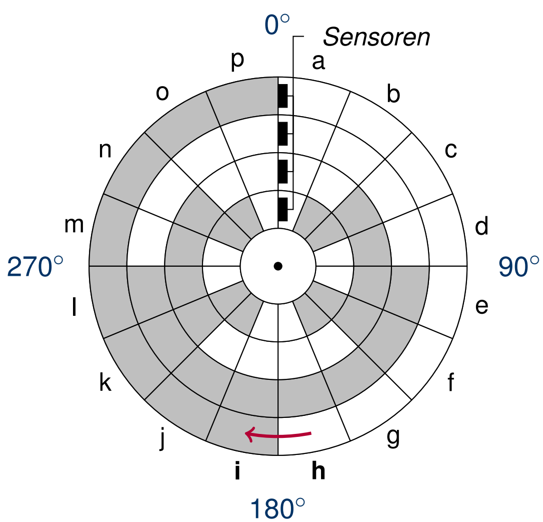
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	
d $\leftrightarrow$ e	$\pm 1{,}15^\circ$	l $\leftrightarrow$ m	
e $\leftrightarrow$ f	$\pm 1{,}91^\circ$	m $\leftrightarrow$ n	
f $\leftrightarrow$ g	$\pm 1{,}42^\circ$	n $\leftrightarrow$ o	
g $\leftrightarrow$ h	$\pm 1{,}91^\circ$	o $\leftrightarrow$ p	
h $\leftrightarrow$ i		p $\leftrightarrow$ a	



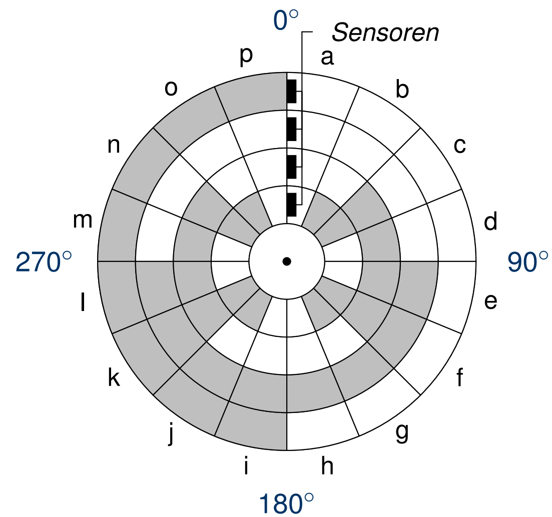
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	
d $\leftrightarrow$ e	$\pm 1{,}15^\circ$	l $\leftrightarrow$ m	
e $\leftrightarrow$ f	$\pm 1{,}91^\circ$	m $\leftrightarrow$ n	
f $\leftrightarrow$ g	$\pm 1{,}42^\circ$	n $\leftrightarrow$ o	
g $\leftrightarrow$ h	$\pm 1{,}91^\circ$	o $\leftrightarrow$ p	
h $\leftrightarrow$ i	$\pm 0{,}96^\circ$	p $\leftrightarrow$ a	



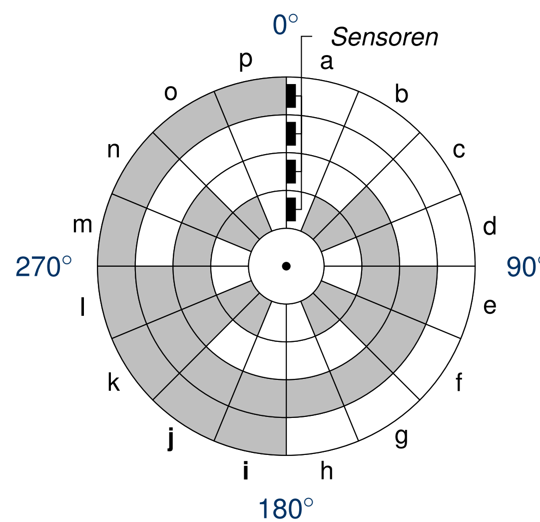
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
$a \leftrightarrow b$	$\pm 1{,}91^\circ$	$i \leftrightarrow j$	
$b \leftrightarrow c$	$\pm 1{,}42^\circ$	$j \leftrightarrow k$	
$c \leftrightarrow d$	$\pm 1{,}91^\circ$	$k \leftrightarrow l$	
$d \leftrightarrow e$	$\pm 1{,}15^\circ$	$l \leftrightarrow m$	
$e \leftrightarrow f$	$\pm 1{,}91^\circ$	$m \leftrightarrow n$	
$f \leftrightarrow g$	$\pm 1{,}42^\circ$	$n \leftrightarrow o$	
$g \leftrightarrow h$	$\pm 1{,}91^\circ$	$o \leftrightarrow p$	
$h \leftrightarrow i$	$\pm 0{,}96^\circ$	$p \leftrightarrow a$	



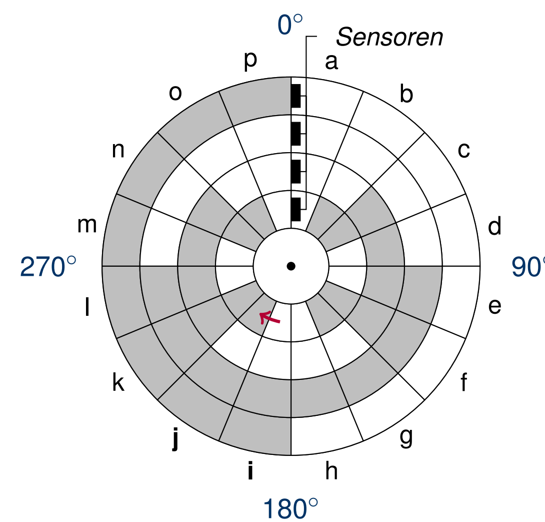
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a ↔ b	$\pm 1{,}91^\circ$	i ↔ j	
b ↔ c	$\pm 1{,}42^\circ$	j ↔ k	
c ↔ d	$\pm 1{,}91^\circ$	k ↔ l	
d ↔ e	$\pm 1{,}15^\circ$	l ↔ m	
e ↔ f	$\pm 1{,}91^\circ$	m ↔ n	
f ↔ g	$\pm 1{,}42^\circ$	n ↔ o	
g ↔ h	$\pm 1{,}91^\circ$	o ↔ p	
h ↔ i	$\pm 0{,}96^\circ$	p ↔ a	



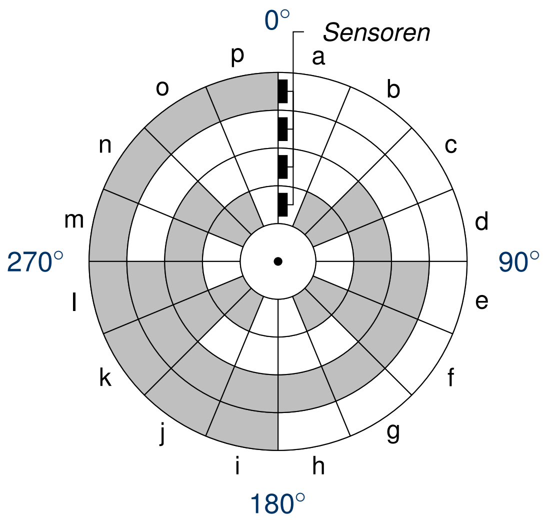
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	$\pm 1{,}91^\circ$
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	
d $\leftrightarrow$ e	$\pm 1{,}15^\circ$	l $\leftrightarrow$ m	
e $\leftrightarrow$ f	$\pm 1{,}91^\circ$	m $\leftrightarrow$ n	
f $\leftrightarrow$ g	$\pm 1{,}42^\circ$	n $\leftrightarrow$ o	
g $\leftrightarrow$ h	$\pm 1{,}91^\circ$	o $\leftrightarrow$ p	
h $\leftrightarrow$ i	$\pm 0{,}96^\circ$	p $\leftrightarrow$ a	



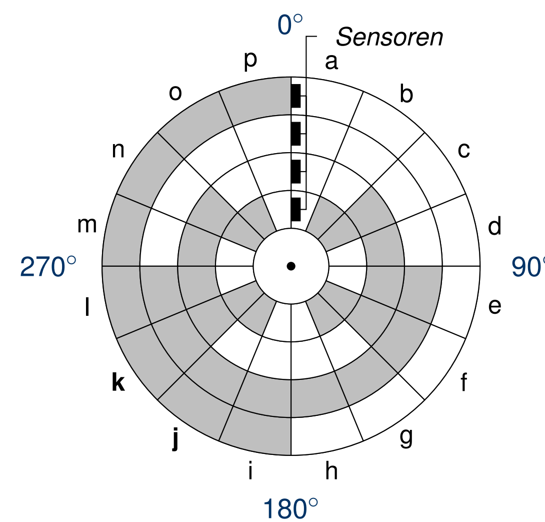
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	$\pm 1{,}91^\circ$
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	
d $\leftrightarrow$ e	$\pm 1{,}15^\circ$	l $\leftrightarrow$ m	
e $\leftrightarrow$ f	$\pm 1{,}91^\circ$	m $\leftrightarrow$ n	
f $\leftrightarrow$ g	$\pm 1{,}42^\circ$	n $\leftrightarrow$ o	
g $\leftrightarrow$ h	$\pm 1{,}91^\circ$	o $\leftrightarrow$ p	
h $\leftrightarrow$ i	$\pm 0{,}96^\circ$	p $\leftrightarrow$ a	



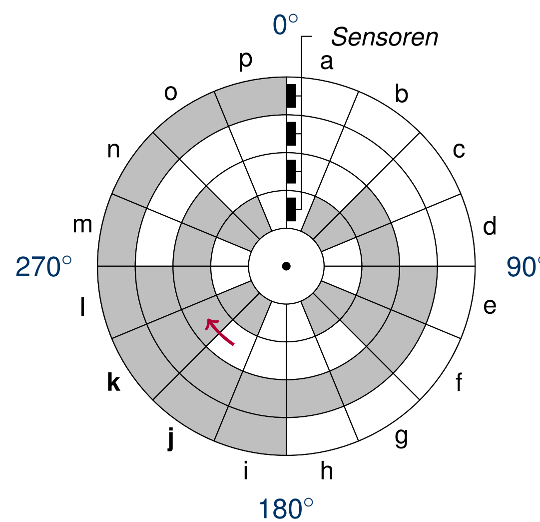
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a ↔ b	$\pm 1{,}91^\circ$	i ↔ j	$\pm 1{,}91^\circ$
b ↔ c	$\pm 1{,}42^\circ$	j ↔ k	
c ↔ d	$\pm 1{,}91^\circ$	k ↔ l	
d ↔ e	$\pm 1{,}15^\circ$	l ↔ m	
e ↔ f	$\pm 1{,}91^\circ$	m ↔ n	
f ↔ g	$\pm 1{,}42^\circ$	n ↔ o	
g ↔ h	$\pm 1{,}91^\circ$	o ↔ p	
h ↔ i	$\pm 0{,}96^\circ$	p ↔ a	



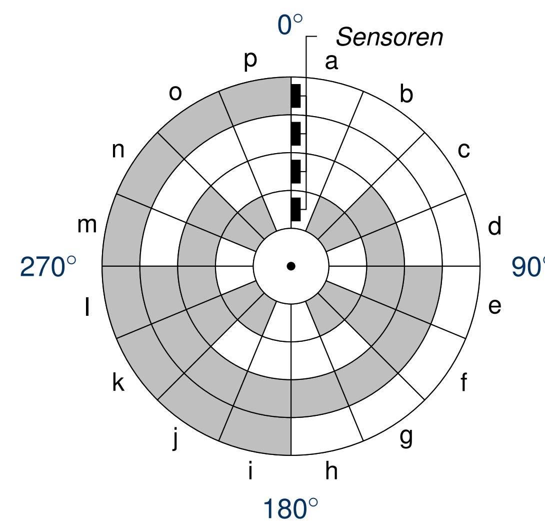
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	$\pm 1{,}91^\circ$
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	$\pm 1{,}42^\circ$
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	
d $\leftrightarrow$ e	$\pm 1{,}15^\circ$	l $\leftrightarrow$ m	
e $\leftrightarrow$ f	$\pm 1{,}91^\circ$	m $\leftrightarrow$ n	
f $\leftrightarrow$ g	$\pm 1{,}42^\circ$	n $\leftrightarrow$ o	
g $\leftrightarrow$ h	$\pm 1{,}91^\circ$	o $\leftrightarrow$ p	
h $\leftrightarrow$ i	$\pm 0{,}96^\circ$	p $\leftrightarrow$ a	



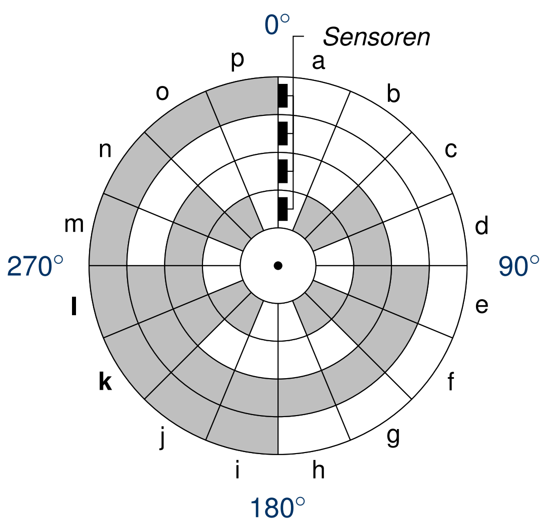
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	$\pm 1{,}91^\circ$
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	$\pm 1{,}42^\circ$
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	
d $\leftrightarrow$ e	$\pm 1{,}15^\circ$	l $\leftrightarrow$ m	
e $\leftrightarrow$ f	$\pm 1{,}91^\circ$	m $\leftrightarrow$ n	
f $\leftrightarrow$ g	$\pm 1{,}42^\circ$	n $\leftrightarrow$ o	
g $\leftrightarrow$ h	$\pm 1{,}91^\circ$	o $\leftrightarrow$ p	
h $\leftrightarrow$ i	$\pm 0{,}96^\circ$	p $\leftrightarrow$ a	



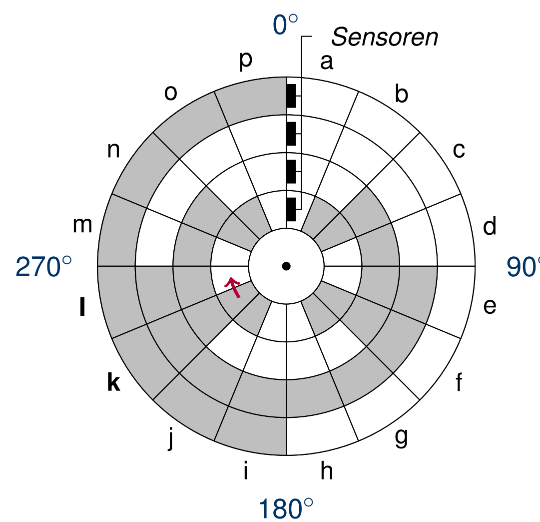
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	$\pm 1{,}91^\circ$
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	$\pm 1{,}42^\circ$
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	
d $\leftrightarrow$ e	$\pm 1{,}15^\circ$	l $\leftrightarrow$ m	
e $\leftrightarrow$ f	$\pm 1{,}91^\circ$	m $\leftrightarrow$ n	
f $\leftrightarrow$ g	$\pm 1{,}42^\circ$	n $\leftrightarrow$ o	
g $\leftrightarrow$ h	$\pm 1{,}91^\circ$	o $\leftrightarrow$ p	
h $\leftrightarrow$ i	$\pm 0{,}96^\circ$	p $\leftrightarrow$ a	



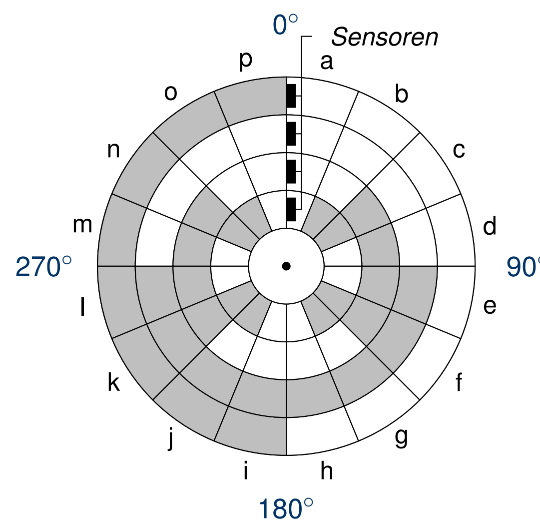
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	$\pm 1{,}91^\circ$
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	$\pm 1{,}42^\circ$
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	$\pm 1{,}91^\circ$
d $\leftrightarrow$ e	$\pm 1{,}15^\circ$	l $\leftrightarrow$ m	
e $\leftrightarrow$ f	$\pm 1{,}91^\circ$	m $\leftrightarrow$ n	
f $\leftrightarrow$ g	$\pm 1{,}42^\circ$	n $\leftrightarrow$ o	
g $\leftrightarrow$ h	$\pm 1{,}91^\circ$	o $\leftrightarrow$ p	
h $\leftrightarrow$ i	$\pm 0{,}96^\circ$	p $\leftrightarrow$ a	



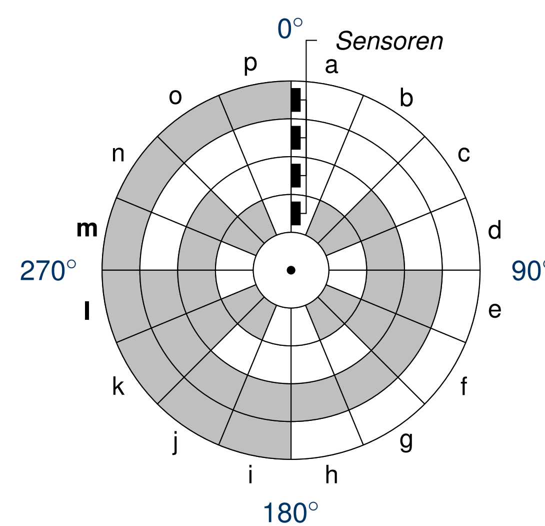
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	$\pm 1{,}91^\circ$
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	$\pm 1{,}42^\circ$
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	$\pm 1{,}91^\circ$
d $\leftrightarrow$ e	$\pm 1{,}15^\circ$	l $\leftrightarrow$ m	
e $\leftrightarrow$ f	$\pm 1{,}91^\circ$	m $\leftrightarrow$ n	
f $\leftrightarrow$ g	$\pm 1{,}42^\circ$	n $\leftrightarrow$ o	
g $\leftrightarrow$ h	$\pm 1{,}91^\circ$	o $\leftrightarrow$ p	
h $\leftrightarrow$ i	$\pm 0{,}96^\circ$	p $\leftrightarrow$ a	



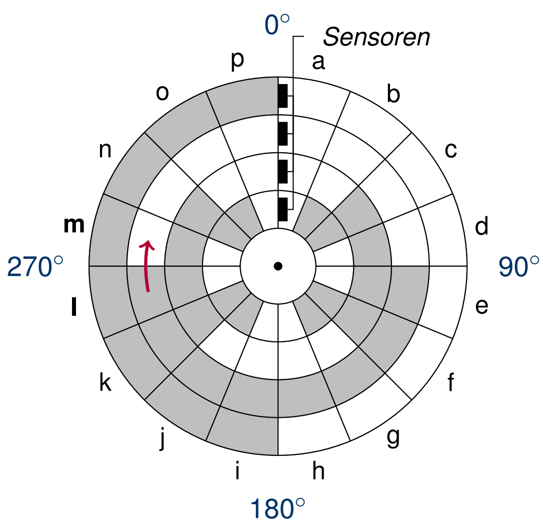
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a ↔ b	$\pm 1{,}91^\circ$	i ↔ j	$\pm 1{,}91^\circ$
b ↔ c	$\pm 1{,}42^\circ$	j ↔ k	$\pm 1{,}42^\circ$
c ↔ d	$\pm 1{,}91^\circ$	k ↔ l	$\pm 1{,}91^\circ$
d ↔ e	$\pm 1{,}15^\circ$	l ↔ m	
e ↔ f	$\pm 1{,}91^\circ$	m ↔ n	
f ↔ g	$\pm 1{,}42^\circ$	n ↔ o	
g ↔ h	$\pm 1{,}91^\circ$	o ↔ p	
h ↔ i	$\pm 0{,}96^\circ$	p ↔ a	



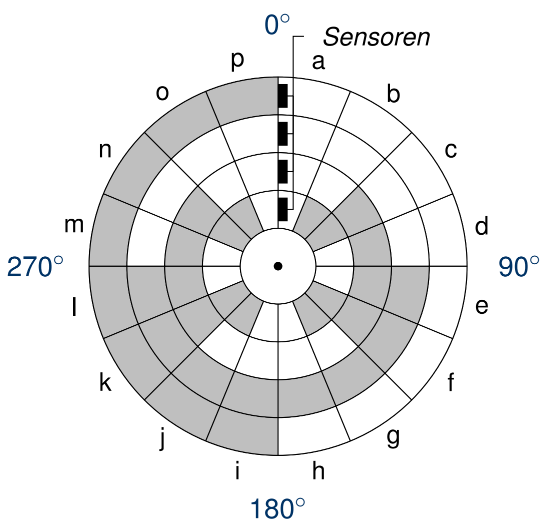
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	$\pm 1{,}91^\circ$
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	$\pm 1{,}42^\circ$
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	$\pm 1{,}91^\circ$
d $\leftrightarrow$ e	$\pm 1{,}15^\circ$	l $\leftrightarrow$ m	$\pm 1{,}15^\circ$
e $\leftrightarrow$ f	$\pm 1{,}91^\circ$	m $\leftrightarrow$ n	
f $\leftrightarrow$ g	$\pm 1{,}42^\circ$	n $\leftrightarrow$ o	
g $\leftrightarrow$ h	$\pm 1{,}91^\circ$	o $\leftrightarrow$ p	
h $\leftrightarrow$ i	$\pm 0{,}96^\circ$	p $\leftrightarrow$ a	



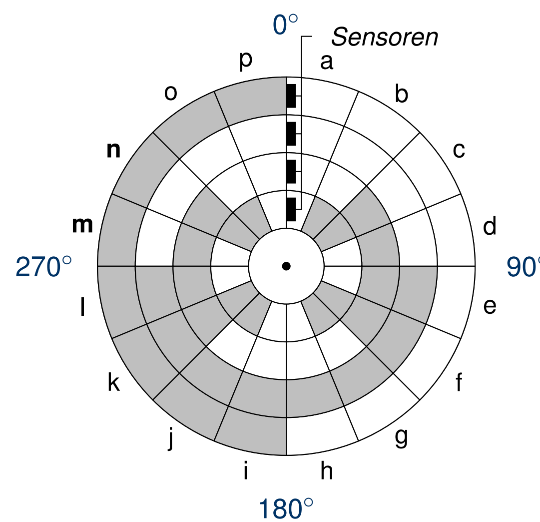
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	$\pm 1{,}91^\circ$
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	$\pm 1{,}42^\circ$
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	$\pm 1{,}91^\circ$
d $\leftrightarrow$ e	$\pm 1{,}15^\circ$	l $\leftrightarrow$ m	$\pm 1{,}15^\circ$
e $\leftrightarrow$ f	$\pm 1{,}91^\circ$	m $\leftrightarrow$ n	
f $\leftrightarrow$ g	$\pm 1{,}42^\circ$	n $\leftrightarrow$ o	
g $\leftrightarrow$ h	$\pm 1{,}91^\circ$	o $\leftrightarrow$ p	
h $\leftrightarrow$ i	$\pm 0{,}96^\circ$	p $\leftrightarrow$ a	



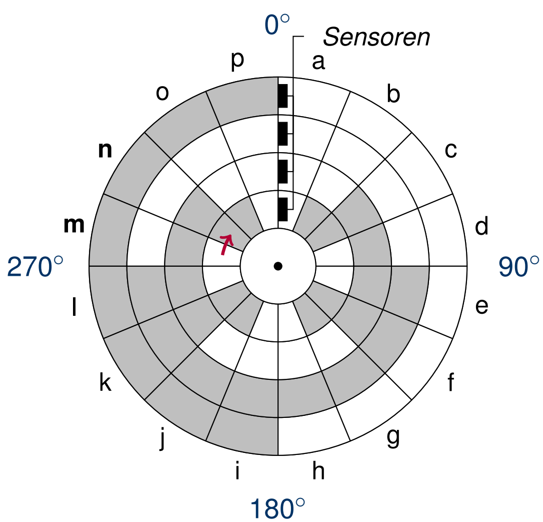
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	$\pm 1{,}91^\circ$
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	$\pm 1{,}42^\circ$
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	$\pm 1{,}91^\circ$
d $\leftrightarrow$ e	$\pm 1{,}15^\circ$	l $\leftrightarrow$ m	$\pm 1{,}15^\circ$
e $\leftrightarrow$ f	$\pm 1{,}91^\circ$	m $\leftrightarrow$ n	
f $\leftrightarrow$ g	$\pm 1{,}42^\circ$	n $\leftrightarrow$ o	
g $\leftrightarrow$ h	$\pm 1{,}91^\circ$	o $\leftrightarrow$ p	
h $\leftrightarrow$ i	$\pm 0{,}96^\circ$	p $\leftrightarrow$ a	



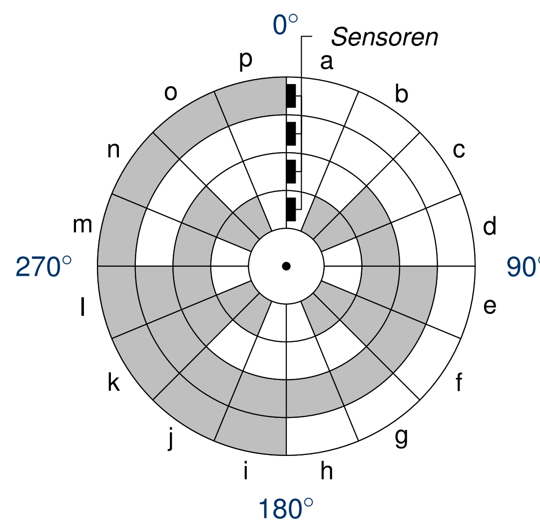
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
$a \leftrightarrow b$	$\pm 1{,}91^\circ$	$i \leftrightarrow j$	$\pm 1{,}91^\circ$
$b \leftrightarrow c$	$\pm 1{,}42^\circ$	$j \leftrightarrow k$	$\pm 1{,}42^\circ$
$c \leftrightarrow d$	$\pm 1{,}91^\circ$	$k \leftrightarrow l$	$\pm 1{,}91^\circ$
$d \leftrightarrow e$	$\pm 1{,}15^\circ$	$l \leftrightarrow m$	$\pm 1{,}15^\circ$
$e \leftrightarrow f$	$\pm 1{,}91^\circ$	$m \leftrightarrow n$	$\pm 1{,}91^\circ$
$f \leftrightarrow g$	$\pm 1{,}42^\circ$	$n \leftrightarrow o$	
$g \leftrightarrow h$	$\pm 1{,}91^\circ$	$o \leftrightarrow p$	
$h \leftrightarrow i$	$\pm 0{,}96^\circ$	$p \leftrightarrow a$	



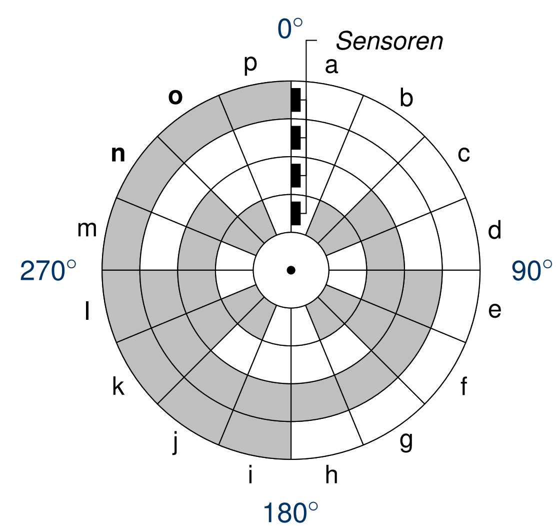
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a ↔ b	$\pm 1{,}91^\circ$	i ↔ j	$\pm 1{,}91^\circ$
b ↔ c	$\pm 1{,}42^\circ$	j ↔ k	$\pm 1{,}42^\circ$
c ↔ d	$\pm 1{,}91^\circ$	k ↔ l	$\pm 1{,}91^\circ$
d ↔ e	$\pm 1{,}15^\circ$	l ↔ m	$\pm 1{,}15^\circ$
e ↔ f	$\pm 1{,}91^\circ$	m ↔ n	$\pm 1{,}91^\circ$
f ↔ g	$\pm 1{,}42^\circ$	n ↔ o	
g ↔ h	$\pm 1{,}91^\circ$	o ↔ p	
h ↔ i	$\pm 0{,}96^\circ$	p ↔ a	



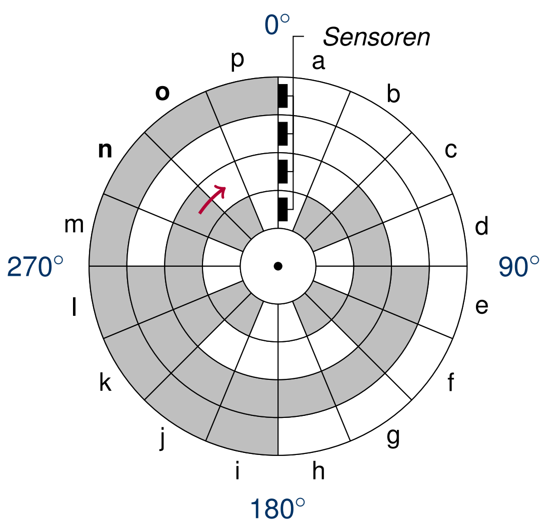
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Lösung

Übergang	α	Übergang	α
a ↔ b	$\pm 1{,}91^\circ$	i ↔ j	$\pm 1{,}91^\circ$
b ↔ c	$\pm 1{,}42^\circ$	j ↔ k	$\pm 1{,}42^\circ$
c ↔ d	$\pm 1{,}91^\circ$	k ↔ l	$\pm 1{,}91^\circ$
d ↔ e	$\pm 1{,}15^\circ$	l ↔ m	$\pm 1{,}15^\circ$
e ↔ f	$\pm 1{,}91^\circ$	m ↔ n	$\pm 1{,}91^\circ$
f ↔ g	$\pm 1{,}42^\circ$	n ↔ o	
g ↔ h	$\pm 1{,}91^\circ$	o ↔ p	
h ↔ i	$\pm 0{,}96^\circ$	p ↔ a	



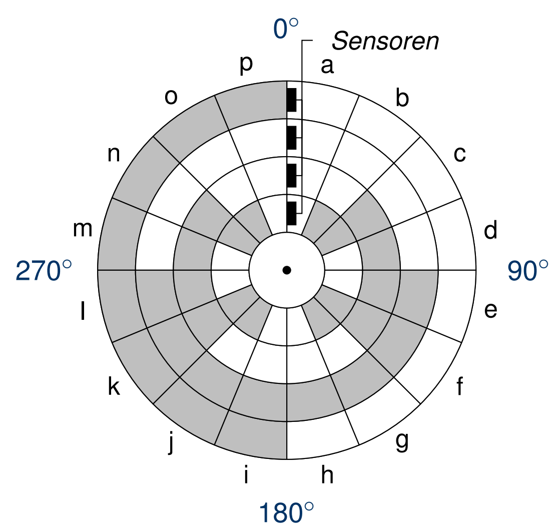
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Lösung

Übergang	α	Übergang	α
a ↔ b	$\pm 1{,}91^\circ$	i ↔ j	$\pm 1{,}91^\circ$
b ↔ c	$\pm 1{,}42^\circ$	j ↔ k	$\pm 1{,}42^\circ$
c ↔ d	$\pm 1{,}91^\circ$	k ↔ l	$\pm 1{,}91^\circ$
d ↔ e	$\pm 1{,}15^\circ$	l ↔ m	$\pm 1{,}15^\circ$
e ↔ f	$\pm 1{,}91^\circ$	m ↔ n	$\pm 1{,}91^\circ$
f ↔ g	$\pm 1{,}42^\circ$	n ↔ o	$\pm 1{,}42^\circ$
g ↔ h	$\pm 1{,}91^\circ$	o ↔ p	
h ↔ i	$\pm 0{,}96^\circ$	p ↔ a	



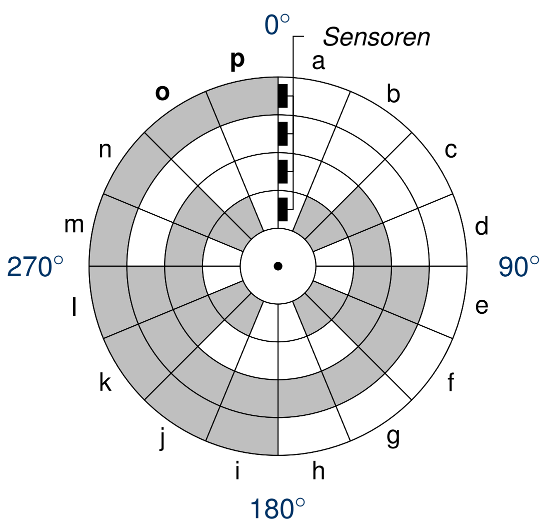
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a ↔ b	$\pm 1{,}91^\circ$	i ↔ j	$\pm 1{,}91^\circ$
b ↔ c	$\pm 1{,}42^\circ$	j ↔ k	$\pm 1{,}42^\circ$
c ↔ d	$\pm 1{,}91^\circ$	k ↔ l	$\pm 1{,}91^\circ$
d ↔ e	$\pm 1{,}15^\circ$	l ↔ m	$\pm 1{,}15^\circ$
e ↔ f	$\pm 1{,}91^\circ$	m ↔ n	$\pm 1{,}91^\circ$
f ↔ g	$\pm 1{,}42^\circ$	n ↔ o	$\pm 1{,}42^\circ$
g ↔ h	$\pm 1{,}91^\circ$	o ↔ p	
h ↔ i	$\pm 0{,}96^\circ$	p ↔ a	



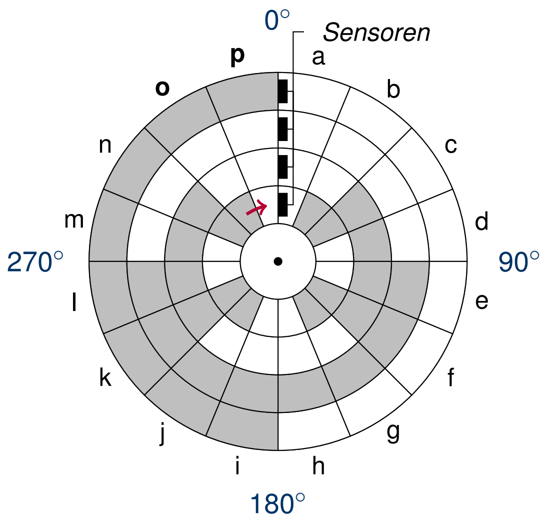
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Lösung

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	$\pm 1{,}91^\circ$
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	$\pm 1{,}42^\circ$
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	$\pm 1{,}91^\circ$
d $\leftrightarrow$ e	$\pm 1{,}15^\circ$	l $\leftrightarrow$ m	$\pm 1{,}15^\circ$
e $\leftrightarrow$ f	$\pm 1{,}91^\circ$	m $\leftrightarrow$ n	$\pm 1{,}91^\circ$
f $\leftrightarrow$ g	$\pm 1{,}42^\circ$	n $\leftrightarrow$ o	$\pm 1{,}42^\circ$
g $\leftrightarrow$ h	$\pm 1{,}91^\circ$	o $\leftrightarrow$ p	
h $\leftrightarrow$ i	$\pm 0{,}96^\circ$	p $\leftrightarrow$ a	



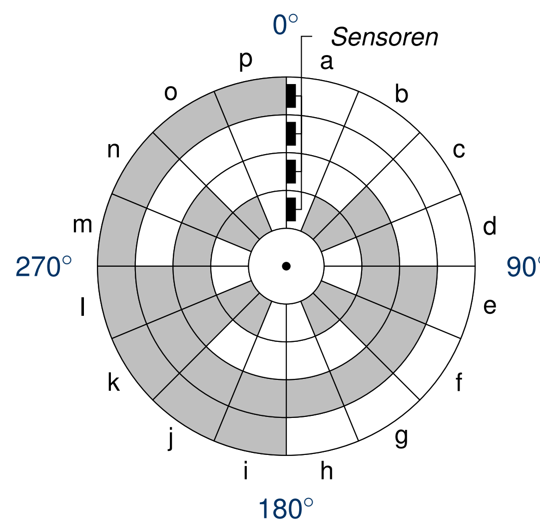
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	$\pm 1{,}91^\circ$
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	$\pm 1{,}42^\circ$
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	$\pm 1{,}91^\circ$
d $\leftrightarrow$ e	$\pm 1{,}15^\circ$	l $\leftrightarrow$ m	$\pm 1{,}15^\circ$
e $\leftrightarrow$ f	$\pm 1{,}91^\circ$	m $\leftrightarrow$ n	$\pm 1{,}91^\circ$
f $\leftrightarrow$ g	$\pm 1{,}42^\circ$	n $\leftrightarrow$ o	$\pm 1{,}42^\circ$
g $\leftrightarrow$ h	$\pm 1{,}91^\circ$	o $\leftrightarrow$ p	$\pm 1{,}91^\circ$
h $\leftrightarrow$ i	$\pm 0{,}96^\circ$	p $\leftrightarrow$ a	



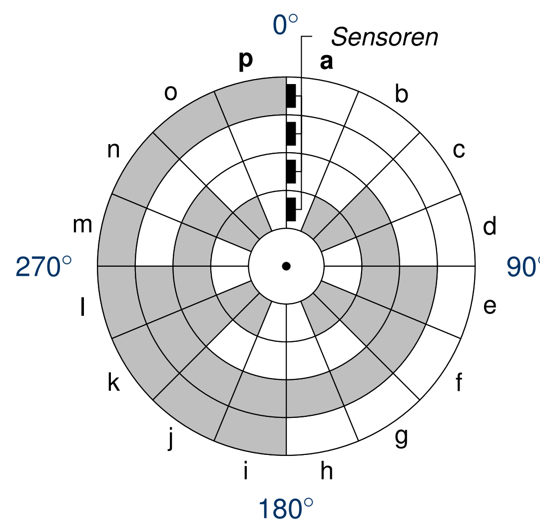
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von $\pm$ 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a $\leftrightarrow$ b	$\pm 1{,}91^\circ$	i $\leftrightarrow$ j	$\pm 1{,}91^\circ$
b $\leftrightarrow$ c	$\pm 1{,}42^\circ$	j $\leftrightarrow$ k	$\pm 1{,}42^\circ$
c $\leftrightarrow$ d	$\pm 1{,}91^\circ$	k $\leftrightarrow$ l	$\pm 1{,}91^\circ$
d $\leftrightarrow$ e	$\pm 1{,}15^\circ$	l $\leftrightarrow$ m	$\pm 1{,}15^\circ$
e $\leftrightarrow$ f	$\pm 1{,}91^\circ$	m $\leftrightarrow$ n	$\pm 1{,}91^\circ$
f $\leftrightarrow$ g	$\pm 1{,}42^\circ$	n $\leftrightarrow$ o	$\pm 1{,}42^\circ$
g $\leftrightarrow$ h	$\pm 1{,}91^\circ$	o $\leftrightarrow$ p	$\pm 1{,}91^\circ$
h $\leftrightarrow$ i	$\pm 0{,}96^\circ$	p $\leftrightarrow$ a	



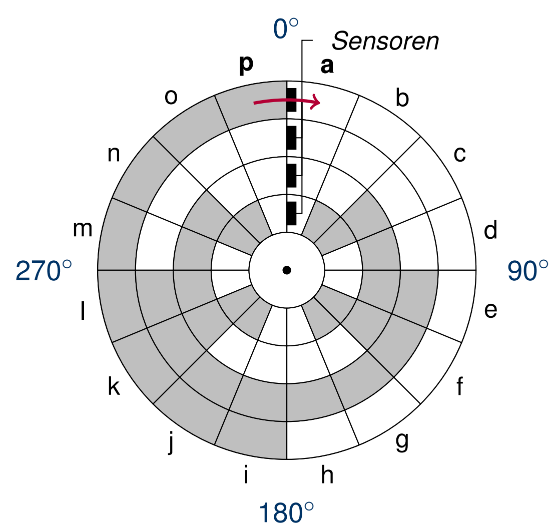
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
a ↔ b	$\pm 1{,}91^\circ$	i ↔ j	$\pm 1{,}91^\circ$
b ↔ c	$\pm 1{,}42^\circ$	j ↔ k	$\pm 1{,}42^\circ$
c ↔ d	$\pm 1{,}91^\circ$	k ↔ l	$\pm 1{,}91^\circ$
d ↔ e	$\pm 1{,}15^\circ$	l ↔ m	$\pm 1{,}15^\circ$
e ↔ f	$\pm 1{,}91^\circ$	m ↔ n	$\pm 1{,}91^\circ$
f ↔ g	$\pm 1{,}42^\circ$	n ↔ o	$\pm 1{,}42^\circ$
g ↔ h	$\pm 1{,}91^\circ$	o ↔ p	$\pm 1{,}91^\circ$
h ↔ i	$\pm 0{,}96^\circ$	p ↔ a	



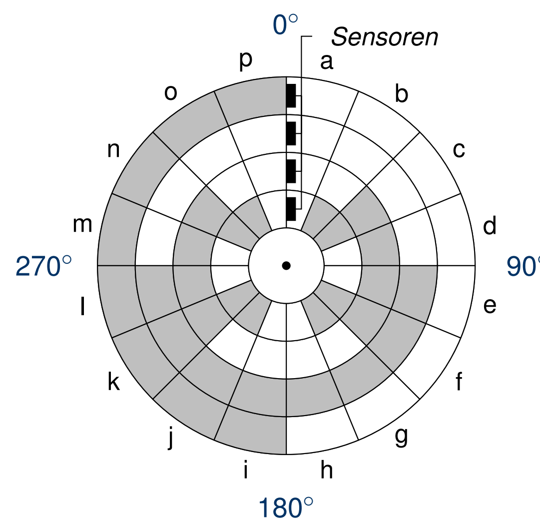
Aufgabe 3 – Genauigkeit und Kodierung

c) Wieviele Winkelgrade umfassen die undefinierten Bereiche jeweils, wenn man annimmt, dass die Schleifkontakte auf den Radien 3, 4, 5 und 6 mm liegen und in tangentialer Richtung eine Toleranz von ± 1 mm aufweisen?

Lösung

Radius r	Winkel α
3	$\pm 1{,}91^\circ$
4	$\pm 1{,}42^\circ$
5	$\pm 1{,}15^\circ$
6	$\pm 0{,}96^\circ$

Übergang	α	Übergang	α
$a \leftrightarrow b$	$\pm 1{,}91^\circ$	$i \leftrightarrow j$	$\pm 1{,}91^\circ$
$b \leftrightarrow c$	$\pm 1{,}42^\circ$	$j \leftrightarrow k$	$\pm 1{,}42^\circ$
$c \leftrightarrow d$	$\pm 1{,}91^\circ$	$k \leftrightarrow l$	$\pm 1{,}91^\circ$
$d \leftrightarrow e$	$\pm 1{,}15^\circ$	$l \leftrightarrow m$	$\pm 1{,}15^\circ$
$e \leftrightarrow f$	$\pm 1{,}91^\circ$	$m \leftrightarrow n$	$\pm 1{,}91^\circ$
$f \leftrightarrow g$	$\pm 1{,}42^\circ$	$n \leftrightarrow o$	$\pm 1{,}42^\circ$
$g \leftrightarrow h$	$\pm 1{,}91^\circ$	$o \leftrightarrow p$	$\pm 1{,}91^\circ$
$h \leftrightarrow i$	$\pm 0{,}96^\circ$	$p \leftrightarrow a$	$\pm 0{,}96^\circ$



Aufgabe 3 – Genauigkeit und Kodierung

d) Wie viele Schleifkontakte werden benötigt, wenn die Winkelauflösung auf 1°genau erfolgen soll? Diskutieren Sie dabei auftretende Probleme.

Aufgabe 3 – Genauigkeit und Kodierung

d) Wie viele Schleifkontakte werden benötigt, wenn die Winkelauflösung auf 1°genau erfolgen soll?
Lösung

Aufgabe 3 – Genauigkeit und Kodierung

d) Wie viele Schleifkontakte werden benötigt, wenn die Winkelauflösung auf 1°genau erfolgen soll?

Lösung

Als erstes ist die Anzahl an benötigten Segmenten gesucht.

Aufgabe 3 – Genauigkeit und Kodierung

d) Wie viele Schleifkontakte werden benötigt, wenn die Winkelauflösung auf 1°genau erfolgen soll?

Lösung

Als erstes ist die Anzahl an benötigten Segmenten gesucht. Es gilt:

$$\#\text{Segmente} = \frac{\varphi_{\text{Gesamt}}}{\text{Auflösung}} = \frac{360^\circ}{1^\circ} = 360$$

Aufgabe 3 – Genauigkeit und Kodierung

d) Wie viele Schleifkontakte werden benötigt, wenn die Winkelauflösung auf 1°genau erfolgen soll?

Lösung

Als erstes ist die Anzahl an benötigten Segmenten gesucht. Es gilt:

$$\#\text{Segmente} = \frac{\varphi_{\text{Gesamt}}}{\text{Auflösung}} = \frac{\overbrace{360^\circ}^{\text{Ein Vollkreis}}}{1^\circ} = 360$$

Aufgabe 3 – Genauigkeit und Kodierung

d) Wie viele Schleifkontakte werden benötigt, wenn die Winkelauflösung auf 1°genau erfolgen soll?

Lösung

Für die 360 Segmente braucht man dann *n* Taster, welche wiederum jeweils eine Binärstelle repräsentieren.

Aufgabe 3 – Genauigkeit und Kodierung

d) Wie viele Schleifkontakte werden benötigt, wenn die Winkelauflösung auf 1°genau erfolgen soll?

Lösung

Für die 360 Segmente braucht man dann n Taster, welche wiederum jeweils eine Binärstelle repräsentieren.

$$360 = 2^n \qquad \equiv \qquad n = \lceil \log_2 360 \rceil = 9$$

Aufgabe 3 – Genauigkeit und Kodierung

d) Wie viele Schleifkontakte werden benötigt, wenn die Winkelauflösung auf 1°genau erfolgen soll?
Lösung
Für die 360 Segmente braucht man dann n Taster, welche wiederum jeweils eine Binärstelle repräsentieren.

$$360 = 2^n \qquad \equiv \qquad n = \lceil \log_2 360 \rceil = 9$$

$\rightsquigarrow$ Man benötigt 9 Schleifkontakte

Aufgabe 3 – Genauigkeit und Kodierung

d) Wie viele Schleifkontakte werden benötigt, wenn die Winkelauflösung auf 1°genau erfolgen soll? Diskutieren Sie dabei auftretende Probleme.

Lösung

Für die 360 Segmente braucht man dann n Taster, welche wiederum jeweils eine Binärstelle repräsentieren.

$$360 = 2^n \qquad \equiv \qquad n = \lceil \log_2 360 \rceil = 9$$

$\rightsquigarrow$ Man benötigt 9 Schleifkontakte

Probleme:

Aufgabe 3 – Genauigkeit und Kodierung

d) Wie viele Schleifkontakte werden benötigt, wenn die Winkelauflösung auf 1°genau erfolgen soll? Diskutieren Sie dabei auftretende Probleme.

Lösung

Für die 360 Segmente braucht man dann n Taster, welche wiederum jeweils eine Binärstelle repräsentieren.

$$360 = 2^n \qquad \equiv \qquad n = \lceil \log_2 360 \rceil = 9$$

$\leadsto$ Man benötigt 9 Schleifkontakte

Probleme:

Ein Problem stellen die undefinierten Bereiche der Schleifkontakte da (siehe ein paar Folien zurück, denn $1^\circ << 1{,}91^\circ$)

$\leadsto$ Lösungsmöglichkeit ist eine drastische Erhöhung des Umfangs (eventuell aufgrund von Anforderungen aber nicht möglich)!

Aufgabe 3 – Genauigkeit und Kodierung

d) Wie viele Schleifkontakte werden benötigt, wenn die Winkelauflösung auf 1°genau erfolgen soll? Diskutieren Sie dabei auftretende Probleme.

Lösung

Für die 360 Segmente braucht man dann *n* Taster, welche wiederum jeweils eine Binärstelle repräsentieren.

$$360 = 2^n \qquad \equiv \qquad n = \lceil \log_2 360 \rceil = 9$$

$\leadsto$ Man benötigt 9 Schleifkontakte

Probleme:

Ein Problem stellen die undefinierten Bereiche der Schleifkontakte da (siehe ein paar Folien zurück, denn $1^\circ << 1{,}91^\circ$)

$\leadsto$ Lösungsmöglichkeit ist eine drastische Erhöhung des Umfangs (eventuell aufgrund von Anforderungen aber nicht möglich)!

Ein weiteres Problem ist, dass unser wunderbarer Gray-Kode nicht mehr ohne Anpassung funktioniert ($2^9 = 512 > 360$).

$\leadsto$ Lösungsmöglichkeit: Wir verschieben unseren Gray-Kode

Aufgabe 3 – Exkurs: Verschieben des Gray-Kodes

- Wir sehen rechts einen „normalen“ Gray-Kode mit 4 Binärstellen, also insgesamt $2^4 = 16$ Wörter.

0000
0001
0011
0010
0110
0111
0101
0100

1100
1101
1111
1110
1010
1011
1001
1000

Aufgabe 3 – Exkurs: Verschieben des Gray-Kodes

- Wir sehen rechts einen „normalen“ Gray-Kode mit 4 Binärstellen, also insgesamt $2^4 = 16$ Wörter.
- Wir sehen, dass aufgrund der Symmetrie sich bei zwei Wörtern, welche den gleichen Abstand zur Achse haben, sich auch nur ein Bit ändert (das äußerste).

0000
0001
0011
0010
0110
0111
0101
0100

1100
1101
1111
1110
1010
1011
1001
1000

Aufgabe 3 – Exkurs: Verschieben des Gray-Kodes

- Wir sehen rechts einen „normalen“ Gray-Kode mit 4 Binärstellen, also insgesamt $2^4 = 16$ Wörter.
- Wir sehen, dass aufgrund der Symmetrie sich bei zwei Wörtern, welche den gleichen Abstand zur Achse haben, sich auch nur ein Bit ändert (das äußerste).

0000
0001
→0011
0010
0110
0111
0101
0100

1100
1101
1111
1110
1010
→1011
1001
1000

Aufgabe 3 – Exkurs: Verschieben des Gray-Kodes

- Wir sehen rechts einen „normalen“ Gray-Kode mit 4 Binärstellen, also insgesamt $2^4 = 16$ Wörter.
- Wir sehen, dass aufgrund der Symmetrie sich bei zwei Wörtern, welche den gleichen Abstand zur Achse haben, sich auch nur ein Bit ändert (das äußerste).
- Möchte man also bei einem n-Bit Gray-Kode (mit 2^n Wörtern) nur m Wörter eigentlich kodieren und das nach Gray, so geht man genau $\frac{2^n - m}{2}$ Wörter von der Symmetrieachse sowohl nach oben als auch nach unten und lässt diese aus.

0000
0001
0011
0010
0110
0111
0101
0100

1100
1101
1111
1110
1010
1011
1001
1000

Aufgabe 3 – Exkurs: Verschieben des Gray-Kodes

- Wir sehen rechts einen „normalen“ Gray-Kode mit 4 Binärstellen, also insgesamt $2^4 = 16$ Wörter.
- Wir sehen, dass aufgrund der Symmetrie sich bei zwei Wörtern, welche den gleichen Abstand zur Achse haben, sich auch nur ein Bit ändert (das äußerste).
- Möchte man also bei einem n-Bit Gray-Kode (mit 2^n Wörtern) nur m Wörter eigentlich kodieren und das nach Gray, so geht man genau $\frac{2^n - m}{2}$ Wörter von der Symmetrieachse sowohl nach oben als auch nach unten und lässt diese aus.
- Beispiel: Wir wollen mit 4 Bits nur 10 Wörter kodieren $\rightsquigarrow$ Wir gehen $\frac{2^4 - 10}{2} = 3$ Wörter nach oben und unten und entfernen diese.

0000
0001
0011
0010
0110
0111
0101
0100

1100
1101
1111
1110
1010
1011
1001
1000

Aufgabe 3 – Exkurs: Verschieben des Gray-Kodes

- Wir sehen rechts einen „normalen“ Gray-Kode mit 4 Binärstellen, also insgesamt $2^4 = 16$ Wörter.
- Wir sehen, dass aufgrund der Symmetrie sich bei zwei Wörtern, welche den gleichen Abstand zur Achse haben, sich auch nur ein Bit ändert (das äußerste).
- Möchte man also bei einem n-Bit Gray-Kode (mit 2^n Wörtern) nur m Wörter eigentlich kodieren und das nach Gray, so geht man genau $\frac{2^n - m}{2}$ Wörter von der Symmetrieachse sowohl nach oben als auch nach unten und lässt diese aus.
- Beispiel: Wir wollen mit 4 Bits nur 10 Wörter kodieren $\rightsquigarrow$ Wir gehen $\frac{2^4 - 10}{2} = 3$ Wörter nach oben und unten und entfernen diese.

0000
0001
0011
0010
0110
0111
0101
~~0100~~

~~1100~~
1101
1111
1110
1010
1011
1001
1000

Aufgabe 3 – Exkurs: Verschieben des Gray-Kodes

- Wir sehen rechts einen „normalen“ Gray-Kode mit 4 Binärstellen, also insgesamt $2^4 = 16$ Wörter.
- Wir sehen, dass aufgrund der Symmetrie sich bei zwei Wörtern, welche den gleichen Abstand zur Achse haben, sich auch nur ein Bit ändert (das äußerste).
- Möchte man also bei einem n-Bit Gray-Kode (mit 2^n Wörtern) nur m Wörter eigentlich kodieren und das nach Gray, so geht man genau $\frac{2^n - m}{2}$ Wörter von der Symmetrieachse sowohl nach oben als auch nach unten und lässt diese aus.
- Beispiel: Wir wollen mit 4 Bits nur 10 Wörter kodieren $\rightsquigarrow$ Wir gehen $\frac{2^4 - 10}{2} = 3$ Wörter nach oben und unten und entfernen diese.

0000
0001
0011
0010
0110
0111
~~0101~~
~~0100~~

~~1100~~
~~1101~~
1111
1110
1010
1011
1001
1000

Aufgabe 3 – Exkurs: Verschieben des Gray-Kodes

- Wir sehen rechts einen „normalen“ Gray-Kode mit 4 Binärstellen, also insgesamt $2^4 = 16$ Wörter.
- Wir sehen, dass aufgrund der Symmetrie sich bei zwei Wörtern, welche den gleichen Abstand zur Achse haben, sich auch nur ein Bit ändert (das äußerste).
- Möchte man also bei einem n-Bit Gray-Kode (mit 2^n Wörtern) nur m Wörter eigentlich kodieren und das nach Gray, so geht man genau $\frac{2^n - m}{2}$ Wörter von der Symmetrieachse sowohl nach oben als auch nach unten und lässt diese aus.
- Beispiel: Wir wollen mit 4 Bits nur 10 Wörter kodieren $\leadsto$ Wir gehen $\frac{2^4 - 10}{2} = 3$ Wörter nach oben und unten und entfernen diese.

0000
0001
0011
0010
0110
~~0111~~
~~0101~~
~~0100~~

~~1100~~
~~1101~~
~~1111~~
1110
1010
1011
1001
1000

Aufgabe 3 – Exkurs: Verschieben des Gray-Kodes

- Wir sehen rechts einen „normalen" Gray-Kode mit 4 Binärstellen, also insgesamt $2^4 = 16$ Wörter.
- Wir sehen, dass aufgrund der Symmetrie sich bei zwei Wörtern, welche den gleichen Abstand zur Achse haben, sich auch nur ein Bit ändert (das äußerste).
- Möchte man also bei einem n-Bit Gray-Kode (mit 2^n Wörtern) nur m Wörter eigentlich kodieren und das nach Gray, so geht man genau $\frac{2^n - m}{2}$ Wörter von der Symmetrieachse sowohl nach oben als auch nach unten und lässt diese aus.
- Beispiel: Wir wollen mit 4 Bits nur 10 Wörter kodieren $\rightsquigarrow$ Wir gehen $\frac{2^4 - 10}{2} = 3$ Wörter nach oben und unten und entfernen diese.
- Problem: Es funktioniert nur für gerade Zahlen!

0000
0001
0011
0010
0110
~~0111~~
~~0101~~
~~0100~~

~~1100~~
~~1101~~
~~1111~~
1110
1010
1011
1001
1000

Übungen zur Grundlagen der Technischen Informatik

Übung 2 – Fehlererkennung, Fehlerkorrektur und Huffman

Florian Frank

Friedrich-Alexander-Universität Erlangen-Nürnberg

Wintersemester 2018/19

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Was machen wir heute?

Aufgabe 1 – Hamming-Distanz

Was machen wir heute?

Aufgabe 1 – Hamming-Distanz

Aufgabe 2 – Fehlererkennung

Was machen wir heute?

Aufgabe 1 – Hamming-Distanz

Aufgabe 2 – Fehlererkennung

Aufgabe 3 – Blocksicherung

Was machen wir heute?

Aufgabe 1 – Hamming-Distanz

Aufgabe 2 – Fehlererkennung

Aufgabe 3 – Blocksicherung

Aufgabe 4 – Fehlerkorrektur

Was machen wir heute?

Aufgabe 1 – Hamming-Distanz

Aufgabe 2 – Fehlererkennung

Aufgabe 3 – Blocksicherung

Aufgabe 4 – Fehlerkorrektur

Aufgabe 5 – Huffman-Code

Aufgabe 1 – Hamming-Distanz

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 1 – Hamming-Distanz (Grundlagen)

Definition 1 (Hamming-Distanz)

Seien x und y zwei gleich lange Wörter der Länge n. Der Hamming-Abstand $\Delta(x, y)$ ist definiert als:

$$\Delta(x, y) := \left|\{j \in \{1, ..., n\} \mid x_j \neq y_j\}\right|$$

Aufgabe 1 – Hamming-Distanz (Grundlagen)

Definition 1 (Hamming-Distanz)

Seien x und y zwei gleich lange Wörter der Länge n. Der Hamming-Abstand $\Delta(x, y)$ ist definiert als:

$$\Delta(x, y) := \left|\{j \in \{1, ..., n\} \mid x_j \neq y_j\}\right|$$

In einfacheren Worten: Die Hamming-Distanz gibt die Anzahl an **unterschiedlichen** Binärstellen in zwei **gleichlangen** Codewörtern an.

Aufgabe 1 – Hamming-Distanz (Grundlagen)

Definition 1 (Hamming-Distanz)

Seien x und y zwei gleich lange Wörter der Länge n. Der Hamming-Abstand $\Delta(x, y)$ ist definiert als:

$$\Delta(x, y) := \left|\{j \in \{1, \ldots, n\} \mid x_j \neq y_j\}\right|$$

In einfacheren Worten: Die Hamming-Distanz gibt die Anzahl an **unterschiedlichen** Binärstellen in zwei **gleichlangen** Codewörtern an.

Definition 2 (Minimale Hamming-Distanz $\mathrm{HD}_{\mathrm{min}}$)

Die minimale Hamming-Distanz $\mathrm{HD}_{\mathrm{min}}$ ist wie folgt definiert:

$$\mathrm{HD}_{\mathrm{min}} = \min \left\{\Delta(x_i, x_j) \mid \forall x_i, x_j \in \mathcal{C}\right\}$$

Aufgabe 1 – Hamming-Distanz (Grundlagen)

Definition 1 (Hamming-Distanz)

Seien x und y zwei gleich lange Wörter der Länge n. Der Hamming-Abstand $\Delta(x, y)$ ist definiert als:

$$\Delta(x, y) := \left|\{j \in \{1, ..., n\} \mid x_j \neq y_j\}\right|$$

In einfacheren Worten: Die Hamming-Distanz gibt die Anzahl an **unterschiedlichen** Binärstellen in zwei **gleichlangen** Codewörtern an.

Definition 2 (Minimale Hamming-Distanz HD_{min})

Die minimale Hamming-Distanz HD_{min} ist wie folgt definiert:

$$HD_{min} = \min\left\{\Delta(x_i, x_j) \mid \forall x_i, x_j \in \mathcal{C}\right\}$$

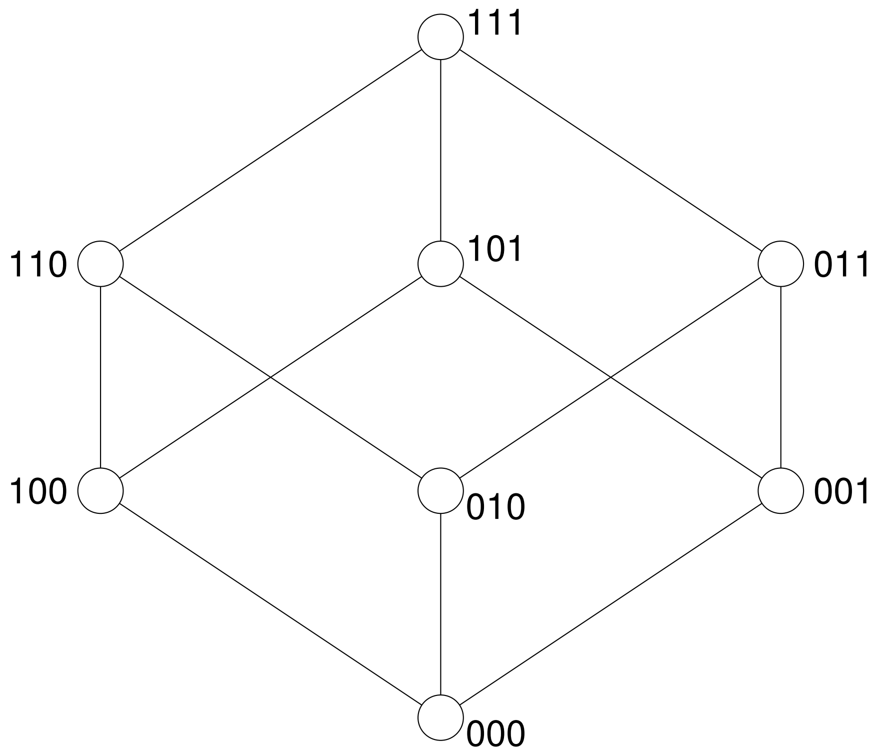
In einfacheren Worten: Die minimale Hamming-Distanz ist das Minimum der Hamming-Distanzen zwischen **allen Codewörtern** eines Codes.

Definition 3 (N-fach Fehler)

Wenn sich durch externe Störeinflüsse bei einem vorher korrekten Binärwort $\mathcal{BW}$ N unterschiedliche Stellen ändern, so spricht man von einem N-fach Fehler.

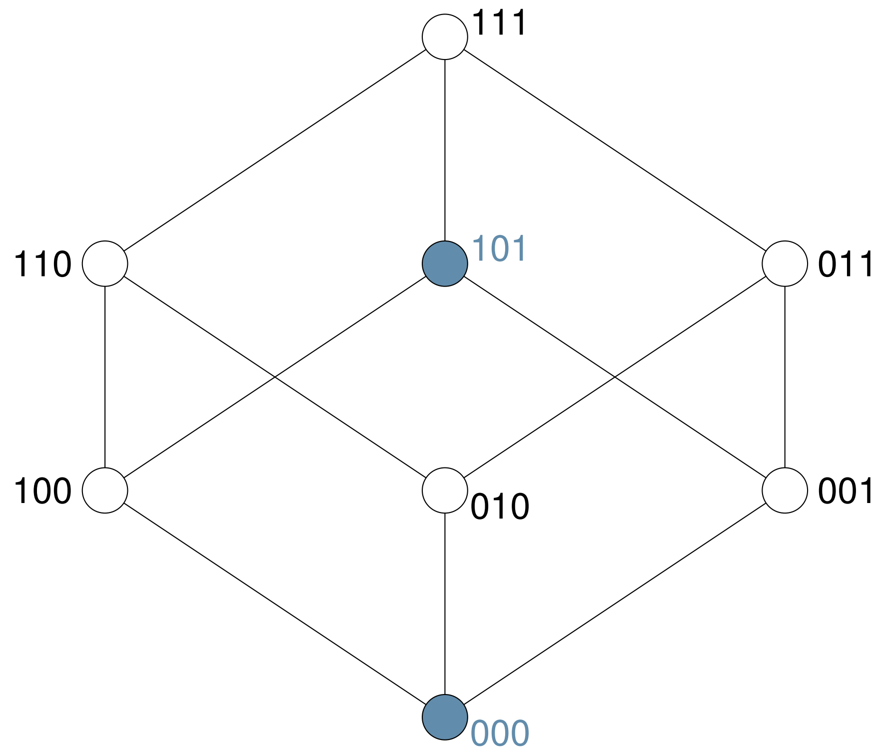
Aufgabe 1 – Hamming-Distanz

Diagramm der Nachbarschaftsbeziehungen für einen Kode mit drei Binärstellen.



Aufgabe 1 – Hamming-Distanz

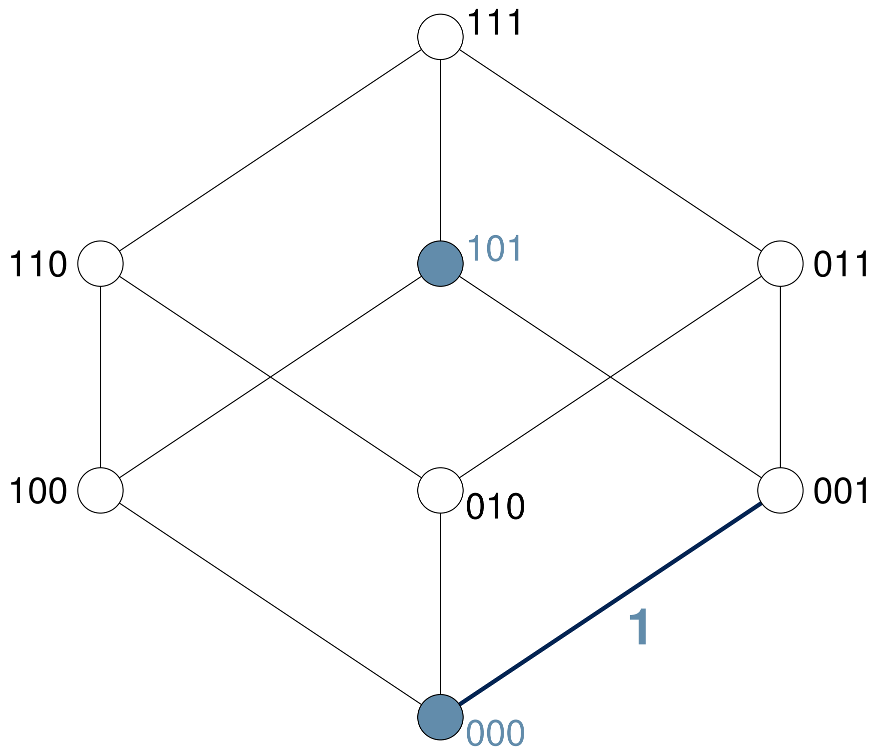
Diagramm der Nachbarschaftsbeziehungen für einen Kode mit drei Binärstellen.



Welche Hamming-Distanz weisen die Kodewörter `000` und `101` auf?

Aufgabe 1 – Hamming-Distanz

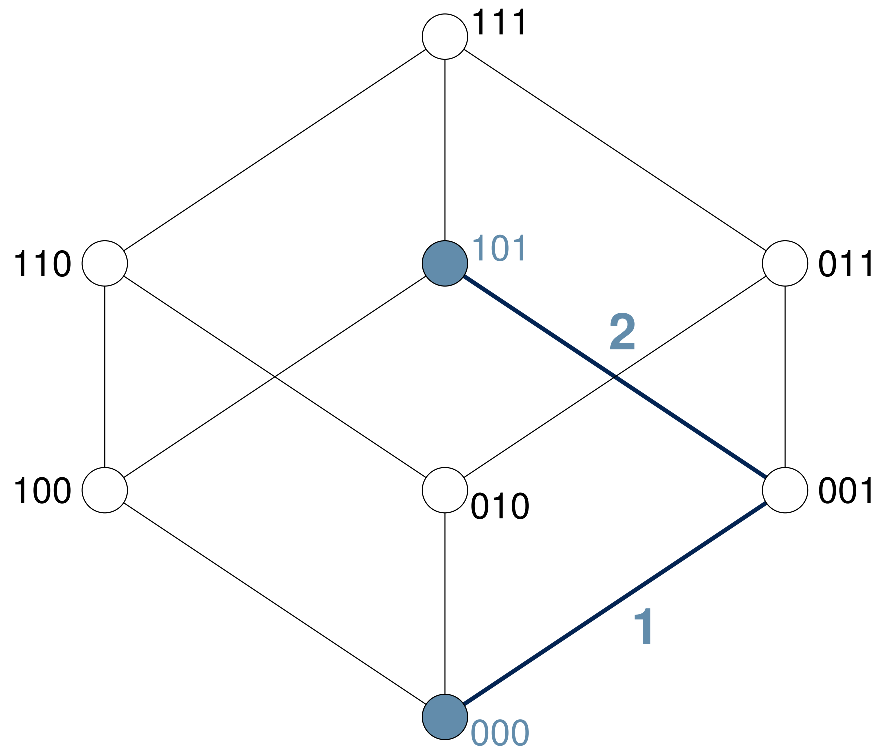
Diagramm der Nachbarschaftsbeziehungen für einen Kode mit drei Binärstellen.



Welche Hamming-Distanz weisen die Kodewörter `000` und `101` auf?

Aufgabe 1 – Hamming-Distanz

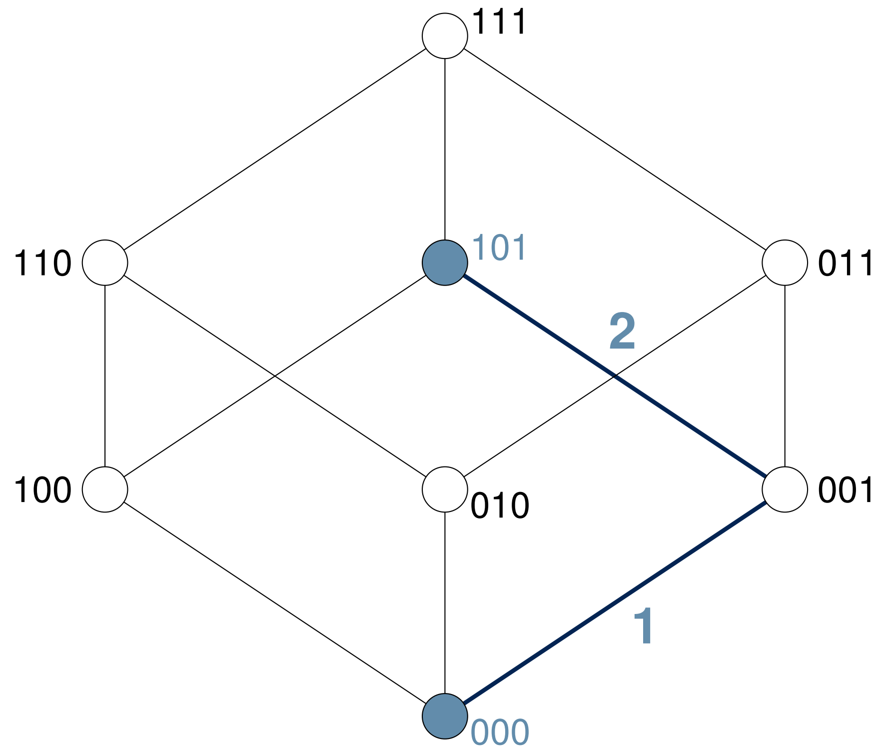
Diagramm der Nachbarschaftsbeziehungen für einen Kode mit drei Binärstellen.



Welche Hamming-Distanz weisen die Kodewörter `000` und `101` auf?

Aufgabe 1 – Hamming-Distanz

Diagramm der Nachbarschaftsbeziehungen für einen Kode mit drei Binärstellen.

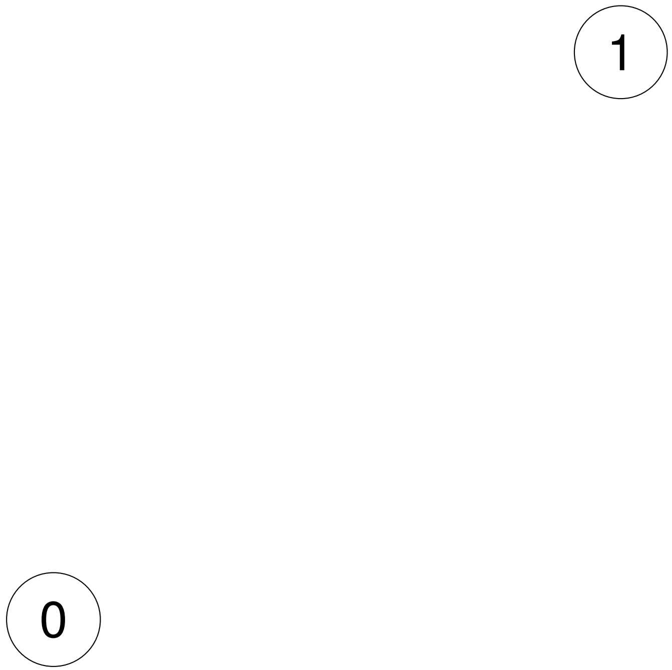


Welche Hamming-Distanz weisen die Kodewörter `000` und `101` auf?

HD(`000`,`101`) = 2

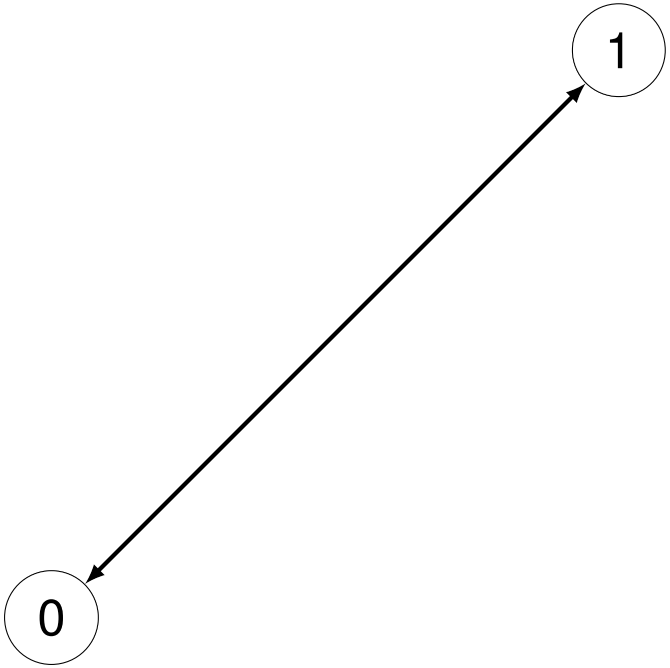
Aufgabe 1 – Hamming-Distanz

a) Welche (minimale) Hamming-Distanz müssen die gültigen Kodewörter aufweisen, damit Einzelfehler erkannt werden können?



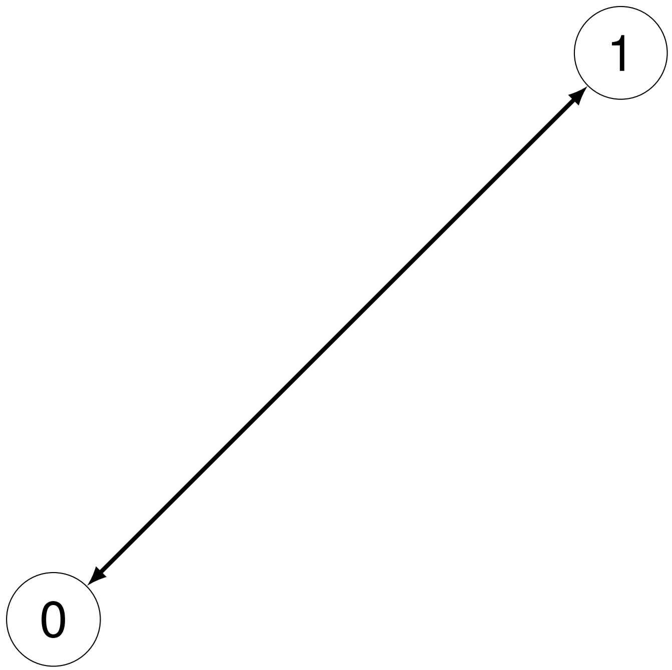
Aufgabe 1 – Hamming-Distanz

a) Welche (minimale) Hamming-Distanz müssen die gültigen Kodewörter aufweisen, damit Einzelfehler erkannt werden können?



Aufgabe 1 – Hamming-Distanz

a) Welche (minimale) Hamming-Distanz müssen die gültigen Kodewörter aufweisen, damit Einzelfehler erkannt werden können?



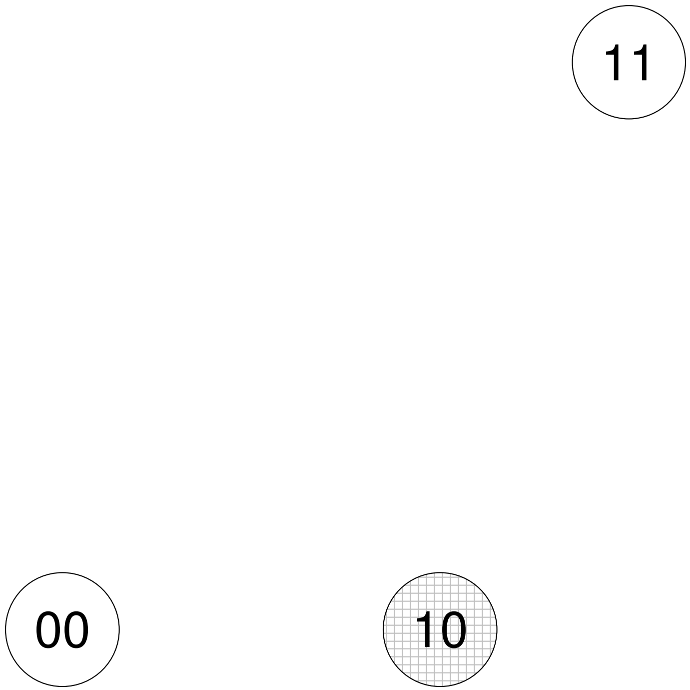
HD(0 , 1) = 1

Problem

Durch Bitfehler kommt man wieder zu einem gültigen Kodewort $\longrightarrow$ Bitfehler sind nicht erkennbar.

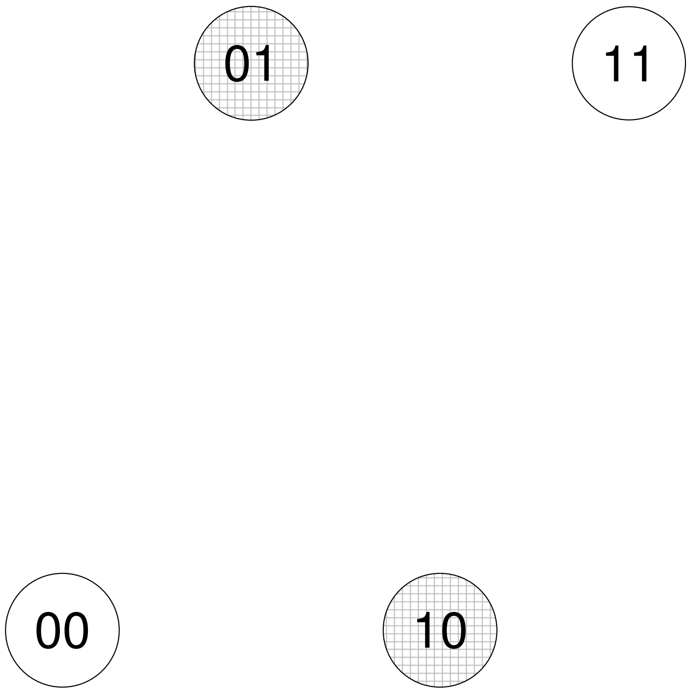
Aufgabe 1 – Hamming-Distanz

a) Welche (minimale) Hamming-Distanz müssen die gültigen Kodewörter aufweisen, damit Einzelfehler erkannt werden können?



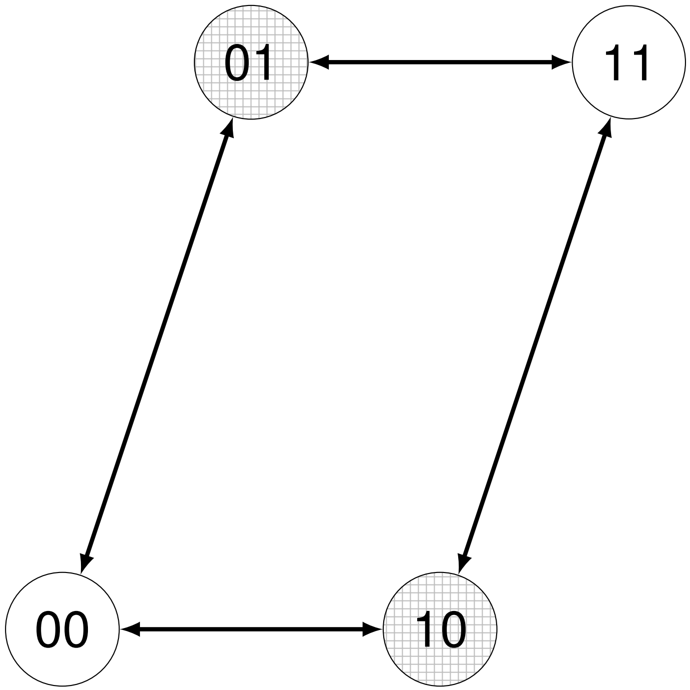
Aufgabe 1 – Hamming-Distanz

a) Welche (minimale) Hamming-Distanz müssen die gültigen Kodewörter aufweisen, damit Einzelfehler erkannt werden können?



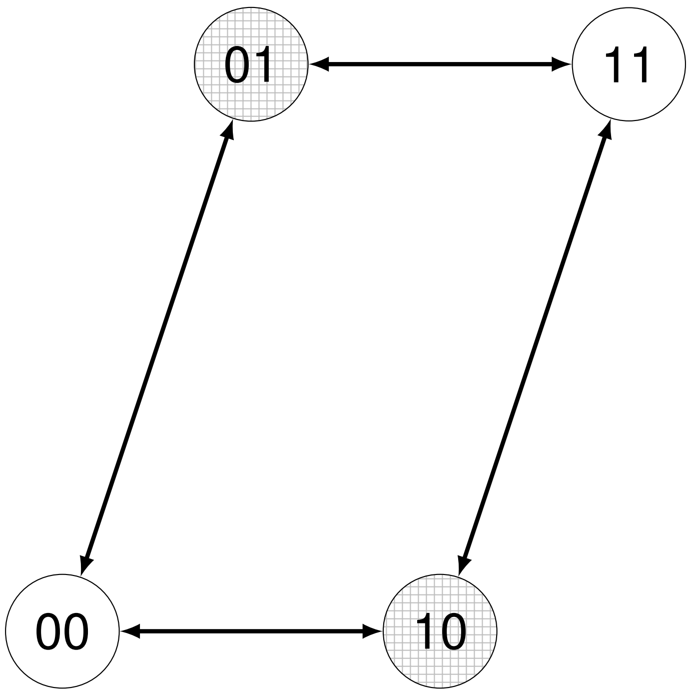
Aufgabe 1 – Hamming-Distanz

a) Welche (minimale) Hamming-Distanz müssen die gültigen Kodewörter aufweisen, damit Einzelfehler erkannt werden können?



Aufgabe 1 – Hamming-Distanz

a) Welche (minimale) Hamming-Distanz müssen die gültigen Kodewörter aufweisen, damit Einzelfehler erkannt werden können?



HD(`00`, `11`) = 2

Erfolg

Einzelfehler sind nun erkennbar, da sie von den gültigen Kodewörtern auf ungültige führen (in der Darstellung gekreuzelte).

Aufgabe 1 – Hamming-Distanz

a) Welche (minimale) Hamming-Distanz müssen die gültigen Kodewörter aufweisen, damit Einzelfehler erkannt werden können?
Sie müssen eine minimale Hamming-Distanz von 2 aufweisen.
Allgemein:

Erkennung von Fehlern

Um einen n-fach Fehler bei der Übertragung zu erkennen, ist eine minimale Hamming-Distanz von $n+1$ erforderlich.
In anderen Worten: Sei d die minimale Hamming-Distanz, so kann der Kode $d-1$-fach Fehler erkennen, jedoch **nicht** korrigieren.

Wie viele Zeichen können so mit drei Binärstellen maximal kodiert werden?
Wir betrachten das Startwort mit 3 Stellen.

Aufgabe 1 – Hamming-Distanz

a) Welche (minimale) Hamming-Distanz müssen die gültigen Kodewörter aufweisen, damit Einzelfehler erkannt werden können?
Sie müssen eine minimale Hamming-Distanz von 2 aufweisen.
Allgemein:

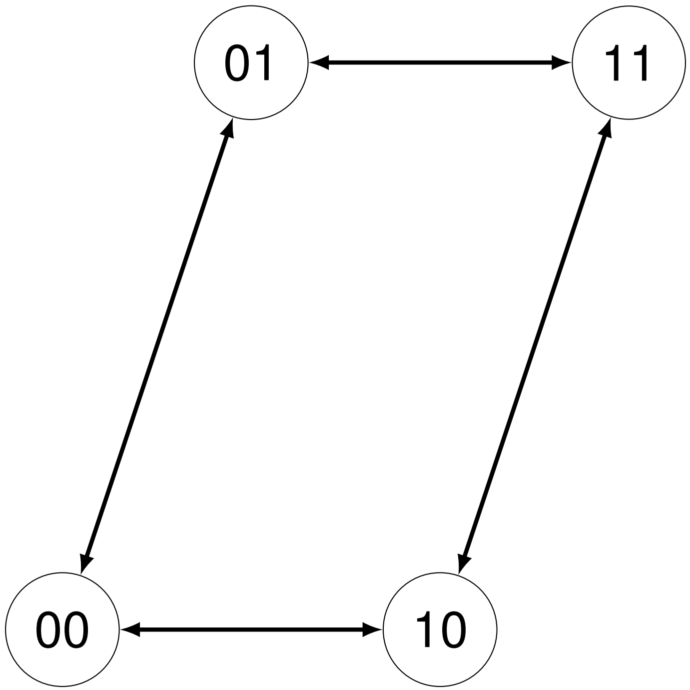
Erkennung von Fehlern

Um einen n-fach Fehler bei der Übertragung zu erkennen, ist eine minimale Hamming-Distanz von $n + 1$ erforderlich.
In anderen Worten: Sei d die minimale Hamming-Distanz, so kann der Kode $d - 1$-fach Fehler erkennen, jedoch **nicht** korrigieren.

Wie viele Zeichen können so mit drei Binärstellen maximal kodiert werden?
Wir betrachten das Startwort mit 3 Stellen. Die Möglichkeiten bei drei Binärstellen zwei Binärstellen zu verändern liegen bei $\binom{3}{2} = 3$.
Zusammen mit dem Ausgangswort macht das 4 Wörter.

Aufgabe 1 – Hamming-Distanz

b) Welche (minimale) Hamming-Distanz müssen die gültigen Kodewörter aufweisen, damit Einzelfehler korrigiert werden können?



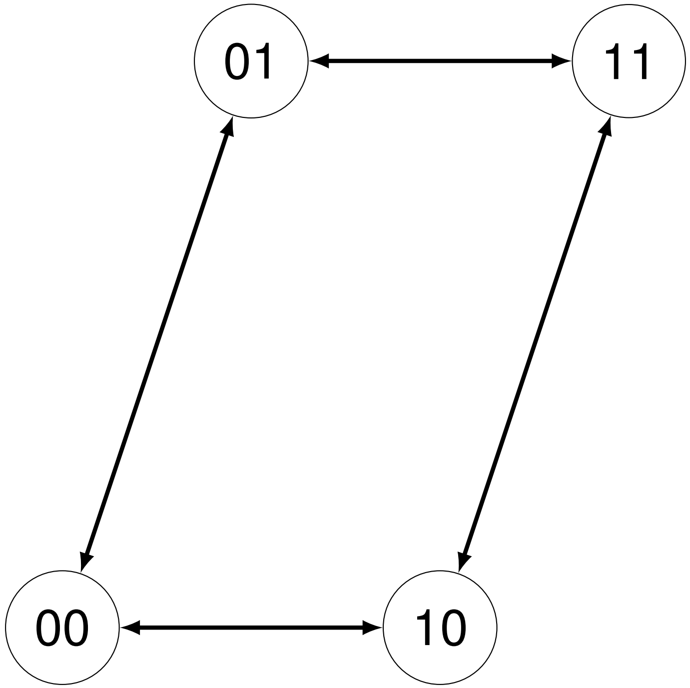
HD(`00`, `11`) = 2

Problem

Einzelfehler sind nun zwar erkennbar, jedoch noch nicht korrigierbar.

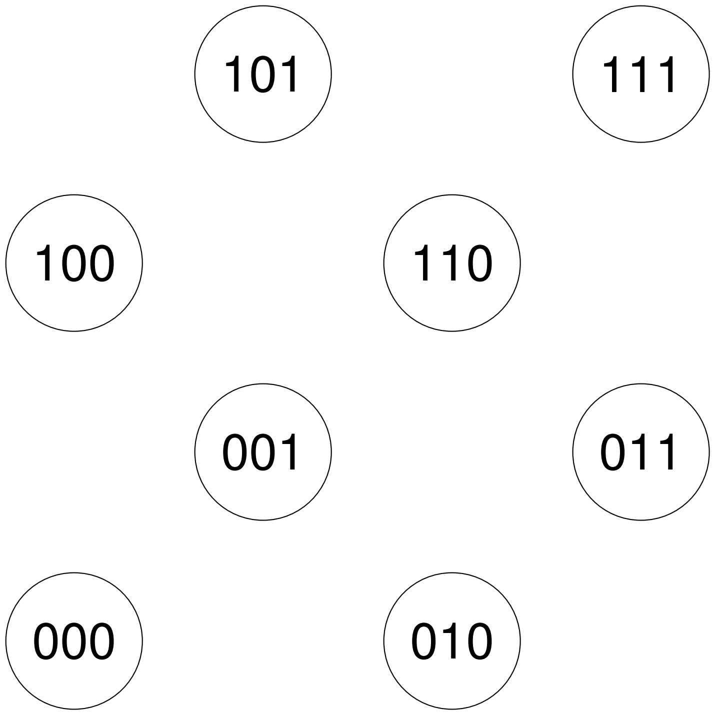
Aufgabe 1 – Hamming-Distanz

b) Welche (minimale) Hamming-Distanz müssen die gültigen Kodewörter aufweisen, damit Einzelfehler korrigiert werden können?



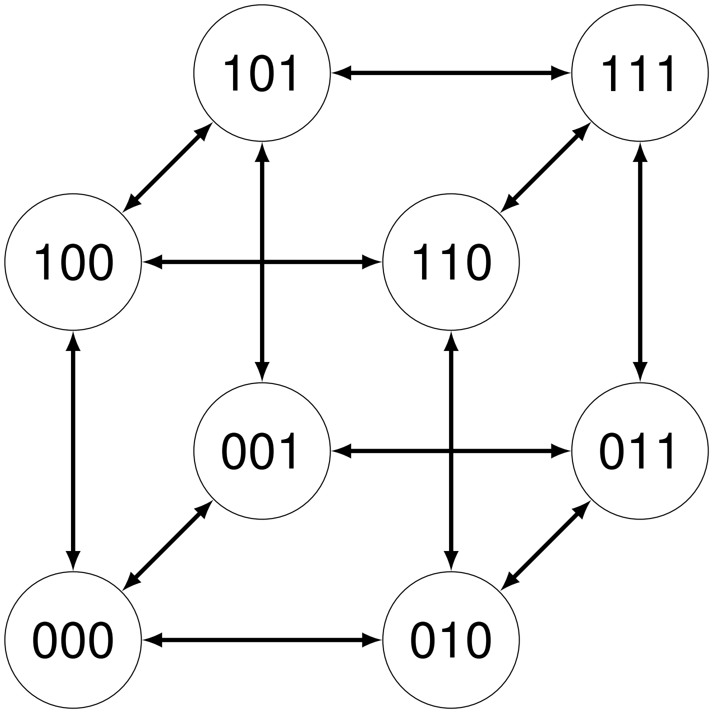
Aufgabe 1 – Hamming-Distanz

b) Welche (minimale) Hamming-Distanz müssen die gültigen Kodewörter aufweisen, damit Einzelfehler korrigiert werden können?



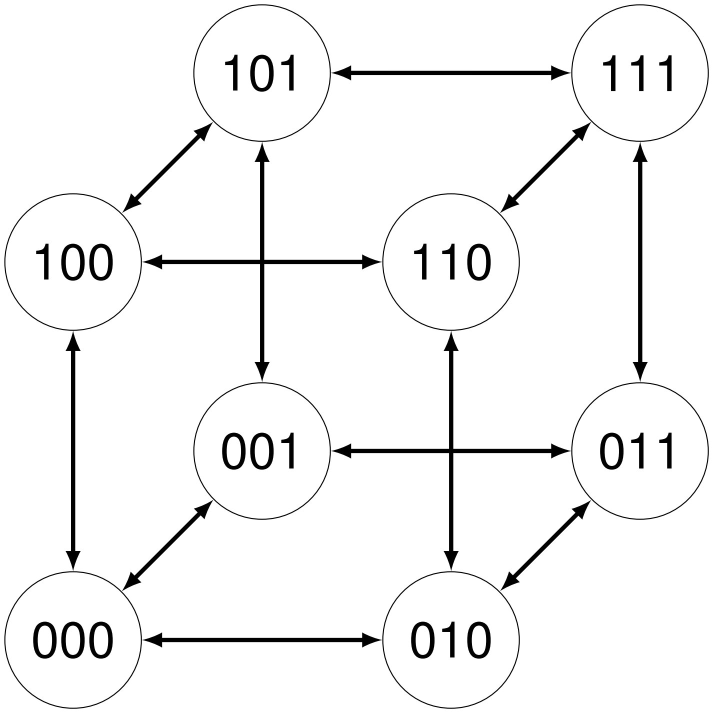
Aufgabe 1 – Hamming-Distanz

b) Welche (minimale) Hamming-Distanz müssen die gültigen Kodewörter aufweisen, damit Einzelfehler korrigiert werden können?



Aufgabe 1 – Hamming-Distanz

b) Welche (minimale) Hamming-Distanz müssen die gültigen Kodewörter aufweisen, damit Einzelfehler korrigiert werden können?



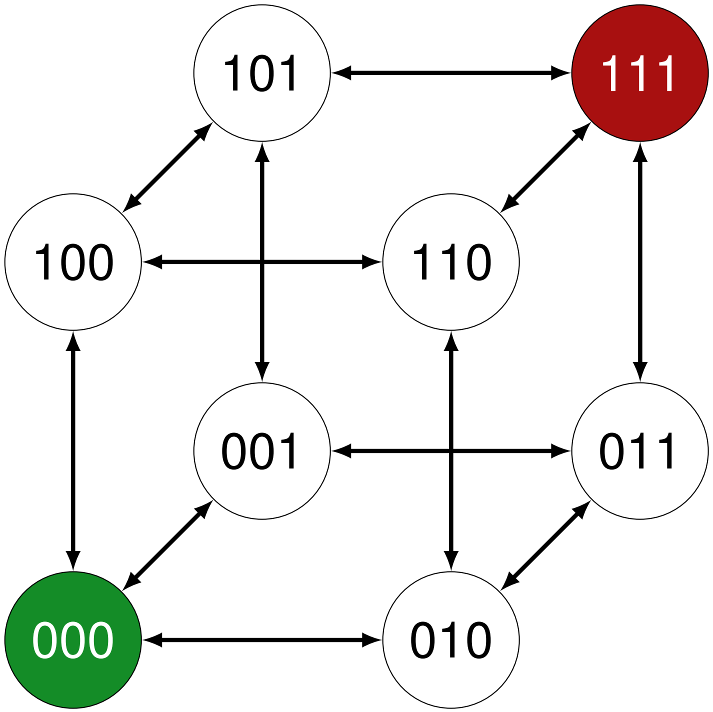
HD(`000`,`111`) = 3

Erfolg

Einzelfehler sind nun sogar korrigierbar, da man sie jeweils einem gültigen Kodewort zuordnen kann.

Aufgabe 1 – Hamming-Distanz

b) Welche (minimale) Hamming-Distanz müssen die gültigen Kodewörter aufweisen, damit Einzelfehler korrigiert werden können?



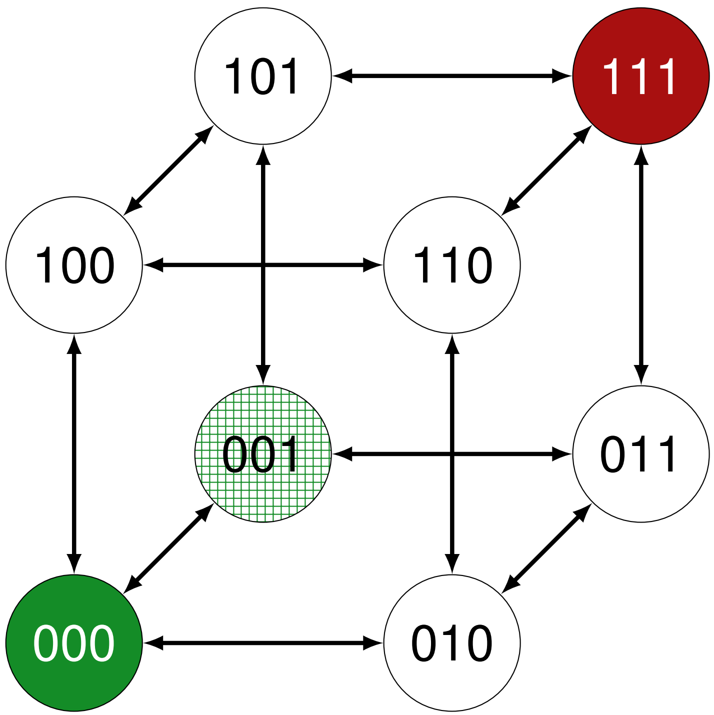
HD(000,111) = 3

Erfolg

Einzelfehler sind nun sogar korrigierbar, da man sie jeweils einem gültigen Kodewort zuordnen kann.

Aufgabe 1 – Hamming-Distanz

b) Welche (minimale) Hamming-Distanz müssen die gültigen Kodewörter aufweisen, damit Einzelfehler korrigiert werden können?



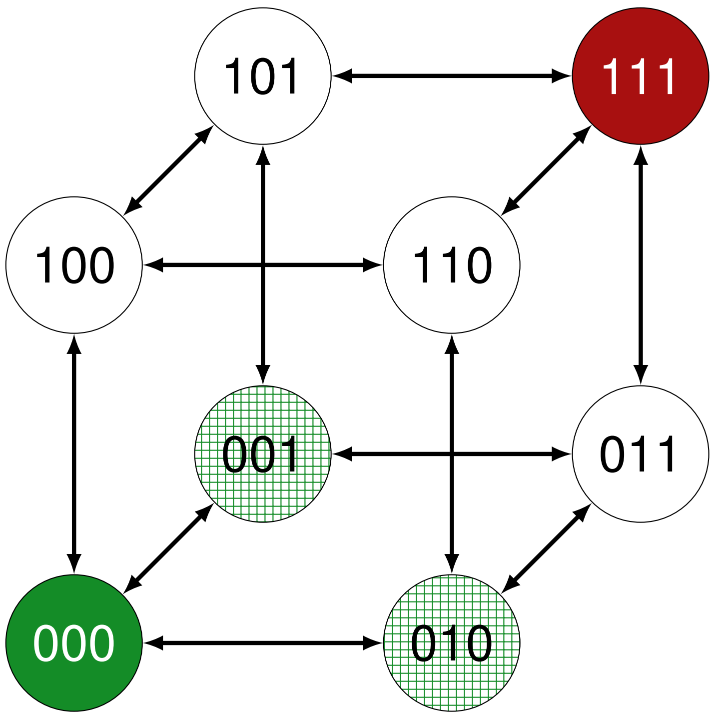
HD(000,111) = 3

Erfolg

Einzelfehler sind nun sogar korrigierbar, da man sie jeweils einem gültigen Kodewort zuordnen kann.

Aufgabe 1 – Hamming-Distanz

b) Welche (minimale) Hamming-Distanz müssen die gültigen Kodewörter aufweisen, damit Einzelfehler korrigiert werden können?



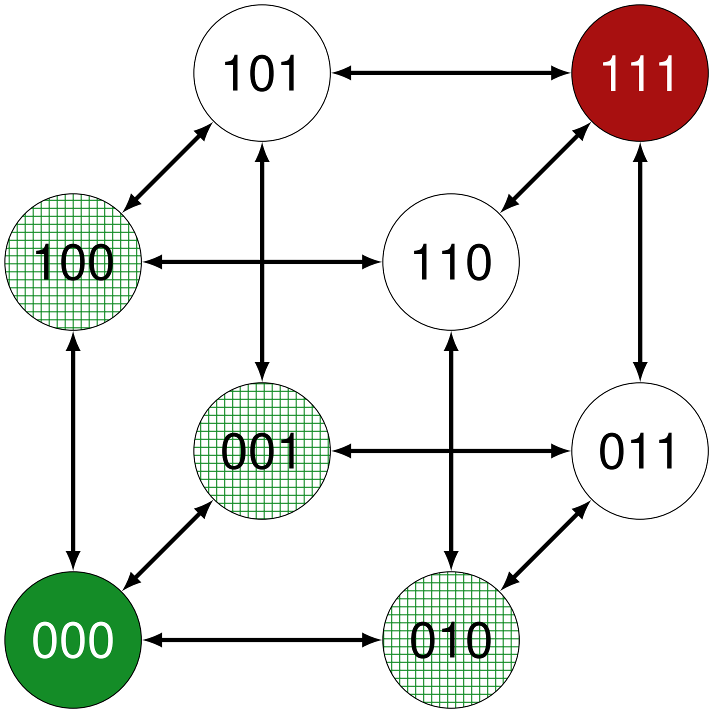
HD(000,111) = 3

Erfolg

Einzelfehler sind nun sogar korrigierbar, da man sie jeweils einem gültigen Kodewort zuordnen kann.

Aufgabe 1 – Hamming-Distanz

b) Welche (minimale) Hamming-Distanz müssen die gültigen Kodewörter aufweisen, damit Einzelfehler korrigiert werden können?



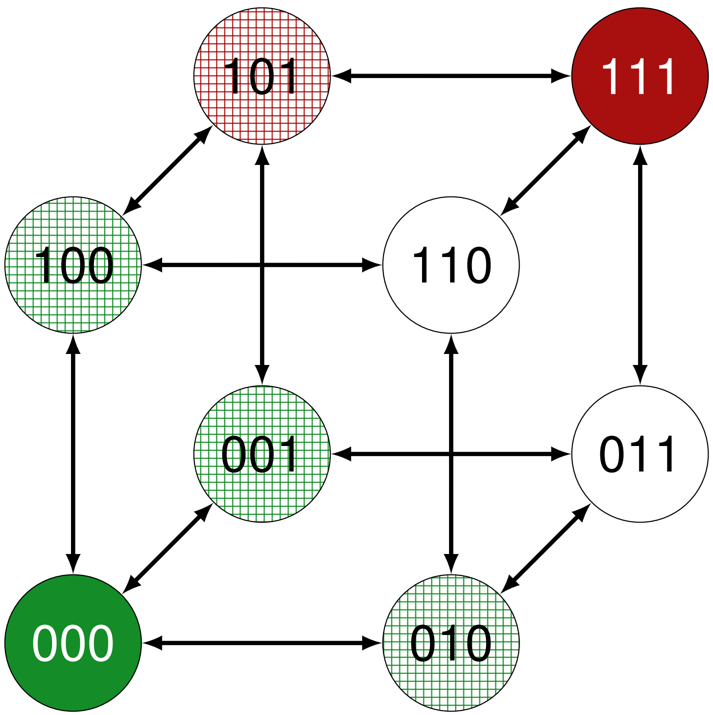
HD(000,111) = 3

Erfolg

Einzelfehler sind nun sogar korrigierbar, da man sie jeweils einem gültigen Kodewort zuordnen kann.

Aufgabe 1 – Hamming-Distanz

b) Welche (minimale) Hamming-Distanz müssen die gültigen Kodewörter aufweisen, damit Einzelfehler korrigiert werden können?



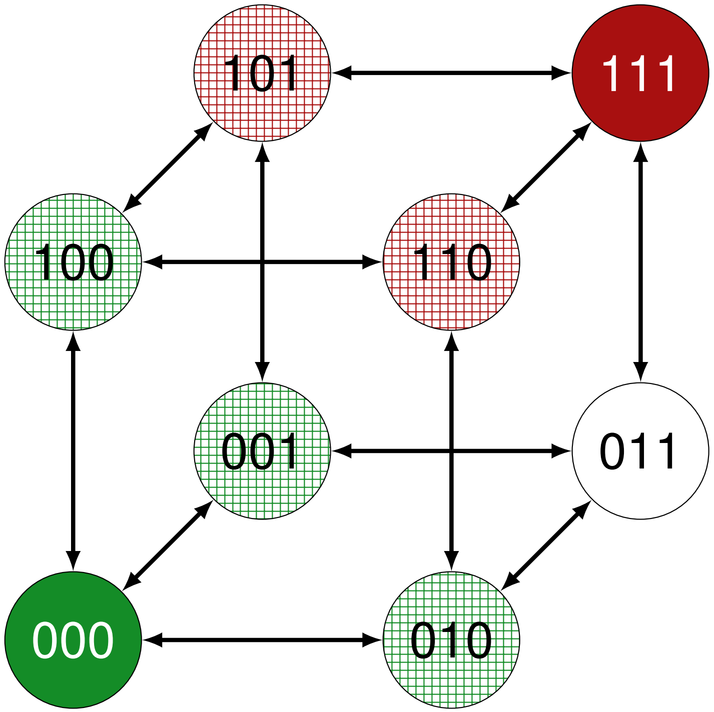
HD(000,111) = 3

Erfolg

Einzelfehler sind nun sogar korrigierbar, da man sie jeweils einem gültigen Kodewort zuordnen kann.

Aufgabe 1 – Hamming-Distanz

b) Welche (minimale) Hamming-Distanz müssen die gültigen Kodewörter aufweisen, damit Einzelfehler korrigiert werden können?



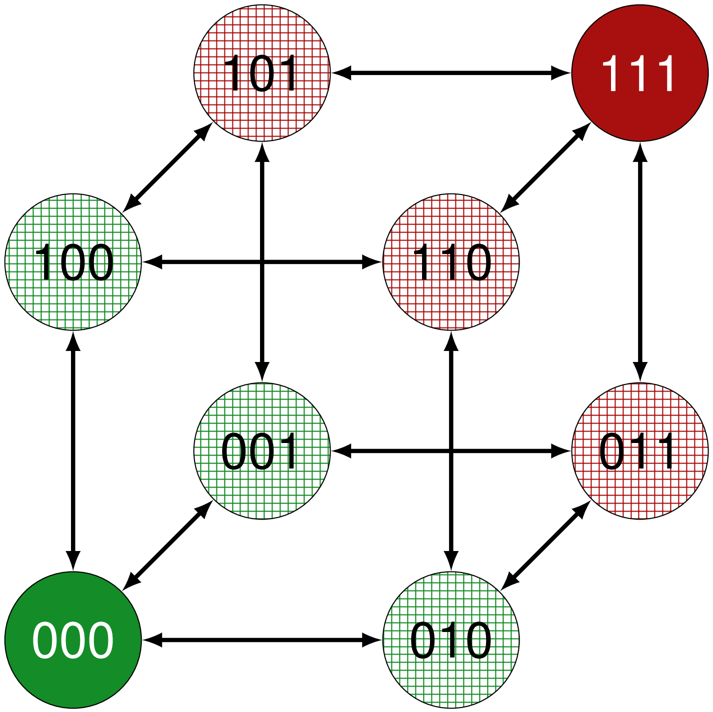
HD($\texttt{000}$,$\texttt{111}$) = 3

Erfolg

Einzelfehler sind nun sogar korrigierbar, da man sie jeweils einem gültigen Kodewort zuordnen kann.

Aufgabe 1 – Hamming-Distanz

b) Welche (minimale) Hamming-Distanz müssen die gültigen Kodewörter aufweisen, damit Einzelfehler korrigiert werden können?



HD(000,111) = 3

Erfolg

Einzelfehler sind nun sogar korrigierbar, da man sie jeweils einem gültigen Kodewort zuordnen kann.

Aufgabe 1 – Hamming-Distanz

b) Welche (minimale) Hamming-Distanz müssen die gültigen Kodewörter aufweisen, damit Einzelfehler korrigiert werden können?
Sie müssen eine minimale Hamming-Distanz von 3 aufweisen.
Allgemein:

Korrektur von Fehlern

Um einen n-fach Fehler bei der Übertragung zu korrigieren, ist eine minimale Hamming-Distanz von $2n + 1$ erforderlich.
In anderen Worten: Sei d die minimale Hamming-Distanz, so kann der Kode $\left\lfloor \frac{d-1}{2} \right\rfloor$-fach Fehler korrigieren.

Wie viele Zeichen können so mit drei Binärstellen maximal kodiert werden?
Wir betrachten das Startwort mit 3 Stellen.

Aufgabe 1 – Hamming-Distanz

b) Welche (minimale) Hamming-Distanz müssen die gültigen Kodewörter aufweisen, damit Einzelfehler korrigiert werden können?
Sie müssen eine minimale Hamming-Distanz von 3 aufweisen.
Allgemein:

Korrektur von Fehlern

Um einen n-fach Fehler bei der Übertragung zu korrigieren, ist eine minimale Hamming-Distanz von $2n + 1$ erforderlich.
In anderen Worten: Sei d die minimale Hamming-Distanz, so kann der Kode $\left\lfloor \frac{d-1}{2} \right\rfloor$-fach Fehler korrigieren.

Wie viele Zeichen können so mit drei Binärstellen maximal kodiert werden?
Wir betrachten das Startwort mit 3 Stellen. Die Möglichkeiten bei drei Binärstellen zwei Binärstellen zu verändern liegen bei $\binom{3}{3} = 1$.
Zusammen mit dem Ausgangswort macht das 2 Wörter.

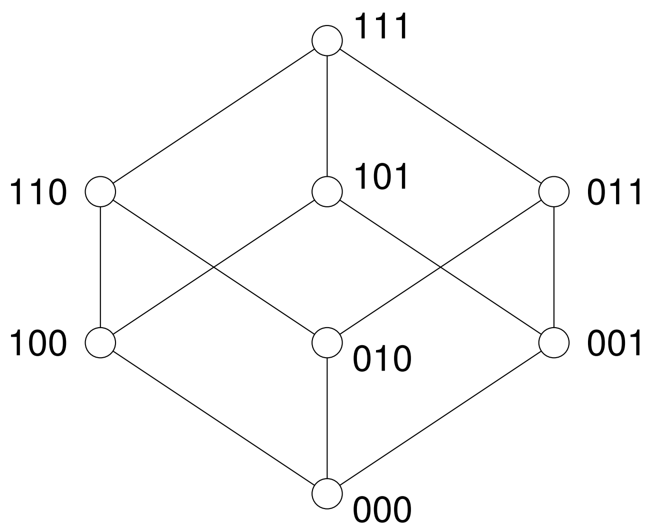
Aufgabe 1 – Hamming-Distanz

c) Die beiden Zeichen *A* und *B* sollen so kodiert werden, dass Einzelfehler korrigierbar sind. Wie viele Lösungen sind für die Kodierung der beiden Zeichen mit drei Binärstellen möglich? Geben Sie eine Lösung an.

d) Bei der Datenübertragung mit einer Kodierung nach c) wurde genau eine Binärstelle falsch übertragen. Die folgenden Daten wurden empfangen:

$$0\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 1\ 1\ 1\ 0$$

Korrigieren Sie den Fehler.



Aufgabe 2 – Fehlererkennung

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Aufgabe 2 – Fehlererkennung

Am Ende einer längeren Übertragungsstrecke wird die folgende Nachricht im ASCII-Code empfangen:

ASCII-Code		Zeichen
0	1 0 0 0 1 1 1	G
1	1 0 1 0 1 0 0	T
1	1 0 0 1 0 0 1	I
1	0 1 0 0 1 0 0	$
0	1 1 0 1 0 0 1	i
1	1 1 1 0 0 1 1	s
0	1 1 1 0 1 0 0	t
1	0 1 0 0 0 0 0	␣
1	1 1 1 0 1 1 0	v
0	1 1 0 1 1 1 1	o
0	1 1 0 1 1 0 0	l
0	1 1 0 1 1 0 0	l

Es ist bekannt, dass der Sender die sieben Bits des ASCII-Codes um ein sogenanntes Paritätsbit (ganz links) ergänzt hat.

a) Welches Zeichen wurde offensichtlich falsch übertragen?

b) Das letzte Wort lautete vor der Übertragung „toll“ und nicht „voll“. Warum ist der von der Übertragungsstrecke verursachte Fehler nicht erkennbar?

Aufgabe 2 – Fehlererkennung: Begriffsklärung (I)

Fehlererkennung mit Paritätsbit

Bei der Fehlererkennung mit Paritätsbits werden zu übertragende Codewörter mit einem zusätlichen Bit gesichert.

Es gibt einen Unterschied zwischen ...

- ... **gerader Parität**: Die Einsen werden auf gerade Anzahl ergänzt. (Ver-XOR-e jede Stelle miteinander).

$$p_B = x_0 \oplus x_1 \oplus \ldots \oplus x_n = \bigoplus_{i=1}^{n} x_i$$

- ... **ungerader Parität**: Die Einsen werden auf ungerade Anzahl ergänzt. (Ver-XNOR-e jede Stelle miteinander).

$$p_B = \overline{x_0 \oplus x_1 \oplus \ldots \oplus x_n} = \overline{\bigoplus_{i=1}^{n}} x_i$$

Aufgabe 2 – Fehlererkennung: Begriffsklärung (II)

ASCII-Kodierung

ASCII $\equiv$ **A**merican **S**tandard **C**ode for **I**nformation **I**nterchange

- Der ASCII-Code kodiert mit 7 Bit insgesamt **128 Zeichen** und reicht damit weitestgehend für die englische Sprache aus (Vorsicht: Keine Sonderzeichen $\rightarrow$ Unicode oder Extended ASCII)
- Die Bitgruppen werden zusammengefasst in die **MSB** (**M**ost **S**ignificant **B**its) und **LSB** (**L**east **S**ignificant **B**its).

$$\underbrace{b_7 b_6 b_5}_{\textbf{MSB}} \underbrace{b_4 b_3 b_2 b_1}_{\textbf{LSB}}$$

Aufgabe 2 – Fehlererkennung: Begriffsklärung (II)

MSBs / LSBs	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
0 0 0 0	NUL	DLE	SP	0	@	P	‘	p
0 0 0 1	SOH	DC1	!	1	A	Q	a	q
0 0 1 0	STX	DC2	"	2	B	R	b	r
0 0 1 1	ETX	DC3	#	3	C	S	c	s
0 1 0 0	EOT	DC4	$	4	D	T	d	t
0 1 0 1	ENQ	NAK	%	5	E	U	e	u
0 1 1 0	ACK	SYN	&	6	F	V	f	v
0 1 1 1	BEL	ETB	’	7	G	W	g	w
1 0 0 0	BS	CAN	(	8	H	X	h	x
1 0 0 1	HT	EM	)	9	I	Y	i	y
1 0 1 0	LF	SUB	*	:	J	Z	j	z
1 0 1 1	VT	ESC	+	;	K	[	k	{
1 1 0 0	FF	FS	,	<	L	\	l	|
1 1 0 1	CR	GS	—	=	M	]	m	}
1 1 1 0	SO	RS	.	>	N	^	n	~
1 1 1 1	SI	US	/	?	O	_	o	DEL

Tabelle 1: ASCII-Kodierung

Aufgabe 2 – Fehlererkennung

Am Ende einer längeren Übertragungsstrecke wird die folgende Nachricht im ASCII-Code empfangen:

ASCII-Code	Zeichen
0 1 0 0 0 1 1 1	G
1 1 0 1 0 1 0 0	T
1 1 0 0 1 0 0 1	I
1 0 1 0 0 1 0 0	$
0 1 1 0 1 0 0 1	i
1 1 1 1 0 0 1 1	s
0 1 1 1 0 1 0 0	t
1 0 1 0 0 0 0 0	␣
1 1 1 1 0 1 1 0	v
0 1 1 0 1 1 1 1	o
0 1 1 0 1 1 0 0	l
0 1 1 0 1 1 0 0	l

Es ist bekannt, dass der Sender die sieben Bits des ASCII-Codes um ein sogenanntes Paritätsbit (ganz links) ergänzt hat.

a) Welches Zeichen wurde offensichtlich falsch übertragen?

b) Das letzte Wort lautete vor der Übertragung „toll“ und nicht „voll“. Warum ist der von der Übertragungsstrecke verursachte Fehler nicht erkennbar?

Aufgabe 3 – Blocksicherung

Aufgabe 3 – Blocksicherung

Es sollen wichtige Daten im ASCII-Code mit einer Blocksicherung geschützt werden, die gerade Parität für die Prüfbits verwendet. Die folgende Tabelle zeigt die empfangenen Daten, welche offensichtlich nicht alle korrekt übermittelt wurden:

	Binärcode		Prüfbit	ASCII
	1 0 1	0 0 1 0	1	R
Block	1 0 0	1 0 0 1	1	I
1	1 0 1	0 0 1 1	1	S
	0 1 0	1 0 0 0	0	H
Prüfbits	1 1 1	0 0 0 0	1	

	Binärcode		Prüfbit	ASCII
	1 0 1	0 0 0 0	1	P
Block	1 0 0	1 0 0 1	1	I
2	1 0 0	0 1 1 1	0	G
	0 1 0	0 0 0 1	0	!
Prüfbits	1 1 1	1 0 1 1	0	

a) Welche Fehler (Anzahl, Einfach-/Mehrfachfehler) sind korrigierbar?

b) Die aufgetretenen Fehler seien korrigierbar. Korrigieren Sie die entsprechenden Binärstellen in der Tabelle. Bestimmen Sie für die korrigierten Codewörter das zugehörige ASCII-Zeichen.

Aufgabe 3 – Blocksicherung: Begriffsklärung (I)

Fehlerkorrektur durch Blocksicherung

Eine Erweiterung der normalen Paritätssicherung stellt die Blocksicherung dar.
Die Nachricht wird in Blöcke von je *n* Codewörtern mit Paritätsbit eingeteilt. **Zusätzlich** wird am Ende eines jeden Blocks ein weiteres Codewort eingefügt, das alle Paritätsbits der Spalten enthält.
Betrachten wir ein Beispiel mit gerader Parität:

$$\begin{array}{cccc|c} 0 & 1 & 0 & 1 & \\ 0 & 1 & 0 & 0 & \\ 0 & 0 & 0 & 1 & \\ 1 & 0 & 0 & 0 & \\ \hline & & & & \end{array}$$

Aufgabe 3 – Blocksicherung: Begriffsklärung (I)

Fehlerkorrektur durch Blocksicherung

Eine Erweiterung der normalen Paritätssicherung stellt die Blocksicherung dar.
Die Nachricht wird in Blöcke von je *n* Codewörtern mit Paritätsbit eingeteilt. **Zusätzlich** wird am Ende eines jeden Blocks ein weiteres Codewort eingefügt, das alle Paritätsbits der Spalten enthält.
Betrachten wir ein Beispiel mit gerader Parität:

$$\begin{array}{cccc|c} 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & \\ 0 & 0 & 0 & 1 & \\ 1 & 0 & 0 & 0 & \\ \hline & & & & \end{array}$$

Aufgabe 3 – Blocksicherung: Begriffsklärung (I)

Fehlerkorrektur durch Blocksicherung

Eine Erweiterung der normalen Paritätssicherung stellt die Blocksicherung dar.
Die Nachricht wird in Blöcke von je *n* Codewörtern mit Paritätsbit eingeteilt. **Zusätzlich** wird am Ende eines jeden Blocks ein weiteres Codewort eingefügt, das alle Paritätsbits der Spalten enthält.
Betrachten wir ein Beispiel mit gerader Parität:

$$\begin{array}{cccc|c} 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & \\ 1 & 0 & 0 & 0 & \\ \hline & & & & \end{array}$$

Aufgabe 3 – Blocksicherung: Begriffsklärung (I)

Fehlerkorrektur durch Blocksicherung

Eine Erweiterung der normalen Paritätssicherung stellt die Blocksicherung dar.

Die Nachricht wird in Blöcke von je *n* Codewörtern mit Paritätsbit eingeteilt. **Zusätzlich** wird am Ende eines jeden Blocks ein weiteres Codewort eingefügt, das alle Paritätsbits der Spalten enthält.

Betrachten wir ein Beispiel mit gerader Parität:

$$
\begin{array}{cccc|c}
0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 1 \\
1 & 0 & 0 & 0 & \\
\hline
 & & & & \\
\end{array}
$$

Aufgabe 3 – Blocksicherung: Begriffsklärung (I)

Fehlerkorrektur durch Blocksicherung

Eine Erweiterung der normalen Paritätssicherung stellt die Blocksicherung dar.
Die Nachricht wird in Blöcke von je n Codewörtern mit Paritätsbit eingeteilt. **Zusätzlich** wird am Ende eines jeden Blocks ein weiteres Codewort eingefügt, das alle Paritätsbits der Spalten enthält.
Betrachten wir ein Beispiel mit gerader Parität:

$$
\begin{array}{cccc|c}
0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 1 \\
1 & 0 & 0 & 0 & 1 \\
\hline
 & & & &
\end{array}
$$

Aufgabe 3 – Blocksicherung: Begriffsklärung (I)

Fehlerkorrektur durch Blocksicherung

Eine Erweiterung der normalen Paritätssicherung stellt die Blocksicherung dar.
Die Nachricht wird in Blöcke von je *n* Codewörtern mit Paritätsbit eingeteilt. **Zusätzlich** wird am Ende eines jeden Blocks ein weiteres Codewort eingefügt, das alle Paritätsbits der Spalten enthält.
Betrachten wir ein Beispiel mit gerader Parität:

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

Aufgabe 3 – Blocksicherung: Begriffsklärung (I)

Fehlerkorrektur durch Blocksicherung

Eine Erweiterung der normalen Paritätssicherung stellt die Blocksicherung dar.
Die Nachricht wird in Blöcke von je *n* Codewörtern mit Paritätsbit eingeteilt. **Zusätzlich** wird am Ende eines jeden Blocks ein weiteres Codewort eingefügt, das alle Paritätsbits der Spalten enthält.
Betrachten wir ein Beispiel mit gerader Parität:

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

Aufgabe 3 – Blocksicherung: Begriffsklärung (I)

Fehlerkorrektur durch Blocksicherung

Eine Erweiterung der normalen Paritätssicherung stellt die Blocksicherung dar.
Die Nachricht wird in Blöcke von je *n* Codewörtern mit Paritätsbit eingeteilt. **Zusätzlich** wird am Ende eines jeden Blocks ein weiteres Codewort eingefügt, das alle Paritätsbits der Spalten enthält.
Betrachten wir ein Beispiel mit gerader Parität:

$$\begin{array}{cccc|c} 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ \hline 1 & 0 & 0 & & \end{array}$$

Aufgabe 3 – Blocksicherung: Begriffsklärung (I)

Fehlerkorrektur durch Blocksicherung

Eine Erweiterung der normalen Paritätssicherung stellt die Blocksicherung dar.
Die Nachricht wird in Blöcke von je n Codewörtern mit Paritätsbit eingeteilt. **Zusätzlich** wird am Ende eines jeden Blocks ein weiteres Codewort eingefügt, das alle Paritätsbits der Spalten enthält.
Betrachten wir ein Beispiel mit gerader Parität:

$$\begin{array}{cccc|c} 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ \hline 1 & 0 & 0 & 0 & \end{array}$$

Aufgabe 3 – Blocksicherung: Begriffsklärung (I)

Fehlerkorrektur durch Blocksicherung

Eine Erweiterung der normalen Paritätssicherung stellt die Blocksicherung dar.
Die Nachricht wird in Blöcke von je *n* Codewörtern mit Paritätsbit eingeteilt. **Zusätzlich** wird am Ende eines jeden Blocks ein weiteres Codewort eingefügt, das alle Paritätsbits der Spalten enthält.
Betrachten wir ein Beispiel mit gerader Parität:

$$\begin{array}{cccc|c} 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ \hline 1 & 0 & 0 & 0 & 1 \end{array}$$

Aufgabe 3 – Blocksicherung: Begriffsklärung (II)

Fehlerkorrektur durch Blocksicherung

Eine Erweiterung der normalen Paritätssicherung stellt die Blocksicherung dar.
Die Nachricht wird in Blöcke von je *n* Codewörtern mit Paritätsbit eingeteilt. **Zusätzlich** wird am Ende eines jeden Blocks ein weiteres Codewort eingefügt, das alle Paritätsbits der Spalten enthält.

Aufgabe 3 – Blocksicherung: Begriffsklärung (II)

Fehlerkorrektur durch Blocksicherung

Eine Erweiterung der normalen Paritätssicherung stellt die Blocksicherung dar. Die Nachricht wird in Blöcke von je *n* Codewörtern mit Paritätsbit eingeteilt. **Zusätzlich** wird am Ende eines jeden Blocks ein weiteres Codewort eingefügt, das alle Paritätsbits der Spalten enthält.

Wie erkenne ich einen Fehler?

Durch die Schnittpunkte von fehlerhaften Zeilen- und Spaltenparitäten.

$$\begin{array}{cccc|c} 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ \hline 1 & 0 & 0 & 0 & 1 \end{array}$$

Aufgabe 3 – Blocksicherung: Begriffsklärung (II)

Fehlerkorrektur durch Blocksicherung

Eine Erweiterung der normalen Paritätssicherung stellt die Blocksicherung dar. Die Nachricht wird in Blöcke von je *n* Codewörtern mit Paritätsbit eingeteilt. **Zusätzlich** wird am Ende eines jeden Blocks ein weiteres Codewort eingefügt, das alle Paritätsbits der Spalten enthält.

Wie erkenne ich einen Fehler?

Durch die Schnittpunkte von fehlerhaften Zeilen- und Spaltenparitäten.

$$
\begin{array}{cccc|c}
0 & 1 & 0 & 1 & \mathbf{0} \\
0 & 1 & 0 & 0 & 1 \\
0 & 0 & 1 & 1 & 1 \\
1 & 0 & 0 & 0 & 1 \\
\hline
\mathbf{1} & 0 & 0 & 0 & 1
\end{array}
$$

Aufgabe 3 – Blocksicherung: Begriffsklärung (II)

Fehlerkorrektur durch Blocksicherung

Eine Erweiterung der normalen Paritätssicherung stellt die Blocksicherung dar. Die Nachricht wird in Blöcke von je *n* Codewörtern mit Paritätsbit eingeteilt. **Zusätzlich** wird am Ende eines jeden Blocks ein weiteres Codewort eingefügt, das alle Paritätsbits der Spalten enthält.

Wie erkenne ich einen Fehler?

Durch die Schnittpunkte von fehlerhaften Zeilen- und Spaltenparitäten.

0	1	0	1	0
0	1	0	0	**1**
0	0	1	1	1
1	0	0	0	1
1	**0**	0	0	1

Aufgabe 3 – Blocksicherung: Begriffsklärung (II)

Fehlerkorrektur durch Blocksicherung

Eine Erweiterung der normalen Paritätssicherung stellt die Blocksicherung dar. Die Nachricht wird in Blöcke von je *n* Codewörtern mit Paritätsbit eingeteilt. **Zusätzlich** wird am Ende eines jeden Blocks ein weiteres Codewort eingefügt, das alle Paritätsbits der Spalten enthält.

Wie erkenne ich einen Fehler?

Durch die Schnittpunkte von fehlerhaften Zeilen- und Spaltenparitäten.

0	1	0	1	0
0	1	0	0	1
0	0	1	1	**1**
1	0	0	0	1
1	0	**0**	0	1

Aufgabe 3 – Blocksicherung: Begriffsklärung (II)

Fehlerkorrektur durch Blocksicherung

Eine Erweiterung der normalen Paritätssicherung stellt die Blocksicherung dar. Die Nachricht wird in Blöcke von je n Codewörtern mit Paritätsbit eingeteilt. **Zusätzlich** wird am Ende eines jeden Blocks ein weiteres Codewort eingefügt, das alle Paritätsbits der Spalten enthält.

Wie erkenne ich einen Fehler?

Durch die Schnittpunkte von fehlerhaften Zeilen- und Spaltenparitäten.

$$
\begin{array}{cccc|c}
0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 1 \\
0 & 0 & 1 & 1 & \mathbf{1} \\
1 & 0 & 0 & 0 & 1 \\
\hline
1 & 0 & \mathbf{0} & 0 & 1
\end{array}
$$

Aufgabe 3 – Blocksicherung: Begriffsklärung (II)

Fehlerkorrektur durch Blocksicherung

Eine Erweiterung der normalen Paritätssicherung stellt die Blocksicherung dar. Die Nachricht wird in Blöcke von je *n* Codewörtern mit Paritätsbit eingeteilt. **Zusätzlich** wird am Ende eines jeden Blocks ein weiteres Codewort eingefügt, das alle Paritätsbits der Spalten enthält.

Wie erkenne ich einen Fehler?

Durch die Schnittpunkte von fehlerhaften Zeilen- und Spaltenparitäten.

$$
\begin{array}{cccc|c}
0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 1 \\
0 & 0 & 1 & 1 & \mathbf{1} \\
1 & 0 & 0 & 0 & 1 \\
\hline
1 & 0 & \mathbf{0} & 0 & 1
\end{array}
$$

Konklusion

Es sind nur Einfachfehler pro Block korrigierbar!

Aufgabe 3 – Blocksicherung

Es sollen wichtige Daten im ASCII-Code mit einer Blocksicherung geschützt werden, die gerade Parität für die Prüfbits verwendet. Die folgende Tabelle zeigt die empfangenen Daten, welche offensichtlich nicht alle korrekt übermittelt wurden:

	Binärcode		Prüfbit	ASCII
	1 0 1	0 0 1 0	1	R
Block	1 0 0	1 0 0 1	1	I
1	1 0 1	0 0 1 1	1	S
	0 1 0	1 0 0 0	0	H
Prüfbits	1 1 1	0 0 0 0	1	

	Binärcode		Prüfbit	ASCII
	1 0 1	0 0 0 0	1	P
Block	1 0 0	1 0 0 1	1	I
2	1 0 0	0 1 1 1	0	G
	0 1 0	0 0 0 1	0	!
Prüfbits	1 1 1	1 0 1 1	0	

b) Die aufgetretenen Fehler seien korrigierbar. Korrigieren Sie die entsprechenden Binärstellen in der Tabelle. Bestimmen Sie für die korrigierten Codewörter das zugehörige ASCII-Zeichen.

Aufgabe 4 – Fehlerkorrektur

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Aufgabe 4 – Fehlerkorrektur

Sei ein nicht fehlertolerantes Kommunikationssystem gegeben, das in der Lage ist, einstellige Hexadezimalzahlen zu übertragen. Es soll nun dahingehend erweitert werden, dass es mittels eines Hamming-Codes Zweifachfehler erkennen oder Einfachfehler korrigieren kann.

a) Welche Hamming-Distanz wird benötigt, um die geforderte Fehlertoleranz zu erreichen?

Aufgabe 4 – Fehlerkorrektur

Sei ein nicht fehlertolerantes Kommunikationssystem gegeben, das in der Lage ist, einstellige Hexadezimalzahlen zu übertragen. Es soll nun dahingehend erweitert werden, dass es mittels eines Hamming-Codes Zweifachfehler erkennen oder Einfachfehler korrigieren kann.

b) Wie viele Bits werden benötigt, um die jeweiligen Informationen und die Paritätsbits nach Hamming zu codieren? Wie lang wird das gesamte zu übertragende Codewort?

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (I)

Hamming-Code

Der Hamming-Code ist ein Beispiel für einen Code mit $HD_{min} = 3$. Dabei werden Prüfsummen nur auf Teilwörtern generiert.

Ein Algorithmus für die Erzeugung eines Hamming-Codes:

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.
2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits
3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.
   Äquivalent dazu ($x_k(i)$ bezeichnet das $i$te Bit des Wortes x_k, d ist die Anzahl an Datenbits):

$$y_i = \bigoplus_{\substack{k=1 \\ x_k(i)=1}}^{d} x_k$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.

Zeile
4
3
2
1

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: $(12, 8)$ Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.

Zeile	1
4	0
3	0
2	0
1	1

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: $(12, 8)$ Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.

Zeile	2	1
4	0	0
3	0	0
2	1	0
1	0	1

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.

Zeile	3	2	1
4	0	0	0
3	0	0	0
2	1	1	0
1	1	0	1

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: $(12, 8)$ Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.

Zeile	4	3	2	1
4	0	0	0	0
3	1	0	0	0
2	0	1	1	0
1	0	1	0	1

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: $(12, 8)$ Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis *n* binär in absteigender Reihenfolge.

Zeile	5	4	3	2	1
4	0	0	0	0	0
3	1	1	0	0	0
2	0	0	1	1	0
1	1	0	1	0	1

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: $(12, 8)$ Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.

Zeile	6	5	4	3	2	1
4	0	0	0	0	0	0
3	1	1	1	0	0	0
2	1	0	0	1	1	0
1	0	1	0	1	0	1

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: $(12, 8)$ Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.

Zeile	7	6	5	4	3	2	1
4	0	0	0	0	0	0	0
3	1	1	1	1	0	0	0
2	1	1	0	0	1	1	0
1	1	0	1	0	1	0	1

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: $(12, 8)$ Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.

Zeile	8	7	6	5	4	3	2	1
4	1	0	0	0	0	0	0	0
3	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: $(12, 8)$ Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.

Zeile	9	8	7	6	5	4	3	2	1
4	1	1	0	0	0	0	0	0	0
3	0	0	1	1	1	1	0	0	0
2	0	0	1	1	0	0	1	1	0
1	1	0	1	0	1	0	1	0	1

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis *n* binär in absteigender Reihenfolge.

Zeile	10	9	8	7	6	5	4	3	2	1
4	1	1	1	0	0	0	0	0	0	0
3	0	0	0	1	1	1	1	0	0	0
2	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.

Zeile	11	10	9	8	7	6	5	4	3	2	1
4	1	1	1	1	0	0	0	0	0	0	0
3	0	0	0	0	1	1	1	1	0	0	0
2	1	1	0	0	1	1	0	0	1	1	0
1	1	0	1	0	1	0	1	0	1	0	1

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.

Zeile	12	11	10	9	8	7	6	5	4	3	2	1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: $(12, 8)$ Hamming-Code

2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits.

Zeile	12	11	10	9	8	7	6	5	4	3	2	1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller x = 1-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: $(12, 8)$ Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

$$y_1 =$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: $(12, 8)$ Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

$$y_1 = x_1$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: $(12, 8)$ Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

$$y_1 = x_1 \oplus x_2$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller x = 1-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

$$y_1 = x_1 \oplus x_2 \oplus x_4$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: $(12, 8)$ Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

$$y_1 = x_1 \oplus x_2 \oplus x_4 \oplus x_5$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

$$y_1 = x_1 \oplus x_2 \oplus x_4 \oplus x_5 \oplus x_7$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller x = 1-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

$$y_1 = x_1 \oplus x_2 \oplus x_4 \oplus x_5 \oplus x_7$$

$$y_2 =$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller x = 1-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

$$y_1 = x_1 \oplus x_2 \oplus x_4 \oplus x_5 \oplus x_7$$

$$y_2 = x_1$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller x = 1-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

$$y_1 = x_1 \oplus x_2 \oplus x_4 \oplus x_5 \oplus x_7$$
$$y_2 = x_1 \oplus x_3$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

$$y_1 = x_1 \oplus x_2 \oplus x_4 \oplus x_5 \oplus x_7$$

$$y_2 = x_1 \oplus x_3 \oplus x_4$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller x = 1-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

$$y_1 = x_1 \oplus x_2 \oplus x_4 \oplus x_5 \oplus x_7$$

$$y_2 = x_1 \oplus x_3 \oplus x_4 \oplus x_6$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller x = 1-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

$$y_1 = x_1 \oplus x_2 \oplus x_4 \oplus x_5 \oplus x_7$$
$$y_2 = x_1 \oplus x_3 \oplus x_4 \oplus x_6 \oplus x_7$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: $(12, 8)$ Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

$$y_1 = x_1 \oplus x_2 \oplus x_4 \oplus x_5 \oplus x_7$$
$$y_2 = x_1 \oplus x_3 \oplus x_4 \oplus x_6 \oplus x_7$$
$$y_3 =$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

$$y_1 = x_1 \oplus x_2 \oplus x_4 \oplus x_5 \oplus x_7$$
$$y_2 = x_1 \oplus x_3 \oplus x_4 \oplus x_6 \oplus x_7$$
$$y_3 = x_2$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller x = 1-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

$$y_1 = x_1 \oplus x_2 \oplus x_4 \oplus x_5 \oplus x_7$$
$$y_2 = x_1 \oplus x_3 \oplus x_4 \oplus x_6 \oplus x_7$$
$$y_3 = x_2 \oplus x_3$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: $(12, 8)$ Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

$$y_1 = x_1 \oplus x_2 \oplus x_4 \oplus x_5 \oplus x_7$$
$$y_2 = x_1 \oplus x_3 \oplus x_4 \oplus x_6 \oplus x_7$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller x = 1-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

$$y_1 = x_1 \oplus x_2 \oplus x_4 \oplus x_5 \oplus x_7$$
$$y_2 = x_1 \oplus x_3 \oplus x_4 \oplus x_6 \oplus x_7$$
$$y_3 = x_2 \oplus x_3 \oplus x_4 \oplus x_8$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller x = 1-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

$$y_1 = x_1 \oplus x_2 \oplus x_4 \oplus x_5 \oplus x_7$$
$$y_2 = x_1 \oplus x_3 \oplus x_4 \oplus x_6 \oplus x_7$$
$$y_3 = x_2 \oplus x_3 \oplus x_4 \oplus x_8$$
$$y_4 =$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

$$y_1 = x_1 \oplus x_2 \oplus x_4 \oplus x_5 \oplus x_7$$
$$y_2 = x_1 \oplus x_3 \oplus x_4 \oplus x_6 \oplus x_7$$
$$y_3 = x_2 \oplus x_3 \oplus x_4 \oplus x_8$$
$$y_4 = x_5$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller x = 1-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

$$y_1 = x_1 \oplus x_2 \oplus x_4 \oplus x_5 \oplus x_7$$
$$y_2 = x_1 \oplus x_3 \oplus x_4 \oplus x_6 \oplus x_7$$
$$y_3 = x_2 \oplus x_3 \oplus x_4 \oplus x_8$$
$$y_4 = x_5 \oplus x_6$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller x = 1-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

$$y_1 = x_1 \oplus x_2 \oplus x_4 \oplus x_5 \oplus x_7$$
$$y_2 = x_1 \oplus x_3 \oplus x_4 \oplus x_6 \oplus x_7$$
$$y_3 = x_2 \oplus x_3 \oplus x_4 \oplus x_8$$
$$y_4 = x_5 \oplus x_6 \oplus x_7$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (II)

Beispiel: (12, 8) Hamming-Code

3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller x = 1-Komponenten seiner Reihe i.

Zeile	x_8	x_7	x_6	x_5	y_4	x_4	x_3	x_2	y_3	x_1	y_2	y_1
4	1	1	1	1	1	0	0	0	0	0	0	0
3	1	0	0	0	0	1	1	1	1	0	0	0
2	0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	0	1	0	1	0	1	0	1	0	1

$$y_1 = x_1 \oplus x_2 \oplus x_4 \oplus x_5 \oplus x_7$$
$$y_2 = x_1 \oplus x_3 \oplus x_4 \oplus x_6 \oplus x_7$$
$$y_3 = x_2 \oplus x_3 \oplus x_4 \oplus x_8$$
$$y_4 = x_5 \oplus x_6 \oplus x_7 \oplus x_8$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (III)

Hamming-Code

Aus dem Bildungsalgorithmus folgt damit unmittelbar:

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (III)

Hamming-Code

Aus dem Bildungsalgorithmus folgt damit unmittelbar:
Bei m Datenbits x_i werden k Prüfbits y_i zur Bildung des Hamming-Codes benötigt und es gilt im Allgemeinen:

$$2^k - k - 1 \geqslant m$$

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (III)

Hamming-Code

Aus dem Bildungsalgorithmus folgt damit unmittelbar:
Bei m Datenbits x_i werden k Prüfbits y_i zur Bildung des Hamming-Codes benötigt und es gilt im Allgemeinen:

$$2^k - k - 1 \geqslant m$$

$\rightsquigarrow$ Denn es gilt für die maximale Gesamtzahl der Bits $n = 2^k - 1$.

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (III)

Hamming-Code

Aus dem Bildungsalgorithmus folgt damit unmittelbar:
Bei m Datenbits x_i werden k Prüfbits y_i zur Bildung des Hamming-Codes benötigt und es gilt im Allgemeinen:

$$2^k - k - 1 \geqslant m$$

- $\rightsquigarrow$ Denn es gilt für die maximale Gesamtzahl der Bits $n = 2^k - 1$.
- $\rightsquigarrow$ Da man aber noch die k Prüfbits braucht, gilt für die Datenbits m die obere Schranke $2^k - \mathbf{k} - 1$.

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (III)

Hamming-Code

Aus dem Bildungsalgorithmus folgt damit unmittelbar:
Bei m Datenbits x_i werden k Prüfbits y_i zur Bildung des Hamming-Codes benötigt und es gilt im Allgemeinen:

$$2^k - k - 1 \geqslant m$$

- ⇝ Denn es gilt für die maximale Gesamtzahl der Bits $n = 2^k - 1$.
- ⇝ Da man aber noch die k Prüfbits braucht, gilt für die Datenbits m die obere Schranke $2^k - \mathbf{k} - 1$.
- ⇝ Weil wir möglichst mit der geringsten Bitanzahl übertragen wollen, suchen wir die kleinste obere Schranke, damit das kleinste k für das die Ungleichung erfüllt ist!

Aufgabe 4 – Fehlerkorrektur: Begriffsklärung (IV)

Hamming-Code

In algorithmischer Form:

Algorithmus 1 : Anzahl an Prüfbits aus Anzahl an Datenbits berechnen

Eingabe : Anzahl an Datenbits m

Ausgabe : Minimale Anzahl an Prüfbits k

int $k \leftarrow 0$;

solange $2^k - k - 1 < m$ **tue**

  $k \leftarrow k + 1$;

Gib k **aus**;

Aufgabe 4 – Fehlerkorrektur

Sei ein nicht fehlertolerantes Kommunikationssystem gegeben, das in der Lage ist, einstellige Hexadezimalzahlen zu übertragen. Es soll nun dahingehend erweitert werden, dass es mittels eines Hamming-Codes Zweifachfehler erkennen oder Einfachfehler korrigieren kann.

b) Wie viele Bits werden benötigt, um die jeweiligen Informationen und die Paritätsbits nach Hamming zu codieren? Wie lang wird das gesamte zu übertragende Codewort?

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Aufgabe 4 – Fehlerkorrektur – Beispiel

Algorithmus zum Erstellen eines Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.
2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits.
3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Aufgabe 4 – Fehlerkorrektur – Beispiel

Algorithmus zum Erstellen eines Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.
2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits.
3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

Zeile	1
1	1
2	0
3	0

Aufgabe 4 – Fehlerkorrektur – Beispiel

Algorithmus zum Erstellen eines Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.
2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits.
3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

Zeile	2	1
1	0	1
2	1	0
3	0	0

Aufgabe 4 – Fehlerkorrektur – Beispiel

Algorithmus zum Erstellen eines Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.
2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits.
3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

Zeile	3	2	1
1	1	0	1
2	1	1	0
3	0	0	0

Aufgabe 4 – Fehlerkorrektur – Beispiel

Algorithmus zum Erstellen eines Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.
2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits.
3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

Zeile	4	3	2	1
1	0	1	0	1
2	0	1	1	0
3	1	0	0	0

Aufgabe 4 – Fehlerkorrektur – Beispiel

Algorithmus zum Erstellen eines Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.
2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits.
3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

Zeile	5	4	3	2	1
1	1	0	1	0	1
2	0	0	1	1	0
3	1	1	0	0	0

Aufgabe 4 – Fehlerkorrektur – Beispiel

Algorithmus zum Erstellen eines Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.
2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits.
3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

Zeile	6	5	4	3	2	1
1	0	1	0	1	0	1
2	1	0	0	1	1	0
3	1	1	1	0	0	0

Aufgabe 4 – Fehlerkorrektur – Beispiel

Algorithmus zum Erstellen eines Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.
2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits.
3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

Zeile	7	6	5	4	3	2	1
1	1	0	1	0	1	0	1
2	1	1	0	0	1	1	0
3	1	1	1	1	0	0	0

Aufgabe 4 – Fehlerkorrektur – Beispiel

Algorithmus zum Erstellen eines Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.
2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits.
3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

Zeile	x_4	x_3	x_2	y_3	x_1	y_2	y_1
1	1	0	1	0	1	0	1
2	1	1	0	0	1	1	0
3	1	1	1	1	0	0	0

Aufgabe 4 – Fehlerkorrektur – Beispiel

Algorithmus zum Erstellen eines Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis *n* binär in absteigender Reihenfolge.
2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits.
3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe *i*.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

Zeile	x_4	x_3	x_2	y_3	x_1	y_2	y_1
1	1	0	1	0	1	0	1
2	1	1	0	0	1	1	0
3	1	1	1	1	0	0	0

$$y_1 =$$

Aufgabe 4 – Fehlerkorrektur – Beispiel

Algorithmus zum Erstellen eines Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.
2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits.
3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

Zeile	x_4	x_3	x_2	y_3	x_1	y_2	y_1
1	1	0	1	0	1	0	1
2	1	1	0	0	1	1	0
3	1	1	1	1	0	0	0

$$y_1 = x_1$$

Aufgabe 4 – Fehlerkorrektur – Beispiel

Algorithmus zum Erstellen eines Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.
2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits.
3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

Zeile	x_4	x_3	x_2	y_3	x_1	y_2	y_1
1	1	0	1	0	1	0	1
2	1	1	0	0	1	1	0
3	1	1	1	1	0	0	0

$$y_1 = x_1 \oplus x_2$$

Aufgabe 4 – Fehlerkorrektur – Beispiel

Algorithmus zum Erstellen eines Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.
2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits.
3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

Zeile	x_4	x_3	x_2	y_3	x_1	y_2	y_1
1	1	0	1	0	1	0	1
2	1	1	0	0	1	1	0
3	1	1	1	1	0	0	0

$$y_1 = x_1 \oplus x_2 \oplus x_4$$

Aufgabe 4 – Fehlerkorrektur – Beispiel

Algorithmus zum Erstellen eines Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.
2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits.
3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

Zeile	x_4	x_3	x_2	y_3	x_1	y_2	y_1
1	1	0	1	0	1	0	1
2	1	1	0	0	1	1	0
3	1	1	1	1	0	0	0

$$y_1 = x_1 \oplus x_2 \oplus x_4$$

$$y_2 =$$

Aufgabe 4 – Fehlerkorrektur – Beispiel

Algorithmus zum Erstellen eines Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.
2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits.
3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

Zeile	x_4	x_3	x_2	y_3	x_1	y_2	y_1
1	1	0	1	0	1	0	1
2	1	1	0	0	1	1	0
3	1	1	1	1	0	0	0

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1$$

Aufgabe 4 – Fehlerkorrektur – Beispiel

Algorithmus zum Erstellen eines Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.
2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits.
3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller x = 1-Komponenten seiner Reihe i.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

Zeile	x_4	x_3	x_2	y_3	x_1	y_2	y_1
1	1	0	1	0	1	0	1
2	1	1	0	0	1	1	0
3	1	1	1	1	0	0	0

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3$$

Aufgabe 4 – Fehlerkorrektur – Beispiel

Algorithmus zum Erstellen eines Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.
2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits.
3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

Zeile	x_4	x_3	x_2	y_3	x_1	y_2	y_1
1	1	0	1	0	1	0	1
2	1	1	0	0	1	1	0
3	1	1	1	1	0	0	0

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$

Aufgabe 4 – Fehlerkorrektur – Beispiel

Algorithmus zum Erstellen eines Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis *n* binär in absteigender Reihenfolge.
2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits.
3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe *i*.

Lösung: Wir brauchen einen $(7,4)$-Hamming-Code

Zeile	x_4	x_3	x_2	y_3	x_1	y_2	y_1
1	1	0	1	0	1	0	1
2	1	1	0	0	1	1	0
3	1	1	1	1	0	0	0

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 =$$

Aufgabe 4 – Fehlerkorrektur – Beispiel

Algorithmus zum Erstellen eines Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.
2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits.
3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

Zeile	x_4	x_3	x_2	y_3	x_1	y_2	y_1
1	1	0	1	0	1	0	1
2	1	1	0	0	1	1	0
3	1	1	1	1	0	0	0

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2$$

Aufgabe 4 – Fehlerkorrektur – Beispiel

Algorithmus zum Erstellen eines Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.
2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits.
3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

Zeile	x_4	x_3	x_2	y_3	x_1	y_2	y_1
1	1	0	1	0	1	0	1
2	1	1	0	0	1	1	0
3	1	1	1	1	0	0	0

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3$$

Aufgabe 4 – Fehlerkorrektur – Beispiel

Algorithmus zum Erstellen eines Hamming-Code

1. Schreibe die natürlichen Zahlen von 1 bis n binär in absteigender Reihenfolge.
2. Die Spalten mit genau einer 1 (also die 2-er Potenzen) werden zu Paritätsbits, der Rest zu Datenbits.
3. Jedes Paritätsbit y_i ist das Ergebnis der Ver-XOR-ung aller $x = 1$-Komponenten seiner Reihe i.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

Zeile	x_4	x_3	x_2	y_3	x_1	y_2	y_1
1	1	0	1	0	1	0	1
2	1	1	0	0	1	1	0
3	1	1	1	1	0	0	0

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0				0
0	0	0	1				1
0	0	1	0				2
0	0	1	1				3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen (7, 4)-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0				0
0	0	0	1				1
0	0	1	0				2
0	0	1	1				3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen (7, 4)-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0			0	0
0	0	0	1				1
0	0	1	0				2
0	0	1	1				3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0			0	0
0	0	0	1				1
0	0	1	0				2
0	0	1	1				3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 0 = 0$$

$$y_2 = 0 \oplus 0 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen (7, 4)-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0		0	0	0
0	0	0	1				1
0	0	1	0				2
0	0	1	1				3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 0 = 0$$

$$y_2 = 0 \oplus 0 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0		0	0	0
0	0	0	1				1
0	0	1	0				2
0	0	1	1				3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 0 = 0$$
$$y_2 = 0 \oplus 0 \oplus 0 = 0$$
$$y_3 = 0 \oplus 0 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1				1
0	0	1	0				2
0	0	1	1				3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 0 = 0$$
$$y_2 = 0 \oplus 0 \oplus 0 = 0$$
$$y_3 = 0 \oplus 0 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen (7, 4)-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1				1
0	0	1	0				2
0	0	1	1				3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.

Lösung: Wir brauchen einen (7, 4)-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1			1	1
0	0	1	0				2
0	0	1	1				3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$

$$y_2 = x_1 \oplus x_3 \oplus x_4$$

$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1			1	1
0	0	1	0				2
0	0	1	1				3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 0 = 1$$
$$y_2 = 1 \oplus 0 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1		1	1	1
0	0	1	0				2
0	0	1	1				3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 0 = 1$$
$$y_2 = 1 \oplus 0 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1		1	1	1
0	0	1	0				2
0	0	1	1				3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 0 = 1$$
$$y_2 = 1 \oplus 0 \oplus 0 = 1$$
$$y_3 = 0 \oplus 0 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0				2
0	0	1	1				3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 0 = 1$$
$$y_2 = 1 \oplus 0 \oplus 0 = 1$$
$$y_3 = 0 \oplus 0 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0				2
0	0	1	1				3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 1 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0			1	2
0	0	1	1				3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 1 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0			1	2
0	0	1	1				3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 1 \oplus 0 = 1$$

$$y_2 = 0 \oplus 0 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0		0	1	2
0	0	1	1				3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 1 \oplus 0 = 1$$

$$y_2 = 0 \oplus 0 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0		0	1	2
0	0	1	1				3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 1 \oplus 0 = 1$$
$$y_2 = 0 \oplus 0 \oplus 0 = 0$$
$$y_3 = 1 \oplus 0 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1				3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 1 \oplus 0 = 1$$
$$y_2 = 0 \oplus 0 \oplus 0 = 0$$
$$y_3 = 1 \oplus 0 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1				3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1			0	3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.

Lösung: Wir brauchen einen (7, 4)-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1			0	3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$

$$y_2 = x_1 \oplus x_3 \oplus x_4$$

$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 0 = 0$$

$$y_2 = 1 \oplus 0 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1		1	0	3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$

$$y_2 = x_1 \oplus x_3 \oplus x_4$$

$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 0 = 0$$

$$y_2 = 1 \oplus 0 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1		1	0	3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 0 = 0$$
$$y_2 = 1 \oplus 0 \oplus 0 = 1$$
$$y_3 = 1 \oplus 0 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 0 = 0$$
$$y_2 = 1 \oplus 0 \oplus 0 = 1$$
$$y_3 = 1 \oplus 0 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0				4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0			0	4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.
Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0			0	4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 0 = 0$$

$$y_2 = 0 \oplus 1 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0		1	0	4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 0 = 0$$
$$y_2 = 0 \oplus 1 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen (7, 4)-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0		1	0	4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 0 = 0$$
$$y_2 = 0 \oplus 1 \oplus 0 = 1$$
$$y_3 = 0 \oplus 1 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0	1	1	0	4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 0 = 0$$
$$y_2 = 0 \oplus 1 \oplus 0 = 1$$
$$y_3 = 0 \oplus 1 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0	1	1	0	4
0	1	0	1				5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen (7, 4)-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0	1	1	0	4
0	1	0	1			1	5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.
Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0	1	1	0	4
0	1	0	1			1	5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 0 = 1$$

$$y_2 = 1 \oplus 1 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0	1	1	0	4
0	1	0	1		0	1	5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 0 = 1$$

$$y_2 = 1 \oplus 1 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.

Lösung: Wir brauchen einen $(7,4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0	1	1	0	4
0	1	0	1		0	1	5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 0 = 1$$
$$y_2 = 1 \oplus 1 \oplus 0 = 0$$
$$y_3 = 0 \oplus 1 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0	1	1	0	4
0	1	0	1	1	0	1	5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 0 = 1$$
$$y_2 = 1 \oplus 1 \oplus 0 = 0$$
$$y_3 = 0 \oplus 1 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.

Lösung: Wir brauchen einen (7, 4)-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0	1	1	0	4
0	1	0	1	1	0	1	5
0	1	1	0				6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0	1	1	0	4
0	1	0	1	1	0	1	5
0	1	1	0			1	6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0	1	1	0	4
0	1	0	1	1	0	1	5
0	1	1	0			1	6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$

$$y_2 = x_1 \oplus x_3 \oplus x_4$$

$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 0 = 0$$

$$y_2 = 0 \oplus 1 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0	1	1	0	4
0	1	0	1	1	0	1	5
0	1	1	0		1	1	6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 0 = 0$$
$$y_2 = 0 \oplus 1 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7,4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0	1	1	0	4
0	1	0	1	1	0	1	5
0	1	1	0		1	1	6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 0 = 0$$
$$y_2 = 0 \oplus 1 \oplus 0 = 1$$
$$y_3 = 1 \oplus 1 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.
Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0	1	1	0	4
0	1	0	1	1	0	1	5
0	1	1	0	0	1	1	6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 0 = 0$$
$$y_2 = 0 \oplus 1 \oplus 0 = 1$$
$$y_3 = 1 \oplus 1 \oplus 0 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen (7, 4)-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0	1	1	0	4
0	1	0	1	1	0	1	5
0	1	1	0	0	1	1	6
0	1	1	1				7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0	1	1	0	4
0	1	0	1	1	0	1	5
0	1	1	0	0	1	1	6
0	1	1	1			0	7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0	1	1	0	4
0	1	0	1	1	0	1	5
0	1	1	0	0	1	1	6
0	1	1	1			0	7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 0 = 0$$

$$y_2 = 1 \oplus 1 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0	1	1	0	4
0	1	0	1	1	0	1	5
0	1	1	0	0	1	1	6
0	1	1	1		0	0	7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 0 = 0$$
$$y_2 = 1 \oplus 1 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0	1	1	0	4
0	1	0	1	1	0	1	5
0	1	1	0	0	1	1	6
0	1	1	1		0	0	7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 0 = 0$$
$$y_2 = 1 \oplus 1 \oplus 0 = 0$$
$$y_3 = 1 \oplus 1 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0	1	1	0	4
0	1	0	1	1	0	1	5
0	1	1	0	0	1	1	6
0	1	1	1	0	0	0	7

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 0 = 0$$
$$y_2 = 1 \oplus 1 \oplus 0 = 0$$
$$y_3 = 1 \oplus 1 \oplus 0 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.
Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0				8
1	0	0	1				9
1	0	1	0				A
1	0	1	1				B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0			1	8
1	0	0	1				9
1	0	1	0				A
1	0	1	1				B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.
Lösung: Wir brauchen einen (7, 4)-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0			1	8
1	0	0	1				9
1	0	1	0				A
1	0	1	1				B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 1 = 1$$
$$y_2 = 0 \oplus 0 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7,4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0		1	1	8
1	0	0	1				9
1	0	1	0				A
1	0	1	1				B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 1 = 1$$
$$y_2 = 0 \oplus 0 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.
Lösung: Wir brauchen einen (7, 4)-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0		1	1	8
1	0	0	1				9
1	0	1	0				A
1	0	1	1				B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 1 = 1$$
$$y_2 = 0 \oplus 0 \oplus 1 = 1$$
$$y_3 = 0 \oplus 0 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0		1	1	8
1	0	0	1				9
1	0	1	0				A
1	0	1	1				B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 1 = 1$$
$$y_2 = 0 \oplus 0 \oplus 1 = 1$$
$$y_3 = 0 \oplus 0 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0		1	1	8
1	0	0	1				9
1	0	1	0				A
1	0	1	1				B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1			0	9
1	0	1	0				A
1	0	1	1				B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen (7, 4)-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1			0	9
1	0	1	0				A
1	0	1	1				B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 1 = 0$$
$$y_2 = 1 \oplus 0 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1		0	0	9
1	0	1	0				A
1	0	1	1				B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 1 = 0$$
$$y_2 = 1 \oplus 0 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1		0	0	9
1	0	1	0				A
1	0	1	1				B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 1 = 0$$
$$y_2 = 1 \oplus 0 \oplus 1 = 0$$
$$y_3 = 0 \oplus 0 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.
Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0				A
1	0	1	1				B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 1 = 0$$
$$y_2 = 1 \oplus 0 \oplus 1 = 0$$
$$y_3 = 0 \oplus 0 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0				A
1	0	1	1				B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 1 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.

Lösung: Wir brauchen einen (7, 4)-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0			0	A
1	0	1	1				B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 1 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen (7, 4)-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0			0	A
1	0	1	1				B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 1 \oplus 1 = 0$$
$$y_2 = 0 \oplus 0 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0		1	0	A
1	0	1	1				B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 1 \oplus 1 = 0$$
$$y_2 = 0 \oplus 0 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0		1	0	A
1	0	1	1				B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 1 \oplus 1 = 0$$
$$y_2 = 0 \oplus 0 \oplus 1 = 1$$
$$y_3 = 1 \oplus 0 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.
Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1				B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 1 \oplus 1 = 0$$
$$y_2 = 0 \oplus 0 \oplus 1 = 1$$
$$y_3 = 1 \oplus 0 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1				B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1			1	B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1			1	B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 1 = 1$$
$$y_2 = 1 \oplus 0 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1		0	1	B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 1 = 1$$
$$y_2 = 1 \oplus 0 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.
Lösung: Wir brauchen einen $(7,4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1		0	1	B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 1 = 1$$
$$y_2 = 1 \oplus 0 \oplus 1 = 0$$
$$y_3 = 1 \oplus 0 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.
Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 1 = 1$$
$$y_2 = 1 \oplus 0 \oplus 1 = 0$$
$$y_3 = 1 \oplus 0 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen (7, 4)-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0				C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen (7, 4)-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0			1	C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0			1	C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 1 = 1$$
$$y_2 = 0 \oplus 1 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0		0	1	C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 1 = 1$$
$$y_2 = 0 \oplus 1 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0		0	1	C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 1 = 1$$
$$y_2 = 0 \oplus 1 \oplus 1 = 0$$
$$y_3 = 0 \oplus 1 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.
Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0	0	0	1	C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 1 = 1$$
$$y_2 = 0 \oplus 1 \oplus 1 = 0$$
$$y_3 = 0 \oplus 1 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.
Lösung: Wir brauchen einen (7, 4)-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0	0	0	1	C
1	1	0	1				D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen (7, 4)-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0	0	0	1	C
1	1	0	1			0	D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0	0	0	1	C
1	1	0	1			0	D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 1 = 0$$
$$y_2 = 1 \oplus 1 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen (7, 4)-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0	0	0	1	C
1	1	0	1		1	0	D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 1 = 0$$
$$y_2 = 1 \oplus 1 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.
Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0	0	0	1	C
1	1	0	1		1	0	D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 1 = 0$$
$$y_2 = 1 \oplus 1 \oplus 1 = 1$$
$$y_3 = 0 \oplus 1 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.
Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0	0	0	1	C
1	1	0	1	0	1	0	D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 0 \oplus 1 = 0$$
$$y_2 = 1 \oplus 1 \oplus 1 = 1$$
$$y_3 = 0 \oplus 1 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen (7, 4)-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0	0	0	1	C
1	1	0	1	0	1	0	D
1	1	1	0				E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0	0	0	1	C
1	1	0	1	0	1	0	D
1	1	1	0			0	E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0	0	0	1	C
1	1	0	1	0	1	0	D
1	1	1	0			0	E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 1 = 1$$
$$y_2 = 0 \oplus 1 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0	0	0	1	C
1	1	0	1	0	1	0	D
1	1	1	0		0	0	E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 1 = 1$$
$$y_2 = 0 \oplus 1 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0	0	0	1	C
1	1	0	1	0	1	0	D
1	1	1	0		0	0	E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 1 = 1$$
$$y_2 = 0 \oplus 1 \oplus 1 = 0$$
$$y_3 = 1 \oplus 1 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0	0	0	1	C
1	1	0	1	0	1	0	D
1	1	1	0	1	0	0	E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 0 \oplus 0 \oplus 1 = 1$$
$$y_2 = 0 \oplus 1 \oplus 1 = 0$$
$$y_3 = 1 \oplus 1 \oplus 1 = 0$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0	0	0	1	C
1	1	0	1	0	1	0	D
1	1	1	0	1	0	0	E
1	1	1	1				F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0	0	0	1	C
1	1	0	1	0	1	0	D
1	1	1	0	1	0	0	E
1	1	1	1			1	F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0	0	0	1	C
1	1	0	1	0	1	0	D
1	1	1	0	1	0	0	E
1	1	1	1			1	F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 1 = 1$$
$$y_2 = 1 \oplus 1 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0	0	0	1	C
1	1	0	1	0	1	0	D
1	1	1	0	1	0	0	E
1	1	1	1		1	1	F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 1 = 1$$
$$y_2 = 1 \oplus 1 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0	0	0	1	C
1	1	0	1	0	1	0	D
1	1	1	0	1	0	0	E
1	1	1	1		1	1	F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 1 = 1$$
$$y_2 = 1 \oplus 1 \oplus 1 = 1$$
$$y_3 = 1 \oplus 1 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \dots x_1 y_k \dots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0	0	0	1	C
1	1	0	1	0	1	0	D
1	1	1	0	1	0	0	E
1	1	1	1	1	1	1	F

$$y_1 = x_1 \oplus x_2 \oplus x_4$$
$$y_2 = x_1 \oplus x_3 \oplus x_4$$
$$y_3 = x_2 \oplus x_3 \oplus x_4$$

$$y_1 = 1 \oplus 1 \oplus 1 = 1$$
$$y_2 = 1 \oplus 1 \oplus 1 = 1$$
$$y_3 = 1 \oplus 1 \oplus 1 = 1$$

Aufgabe 4 – Fehlerkorrektur

c) Erstellen Sie nun den Hamming-Code und ordnen Sie den Codewörtern die entsprechenden Hexadezimalwerte zu, die der Wertigkeit der Informationsstellen entsprechen sollen. Der Aufbau der Codewörter soll wie folgt aussehen: $x_m \ldots x_1 y_k \ldots y_1$.

Lösung: Wir brauchen einen $(7, 4)$-Hamming-Code

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0	1	1	0	4
0	1	0	1	1	0	1	5
0	1	1	0	0	1	1	6
0	1	1	1	0	0	0	7

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0	0	0	1	C
1	1	0	1	0	1	0	D
1	1	1	0	1	0	0	E
1	1	1	1	1	1	1	F

Aufgabe 4 – Fehlerkorrektur

Bei einer Übertragung mit diesem Kommunikationssystem wurde folgende Binärfolge empfangen:

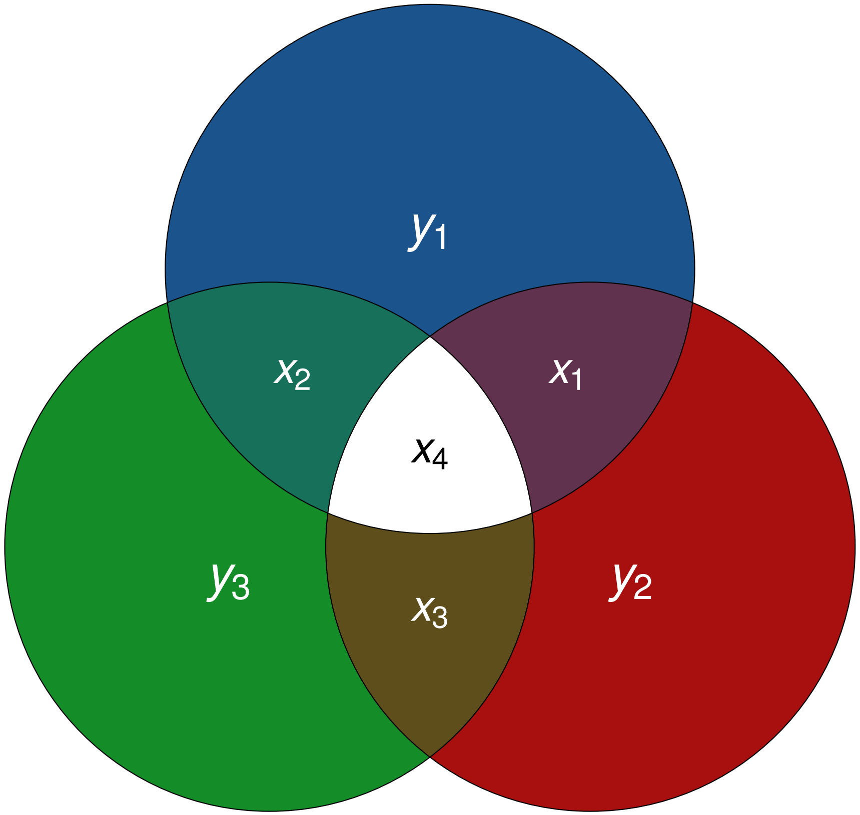
0 1 0 1 0 0 0 0 1 0 0 1 0 0 1 0 0 1 1 0 0 1 1 1 0 1 1 1 1 1 1 0 1 0 0

d) Überprüfen Sie anhand Ihrer Code-Tabelle, ob der Empfang der Codewörter fehlerfrei erfolgt ist, und führen Sie falls notwendig eine Korrektur durch.

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
0	0	0	0	0	0	0	0
0	0	0	1	0	1	1	1
0	0	1	0	1	0	1	2
0	0	1	1	1	1	0	3
0	1	0	0	1	1	0	4
0	1	0	1	1	0	1	5
0	1	1	0	0	1	1	6
0	1	1	1	0	0	0	7

x_4	x_3	x_2	x_1	y_3	y_2	y_1	HEX
1	0	0	0	1	1	1	8
1	0	0	1	1	0	0	9
1	0	1	0	0	1	0	A
1	0	1	1	0	0	1	B
1	1	0	0	0	0	1	C
1	1	0	1	0	1	0	D
1	1	1	0	1	0	0	E
1	1	1	1	1	1	1	F

Aufgabe 4 – Fehlerkorrektur: Hamming-Code im Venndiagramm



Aufgabe 5 – Huffman-Code

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

`ABRAKADABRASIMSALABIM`

b) Wieviel Bits werden durch diese Codierung im Vergleich zu einer Codierung mit einer festen Codewort-Länge eingespart? (Das Codebuch ist zu vernachlässigen.)

c) Wieviel Bits sind minimal nötig (optimale Codierung)? Wieviel Prozent schlechter ist der Huffman-Code?

Aufgabe 5 – Huffman-Code

Definition 4 (Optimale Codes)

Wir bezeichnen einen Code als optimal bzgl. einer Wahrscheinlichkeitsverteilung p, wenn die durchschnittliche Codewortlänge minimal sind.
Die **durchschnittliche Codewortlänge** $\overline{m}$ berechnet sich durch:

$$\overline{m} = \sum_{i=1}^{n} p(x_i) \cdot m(x_i)$$

Der Idealwert eines optimalen Codes bezeichnen wir als **Entropie** und berechnet sich durch:

$$H = \sum_{i=1}^{n} p(x_i) \cdot I(x_i)$$

Aufgabe 5 – Huffman-Code

Definition 5 (Huffman-Code)

Der Huffman-Code stellt einen nahezu optimalen, präfixfreien Code dar, bei dem eine Codierung mit variabler Bitlänge verwendet wird.

Huffman-Kodierungs-Algorithmus[a]:

Schritt 1: Sortiere die vorkommenden Zeichen der zu codierenden Nachricht N **aufsteigend** nach ihrer Häufigkeit ($\equiv$ Liste Q).

Schritt 2: Finde aus der sortierten Liste Q die beiden Minimas z_l und z_r.

Schritt 3: Verschmelze z_l und z_r zu einem neuen Element z. Die Häufigkeit von z ist die Summe der Häufigkeiten von z_l und z_r.

Schritt 4: Sortiere z in Q gemäß seiner Häufigkeit ein.

Schritt 5: Ist nur noch ein Element in Q vorhanden, so ist dieses Element die Wurzel des Codierungsbaums, breche den Algorithmus ab.

Schritt 6: Sonst: Springe zu Schritt 2.

[a] nach Folie 09-16f.

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

`ABRAKADABRASIMSALABIM`

Aufgabe 5 – Huffman-Code

Sei folgende Kodierung nun gegeben:

i	Zeichen x_i	Code	Anzahl N_i
1	A	11	7
2	B	101	3
3	I	000	2
4	M	001	2
5	R	010	2
6	S	011	2
7	L	1000	1
8	K	10010	1
9	D	10011	1

b) Wieviel Bits werden durch diese Codierung im Vergleich zu einer Codierung mit einer festen Codewort-Länge eingespart? (Das Codebuch ist zu vernachlässigen.)

Aufgabe 5 – Huffman-Code

Seien die Häufigkeiten nun nochmal gegeben:

Buchstabe	A	B	I	M	R	S	L	K	D
Anzahl	7	3	2	2	2	2	1	1	1

c) Wieviel Bits sind minimal nötig (optimale Codierung)? Wieviel Prozent schlechter ist der Huffman-Code?

Aufgabe 5 – Huffman-Code

Seien die Häufigkeiten nun nochmal gegeben:

Buchstabe	A	B	I	M	R	S	L	K	D
Anzahl	7	3	2	2	2	2	1	1	1

c) Wieviel Bits sind minimal nötig (optimale Codierung)? Wieviel Prozent schlechter ist der Huffman-Code?

Optimale Codierung

Die theoretisch minimale Anzahl an Bits zur Codierung eines Zeichens x entspricht dessen Informationsgehalt

$$I(x) = -\, ld\left(\frac{\text{Anzahl}(x)}{\text{Gesamtzeichenanzahl}}\right)$$

Übungen zur Grundlagen der Technischen Informatik

Übung 2A – Huffman-Codierung

Florian Frank

Friedrich-Alexander-Universität Erlangen-Nürnberg

Wintersemester 2018/19

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung:

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Zählen wir zuerst die Häufigkeiten aller vorkommenden Zeichen:

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Zählen wir zuerst die Häufigkeiten aller vorkommenden Zeichen:

Buchstabe	A
Anzahl	1

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Zählen wir zuerst die Häufigkeiten aller vorkommenden Zeichen:

Buchstabe	A	B
Anzahl	1	1

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Zählen wir zuerst die Häufigkeiten aller vorkommenden Zeichen:

Buchstabe	A	B	R
Anzahl	1	1	1

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Zählen wir zuerst die Häufigkeiten aller vorkommenden Zeichen:

Buchstabe	A	B	R
Anzahl	2	1	1

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRA**K**ADABRASIMSALABIM

Lösung: Zählen wir zuerst die Häufigkeiten aller vorkommenden Zeichen:

Buchstabe	A	B	R	K
Anzahl	2	1	1	1

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKA**A**DABRASIMSALABIM

Lösung: Zählen wir zuerst die Häufigkeiten aller vorkommenden Zeichen:

Buchstabe	A	B	R	K
Anzahl	3	1	1	1

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKA**D**ABRASIMSALABIM

Lösung: Zählen wir zuerst die Häufigkeiten aller vorkommenden Zeichen:

Buchstabe	A	B	R	K	D
Anzahl	3	1	1	1	1

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKAD**A**BRASIMSALABIM

Lösung: Zählen wir zuerst die Häufigkeiten aller vorkommenden Zeichen:

Buchstabe	A	B	R	K	D
Anzahl	4	1	1	1	1

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADA**B**RASIMSALABIM

Lösung: Zählen wir zuerst die Häufigkeiten aller vorkommenden Zeichen:

Buchstabe	A	B	R	K	D
Anzahl	4	2	1	1	1

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADAB**R**ASIMSALABIM

Lösung: Zählen wir zuerst die Häufigkeiten aller vorkommenden Zeichen:

Buchstabe	A	B	R	K	D
Anzahl	4	2	2	1	1

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABR**A**SIMSALABIM

Lösung: Zählen wir zuerst die Häufigkeiten aller vorkommenden Zeichen:

Buchstabe	A	B	R	K	D
Anzahl	5	2	2	1	1

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRA**S**IMSALABIM

Lösung: Zählen wir zuerst die Häufigkeiten aller vorkommenden Zeichen:

Buchstabe	A	B	R	K	D	S
Anzahl	5	2	2	1	1	1

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Zählen wir zuerst die Häufigkeiten aller vorkommenden Zeichen:

Buchstabe	A	B	R	K	D	S	I
Anzahl	5	2	2	1	1	1	1

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASI**M**SALABIM

Lösung: Zählen wir zuerst die Häufigkeiten aller vorkommenden Zeichen:

Buchstabe	A	B	R	K	D	S	I	M
Anzahl	5	2	2	1	1	1	1	1

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIM**S**ALABIM

Lösung: Zählen wir zuerst die Häufigkeiten aller vorkommenden Zeichen:

Buchstabe	A	B	R	K	D	S	I	M
Anzahl	5	2	2	1	1	2	1	1

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Zählen wir zuerst die Häufigkeiten aller vorkommenden Zeichen:

Buchstabe	A	B	R	K	D	S	I	M
Anzahl	6	2	2	1	1	2	1	1

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Zählen wir zuerst die Häufigkeiten aller vorkommenden Zeichen:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	6	2	2	1	1	2	1	1	1

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSAL**A**BIM

Lösung: Zählen wir zuerst die Häufigkeiten aller vorkommenden Zeichen:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	2	2	1	1	2	1	1	1

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALA**B**IM

Lösung: Zählen wir zuerst die Häufigkeiten aller vorkommenden Zeichen:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	1	1	1

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Zählen wir zuerst die Häufigkeiten aller vorkommenden Zeichen:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	1	1

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABI**M**

Lösung: Zählen wir zuerst die Häufigkeiten aller vorkommenden Zeichen:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten → Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1

`K: 1` `D: 1` `L: 1` `R: 2` `S: 2` `I: 2` `M: 2` `B: 3` `A: 7`

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten $\rightarrow$ Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1

`K: 1` `D: 1` `L: 1` `R: 2` `S: 2` `I: 2` `M: 2` `B: 3` `A: 7`

Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten → Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1

`K: 1` `D: 1` `L: 1` `R: 2` `S: 2` `I: 2` `M: 2` `B: 3` `A: 7`

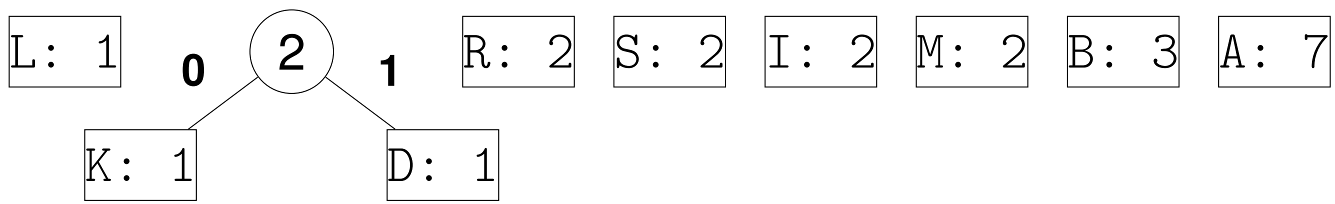
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten → Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1



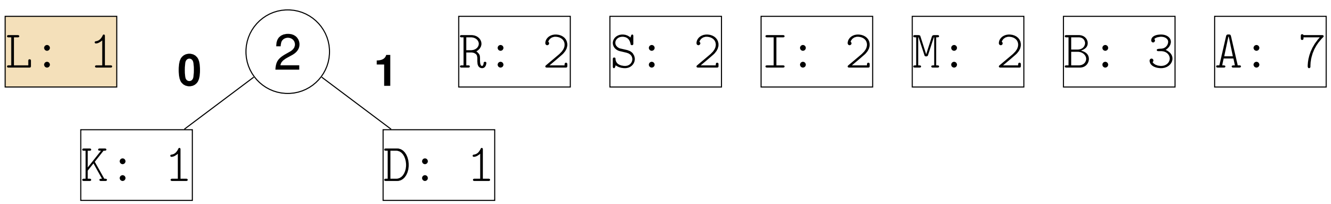
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten $\rightarrow$ Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1



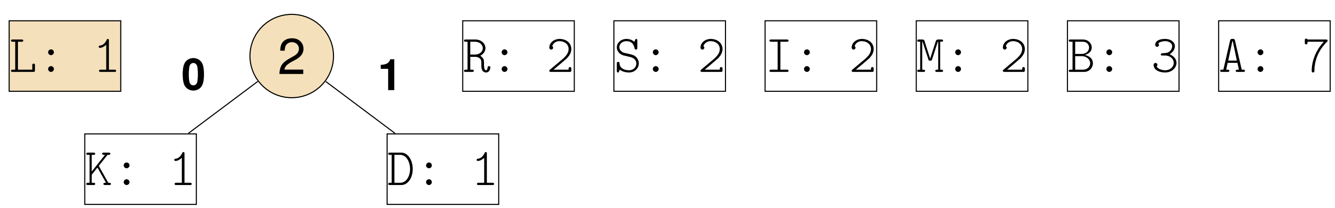
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten → Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1



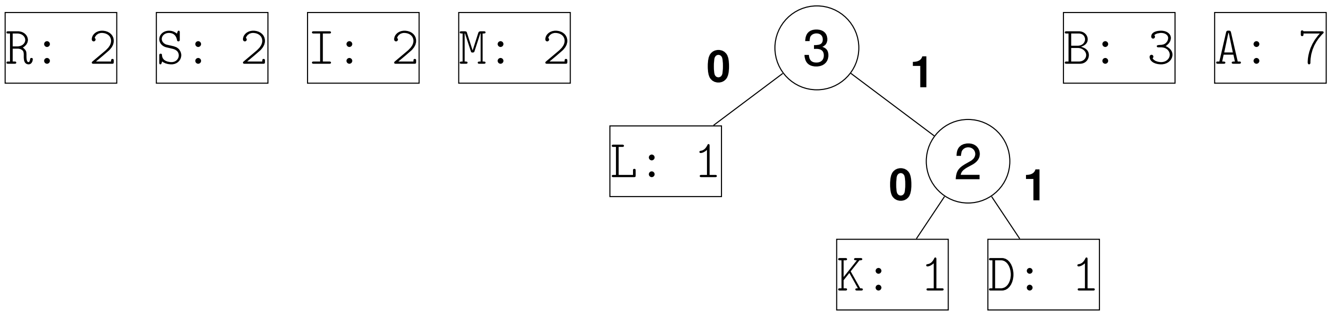
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten $\rightarrow$ Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1



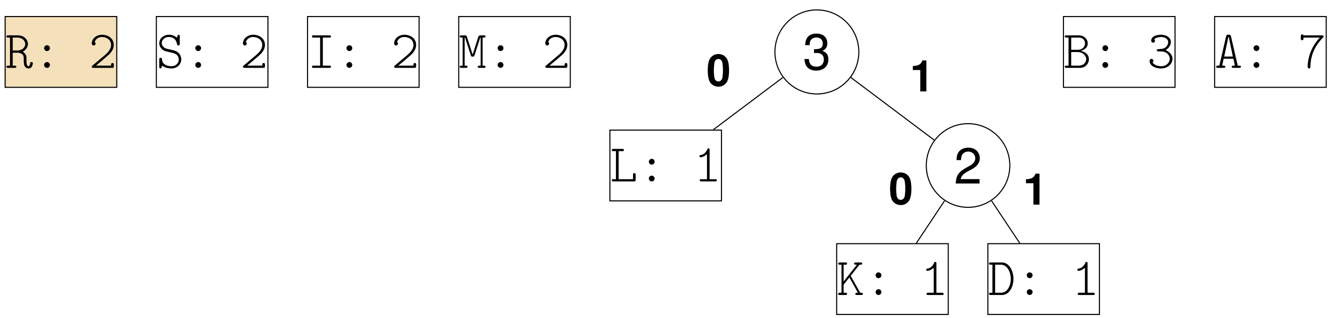
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten → Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1



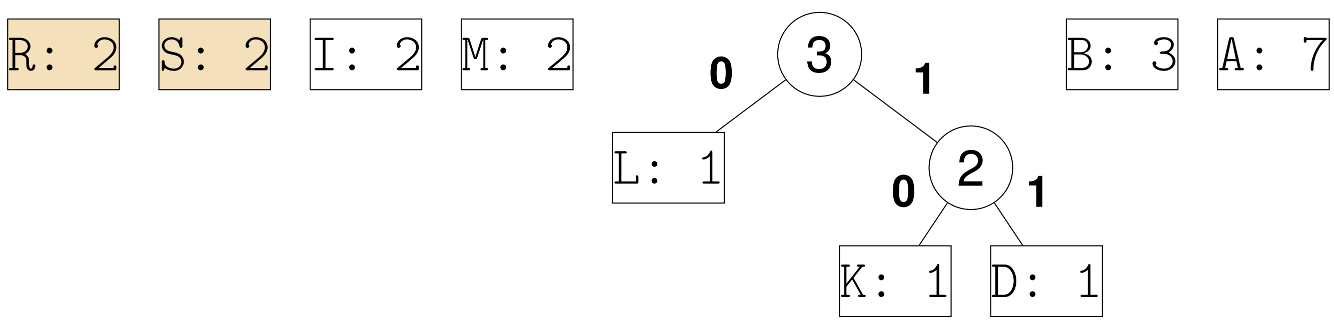
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten $\rightarrow$ Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1



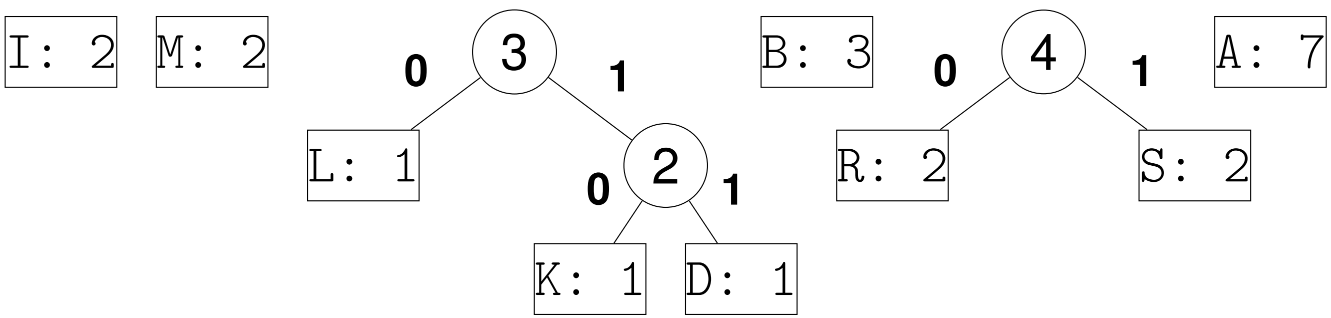
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten → Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1



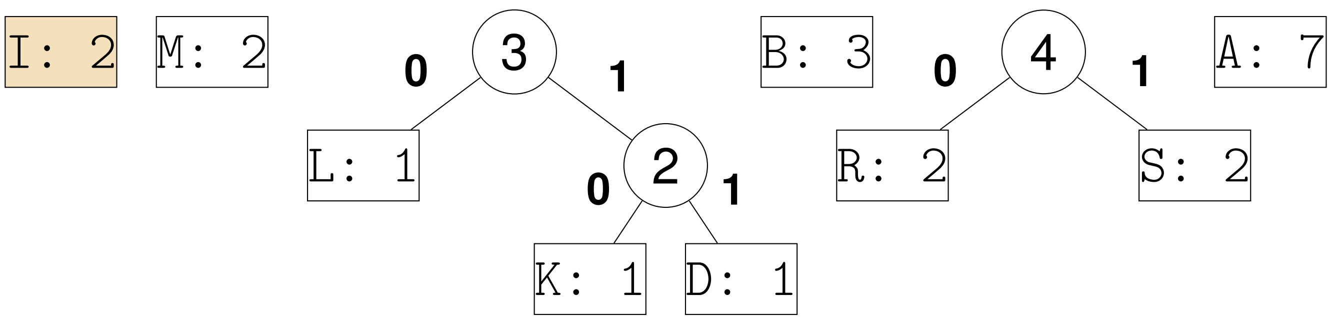
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten → Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1



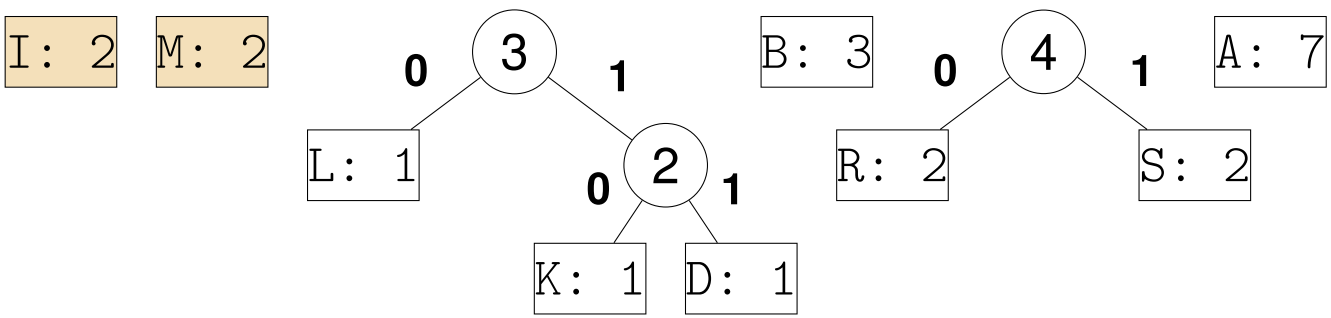
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten → Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1



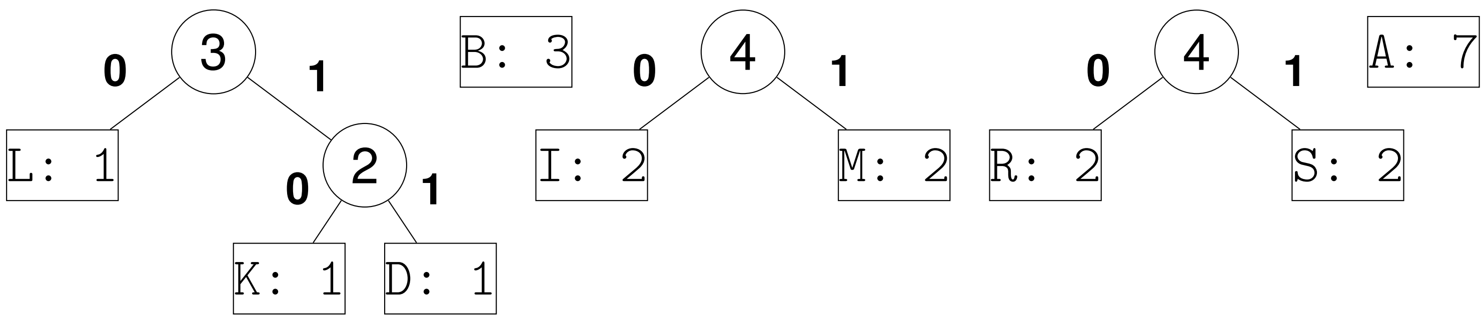
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten → Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1



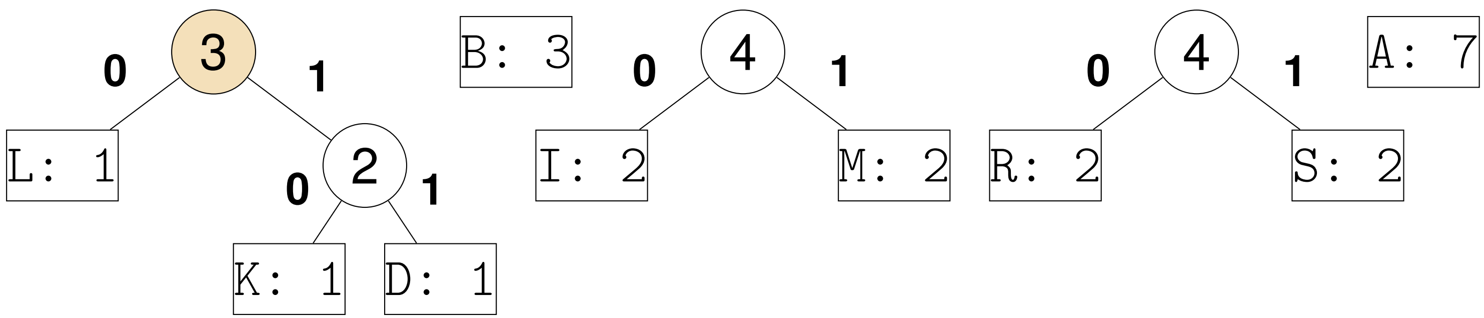
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten → Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1



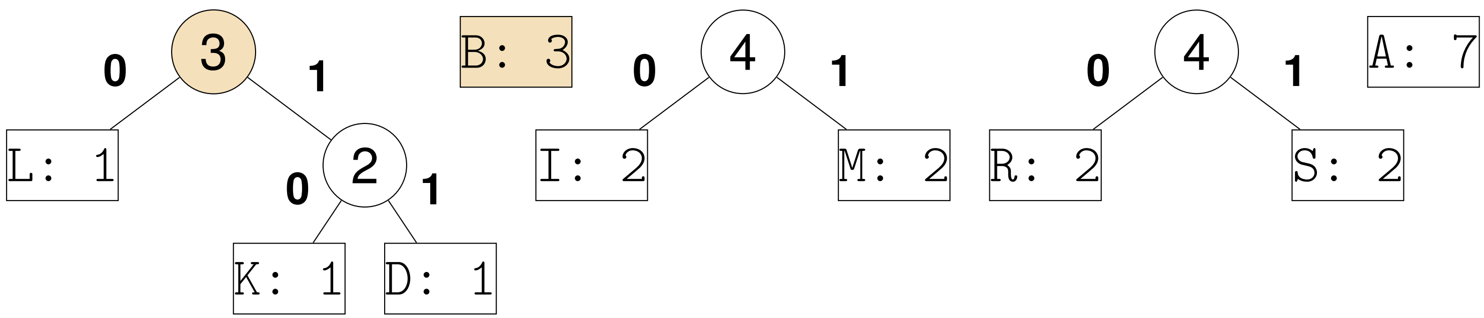
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten → Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1



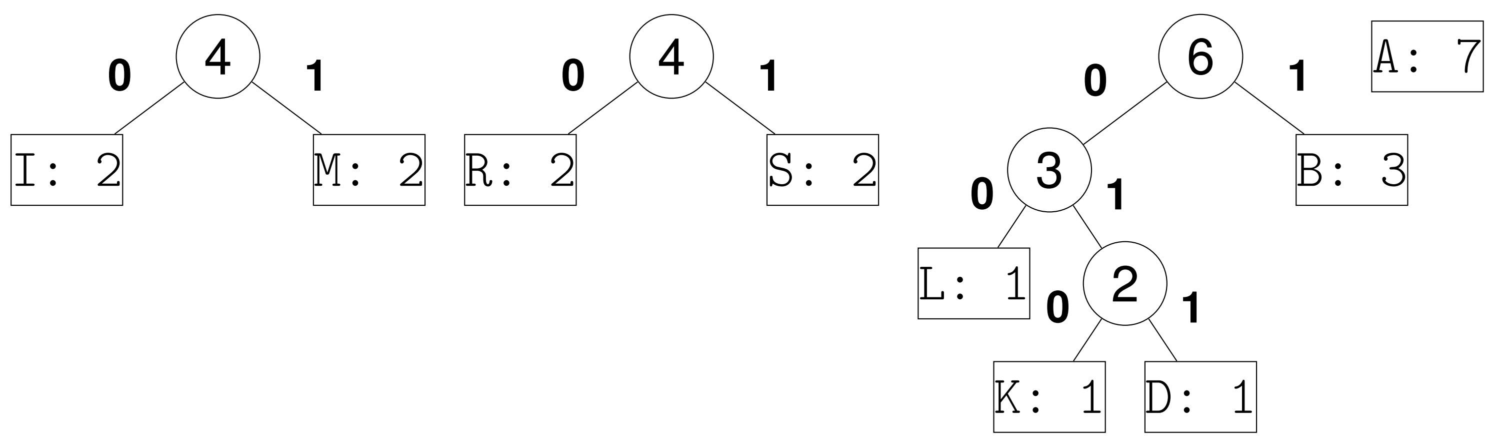
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten → Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1



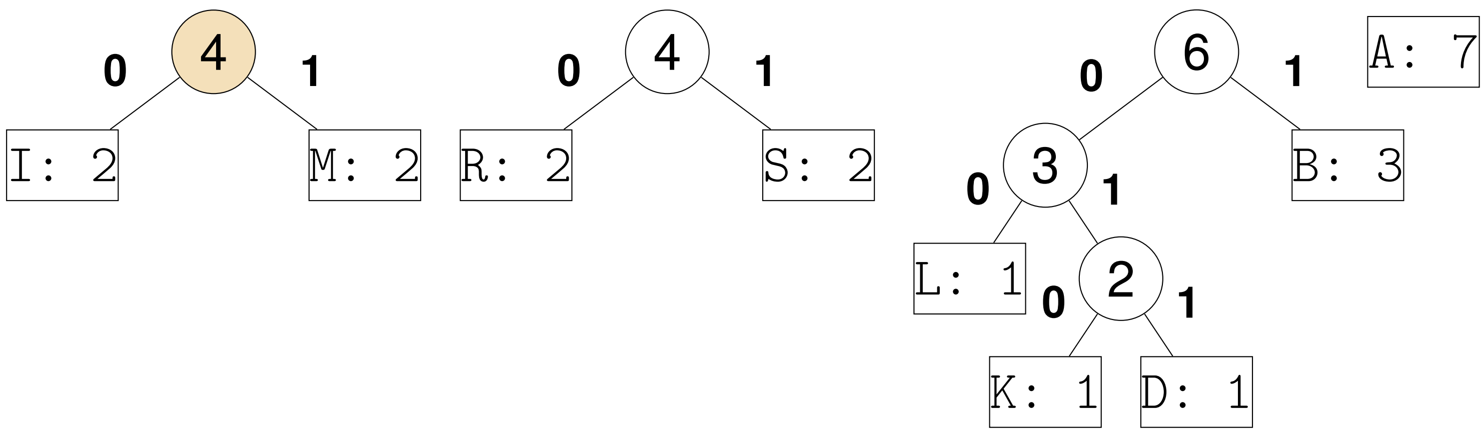
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten $\rightarrow$ Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1



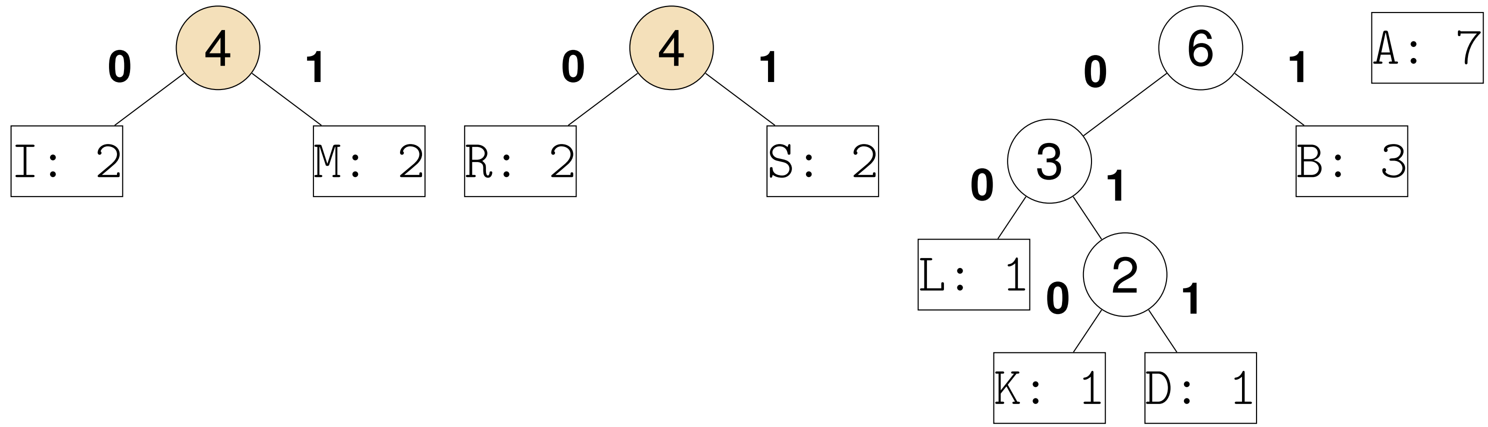
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten → Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1



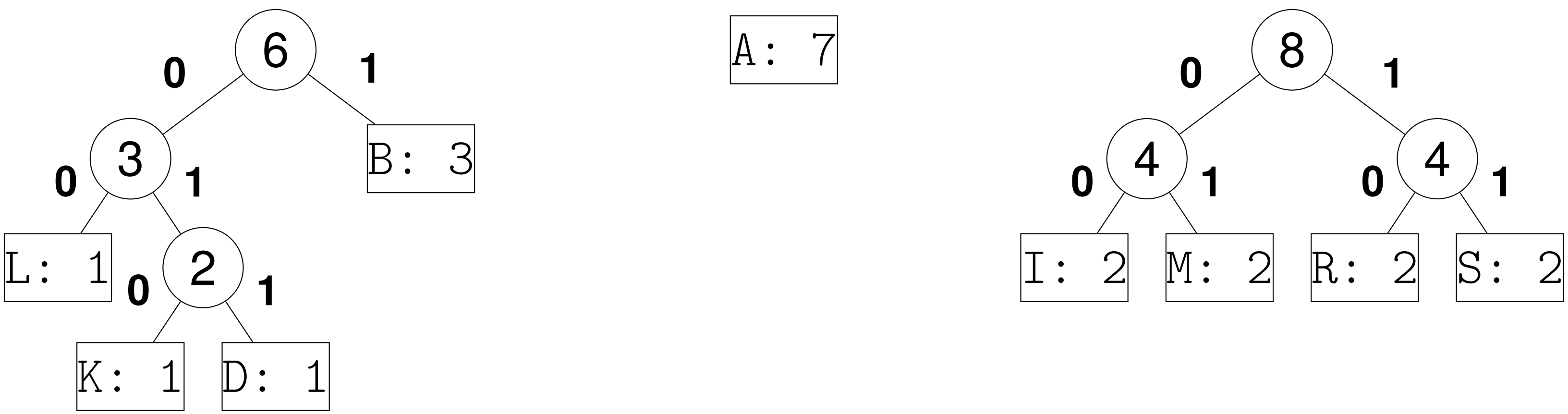
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten $\rightarrow$ Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1



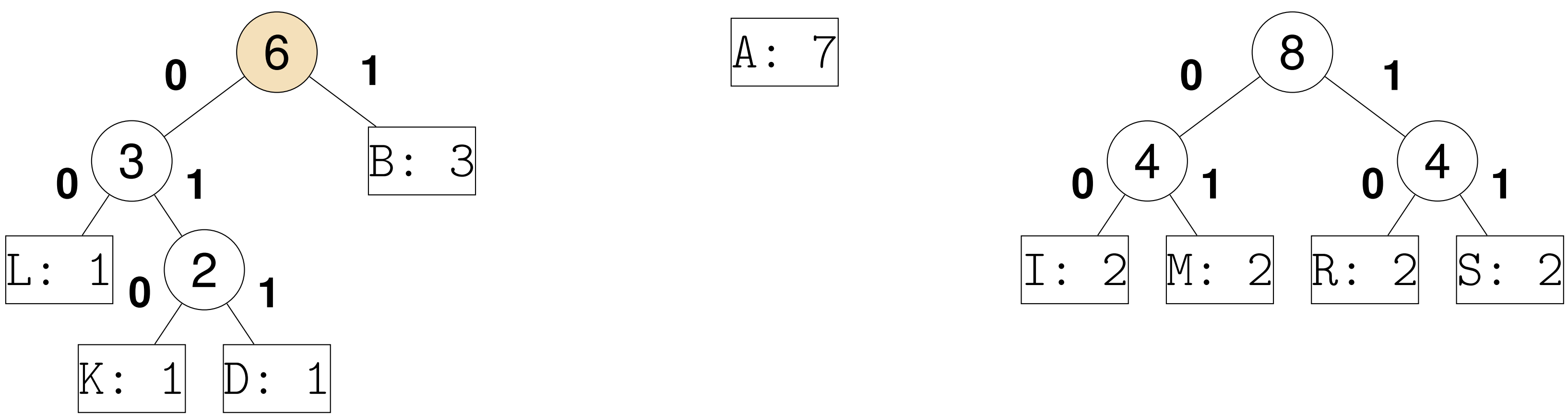
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten → Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1



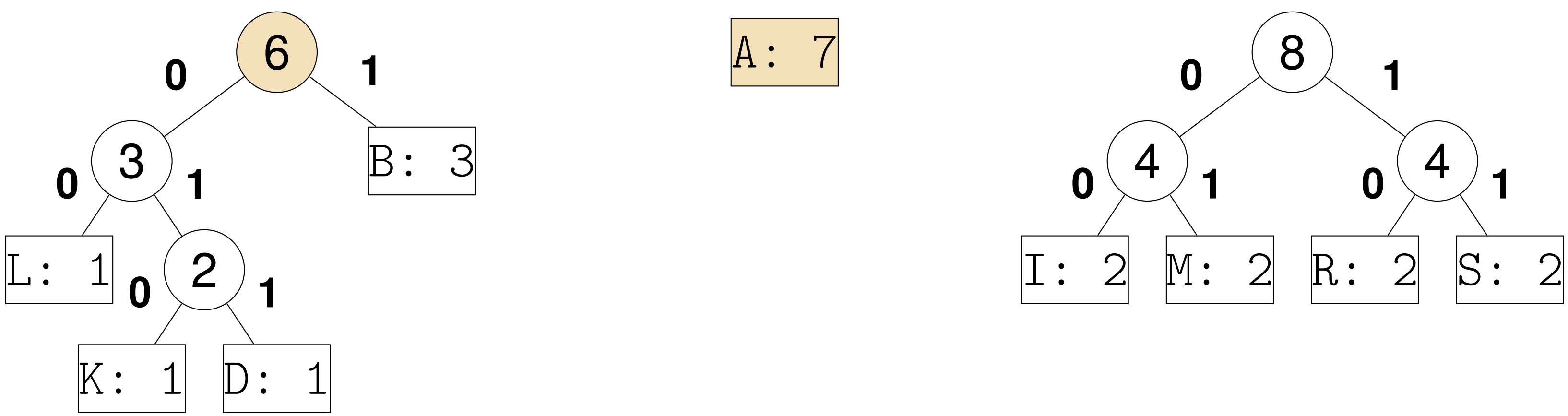
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten $\rightarrow$ Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1



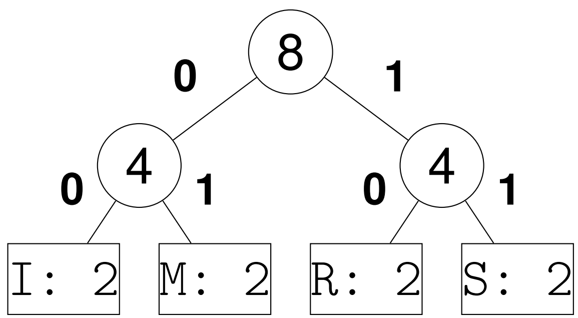
Aufgabe 5 – Huffman-Code

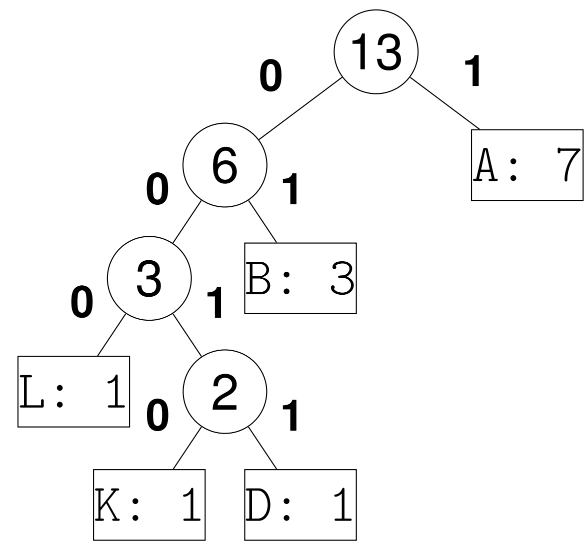
a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten $\rightarrow$ Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1





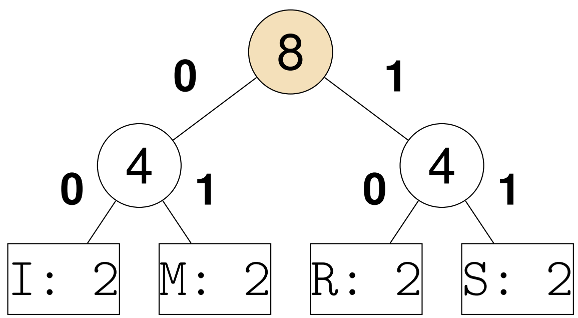
Aufgabe 5 – Huffman-Code

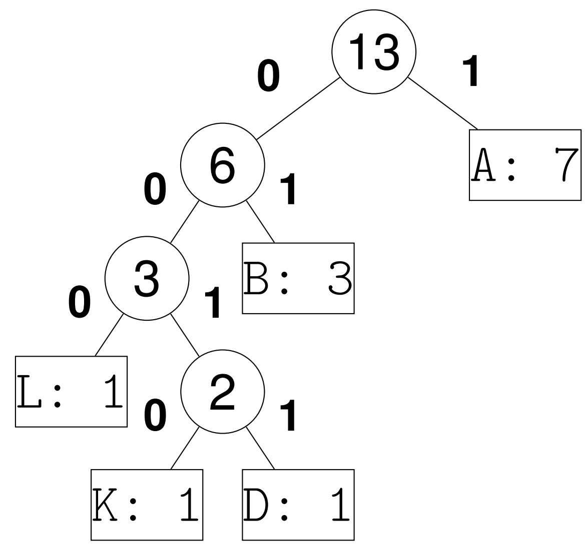
a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten $\rightarrow$ Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1





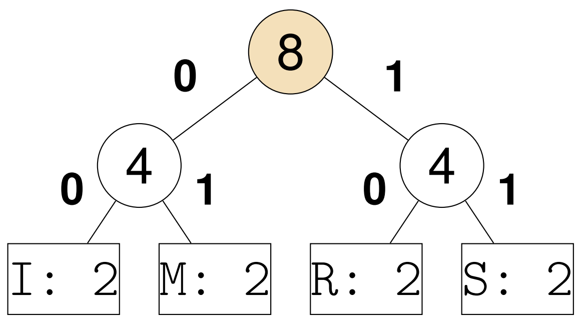
Aufgabe 5 – Huffman-Code

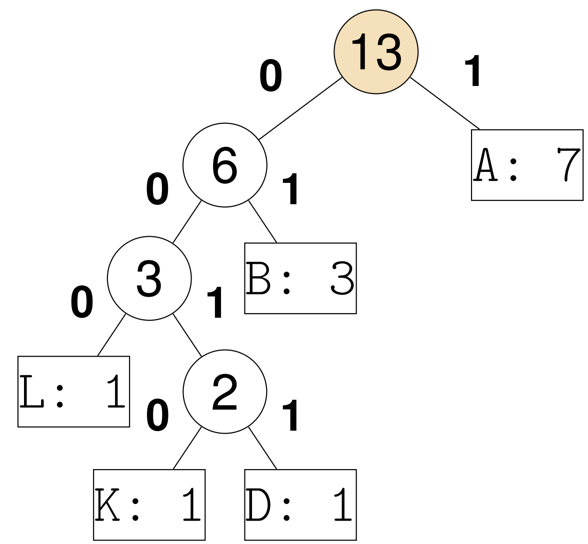
a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten → Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1





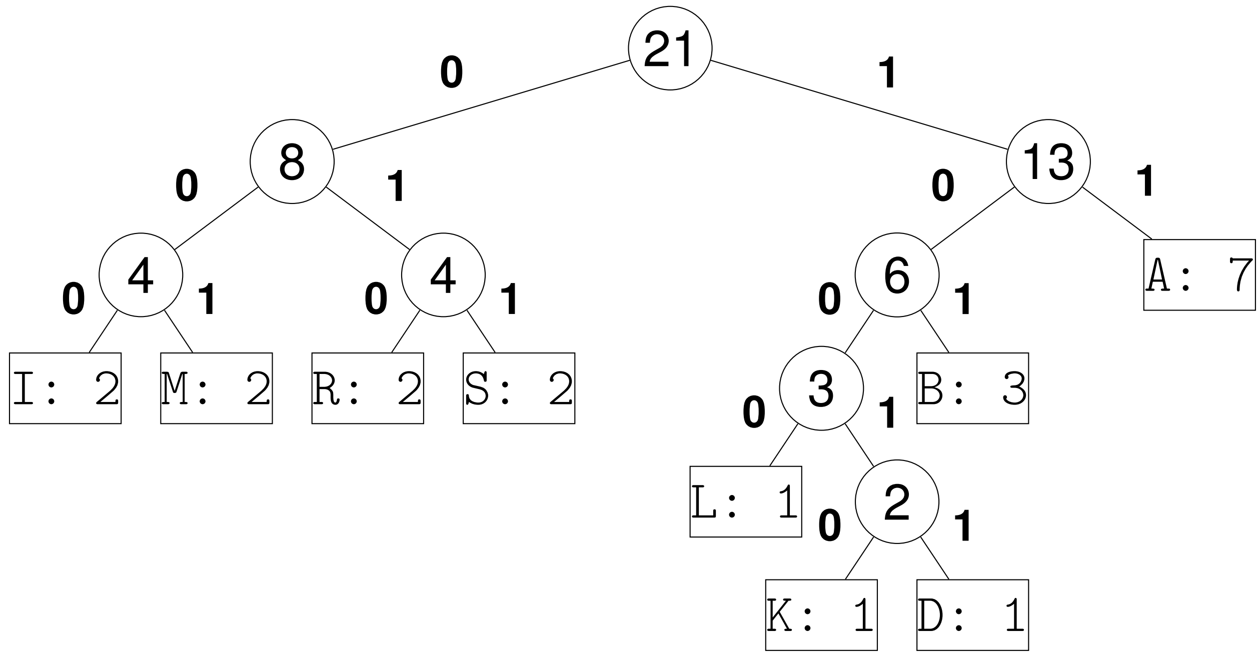
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Häufigkeiten → Huffman-Kodierungs-Baum:

Buchstabe	A	B	R	K	D	S	I	M	L
Anzahl	7	3	2	1	1	2	2	2	1



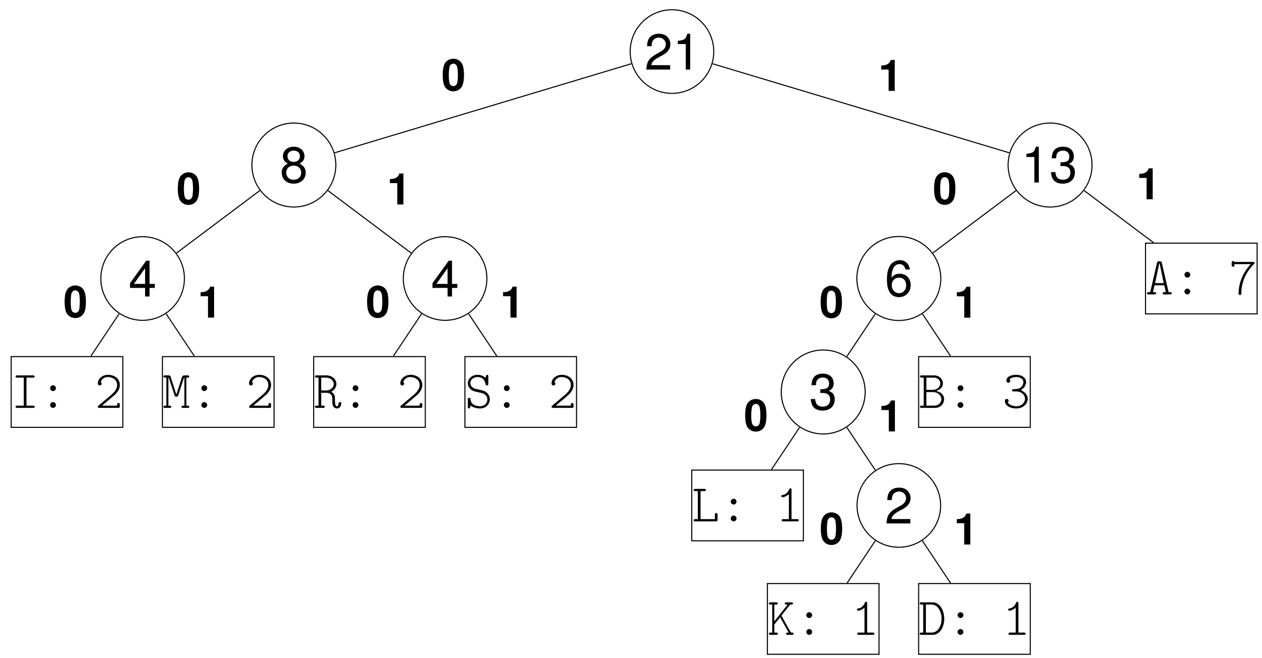
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung:

Zeichen	Code



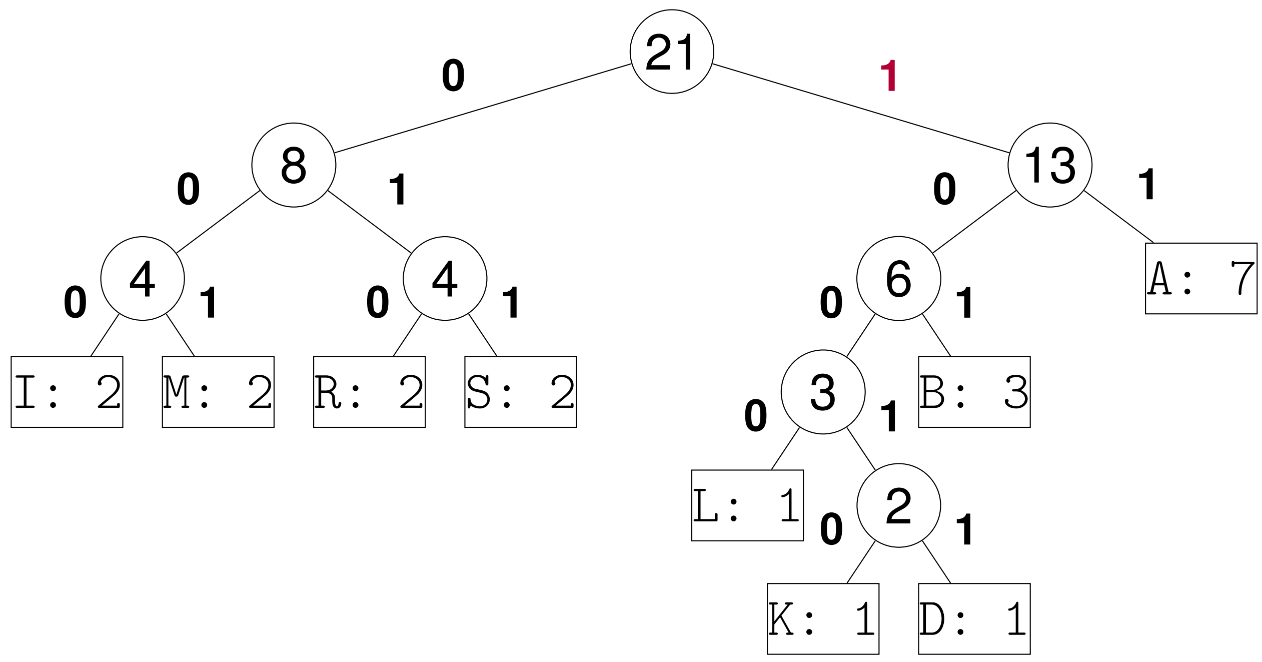
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	



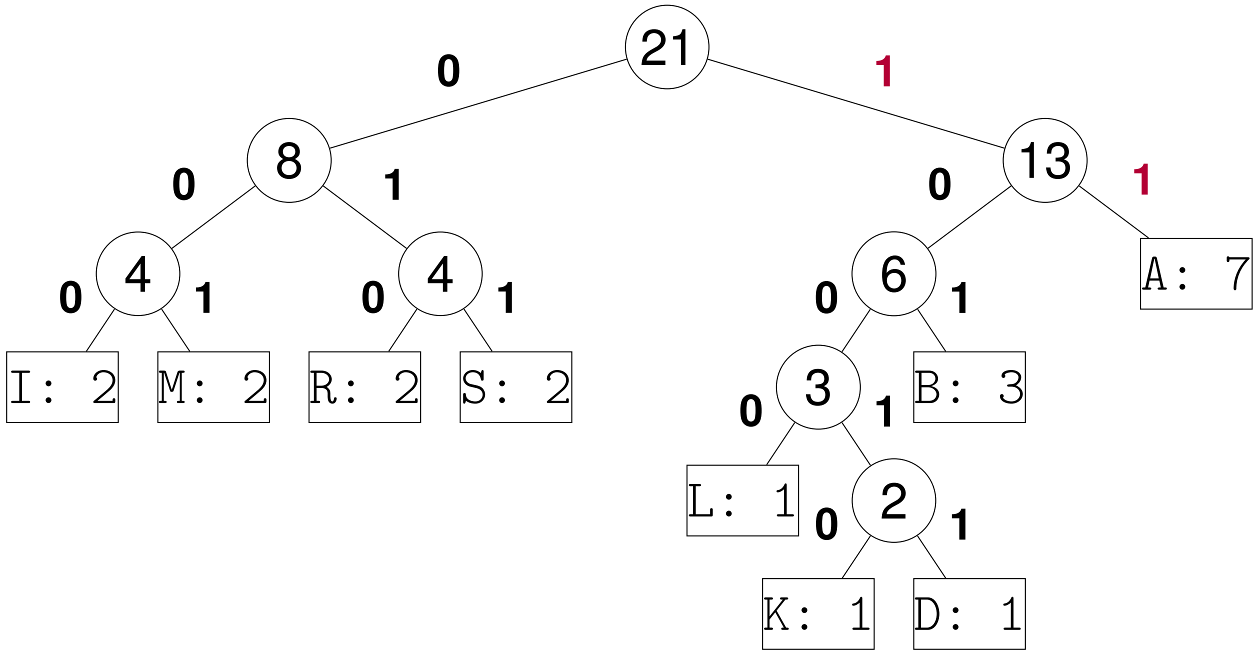
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	



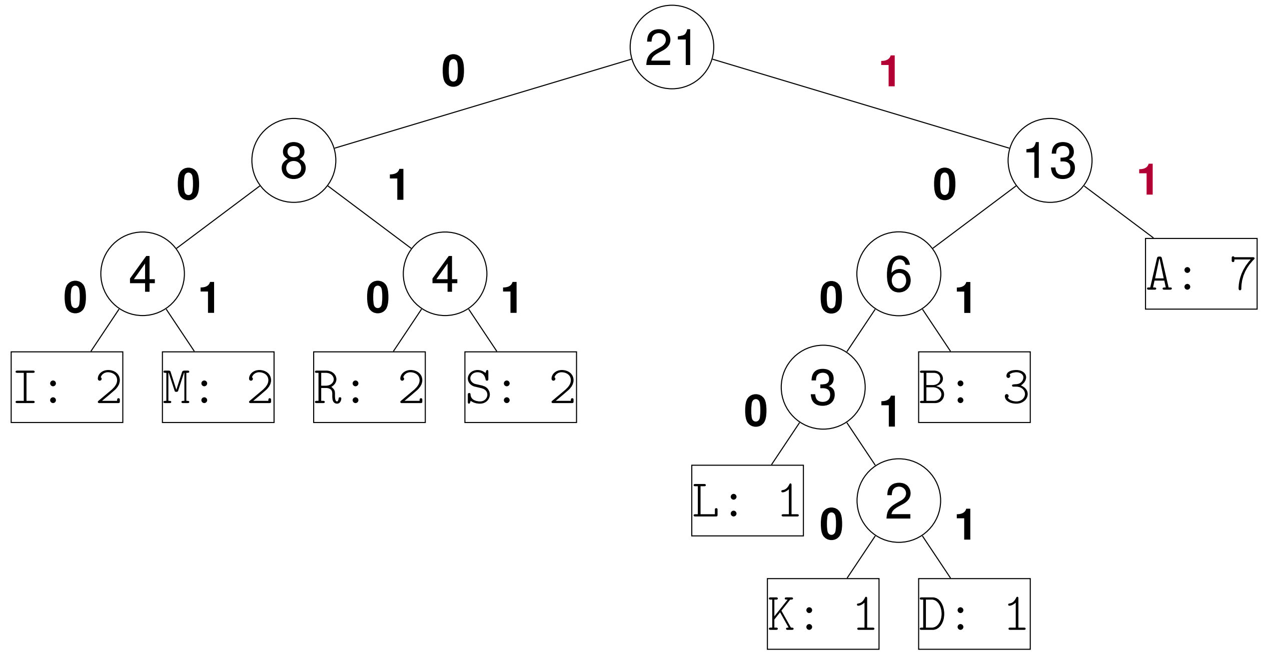
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11



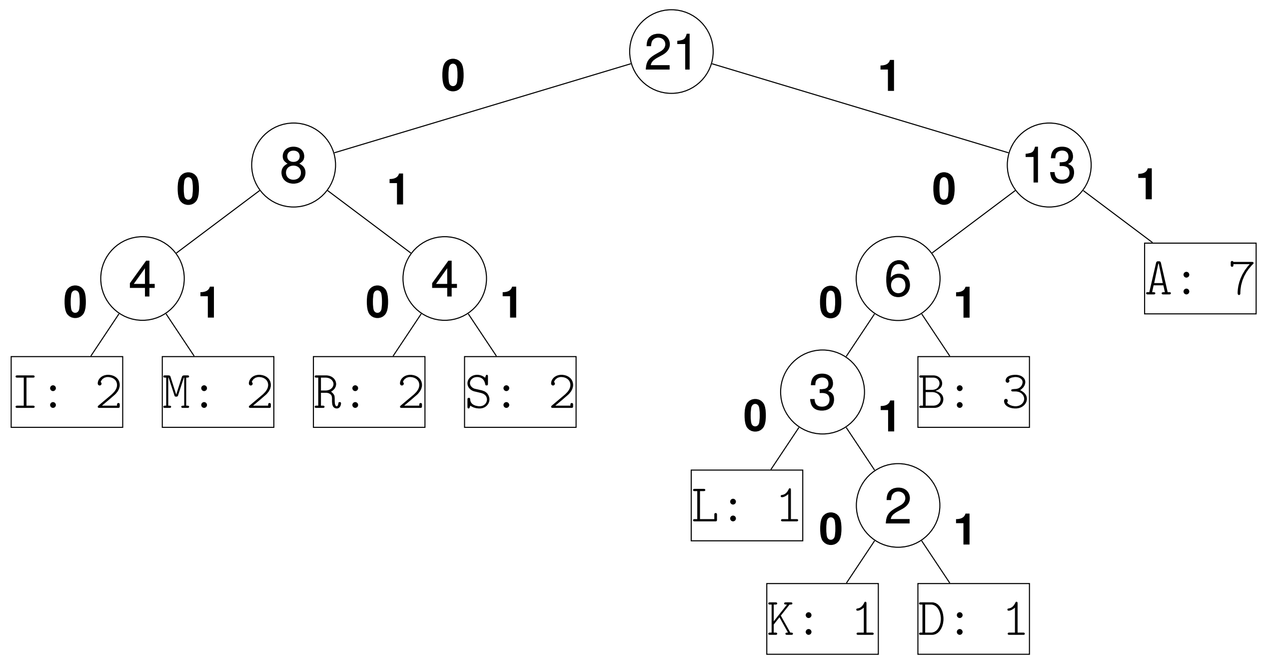
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	



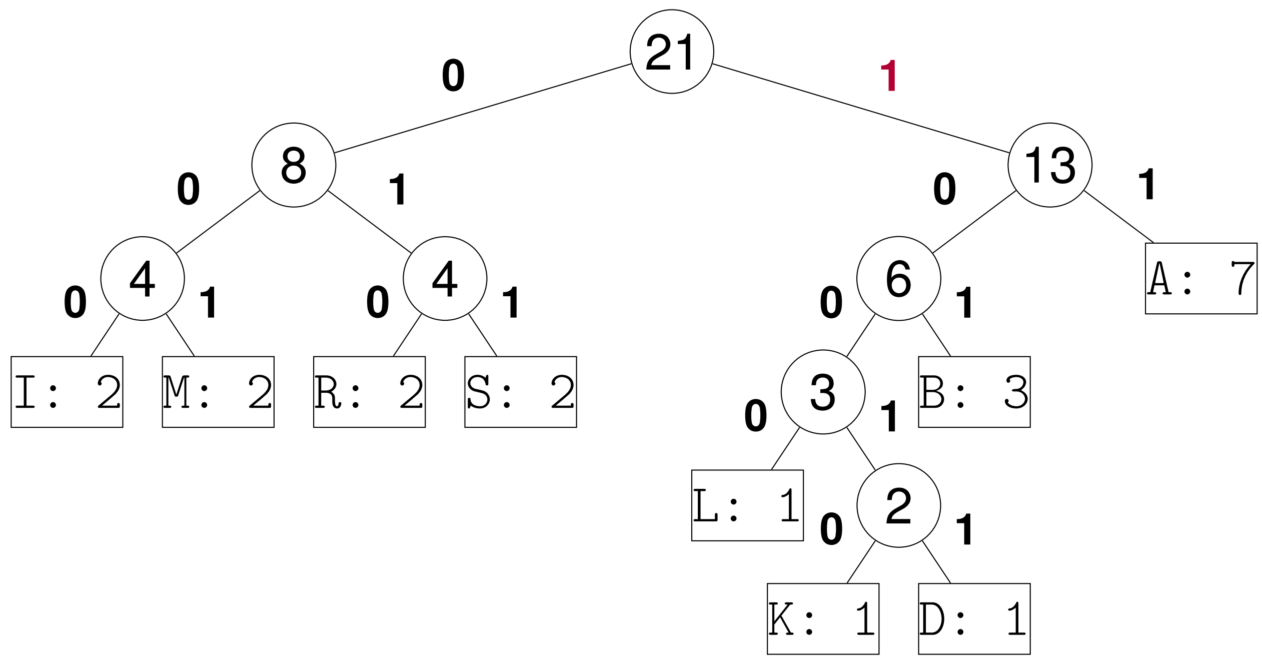
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	



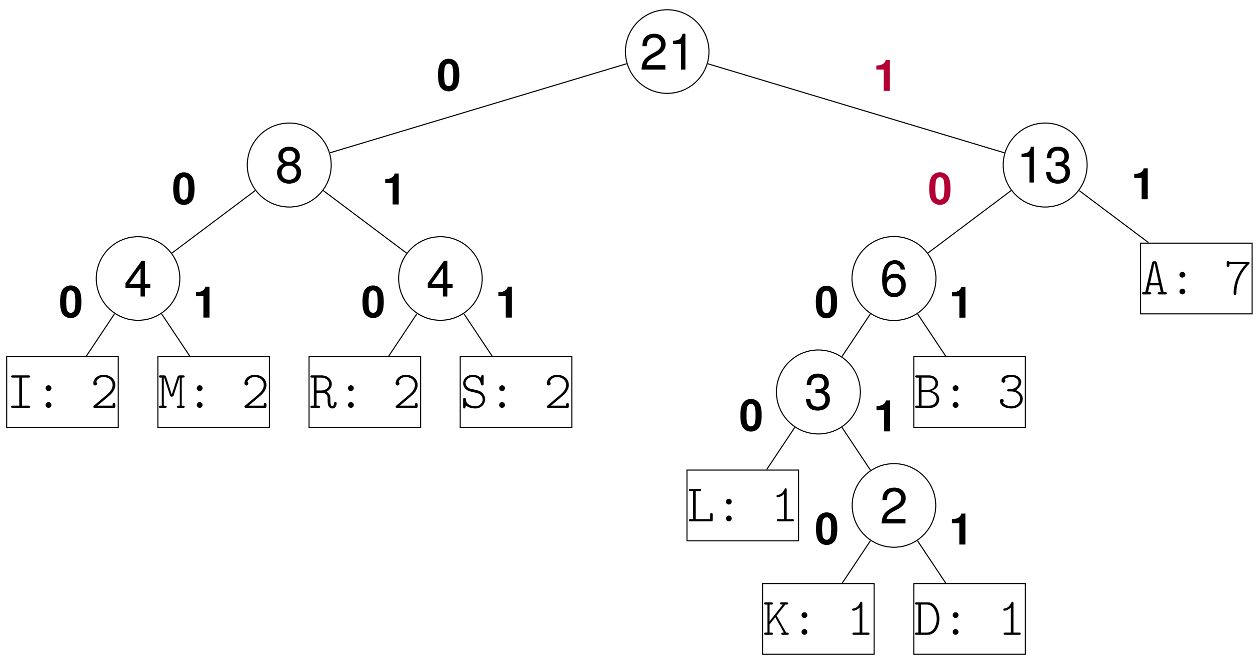
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	



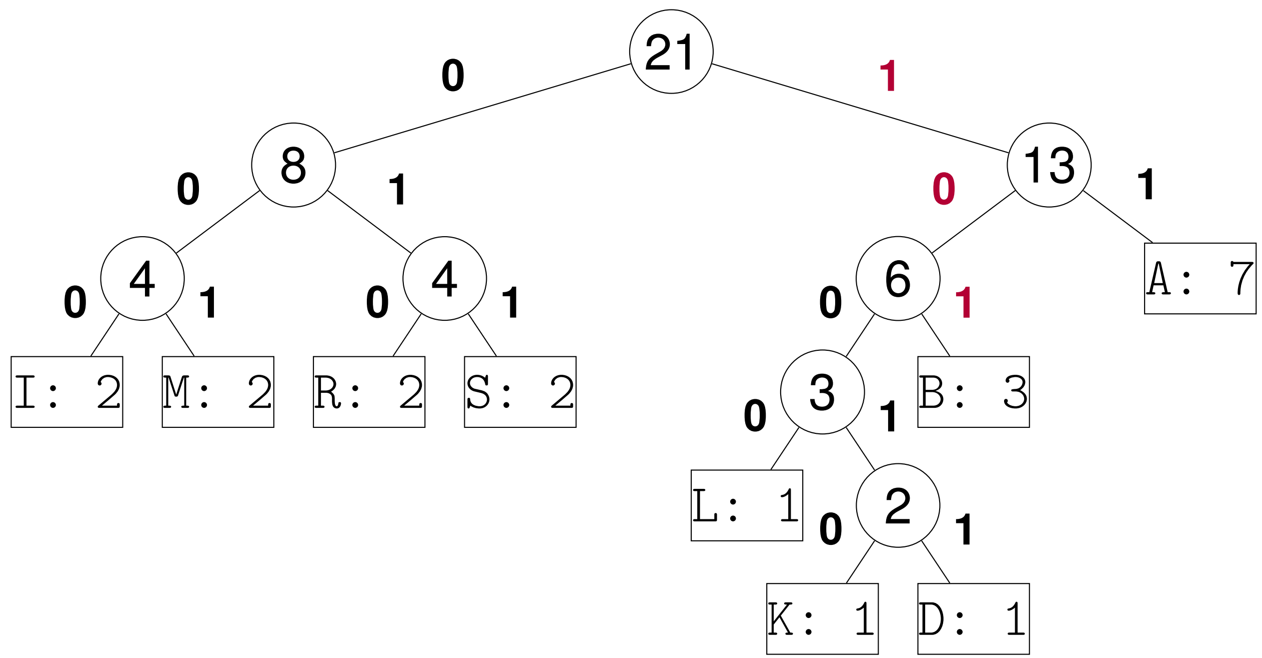
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	



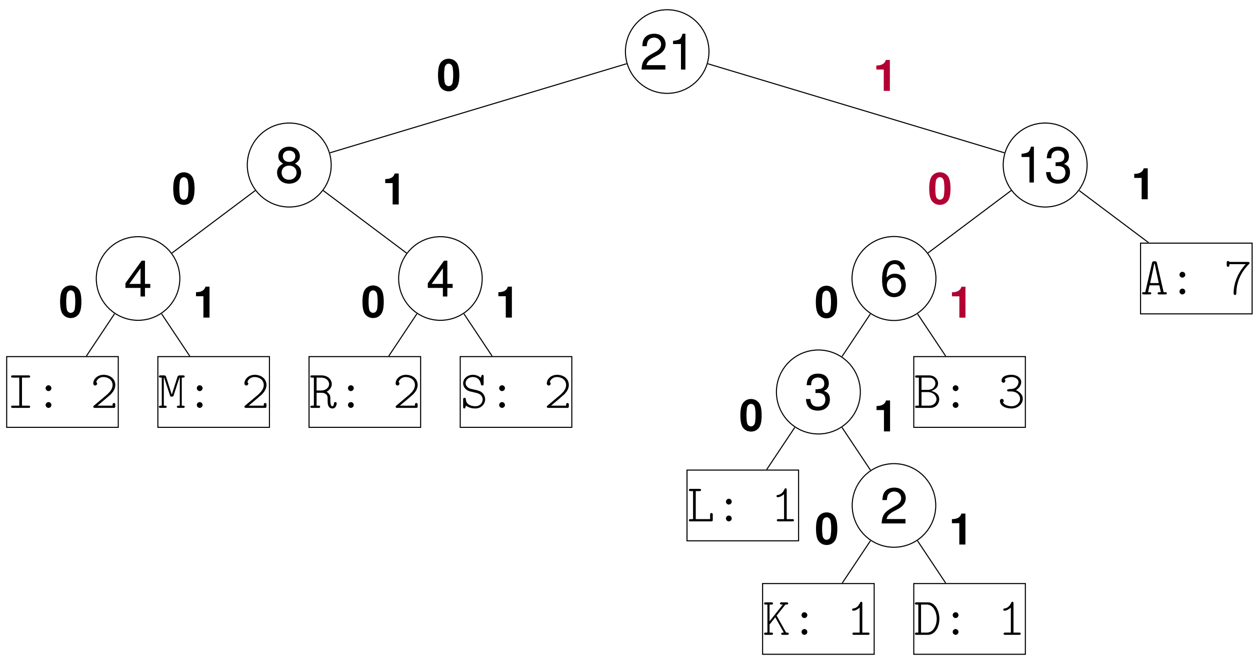
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101



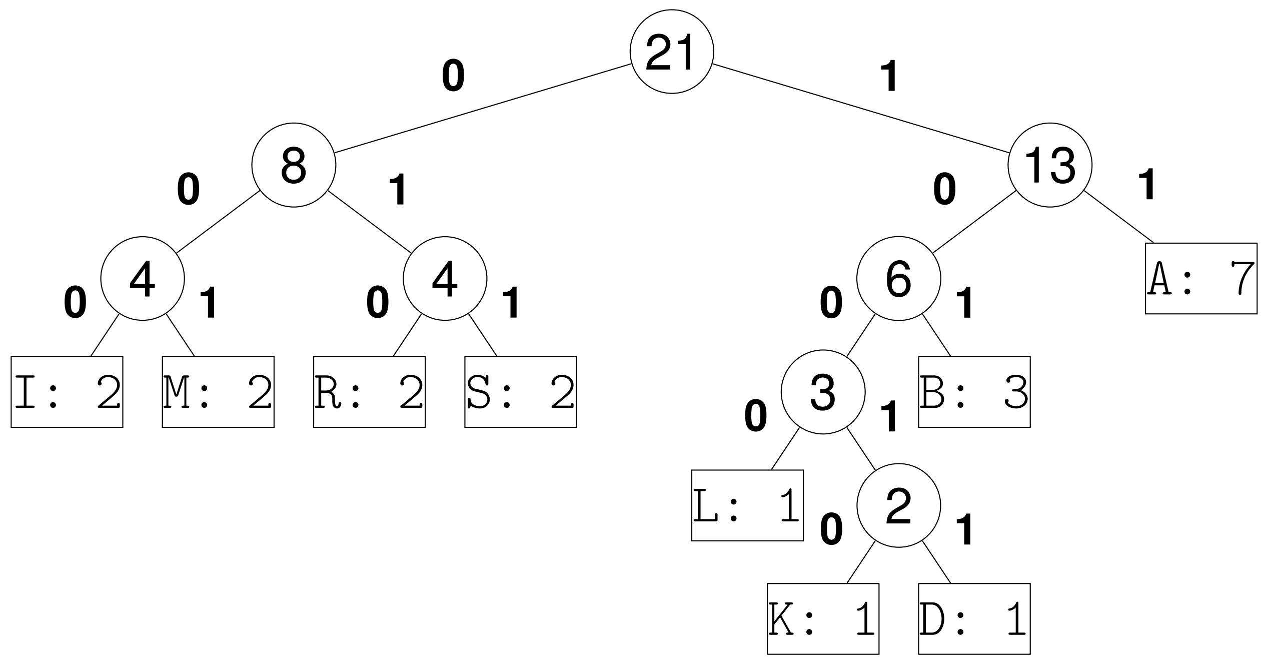
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	



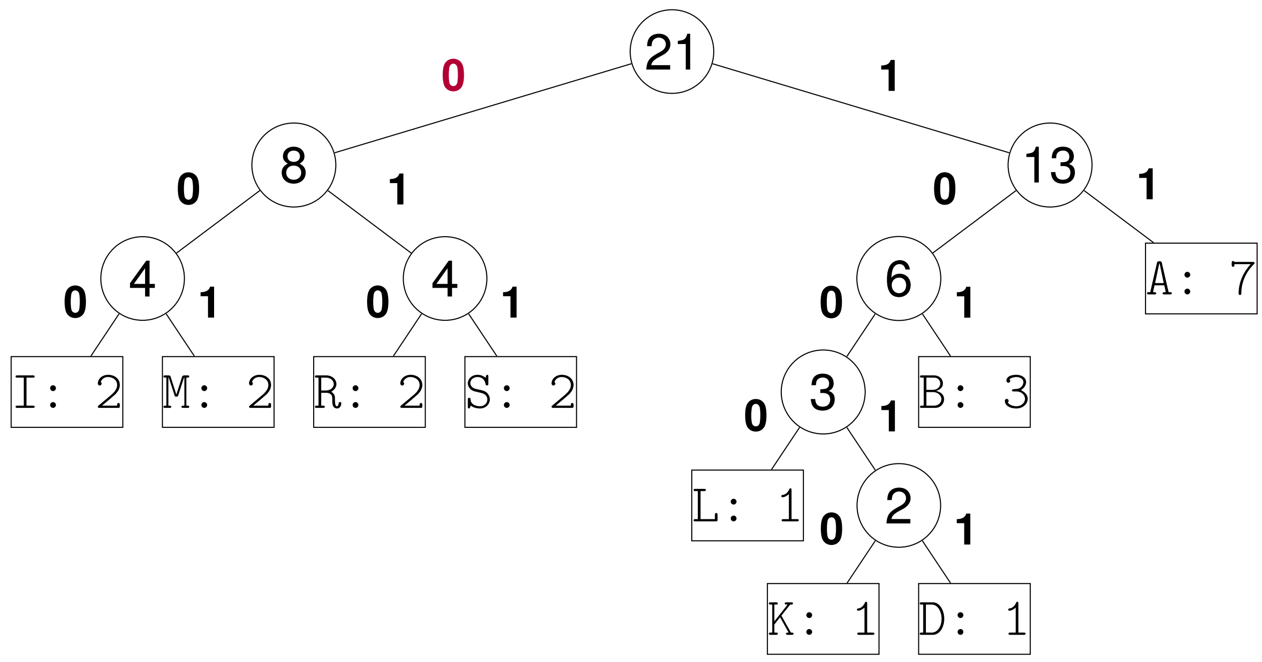
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	



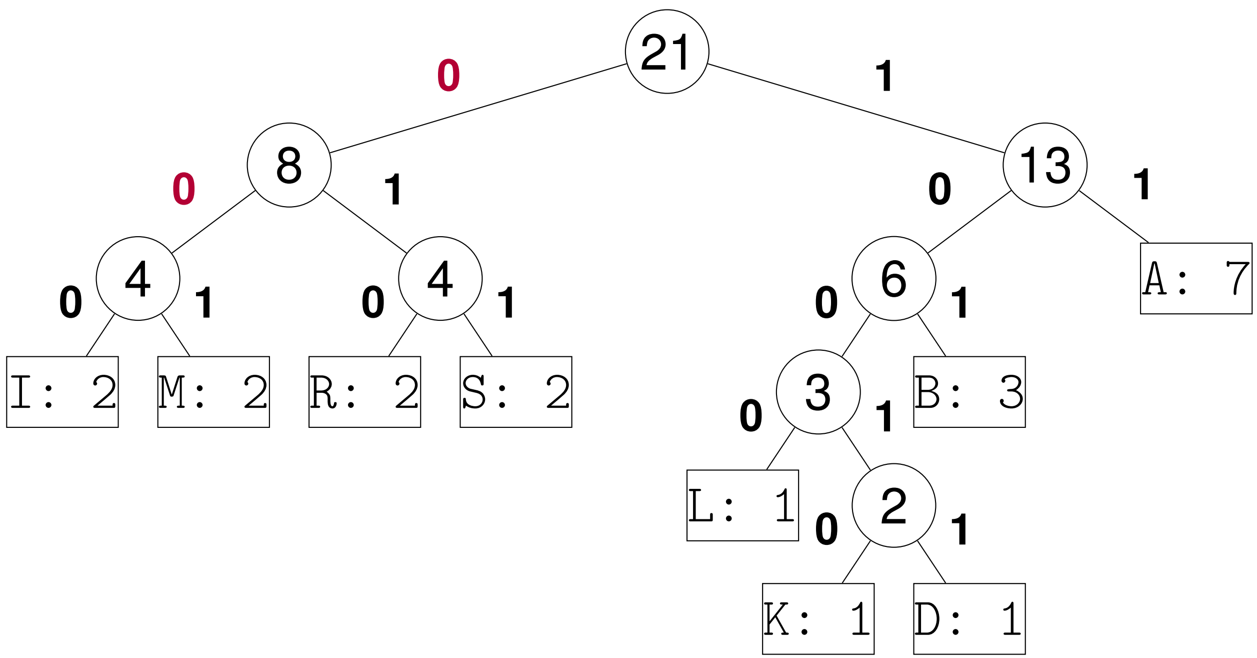
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	



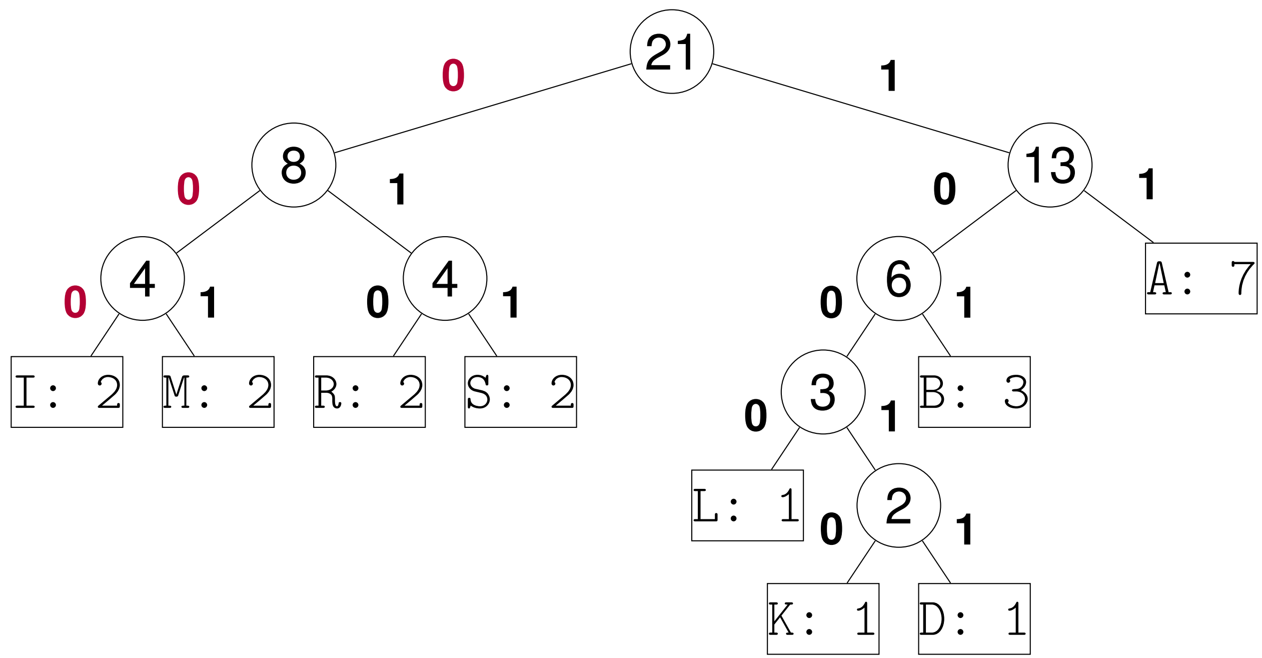
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	



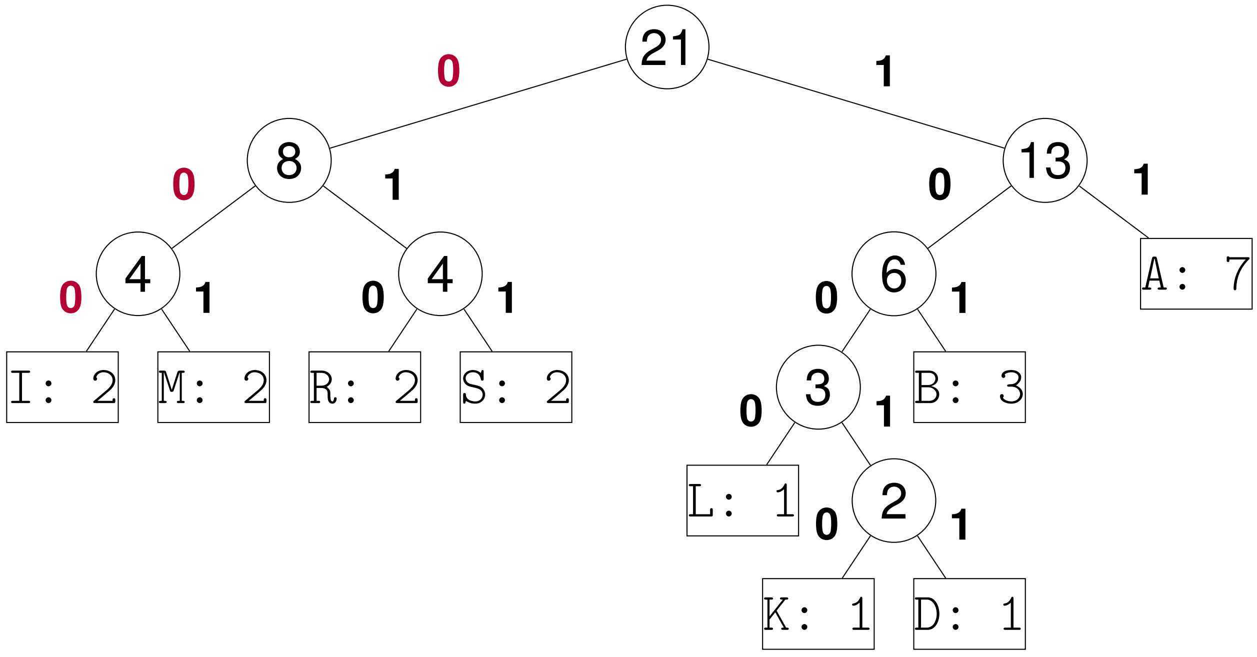
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000



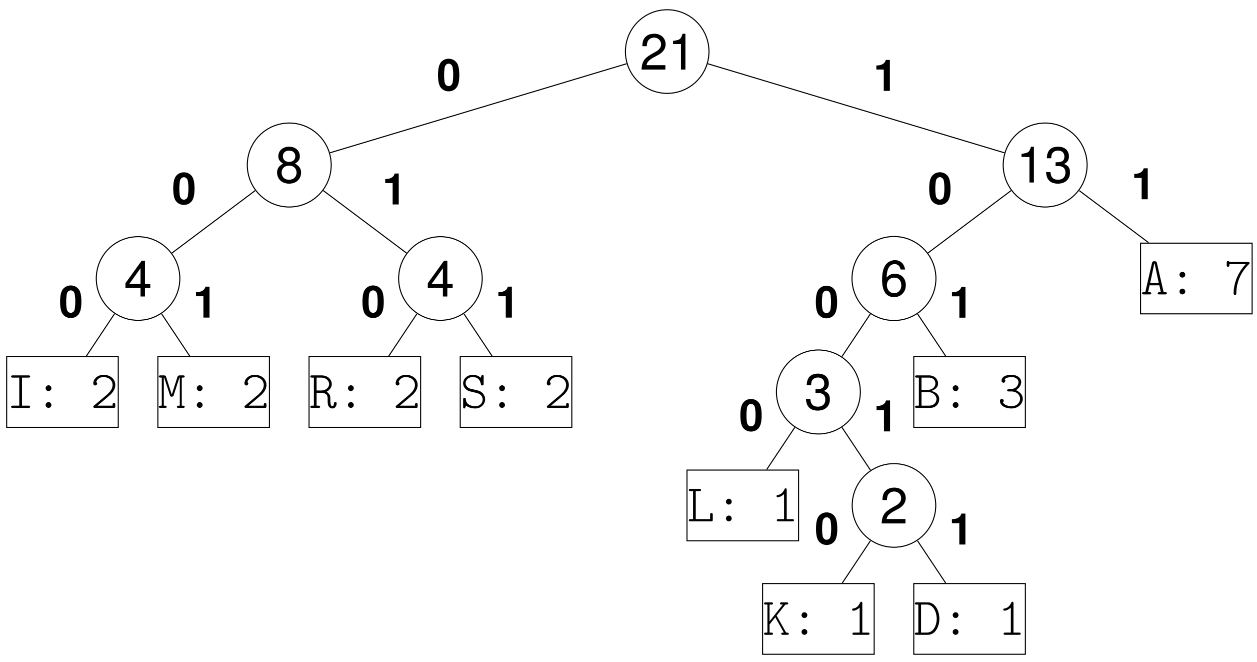
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	



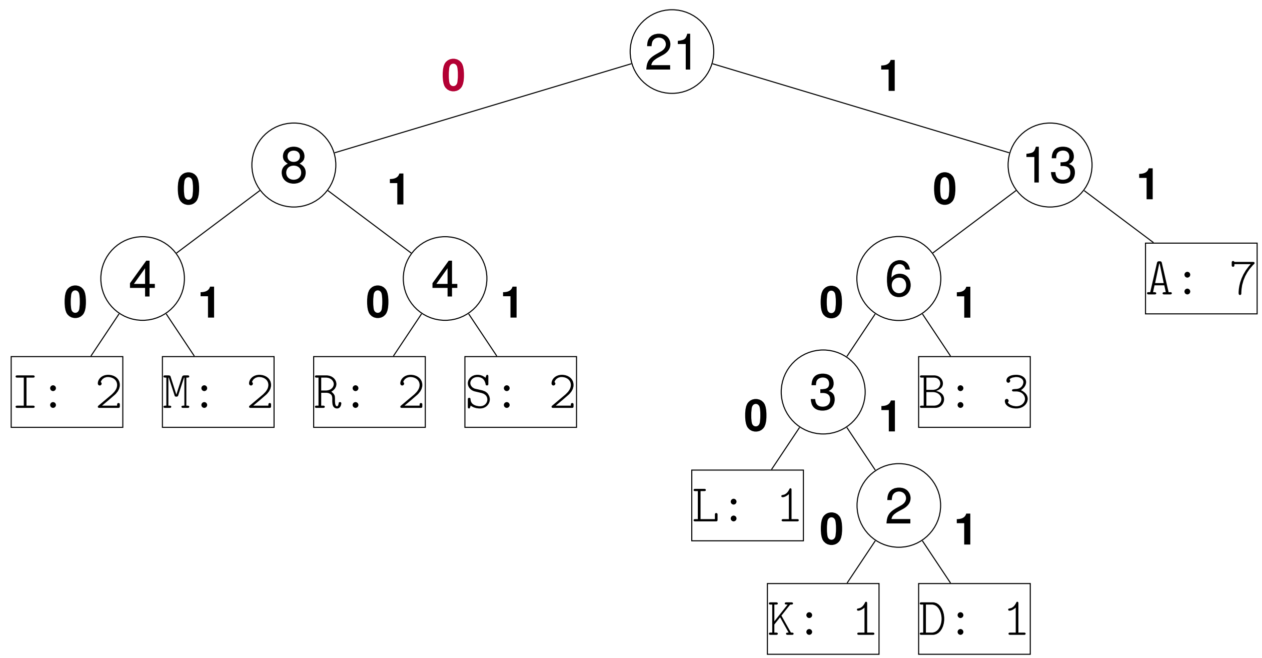
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	



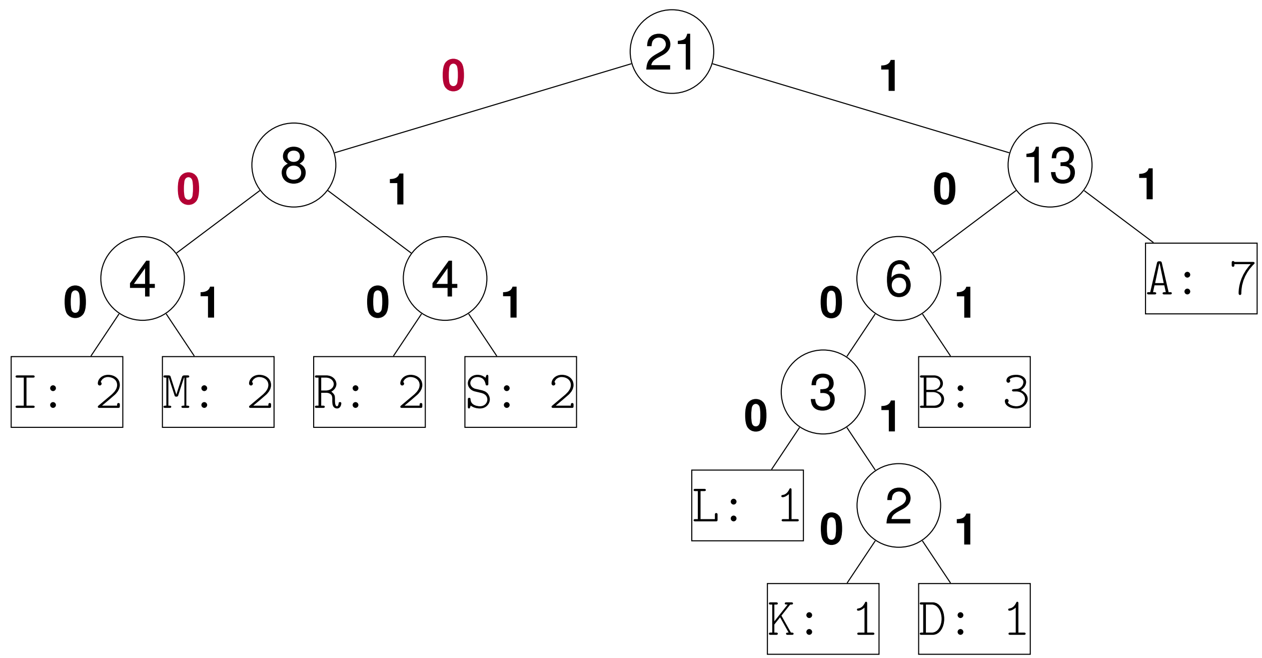
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	



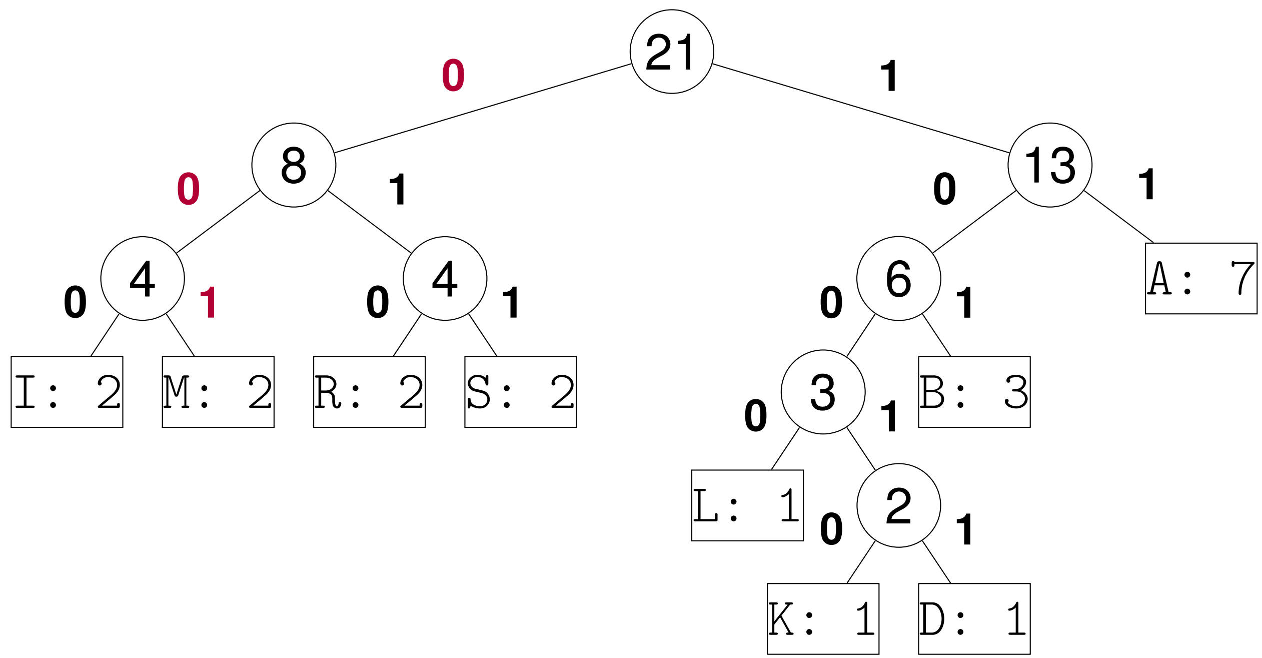
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	



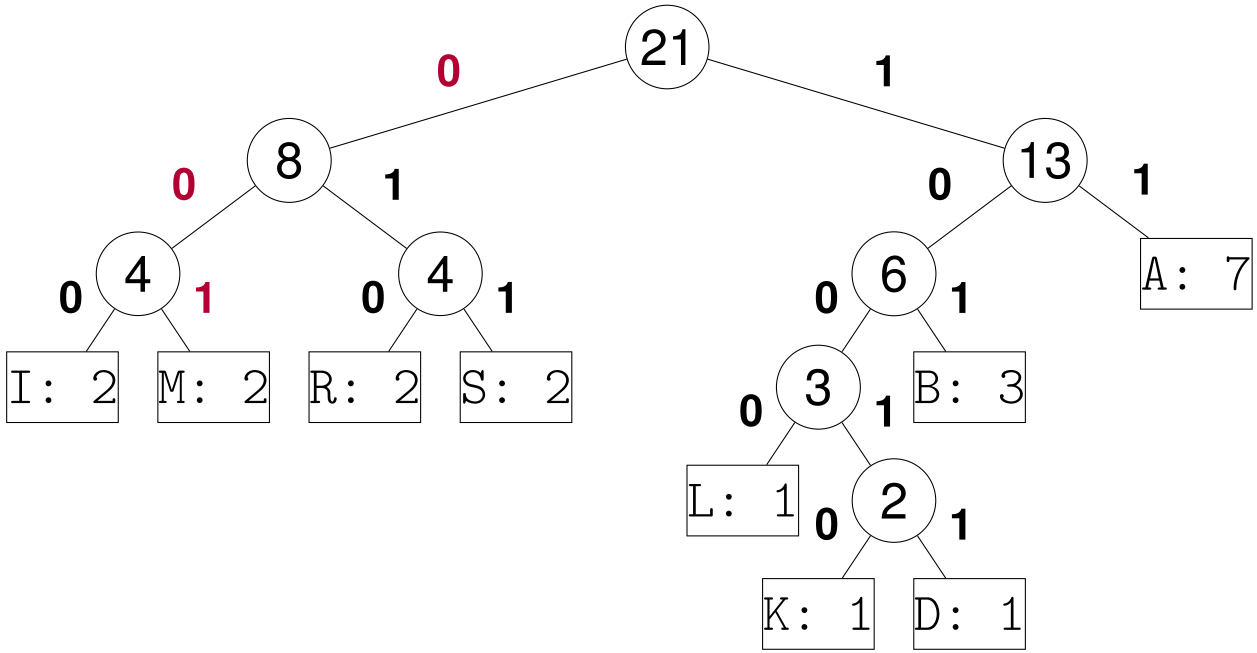
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001



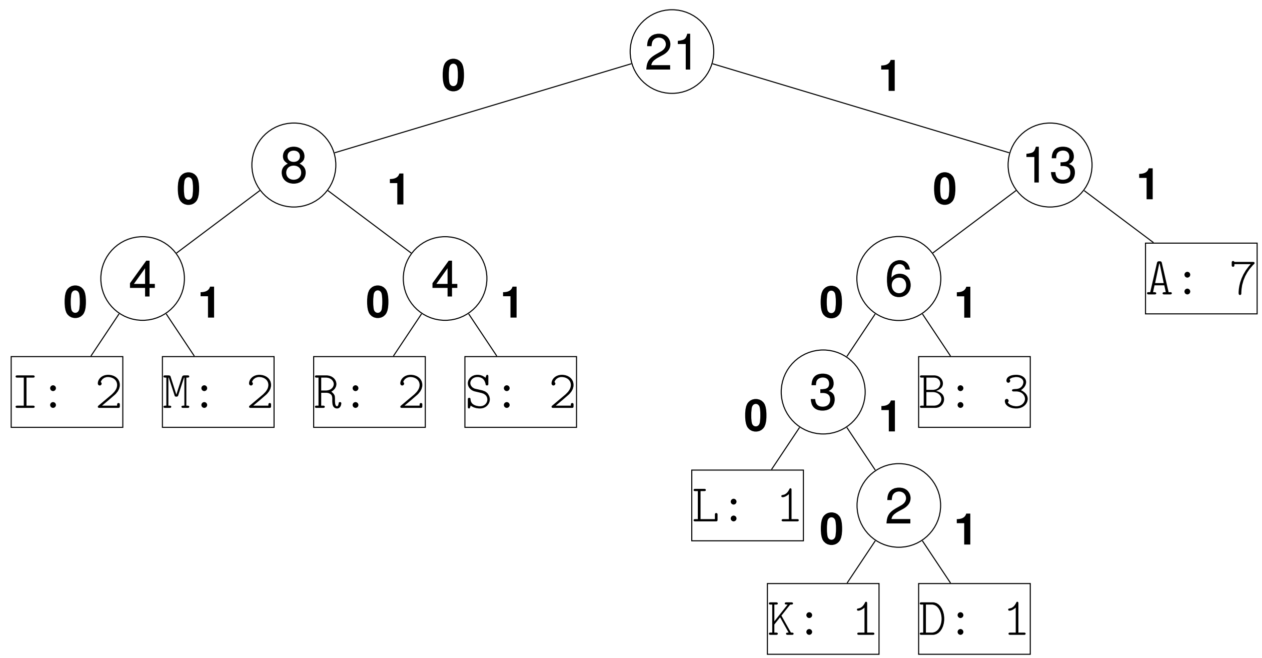
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	



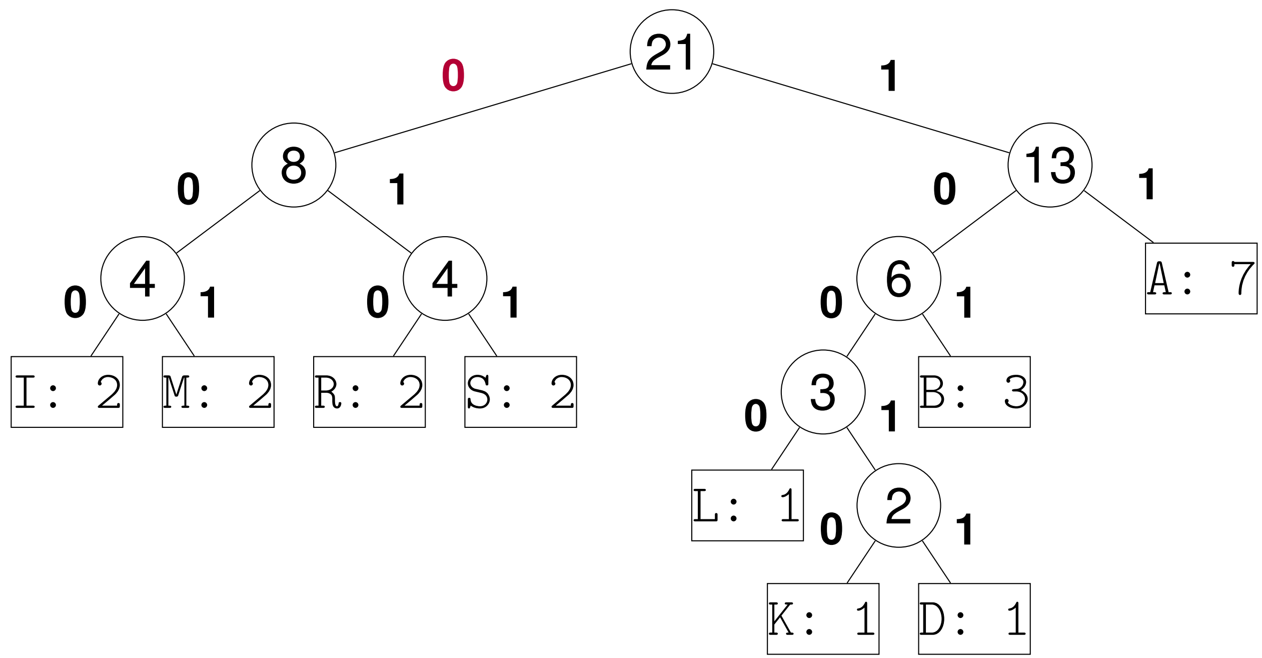
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	



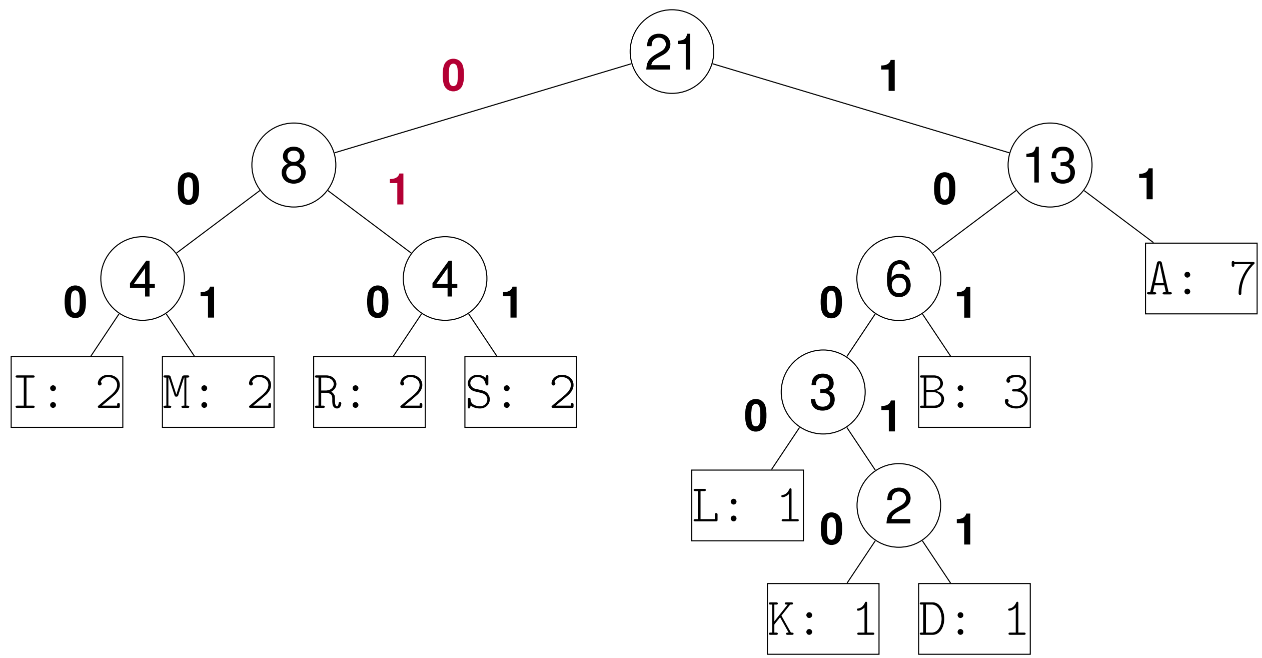
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	



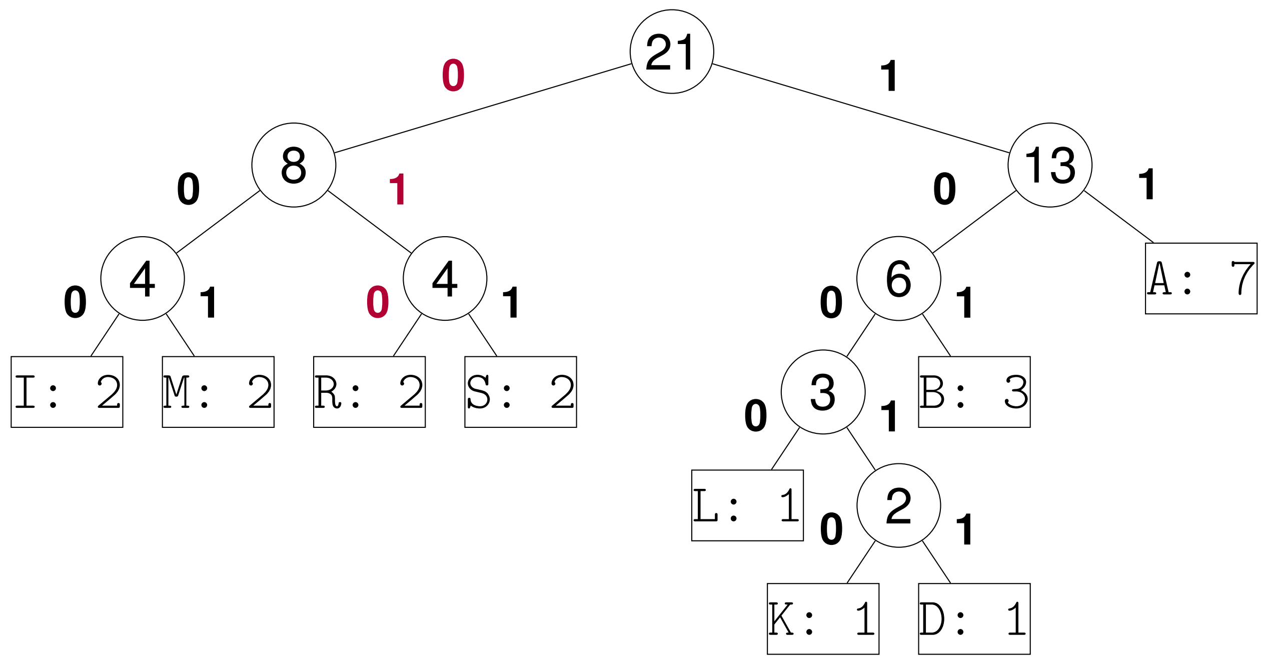
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	



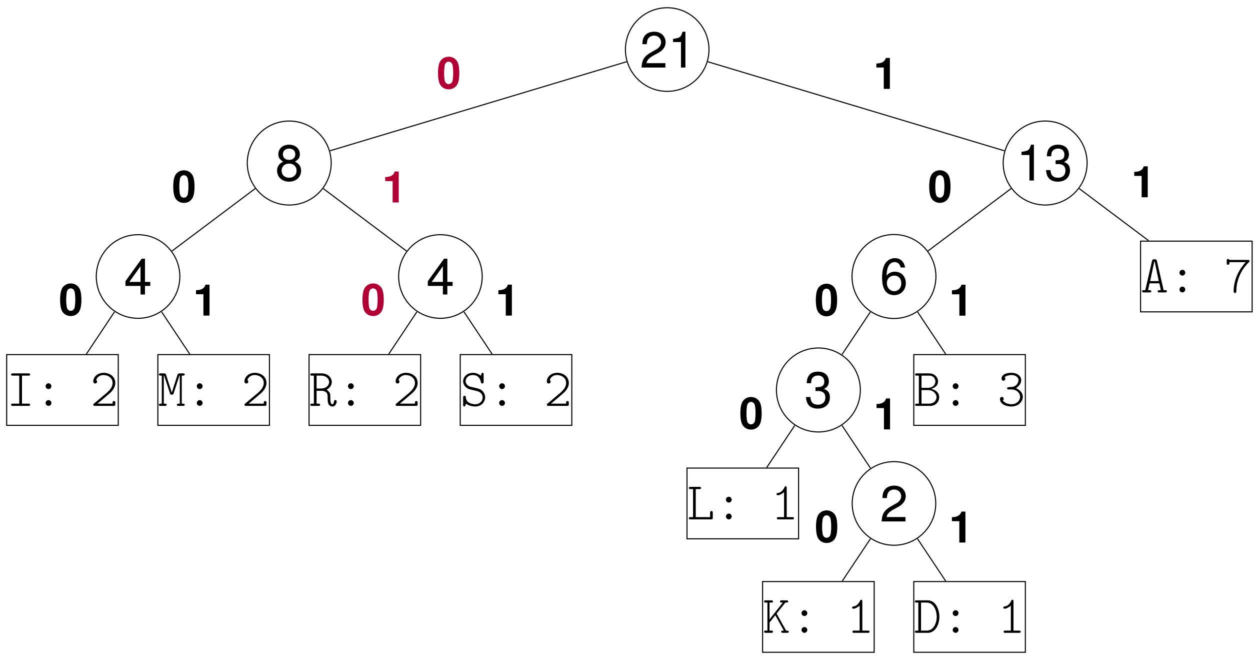
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010



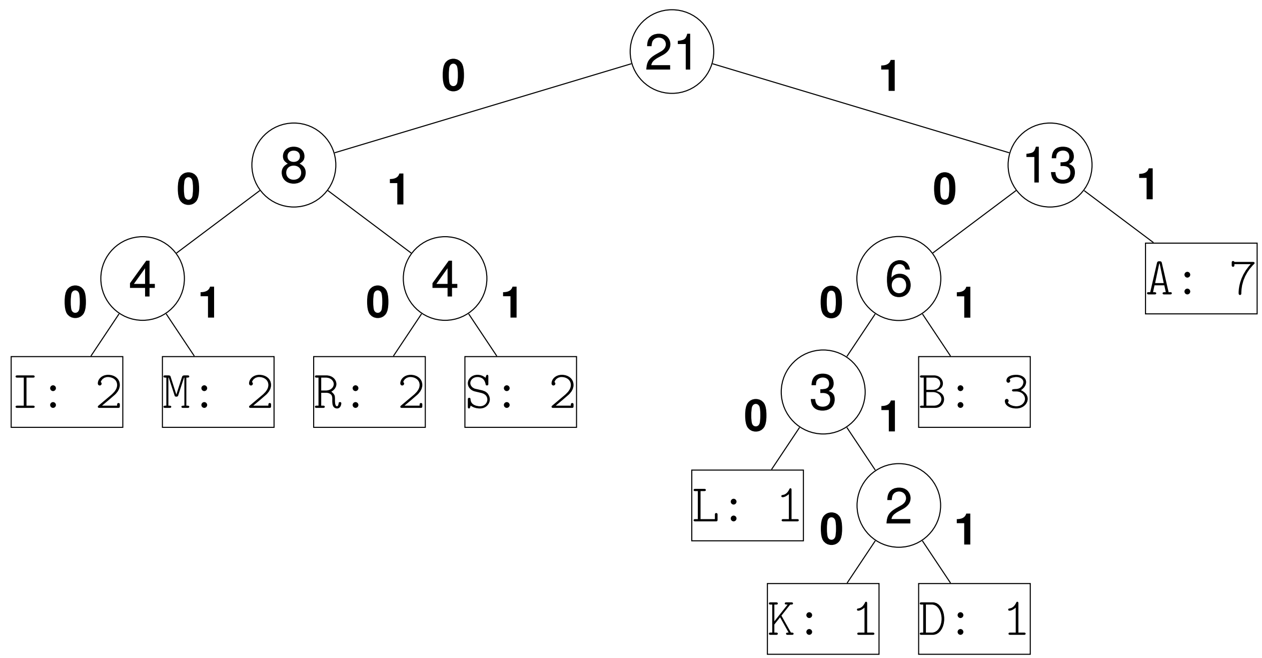
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	



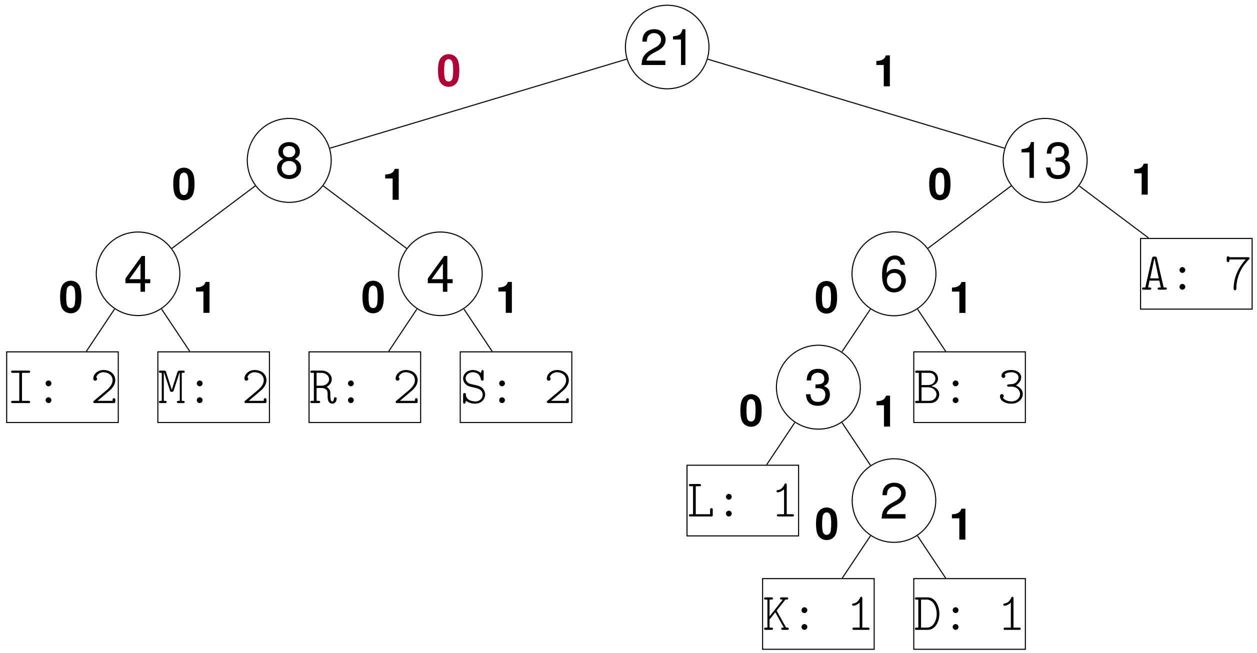
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	



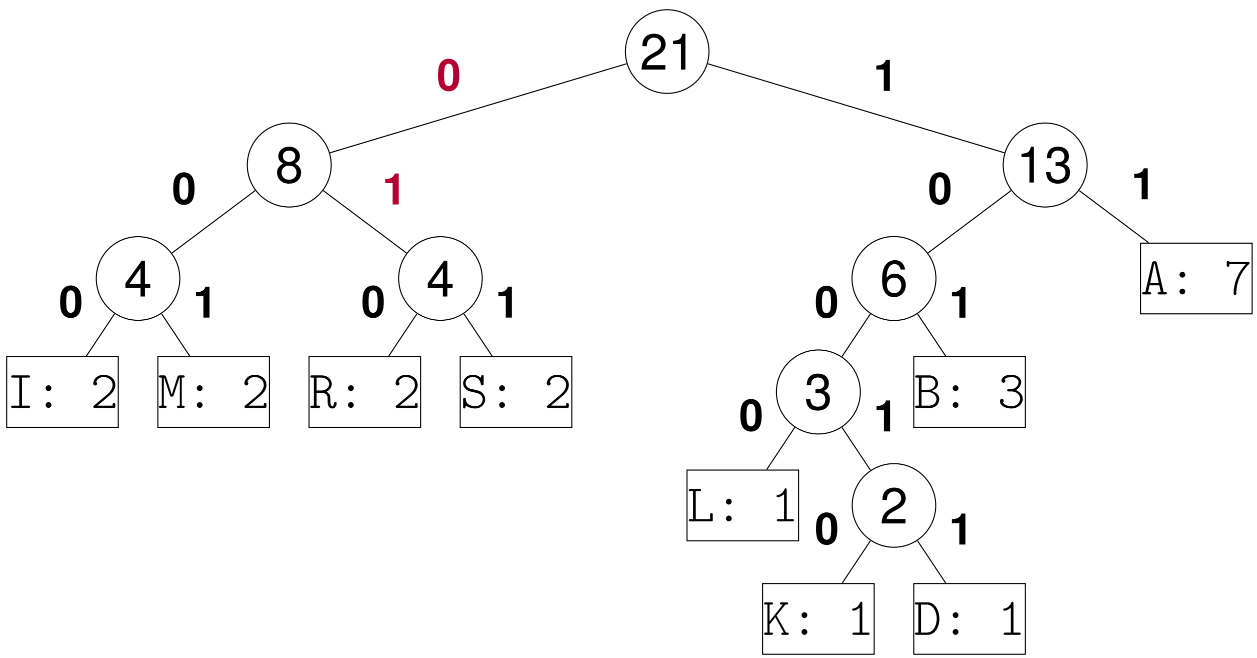
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	



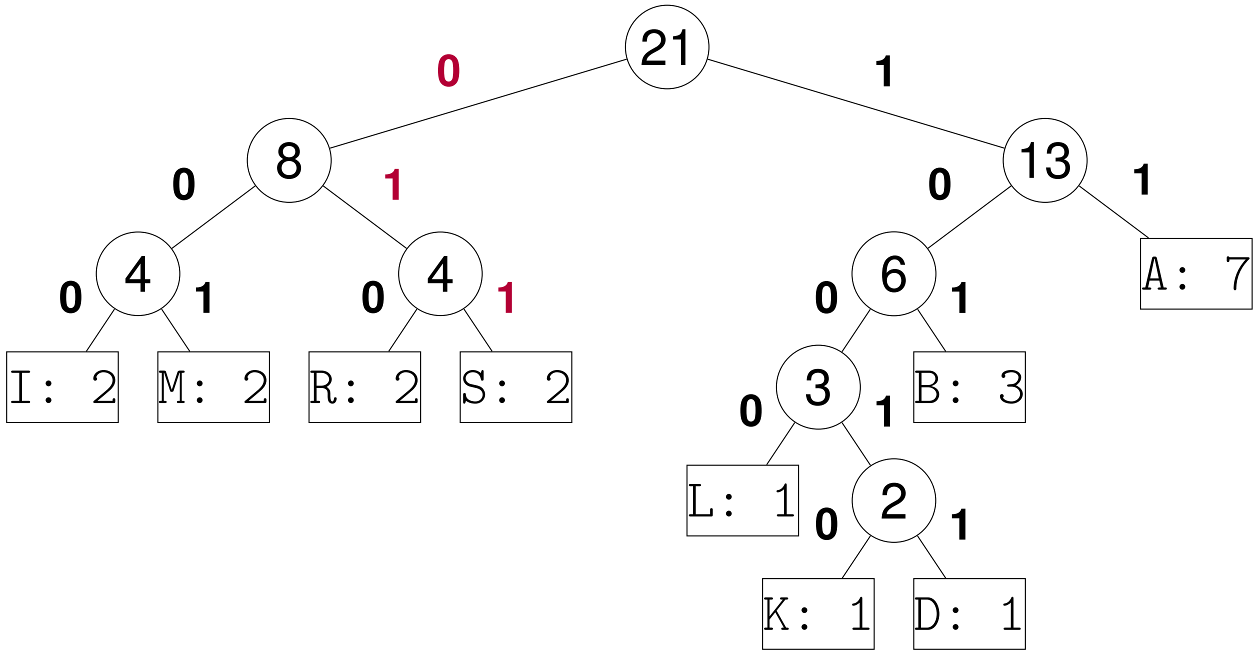
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	



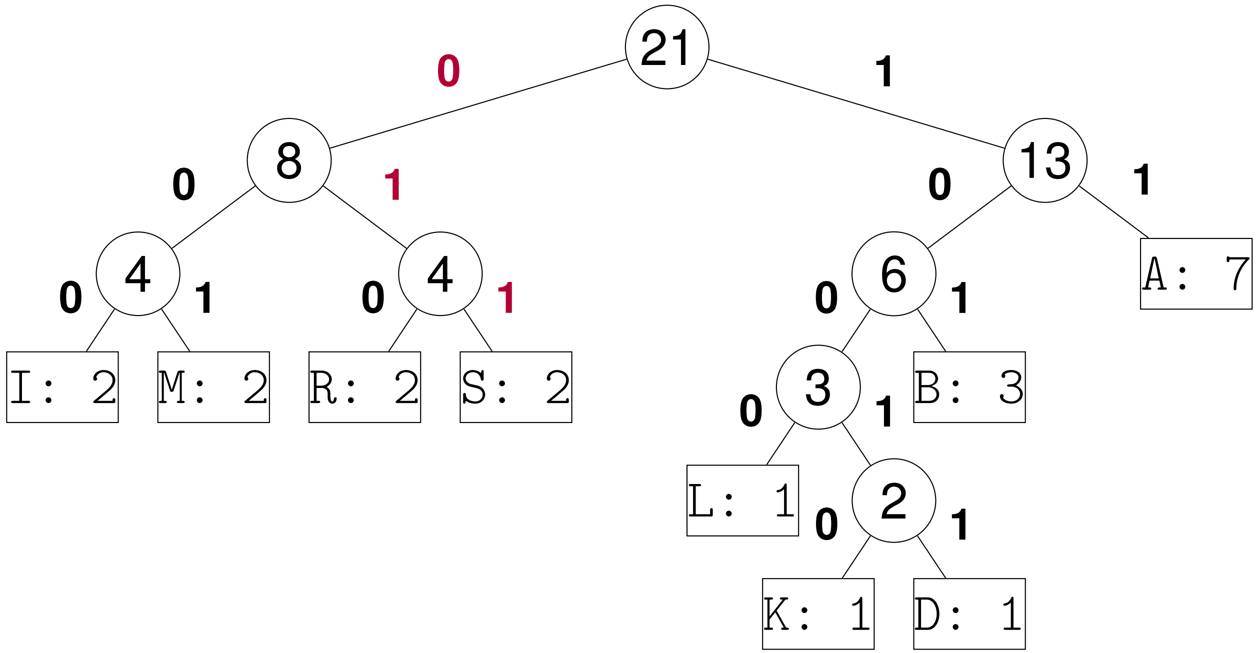
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	011



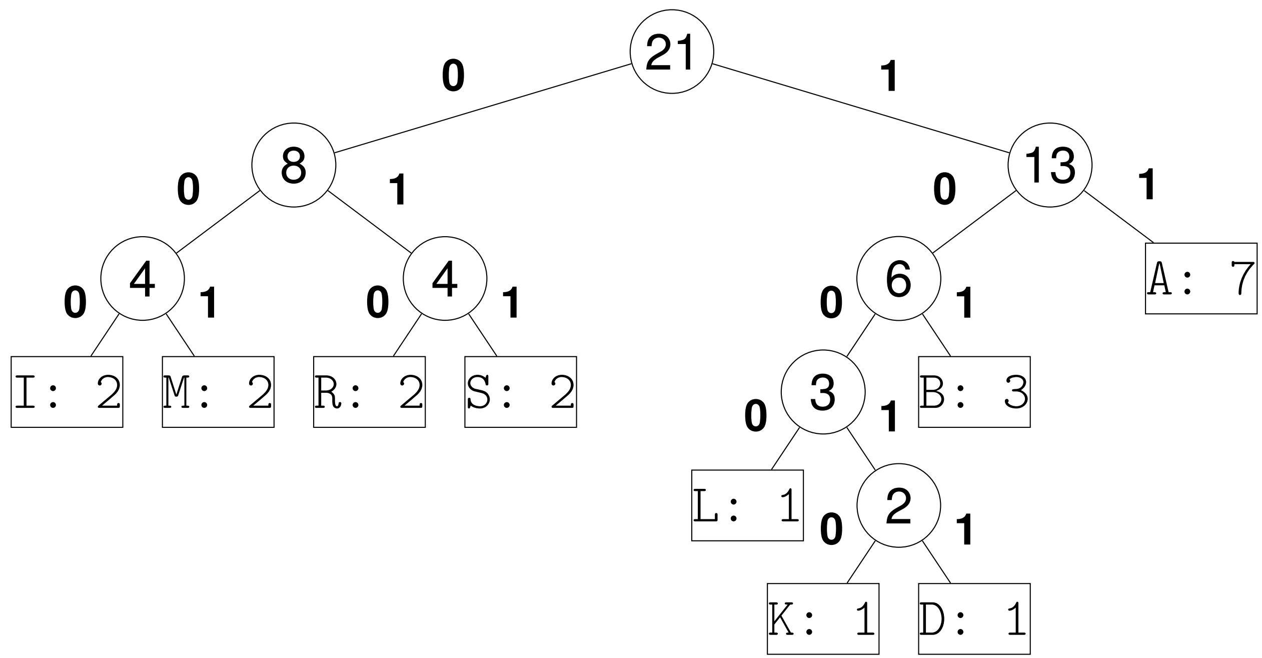
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	011
L	



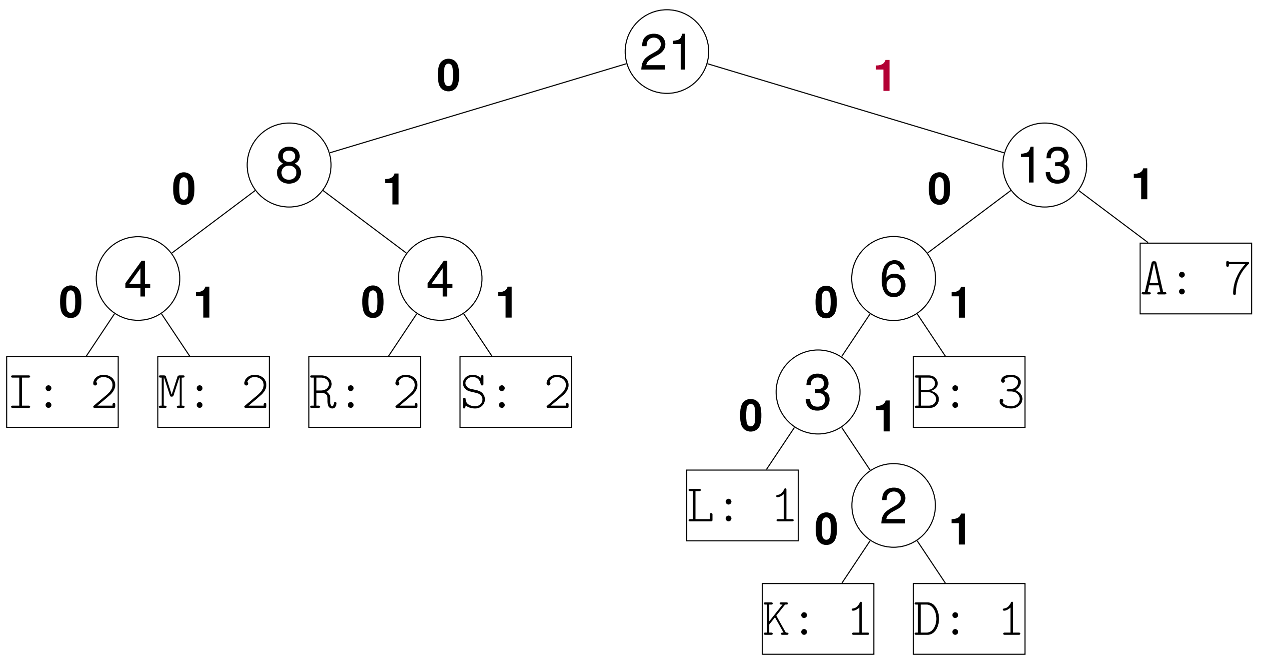
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	011
L	



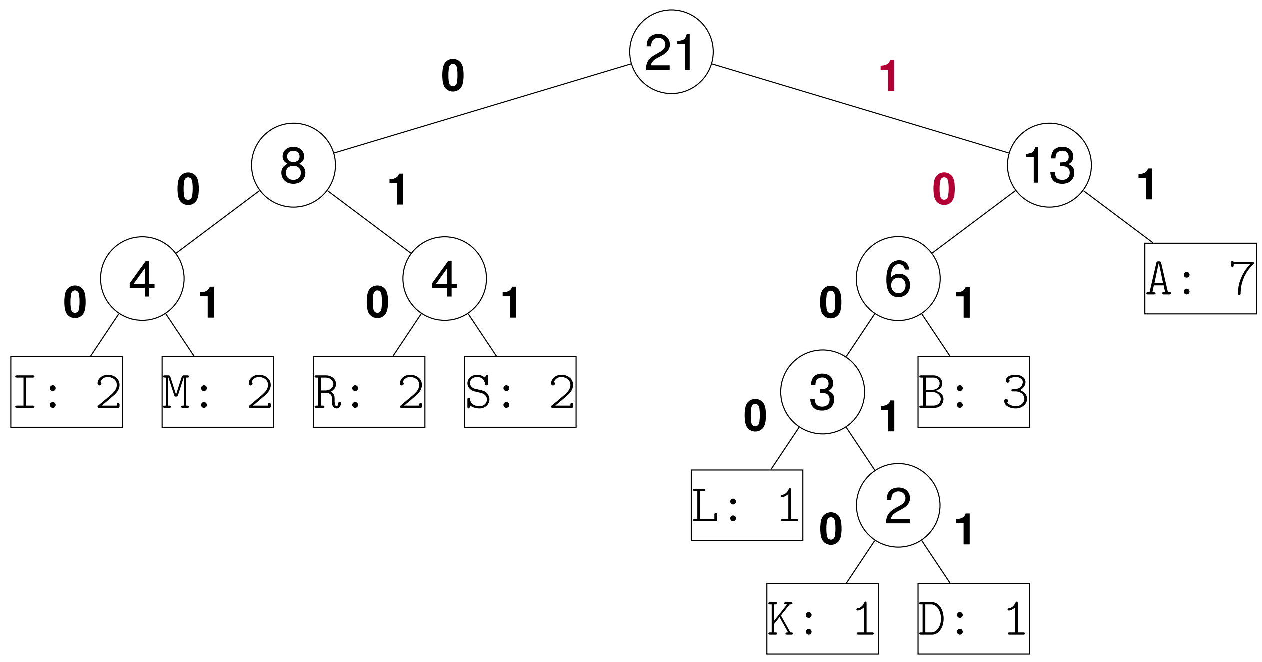
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	011
L	



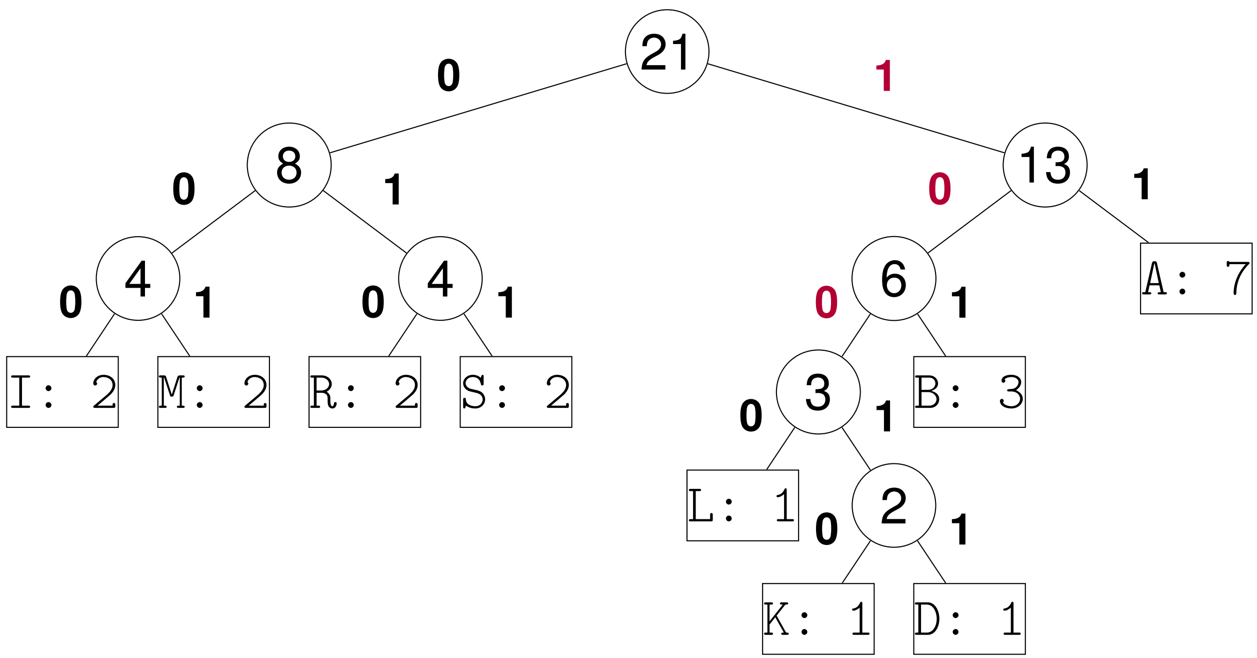
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	011
L	



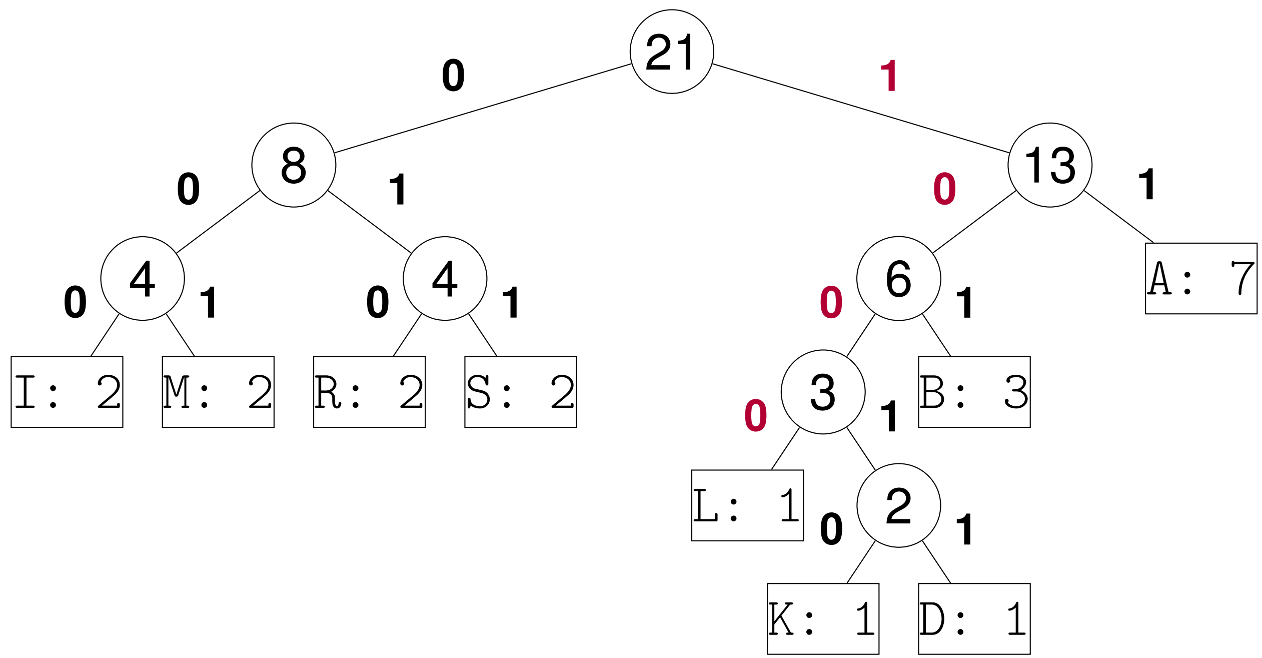
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	011
L	



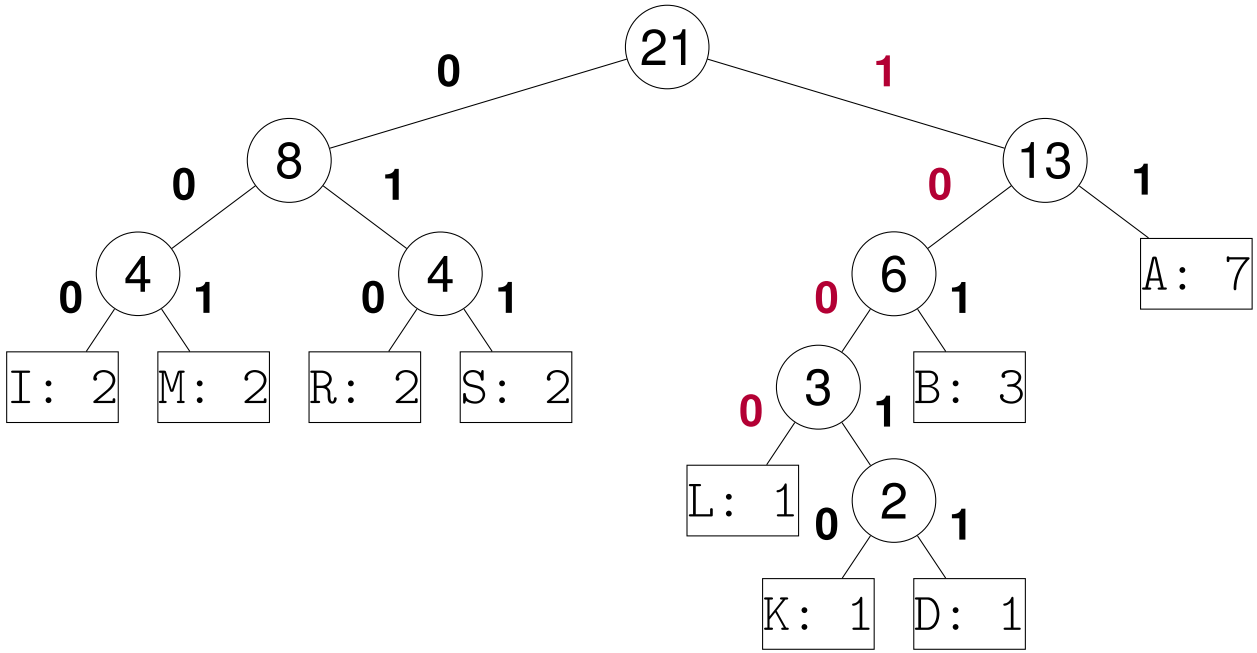
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	011
L	1000



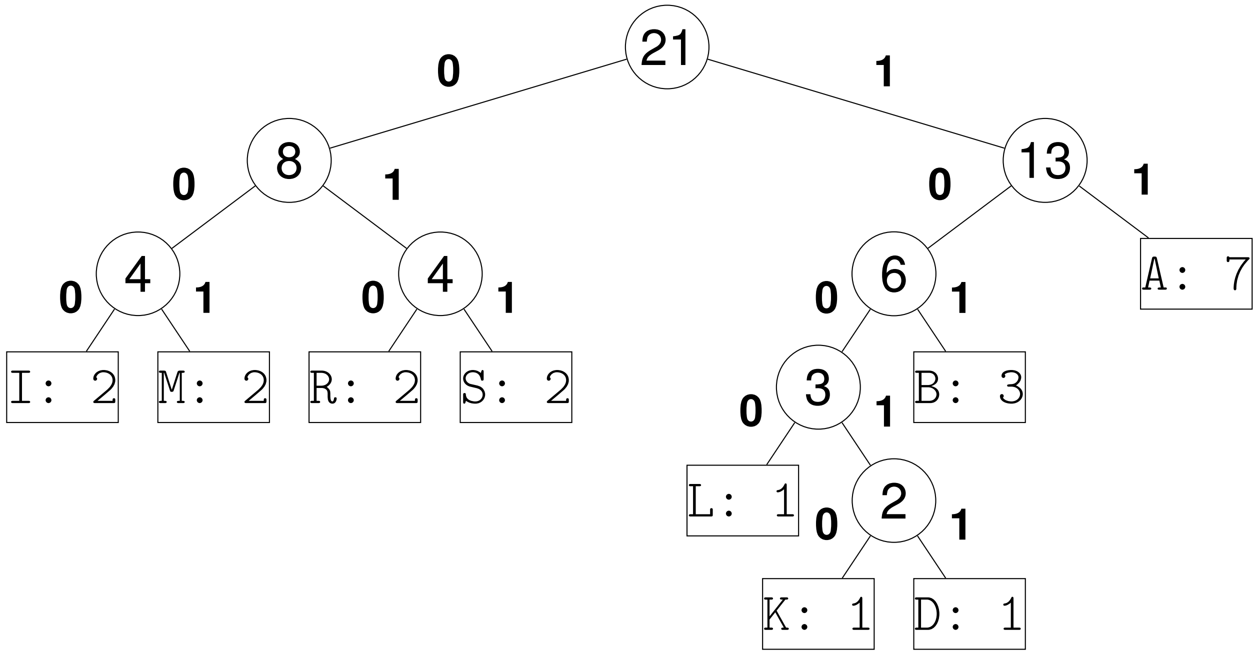
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	011
L	1000
K	



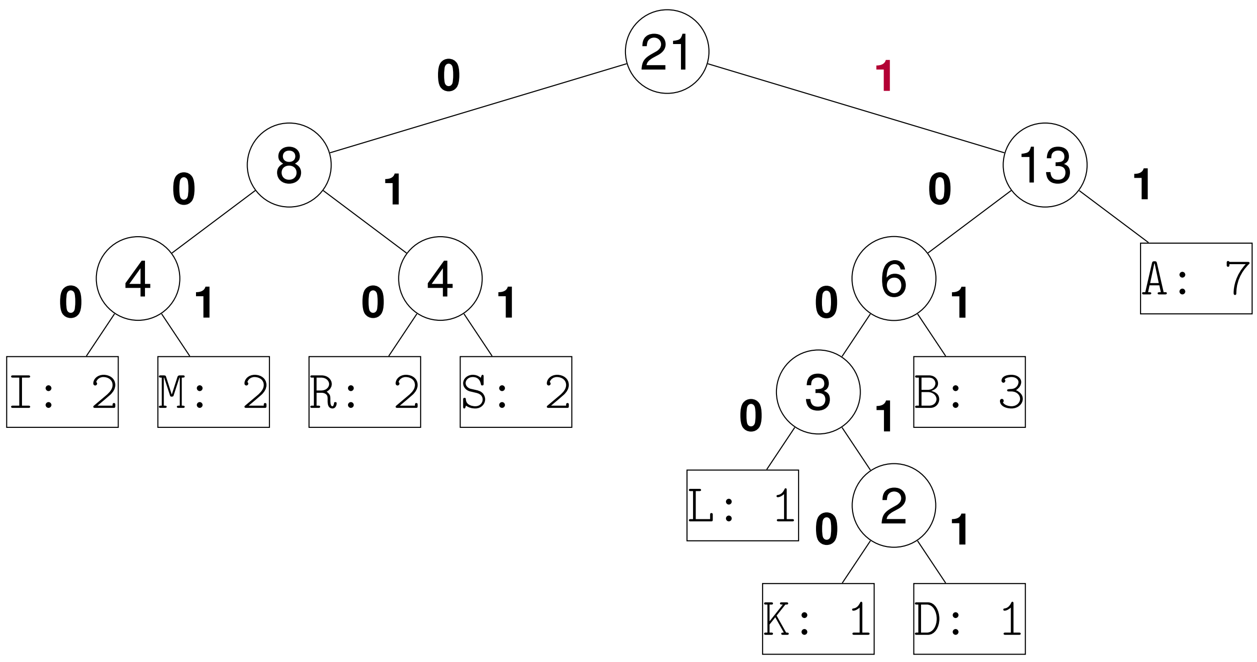
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	011
L	1000
K	



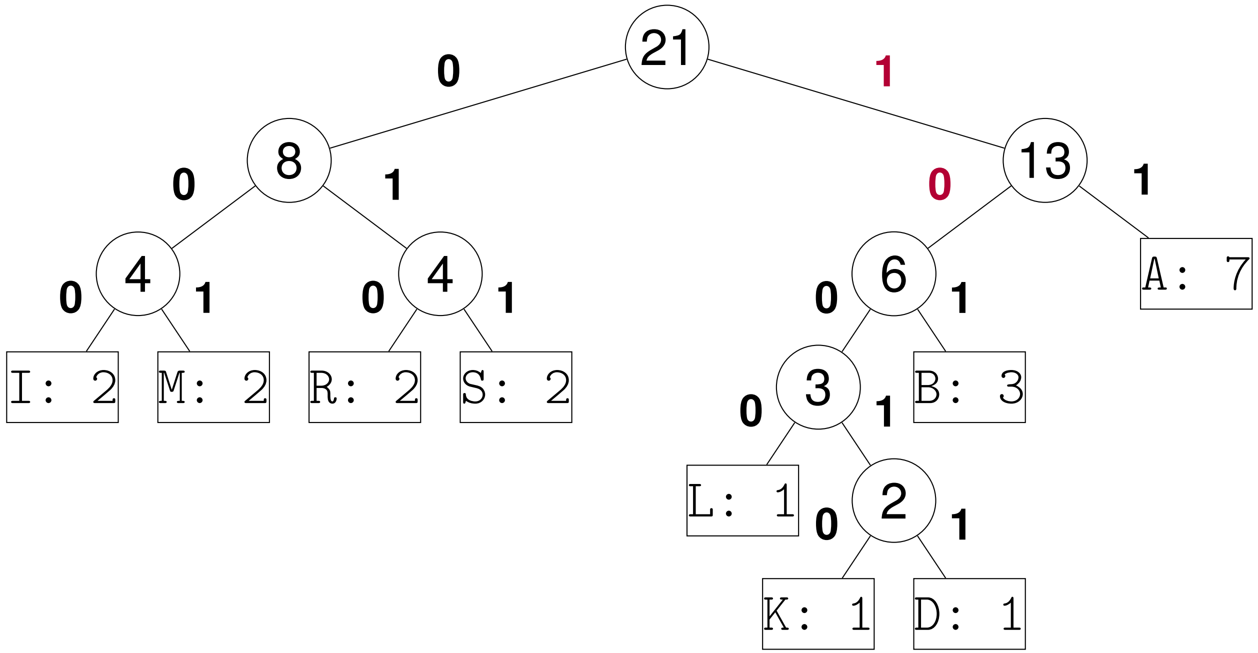
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	011
L	1000
K	



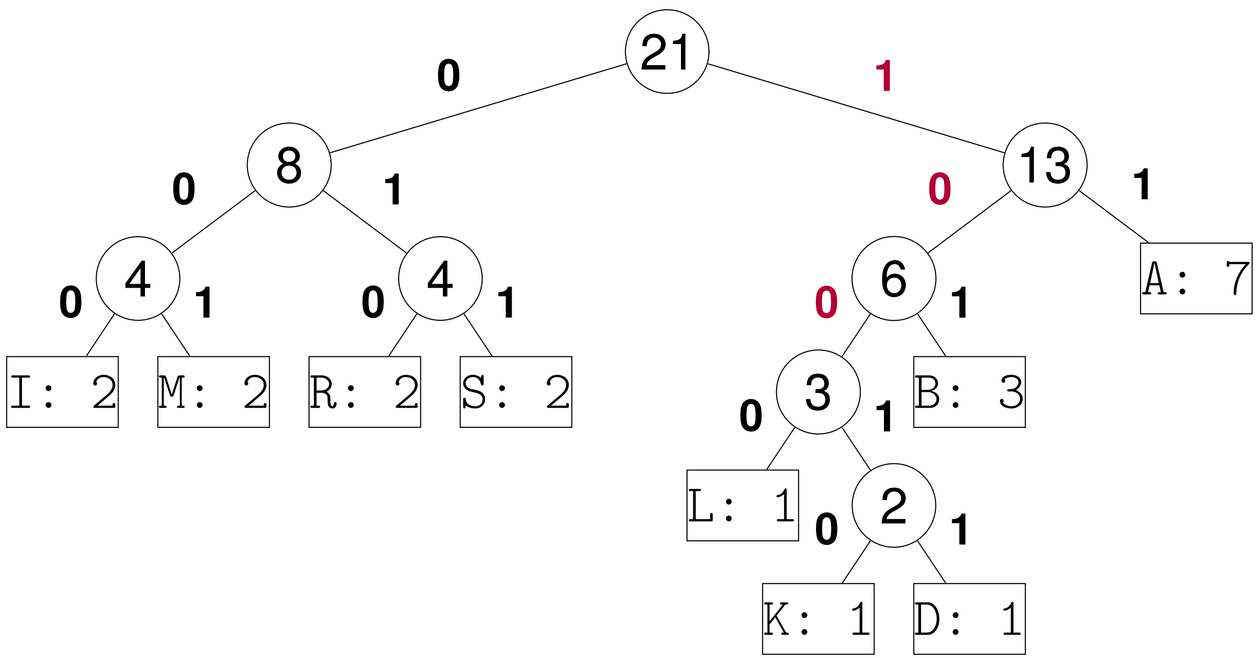
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	011
L	1000
K	



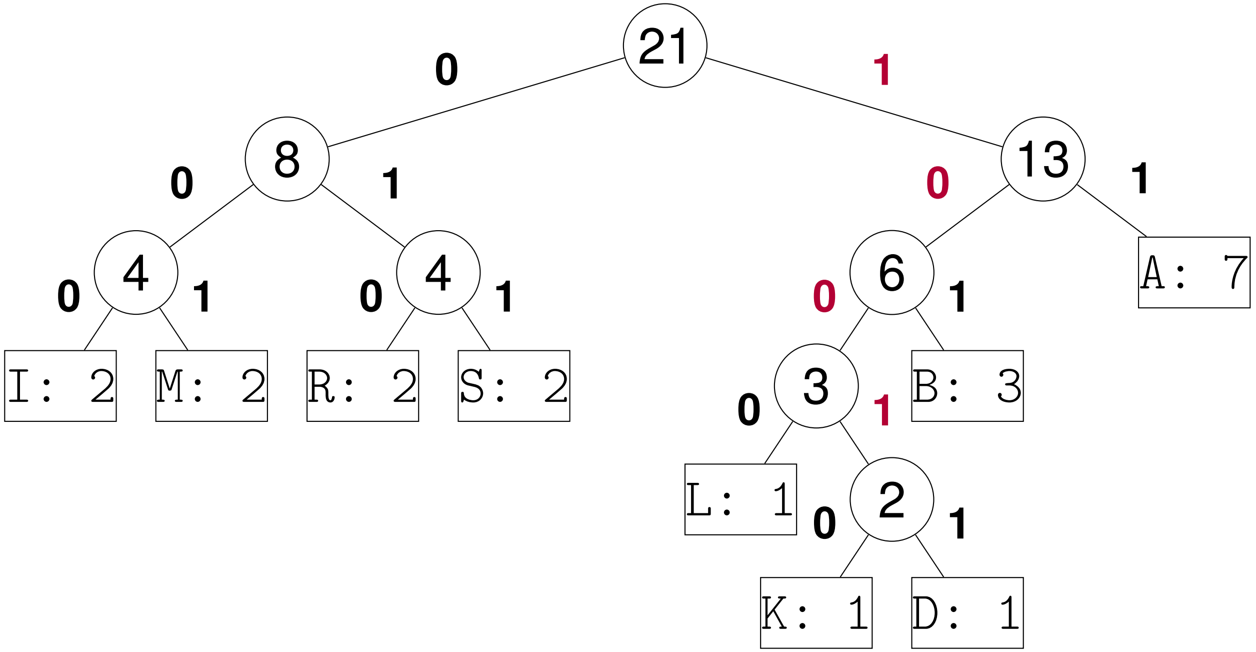
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	011
L	1000
K	



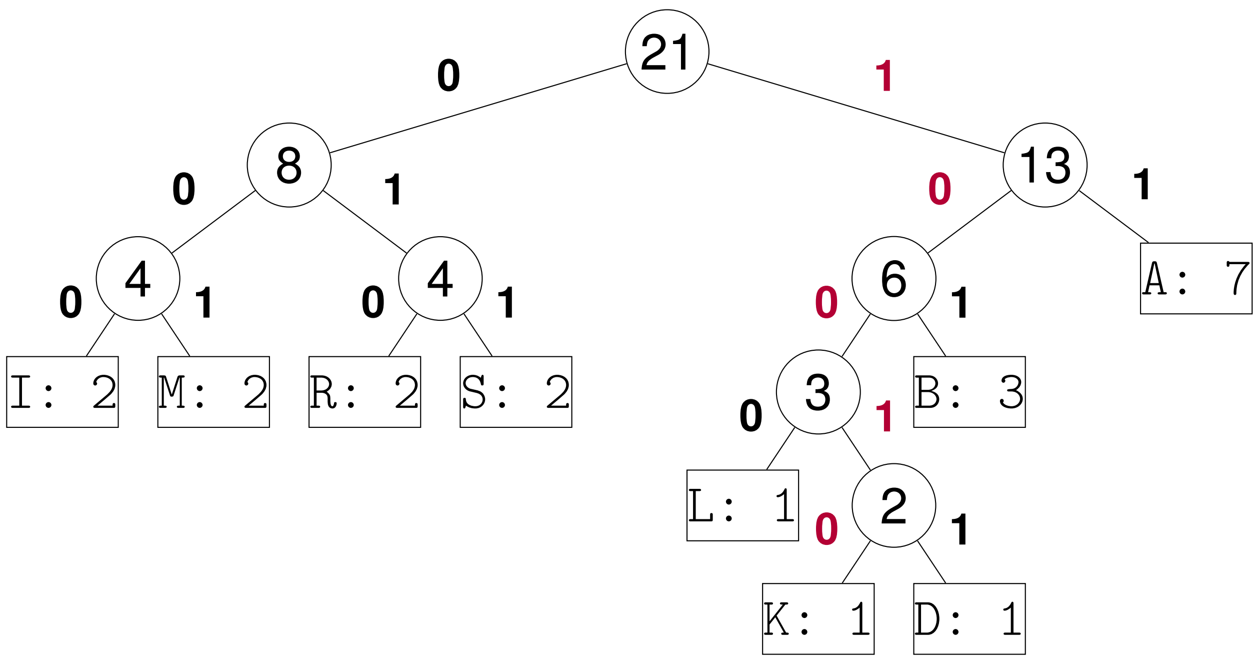
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	011
L	1000
K	



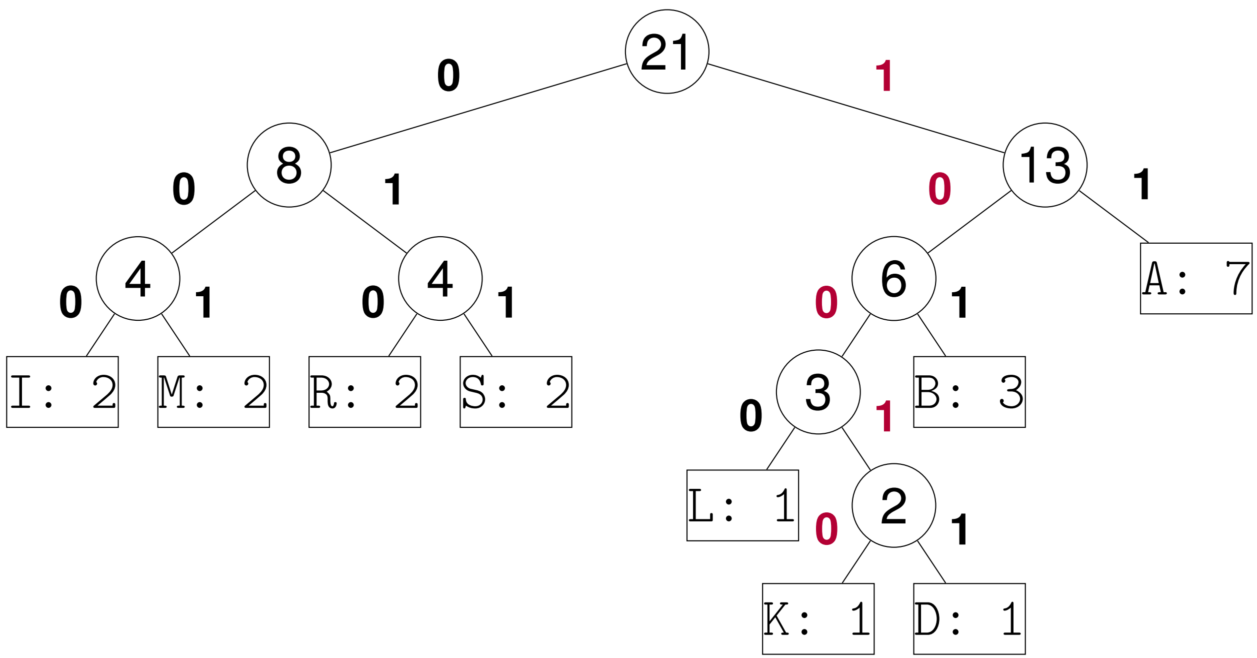
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	011
L	1000
K	10010



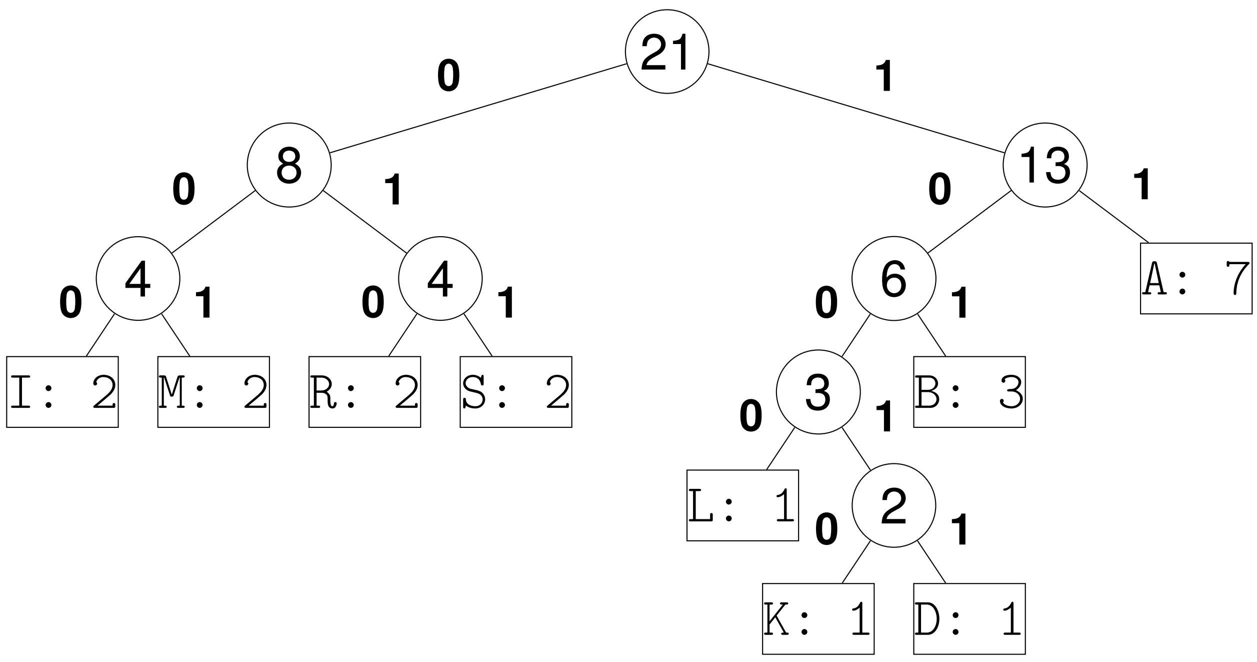
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	011
L	1000
K	10010
D	



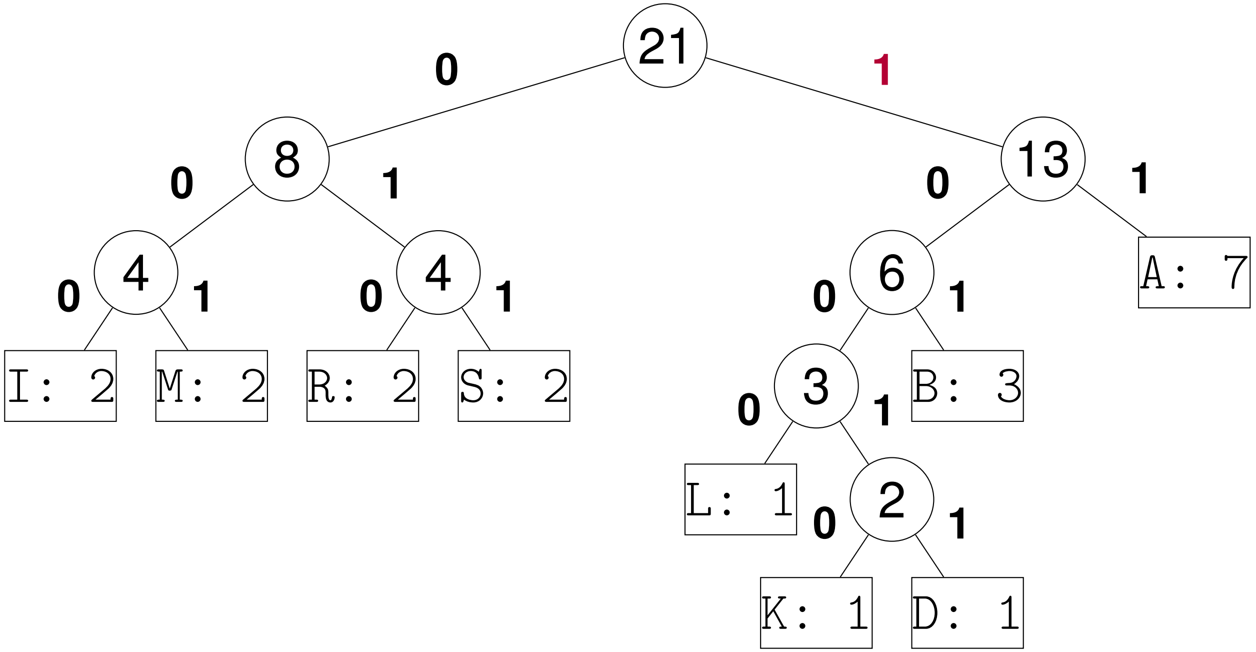
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	011
L	1000
K	10010
D	



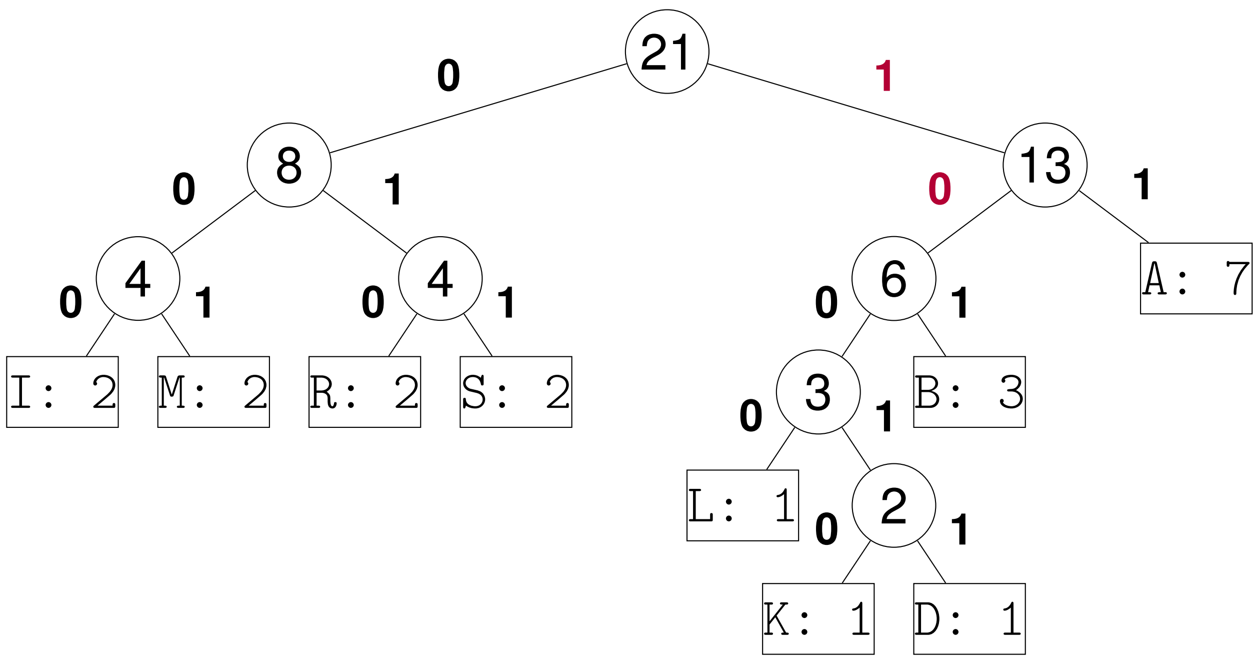
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	011
L	1000
K	10010
D	



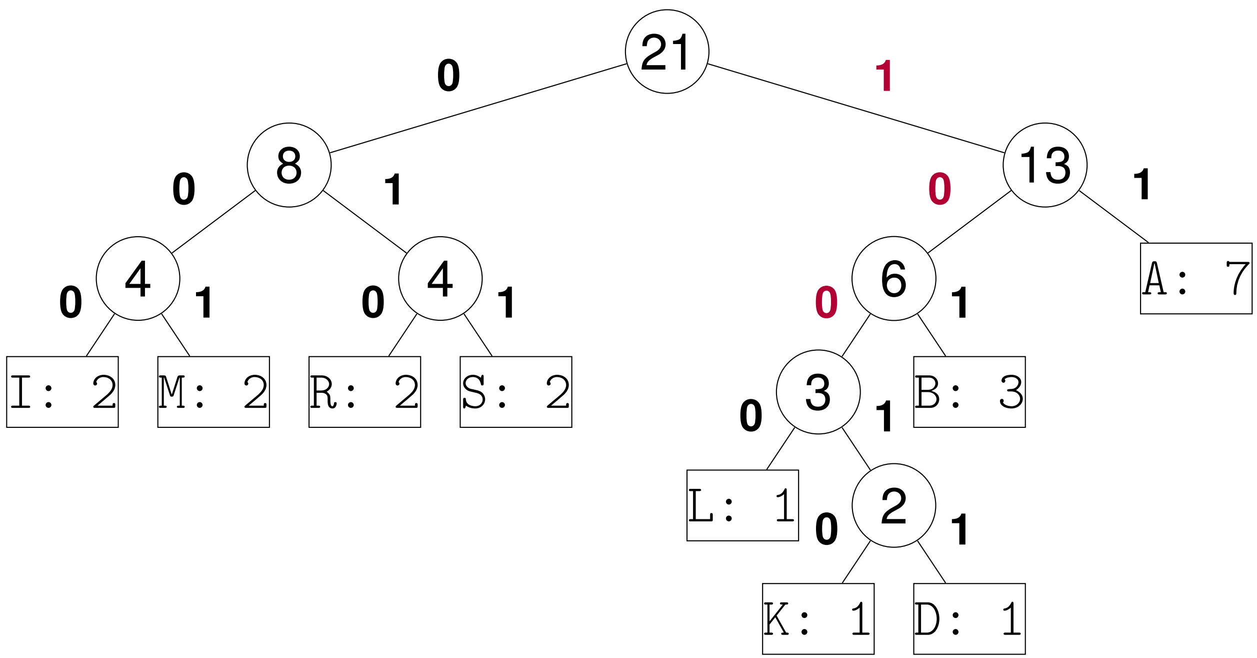
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	011
L	1000
K	10010
D	



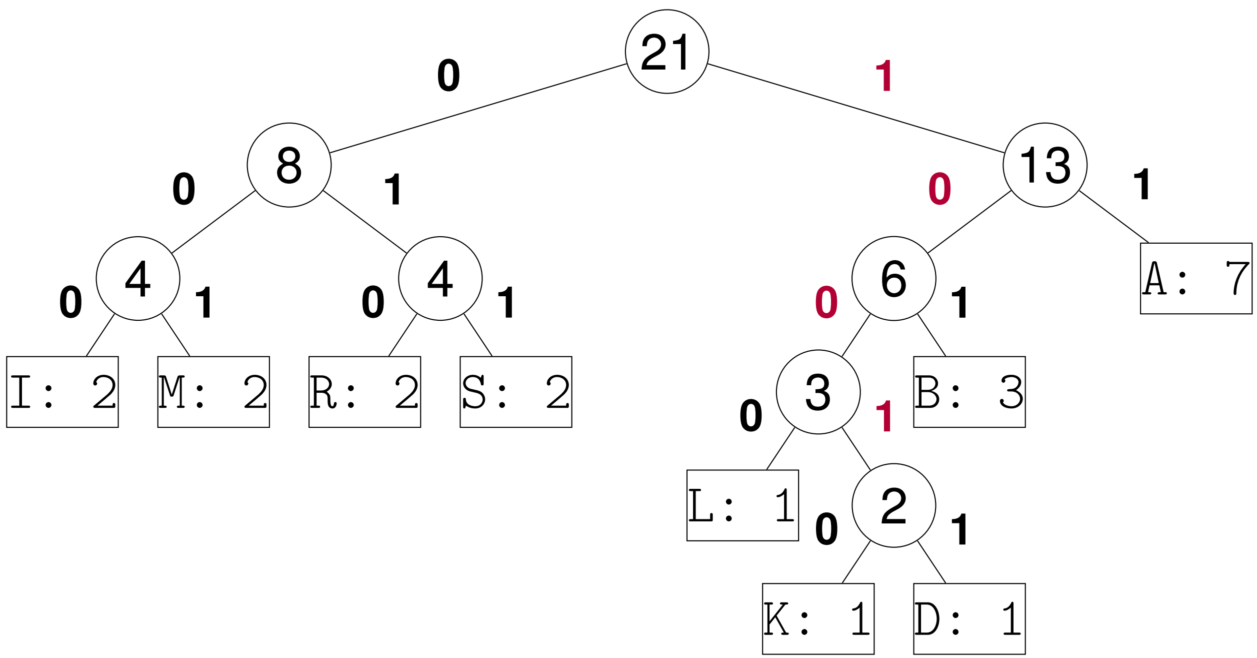
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	011
L	1000
K	10010
D	



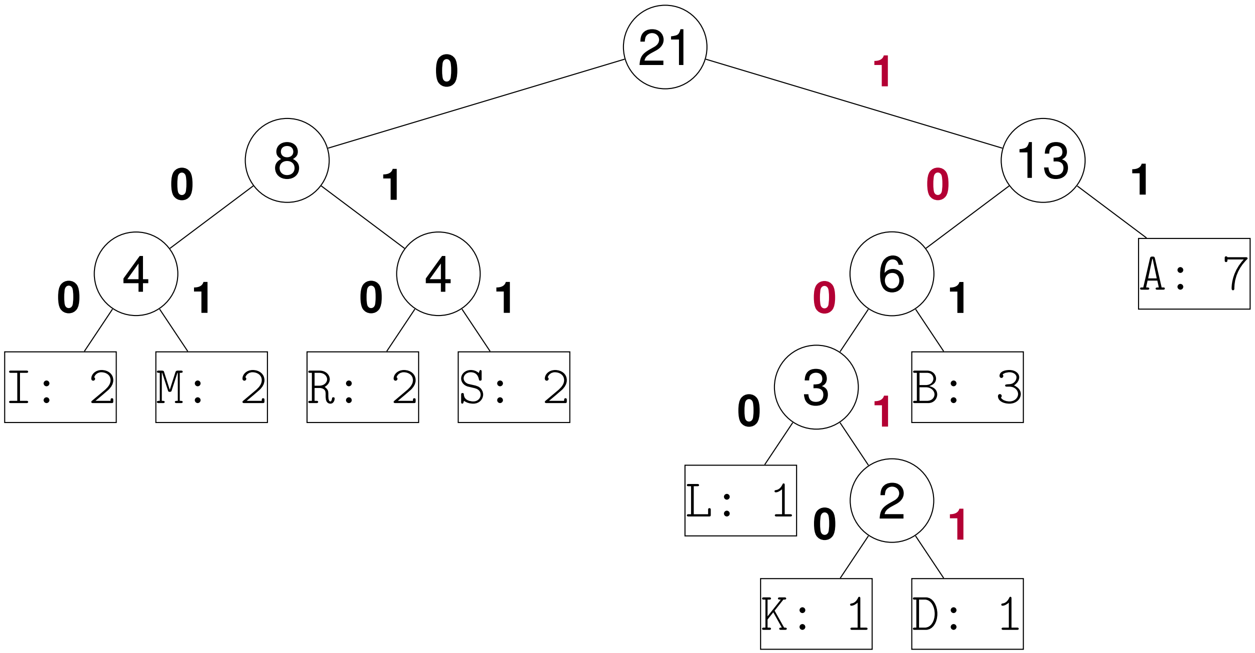
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	011
L	1000
K	10010
D	



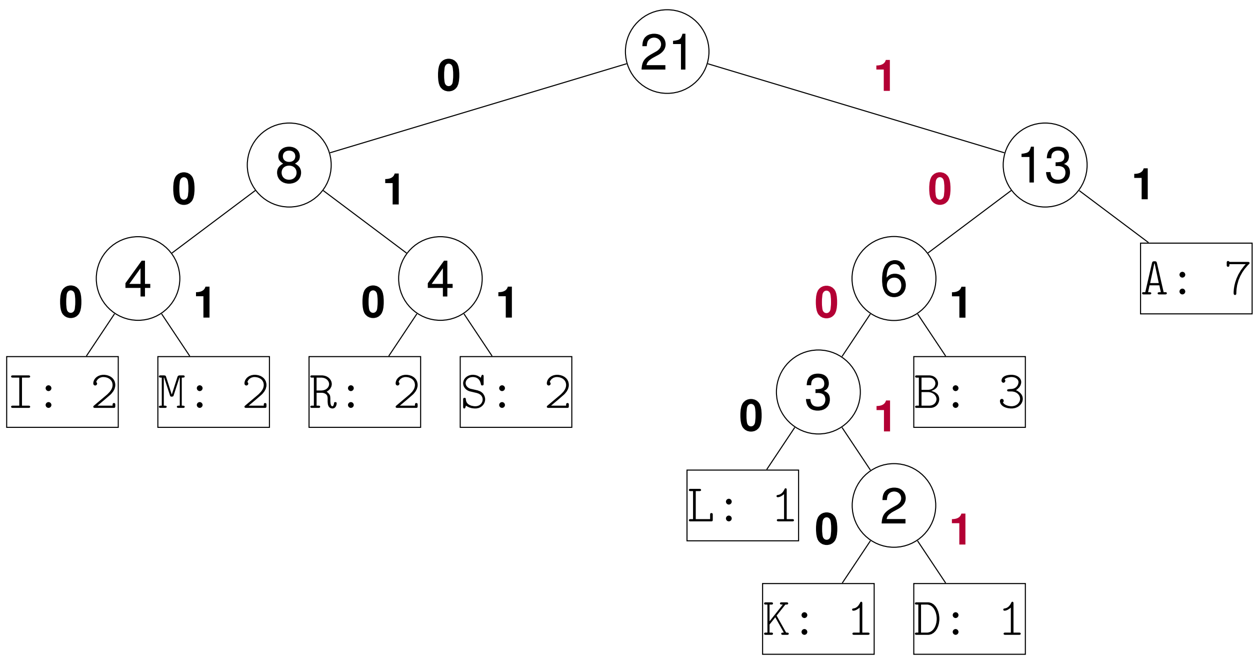
Aufgabe 5 – Huffman-Code

a) Erstellen Sie einen Huffman-Codierungsbaum für die folgende Zeichenkette:

ABRAKADABRASIMSALABIM

Lösung: Wir lesen die Codierung nun von **oben** nach **unten** ab.

Zeichen	Code
A	11
B	101
I	000
M	001
R	010
S	011
L	1000
K	10010
D	10011



Aufgabe 5 – Huffman-Code

Sei folgende Kodierung nun gegeben:

i	Zeichen x_i	Code	Anzahl N_i
1	A	11	7
2	B	101	3
3	I	000	2
4	M	001	2
5	R	010	2
6	S	011	2
7	L	1000	1
8	K	10010	1
9	D	10011	1

b) Wieviel Bits werden durch diese Codierung im Vergleich zu einer Codierung mit einer festen Codewort-Länge eingespart? (Das Codebuch ist zu vernachlässigen.)

Aufgabe 5 – Huffman-Code

b) Wieviel Bits werden durch diese Codierung im Vergleich zu einer Codierung mit einer festen Codewort-Länge eingespart?

i	x_i	Code	N_i
1	A	11	7
2	B	101	3
3	I	000	2
4	M	001	2
5	R	010	2
6	S	011	2
7	L	1000	1
8	K	10010	1
9	D	10011	1

Lösung: Wir fragen uns zunächst wie viele Bits der Huffmancode bräuchte:

$$\sum_{\text{Huff}} = \sum_A + \sum_B + \sum_I + \sum_M + \sum_R + \sum_S + \sum_L + \sum_K + \sum_D$$

Aufgabe 5 – Huffman-Code

b) Wieviel Bits werden durch diese Codierung im Vergleich zu einer Codierung mit einer festen Codewort-Länge eingespart?

i	x_i	Code	N_i
1	A	11	7
2	B	101	3
3	I	000	2
4	M	001	2
5	R	010	2
6	S	011	2
7	L	1000	1
8	K	10010	1
9	D	10011	1

Lösung: Wir fragen uns zunächst wie viele Bits der Huffmancode bräuchte:

$$\sum_A = 7 \cdot 2 \text{ bit}$$

$$\sum_{\text{Huff}} = \sum_A + \sum_B + \sum_I + \sum_M + \sum_R + \sum_S + \sum_L + \sum_K + \sum_D$$

Aufgabe 5 – Huffman-Code

b) Wieviel Bits werden durch diese Codierung im Vergleich zu einer Codierung mit einer festen Codewort-Länge eingespart?

i	x_i	Code	N_i
1	A	11	7
2	B	101	3
3	I	000	2
4	M	001	2
5	R	010	2
6	S	011	2
7	L	1000	1
8	K	10010	1
9	D	10011	1

Lösung: Wir fragen uns zunächst wie viele Bits der Huffmancode bräuchte:

$$\sum_A = 7 \cdot 2 \text{ bit} = 14 \text{ bit}$$

$$\sum_{\text{Huff}} = \sum_A + \sum_B + \sum_I + \sum_M + \sum_R + \sum_S + \sum_L + \sum_K + \sum_D$$

Aufgabe 5 – Huffman-Code

b) Wieviel Bits werden durch diese Codierung im Vergleich zu einer Codierung mit einer festen Codewort-Länge eingespart?

i	x_i	Code	N_i
1	A	11	7
2	B	101	3
3	I	000	2
4	M	001	2
5	R	010	2
6	S	011	2
7	L	1000	1
8	K	10010	1
9	D	10011	1

Lösung: Wir fragen uns zunächst wie viele Bits der Huffmancode bräuchte:

$$\sum_A = 7 \cdot 2 \text{ bit} = 14 \text{ bit}$$

$$\sum_B = 3 \cdot 3 \text{ bit}$$

$$\sum_{\text{Huff}} = 14 \text{ bit} + \sum_B + \sum_I + \sum_M + \sum_R + \sum_S + \sum_L + \sum_K + \sum_D$$

Aufgabe 5 – Huffman-Code

b) Wieviel Bits werden durch diese Codierung im Vergleich zu einer Codierung mit einer festen Codewort-Länge eingespart?

i	x_i	Code	N_i
1	A	11	7
2	B	101	3
3	I	000	2
4	M	001	2
5	R	010	2
6	S	011	2
7	L	1000	1
8	K	10010	1
9	D	10011	1

Lösung: Wir fragen uns zunächst wie viele Bits der Huffmancode bräuchte:

$$\sum_A = 7 \cdot 2 \text{ bit} = 14 \text{ bit}$$

$$\sum_B = 3 \cdot 3 \text{ bit} = 9 \text{ bit}$$

$$\sum_{\text{Huff}} = 14 \text{ bit} + 9 \text{ bit} + \sum_I + \sum_M + \sum_R + \sum_S + \sum_L + \sum_K + \sum_D$$

Aufgabe 5 – Huffman-Code

b) Wieviel Bits werden durch diese Codierung im Vergleich zu einer Codierung mit einer festen Codewort-Länge eingespart?

i	x_i	Code	N_i
1	A	11	7
2	B	101	3
3	I	000	2
4	M	001	2
5	R	010	2
6	S	011	2
7	L	1000	1
8	K	10010	1
9	D	10011	1

Lösung: Wir fragen uns zunächst wie viele Bits der Huffmancode bräuchte:

$$\sum_A = 7 \cdot 2 \text{ bit} = 14 \text{ bit}$$

$$\sum_B = 3 \cdot 3 \text{ bit} = 9 \text{ bit}$$

$$\sum_I = \sum_M = \sum_R = \sum_S = 2 \cdot 3 \text{ bit} = 6 \text{ bit}$$

$$\sum_{\text{Huff}} = 14 \text{ bit} + 9 \text{ bit} + 6 \text{ bit} + 6 \text{ bit} + 6 \text{ bit} + 6 \text{ bit} + \sum_L + \sum_K + \sum_D$$

Aufgabe 5 – Huffman-Code

b) Wieviel Bits werden durch diese Codierung im Vergleich zu einer Codierung mit einer festen Codewort-Länge eingespart?

i	x_i	Code	N_i
1	A	11	7
2	B	101	3
3	I	000	2
4	M	001	2
5	R	010	2
6	S	011	2
7	L	1000	1
8	K	10010	1
9	D	10011	1

Lösung: Wir fragen uns zunächst wie viele Bits der Huffmancode bräuchte:

$$\sum_A = 7 \cdot 2 \text{ bit} = 14 \text{ bit}$$

$$\sum_B = 3 \cdot 3 \text{ bit} = 9 \text{ bit}$$

$$\sum_I = \sum_M = \sum_R = \sum_S = 2 \cdot 3 \text{ bit} = 6 \text{ bit}$$

$$\sum_L = 1 \cdot 4 \text{ bit} = 4 \text{ bit}$$

$$\sum_{\text{Huff}} = 14 \text{ bit} + 9 \text{ bit} + 6 \text{ bit} + 6 \text{ bit} + 6 \text{ bit} + 6 \text{ bit} + 4 \text{ bit} + \sum_K + \sum_D$$

Aufgabe 5 – Huffman-Code

b) Wieviel Bits werden durch diese Codierung im Vergleich zu einer Codierung mit einer festen Codewort-Länge eingespart?

i	x_i	Code	N_i
1	A	11	7
2	B	101	3
3	I	000	2
4	M	001	2
5	R	010	2
6	S	011	2
7	L	1000	1
8	K	10010	1
9	D	10011	1

Lösung: Wir fragen uns zunächst wie viele Bits der Huffmancode bräuchte:

$$\Sigma_A = 7 \cdot 2\text{ bit} = 14\text{ bit}$$
$$\Sigma_B = 3 \cdot 3\text{ bit} = 9\text{ bit}$$
$$\Sigma_I = \Sigma_M = \Sigma_R = \Sigma_S = 2 \cdot 3\text{ bit} = 6\text{ bit}$$
$$\Sigma_L = 1 \cdot 4\text{ bit} = 4\text{ bit}$$
$$\Sigma_K = \Sigma_D = 5\text{ bit}$$

$$\Sigma_{\text{Huff}} = 14\text{ bit} + 9\text{ bit} + 6\text{ bit} + 6\text{ bit} + 6\text{ bit} + 6\text{ bit} + 4\text{ bit} + 5\text{ bit} + 5\text{ bit} = 61\text{ bit}$$

Aufgabe 5 – Huffman-Code

b) Wieviel Bits werden durch diese Codierung im Vergleich zu einer Codierung mit einer festen Codewort-Länge eingespart?
Lösung: Wenn der Huffmancode nun 61 bit benötigt, stellt sich die Frage wie viele Bit man für eine „naive" Codierung mit fester Länge benötigt:

Aufgabe 5 – Huffman-Code

b) Wieviel Bits werden durch diese Codierung im Vergleich zu einer Codierung mit einer festen Codewort-Länge eingespart?
Lösung: Wenn der Huffmancode nun 61 bit benötigt, stellt sich die Frage wie viele Bit man für eine „naive“ Codierung mit fester Länge benötigt:
Wir haben 9 unterschiedliche Zeichen zu kodieren, sprich wir brauchen eine Codewortlänge von

Aufgabe 5 – Huffman-Code

b) Wieviel Bits werden durch diese Codierung im Vergleich zu einer Codierung mit einer festen Codewort-Länge eingespart?
Lösung: Wenn der Huffmancode nun 61 bit benötigt, stellt sich die Frage wie viele Bit man für eine „naive“ Codierung mit fester Länge benötigt: Wir haben 9 unterschiedliche Zeichen zu kodieren, sprich wir brauchen eine Codewortlänge von $\lceil ld\, 9 \rceil = 4$ bit.

Aufgabe 5 – Huffman-Code

b) Wieviel Bits werden durch diese Codierung im Vergleich zu einer Codierung mit einer festen Codewort-Länge eingespart?
Lösung: Wenn der Huffmancode nun 61 bit benötigt, stellt sich die Frage wie viele Bit man für eine „naive“ Codierung mit fester Länge benötigt: Wir haben 9 unterschiedliche Zeichen zu kodieren, sprich wir brauchen eine Codewortlänge von $\lceil ld\, 9 \rceil = 4$ bit. Mit 21 zu kodierenden Zeichen ergibt sich eine Kodewortlänge von 84 bit.

Aufgabe 5 – Huffman-Code

b) Wieviel Bits werden durch diese Codierung im Vergleich zu einer Codierung mit einer festen Codewort-Länge eingespart?
Lösung: Wenn der Huffmancode nun 61 bit benötigt, stellt sich die Frage wie viele Bit man für eine „naive“ Codierung mit fester Länge benötigt: Wir haben 9 unterschiedliche Zeichen zu kodieren, sprich wir brauchen eine Codewortlänge von $\lceil ld\, 9 \rceil = 4$ bit. Mit 21 zu kodierenden Zeichen ergibt sich eine Kodewortlänge von 84 bit.
Wir erhalten somit eine Einsparung von

$$\blacksquare = 84 \text{ bit} - 61 \text{ bit} = 23 \text{ bit}.$$

Aufgabe 5 – Huffman-Code

Seien die Häufigkeiten nun nochmal gegeben:

Buchstabe	A	B	I	M	R	S	L	K	D
Anzahl	7	3	2	2	2	2	1	1	1

c) Wieviel Bits sind minimal nötig (optimale Codierung)? Wieviel Prozent schlechter ist der Huffman-Code?

Aufgabe 5 – Huffman-Code

Seien die Häufigkeiten nun nochmal gegeben:

Buchstabe	A	B	I	M	R	S	L	K	D
Anzahl	7	3	2	2	2	2	1	1	1

c) Wieviel Bits sind minimal nötig (optimale Codierung)? Wieviel Prozent schlechter ist der Huffman-Code?

Optimale Codierung

Die theoretisch minimale Anzahl an Bits zur Codierung eines Zeichens x entspricht dessen Informationsgehalt

$$I(x) = -ld\left(\frac{\text{Anzahl}(x)}{\text{Gesamtzeichenanzahl}}\right)$$

Übungen zur Grundlagen der Technischen Informatik

Übung 3 – Zahlendarstellung, -konversion und IEEE754

Florian Frank

Friedrich-Alexander-Universität Erlangen-Nürnberg

Wintersemester 2018/19

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Was machen wir heute?

Aufgabe 1 – Zahlendarstellungen in der Theorie

Was machen wir heute?

Aufgabe 1 – Zahlendarstellungen in der Theorie

Aufgabe 2 – Zahlendarstellungen in der Praxis

Was machen wir heute?

Aufgabe 1 – Zahlendarstellungen in der Theorie

Aufgabe 2 – Zahlendarstellungen in der Praxis

Aufgabe 3 – Zahlendarstellungen

Was machen wir heute?

Aufgabe 1 – Zahlendarstellungen in der Theorie

Aufgabe 2 – Zahlendarstellungen in der Praxis

Aufgabe 3 – Zahlendarstellungen

Aufgabe 4 – Zahlenkonversion

Was machen wir heute?

Aufgabe 1 – Zahlendarstellungen in der Theorie

Aufgabe 2 – Zahlendarstellungen in der Praxis

Aufgabe 3 – Zahlendarstellungen

Aufgabe 4 – Zahlenkonversion

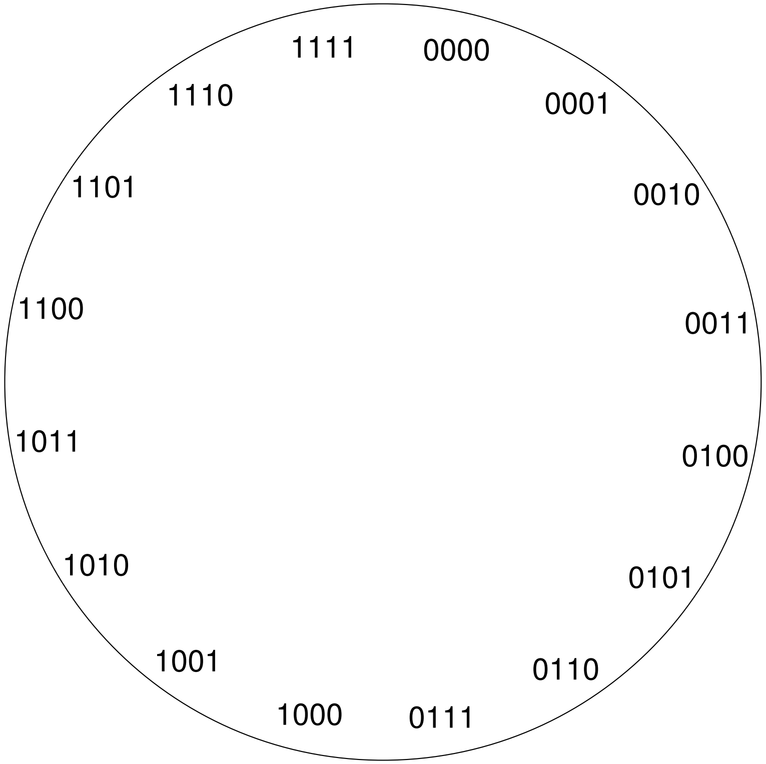
Aufgabe 5 – Konversion von Gleitkommazahlen

Aufgabe 1 – Zahlendarstellungen in der Theorie

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

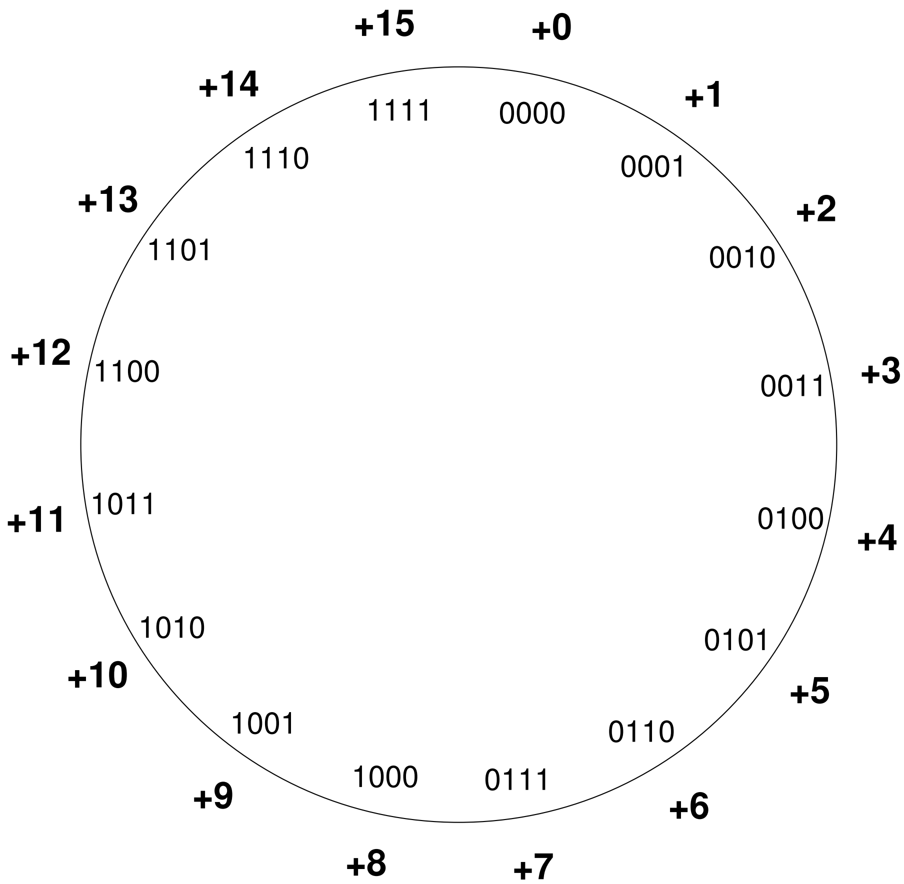
Aufgabe 1 – Zahlendarstellungen: Theorie

Vorzeichenlose Zahlendarstellung – `unsigned int`



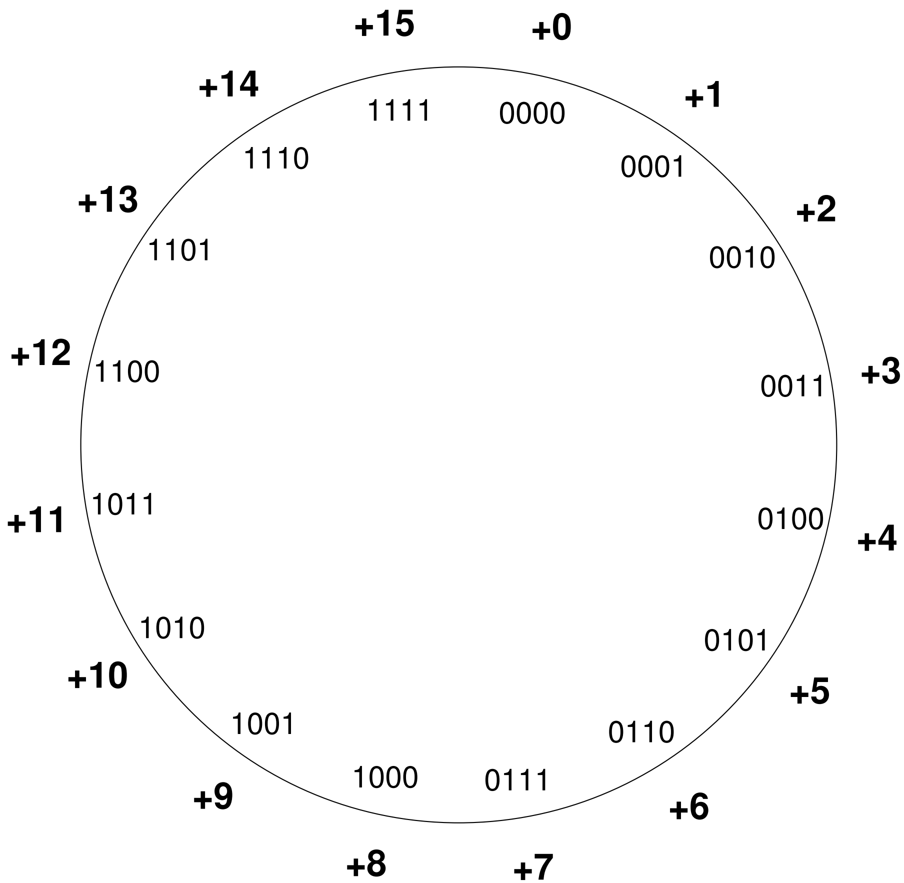
Aufgabe 1 – Zahlendarstellungen: Theorie

Vorzeichenlose Zahlendarstellung – `unsigned int`



Aufgabe 1 – Zahlendarstellungen: Theorie

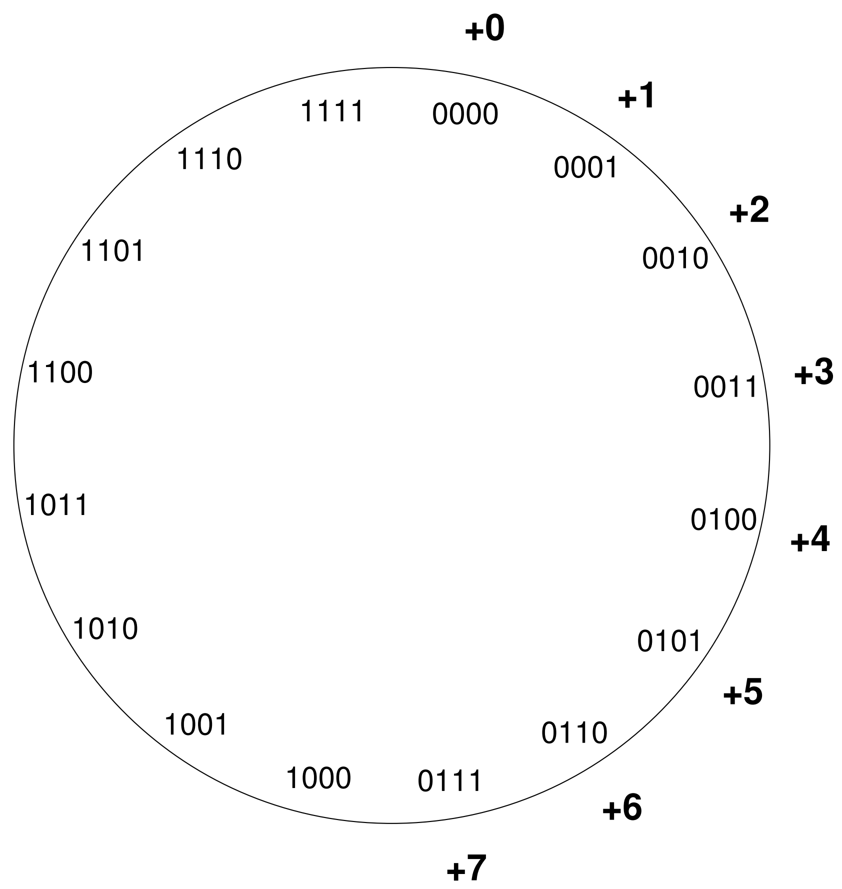
Vorzeichenlose Zahlendarstellung – `unsigned int`



Wertebereich einer n bit breiten Zahl: $\left[0, 2^n - 1\right]$

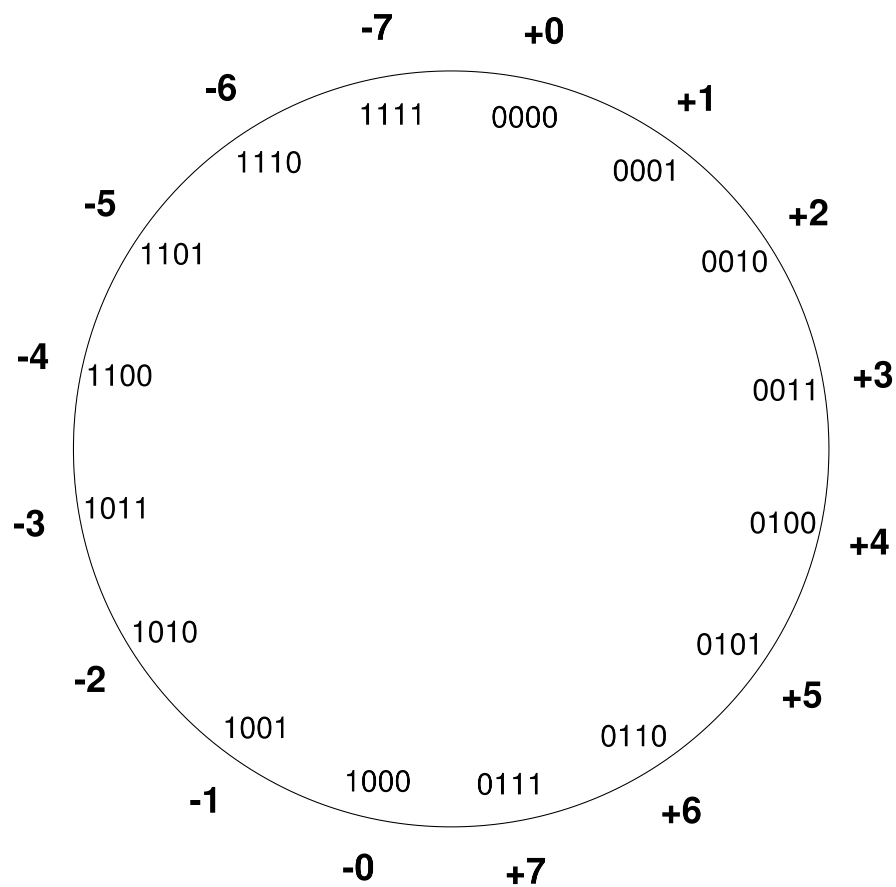
Aufgabe 1 – Zahlendarstellungen: Theorie

Vorzeichen-/Betragsdarstellung



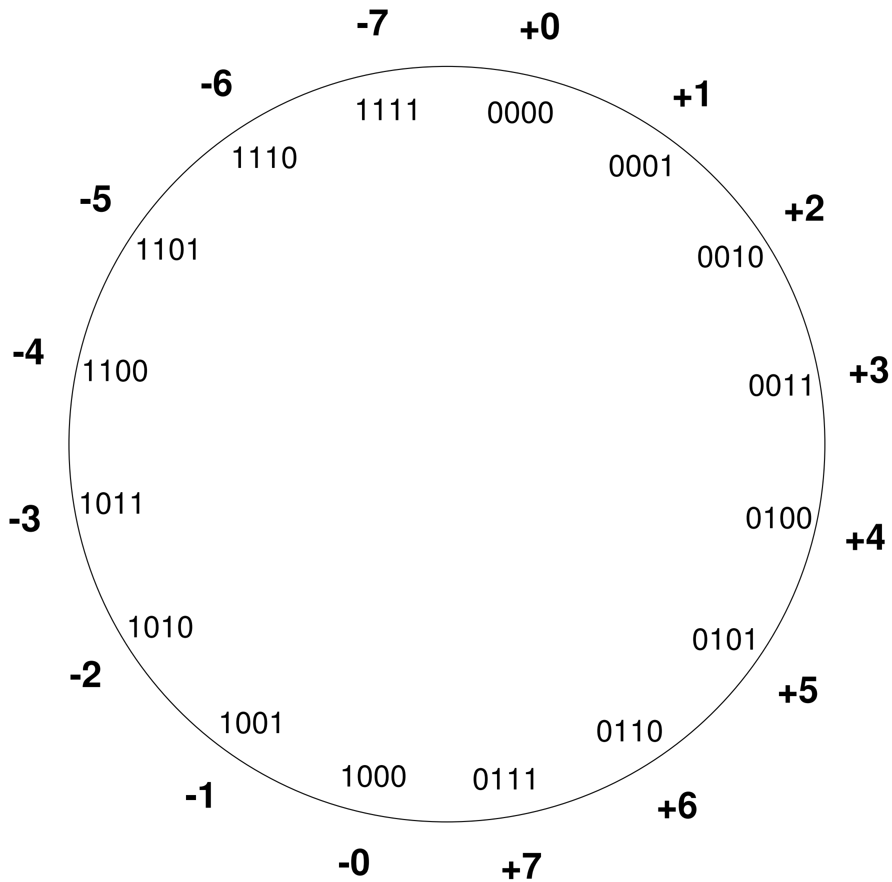
Aufgabe 1 – Zahlendarstellungen: Theorie

Vorzeichen-/Betragsdarstellung



Aufgabe 1 – Zahlendarstellungen: Theorie

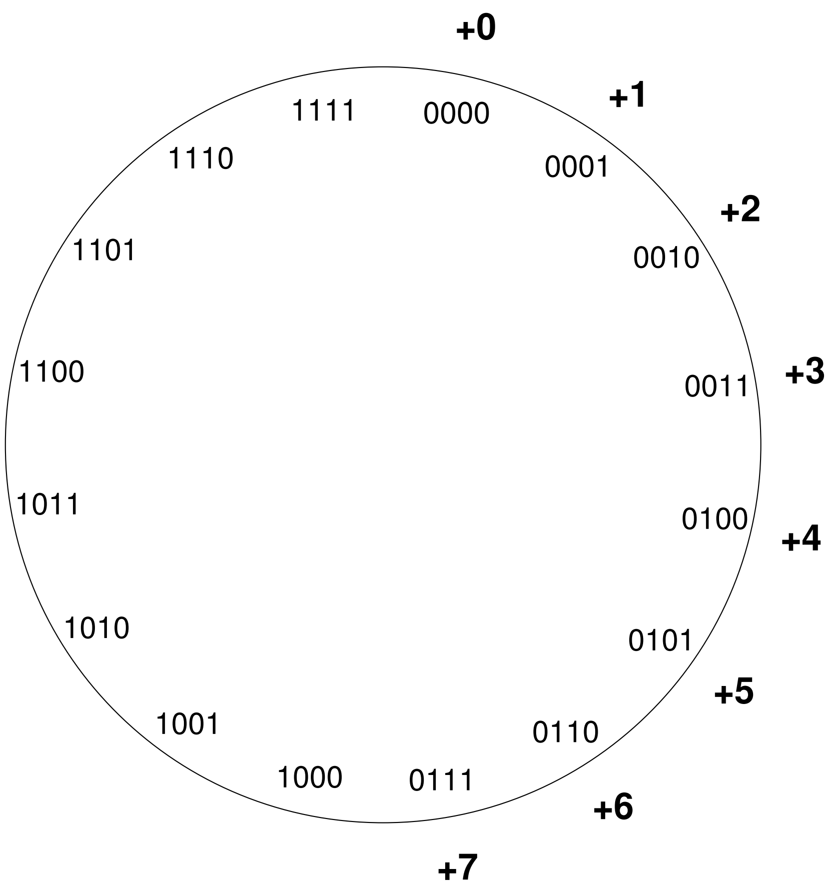
Vorzeichen-/Betragsdarstellung



Wertebereich einer n bit breiten Zahl: $\left[-2^{n-1}+1, 2^{n-1}-1\right]$

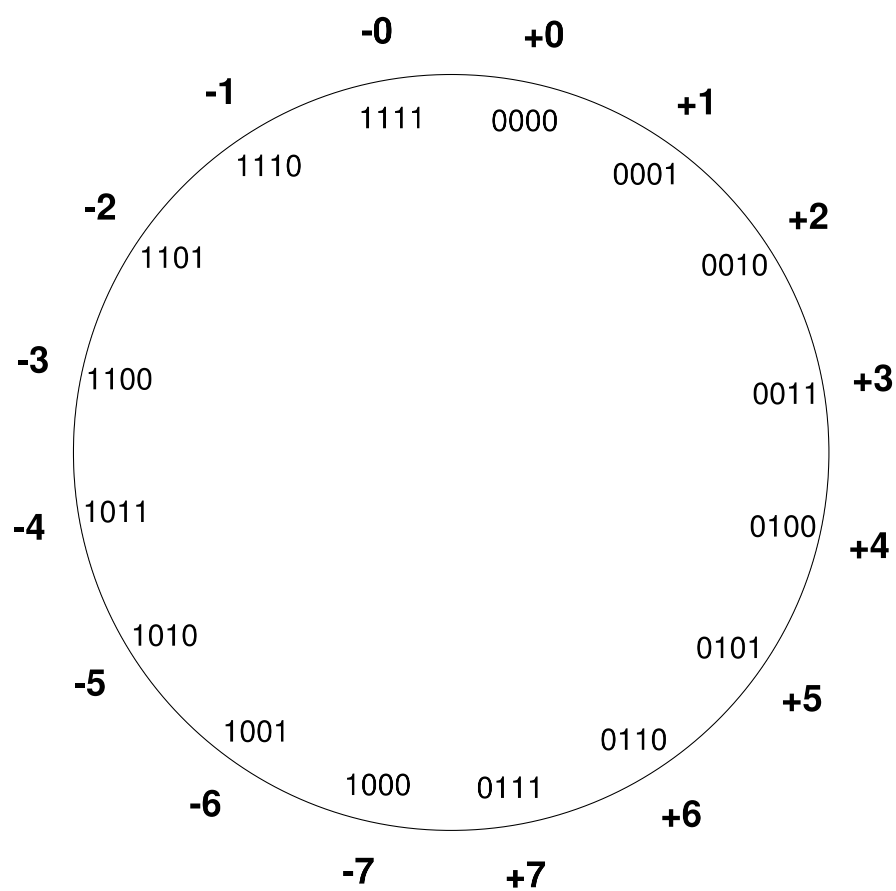
Aufgabe 1 – Zahlendarstellungen: Theorie

Einerkomplement



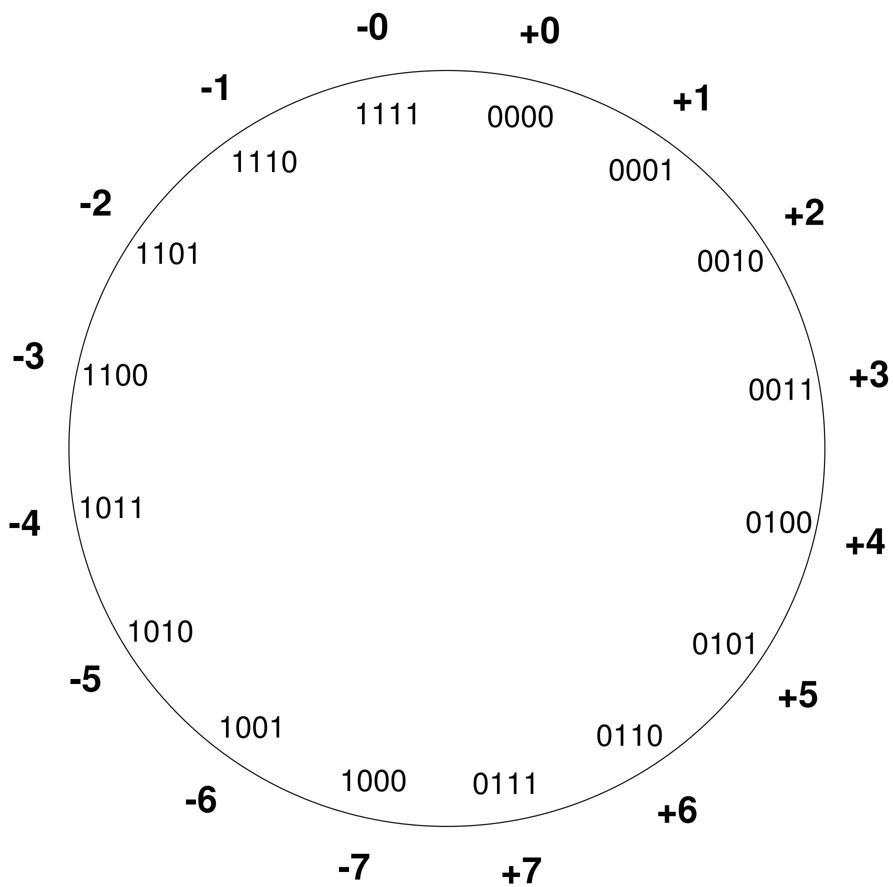
Aufgabe 1 – Zahlendarstellungen: Theorie

Einerkomplement



Aufgabe 1 – Zahlendarstellungen: Theorie

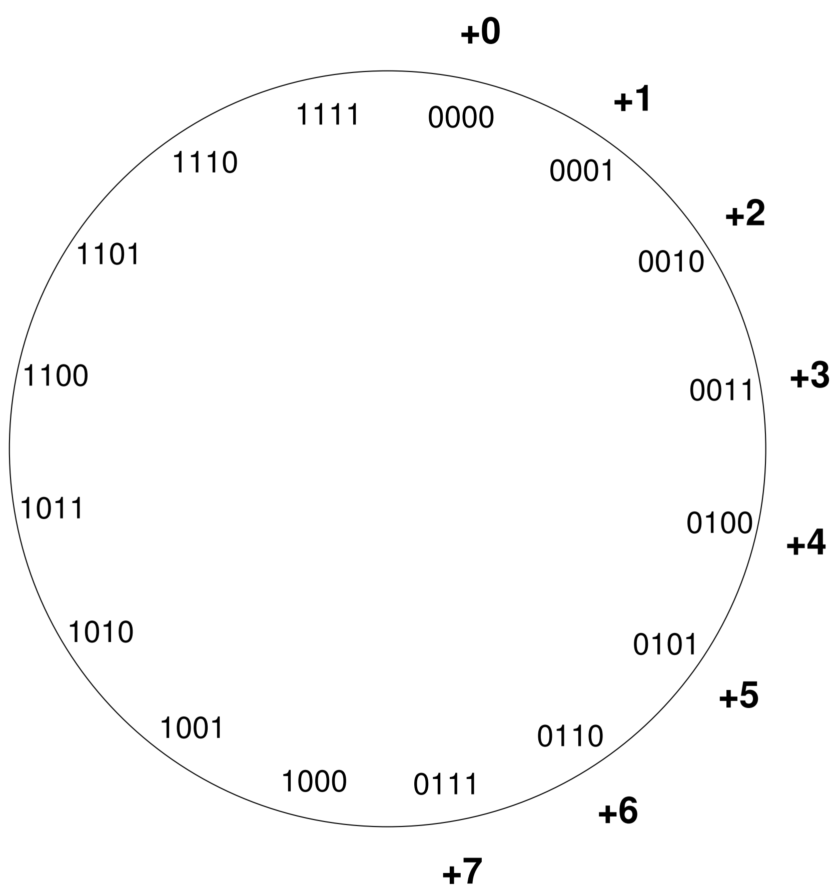
Einerkomplement



Wertebereich einer n bit breiten Zahl: $\left[-2^{n-1}+1, 2^{n-1}-1\right]$

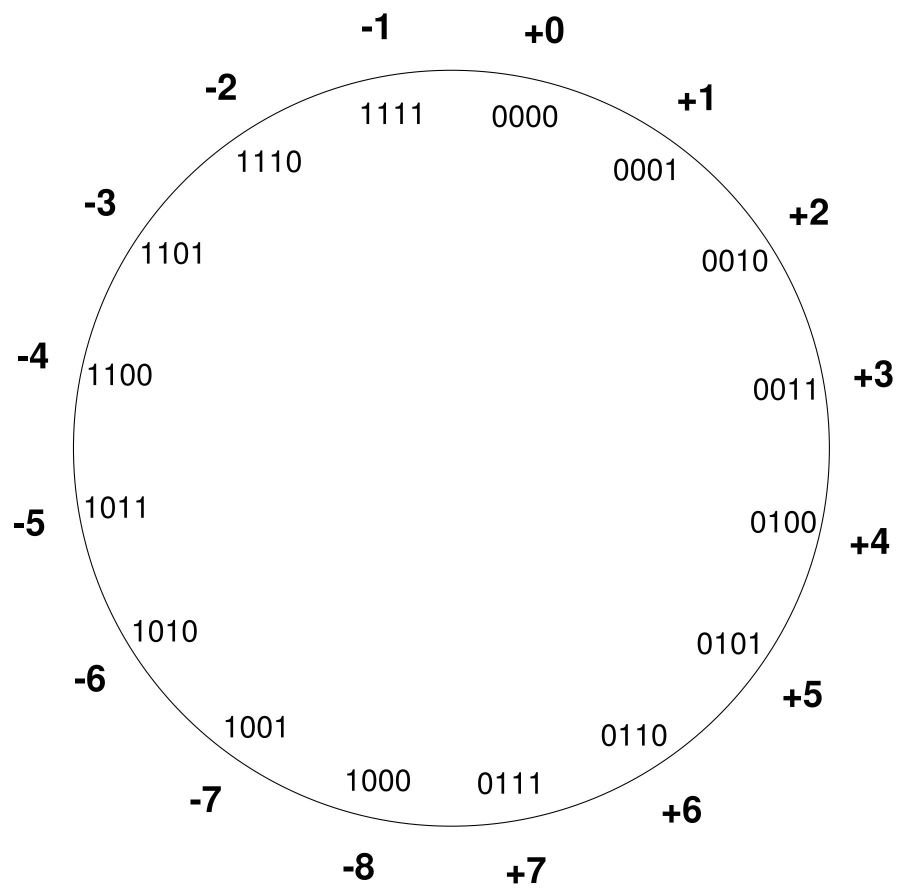
Aufgabe 1 – Zahlendarstellungen: Theorie

Zweierkomplement



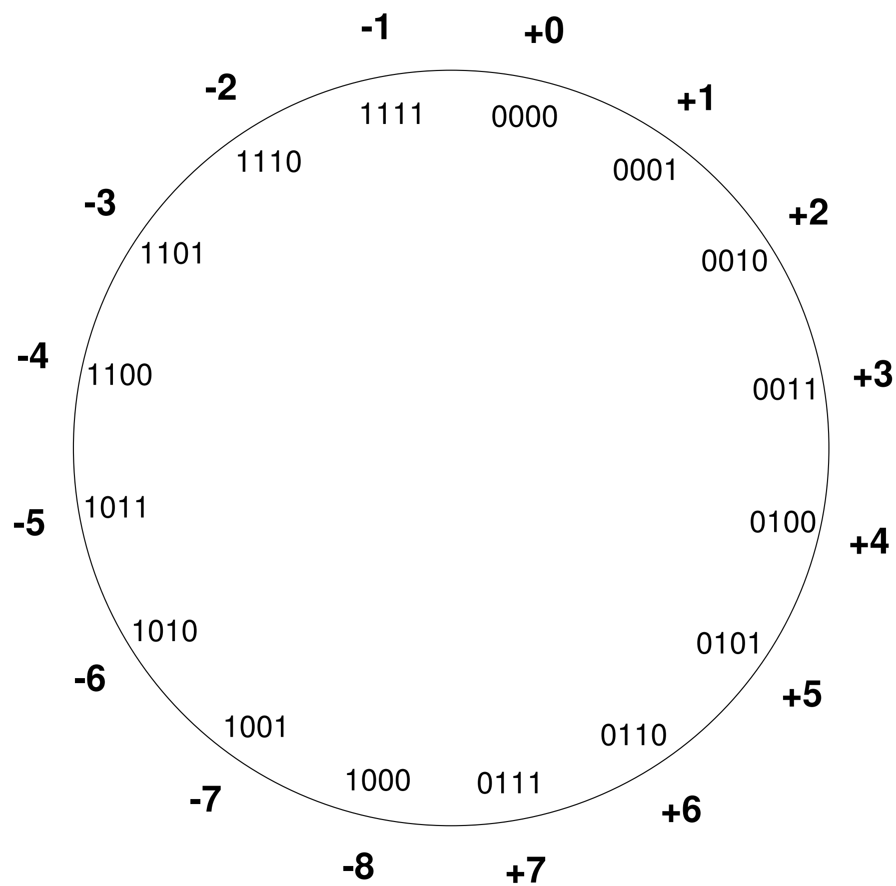
Aufgabe 1 – Zahlendarstellungen: Theorie

Zweierkomplement



Aufgabe 1 – Zahlendarstellungen: Theorie

Zweierkomplement



Wertebereich einer n bit breiten Zahl: $\left[-2^{n-1}, 2^{n-1} - 1\right]$

Aufgabe 2 – Zahlendarstellungen in der Praxis

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Aufgabe 2 – Zahlendarstellungen: Praxis

Gegeben seien die folgenden Dezimalzahlen:

- 2
- 64
- 255
- -254
- -32

Stellen Sie diese Zahlen **jeweils** in ...

i ... Vorzeichen/Betragsdarstellung dar.

ii ... 1er-Komplementdarstellung dar.

iii ... 2er-Komplementdarstellung dar.

Aufgabe 3 – Zahlendarstellungen

design

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Aufgabe 3 – Zahlendarstellungen

Gegeben seien die folgenden positiven Binärzahlen:

- 10_2
- 10100_2
- 111111_2
- 1000000_2
- 11111111_2
- 11111110_2.

Stellen Sie diese Zahlen jeweils als ...

i ... Hexadezimalzahl dar.

ii ... Oktalzahl dar.

iii ... BCD-Zahl dar.

Aufgabe 3 – Begriffsklärung

BCD-Zahl

BCD-Zahl $\equiv$ **B**inary **C**oded **D**igit
Jede Dezimalziffer wird durch vier Binärstellen repräsentiert. Damit ist die Dezimalzahl leicht rekonstruierbar, die Kodierung aber vergleichsweise ineffizient.

Aufgabe 3 – Begriffsklärung

BCD-Zahl

BCD-Zahl $\equiv$ **B**inary **C**oded **D**igit

Jede Dezimalziffer wird durch vier Binärstellen repräsentiert. Damit ist die Dezimalzahl leicht rekonstruierbar, die Kodierung aber vergleichsweise ineffizient.

Oktalsystem/Hexadezimalsystem

Ein polyadisches System zur Basis 8 heißt **Oktalsystem**.
Ein polyadisches System zur Basis 16 heißt **Hexadezimalsystem**.

Aufgabe 3 – Begriffsklärung

Umwandlung verwandter Systeme

Sei eine Zahl Z aus dem System zur Basis a in das System zur Basis b (mit $a, b \in \mathbb{N}$) zu kodieren und es gilt:

Aufgabe 3 – Begriffsklärung

Umwandlung verwandter Systeme

Sei eine Zahl Z aus dem System zur Basis a in das System zur Basis b (mit $a, b \in \mathbb{N}$) zu kodieren und es gilt:

- $a = b^n$ mit $n \in \mathbb{N}$.
  Dann wird jede Stelle von Z n-äquidistant zerteilt (sprich: in n-Stellen (zur Basis b) zerlegt).

Aufgabe 3 – Begriffsklärung

Umwandlung verwandter Systeme

Sei eine Zahl Z aus dem System zur Basis a in das System zur Basis b (mit $a, b \in \mathbb{N}$) zu kodieren und es gilt:

- $a = b^n$ mit $n \in \mathbb{N}$.
  Dann wird jede Stelle von Z n-äquidistant zerteilt (sprich: in n-Stellen (zur Basis b) zerlegt).
- $a^n = b$ mit $n \in \mathbb{N}$.
  Dann werden jeweils n Stellen zu einer neuen Stelle (der Basis b) zusammengefasst.

Aufgabe 3 – Begriffsklärung

Umwandlung verwandter Systeme

Sei eine Zahl Z aus dem System zur Basis a in das System zur Basis b (mit $a, b \in \mathbb{N}$) zu kodieren und es gilt:

- $a = b^n$ mit $n \in \mathbb{N}$.
  Dann wird jede Stelle von Z n-äquidistant zerteilt (sprich: in n-Stellen (zur Basis b) zerlegt).
- $a^n = b$ mit $n \in \mathbb{N}$.
  Dann werden jeweils n Stellen zu einer neuen Stelle (der Basis b) zusammengefasst.

Umwandlung Binär $\mapsto$ BCD

Zuerst ist eine Dezimalzahl zu bilden, die dann jeweils stellenweise kodiert wird.

Aufgabe 3 – Begriffsklärung

Umwandlung verwandter Systeme

Sei eine Zahl Z aus dem System zur Basis a in das System zur Basis b (mit $a, b \in \mathbb{N}$) zu kodieren und es gilt:

- $a = b^n$ mit $n \in \mathbb{N}$.
  Dann wird jede Stelle von Z n-äquidistant zerteilt (sprich: in n-Stellen (zur Basis b) zerlegt).
- $a^n = b$ mit $n \in \mathbb{N}$.
  Dann werden jeweils n Stellen zu einer neuen Stelle (der Basis b) zusammengefasst.

Umwandlung Binär $\mapsto$ BCD

Zuerst ist eine Dezimalzahl zu bilden, die dann jeweils stellenweise kodiert wird.

Umwandlung Binär $\mapsto$ Dezimal

Für jede vorzeichenlose Binärzahl Z mit n Stellen z_i gilt:

$$Z = \sum_{i=0}^{n-1} z_i \cdot 2^i$$

Aufgabe 4 – Zahlenkonversion

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 4 – Zahlenkonversion

a) Konvertieren Sie die Hexadezimalzahl $\mathrm{A03}_{16}$ mit sukzessiver Division unter ausschließlicher Verwendung der angegebenen Zahlensysteme ins Binär- bzw. Ternärsystem.

b) Konvertieren Sie die Binärzahl $1110\,0111_2$ unter ausschließlicher Verwendung der angegebenen Zahlensysteme ins Oktal- bzw. Ternärsystem.

c) Konvertieren Sie die Dezimalzahl $234{,}28125_{10}$ ins Binärformat. Verwenden Sie für die Nachkommastellen maximal 4 Bit.

Aufgabe 4 – Begriffsklärung

Umwandlung verwandter Systeme

Sei eine Zahl Z aus dem System zur Basis a in das System zur Basis b (mit $a, b \in \mathbb{N}$) zu kodieren und es gilt:

- $a = b^n$ mit $n \in \mathbb{N}$.
  Dann wird jede Stelle von Z n-äquidistant zerteilt (sprich: in n-Stellen (zur Basis b) zerlegt).
- $a^n = b$ mit $n \in \mathbb{N}$.
  Dann werden jeweils n Stellen zu einer neuen Stelle (der Basis b) zusammengefasst.

Aufgabe 4 – Begriffsklärung

Umwandlung fremder Systeme

Sei eine Zahl Z aus dem System zur Basis a in das System zur Basis b (mit $a, b \in \mathbb{N}$) zu kodieren und die Systeme nicht verwandt (wie in Aufgabe 3). Dann erfolgt die Kodierung mit folgendem Algorithmus:

Aufgabe 4 – Begriffsklärung

Umwandlung fremder Systeme

Sei eine Zahl Z aus dem System zur Basis a in das System zur Basis b (mit $a, b \in \mathbb{N}$) zu kodieren und die Systeme nicht verwandt (wie in Aufgabe 3). Dann erfolgt die Kodierung mit folgendem Algorithmus:

Schritt 1: Dividiere b im System a von Zahl s im System a

Aufgabe 4 – Begriffsklärung

Umwandlung fremder Systeme

Sei eine Zahl Z aus dem System zur Basis a in das System zur Basis b (mit $a, b \in \mathbb{N}$) zu kodieren und die Systeme nicht verwandt (wie in Aufgabe 3). Dann erfolgt die Kodierung mit folgendem Algorithmus:

Schritt 1: Dividiere b im System a von Zahl s im System a

Schritt 2: Merke das Ergebnis $e = \frac{s}{b}$ und den Rest $r = s \mod b$

Aufgabe 4 – Begriffsklärung

Umwandlung fremder Systeme

Sei eine Zahl Z aus dem System zur Basis a in das System zur Basis b (mit $a, b \in \mathbb{N}$) zu kodieren und die Systeme nicht verwandt (wie in Aufgabe 3). Dann erfolgt die Kodierung mit folgendem Algorithmus:

Schritt 1: Dividiere b im System a von Zahl s im System a

Schritt 2: Merke das Ergebnis $e = \frac{s}{b}$ und den Rest $r = s \mod b$

Schritt 3: r repräsentiert eine Stelle des Ergebnisses in System b

Aufgabe 4 – Begriffsklärung

Umwandlung fremder Systeme

Sei eine Zahl Z aus dem System zur Basis a in das System zur Basis b (mit $a, b \in \mathbb{N}$) zu kodieren und die Systeme nicht verwandt (wie in Aufgabe 3). Dann erfolgt die Kodierung mit folgendem Algorithmus:

Schritt 1: Dividiere b im System a von Zahl s im System a

Schritt 2: Merke das Ergebnis $e = \frac{s}{b}$ und den Rest $r = s \mod b$

Schritt 3: r repräsentiert eine Stelle des Ergebnisses in System b

Schritt 4: Ist $e > 0$, so setze b zu e und mache weiter mit Schritt 1.

Aufgabe 4 – Begriffsklärung

Umwandlung fremder Systeme

Sei eine Zahl Z aus dem System zur Basis a in das System zur Basis b (mit $a, b \in \mathbb{N}$) zu kodieren und die Systeme nicht verwandt (wie in Aufgabe 3). Dann erfolgt die Kodierung mit folgendem Algorithmus:

Schritt 1: Dividiere b im System a von Zahl s im System a

Schritt 2: Merke das Ergebnis $e = \frac{s}{b}$ und den Rest $r = s \mod b$

Schritt 3: r repräsentiert eine Stelle des Ergebnisses in System b

Schritt 4: Ist $e > 0$, so setze b zu e und mache weiter mit Schritt 1.

Schritt 5: Lese das Ergebnis rückwärts aus

Aufgabe 4 – Zahlenkonversion

a) Konvertieren Sie die Hexadezimalzahl $\mathrm{A03}_{16}$ mit sukzessiver Division unter ausschließlicher Verwendung der angegebenen Zahlensysteme ins Binär- bzw. Ternärsystem.

Aufgabe 4 – Zahlenkonversion

b) Konvertieren Sie die Binärzahl $1110\,0111_2$ unter ausschließlicher Verwendung der angegebenen Zahlensysteme ins Oktal- bzw. Ternärsystem.

Aufgabe 4 – Zahlenkonversion

c) Konvertieren Sie die Dezimalzahl $234{,}28125_{10}$ ins Binärformat. Verwenden Sie für die Nachkommastellen maximal 4 Bit.

Aufgabe 4 – Begriffsklärung

Festkommazahl

Eine Festkommazahl ist eine Zahl, die aus einer festen Anzahl von Ziffern besteht. Die Position des Kommas ist dabei **fest** vorgegeben, daher der Name.

Aufgabe 4 – Begriffsklärung

Festkommazahl

Eine Festkommazahl ist eine Zahl, die aus einer festen Anzahl von Ziffern besteht. Die Position des Kommas ist dabei **fest** vorgegeben, daher der Name.
Dabei wird das polyadische System einfach fortgeführt, es gilt also bei einer n-stelligen B-adischen Festkommazahl Z mit k Nachkommastellen:

Aufgabe 4 – Begriffsklärung

Festkommazahl

Eine Festkommazahl ist eine Zahl, die aus einer festen Anzahl von Ziffern besteht. Die Position des Kommas ist dabei **fest** vorgegeben, daher der Name.
Dabei wird das polyadische System einfach fortgeführt, es gilt also bei einer n-stelligen B-adischen Festkommazahl Z mit k Nachkommastellen:

$$Z = \sum_{i=1}^{k} z_i \cdot B^{-i} + \sum_{i=0}^{(n-1)-k} z_i \cdot B^{i} = \sum_{i=-k}^{(n-1)-k} z_i \cdot B^{i}$$

Aufgabe 4 – Begriffsklärung

Umwandlung einer rationalen Zahl in eine Festkommazahl

Sei eine Zahl $(z, n) \in \mathbb{Q}$ gegeben, wobei $z \in \mathbb{Z}, n \in \mathbb{N}$. Diese soll nun als binäre Festkommazahl mit k Nachkommastellen dargestellt werden.

Aufgabe 4 – Begriffsklärung

Umwandlung einer rationalen Zahl in eine FESTKOMMAzahl

Sei eine Zahl $(z, n) \in \mathbb{Q}$ gegeben, wobei $z \in \mathbb{Z}, n \in \mathbb{N}$. Diese soll nun als binäre Festkommazahl mit k Nachkommastellen dargestellt werden.
Dann wandle zuerst die Zahl z um

Aufgabe 4 – Begriffsklärung

Umwandlung einer rationalen Zahl in eine Festkommazahl

Sei eine Zahl $(z, n) \in \mathbb{Q}$ gegeben, wobei $z \in \mathbb{Z}, n \in \mathbb{N}$. Diese soll nun als binäre Festkommazahl mit k Nachkommastellen dargestellt werden.
Dann wandle zuerst die Zahl z um und danach die Nachkommazahl n mit dem binären Verdopplungsverfahren.

Aufgabe 4 – Begriffsklärung

Umwandlung einer rationalen Zahl in eine Festkommazahl

Sei eine Zahl $(z, n) \in \mathbb{Q}$ gegeben, wobei $z \in \mathbb{Z}, n \in \mathbb{N}$. Diese soll nun als binäre Festkommazahl mit k Nachkommastellen dargestellt werden.
Dann wandle zuerst die Zahl z um und danach die Nachkommazahl n mit dem binären Verdopplungsverfahren.

Binäres Verdopplungsverfahren

Sei eine Zahl $(z, n) \in \mathbb{Q}$ gegeben, wobei $z \in \mathbb{Z}, n \in \mathbb{N}$. Betrachte nun nur n:

Aufgabe 4 – Begriffsklärung

Umwandlung einer rationalen Zahl in eine FESTKOMMAzahl

Sei eine Zahl $(z, n) \in \mathbb{Q}$ gegeben, wobei $z \in \mathbb{Z}, n \in \mathbb{N}$. Diese soll nun als binäre Festkommazahl mit k Nachkommastellen dargestellt werden.
Dann wandle zuerst die Zahl z um und danach die Nachkommazahl n mit dem binären Verdopplungsverfahren.

Binäres Verdopplungsverfahren

Sei eine Zahl $(z, n) \in \mathbb{Q}$ gegeben, wobei $z \in \mathbb{Z}, n \in \mathbb{N}$. Betrachte nun nur n:

Schritt 1: Multipliziere n mit 2, merke das Ergebnis e.

Aufgabe 4 – Begriffsklärung

Umwandlung einer rationalen Zahl in eine FESTKOMMAzahl

Sei eine Zahl $(z, n) \in \mathbb{Q}$ gegeben, wobei $z \in \mathbb{Z}, n \in \mathbb{N}$. Diese soll nun als binäre Festkommazahl mit k Nachkommastellen dargestellt werden.
Dann wandle zuerst die Zahl z um und danach die Nachkommazahl n mit dem binären Verdopplungsverfahren.

Binäres Verdopplungsverfahren

Sei eine Zahl $(z, n) \in \mathbb{Q}$ gegeben, wobei $z \in \mathbb{Z}, n \in \mathbb{N}$. Betrachte nun nur n:

Schritt 1: Multipliziere n mit 2, merke das Ergebnis e.

Schritt 2: Ist $e > 1$, so setze e^* auf $e - 1$, notiere binär eine 1.

Aufgabe 4 – Begriffsklärung

Umwandlung einer rationalen Zahl in eine FESTKOMMAzahl

Sei eine Zahl $(z, n) \in \mathbb{Q}$ gegeben, wobei $z \in \mathbb{Z}, n \in \mathbb{N}$. Diese soll nun als binäre Festkommazahl mit k Nachkommastellen dargestellt werden.
Dann wandle zuerst die Zahl z um und danach die Nachkommazahl n mit dem binären Verdopplungsverfahren.

Binäres Verdopplungsverfahren

Sei eine Zahl $(z, n) \in \mathbb{Q}$ gegeben, wobei $z \in \mathbb{Z}, n \in \mathbb{N}$. Betrachte nun nur n:

Schritt 1: Multipliziere n mit 2, merke das Ergebnis e.

Schritt 2: Ist $e > 1$, so setze e^* auf $e - 1$, notiere binär eine 1.

Schritt 3: Ist $e < 1$, so notiere binär eine 0.

Aufgabe 4 – Begriffsklärung

Umwandlung einer rationalen Zahl in eine FESTKOMMAzahl

Sei eine Zahl $(z, n) \in \mathbb{Q}$ gegeben, wobei $z \in \mathbb{Z}, n \in \mathbb{N}$. Diese soll nun als binäre Festkommazahl mit k Nachkommastellen dargestellt werden.
Dann wandle zuerst die Zahl z um und danach die Nachkommazahl n mit dem binären Verdopplungsverfahren.

Binäres Verdopplungsverfahren

Sei eine Zahl $(z, n) \in \mathbb{Q}$ gegeben, wobei $z \in \mathbb{Z}, n \in \mathbb{N}$. Betrachte nun nur n:

Schritt 1: Multipliziere n mit 2, merke das Ergebnis e.

Schritt 2: Ist $e > 1$, so setze e^* auf $e - 1$, notiere binär eine 1.

Schritt 3: Ist $e < 1$, so notiere binär eine 0.

Schritt 4: Ist die Anzahl an notierten Stellen identisch mit k oder $e = 1$, so breche ab.

Aufgabe 4 – Begriffsklärung

Umwandlung einer rationalen Zahl in eine FESTKOMMAzahl

Sei eine Zahl $(z, n) \in \mathbb{Q}$ gegeben, wobei $z \in \mathbb{Z}, n \in \mathbb{N}$. Diese soll nun als binäre Festkommazahl mit k Nachkommastellen dargestellt werden.
Dann wandle zuerst die Zahl z um und danach die Nachkommazahl n mit dem binären Verdopplungsverfahren.

Binäres Verdopplungsverfahren

Sei eine Zahl $(z, n) \in \mathbb{Q}$ gegeben, wobei $z \in \mathbb{Z}, n \in \mathbb{N}$. Betrachte nun nur n:

Schritt 1: Multipliziere n mit 2, merke das Ergebnis e.

Schritt 2: Ist $e > 1$, so setze e^* auf $e - 1$, notiere binär eine 1.

Schritt 3: Ist $e < 1$, so notiere binär eine 0.

Schritt 4: Ist die Anzahl an notierten Stellen identisch mit k oder $e = 1$, so breche ab.

Schritt 5: Sonst wiederhole Schritt 1.

Aufgabe 4 – Zahlenkonversion

c) Konvertieren Sie die Dezimalzahl $234,28125_{10}$ ins Binärformat.
Verwenden Sie für die Nachkommastellen maximal 4 Bit.

Aufgabe 4 – Zahlenkonversion: Festkommazahlen

Unstabile Polygone bei der PSX:
https://www.youtube.com/watch?v=nqw2HMUrNiA

Aufgabe 4 – Zahlenkonversion: Festkommazahlen

Unstabile Polygone bei der PSX: https://www.youtube.com/watch?v=nqw2HMUrNiA

Was ist das Problem?

... The real problem is that the PS1 didn't have a FPU; a co-processor for math dealing with real numbers called a Floating Point Unit, so consequently it used fixed point math that has limited precision [...]. The limited precision causes the polygon vertices themselves to jump around as the camera moves around the scene because there isn't enough bits to finely position the vertex ...

(Kommentar unter dem Video)

Aufgabe 5 – Konversion von Gleitkommazahlen

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 5 – Konversion von Gleitkommazahlen

Das Format für Gleitkommazahlen im IEEE-Standard 754 einfacher Genauigkeit lautet:

V 31	E (8) 30 … 23	M (23) 22 … 0

a) Konvertieren Sie die folgenden nach obigem Standard codierten Zahlen in das Dezimalsystem:

i) 0 1001 1001 1001 1001 1001 0000 0000 000

i) 1 0001 1001 1001 1001 0000 0000 0000 000

b) Wandeln Sie die folgenden Zahlen in den obigen IEEE-Standard um:

ii) $-6,25_{10} \cdot 10^{-3} = -0,00000\overline{0011}_2$

ii) $3,14159_{10} = 11,0010\,0100\,0011\,1111\,0011\,11(1\ldots)_2$

c) Warum kann einer `float`-Variablen der Wert $1 \cdot 10^{-42}$, nicht aber der Wert $1 \cdot 10^{42}$ zugewiesen werden?

Aufgabe 5 – Begriffsklärung

IEEE

Das **Institute of Electrical and Electronics Engineers** (**IEEE**) ist ein globaler Verband von Ingenieuren – haupsächlich aus – der Elektro- und Informationstechnik. Mit ihr kommen verschiedene Gremien zur Standardisierung von Techniken, Hardware und Software zusammen.

Aufgabe 5 – Begriffsklärung

V 31	E (8) 30 … 23	M (23) 22 … 0

Gleitkommaarithmetik

Wie kann man eine Gleitkommazahl berechnen?
Es gilt:

$$Z = (-1)^V \cdot 2^{(E-B)} \cdot (1 + M)$$

- V beschreibt das Vorzeichenbit. Ist $V = 0$, so ist Z positiv, sonst negativ.
- E beschreibt den *biased exponent*[a]. Sie berechnet zusammen mit dem BIAS B den realen Exponenten.
- B beschreibt den BIAS. Er ist für Gleitkommazahlen fest und berechnet sich durch $B = 2^{|E|-1} - 1$. Mit dem BIAS ist es möglich negaive Exponenten darzustellen.
- M beschreibt die Nachkommastellen der Mantisse. Man berechnet die reale Mantisse mit $1 + M$.

[a] auch Charakteristik genannt

Aufgabe 5 – Begriffsklärung

V	E (8)	M (23)
31	30 … 23	22 … 0

Gleitkommaarithmetik – Sonderfälle

Wie kann man eine Gleitkommazahl berechnen?
Es gilt:

$$Z = (-1)^V \cdot 2^{(E-B)} \cdot (1 + M)$$

Allerdings ist bei der Berechnung von Gleitkommazahlen auf einige Sonderfälle zu achten:

E	M	Wert
$0 < E < 2^{\lvert E\rvert} - 1$	M	$(-1)^V \cdot 2^{(E-B)} \cdot (1 + M)$
$2^{\lvert E\rvert} - 1$	$\neq 0$	NaN (Not a Number)
	0	$\pm\infty$ (je nach Vorzeichenbit)
0	M	Denormalisierte Zahlen

Warum die Fehlerbehandlung wichtig sein kann ...

Aufgabe 5 – Konversion von Gleitkommazahlen

Das Format für Gleitkommazahlen im IEEE-Standard 754 einfacher Genauigkeit lautet:

V	E (8)		M (23)	
31	30	23	22	0

a) Konvertieren Sie die folgenden nach obigem Standard codierten Zahlen in das Dezimalsystem:
   i) 0 1001 1001 1001 1001 1001 0000 0000 000
   i) 1 0001 1001 1001 1001 0000 0000 0000 000

Aufgabe 5 – Konversion von Gleitkommazahlen

Das Format für Gleitkommazahlen im IEEE-Standard 754 einfacher Genauigkeit lautet:

V	E (8)	M (23)
31	30 … 23	22 … 0

b) Wandeln Sie die folgenden Zahlen in den obigen IEEE-Standard um:

ii) $-6,25_{10} \cdot 10^{-3} = -0,00000\overline{0011}_2$

ii) $3,14159_{10} = 11,0010\,0100\,0011\,1111\,0011\,11(1\ldots)_2$

Aufgabe 5 – Algorithmus Dezimal $\mapsto$ IEEE

Sei die Dezimalzahl d_{10} gegeben:

Aufgabe 5 – Algorithmus Dezimal $\mapsto$ IEEE

Sei die Dezimalzahl d_{10} gegeben:

Schritt 1) Wandle d_{10} ins Binärsystem um

Aufgabe 5 – Algorithmus Dezimal $\mapsto$ IEEE

Sei die Dezimalzahl d_{10} gegeben:

Schritt 1) Wandle d_{10} ins Binärsystem um

Schritt 2) Normalisiere auf $(1, M)_2 \cdot 2^E$ und runde auf $\text{len}(M)$ bit Nachkommastellen.

Aufgabe 5 – Algorithmus Dezimal $\mapsto$ IEEE

Sei die Dezimalzahl d_{10} gegeben:

Schritt 1) Wandle d_{10} ins Binärsystem um

Schritt 2) Normalisiere auf $(1, M)_2 \cdot 2^E$ und runde auf len(M) bit Nachkommastellen.

Schritt 3) Bestimme den *biased exponent*, den „voreingenommenen“ Exponenten.

Aufgabe 5 – Algorithmus Dezimal $\mapsto$ IEEE

Sei die Dezimalzahl d_{10} gegeben:

Schritt 1) Wandle d_{10} ins Binärsystem um

Schritt 2) Normalisiere auf $(1, M)_2 \cdot 2^E$ und runde auf $\text{len}(M)$ bit Nachkommastellen.

Schritt 3) Bestimme den *biased exponent*, den „voreingenommenen“ Exponenten.

Schritt 4) Bestimme je nach Vorzeichen das Vorzeichenbit V und setze die Zahl zusammen.

Aufgabe 5 – Konversion von Gleitkommazahlen

c) Warum kann einer `float`-Variablen der Wert $1 \cdot 10^{-42}$, nicht aber der Wert $1 \cdot 10^{42}$ zugewiesen werden?

```java
//...
public static void main(String[] args) {
    DecimalIEEE dI = new DecimalIEEE((float) 1e-42); dI.printString();
    DecimalIEEE dI2 = new DecimalIEEE((float) 1e42); dI2.printString();
}
//...
```

Aufgabe 5 – Konversion von Gleitkommazahlen

c) Warum kann einer `float`-Variablen der Wert $1 \cdot 10^{-42}$, nicht aber der Wert $1 \cdot 10^{42}$ zugewiesen werden?

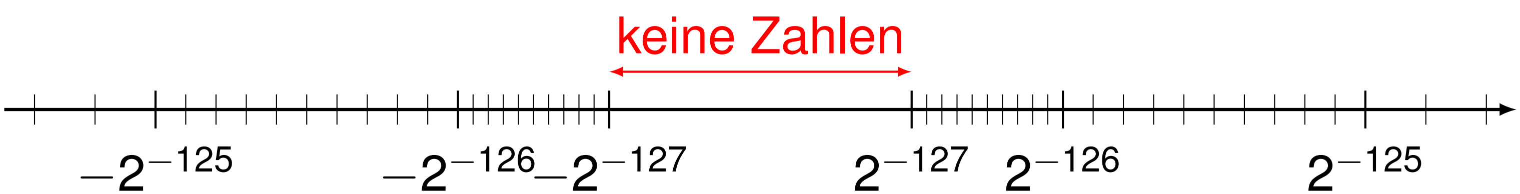
```
//...
public static void main(String[] args) {
    DecimalIEEE dI = new DecimalIEEE((float) 1e-42); dI.printString();
    DecimalIEEE dI2 = new DecimalIEEE((float) 1e42); dI2.printString();
}
//...
```

Ausgabe

```
usercca:~/gti$java DecimalIEEE
DecimalIEEE -- 00000000000000000000001011001010
    +1.0E-42
DecimalIEEE -- 01111111100000000000000000000000
    +Infinity
```

IEEE-Standard 754: Normalisierte Darstellung

V	E (8)	M (23)
31	30 … 23	22 … 0

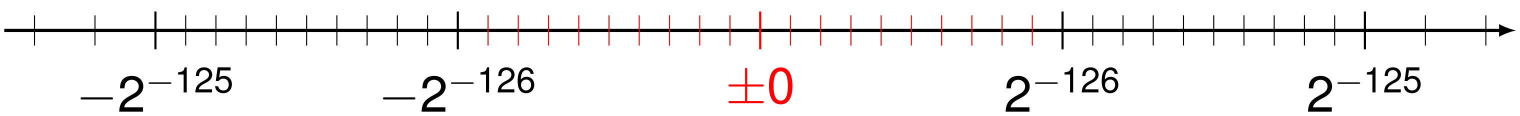


$$Z = (-1)^V \cdot 2^{(E-B)} \cdot (1 + M)$$

IEEE-Standard 754: Denormalisierte Darstellung

V	E (8)		M (23)	
31	30	23	22	0

$\rightsquigarrow$ $E = 0$ heißt die Zahl ist denormalisiert.



$$Z = (-1)^V \cdot 2^{-126} \cdot (0 + M)$$

Übungen zur Grundlagen der Technischen Informatik

Übung 4 – Binär-, Hexadezimal- und Gleitkommaarithmetik

Florian Frank

Friedrich-Alexander-Universität Erlangen-Nürnberg

Wintersemester 2018/19

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Was machen wir heute?

Prüfungsanmeldung

Was machen wir heute?

Prüfungsanmeldung

Aufgabe 1 – Binär- und Hexadezimalarithmetik

Was machen wir heute?

Prüfungsanmeldung

Aufgabe 1 – Binär- und Hexadezimalarithmetik

Blatt 3, Aufgabe 5 – Konversion von Gleitkommazahlen

Was machen wir heute?

Prüfungsanmeldung

Aufgabe 1 – Binär- und Hexadezimalarithmetik

Blatt 3, Aufgabe 5 – Konversion von Gleitkommazahlen

Aufgabe 2 – Addition und Subtraktion von Gleitkommazahlen

Was machen wir heute?

Prüfungsanmeldung

Aufgabe 1 – Binär- und Hexadezimalarithmetik

Blatt 3, Aufgabe 5 – Konversion von Gleitkommazahlen

Aufgabe 2 – Addition und Subtraktion von Gleitkommazahlen

Aufgabe 3 – Multiplikation von Gleitkommazahlen

Was machen wir heute?

Prüfungsanmeldung

Aufgabe 1 – Binär- und Hexadezimalarithmetik

Blatt 3, Aufgabe 5 – Konversion von Gleitkommazahlen

Aufgabe 2 – Addition und Subtraktion von Gleitkommazahlen

Aufgabe 3 – Multiplikation von Gleitkommazahlen

Aufgabe 4 – Assoziativität bei Gleitkommadarstellung

Prüfungsanmeldung

Achtung – Prüfungsanmeldung

Prüfungsanmeldung

Im Wintersemester 2018/19 findet die Anmeldung zu den Prüfungen des Semesters ab dem 19. November 2018, 0:01 Uhr über „mein campus“ statt. Die Prüfungsanmeldephase endet am 07. Dezember um 12:00 Uhr.
Ihr müsst euch für die Prüfungen anmelden, an denen ihr teilnehmen wollt.
Nach dem 07. Dezember um 12:00 Uhr ist **keine** Anmeldung mehr möglich!

Achtung – Prüfungsanmeldung

Prüfungsanmeldung

Im Wintersemester 2018/19 findet die Anmeldung zu den Prüfungen des Semesters ab dem 19. November 2018, 0:01 Uhr über „mein campus“ statt. Die Prüfungsanmeldephase endet am 07. Dezember um 12:00 Uhr.
Ihr müsst euch für die Prüfungen anmelden, an denen ihr teilnehmen wollt.
Nach dem 07. Dezember um 12:00 Uhr ist **keine** Anmeldung mehr möglich!

Achtung – Prüfungsanmeldung

Prüfungsanmeldung

Im Wintersemester 2018/19 findet die Anmeldung zu den Prüfungen des Semesters ab dem 19. November 2018, 0:01 Uhr über „mein campus“ statt. Die Prüfungsanmeldephase endet am 07. Dezember um 12:00 Uhr.
Ihr müsst euch für die Prüfungen anmelden, an denen ihr teilnehmen wollt.
Nach dem 07. Dezember um 12:00 Uhr ist **keine** Anmeldung mehr möglich!

Achtung – Prüfungsanmeldung

Prüfungsanmeldung

Im Wintersemester 2018/19 findet die Anmeldung zu den Prüfungen des Semesters ab dem 19. November 2018, 0:01 Uhr über „mein campus“ statt. Die Prüfungsanmeldephase endet am 07. Dezember um 12:00 Uhr.

Ihr müsst euch für die Prüfungen anmelden, an denen ihr teilnehmen wollt.

Nach dem 07. Dezember um 12:00 Uhr ist **keine** Anmeldung mehr möglich!

Prüfungsan- und abmeldung

Wichtige Informationen zur Prüfungsanmeldung. Bitte sorgfältig lesen!

Bitte beachten Sie:

Sie sind verpflichtet, die ordnungsgemäße Erfassung Ihrer An- oder Abmeldung rechtzeitig vor den Prüfungen durch Einsichtnahme in *mein campus* zu kontrollieren. Setzen Sie sich bei Unstimmigkeiten bitte sofort mit dem Prüfungsamt in Verbindung und legen Sie den Ausdruck über die angemeldeten Prüfungen vor.

Die Anmeldung zu den einzelnen Modulprüfungen oder Modulteilprüfungen erfolgt gegebenenfalls unter dem Vorbehalt einer endgültig verabschiedeten Prüfungsordnung.

Die Anmeldungen zu Modulen, die als Schlüsselqualifikationen gewertet werden sollen, erfolgt seit diesem Semester ebenfalls über *mein campus*.
Nicht bestandene Prüfungen sind innerhalb der in der geltenden Prüfungsordnung genannten Fristen zu wiederholen.
Die Anmeldung zur Wiederholungsprüfung ist zu kontrollieren. Soweit eine Anmeldung für die Wiederholungsprüfung nicht eingetragen ist und Sie sich auch nicht selbst für diese Prüfung über *mein campus* anmelden können, verständigen Sie bitte das Prüfungsamt.
Für Wiederholungen gelten die Vorschriften der jeweiligen Prüfungsordnung.

Hiermit erkläre ich, dass ich in dem oder einem verwandten Studiengang, für den Prüfungen angemeldet werden (Diplom-, Bachelor- oder Master- Studium), noch keine Prüfung endgültig nicht bestanden habe bzw. unter Verlust des Prüfungsanspruches exmatrikuliert wurde. Soweit ich mich in einem schwebenden Prüfungsverfahren an einer anderen Hochschule befinde, lege ich umgehend eine Bestätigung der bisherigen Hochschule im Prüfungsamt vor. Bis zu diesem Zeitpunkt erfolgt die Anmeldung bzw. Zulassung zu Prüfungen unter Vorbehalt.

Von den Bestimmungen der Prüfungsordnungen habe ich Kenntnis genommen.

Erst wenn Sie diesen Hinweis durch Anklicken des unten stehenden Feldes akzeptiert haben, können Sie mit Ihren gewünschten Aktionen fortfahren.
Klicken Sie dazu mit der linken Maustaste auf den "Weiter"-Button und wählen anschließend den entsprechenden Link aus.

Ich akzeptiere die obigen Bedingungen. – Bitte bestätigen

« Zurück zur Übersicht | Weiter »

Achtung – Prüfungsanmeldung

Prüfungsanmeldung

Im Wintersemester 2018/19 findet die Anmeldung zu den Prüfungen des Semesters ab dem 19. November 2018, 0:01 Uhr über „mein campus“ statt. Die Prüfungsanmeldephase endet am 07. Dezember um 12:00 Uhr.
Ihr müsst euch für die Prüfungen anmelden, an denen ihr teilnehmen wollt.
Nach dem 07. Dezember um 12:00 Uhr ist **keine** Anmeldung mehr möglich!

Von den Bestimmungen der Prüfungsordnungen habe ich Kenntnis genommen.

Erst wenn Sie diesen Hinweis durch Anklicken des unten stehenden Feldes akzeptiert haben, können Sie mit Ihren gewünschten Aktionen fortfahren.
Klicken Sie dazu mit der linken Maustaste auf den "Weiter"-Button und wählen anschließend den entsprechenden Link aus.

Ich akzeptiere die obigen Bedingungen. – Bitte bestätigen

« Zurück zur Übersicht | Weiter »

Achtung – Prüfungsanmeldung

Prüfungsanmeldung

Im Wintersemester 2018/19 findet die Anmeldung zu den Prüfungen des Semesters ab dem 19. November 2018, 0:01 Uhr über „mein campus“ statt. Die Prüfungsanmeldephase endet am 07. Dezember um 12:00 Uhr.

Ihr müsst euch für die Prüfungen anmelden, an denen ihr teilnehmen wollt.

Nach dem 07. Dezember um 12:00 Uhr ist **keine** Anmeldung mehr möglich!

3020 Implementierung von Datenbanksystemen

TEC 30201 Implementierung von Datenbanksystemen (ECTS: 5.0)

Datum: 16.02.2018, Prüfer: Richard Lenz, **Prüfung anmelden**

Aufgabe 1 – Binär- und Hexadezimalarithmetik

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Aufgabe 1 – Binär- und Hexadezimalarithmetik

Führen Sie die folgenden Berechnungen im angegebenen Zahlensystem aus, ohne die Zahlen ins Dezimalsystem umzuwandeln.

a) Addition im Binär- und Hexadezimalsystem:

$$111010100110_2 + 010101111110_2$$

$$\mathrm{B674FC12}_{16} + \mathrm{2DA9D4B2}_{16}$$

Aufgabe 1 – Binär- und Hexadezimalarithmetik

Wie können wir zwei Zahlen in einem fremdem System zur Basis *a* addieren?
Erinnerung, wie haben wir das Addieren in der Grundschule gelernt?

$$
\begin{array}{rr}
 & 5\,9 \\
+ & 6\,2 \\
\hline
= &
\end{array}
$$

Aufgabe 1 – Binär- und Hexadezimalarithmetik

Wie können wir zwei Zahlen in einem fremdem System zur Basis *a* addieren?
Erinnerung, wie haben wir das Addieren in der Grundschule gelernt?

$$
\begin{array}{rcc}
 & 5 & 9 \\
+ & 6 & 2 \\
+ & 1 & \\
\hline
= & & 1
\end{array}
$$

Aufgabe 1 – Binär- und Hexadezimalarithmetik

Wie können wir zwei Zahlen in einem fremdem System zur Basis *a* addieren?
Erinnerung, wie haben wir das Addieren in der Grundschule gelernt?

$$
\begin{array}{rcc}
 & 5 & 9 \\
+ & 6 & 2 \\
{\scriptstyle +} \; {\scriptstyle 1} & {\scriptstyle 1} & \\
\hline
= & 2 & 1
\end{array}
$$

Aufgabe 1 – Binär- und Hexadezimalarithmetik

Wie können wir zwei Zahlen in einem fremdem System zur Basis *a* addieren?
Erinnerung, wie haben wir das Addieren in der Grundschule gelernt?

$$
\begin{array}{rrrr}
 & & 5 & 9 \\
+ & & 6 & 2 \\
\scriptstyle + & \scriptstyle & \scriptstyle 1 & \scriptstyle 1 \\
\hline
= & 1 & 2 & 1
\end{array}
$$

Aufgabe 1 – Binär- und Hexadezimalarithmetik

Wie können wir zwei Zahlen in einem fremdem System zur Basis *a* addieren?
Erinnerung, wie haben wir das Addieren in der Grundschule gelernt?

$$
\begin{array}{rccc}
 & & 5 & 9 \\
+ & & 6 & 2 \\
{\scriptstyle +} & {\scriptstyle 1} & {\scriptstyle 1} & \\
\hline
= & 1 & 2 & 1
\end{array}
$$

(1) Addiere jeweils **stellenweise**!

Aufgabe 1 – Binär- und Hexadezimalarithmetik

Wie können wir zwei Zahlen in einem fremdem System zur Basis *a* addieren?
Erinnerung, wie haben wir das Addieren in der Grundschule gelernt?

$$
\begin{array}{rrrr}
 & & 5 & 9 \\
+ & & 6 & 2 \\
{\scriptstyle +} & {\scriptstyle 1} & {\scriptstyle 1} & \\
\hline
= & 1 & 2 & 1
\end{array}
$$

(1) Addiere jeweils **stellenweise**!
Nimm auch notfalls die Hände dazu ...

Aufgabe 1 – Binär- und Hexadezimalarithmetik

Wie können wir zwei Zahlen in einem fremdem System zur Basis *a* addieren?
Erinnerung, wie haben wir das Addieren in der Grundschule gelernt?

$$
\begin{array}{rccc}
 & & 5 & 9 \\
+ & & 6 & 2 \\
{\scriptstyle +} & {\scriptstyle 1} & {\scriptstyle 1} & \\
\hline
= & 1 & 2 & 1
\end{array}
$$

(1) Addiere jeweils **stellenweise**!
Nimm auch notfalls die Hände dazu ...

(2) Hat das Ergebnis der Addition **mehrere** Stellen, so schreibe nur die letzte Stelle nieder!

Aufgabe 1 – Binär- und Hexadezimalarithmetik

Wie können wir zwei Zahlen in einem fremdem System zur Basis *a* addieren?
Erinnerung, wie haben wir das Addieren in der Grundschule gelernt?

$$
\begin{array}{rrrr}
 & & 5 & 9 \\
+ & & 6 & 2 \\
{\scriptstyle +} & {\scriptstyle 1} & {\scriptstyle 1} & \\
\hline
= & 1 & 2 & 1
\end{array}
$$

(1) Addiere jeweils **stellenweise**!
Nimm auch notfalls die Hände dazu ...

(2) Hat das Ergebnis der Addition **mehrere** Stellen, so schreibe nur die letzte Stelle nieder!

(3) Der Rest *e* wandert eine Stelle weiter als Übertrag (*carry*)

Aufgabe 1 – Binär- und Hexadezimalarithmetik

Wie können wir zwei Zahlen in einem fremdem System zur Basis *a* addieren?
Erinnerung, wie haben wir das Addieren in der Grundschule gelernt?

$$
\begin{array}{rrrr}
 & & 5 & 9 \\
+ & & 6 & 2 \\
{\scriptstyle +} & {\scriptstyle 1} & {\scriptstyle 1} & \\
\hline
= & 1 & 2 & 1
\end{array}
$$

(1) Addiere jeweils **stellenweise**!
Nimm auch notfalls die Hände dazu ...

(2) Hat das Ergebnis der Addition **mehrere** Stellen, so schreibe nur die letzte Stelle nieder!

(3) Der Rest *e* wandert eine Stelle weiter als Übertrag (*carry*)

(4) Berechne die nächste Stelle dann durch Addition der einzelnen Stellen mit dem Übertrag von der letzten Stelle.

Aufgabe 1 – Binär- und Hexadezimalarithmetik

Wie können wir zwei Zahlen in einem fremdem System zur Basis *a* addieren?
Erinnerung, wie haben wir das Addieren in der Grundschule gelernt?

$$
\begin{array}{rccc}
 & & 5 & 9 \\
+ & & 6 & 2 \\
{\scriptstyle +} & {\scriptstyle 1} & {\scriptstyle 1} & \\
\hline
= & 1 & 2 & 1
\end{array}
$$

(1) Addiere jeweils **stellenweise**!
Nimm auch notfalls die Hände dazu ...

(2) Hat das Ergebnis der Addition **mehrere** Stellen, so schreibe nur die letzte Stelle nieder!

(3) Der Rest *e* wandert eine Stelle weiter als Übertrag (*carry*)

(4) Berechne die nächste Stelle dann durch Addition der einzelnen Stellen mit dem Übertrag von der letzten Stelle.

(5) Wiederhole diesen Prozess, bis man beim Ende der Ziffern angekommen ist.

Aufgabe 1 – Binär- und Hexadezimalarithmetik

Führen Sie die folgenden Berechnungen im angegebenen Zahlensystem aus, ohne die Zahlen ins Dezimalsystem umzuwandeln.

a) Addition im Binär- und Hexadezimalsystem:

$$111010100110_2 + 010101111110_2$$

$$\mathrm{B674FC12}_{16} + \mathrm{2DA9D4B2}_{16}$$

Einschub — BCD-Addition

BCD-Zahl

BCD-Zahl $\equiv$ **B**inary **C**oded **D**igit
Jede Dezimalziffer wird durch vier Binärstellen repräsentiert. Damit ist die Dezimalzahl leicht rekonstruierbar, die Kodierung aber vergleichsweise ineffizient.
Neben den 10 gültigen Repräsentanten gibt es noch 6 sogenannte Pseudotetraden.

Pseudotetraden

Binärdarstellung von nicht dezimalen Ziffern, also 1010 bis 1111 (10 bis 15).

Die Addition im BCD-System funktioniert im Prinzip genau so wie die Addition bei Binärzahlen, jedoch mit folgenden **wesentlichen** Ausnahmen:

- Stellen (je 4 bit) werden getrennt addiert und der Übertrag in die nächste Stelle gezogen werden.
- Pseudotetraden müssen korrigiert werden.
- Ein Überlauf muss korrigiert werden, wenn er nicht durch einen Korrekturschritt verursacht wurde.
- Ein Korrekturschritt ist die Addtion von 6 (0110_{BCD})

Einschub — BCD-Addition: Beispiel

Berechne $775_{10} + 498_{10}$:

		0 1 1 1	0 1 1 1	0 1 0 1	(775_{10})
+		0 1 0 0	1 0 0 1	1 0 0 0	(498_{10})
+		1 1 1 1	1 1 1		
=		1 1 0 0	0 0 0 0	1 1 0 1	Pseudotetraden
+		0 1 1 0		0 1 1 0	Korrektur
+	1	1	1	1	
	1	$\boxed{0\ 0\ 1\ 0}$	$\boxed{0\ 0\ 0\ 1}$	0 0 1 1	Überlauf
+			0 1 1 0		Korrektur
	1	0 0 1 0	0 1 1 1	0 0 1 1	(1273_{10})

Keine Korrektur
Verursacht durch Korrekturschritt

Korrektur
Überlauf

Aufgabe 1 – Binär- und Hexadezimalarithmetik

Führen Sie die folgenden Berechnungen im angegebenen Zahlensystem aus, ohne die Zahlen ins Dezimalsystem umzuwandeln.

b) Subtraktion im Binärsystem:

$$11010010_2 - 10110101_2$$
$$01110110_2 - 10011001_2$$

Aufgabe 1 – Binär- und Hexadezimalarithmetik

Führen Sie die folgenden Berechnungen im angegebenen Zahlensystem aus, ohne die Zahlen ins Dezimalsystem umzuwandeln.

c) Multiplikation im Binärsystem:

$$11101010_2 \cdot 1011_2$$

Aufgabe 1 – Binär- und Hexadezimalarithmetik

Führen Sie die folgenden Berechnungen im angegebenen Zahlensystem aus, ohne die Zahlen ins Dezimalsystem umzuwandeln.

d) Prozessoren arbeiten mit einer endlichen Wortbreite, was den darstellbaren Zahlenbereich einschränkt. Welche Auswirkungen kann dies bei der Subtraktion und Addition haben? Können Sie sich vorstellen, wie diese Probleme in der Praxis gelöst werden?

Blatt 3, Aufgabe 5 – Konversion von Gleitkomma-zahlen

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 5 – Konversion von Gleitkommazahlen

Das Format für Gleitkommazahlen im IEEE-Standard 754 einfacher Genauigkeit lautet:

V 31	E (8) 30 … 23	M (23) 22 … 0

a) Konvertieren Sie die folgenden nach obigem Standard codierten Zahlen in das Dezimalsystem:
   i) 0 1001 1001 1001 1001 1001 0000 0000 000
   i) 1 0001 1001 1001 1001 0000 0000 0000 000

b) Wandeln Sie die folgenden Zahlen in den obigen IEEE-Standard um:
   ii) $-6,25_{10} \cdot 10^{-3} = -0,00000\overline{0011}_2$
   ii) $3,14159_{10} = 11,0010\,0100\,0011\,1111\,0011\,11(1\ldots)_2$

c) Warum kann einer `float`-Variablen der Wert $1 \cdot 10^{-42}$, nicht aber der Wert $1 \cdot 10^{42}$ zugewiesen werden?

Aufgabe 5 – Begriffsklärung

IEEE

Das **Institute of Electrical and Electronics Engineers** (**IEEE**) ist ein globaler Verband von Ingenieuren – haupsächlich aus – der Elektro- und Informationstechnik. Mit ihr kommen verschiedene Gremien zur Standardisierung von Techniken, Hardware und Software zusammen.

Aufgabe 5 – Begriffsklärung

V	E (8)		M (23)	
31	30	23	22	0

Gleitkommaarithmetik

Wie kann man eine Gleitkommazahl berechnen?
Es gilt:

$$Z = (-1)^V \cdot 2^{(E-B)} \cdot (1 + M)$$

- V beschreibt das Vorzeichenbit. Ist $V = 0$, so ist Z positiv, sonst negativ.
- E beschreibt den *biased exponent*[a]. Sie berechnet zusammen mit dem BIAS B den realen Exponenten.
- B beschreibt den BIAS. Er ist für Gleitkommazahlen fest und berechnet sich durch $B = 2^{|E|-1} - 1$. Mit dem BIAS ist es möglich negaive Exponenten darzustellen.
- M beschreibt die Nachkommastellen der Mantisse. Man berechnet die reale Mantisse mit $1 + M$.

[a] auch Charakteristik genannt

Aufgabe 5 – Begriffsklärung

V	E (8)		M (23)	
31	30	23	22	0

Gleitkommaarithmetik – Sonderfälle

Wie kann man eine Gleitkommazahl berechnen?
Es gilt:

$$Z = (-1)^V \cdot 2^{(E-B)} \cdot (1 + M)$$

Allerdings ist bei der Berechnung von Gleitkommazahlen auf einige Sonderfälle zu achten:

E	M	Wert
$0 < E < 2^{\|E\|} - 1$	M	$(-1)^V \cdot 2^{(E-B)} \cdot (1 + M)$
$2^{\|E\|} - 1$	$\neq 0$	NaN (Not a Number)
	0	$\pm\infty$ (je nach Vorzeichenbit)
0	M	Denormalisierte Zahlen

Warum die Fehlerbehandlung wichtig sein kann ...

Aufgabe 5 – Konversion von Gleitkommazahlen

Das Format für Gleitkommazahlen im IEEE-Standard 754 einfacher Genauigkeit lautet:

V	E (8)		M (23)	
31	30	23	22	0

a) Konvertieren Sie die folgenden nach obigem Standard codierten Zahlen in das Dezimalsystem:

   i) 0 1001 1001 1001 1001 1001 0000 0000 000

   i) 1 0001 1001 1001 1001 0000 0000 0000 000

Aufgabe 5 – Konversion von Gleitkommazahlen

Das Format für Gleitkommazahlen im IEEE-Standard 754 einfacher Genauigkeit lautet:

V	E (8)	M (23)
31	30 … 23	22 … 0

b) Wandeln Sie die folgenden Zahlen in den obigen IEEE-Standard um:

ii) $-6,25_{10} \cdot 10^{-3} = -0,00000\overline{0011}_2$

ii) $3,14159_{10} = 11,0010\,0100\,0011\,1111\,0011\,11(1\ldots)_2$

Aufgabe 5 – Algorithmus Dezimal $\mapsto$ IEEE

Sei die Dezimalzahl d_{10} gegeben:

Aufgabe 5 – Algorithmus Dezimal $\mapsto$ IEEE

Sei die Dezimalzahl d_{10} gegeben:

Schritt 1) Wandle d_{10} ins Binärsystem um

Aufgabe 5 – Algorithmus Dezimal $\mapsto$ IEEE

Sei die Dezimalzahl d_{10} gegeben:

Schritt 1) Wandle d_{10} ins Binärsystem um

Schritt 2) Normalisiere auf $(1, M)_2 \cdot 2^E$ und runde auf $\mathrm{len}(M)$ bit Nachkommastellen.

Aufgabe 5 – Algorithmus Dezimal $\mapsto$ IEEE

Sei die Dezimalzahl d_{10} gegeben:

Schritt 1) Wandle d_{10} ins Binärsystem um

Schritt 2) Normalisiere auf $(1,M)_2 \cdot 2^E$ und runde auf $\mathrm{len}(M)$ bit Nachkommastellen.

Schritt 3) Bestimme den *biased exponent*, den „voreingenommenen“ Exponenten.

Aufgabe 5 – Algorithmus Dezimal $\mapsto$ IEEE

Sei die Dezimalzahl d_{10} gegeben:

Schritt 1) Wandle d_{10} ins Binärsystem um

Schritt 2) Normalisiere auf $(1,M)_2 \cdot 2^E$ und runde auf len(M) bit Nachkommastellen.

Schritt 3) Bestimme den *biased exponent*, den „voreingenommenen“ Exponenten.

Schritt 4) Bestimme je nach Vorzeichen das Vorzeichenbit V und setze die Zahl zusammen.

Aufgabe 5 – Konversion von Gleitkommazahlen

c) Warum kann einer `float`-Variablen der Wert $1 \cdot 10^{-42}$, nicht aber der Wert $1 \cdot 10^{42}$ zugewiesen werden?

```java
//...
public static void main(String[] args) {
    DecimalIEEE dI = new DecimalIEEE((float) 1e-42); dI.printString();
    DecimalIEEE dI2 = new DecimalIEEE((float) 1e42); dI2.printString();
}
//...
```

Aufgabe 5 – Konversion von Gleitkommazahlen

c) Warum kann einer `float`-Variablen der Wert $1 \cdot 10^{-42}$, nicht aber der Wert $1 \cdot 10^{42}$ zugewiesen werden?

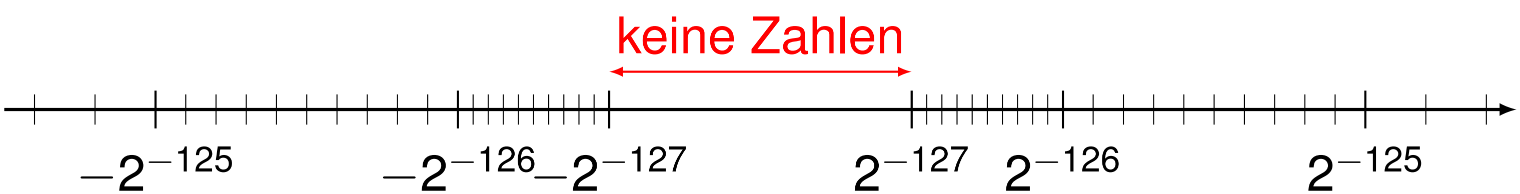
```java
//...
public static void main(String[] args) {
    DecimalIEEE dI = new DecimalIEEE((float) 1e-42); dI.printString();
    DecimalIEEE dI2 = new DecimalIEEE((float) 1e42); dI2.printString();
}
//...
```

Ausgabe

```
usercca:~/gti$java DecimalIEEE
DecimalIEEE -- 00000000000000000000001011001010
    +1.0E-42
DecimalIEEE -- 01111111100000000000000000000000
    +Infinity
```

IEEE-Standard 754: Normalisierte Darstellung

V	E (8)	M (23)
31	30 … 23	22 … 0

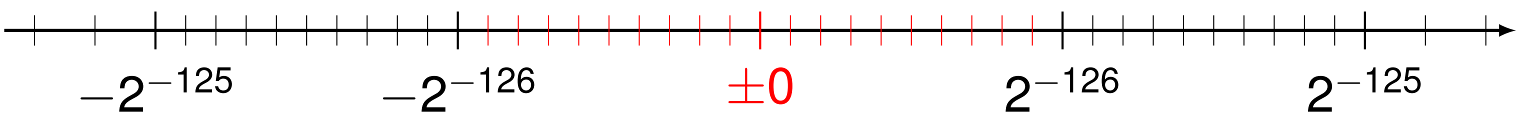


$$Z = (-1)^{V} \cdot 2^{(E-B)} \cdot (1 + M)$$

IEEE-Standard 754: Denormalisierte Darstellung

V	E (8)	M (23)
31	30 … 23	22 … 0

$\rightsquigarrow$ $E = 0$ heißt die Zahl ist denormalisiert.



$$Z = (-1)^V \cdot 2^{-126} \cdot (0 + M)$$

Aufgabe 2 – Addition und Subtraktion von Gleitkommazahlen

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 2 – Addition und Subtraktion von Gleitkommazahlen

Seien folgende Zahlen im IEEE-Standard 754 einfacher Genauigkeit gegeben:

$$x_1 = 0 \;\; 1001\,1001 \;\; 0000\,1001\,1001\,0000\,0000\,000$$
$$x_2 = 0 \;\; 1001\,1001 \;\; 1111\,1111\,0000\,0000\,0000\,000$$
$$x_3 = 1 \;\; 1001\,0100 \;\; 0011\,0110\,0000\,0000\,0000\,000$$

1. Berechnen Sie $x_4 = x_1 + x_2$.
2. Berechnen Sie $x_5 = x_1 + x_3$.

Aufgabe 2 – Add. und Sub. von GKZ – Verfahrensidee

Wie könnte man zwei Zahlen $x_1 = (-1)^{V_1} \cdot 2^{(E_1 - B)} \cdot (M_1)$ und $x_2 = (-1)^{V_2} \cdot 2^{(E_2 - B)} \cdot (M_2)$ addieren?

Aufgabe 2 – Add. und Sub. von GKZ – Verfahrensidee

Wie könnte man zwei Zahlen $x_1 = (-1)^{V_1} \cdot 2^{(E_1 - B)} \cdot (M_1)$ und $x_2 = (-1)^{V_2} \cdot 2^{(E_2 - B)} \cdot (M_2)$ addieren?

(1) Wir „machen" die Exponenten „gleich".
Dazu passe durch Rechtsschieben den Exponenten der kleineren Zahl auf den Exponenten der größeren an.

Aufgabe 2 – Add. und Sub. von GKZ – Verfahrensidee

Wie könnte man zwei Zahlen $x_1 = (-1)^{V_1} \cdot 2^{(E_1-B)} \cdot (M_1)$ und $x_2 = (-1)^{V_2} \cdot 2^{(E_2-B)} \cdot (M_2)$ addieren?

(1) Wir „machen“ die Exponenten „gleich“.
Dazu passe durch Rechtsschieben den Exponenten der kleineren Zahl auf den Exponenten der größeren an.

(2) Wir addieren/subtrahieren die Mantissen ($\rightsquigarrow$ bilde dazu – falls notwendig – das Zweierkomplement).
Ist das Ergebnis < 0, setze das Vorzeichen bit des Ergebnis und bilde das Zweierkomplement.

Aufgabe 2 – Add. und Sub. von GKZ – Verfahrensidee

Wie könnte man zwei Zahlen $x_1 = (-1)^{V_1} \cdot 2^{(E_1 - B)} \cdot (M_1)$ und $x_2 = (-1)^{V_2} \cdot 2^{(E_2 - B)} \cdot (M_2)$ addieren?

(1) Wir „machen“ die Exponenten „gleich“.
Dazu passe durch Rechtsschieben den Exponenten der kleineren Zahl auf den Exponenten der größeren an.

(2) Wir addieren/subtrahieren die Mantissen ($\rightsquigarrow$ bilde dazu – falls notwendig – das Zweierkomplement).
Ist das Ergebnis < 0, setze das Vorzeichen bit des Ergebnis und bilde das Zweierkomplement.

(3) Normalisiere das Ergebnis. Dazu:

Aufgabe 2 – Add. und Sub. von GKZ – Verfahrensidee

Wie könnte man zwei Zahlen $x_1 = (-1)^{V_1} \cdot 2^{(E_1 - B)} \cdot (M_1)$ und $x_2 = (-1)^{V_2} \cdot 2^{(E_2 - B)} \cdot (M_2)$ addieren?

(1) Wir „machen“ die Exponenten „gleich“.
Dazu passe durch Rechtsschieben den Exponenten der kleineren Zahl auf den Exponenten der größeren an.

(2) Wir addieren/subtrahieren die Mantissen ($\rightsquigarrow$ bilde dazu – falls notwendig – das Zweierkomplement).
Ist das Ergebnis < 0, setze das Vorzeichen bit des Ergebnis und bilde das Zweierkomplement.

(3) Normalisiere das Ergebnis. Dazu:
   A) Falls das Ergebnis $\geqslant 2$, schiebe das Ergebnis um eins (ggf. mit Rundung) nach rechts und inkrementiere den Exponenten.

Aufgabe 2 – Add. und Sub. von GKZ – Verfahrensidee

Wie könnte man zwei Zahlen $x_1 = (-1)^{V_1} \cdot 2^{(E_1 - B)} \cdot (M_1)$ und $x_2 = (-1)^{V_2} \cdot 2^{(E_2 - B)} \cdot (M_2)$ addieren?

(1) Wir „machen" die Exponenten „gleich".
Dazu passe durch Rechtsschieben den Exponenten der kleineren Zahl auf den Exponenten der größeren an.

(2) Wir addieren/subtrahieren die Mantissen ($\rightsquigarrow$ bilde dazu – falls notwendig – das Zweierkomplement).
Ist das Ergebnis < 0, setze das Vorzeichen bit des Ergebnis und bilde das Zweierkomplement.

(3) Normalisiere das Ergebnis. Dazu:
   A) Falls das Ergebnis $\geqslant 2$, schiebe das Ergebnis um eins (ggf. mit Rundung) nach rechts und inkrementiere den Exponenten.
   B) Falls das Ergebnis < 1, so schiebe das Ergebnis um ein nach links und dekrementiere den Exponenten.

Aufgabe 2 – Add. und Sub. von GKZ – Verfahrensidee

Wie könnte man zwei Zahlen $x_1 = (-1)^{V_1} \cdot 2^{(E_1 - B)} \cdot (M_1)$ und $x_2 = (-1)^{V_2} \cdot 2^{(E_2 - B)} \cdot (M_2)$ addieren?

(1) Wir „machen“ die Exponenten „gleich“.
Dazu passe durch Rechtsschieben den Exponenten der kleineren Zahl auf den Exponenten der größeren an.

(2) Wir addieren/subtrahieren die Mantissen ($\leadsto$ bilde dazu – falls notwendig – das Zweierkomplement).
Ist das Ergebnis < 0, setze das Vorzeichen bit des Ergebnis und bilde das Zweierkomplement.

(3) Normalisiere das Ergebnis. Dazu:

   A) Falls das Ergebnis $\geqslant 2$, schiebe das Ergebnis um eins (ggf. mit Rundung) nach rechts und inkrementiere den Exponenten.

   B) Falls das Ergebnis < 1, so schiebe das Ergebnis um ein nach links und dekrementiere den Exponenten.

   $\Rightarrow$ Wiederhole A) und B) solange bis das Ergebnis entweder = 0 oder $1 \leqslant$ Ergebnis < 2.

Aufgabe 2 – Add. und Sub. von GKZ – Verfahrensidee

Wie könnte man zwei Zahlen $x_1 = (-1)^{V_1} \cdot 2^{(E_1-B)} \cdot (M_1)$ und $x_2 = (-1)^{V_2} \cdot 2^{(E_2-B)} \cdot (M_2)$ addieren?

(1) Wir „machen“ die Exponenten „gleich“.
Dazu passe durch Rechtsschieben den Exponenten der kleineren Zahl auf den Exponenten der größeren an.

(2) Wir addieren/subtrahieren die Mantissen ($\rightsquigarrow$ bilde dazu – falls notwendig – das Zweierkomplement).
Ist das Ergebnis < 0, setze das Vorzeichen bit des Ergebnis und bilde das Zweierkomplement.

(3) Normalisiere das Ergebnis. Dazu:

- A) Falls das Ergebnis $\geqslant 2$, schiebe das Ergebnis um eins (ggf. mit Rundung) nach rechts und inkrementiere den Exponenten.
- B) Falls das Ergebnis < 1, so schiebe das Ergebnis um ein nach links und dekrementiere den Exponenten.
- $\Rightarrow$ Wiederhole A) und B) solange bis das Ergebnis entweder $= 0$ oder $1 \leqslant$ Ergebnis < 2.

(4) Behandle auftretende Sonderfälle – wie zum Beispiel einen Über-/Unterlauf oder Null.

Aufgabe 3 – Multiplikation von Gleitkommazahlen

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 3 – Multiplikation von Gleitkommazahlen

Sei folgendes Format für Gleitkommazahlen gegeben, das analog zum IEEE-Standard 754 definiert ist:

V	E (7)	M (8)
15	14 … 8	7 … 0

Multiplizieren Sie folgende Gleitkommazahlen dieses Formats miteinander:

$$x_1 = 0 \;\; 1001\,000 \;\; 1001\,1011$$
$$x_2 = 1 \;\; 1001\,010 \;\; 1110\,1000$$

Aufgabe 3 – Mult./Division von GKZ – Verfahrensidee

Wie könnte man zwei Zahlen $x_1 = (-1)^{V_1} \cdot 2^{(E_1 - B)} \cdot (M_1)$ und $x_2 = (-1)^{V_2} \cdot 2^{(E_2 - B)} \cdot (M_2)$ multiplizieren?

Aufgabe 3 – Mult./Division von GKZ – Verfahrensidee

Wie könnte man zwei Zahlen $x_1 = (-1)^{V_1} \cdot 2^{(E_1 - B)} \cdot (M_1)$ und $x_2 = (-1)^{V_2} \cdot 2^{(E_2 - B)} \cdot (M_2)$ multiplizieren?

(1) Wir multiplizieren/dividieren die Mantissen.
Hier ist aufzupasen, ob normalisierte oder denormalisierte Zahlen vorliegen.

Aufgabe 3 – Mult./Division von GKZ – Verfahrensidee

Wie könnte man zwei Zahlen $x_1 = (-1)^{V_1} \cdot 2^{(E_1 - B)} \cdot (M_1)$ und $x_2 = (-1)^{V_2} \cdot 2^{(E_2 - B)} \cdot (M_2)$ multiplizieren?

(1) Wir multiplizieren/dividieren die Mantissen.
Hier ist aufzupasen, ob normalisierte oder denormalisierte Zahlen vorliegen.

(2) Wir addieren/subtrahieren die „voreingenommenen" Exponenten ($\rightsquigarrow$ bilde dazu – falls notwendig – das Zweierkomplement).
Vorsicht: Der Bias kommt als Hilfsgröße in beiden Faktoren vor, deshalb wird der Bias einmal abgezogen/hinzuaddiert

Aufgabe 3 – Mult./Division von GKZ – Verfahrensidee

Wie könnte man zwei Zahlen $x_1 = (-1)^{V_1} \cdot 2^{(E_1 - B)} \cdot (M_1)$ und $x_2 = (-1)^{V_2} \cdot 2^{(E_2 - B)} \cdot (M_2)$ multiplizieren?

(1) Wir multiplizieren/dividieren die Mantissen.
Hier ist aufzupasen, ob normalisierte oder denormalisierte Zahlen vorliegen.

(2) Wir addieren/subtrahieren die „voreingenommenen“ Exponenten ($\rightsquigarrow$ bilde dazu – falls notwendig – das Zweierkomplement).
Vorsicht: Der Bias kommt als Hilfsgröße in beiden Faktoren vor, deshalb wird der Bias einmal abgezogen/hinzuaddiert

(3) Normalisiere die Mantisse.
Binge die Mantisse auf die Form $1, M$ und verschiebe ggf. den Exponenten.

Aufgabe 3 – Mult./Division von GKZ – Verfahrensidee

Wie könnte man zwei Zahlen $x_1 = (-1)^{V_1} \cdot 2^{(E_1 - B)} \cdot (M_1)$ und $x_2 = (-1)^{V_2} \cdot 2^{(E_2 - B)} \cdot (M_2)$ multiplizieren?

(1) Wir multiplizieren/dividieren die Mantissen.
Hier ist aufzupasen, ob normalisierte oder denormalisierte Zahlen vorliegen.

(2) Wir addieren/subtrahieren die „voreingenommenen“ Exponenten ($\rightsquigarrow$ bilde dazu – falls notwendig – das Zweierkomplement).
Vorsicht: Der Bias kommt als Hilfsgröße in beiden Faktoren vor, deshalb wird der Bias einmal abgezogen/hinzuaddiert

(3) Normalisiere die Mantisse.
Binge die Mantisse auf die Form $1, M$ und verschiebe ggf. den Exponenten.

(4) Behandle die Vorzeichen getrennt. Dazu wende ein XOR an!
$1 \oplus 0 = 1$ und $0 \oplus 0 = 0$ ...

Aufgabe 3 – Mult./Division von GKZ – Verfahrensidee

Wie könnte man zwei Zahlen $x_1 = (-1)^{V_1} \cdot 2^{(E_1-B)} \cdot (M_1)$ und $x_2 = (-1)^{V_2} \cdot 2^{(E_2-B)} \cdot (M_2)$ multiplizieren?

(1) Wir multiplizieren/dividieren die Mantissen.
Hier ist aufzupasen, ob normalisierte oder denormalisierte Zahlen vorliegen.

(2) Wir addieren/subtrahieren die „voreingenommenen" Exponenten ($\rightsquigarrow$ bilde dazu – falls notwendig – das Zweierkomplement).
Vorsicht: Der Bias kommt als Hilfsgröße in beiden Faktoren vor, deshalb wird der Bias einmal abgezogen/hinzuaddiert

(3) Normalisiere die Mantisse.
Binge die Mantisse auf die Form $1, M$ und verschiebe ggf. den Exponenten.

(4) Behandle die Vorzeichen getrennt. Dazu wende ein XOR an!
$1 \oplus 0 = 1$ und $0 \oplus 0 = 0$ ...

(5) Behandle auftretende Sonderfälle – wie zum Beispiel einen Über-/Unterlauf, Null oder auch NaN.

Aufgabe 4 – Assoziativität bei Gleitkommadarstellung

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 4 – Assoziativität bei Gleitkommadarstellung

Addition und Multiplikation von Operanden in einer Gleitkommadarstellung sind im Allgemeinen **nicht** assoziativ.
Belegen Sie dies, indem Sie mit einem PC-Tabellenkalkulationsprogramm experimentieren. Bestimmen Sie dadurch auch die Anzahl der Bits, die zur Speicherung der Mantisse verwendet wird.
(Beispiel: LibreOffice und Excel liefern bei der Berechnung von:
$10^{20} + 17 - 10 - 10^{20} + 130$ als Ergebnis 130 – dies übrigens ohne irgendeine Warnung.)

Aufgabe 4 – Assoziativität bei Gleitkommadarstellung

Sowohl Excel 2013 als auch LibreOffice 4.2 liefern zum Beispiel folgende Ergebnisse:

	Formel	Ergebnis
A	$10^{20} - 10^{20} + 17 - 10 + 130$	$= 137$
B	$10^{15} - 10^{15} + 17 - 10 + 130$	$= 137$
C	$10^{16} - 10^{16} + 17 - 10 + 130$	$= 137$
D	$10^{20} + 17 - 10 - 10^{20} + 130$	$= 130$
E	$10^{15} + 17 - 10 - 10^{15} + 130$	$= 137$
F	$10^{16} + 17 - 10 - 10^{16} + 130$	$= 136$
G	$10^{20} + 17 + 130 - 10^{20} - 10$	$= -10$
H	$10^{15} + 17 + 130 - 10^{15} - 10$	$= 137$
I	$10^{16} + 17 + 130 - 10^{16} - 10$	$= 136$

Wo liegen die Probleme?

Aufgabe 4 – Assoziativität bei Gleitkommadarstellung

Sowohl Excel 2013 als auch LibreOffice 4.2 liefern zum Beispiel folgende Ergebnisse:

	Formel	Ergebnis
⋮	⋮	⋮
D	$10^{20} + 17 - 10 - 10^{20} + 130$	$= 130$
⋮	⋮	⋮
G	$10^{20} + 17 + 130 - 10^{20} - 10$	$= -10$
⋮	⋮	⋮

Aufgabe 4 – Assoziativität bei Gleitkommadarstellung

Sowohl Excel 2013 als auch LibreOffice 4.2 liefern zum Beispiel folgende Ergebnisse:

	Formel	Ergebnis
⋮	⋮	⋮
D	$10^{20} + 17 - 10 - 10^{20} + 130$	$= 130$
⋮	⋮	⋮
G	$10^{20} + 17 + 130 - 10^{20} - 10$	$= -10$
⋮	⋮	⋮

Problem bei D und G

Aufgabe 4 – Assoziativität bei Gleitkommadarstellung

Sowohl Excel 2013 als auch LibreOffice 4.2 liefern zum Beispiel folgende Ergebnisse:

	Formel	Ergebnis
$\vdots$	$\vdots$	$\vdots$
D	$10^{20} + 17 - 10 - 10^{20} + 130$	$= 130$
$\vdots$	$\vdots$	$\vdots$
G	$10^{20} + 17 + 130 - 10^{20} - 10$	$= -10$
$\vdots$	$\vdots$	$\vdots$

Problem bei D und G

Formel D liefert das falsche Ergebnis, weil $10^{20} + 7$ nicht dargestellt werden kann.

Aufgabe 4 – Assoziativität bei Gleitkommadarstellung

Sowohl Excel 2013 als auch LibreOffice 4.2 liefern zum Beispiel folgende Ergebnisse:

	Formel	Ergebnis
⋮	⋮	⋮
D	$10^{20} + 17 - 10 - 10^{20} + 130$	$= 130$
⋮	⋮	⋮
G	$10^{20} + 17 + 130 - 10^{20} - 10$	$= -10$
⋮	⋮	⋮

Problem bei D und G

Formel D liefert das falsche Ergebnis, weil $10^{20} + 7$ nicht dargestellt werden kann. 10^{20} ist zwar darstellbar, aber aufgrund des hohen Exponenten ist die Genauigkeit für die 7 nicht ausreichend, weshalb auf 10^{20} gerundet wird.

Aufgabe 4 – Assoziativität bei Gleitkommadarstellung

Sowohl Excel 2013 als auch LibreOffice 4.2 liefern zum Beispiel folgende Ergebnisse:

	Formel	Ergebnis
⋮	⋮	⋮
D	$10^{20} + 17 - 10 - 10^{20} + 130$	$= 130$
⋮	⋮	⋮
G	$10^{20} + 17 + 130 - 10^{20} - 10$	$= -10$
⋮	⋮	⋮

Problem bei D und G

Formel D liefert das falsche Ergebnis, weil $10^{20} + 7$ nicht dargestellt werden kann. 10^{20} ist zwar darstellbar, aber aufgrund des hohen Exponenten ist die Genauigkeit für die 7 nicht ausreichend, weshalb auf 10^{20} gerundet wird.
Anschließend wird 10^{20} (= 0) wieder abgezogen und 130 addiert, was das Ergebnis von 130 erklärt.

Aufgabe 4 – Assoziativität bei Gleitkommadarstellung

Sowohl Excel 2013 als auch LibreOffice 4.2 liefern zum Beispiel folgende Ergebnisse:

	Formel	Ergebnis
$\vdots$	$\vdots$	$\vdots$
D	$10^{20} + 17 - 10 - 10^{20} + 130$	$= 130$
$\vdots$	$\vdots$	$\vdots$
G	$10^{20} + 17 + 130 - 10^{20} - 10$	$= -10$
$\vdots$	$\vdots$	$\vdots$

Problem bei D und G

Formel D liefert das falsche Ergebnis, weil $10^{20} + 7$ nicht dargestellt werden kann.
10^{20} ist zwar darstellbar, aber aufgrund des hohen Exponenten ist die Genauigkeit für die 7 nicht ausreichend, weshalb auf 10^{20} gerundet wird.
Anschließend wird 10^{20} (= 0) wieder abgezogen und 130 addiert, was das Ergebnis von 130 erklärt.
Analog bei Formel G.

Aufgabe 4 – Assoziativität bei Gleitkommadarstellung

Sowohl Excel 2013 als auch LibreOffice 4.2 liefern zum Beispiel folgende Ergebnisse:

	Formel	Ergebnis
⋮	⋮	⋮
F	$10^{16} + 17 - 10 - 10^{16} + 130$	$= 136$
⋮	⋮	⋮
I	$10^{16} + 17 + 130 - 10^{16} - 10$	$= 136$

Aufgabe 4 – Assoziativität bei Gleitkommadarstellung

Sowohl Excel 2013 als auch LibreOffice 4.2 liefern zum Beispiel folgende Ergebnisse:

	Formel	Ergebnis
$\vdots$	$\vdots$	$\vdots$
F	$10^{16} + 17 - 10 - 10^{16} + 130$	$= 136$
$\vdots$	$\vdots$	$\vdots$
I	$10^{16} + 17 + 130 - 10^{16} - 10$	$= 136$

Problem bei F und I

Aufgabe 4 – Assoziativität bei Gleitkommadarstellung

Sowohl Excel 2013 als auch LibreOffice 4.2 liefern zum Beispiel folgende Ergebnisse:

	Formel	Ergebnis
$\vdots$	$\vdots$	$\vdots$
F	$10^{16} + 17 - 10 - 10^{16} + 130$	$= 136$
$\vdots$	$\vdots$	$\vdots$
I	$10^{16} + 17 + 130 - 10^{16} - 10$	$= 136$

Problem bei F und I

Formeln F und I liefern das falsche Ergebnis, weil bei der Darstellung von 10^{16} die niederstwertige 1 von $17 = 10001_2$ bei der Addition genau auf die 54. Stelle der Mantisse fallen würde, die Mantisse bei doppelter Genauigkeit (64 Bit) jedoch nur 53 Stellen hat.

Aufgabe 4 – Assoziativität bei Gleitkommadarstellung

Sowohl Excel 2013 als auch LibreOffice 4.2 liefern zum Beispiel folgende Ergebnisse:

	Formel	Ergebnis
$\vdots$	$\vdots$	$\vdots$
F	$10^{16} + 17 - 10 - 10^{16} + 130$	$= 136$
$\vdots$	$\vdots$	$\vdots$
I	$10^{16} + 17 + 130 - 10^{16} - 10$	$= 136$

Problem bei F und I

Formeln F und I liefern das falsche Ergebnis, weil bei der Darstellung von 10^{16} die niederstwertige 1 von $17 = 10001_2$ bei der Addition genau auf die 54. Stelle der Mantisse fallen würde, die Mantisse bei doppelter Genauigkeit (64 Bit) jedoch nur 53 Stellen hat.
Effektiv wird also der Summand 1 abgeschnitten und nur 16 addiert.

Aufgabe 4 – Assoziativität bei Gleitkommadarstellung

Sowohl Excel 2013 als auch LibreOffice 4.2 liefern zum Beispiel folgende Ergebnisse:

	Formel	Ergebnis
⋮	⋮	⋮
F	$10^{16} + 17 - 10 - 10^{16} + 130$	$= 136$
⋮	⋮	⋮
I	$10^{16} + 17 + 130 - 10^{16} - 10$	$= 136$

Problem bei F und I

Formeln F und I liefern das falsche Ergebnis, weil bei der Darstellung von 10^{16} die niederstwertige 1 von $17 = 10001_2$ bei der Addition genau auf die 54. Stelle der Mantisse fallen würde, die Mantisse bei doppelter Genauigkeit (64 Bit) jedoch nur 53 Stellen hat.
Effektiv wird also der Summand 1 abgeschnitten und nur 16 addiert.

Diese Beispiele zeigen auch, dass doppelte Genauigkeit verwendet wird, denn sonst würde auch die 16 verloren gehen.

Übungen zur Grundlagen der Technischen Informatik

Übung 5 – Schaltfunktionen und Logik

Florian Frank

Friedrich-Alexander-Universität Erlangen-Nürnberg

Wintersemester 2018/19

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Was machen wir heute?

Organisatorisches

Was machen wir heute?

Organisatorisches

Aufgabe 1 – Darstellung von Schaltfunktionen

Was machen wir heute?

Organisatorisches

Aufgabe 1 – Darstellung von Schaltfunktionen

Aufgabe 2 – Logik

Organisatorisches

design

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Achtung – Miniklausur

Achtung – Miniklausur

Diesen Donnerstag – am 29. November 2018 – findet die erste Miniklausur statt.

Wo?	im H7
Wann?	Wir starten um 16:15 Uhr!
	*Seid bitte deswegen schon **ungefähr*** 3 – 5 ***Minuten** vor Beginn da.*
Worum gehts?	Um den Übungsstoff bis Blatt 5 – sprich heute –
	und den Vorlesungsstoff bis einschließlich zum 27.11.2018

Aufgabe 1 – Darstellung von Schaltfunktionen

Aufgabe 1 – Darstellung von Schaltfunktionen

a) Geben Sie für die folgenden Schaltfunktionen jeweils die Funktionstabelle, ein Symmetriediagramm, ein Binary Decision Diagram und ein Gatterschaltnetz an:

i) $f_1(x, y) = x \oplus y$

ii) $f_2(x, y, z) = (x + y)z$

iii) $f_3(x, y, z) = xy\bar{z} + \overline{x + \bar{y} + z}$

Aufgabe 1 – Darstellung von Schaltfunktionen: Begriffsklärung

Konjunktion – $\wedge$

Wir definieren die Konjunktion $x \wedge y \equiv x \cdot y$ über die Wahrheitstabelle rechts.
Konjunktionen werden in der theoretischen Informatik und Mathematik meistens mit dem Symbol $\wedge$ dargestellt, wir verwenden dafür das Multiplikationssymbol $\cdot$.

$x\,y$	$x \cdot y$
0 0	0
0 1	0
1 0	0
1 1	1

Disjunktion – $\vee$

Wir definieren die Disjunktion $x \vee y \equiv x + y$ über die Wahrheitstabelle rechts.
Konjunktionen werden in der theoretischen Informatik und Mathematik meistens mit dem Symbol $\vee$ dargestellt, wir verwenden dafür das Additionssymbol $+$.

$x\,y$	$x + y$
0 0	0
0 1	1
1 0	1
1 1	1

Aufgabe 1 – Darstellung von Schaltfunktionen: Begriffsklärung

Negation – $\overline{}$

Wir definieren die Negation $\bar{x}$ über die Wahrheitstabelle rechts.
Wir stellen Negationen durch einen Querstrich über der negierten Formel $\overline{}$ dar.

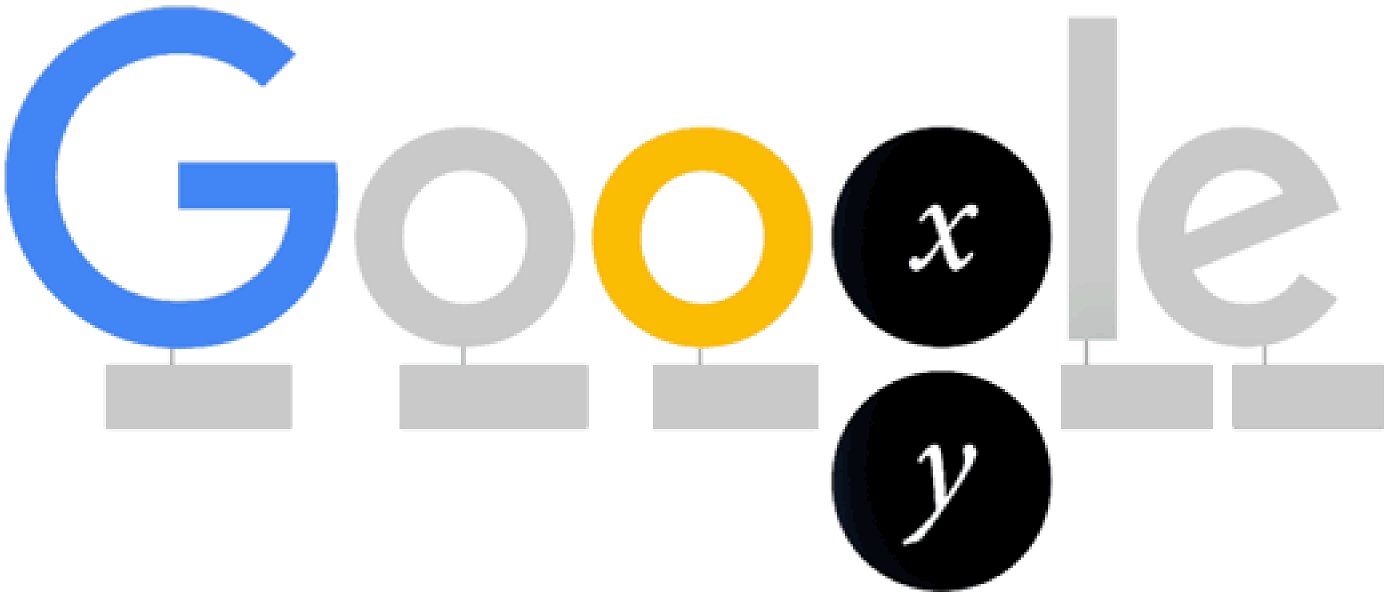
x	$\bar{x}$
0	1
1	0

Exklusive Disjunktion – $\oplus$

Wir definieren die exklusive Disjunktion $x \oplus y$ wie folgt:

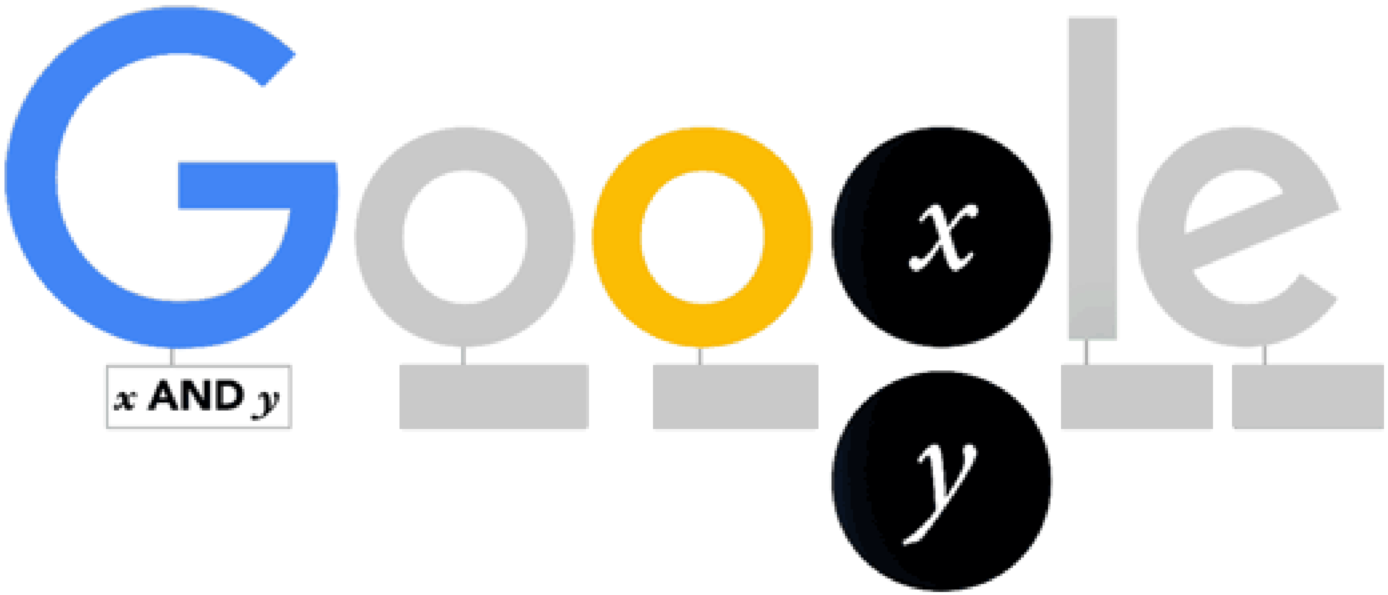
$$x \oplus y \equiv x \cdot \bar{y} + \bar{x} \cdot y$$

Schaltfunktionen – Google-Doodle



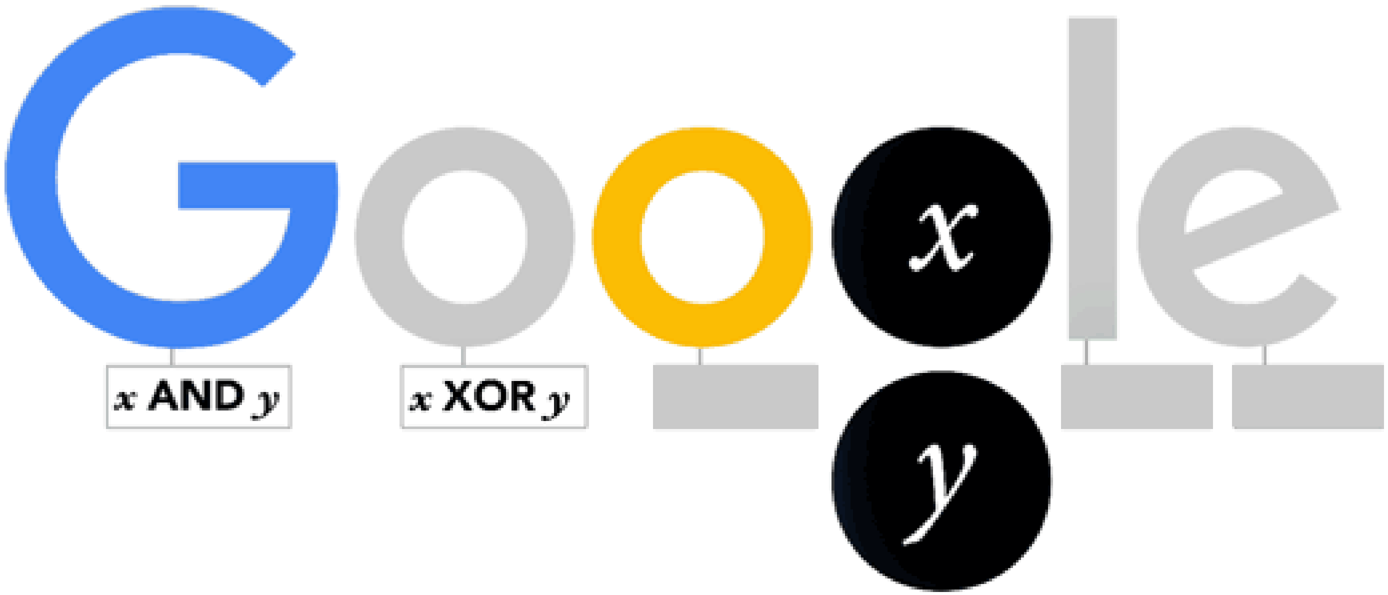
Quelle: https://www.google.com/doodles/george-booles-200th-birthday

Schaltfunktionen – Google-Doodle



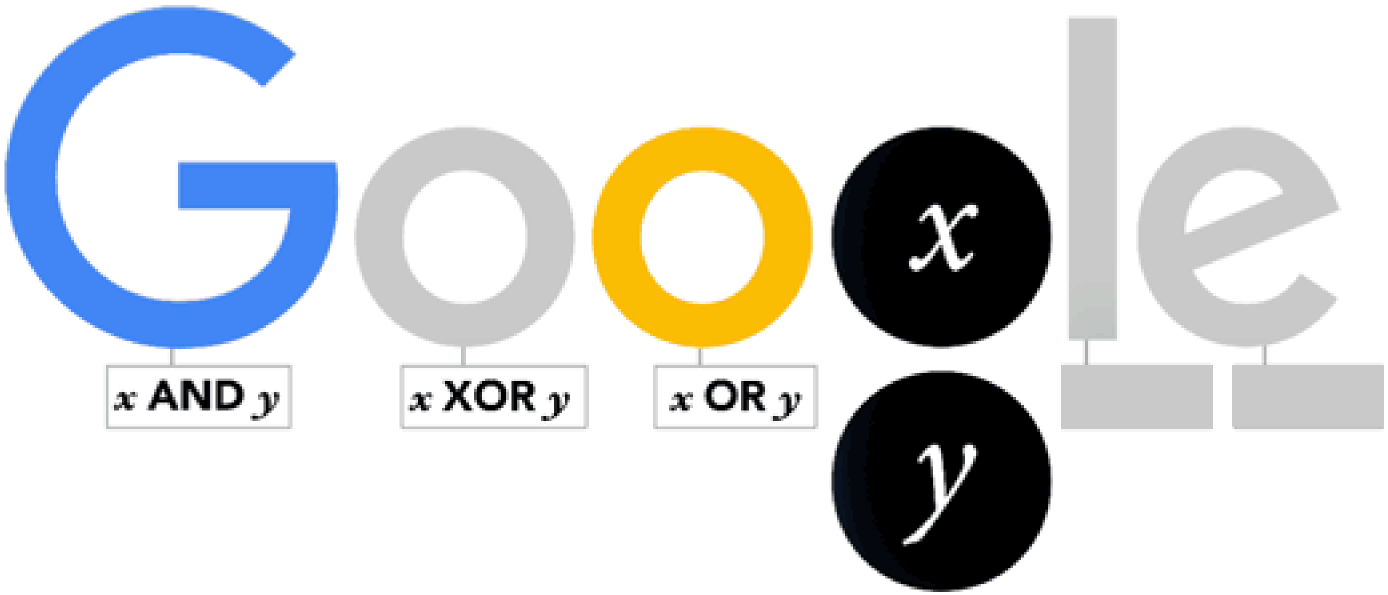
Quelle: https://www.google.com/doodles/george-booles-200th-birthday

Schaltfunktionen – Google-Doodle



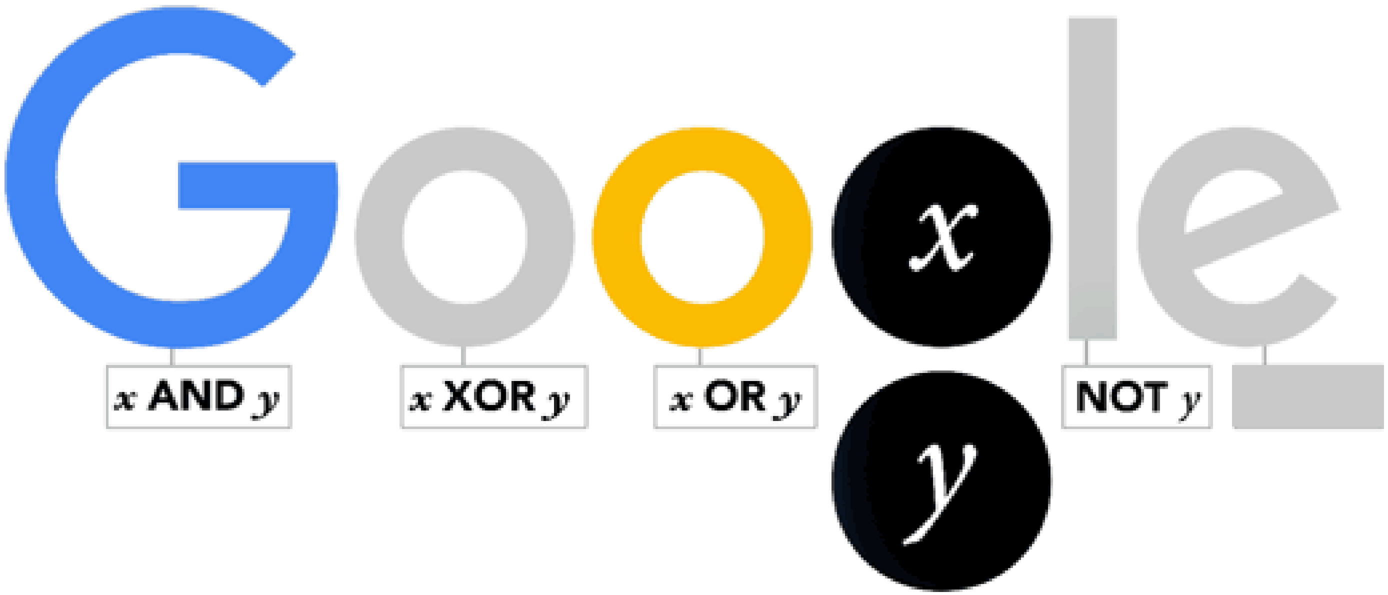
Quelle: https://www.google.com/doodles/george-booles-200th-birthday

Schaltfunktionen – Google-Doodle



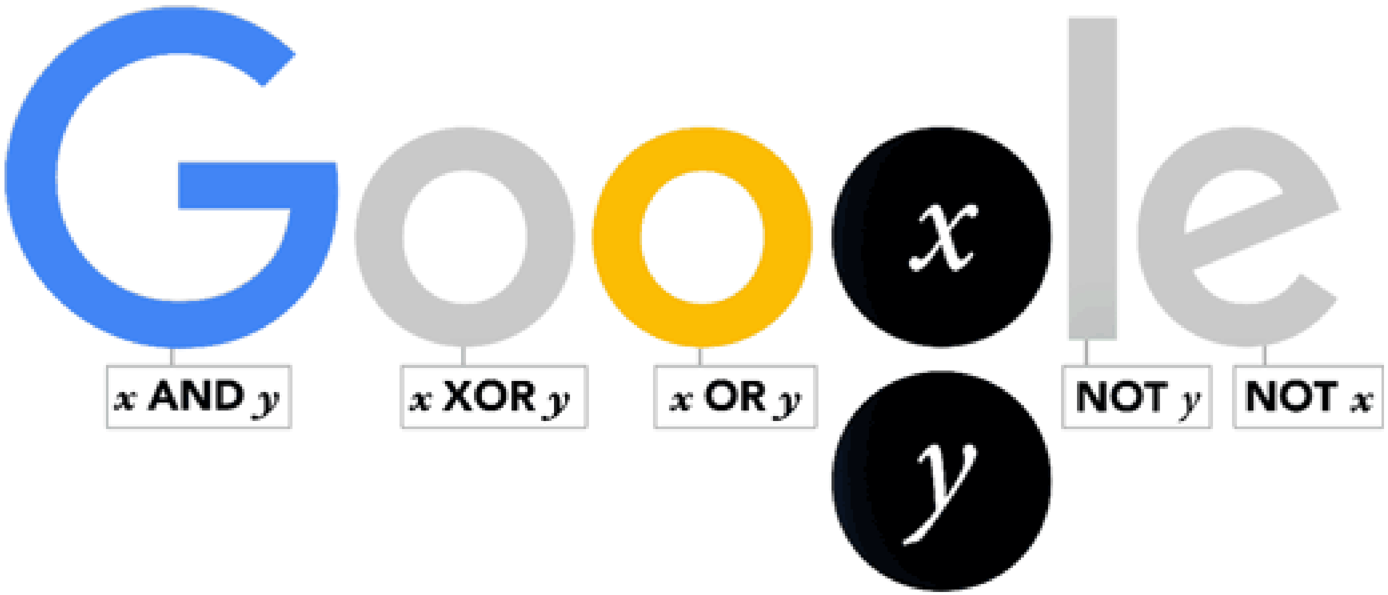
Quelle: https://www.google.com/doodles/george-booles-200th-birthday

Schaltfunktionen – Google-Doodle



Quelle: https://www.google.com/doodles/george-booles-200th-birthday

Schaltfunktionen – Google-Doodle



Quelle: https://www.google.com/doodles/george-booles-200th-birthday

Aufgabe 1 – Darstellung von Schaltfunktionen: Begriffsklärung

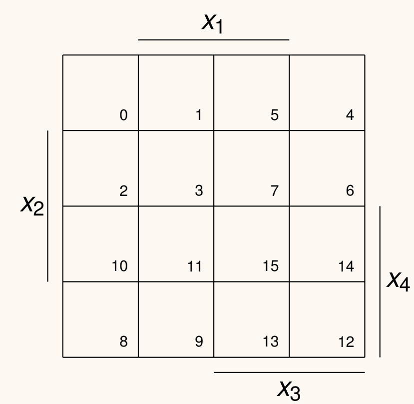
Wahrheitstabelle oder Funktionstabelle

Die Funktionstabelle einer Schaltfunktion f enthält die Funktionswerte für **alle** Permutationen der Variablenwerte.
Bei n Variablen gibt es somit 2^n Zeilen in der Funktionstabelle.

Aufgabe 1 – Darstellung von Schaltfunktionen: Begriffsklärung

Symmetriediagramm

Darstellung der Funktionswerte in einer Matrix. Dabei steht jede Zelle für eine mögliche Wahrheitsbelegung. Rechts ein Beispiel für ein Symmetriediagramm mit 4 Variablen.

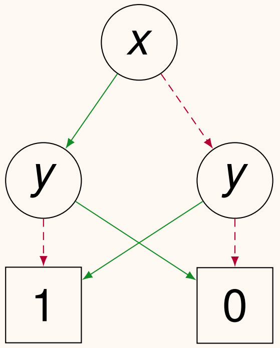


⇝ Die Konstruktion erfolgt über Spiegelung an abwechselnden Seiten (siehe Tafel)

Aufgabe 1 – Darstellung von Schaltfunktionen: Begriffsklärung

Binary Decision Diagram

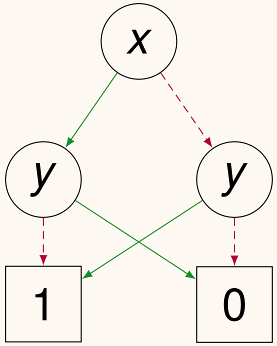
Bei Binary Decision Diagrams (BDD) werden die Funktionstabellen graphisch in Baumform dargestellt. Man erzeugt einen BDD durch eine Fallunterscheidung bei allen in dem Term vorkommenden Variablen. Ein Beispiel ist rechts zu sehen.



Aufgabe 1 – Darstellung von Schaltfunktionen: Begriffsklärung

Binary Decision Diagram

Bei Binary Decision Diagrams (BDD) werden die Funktionstabellen graphisch in Baumform dargestellt. Man erzeugt einen BDD durch eine Fallunterscheidung bei allen in dem Term vorkommenden Variablen. Ein Beispiel ist rechts zu sehen.



Problem 1

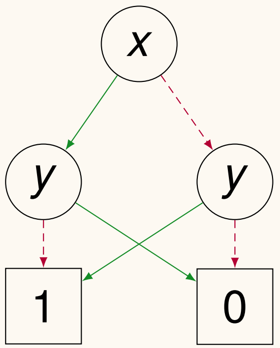
Zu einer Schaltfunktion mit n Variablen ergeben sich $n!$ verschiedene BDDs.
$\rightsquigarrow$ Wir bemühen uns die „Entwicklungsreihenfolge“ fest vorzugeben, dann ist der BDD eindeutig.

ORDERED BINARY DECISION DIAGRAMM (OBDD)

Aufgabe 1 – Darstellung von Schaltfunktionen: Begriffsklärung

Binary Decision Diagram

Bei Binary Decision Diagrams (BDD) werden die Funktionstabellen graphisch in Baumform dargestellt. Man erzeugt einen BDD durch eine Fallunterscheidung bei allen in dem Term vorkommenden Variablen. Ein Beispiel ist rechts zu sehen.



Problem 2

Zu einer Schaltfunktion mit n Variablen gibt es bei einem BDD insgesamt $\sum_{i=0}^{n-1} 2^i = 2^n - 1$ Knoten.

$\rightsquigarrow$ Fasse deswegen „isomorphe" Teilbäume zusammen und eliminiere Knoten, deren beider Kindknoten „isomorph" sind.

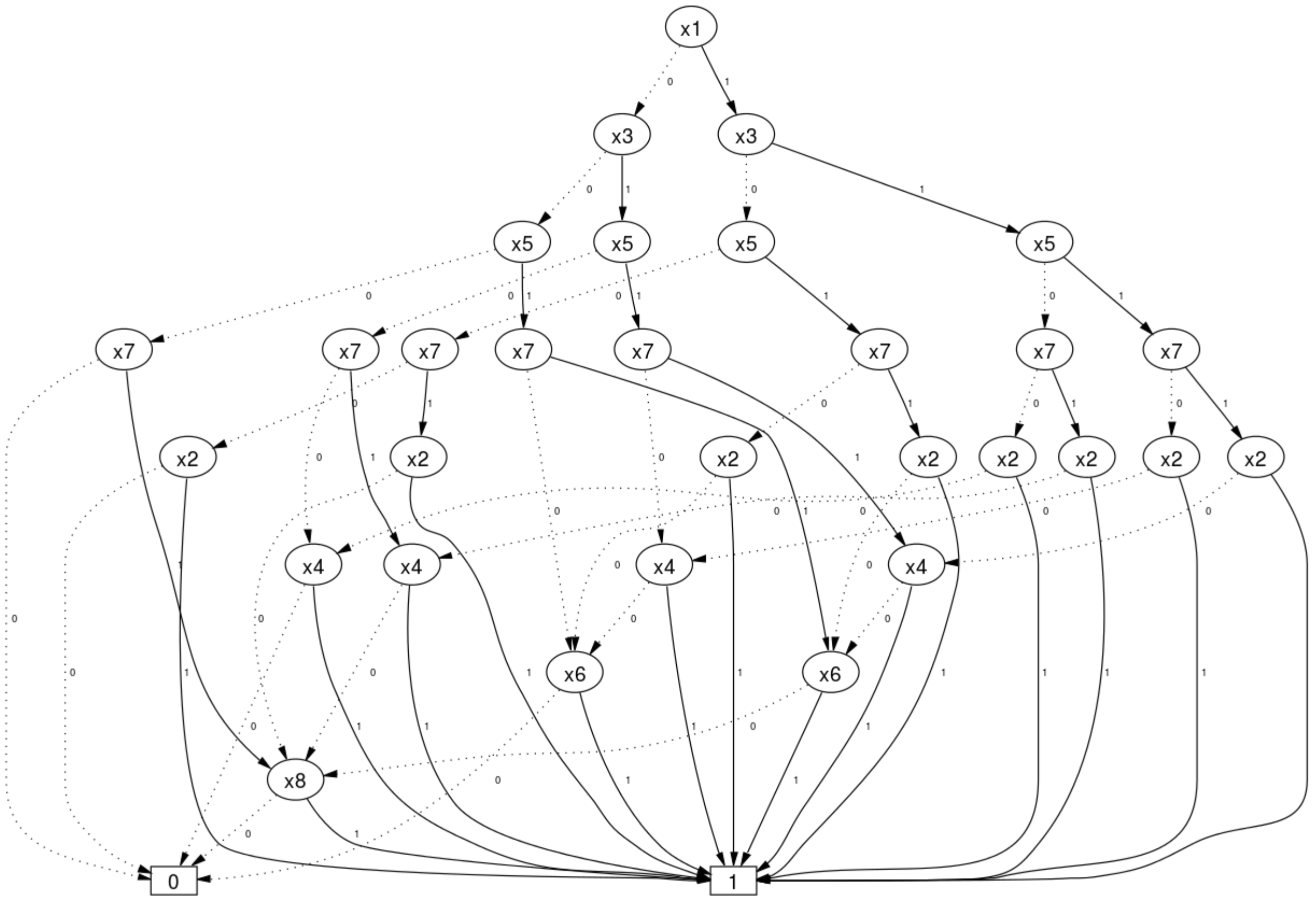
Reduced Binary Decision Diagramm (RBDD)

Warum die Reihenfolge eine Rolle spielt ...

Sei $f_9(x_1, ..., x_8) = (x_1 x_2) + (x_3 x_4) + (x_5 x_6) + (x_7 x_8)$ gegeben. Stellen wir nun den Baum in Reihenfolge $(x_1, x_3, x_5, x_7, x_2, x_4, x_6, x_8)$.
Dabei kommt folgender BDD raus:

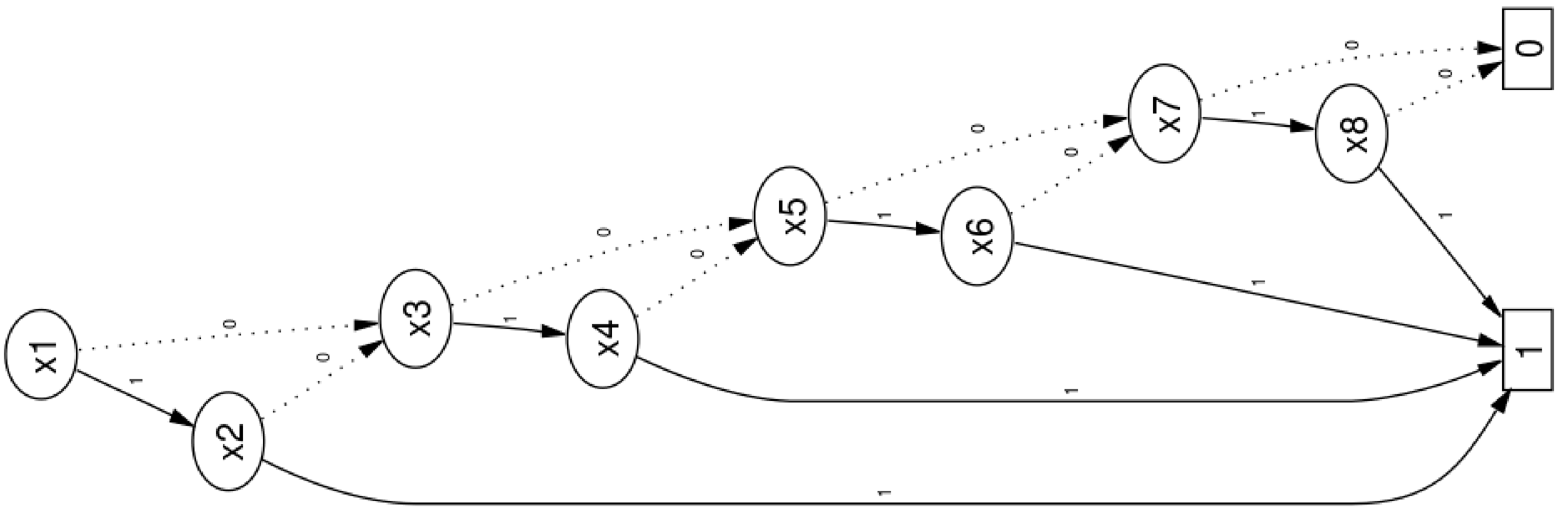
Warum die Reihenfolge eine Rolle spielt ...

Sei $f_9(x_1, ..., x_8) = (x_1 x_2) + (x_3 x_4) + (x_5 x_6) + (x_7 x_8)$ gegeben. Stellen wir nun den Baum in Reihenfolge $(x_1, x_3, x_5, x_7, x_2, x_4, x_6, x_8)$.
Dabei kommt folgender BDD raus:



Warum die Reihenfolge eine Rolle spielt ...

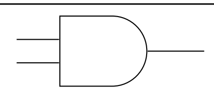
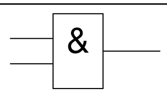
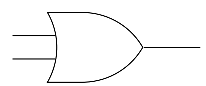
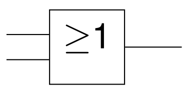
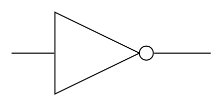
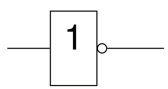
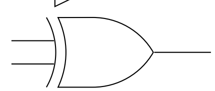
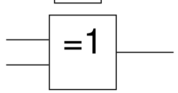
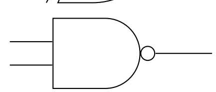
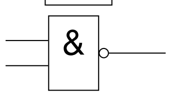
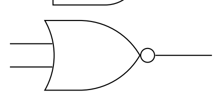
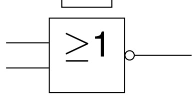
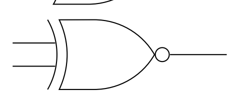
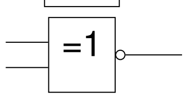
Sei $f_9(x_1, ..., x_8) = (x_1 x_2) + (x_3 x_4) + (x_5 x_6) + (x_7 x_8)$ gegeben. Stellen wir nun den Baum in Reihenfolge $(x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8)$.
Dabei kommt folgender BDD raus:



Aufgabe 1 – Darstellung von Schaltfunktionen: Begriffsklärung

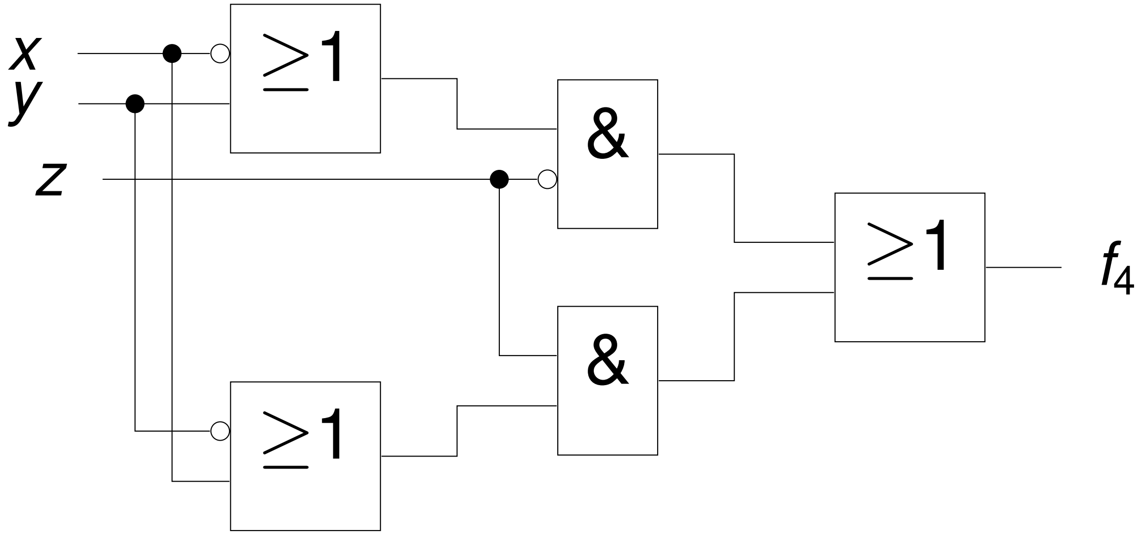
Logikgatter

Schaltfunktionen können auch durch Gattersymbole visualisiert werden. Dabei wird jeder Operation (jedem Junktor) ein Symbol zugewiesen, der Ausgang ist die Funktion des Gatters angewandt auf **alle** Eingänge.

Funktion	Symbol (IEEE)	Symbol (DIN)
AND		 &
OR		 ≥ 1
NOT		 1
XOR		 =1
NAND		 &
NOR		 ≥ 1
XNOR		 =1

Aufgabe 1 – Darstellung von Schaltfunktionen

b) Geben Sie für folgendes Gatterschaltnetz eine Schaltfunktion $f_4(x, y, z)$ an:



Aufgabe 1 – Darstellung von Schaltfunktionen

c) Geben Sie für die in folgender Funktionstabelle beschriebene Schaltfunktion $f_5(x_3, x_2, x_1, x_0)$ sowohl eine disjunktive als auch eine konjunktive Normalform an:

$x_3x_2x_1x_0$	f_5	$x_3x_2x_1x_0$	f_5	$x_3x_2x_1x_0$	f_5	$x_3x_2x_1x_0$	f_5
0 0 0 0	1	0 1 0 0	0	1 0 0 0	0	1 1 0 0	1
0 0 0 1	0	0 1 0 1	1	1 0 0 1	1	1 1 0 1	0
0 0 1 0	0	0 1 1 0	1	1 0 1 0	1	1 1 1 0	0
0 0 1 1	1	0 1 1 1	0	1 0 1 1	0	1 1 1 1	1

Aufgabe 1 – Darstellung von Schaltfunktionen: Begriffsklärung

Minterm (Vollkonjunktion)

Minterme sind eine reine Konjunktion **aller** existierenden Literale in negierter oder nicht negierter Form, deren Funktionswert 1 ist.

Aufgabe 1 – Darstellung von Schaltfunktionen: Begriffsklärung

Minterm (Vollkonjunktion)

Minterme sind eine reine Konjunktion **aller** existierenden Literale in negierter oder nicht negierter Form, deren Funktionswert 1 ist.

Merkhilfe: Minterme

Minterme überdecken nur eine Einsstelle in unserem Symmetriediagramm.

Aufgabe 1 – Darstellung von Schaltfunktionen: Begriffsklärung

Minterm (Vollkonjunktion)

Minterme sind eine reine Konjunktion **aller** existierenden Literale in negierter oder nicht negierter Form, deren Funktionswert 1 ist.

Merkhilfe: Minterme

Minterme überdecken nur eine Einsstelle in unserem Symmetriediagramm.

Disjunktive Normalform (DNF)

Disjunktion **aller** Minterme. Heißt:

$$\mathrm{DNF}(\varphi) = \bigvee_{i=1}^{\text{Anzahl an Mintermen}} \left(\bigwedge_{j=1}^{\text{Anzahl an Literalen}} L_j \right)$$

Aufgabe 1 – Darstellung von Schaltfunktionen: Begriffsklärung

Minterm (Vollkonjunktion)

Minterme sind eine reine Konjunktion **aller** existierenden Literale in negierter oder nicht negierter Form, deren Funktionswert 1 ist.

Merkhilfe: Minterme

Minterme überdecken nur eine Einsstelle in unserem Symmetriediagramm.

Disjunktive Normalform (DNF)

Disjunktion **aller** Minterme. Heißt:

$$\text{DNF}(\varphi) = \bigvee_{i=1}^{\text{Anzahl an Mintermen}} \left(\bigwedge_{j=1}^{\text{Anzahl an Literalen}} L_j \right)$$

Beispiel: Minterme und DNF

Seien die Literale festgelegt als x, y und z. Sei $f_{ed}(x, y, z) = \bar{y}$ gegeben. So sind die Terme $x \cdot \bar{y} \cdot z$ oder $\bar{x} \cdot \bar{y} \cdot \bar{z}$ beispielsweise Minterme.
Die DNF von f_{ed} hingegen lautet $(x \cdot \bar{y} \cdot z) + (\bar{x} \cdot \bar{y} \cdot z) + (x \cdot \bar{y} \cdot \bar{z}) + (\bar{x} \cdot \bar{y} \cdot \bar{z})$

Aufgabe 1 – Darstellung von Schaltfunktionen: Begriffsklärung

Maxterm (Volldisjunktion)

Maxterme sind eine reine Disjunktion **aller** existierenden Literale in negierter oder nicht negierter Form, deren Funktionswert 0 ist.

Aufgabe 1 – Darstellung von Schaltfunktionen: Begriffsklärung

Maxterm (Volldisjunktion)

Maxterme sind eine reine Disjunktion **aller** existierenden Literale in negierter oder nicht negierter Form, deren Funktionswert 0 ist.

Merkhilfe: Maxterme

Maxterme überdecken nur eine Nullstelle in unserem Symmetriediagramm.

Aufgabe 1 – Darstellung von Schaltfunktionen: Begriffsklärung

Maxterm (Volldisjunktion)

Maxterme sind eine reine Disjunktion **aller** existierenden Literale in negierter oder nicht negierter Form, deren Funktionswert 0 ist.

Merkhilfe: Maxterme

Maxterme überdecken nur eine Nullstelle in unserem Symmetriediagramm.

Konjunktive Normalform (KNF)

Konjunktion **aller** Maxterme. Heißt:

$$\mathrm{KNF}(\varphi) = \bigwedge_{i=1}^{\text{Anzahl an Maxtermen}} \left(\bigvee_{j=1}^{\text{Anzahl an Literalen}} L_j \right)$$

Aufgabe 1 – Darstellung von Schaltfunktionen: Begriffsklärung

Maxterm (Volldisjunktion)

Maxterme sind eine reine Disjunktion **aller** existierenden Literale in negierter oder nicht negierter Form, deren Funktionswert 0 ist.

Merkhilfe: Maxterme

Maxterme überdecken nur eine Nullstelle in unserem Symmetriediagramm.

Konjunktive Normalform (KNF)

Konjunktion **aller** Maxterme. Heißt:

$$\text{KNF}(\varphi) = \bigwedge_{i=1}^{\text{Anzahl an Maxtermen}} \left(\bigvee_{j=1}^{\text{Anzahl an Literalen}} L_j \right)$$

Beispiel: Maxterme und KNF

Seien die Literale festgelegt als x, y und z. Sei $f_{ed}(x, y, z) = \bar{y}$ gegeben. So sind die Terme $\bar{x} + \bar{y} + z$ oder $\bar{x} + \bar{y} + \bar{z}$ beispielsweise Maxterme.
Die KNF von f_{ed} hingegen lautet $(x + \bar{y} + z) \cdot (\bar{x} + \bar{y} + z) \cdot (x + \bar{y} + \bar{z}) \cdot (\bar{x} + \bar{y} + \bar{z})$

Aufgabe 1 – Darstellung von Schaltfunktionen

d) Geben Sie für die in folgendem Symmetriediagramm beschriebene Schaltfunktion $f_6(x_3, x_2, x_1, x_0)$ sowohl eine disjunktive als auch eine konjunktive Normalform an:

		x_0	x_0		
	1_{0}	0_{1}	0_{5}	1_{4}	
x_1	0_{2}	1_{3}	1_{7}	0_{6}	
x_1	0_{10}	1_{11}	1_{15}	0_{14}	x_3
	1_{8}	0_{9}	0_{13}	1_{12}	x_3
			x_2	x_2	

Aufgabe 2 – Logik

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Aufgabe 2 – Logik

Fünf ehemalige Zuschauer eines Fußballturniers haben versucht, sich an die damalige Rangliste zu erinnern und machten dabei die folgenden Aussagen:

- Mannschaft A belegte den 2. Platz und Mannschaft B den 5.
- Mannschaft C belegte den 2. Platz und Mannschaft D den 3.
- Mannschaft F belegte den 1. Platz und Mannschaft B den 3.
- Mannschaft A belegte den 3. Platz und Mannschaft E den 6.
- Mannschaft C belegte den 3. Platz und Mannschaft E den 4.

Später stellte sich heraus, dass jeder Zuschauer sich in einer seiner beiden Aussagen geirrt hatte.
Rekonstruieren Sie die richtige Platzverteilung.

Übungen zur Grundlagen der Technischen Informatik

Übung 6 – Normalformen, Minimalformen und der Entwicklungssatz

Florian Frank

Friedrich-Alexander-Universität Erlangen-Nürnberg

Wintersemester 2018/19

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Was machen wir heute?

Aufgabe 1 – Boolesche Algebra — Beweise

Aufgabe 2 – DNF — DF — KF

Aufgabe 3 – Relaisschaltnetze

Aufgabe 4 – Entwicklungssatz

Zusatzaufgabe – Nutzen des Entwicklungssatzes

Aufgabe 1 – Boolesche Algebra — Beweise

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Aufgabe 1 – Boolesche Algebra — Beweise

Beweisen oder widerlegen Sie die folgenden Aussagen, ohne Wahrheitstabellen zu verwenden. Für Aussagen, die nicht wahr sind, geben Sie ein Gegenbeispiel an. Für Aussagen, die wahr sind, geben Sie die entsprechenden Regeln an, mit welchen die Eigenschaft bewiesen werden kann.

a) $\overline{x} \oplus (x + y) = x + \overline{y}$

b) $(x + y) \cdot (x + y \cdot z) = x + y \cdot z$

c) $a \cdot b + \overline{b} \cdot c \cdot \overline{d} + a \cdot c \cdot \overline{d} = a \cdot b + \overline{b} \cdot c \cdot \overline{d}$

d) $x \cdot y + x \cdot (y + z) = x \cdot (y + z)$

e) $a \cdot b + \overline{b} \cdot c \cdot \overline{d} + a \cdot c \cdot \overline{d} = a \cdot b + c \cdot \overline{d}$

Aufgabe 1 – Boolesche Algebra: Begriffsklärung

Definition (Boolesche Algebra)

Eine Menge B mit zwei Verknüpfungen $\top$ und $\perp$ auf B ist eine **boolesche Algebra** gdw. für alle Elemente $a, b, c \in B$ gilt:

Aufgabe 1 – Boolesche Algebra: Begriffsklärung

Definition (Boolesche Algebra)

Eine Menge B mit zwei Verknüpfungen $\top$ und $\perp$ auf B ist eine **boolesche Algebra** gdw. für alle Elemente $a, b, c \in B$ gilt:

(1) **Abgeschlossenheit** der Operationen – implizit mit „... Verknüpfungen auf B“ gefordert.

Aufgabe 1 – Boolesche Algebra: Begriffsklärung

Definition (Boolesche Algebra)

Eine Menge B mit zwei Verknüpfungen $\top$ und $\perp$ auf B ist eine **boolesche Algebra** gdw. für alle Elemente $a, b, c \in B$ gilt:

(1) **Abgeschlossenheit** der Operationen – implizit mit „... Verknüpfungen auf B“ gefordert.

(2) **Kommutativität** der Operationen:

$$a \top b \equiv b \top a \quad \text{und} \quad a \perp b \equiv b \perp a$$

Aufgabe 1 – Boolesche Algebra: Begriffsklärung

Definition (Boolesche Algebra)

Eine Menge B mit zwei Verknüpfungen $\top$ und $\perp$ auf B ist eine **boolesche Algebra** gdw. für alle Elemente $a, b, c \in B$ gilt:

(1) **Abgeschlossenheit** der Operationen – implizit mit „... Verknüpfungen auf B“ gefordert.

(2) **Kommutativität** der Operationen:

$$a \top b \equiv b \top a \quad \text{und} \quad a \perp b \equiv b \perp a$$

(3) **Distributivität** der Operationen:

$$a \top (b \perp c) \equiv (a \top b) \perp (a \top c) \quad \text{und} \quad a \perp (b \top c) \equiv (a \perp b) \top (a \perp c)$$

Aufgabe 1 – Boolesche Algebra: Begriffsklärung

Definition (Boolesche Algebra)

Eine Menge B mit zwei Verknüpfungen $\top$ und $\perp$ auf B ist eine **boolesche Algebra** gdw. für alle Elemente $a, b, c \in B$ gilt:

(1) **Abgeschlossenheit** der Operationen – implizit mit „... Verknüpfungen auf B“ gefordert.

(2) **Kommutativität** der Operationen:

$$a \top b \equiv b \top a \quad \text{und} \quad a \perp b \equiv b \perp a$$

(3) **Distributivität** der Operationen:

$$a \top (b \perp c) \equiv (a \top b) \perp (a \top c) \quad \text{und} \quad a \perp (b \top c) \equiv (a \perp b) \top (a \perp c)$$

(4) Existenz **neutraler** Elemente:

$$\exists 0, 1 \in B : a \top 1 \equiv a \quad \text{und} \quad a \perp 0 \equiv a$$

Aufgabe 1 – Boolesche Algebra: Begriffsklärung

Definition (Boolesche Algebra)

Eine Menge B mit zwei Verknüpfungen $\top$ und $\perp$ auf B ist eine **boolesche Algebra** gdw. für alle Elemente $a, b, c \in B$ gilt:

(1) **Abgeschlossenheit** der Operationen – implizit mit „... Verknüpfungen auf B“ gefordert.

(2) **Kommutativität** der Operationen:

$$a \top b \equiv b \top a \quad \text{und} \quad a \perp b \equiv b \perp a$$

(3) **Distributivität** der Operationen:

$$a \top (b \perp c) \equiv (a \top b) \perp (a \top c) \quad \text{und} \quad a \perp (b \top c) \equiv (a \perp b) \top (a \perp c)$$

(4) Existenz **neutraler** Elemente:

$$\exists 0, 1 \in B : a \top 1 \equiv a \quad \text{und} \quad a \perp 0 \equiv a$$

(5) Existenz von **Komplementen** $\overline{a}$:

$$\exists \overline{a} \in B : a \top \overline{a} \equiv 0 \quad \text{und} \quad a \perp \overline{a} \equiv 1$$

Aufgabe 1 – Boolesche Algebra: Begriffsklärung

Dualitätsprinzip von de Morgan

Zu jeder existierenden Formel für eine boolesche Algebra existiert auch eine **duale** Formel, die entsteht, wenn man $\top$ und $\perp$ sowie 1 und 0 vertauscht.

Aufgabe 1 – Boolesche Algebra: Begriffsklärung

Mit der axiomatischen Definition ergeben sich folgende „Rechenregeln“:

Abk.	Regelname	Verwendung
(KG)	Kommutativgesetz	$a + b \equiv b + a$ und $a \cdot b \equiv b \cdot a$
(AG)	Assoziativgesetz	$a + (b + c) \equiv (a + b) + c$ $a \cdot (b \cdot c) \equiv (a \cdot b) \cdot c$
(IG)	Idempotenzgesetz	$a + a \equiv a$ und $a \cdot a \equiv a$
(DG)	Distribution	$a + (b \cdot c) \equiv (a + b) \cdot (a + c)$ $a \cdot (b + c) \equiv (a \cdot b) + (a \cdot c)$
(NG)	Neutralitätsgesetz	$a + 0 \equiv a$ und $a \cdot 1 \equiv a$
(EG)	Extremalgesetz	$a + 1 \equiv 1$ und $a \cdot 0 \equiv 0$
(IN)	Involution	$\overline{\overline{a}} \equiv a$
(DM)	De Morgansche Gesetze	$\overline{a + b} \equiv \overline{a} \cdot \overline{b}$ und $\overline{ab} \equiv \overline{a} + \overline{b}$
($\overline{\text{K}}$G)	Komplementärgesetz	$a + \overline{a} \equiv 1$ und $a \cdot \overline{a} \equiv 0$
(DT)	Dualitätsgesetz	$\overline{0} \equiv 1$ und $\overline{1} \equiv 0$
(AB)	Absorption	$a \cdot (a + b) \equiv a$ und $a + (a \cdot b) \equiv a$
(KO)	Konsensus	$(a \cdot b) + (\overline{a} \cdot c) + (b \cdot c) \equiv (a \cdot b) + (\overline{a} \cdot c)$ $(a + b) \cdot (\overline{a} + c) \cdot (b + c) \equiv (a + b) \cdot (\overline{a} + c)$

Aufgabe 1 – Boolesche Algebra — Beweise

Beweisen oder widerlegen Sie die folgenden Aussagen, ohne Wahrheitstabellen zu verwenden. Für Aussagen, die nicht wahr sind, geben Sie ein Gegenbeispiel an. Für Aussagen, die wahr sind, geben Sie die entsprechenden Regeln an, mit welchen die Eigenschaft bewiesen werden kann.

a) $\overline{x} \oplus (x + y) = x + \overline{y}$

b) $(x + y) \cdot (x + y \cdot z) = x + y \cdot z$

c) $a \cdot b + \overline{b} \cdot c \cdot \overline{d} + a \cdot c \cdot \overline{d} = a \cdot b + \overline{b} \cdot c \cdot \overline{d}$

d) $x \cdot y + x \cdot (y + z) = x \cdot (y + z)$

e) $a \cdot b + \overline{b} \cdot c \cdot \overline{d} + a \cdot c \cdot \overline{d} = a \cdot b + c \cdot \overline{d}$

Aufgabe 2 – DNF — DF — KF

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 2 – DNF — DF — KF

a) Bestimmen Sie jeweils die disjunktive Normalform (DNF) der folgenden Funktionen:

   i) $f_1(x, y, z) = xyz + \overline{x}y + \overline{y}z$

   ii) $f_2(x, y, z) = xy + \overline{x}\,\overline{y}z + x\overline{y}\,\overline{z}$

b) Bestimmen Sie die primären (Summe der Anzahl verundeter Literale) und sekundären Kosten (Anzahl veroderter Terme) von f_1 und f_2 sowie derer DNFs. Geben Sie auch die Gesamtkosten an.

c) Bringen Sie die folgende Funktion in disjunktive Form (DF):

$$f_3(w, x, y, z) = (\overline{w} + z) \cdot (\overline{w} + x + y) \cdot (x + \overline{y} + z)$$

d) Bringen Sie die folgenden Funktionen in konjunktive Form (KF):

   i) $f_4(x, y, z) = xyz + \overline{x}y + \overline{y}z$

   ii) $f_5(w, x, y, z) = wx + \overline{wz} + w\overline{y}\,\overline{z}$

e) Wie unterscheiden sich die technischen Realisierungen der beiden Formen?

Wiederholung – Übung 5, Begriffsklärung

Minterm (Vollkonjunktion)

Minterme sind eine reine Konjunktion **aller** existierenden Literale in negierter oder nicht negierter Form, deren Funktionswert 1 ist.

Wiederholung – Übung 5, Begriffsklärung

Minterm (Vollkonjunktion)

Minterme sind eine reine Konjunktion **aller** existierenden Literale in negierter oder nicht negierter Form, deren Funktionswert 1 ist.

Merkhilfe: Minterme

Minterme überdecken nur eine Einsstelle in unserem Symmetriediagramm.

Wiederholung – Übung 5, Begriffsklärung

Minterm (Vollkonjunktion)

Minterme sind eine reine Konjunktion **aller** existierenden Literale in negierter oder nicht negierter Form, deren Funktionswert 1 ist.

Merkhilfe: Minterme

Minterme überdecken nur eine Einsstelle in unserem Symmetriediagramm.

Disjunktive Normalform (DNF)

Disjunktion **aller** Minterme. Heißt:

$$\text{DNF}(\varphi) = \bigvee_{i=1}^{\text{Anzahl an Mintermen}} \left(\bigwedge_{j=1}^{\text{Anzahl an Literalen}} L_j \right)$$

Wiederholung – Übung 5, Begriffsklärung

Minterm (Vollkonjunktion)

Minterme sind eine reine Konjunktion **aller** existierenden Literale in negierter oder nicht negierter Form, deren Funktionswert 1 ist.

Merkhilfe: Minterme

Minterme überdecken nur eine Einsstelle in unserem Symmetriediagramm.

Disjunktive Normalform (DNF)

Disjunktion **aller** Minterme. Heißt:

$$\text{DNF}(\varphi) = \bigvee_{i=1}^{\text{Anzahl an Mintermen}} \left(\bigwedge_{j=1}^{\text{Anzahl an Literalen}} L_j \right)$$

Disjunktive Form (DF)

Ein Term ist in disjunktiver Form, wenn er als Disjunktion von Konjunktionen dargestellt werden kann (Summe von Produkten (SoP)).

Beispiel: $(x_1 \cdot x_2) + (x_3 \cdot \overline{x_1})$ ist in disjunktiver Form.

Beispiel: $((x_1 \cdot x_2) + x_3) \cdot \overline{x_1}$ ist **nicht** in disjunktiver Form.

Wiederholung – Übung 5, Begriffsklärung

Maxterm (Volldisjunktion)

Maxterme sind eine reine Disjunktion **aller** existierenden Literale in negierter oder nicht negierter Form, deren Funktionswert 0 ist.

Wiederholung – Übung 5, Begriffsklärung

Maxterm (Volldisjunktion)

Maxterme sind eine reine Disjunktion **aller** existierenden Literale in negierter oder nicht negierter Form, deren Funktionswert 0 ist.

Merkhilfe: Maxterme

Maxterme überdecken nur eine Nullstelle in unserem Symmetriediagramm.

Wiederholung – Übung 5, Begriffsklärung

Maxterm (Volldisjunktion)

Maxterme sind eine reine Disjunktion **aller** existierenden Literale in negierter oder nicht negierter Form, deren Funktionswert 0 ist.

Merkhilfe: Maxterme

Maxterme überdecken nur eine Nullstelle in unserem Symmetriediagramm.

Konjunktive Normalform (KNF)

Konjunktion **aller** Maxterme. Heißt:

$$\text{KNF}(\varphi) = \bigwedge_{i=1}^{\text{Anzahl an Mintermen}} \left(\bigvee_{j=1}^{\text{Anzahl an Literalen}} L_j \right)$$

Wiederholung – Übung 5, Begriffsklärung

Maxterm (Volldisjunktion)

Maxterme sind eine reine Disjunktion **aller** existierenden Literale in negierter oder nicht negierter Form, deren Funktionswert 0 ist.

Merkhilfe: Maxterme

Maxterme überdecken nur eine Nullstelle in unserem Symmetriediagramm.

Konjunktive Normalform (KNF)

Konjunktion **aller** Maxterme. Heißt:

$$\mathrm{KNF}(\varphi) = \bigwedge_{i=1}^{\text{Anzahl an Mintermen}} \left(\bigvee_{j=1}^{\text{Anzahl an Literalen}} L_j \right)$$

Konjunktive Form (KF)

Ein Term ist in konjunktiver Form, wenn er als Konjunktion von Disjunktionen dargestellt werden kann (Produkt von Summen (PoS)).
Beispiel: $(x_1 + x_2) \cdot (x_3 + \overline{x_1})$ ist in konjunktiver Form.
Beispiel: $((x_1 \cdot x_2) + x_3) \cdot \overline{x_1}$ ist **nicht** in konjunktiver Form.

Aufgabe 2 – DNF — DF — KF

a) Bestimmen Sie jeweils die disjunktive Normalform (DNF) der folgenden Funktionen:
   - i) $f_1(x, y, z) = xyz + \overline{x}y + \overline{y}z$
   - ii) $f_2(x, y, z) = xy + \overline{x}\,\overline{y}z + x\overline{y}\,\overline{z}$

b) Bestimmen Sie die primären (Summe der Anzahl verundeter Literale) und sekundären Kosten (Anzahl veroderter Terme) von f_1 und f_2 sowie derer DNFs. Geben Sie auch die Gesamtkosten an.

c) Bringen Sie die folgende Funktion in disjunktive Form (DF):

$$f_3(w, x, y, z) = (\overline{w} + z) \cdot (\overline{w} + x + y) \cdot (x + \overline{y} + z)$$

d) Bringen Sie die folgenden Funktionen in konjunktive Form (KF):
   - i) $f_4(x, y, z) = xyz + \overline{x}y + \overline{y}z$
   - ii) $f_5(w, x, y, z) = wx + \overline{w}z + w\overline{y}\,\overline{z}$

e) Wie unterscheiden sich die technischen Realisierungen der beiden Formen?

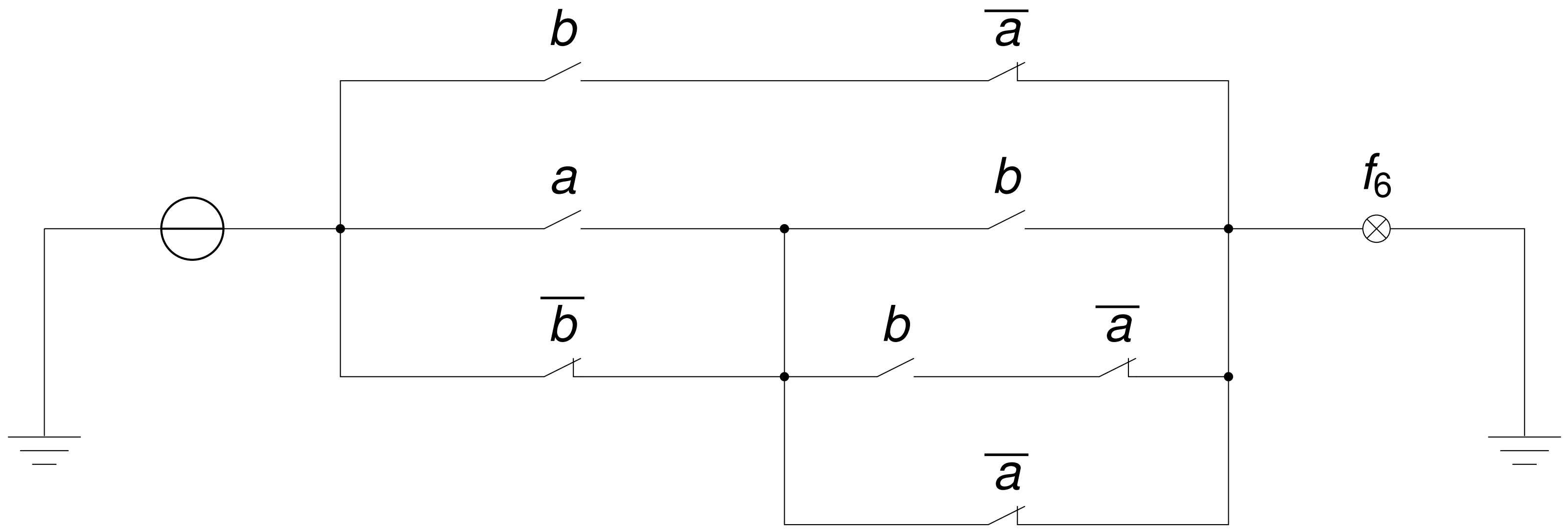
Aufgabe 3 – Relaisschaltnetze

design

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 3 – Relaisschaltnetze

Gegeben sei das im nachstehenden Bild dargestellte Relaisschaltnetz.



a) Bilden Sie daraus den entsprechenden schaltalgebraischen Ausdruck und vereinfachen Sie ihn. Wie wird die realisierte Funktion $f_6(a, b)$ bezeichnet?

b) Entwerfen Sie ein Relaisschaltnetz, das die negierte Konjunktion $f_{nk}(a, b) := \overline{\overline{a} \cdot \overline{b}}$ realisiert.

Aufgabe 4 – Entwicklungssatz

Aufgabe 4 – Entwicklungssatz

Sei folgende Schaltfunktion gegeben:

$$f_7(a, b, c, d) = \overline{a\,c} + b + \overline{d}\,\overline{c} + adc$$

a) Entwickeln Sie f_7 mit Variablenordnung b, c, a, d, bis als Restfunktionen nur noch Konstanten (0 oder 1) übrig bleiben. Geben Sie alle Zwischenschritte an.

b) Zeichnen Sie das resultierende binäre Entscheidungsdiagramm (BDD).

Aufgabe 4 – Entwicklungssatz

Sei folgende Schaltfunktion gegeben:

$$f_7(a, b, c, d) = \overline{a\ c} + b + \overline{d}\ \overline{c} + adc$$

a) Entwickeln Sie f_7 mit Variablenordnung b, c, a, d, bis als Restfunktionen nur noch Konstanten (0 oder 1) übrig bleiben. Geben Sie alle Zwischenschritte an.

b) Zeichnen Sie das resultierende binäre Entscheidungsdiagramm (BDD).

Shannon'scher Entwicklungssatz

Sei $f : B^n \rightarrow B$ eine beliebige boolesche Funktion und $x = (x_1, \ldots, x_i, \ldots, x_n)^T \in B^n$ ein Vektor, an dessen Stelle f ausgewertet wird. Für den Wert der Funktion $f(x)$ gilt dann – unter **Entwicklung** von f nach x_i:

$$f(x_1, \ldots, x_i, \ldots, x_n) = x_i \cdot \underbrace{f(x_1, \ldots, x_{i-1}, 1, x_{i+1}, \ldots, x_n)}_{\text{Kofaktor } f_{x_i}} + \overline{x_i} \cdot \overbrace{f(x_1, \ldots, x_{i-1}, 0, x_{i+1}, \ldots, x_n)}^{\text{Kofaktor } f_{\overline{x_i}}}$$

Zusatzaufgabe – Nutzen des Entwicklungssatzes

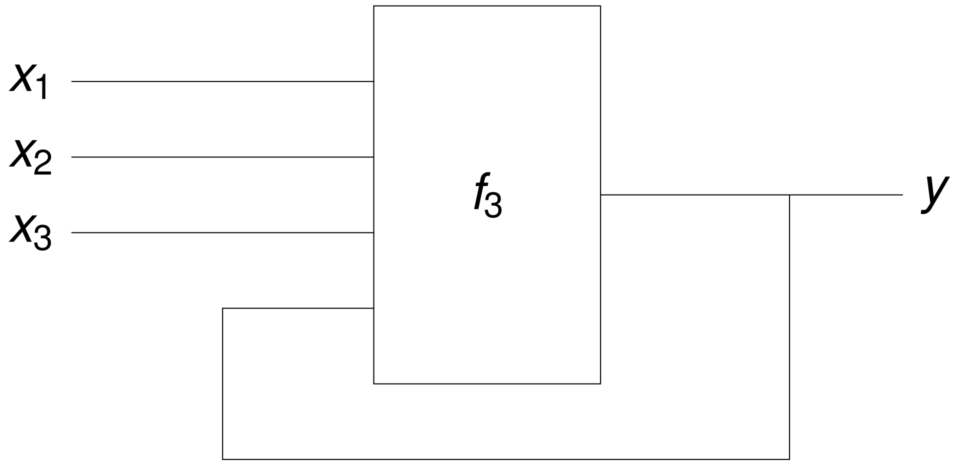
FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Zusatzaufgabe – Nutzen des Entwicklungssatzes

Nehmen Sie an, ein Schaltnetz mit langer Laufzeit und zugehöriger Schaltfunktion

$$y = f_3(x_1, x_2, x_3, x_4)$$

sei folgendermaßen rückgekoppelt:



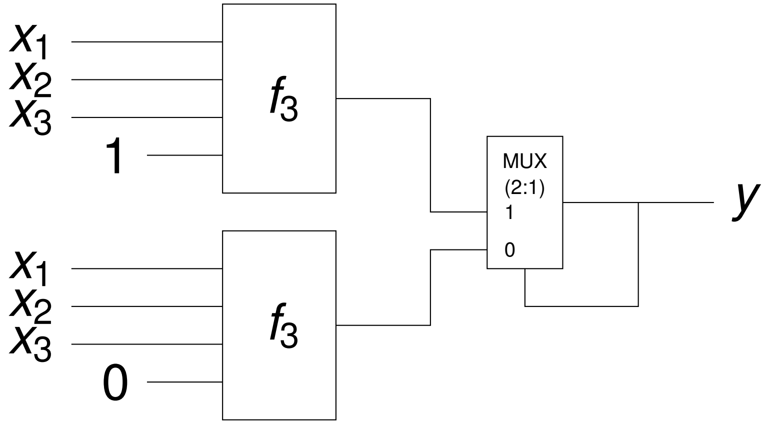
Nutzen Sie den Entwicklungssatz der Schaltalgebra, um den Einfluss der Rückkopplung auf die Laufzeit zu reduzieren und zeichnen Sie das resultierende Schaltnetz.

Lösung der Zusatzaufgabe

Der Trick besteht darin, nach y zu entwickeln:

$$f_3(x_3, x_2, x_1, y) \Leftrightarrow y \cdot f_3(x_3, x_2, x_1, 1) + \overline{y} \cdot f_3(x_3, x_2, x_1, 0)$$

Dadurch besteht die Rückkopplung nur noch daraus, dass y zwischen zwei nicht rückgekoppelten Funktionen auswählt (multiplext):



Entsprechend wird f_3 parallel für beide möglichen Werte von y „vorberechnet“ und die langen Schaltzeiten reduziert, falls sich f_3 durch die Rückkopplung ändern sollte. Der Nachteil liegt darin, dass diese Lösung mehr Fläche benötigt (d.h. Anzahl an Gattern und Leitungen).

Übungen zur Grundlagen der Technischen Informatik

Übung 7 – Symmetriediagramme

Florian Frank

Friedrich-Alexander-Universität Erlangen-Nürnberg

Wintersemester 2018/19

Was machen wir heute?

Aufgabe 1 – Symmetriediagramme

Was machen wir heute?

Aufgabe 1 – Symmetriediagramme

Korrektur und Besprechung der ersten Miniklausur

Aufgabe 1 – Symmetriediagramme

design

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Aufgabe 1 – Symmetriediagramme

a) Seien die vier in der folgenden Funktionstabelle abgebildeten Schaltfunktionen y_1 bis y_4 gegeben, die jeweils abhängig vom Eingangsvektor (x_4, x_3, x_2, x_1) sind.
Geben Sie mithilfe von Symmetriediagrammen jeweils eine disjunktive Minimalform (DMF) und eine konjunktive Minimalform (KMF) an.

Hex	x_4	x_3	x_2	x_1	y_1	y_2	y_3	y_4
0	0	0	0	0	1	—	1	1
1	0	0	0	1	0	-	0	—
2	0	0	1	0	0	0	0	—
3	0	0	1	1	1	1	1	1
4	0	1	0	0	1	—	0	0
5	0	1	0	1	1	—	1	1
6	0	1	1	0	1	1	1	1
7	0	1	1	1	1	0	1	1
8	1	0	0	0	0	—	0	0
9	1	0	0	1	0	—	0	0
A	1	0	1	0	0	1	0	—
B	1	0	1	1	0	0	0	0
C	1	1	0	0	0	—	1	1
D	1	1	0	1	0	—	1	1
E	1	1	1	0	0	0	1	1
F	1	1	1	1	0	1	0	—

Aufgabe 1 – Symmetriediagramme

b) Bestimmen Sie mithilfe des unten gegebenen Symmetriediagramms alle **Primimplikate** der darin spezifizierten Schaltfunktion $f_5(x_4, x_3, x_2, x_1, x_0)$ und geben Sie deren schaltalgebraische Ausdrücke an. Kennzeichnen Sie durch Unterstreichen alle **Kernimplikate**.

x_4

x_0 x_0

–	0	1	0	0	1	0	–
0	1	1	0	0	1	1	0
1	1	1	–	1	1	1	1
–	1	1	0	0	1	1	1

x_1

x_3

x_2

Aufgabe 1 – Begriffsklärung

Wiederholen wir in diesem Zusammenhang einige Begrifflichkeiten aus Vorlesung und Übung:

(1) Was sind Minterme?

(2) Was sind Maxterme?

Aufgabe 1 – Begriffsklärung

Wiederholen wir in diesem Zusammenhang einige Begrifflichkeiten aus Vorlesung und Übung:

(1) Was sind Minterme?

Minterm

Minterme sind eine reine Konjunktion **aller** existierenden Literale in negierter oder nicht negierter Form, deren Funktionswert 1 ist.
graphisch: Terme, die genau eine Einsstelle im Symmetriediagramm überdecken.

(2) Was sind Maxterme?

Aufgabe 1 – Begriffsklärung

Wiederholen wir in diesem Zusammenhang einige Begrifflichkeiten aus Vorlesung und Übung:

(1) Was sind Minterme?

Minterm

Minterme sind eine reine Konjunktion **aller** existierenden Literale in negierter oder nicht negierter Form, deren Funktionswert 1 ist.
graphisch: Terme, die genau eine Einsstelle im Symmetriediagramm überdecken.

(2) Was sind Maxterme?

Maxterm

Maxterme sind eine reine Disjunktion **aller** existierenden Literale in negierter oder nicht negierter Form, deren Funktionswert 0 ist.
graphisch: Terme, die genau eine Nullstelle im Symmetriediagramm überdecken.

Aufgabe 1 – Begriffsklärung

Wiederholen wir in diesem Zusammenhang einige Begrifflichkeiten aus Vorlesung und Übung:

(3) Was sind Primterme?

(4) Was sind Primimplikate?

(5) Was sind Primimplikanten?

Aufgabe 1 – Begriffsklärung

Wiederholen wir in diesem Zusammenhang einige Begrifflichkeiten aus Vorlesung und Übung:

(3) Was sind Primterme?

Primterm

Primterme sind Terme mit minimaler Anzahl von Literalen, die **nur Einsstellen** bzw. **Nullstellen** überdecken.
graphisch: die größtmögliche Eins- bzw. Nullblocküberdeckung im Symmetriediagramm.

(4) Was sind Primimplikate?

(5) Was sind Primimplikanten?

Aufgabe 1 – Begriffsklärung

Wiederholen wir in diesem Zusammenhang einige Begrifflichkeiten aus Vorlesung und Übung:

(3) Was sind Primterme?

(4) Was sind Primimplikate?

Primimplikate

Primimplikate sind Primterme, die nur Nullstellen (mit Freistellen) überdecken
graphisch: die größtmögliche Nullblocküberdeckung (das inkludiert Redundanzstellen) im Symmetriediagramm.

(5) Was sind Primimplikanten?

Aufgabe 1 – Begriffsklärung

Wiederholen wir in diesem Zusammenhang einige Begrifflichkeiten aus Vorlesung und Übung:

(3) Was sind Primterme?

(4) Was sind Primimplikate?

> **Primimplikate**
>
> Primimplikate sind Primterme, die nur Nullstellen (mit Freistellen) überdecken *graphisch*: die größtmögliche Nullblocküberdeckung (das inkludiert Redundanzstellen) im Symmetriediagramm.

(5) Was sind Primimplikanten?

> **Primimplikanten**
>
> Primimplikanten sind Primterme, die nur Einsstellen (mit Freistellen) überdecken. *graphisch*: die größtmögliche Einsblocküberdeckung (das inkludiert Redundanzstellen) im Symmetriediagramm.

Aufgabe 1 – Begriffsklärung

Wiederholen wir in diesem Zusammenhang einige Begrifflichkeiten aus Vorlesung und Übung:

(6) Was ist eine DNF?

(7) Was ist eine KNF?

Aufgabe 1 – Begriffsklärung

Wiederholen wir in diesem Zusammenhang einige Begrifflichkeiten aus Vorlesung und Übung:

(6) Was ist eine DNF?

DNF

Ein Term ist in DNF, wenn er eine Disjunktion **aller** Minterme darstellt.

(7) Was ist eine KNF?

Aufgabe 1 – Begriffsklärung

Wiederholen wir in diesem Zusammenhang einige Begrifflichkeiten aus Vorlesung und Übung:

(6) Was ist eine DNF?

DNF

Ein Term ist in DNF, wenn er eine Disjunktion **aller** Minterme darstellt.

(7) Was ist eine KNF?

KNF

Ein Term ist in KNF, wenn er eine Konjunktion **aller** Maxterme darstellt.

Aufgabe 1 – Begriffsklärung

Wiederholen wir in diesem Zusammenhang einige Begrifflichkeiten aus Vorlesung und Übung:

(8) Was ist eine DF?

(9) Was ist eine KF?

Aufgabe 1 – Begriffsklärung

Wiederholen wir in diesem Zusammenhang einige Begrifflichkeiten aus Vorlesung und Übung:

(8) Was ist eine DF?

DF

Ein Term ist in DF, wenn er als Disjunktion von Konjunktionen dargestellt werden kann (Summe von Produkten (SoP))

(9) Was ist eine KF?

Aufgabe 1 – Begriffsklärung

Wiederholen wir in diesem Zusammenhang einige Begrifflichkeiten aus Vorlesung und Übung:

(8) Was ist eine DF?

DF

Ein Term ist in DF, wenn er als Disjunktion von Konjunktionen dargestellt werden kann (Summe von Produkten (SoP))

(9) Was ist eine KF?

KF

Ein Term ist in KF, wenn er als Konjunktion von Disjunktionen dargestellt werden kann (Produkt von Summen (PoS))

Aufgabe 1 – Begriffsklärung

Wiederholen wir in diesem Zusammenhang einige Begrifflichkeiten aus Vorlesung und Übung:

(10) Was ist eine DMF?

(11) Was ist eine KMF?

Aufgabe 1 – Begriffsklärung

Wiederholen wir in diesem Zusammenhang einige Begrifflichkeiten aus Vorlesung und Übung:

(10) Was ist eine DMF?

DMF

Eine DMF besteht aus einer kostenminimalen Kombination von Primimplikanten, die alle Einsstellen überdecken (kann mehrere geben).
informell: Eine DF, die nicht mehr weiter vereinfacht werden kann.

(11) Was ist eine KMF?

Aufgabe 1 – Begriffsklärung

Wiederholen wir in diesem Zusammenhang einige Begrifflichkeiten aus Vorlesung und Übung:

(10) Was ist eine DMF?

DMF

Eine DMF besteht aus einer kostenminimalen Kombination von Primimplikanten, die alle Einsstellen überdecken (kann mehrere geben).
informell: Eine DF, die nicht mehr weiter vereinfacht werden kann.

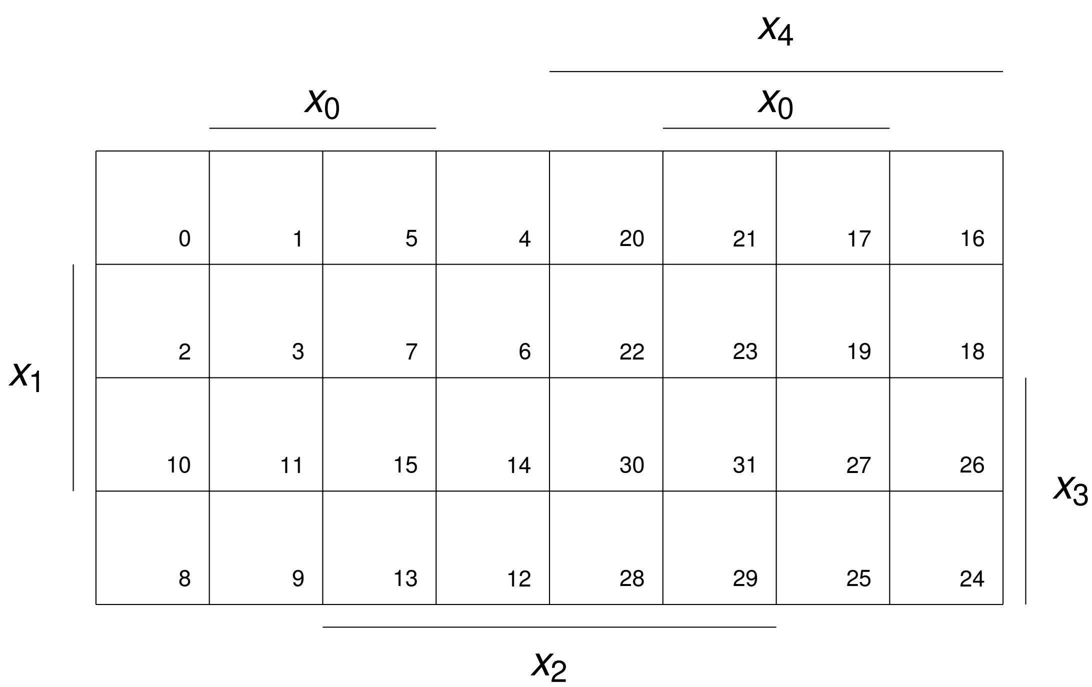
(11) Was ist eine KMF?

KMF

Eine KMF besteht aus einer kostenminimalen Kombination von Primimplikaten, die alle Nullstellen überdecken (kann mehrere geben).
informell: Eine KF, die nicht mehr weiter vereinfacht werden kann.

Aufgabe 1 – Wie bestimme ich Primterme graphisch?

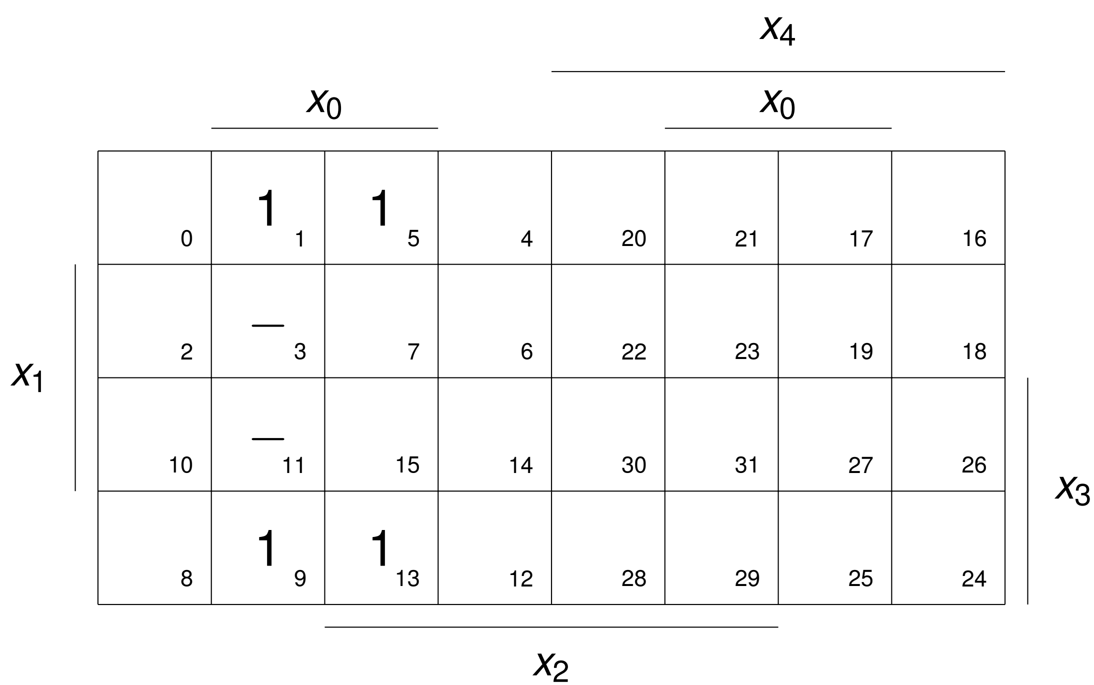
Wir suchen bspw. eine möglichst große Einsstellenüberdeckung. Dazu ...



Aufgabe 1 – Wie bestimme ich Primterme graphisch?

Wir suchen bspw. eine möglichst große Einsstellenüberdeckung. Dazu ...

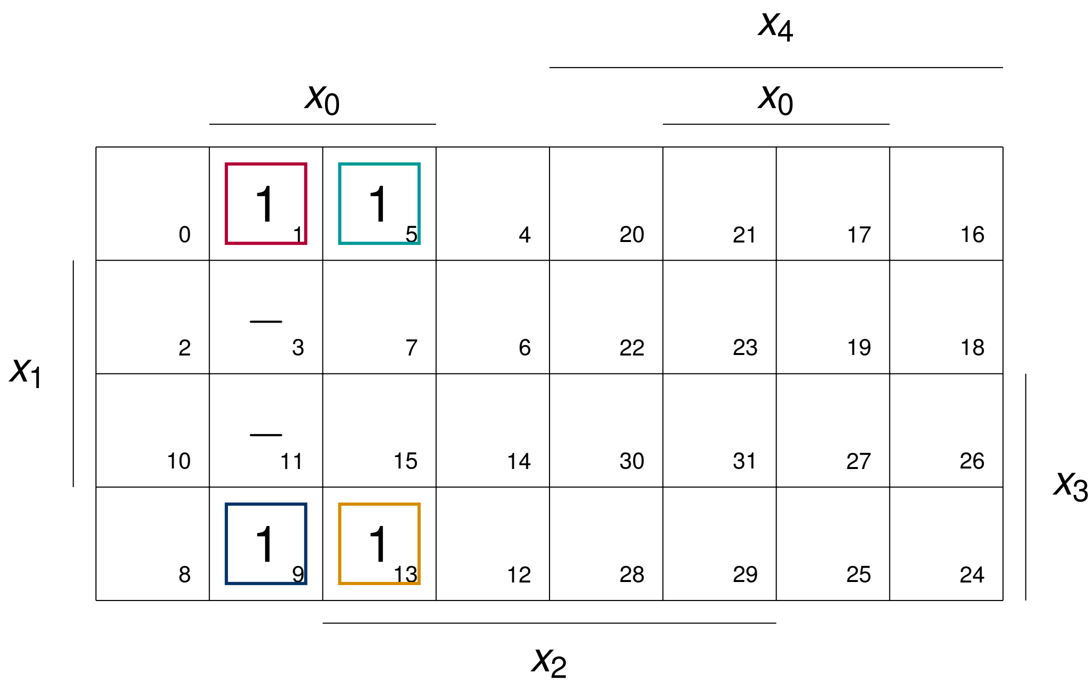
- ... schauen wir mit welchen Symmetrieachsen wir unsere Blöcke vergrößern können.



Aufgabe 1 – Wie bestimme ich Primterme graphisch?

Wir suchen bspw. eine möglichst große Einsstellenüberdeckung. Dazu ...

- ... schauen wir mit welchen Symmetrieachsen wir unsere Blöcke vergrößern können.

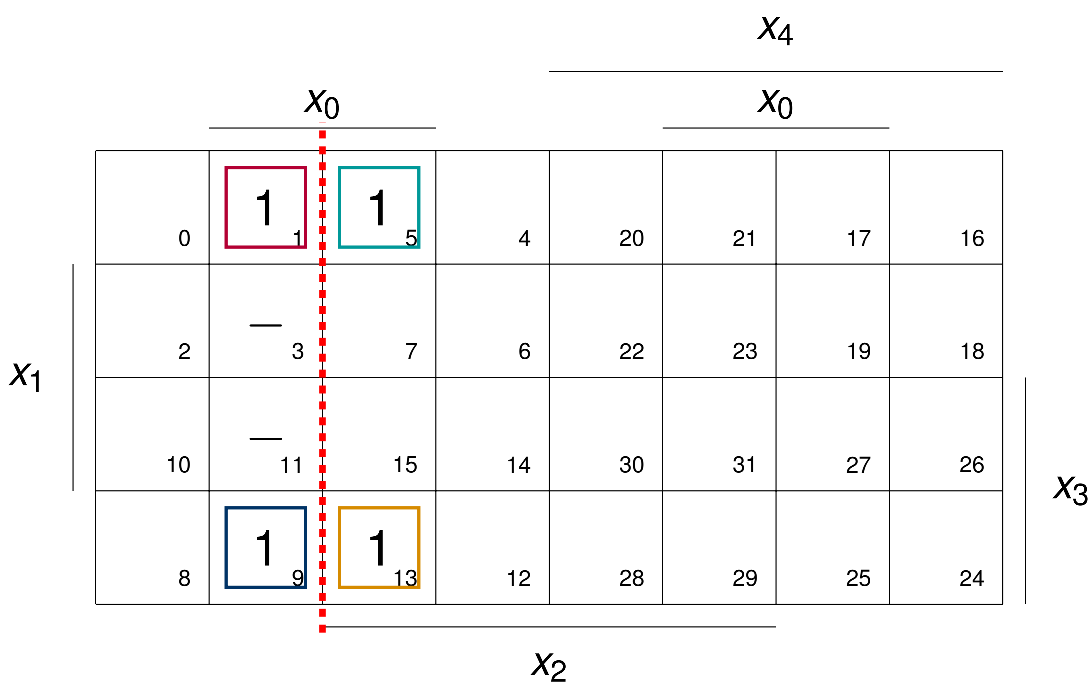


0	1 (1)	1 (5)	4	20	21	17	16
2	— (3)	7	6	22	23	19	18
10	— (11)	15	14	30	31	27	26
8	1 (9)	1 (13)	12	28	29	25	24

Aufgabe 1 – Wie bestimme ich Primterme graphisch?

Wir suchen bspw. eine möglichst große Einsstellenüberdeckung. Dazu ...

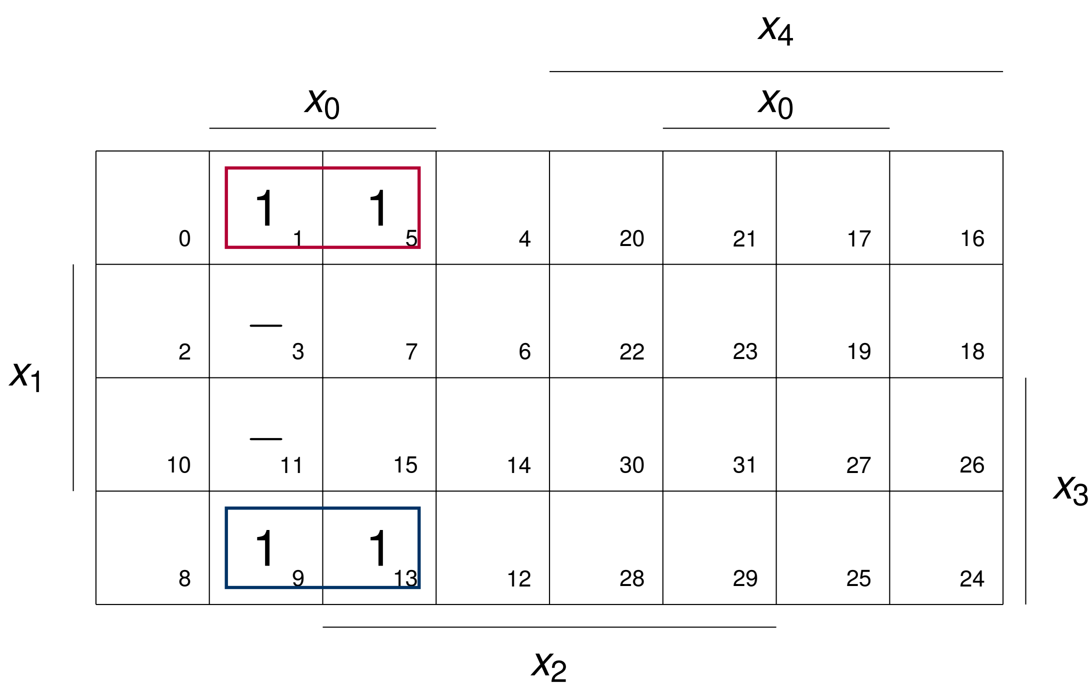
- ... schauen wir mit welchen Symmetrieachsen wir unsere Blöcke vergrößern können.



Aufgabe 1 – Wie bestimme ich Primterme graphisch?

Wir suchen bspw. eine möglichst große Einsstellenüberdeckung. Dazu ...

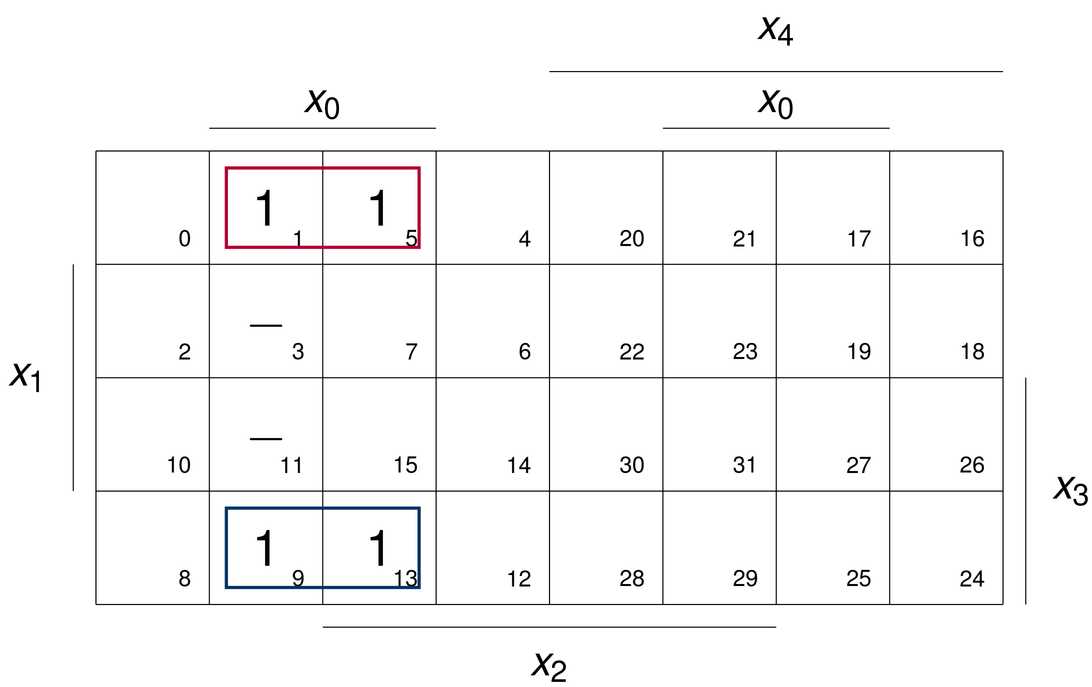
- ... schauen wir mit welchen Symmetrieachsen wir unsere Blöcke vergrößern können.



Aufgabe 1 – Wie bestimme ich Primterme graphisch?

Wir suchen bspw. eine möglichst große Einsstellenüberdeckung. Dazu ...

- ... schauen wir mit welchen Symmetrieachsen wir unsere Blöcke vergrößern können.



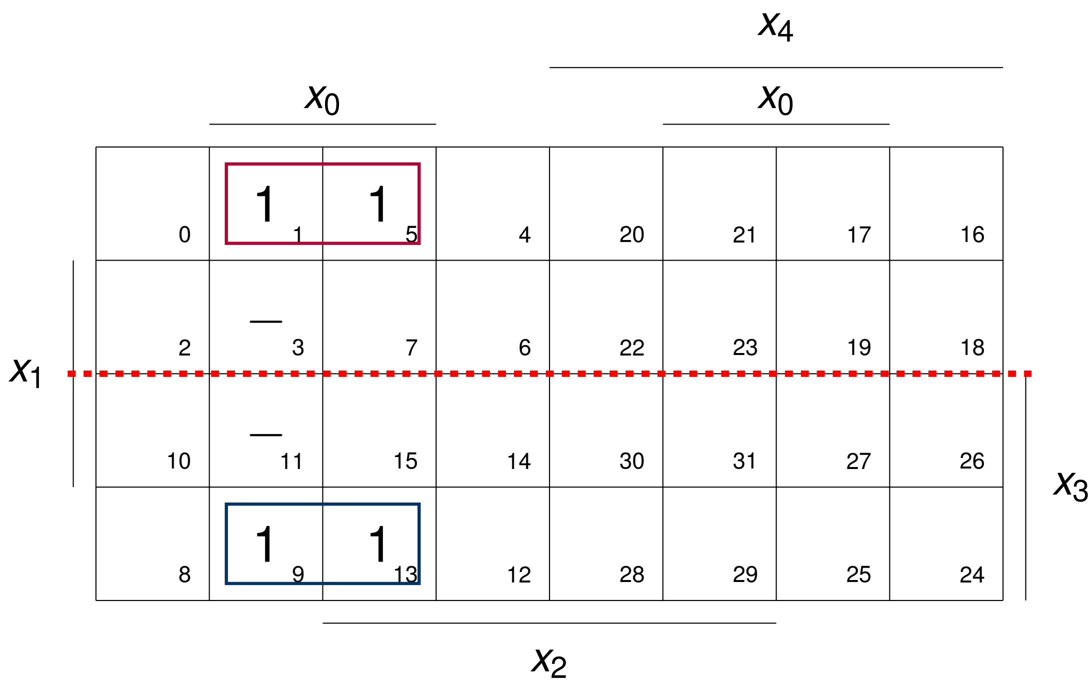
Aufpassen bei Symmetrieblöcken

Blöcke können auch „über“ das Symmetriediagramm „hinausgehen“.

Aufgabe 1 – Wie bestimme ich Primterme graphisch?

Wir suchen bspw. eine möglichst große Einsstellenüberdeckung. Dazu ...

- ... schauen wir mit welchen Symmetrieachsen wir unsere Blöcke vergrößern können.



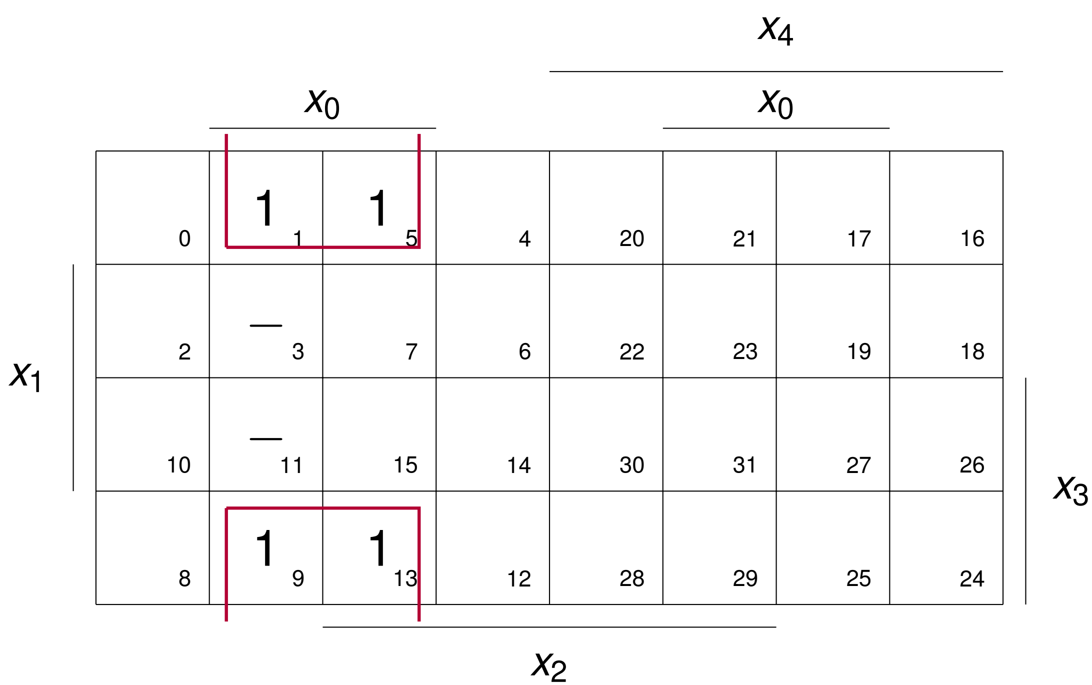
Aufpassen bei Symmetrieblöcken

Blöcke können auch „**über**“ das Symmetriediagramm „**hinaus**gehen“.

Aufgabe 1 – Wie bestimme ich Primterme graphisch?

Wir suchen bspw. eine möglichst große Einsstellenüberdeckung. Dazu ...

- ... schauen wir mit welchen Symmetrieachsen wir unsere Blöcke vergrößern können.



Aufpassen bei Symmetrieblöcken

Blöcke können auch „**über**“ das Symmetriediagramm „**hinaus**gehen“.

Aufgabe 1 – Wie bestimme ich Primterme graphisch?

Wir suchen bspw. eine möglichst große Einsstellenüberdeckung. Dazu ...

- ... schauen wir mit welchen Symmetrieachsen wir unsere Blöcke vergrößern können.

		x_0	x_0		x_4	x_4, x_0	x_4, x_0	x_4
	(0)	1 (1)	1 (5)	(4)	(20)	(21)	(17)	(16)
x_1	1 (2)	— (3)	(7)	(6)	(22)	(23)	1 (19)	— (18)
x_1, x_3	1 (10)	— (11)	(15)	(14)	(30)	(31)	— (27)	— (26)
x_3	(8)	1 (9)	1 (13)	(12)	(28)	(29)	(25)	(24)
			x_2	x_2	x_2	x_2		

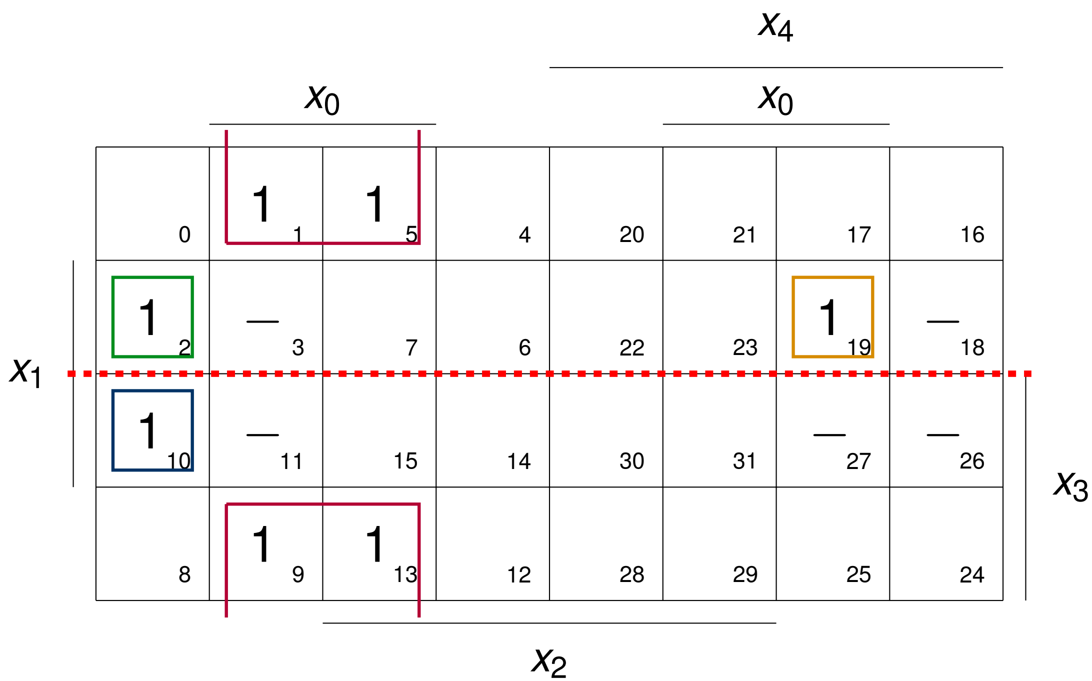
Aufpassen bei Symmetrieblöcken

Blöcke können auch „über“ das Symmetriediagramm „hinausgehen“.

Aufgabe 1 – Wie bestimme ich Primterme graphisch?

Wir suchen bspw. eine möglichst große Einsstellenüberdeckung. Dazu ...

- ... schauen wir mit welchen Symmetrieachsen wir unsere Blöcke vergrößern können.



Aufpassen bei Symmetrieblöcken

Blöcke können auch „über“ das Symmetriediagramm „hinausgehen“.

Aufgabe 1 – Wie bestimme ich Primterme graphisch?

Wir suchen bspw. eine möglichst große Einsstellenüberdeckung. Dazu ...

- ... schauen wir mit welchen Symmetrieachsen wir unsere Blöcke vergrößern können.

					x_4	x_4	x_4	x_4	
		x_0	x_0			x_0	x_0		
	$_{0}$	1 $_{1}$	1 $_{5}$	$_{4}$	$_{20}$	$_{21}$	$_{17}$	$_{16}$	
x_1	1 $_{2}$	— $_{3}$	$_{7}$	$_{6}$	$_{22}$	$_{23}$	1 $_{19}$	— $_{18}$	
x_1	1 $_{10}$	— $_{11}$	$_{15}$	$_{14}$	$_{30}$	$_{31}$	— $_{27}$	— $_{26}$	x_3
	$_{8}$	1 $_{9}$	1 $_{13}$	$_{12}$	$_{28}$	$_{29}$	$_{25}$	$_{24}$	x_3
			x_2	x_2	x_2	x_2			

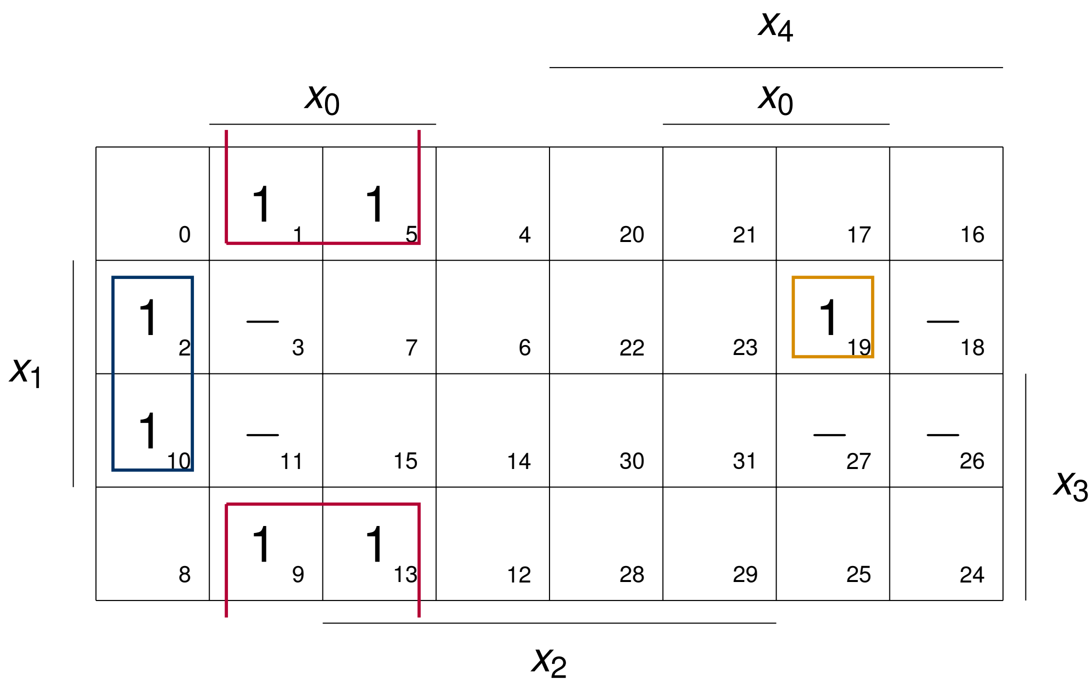
Aufpassen bei Symmetrieblöcken

Blöcke können auch „über“ das Symmetriediagramm „hinausgehen“.

Aufgabe 1 – Wie bestimme ich Primterme graphisch?

Wir suchen bspw. eine möglichst große Einsstellenüberdeckung. Dazu ...

- ... schauen wir mit welchen Symmetrieachsen wir unsere Blöcke vergrößern können.
- ... nehmen wir auch Redundanzstellen mit auf, wenn wir unseren Block dadurch vergrößern können.



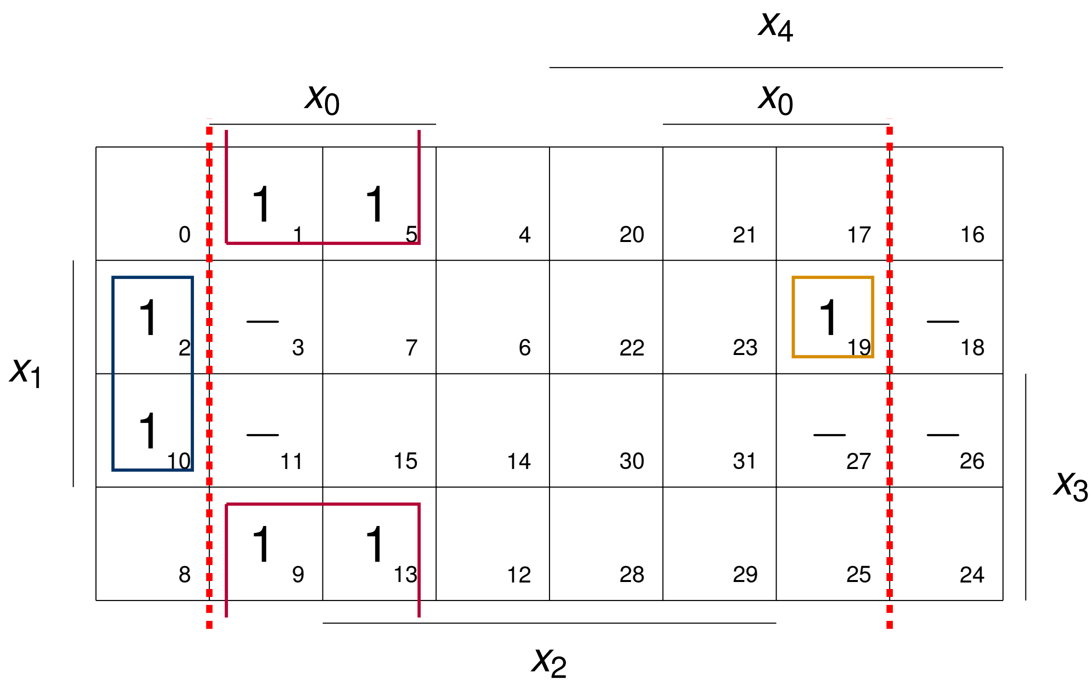
Aufpassen bei Symmetrieblöcken

Blöcke können auch „über“ das Symmetriediagramm „hinausgehen“.

Aufgabe 1 – Wie bestimme ich Primterme graphisch?

Wir suchen bspw. eine möglichst große Einsstellenüberdeckung. Dazu ...

- ... schauen wir mit welchen Symmetrieachsen wir unsere Blöcke vergrößern können.
- ... nehmen wir auch Redundanzstellen mit auf, wenn wir unseren Block dadurch vergrößern können.



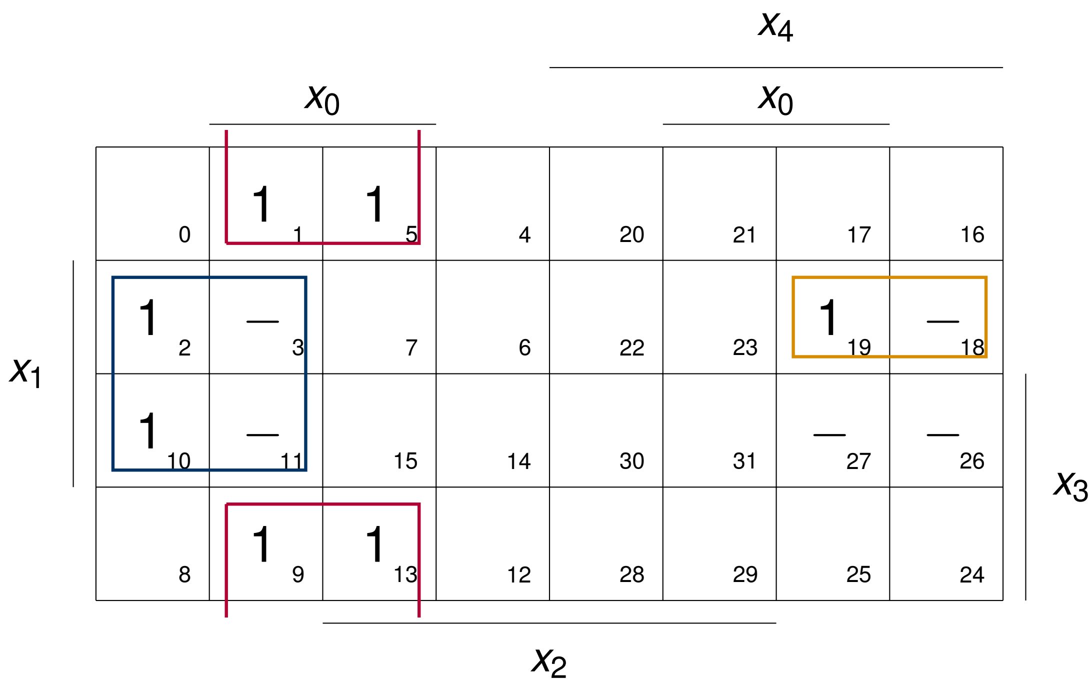
Aufpassen bei Symmetrieblöcken

Blöcke können auch „über“ das Symmetriediagramm „hinausgehen“.

Aufgabe 1 – Wie bestimme ich Primterme graphisch?

Wir suchen bspw. eine möglichst große Einsstellenüberdeckung. Dazu ...

- ... schauen wir mit welchen Symmetrieachsen wir unsere Blöcke vergrößern können.
- ... nehmen wir auch Redundanzstellen mit auf, wenn wir unseren Block dadurch vergrößern können.



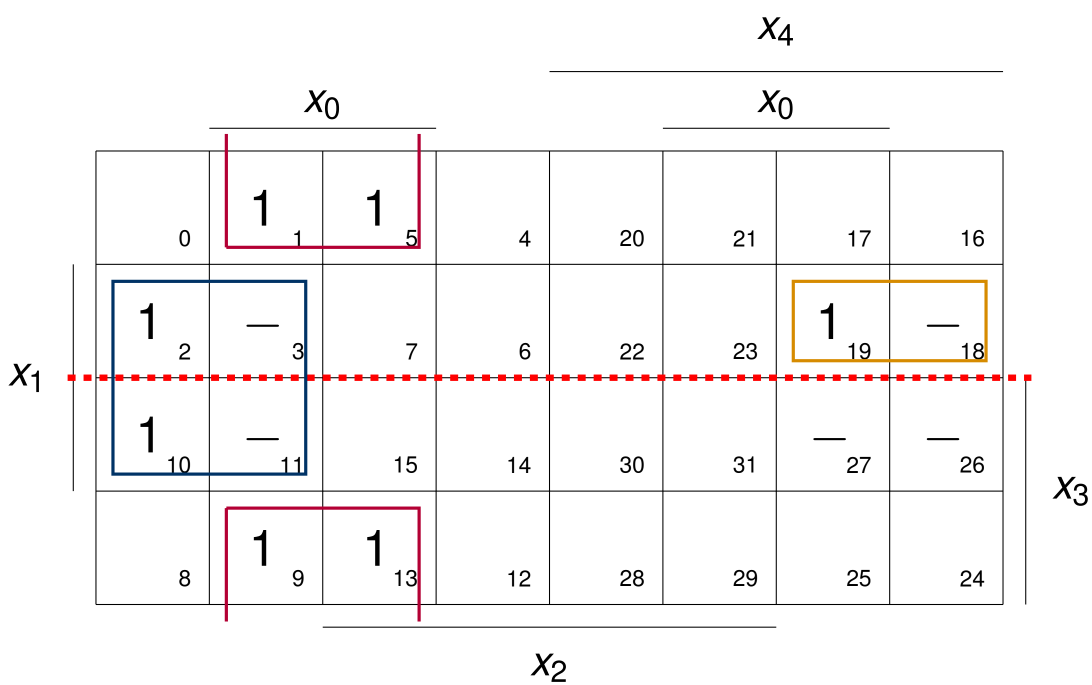
Aufpassen bei Symmetrieblöcken

Blöcke können auch „über“ das Symmetriediagramm „hinausgehen“.

Aufgabe 1 – Wie bestimme ich Primterme graphisch?

Wir suchen bspw. eine möglichst große Einsstellenüberdeckung. Dazu ...

- ... schauen wir mit welchen Symmetrieachsen wir unsere Blöcke vergrößern können.
- ... nehmen wir auch Redundanzstellen mit auf, wenn wir unseren Block dadurch vergrößern können.



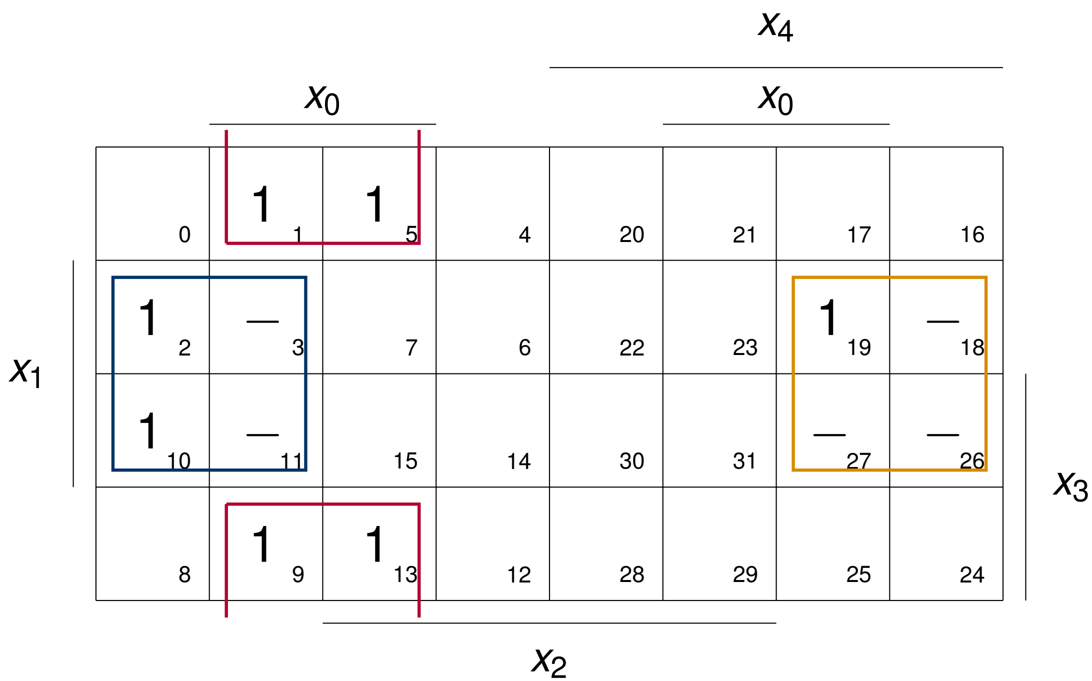
Aufpassen bei Symmetrieblöcken

Blöcke können auch „über“ das Symmetriediagramm „hinausgehen“.

Aufgabe 1 – Wie bestimme ich Primterme graphisch?

Wir suchen bspw. eine möglichst große Einsstellenüberdeckung. Dazu ...

- ... schauen wir mit welchen Symmetrieachsen wir unsere Blöcke vergrößern können.
- ... nehmen wir auch Redundanzstellen mit auf, wenn wir unseren Block dadurch vergrößern können.



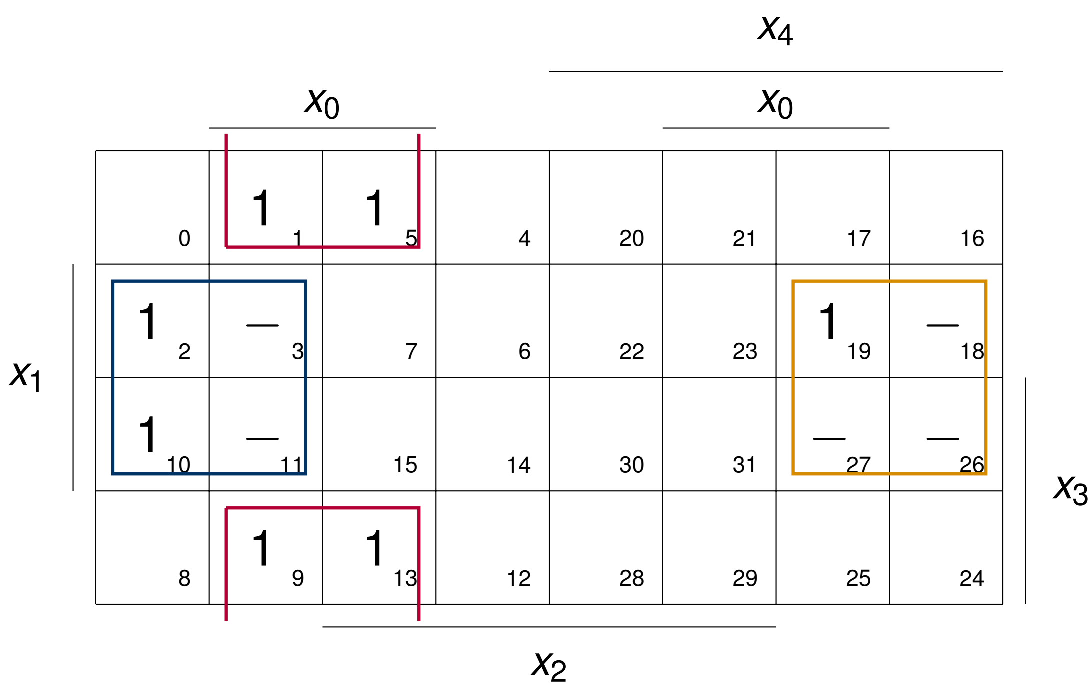
Aufpassen bei Symmetrieblöcken

Blöcke können auch „über“ das Symmetriediagramm „hinausgehen“.

Aufgabe 1 – Wie bestimme ich Primterme graphisch?

Wir suchen bspw. eine möglichst große Einsstellenüberdeckung. Dazu ...

- ... schauen wir mit welchen Symmetrieachsen wir unsere Blöcke vergrößern können.
- ... nehmen wir auch Redundanzstellen mit auf, wenn wir unseren Block dadurch vergrößern können.



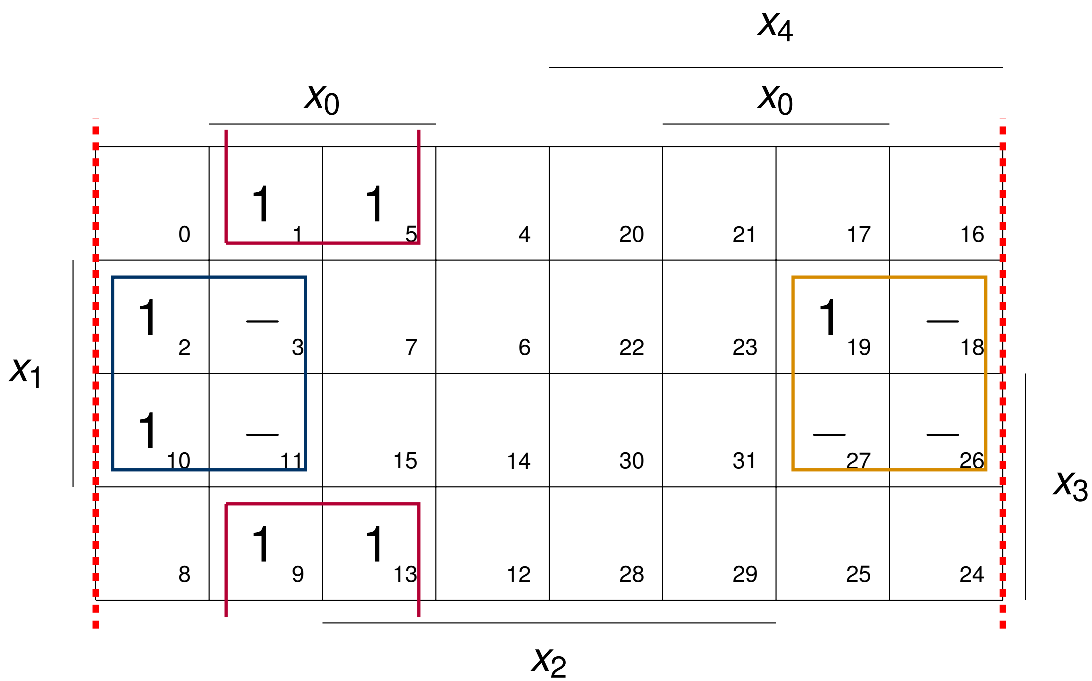
Aufpassen bei Symmetrieblöcken

Blöcke können auch „**über**“ das Symmetriediagramm „**hinaus**gehen“.

Aufgabe 1 – Wie bestimme ich Primterme graphisch?

Wir suchen bspw. eine möglichst große Einsstellenüberdeckung. Dazu ...

- ... schauen wir mit welchen Symmetrieachsen wir unsere Blöcke vergrößern können.
- ... nehmen wir auch Redundanzstellen mit auf, wenn wir unseren Block dadurch vergrößern können.



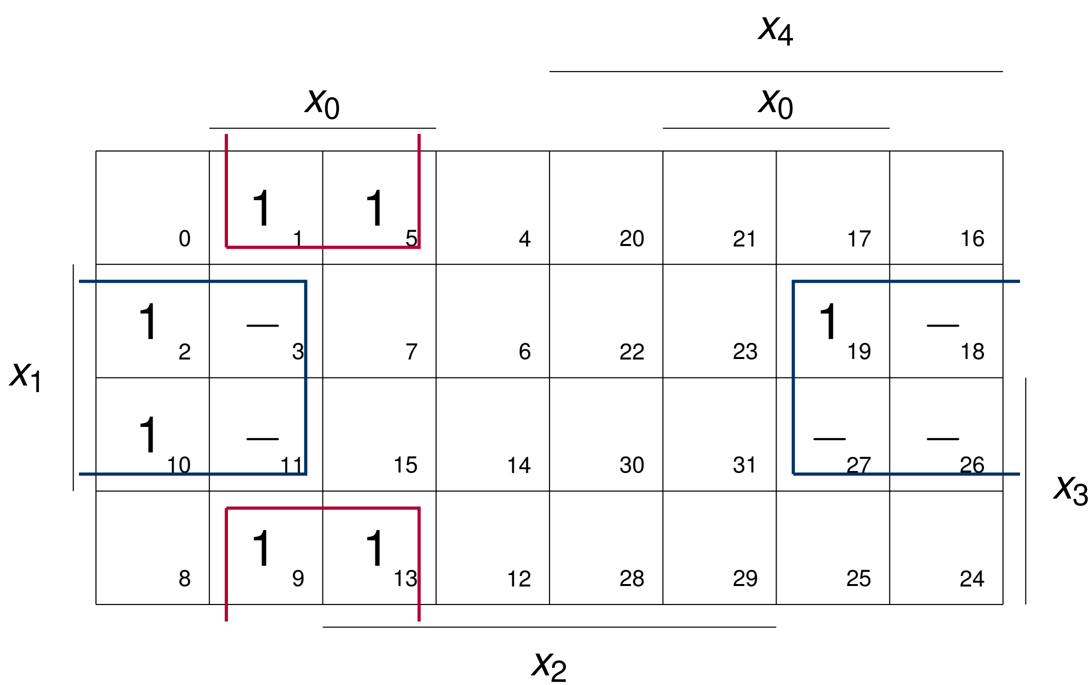
Aufpassen bei Symmetrieblöcken

Blöcke können auch „**über**“ das Symmetriediagramm „**hinaus**gehen“.

Aufgabe 1 – Wie bestimme ich Primterme graphisch?

Wir suchen bspw. eine möglichst große Einsstellenüberdeckung. Dazu ...

- ... schauen wir mit welchen Symmetrieachsen wir unsere Blöcke vergrößern können.
- ... nehmen wir auch Redundanzstellen mit auf, wenn wir unseren Block dadurch vergrößern können.



Aufpassen bei Symmetrieblöcken

Blöcke können auch „**über**“ das Symmetriediagramm „**hinaus**gehen“.

Aufgabe 1 – Symmetriediagramme

a) Seien die vier in der folgenden Funktionstabelle abgebildeten Schaltfunktionen y_1 bis y_4 gegeben, die jeweils abhängig vom Eingangsvektor (x_4, x_3, x_2, x_1) sind.
Geben Sie mithilfe von Symmetriediagrammen jeweils eine disjunktive Minimalform (DMF) und eine konjunktive Minimalform (KMF) an.

Hex	x_4	x_3	x_2	x_1	y_1	y_2	y_3	y_4
0	0	0	0	0	1	—	1	1
1	0	0	0	1	0	-	0	—
2	0	0	1	0	0	0	0	—
3	0	0	1	1	1	1	1	1
4	0	1	0	0	1	—	0	0
5	0	1	0	1	1	—	1	1
6	0	1	1	0	1	1	1	1
7	0	1	1	1	1	0	1	1
8	1	0	0	0	0	—	0	0
9	1	0	0	1	0	—	0	0
A	1	0	1	0	0	1	0	—
B	1	0	1	1	0	0	0	0
C	1	1	0	0	0	—	1	1
D	1	1	0	1	0	—	1	1
E	1	1	1	0	0	0	1	1
F	1	1	1	1	0	1	0	—

Aufgabe 1 – Symmetriediagramme

b) Bestimmen Sie mithilfe des unten gegebenen Symmetriediagramms alle **Primimplikate** der darin spezifizierten Schaltfunktion $f_5(x_4, x_3, x_2, x_1, x_0)$ und geben Sie deren schaltalgebraische Ausdrücke an. Kennzeichnen Sie durch Unterstreichen alle **Kernimplikate**.

					x_4	x_4	x_4	x_4	
		x_0	x_0			x_0	x_0		
	—	0	1	0	0	1	0	—	
x_1	0	1	1	0	0	1	1	0	
x_1	1	1	1	—	1	1	1	1	x_3
	—	1	1	0	0	1	1	1	x_3
			x_2	x_2	x_2	x_2			

Korrektur und Besprechung der ersten Miniklausur

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Übungen zur Grundlagen der Technischen Informatik

Übung 8 – Nelson/Petrick, Überdeckungstabellen und Quine/McCluskey

Florian Frank

Friedrich-Alexander-Universität Erlangen-Nürnberg

Wintersemester 2018/19

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Was machen wir heute?

Aufgabe 1 – Nelson/Petrick-Verfahren

Was machen wir heute?

Aufgabe 1 – Nelson/Petrick-Verfahren

Aufgabe 2 – Überdeckungstabelle

Was machen wir heute?

Aufgabe 1 – Nelson/Petrick-Verfahren

Aufgabe 2 – Überdeckungstabelle

Aufgabe 3 – Quine/McCluskey-Verfahren

Was machen wir heute?

Aufgabe 1 – Nelson/Petrick-Verfahren

Aufgabe 2 – Überdeckungstabelle

Aufgabe 3 – Quine/McCluskey-Verfahren

Aufgabe 4 – Relation zur Vorweihnachtszeit

Aufgabe 1 – Nelson/Petrick-Verfahren

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 1 – Nelson/Petrick-Verfahren

Sei die folgende Funktionstabelle gegeben:

Dezimal	x_4	x_3	x_2	x_1	y_0	y_1	Dezimal	x_4	x_3	x_2	x_1	y_0	y_1
0	0	0	0	0	0	0	8	1	0	0	0	1	1
1	0	0	0	1	0	0	9	1	0	0	1	1	1
2	0	0	1	0	1	1	10	1	0	1	0	1	1
3	0	0	1	1	1	1	11	1	0	1	1	1	1
4	0	1	0	0	0	0	12	1	1	0	0	1	1
5	0	1	0	1	1	1	13	1	1	0	1	1	0
6	0	1	1	0	1	0	14	1	1	1	0	1	1
7	0	1	1	1	1	1	15	1	1	1	1	1	1

a) Ermitteln Sie alle Primimplikanten für die Funktion y_0 mithilfe des Nelson-Verfahrens.

b) Bestimmen Sie eine disjunktive Minimalform (DMF) für die Funktion y_1 mittels des Nelson/Petrick-Verfahrens.

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Nelson-Verfahren (Bestimmung von Primimplikanten)

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Nelson-Verfahren (Bestimmung von Primimplikanten)

Schritt 1 – Einsstellenergänzung

Verfüge **alle** Redundanzstellen (Freistellen) zu Einsstellen

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Nelson-Verfahren (Bestimmung von Primimplikanten)

Schritt 1 – Einsstellenergänzung

Verfüge **alle** Redundanzstellen (Freistellen) zu Einsstellen

Schritt 2 – Nullblocküberdeckung

Erstelle eine Nullblocküberdeckung für die Einsstellenergänzung und ...

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Nelson-Verfahren (Bestimmung von Primimplikanten)

Schritt 1 – Einsstellenergänzung

Verfüge **alle** Redundanzstellen (Freistellen) zu Einsstellen

Schritt 2 – Nullblocküberdeckung

Erstelle eine Nullblocküberdeckung für die Einsstellenergänzung und ...

Schritt 3 – Aufstellen einer konjunktiven Form

... stelle daraus eine KF für die Einsstellenergänzung auf.

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Nelson-Verfahren (Bestimmung von Primimplikanten)

Schritt 1 – Einsstellenergänzung
Verfüge **alle** Redundanzstellen (Freistellen) zu Einsstellen

Schritt 2 – Nullblocküberdeckung
Erstelle eine Nullblocküberdeckung für die Einsstellenergänzung und ...

Schritt 3 – Aufstellen einer konjunktiven Form
... stelle daraus eine KF für die Einsstellenergänzung auf.

Schritt 4 – Mache die KF zur DF
Umwandeln der eben erstellten KF in eine äquivalente DF durch Anwenden von logischen Umformungen

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Nelson-Verfahren (Bestimmung von Primimplikanten)

Schritt 1 – Einsstellenergänzung

Verfüge **alle** Redundanzstellen (Freistellen) zu Einsstellen

Schritt 2 – Nullblocküberdeckung

Erstelle eine Nullblocküberdeckung für die Einsstellenergänzung und ...

Schritt 3 – Aufstellen einer konjunktiven Form

... stelle daraus eine KF für die Einsstellenergänzung auf.

Schritt 4 – Mache die KF zur DF

Umwandeln der eben erstellten KF in eine äquivalente DF durch Anwenden von logischen Umformungen

Schritt 5 – Streichen aller reinen Redundanzterme

Streiche alle Terme der eben erstellten DF, die nur Freistellen überdecken.

Aufgabe 1 – Nelson/Petrick-Verfahren

Sei die folgende Funktionstabelle gegeben:

Dezimal	x_4	x_3	x_2	x_1	y_0	y_1	Dezimal	x_4	x_3	x_2	x_1	y_0	y_1
0	0	0	0	0	0	0	8	1	0	0	0	1	1
1	0	0	0	1	0	0	9	1	0	0	1	1	1
2	0	0	1	0	1	1	10	1	0	1	0	1	1
3	0	0	1	1	1	1	11	1	0	1	1	1	1
4	0	1	0	0	0	0	12	1	1	0	0	1	1
5	0	1	0	1	1	1	13	1	1	0	1	1	0
6	0	1	1	0	1	0	14	1	1	1	0	1	1
7	0	1	1	1	1	1	15	1	1	1	1	1	1

a) Ermitteln Sie alle Primimplikanten für die Funktion y_0 mithilfe des Nelson-Verfahrens.

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Petrick-Verfahren (Kostenminimale Auswahl der Primterme)

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Petrick-Verfahren (Kostenminimale Auswahl der Primterme)

Schritt 1 – Bilden des Petrick-Ausdrucks

Konjungiere die in der Überdeckungstabelle spaltenweise disjungierten Implikanten

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Petrick-Verfahren (Kostenminimale Auswahl der Primterme)

Schritt 1 – Bilden des Petrick-Ausdrucks

Konjungiere die in der Überdeckungstabelle spaltenweise disjungierten Implikanten

Schritt 2 – Vereinfachen des Petrick-Ausdrucks

Wende dazu das Absorptions- und Distributivgesetz auf den Petrick-Ausdruck an.

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Petrick-Verfahren (Kostenminimale Auswahl der Primterme)

Schritt 1 – Bilden des Petrick-Ausdrucks
: Konjungiere die in der Überdeckungstabelle spaltenweise disjungierten Implikanten

Schritt 2 – Vereinfachen des Petrick-Ausdrucks
: Wende dazu das Absorptions- und Distributivgesetz auf den Petrick-Ausdruck an.

Schritt 3 – Herausfinden der kostenminimalen Lösung(en)
: Wähle den/die kostenminimalsten Disjunkt(e) des vereinfachten Petrick-Ausdrucks aus $\rightarrow$ Minimalform.

Aufgabe 1 – Nelson/Petrick-Verfahren

Sei die folgende Funktionstabelle gegeben:

Dezimal	x_4	x_3	x_2	x_1	y_0	y_1	Dezimal	x_4	x_3	x_2	x_1	y_0	y_1
0	0	0	0	0	0	0	8	1	0	0	0	1	1
1	0	0	0	1	0	0	9	1	0	0	1	1	1
2	0	0	1	0	1	1	10	1	0	1	0	1	1
3	0	0	1	1	1	1	11	1	0	1	1	1	1
4	0	1	0	0	0	0	12	1	1	0	0	1	1
5	0	1	0	1	1	1	13	1	1	0	1	1	0
6	0	1	1	0	1	0	14	1	1	1	0	1	1
7	0	1	1	1	1	1	15	1	1	1	1	1	1

b) Bestimmen Sie eine disjunktive Minimalform (DMF) für die Funktion y_1 mittels des Nelson/Petrick-Verfahrens.

Aufgabe 2 – Überdeckungstabelle

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 2 – Überdeckungstabelle

Lösen Sie das folgende Überdeckungsproblem tabellarisch mittels einer Überdeckungstabelle unter Angabe der verwendeten Regeln. Geben Sie zudem eine DMF der beschriebenen Schaltfunktion $g(e, d, c, b, a)$ an.

Primimplikant	2	8	10	11	26	29	31	Kosten c_k
$\bar{e}\bar{c}b$	$\times$		$\times$	$\times$				5
$\bar{e}ba$				$\times$				5
$d\bar{c}b$			$\times$	$\times$	$\times$			5
dba							$\times$	5
dca						$\times$	$\times$	5
$\bar{c}\bar{a}$	$\times$	$\times$	$\times$		$\times$			2
$\bar{e}d$		$\times$	$\times$	$\times$				2

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Überdeckungstabellen (Graphisches Äquivalent zum Petrick-Verfahren)

Grundsatzidee: Minimierung des logischen Ausdrucks durch optimale Selektion der Primterme, die zu einer vollständigen Überdeckung aller Einsstellen führt.
Dazu geht man folgende Schritte:

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Überdeckungstabellen (Graphisches Äquivalent zum Petrick-Verfahren)

Grundsatzidee: Minimierung des logischen Ausdrucks durch optimale Selektion der Primterme, die zu einer vollständigen Überdeckung aller Einsstellen führt.
Dazu geht man folgende Schritte:

Schritt 1 – Regel der Kernimplikanten
: Finde Kerne (einziges Kreuz in einer Spalte) und streiche die Spalte und Schnittspalten.

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Überdeckungstabellen (Graphisches Äquivalent zum Petrick-Verfahren)

Grundsatzidee: Minimierung des logischen Ausdrucks durch optimale Selektion der Primterme, die zu einer vollständigen Überdeckung aller Einsstellen führt.
Dazu geht man folgende Schritte:

Schritt 1 – Regel der Kernimplikanten
Finde Kerne (einziges Kreuz in einer Spalte) und streiche die Spalte und Schnittspalten.

Schritt 2 – Regel der Spaltendominanz
Streiche dominierende Spalten (eine Obermenge einer anderen Spalte).

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Überdeckungstabellen (Graphisches Äquivalent zum Petrick-Verfahren)

Grundsatzidee: Minimierung des logischen Ausdrucks durch optimale Selektion der Primterme, die zu einer vollständigen Überdeckung aller Einsstellen führt.
Dazu geht man folgende Schritte:

Schritt 1 – Regel der Kernimplikanten
: Finde Kerne (einziges Kreuz in einer Spalte) und streiche die Spalte und Schnittspalten.

Schritt 2 – Regel der Spaltendominanz
: Streiche dominierende Spalten (eine Obermenge einer anderen Spalte).

Schritt 3 – Regel der Zeilendominanz
: Streiche dominierte Zeilen **nur** wenn sie mehr kostet als ihre dominierende Zeile oder keine Zeile(-nkombination) existiert, welche die fehlenden Einsstellen überdeckt und weniger kostet als die Differenz.

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Überdeckungstabellen (Graphisches Äquivalent zum Petrick-Verfahren)

Grundsatzidee: Minimierung des logischen Ausdrucks durch optimale Selektion der Primterme, die zu einer vollständigen Überdeckung aller Einsstellen führt.
Dazu geht man folgende Schritte:

Schritt 1 – Regel der Kernimplikanten
Finde Kerne (einziges Kreuz in einer Spalte) und streiche die Spalte und Schnittspalten.

Schritt 2 – Regel der Spaltendominanz
Streiche dominierende Spalten (eine Obermenge einer anderen Spalte).

Schritt 3 – Regel der Zeilendominanz
Streiche dominierte Zeilen **nur** wenn sie mehr kostet als ihre dominierende Zeile oder keine Zeile(-nkombination) existiert, welche die fehlenden Einsstellen überdeckt und weniger kostet als die Differenz.

Schritt 4 – Wiederhole Schritte 1 bis 3 solange bis kein Schritt mehr anwendbar ist

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Überdeckungstabellen (Graphisches Äquivalent zum Petrick-Verfahren)

Grundsatzidee: Minimierung des logischen Ausdrucks durch optimale Selektion der Primterme, die zu einer vollständigen Überdeckung aller Einsstellen führt.
Dazu geht man folgende Schritte:

Schritt 1 – Regel der Kernimplikanten
Finde Kerne (einziges Kreuz in einer Spalte) und streiche die Spalte und Schnittspalten.

Schritt 2 – Regel der Spaltendominanz
Streiche dominierende Spalten (eine Obermenge einer anderen Spalte).

Schritt 3 – Regel der Zeilendominanz
Streiche dominierte Zeilen **nur** wenn sie mehr kostet als ihre dominierende Zeile oder keine Zeile(-nkombination) existiert, welche die fehlenden Einsstellen überdeckt und weniger kostet als die Differenz.

Schritt 4 – Wiederhole Schritte 1 bis 3 solange bis kein Schritt mehr anwendbar ist

Problem: Es kann zu zyklischen Resttabellen kommen ($\rightarrow$ wende dann das Petrick-Verfahren an.)

Aufgabe 2 – Überdeckungstabelle

Lösen Sie das folgende Überdeckungsproblem tabellarisch mittels einer Überdeckungstabelle unter Angabe der verwendeten Regeln. Geben Sie zudem eine DMF der beschriebenen Schaltfunktion $g(e, d, c, b, a)$ an.

Primimplikant	2	8	10	11	26	29	31	Kosten c_k
$\bar{e}\bar{c}b$	$\times$		$\times$	$\times$				5
$\bar{e}ba$				$\times$				5
$d\bar{c}b$			$\times$	$\times$	$\times$			5
dba							$\times$	5
dca						$\times$	$\times$	5
$\bar{c}\bar{a}$	$\times$	$\times$	$\times$		$\times$			2
$\bar{e}d$		$\times$	$\times$	$\times$				2

Aufgabe 3 – Quine/McCluskey-Verfahren

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 3 – Quine/McCluskey-Verfahren

Auf einer Siebensegmentanzeige soll die einstellige Hexadezimaldarstellung des Bitvektors *DCBA* angezeigt werden. Die folgende Funktionstabelle gibt die Ansteuerfunkionen für die Leuchtbalken *a* bis *g* an (vergleiche Abbildung 2). Optimieren Sie mithilfe des Verfahrens von Quine und McCluskey die Schaltfunktion für den Leuchtbalken *a*.

Abbildung 2: Darstellung von Hexadezimalzahlen auf einer Sieben-Segment-Anzeige.

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Quine/McCluskey-Verfahren (Bestimmung von Primimplikanten)

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Quine/McCluskey-Verfahren (Bestimmung von Primimplikanten)

Einsstellenergänzung

Verfüge **alle** Redundanzstellen (Freistellen) zu Einsstellen

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Quine/McCluskey-Verfahren (Bestimmung von Primimplikanten)

Einsstellenergänzung

Verfüge **alle** Redundanzstellen (Freistellen) zu Einsstellen

Schritt 1 – Bilden einer disjunktiven Normalform

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Quine/McCluskey-Verfahren (Bestimmung von Primimplikanten)

Einsstellenergänzung
Verfüge **alle** Redundanzstellen (Freistellen) zu Einsstellen

Schritt 2 – Bilden einer disjunktiven Normalform

Schritt 3 – Einteilung der Implikanten zu Klassen $Q_{i,j}$
$Q_{i,j}$ bezeichne dabei die Klasse der Implikanten mit i Literalen, von denen j negiert vorkommen (also gilt logischerweise $0 \leqslant j \leqslant i$)

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Quine/McCluskey-Verfahren (Bestimmung von Primimplikanten)

Einsstellenergänzung
Verfüge **alle** Redundanzstellen (Freistellen) zu Einsstellen

Schritt 2 – Bilden einer disjunktiven Normalform

Schritt 3 – Einteilung der Implikanten zu Klassen $Q_{i,j}$
$Q_{i,j}$ bezeichne dabei die Klasse der Implikanten mit i Literalen, von denen j negiert vorkommen (also gilt logischerweise $0 \leqslant j \leqslant i$)

Schritt 4 – Reduktion zweier „benachbarter“ Klassen $Q_{i,j}$ und $Q_{i,j-1}$
Wende dazu das Distributivgesetz i.V.m. den Komplementärgesetzen an. Fasse die entstehenden Terme wieder in einer neuen Klasse $Q_{i-1,j-1}$ zusammen. Markiere bereits berücksichtigte Terme.

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Quine/McCluskey-Verfahren (Bestimmung von Primimplikanten)

Einsstellenergänzung
Verfüge **alle** Redundanzstellen (Freistellen) zu Einsstellen

Schritt 2 – Bilden einer disjunktiven Normalform

Schritt 3 – Einteilung der Implikanten zu Klassen $Q_{i,j}$
$Q_{i,j}$ bezeichne dabei die Klasse der Implikanten mit i Literalen, von denen j negiert vorkommen (also gilt logischerweise $0 \leqslant j \leqslant i$)

Schritt 4 – Reduktion zweier „benachbarter" Klassen $Q_{i,j}$ und $Q_{i,j-1}$
Wende dazu das Distributivgesetz i.V.m. den Komplementärgesetzen an. Fasse die entstehenden Terme wieder in einer neuen Klasse $Q_{i-1,j-1}$ zusammen. Markiere bereits berücksichtigte Terme.

Schritt 5 – Wiederhole Schritt 4 solange bis keine Reduktion mehr möglich ist

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Quine/McCluskey-Verfahren (Bestimmung von Primimplikanten)

Einsstellenergänzung
Verfüge **alle** Redundanzstellen (Freistellen) zu Einsstellen

Schritt 2 – Bilden einer disjunktiven Normalform

Schritt 3 – Einteilung der Implikanten zu Klassen $Q_{i,j}$
$Q_{i,j}$ bezeichne dabei die Klasse der Implikanten mit i Literalen, von denen j negiert vorkommen (also gilt logischerweise $0 \leqslant j \leqslant i$)

Schritt 4 – Reduktion zweier „benachbarter" Klassen $Q_{i,j}$ und $Q_{i,j-1}$
Wende dazu das Distributivgesetz i.V.m. den Komplementärgesetzen an. Fasse die entstehenden Terme wieder in einer neuen Klasse $Q_{i-1,j-1}$ zusammen. Markiere bereits berücksichtigte Terme.

Schritt 5 – Wiederhole Schritt 4 solange bis keine Reduktion mehr möglich ist

Schritt 6 – „Streiche" all diejenigen Terme, die nur Redundanzstellen überdecken
Die restlichen Terme sind die gesuchten Primimplikanten.

Minimierung von Schaltfunktionen – Verschiedene Verfahren im Überblick

Quine/McCluskey-Verfahren (Bestimmung von Primimplikanten)

Einsstellenergänzung
Verfüge **alle** Redundanzstellen (Freistellen) zu Einsstellen

Schritt 2 – Bilden einer disjunktiven Normalform

Schritt 3 – Einteilung der Implikanten zu Klassen $Q_{i,j}$
$Q_{i,j}$ bezeichne dabei die Klasse der Implikanten mit i Literalen, von denen j negiert vorkommen (also gilt logischerweise $0 \leqslant j \leqslant i$)

Schritt 4 – Reduktion zweier „benachbarter“ Klassen $Q_{i,j}$ und $Q_{i,j-1}$
Wende dazu das Distributivgesetz i.V.m. den Komplementärgesetzen an. Fasse die entstehenden Terme wieder in einer neuen Klasse $Q_{i-1,j-1}$ zusammen. Markiere bereits berücksichtigte Terme.

Schritt 5 – Wiederhole Schritt 4 solange bis keine Reduktion mehr möglich ist

Schritt 6 – „Streiche“ all diejenigen Terme, die nur Redundanzstellen überdecken
Die restlichen Terme sind die gesuchten Primimplikanten.

Suche die kostenminimalste Überdeckung weiterhin mit dem Petrick-Verfahren oder einer Überdeckungstabelle.

Aufgabe 3 – Quine/McCluskey-Verfahren

Auf einer Siebensegmentanzeige soll die einstellige Hexadezimaldarstellung des Bitvektors *DCBA* angezeigt werden. Die folgende Funktionstabelle gibt die Ansteuerfunkionen für die Leuchtbalken *a* bis *g* an (vergleiche Abbildung 2). Optimieren Sie mithilfe des Verfahrens von Quine und McCluskey die Schaltfunktion für den Leuchtbalken *a*.

Hex	*D*	*C*	*B*	*A*	*a*	*b*	*c*	*d*	*e*	*f*	*g*
0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	1	1	0	0	0	0
2	0	0	1	0	1	1	0	1	1	0	1
3	0	0	1	1	1	1	1	1	0	0	1
4	0	1	0	0	0	1	1	0	0	1	1
5	0	1	0	1	1	0	1	1	0	1	1
6	0	1	1	0	1	0	1	1	1	1	1
7	0	1	1	1	1	1	1	0	0	0	0
8	1	0	0	0	1	1	1	1	1	1	1
9	1	0	0	1	1	1	1	1	0	1	1
A	1	0	1	0	1	1	1	0	1	1	1
b	1	0	1	1	0	0	1	1	1	1	1
c	1	1	0	0	0	0	0	1	1	0	1
d	1	1	0	1	0	1	1	1	1	0	1
E	1	1	1	0	1	0	0	1	1	1	1
F	1	1	1	1	1	0	0	0	1	1	1

Aufgabe 4 – Relation zur Vorweihnachtszeit

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 4 – Relation zur Vorweihnachtszeit

In der Vorweihnachtszeit stellt sich die Frage, was besser ist: *ewiges Glück* oder ein **Lebkuchenherz**?
Man sollte meinen, dass nichts besser ist als *ewiges Glück*. Andererseits ist ein **Lebkuchenherz** sicherlich besser als nichts.
„Besser“ ist bekanntlich eine transitive Relation. Was folgt daraus?

Übungen zur Grundlagen der Technischen Informatik

Übung 9 – CMOS, PAL, NAND, Latches und Flipflops

Florian Frank

Friedrich-Alexander-Universität Erlangen-Nürnberg

Wintersemester 2018/19

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Was machen wir heute?

Aufgabe 1 – CMOS-Gatterschaltungen

Aufgabe 2 – NAND-Technik

Aufgabe 3 – PAL-Implementierung

Aufgabe 4 – Latches und Flipflops

Aufgabe 1 – CMOS-Gatterschaltungen

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 1 – CMOS-Gatterschaltungen

Sei die Schaltfunktion $f_1(x_3, x_2, x_2, x_0) = x_0\overline{x_1\,x_2} + x_0\overline{x_1\,x_3}$ gegeben.

a) Standardzellen sind vorgefertigte CMOS-Realisierungen einfacher Schaltfunktionen, wie zum Beispiel Und, Oder oder Nicht, die im Baukastenprinzip zusammengesetzt werden können. Schalten Sie die folgenden Standardzellen so zusammen, dass sie f_1 realisieren (Hinweis: die einzelnen Standardzellen sind gestrichelt umrahmt).

Aufgabe 1 – CMOS: Begriffsklärung

Transistor (TRANSFER RESISTOR)

Ein Transistor ist ein steuerbarer Widerstand. Wir nutzen ihn als einen durch Spannung steuerbaren Schalter .

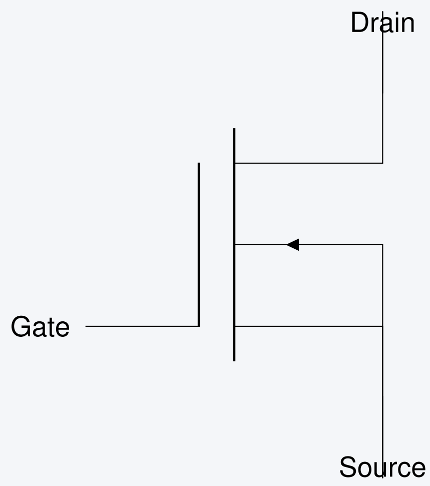
Aufgabe 1 – CMOS: Begriffsklärung

Transistor (Transfer Resistor)

Ein Transistor ist ein steuerbarer Widerstand. Wir nutzen ihn als einen durch Spannung steuerbaren Schalter .

MOSFETs (*metal-oxid-semiconductor-field-effect-transistor*)

Ein MOSFET ist ein Feldeffekttransistor mit isoliertem Gate mit – historisch – einer Metall-Isolator-Halbleiter-Struktur.
Wir unterscheiden zwischen NMOS und PMOS-Transistoren:



Aufgabe 1 – CMOS: Begriffsklärung

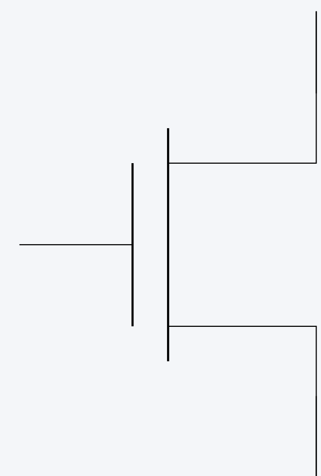
Transistor (TRANSFER RESISTOR)

Ein Transistor ist ein steuerbarer Widerstand. Wir nutzen ihn als einen durch Spannung steuerbaren Schalter .

MOSFETs (*metal-oxid-semiconductor-field-effect-transistor*)

Ein MOSFET ist ein Feldeffekttransistor mit isoliertem Gate mit – historisch – einer Metall-Isolator-Halbleiter-Struktur.
Wir unterscheiden zwischen NMOS und PMOS-Transistoren:

NMOS **n**-dotiert $\rightarrow$ öffnet bei logischer 1
Kommt im „pull-dow**n**“-Netwerk vor.



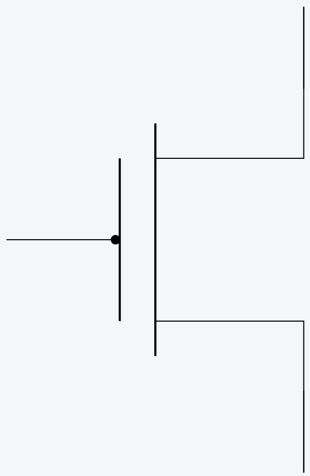
Aufgabe 1 – CMOS: Begriffsklärung

Transistor (TRANSFER RESISTOR)

Ein Transistor ist ein steuerbarer Widerstand. Wir nutzen ihn als einen durch Spannung steuerbaren Schalter .

MOSFETs (*metal-oxid-semiconductor-field-effect-transistor*)

Ein MOSFET ist ein Feldeffekttransistor mit isoliertem Gate mit – historisch – einer Metall-Isolator-Halbleiter-Struktur.
Wir unterscheiden zwischen NMOS und PMOS-Transistoren:



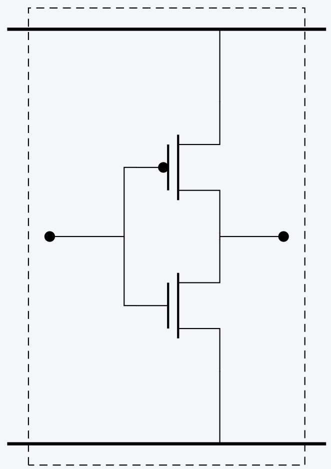
PMOS **p**-dotiert $\rightarrow$ schließt bei logischer 1
Kommt im „pull-u**p**"-Netwerk vor.

Aufgabe 1 – CMOS: Begriffsklärung

MOSFETs (*metal-oxid-semiconductor-field-effect-transistor*)

Ein MOSFET ist ein Feldeffekttransistor mit isoliertem Gate mit – historisch – einer Metall-Isolator-Halbleiter-Struktur.

Wir unterscheiden zwischen NMOS und PMOS-Transistoren:



- NMOS **n**-dotiert → öffnet bei logischer 1
  Kommt im „pull-dow**n**"-Netwerk vor.
- PMOS **p**-dotiert → schließt bei logischer 1
  Kommt im „pull-u**p**"-Netwerk vor.

CMOS (COMPLEMENTARY MOS(FET)S)

Man versteht unter der CMOS-Technologie eine Logikfamilie, sowie den dazu verwendeten Halbleiterprozess.

Der Grundgedanke dieser ist die Kombination von PMOS und NMOS-Transistoren.

Die gewünschte Operation wird dabei sowohl im PMOS-Netz (das „pull-up"-Netz) als auch im NMOS-Netz (das „pull-down"-Netz) realisiert.

Aufgabe 1 – CMOS: Ein Beispiel (I)

Realisieren Sie die Schaltfunktion $f_B(x, y) = \overline{x + y}$.
Wir gehen dazu wie folgt vor:

Aufgabe 1 – CMOS: Ein Beispiel (I)

Realisieren Sie die Schaltfunktion $f_B(x, y) = \overline{x + y}$.
Wir gehen dazu wie folgt vor:

Pull-Up-Netzwerk Hier müssen alle Literale negiert auftreten („ein PMOS ist ein negierter NMOS“). Wir formen also die Funktion wie folgt um:

Aufgabe 1 – CMOS: Ein Beispiel (I)

Realisieren Sie die Schaltfunktion $f_B(x, y) = \overline{x + y}$.
Wir gehen dazu wie folgt vor:

Pull-Up-Netzwerk Hier müssen alle Literale negiert auftreten („ein PMOS ist ein negierter NMOS“). Wir formen also die Funktion wie folgt um:

$$f_{B_{PMOS}} = \overline{x + y} = \overline{x} \cdot \overline{y}$$

Aufgabe 1 – CMOS: Ein Beispiel (I)

Realisieren Sie die Schaltfunktion $f_B(x, y) = \overline{x + y}$.
Wir gehen dazu wie folgt vor:

Pull-Up-Netzwerk Hier müssen alle Literale negiert auftreten („ein PMOS ist ein negierter NMOS"). Wir formen also die Funktion wie folgt um:

$$f_{B_{PMOS}} = \overline{x + y} = \overline{x} \cdot \overline{y}$$

Wir sehen, dass alle Literale negiert vorkommen. Wir brauchen also zwei PMOS-Transistoren, die

Aufgabe 1 – CMOS: Ein Beispiel (I)

Realisieren Sie die Schaltfunktion $f_B(x, y) = \overline{x + y}$.
Wir gehen dazu wie folgt vor:

Pull-Up-Netzwerk Hier müssen alle Literale negiert auftreten („ein PMOS ist ein negierter NMOS“). Wir formen also die Funktion wie folgt um:

$$f_{B_{PMOS}} = \overline{x + y} = \overline{x} \cdot \overline{y}$$

Wir sehen, dass alle Literale negiert vorkommen. Wir brauchen also zwei PMOS-Transistoren, die in Serie geschalten sind.

Aufgabe 1 – CMOS: Ein Beispiel (I)

Realisieren Sie die Schaltfunktion $f_B(x, y) = \overline{x + y}$.
Wir gehen dazu wie folgt vor:

Pull-Up-Netzwerk Hier müssen alle Literale negiert auftreten („ein PMOS ist ein negierter NMOS“). Wir formen also die Funktion wie folgt um:

$$f_{B_{PMOS}} = \overline{x + y} = \overline{x} \cdot \overline{y}$$

Wir sehen, dass alle Literale negiert vorkommen. Wir brauchen also zwei PMOS-Transistoren, die in Serie geschalten sind.
Die Eingänge sind dann aber x und y.

Aufgabe 1 – CMOS: Ein Beispiel (I)

Realisieren Sie die Schaltfunktion $f_B(x, y) = \overline{x + y}$.
Wir gehen dazu wie folgt vor:

Pull-Up-Netzwerk Hier müssen alle Literale negiert auftreten („ein PMOS ist ein negierter NMOS“). Wir formen also die Funktion wie folgt um:

$$f_{B_{PMOS}} = \overline{x + y} = \overline{x} \cdot \overline{y}$$

Wir sehen, dass alle Literale negiert vorkommen. Wir brauchen also zwei PMOS-Transistoren, die in Serie geschalten sind.
Die Eingänge sind dann aber x und y.

Pull-Down-Netzwerk Hier muss die Funktion negiert werden („das PDN ist zum PUN komplementär“):

Aufgabe 1 – CMOS: Ein Beispiel (I)

Realisieren Sie die Schaltfunktion $f_B(x, y) = \overline{x + y}$.
Wir gehen dazu wie folgt vor:

Pull-Up-Netzwerk Hier müssen alle Literale negiert auftreten („ein PMOS ist ein negierter NMOS"). Wir formen also die Funktion wie folgt um:

$$f_{B_{PMOS}} = \overline{x + y} = \overline{x} \cdot \overline{y}$$

Wir sehen, dass alle Literale negiert vorkommen. Wir brauchen also zwei PMOS-Transistoren, die in Serie geschalten sind.
Die Eingänge sind dann aber x und y.

Pull-Down-Netzwerk Hier muss die Funktion negiert werden („das PDN ist zum PUN komplementär"):

$$f_{B_{NMOS}}(x, y) = \overline{\overline{x + y}} = x + y$$

Aufgabe 1 – CMOS: Ein Beispiel (I)

Realisieren Sie die Schaltfunktion $f_B(x, y) = \overline{x + y}$.
Wir gehen dazu wie folgt vor:

Pull-Up-Netzwerk Hier müssen alle Literale negiert auftreten („ein PMOS ist ein negierter NMOS"). Wir formen also die Funktion wie folgt um:

$$f_{B_{PMOS}} = \overline{x + y} = \overline{x} \cdot \overline{y}$$

Wir sehen, dass alle Literale negiert vorkommen. Wir brauchen also zwei PMOS-Transistoren, die in Serie geschalten sind.
Die Eingänge sind dann aber x und y.

Pull-Down-Netzwerk Hier muss die Funktion negiert werden („das PDN ist zum PUN komplementär"):

$$f_{B_{NMOS}}(x, y) = \overline{\overline{x + y}} = x + y$$

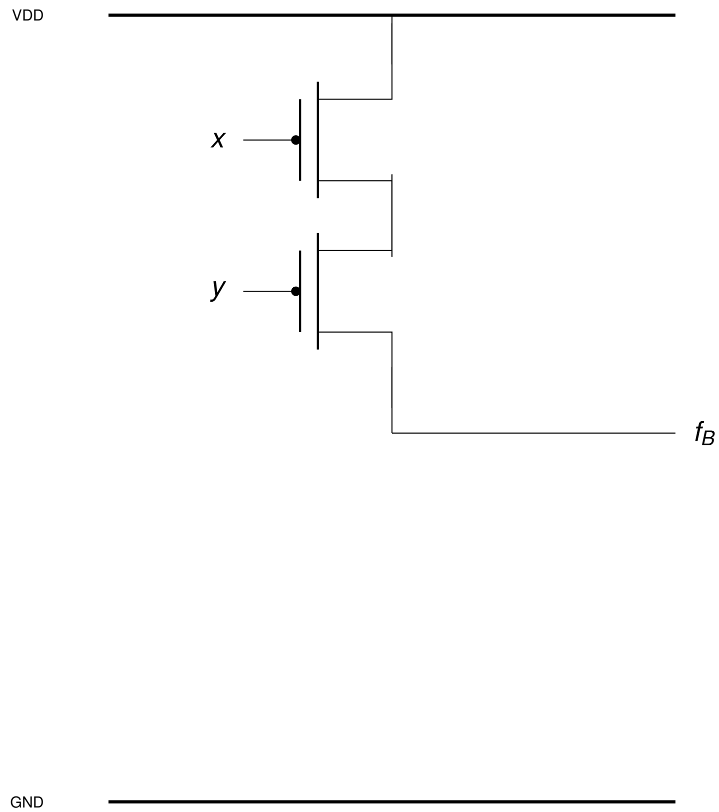
Mit diesen Funktionen lässt sich nun das CMOS-Netz bilden.

Aufgabe 1 – CMOS: Ein Beispiel (II)

Realisieren Sie die Schaltfunktion $f_B(x, y) = \overline{x + y}$.
Bauen des Pull-Up-Netzwerkes: $f_{PMOS} = \overline{x} \cdot \overline{y}$

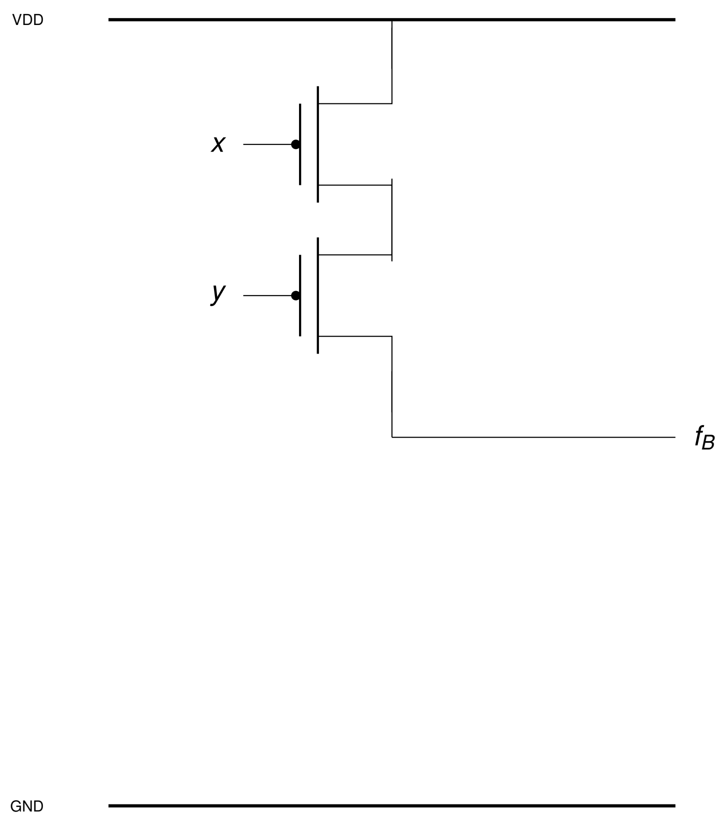
Aufgabe 1 – CMOS: Ein Beispiel (II)

Realisieren Sie die Schaltfunktion $f_B(x, y) = \overline{x + y}$.
Bauen des Pull-Up-Netzwerkes: $f_{PMOS} = \overline{x} \cdot \overline{y}$



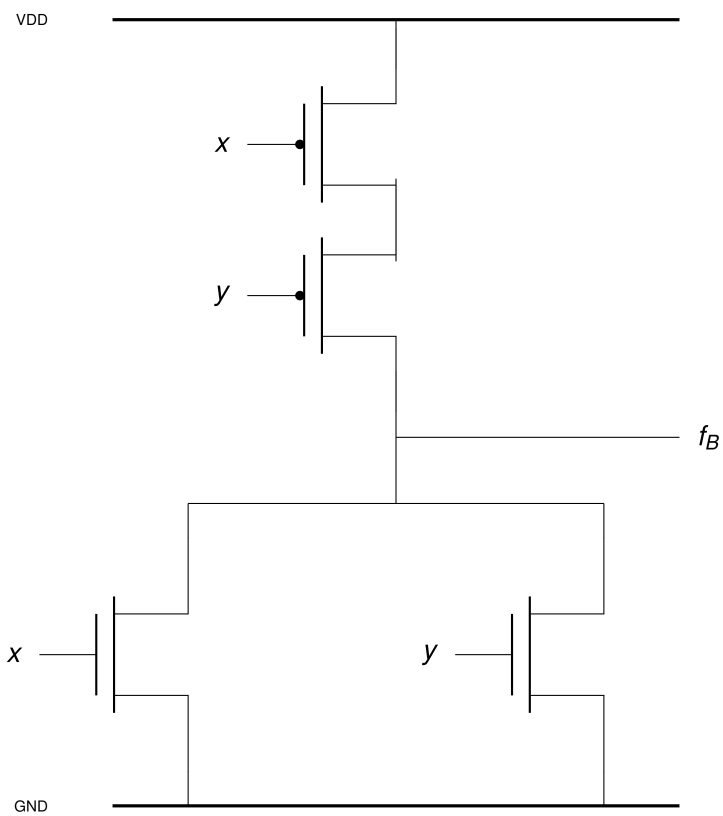
Aufgabe 1 – CMOS: Ein Beispiel (II)

Realisieren Sie die Schaltfunktion $f_B(x, y) = \overline{x + y}$.
Bauen des Pull-Down-Netzwerkes: $f_{NMOS} = x + y$



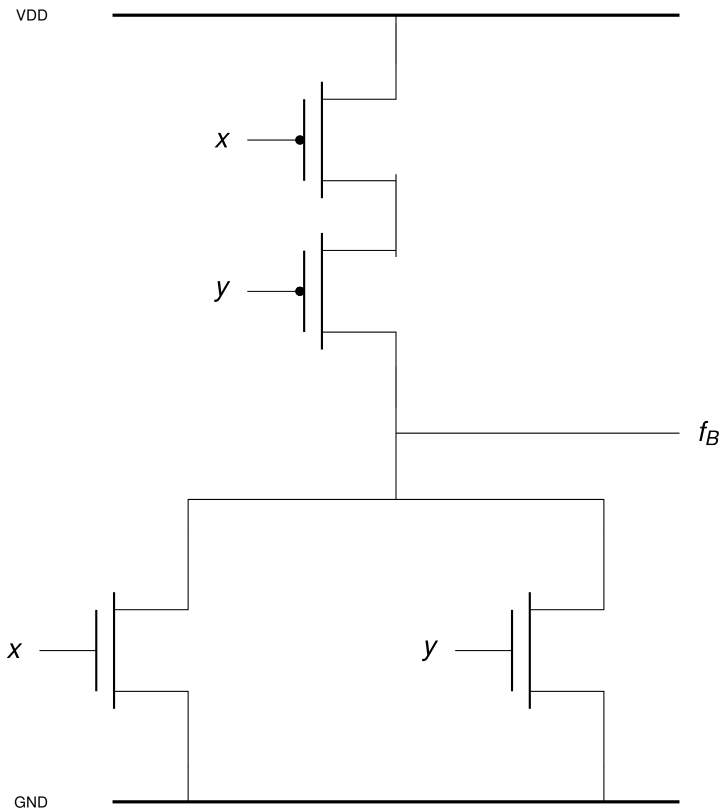
Aufgabe 1 – CMOS: Ein Beispiel (II)

Realisieren Sie die Schaltfunktion $f_B(x, y) = \overline{x + y}$.
Bauen des Pull-Down-Netzwerkes: $f_{NMOS} = x + y$



Aufgabe 1 – CMOS: Ein Beispiel (II)

Realisieren Sie die Schaltfunktion $f_B(x, y) = \overline{x + y}$.
Bauen des Pull-Down-Netzwerkes: $f_{NMOS} = x + y$



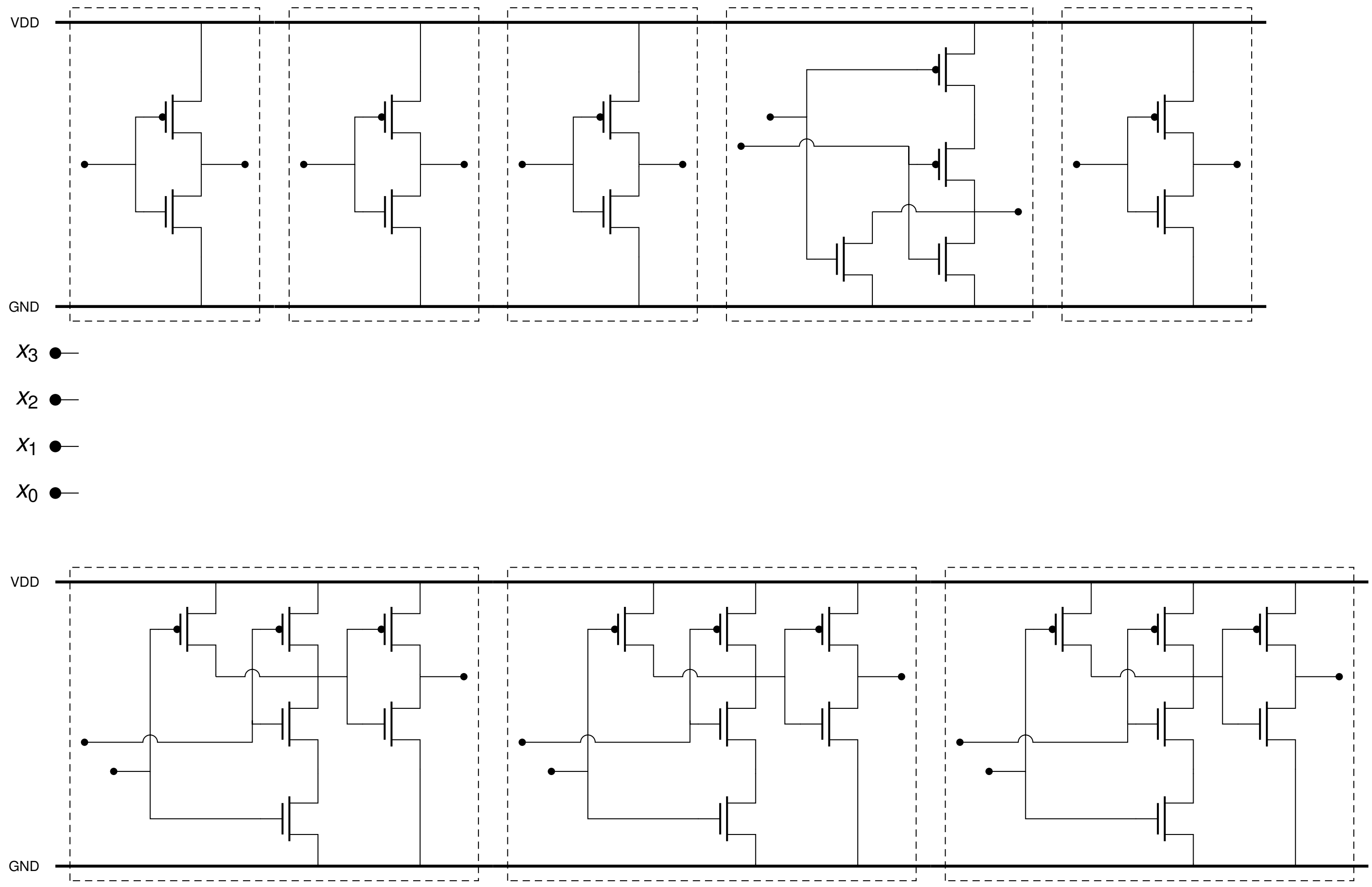
Weiteres Tafelbeispiel: $f_{B2}(x_1, x_2) = x_1 + x_2$

Aufgabe 1 – CMOS-Gatterschaltungen

Sei die Schaltfunktion $f_1(x_3, x_2, x_2, x_0) = x_0 \overline{x_1\, x_2} + x_0 \overline{x_1\, x_3}$ gegeben.

a) Standardzellen sind vorgefertigte CMOS-Realisierungen einfacher Schaltfunktionen, wie zum Beispiel Und, Oder oder Nicht, die im Baukastenprinzip zusammengesetzt werden können. Schalten Sie die folgenden Standardzellen so zusammen, dass sie f_1 realisieren (Hinweis: die einzelnen Standardzellen sind gestrichelt umrahmt).

Aufgabe 1 – CMOS-Gatterschaltungen



Aufgabe 1 – CMOS-Gatterschaltungen

Sei die Schaltfunktion $f_1(x_3, x_2, x_2, x_0) = x_0 \overline{x_1\, x_2} + x_0 \overline{x_1\, x_3}$ gegeben.

b) Realisieren Sie die Schaltfunktion f_1 als CMOS-Schaltung mit möglichst wenig Transistoren, wobei alle Eingänge nur in der nicht invertierten Form zur Verfügung stehen.

Aufgabe 1 – CMOS-Gatterschaltungen

Sei die Schaltfunktion $f_1(x_3, x_2, x_2, x_0) = x_0\overline{x_1\,x_2} + x_0\overline{x_1\,x_3}$ gegeben.

c) Vergleichen Sie die Anzahl benötigter Transistoren in a) und b). Für welche Anwendungsfälle eignen sich die beiden Entwurfsmethoden jeweils?

Aufgabe 2 – NAND-Technik

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Aufgabe 2 – NAND-Technik

Realisieren Sie die Schaltfunktion

$$f_2 = \overline{D}CA + \overline{C}\,\overline{A} + CB + \overline{D}B$$

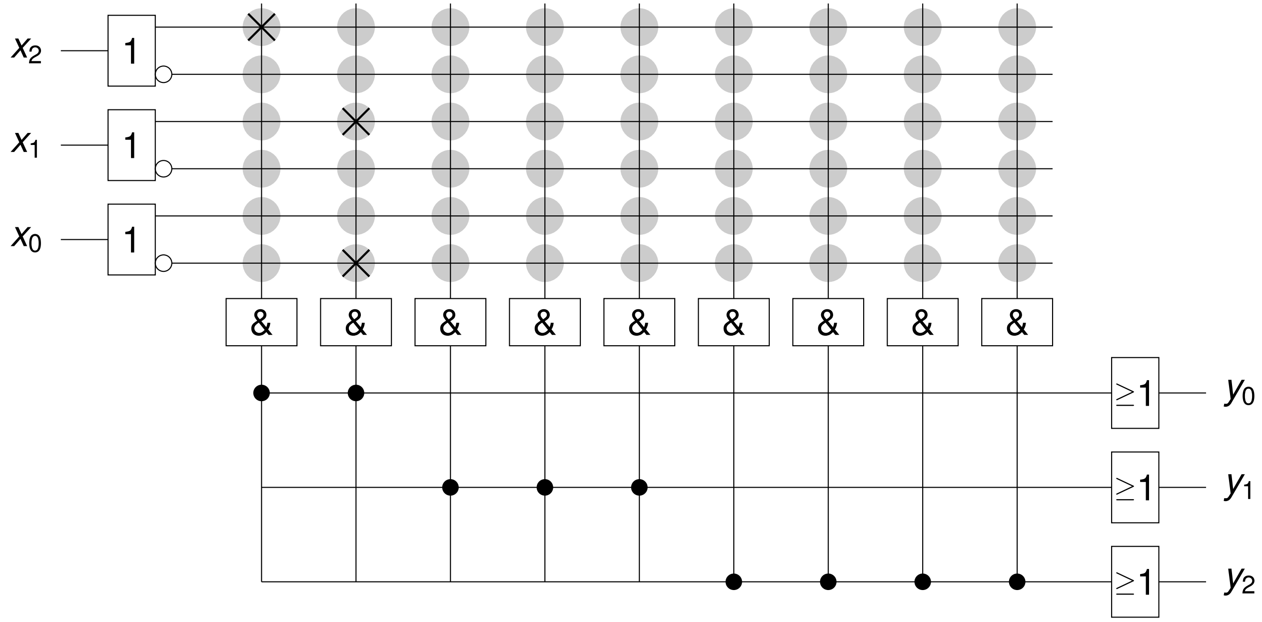
unter ausschließlicher Verwendung von NAND-Gattern mit zwei Eingängen. Wie viele NAND-Gatter sind dann erforderlich?

Aufgabe 3 – PAL-Implementierung

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 3 – PAL-Implementierung

Realisieren Sie einen Codeumsetzer, der eine 3-Bit-Binärzahl in einen zyklischen Gray-Code umwandelt. Orientieren Sie sich dazu an unten gegebener *Programmable Array Logic* (PAL), in der beispielhaft die Funktion $y_0(x_2, x_1, x_0) = x_2 + (x_1 \cdot \overline{x_0})$ programmiert ist:



Aufgabe 3 – PAL-Implementierung: Begriffsklärung

PAL (Programmable Array Logic)

Effektiv ein Baustein, in dem konjunktive Terme über ein Feld frei programmiert werden können. Damit ist ein PAL ein Baustein zur Repräsentation einer DMF:

Erste Stufe Auswahl der Literale für konjunktive Terme (programmierbar)

Zweite Stufe Auswahl der konjungierten Terme (fest)

Aufgabe 3 – PAL-Implementierung: Begriffsklärung

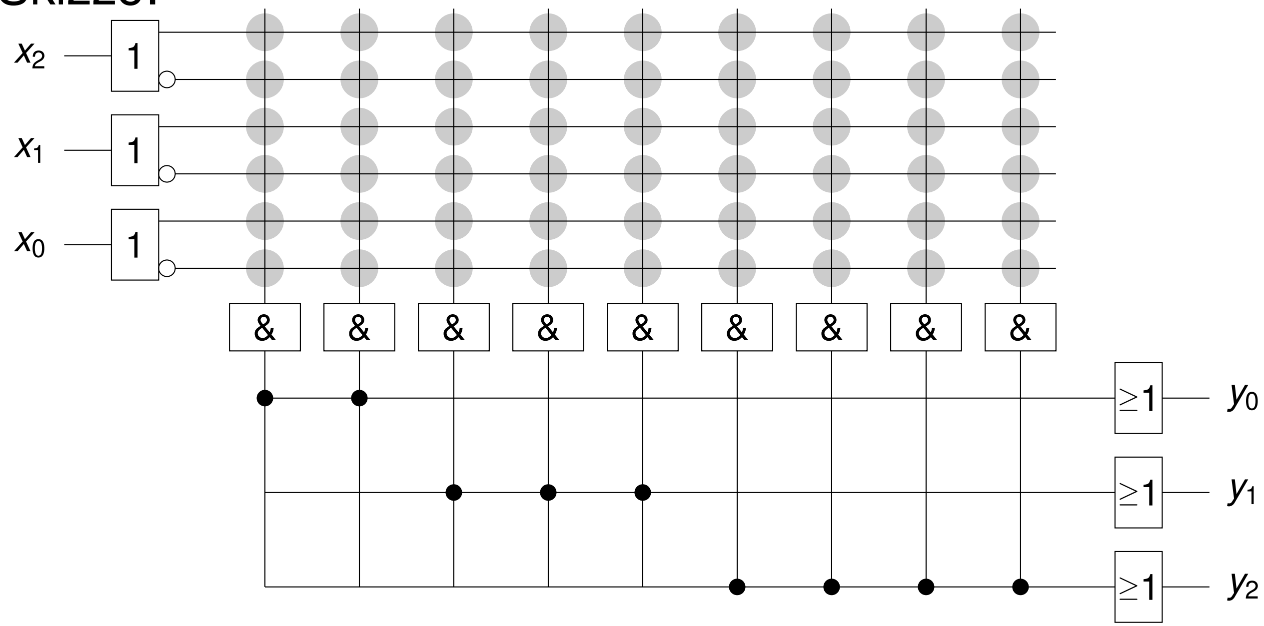
PAL (Programmable Array Logic)

Effektiv ein Baustein, in dem konjunktive Terme über ein Feld frei programmiert werden können. Damit ist ein PAL ein Baustein zur Repräsentation einer DMF:

Erste Stufe Auswahl der Literale für konjunktive Terme (programmierbar)

Zweite Stufe Auswahl der konjungierten Terme (fest)

Skizze:



Aufgabe 3 – PAL-Implementierung: Begriffsklärung

PAL (Programmable Array Logic)

Effektiv ein Baustein, in dem konjunktive Terme über ein Feld frei programmiert werden können. Damit ist ein PAL ein Baustein zur Repräsentation einer DMF:

Erste Stufe Auswahl der Literale für konjunktive Terme (programmierbar)

Zweite Stufe Auswahl der konjungierten Terme (fest)

Es gibt auch noch andere Schaltungstypen, wie zum Beispiel:

PAL Programmable Array Logic – UND-Terme sind programmierbar, ODER-Terme fest.

Aufgabe 3 – PAL-Implementierung: Begriffsklärung

PAL (Programmable Array Logic)

Effektiv ein Baustein, in dem konjunktive Terme über ein Feld frei programmiert werden können. Damit ist ein PAL ein Baustein zur Repräsentation einer DMF:

Erste Stufe Auswahl der Literale für konjunktive Terme (programmierbar)

Zweite Stufe Auswahl der konjungierten Terme (fest)

Es gibt auch noch andere Schaltungstypen, wie zum Beispiel:

PLA Programmable Logic Array – UND-Terme sind programmierbar, ODER-Terme ebenfalls.

Aufgabe 3 – PAL-Implementierung: Begriffsklärung

PAL (Programmable Array Logic)

Effektiv ein Baustein, in dem konjunktive Terme über ein Feld frei programmiert werden können. Damit ist ein PAL ein Baustein zur Repräsentation einer DMF:

Erste Stufe Auswahl der Literale für konjunktive Terme (programmierbar)

Zweite Stufe Auswahl der konjungierten Terme (fest)

Es gibt auch noch andere Schaltungstypen, wie zum Beispiel:

ULA Universal Logic Array – UND-Terme sind fest, ODER-Terme ebenfalls, man kann aber programmieren, welche Minterme ausgewählt werden.

Aufgabe 3 – PAL-Implementierung: Begriffsklärung

PAL (Programmable Array Logic)

Effektiv ein Baustein, in dem konjunktive Terme über ein Feld frei programmiert werden können. Damit ist ein PAL ein Baustein zur Repräsentation einer DMF:

Erste Stufe Auswahl der Literale für konjunktive Terme (programmierbar)

Zweite Stufe Auswahl der konjungierten Terme (fest)

Es gibt auch noch andere Schaltungstypen, wie zum Beispiel:

ROM Read-Only Memory – UND-Terme sind fest, ODER-Terme dafür programmierbar.

Aufgabe 3 – PAL-Implementierung: Begriffsklärung

PAL (PROGRAMMABLE ARRAY LOGIC)

Effektiv ein Baustein, in dem konjunktive Terme über ein Feld frei programmiert werden können. Damit ist ein PAL ein Baustein zur Repräsentation einer DMF:

Erste Stufe Auswahl der Literale für konjunktive Terme (programmierbar)

Zweite Stufe Auswahl der konjungierten Terme (fest)

Es gibt auch noch andere Schaltungstypen, wie zum Beispiel:

PAL PROGRAMMABLE ARRAY LOGIC – UND-Terme sind programmierbar, ODER-Terme fest.

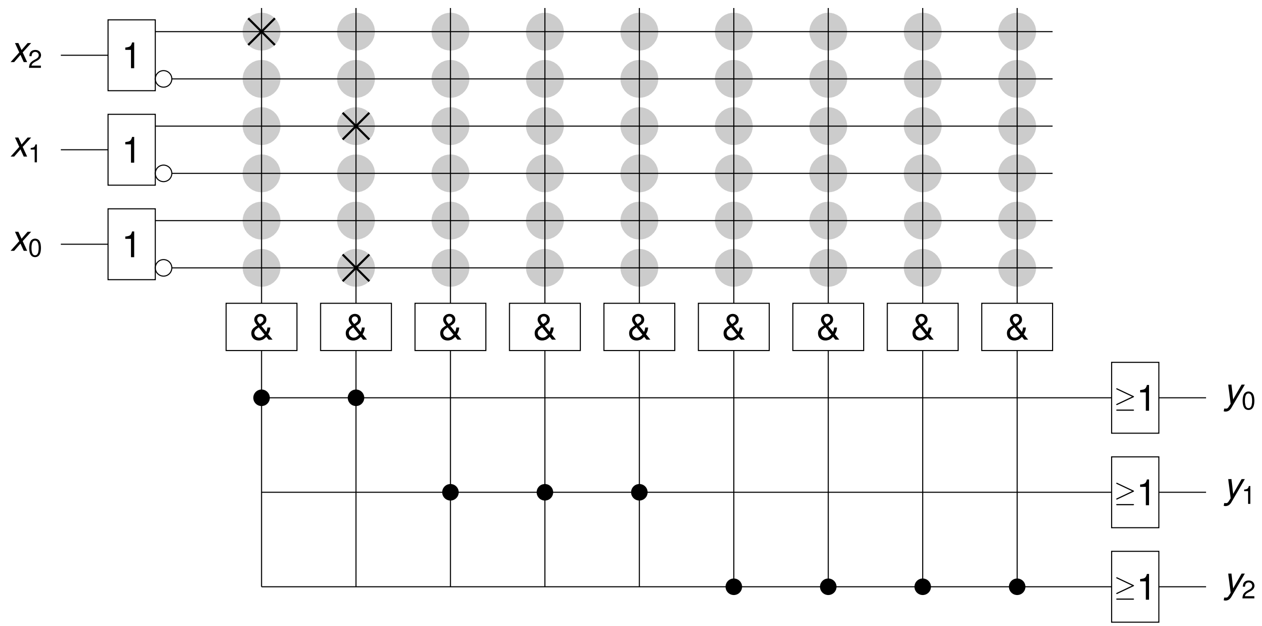
PLA PROGRAMMABLE LOGIC ARRAY – UND-Terme sind programmierbar, ODER-Terme ebenfalls.

ULA UNIVERSAL LOGIC ARRAY – UND-Terme sind fest, ODER-Terme ebenfalls, man kann aber programmieren, welche Minterme ausgewählt werden.

ROM READ-ONLY MEMORY – UND-Terme sind fest, ODER-Terme dafür programmierbar.

Aufgabe 3 – PAL-Implementierung

Realisieren Sie einen Codeumsetzer, der eine 3-Bit-Binärzahl in einen zyklischen Gray-Code umwandelt. Orientieren Sie sich dazu an unten gegebener *Programmable Array Logic* (PAL), in der beispielhaft die Funktion $y_0(x_2, x_1, x_0) = x_2 + (x_1 \cdot \overline{x_0})$ programmiert ist:



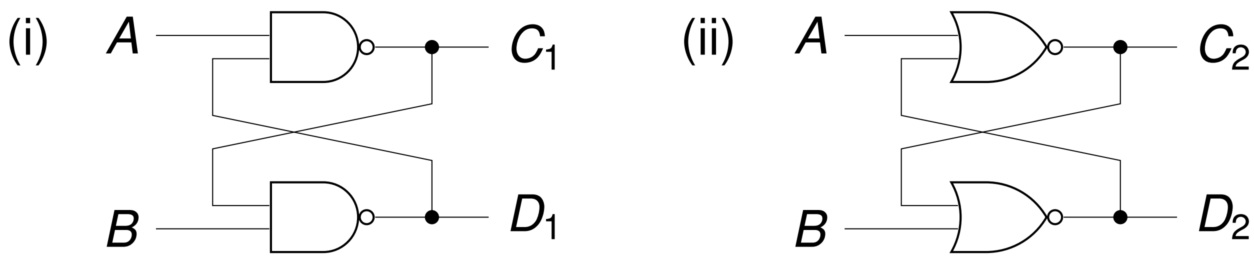
Aufgabe 4 – Latches und Flipflops

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 4 – Latches und Flipflops

Die Verzögerungszeit jedes Logikgatters in dieser Aufgabe betrage τ = 1 ns.

a) An die Eingänge (A, B) der unten stehenden Schaltungen (i) und (ii) werden nacheinander folgende Werte angelegt: $(0, 1)$, $(0, 0)$, $(1, 1)$, $(1, 0)$, $(1, 1)$ und $(0, 0)$. Geben Sie jeweils die Ausgangswerte von (i) und (ii) an und benennen Sie die Signale A bis D_2 sinnvoll.



Aufgabe 4 – Latches und Flipflops: Begriffklärung

Speicherelement

Ein Speicherelement ist ein Gerät/Modul/Element, das einen vorher angelegten Wert speichert.

Aufgabe 4 – Latches und Flipflops: Begriffklärung

Speicherelement

Ein Speicherelement ist ein Gerät/Modul/Element, das einen vorher angelegten Wert speichert.

Eigenschaften von Speicherelementen

Speicherelemente können ...

Operation	Q^t	Q^{t+1}
S(etze)	—	1
R(ücksetze)	—	0
N(ix)	0	0
	1	1

Operationen eines Speicherelements

Aufgabe 4 – Latches und Flipflops: Begriffklärung

Speicherelement

Ein Speicherelement ist ein Gerät/Modul/Element, das einen vorher angelegten Wert speichert.

Eigenschaften von Speicherelementen

Speicherelemente können ...

asynchron sein, wenn Änderungen jederzeit möglich sind.

Operation	Q^t	Q^{t+1}
S(etze)	—	1
R(ücksetze)	—	0
N(ix)	0	0
	1	1

Operationen eines Speicherelements

Aufgabe 4 – Latches und Flipflops: Begriffklärung

Speicherelement

Ein Speicherelement ist ein Gerät/Modul/Element, das einen vorher angelegten Wert speichert.

Eigenschaften von Speicherelementen

Speicherelemente können ...

asynchron sein, wenn Änderungen jederzeit möglich sind.

synchron sein, wenn Änderungen nur zu vorher festgelegten Momenten möglich sind. Diese Momente können ...

Operation	Q^t	Q^{t+1}
S(etze)	—	1
R(ücksetze)	—	0
N(ix)	0	0
	1	1

Operationen eines Speicherelements

Aufgabe 4 – Latches und Flipflops: Begriffklärung

Speicherelement

Ein Speicherelement ist ein Gerät/Modul/Element, das einen vorher angelegten Wert speichert.

Eigenschaften von Speicherelementen

Speicherelemente können ...

asynchron sein, wenn Änderungen jederzeit möglich sind.

synchron sein, wenn Änderungen nur zu vorher festgelegten Momenten möglich sind. Diese Momente können ...

pegelabhängig sein. Mann nennt diese Elemente dann auch pegelgesteuert.

Operation	Q^t	Q^{t+1}
S(etze)	—	1
R(ücksetze)	—	0
N(ix)	0	0
	1	1

Operationen eines Speicherelements

Aufgabe 4 – Latches und Flipflops: Begriffklärung

Speicherelement

Ein Speicherelement ist ein Gerät/Modul/Element, das einen vorher angelegten Wert speichert.

Eigenschaften von Speicherelementen

Speicherelemente können ...

asynchron sein, wenn Änderungen jederzeit möglich sind.

synchron sein, wenn Änderungen nur zu vorher festgelegten Momenten möglich sind. Diese Momente können ...

- pegelabhängig sein. Mann nennt diese Elemente dann auch pegelgesteuert.
- taktflankenabhängig sein. Mann nennt diese Elemente dann auch taktflankengesteuert. Es können zu **steigender**, **fallender** oder **zu beiden** Flanken Änderungen möglich sein.

Operation	Q^t	Q^{t+1}
S(etze)	—	1
R(ücksetze)	—	0
N(ix)	0	0
	1	1

Operationen eines Speicherelements

Aufgabe 4 – Latches und Flipflops: Begriffklärung

Speicherelement

Ein Speicherelement ist ein Gerät/Modul/Element, das einen vorher angelegten Wert speichert.

Flipflops

Flipflops sind Elemente zur Speicherung eines Bits.
Sie sind – bei uns – **rein taktflankengesteuert**.

Aufgabe 4 – Latches und Flipflops: Begriffklärung

Speicherelement

Ein Speicherelement ist ein Gerät/Modul/Element, das einen vorher angelegten Wert speichert.

Latches

Latches sind ebenfalls Elemente zur Speicherung eines Bits, ähnlich zu den Flipflops.
Im Gegensatz zu Flipflops sind sie aber **rein pegelgesteuert.**.

Aufgabe 4 – Latches und Flipflops: Begriffklärung

Speicherelement

Ein Speicherelement ist ein Gerät/Modul/Element, das einen vorher angelegten Wert speichert.

Flipflops

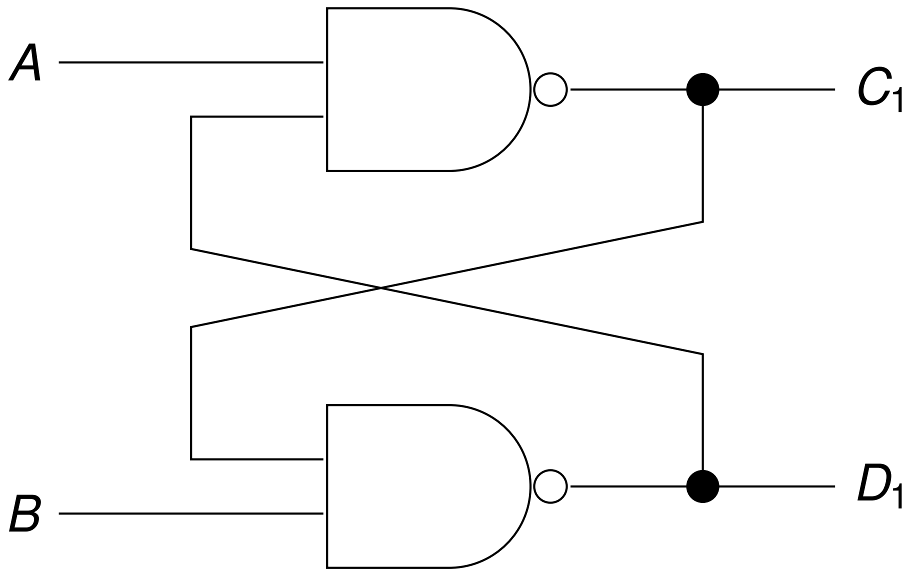
Flipflops sind Elemente zur Speicherung eines Bits.
Sie sind – bei uns – **rein taktflanken-gesteuert**.

Latches

Latches sind ebenfalls Elemente zur Speicherung eines Bits, ähnlich zu den Flipflops.
Im Gegensatz zu Flipflops sind sie aber **rein pegelgesteuert.**.

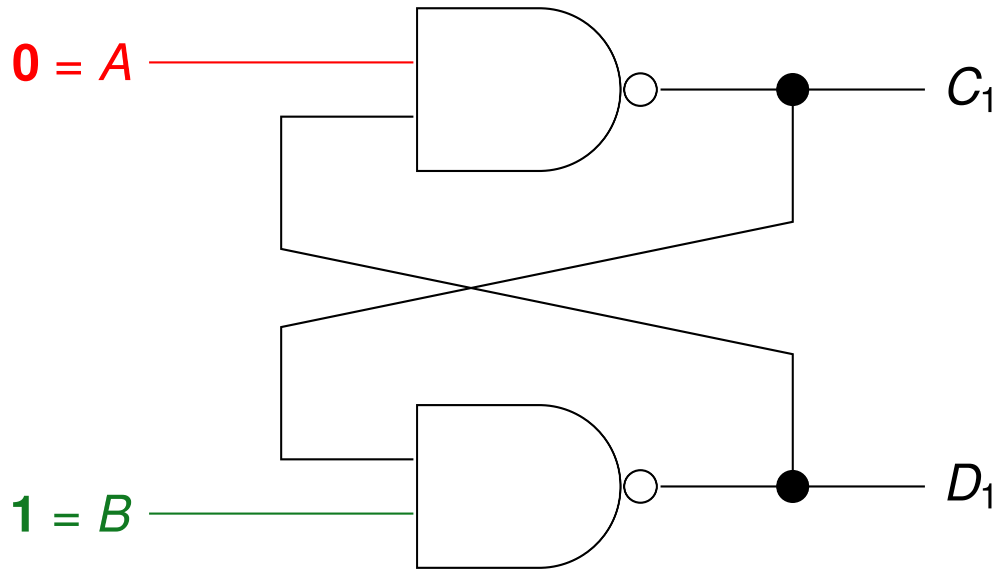
Aufgabe 4 – Latches und Flipflops

Anliegender Wert (**fett** hervorgehoben):



Aufgabe 4 – Latches und Flipflops

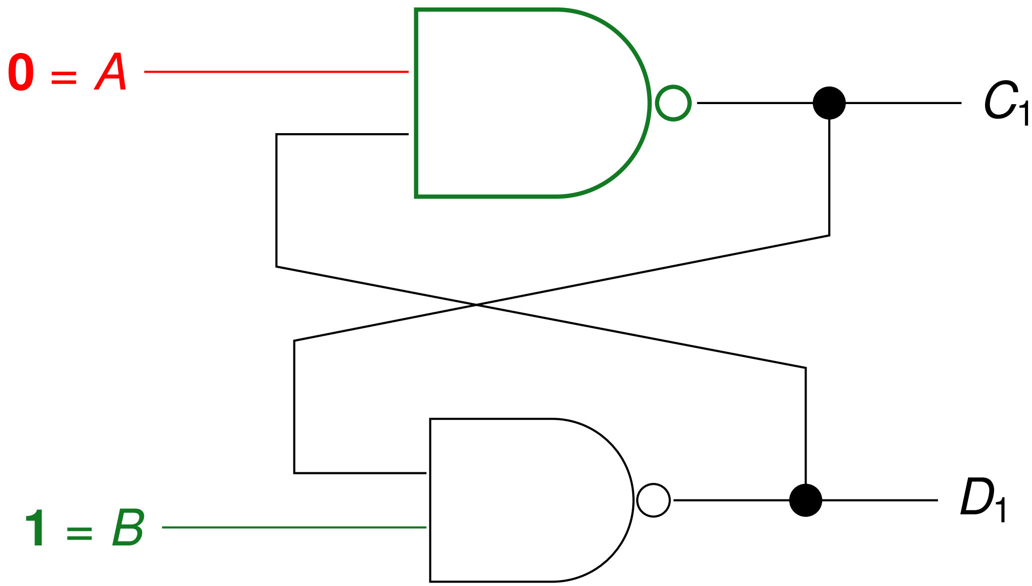
Anliegender Wert (**fett** hervorgehoben): **(0,1)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

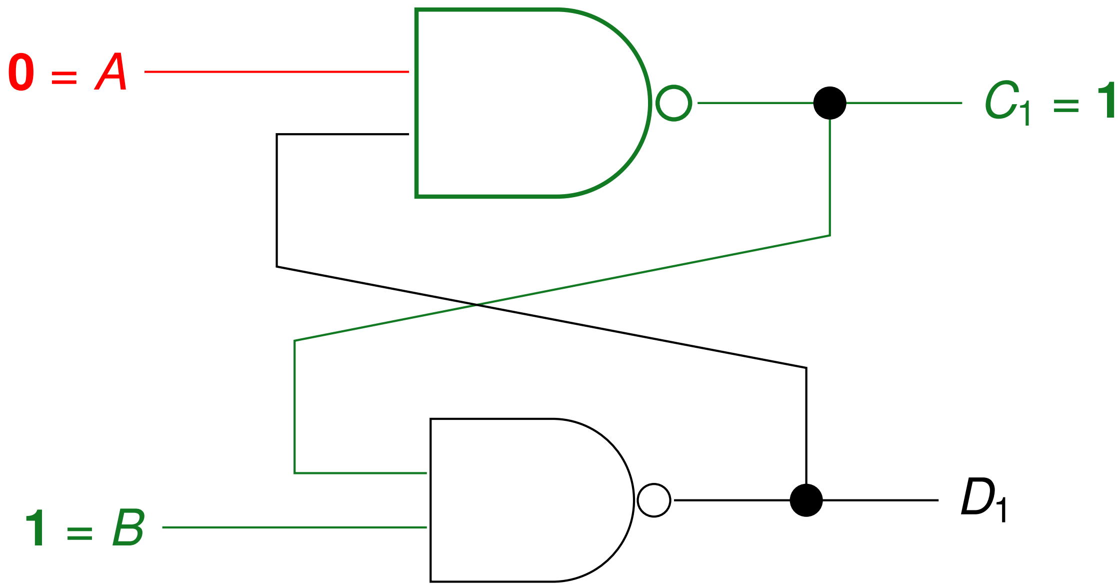
Anliegender Wert (**fett** hervorgehoben): **(0,1)**



Hat ein NAND mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein NAND **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

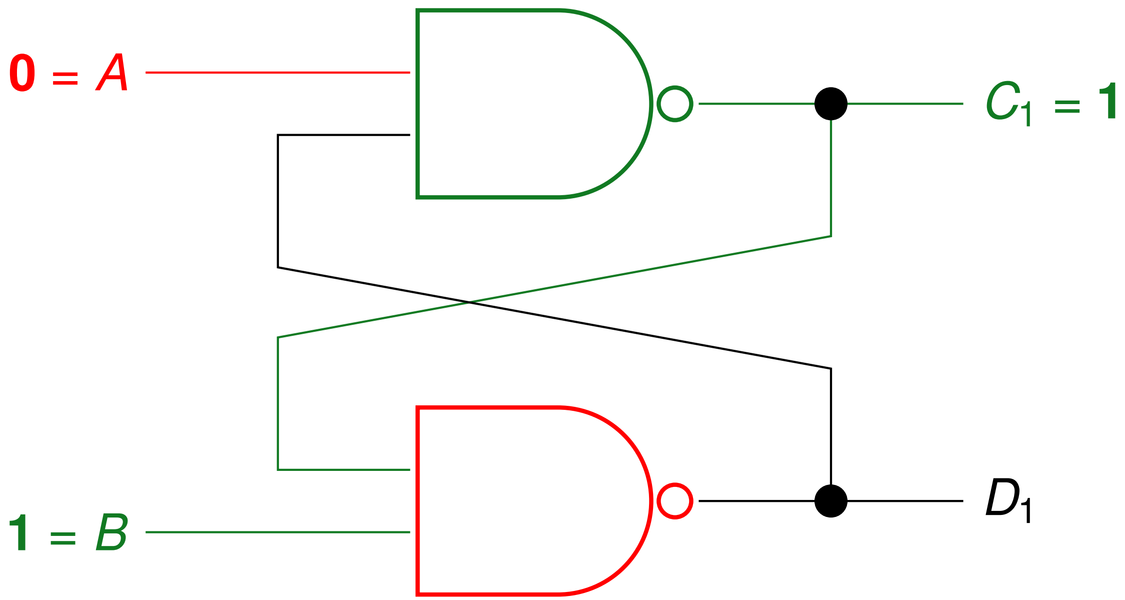
Anliegender Wert (**fett** hervorgehoben): **(0,1)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

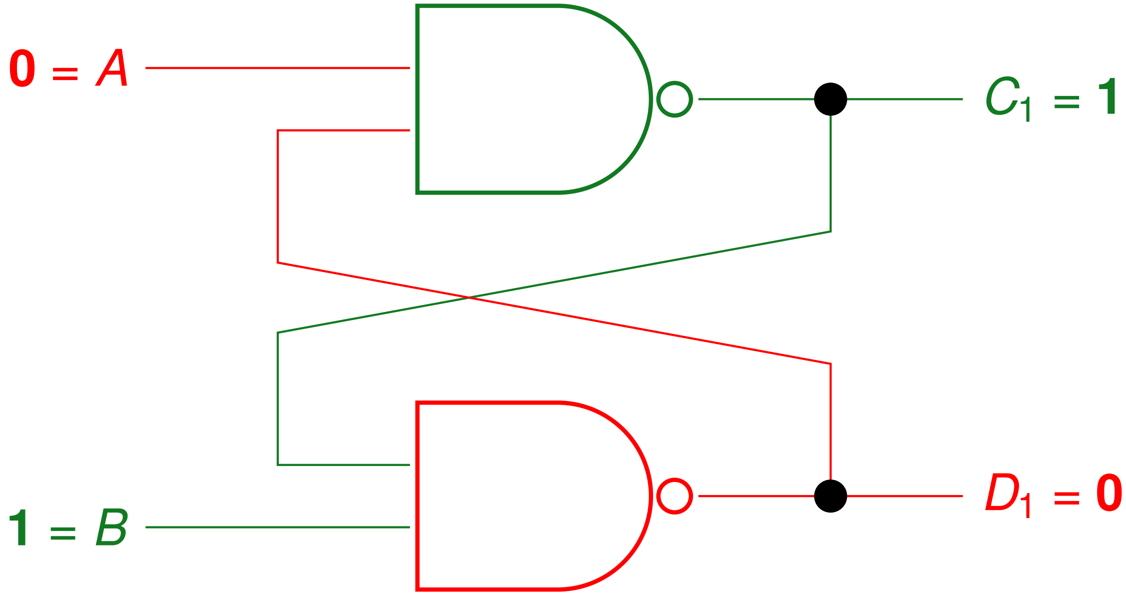
Anliegender Wert (**fett** hervorgehoben): **(0,1)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

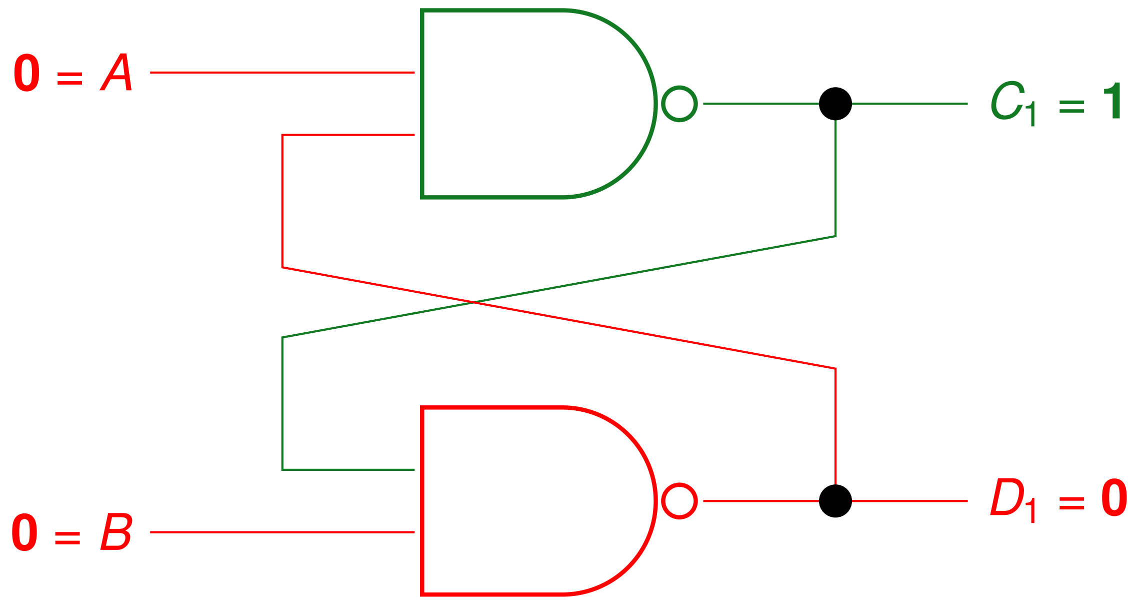
Anliegender Wert (**fett** hervorgehoben): **(0,1)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

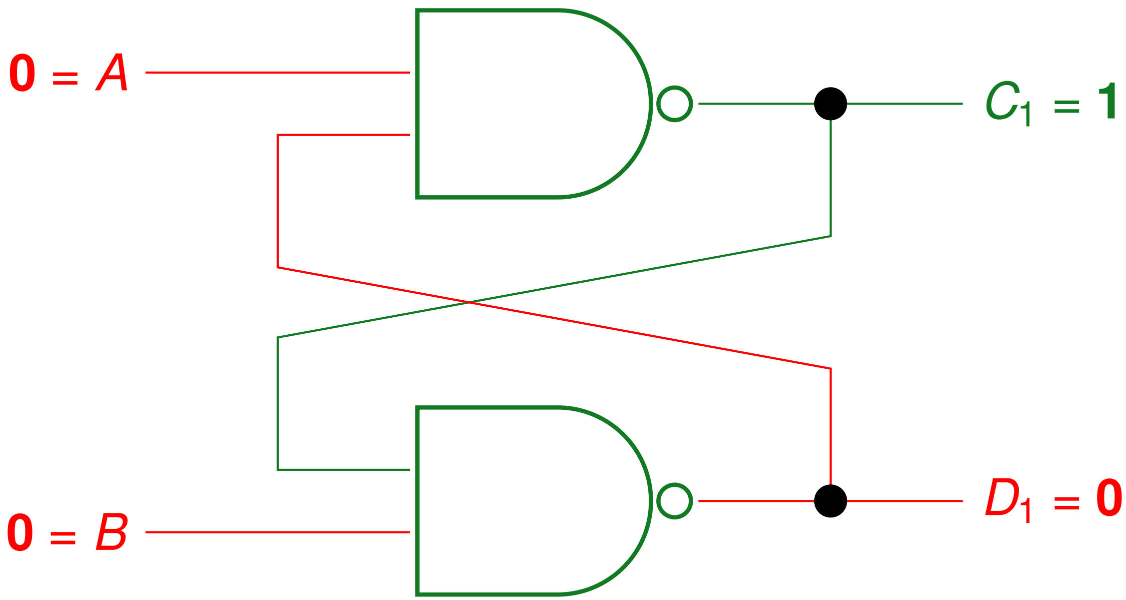
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ **(0, 0)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

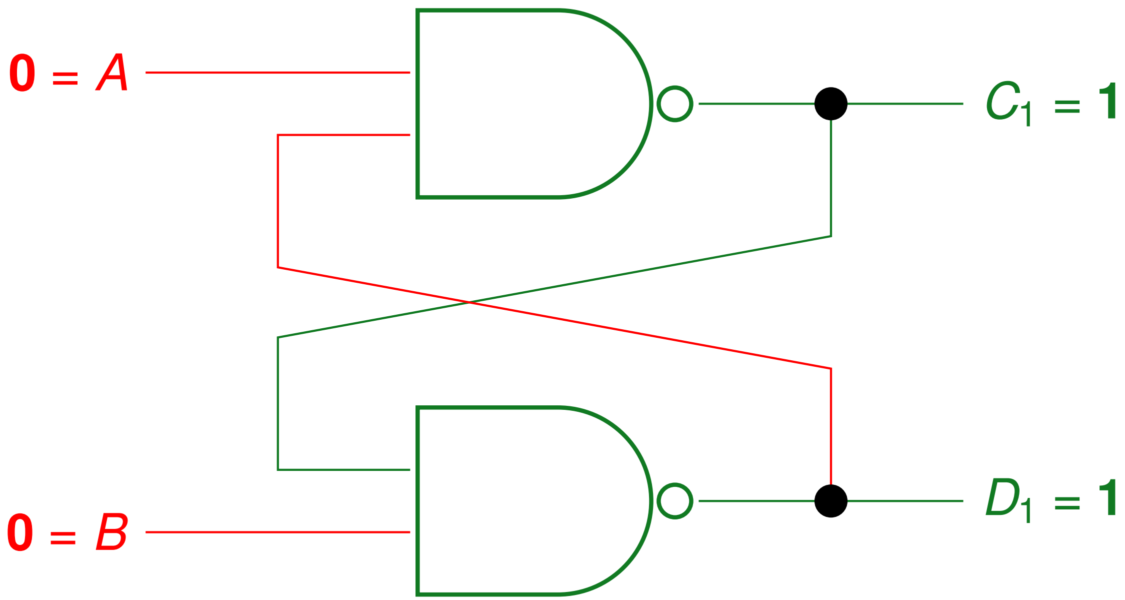
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ **(0, 0)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

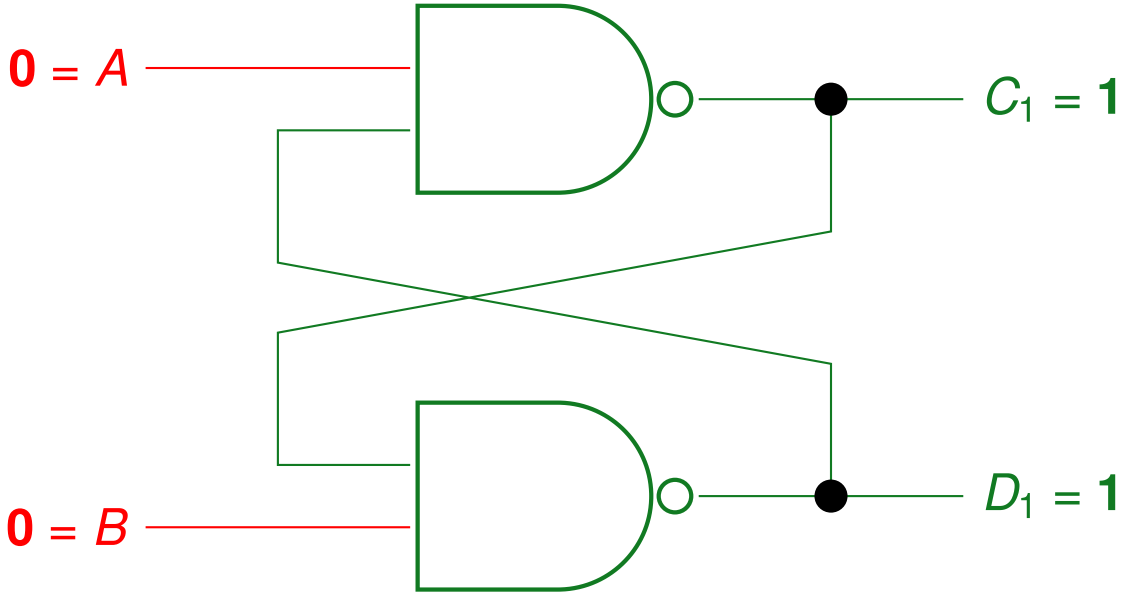
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ **(0, 0)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

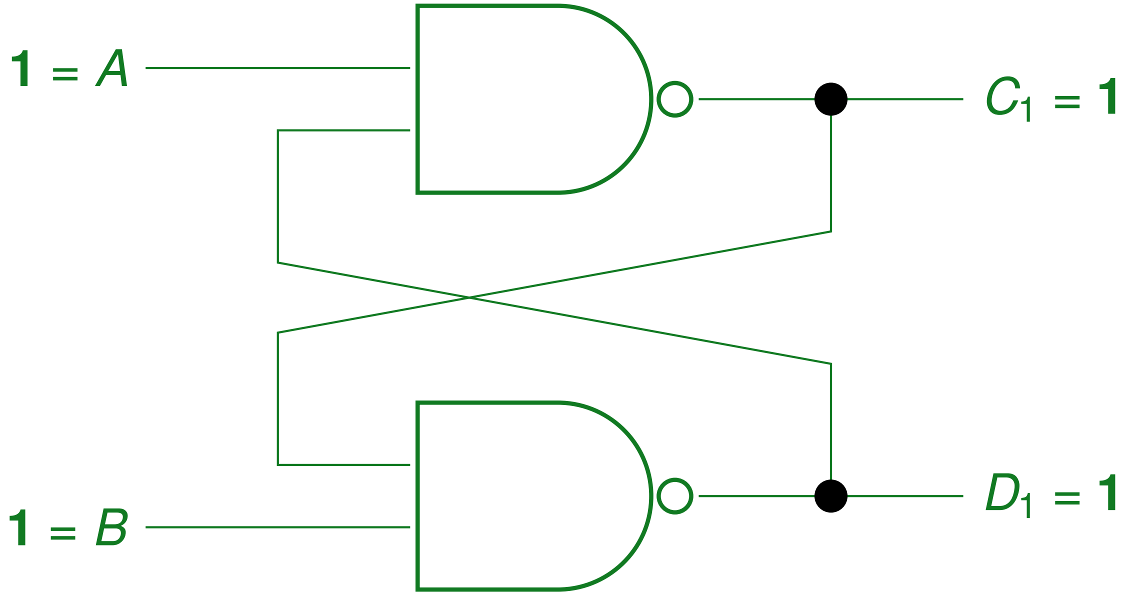
Anliegender Wert (**fett** hervorgehoben): $(0,1) \mapsto$ **(0, 0)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

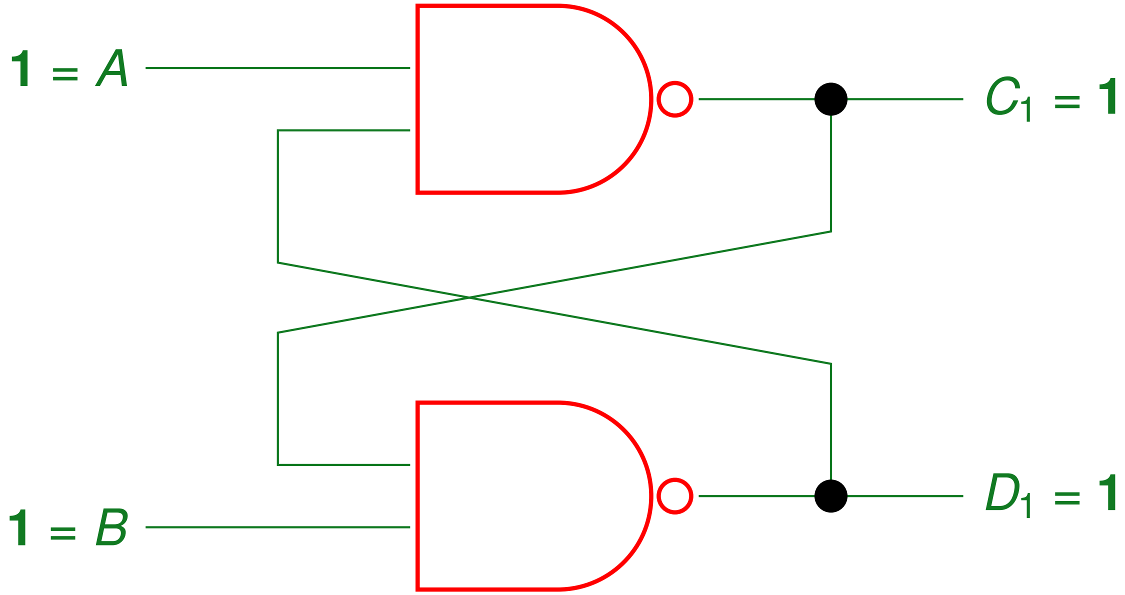
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ (0, 0) $\mapsto$ **(1, 1)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

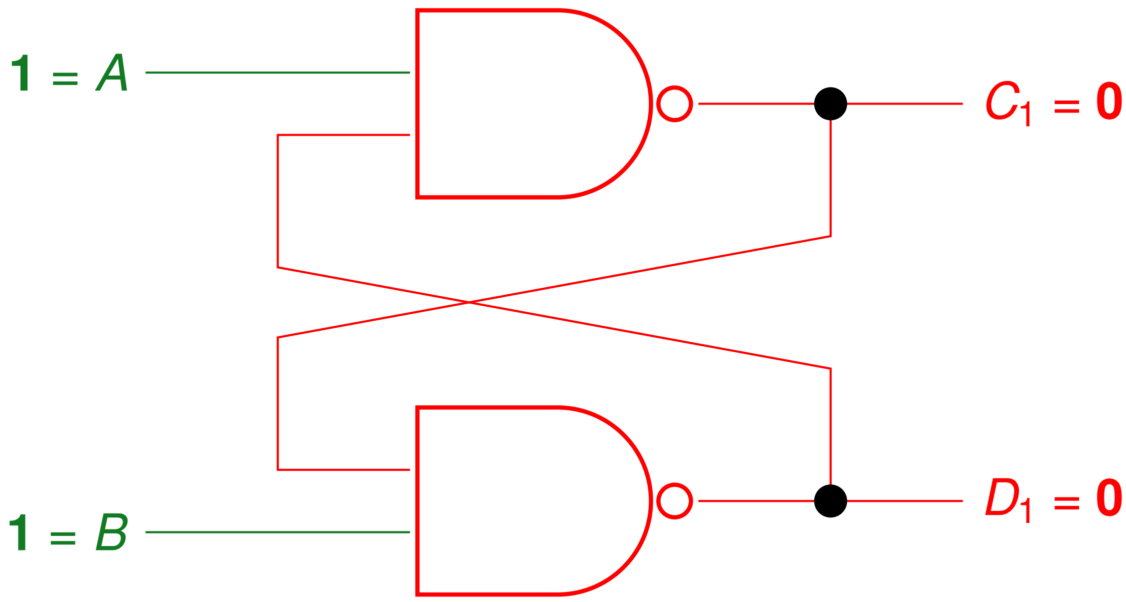
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ (0, 0) $\mapsto$ **(1, 1)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

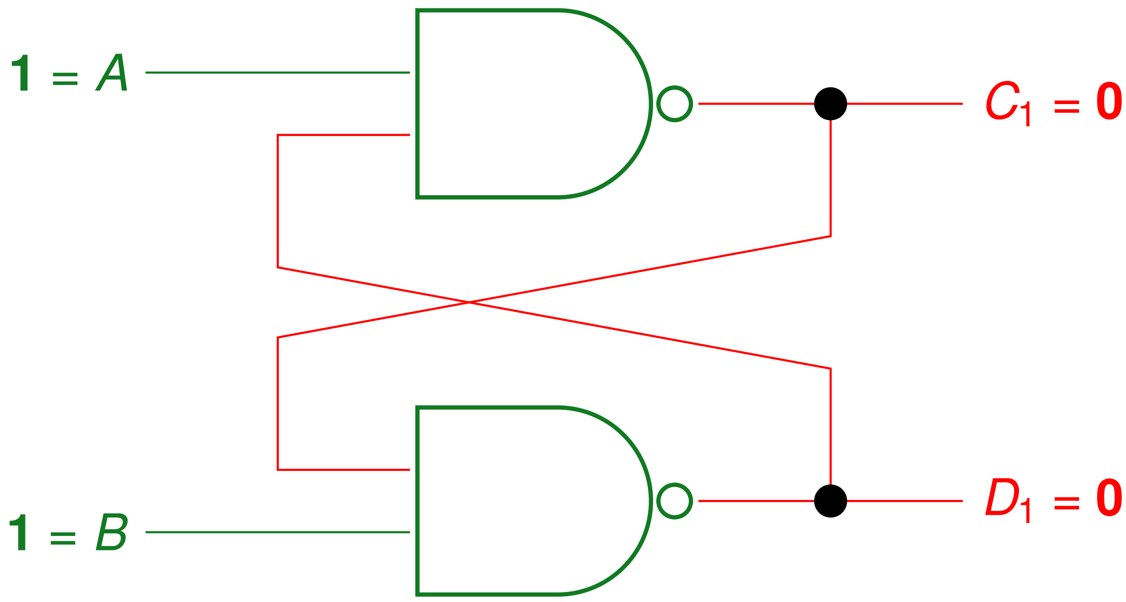
Anliegender Wert (**fett** hervorgehoben): $(0,1) \mapsto (0,0) \mapsto$ **(1, 1)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

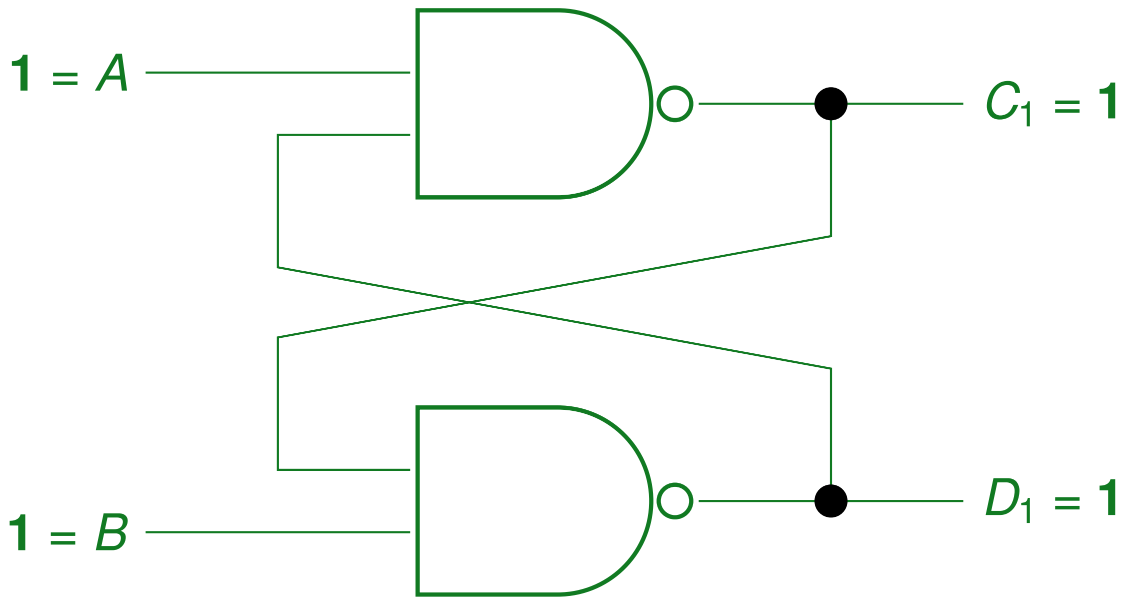
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ (0, 0) $\mapsto$ **(1, 1)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

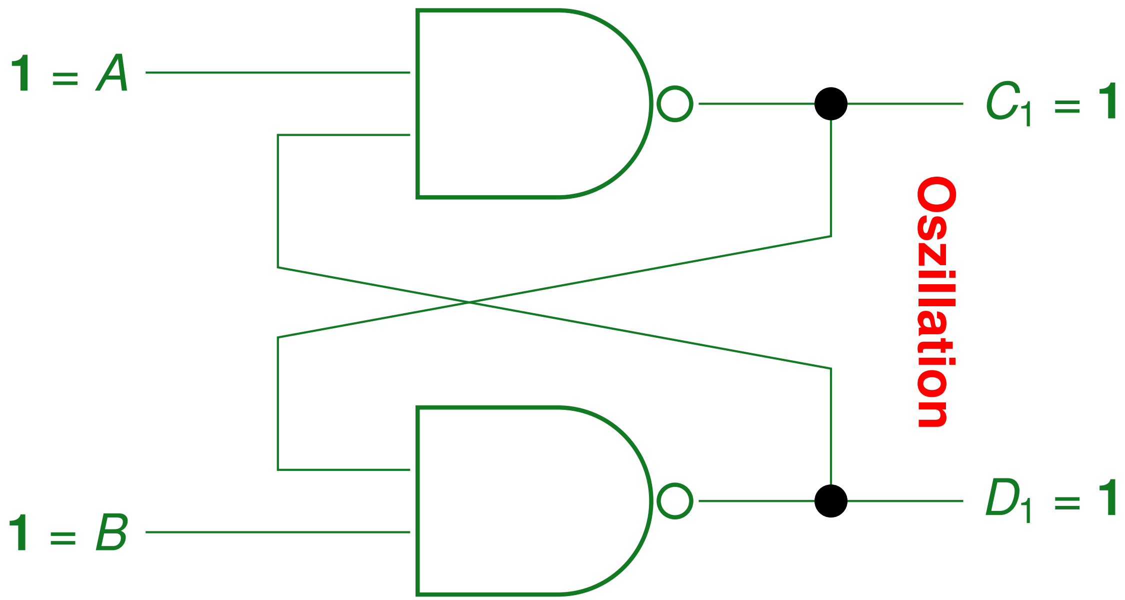
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ (0, 0) $\mapsto$ **(1, 1)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

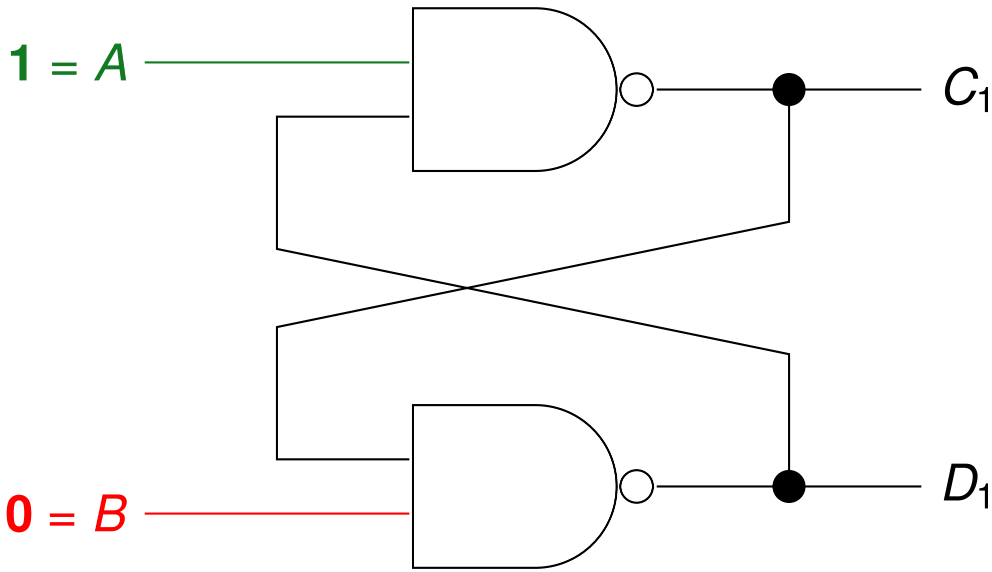
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ (0, 0) $\mapsto$ **(1, 1)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

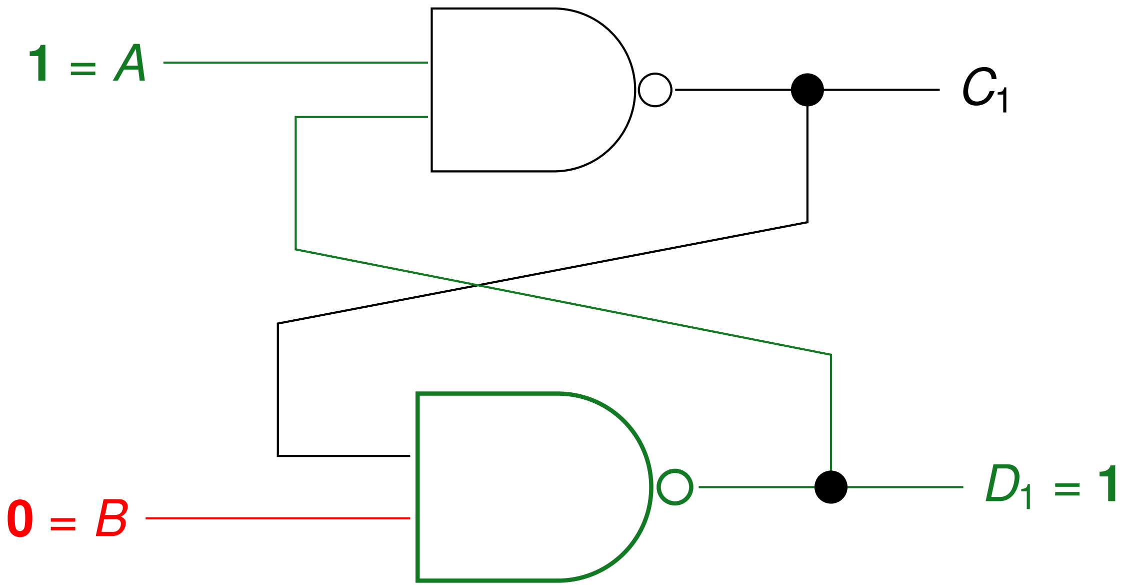
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (0, 0) $\mapsto$ (1, 1) $\mapsto$ **(1, 0)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

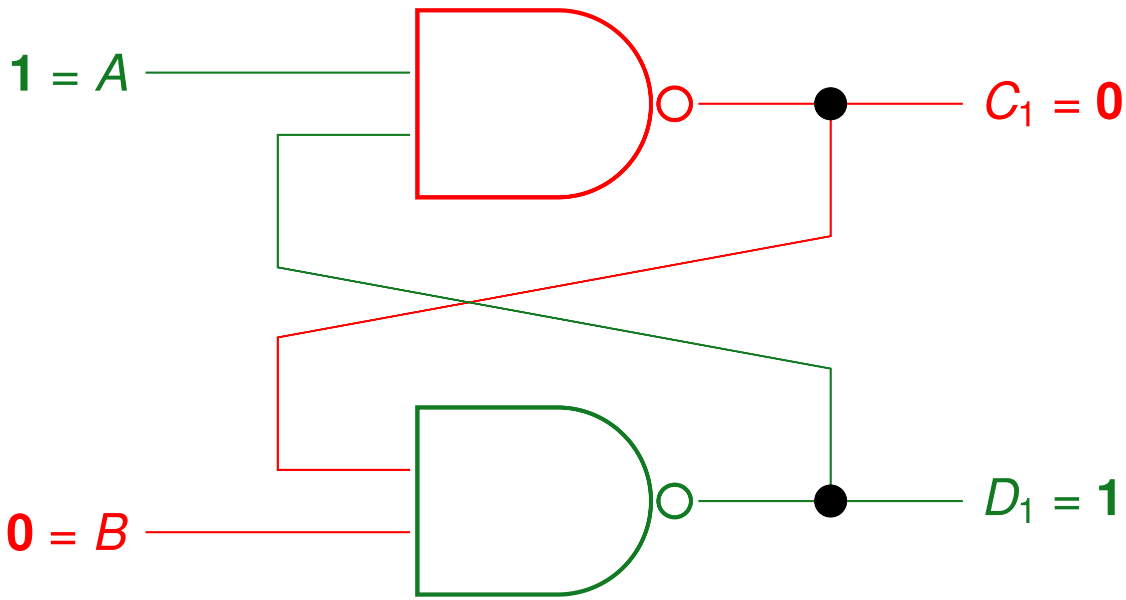
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (0, 0) $\mapsto$ (1, 1) $\mapsto$ **(1, 0)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

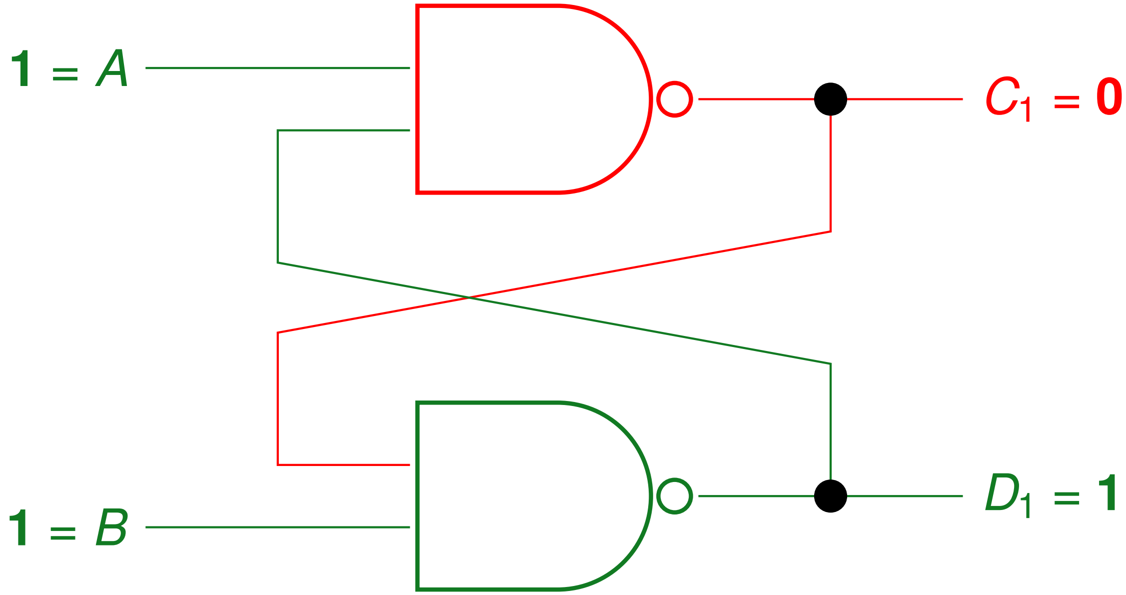
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (0, 0) $\mapsto$ (1, 1) $\mapsto$ **(1, 0)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

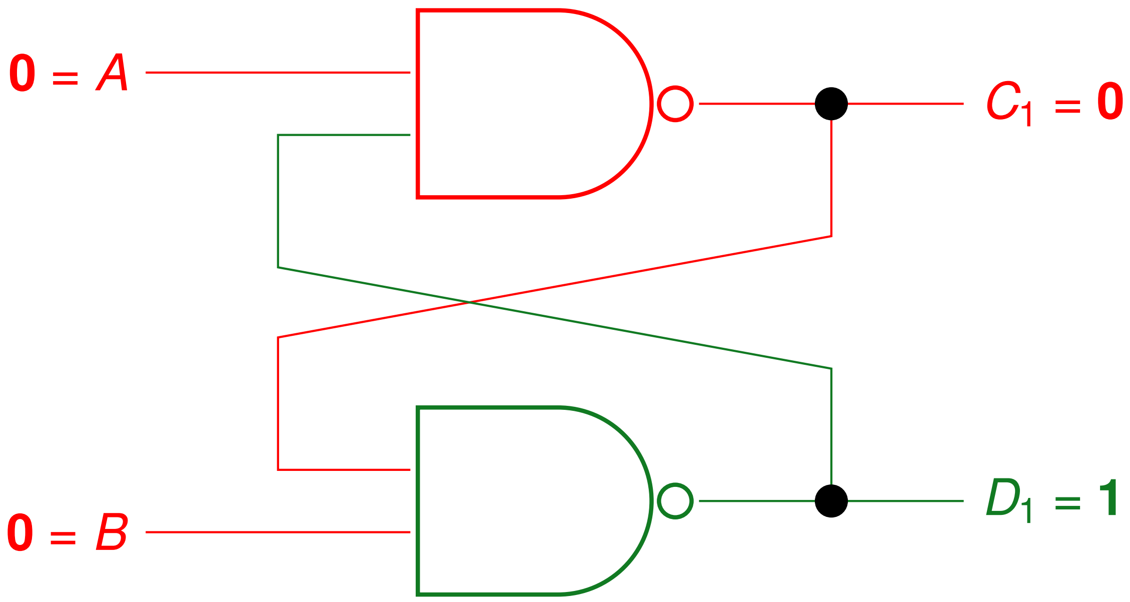
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 1) $\mapsto$ (1, 0) $\mapsto$ **(1, 1)**



> Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
> Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

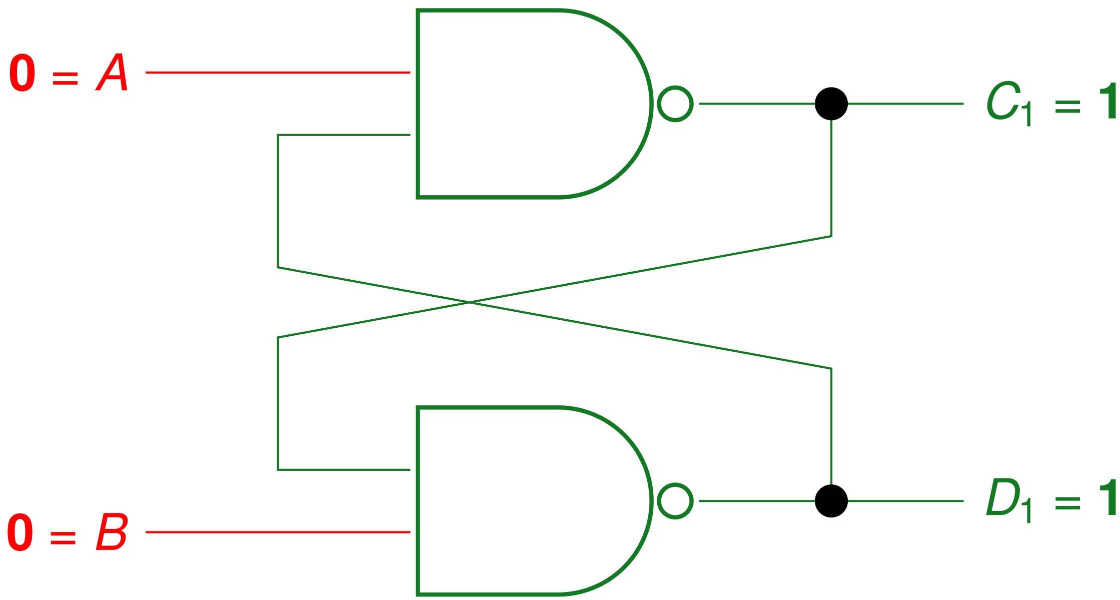
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 0) $\mapsto$ (1, 1) $\mapsto$ **(0, 0)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

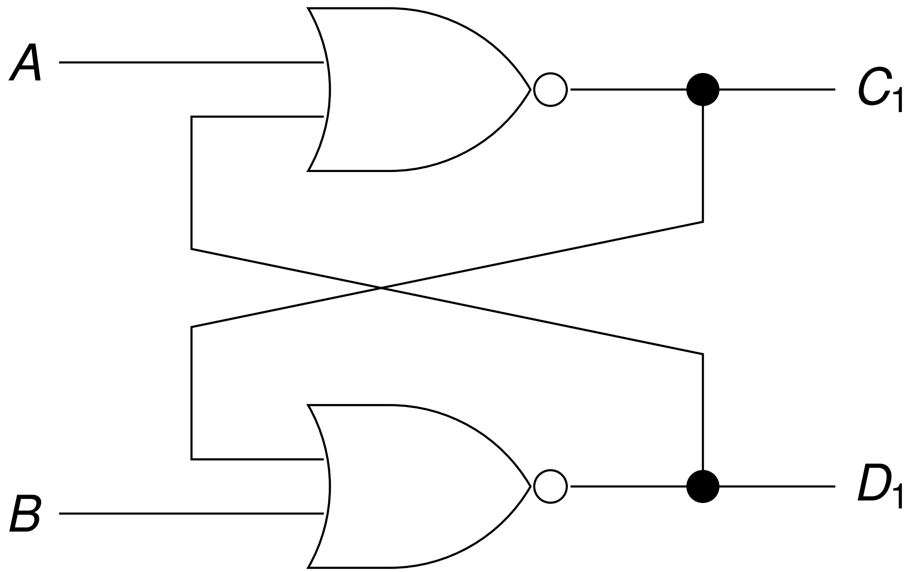
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 0) $\mapsto$ (1, 1) $\mapsto$ **(0, 0)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

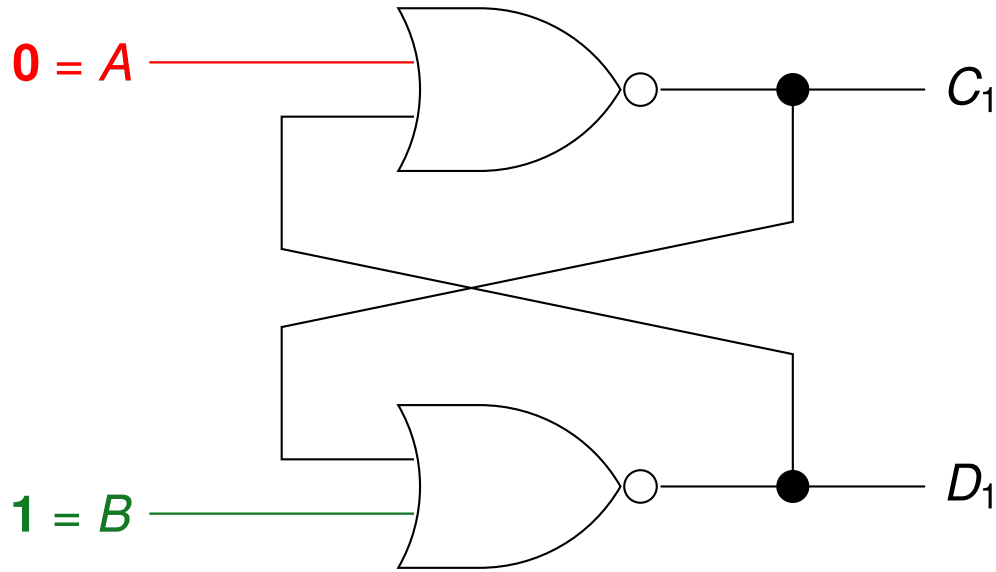
Aufgabe 4 – Latches und Flipflops

Anliegender Wert (**fett** hervorgehoben):



Aufgabe 4 – Latches und Flipflops

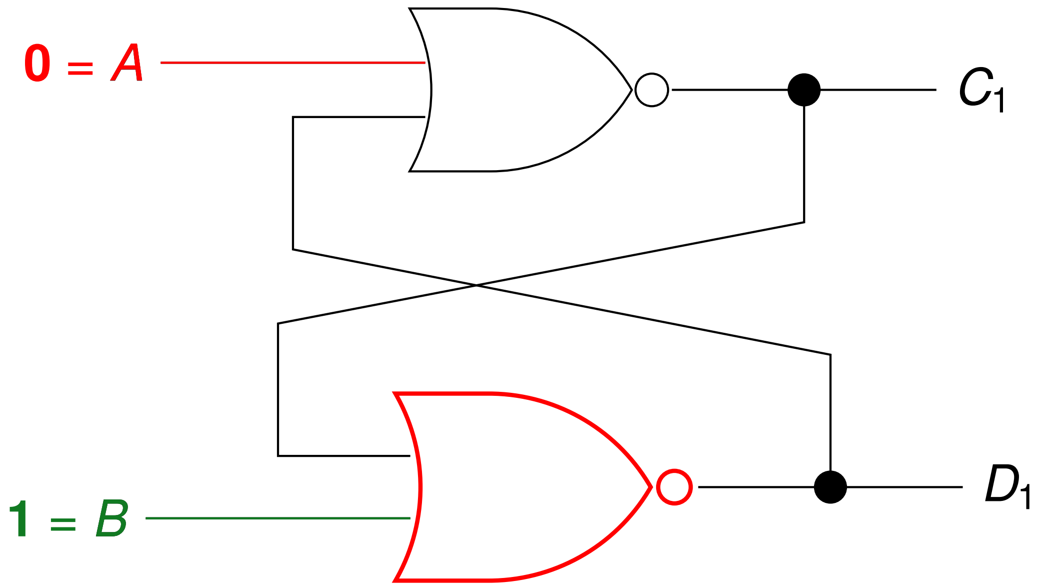
Anliegender Wert (**fett** hervorgehoben): **(0,1)**



Hat ein NOR mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein NOR **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

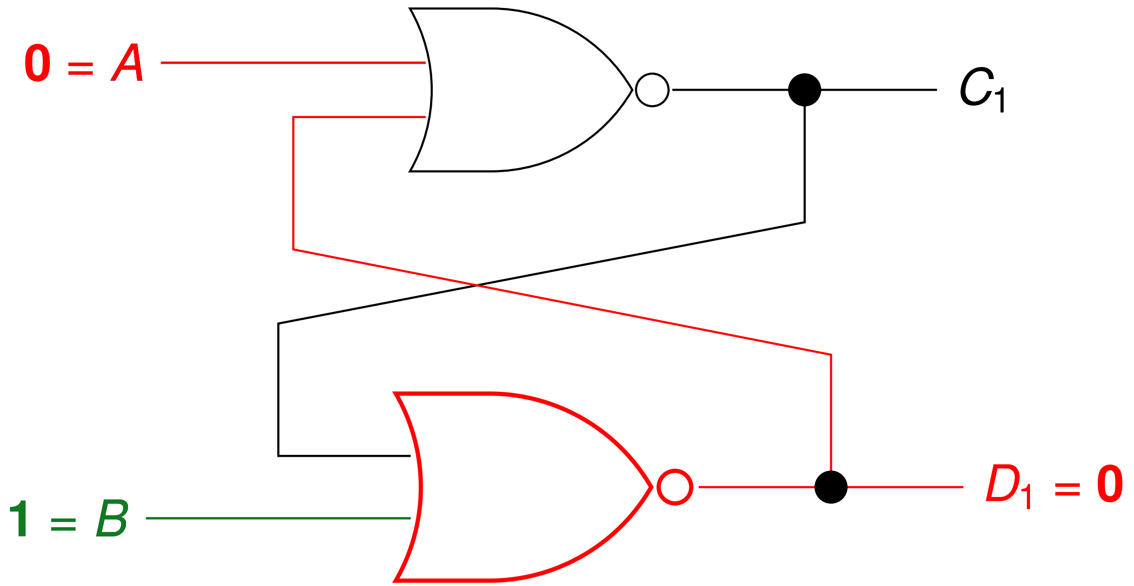
Anliegender Wert (**fett** hervorgehoben): **(0,1)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

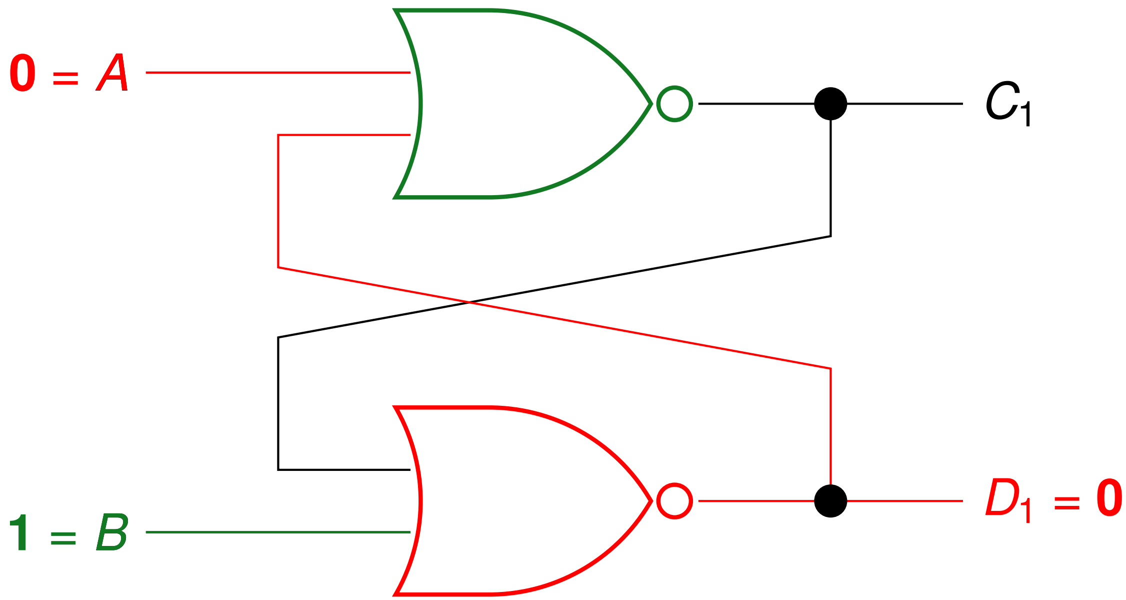
Anliegender Wert (**fett** hervorgehoben): **(0,1)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

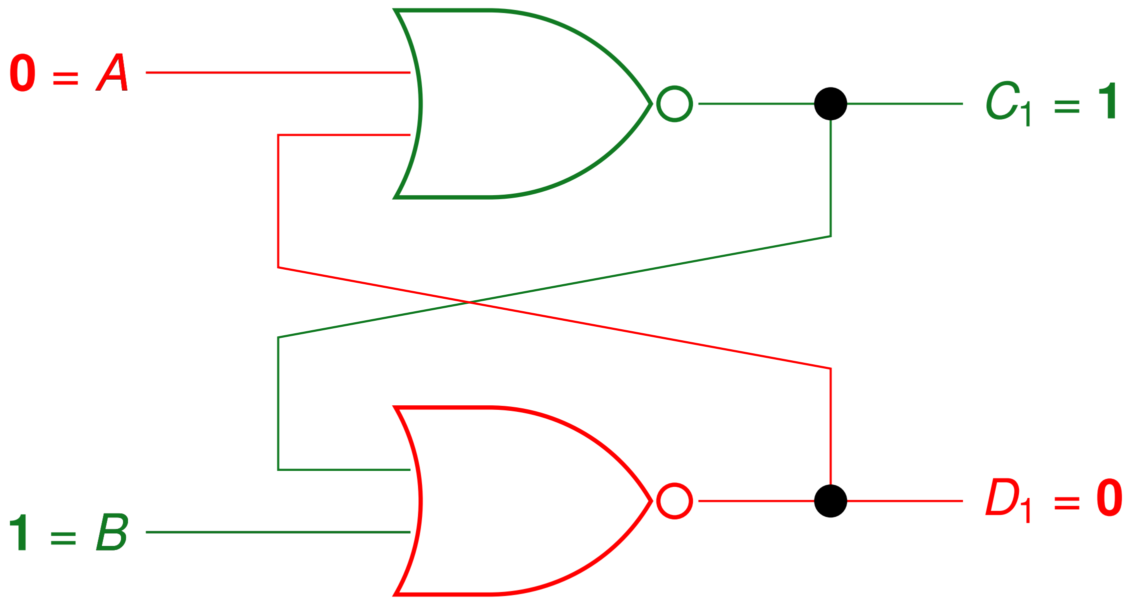
Anliegender Wert (**fett** hervorgehoben): **(0,1)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

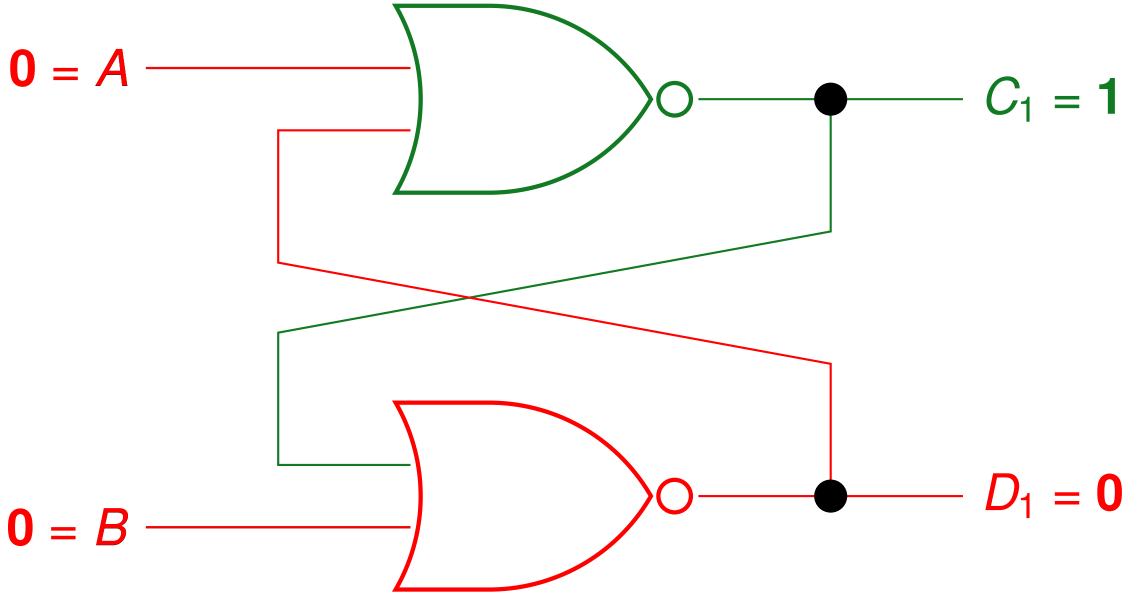
Anliegender Wert (**fett** hervorgehoben): **(0,1)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

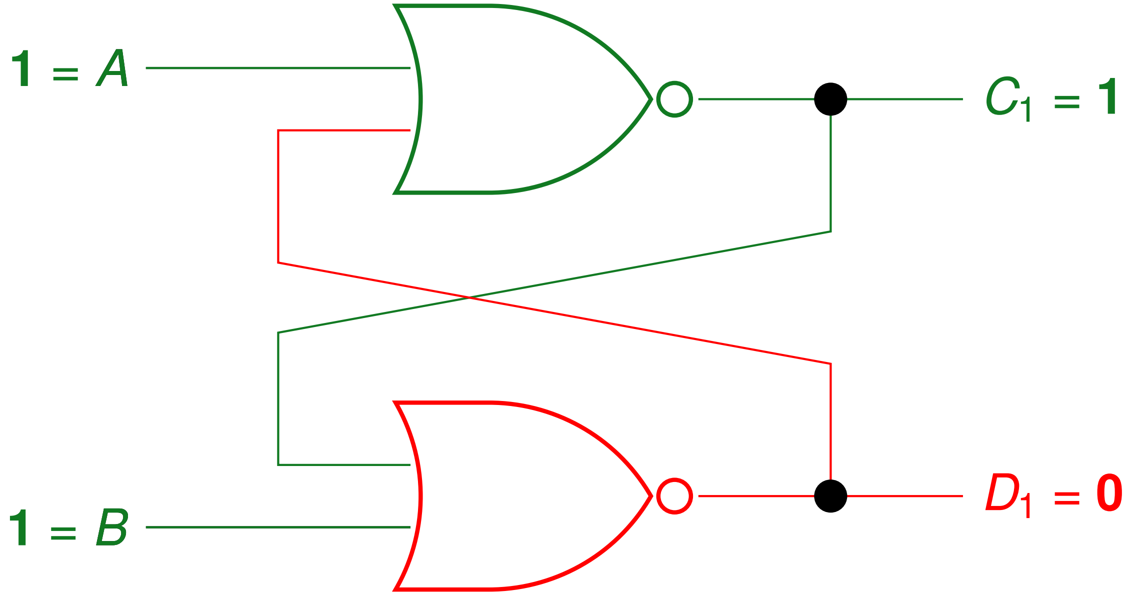
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ **(0, 0)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

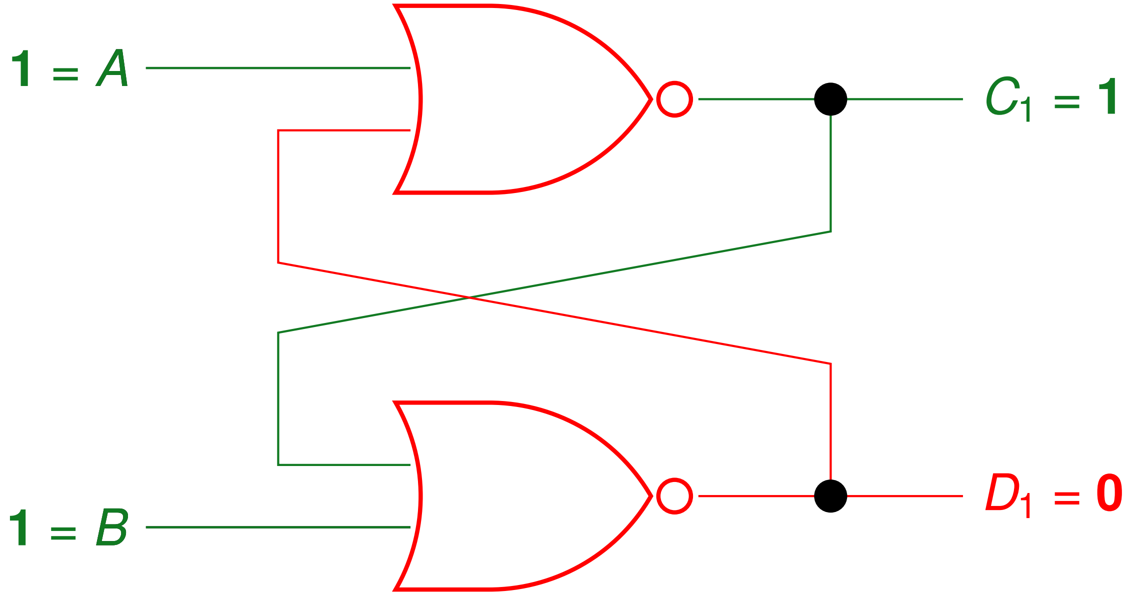
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ (0, 0) $\mapsto$ **(1, 1)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

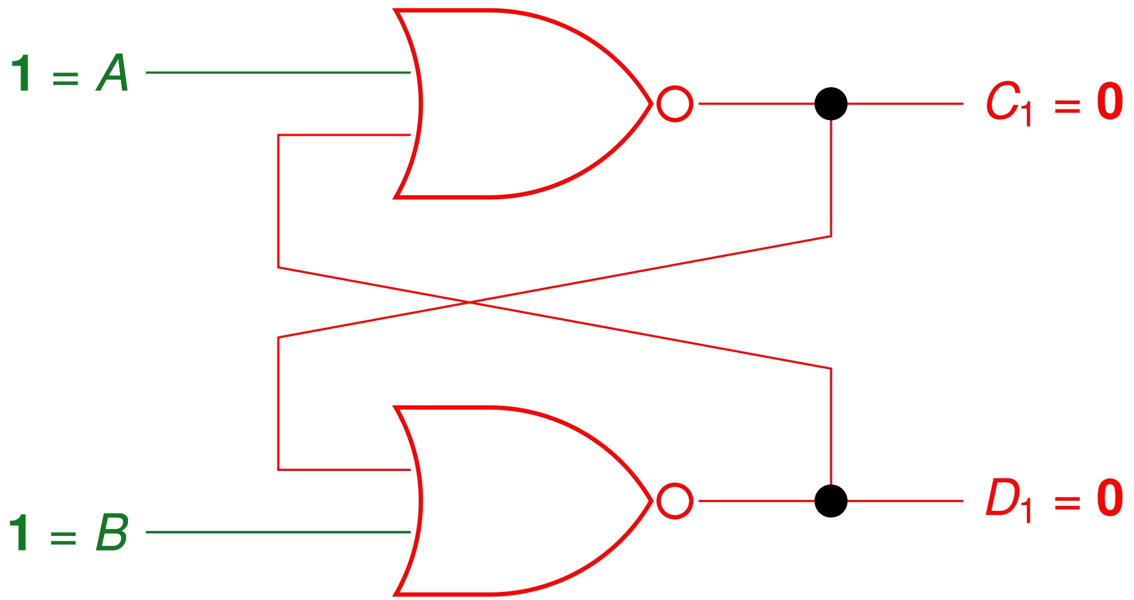
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ (0, 0) $\mapsto$ **(1, 1)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

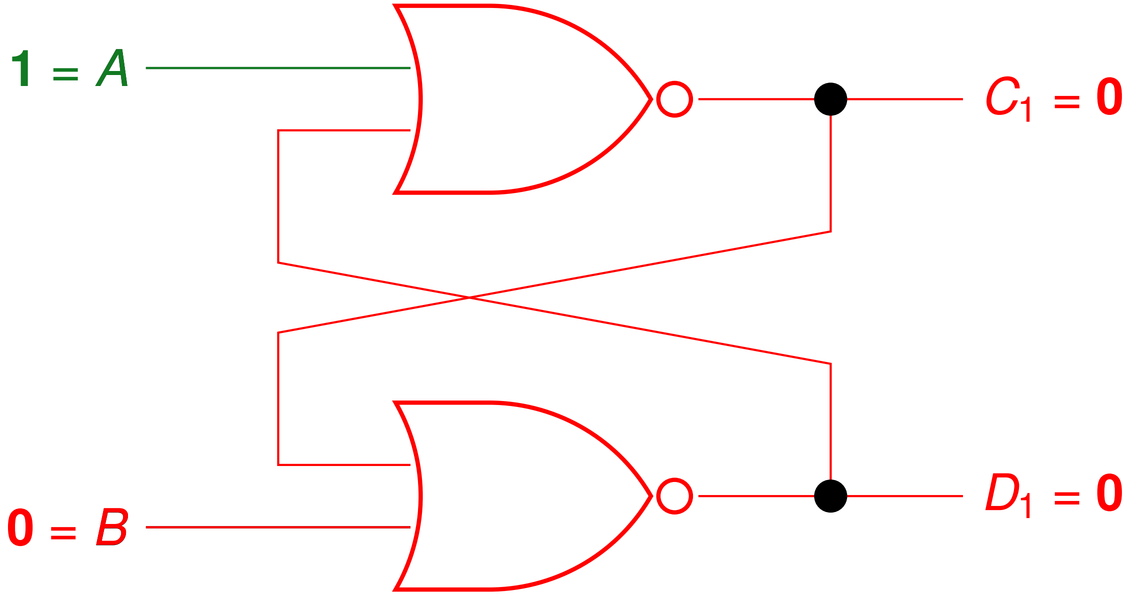
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ (0, 0) $\mapsto$ **(1, 1)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

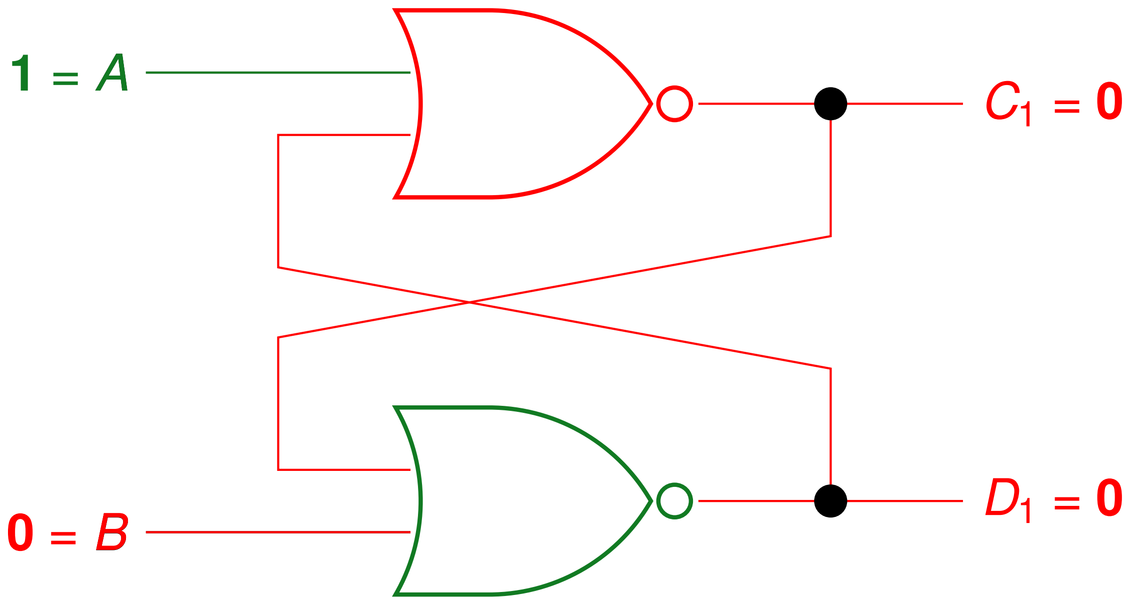
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (0, 0) $\mapsto$ (1, 1) $\mapsto$ **(1, 0)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

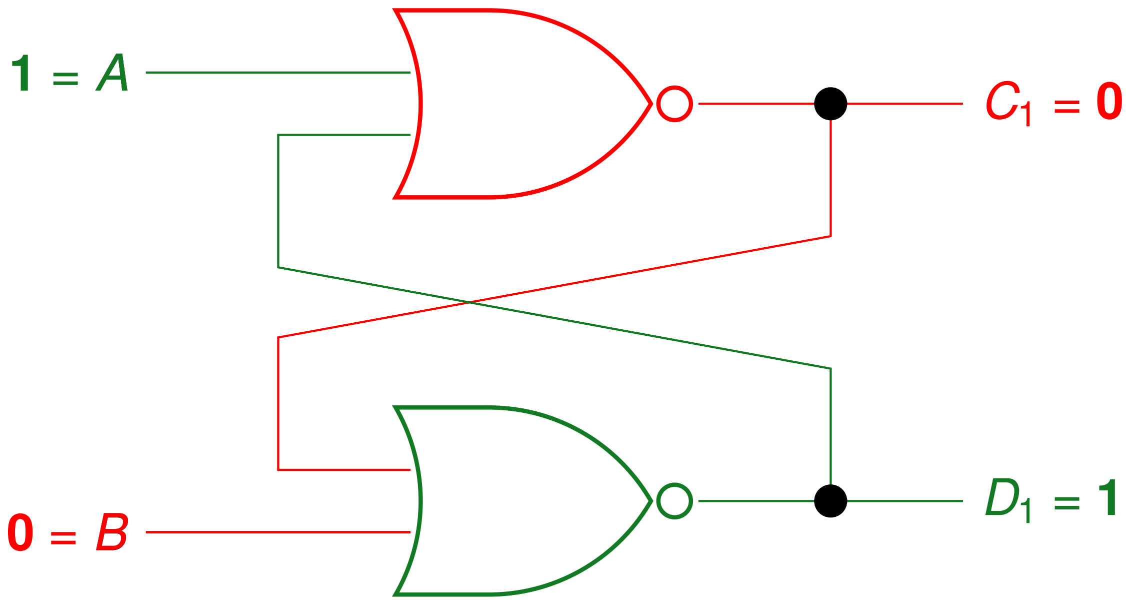
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (0, 0) $\mapsto$ (1, 1) $\mapsto$ **(1, 0)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

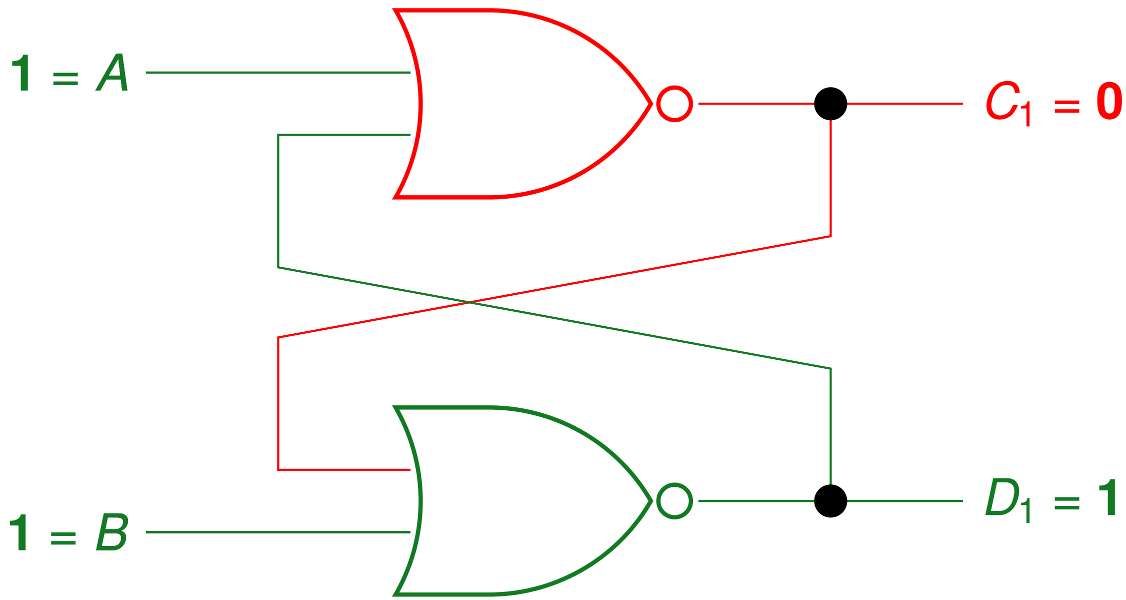
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (0, 0) $\mapsto$ (1, 1) $\mapsto$ **(1, 0)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

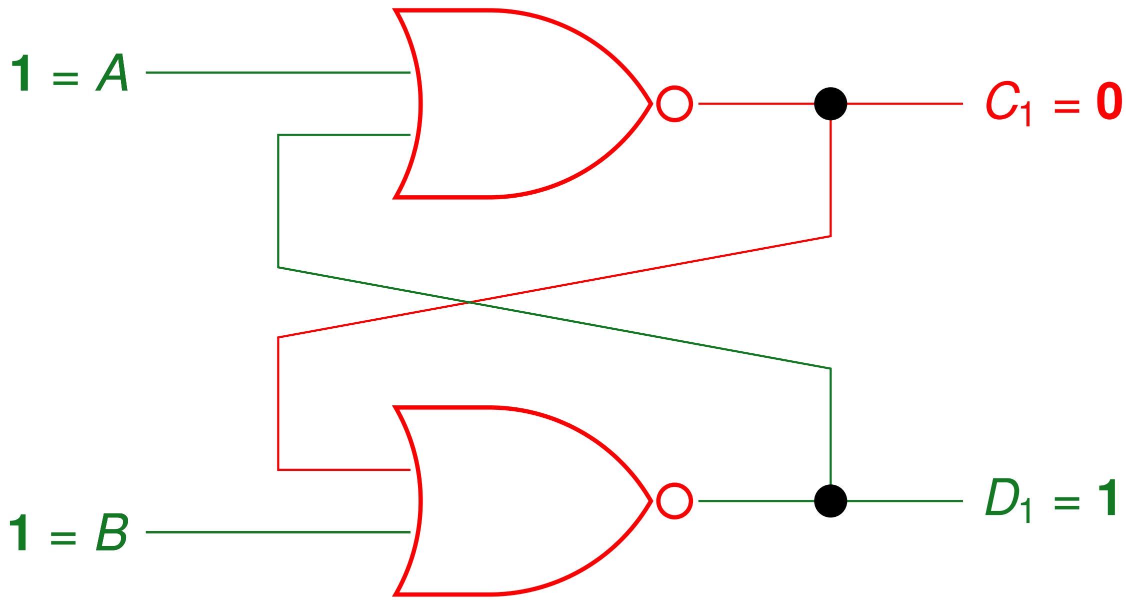
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 1) $\mapsto$ (1, 0) $\mapsto$ **(1, 1)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

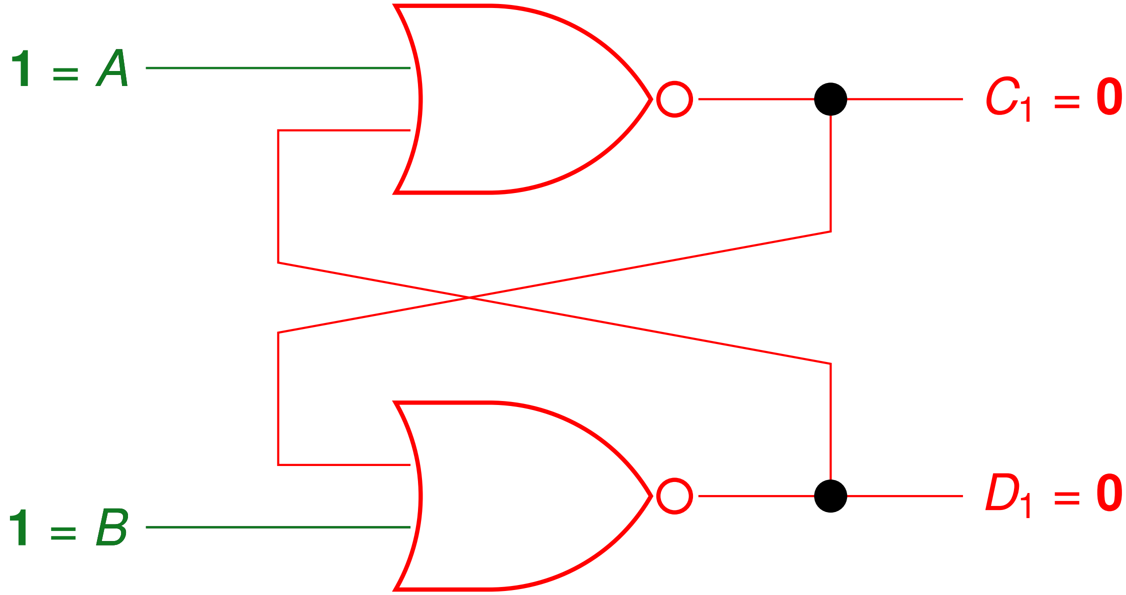
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 1) $\mapsto$ (1, 0) $\mapsto$ **(1, 1)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

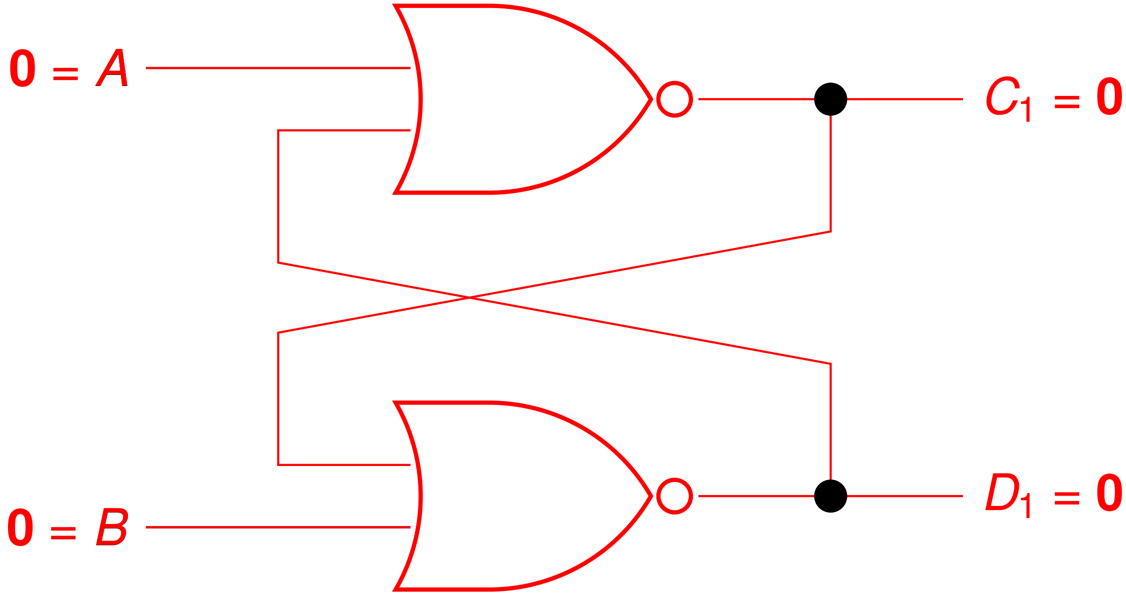
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 1) $\mapsto$ (1, 0) $\mapsto$ **(1, 1)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

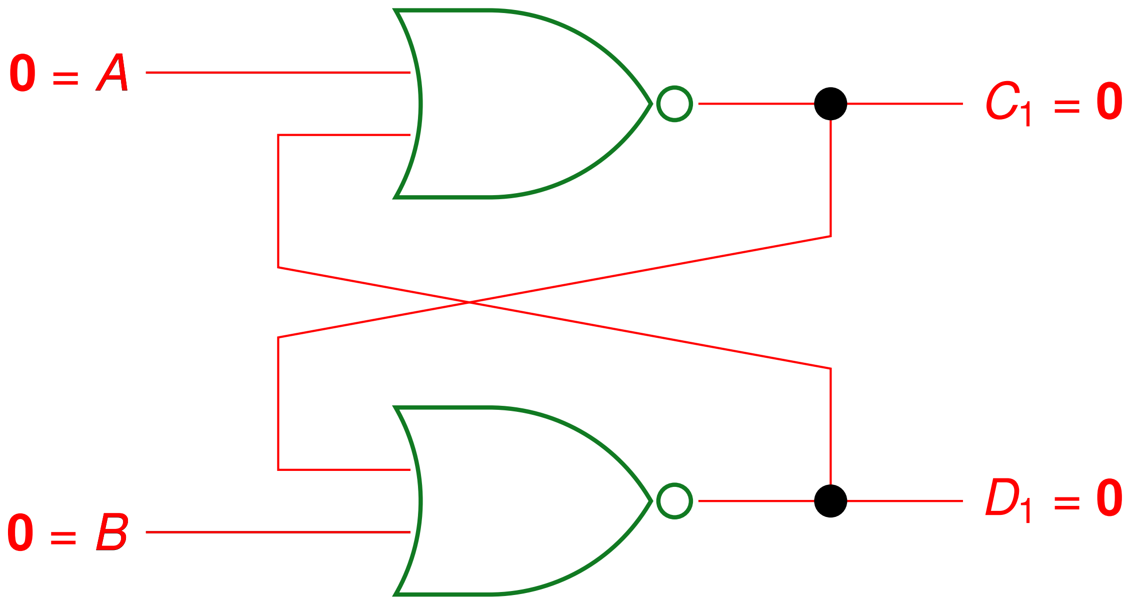
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 0) $\mapsto$ (1, 1) $\mapsto$ **(0, 0)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

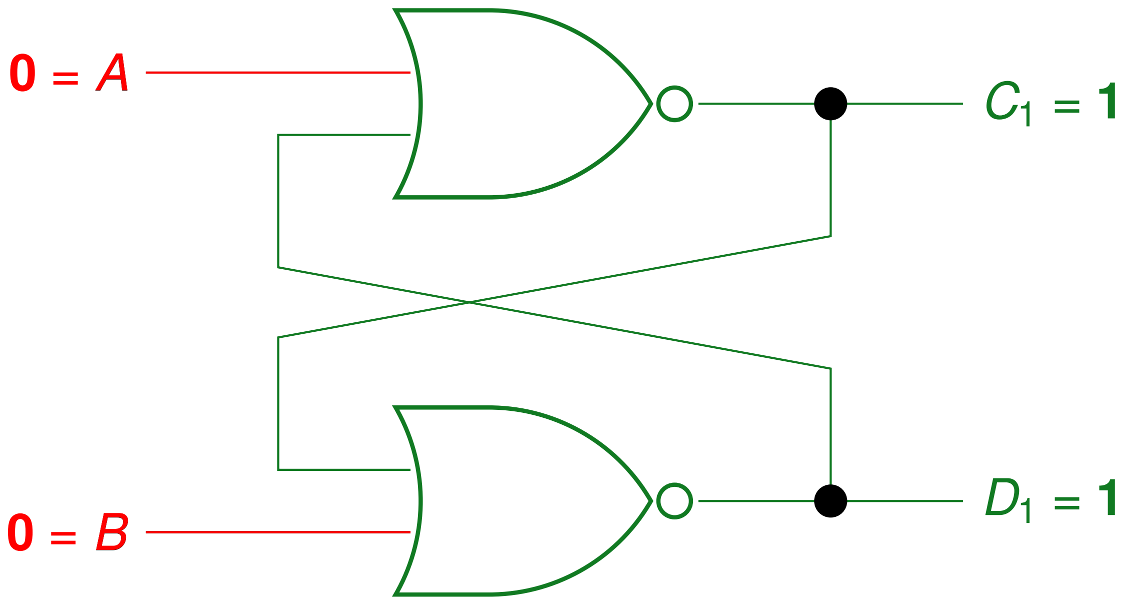
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 0) $\mapsto$ (1, 1) $\mapsto$ **(0, 0)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

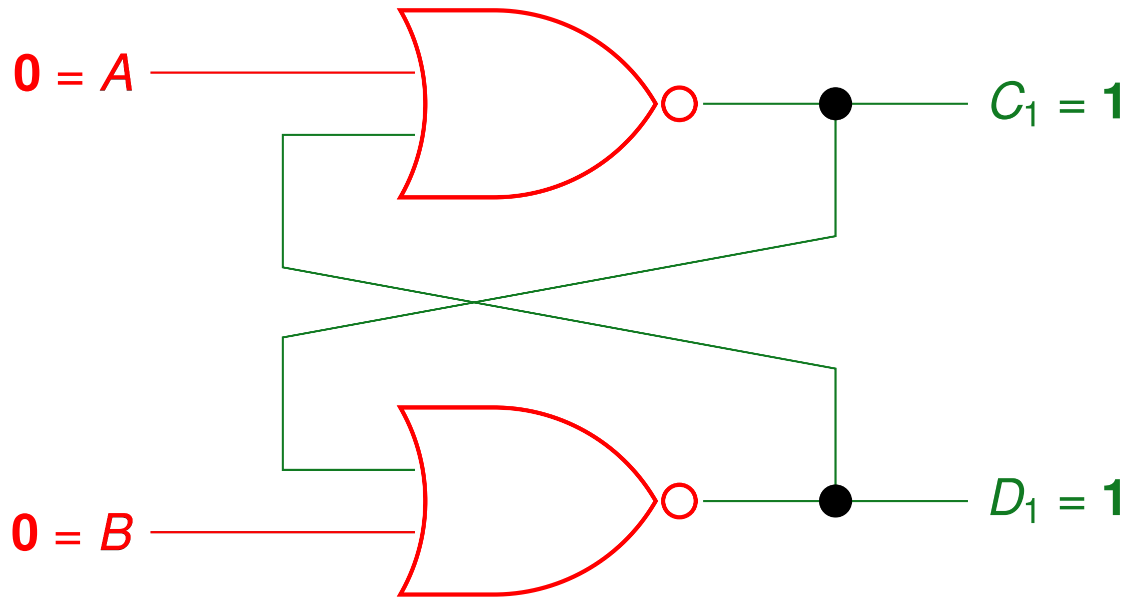
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 0) $\mapsto$ (1, 1) $\mapsto$ **(0, 0)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

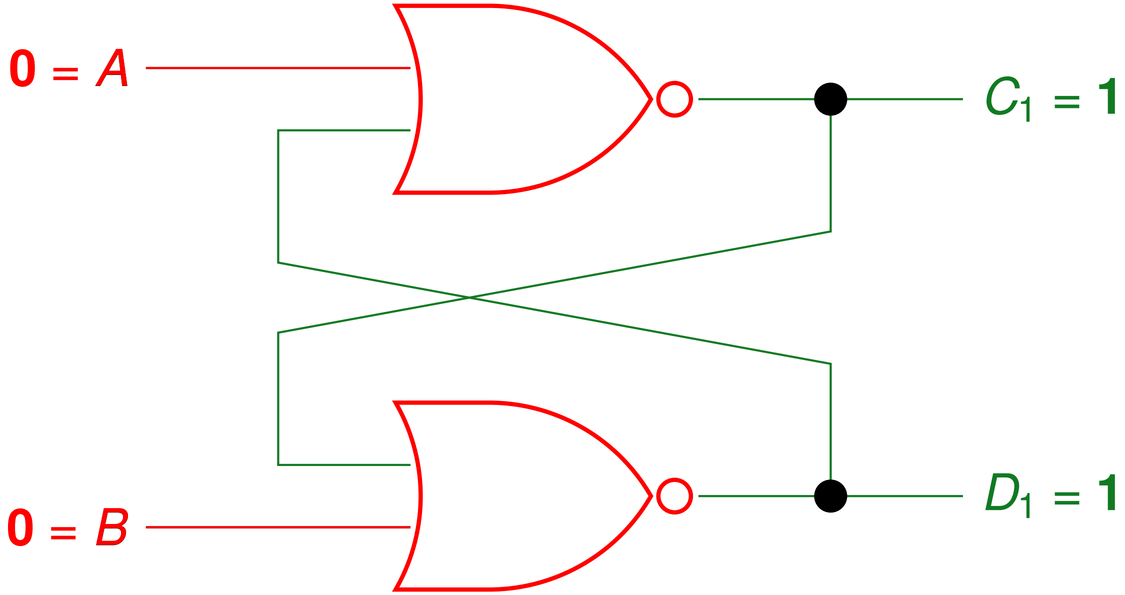
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 0) $\mapsto$ (1, 1) $\mapsto$ **(0, 0)**



Hat ein NOR mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein NOR **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

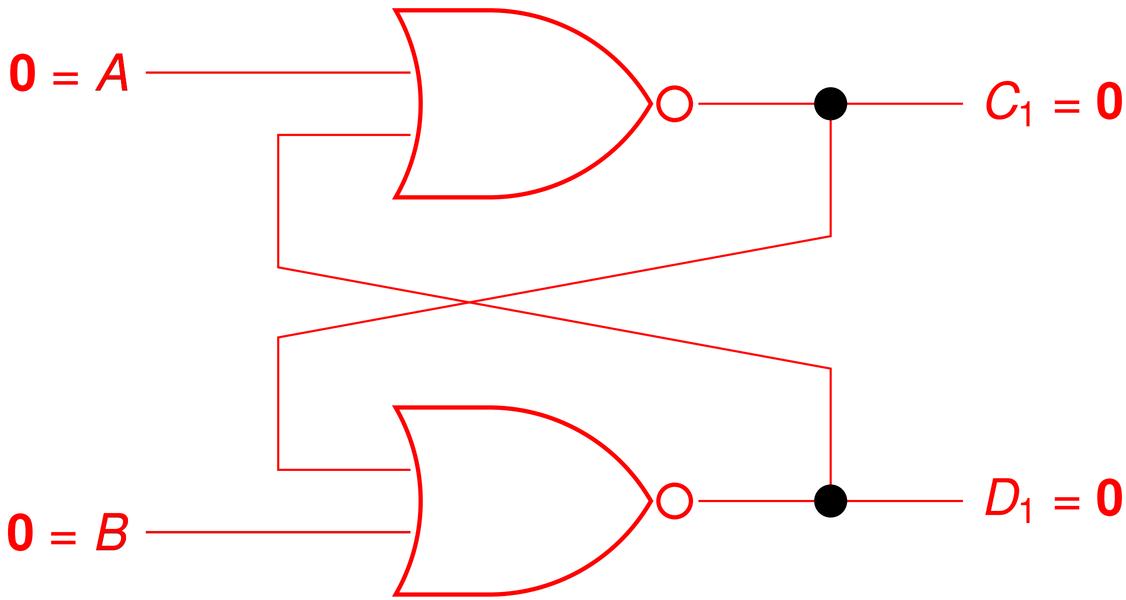
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 0) $\mapsto$ (1, 1) $\mapsto$ **(0, 0)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

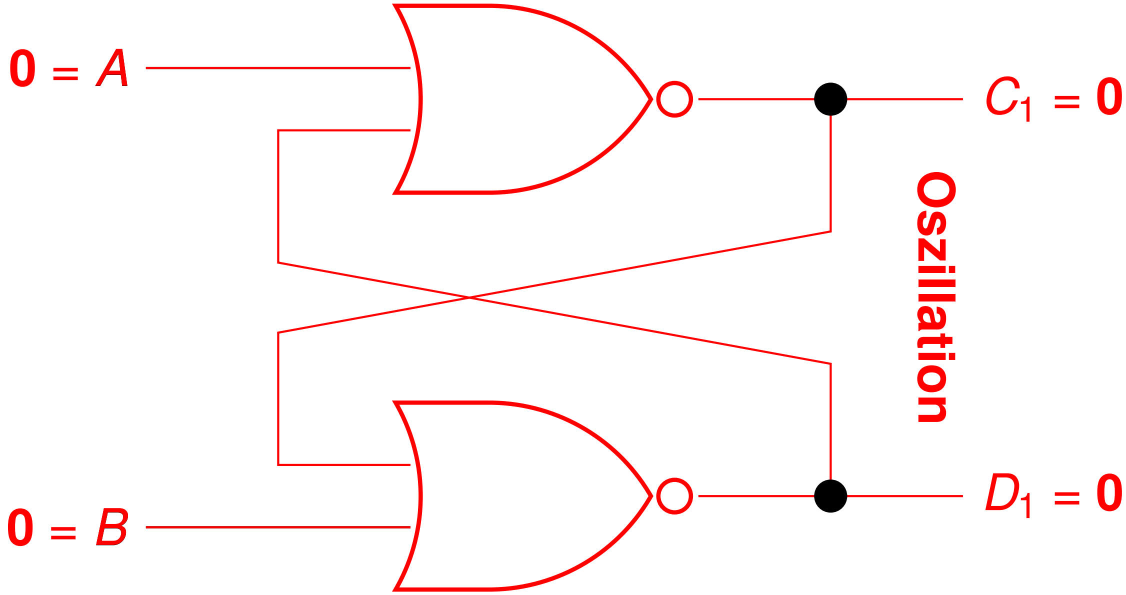
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 0) $\mapsto$ (1, 1) $\mapsto$ **(0, 0)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

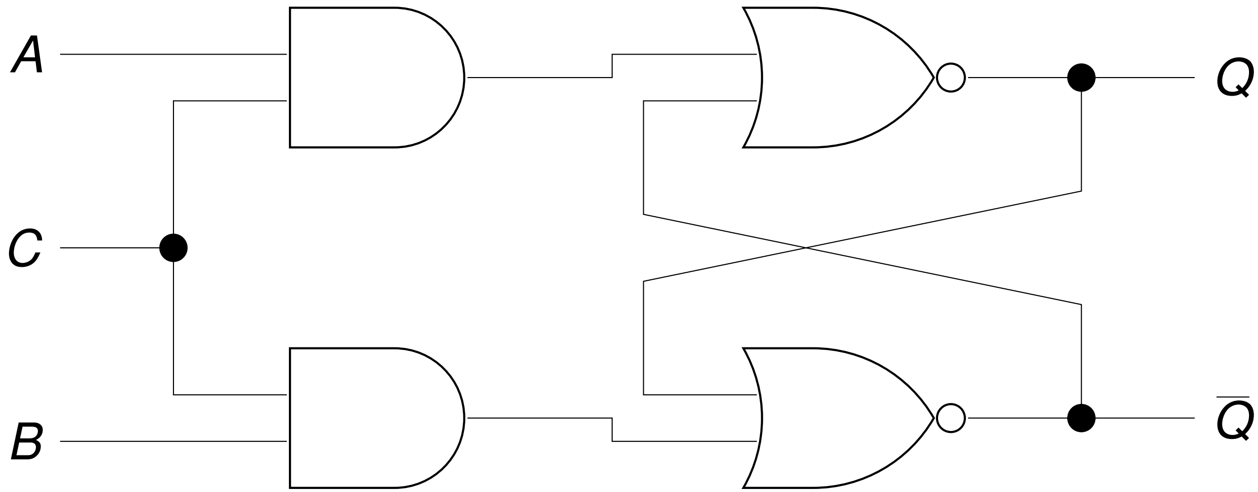
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 0) $\mapsto$ (1, 1) $\mapsto$ **(0, 0)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

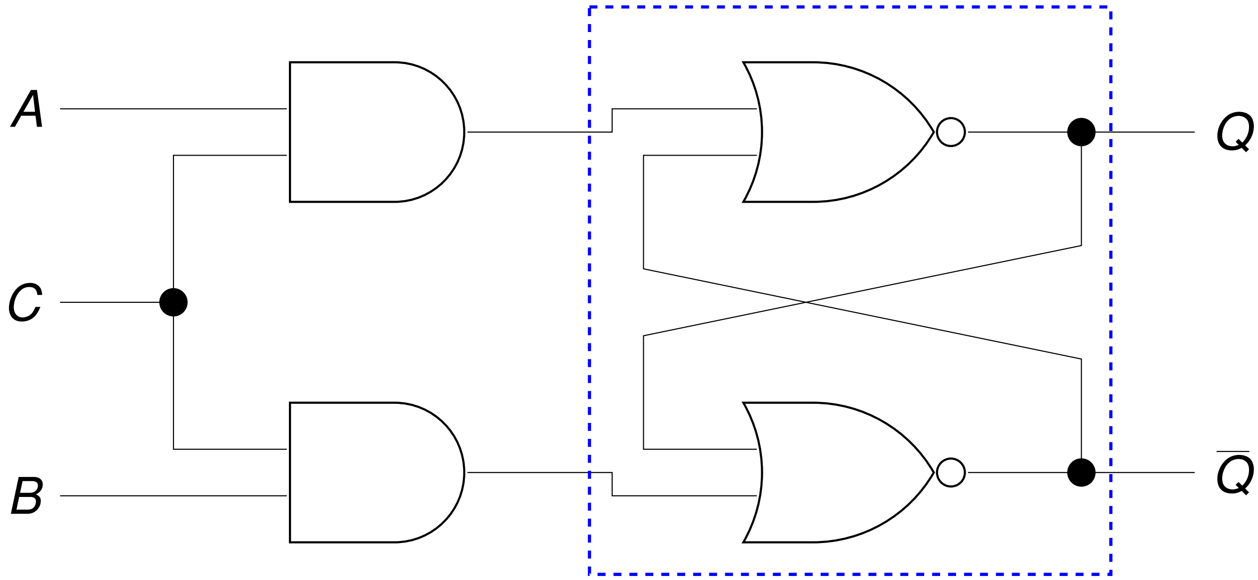
Aufgabe 4 – Latches und Flipflops

b) Sei C ein Taktsignal. Wie bezeichnet man dann das hier abgebildete Speicherelement?



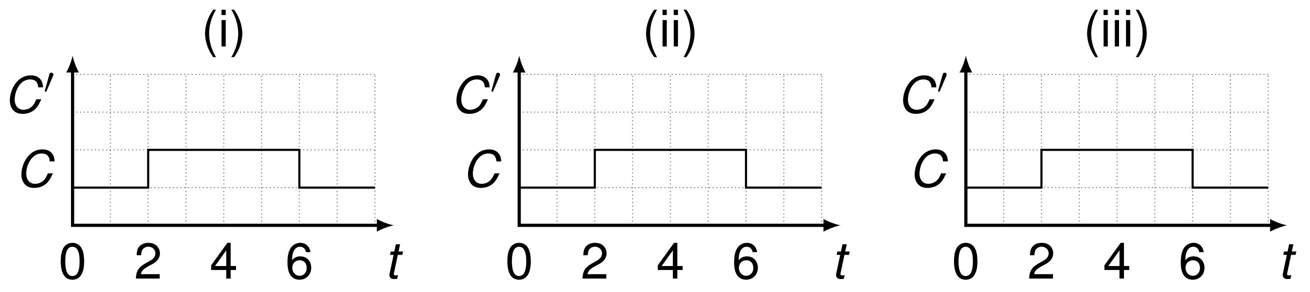
Aufgabe 4 – Latches und Flipflops

b) Sei C ein Taktsignal. Wie bezeichnet man dann das hier abgebildete Speicherelement?



Aufgabe 4 – Latches und Flipflops

b) Dieses soll so erweitert werden, dass es (i) nur bei der steigenden, (ii) der fallenden und (iii) bei jeder Flanke von C auf die Eingänge A und B reagiert. Geben Sie jeweils das Schaltnetz der Flankenerkennung $C \mapsto C'$ an und vervollständigen Sie die folgenden Wellenformdiagramme:



Aufgabe 4 – Latches und Flipflops

c) Erweitern Sie nun die Schaltung aus Teilaufgabe b) dahingehend, dass keine undefinierten Zustände, wie sie in Aufgabe a) der Fall waren, mehr auftreten.

Übungen zur Grundlagen der Technischen Informatik

Übung 10 – Master-Slave, Multiplexer, Shifter und Register

Florian Frank

Friedrich-Alexander-Universität Erlangen-Nürnberg

Wintersemester 2018/19

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Was machen wir heute?

Organisatorisches: Vorlesungsevaluation

Organisatorisches – Miniklausur

Blatt 9, Aufgabe 4 – Latches und Flipflops *revisited*

Aufgabe 1 – Master-Slave-Flipflops

Aufgabe 2 – Multiplexer und Demultiplexer

Aufgabe 3 – Barrel-Shifter

Aufgabe 4 – Schieberegister

Organisatorisches: Vorlesungsevaluation

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Organisatorisches – Vorlesungsevaluation

Bitte evaluiert die Veranstaltung, nur so können Dinge verbessert werden!
Auch bei keinen Verbesserungsvorschlägen freuen wir uns immer über positives Feedback, damit wir sehen, dass alles so gepasst hat!

- Bei Kommentaren in Freitextfeldern, die sich auf einen **bestimmten** Übungsleiter beziehen, gebt bitte **dessen Name bei diesen Kommentaren** mit an.
  - ⇝ Kommentare werden durcheinandergewürfelt ...
    — Das heißt nicht nur einmal den Namen angeben, sondern immer!
  - ⇝ Ihr evaluiert die Gesamtveranstaltung „Übungen zu den Grundlagen der Technischen Informatik“ und nicht – wie in AuD oder GRa bspw. – die einzelnen Übungen, deswegen gebt bitte die Namen mit an, wir wären euch sehr verbunden.

Ihr habt noch bis zum **26.01 um 12$^{\underline{00}}$ Uhr** Zeit zu evaluieren!

Web-basierte Evaluation an der Technischen Fakultät

Evaluation der Vorlesungen, Übungen, Seminare u. Praktika

- Fragen zur Lehrveranstaltung, zur Dozentin/zum Dozent
  + optionale Zusatzfragen (von der Dozentin/vom Dozenten gestellt)

- Die **TAN-Zettel** für die Lehrveranstaltungen erhalten Sie in der LV jeweils von der Dozentin/vom Dozenten. Bei Nichtbenutzung bitte zerstören!

Informationen und Evaluation → **http://eva.tf.fau.de**

Die Auswertung der LV-Umfragen erfolgt automatisiert und kurzfristig, um die Erkenntnisse noch vor Semesterende mit den Studierenden diskutieren zu können.

Tragen Sie zu einem repräsentativen Ergebnis bei!

Evaluieren Sie unter:

http://eva.tf.fau.de

Frist: Sa., 27. Januar 2018, 12^{00} Uhr (!)

Organisatorisches – Miniklausur

design

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Achtung – Miniklausur

Achtung – Miniklausur

Diesen Donnerstag – am 17. Januar 2019 – findet die **zweite** Miniklausur statt.

Wo?	im H7
Wann?	Wir starten um 16:15 Uhr!
	*Seid bitte deswegen schon **ungefähr** 3 – 5 **Minuten** vor Beginn da.*
Worum gehts?	Um den Übungsstoff bis Blatt 10 – sprich heute –
	und den Vorlesungsstoff bis einschließlich zum 15.01.2019

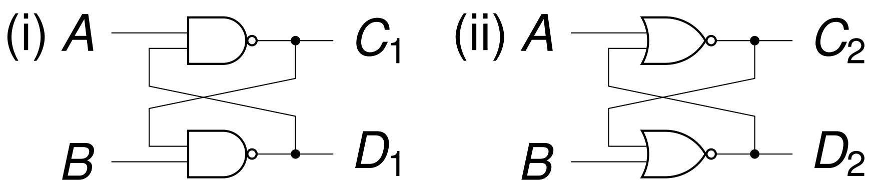
Blatt 9, Aufgabe 4 – Latches und Flipflops *revisited*

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 4 – Latches und Flipflops

Die Verzögerungszeit jedes Logikgatters in dieser Aufgabe betrage τ = 1 ns.

a) An die Eingänge (A, B) der unten stehenden Schaltungen (i) und (ii) werden nacheinander folgende Werte angelegt: $(0, 1)$, $(0, 0)$, $(1, 1)$, $(1, 0)$, $(1, 1)$ und $(0, 0)$. Geben Sie jeweils die Ausgangswerte von (i) und (ii) an und benennen Sie die Signale A bis D_2 sinnvoll.



Aufgabe 4 – Latches und Flipflops: Begriffklärung

Speicherelement

Ein Speicherelement ist ein Gerät/Modul/Element, das einen vorher angelegten Wert speichert.

Aufgabe 4 – Latches und Flipflops: Begriffklärung

Speicherelement

Ein Speicherelement ist ein Gerät/Modul/Element, das einen vorher angelegten Wert speichert.

Eigenschaften von Speicherelementen

Speicherelemente können ...

Operation	Q^t	Q^{t+1}
S(etze)	—	1
R(ücksetze)	—	0
N(ix)	0	0
	1	1

Operationen eines Speicherelements

Aufgabe 4 – Latches und Flipflops: Begriffklärung

Speicherelement

Ein Speicherelement ist ein Gerät/Modul/Element, das einen vorher angelegten Wert speichert.

Eigenschaften von Speicherelementen

Speicherelemente können ...

asynchron sein, wenn Änderungen jederzeit möglich sind.

Operation	Q^t	Q^{t+1}
S(etze)	—	1
R(ücksetze)	—	0
N(ix)	0	0
	1	1

Operationen eines Speicherelements

Aufgabe 4 – Latches und Flipflops: Begriffklärung

Speicherelement

Ein Speicherelement ist ein Gerät/Modul/Element, das einen vorher angelegten Wert speichert.

Eigenschaften von Speicherelementen

Speicherelemente können ...

asynchron sein, wenn Änderungen jederzeit möglich sind.

synchron sein, wenn Änderungen nur zu vorher festgelegten Momenten möglich sind. Diese Momente können ...

Operation	Q^t	Q^{t+1}
S(etze)	—	1
R(ücksetze)	—	0
N(ix)	0	0
	1	1

Operationen eines Speicherelements

Aufgabe 4 – Latches und Flipflops: Begriffklärung

Speicherelement

Ein Speicherelement ist ein Gerät/Modul/Element, das einen vorher angelegten Wert speichert.

Eigenschaften von Speicherelementen

Speicherelemente können ...

asynchron sein, wenn Änderungen jederzeit möglich sind.

synchron sein, wenn Änderungen nur zu vorher festgelegten Momenten möglich sind. Diese Momente können ...

pegelabhängig sein. Mann nennt diese Elemente dann auch pegelgesteuert.

Operation	Q^t	Q^{t+1}
S(etze)	—	1
R(ücksetze)	—	0
N(ix)	0	0
	1	1

Operationen eines Speicherelements

Aufgabe 4 – Latches und Flipflops: Begriffklärung

Speicherelement

Ein Speicherelement ist ein Gerät/Modul/Element, das einen vorher angelegten Wert speichert.

Eigenschaften von Speicherelementen

Speicherelemente können ...

asynchron sein, wenn Änderungen jederzeit möglich sind.

synchron sein, wenn Änderungen nur zu vorher festgelegten Momenten möglich sind. Diese Momente können ...

- **pegelabhängig** sein. Mann nennt diese Elemente dann auch pegelgesteuert.
- **taktflankenabhängig** sein. Mann nennt diese Elemente dann auch taktflankengesteuert. Es können zu **steigender**, **fallender** oder **zu beiden** Flanken Änderungen möglich sein.

Operation	Q^t	Q^{t+1}
S(etze)	—	1
R(ücksetze)	—	0
N(ix)	0	0
	1	1

Operationen eines Speicherelements

Aufgabe 4 – Latches und Flipflops: Begriffklärung

Speicherelement

Ein Speicherelement ist ein Gerät/Modul/Element, das einen vorher angelegten Wert speichert.

Flipflops

Flipflops sind Elemente zur Speicherung eines Bits.
Sie sind – bei uns – **rein taktflanken-gesteuert**.

Aufgabe 4 – Latches und Flipflops: Begriffklärung

Speicherelement

Ein Speicherelement ist ein Gerät/Modul/Element, das einen vorher angelegten Wert speichert.

Latches

Latches sind ebenfalls Elemente zur Speicherung eines Bits, ähnlich zu den Flipflops.
Im Gegensatz zu Flipflops sind sie aber **rein pegelgesteuert.**.

Aufgabe 4 – Latches und Flipflops: Begriffklärung

Speicherelement

Ein Speicherelement ist ein Gerät/Modul/Element, das einen vorher angelegten Wert speichert.

Flipflops

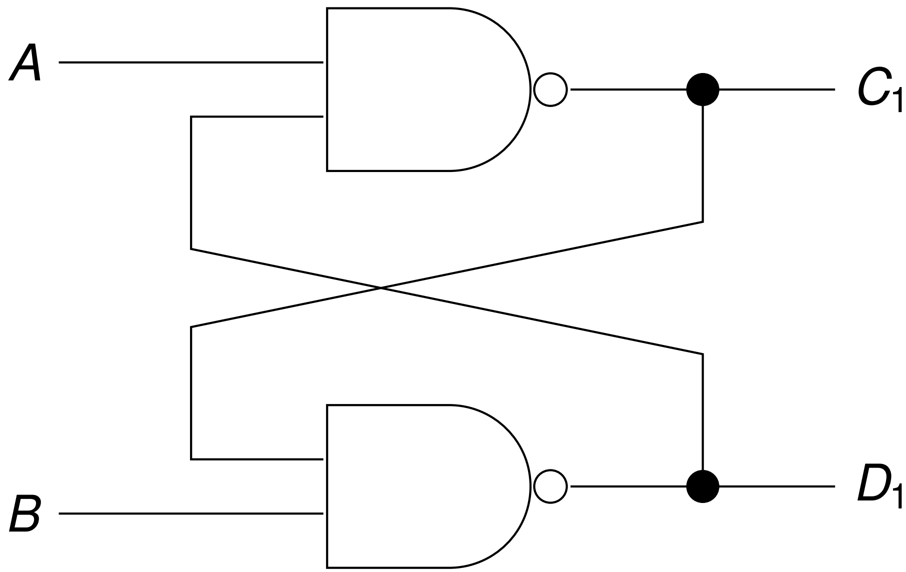
Flipflops sind Elemente zur Speicherung eines Bits.
Sie sind – bei uns – **rein taktflankengesteuert**.

Latches

Latches sind ebenfalls Elemente zur Speicherung eines Bits, ähnlich zu den Flipflops.
Im Gegensatz zu Flipflops sind sie aber **rein pegelgesteuert.**.

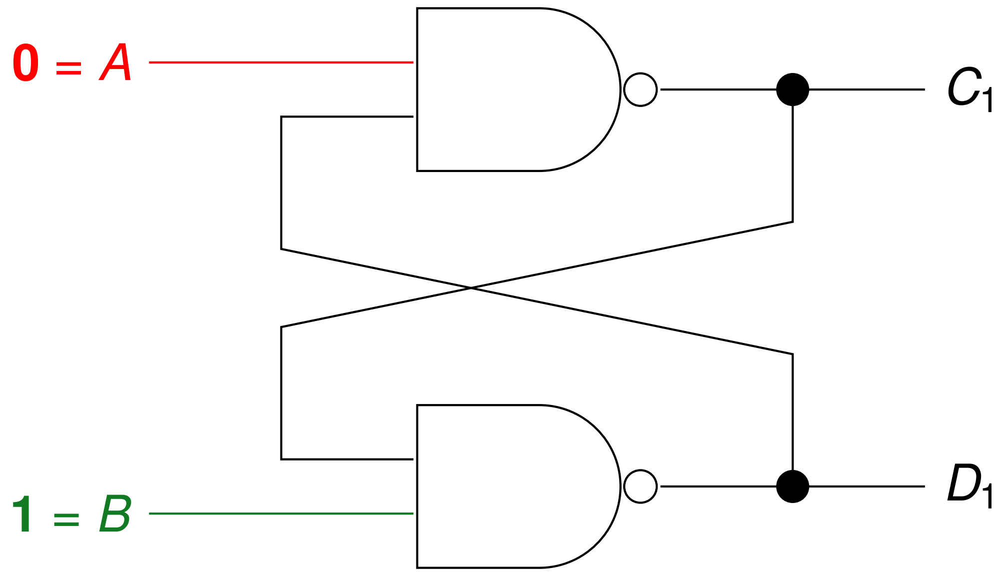
Aufgabe 4 – Latches und Flipflops

Anliegender Wert (**fett** hervorgehoben):



Aufgabe 4 – Latches und Flipflops

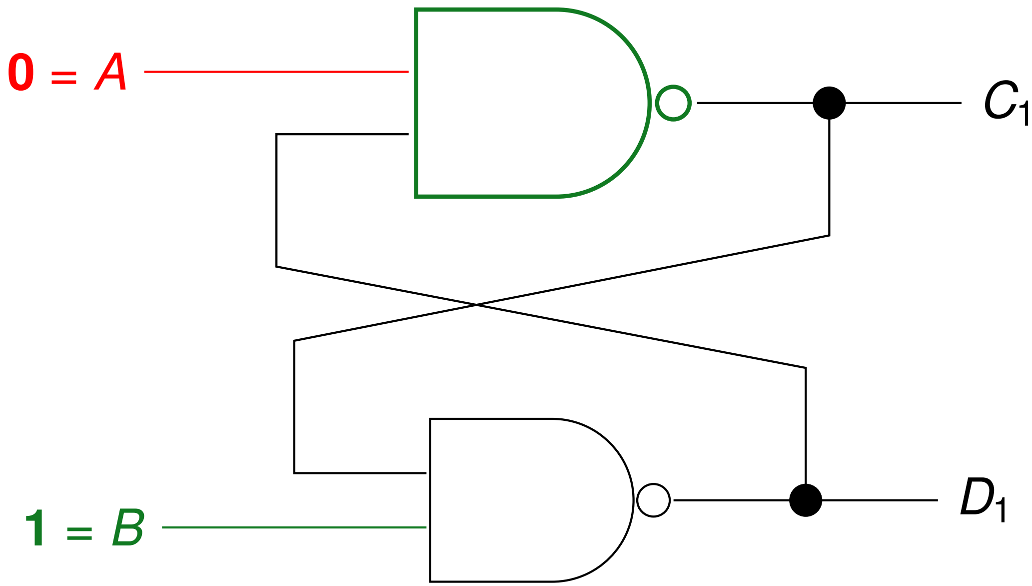
Anliegender Wert (**fett** hervorgehoben): **(0,1)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

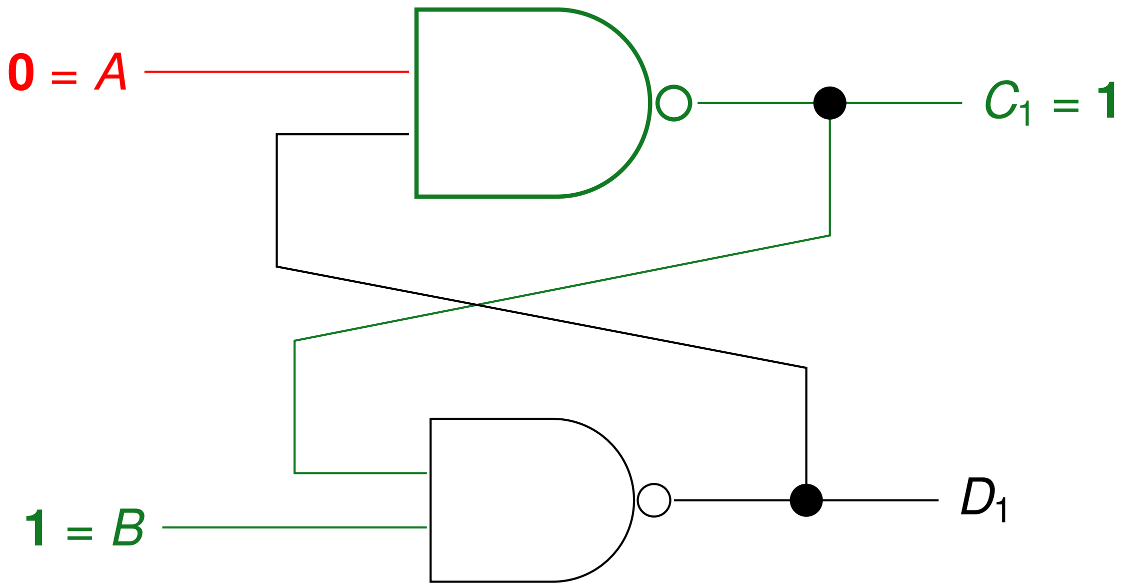
Anliegender Wert (**fett** hervorgehoben): **(0,1)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

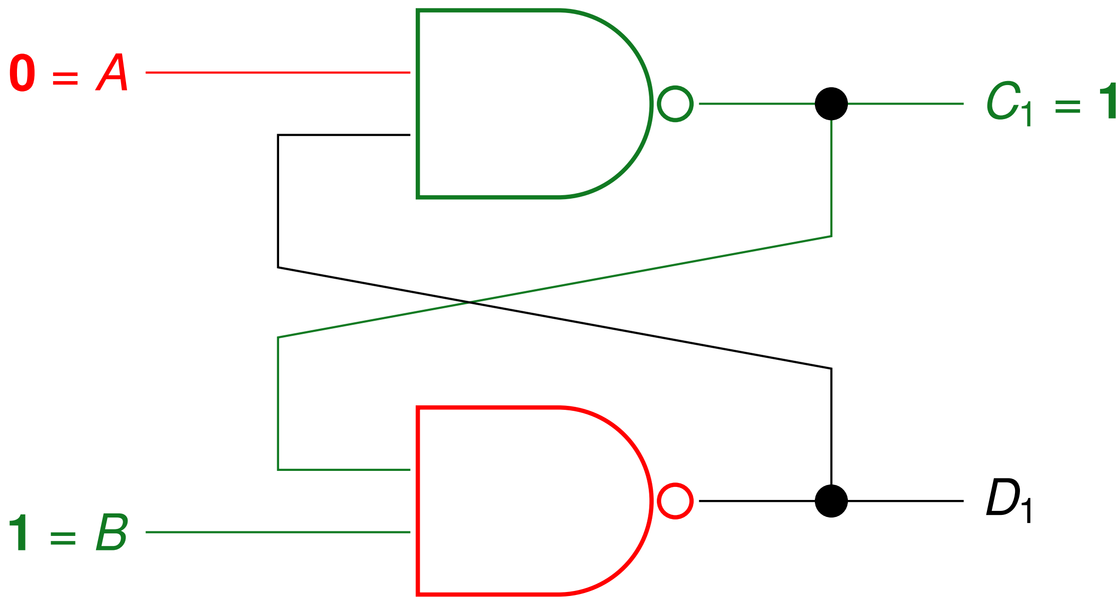
Anliegender Wert (**fett** hervorgehoben): **(0,1)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

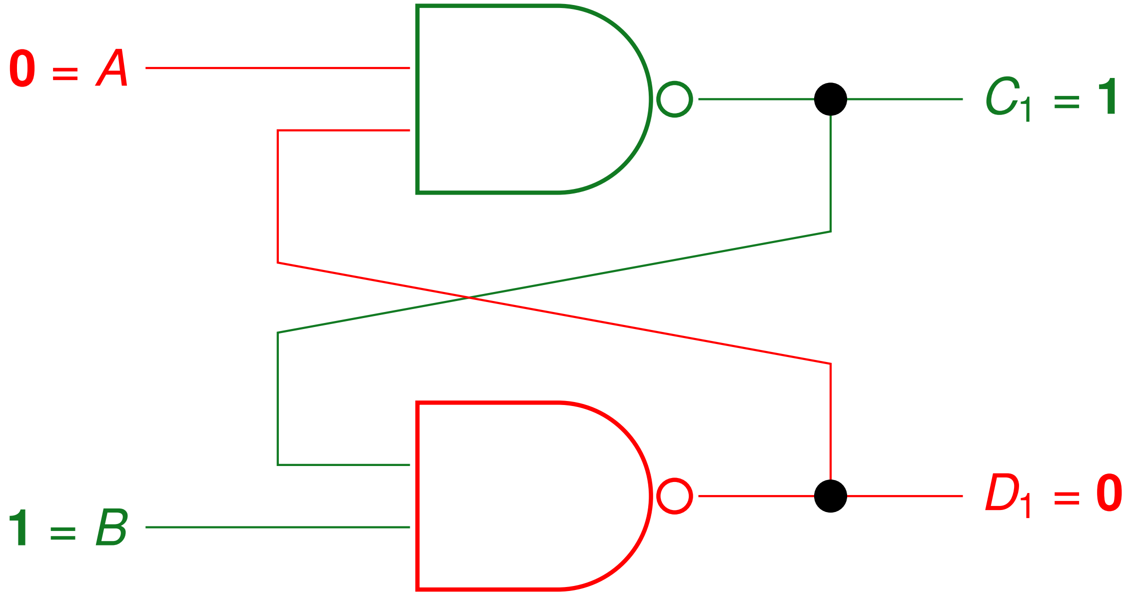
Anliegender Wert (**fett** hervorgehoben): **(0,1)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

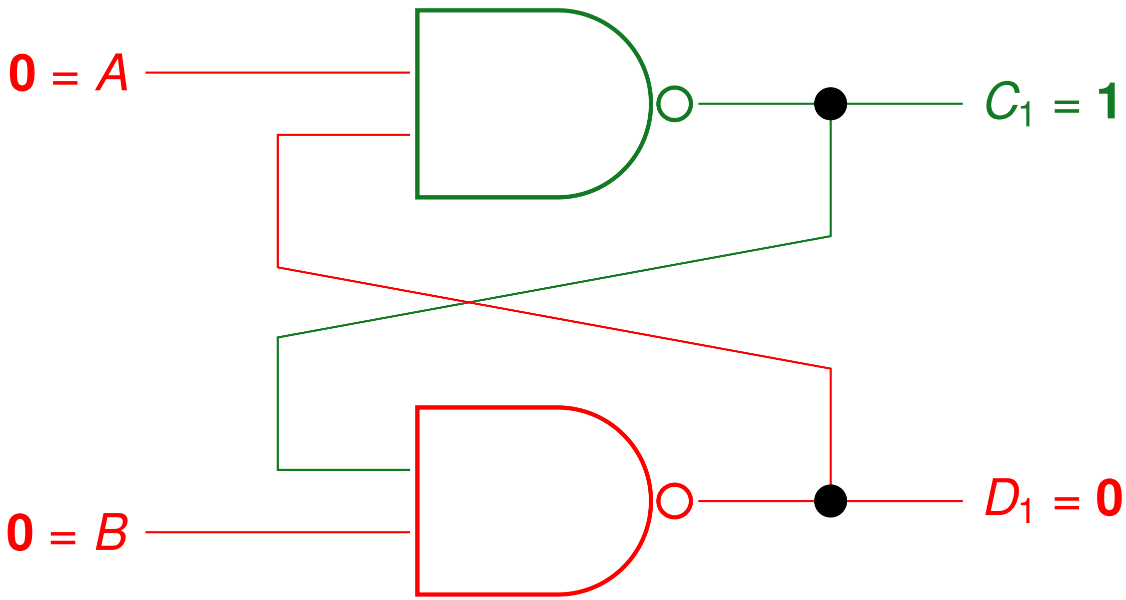
Anliegender Wert (**fett** hervorgehoben): **(0,1)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

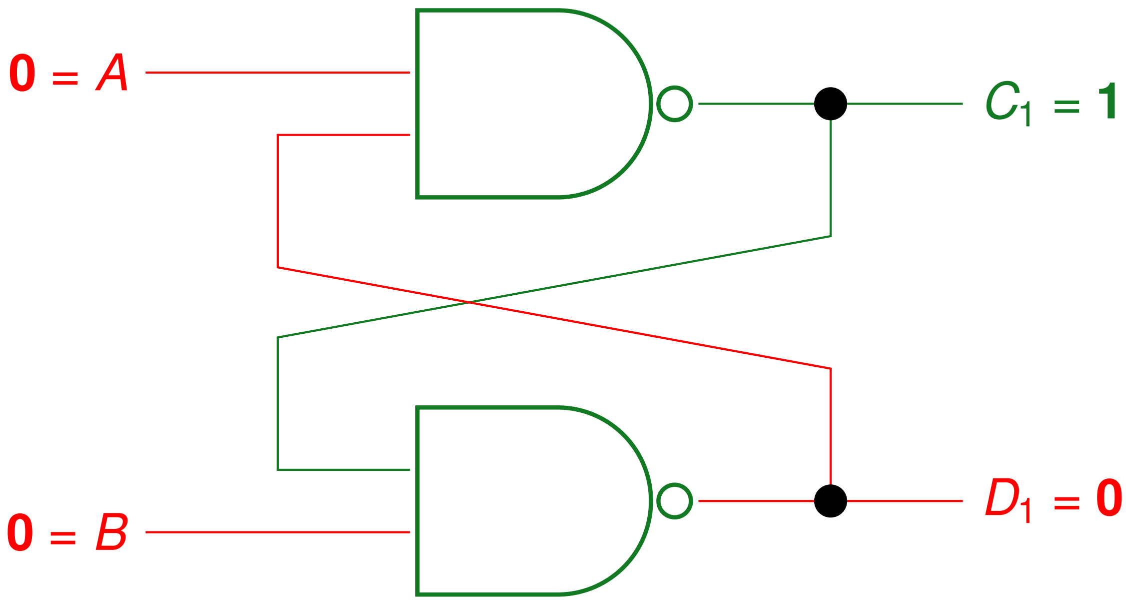
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ **(0, 0)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

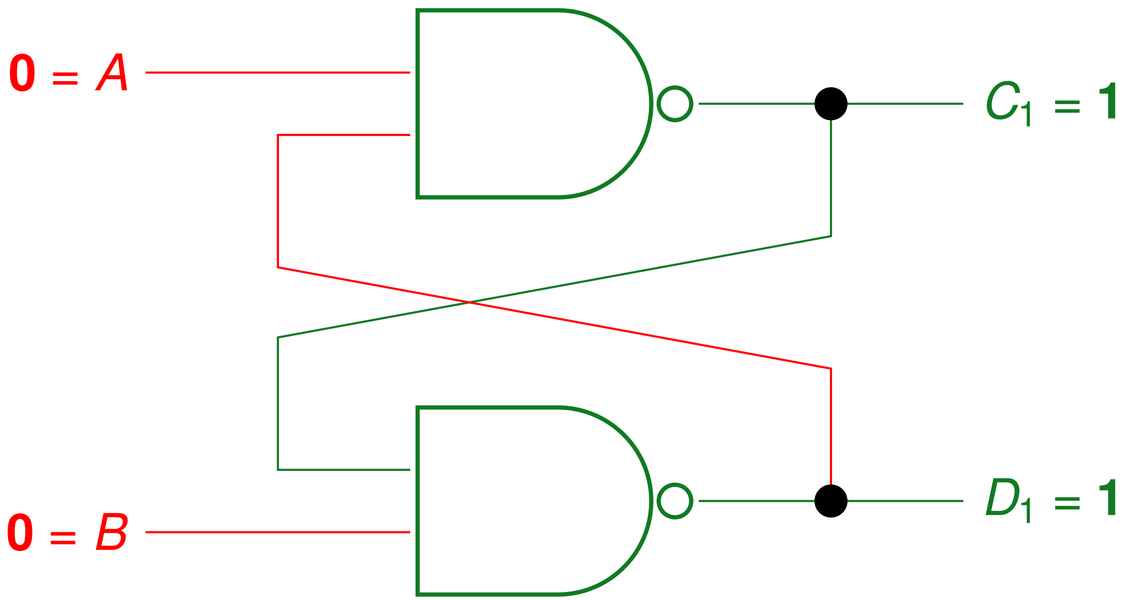
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ **(0, 0)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

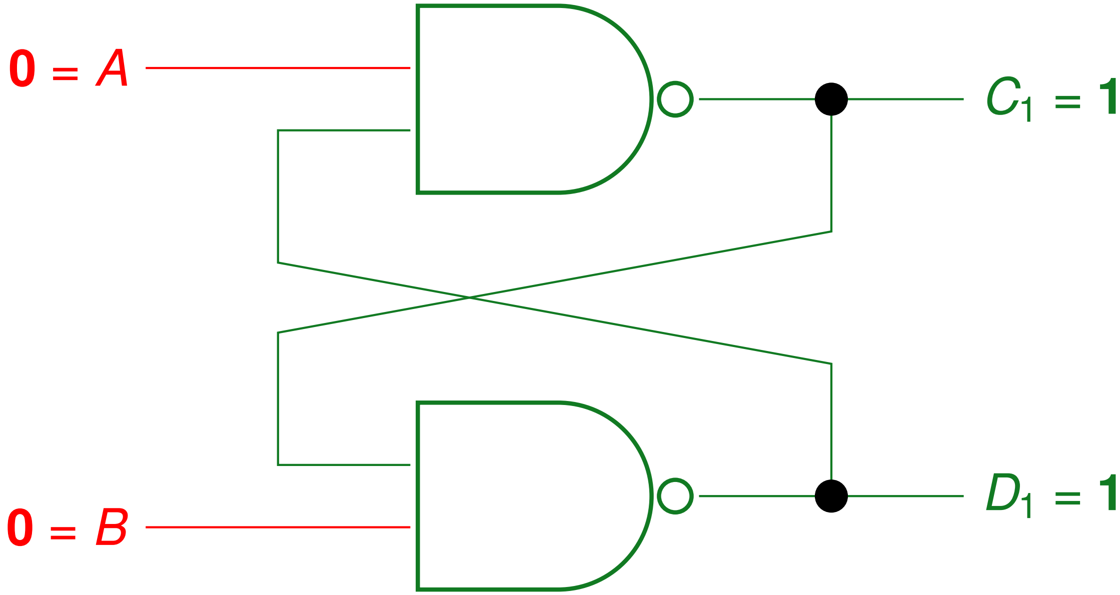
Anliegender Wert (**fett** hervorgehoben): $(0,1) \mapsto \mathbf{(0, 0)}$



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

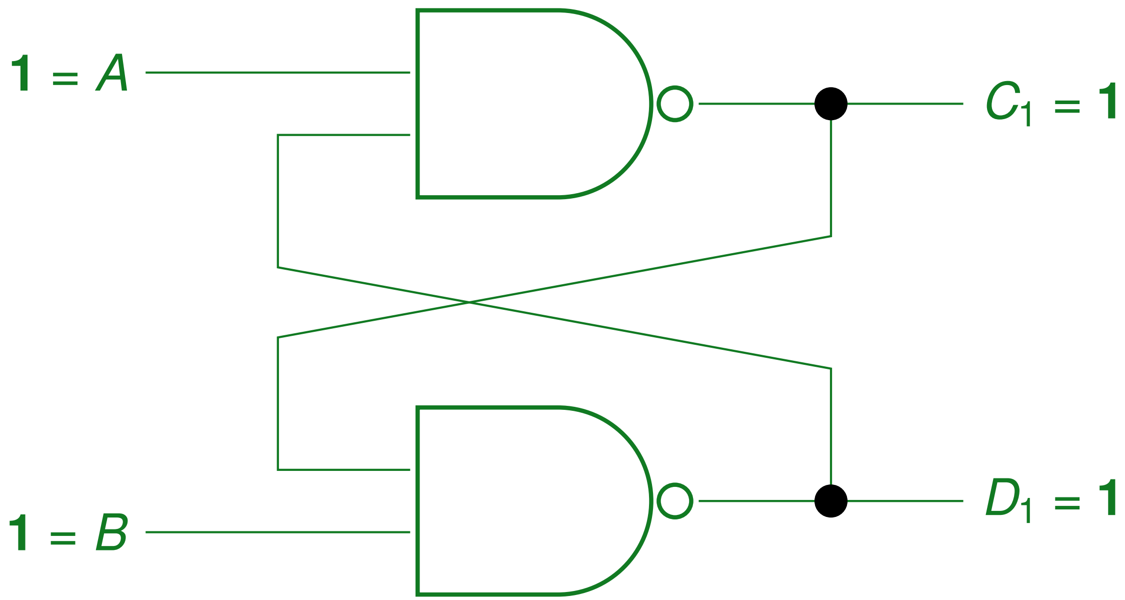
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ **(0, 0)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

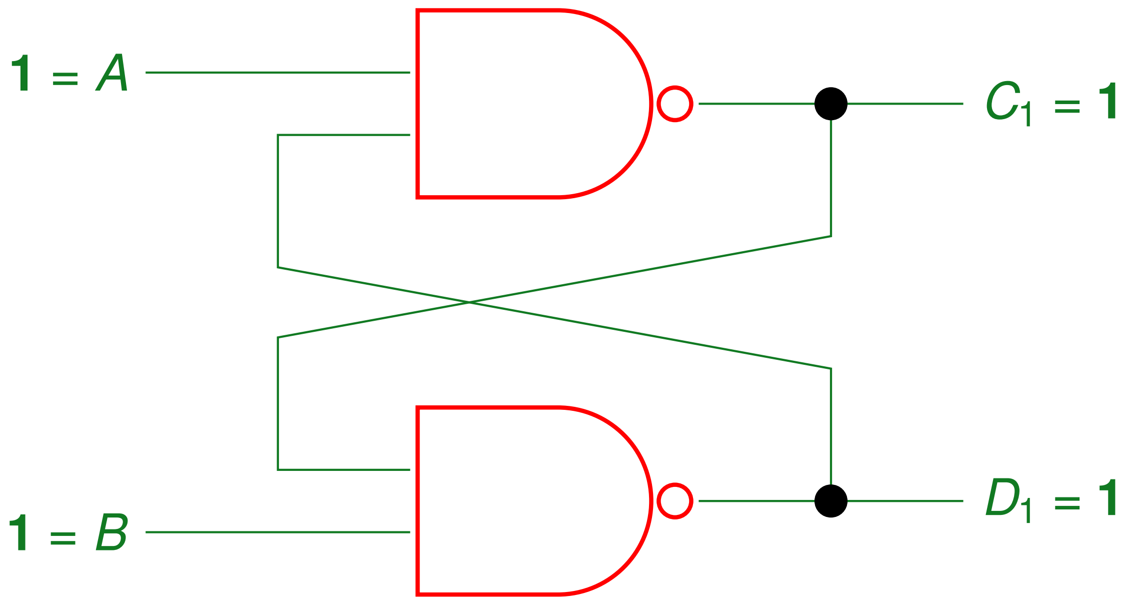
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ (0, 0) $\mapsto$ **(1, 1)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

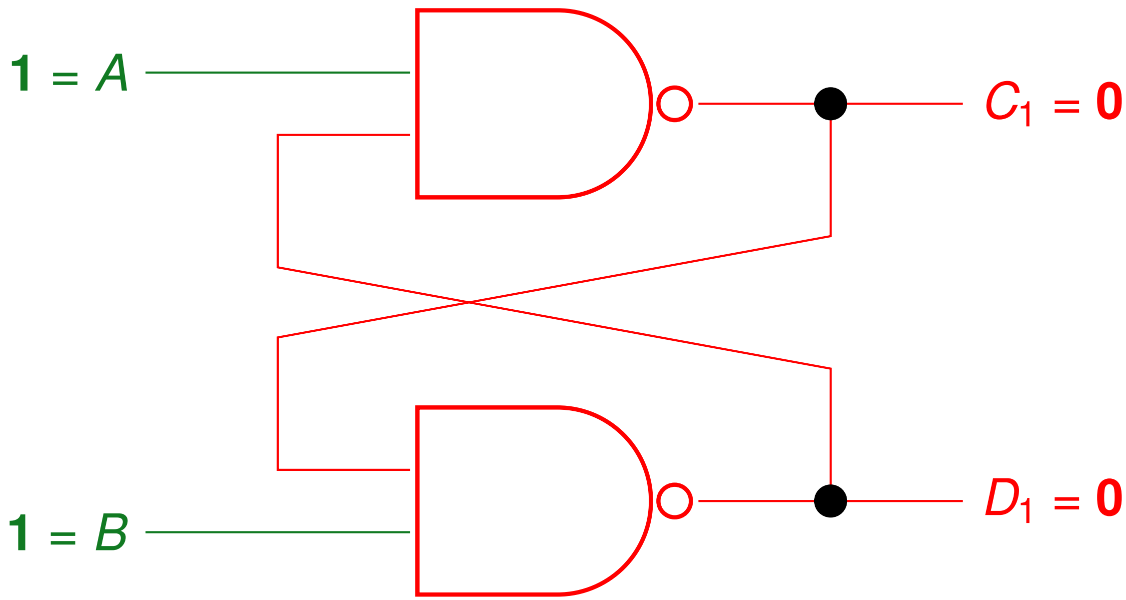
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ (0, 0) $\mapsto$ **(1, 1)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

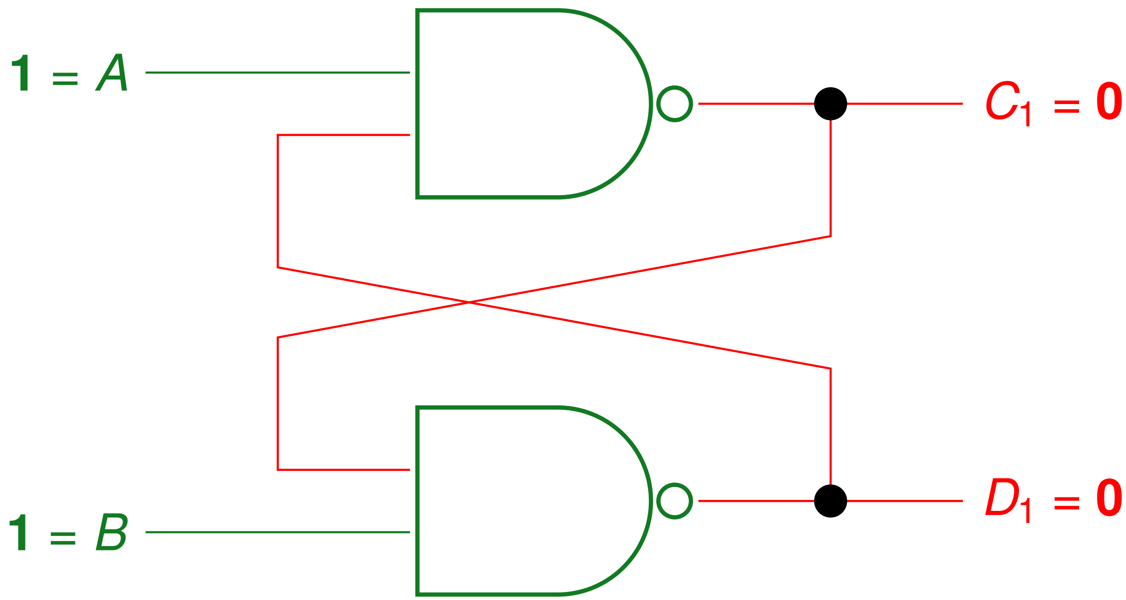
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ (0, 0) $\mapsto$ **(1, 1)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

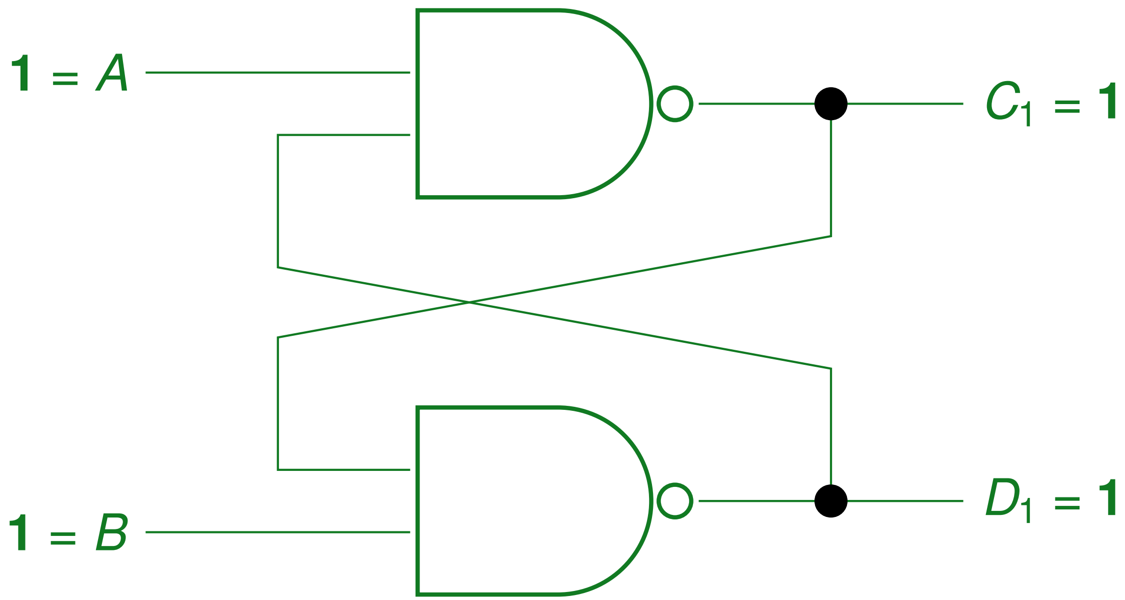
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ (0, 0) $\mapsto$ **(1, 1)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

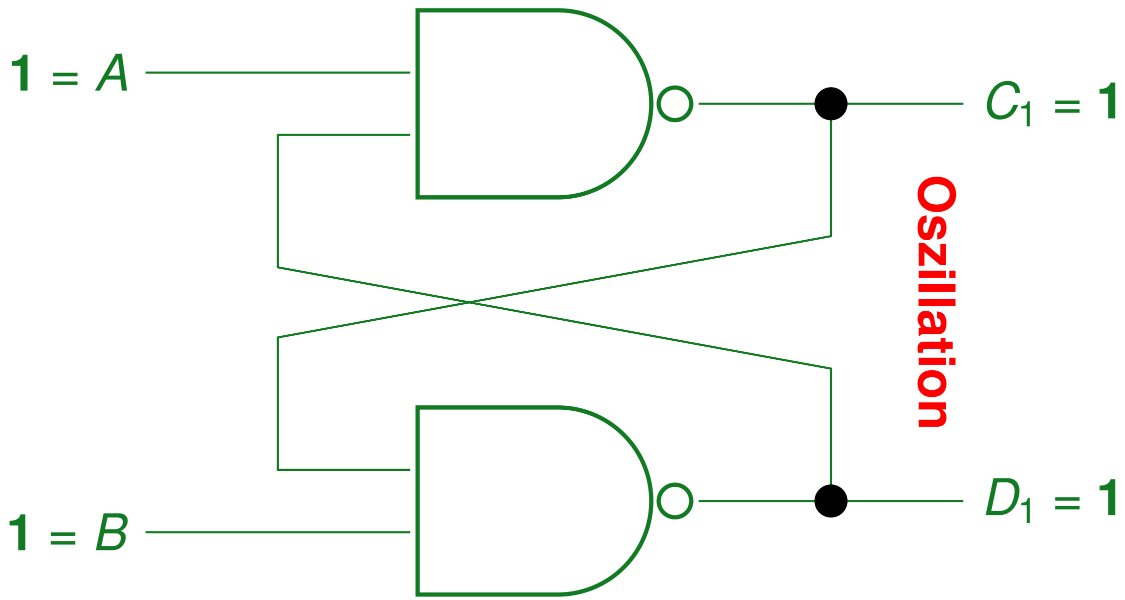
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ (0, 0) $\mapsto$ **(1, 1)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

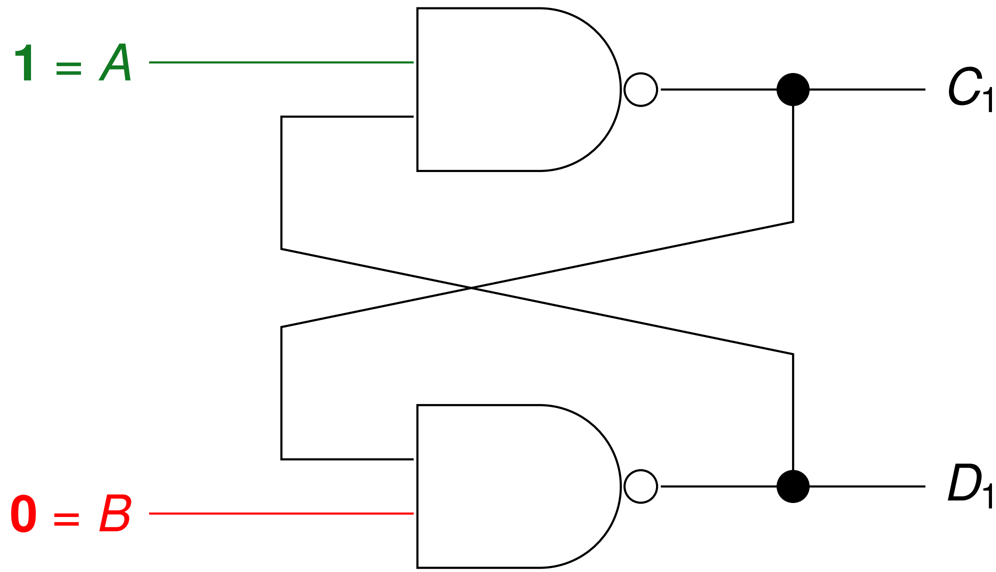
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ (0, 0) $\mapsto$ **(1, 1)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

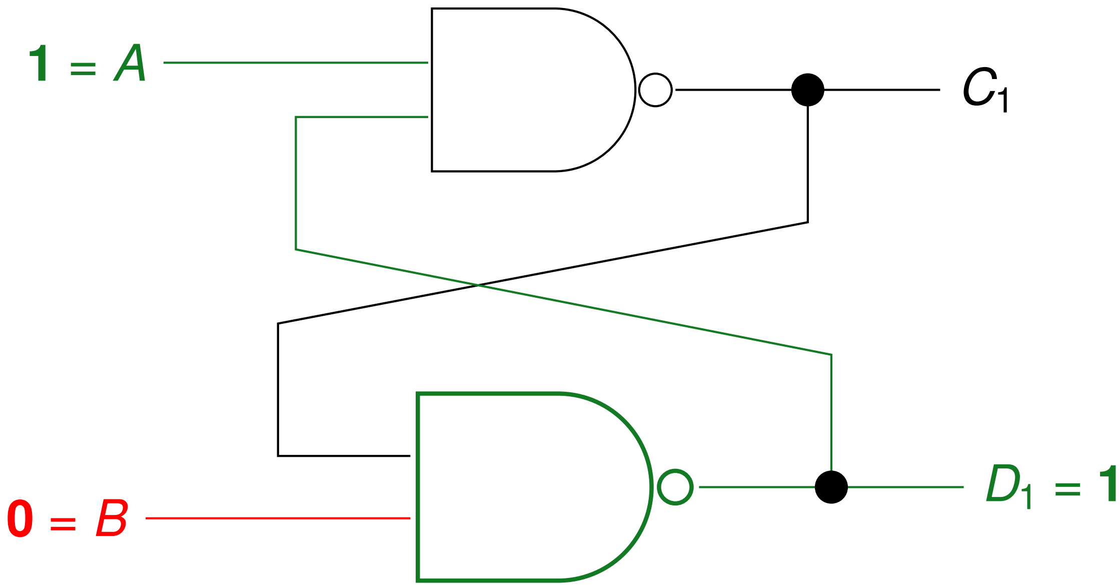
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (0, 0) $\mapsto$ (1, 1) $\mapsto$ **(1, 0)**



Hat ein NAND mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein NAND **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

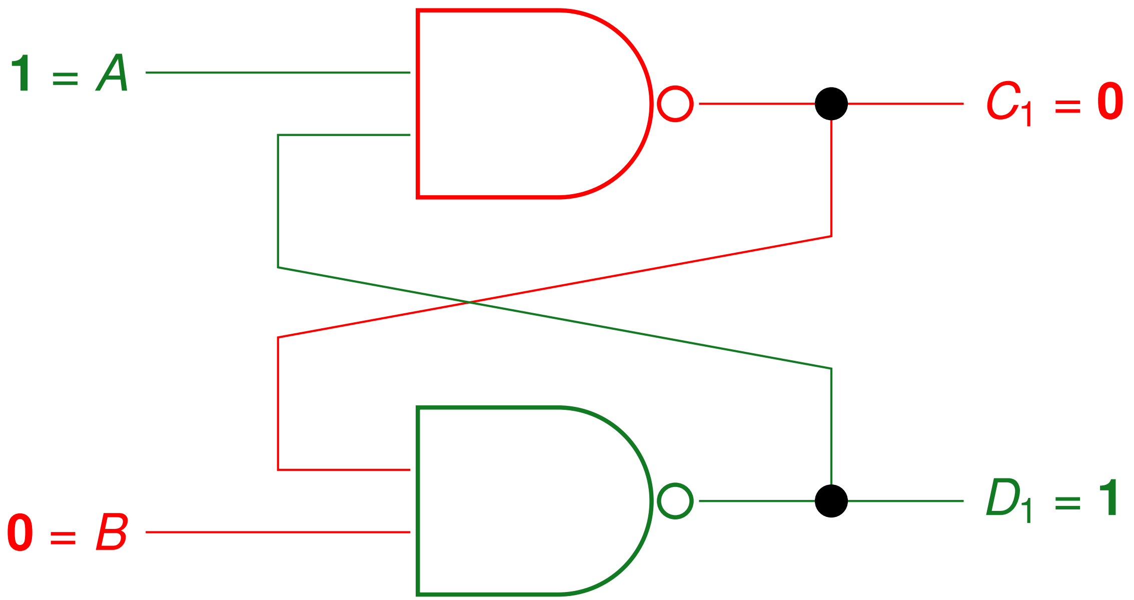
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (0, 0) $\mapsto$ (1, 1) $\mapsto$ **(1, 0)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

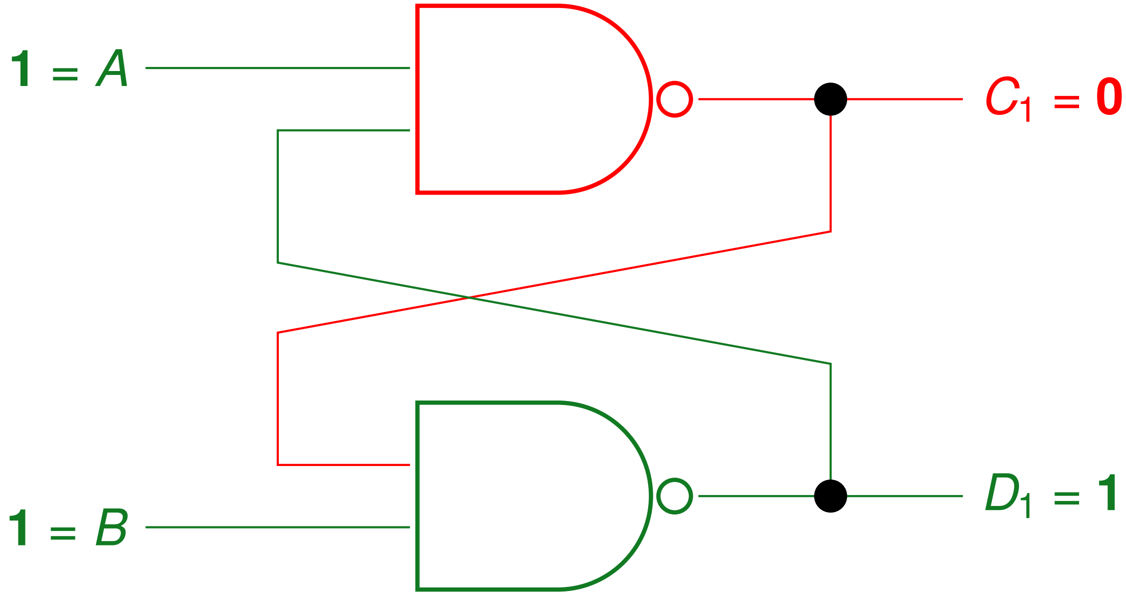
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (0, 0) $\mapsto$ (1, 1) $\mapsto$ **(1, 0)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

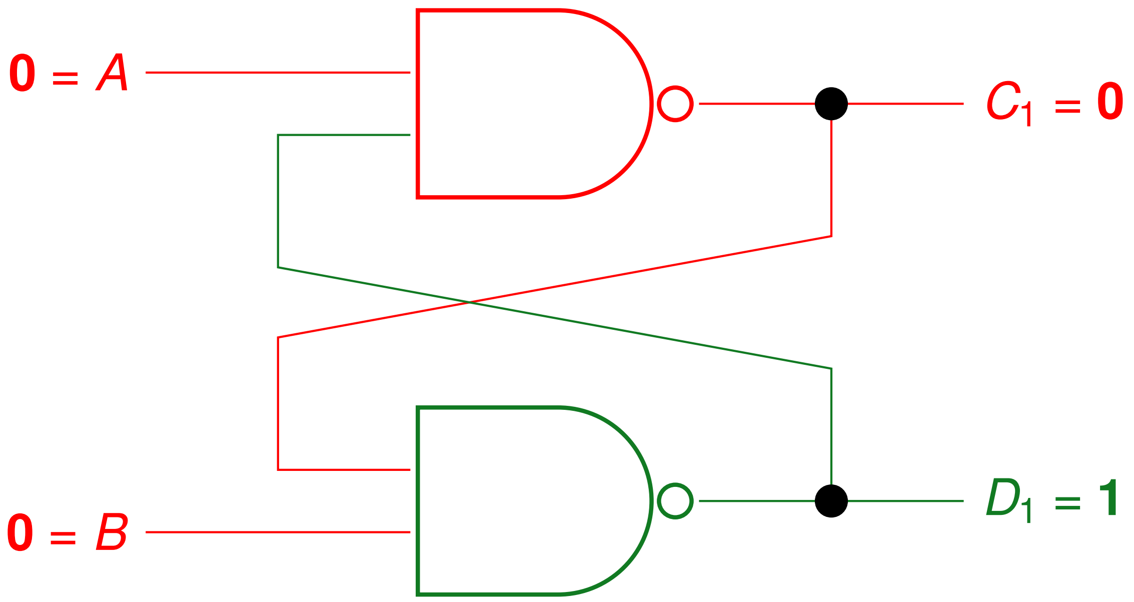
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 1) $\mapsto$ (1, 0) $\mapsto$ **(1, 1)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

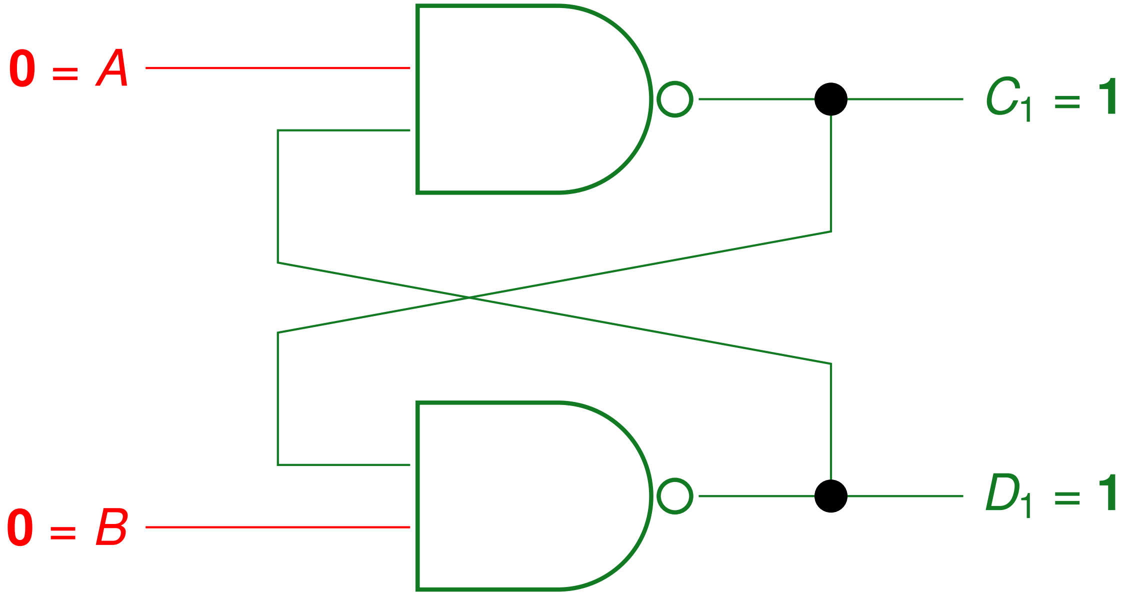
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 0) $\mapsto$ (1, 1) $\mapsto$ **(0, 0)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

Aufgabe 4 – Latches und Flipflops

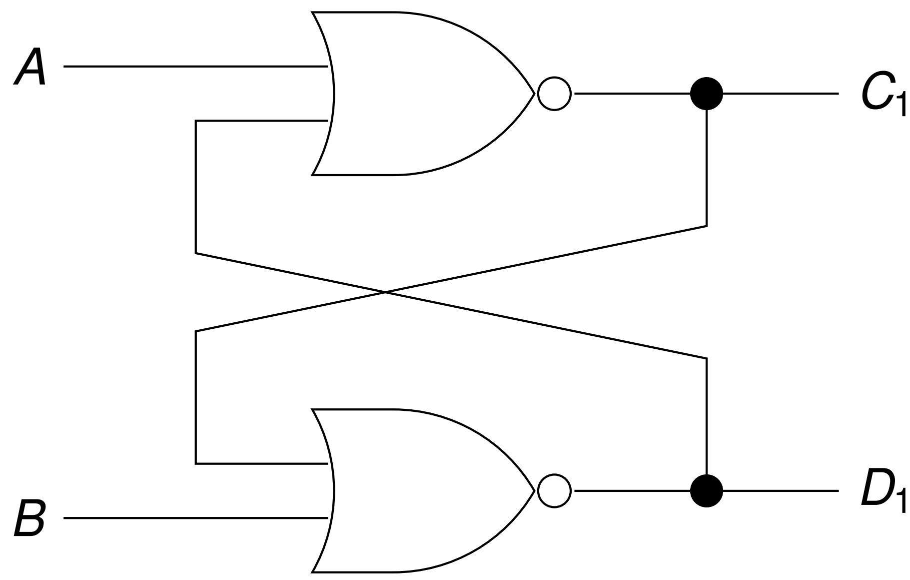
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 0) $\mapsto$ (1, 1) $\mapsto$ **(0, 0)**



Hat ein `NAND` mindestens **eine** 0 im Eingang, so ist der Ausgang 1.
Hat ein `NAND` **keine** 0 im Eingang (sprich: nur Einsen), so ist der Ausgang 0.

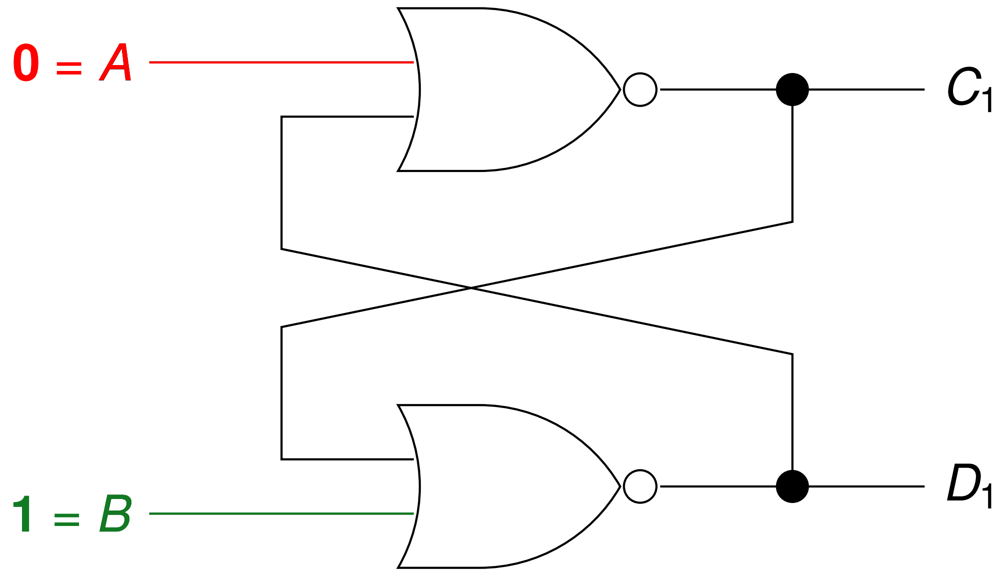
Aufgabe 4 – Latches und Flipflops

Anliegender Wert (**fett** hervorgehoben):



Aufgabe 4 – Latches und Flipflops

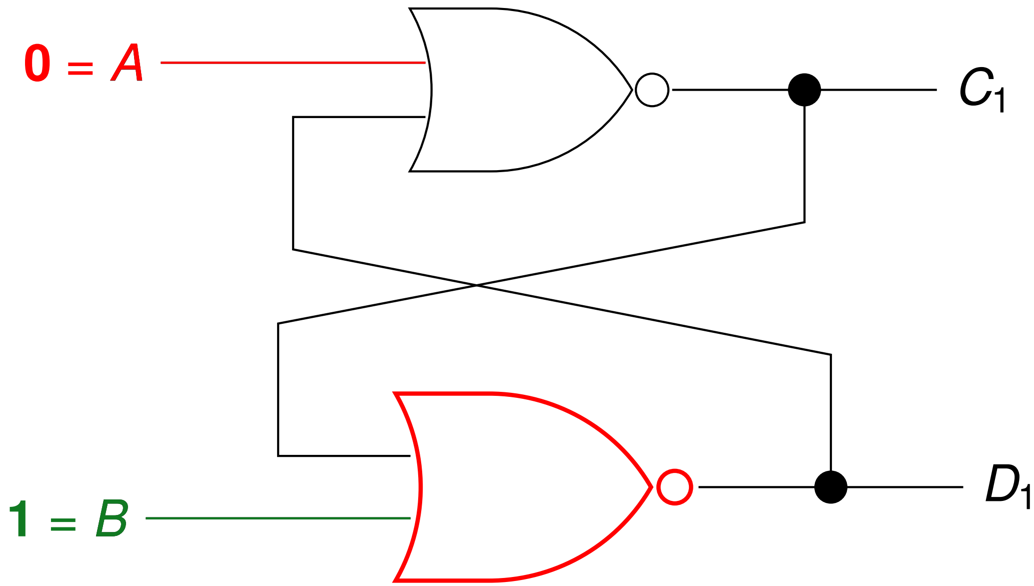
Anliegender Wert (**fett** hervorgehoben): **(0,1)**



Hat ein NOR mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein NOR **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

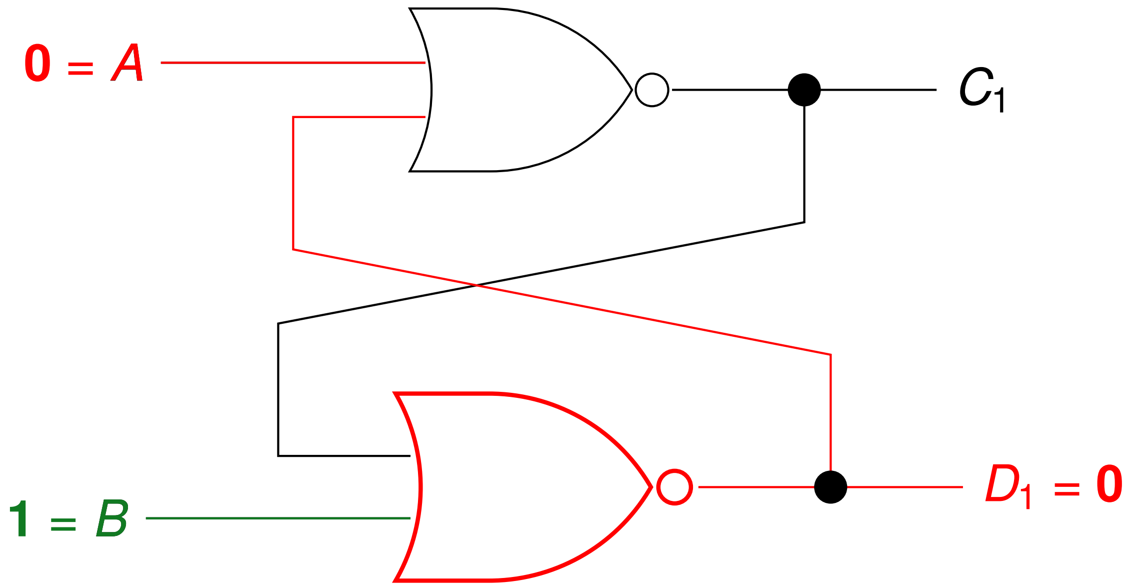
Anliegender Wert (**fett** hervorgehoben): **(0,1)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

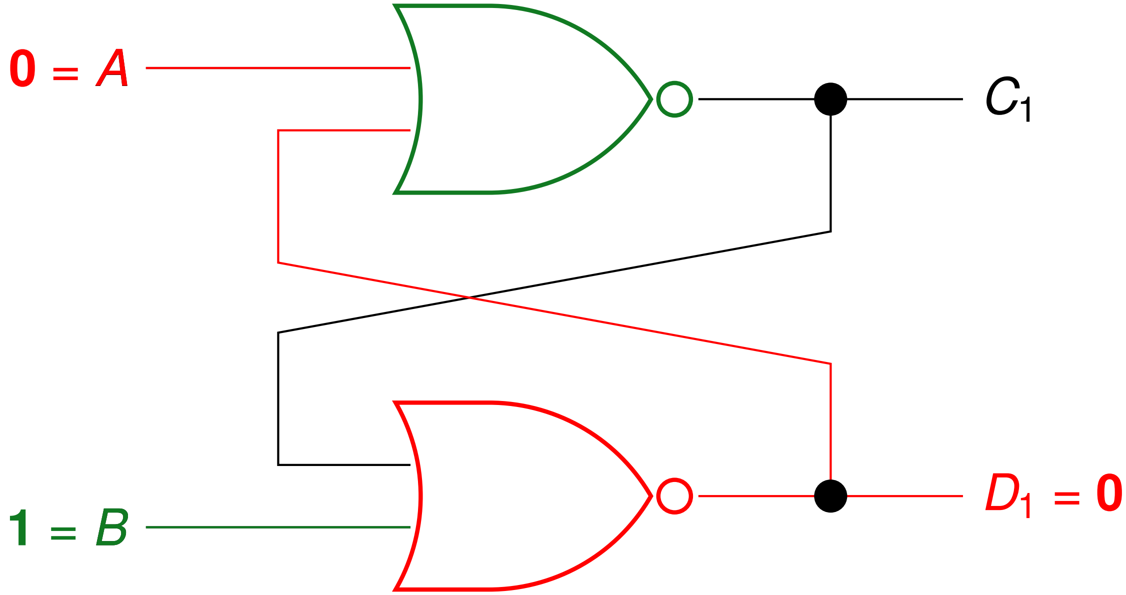
Anliegender Wert (**fett** hervorgehoben): **(0,1)**



Hat ein NOR mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein NOR **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

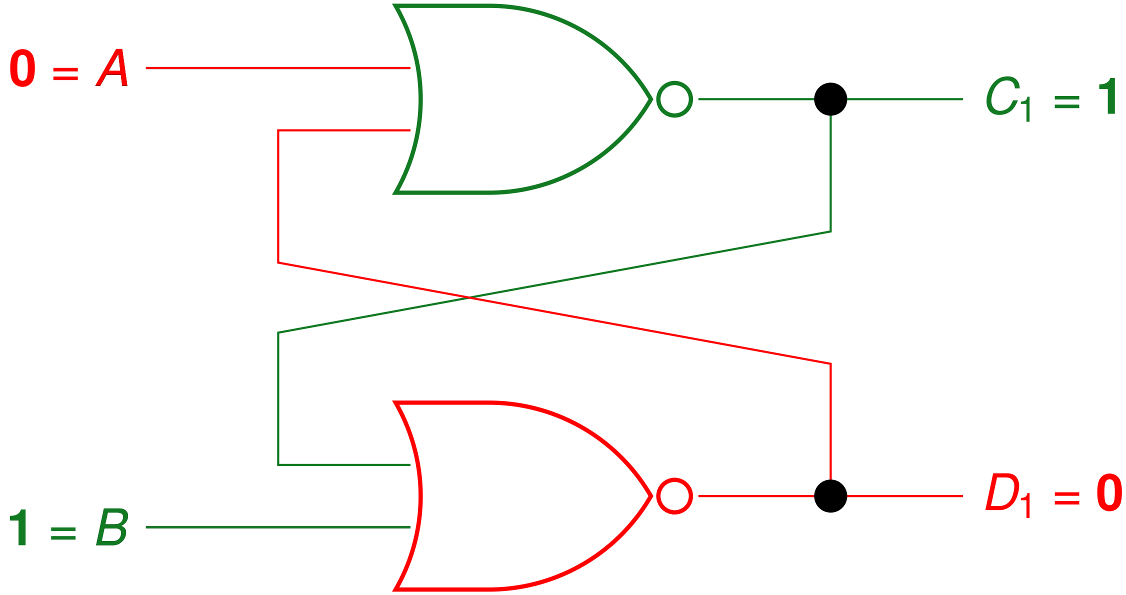
Anliegender Wert (**fett** hervorgehoben): **(0,1)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

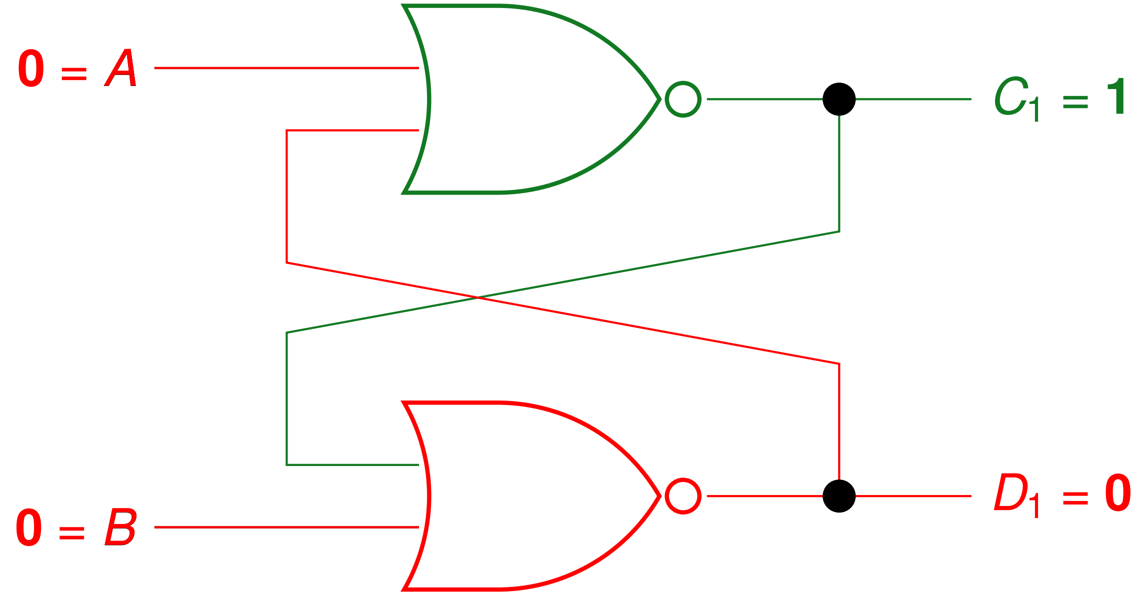
Anliegender Wert (**fett** hervorgehoben): **(0,1)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

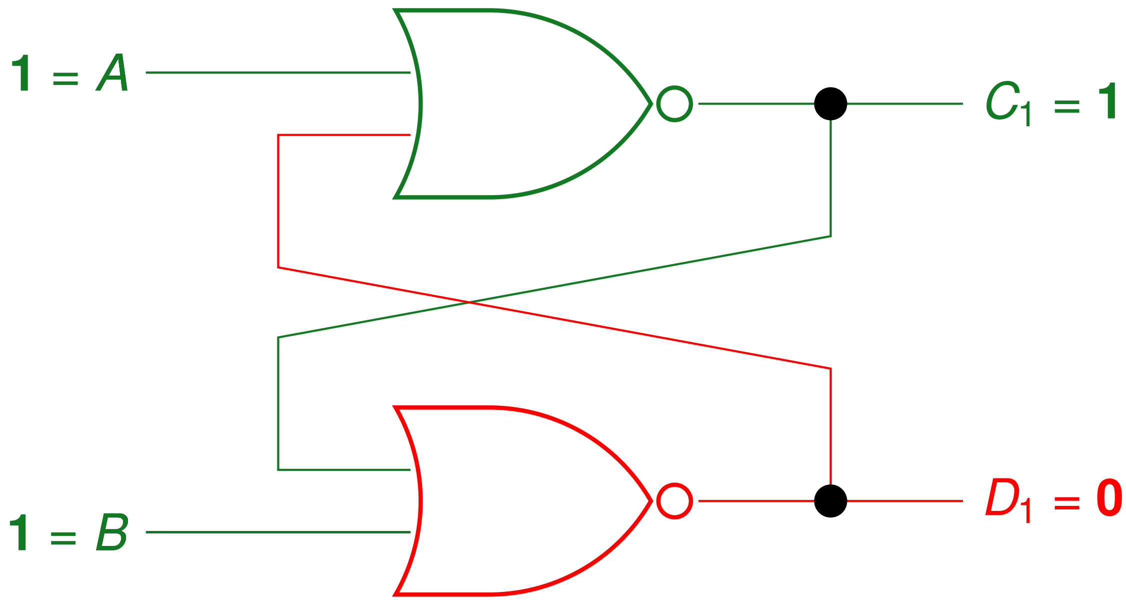
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ **(0, 0)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

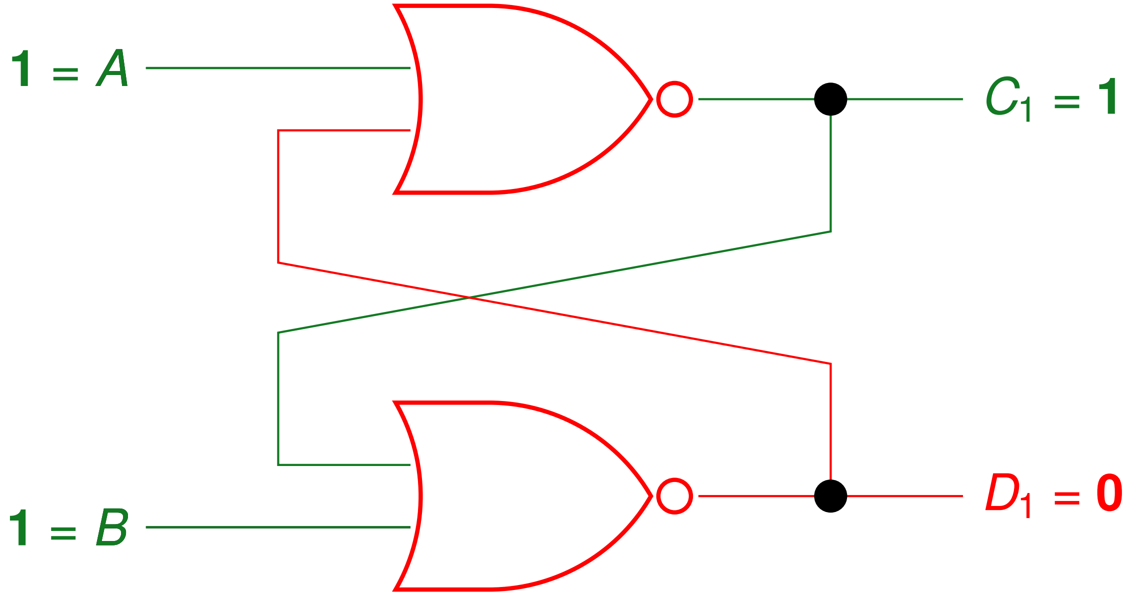
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ (0, 0) $\mapsto$ **(1, 1)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

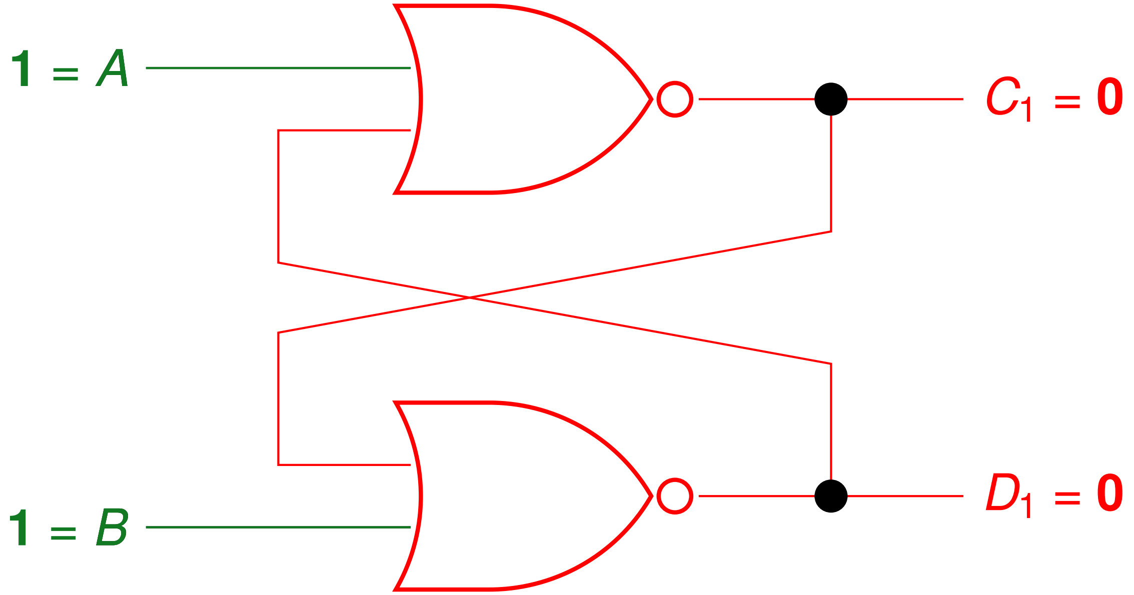
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ (0, 0) $\mapsto$ **(1, 1)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

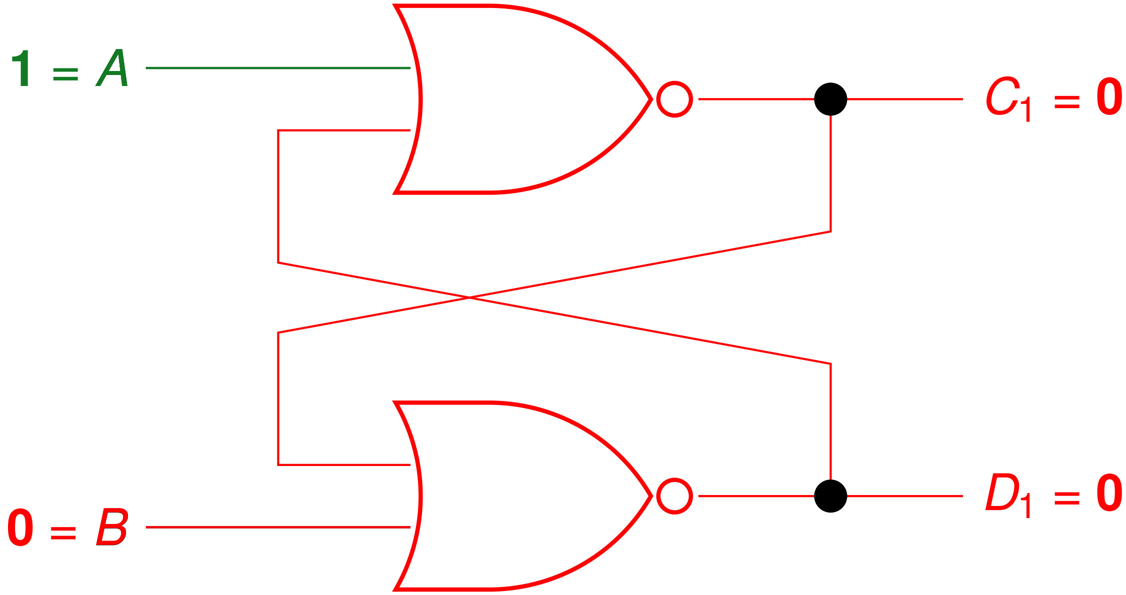
Anliegender Wert (**fett** hervorgehoben): (0,1) $\mapsto$ (0, 0) $\mapsto$ **(1, 1)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

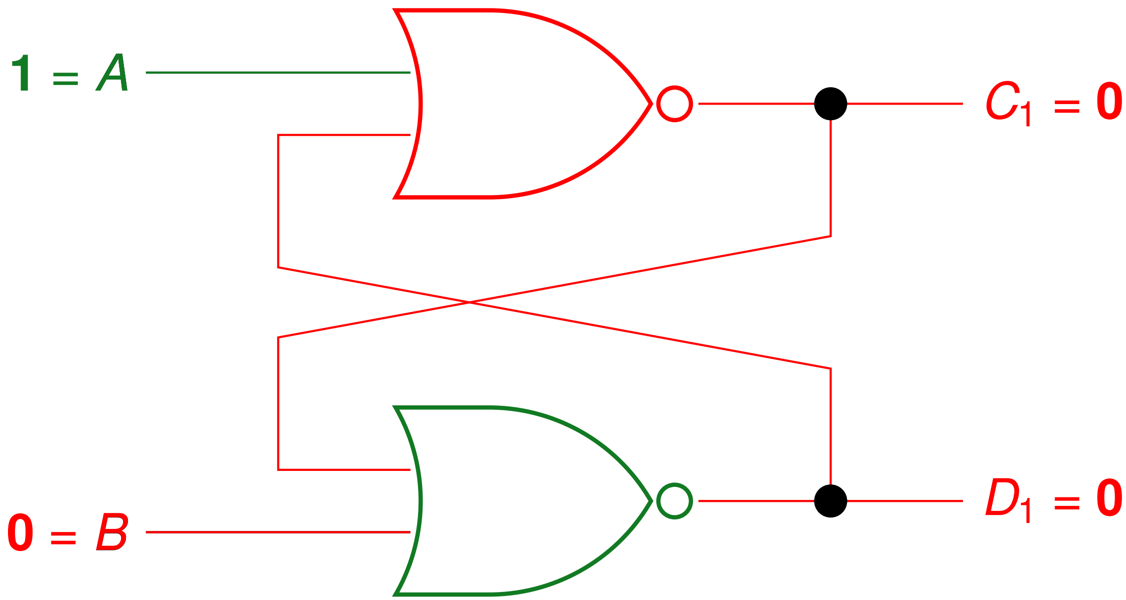
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (0, 0) $\mapsto$ (1, 1) $\mapsto$ **(1, 0)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

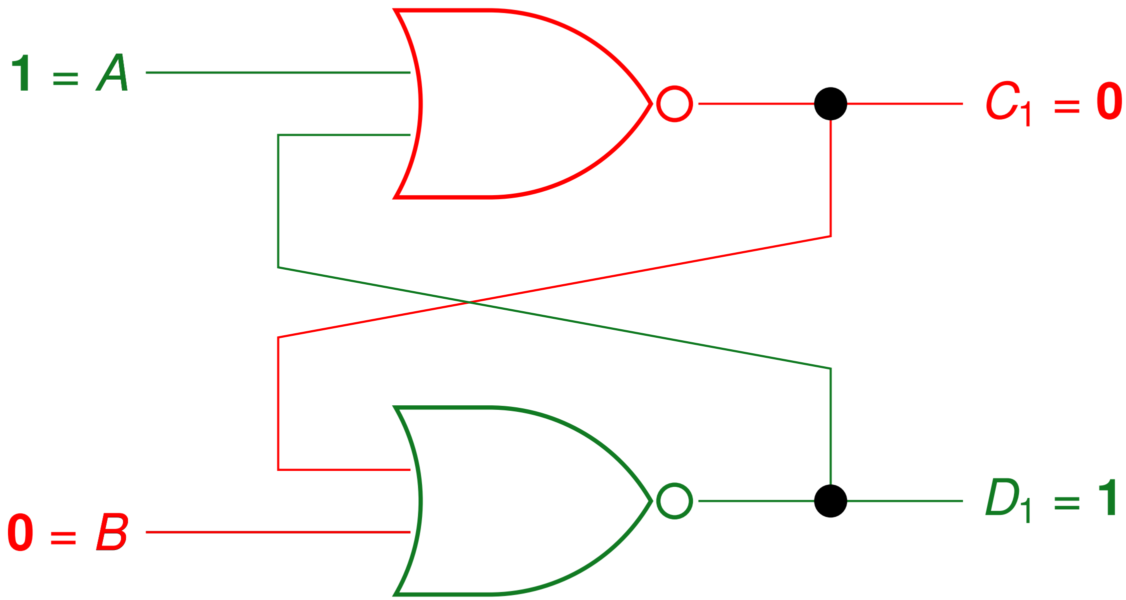
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (0, 0) $\mapsto$ (1, 1) $\mapsto$ **(1, 0)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

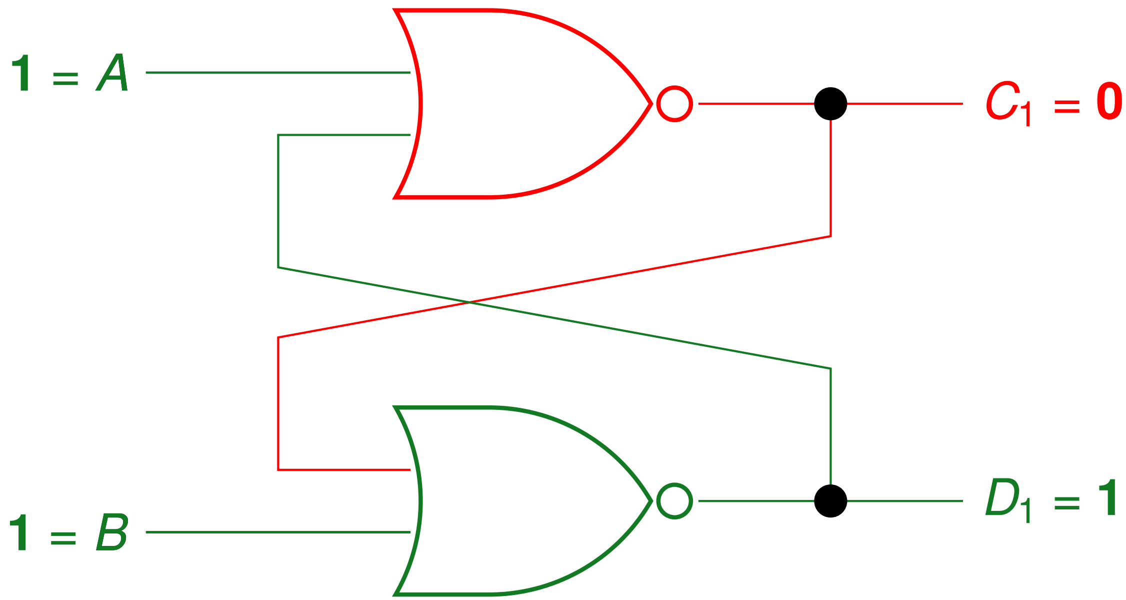
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (0, 0) $\mapsto$ (1, 1) $\mapsto$ **(1, 0)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

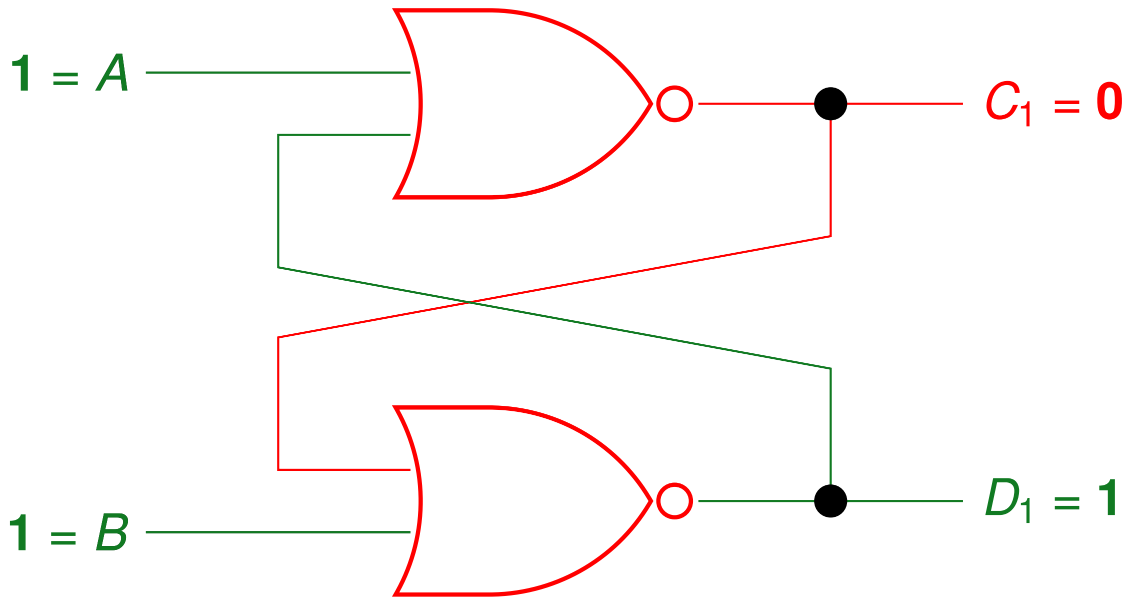
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 1) $\mapsto$ (1, 0) $\mapsto$ **(1, 1)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

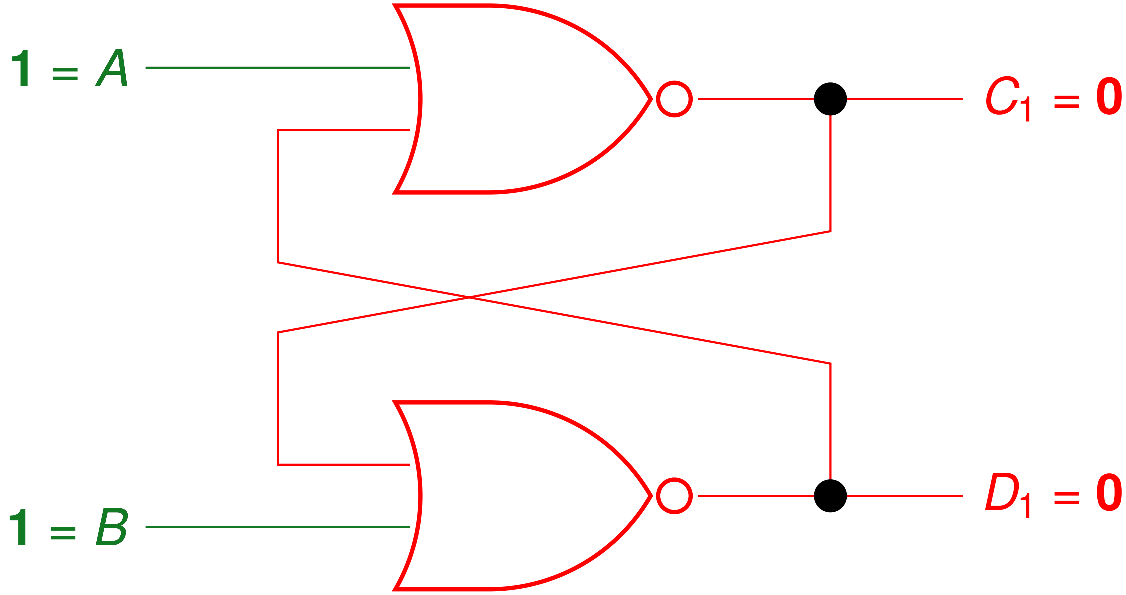
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 1) $\mapsto$ (1, 0) $\mapsto$ **(1, 1)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

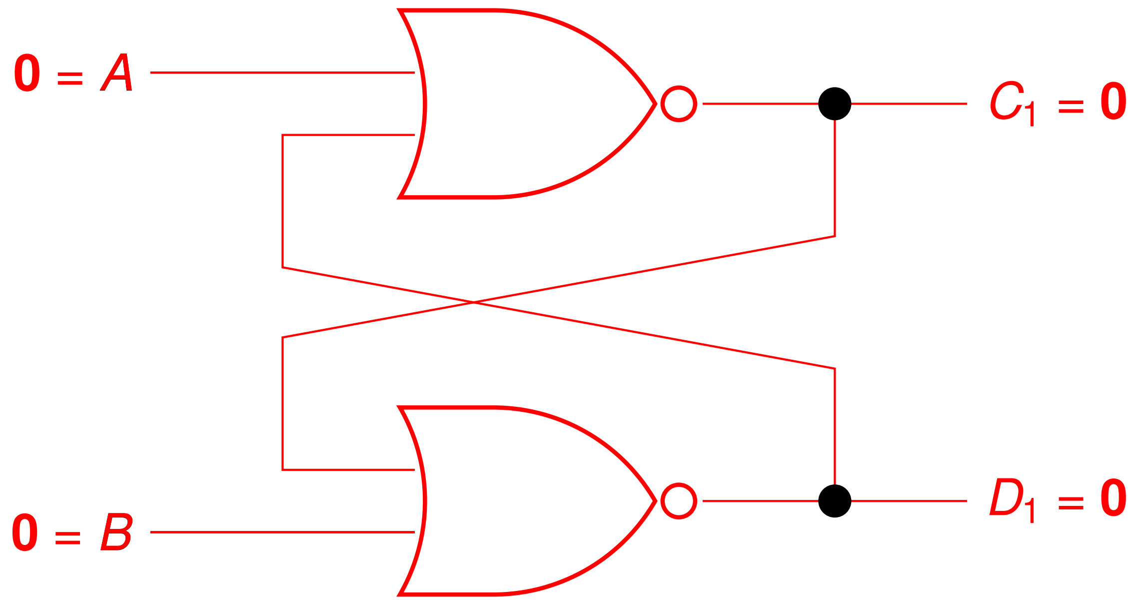
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 1) $\mapsto$ (1, 0) $\mapsto$ **(1, 1)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

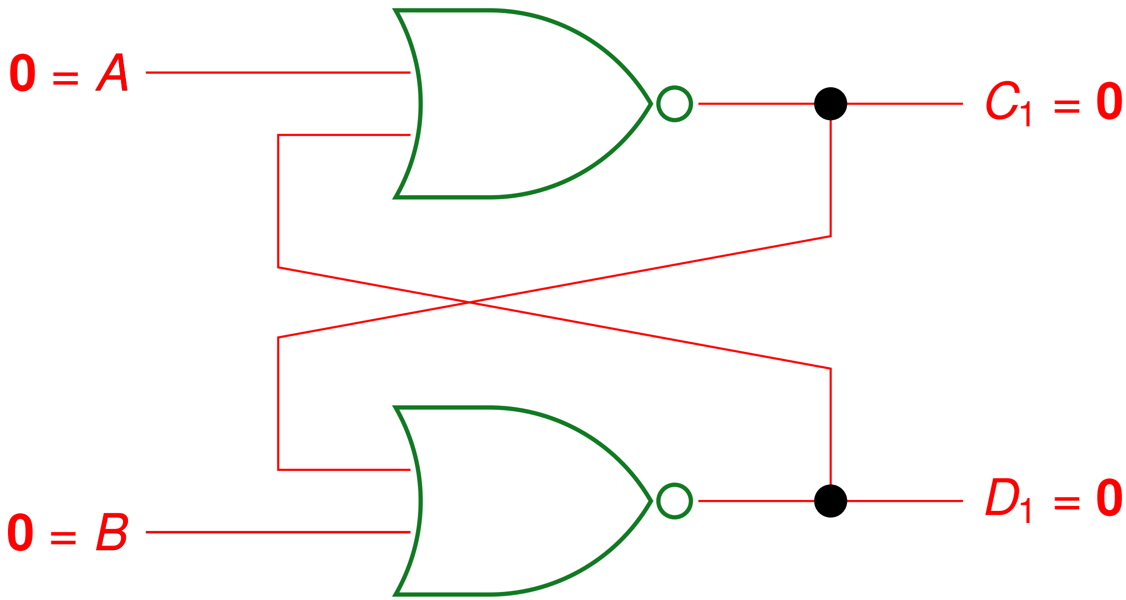
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 0) $\mapsto$ (1, 1) $\mapsto$ **(0, 0)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

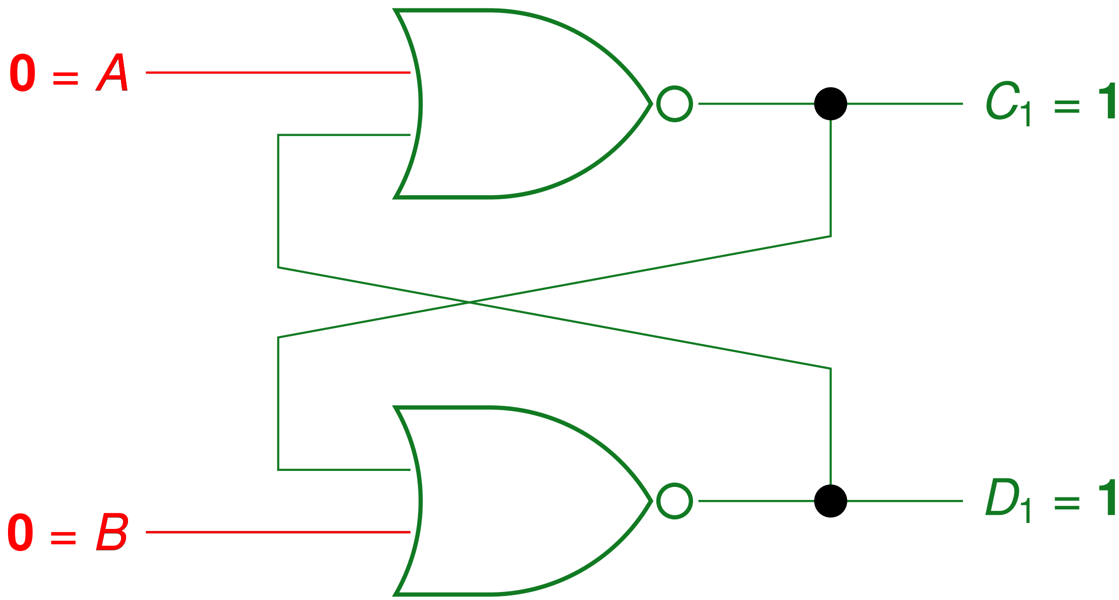
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 0) $\mapsto$ (1, 1) $\mapsto$ **(0, 0)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

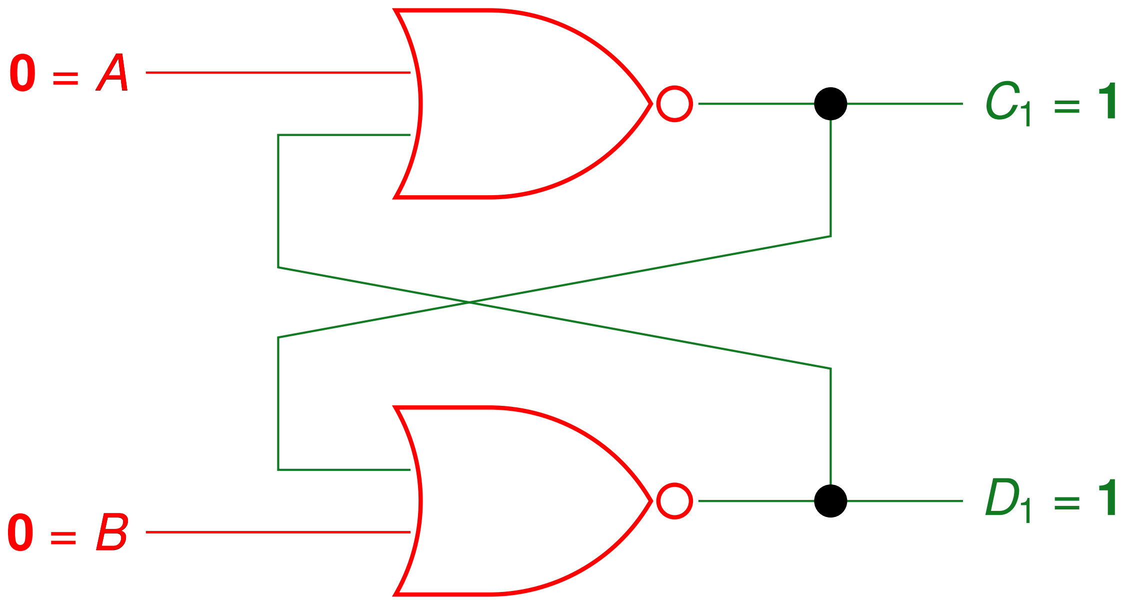
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 0) $\mapsto$ (1, 1) $\mapsto$ **(0, 0)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

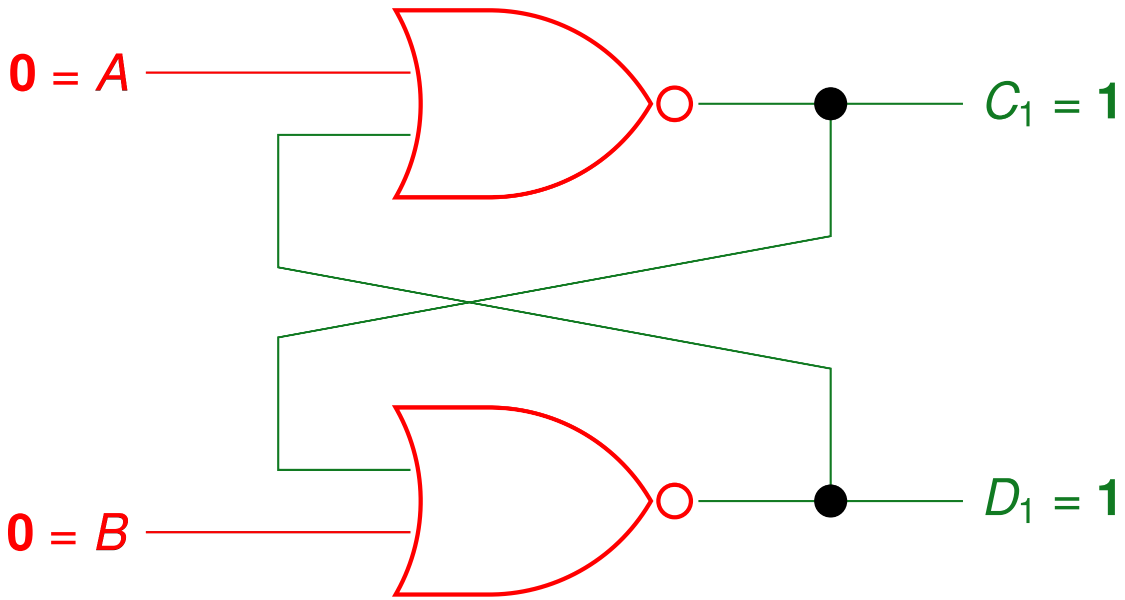
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 0) $\mapsto$ (1, 1) $\mapsto$ **(0, 0)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

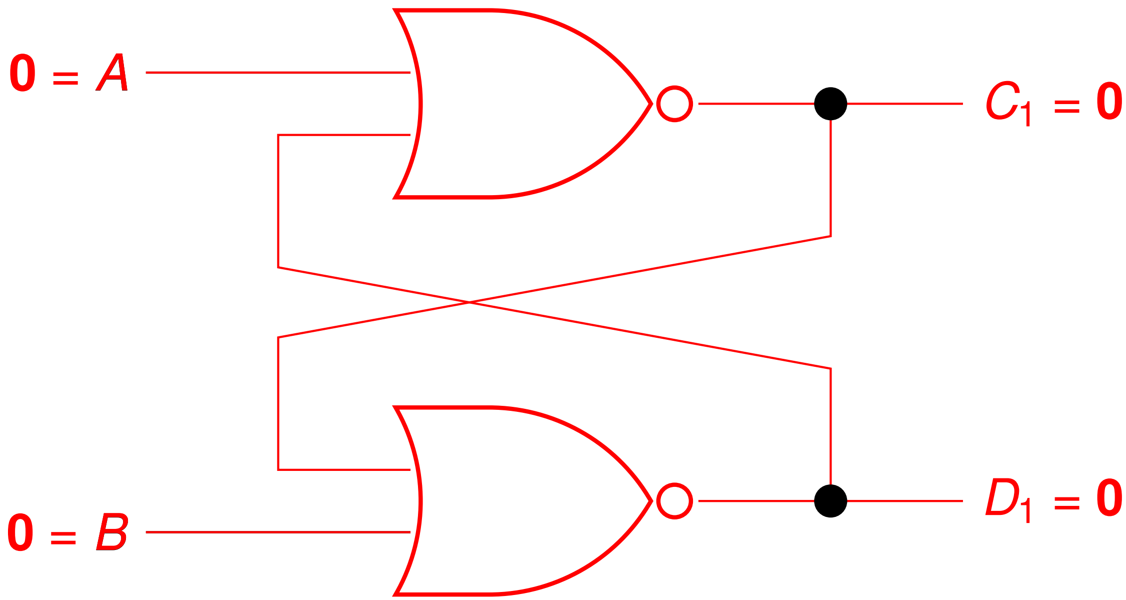
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 0) $\mapsto$ (1, 1) $\mapsto$ **(0, 0)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

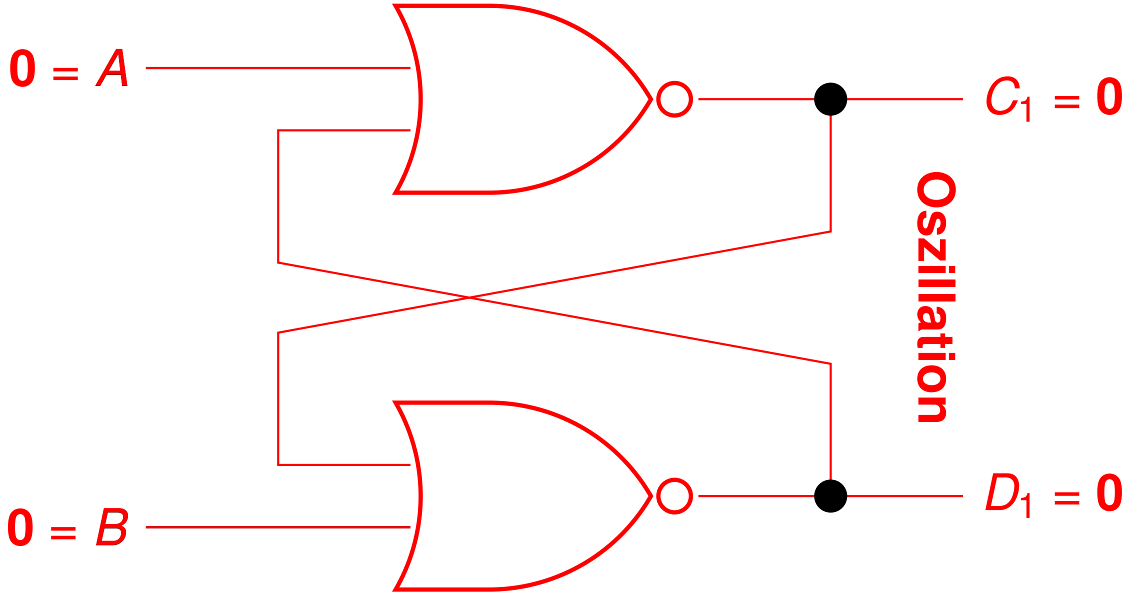
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 0) $\mapsto$ (1, 1) $\mapsto$ **(0, 0)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

Aufgabe 4 – Latches und Flipflops

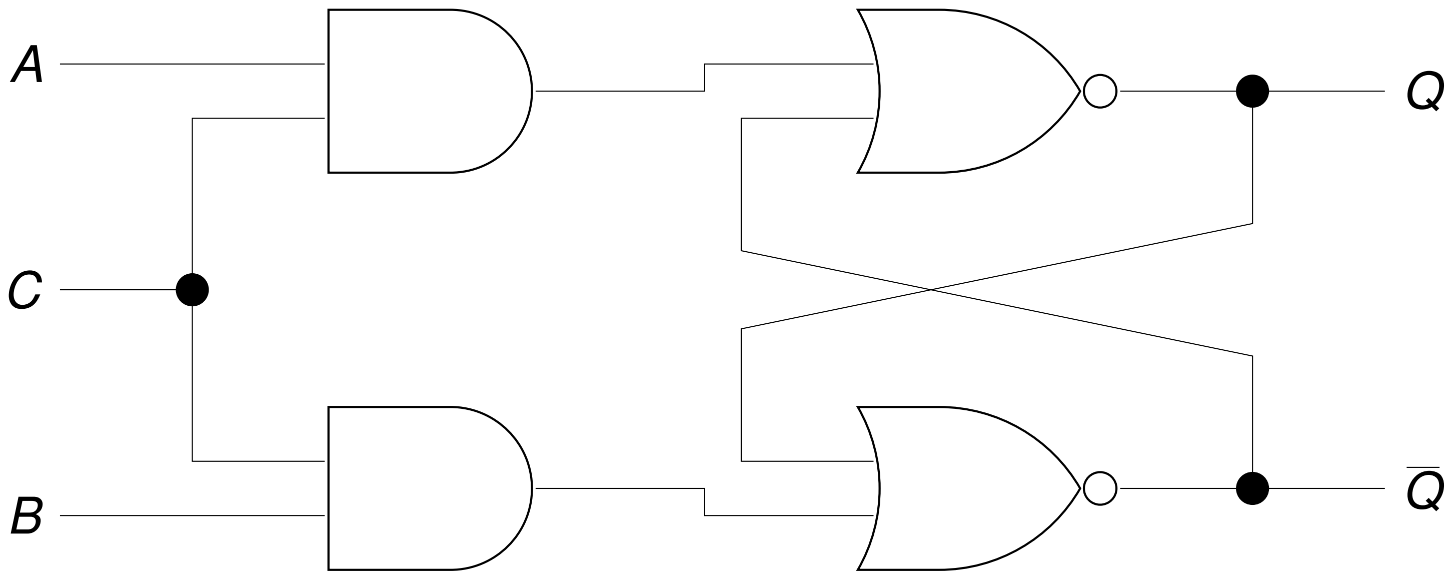
Anliegender Wert (**fett** hervorgehoben): ... $\mapsto$ (1, 0) $\mapsto$ (1, 1) $\mapsto$ **(0, 0)**



Hat ein `NOR` mindestens **eine** 1 im Eingang, so ist der Ausgang 0.
Hat ein `NOR` **keine** 1 im Eingang (sprich: nur Nullen), so ist der Ausgang 1.

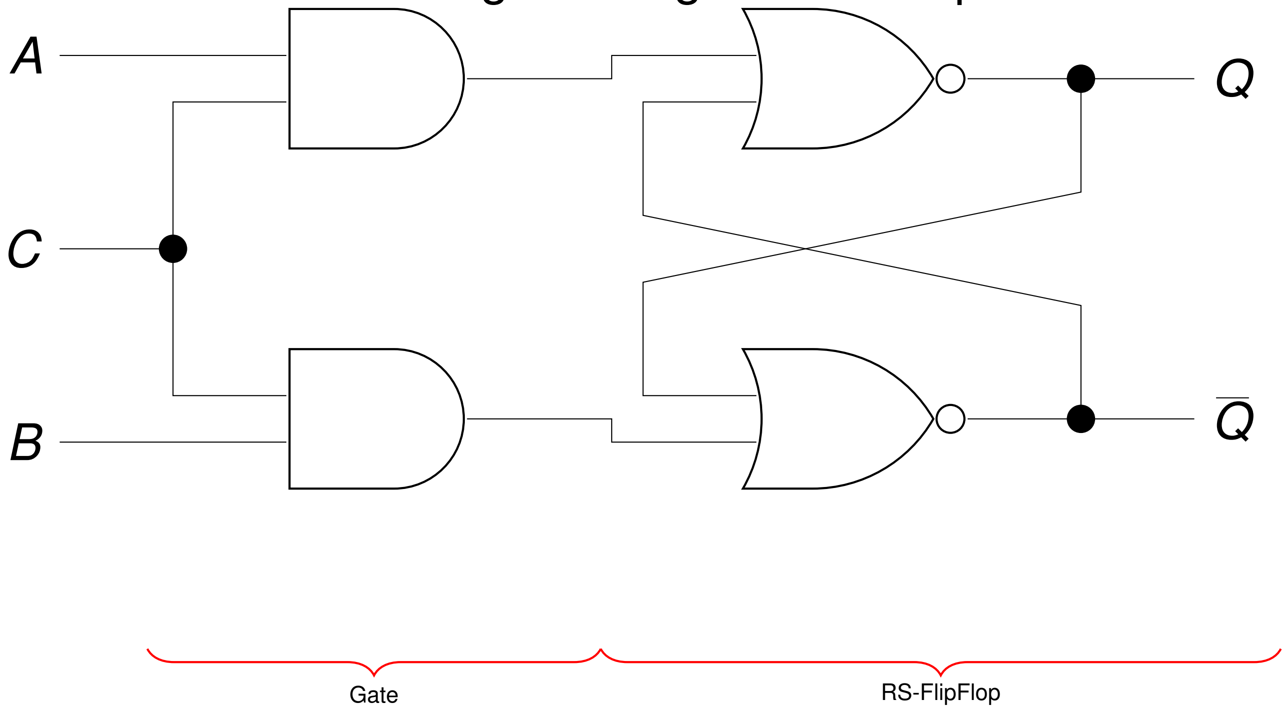
Aufgabe 4 – Latches und Flipflops

b) Sei C ein Taktsignal. Wie bezeichnet man dann das hier abgebildete Speicherelement?



Bestimmung

Gesucht: Bezeichnung des abgebildeten Speicherelements.

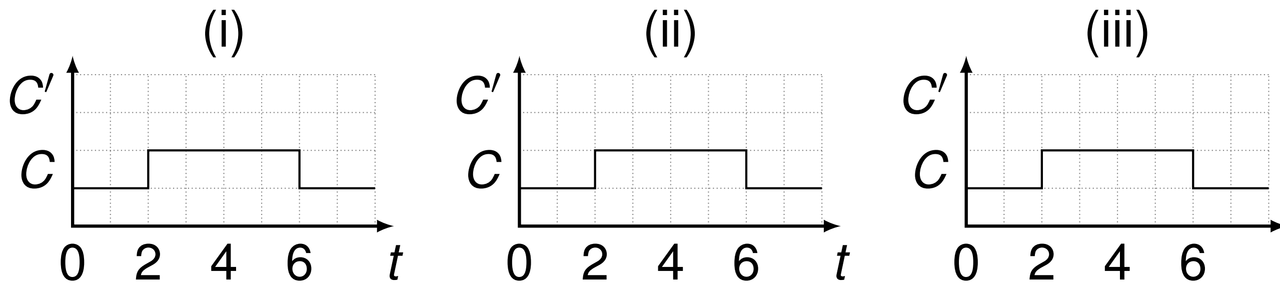


- C = 0: Gate deaktiviert, da an den Ausgängen immer (0,0) anliegt
- C = 1: A und B werden weitergeleitet (UND wegdenken)

Das Flipflop ist also ein pegelgesteuertes RS-Flipflop.
Außerdem ist es wieder Active-high.

Aufgabe 4 – Latches und Flipflops

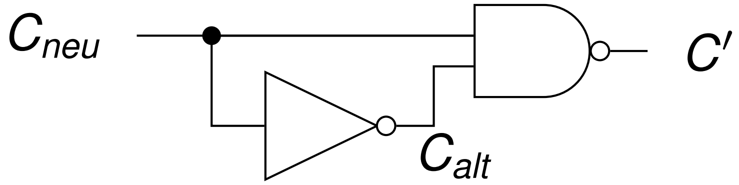
b) Dieses soll so erweitert werden, dass es (i) nur bei der steigenden, (ii) der fallenden und (iii) bei jeder Flanke von C auf die Eingänge A und B reagiert. Geben Sie jeweils das Schaltnetz der Flankenerkennung $C \mapsto C'$ an und vervollständigen Sie die folgenden Wellenformdiagramme:



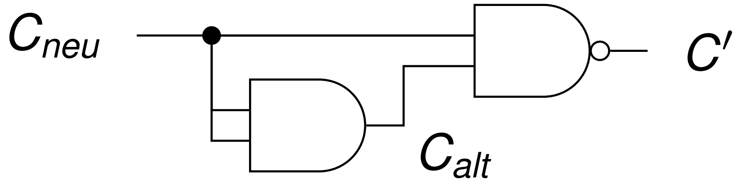
Verzögerungsstrategien

Es gibt verschiedene Verzögerungsarten, um ein C_{alt} zu bekommen.

Inverter:
Man erhält ein negiertes C_{alt}



UND/ODER:
Man erhält ein nicht negiertes C_{alt}



Sonstige Gatter:

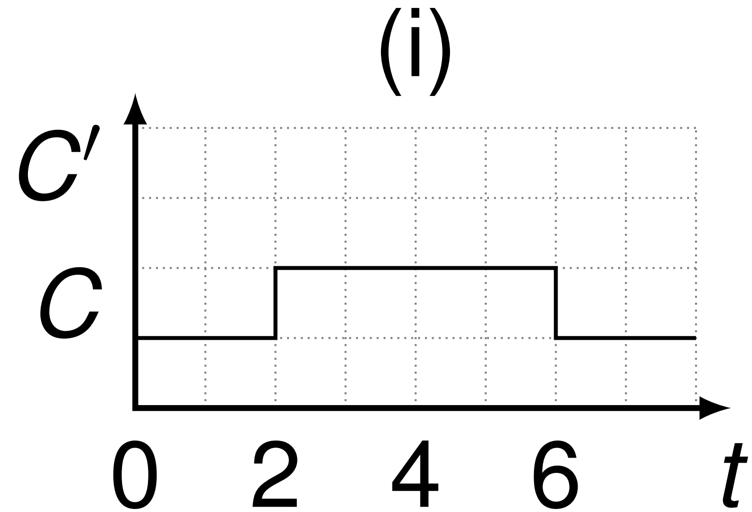
- NAND/NOR genutzt, da geringere Latenz als UND/ODER
- Andere Gatter möglich

Steigende Flanke I

Wir wissen, dass die Verzögerungszeit $\tau = 1$ beträgt.
Wir wollen folgendes Verhalten für jeden Zeitschritt umsetzen:

C_{alt}	C_{neu}	C'
0	0	0
0	1	1 (steigend)
1	0	0 (fallend)
1	1	0

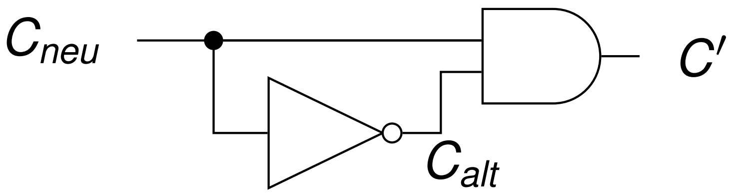
$$f(C_{alt}, C_{neu}) = \overline{C_{alt}} \cdot C_{neu}$$

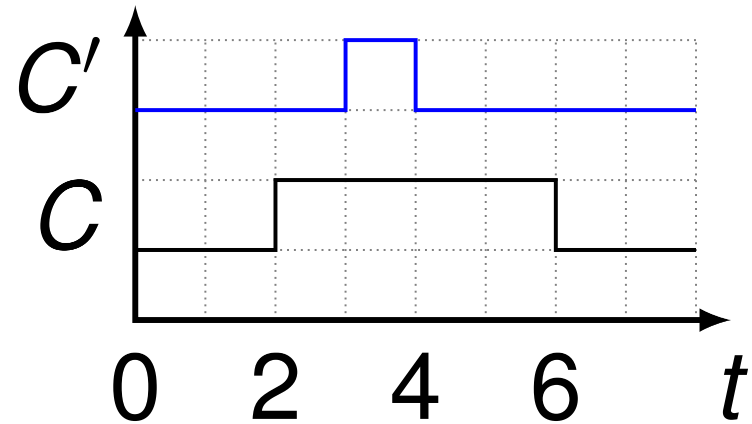


Steigende Flanke II

Wir wissen, dass die Verzögerungszeit $\tau = 1$ beträgt. Dadurch entsteht ein C_{alt} nach dem Inverter, da die obere Leitung keine Verzögerung besitzt.

$f(C_{alt}, C_{neu}) = \overline{C_{alt}} \cdot C_{neu}$





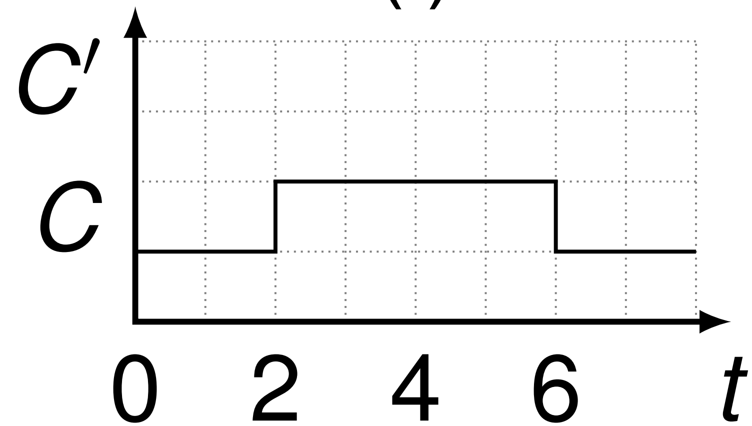
Das UND-Gatter verzögert das C‘ um eine Zeiteinheit.

Fallende Flanke I

Wir wissen, dass die Verzögerungszeit $\tau = 1$ beträgt.
Wir wollen folgendes Verhalten für jeden Zeitschritt umsetzen:

C_{alt}	C_{neu}	C'
0	0	0
0	1	0 (steigend)
1	0	1 (fallend)
1	1	0

(i)

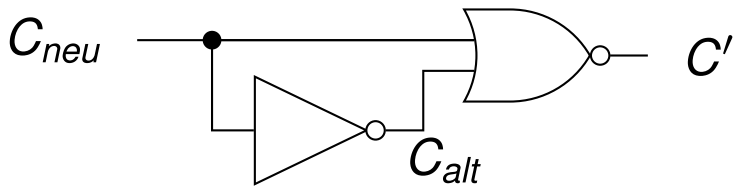


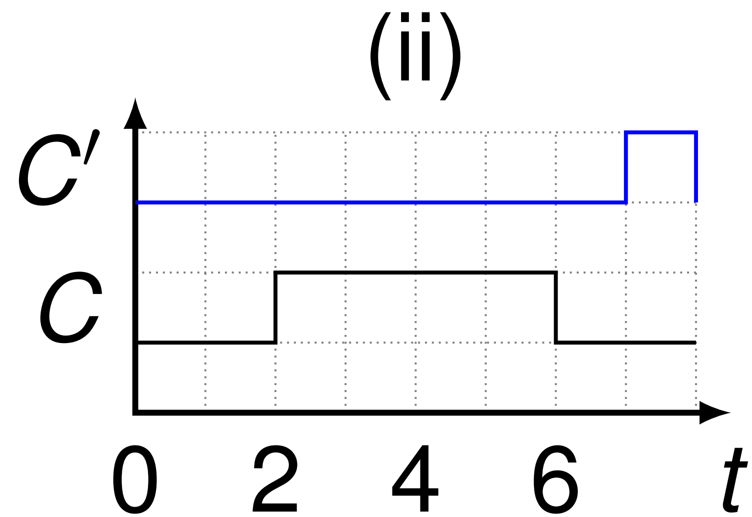
Wir bekommen durch einen Inverter nur ein $\overline{C_{alt}}$ deshalb formen wir um:
$f(C_{alt}, C_{neu}) = \overline{C_{neu}} \cdot C_{alt} = \overline{\overline{C_{alt}} \cdot C_{neu}}$

Fallende Flanke II

Wir wissen, dass die Verzögerungszeit $\tau = 1$ beträgt. Dadurch entsteht ein C_{alt} nach dem Inverter, da die obere Leitung keine Verzögerung besitzt.

$$f(C_{alt}, C_{neu}) = \overline{\overline{C_{alt}} \cdot C_{neu}}$$





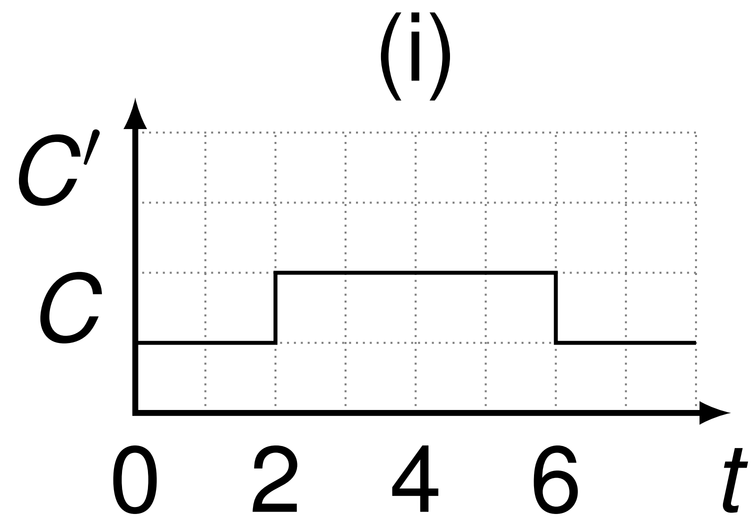
Das NOR-Gatter verzögert das C‘ um eine Zeiteinheit.

Steigende und Fallende Flanke I

Wir wissen, dass die Verzögerungszeit $\tau = 1$ beträgt.
Wir wollen folgendes Verhalten für jeden Zeitschritt umsetzen:

C_{alt}	C_{neu}	C'
0	0	0
0	1	1 (steigend)
1	0	1 (fallend)
1	1	0

$$f(C_{alt}, C_{neu}) = \overline{C_{neu}} \cdot C_{alt} + \overline{C_{alt}} \cdot C_{neu}$$

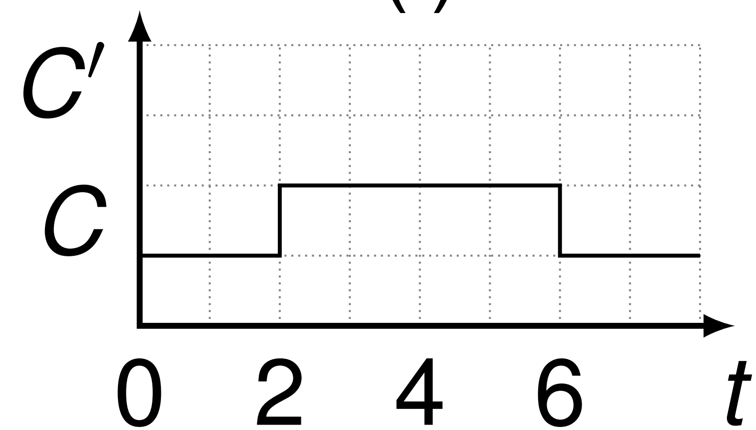


Steigende und Fallende Flanke I

Wir wissen, dass die Verzögerungszeit $\tau = 1$ beträgt.
Wir wollen folgendes Verhalten für jeden Zeitschritt umsetzen:

C_{alt}	C_{neu}	C'
0	0	0
0	1	1 (steigend)
1	0	1 (fallend)
1	1	0

(i)



$$f(C_{alt}, C_{neu}) = \overline{C_{neu}} \cdot C_{alt} + \overline{C_{alt}} \cdot C_{neu}$$
$$= (\overline{C_{neu}} + \overline{C_{alt}}) \cdot (C_{alt} + C_{neu})$$

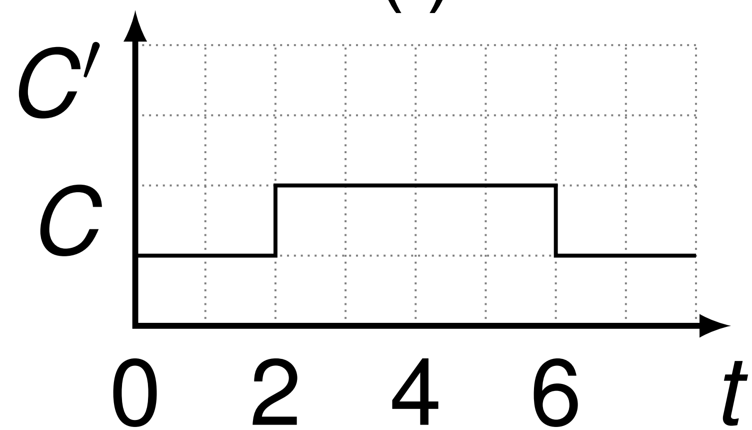
Steigende und Fallende Flanke I

Wir wissen, dass die Verzögerungszeit $\tau = 1$ beträgt.
Wir wollen folgendes Verhalten für jeden Zeitschritt umsetzen:

C_{alt}	C_{neu}	C'
0	0	0
0	1	1 (steigend)
1	0	1 (fallend)
1	1	0

$$f(C_{alt}, C_{neu}) = \overline{C_{neu}} \cdot C_{alt} + \overline{C_{alt}} \cdot C_{neu}$$
$$= (\overline{C_{neu}} + \overline{C_{alt}}) \cdot (C_{alt} + C_{neu})$$
$$= \overline{\overline{C_{alt}}\,\overline{C_{neu}} + C_{alt} C_{neu}} = \overline{\overline{C_{alt}} \oplus C_{neu}}$$

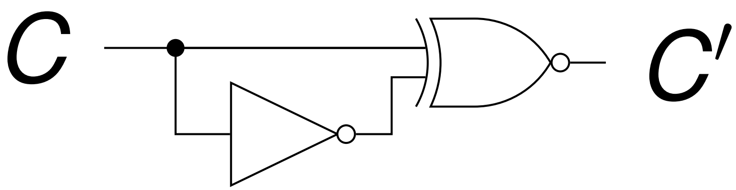
(i)

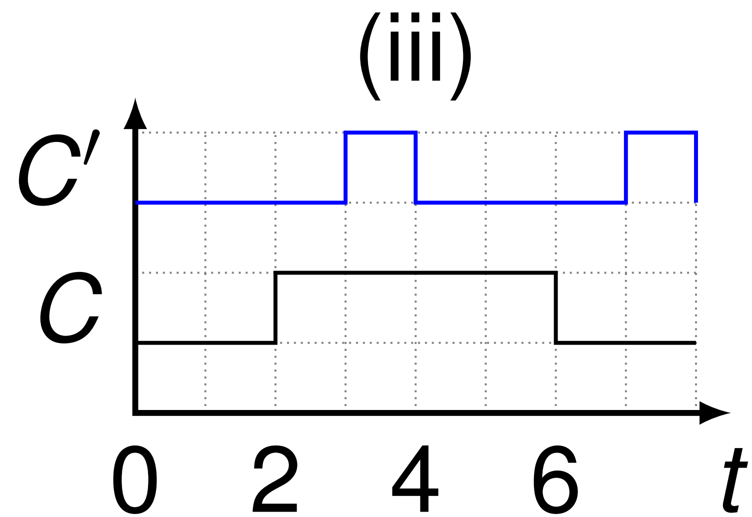


Steigende und Fallende Flanke II

Wir wissen, dass die Verzögerungszeit $\tau = 1$ beträgt. Dadurch entsteht ein C_{alt} nach dem Inverter, da die obere Leitung keine Verzögerung besitzt.

$f(C_{alt}, C_{neu}) = \overline{\overline{C_{alt}} \oplus C_{neu}}$

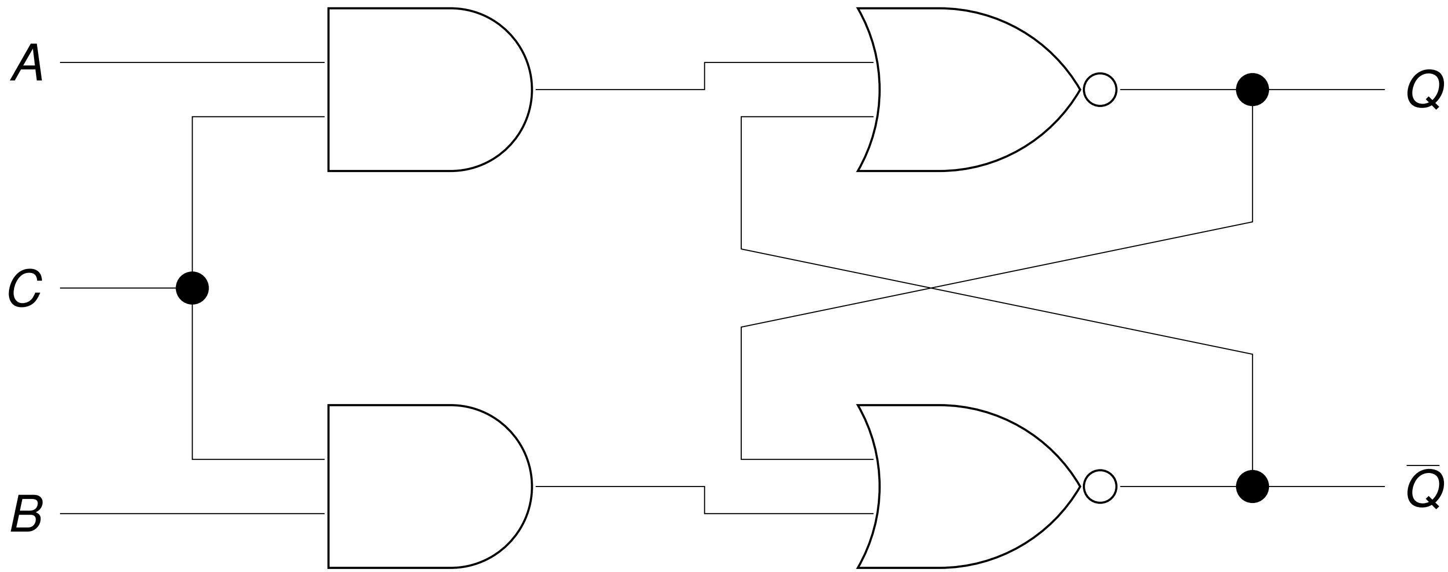




Das NOR-Gatter verzögert das C‘ um eine Zeiteinheit.

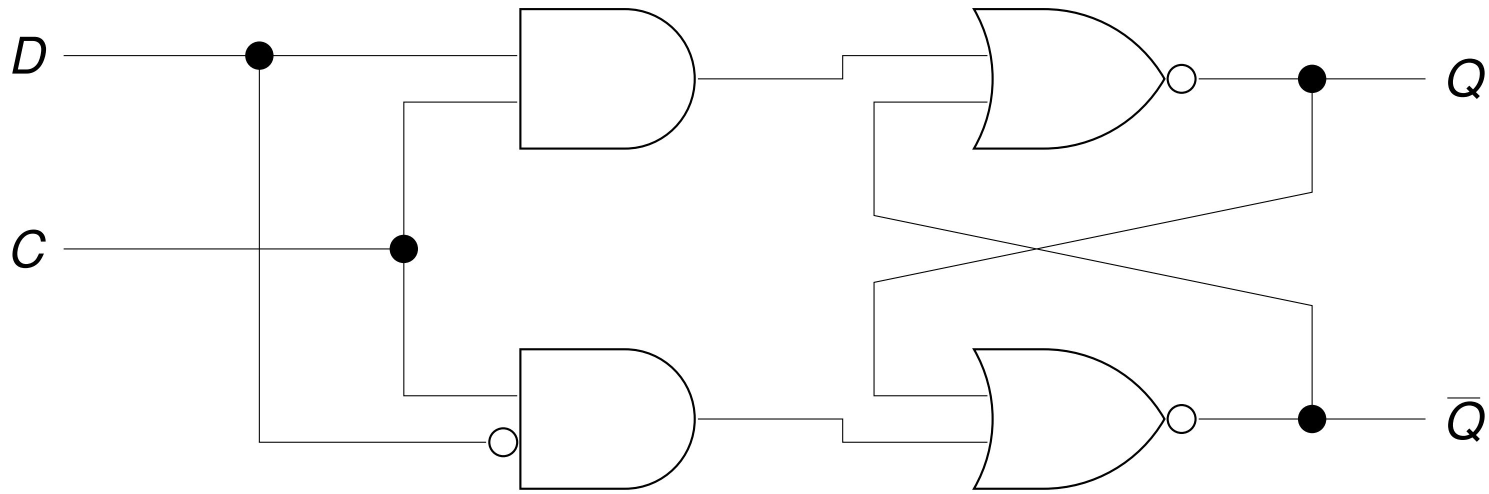
Aufgabe 4 – Latches und Flipflops

c) Erweitern Sie nun die Schaltung aus Teilaufgabe b) dahingehend, dass keine undefinierten Zustände, wie sie in Aufgabe a) der Fall waren, mehr auftreten.



Zum Beispiel: D-Flipflop

Gesucht: Elimination der ungültigen Zustände
Lösung: Nur noch eine Datenleitung (D-Flipflop)



Der zweite Eingang ist stets die Negation des ersten, weshalb der ungültige Zustand nicht auftreten kann.

Aufgabe 1 – Master-Slave-Flipflops

design

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

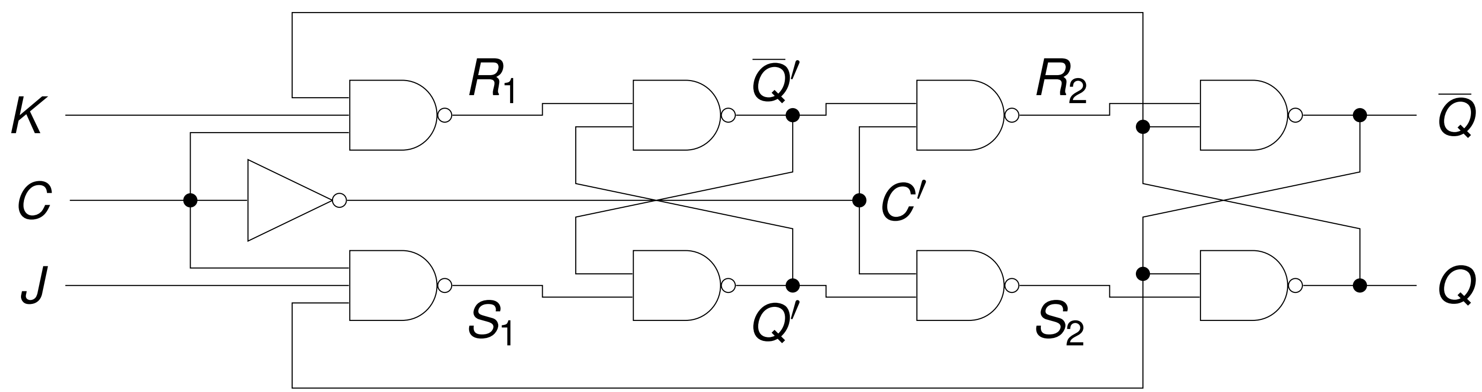
TECHNISCHE FAKULTÄT

Aufgabe 1 – Master-Slave-Flipflops

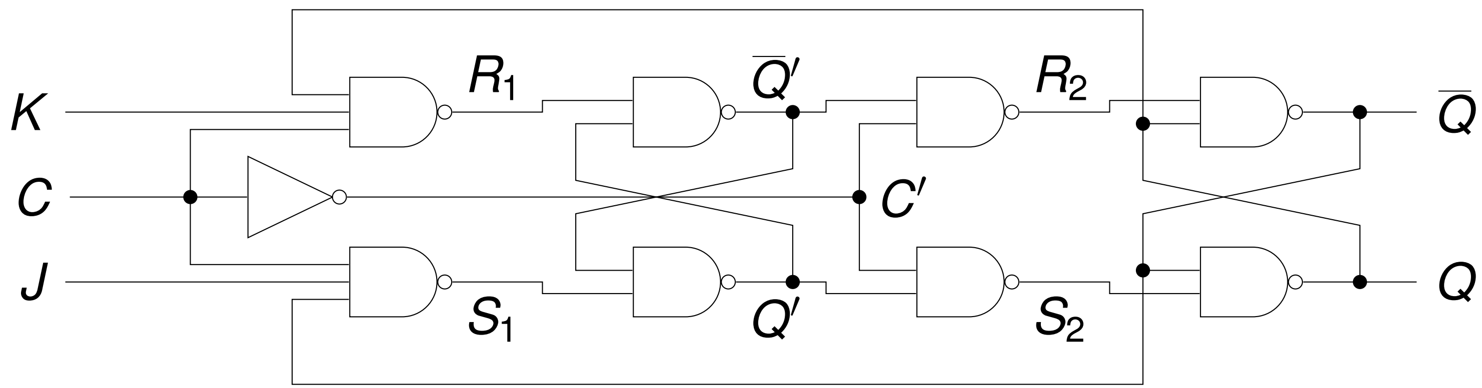
a) Welches Problem pegelgesteuerter JK-Flipflops löst das gegebene JK-Master-Slave-Flipflop?

Aufgabe 1 – Master-Slave-Flipflops

a) Welches Problem pegelgesteuerter JK-Flipflops löst das gegebene JK-Master-Slave-Flipflop?



Aufgabe 1 – Master-Slave-Flipflops

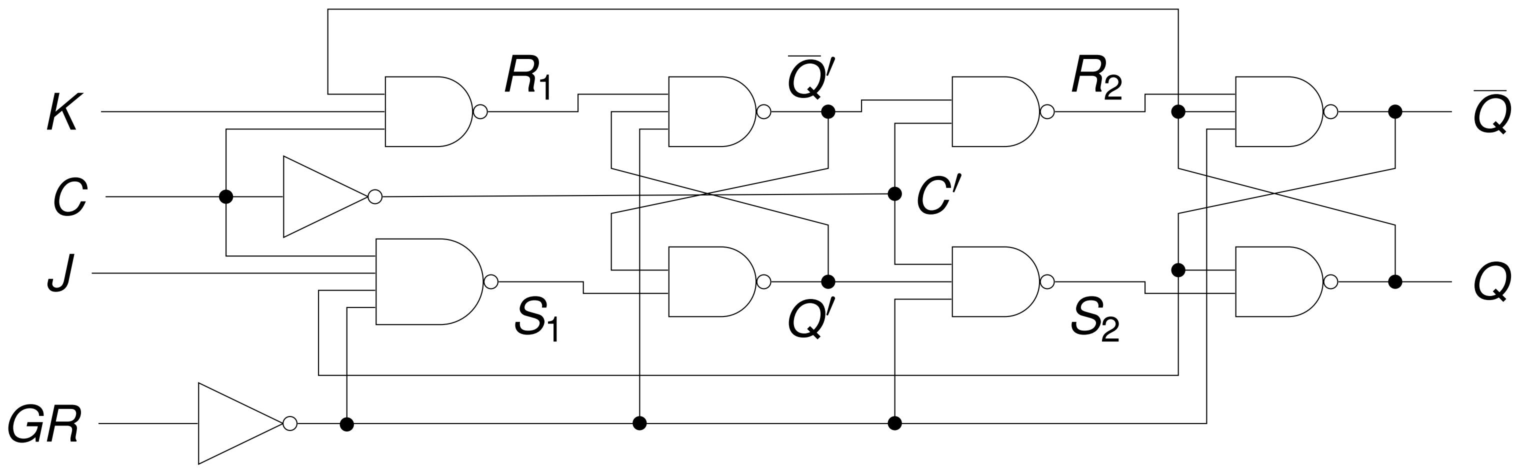


b) Erweitern Sie das gegebene JK-Master-Slave-Flipflop um ein Reset-Signal *GR*, das unabhängig von allen anderen Signalen das Flipflop asynchron zurücksetzt.

Asynchrones Reset I

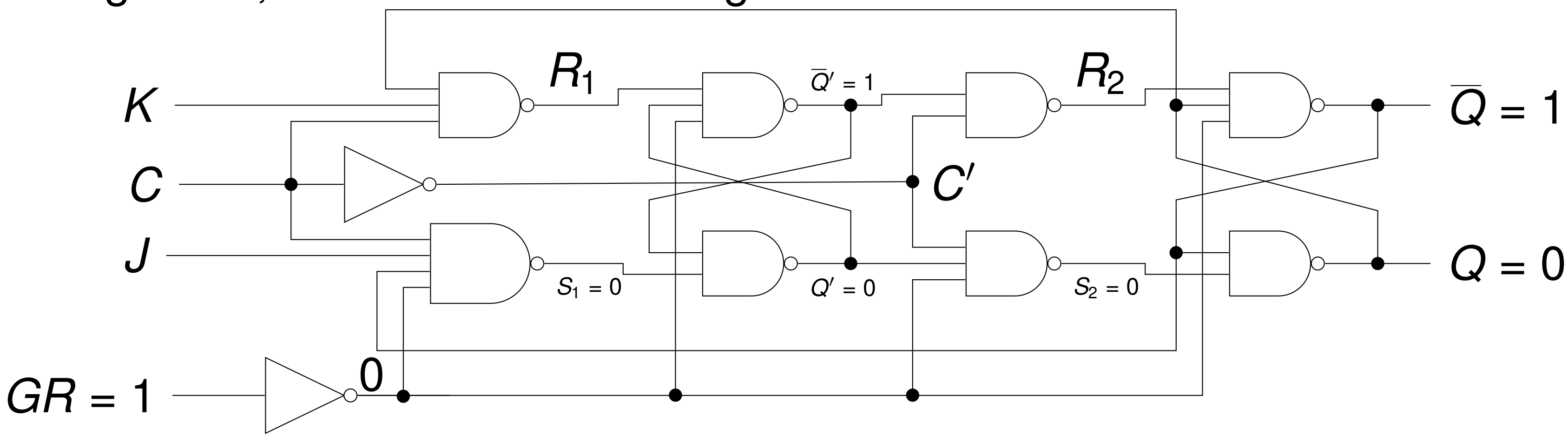
Zwei Aufgaben:

- Zurücksetzen des Master- und des Slave-Flipflop unabhängig der Werte K, R_1 und R_2.
- Setzen von S_1 und S_2 auf inaktiv, um den Toggle-Zustand zu verhindern.



Asynchrones Reset II

Ist $GR = 1$, wird deshalb durch den Inverter jeweils eine 0 an S_1 und S_2 angelegt und die beiden Gatter auf 1 gesetzt. Analog werden auch $\overline{Q}'$ und $\overline{Q}$ auf 1 gesetzt, also Q und Q' zurückgesetzt.



Aufgabe 1 – Master-Slave-Flipflops

c) Welches Problem ergibt sich für dieses JK-Master-Slave-Flipflop, wenn sich während der Einsphase des Taktes C die Eingänge J und K ändern?

Aufgabe 2 – Multiplexer und Demultiplexer

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 2 – Multiplexer und Demultiplexer

a) Entwerfen Sie eine digitale Schaltung, die bei einer 0 am Steuereingang s den Wert des Eingangs x_0, bei einer 1 am Steuereingang s den Wert des Eingangs x_1 am Ausgang y erzeugt:

s	x_0	x_1	y
0	0	-	0
0	1	-	1
1	-	0	0
1	-	1	1

Aufgabe 2 – Multiplexer und Demultiplexer

b) Die Schaltung soll nun so erweitert werden, dass wahlweise genau einer von vier Eingängen $x_0 \ldots x_3$ am Ausgang y erscheint.

Aufgabe 2 – Multiplexer und Demultiplexer

b) Die Schaltung soll nun so erweitert werden, dass wahlweise genau einer von vier Eingängen $x_0 \dots x_3$ am Ausgang y erscheint.

- Welche Auswirkungen hat dies für den Steuereingang s?

Aufgabe 2 – Multiplexer und Demultiplexer

b) Die Schaltung soll nun so erweitert werden, dass wahlweise genau einer von vier Eingängen $x_0 \dots x_3$ am Ausgang y erscheint.

- Realisieren Sie die Schaltung mit Und-/Oder-Gattern sowie Invertern.

Aufgabe 2 – Multiplexer und Demultiplexer

b) Die Schaltung soll nun so erweitert werden, dass wahlweise genau einer von vier Eingängen $x_0 \ldots x_3$ am Ausgang y erscheint.

- Welche Auswirkungen hat dies für den Steuereingang s?
- Realisieren Sie die Schaltung mit Und-/Oder-Gattern sowie Invertern.

Aufgabe 2 – Multiplexer und Demultiplexer

c) Schließlich soll eine Demultiplexer-Schaltung entworfen werden. Sie besitzt einen Eingang x, der in Abhängigkeit des Steuereingangs s den Wert von x an einem der vier Ausgänge $y_0 \ldots y_3$ erzeugt (siehe folgende Wahrheitstabelle).

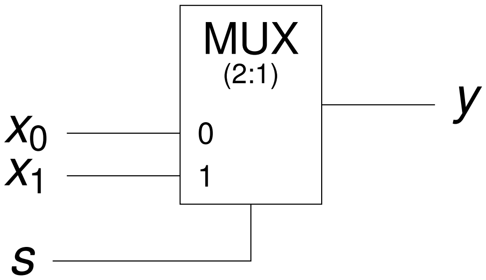
$s_1 s_0$	x	y_0	y_1	y_2	y_3
00	$^0/_1$	$^0/_1$	0	0	0
01	$^0/_1$	0	$^0/_1$	0	0
10	$^0/_1$	0	0	$^0/_1$	0
11	$^0/_1$	0	0	0	$^0/_1$

Aufgabe 3 – Barrel-Shifter

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 3 – Barrel-Shifter

Erstellen Sie aus 2:1-Multiplexern, wie im nebenstehenden Blockschaltbild dargestellt, einen Barrel-Shifter, der den Eingang $A = (a_3, a_2, a_1, a_0)$ zyklisch um $N = 0 \ldots 3$ Stellen nach links verschiebt und auf dem Ausgang $B = (b_3, b_2, b_1, b_0)$ ausgibt. Ist zum Beispiel $N = 1$, so soll $b_3 = a_2$, $b_2 = a_1$, $b_1 = a_0$, $b_0 = a_3$ gelten.

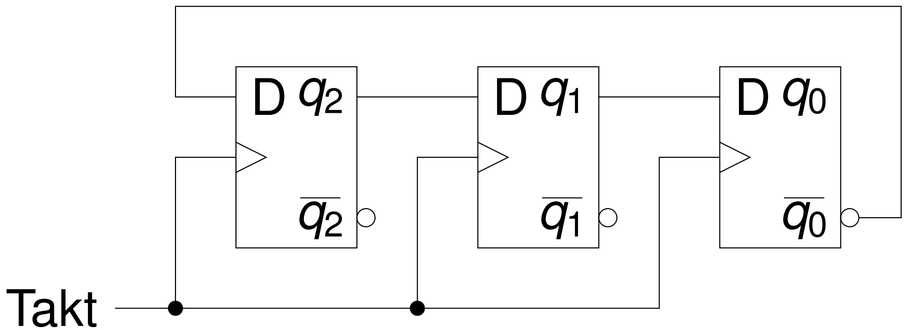


Aufgabe 4 – Schieberegister

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 4 – Schieberegister

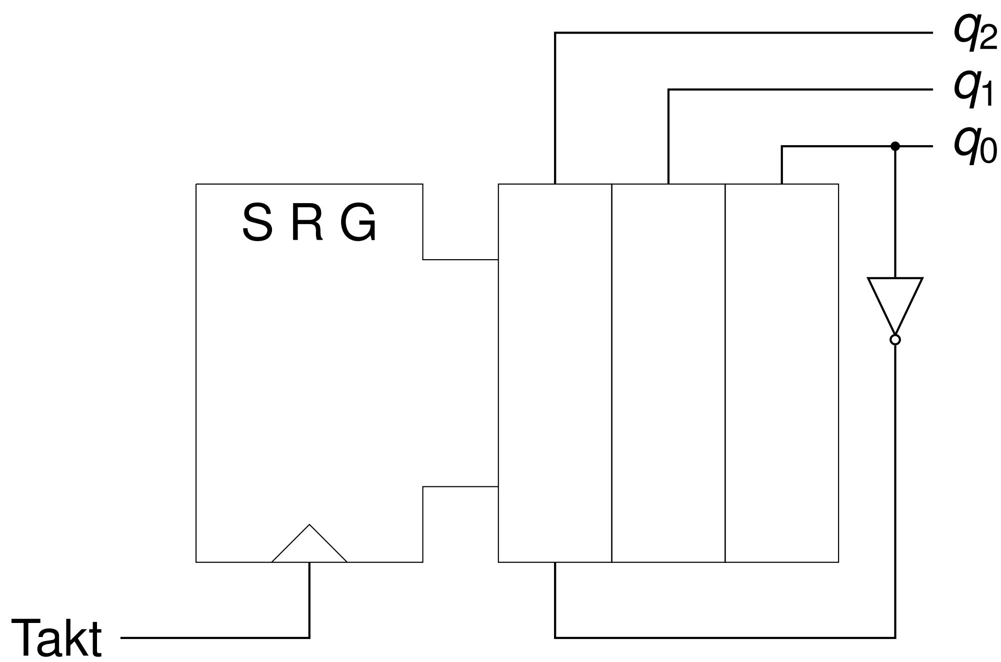
Das folgende Bild zeigt einen sogenannten Johnson-Zähler, bei dem das invertierte Signal q_0 des letzten Flipflops in der Kette an den Eingang D des ersten angeschlossen wird.



a) Der aktuelle Wert der Speicherzellen sei `000`. Stellen Sie die Folge der Speicherinhalte als gerichteten Graphen dar und bestimmen Sie die Anzahl N der durchlaufenen Zustände.

Aufgabe 4 – Schieberegister: Johnson-Zähler

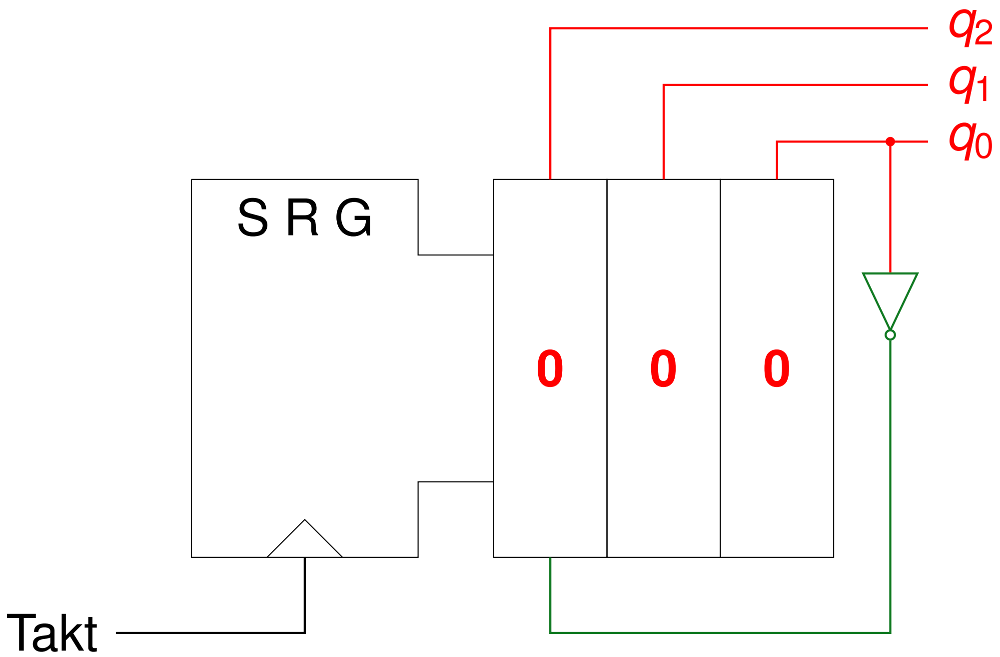
Startwert: 000



000

Aufgabe 4 – Schieberegister: Johnson-Zähler

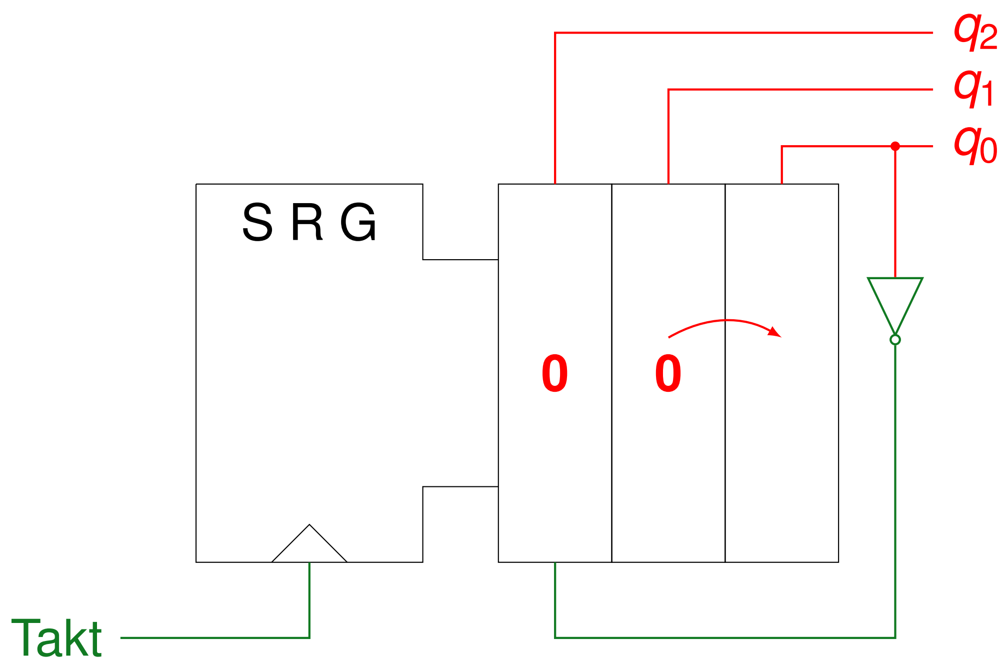
Startwert: 000



000

Aufgabe 4 – Schieberegister: Johnson-Zähler

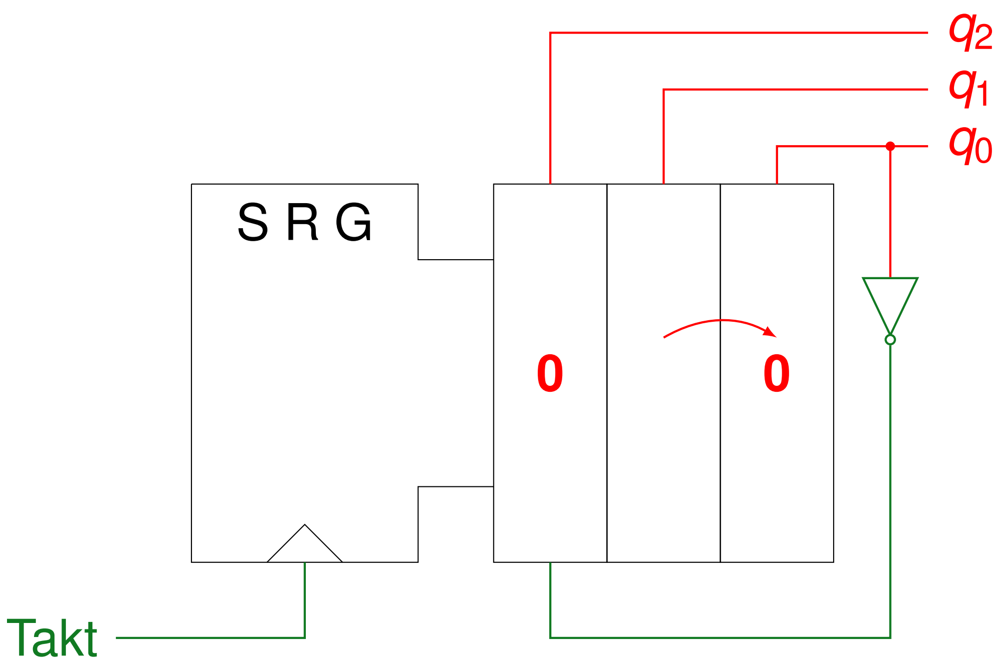
Startwert: 000



000

Aufgabe 4 – Schieberegister: Johnson-Zähler

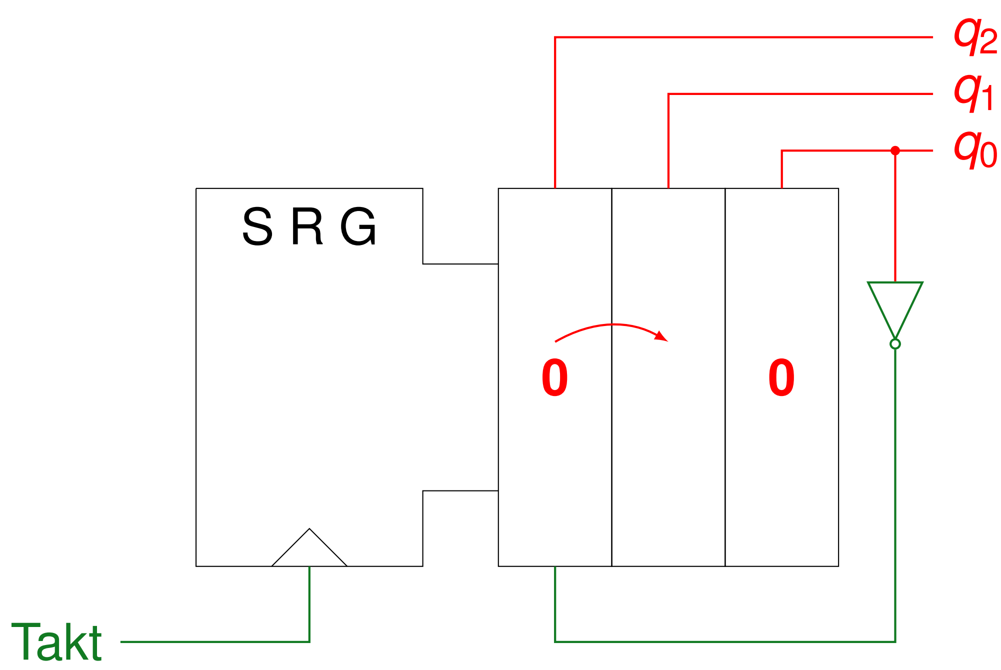
Startwert: 000



000

Aufgabe 4 – Schieberegister: Johnson-Zähler

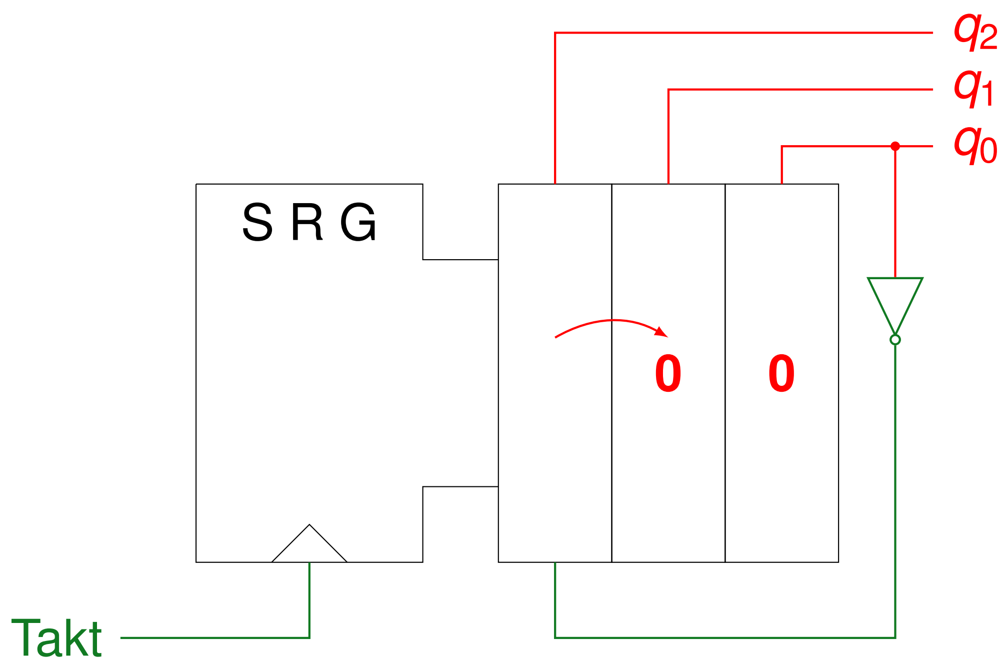
Startwert: 000



000

Aufgabe 4 – Schieberegister: Johnson-Zähler

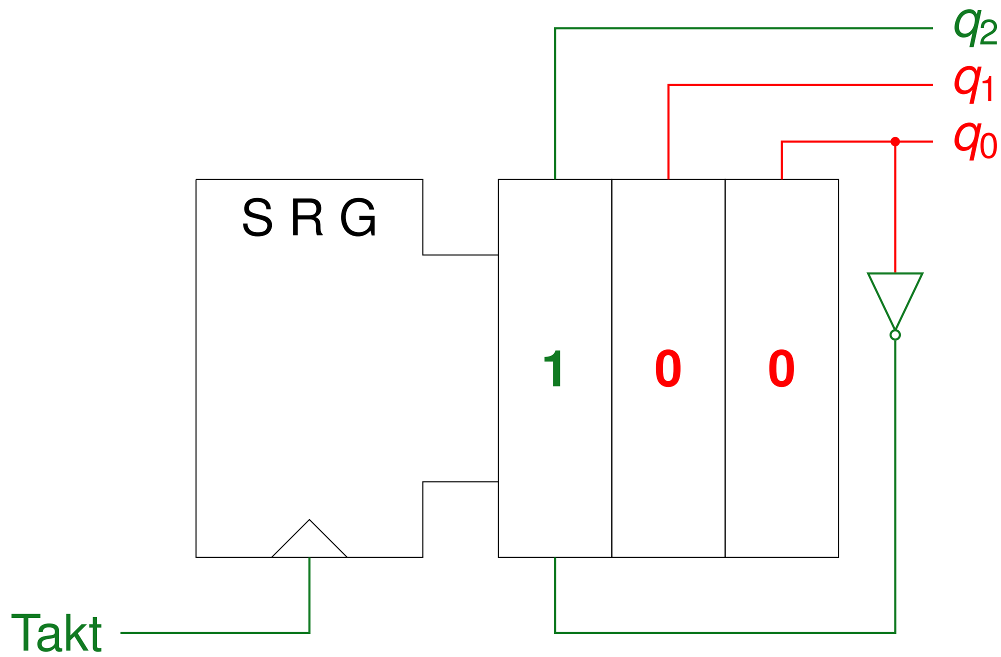
Startwert: 000



000

Aufgabe 4 – Schieberegister: Johnson-Zähler

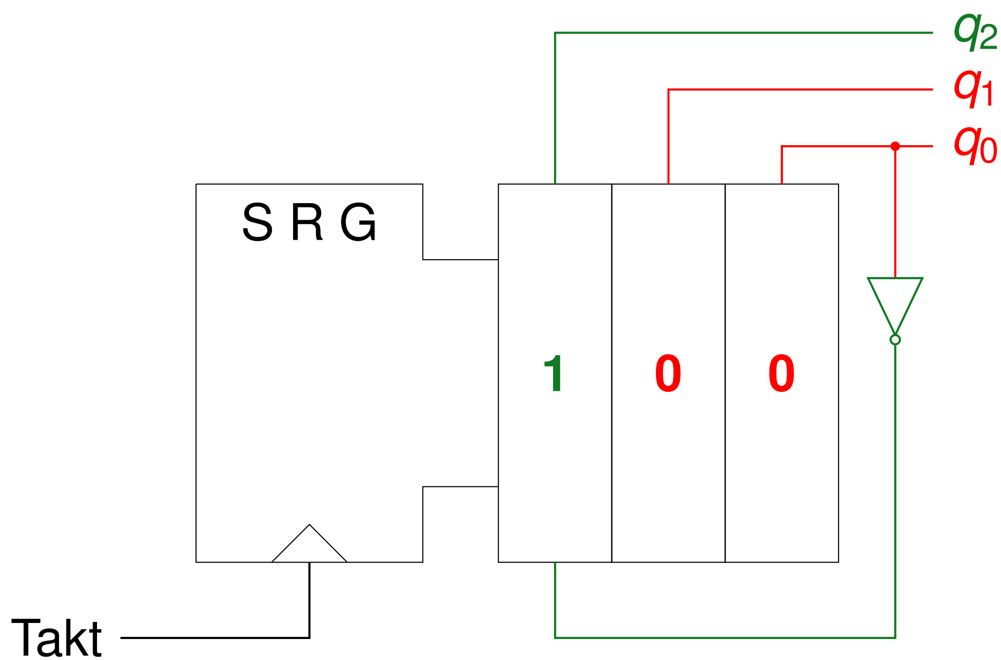
Startwert: 000



000

Aufgabe 4 – Schieberegister: Johnson-Zähler

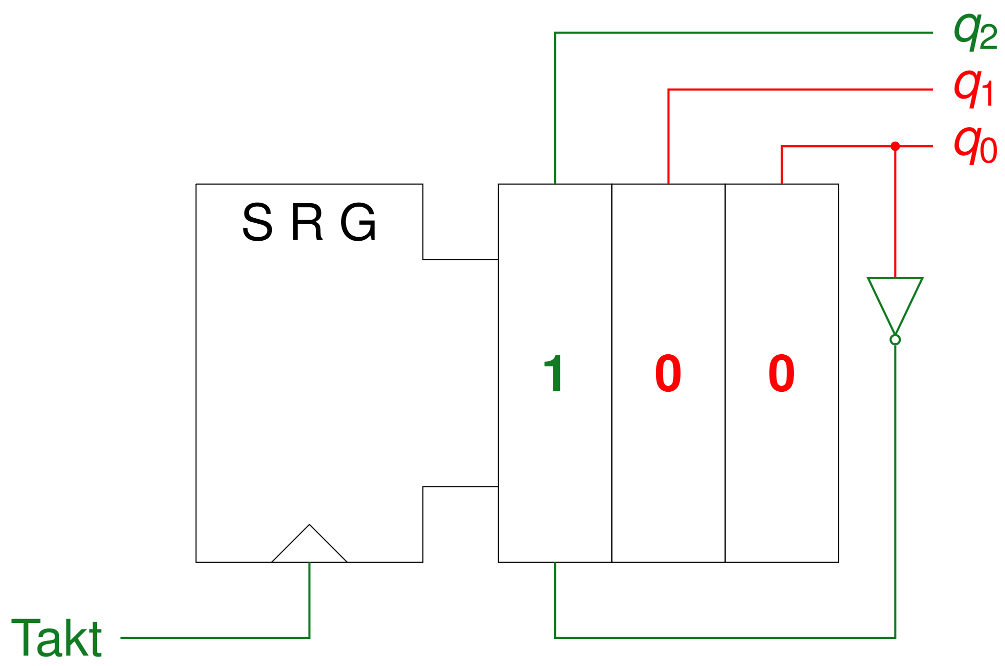
Startwert: 000



$$000 \mapsto 100$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

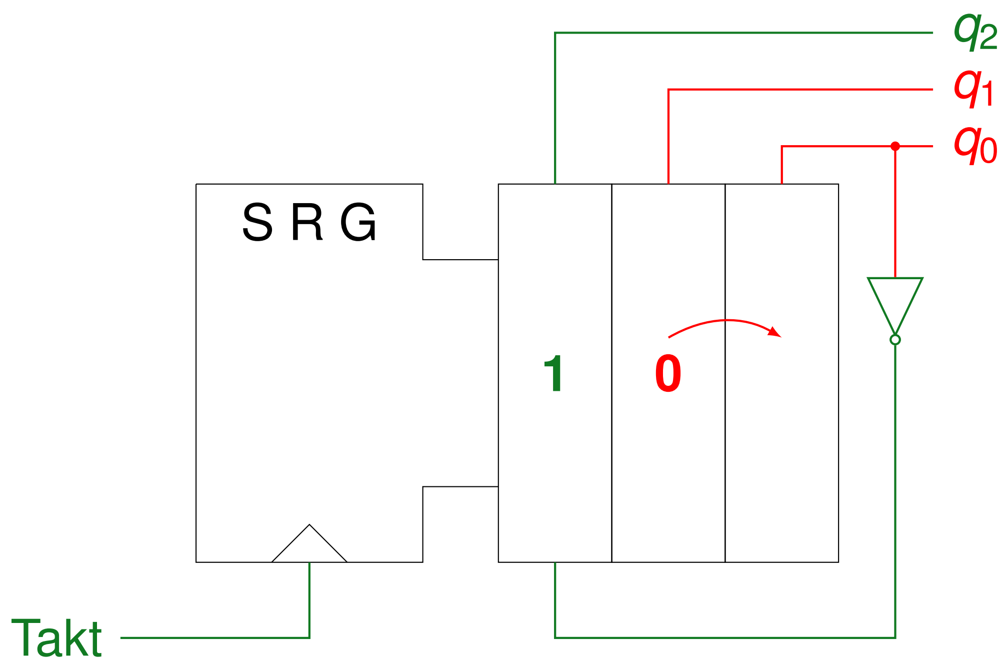
Startwert: 000



$$000 \mapsto 100$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

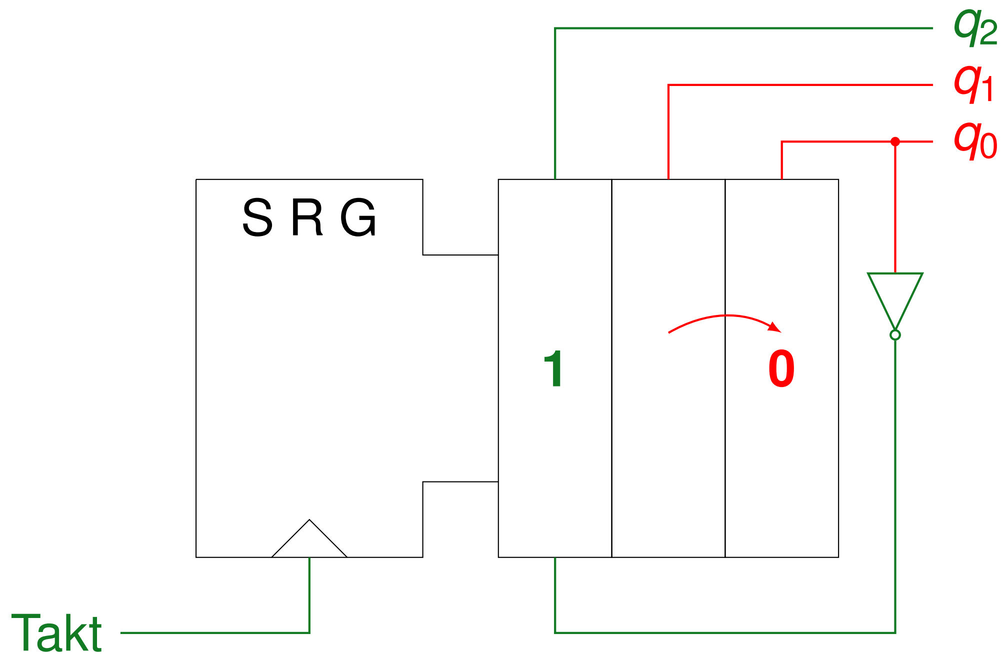
Startwert: 000



$$000 \mapsto 100$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

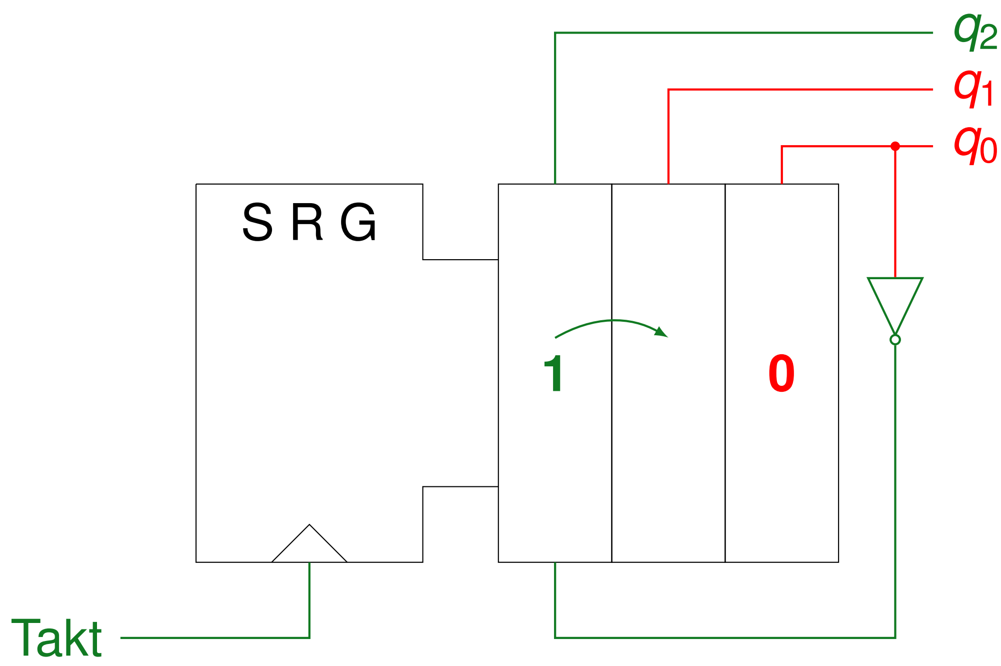
Startwert: 000



$$000 \mapsto 100$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

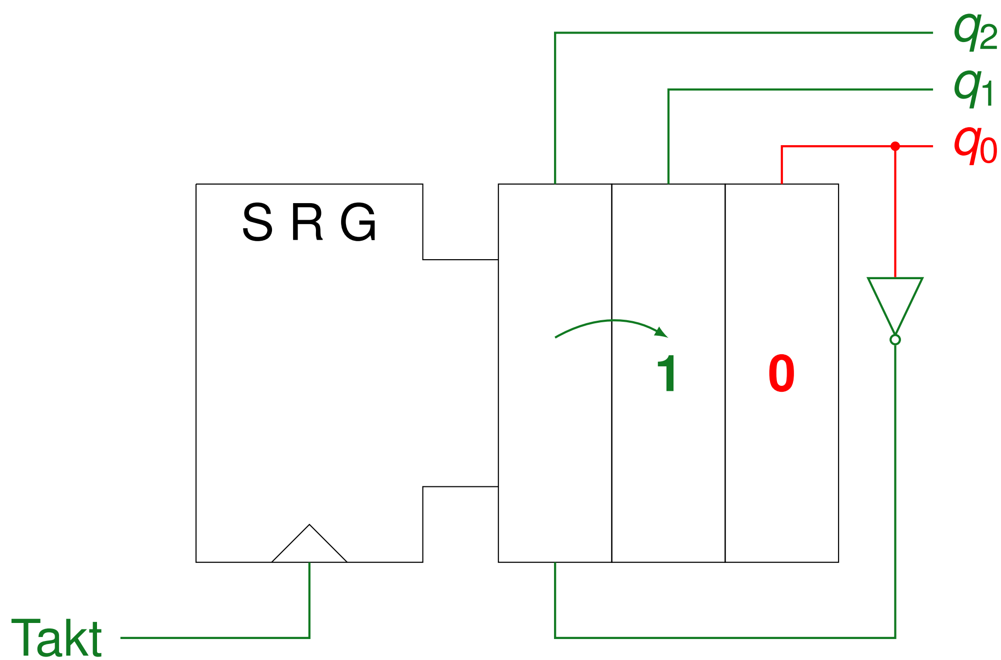
Startwert: 000



$$000 \mapsto 100$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

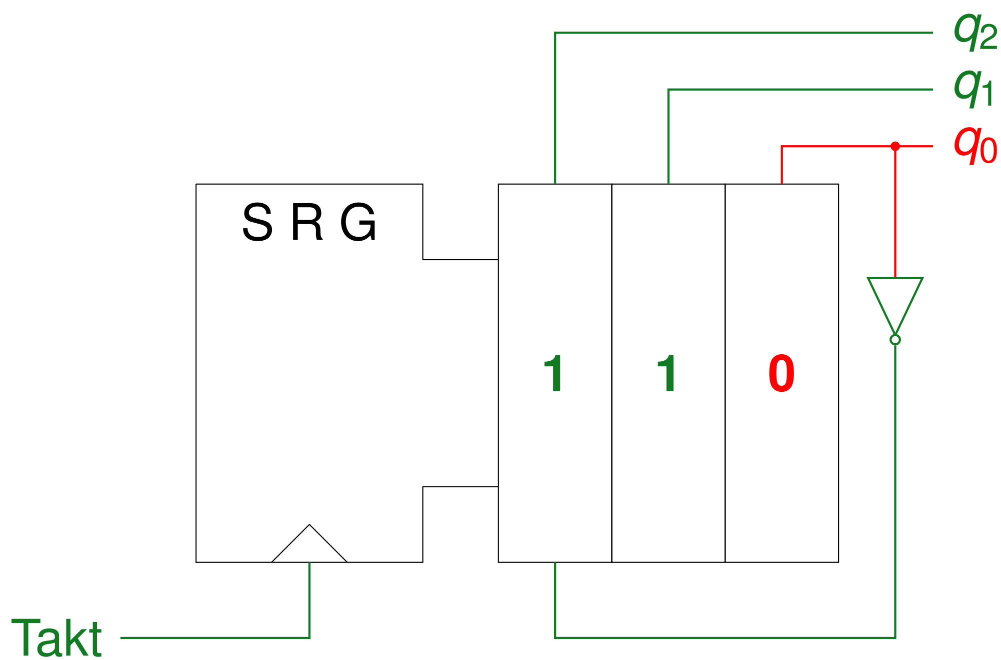
Startwert: 000



$000 \mapsto 100$

Aufgabe 4 – Schieberegister: Johnson-Zähler

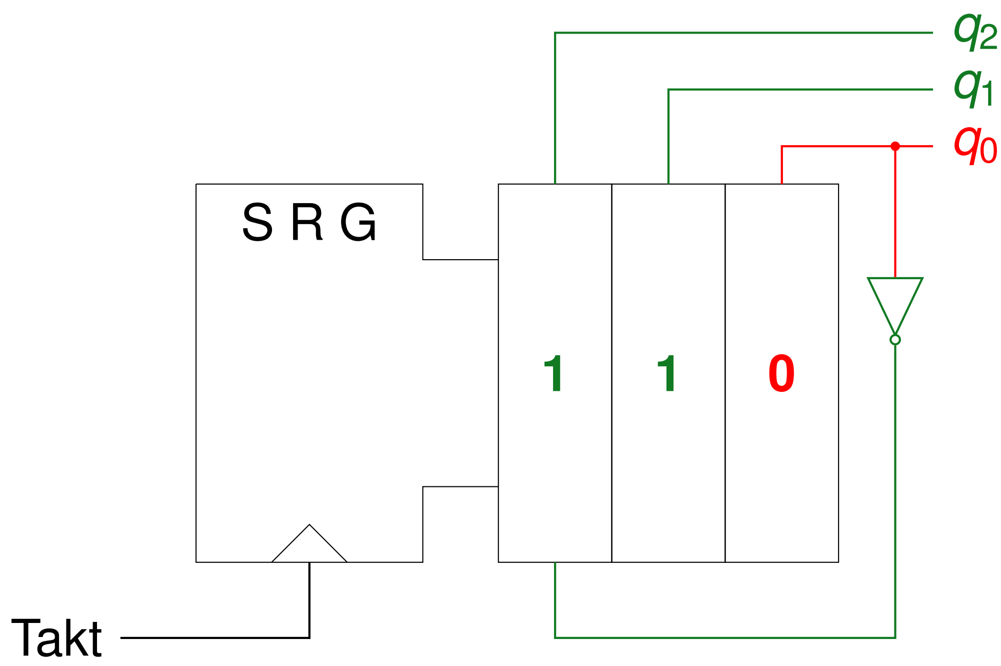
Startwert: 000



$$000 \mapsto 100$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

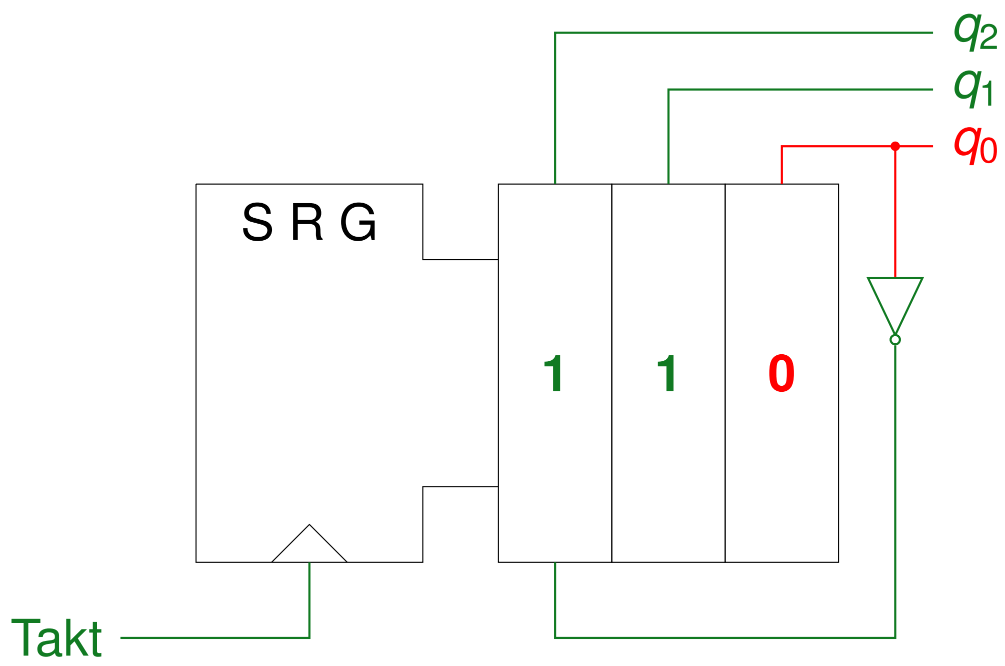
Startwert: 000



$$000 \mapsto 100 \mapsto 110$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

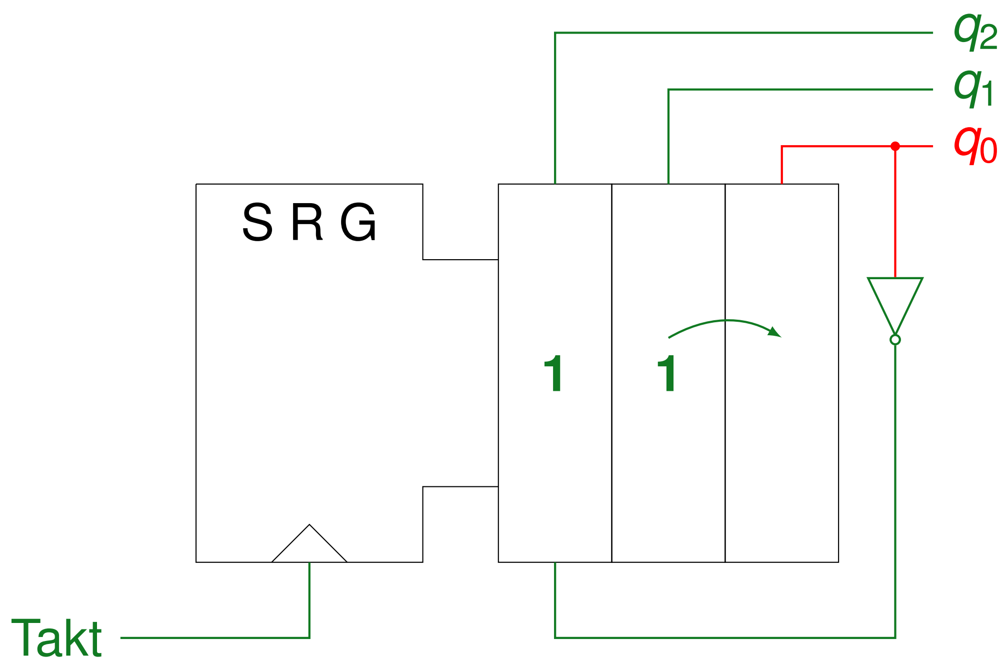
Startwert: 000



$$000 \mapsto 100 \mapsto 110$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

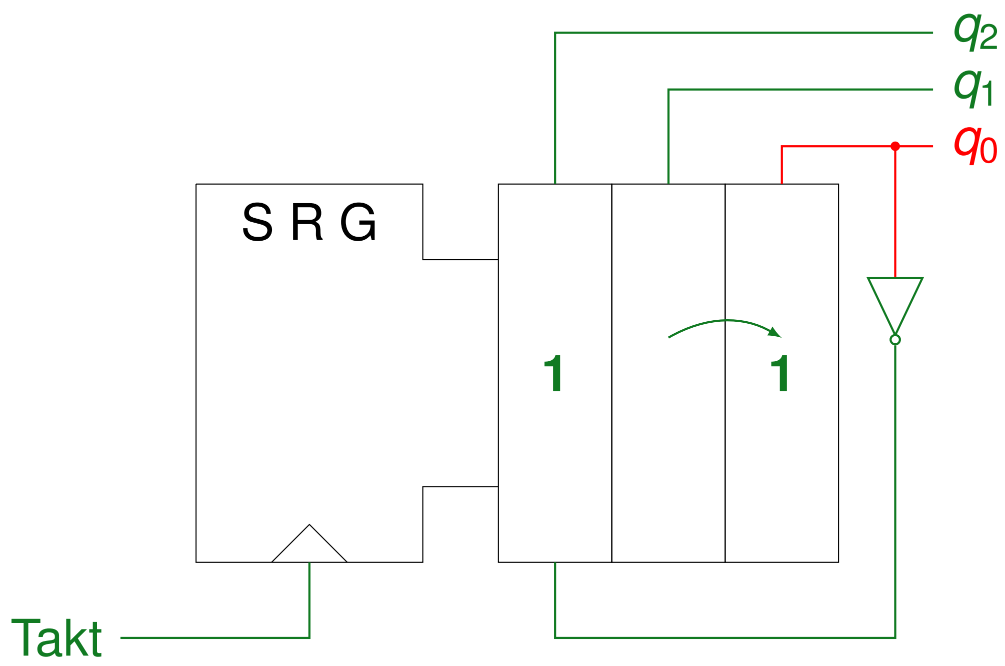
Startwert: 000



$$000 \mapsto 100 \mapsto 110$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

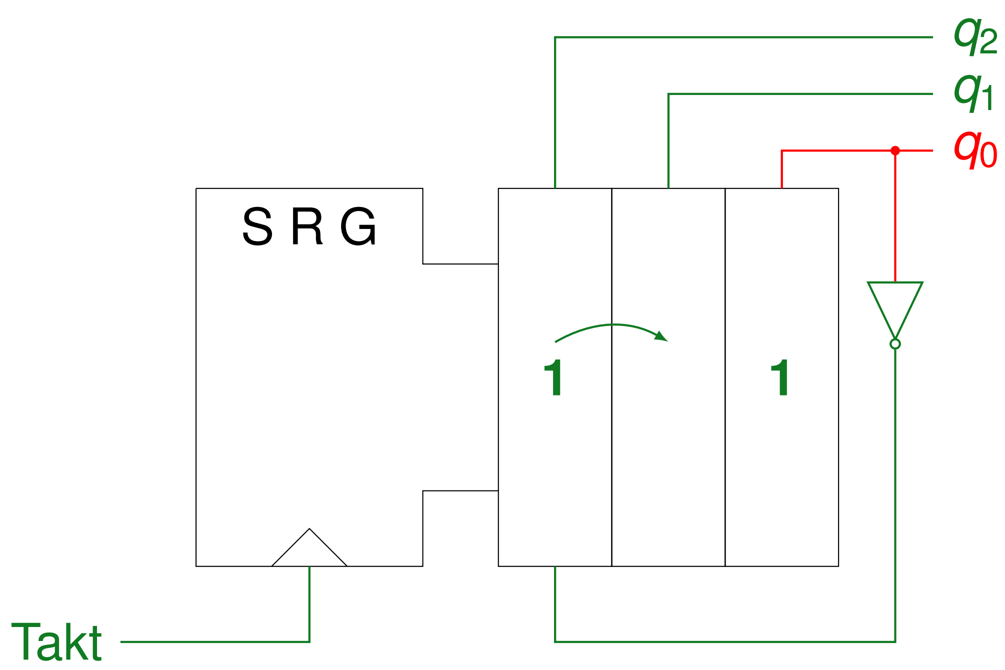
Startwert: 000



$$000 \mapsto 100 \mapsto 110$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

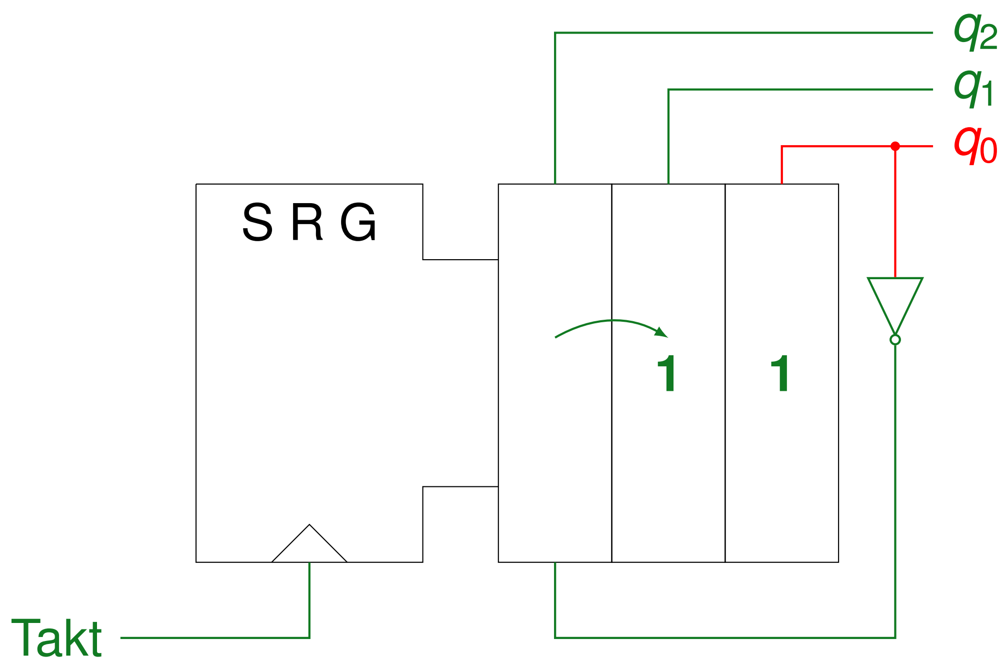
Startwert: 000



$$000 \mapsto 100 \mapsto 110$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

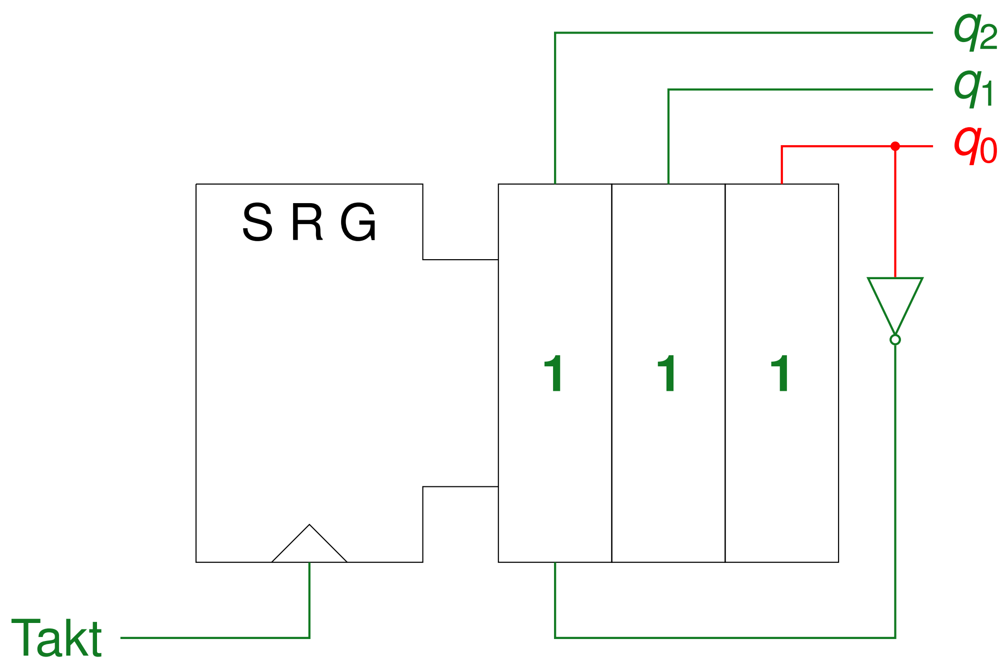
Startwert: 000



$$000 \mapsto 100 \mapsto 110$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

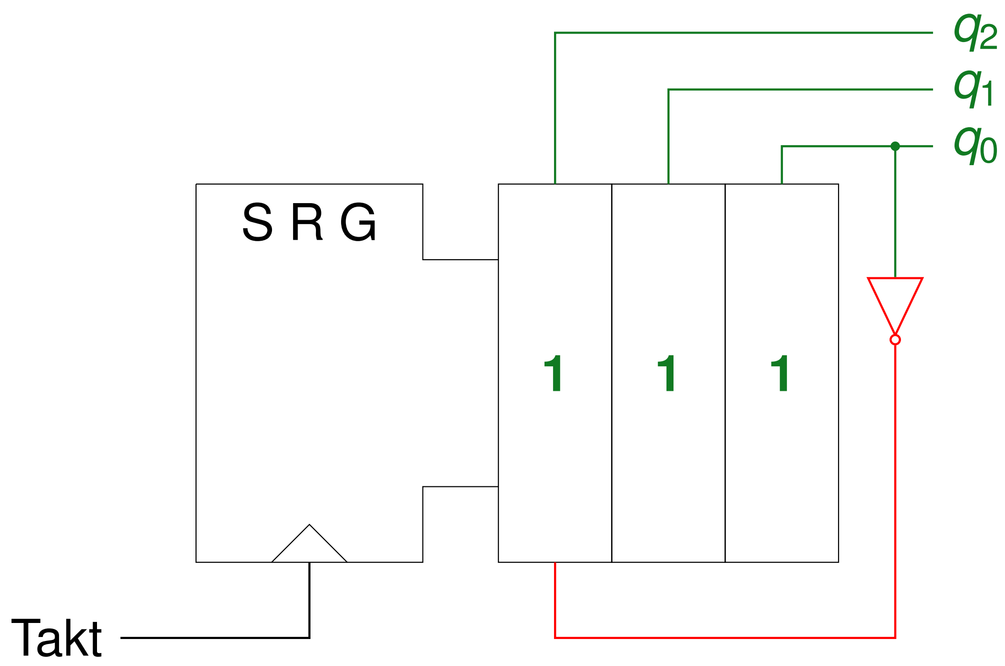
Startwert: 000



$000 \mapsto 100 \mapsto 110$

Aufgabe 4 – Schieberegister: Johnson-Zähler

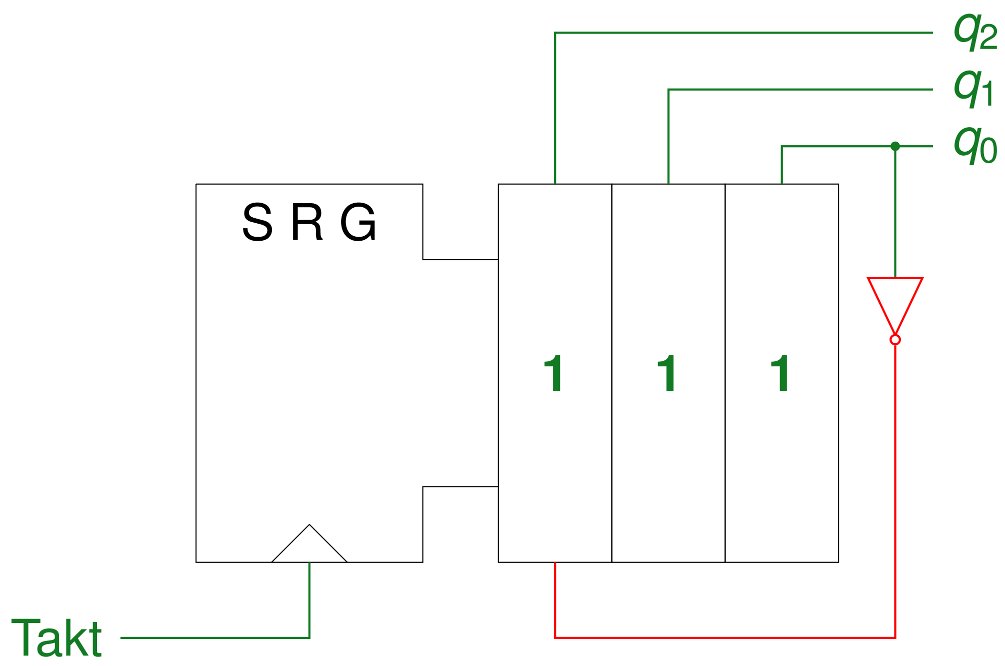
Startwert: 000



$$000 \mapsto 100 \mapsto 110 \mapsto 111$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

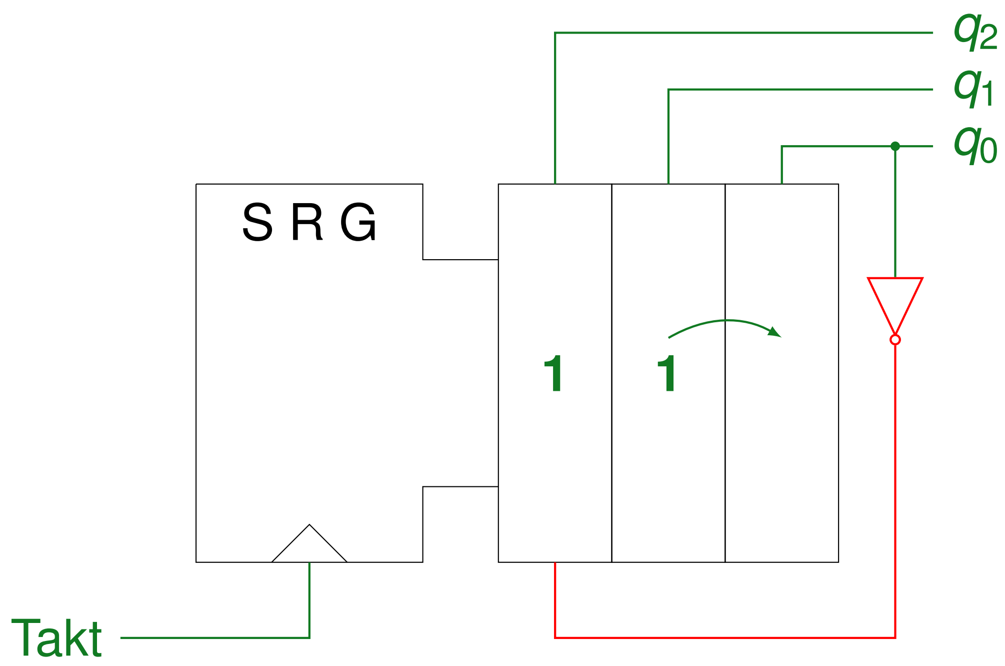
Startwert: 000



$$000 \mapsto 100 \mapsto 110 \mapsto 111$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

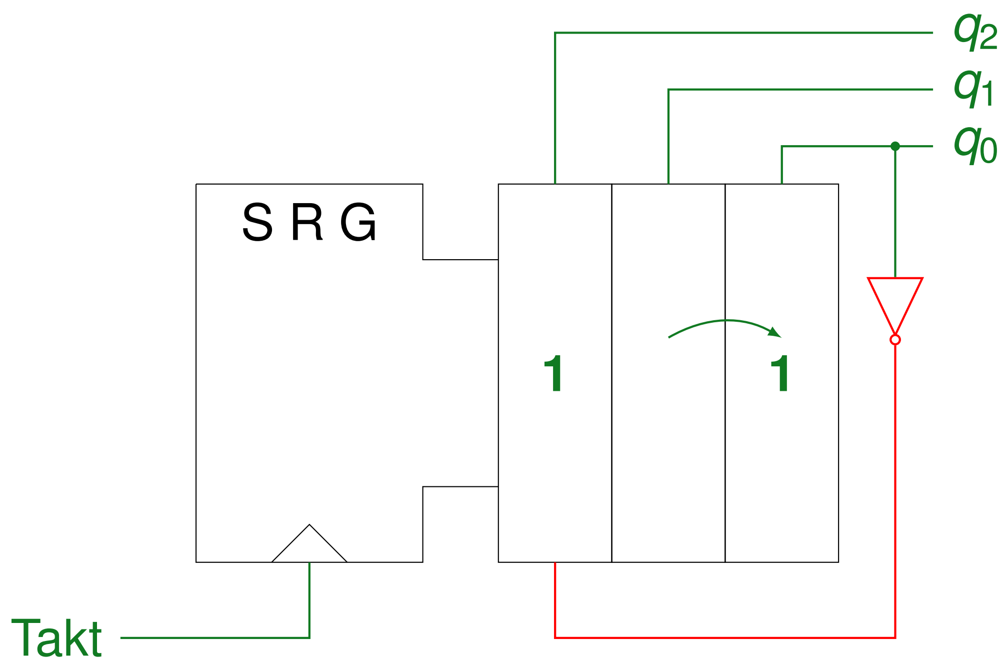
Startwert: 000



$$000 \mapsto 100 \mapsto 110 \mapsto 111$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

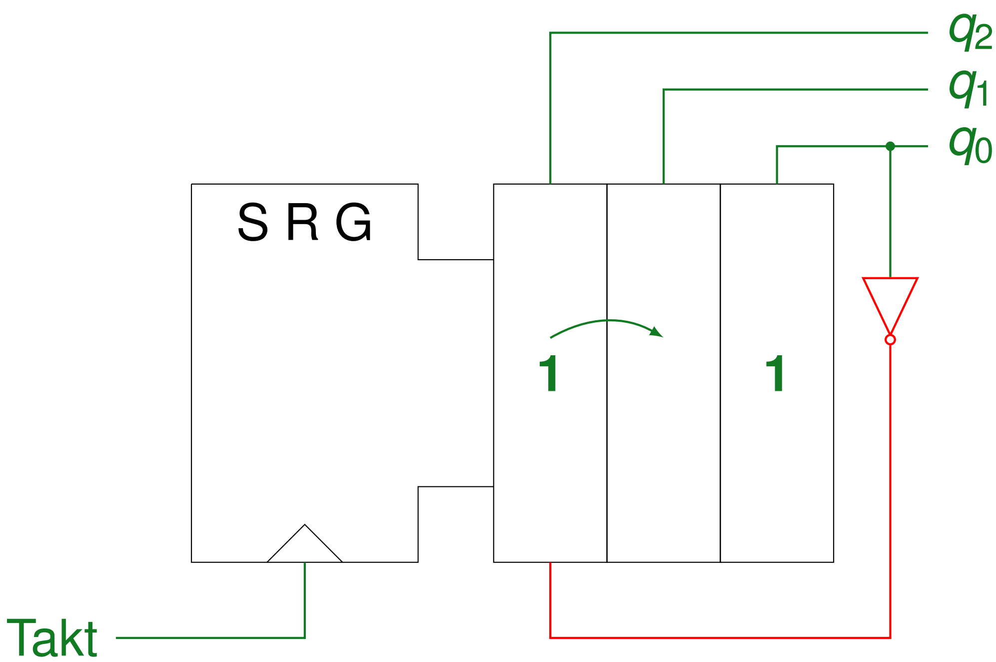
Startwert: 000



$$000 \mapsto 100 \mapsto 110 \mapsto 111$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

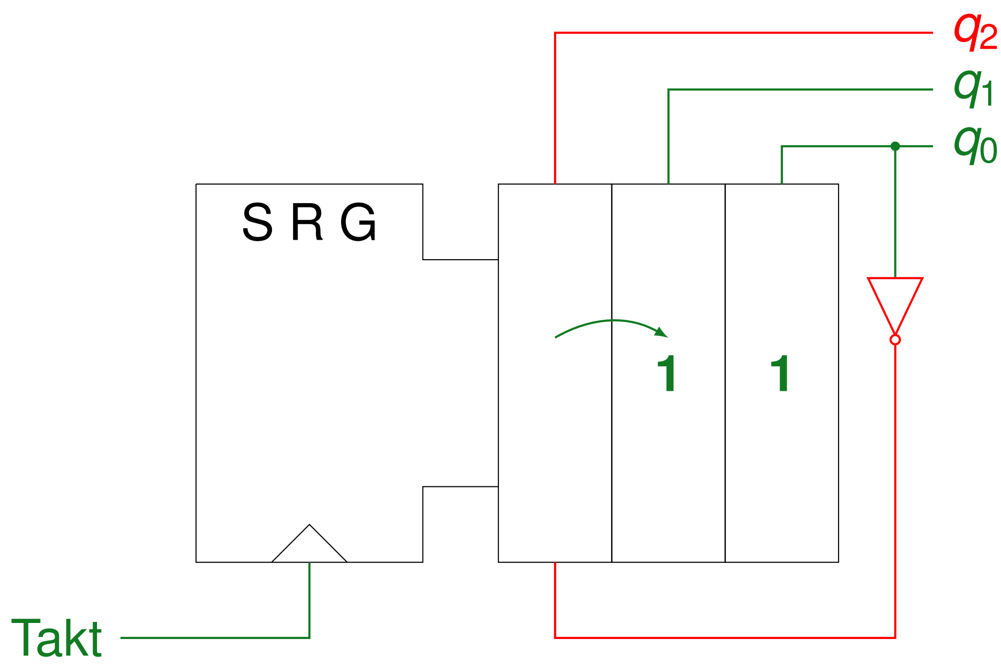
Startwert: 000



$$000 \mapsto 100 \mapsto 110 \mapsto 111$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

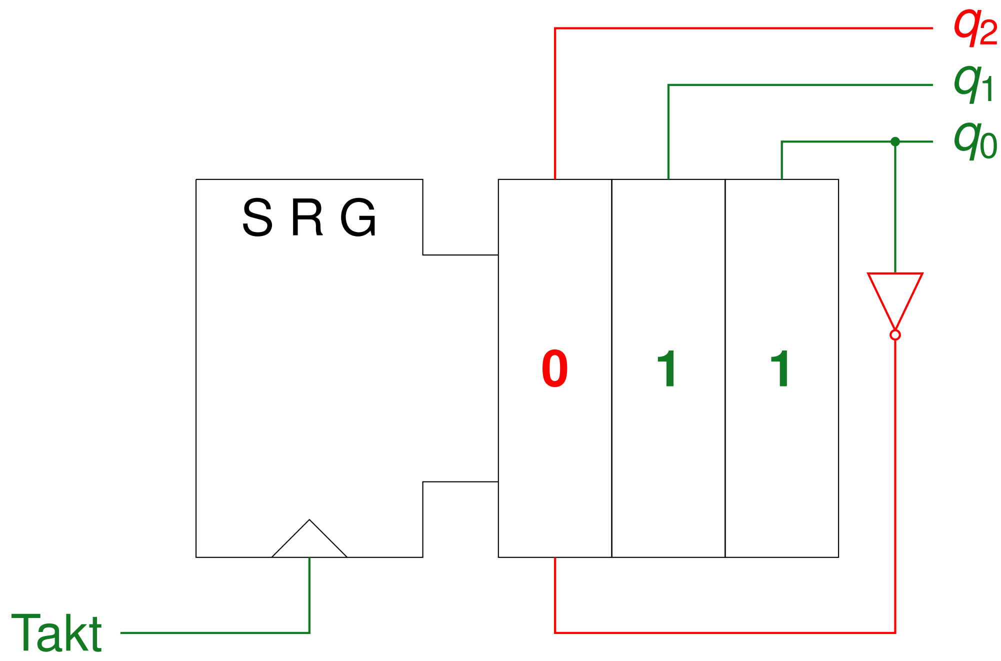
Startwert: 000



$$000 \mapsto 100 \mapsto 110 \mapsto 111$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

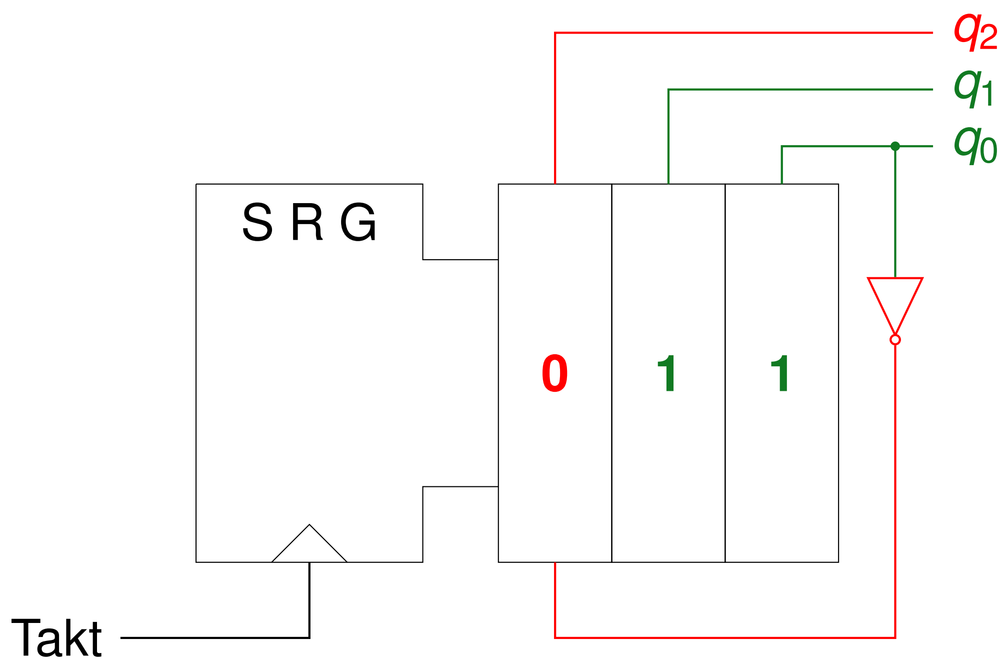
Startwert: 000



$$000 \mapsto 100 \mapsto 110 \mapsto 111$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

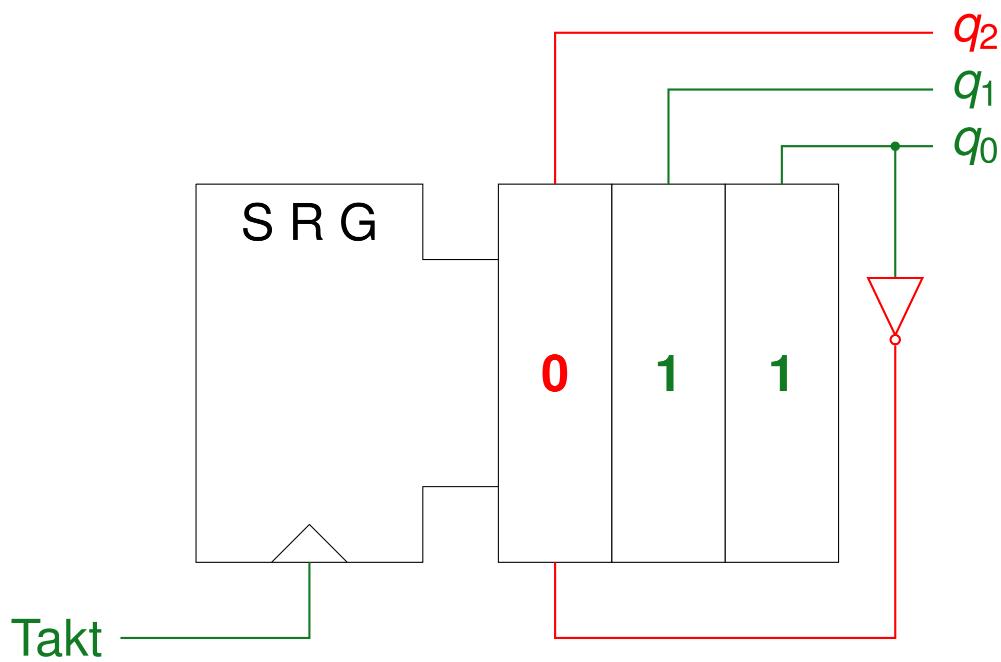
Startwert: 000



$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

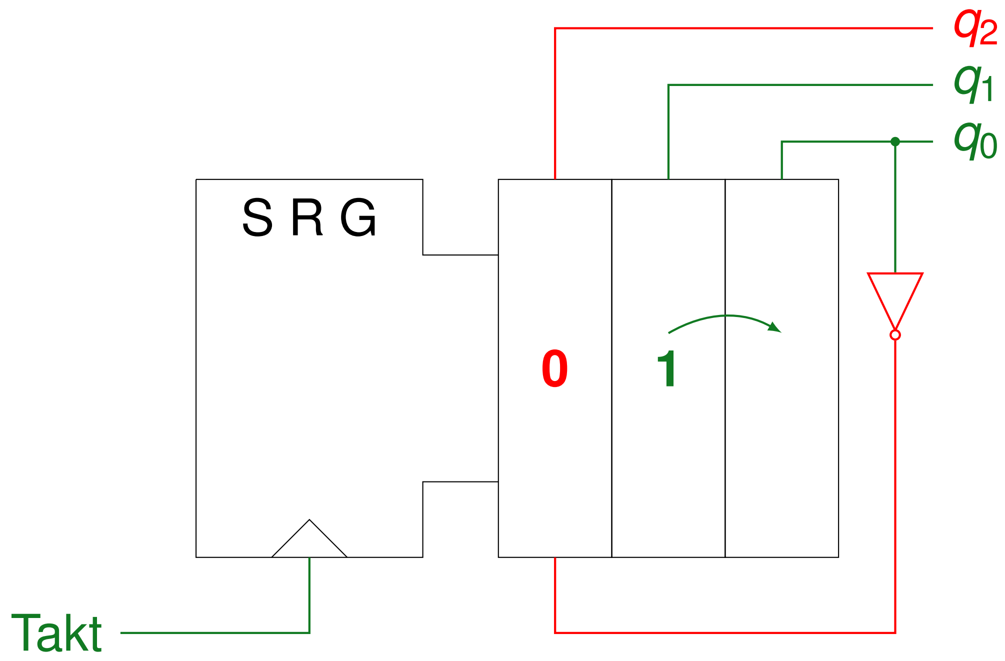
Startwert: 000



$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

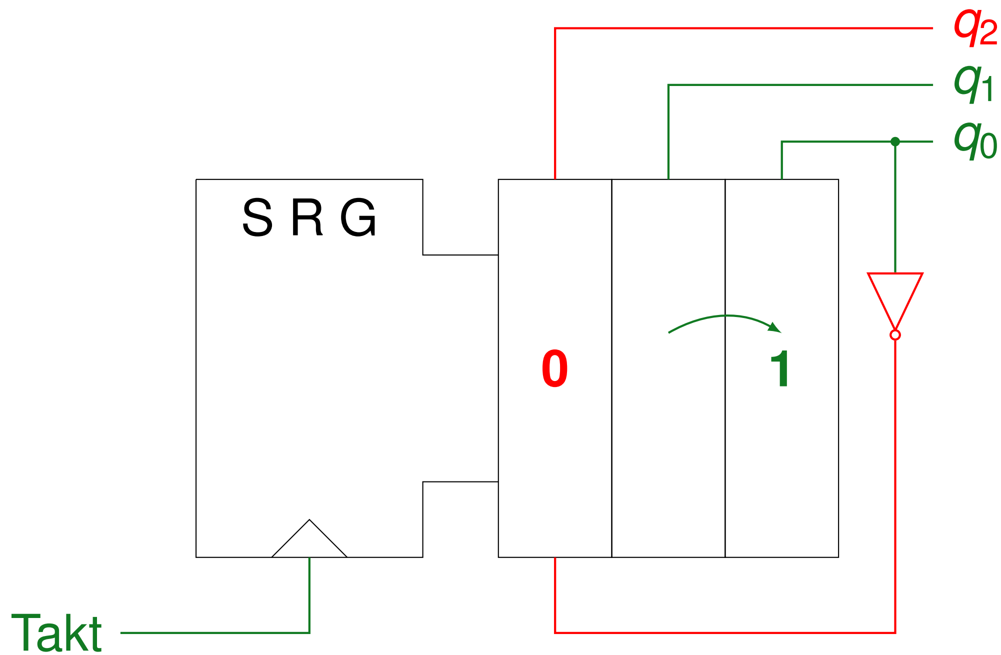
Startwert: 000



$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

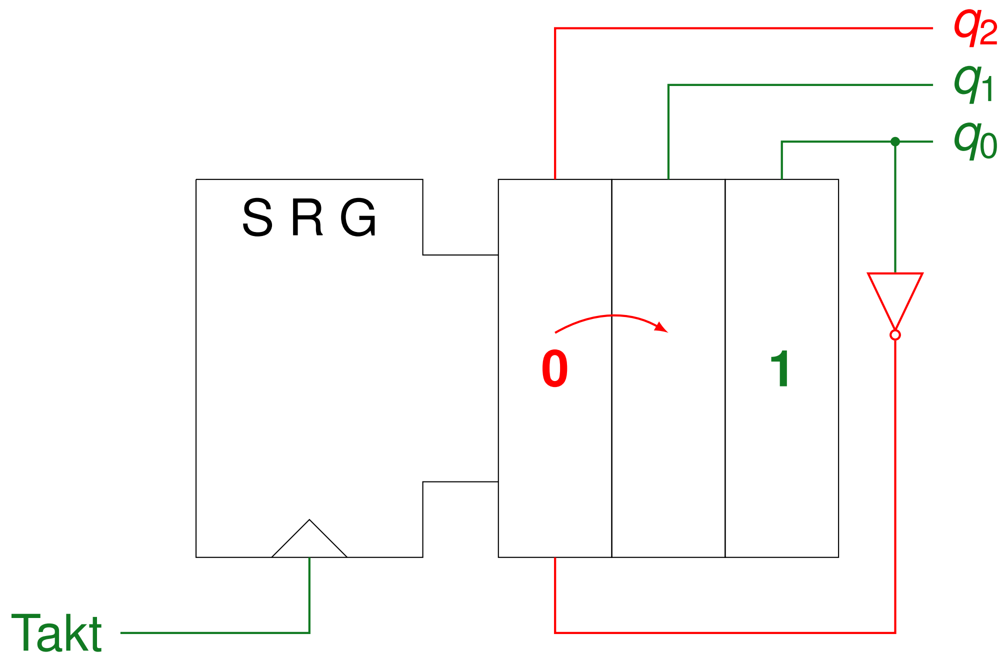
Startwert: 000



$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

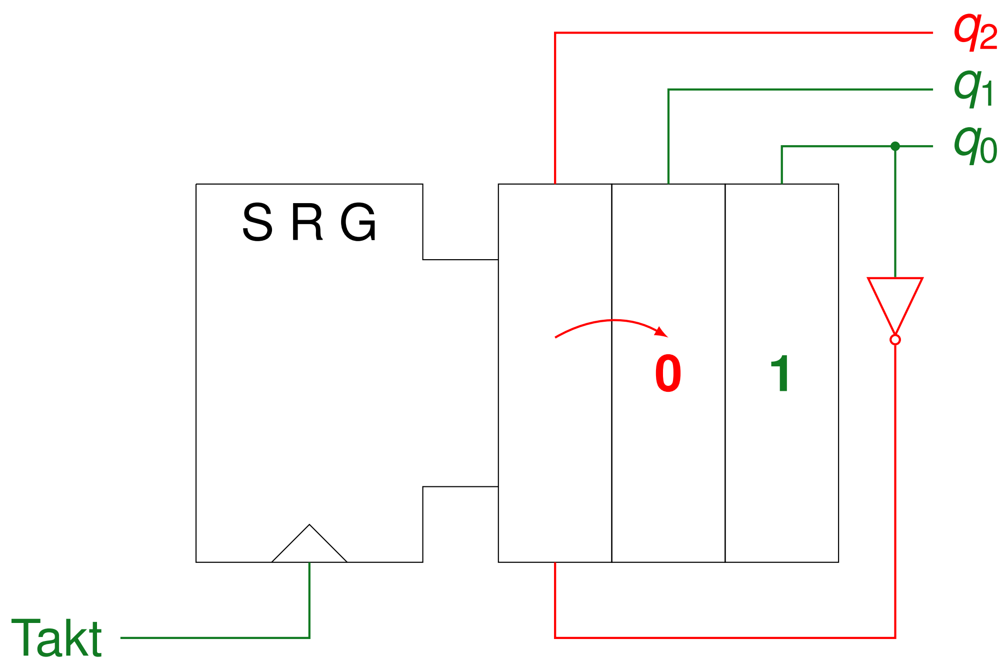
Startwert: 000



$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

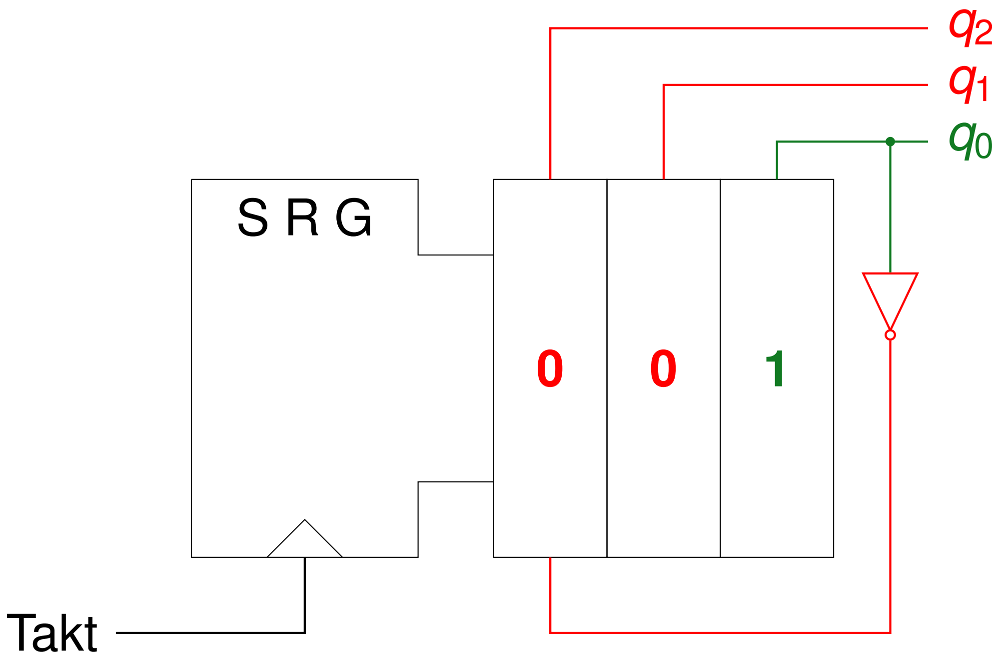
Startwert: 000



$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

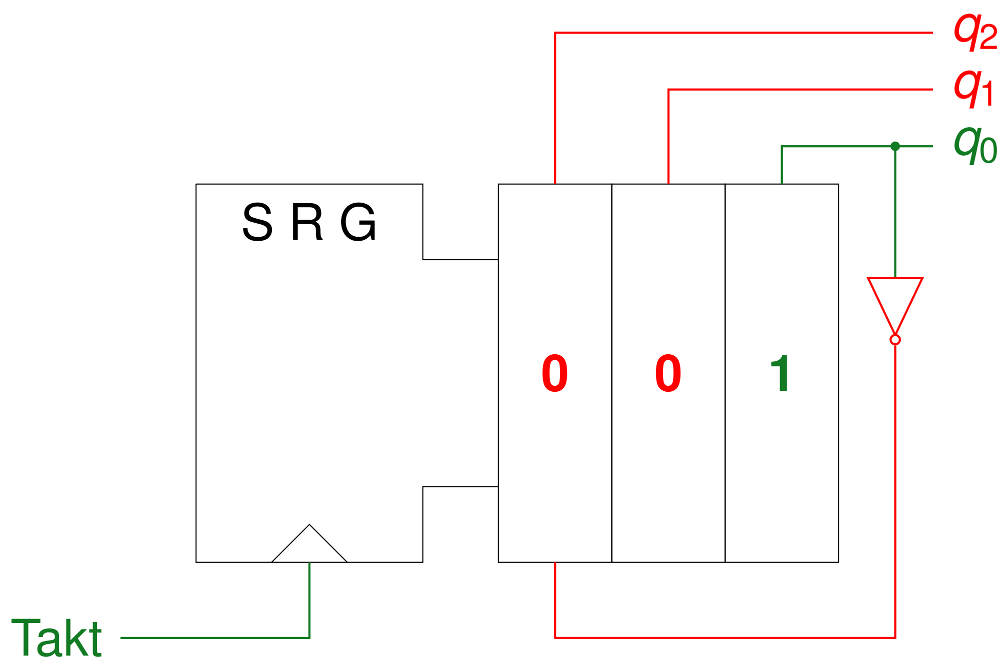
Startwert: 000



$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011 \mapsto 001$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

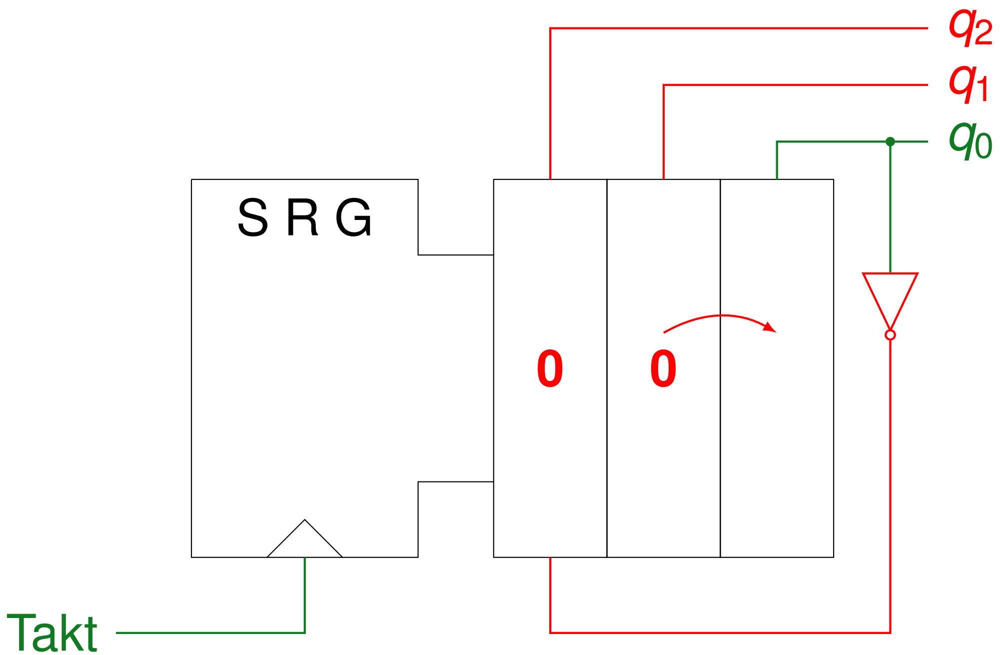
Startwert: 000



$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011 \mapsto 001$

Aufgabe 4 – Schieberegister: Johnson-Zähler

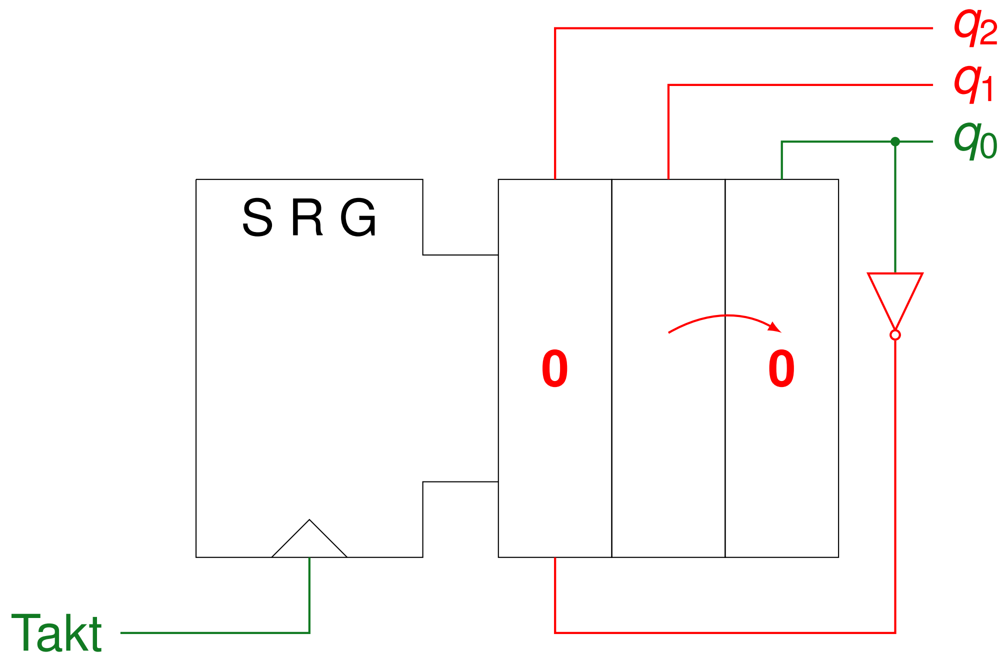
Startwert: 000



$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011 \mapsto 001$

Aufgabe 4 – Schieberegister: Johnson-Zähler

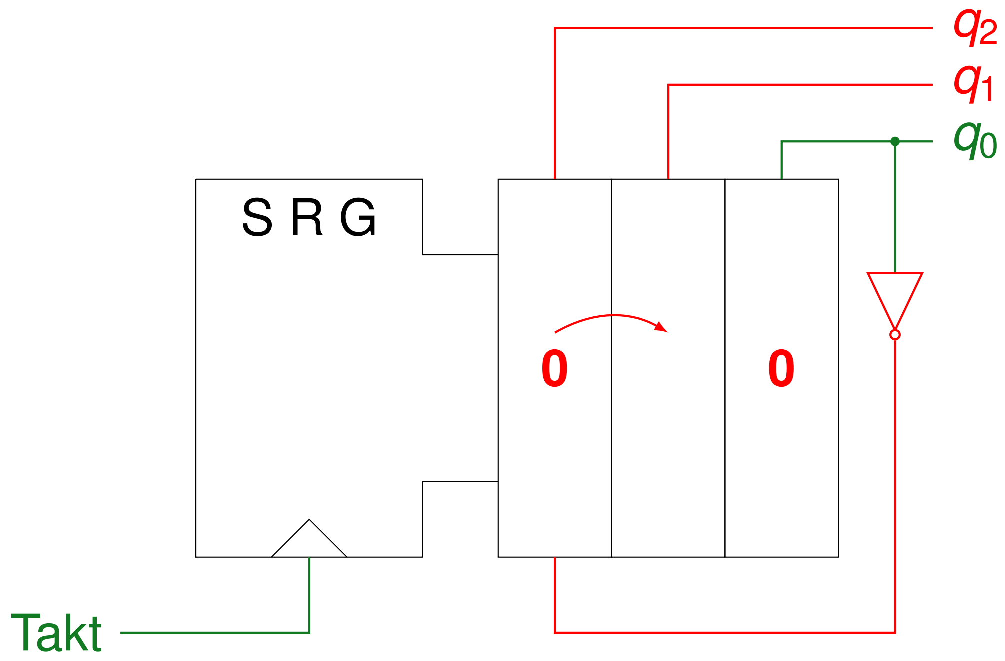
Startwert: 000



$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011 \mapsto 001$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

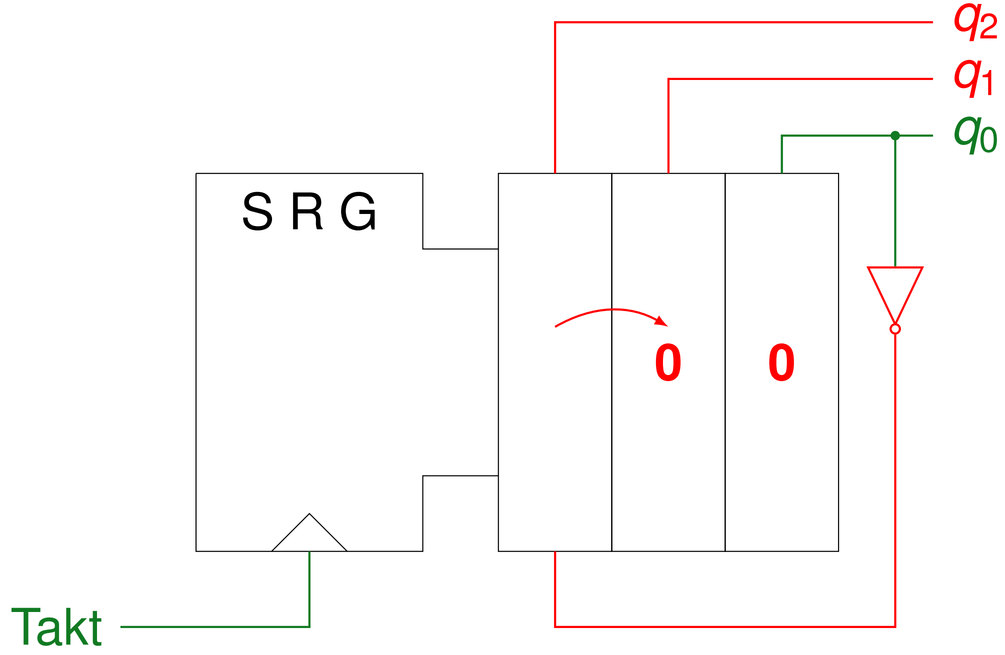
Startwert: 000



$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011 \mapsto 001$

Aufgabe 4 – Schieberegister: Johnson-Zähler

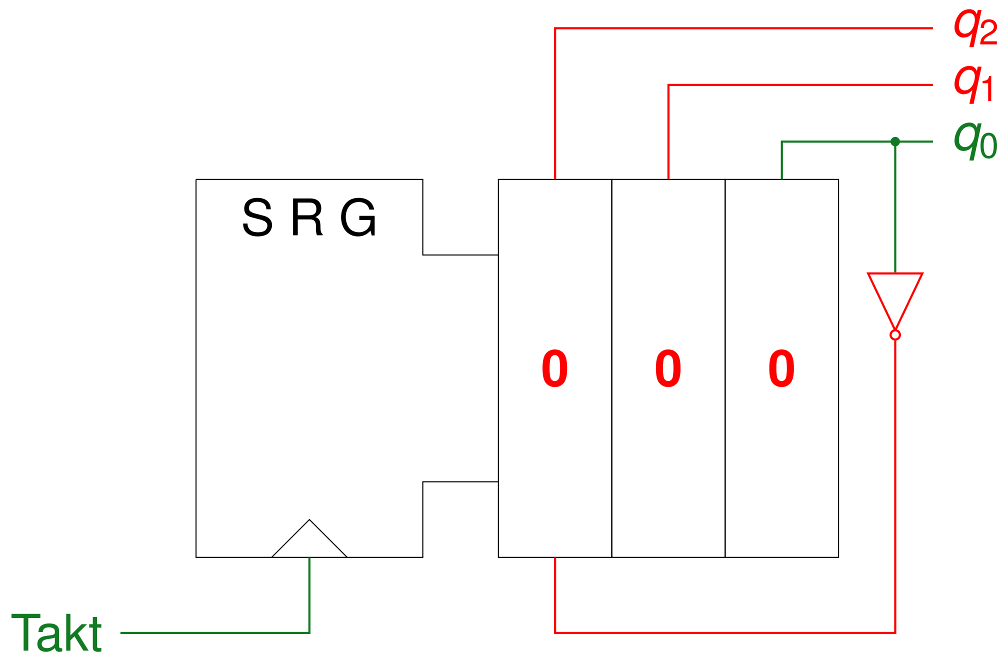
Startwert: 000



$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011 \mapsto 001$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

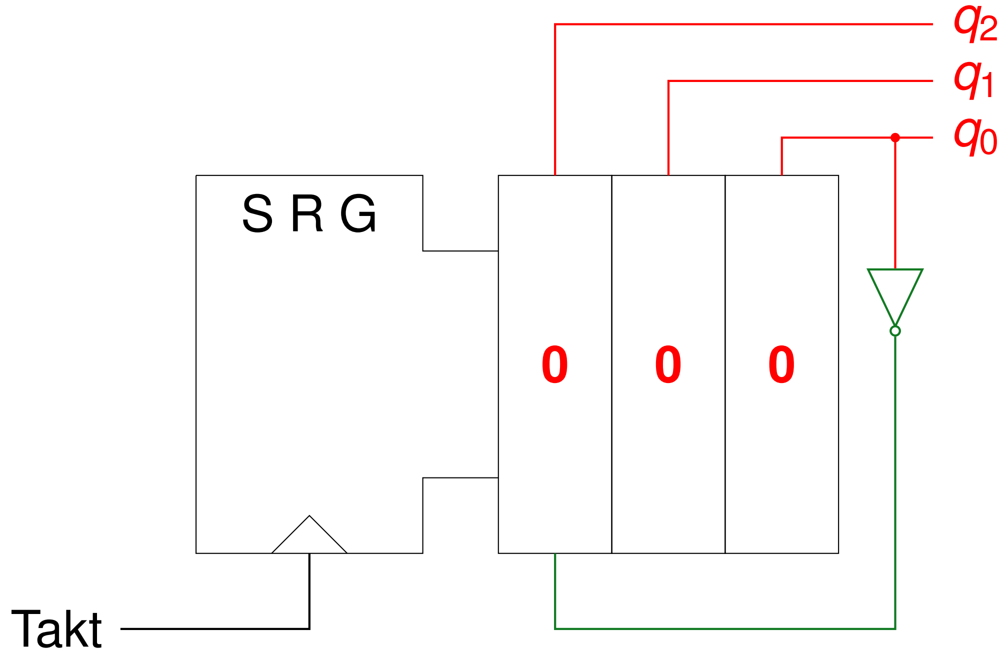
Startwert: 000



$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011 \mapsto 001$$

Aufgabe 4 – Schieberegister: Johnson-Zähler

Startwert: 000



$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011 \mapsto 001 \mapsto 000$$

Aufgabe 4 – Schieberegister

$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011 \mapsto 001 \mapsto 000$$

a)

Aufgabe 4 – Schieberegister

$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011 \mapsto 001 \mapsto 000$$

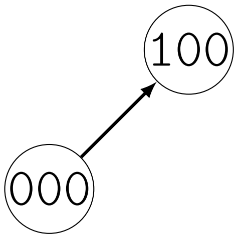
a)

000

Aufgabe 4 – Schieberegister

$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011 \mapsto 001 \mapsto 000$$

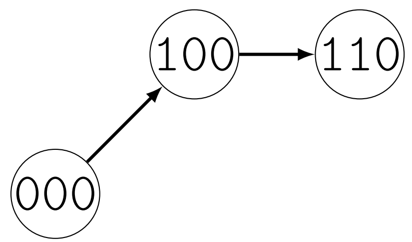
a)



Aufgabe 4 – Schieberegister

$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011 \mapsto 001 \mapsto 000$$

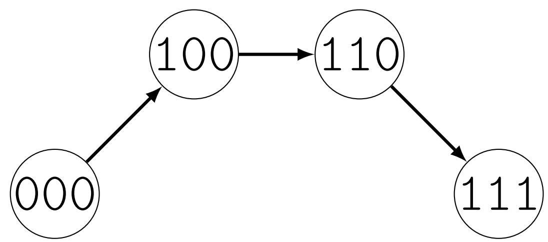
a)



Aufgabe 4 – Schieberegister

$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011 \mapsto 001 \mapsto 000$$

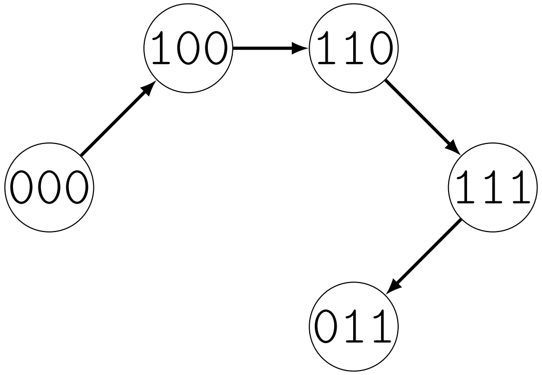
a)



Aufgabe 4 – Schieberegister

$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011 \mapsto 001 \mapsto 000$$

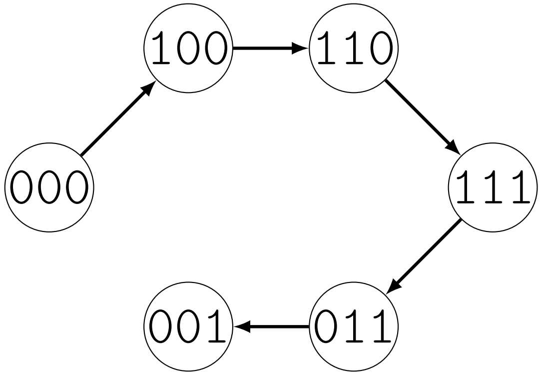
a)



Aufgabe 4 – Schieberegister

$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011 \mapsto 001 \mapsto 000$$

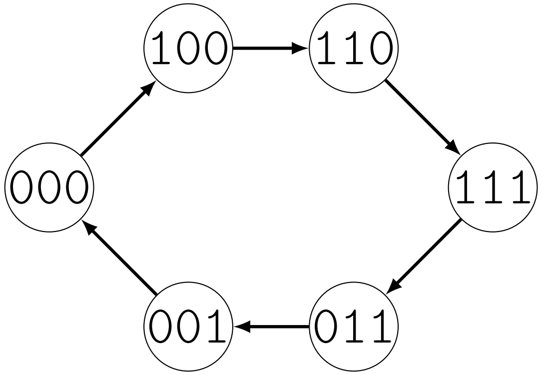
a)



Aufgabe 4 – Schieberegister

$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011 \mapsto 001 \mapsto 000$$

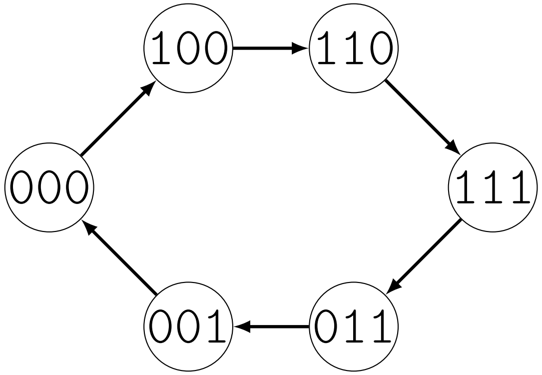
a)



Aufgabe 4 – Schieberegister

$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011 \mapsto 001 \mapsto 000$$

a)

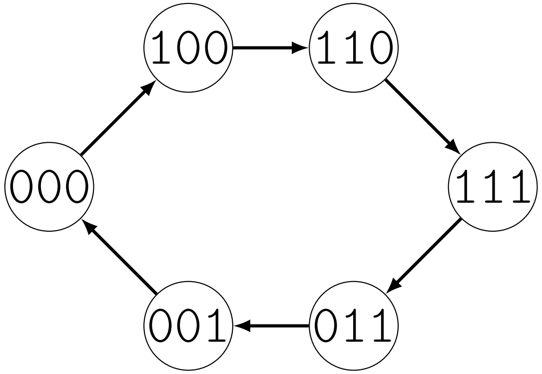


b) Welche Speicherbelegungen kommen in der Folge nicht vor? Vervollständigen Sie den Graphen aus a) entsprechend.

Aufgabe 4 – Schieberegister

$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011 \mapsto 001 \mapsto 000$$

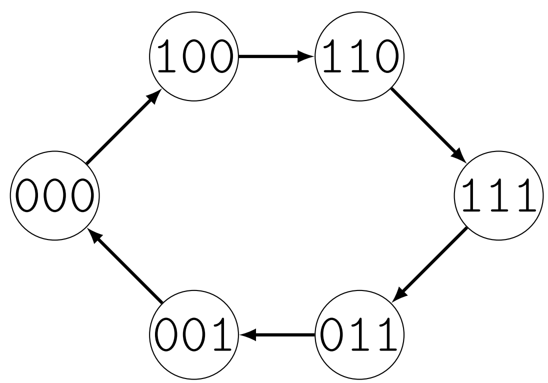
b) Welche Speicherbelegungen kommen in der Folge nicht vor?



Aufgabe 4 – Schieberegister

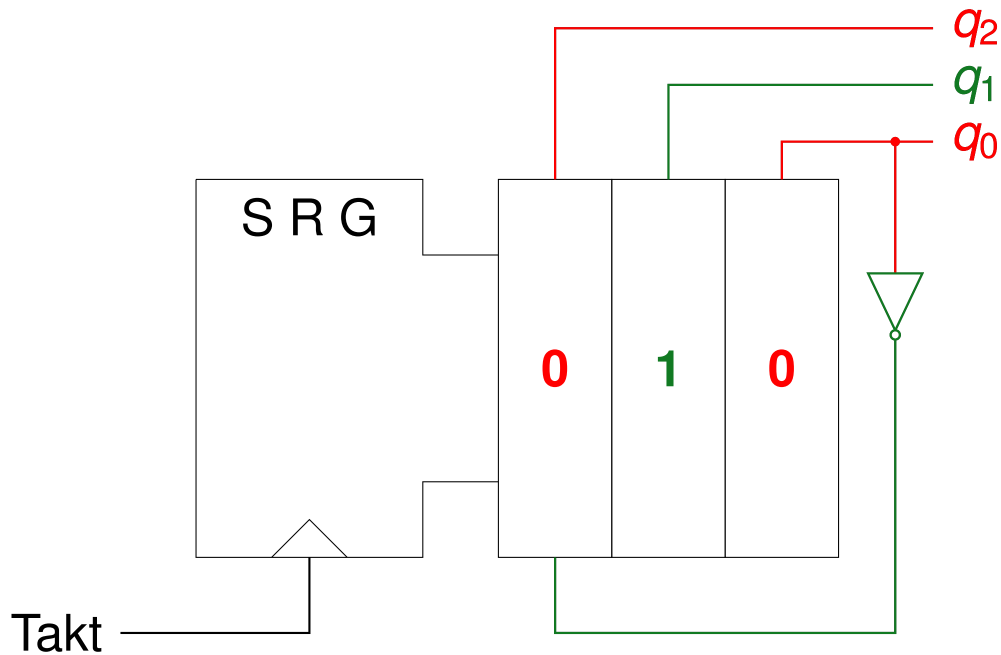
$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011 \mapsto 001 \mapsto 000$$

b) Welche Speicherbelegungen kommen in der Folge nicht vor?
$\rightarrow$ 101 und 010



Aufgabe 4 – Schieberegister: Johnson-Zähler *revisited*

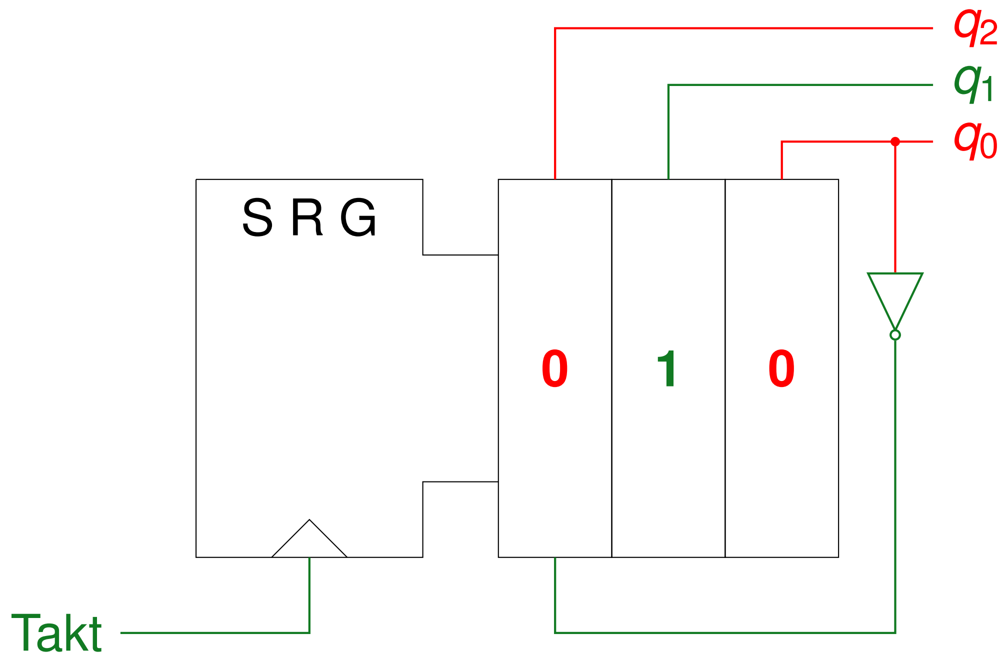
Startwert: `010`



`010`

Aufgabe 4 – Schieberegister: Johnson-Zähler *revisited*

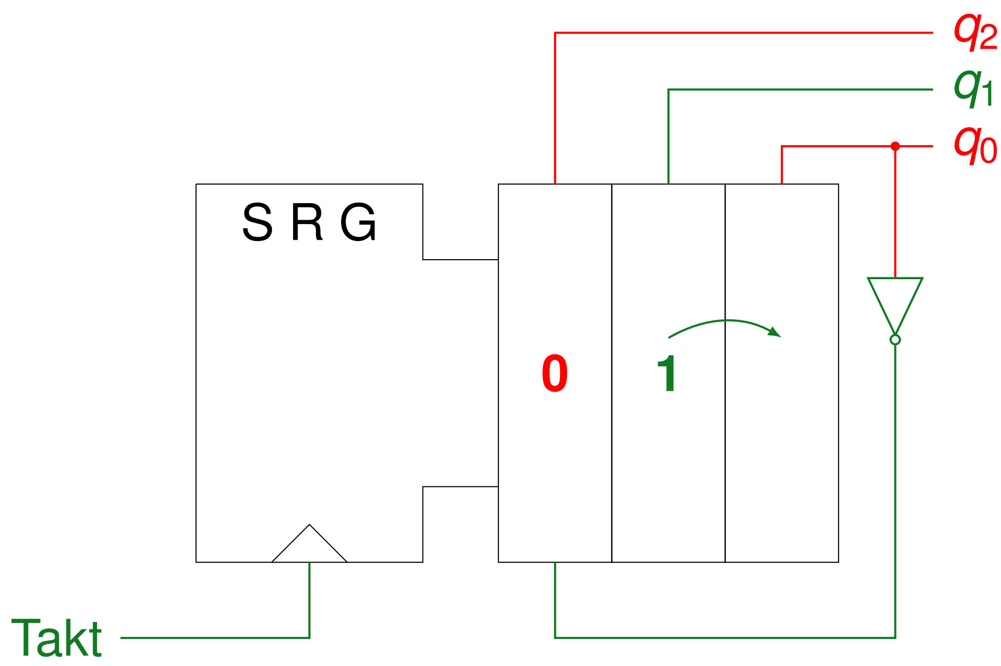
Startwert: 010



010

Aufgabe 4 – Schieberegister: Johnson-Zähler *revisited*

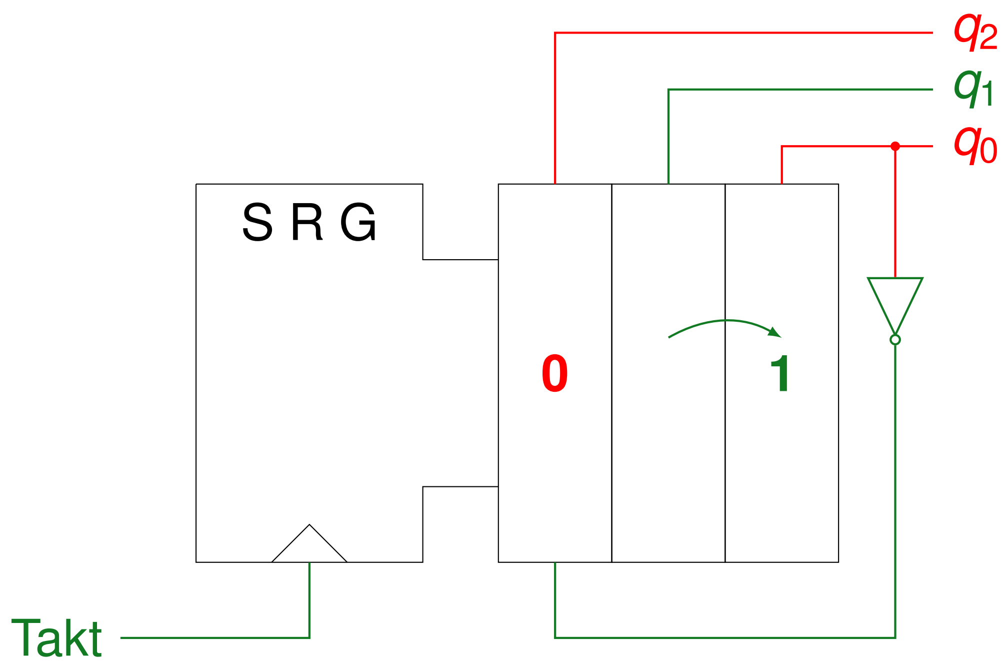
Startwert: 010



010

Aufgabe 4 – Schieberegister: Johnson-Zähler *revisited*

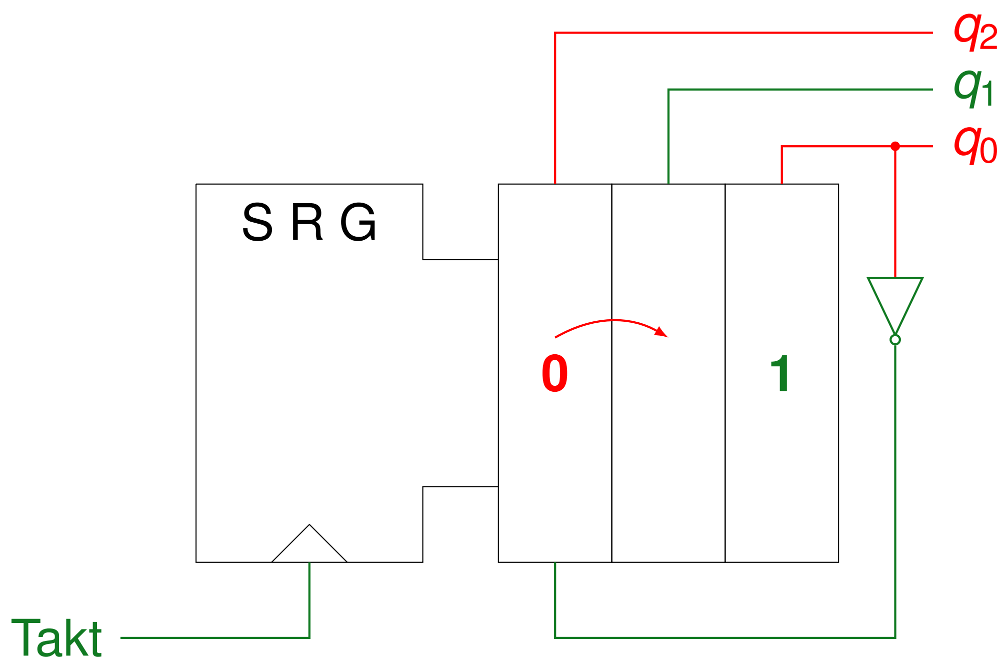
Startwert: `010`



010

Aufgabe 4 – Schieberegister: Johnson-Zähler *revisited*

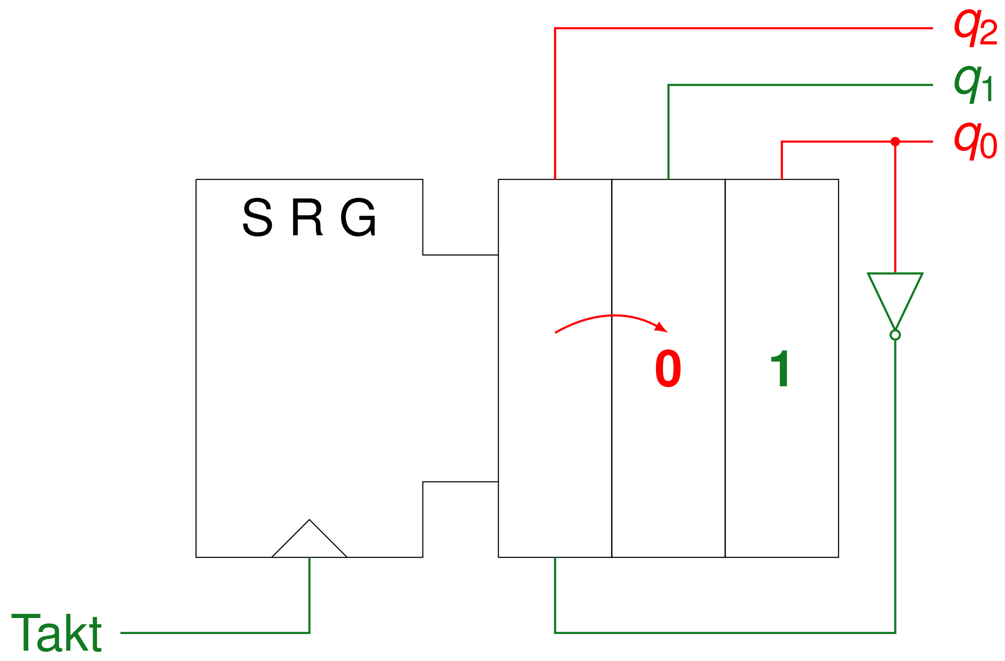
Startwert: 010



010

Aufgabe 4 – Schieberegister: Johnson-Zähler *revisited*

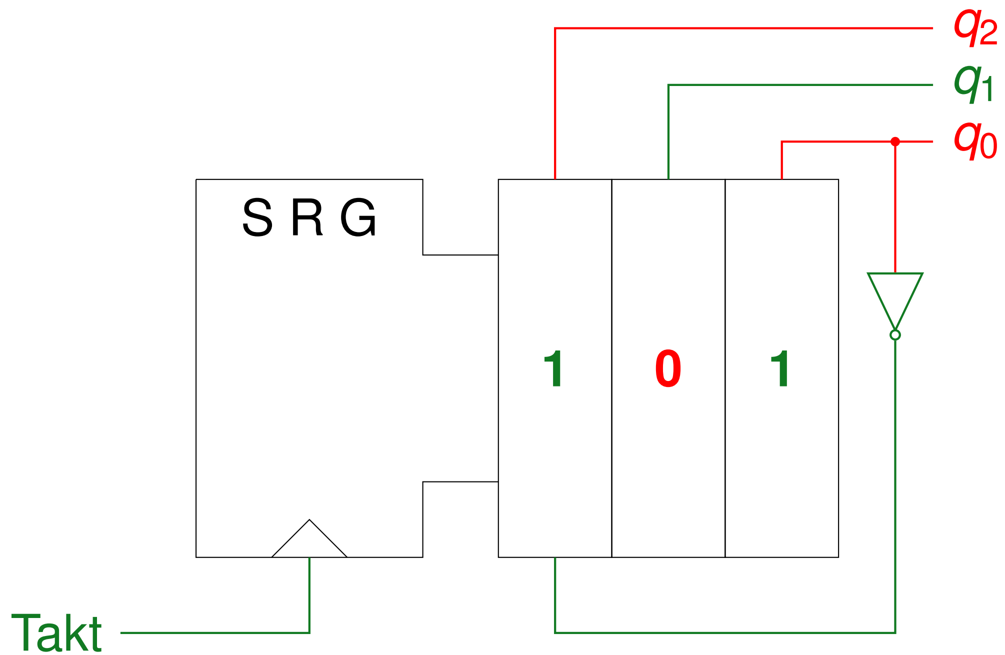
Startwert: 010



010

Aufgabe 4 – Schieberegister: Johnson-Zähler *revisited*

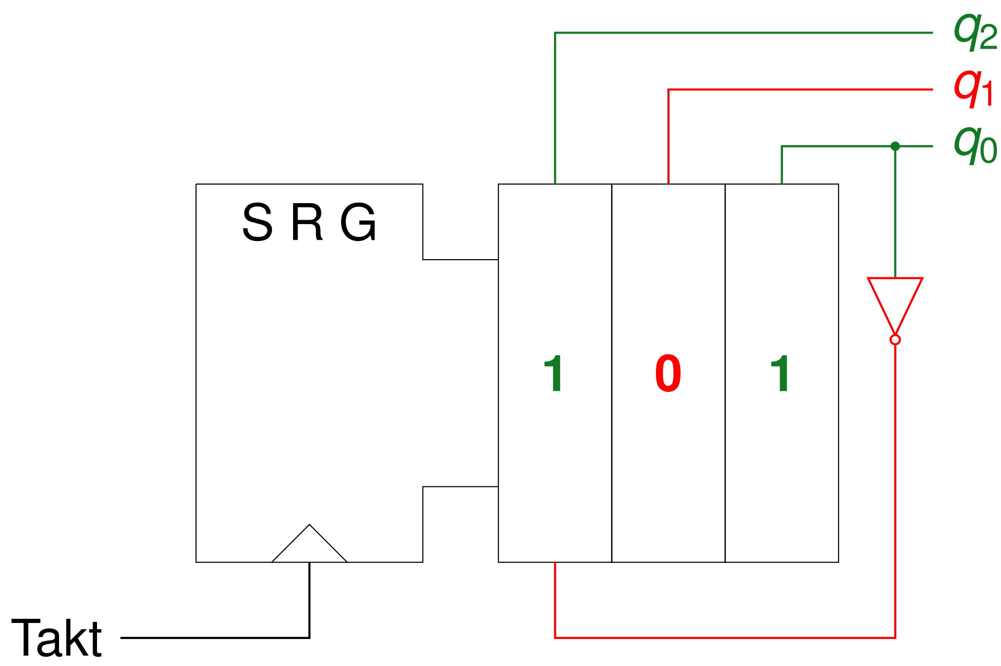
Startwert: 010



010

Aufgabe 4 – Schieberegister: Johnson-Zähler *revisited*

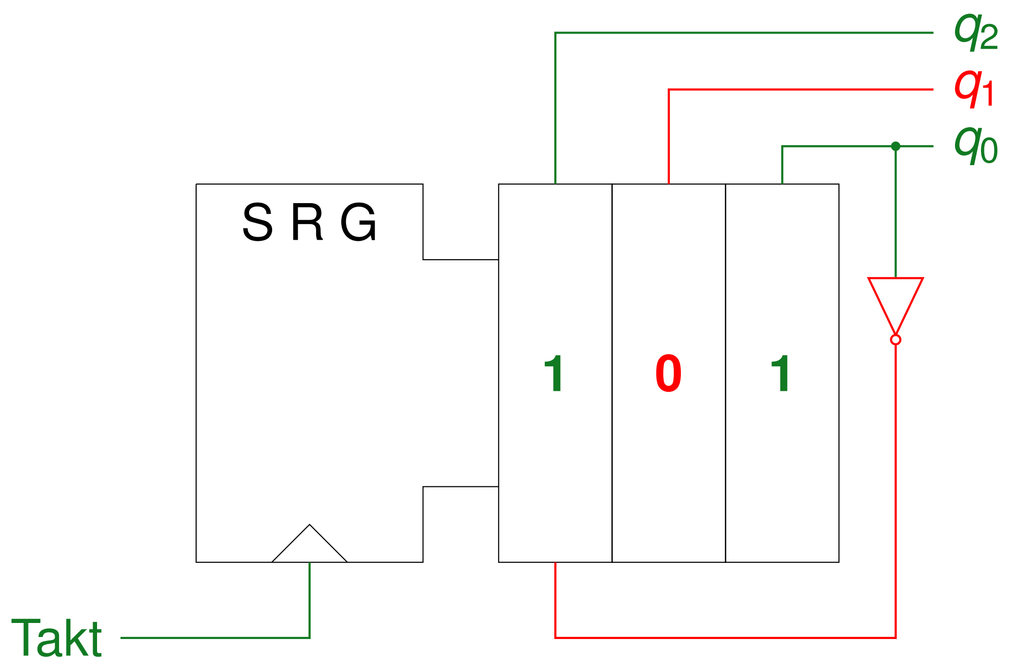
Startwert: 010



$010 \mapsto 101$

Aufgabe 4 – Schieberegister: Johnson-Zähler *revisited*

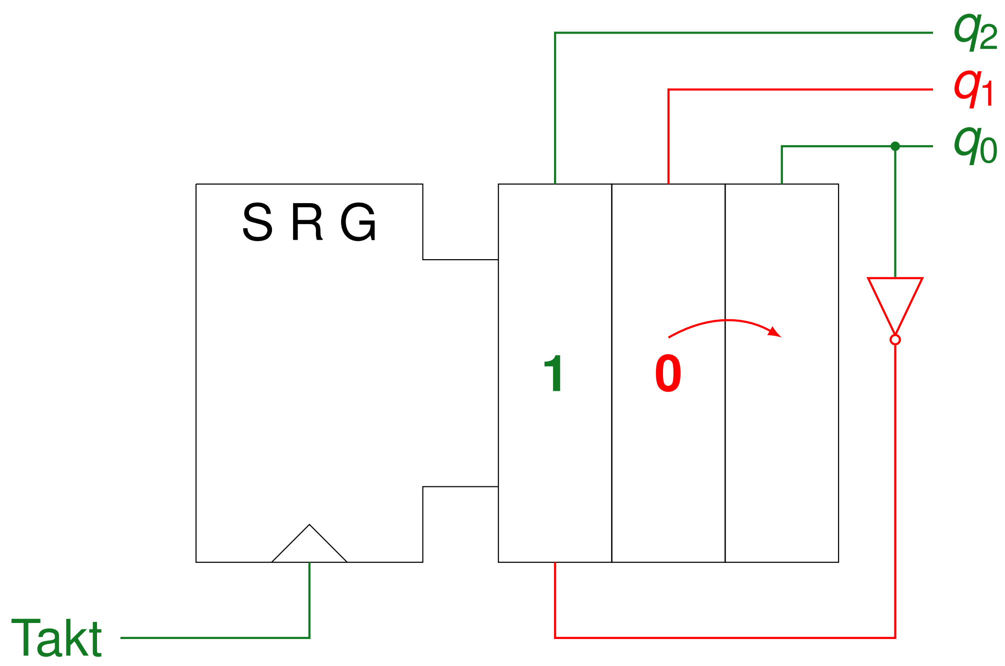
Startwert: 010



$$010 \mapsto 101$$

Aufgabe 4 – Schieberegister: Johnson-Zähler *revisited*

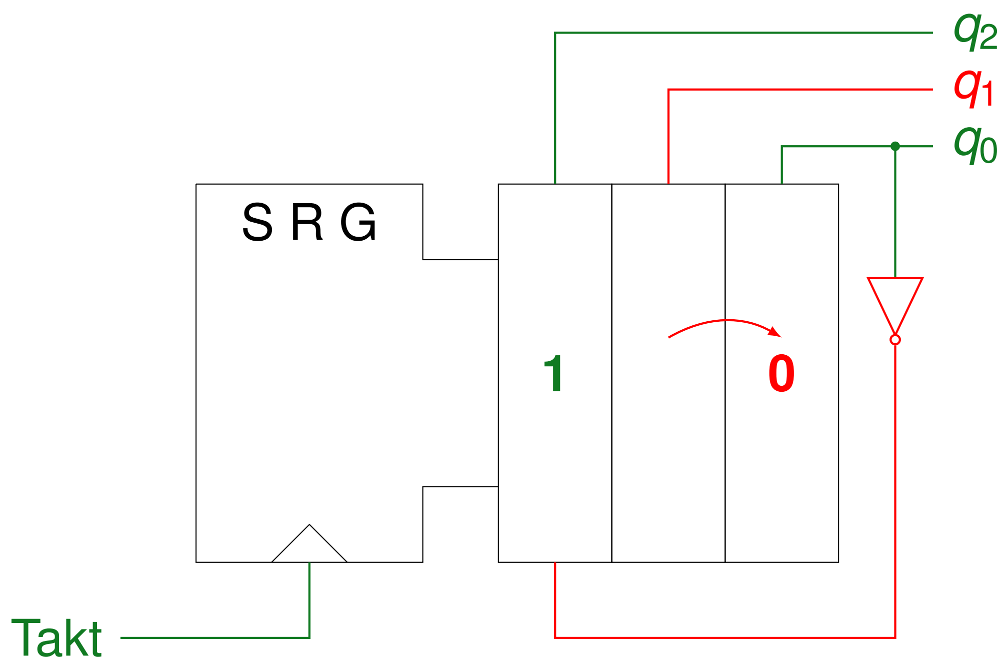
Startwert: 010



$$010 \mapsto 101$$

Aufgabe 4 – Schieberegister: Johnson-Zähler *revisited*

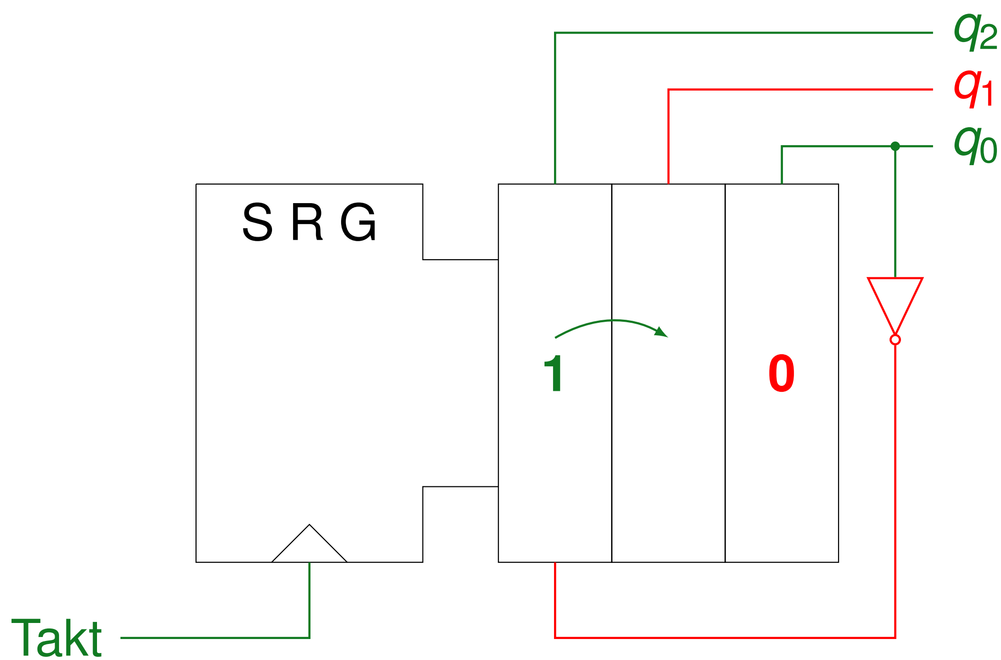
Startwert: 010



$010 \mapsto 101$

Aufgabe 4 – Schieberegister: Johnson-Zähler *revisited*

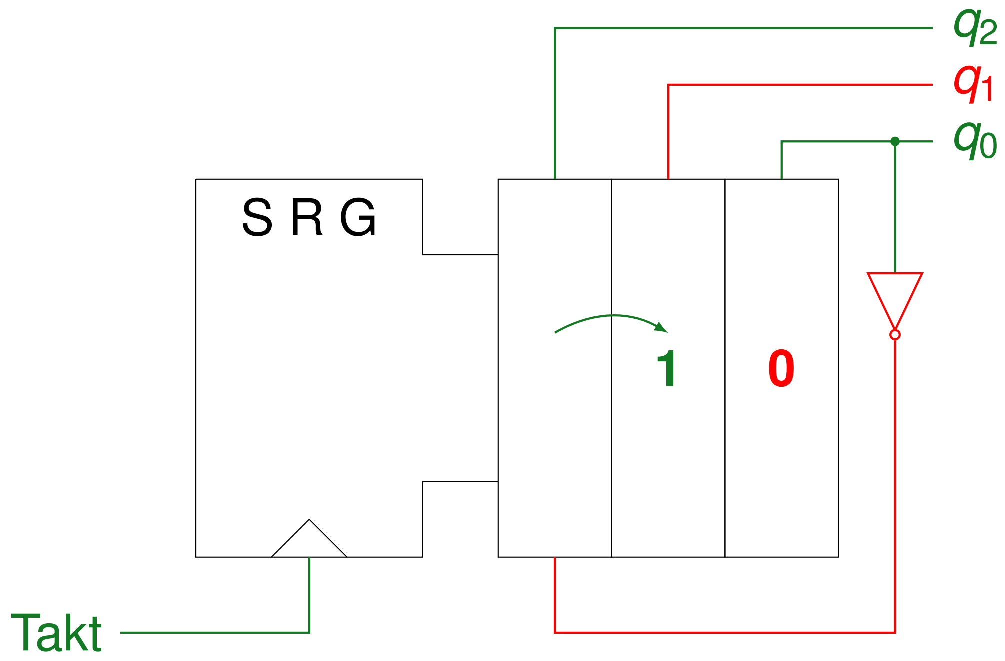
Startwert: 010



$010 \mapsto 101$

Aufgabe 4 – Schieberegister: Johnson-Zähler *revisited*

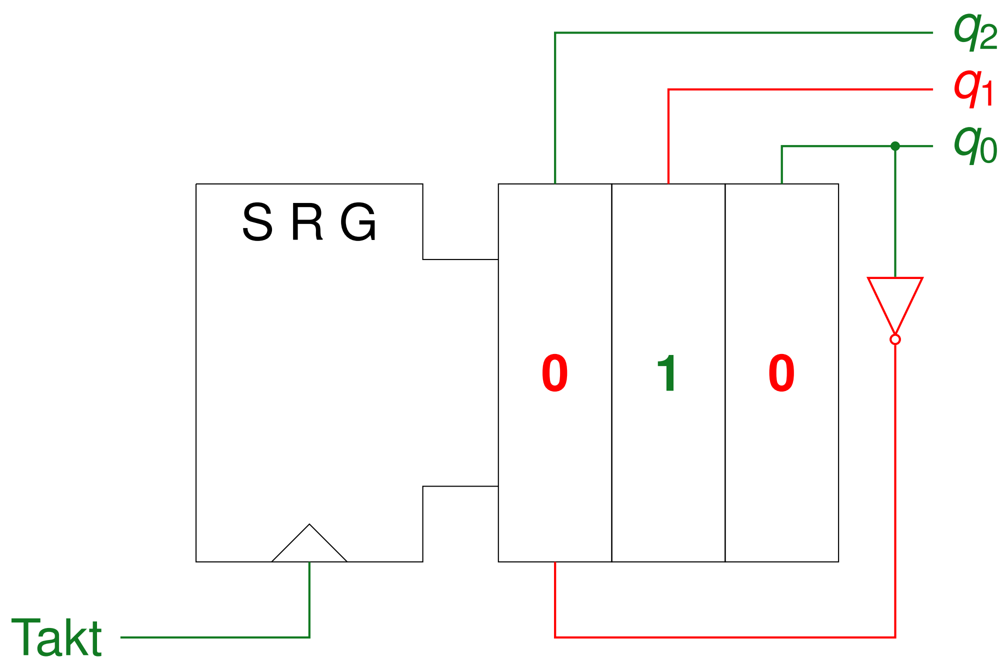
Startwert: 010



$$010 \mapsto 101$$

Aufgabe 4 – Schieberegister: Johnson-Zähler *revisited*

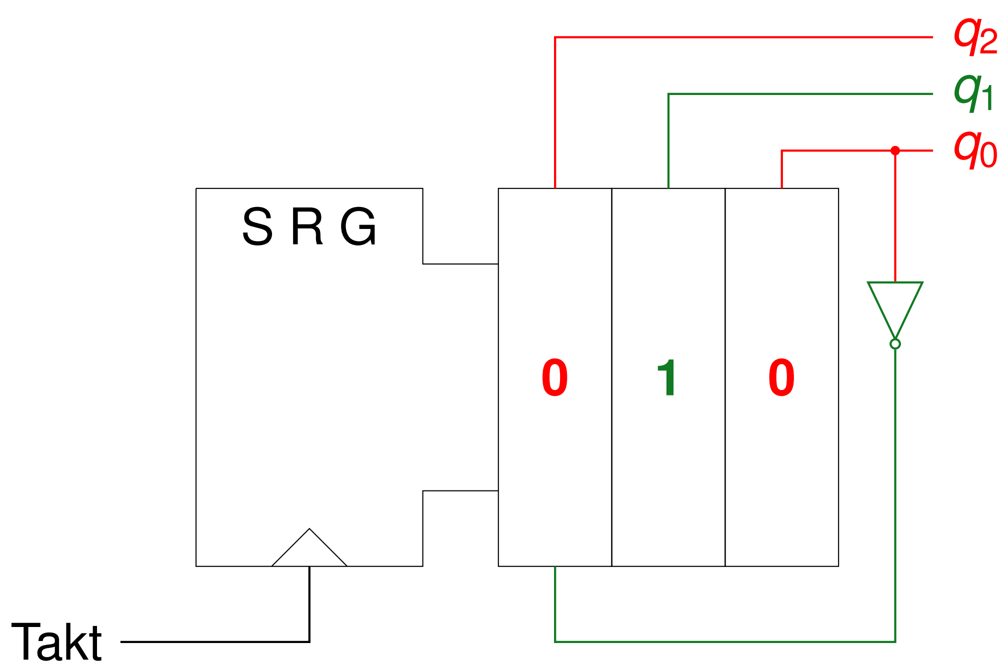
Startwert: 010



$$010 \mapsto 101$$

Aufgabe 4 – Schieberegister: Johnson-Zähler *revisited*

Startwert: 010



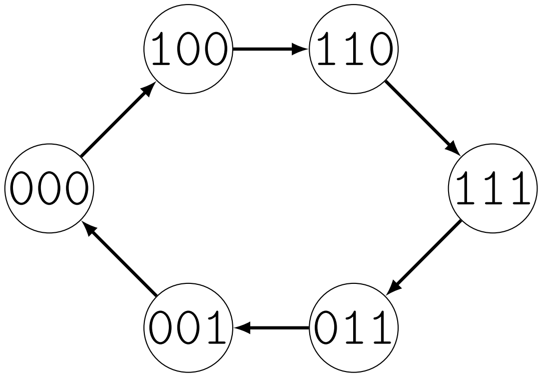
$010 \mapsto 101 \mapsto 010$

Aufgabe 4 – Schieberegister

$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011 \mapsto 001 \mapsto 000$$

$$010 \mapsto 101 \mapsto 010$$

b) Welche Speicherbelegungen kommen in der Folge nicht vor?
$\rightarrow$ 101 und 010



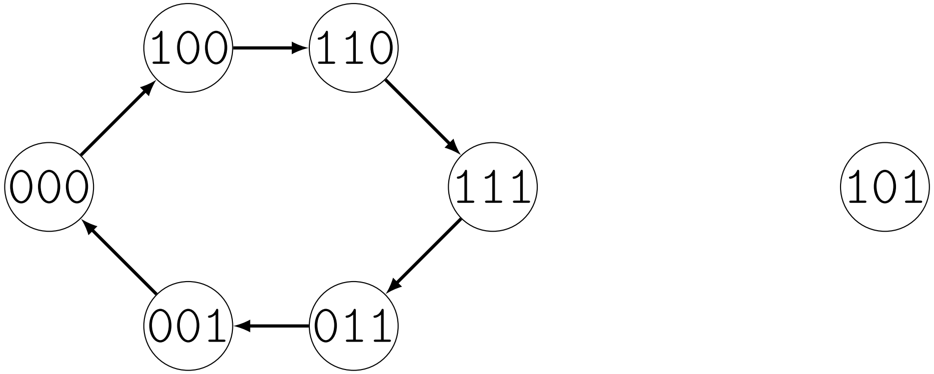
Aufgabe 4 – Schieberegister

$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011 \mapsto 001 \mapsto 000$$

$$010 \mapsto 101 \mapsto 010$$

b) Welche Speicherbelegungen kommen in der Folge nicht vor?

$\rightarrow$ 101 und 010



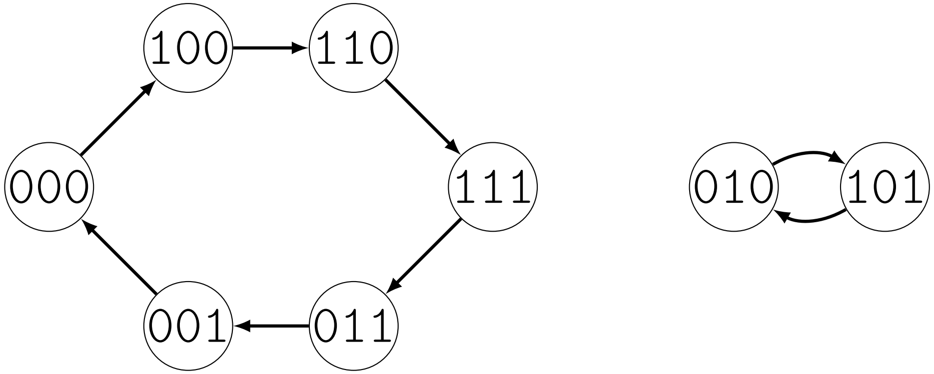
Aufgabe 4 – Schieberegister

$$000 \mapsto 100 \mapsto 110 \mapsto 111 \mapsto 011 \mapsto 001 \mapsto 000$$

$$010 \mapsto 101 \mapsto 010$$

b) Welche Speicherbelegungen kommen in der Folge nicht vor?

$\rightarrow$ 101 und 010



Aufgabe 4 – Schieberegister

c) Realisieren Sie ein Schaltnetz, welches aus allen N Zuständen des Johnson-Zählers aus der Abbildung ein Lauflicht für 6 einzelne Lampen $L_1 \ldots L_6$ ansteuern kann. Sie können dazu auf die Ausgangswerte q_0, q_1, q_2 sowie deren negierte Signale $\overline{q_0}, \overline{q_1}, \overline{q_2}$ zurückgreifen. Beginnen Sie mit einer Wahrheitstabelle.

Aufgabe 4 – Schieberegister

c) Lauflicht $\rightarrow$ Leuchtende Lampe „läuft“ von links nach rechts. Sprich: Nach L_1 leuchtet L_2, dann L_3 ... und nach L_6 leuchtet wieder L_1.

Aufgabe 4 – Schieberegister

c) Lauflicht $\rightarrow$ Leuchtende Lampe „läuft“ von links nach rechts. Sprich: Nach L_1 leuchtet L_2, dann L_3 ... und nach L_6 leuchtet wieder L_1.
Unser Johnson-Zähler hat **genau** 6 zyklische Zustände. Wir verwenden diese also und stellen unsere Wahrheitstabelle auf:

Aufgabe 4 – Schieberegister

c) Lauflicht $\rightarrow$ Leuchtende Lampe „läuft" von links nach rechts. Sprich: Nach L_1 leuchtet L_2, dann L_3 ... und nach L_6 leuchtet wieder L_1.
Unser Johnson-Zähler hat **genau** 6 zyklische Zustände. Wir verwenden diese also und stellen unsere Wahrheitstabelle auf:

q_2	q_1	q_0	L_1	L_2	L_3	L_4	L_5	L_6
0	0	0	1	0	0	0	0	0
1	0	0	0	1	0	0	0	0
1	1	0	0	0	1	0	0	0
1	1	1	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0
0	0	1	0	0	0	0	0	1
0	1	0	-	-	-	-	-	-
1	0	1	-	-	-	-	-	-

Aufgabe 4 – Schieberegister

c) Lauflicht $\rightarrow$ Leuchtende Lampe „läuft“ von links nach rechts. Sprich: Nach L_1 leuchtet L_2, dann L_3 ... und nach L_6 leuchtet wieder L_1.
Unser Johnson-Zähler hat **genau** 6 zyklische Zustände. Wir verwenden diese also und stellen unsere Wahrheitstabelle auf:

q_2	q_1	q_0	L_1	L_2	L_3	L_4	L_5	L_6
0	0	0	1	0	0	0	0	0
1	0	0	0	1	0	0	0	0
1	1	0	0	0	1	0	0	0
1	1	1	0	0	0	1	0	0
0	1	1	0	0	0	0	1	0
0	0	1	0	0	0	0	0	1
0	1	0	-	-	-	-	-	-
1	0	1	-	-	-	-	-	-

Da die Zustände 010 und 101 nie auftreten (siehe Teilaufgabe b), können sie als Freistellen angenommen werden $\rightarrow$ Damit ergeben sich „minimalere“ Funktionen.

Aufgabe 4 – Schieberegister: Teilaufgabe c

Wir minimieren also per Symmetriediagramm:

Dec	q_2	q_1	q_0	L_1	L_2	L_3	L_4	L_5	L_6
0	0	0	0	1	0	0	0	0	0
4	1	0	0	0	1	0	0	0	0
6	1	1	0	0	0	1	0	0	0
7	1	1	1	0	0	0	1	0	0
3	0	1	1	0	0	0	0	1	0
1	0	0	1	0	0	0	0	0	1
2	0	1	0	-	-	-	-	-	-
5	1	0	1	-	-	-	-	-	-

Aufgabe 4 – Schieberegister: Teilaufgabe c

Wir minimieren also per Symmetriediagramm:

L_1:

		q_2	q_2	
	1 ₀	0 ₁	— ₅	0 ₄
q_1	— ₂	0 ₃	0 ₇	0 ₆
			q_0	q_0

Dec	q_2 q_1 q_0	L_1	L_2	L_3	L_4	L_5	L_6
0	0 0 0	1	0	0	0	0	0
4	1 0 0	0	1	0	0	0	0
6	1 1 0	0	0	1	0	0	0
7	1 1 1	0	0	0	1	0	0
3	0 1 1	0	0	0	0	1	0
1	0 0 1	0	0	0	0	0	1
2	0 1 0	-	-	-	-	-	-
5	1 0 1	-	-	-	-	-	-

Aufgabe 4 – Schieberegister: Teilaufgabe c

Wir minimieren also per Symmetriediagramm:

L_1:

		q_2	q_2	
	1 (0)	0 (1)	— (5)	0 (4)
q_1	— (2)	0 (3)	0 (7)	0 (6)
			q_0	q_0

Dec	q_2 q_1 q_0	L_1	L_2	L_3	L_4	L_5	L_6
0	0 0 0	1	0	0	0	0	0
4	1 0 0	0	1	0	0	0	0
6	1 1 0	0	0	1	0	0	0
7	1 1 1	0	0	0	1	0	0
3	0 1 1	0	0	0	0	1	0
1	0 0 1	0	0	0	0	0	1
2	0 1 0	-	-	-	-	-	-
5	1 0 1	-	-	-	-	-	-

Aufgabe 4 – Schieberegister: Teilaufgabe c

Wir minimieren also per Symmetriediagramm:

L_1:

		q_2	q_2	
	$\boxed{1}$ [0]	0 [1]	– [5]	0 [4]
q_1	$\boxed{–}$ [2]	0 [3]	0 [7]	0 [6]
			q_0	q_0

$$\rightarrow L_1 = \overline{q_2}\ \overline{q_0}$$

Dec	q_2 q_1 q_0	L_1	L_2	L_3	L_4	L_5	L_6
0	0 0 0	1	0	0	0	0	0
4	1 0 0	0	1	0	0	0	0
6	1 1 0	0	0	1	0	0	0
7	1 1 1	0	0	0	1	0	0
3	0 1 1	0	0	0	0	1	0
1	0 0 1	0	0	0	0	0	1
2	0 1 0	-	-	-	-	-	-
5	1 0 1	-	-	-	-	-	-

Aufgabe 4 – Schieberegister: Teilaufgabe c

Wir minimieren also per Symmetriediagramm:

L_2:

		q_0		
	0 (0)	0 (1)	— (5)	1 (4)
q_1	— (2)	0 (3)	0 (7)	0 (6)
			q_2	

Dec	q_2 q_1 q_0	L_1	L_2	L_3	L_4	L_5	L_6
0	0 0 0	1	0	0	0	0	0
4	1 0 0	0	1	0	0	0	0
6	1 1 0	0	0	1	0	0	0
7	1 1 1	0	0	0	1	0	0
3	0 1 1	0	0	0	0	1	0
1	0 0 1	0	0	0	0	0	1
2	0 1 0	-	-	-	-	-	-
5	1 0 1	-	-	-	-	-	-

Aufgabe 4 – Schieberegister: Teilaufgabe c

Wir minimieren also per Symmetriediagramm:

L_2:

		q_0	q_0	
	0 $_{0}$	0 $_{1}$	– $_{5}$	1 $_{4}$
q_1	– $_{2}$	0 $_{3}$	0 $_{7}$	0 $_{6}$
			q_2	q_2

Dec	q_2	q_1	q_0	L_1	L_2	L_3	L_4	L_5	L_6
0	0	0	0	1	0	0	0	0	0
4	1	0	0	0	1	0	0	0	0
6	1	1	0	0	0	1	0	0	0
7	1	1	1	0	0	0	1	0	0
3	0	1	1	0	0	0	0	1	0
1	0	0	1	0	0	0	0	0	1
2	0	1	0	-	-	-	-	-	-
5	1	0	1	-	-	-	-	-	-

Aufgabe 4 – Schieberegister: Teilaufgabe c

Wir minimieren also per Symmetriediagramm:

L_2:

		q_0	q_0	
	0 $_0$	0 $_1$	– $_5$	1 $_4$
q_1	– $_2$	0 $_3$	0 $_7$	0 $_6$
			q_2	q_2

$$\rightarrow L_2 = q_2 \; \overline{q_1}$$

Dec	q_2	q_1	q_0	L_1	L_2	L_3	L_4	L_5	L_6
0	0	0	0	1	0	0	0	0	0
4	1	0	0	0	1	0	0	0	0
6	1	1	0	0	0	1	0	0	0
7	1	1	1	0	0	0	1	0	0
3	0	1	1	0	0	0	0	1	0
1	0	0	1	0	0	0	0	0	1
2	0	1	0	-	-	-	-	-	-
5	1	0	1	-	-	-	-	-	-

Aufgabe 4 – Schieberegister: Teilaufgabe c

Wir minimieren also per Symmetriediagramm:

L_3:

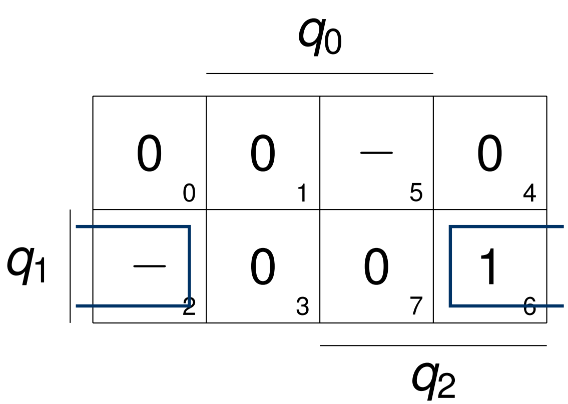
		q_0	q_0	
	0 (0)	0 (1)	— (5)	0 (4)
q_1	— (2)	0 (3)	0 (7)	1 (6)
			q_2	q_2

Dec	q_2 q_1 q_0	L_1	L_2	L_3	L_4	L_5	L_6
0	0 0 0	1	0	0	0	0	0
4	1 0 0	0	1	0	0	0	0
6	1 1 0	0	0	1	0	0	0
7	1 1 1	0	0	0	1	0	0
3	0 1 1	0	0	0	0	1	0
1	0 0 1	0	0	0	0	0	1
2	0 1 0	-	-	-	-	-	-
5	1 0 1	-	-	-	-	-	-

Aufgabe 4 – Schieberegister: Teilaufgabe c

Wir minimieren also per Symmetriediagramm:

L_3:



Dec	q_2 q_1 q_0	L_1	L_2	L_3	L_4	L_5	L_6
0	0 0 0	1	0	0	0	0	0
4	1 0 0	0	1	0	0	0	0
6	1 1 0	0	0	1	0	0	0
7	1 1 1	0	0	0	1	0	0
3	0 1 1	0	0	0	0	1	0
1	0 0 1	0	0	0	0	0	1
2	0 1 0	-	-	-	-	-	-
5	1 0 1	-	-	-	-	-	-

Aufgabe 4 – Schieberegister: Teilaufgabe c

Wir minimieren also per Symmetriediagramm:

L_3:

		q_0	q_0	
	0 (0)	0 (1)	– (5)	0 (4)
q_1	– (2)	0 (3)	0 (7)	1 (6)
			q_2	q_2

$\rightarrow L_3 = q_1 \; \overline{q_0}$

Dec	q_2	q_1	q_0	L_1	L_2	L_3	L_4	L_5	L_6
0	0	0	0	1	0	0	0	0	0
4	1	0	0	0	1	0	0	0	0
6	1	1	0	0	0	1	0	0	0
7	1	1	1	0	0	0	1	0	0
3	0	1	1	0	0	0	0	1	0
1	0	0	1	0	0	0	0	0	1
2	0	1	0	-	-	-	-	-	-
5	1	0	1	-	-	-	-	-	-

Aufgabe 4 – Schieberegister: Teilaufgabe c

Wir minimieren also per Symmetriediagramm:

L_4:

		q_0	q_0	
	0_{0}	0_{1}	$-_{5}$	0_{4}
q_1	$-_{2}$	0_{3}	1_{7}	0_{6}
			q_2	q_2

Dec	q_2 q_1 q_0	L_1	L_2	L_3	L_4	L_5	L_6
0	0 0 0	1	0	0	0	0	0
4	1 0 0	0	1	0	0	0	0
6	1 1 0	0	0	1	0	0	0
7	1 1 1	0	0	0	1	0	0
3	0 1 1	0	0	0	0	1	0
1	0 0 1	0	0	0	0	0	1
2	0 1 0	-	-	-	-	-	-
5	1 0 1	-	-	-	-	-	-

Aufgabe 4 – Schieberegister: Teilaufgabe c

Wir minimieren also per Symmetriediagramm:

L_4:

		q_0	q_0	
	0_{0}	0_{1}	$-_{5}$	0_{4}
q_1	$-_{2}$	0_{3}	1_{7}	0_{6}
			q_2	q_2

Dec	q_2 q_1 q_0	L_1	L_2	L_3	L_4	L_5	L_6
0	0 0 0	1	0	0	0	0	0
4	1 0 0	0	1	0	0	0	0
6	1 1 0	0	0	1	0	0	0
7	1 1 1	0	0	0	1	0	0
3	0 1 1	0	0	0	0	1	0
1	0 0 1	0	0	0	0	0	1
2	0 1 0	-	-	-	-	-	-
5	1 0 1	-	-	-	-	-	-

Aufgabe 4 – Schieberegister: Teilaufgabe c

Wir minimieren also per Symmetriediagramm:

L_4:

		q_0	q_0	
	0 (0)	0 (1)	– (5)	0 (4)
q_1	– (2)	0 (3)	1 (7)	0 (6)
			q_2	q_2

$\rightarrow L_4 = q_2 \; q_0$

Dec	q_2 q_1 q_0	L_1	L_2	L_3	L_4	L_5	L_6
0	0 0 0	1	0	0	0	0	0
4	1 0 0	0	1	0	0	0	0
6	1 1 0	0	0	1	0	0	0
7	1 1 1	0	0	0	1	0	0
3	0 1 1	0	0	0	0	1	0
1	0 0 1	0	0	0	0	0	1
2	0 1 0	-	-	-	-	-	-
5	1 0 1	-	-	-	-	-	-

Aufgabe 4 – Schieberegister: Teilaufgabe c

Wir minimieren also per Symmetriediagramm:

L_5:

		q_0	q_0	
	0_{0}	0_{1}	$-_{5}$	0_{4}
q_1	$-_{2}$	1_{3}	0_{7}	0_{6}
			q_2	q_2

Dec	q_2	q_1	q_0	L_1	L_2	L_3	L_4	L_5	L_6
0	0	0	0	1	0	0	0	0	0
4	1	0	0	0	1	0	0	0	0
6	1	1	0	0	0	1	0	0	0
7	1	1	1	0	0	0	1	0	0
3	0	1	1	0	0	0	0	1	0
1	0	0	1	0	0	0	0	0	1
2	0	1	0	-	-	-	-	-	-
5	1	0	1	-	-	-	-	-	-

Aufgabe 4 – Schieberegister: Teilaufgabe c

Wir minimieren also per Symmetriediagramm:

L_5:

		q_0	q_0	
	0_{0}	0_{1}	$-_{5}$	0_{4}
q_1	$-_{2}$	1_{3}	0_{7}	0_{6}
			q_2	q_2

Dec	q_2	q_1	q_0	L_1	L_2	L_3	L_4	L_5	L_6
0	0	0	0	1	0	0	0	0	0
4	1	0	0	0	1	0	0	0	0
6	1	1	0	0	0	1	0	0	0
7	1	1	1	0	0	0	1	0	0
3	0	1	1	0	0	0	0	1	0
1	0	0	1	0	0	0	0	0	1
2	0	1	0	-	-	-	-	-	-
5	1	0	1	-	-	-	-	-	-

Aufgabe 4 – Schieberegister: Teilaufgabe c

Wir minimieren also per Symmetriediagramm:

L_5:

		q_0	q_0	
	0 (0)	0 (1)	– (5)	0 (4)
q_1	– (2)	1 (3)	0 (7)	0 (6)
			q_2	q_2

$$\rightarrow L_5 = \overline{q_2}\ q_1$$

Dec	q_2	q_1	q_0	L_1	L_2	L_3	L_4	L_5	L_6
0	0	0	0	1	0	0	0	0	0
4	1	0	0	0	1	0	0	0	0
6	1	1	0	0	0	1	0	0	0
7	1	1	1	0	0	0	1	0	0
3	0	1	1	0	0	0	0	1	0
1	0	0	1	0	0	0	0	0	1
2	0	1	0	-	-	-	-	-	-
5	1	0	1	-	-	-	-	-	-

Aufgabe 4 – Schieberegister: Teilaufgabe c

Wir minimieren also per Symmetriediagramm:

L_6:

		q_0	q_0	
	0 (0)	1 (1)	– (5)	0 (4)
q_1	– (2)	0 (3)	0 (7)	0 (6)
			q_2	q_2

Dec	q_2 q_1 q_0	L_1	L_2	L_3	L_4	L_5	L_6
0	0 0 0	1	0	0	0	0	0
4	1 0 0	0	1	0	0	0	0
6	1 1 0	0	0	1	0	0	0
7	1 1 1	0	0	0	1	0	0
3	0 1 1	0	0	0	0	1	0
1	0 0 1	0	0	0	0	0	1
2	0 1 0	-	-	-	-	-	-
5	1 0 1	-	-	-	-	-	-

Aufgabe 4 – Schieberegister: Teilaufgabe c

Wir minimieren also per Symmetriediagramm:

L_6:

		q_0	q_0		
	0 (0)	1 (1)	– (5)	0 (4)	
q_1	– (2)	0 (3)	0 (7)	0 (6)	
			q_2	q_2	

Dec	q_2	q_1	q_0	L_1	L_2	L_3	L_4	L_5	L_6
0	0	0	0	1	0	0	0	0	0
4	1	0	0	0	1	0	0	0	0
6	1	1	0	0	0	1	0	0	0
7	1	1	1	0	0	0	1	0	0
3	0	1	1	0	0	0	0	1	0
1	0	0	1	0	0	0	0	0	1
2	0	1	0	-	-	-	-	-	-
5	1	0	1	-	-	-	-	-	-

Aufgabe 4 – Schieberegister: Teilaufgabe c

Wir minimieren also per Symmetriediagramm:

L_6:

		q_0	q_0	
	0 (0)	1 (1)	– (5)	0 (4)
q_1	– (2)	0 (3)	0 (7)	0 (6)
			q_2	q_2

$\rightarrow L_6 = \overline{q_1}\ q_0$

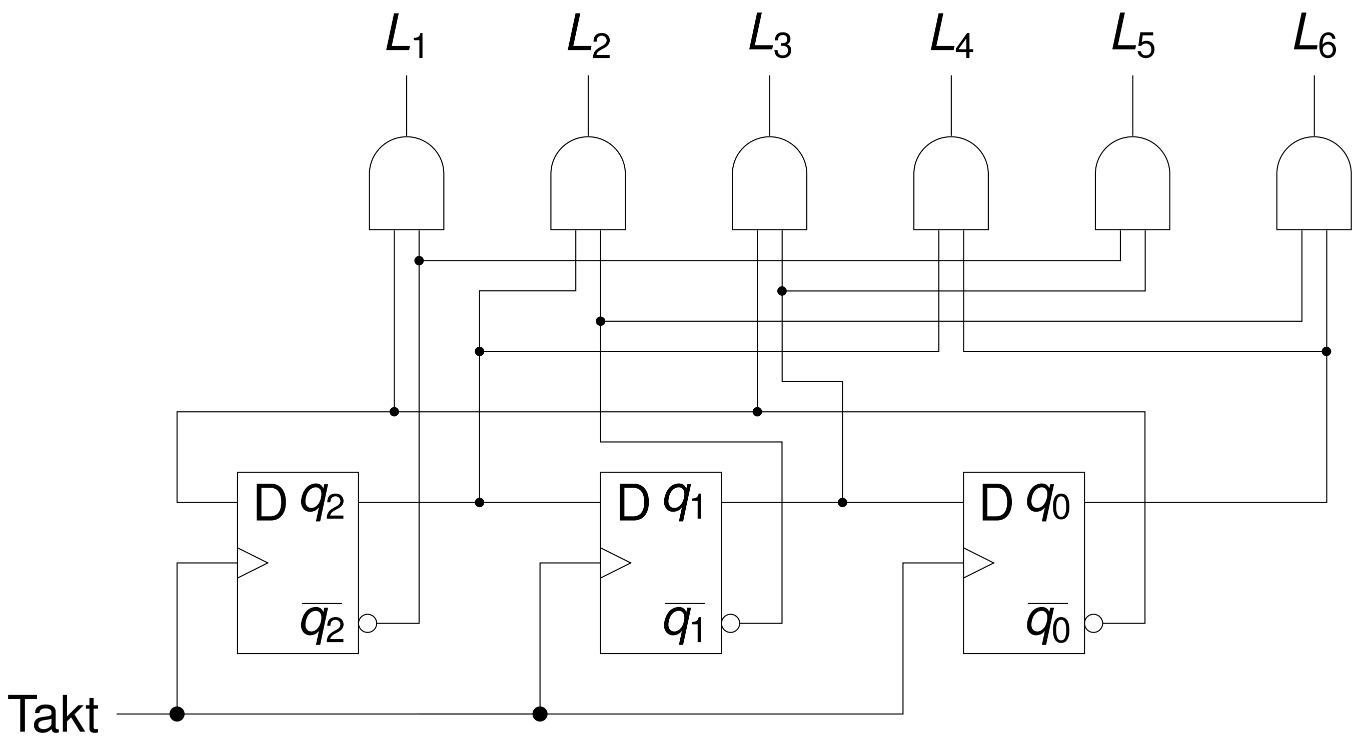
Dec	q_2	q_1	q_0	L_1	L_2	L_3	L_4	L_5	L_6
0	0	0	0	1	0	0	0	0	0
4	1	0	0	0	1	0	0	0	0
6	1	1	0	0	0	1	0	0	0
7	1	1	1	0	0	0	1	0	0
3	0	1	1	0	0	0	0	1	0
1	0	0	1	0	0	0	0	0	1
2	0	1	0	-	-	-	-	-	-
5	1	0	1	-	-	-	-	-	-

Aufgabe 4 – Schieberegister: Teilaufgabe c

Durch Minimierung erhalten wir:

$$L_1 = \overline{q_2} \cdot \overline{q_0}, \quad L_2 = q_2 \cdot \overline{q_1}, \quad L_3 = q_1 \cdot \overline{q_0}, \quad L_4 = q_2 \cdot q_0, \quad L_5 = \overline{q_2} \cdot q_1, \quad L_6 = \overline{q_1} \cdot q_0.$$

Damit können wir alles in einem Schaltwerk zusammenschalten:



Übungen zur Grundlagen der Technischen Informatik

Übung 11 – Automaten

Florian Frank

Friedrich-Alexander-Universität Erlangen-Nürnberg

Wintersemester 2018/19

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Was machen wir heute?

Organisatorisches: Vorlesungsevaluation

Aufgabe 1 – Automaten

Aufgabe 2 – Flipflops und Automaten

Aufgabe 3 – Synchrones Schaltwerk

Organisatorisches: Vorlesungsevaluation

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Organisatorisches – Vorlesungsevaluation

Bitte evaluiert die Veranstaltung, nur so können Dinge verbessert werden!
Auch bei keinen Verbesserungsvorschlägen freuen wir uns immer über positives Feedback, damit wir sehen, dass alles so gepasst hat!

- Bei Kommentaren in Freitextfeldern, die sich auf einen **bestimmten** Übungsleiter beziehen, gebt bitte **dessen Name bei diesen Kommentaren** mit an.
  - ⇝ Kommentare werden durcheinandergewürfelt ...
    — Das heißt nicht nur einmal den Namen angeben, sondern immer!
  - ⇝ Ihr evaluiert die Gesamtveranstaltung „Übungen zu den Grundlagen der Technischen Informatik“ und nicht – wie in AuD oder GRa bspw. – die einzelnen Übungen, deswegen gebt bitte die Namen mit an, wir wären euch sehr verbunden.

Ihr habt noch bis zum **26.01 um $12^{\underline{00}}$ Uhr** Zeit zu evaluieren!

Aufgabe 1 – Automaten

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Aufgabe 1 – Automaten

Entwerfen Sie einen endlichen Zustandsautomat (FSM) für eine Armbanduhr, der eines von vier internen Registern auf dem Display anzeigt. Die Auswahl des Registers erfolgt durch einen 4:1-Multiplexer, dessen Kontrolleingänge mit s_0 und s_1 bezeichnet werden. Die Register entsprechen den aktuellen Werten der Uhrzeit ($s_1 s_0 = 00$), der Alarmeinstellung ($s_1 s_0 = 01$), des Datums (10) und der Stoppuhr (11).
Durch wiederholtes Drücken des Knopfes b soll es möglich sein, die vier Register in der oben genannten Reihenfolge zyklisch auszulesen. Gehen Sie davon aus, dass durch Drücken des Knopfes der Wert von b synchron zum Takt für eine Taktperiode auf 1 gesetzt wird. Zusätzlich soll der Wechsel des Registers durch einen hörbaren Ton angezeigt werden, indem der Ausgang p bei jedem Drücken des Knopfes für eine Taktperiode auf 1 gesetzt wird.

a) Modellieren Sie den Zustandsautomat als Moore-Automat.
b) Modellieren Sie den Zustandsautomat als Mealy-Automat.
c) Welche Vorteile bietet die Realisierung des Zustandsautomats als Mealy-Automat und welche potentiellen Probleme müssen beachtet werden?

Aufgabe 1 – Automaten: Begriffsklärung

Schaltnetz

Die Ausgabe eines Schaltnetzes hängt **alleinig** von der zugehörigen **Eingabe** ab.

Aufgabe 1 – Automaten: Begriffsklärung

Schaltwerk

Die Ausgabe eines Schaltwerkes hängt **mitunter** von dem aktuellen **Zustand** ab.

Aufgabe 1 – Automaten: Begriffsklärung

Schaltnetz

Die Ausgabe eines Schaltnetzes hängt **alleinig** von der zugehörigen **Eingabe** ab.

Schaltwerk

Die Ausgabe eines Schaltwerkes hängt **mitunter** von dem aktuellen **Zustand** ab.

Aufgabe 1 – Automaten: Begriffsklärung

Schaltnetz

Die Ausgabe eines Schaltnetzes hängt **alleinig** von der zugehörigen **Eingabe** ab.

Schaltwerk

Die Ausgabe eines Schaltwerkes hängt **mitunter** von dem aktuellen **Zustand** ab.

Vollständige Spezifiziertheit

Aufgabe 1 – Automaten: Begriffsklärung

Schaltnetz

Die Ausgabe eines Schaltnetzes hängt **alleinig** von der zugehörigen **Eingabe** ab.

Schaltwerk

Die Ausgabe eines Schaltwerkes hängt **mitunter** von dem aktuellen **Zustand** ab.

Vollständige Spezifiziertheit

Automaten Ein Automat ist vollständig spezifiziert gdw. all seine Zustände vollständig spezifiziert sind.

Aufgabe 1 – Automaten: Begriffsklärung

Schaltnetz

Die Ausgabe eines Schaltnetzes hängt **alleinig** von der zugehörigen **Eingabe** ab.

Schaltwerk

Die Ausgabe eines Schaltwerkes hängt **mitunter** von dem aktuellen **Zustand** ab.

Vollständige Spezifiziertheit

Zustände Ein Zustand ist vollständig spezifiziert gdw. das Verhalten für alle möglichen Eingaben spezifiziert ist. (Übergangsfunktion liefert für jede mögliche Eingabe mit diesem Zustand eine gültige Antwort).

Aufgabe 1 – Automaten: Begriffsklärung

Schaltnetz

Die Ausgabe eines Schaltnetzes hängt **alleinig** von der zugehörigen **Eingabe** ab.

Schaltwerk

Die Ausgabe eines Schaltwerkes hängt **mitunter** von dem aktuellen **Zustand** ab.

Vollständige Spezifiziertheit

Automaten
: Ein Automat ist vollständig spezifiziert gdw. all seine Zustände vollständig spezifiziert sind.

Zustände
: Ein Zustand ist vollständig spezifiziert gdw. das Verhalten für alle möglichen Eingaben spezifiziert ist. (Übergangsfunktion liefert für jede mögliche Eingabe mit diesem Zustand eine gültige Antwort).

Aufgabe 1 – Automaten: Begriffsklärung

Schaltblock eines Automaten

e^t	Die Eingabe e zum Zeitpunkt t
s^t / s^{t+1}	Der aktuelle Zustand s zum Zeitpunkt t / $t+1$
δ	Zustandsübergangsfunktion
λ	Ausgabefunktion
a^t	Die Ausgabe a zum Zeitpunkt t

Aufgabe 1 – Automaten: Begriffsklärung

Wir unterscheiden zwischen folgenden Automatentypen:

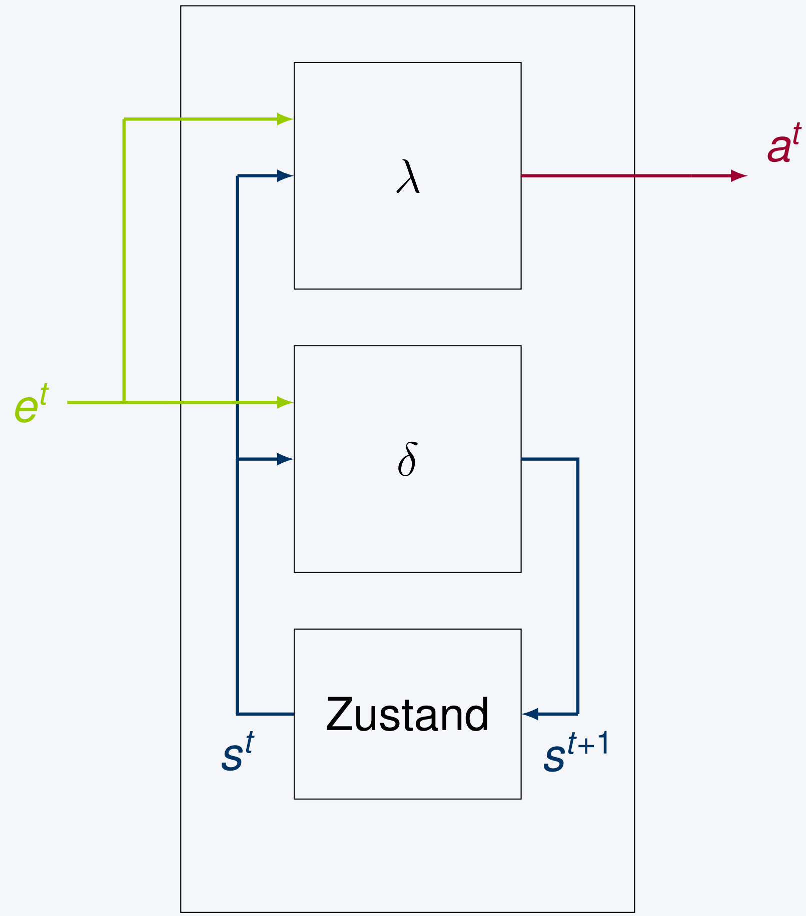
Aufgabe 1 – Automaten: Begriffsklärung

Wir unterscheiden zwischen folgenden Automatentypen:

MEALY-Automat

Die Ausgabe hängt nicht nur vom jeweiligen Zustand, sondern auch von der dazugehörigen Eingabe ab:

$$a^t = \lambda(e^t, s^t)$$



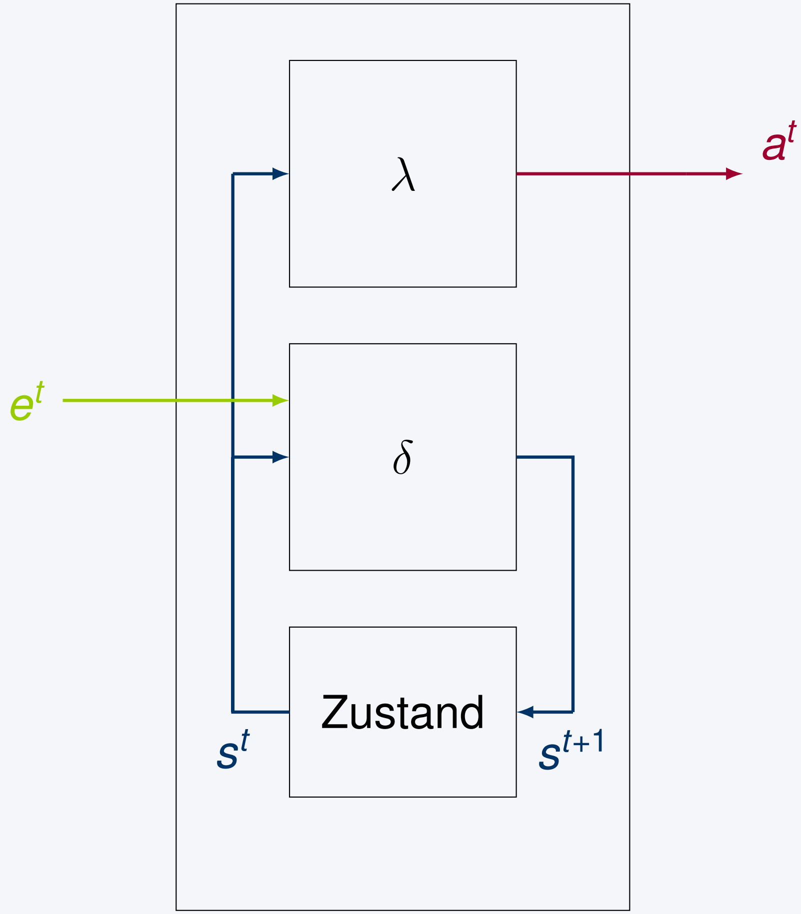
Aufgabe 1 – Automaten: Begriffsklärung

Wir unterscheiden zwischen folgenden Automatentypen:

MOORE-Automat

Die Ausgabe hängt nur vom jeweiligen Zustand ab:

$$a^t = \lambda(s^t)$$



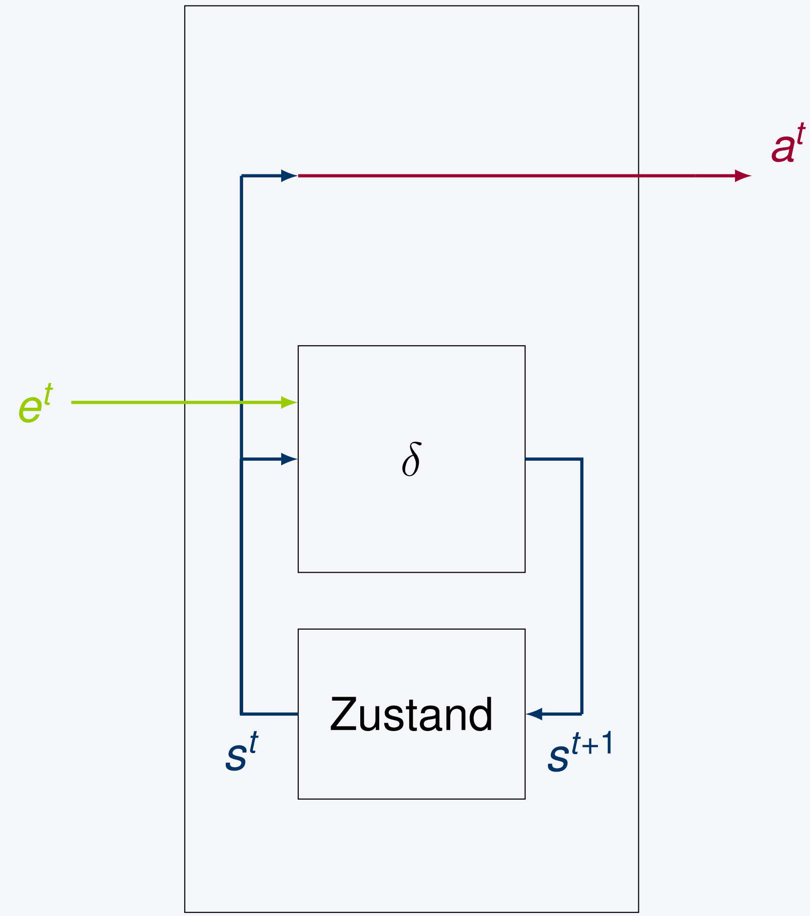
Aufgabe 1 – Automaten: Begriffsklärung

Wir unterscheiden zwischen folgenden Automatentypen:

MEDWEDEW-Automat

Die Ausgabe ist der Zustand selbst, die Funktion λ wird nicht mehr auf s^t angewendet:

$$a^t = s^t$$



Aufgabe 1 – Automaten: Begriffsklärung

Wir unterscheiden zwischen folgenden Automatentypen:

Mealy-Automat

Die Ausgabe hängt nicht nur vom jeweiligen Zustand, sondern auch von der dazugehörigen Eingabe ab:

$$a^t = \lambda(e^t, s^t)$$

Moore-Automat

Die Ausgabe hängt nur vom jeweiligen Zustand ab:

$$a^t = \lambda(s^t)$$

Medwedew-Automat

Die Ausgabe ist der Zustand selbst, die Funktion λ wird nicht mehr auf s^t angewendet:

$$a^t = s^t$$

Unterschied

Der **Unterschied** aller dreier Automatentypen liegt **alleine** in der **Ausgabefunktion**.

Aufgabe 1 – Automaten

Entwerfen Sie einen endlichen Zustandsautomat (FSM) für eine Armbanduhr, der eines von vier internen Registern auf dem Display anzeigt. Die Auswahl des Registers erfolgt durch einen 4:1-Multiplexer, dessen Kontrolleingänge mit s_0 und s_1 bezeichnet werden. Die Register entsprechen den aktuellen Werten der Uhrzeit ($s_1 s_0 = 00$), der Alarmeinstellung ($s_1 s_0 = 01$), des Datums (10) und der Stoppuhr (11).
Durch wiederholtes Drücken des Knopfes b soll es möglich sein, die vier Register in der oben genannten Reihenfolge zyklisch auszulesen. Gehen Sie davon aus, dass durch Drücken des Knopfes der Wert von b synchron zum Takt für eine Taktperiode auf 1 gesetzt wird. Zusätzlich soll der Wechsel des Registers durch einen hörbaren Ton angezeigt werden, indem der Ausgang p bei jedem Drücken des Knopfes für eine Taktperiode auf 1 gesetzt wird.

a) Modellieren Sie den Zustandsautomat als Moore-Automat.

b) Modellieren Sie den Zustandsautomat als Mealy-Automat.

c) Welche Vorteile bietet die Realisierung des Zustandsautomats als Mealy-Automat und welche potentiellen Probleme müssen beachtet werden?

Aufgabe 2 – Flipflops und Automaten

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 2 – Flipflops und Automaten

Realisieren Sie unter Verwendung von JK-Flipflops das Schaltwerk eines Automaten, dessen Überführungsfunktion durch folgende Automatentafel gegeben ist.

Q^n		X		Q^{n+1}		Flipflop 2		Flipflop 1	
q_2^n	q_1^n	b	a	q_2^{n+1}	q_1^{n+1}	J_2	K_2	J_1	K_1
0	0	0	0	0	1				
0	0	0	1	0	1				
0	0	1	-	1	1				
0	1	-	0	1	0				
0	1	-	1	0	1				
1	0	-	0	0	0				
1	0	0	1	1	1				
1	0	1	1	1	0				
1	1	-	-	0	0				

Aufgabe 2 – Flipflops und Automaten

Realisieren Sie unter Verwendung von JK-Flipflops das Schaltwerk eines Automaten, dessen Überführungsfunktion durch die vorherige Automatentafel gegeben ist.

a) Bestimmen Sie die Ansteuerfunktionen (J_2, K_2) und (J_1, K_1) der beiden JK-Flipflops.

b) Können Sie anhand der Tabelle den Automatentyp angeben (mit Begründung)?

c) Tragen Sie die Ansteuerfunktionen J_2, K_2, J_1, und K_1 in Symmetriediagramme ein und bestimmen Sie jeweils eine disjunktive Minimalform.

d) Realisieren sie die Ansteuerfunktionen unter Verwendung eines PAL-Bausteins und zeichnen Sie das vollständige daraus resultierende Schaltwerk.

Aufgabe 3 – Synchrones Schaltwerk

design

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 3 – Synchrones Schaltwerk

Entwerfen Sie eine synchrone Schaltung mit den Zuständen A, B, C, D. Diese soll abhängig von den Eingangssignalen R (Rücksetzen) und V (Vorwärts)

- für die Belegung $R = 0$, $V = 1$ den Zyklus $A, B, C, D, A, B, \ldots$ durchlaufen.
- für die Belegung $R = 0$, $V = 0$ den Zyklus $A, D, C, B, A, D, \ldots$ durchlaufen.
- für $R = 1$ unabhängig von V in den Zustand A übergehen.

a) Zeichnen Sie das Zustandsdiagramm.

b) Stellen Sie die Automatentafel auf.

c) Realisieren Sie die Schaltung mit zwei D-Flipflops und unter ausschließlicher Verwendung von 2:1-Multiplexern.

Aufgabe 3 – Synchrones Schaltwerk

Entwerfen Sie eine synchrone Schaltung mit den Zuständen A, B, C, D. Diese soll abhängig von den Eingangssignalen R (Rücksetzen) und V (Vorwärts)

- für die Belegung $R = 0$, $V = 1$ den Zyklus $A, B, C, D, A, B, \ldots$ durchlaufen.
- für die Belegung $R = 0$, $V = 0$ den Zyklus $A, D, C, B, A, D, \ldots$ durchlaufen.
- für $R = 1$ unabhängig von V in den Zustand A übergehen.

a) Zeichnen Sie das Zustandsdiagramm.

Wie viele Zustände?

Aufgabe 3 – Synchrones Schaltwerk

Entwerfen Sie eine synchrone Schaltung mit den Zuständen A, B, C, D. Diese soll abhängig von den Eingangssignalen R (Rücksetzen) und V (Vorwärts)

- für die Belegung $R = 0$, $V = 1$ den Zyklus $A, B, C, D, A, B, \ldots$ durchlaufen.
- für die Belegung $R = 0$, $V = 0$ den Zyklus $A, D, C, B, A, D, \ldots$ durchlaufen.
- für $R = 1$ unabhängig von V in den Zustand A übergehen.

a) Zeichnen Sie das Zustandsdiagramm.

Wie viele Zustände?

Aufgabe 3 – Synchrones Schaltwerk

Entwerfen Sie eine synchrone Schaltung mit den Zuständen A, B, C, D. Diese soll abhängig von den Eingangssignalen R (Rücksetzen) und V (Vorwärts)

- für die Belegung $R = 0$, $V = 1$ den Zyklus $A, B, C, D, A, B, \ldots$ durchlaufen.
- für die Belegung $R = 0$, $V = 0$ den Zyklus $A, D, C, B, A, D, \ldots$ durchlaufen.
- für $R = 1$ unabhängig von V in den Zustand A übergehen.

a) Zeichnen Sie das Zustandsdiagramm.

Wie viele Zustände? 4

Aufgabe 3 – Synchrones Schaltwerk

Entwerfen Sie eine synchrone Schaltung mit den Zuständen A, B, C, D. Diese soll abhängig von den Eingangssignalen R (Rücksetzen) und V (Vorwärts)

- für die Belegung $R = 0$, $V = 1$ den Zyklus $A, B, C, D, A, B, \ldots$ durchlaufen.
- für die Belegung $R = 0$, $V = 0$ den Zyklus $A, D, C, B, A, D, \ldots$ durchlaufen.
- für $R = 1$ unabhängig von V in den Zustand A übergehen.

a) Zeichnen Sie das Zustandsdiagramm.

Wie viele Zustände? 4

Eingabe/Übergänge:

Aufgabe 3 – Synchrones Schaltwerk

Entwerfen Sie eine synchrone Schaltung mit den Zuständen A, B, C, D.
Diese soll abhängig von den Eingangssignalen R (Rücksetzen) und V (Vorwärts)

- für die Belegung $R = 0$, $V = 1$ den Zyklus $A, B, C, D, A, B, \ldots$ durchlaufen.
- für die Belegung $R = 0$, $V = 0$ den Zyklus $A, D, C, B, A, D, \ldots$ durchlaufen.
- für $R = 1$ unabhängig von V in den Zustand A übergehen.

a) Zeichnen Sie das Zustandsdiagramm.

Wie viele Zustände? 4
Eingabe/Übergänge:

Aufgabe 3 – Synchrones Schaltwerk

Entwerfen Sie eine synchrone Schaltung mit den Zuständen A, B, C, D. Diese soll abhängig von den Eingangssignalen R (Rücksetzen) und V (Vorwärts)

- für die Belegung $R = 0$, $V = 1$ den Zyklus $A, B, C, D, A, B, \ldots$ durchlaufen.
- für die Belegung $R = 0$, $V = 0$ den Zyklus $A, D, C, B, A, D, \ldots$ durchlaufen.
- für $R = 1$ unabhängig von V in den Zustand A übergehen.

a) Zeichnen Sie das Zustandsdiagramm.

Wie viele Zustände? 4

Eingabe/Übergänge: R und V

Aufgabe 3 – Synchrones Schaltwerk

Entwerfen Sie eine synchrone Schaltung mit den Zuständen A, B, C, D. Diese soll abhängig von den Eingangssignalen R (Rücksetzen) und V (Vorwärts)

- für die Belegung $R = 0$, $V = 1$ den Zyklus $A, B, C, D, A, B, \dots$ durchlaufen.
- für die Belegung $R = 0$, $V = 0$ den Zyklus $A, D, C, B, A, D, \dots$ durchlaufen.
- für $R = 1$ unabhängig von V in den Zustand A übergehen.

a) Zeichnen Sie das Zustandsdiagramm.

Wie viele Zustände? 4
Eingabe/Übergänge: R und V
Startzustand? A
Da in beiden Zyklen mit dem Knoten A gestartet wird.

Aufgabe 3 – Synchrones Schaltwerk: Zustandsdiagramm

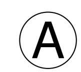
Wie viele Zustände? 4
Eingabe: R und V
Übergänge:

- für die Belegung $R = 0,\ V = 1$ den Zyklus $A, B, C, D, A, B, \ldots$ durchlaufen.
- für die Belegung $R = 0,\ V = 0$ den Zyklus $A, D, C, B, A, D, \ldots$ durchlaufen.
- für $R = 1$ unabhängig von V in den Zustand A übergehen.

Aufgabe 3 – Synchrones Schaltwerk: Zustandsdiagramm

Wie viele Zustände? 4
Eingabe: R und V
Übergänge:

- für die Belegung $R = 0,\ V = 1$ den Zyklus $A, B, C, D, A, B, \ldots$ durchlaufen.
- für die Belegung $R = 0,\ V = 0$ den Zyklus $A, D, C, B, A, D, \ldots$ durchlaufen.
- für $R = 1$ unabhängig von V in den Zustand A übergehen.



Aufgabe 3 – Synchrones Schaltwerk: Zustandsdiagramm

Wie viele Zustände? 4

Eingabe: R und V

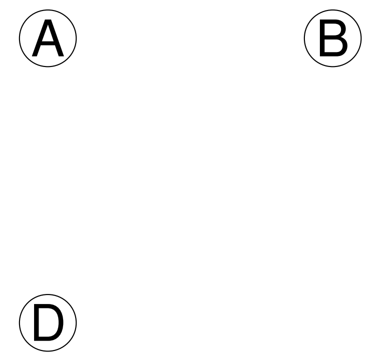
Übergänge:

- für die Belegung $R = 0,\ V = 1$ den Zyklus $A, B, C, D, A, B, \ldots$ durchlaufen.
- für die Belegung $R = 0,\ V = 0$ den Zyklus $A, D, C, B, A, D, \ldots$ durchlaufen.
- für $R = 1$ unabhängig von V in den Zustand A übergehen.

Aufgabe 3 – Synchrones Schaltwerk: Zustandsdiagramm

Wie viele Zustände? 4
Eingabe: R und V
Übergänge:

- für die Belegung $R = 0$, $V = 1$ den Zyklus $A, B, C, D, A, B, \ldots$ durchlaufen.
- für die Belegung $R = 0$, $V = 0$ den Zyklus $A, D, C, B, A, D, \ldots$ durchlaufen.
- für $R = 1$ unabhängig von V in den Zustand A übergehen.



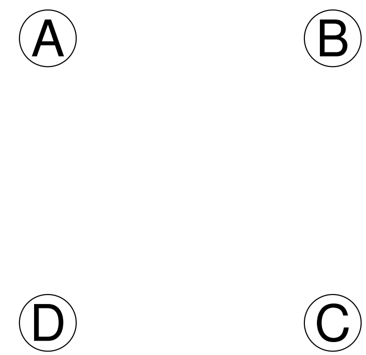
Aufgabe 3 – Synchrones Schaltwerk: Zustandsdiagramm

Wie viele Zustände? 4

Eingabe: R und V

Übergänge:

- für die Belegung $R = 0,\ V = 1$ den Zyklus $A, B, C, D, A, B, \ldots$ durchlaufen.
- für die Belegung $R = 0,\ V = 0$ den Zyklus $A, D, C, B, A, D, \ldots$ durchlaufen.
- für $R = 1$ unabhängig von V in den Zustand A übergehen.



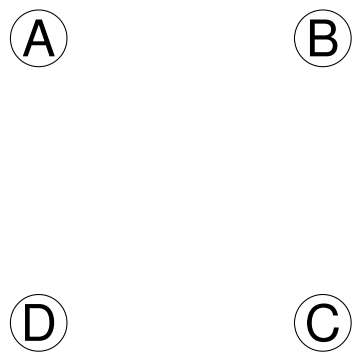
Aufgabe 3 – Synchrones Schaltwerk: Zustandsdiagramm

Wie viele Zustände? 4

Eingabe: R und V

Übergänge:

- für die Belegung $R = 0,\ V = 1$ den Zyklus $A, B, C, D, A, B, \ldots$ durchlaufen. $V \cdot \overline{R}$
- für die Belegung $R = 0,\ V = 0$ den Zyklus $A, D, C, B, A, D, \ldots$ durchlaufen.
- für $R = 1$ unabhängig von V in den Zustand A übergehen.



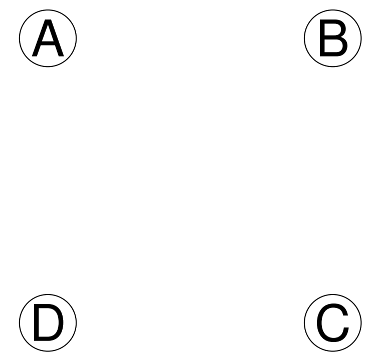
Aufgabe 3 – Synchrones Schaltwerk: Zustandsdiagramm

Wie viele Zustände? 4

Eingabe: R und V

Übergänge:

- für die Belegung $R = 0,\ V = 1$ den Zyklus $A, B, C, D, A, B, \ldots$ durchlaufen. $V \cdot \overline{R}$
- für die Belegung $R = 0,\ V = 0$ den Zyklus $A, D, C, B, A, D, \ldots$ durchlaufen. $\overline{V} \cdot \overline{R}$
- für $R = 1$ unabhängig von V in den Zustand A übergehen.



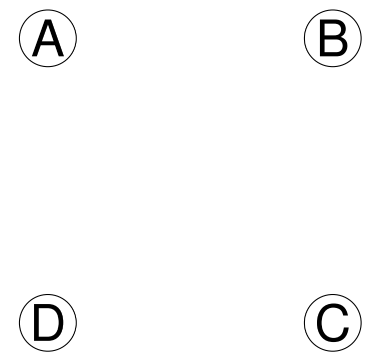
Aufgabe 3 – Synchrones Schaltwerk: Zustandsdiagramm

Wie viele Zustände? 4

Eingabe: R und V

Übergänge:

- für die Belegung $R = 0,\ V = 1$ den Zyklus $A, B, C, D, A, B, \dots$ durchlaufen. $V \cdot \overline{R}$
- für die Belegung $R = 0,\ V = 0$ den Zyklus $A, D, C, B, A, D, \dots$ durchlaufen. $\overline{V} \cdot \overline{R}$
- für $R = 1$ unabhängig von V in den Zustand A übergehen. R



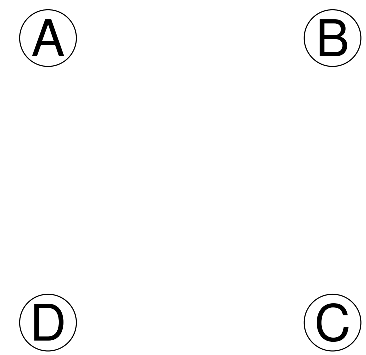
Aufgabe 3 – Synchrones Schaltwerk: Zustandsdiagramm

Wie viele Zustände? 4

Eingabe: R und V

Übergänge:

- für die Belegung $R = 0,\ V = 1$ den Zyklus $A, B, C, D, A, B, \ldots$ durchlaufen. $V \cdot \overline{R}$
- für die Belegung $R = 0,\ V = 0$ den Zyklus $A, D, C, B, A, D, \ldots$ durchlaufen. $\overline{V} \cdot \overline{R}$
- für $R = 1$ unabhängig von V in den Zustand A übergehen. R



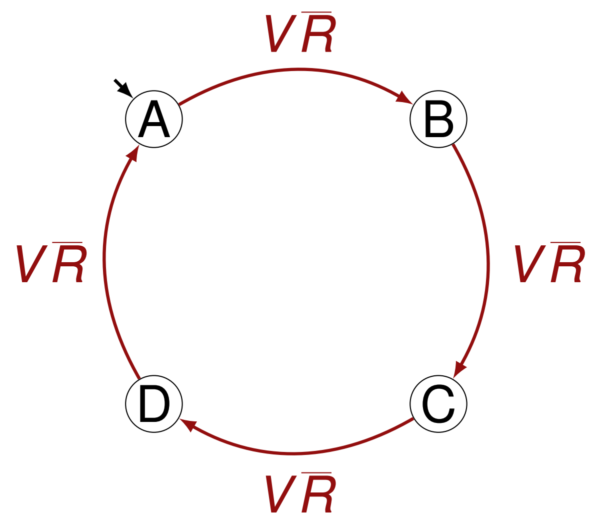
Aufgabe 3 – Synchrones Schaltwerk: Zustandsdiagramm

Wie viele Zustände? 4

Eingabe: R und V

Übergänge:

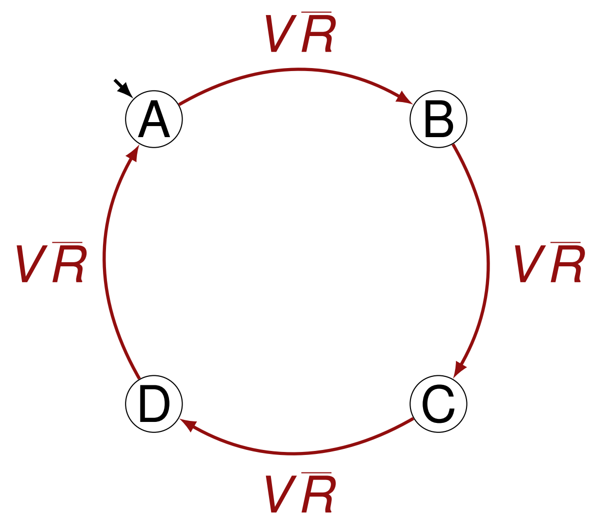
- für die Belegung $R = 0,\ V = 1$ den Zyklus $A, B, C, D, A, B, \ldots$ durchlaufen. $V \cdot \overline{R}$
- für die Belegung $R = 0,\ V = 0$ den Zyklus $A, D, C, B, A, D, \ldots$ durchlaufen. $\overline{V} \cdot \overline{R}$
- für $R = 1$ unabhängig von V in den Zustand A übergehen. R



Aufgabe 3 – Synchrones Schaltwerk: Zustandsdiagramm

Wie viele Zustände? 4
Eingabe: R und V
Übergänge:

- für die Belegung $R = 0$, $V = 1$ den Zyklus $A, B, C, D, A, B, \ldots$ durchlaufen. $V \cdot \overline{R}$
- für die Belegung $R = 0$, $V = 0$ den Zyklus $A, D, C, B, A, D, \ldots$ durchlaufen. $\overline{V} \cdot \overline{R}$
- für $R = 1$ unabhängig von V in den Zustand A übergehen. R



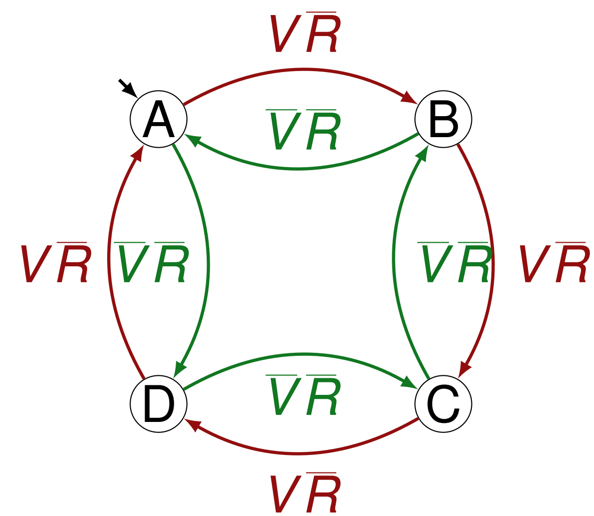
Aufgabe 3 – Synchrones Schaltwerk: Zustandsdiagramm

Wie viele Zustände? 4

Eingabe: R und V

Übergänge:

- für die Belegung $R = 0,\ V = 1$ den Zyklus $A, B, C, D, A, B, \ldots$ durchlaufen. $V \cdot \overline{R}$
- für die Belegung $R = 0,\ V = 0$ den Zyklus $A, D, C, B, A, D, \ldots$ durchlaufen. $\overline{V} \cdot \overline{R}$
- für $R = 1$ unabhängig von V in den Zustand A übergehen. R



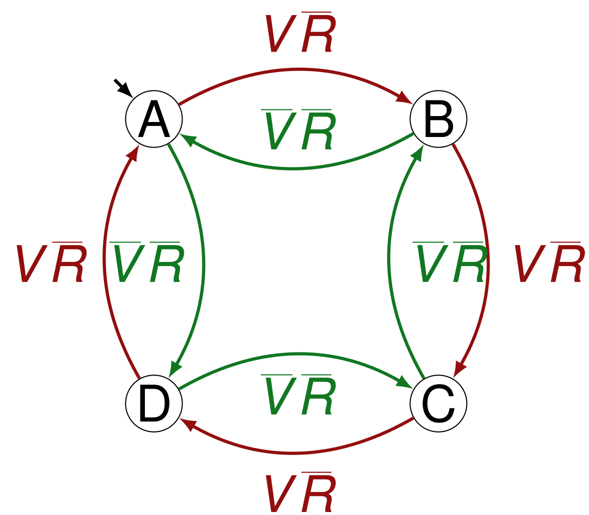
Aufgabe 3 – Synchrones Schaltwerk: Zustandsdiagramm

Wie viele Zustände? 4

Eingabe: R und V

Übergänge:

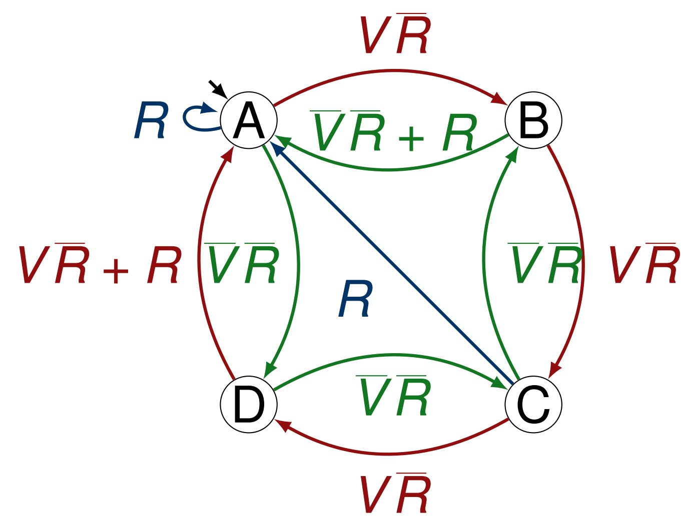
- für die Belegung $R = 0,\ V = 1$ den Zyklus $A, B, C, D, A, B, \ldots$ durchlaufen. $V \cdot \overline{R}$
- für die Belegung $R = 0,\ V = 0$ den Zyklus $A, D, C, B, A, D, \ldots$ durchlaufen. $\overline{V} \cdot \overline{R}$
- für $R = 1$ unabhängig von V in den Zustand A übergehen. R



Aufgabe 3 – Synchrones Schaltwerk: Zustandsdiagramm

Wie viele Zustände? 4
Eingabe: R und V
Übergänge:

- für die Belegung $R = 0,\ V = 1$ den Zyklus $A, B, C, D, A, B, \ldots$ durchlaufen. $V \cdot \overline{R}$
- für die Belegung $R = 0,\ V = 0$ den Zyklus $A, D, C, B, A, D, \ldots$ durchlaufen. $\overline{V} \cdot \overline{R}$
- für $R = 1$ unabhängig von V in den Zustand A übergehen. R



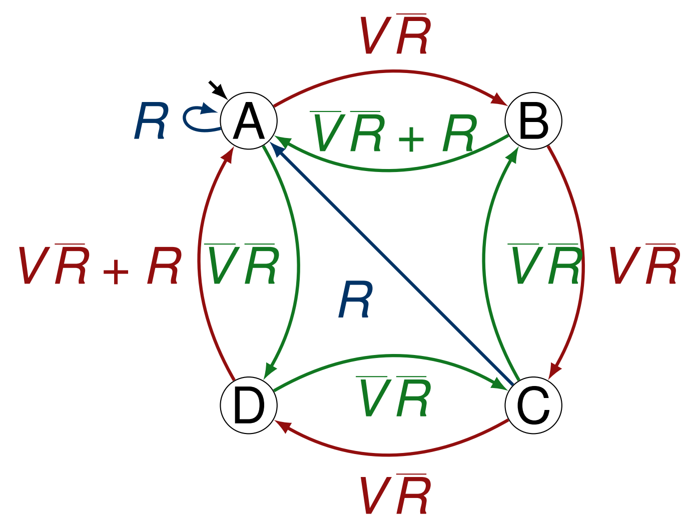
Aufgabe 3 – Synchrones Schaltwerk: Zustandsdiagramm

Wie viele Zustände? 4

Eingabe: R und V

Übergänge:

- für die Belegung $R = 0,\ V = 1$ den Zyklus $A, B, C, D, A, B, \ldots$ durchlaufen. $V \cdot \overline{R}$
- für die Belegung $R = 0,\ V = 0$ den Zyklus $A, D, C, B, A, D, \ldots$ durchlaufen. $\overline{V} \cdot \overline{R}$
- für $R = 1$ unabhängig von V in den Zustand A übergehen. R



$\rightarrow$ Minimieren der Funktionen $\overline{V}\,\overline{R}+R$ und $V\overline{R}+R$ liefert jeweils $\overline{V}+R$ und $V+R$.

Aufgabe 3 – Synchrones Schaltwerk: Zustandsdiagramm

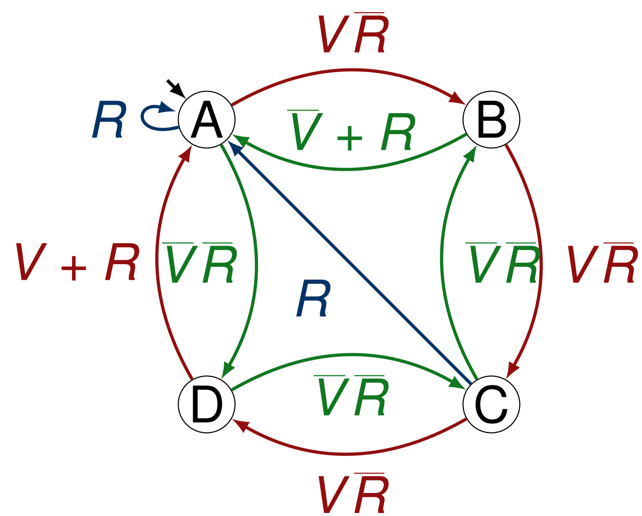
Wie viele Zustände? 4

Eingabe: R und V

Übergänge:

- für die Belegung $R = 0,\ V = 1$ den Zyklus $A, B, C, D, A, B, \ldots$ durchlaufen. $V \cdot \overline{R}$
- für die Belegung $R = 0,\ V = 0$ den Zyklus $A, D, C, B, A, D, \ldots$ durchlaufen. $\overline{V} \cdot \overline{R}$
- für $R = 1$ unabhängig von V in den Zustand A übergehen. R

$\rightarrow$ Minimieren der Funktionen $\overline{V}\,\overline{R}+R$ und $V\overline{R}+R$ liefert jeweils $\overline{V} + R$ und $V + R$.



Aufgabe 3 – Synchrones Schaltwerk

b) Stellen Sie die Automatentafel auf.

Wichtig: Automatentafeln

Automatentafeln sind die tabellarische Darstellung eines Automaten. Wichtig ist, dass **alle** möglichen Zustandsübergänge vorkommen.

Aufgabe 3 – Synchrones Schaltwerk

b) Stellen Sie die Automatentafel auf.

Wichtig: Automatentafeln

Automatentafeln sind die tabellarische Darstellung eines Automaten. Wichtig ist, dass **alle** möglichen Zustandsübergänge vorkommen.

Zustand q_2 q_1			$\overline{R} \cdot \overline{V}$ (0 0)	$\overline{R} \cdot V$ (0 1)	$R \cdot (\overline{V}/V)$ (1 -)
A	0	0	D (1 1)	B (0 1)	A (0 0)
B	0	1	A (0 0)	C (1 0)	A (0 0)
C	1	0	B (0 1)	D (1 1)	A (0 0)
D	1	1	C (1 0)	A (0 0)	A (0 0)

Aufgabe 3 – Synchrones Schaltwerk

c) Realisieren Sie die Schaltung mit zwei D-Flipflops und unter ausschließlicher Verwendung von 2:1-Multiplexern.

Aufstellen der Wertetabelle. Bei D-Flipflops gilt: **Der Ansteuerwert der Flipflops sind genau der zu speichernde Wert!**

Aufgabe 3 – Synchrones Schaltwerk

c) Realisieren Sie die Schaltung mit zwei D-Flipflops und unter ausschließlicher Verwendung von 2:1-Multiplexern.

q_2	q_1	R	V	q'_2	q'_1	D_2	D_1
0	0	0	0	1	1	1	1
0	0	0	1	0	1	0	1
0	0	1	-	0	0	0	0
0	1	0	0	0	0	0	0
0	1	0	1	1	0	1	0
0	1	1	-	0	0	0	0
1	0	0	0	0	1	0	1
1	0	0	1	1	1	1	1
1	0	1	-	0	0	0	0
1	1	0	0	1	0	1	0
1	1	0	1	0	0	0	0
1	1	1	-	0	0	0	0

Aufgabe 3 – Synchrones Schaltwerk

c) Realisieren Sie die Schaltung mit zwei D-Flipflops und unter ausschließlicher Verwendung von 2:1-Multiplexern.

Wir fangen an zu minimieren (Symmetriediagramme):

		V	V		
	1 $_{0}$	0 $_{1}$	1 $_{5}$	0 $_{4}$	
R	0 $_{2}$	0 $_{3}$	0 $_{7}$	0 $_{6}$	
R	0 $_{10}$	0 $_{11}$	0 $_{15}$	0 $_{14}$	q_2
	0 $_{8}$	1 $_{9}$	0 $_{13}$	1 $_{12}$	q_2
			q_1	q_1	

Aufgabe 3 – Synchrones Schaltwerk

c) Realisieren Sie die Schaltung mit zwei D-Flipflops und unter ausschließlicher Verwendung von 2:1-Multiplexern.

Wir fangen an zu minimieren (Symmetriediagramme):

		V	V		
	$\boxed{1}_{0}$	0_{1}	$\boxed{1}_{5}$	0_{4}	
R	0_{2}	0_{3}	0_{7}	0_{6}	
R	0_{10}	0_{11}	0_{15}	0_{14}	q_2
	0_{8}	$\boxed{1}_{9}$	0_{13}	$\boxed{1}_{12}$	q_2
			q_1	q_1	

Aufgabe 3 – Synchrones Schaltwerk

c) Realisieren Sie die Schaltung mit zwei D-Flipflops und unter ausschließlicher Verwendung von 2:1-Multiplexern.

Wir fangen an zu minimieren (Symmetriediagramme):

		V	V		
	1 (0)	0 (1)	1 (5)	0 (4)	
R	0 (2)	0 (3)	0 (7)	0 (6)	
R	0 (10)	0 (11)	0 (15)	0 (14)	q_2
	0 (8)	1 (9)	0 (13)	1 (12)	q_2
			q_1	q_1	

$$D_2 = V\overline{R}(\overline{q_2}q_1 + q_2\overline{q_1}) + \overline{V}\,\overline{R}(\overline{q_2}\,\overline{q_1} + q_2q_1)$$

Aufgabe 3 – Synchrones Schaltwerk

c) Realisieren Sie die Schaltung mit zwei D-Flipflops und unter ausschließlicher Verwendung von 2:1-Multiplexern.

Wir fangen an zu minimieren (Symmetriediagramme):

		V	V		
	1 $_{0}$	1 $_{1}$	0 $_{5}$	0 $_{4}$	
R	0 $_{2}$	0 $_{3}$	0 $_{7}$	0 $_{6}$	
R	0 $_{10}$	0 $_{11}$	0 $_{15}$	0 $_{14}$	q_2
	1 $_{8}$	1 $_{9}$	0 $_{13}$	0 $_{12}$	q_2
			q_1	q_1	

$$D_2 = V\overline{R}(\overline{q_2}q_1 + q_2\overline{q_1}) + \overline{V}\,\overline{R}(\overline{q_2}\,\overline{q_1} + q_2 q_1)$$

Aufgabe 3 – Synchrones Schaltwerk

c) Realisieren Sie die Schaltung mit zwei D-Flipflops und unter ausschließlicher Verwendung von 2:1-Multiplexern.

Wir fangen an zu minimieren (Symmetriediagramme):

V

1 $_{0}$	1 $_{1}$	0 $_{5}$	0 $_{4}$
0 $_{2}$	0 $_{3}$	0 $_{7}$	0 $_{6}$
0 $_{10}$	0 $_{11}$	0 $_{15}$	0 $_{14}$
1 $_{8}$	1 $_{9}$	0 $_{13}$	0 $_{12}$

R q_2

q_1

$$D_2 = V\overline{R}(\overline{q_2}q_1 + q_2\overline{q_1}) + \overline{V}\,\overline{R}(\overline{q_2}\,\overline{q_1} + q_2q_1)$$

Aufgabe 3 – Synchrones Schaltwerk

c) Realisieren Sie die Schaltung mit zwei D-Flipflops und unter ausschließlicher Verwendung von 2:1-Multiplexern.

Wir fangen an zu minimieren (Symmetriediagramme):

1 (0)	1 (1)	0 (5)	0 (4)
0 (2)	0 (3)	0 (7)	0 (6)
0 (10)	0 (11)	0 (15)	0 (14)
1 (8)	1 (9)	0 (13)	0 (12)

$$D_2 = V\overline{R}(\overline{q_2}q_1 + q_2\overline{q_1}) + \overline{V}\,\overline{R}(\overline{q_2}\,\overline{q_1} + q_2q_1)$$

$$D_1 = \overline{R}\overline{q_1}$$

Aufgabe 3 – Synchrones Schaltwerk: Realisierung

c) Realisieren Sie die Schaltung mit zwei D-Flipflops und unter ausschließlicher Verwendung von 2:1-Multiplexern.

$$D_1 = \overline{q_1}\,\overline{R}$$

$$D_2 = V\overline{R}(\overline{q_2}q_1 + q_2\overline{q_1}) + \overline{V}\,\overline{R}(\overline{q_2}\,\overline{q_1} + q_2q_1)$$

Aufgabe 3 – Synchrones Schaltwerk: Realisierung

c) Realisieren Sie die Schaltung mit zwei D-Flipflops und unter ausschließlicher Verwendung von 2:1-Multiplexern.

$$D_1 = \overline{q_1}\,\overline{R}$$
$$D_2 = V\overline{R}(\overline{q_2}q_1 + q_2\overline{q_1}) + \overline{V}\,\overline{R}(\overline{q_2}\,\overline{q_1} + q_2q_1)$$

Um die Schaltungen ausschließlich mit Multiplexern aufzubauen, verwenden wir den Entwicklungssatz von Shannon, zum Beispiel in der Reihenfolge R, V und q_1:

Aufgabe 3 – Synchrones Schaltwerk: Realisierung

c) Realisieren Sie die Schaltung mit zwei D-Flipflops und unter ausschließlicher Verwendung von 2:1-Multiplexern.

$$D_1 = \overline{q_1}\,\overline{R}$$
$$D_2 = V\overline{R}(\overline{q_2}q_1 + q_2\overline{q_1}) + \overline{V}\,\overline{R}(\overline{q_2}\,\overline{q_1} + q_2q_1)$$

Um die Schaltungen ausschließlich mit Multiplexern aufzubauen, verwenden wir den Entwicklungssatz von Shannon, zum Beispiel in der Reihenfolge R, V und q_1:

$$D_1 = \overline{q_1}\,\overline{R}$$

Aufgabe 3 – Synchrones Schaltwerk: Realisierung

c) Realisieren Sie die Schaltung mit zwei D-Flipflops und unter ausschließlicher Verwendung von 2:1-Multiplexern.

$$D_1 = \overline{q_1}\,\overline{R}$$
$$D_2 = V\overline{R}(\overline{q_2}q_1 + q_2\overline{q_1}) + \overline{V}\,\overline{R}(\overline{q_2}\,\overline{q_1} + q_2q_1)$$

Um die Schaltungen ausschließlich mit Multiplexern aufzubauen, verwenden wir den Entwicklungssatz von Shannon, zum Beispiel in der Reihenfolge R, V und q_1:

$$\begin{aligned} D_1 &= \overline{q_1}\,\overline{R} \\ &\equiv R \cdot (0) + \overline{R} \cdot (\overline{q_1}) \end{aligned}$$

Aufgabe 3 – Synchrones Schaltwerk: Realisierung

c) Realisieren Sie die Schaltung mit zwei D-Flipflops und unter ausschließlicher Verwendung von 2:1-Multiplexern.

$$D_1 = \overline{q_1}\,\overline{R}$$
$$D_2 = V\overline{R}(\overline{q_2}q_1 + q_2\overline{q_1}) + \overline{V}\,\overline{R}(\overline{q_2}\,\overline{q_1} + q_2q_1)$$

Um die Schaltungen ausschließlich mit Multiplexern aufzubauen, verwenden wir den Entwicklungssatz von Shannon, zum Beispiel in der Reihenfolge R, V und q_1:

$$\begin{aligned} D_1 &= \overline{q_1}\,\overline{R} \\ &\equiv R \cdot (0) + \overline{R} \cdot (\overline{q_1}) \\ D_2 &= R \cdot (0) + \overline{R} \cdot (V\overline{q_2}q_1 + Vq_2\overline{q_1} + \overline{V}\,\overline{q_2}\,\overline{q_1} + \overline{V}q_2q_1) \end{aligned}$$

Aufgabe 3 – Synchrones Schaltwerk: Realisierung

c) Realisieren Sie die Schaltung mit zwei D-Flipflops und unter ausschließlicher Verwendung von 2:1-Multiplexern.

$$D_1 = \overline{q_1}\,\overline{R}$$
$$D_2 = V\overline{R}(\overline{q_2}q_1 + q_2\overline{q_1}) + \overline{V}\,\overline{R}(\overline{q_2}\,\overline{q_1} + q_2q_1)$$

Um die Schaltungen ausschließlich mit Multiplexern aufzubauen, verwenden wir den Entwicklungssatz von Shannon, zum Beispiel in der Reihenfolge R, V und q_1:

$$\begin{aligned} D_1 &= \overline{q_1}\,\overline{R} \\ &\equiv R \cdot (0) + \overline{R} \cdot (\overline{q_1}) \\ D_2 &= R \cdot (0) + \overline{R} \cdot (V\overline{q_2}q_1 + Vq_2\overline{q_1} + \overline{V}\,\overline{q_2}\,\overline{q_1} + \overline{V}q_2q_1) \\ &\equiv R \cdot (0) + \overline{R} \cdot (V \cdot (\overline{q_2}q_1 + q_2\overline{q_1}) + \overline{V} \cdot (\overline{q_2}\,\overline{q_1} + q_2q_1)) \end{aligned}$$

Aufgabe 3 – Synchrones Schaltwerk: Realisierung

c) Realisieren Sie die Schaltung mit zwei D-Flipflops und unter ausschließlicher Verwendung von 2:1-Multiplexern.

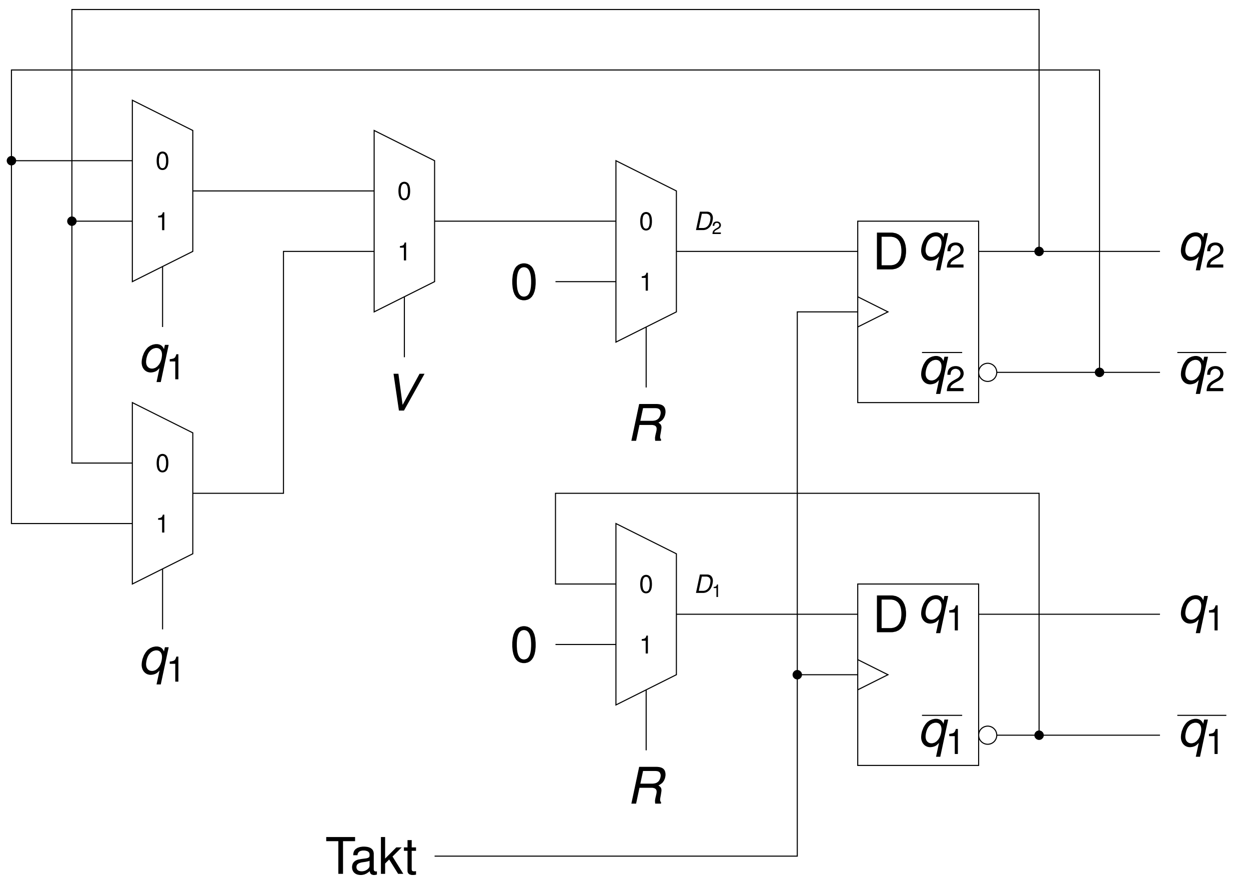
$$D_1 = \overline{q_1}\,\overline{R}$$
$$D_2 = V\overline{R}(\overline{q_2}q_1 + q_2\overline{q_1}) + \overline{V}\,\overline{R}(\overline{q_2}\,\overline{q_1} + q_2q_1)$$

Um die Schaltungen ausschließlich mit Multiplexern aufzubauen, verwenden wir den Entwicklungssatz von Shannon, zum Beispiel in der Reihenfolge R, V und q_1:

$$\begin{aligned}
D_1 &= \overline{q_1}\,\overline{R} \\
&\equiv R\cdot(0) + \overline{R}\cdot(\overline{q_1}) \\
D_2 &= R\cdot(0) + \overline{R}\cdot(V\overline{q_2}q_1 + Vq_2\overline{q_1} + \overline{V}\,\overline{q_2}\,\overline{q_1} + \overline{V}q_2q_1) \\
&\equiv R\cdot(0) + \overline{R}\cdot(V\cdot(\overline{q_2}q_1 + q_2\overline{q_1}) + \overline{V}\cdot(\overline{q_2}\,\overline{q_1} + q_2q_1)) \\
&\equiv R\cdot(0) + \overline{R}\cdot\big(V\cdot(q_1\cdot(\overline{q_2}) + \overline{q_1}(q_2)) + \overline{V}\cdot(q_1\cdot(q_2) + \overline{q_1}\cdot(\overline{q_2}))\big)
\end{aligned}$$

Aufgabe 3 – Synchrones Schaltwerk: Realisierung

c) Realisieren Sie die Schaltung mit zwei D-Flipflops und unter ausschließlicher Verwendung von 2:1-Multiplexern.



Übungen zur Grundlagen der Technischen Informatik

Übung 12 – VHDL und Komparatoren

Florian Frank

Friedrich-Alexander-Universität Erlangen-Nürnberg

Wintersemester 2018/19

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Was machen wir heute?

Organisatorisches: Vorlesungsevaluation

Was machen wir heute?

Organisatorisches: Vorlesungsevaluation

Aufgabe 1 – VHDL: Funktionen

Was machen wir heute?

Organisatorisches: Vorlesungsevaluation

Aufgabe 1 – VHDL: Funktionen

Aufgabe 2 – ALU in VHDL

Was machen wir heute?

Organisatorisches: Vorlesungsevaluation

Aufgabe 1 – VHDL: Funktionen

Aufgabe 2 – ALU in VHDL

Aufgabe 3 – VHDL: Automaten

Was machen wir heute?

Organisatorisches: Vorlesungsevaluation

Aufgabe 1 – VHDL: Funktionen

Aufgabe 2 – ALU in VHDL

Aufgabe 3 – VHDL: Automaten

Aufgabe 4 – Komparator

Was machen wir heute?

Organisatorisches: Vorlesungsevaluation

Aufgabe 1 – VHDL: Funktionen

Aufgabe 2 – ALU in VHDL

Aufgabe 3 – VHDL: Automaten

Aufgabe 4 – Komparator

Korrektur und Besprechung der ersten Miniklausur

Organisatorisches: Vorlesungsevaluation

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Aufgabe 1 – VHDL: Funktionen

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 1 – VHDL: Funktionen

Im VHDL-2008-Standard wurde ein unärer `or`-Operator eingeführt, der die Elemente eines beliebig langen Vektors vom Typ `std_logic_vector` mittels sukzessiver Veroderung auf eine 1 Bit lange Ausgabe vom Typ `std_logic` reduziert.

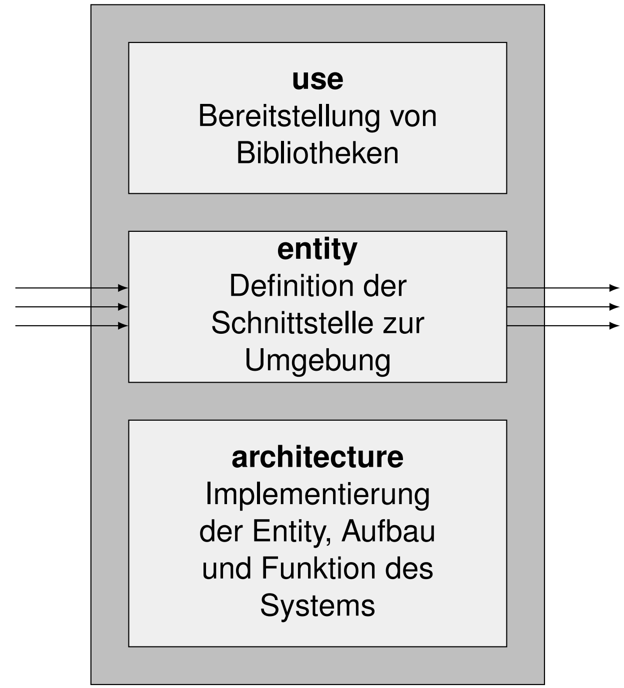
Entwickeln Sie eine Funktion `or_reduce`, die dieselbe Semantik hat wie der beschriebene Operator, um auch Werkzeuge zu unterstützen, die den Standard noch nicht implementieren. Verwenden Sie dazu eine `for`-Schleife und erläutern Sie, wie sich deren Semantik von Schleifen in Software-Programmiersprachen unterscheidet.

VHDL: VHSIC Hardware Description Language (I)

Mit VHDL „programmieren“ wir unsere Hardware und das auf einem möglichst hohen Abstraktionsniveau.

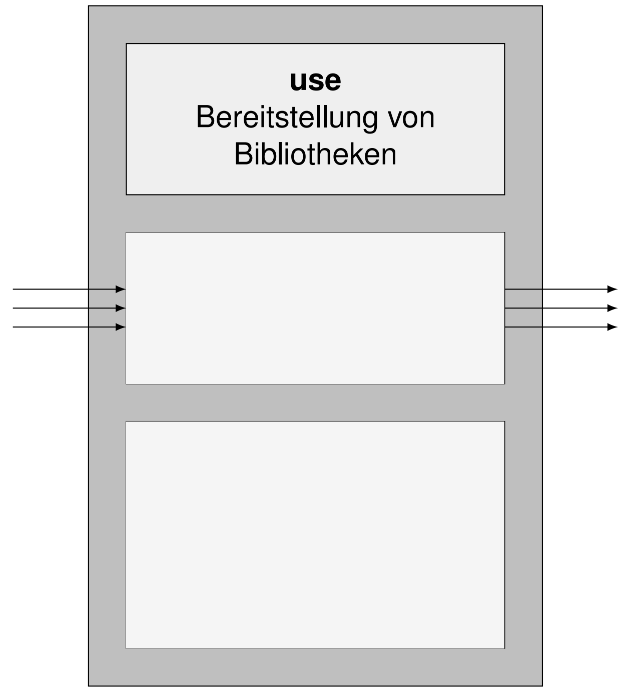
In der Darstellung rechts ist der allgemeine Aufbau einer VHDL-Beschreibung dargestellt.

```
use IEEE.std_logic_1164.all; -- Inkludiert das
   Paket std_logic_1164 aus der Bibliothek IEEE

```



VHDL: VHSIC Hardware Description Language (I)

Mit VHDL „programmieren“ wir unsere Hardware und das auf einem möglichst hohen Abstraktionsniveau.
In der Darstellung rechts ist der allgemeine Aufbau einer VHDL-Beschreibung dargestellt.



use – Bereitstellung von Bibliotheken

- Genauso wie in Java, gibt es auch in VHDL Pakete, in denen verschiedene Typen oder Funktionen definiert sind (zum Beispiel der Typ `std_logic` aus der `IEEE`-Bibliothek) .
- Diese werden dann über den `use`-Befehl „importiert“. Am Beispiel:

```
use IEEE.std_logic_1164.all; -- Inkludiert das Paket std_logic_1164 aus der Bibliothek IEEE
```

VHDL: VHSIC Hardware Description Language (I)

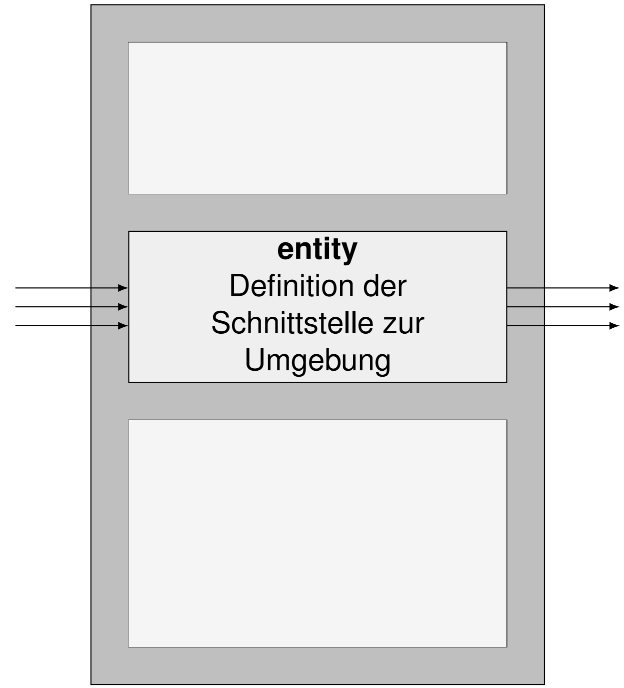
Mit VHDL „programmieren“ wir unsere Hardware und das auf einem möglichst hohen Abstraktionsniveau.

In der Darstellung rechts ist der allgemeine Aufbau einer VHDL-Beschreibung dargestellt.

entity – Definition einer Schnittstelle

Eine „Entität“ beschreibt die „*black-box*“ einer Komponente. Sie besteht nur aus den **Ein-** sowie *Ausgängen* derselbigen.

```
entity entity_name is
  port (
    port_list -- hier stehen Ein- und Ausgaenge
  );
end [entity_name];
```



VHDL: VHSIC Hardware Description Language (I)

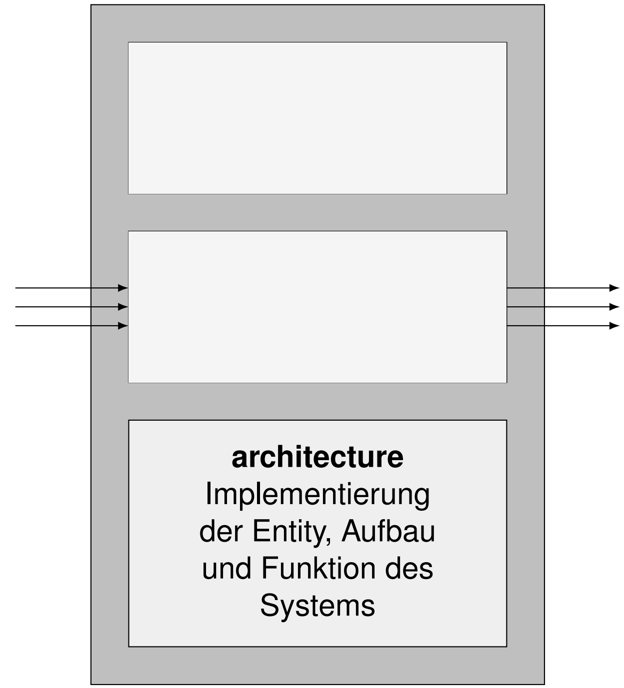
Mit VHDL „programmieren“ wir unsere Hardware und das auf einem möglichst hohen Abstraktionsniveau.

In der Darstellung rechts ist der allgemeine Aufbau einer VHDL-Beschreibung dargestellt.

architecture – Implementierung

Die „Architektur“ einer Entität beschreibt nun den inneren Aufbau, sowie die Funktionalität derselbigen.

```vhdl
architecture architecture_name of entity_name is
    -- hier stehen nur intern genutzte Signale
begin
    -- hier steht, was die Entitaet macht
end [architecture] architecture_name;
```



VHDL: VHSIC Hardware Description Language (II) – Grundlegendes

- Kommentare

```
-- Ein Kommentar muss immer mit einem doppelten Minus (--) beginnen und geht bis
   zum Ende der Zeile
Hier ist kein Kommentar
-- Hier schon
```

- Bezeichner

  Bezeichner sind in VHDL *case-insensitive* (sprich: Groß- und Kleinschreibung werden nicht unterschieden).
  Sie **müssen** mit einem Buchstaben anfangen, anschließend können Buchstaben, Zahlen oder Unterstriche folgen. Zwei Unterstriche dürfen sich dabei **nicht** unmittelbar folgen.

Reservierte Wörter

after	else	library	port	sll
alias	elsif	linkage	postponed	sra
all	end	literal	procedure	srl
and	entity	loop	process	subtype
architecture	exit	map	pure	then
array	file	mod	range	to
assert	for	nand	record	transport
attribute	function	new	register	type
begin	generate	next	reject	unaffected
block	generic	nor	rem	units
body	group	not	report	until
buffer	guarded	null	return	use
bus	if	of	rol	variable
case	impure	on	ror	wait
component	in	open	select	when
configuration	inertial	or	severity	while
constant	inout	others	signal	with
disconnect	is	out	shared	xnor
downto	label	package	sla	xor

VHDL: VHSIC Hardware Description Language (II) – Grundlegendes

- **Variablen**
  Genauso wie in Java oder C, enthält eine Variable in VHDL nur eine Information: den aktuellen Wert.

```vhdl
variable v1 :  std_logic;       -- Variablendeklaration
v1          := 1;               -- Wertzuweisung
```

- **Konstanten**
  Konstanten verhalten sich ähnlich zu Variablen mit dem Unterschied, dass sie nicht verändert werden können.

```vhdl
constant c1 : std_logic := X; -- Konstantendeklaration
```

- **Signale**
  Signale und Variablen sind ähnlich, es gibt aber einige Unterschiede bei der Verwendung der beiden.

```vhdl
signal s1  :  std_logic;        -- Signaldeklaration
s1 <= X;                        -- Wertzuweisung
```

VHDL: VHSIC Hardware Description Language (II) – Grundlegendes

- Variablen
  Genauso wie in Java oder C, enthält eine Variable in VHDL nur eine Information: den aktuellen Wert
  Variablen verhalten sich insbesondere gleich dem Programmiersprachenkonstrukt, als dass sie **sequentiell** genutzt werden und **überschreibbar** sind.
- Konstanten
  Konstanten verhalten sich ähnlich zu Variablen mit dem Unterschied, dass sie nicht verändert werden können.

```
constant c1 : std_logic := X; -- Konstantendeklaration
```

- Signale
  Signale und Variablen sind ähnlich, es gibt aber einige Unterschiede bei der Verwendung der beiden.
  Signale sind vorzustellen als verdrahtete Leitungen, was ihre **nebenläufige** Natur begründet. Ihnen kann in einem Abschnitt nicht mehrfach Werte zugewiesen werden, der zuletzt zugewiesene Wert gilt.

VHDL: VHSIC Hardware Description Language (II) – Beispiel

Welchen Wert enthalten s_2 und v_2 am Ende der Prozedur?

```
procedure p_wertezuweisung (
    variable  v1  : integer;
    variable  v2  : integer;
    signal    s1  : out integer := 5;
    signal    s2  : out integer
  ) is
  begin
    for I in 1 to 10 loop
        v1 <= I;
        s1 <= I;
    end loop;
    v2 <= v1;
    s2 <= s1;
end p_wertezuweisung;
```

VHDL: VHSIC Hardware Description Language (II) – Beispiel

Welchen Wert enthalten s_2 und v_2 am Ende der Prozedur?

```
procedure p_wertezuweisung (
    variable  v1  : integer;
    variable  v2  : integer;
    signal    s1  : out integer := 5;
    signal    s2  : out integer
  ) is
  begin
    for I in 1 to 10 loop
        v1 <= I;
        s1 <= I;
    end loop;
    v2 <= v1;
    s2 <= s1;
end p_wertezuweisung;
```

v_2 enthält den Wert 10, s_2 aber 5.

VHDL: VHSIC Hardware Description Language (III) – Funktionen und Prozeduren

Funktionen

Funktionen stellen ein *sequentielles* Unterprogramm mit Rückgabewert dar und können auch rekursiv aufgerufen werden. Sie dürfen ihre Parameter **nicht** verändern.

```
function identifier [ ( formal parameter list ) ] return a_type is
    [ declarations, see allowed list below ]
begin
    sequential statement(s)
    return some_value; -- muss vom Typ a_type sein
end function identifier ;
```

Die Elemente der „*formal parameter list*“ werden durch ein Semikolon (`;`) von einander getrennt, dem letzten folgt aber **keines**. Ebenfalls darf kein Parameter vom Modus **inout** oder **out** sein.

Erlaubte Deklarationen enthalten unter anderem ...

- ... die Deklaration und der Körper eines Unterprogramms
- ... Konstanten
- ... Variablen
- ... Typen und Subtypen

... aber **nicht** die Deklaration von Signalen.

VHDL: VHSIC Hardware Description Language (III) – Funktionen und Prozeduren

- Prozeduren

  Prozeduren stellen ein *sequentielles* Unterprogramm ohne Rückgabewert dar. Sie geben Werte zurück, indem sie ihre Parameter oder globale Objekte verändern.

```
procedure identifier [ ( formal parameter list ) ] is
    [ declarations, see allowed list below ]
begin
    sequential statement(s)
end procedure identifier ;
```

  Die Elemente der „*formal parameter list*“ werden durch ein Semikolon (;) von einander getrennt, dem letzten folgt aber **keines**.
  Erlaubte Deklarationen enthalten unter anderem ...
  - ... die Deklaration und der Körper eines Unterprogramms
  - ... Konstanten
  - ... Variablen
  - ... Typen und Subtypen

  ... aber **nicht** die Deklaration von Signalen.

VHDL: VHSIC Hardware Description Language (III) – Funktionen und Prozeduren

```vhdl
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;

-- Purpose: This function performs a bitwise xor on the input vector
function f_BITWISE_XOR (
  r_SLV_IN     : in std_logic_vector
) return std_logic is
  variable v_XOR : std_logic := '0';
begin
  for i in 0 to r_SLV_IN'length-1 loop
    v_XOR := v_XOR xor r_SLV_IN(i);
  end loop;
  return v_XOR;
end function f_BITWISE_XOR;
```

VHDL: VHSIC Hardware Description Language (III) – Funktionen und Prozeduren

- Prozeduren

  Prozeduren stellen ein *sequentielles* Unterprogramm ohne Rückgabewert dar. Sie geben Werte zurück, indem sie ihre Parameter oder globale Objekte verändern.

```
procedure identifier [ ( formal parameter list ) ] is
    [ declarations, see allowed list below ]
begin
    sequential statement(s)
end procedure identifier ;
```

  Die Elemente der „*formal parameter list*“ werden durch ein Semikolon (;) von einander getrennt, dem letzten folgt aber **keines**.
  Erlaubte Deklarationen enthalten unter anderem ...
  - ... die Deklaration und der Körper eines Unterprogramms
  - ... Konstanten
  - ... Variablen
  - ... Typen und Subtypen

  ... aber **nicht** die Deklaration von Signalen.

VHDL: VHSIC Hardware Description Language (IV) – Sequentielle Anweisungen

- Prozedur-/Funktionaufruf
  Ruft eine Prozedur oder Funktion auf.

```
[ label: ] procedure-name [ ( actual parameters ) ] ;
```

- if-Abfragen

```
[ label: ] if  condition1  then
     sequence-of-statements
elsif  condition2  then      \_ optional
     sequence-of-statements  /
elsif  condition3  then      \_ optional
     sequence-of-statements  /
...
else                         \_ optional
     sequence-of-statements  /
end if [ label ] ;
```

VHDL: VHSIC Hardware Description Language (IV) – Sequentielle Anweisungen

- switch-case
  Führt aufgrund einer gewissen Wahl einen spezifischen Fall aus. Wahlmöglichkeiten müssen Konstanten desselben diskreten Typen wie der Ausdruck sein.

```
[ label: ] case  expression  is
    when choice1 =>
        sequence-of-statements
    when choice2 =>               \_ optional
        sequence-of-statements  /
    ...
    when others =>                \_ optional if all choices covered
        sequence-of-statements  /
end case [ label ] ;
```

VHDL: VHSIC Hardware Description Language (IV) – Sequentielle Anweisungen

- Schleifen

  Wiederholte Ausfürhung von Code, kommt in dreierlei verschiedenen Ausführungen.

```
[ label: ] loop
     sequence-of-statements -- use exit statement to get out
      end loop [ label ] ;

[ label: ] for variable in range loop
     sequence-of-statements
end loop [ label ] ;

[ label: ] while  condition  loop
     sequence-of-statements
end loop [ label ] ;
```

VHDL: VHSIC Hardware Description Language (IV) – Sequentielle Anweisungen

- next
  Das `continue` von VHDL, kann auch gleichzeitig noch Bedingung für Fortsetzung enthalten.

```
[ label: ] next [ label2 ] [ when condition ] ;
```

- exit
  Das `break` von VHDL, kann auch gleichzeitig noch Bedingung für „Ausbruch“ enthalten.

```
[ label: ] exit [ label2 ] [ when condition ] ;
```

- return
  Gibt einen Wert in Funktionen zurück.

```
[ label: ] return [ expression ] ;
```

- null
  Wird ein „Statement“ gebraucht, man will nichts tun, so hilft null im nun.

```
null ;
```

Aufgabe 1 – VHDL: Funktionen, Lösung der Aufgabe

```
function or_reduce(arg: std_logic_vector) return std_logic is
  variable result : std_logic;
begin
  result := '0';
  for i in arg'range loop
    result := result or arg(i);
  end loop;
  return result;
end or_reduce;
```

Die Ähnlichkeit zu Software-Programmiersprachen trügt:
Wird `or_reduce` in zu synthetisierendem Code (das heißt in tatsächlichen Hardware-Designs und nicht der Simulation) verwendet, wird die Schleife *nicht* sequentiell realisiert. Stattdessen werden für jede Iteration Hardware-Komponenten inferriert (im einfachsten, unoptimierten Fall hier zum Beispiel schlicht `arg'high` Oder-Gatter). Dies ist auch der Grund, wieso in synthetisierbarem VHDL-Code die Schleifengrenzen zur Übersetzungszeit bekannt sein müssen.

Aufgabe 2 – ALU in VHDL

Aufgabe 2 – ALU in VHDL

Entwerfen Sie ein Rechenwerk `alu` (*Arithmetic Logic Unit*), das in Abhängigkeit eines Steuersignals *op* auf zwei 8 Bit lange Eingabevektoren *a* und *b* die folgenden Operationen durchführt und das Ergebnis auf dem ebenfalls 8 Bit langen Ausgabevektor *result* ausgibt:

- $result \leftarrow a + b$ falls $op = 00$
- $result \leftarrow a - b$ falls $op = 01$
- $result \leftarrow a \wedge b$ falls $op = 10$
- $result \leftarrow a \cdot 2$ (um 1 Linksschieben) falls $op = 11$

Verwenden Sie Signale des Typs `std_logic_vector` für die Schnittstelle und die in der IEEE-Bibliothek `numeric_std` definierten Operationen für die Berechnungen.

Aufgabe 2 – ALU in VHDL, Lösung

Zuerst müssen die benötigten Bibliotheken eingebunden werden:

```
library ieee;
use ieee.std_logic_1164.all; -- std_logic_vector
use ieee.numeric_std.all; -- fuer Arithmetik
```

Aufgabe 2 – ALU in VHDL, Lösung

Zuerst müssen die benötigten Bibliotheken eingebunden werden:

```vhdl
library ieee;
use ieee.std_logic_1164.all; -- std_logic_vector
use ieee.numeric_std.all; -- fuer Arithmetik
```

Anschließend wird die zu entwerfende `entity` beschrieben mit ihren Ein- und Ausgängen. Für die ALU benötigen wir die beiden Operanden *a* und *b* sowie die auszuführende Operation *op* als Eingänge und das Ergebnis *result* als Ausgang:

```vhdl
entity alu is
    port(
        a, b : in std_logic_vector(7 downto 0);
        op : in std_logic_vector(1 downto 0);
        result : out std_logic_vector(7 downto 0)
    );
end entity alu;
```

Aufgabe 2 – ALU in VHDL, Lösung

Als nächstes folgt die eigentliche Implementierung der `entity`[1], innerhalb einer `architecture`. Zu Beginn der `architecture` müssen zunächst alle internen Signale deklariert werden.

```
architecture behavioral of alu is
  signal signed_result : signed(8 downto 0);
  signal integer_b : integer;
```

[1] Eine `entity` kann mehrere `architecture`s haben.

Aufgabe 2 – ALU in VHDL, Lösung

Als nächstes folgt die eigentliche Implementierung der `entity`[1], innerhalb einer `architecture`. Zu Beginn der `architecture` müssen zunächst alle internen Signale deklariert werden.

```
architecture behavioral of alu is
  signal signed_result : signed(8 downto 0);
  signal integer_b : integer;
```

Schließlich beschreiben wir das Verhalten der `architecture`. Der `sll`-Operator (Linksschieben) benötigt einen zweiten Operanden vom Typ `integer`, weshalb wir ein Hilfssignal `integer_b` verwenden. Dem Compiler muss bei der Umwandlung in einen `integer` mitgeteilt werden, ob der Bitvektor *b* vorzeichenbehaftet interpretiert werden soll oder nicht, weshalb zwei Typwandlungen notwendig sind:

```
begin
  integer_b <= to_integer(signed(b));
```

[1] Eine `entity` kann mehrere `architecture`s haben.

Aufgabe 2 – ALU in VHDL, Lösung

Nun folgen die eigentliche durchzuführende Operation, welche wir kombinatorisch implementieren und somit keinen `process` benötigen.

Aufgabe 2 – ALU in VHDL, Lösung

Nun folgen die eigentliche durchzuführende Operation, welche wir kombinatorisch implementieren und somit keinen `process` benötigen. Die arithmetischen Bitoperationen aus `numeric_std` sind für vorzeichenbehaftete und nicht vorzeichenbehaftete Operanden unterschiedlich überladen, weshalb wir, falls nötig, die geforderte Typumwandlung vornehmen:

Aufgabe 2 – ALU in VHDL, Lösung

Nun folgen die eigentliche durchzuführende Operation, welche wir kombinatorisch implementieren und somit keinen `process` benötigen. Die arithmetischen Bitoperationen aus `numeric_std` sind für vorzeichenbehaftete und nicht vorzeichenbehaftete Operanden unterschiedlich überladen, weshalb wir, falls nötig, die geforderte Typumwandlung vornehmen:

```vhdl
with op select signed_result <=
  (signed(a) + signed(b)) when "00",
  (signed(a) - signed(b)) when "01",
  signed(a and b) when "10",
  signed(a sll 1) when "11",
  "00000000" when others;
```

Aufgabe 2 – ALU in VHDL, Lösung

```
with op select signed_result <=
  (signed(a) + signed(b)) when "00",
  (signed(a) - signed(b)) when "01",
  signed(a and b) when "10",
  signed(a sll 1) when "11",
  "00000000" when others;
```

Das `with` *signal* `select`-Konstrukt inferriert einen Multiplexer, der zwischen den vier Ergebnissen auswählt. Entsprechend werden alle vier möglichen Operationen *gleichzeitig* berechnet.

Aufgabe 2 – ALU in VHDL, Lösung

```
with op select signed_result <=
  (signed(a) + signed(b)) when "00",
  (signed(a) - signed(b)) when "01",
  signed(a and b) when "10",
  signed(a sll 1) when "11",
  "00000000" when others;
```

Das `with` *signal* `select`-Konstrukt inferriert einen Multiplexer, der zwischen den vier Ergebnissen auswählt. Entsprechend werden alle vier möglichen Operationen *gleichzeitig* berechnet. Zuletzt muss das Ergebnis zurück in einen `std_logic_vector` umgewandelt und dem Ausgangs-`port` zugewiesen werden:

Aufgabe 2 – ALU in VHDL, Lösung

```
  with op select signed_result <=
    (signed(a) + signed(b)) when "00",
    (signed(a) - signed(b)) when "01",
    signed(a and b) when "10",
    signed(a sll 1) when "11",
    "00000000" when others;
```

Das `with` *signal* `select`-Konstrukt inferriert einen Multiplexer, der zwischen den vier Ergebnissen auswählt. Entsprechend werden alle vier möglichen Operationen *gleichzeitig* berechnet. Zuletzt muss das Ergebnis zurück in einen `std_logic_vector` umgewandelt und dem Ausgangs-`port` zugewiesen werden:

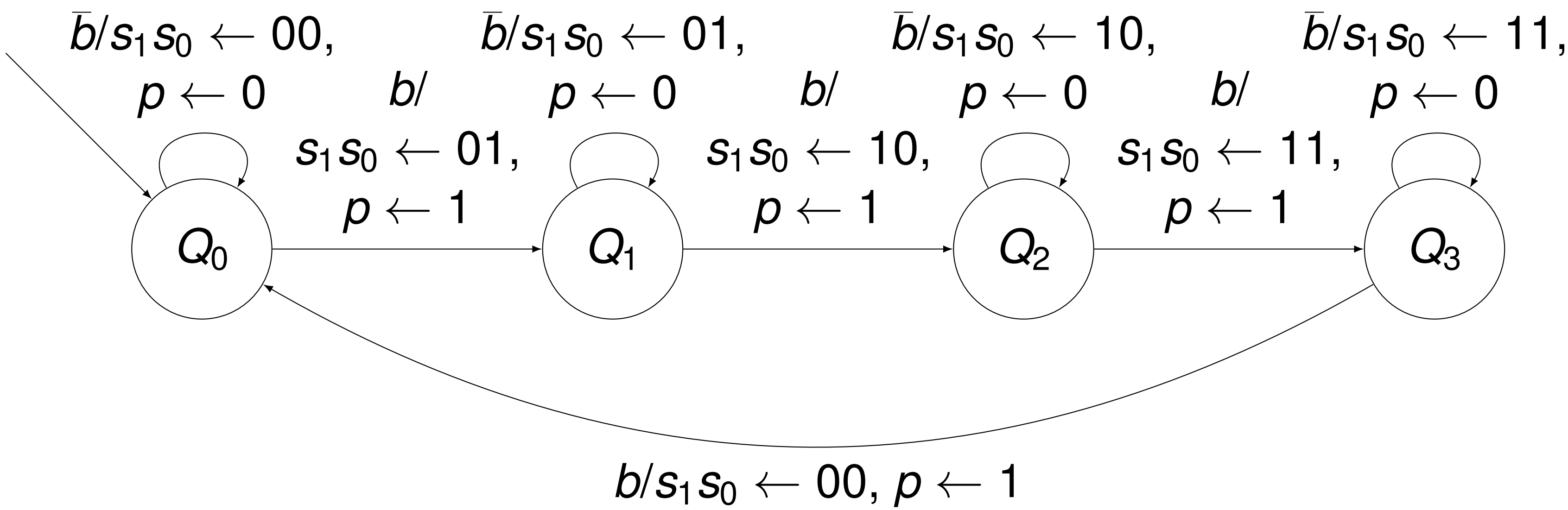
```
  result <= std_logic_vector(signed_result)(7 downto 0);
end architecture;
```

```vhdl
library ieee;
use ieee.std_logic_1164.all; -- std_logic_vector
use ieee.numeric_std.all; -- fuer Arithmetik

entity alu is
    port(
        a, b : in std_logic_vector(7 downto 0);
        op : in std_logic_vector(1 downto 0);
        result : out std_logic_vector(7 downto 0)
    );
end entity alu;

architecture behavioral of alu is
  signal signed_result : signed(8 downto 0);
  signal integer_b : integer;
begin
  integer_b <= to_integer(signed(b));

  with op select signed_result <=
    (signed(a) + signed(b)) when "00",
    (signed(a) - signed(b)) when "01",
    signed(a and b) when "10",
    signed(a sll 1) when "11",
    "00000000" when others;
  result <= std_logic_vector(signed_result)(7 downto 0);
end architecture;
```

Aufgabe 3 – VHDL: Automaten

Aufgabe 3 – VHDL: Automaten

Implementieren Sie den folgenden Mealy-Automaten, der die Armbanduhr aus Übung 11 beschreibt, in VHDL. Der Automat soll mit einem synchronen Reset-Signal in den Anfangszustand zurückgesetzt werden können.



Aufgabe 4 – Komparator

design

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Aufgabe 4 – Komparator

a) Entwickeln Sie einen 1-Bit-Komparator, der zwei Bits a und b miteinander vergleicht und auf den Ausgängen $<$, $>$ und = die Gültigkeit der drei Relationen $a < b$, $a > b$ und $a = b$ ausgibt (1 entspreche wahr). Achten Sie darauf, dass stets genau einer der drei Ausgänge aktiv ist.

Aufgabe 4 – Komparator

b) Entwerfen Sie nun eine digitale Schaltung, deren Eingänge die Ausgänge $K_0 = (<_0, >_0, =_0)$ und $K_1 = (<_1, >_1, =_1)$ zweier Komparatoren sind und die diese *lexikographisch* auf die Ausgabe $(<, >, =)$ reduziert. Dabei soll K_0 nieder- und K_1 höherwertig sein.

Aufgabe 4 – Komparator: lexikographisch?

Lexikographisch

Gegeben sei ein quasigeordnetes Alphabet $(\Sigma, \leq)$, d. i. eine Menge von Zeichen $a_i, b_j \in \Sigma$. Eine Zeichenkette $a = (a_1, a_2, \ldots)$ ist lexikographisch kleiner als eine Zeichenkette $b = (b_1, b_2, \ldots)$, das heißt a liegt in der Sortierung vor b, wenn beim komponentenweisen Vergleich Zeichen für Zeichen ...

(entnommen aus: *Wikipediaeintrag zu „Lexikographischer Ordnung“*)

Aufgabe 4 – Komparator: lexikographisch?

Lexikographisch

Gegeben sei ein quasigeordnetes Alphabet $(\Sigma, \leq)$, d. i. eine Menge von Zeichen $a_i, b_j \in \Sigma$. Eine Zeichenkette $a = (a_1, a_2, \ldots)$ ist lexikographisch kleiner als eine Zeichenkette $b = (b_1, b_2, \ldots)$, das heißt a liegt in der Sortierung vor b, wenn beim komponentenweisen Vergleich Zeichen für Zeichen ...

1. das Zeichen a_i von a mit dem niedrigsten Index i, in dem sich die beiden Zeichenketten unterscheiden, (echt) kleiner ist als das entsprechende Zeichen b_i von b,

(entnommen aus: *Wikipediaeintrag zu „Lexikographischer Ordnung“*)

Aufgabe 4 – Komparator: lexikographisch?

Lexikographisch

Gegeben sei ein quasigeordnetes Alphabet $(\Sigma, \leq)$, d. i. eine Menge von Zeichen $a_i, b_j \in \Sigma$. Eine Zeichenkette $a = (a_1, a_2, \ldots)$ ist lexikographisch kleiner als eine Zeichenkette $b = (b_1, b_2, \ldots)$, das heißt a liegt in der Sortierung vor b, wenn beim komponentenweisen Vergleich Zeichen für Zeichen ...

1. $a_i \leq b_i \wedge b_i \not\leq a_i$,

(entnommen aus: *Wikipediaeintrag zu „Lexikographischer Ordnung“*)

Aufgabe 4 – Komparator: lexikographisch?

Lexikographisch

Gegeben sei ein quasigeordnetes Alphabet $(\Sigma, \leq)$, d. i. eine Menge von Zeichen $a_i, b_j \in \Sigma$. Eine Zeichenkette $a = (a_1, a_2, \ldots)$ ist lexikographisch kleiner als eine Zeichenkette $b = (b_1, b_2, \ldots)$, das heißt a liegt in der Sortierung vor b, wenn beim komponentenweisen Vergleich Zeichen für Zeichen ...

1. $a_i \leq b_i \wedge b_i \not\leq a_i$,
2. oder wenn a ein Präfix von b (d. h. $a_i \leq b_i \wedge b_i \leq a_i$ für alle verfügbaren i), aber kürzer ist.

(entnommen aus: *Wikipediaeintrag zu „Lexikographischer Ordnung“*)

Aufgabe 4 – Komparator: lexikographisch?

Lexikographisch *bei Wörtern mit festen Längen*

Gegeben sei ein quasigeordnetes Alphabet $(\Sigma, \leq)$, d. i. eine Menge von Zeichen $a_i, b_j \in \Sigma$. Ein geordnetes Paar $(a_1, a_2) \in \Sigma^2$ ist *lexikographisch kleiner* als ein geordnetes Paar $(b_1, b_2) \in \Sigma^2$, wenn ...

1. $a_1 < b_1$ oder
2. $a_1 = b_1$ und $a_2 < b_2$

(entnommen aus: *Wikipediaeintrag zu „Lexikographischer Ordnung*“)

Aufgabe 4 – Komparator

c) Betrachten Sie schließlich die in a) und b) entworfenen Schaltungen jeweils als Blackbox mit den gegebenen Schaltsymbolen.
Entwerfen Sie ausschließlich mit diesen Komponenten einen Komparator für vorzeichenlose 4-Bit-Binärzahlen. Welche Möglichkeiten gibt es, die Komponenten zusammenzuschalten, und welche Auswirkungen hat dies auf die benötigte Fläche und den kritischen Pfad?

Korrektur und Besprechung der ersten Miniklausur

design

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Übungen zur Grundlagen der Technischen Informatik

Übung 13 – Arithmetik

Florian Frank

Friedrich-Alexander-Universität Erlangen-Nürnberg

Wintersemester 2018/19

FRIEDRICH-ALEXANDER UNIVERSITÄT ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Was machen wir heute?

Organisatorisches: Vorlesungsevaluation

Was machen wir heute?

Organisatorisches: Vorlesungsevaluation

Aufgabe 1 – Addierer/Subtrahierer

Was machen wir heute?

Organisatorisches: Vorlesungsevaluation

Aufgabe 1 – Addierer/Subtrahierer

Aufgabe 2 – Mehr-Operanden-Addierer

Was machen wir heute?

Organisatorisches: Vorlesungsevaluation

Aufgabe 1 – Addierer/Subtrahierer

Aufgabe 2 – Mehr-Operanden-Addierer

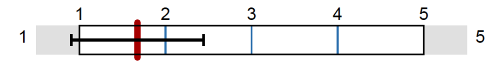
Aufgabe 3 – Arithmetik

Organisatorisches: Vorlesungsevaluation

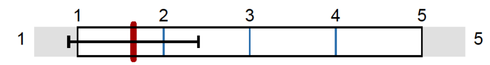
FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Organisatorisches – Vorlesungsevaluation (I)

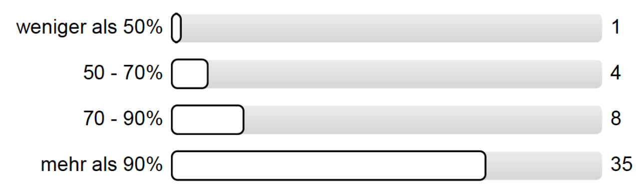
3. Hauptfragen zu Lehrveranstaltung und Übungsleiterin/Übungsleiter



5. Weitere Fragen zu Lehrveranstaltung und Übungsleiterin/Übungsleiter

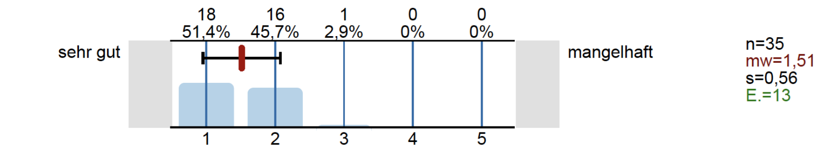


2.7) Ich besuche etwa . . . . Prozent dieser Übung.

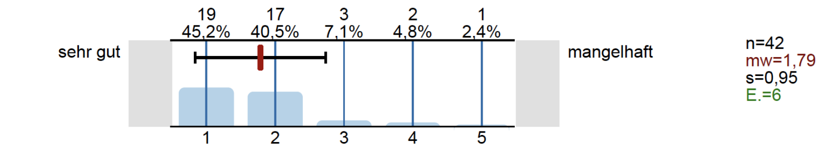


Organisatorisches – Vorlesungsevaluation (II)

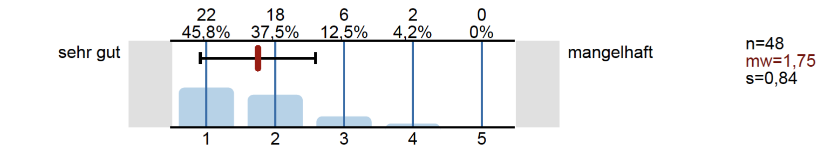
3.1) ►► Die Übung entspricht den im Modulhandbuch eingetragenen Inhalten und Kompetenzen.



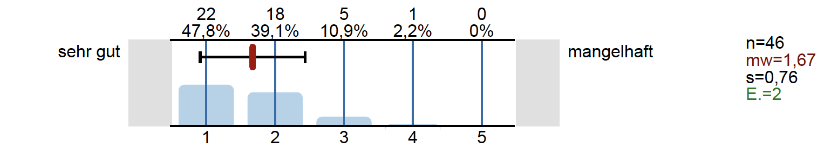
3.2) ►► Wie ist die Einpassung in den Studienverlauf Ihres Studienganges?



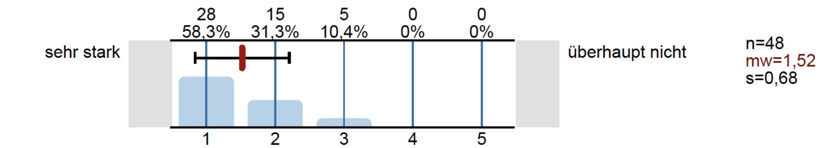
3.3) ►► Wie ist die Übung selbst strukturiert?



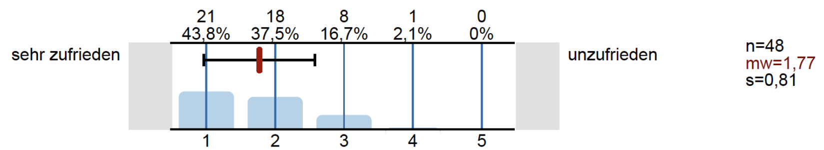
3.4) ►► Wie ist die Übung inhaltlich und organisatorisch mit der zugehörigen Vorlesung abgestimmt?



3.5) ►► Die Übungsleiterin/Der Übungsleiter wirkt engagiert und motiviert bei der Durchführung der Übung.



3.6) ►► Wie zufrieden sind Sie insgesamt mit der Übung:



Organisatorisches – Vorlesungsevaluation (III)

An der Übung hat mir besonders gut gefallen ...

- Wiederholung von Themen aus der Vorlesung, sowie viele Übungsaufgaben
- trägt deutlich zum Verständnis der VL bei
- 1 und 0
- Alles ist sehr gut verständlich und hilfreich
- Ausfuhrlich erklärt
- Der Tutor ist sehr engagiert. Die Übungsgruppen habe eine sinnvolle Anzahl an Teilnehmern. Das Angebot hilft, den Vorlesungsstoff zu uben und zu festigen. Man erhält eine physische Kopie des Übungsblattes.
- Die Art des Unterrichts, Lieblings Dozent.
- Es wird immer erst noch einmal die zugrundeliegende Grammatik wiederholt und danach erst die Aufgaben gemacht.
- Fragen werden beantwortet. Wenn nicht sofort dann in der nächsten Übungsstunde.
- Gute Beispiel zum Inhalt der Vorlesung.
- Lösungen und step-to-step Lösung in powerpoint Form sind sehr gut.

Organisatorisches – Vorlesungsevaluation (IV)

An der Übung hat mir besonders gut gefallen ...

- Mein Tutor (Florian Frank) ist unglaublich engagiert und motiviert! Einer der besten Tutoren, die ich je hatte. Die Tafelanschriften sind super; lesbar, farbig, ubersichtlich. Weiter so!
- Sehr gute Folien mit Erklärung und Lösung zu jeder Übung! Erleichtert das Vor- oder Nachbereiten massiv
- Wiederholung der teilweise zu theoretischen Vorlesung
- Übungsaufgaben sind nah an Klausuraufgaben, ähnlich gestellt etc
- Sehr gute Tafelubungsfolien
- Gut Strukturiert

Organisatorisches – Vorlesungsevaluation (V)

An der Übung hat mir nicht so gut gefallen und ich schlage zur Verbesserung vor ...

- Das Rechnen mit Fließkommazahlen ist etwas, das entweder garnicht oder ausfuhrlicher besprochen werden sollte.
  Die Komplexität dessen ist zu hoch, als dass dafur die wenigen Beispiele ausreichen, die in den Übungen ind noch dazu kurzer Zeit präsentiert wurden.
- Der "Dozierstil der einzelnen Übungsleiter ist SEHR verschieden.
- Der Umfang der Aufgaben ist fur die Art und Weise wie die Übung gehalten wird zu groß! Die Aufgaben sollen, laut Aussage des Übungsleiters, in der Übung gemeinsam bearbeitet werden. Dafur reicht die Zeit nicht und der Tutor uberzieht nahezu jedes Mal mindestens 20 Minuten. Die Meiste Zeit wird darauf verwendet die Aufgaben niederzuschreiben. Dadurch wird man so uberfahren, dass es unmöglich fur mich ist Fragen zu Lucken zu stellen, da ich weiß, dass zu wenig Zeit fur die Übung zur Verfugung steht! Eventuell ist es zielfuhrender, wenn weniger Aufgaben gestellt werden, welche dann intensiver besprochen werden können. (*mehrfache Vorkommen*)

Organisatorisches – Vorlesungsevaluation (VI)

An der Übung hat mir nicht so gut gefallen und ich schlage zur Verbesserung vor ...

- Die Stoffverteilung ist sehr ungleichmäßig und tendiert auf zu viel Stoff pro Übung, bzw. zu viele Aufgaben pro Übung. Weniger, dafur detailliertere Übungen wären den Jetzigen Vorzuziehen
- Die Folien könnten ggf. schon etwas vorher online sein, hat man zB am Montag seine Übung und möchte am Dienstag oder Mittwoch nochmal etwas in den Folien nachsehen, geht das nicht..
- Die Übungen sind so, das zumindestens bei uns keiner die Aufgaben in der Übung gemacht hat, sonder nur in der Übung mitgemacht. Ausserdem wären (mini)Klausur Lösungen gut
- In manchen Wochen zu wenig Zeit fur die Übungsaufgaben
- Manche relevanten Inhalte der Vorlesung werden in der Übung nicht explizit behandelt (hier v.a. JPEG).
- Langsamer, freier und vorbereiter
- Wir hängen eine bis zwei Wochen hinter der Vorlesung her
- Keine

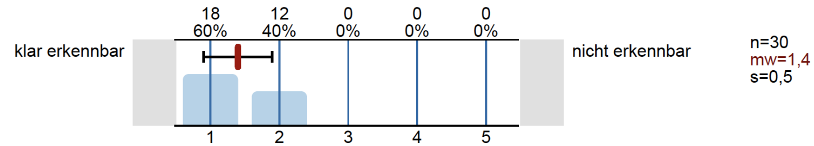
Organisatorisches – Vorlesungsevaluation (VII)

Zur Lehrveranstaltung möchte ich im Übrigen anmerken ...

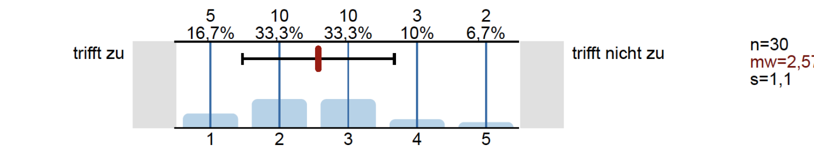
- All in all a very useful course that is worth attending. Very helpful in clearing up topics which have been cut short in the lecture or just haven't been understood by the student
- Das ist jetzt mein 3. Mal, die Übung macht so viel Spaß, dass ich einfach mehrmals kommen wollte
- Der Tutor scheint ein wirklicher Streber zu sein :)
- Die Übungen und die Vorlesung treffen sich inhaltlich schon, jedoch fuhle ich mich einzig durch durcharbeiten der Vorlesungsmaterialien nicht auf die Aufgaben vorbereitet. Es bedarf weiterer Ressourcen sich hier effektiv vorzubereiten bzw die Hausaufgaben tatsächlich im Vorhinhein zuhause zu erledigen.
- Florian macht phänomenale Tafelanschriften!
- Florian Frank ist ein hervorragender Tutor
- Zu viel Stoff, dass man selbst etwas lösen könnte. Da es aber insgesamt nur 5ECTS fur GTI-VL gibt, kann man da unmöglich noch zusätzliche Zeit reinstecken die Übungsaifgaben schon im Voraus zu lösen

Organisatorisches – Vorlesungsevaluation

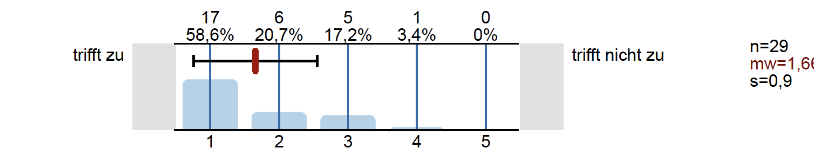
5.2) Zielsetzungen und Schwerpunkte des Übungsinhalts sind:



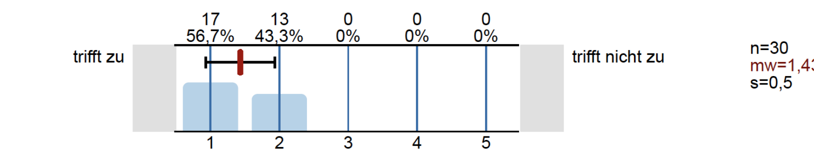
5.3) Ich werde gut zum selbstständigen Lösen von Aufgaben angeleitet.



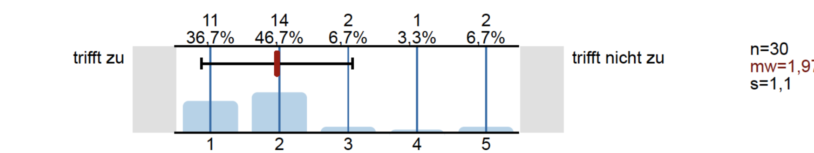
5.4) Die Anwendbarkeit des Übungsstoffes wird z.B. durch Beispiele gut verdeutlicht.



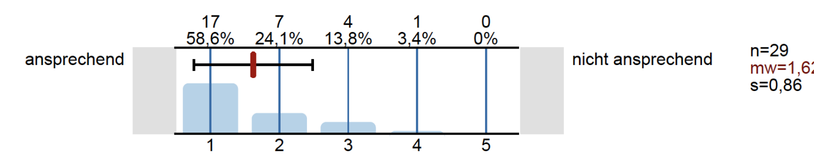
5.5) Die Übungsform (Aufgabenbehandlung, Programmieren, etc.) ist gut zur Vermittlung des Stoffes geeignet.



5.6) Die Präsentation von Aufgaben und Lösungen ist nachvollziehbar, es ist genügend Zeit zum Mitdenken vorhanden.

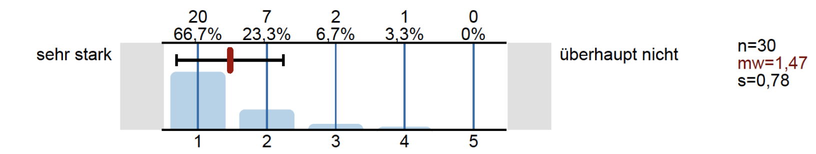


5.7) Der Präsentationsstil der Übungsleiterin/des Übungsleiters ist:

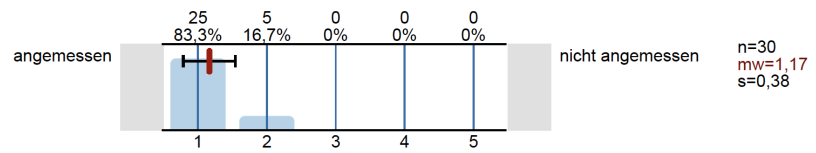


Organisatorisches – Vorlesungsevaluation

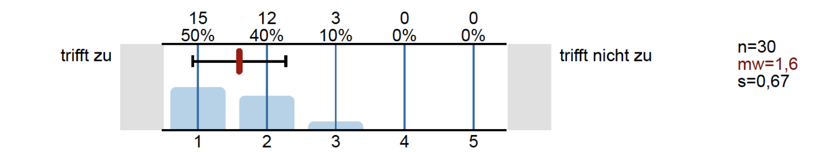
5.8) Die Übungsleiterin/Der Übungsleiter geht auf Fragen und Belange der Studierenden ein.



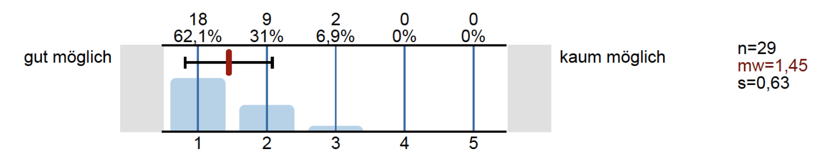
5.9) Der Einsatz und das Zusammenspiel von Medien (Tafel, Overhead-Projektor, Beamer, etc.) ist:



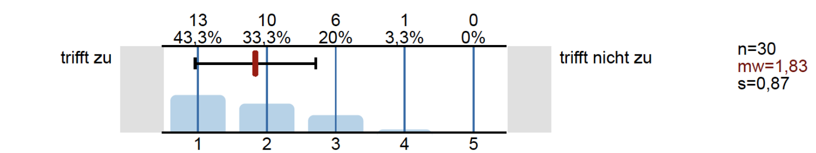
5.10) Die zur Verfügung gestellten Unterlagen sind in Menge und Qualität den Zielen der Übung angemessen.



5.11) Anhand des erarbeiteten Übungsmaterials ist die Vertiefung des Vorlesungs-/Modulinhalts:

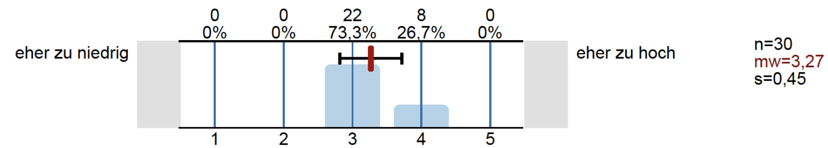


5.12) Der Bezug zu den Prüfungsanforderungen wird hergestellt.

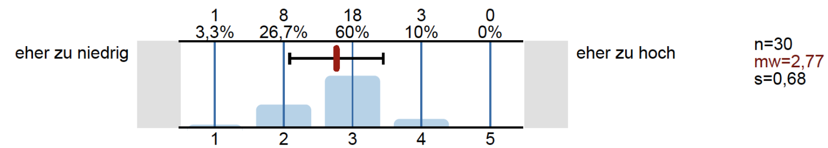


Organisatorisches – Vorlesungsevaluation

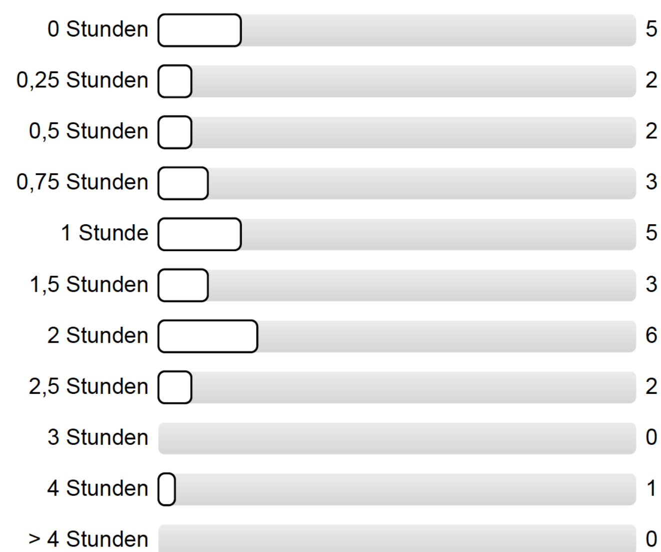
6.1) Der Schwierigkeitsgrad der Übung ist:



6.3) Meinen zeitlichen Durchschnittsaufwand für diese Übung finde ich:



6.2) Mein Durchschnittsaufwand für Vor- und Nachbereitung dieser Übung beträgt pro Woche:



n=29

Aufgabe 1 – Addierer/Subtrahierer

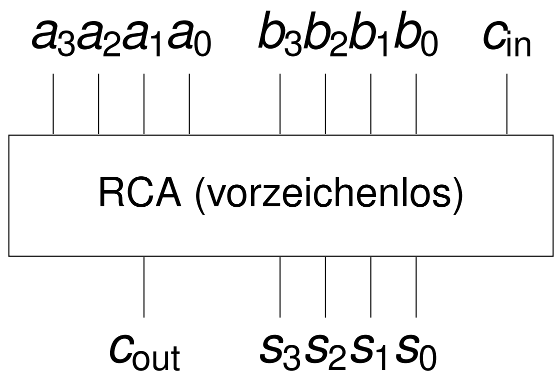
FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG

TECHNISCHE FAKULTÄT

Aufgabe 1 – Addierer/Subtrahierer

a) Realisieren Sie sowohl einen Halbaddierer als auch einen Volladdierer ausschließlich mit NAND-Gattern. Bestimmen Sie jeweils die Anzahl der verwendeten Gatter und die Länge des kritischen Pfades.

b) Erstellen Sie aus den Volladdiererzellen aus a) einen Ripple-Carry-Addierer (RCA) für 4 Bit breite Operanden:



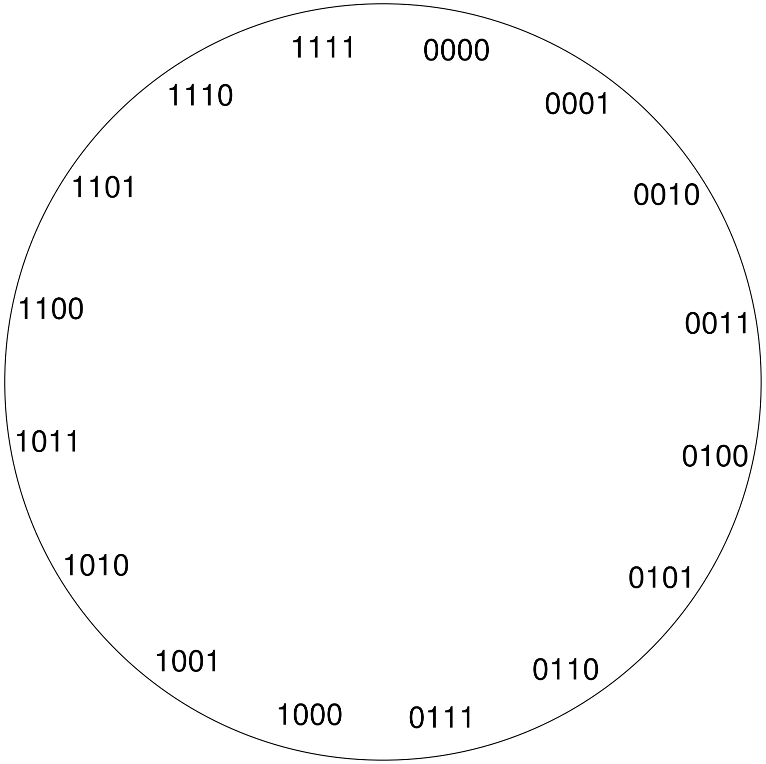
Wieviele Gatter enthält der kritische Pfad des gesamten Schaltnetzes nun?

Aufgabe 1 – Addierer/Subtrahierer

c) Erweitern Sie den RCA aus b) nun um eine Subktraktionsfunktion. Es soll $A - B$ berechnet werden, wenn der zusätzliche Steuereingang *sub* aktiv ist (ist *sub* inaktiv, soll weiterhin $A + B$ berechnet werden). Geben Sie jeweils eine Lösung an, die i) das 1er-Komplement und ii) das 2er-Komplement zur Berechnung nutzt.

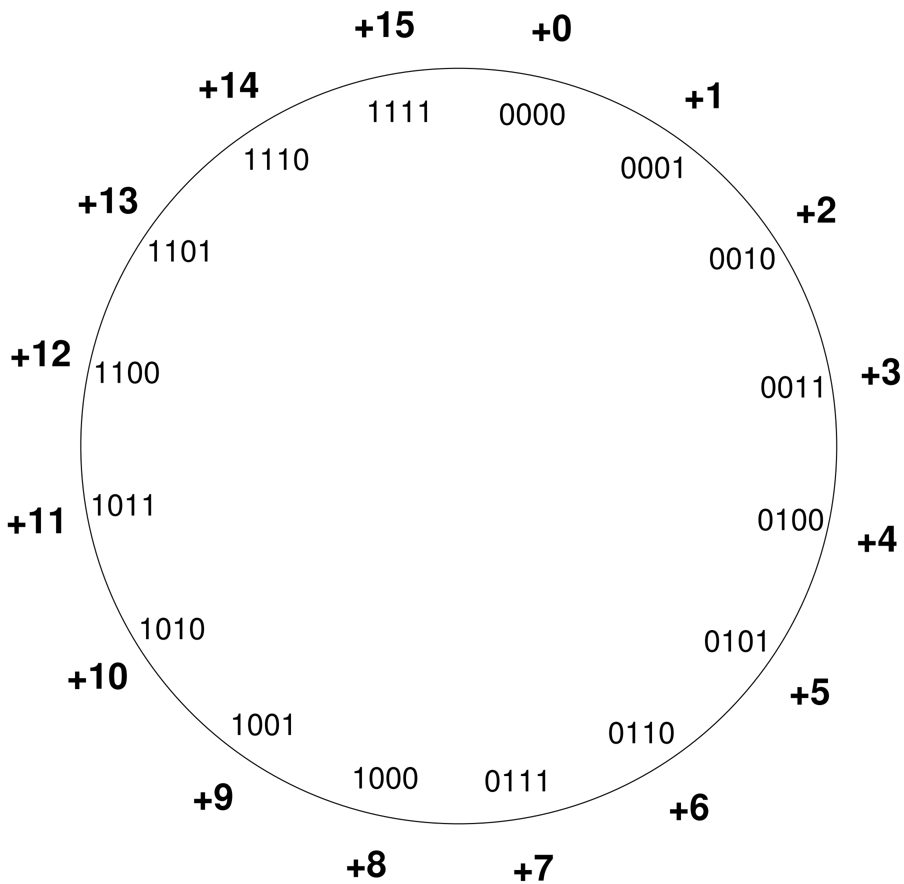
Wiederholung: Blatt 3, Aufgabe 1 – Zahlendarstellungen: Theorie

Vorzeichenlose Zahlendarstellung –



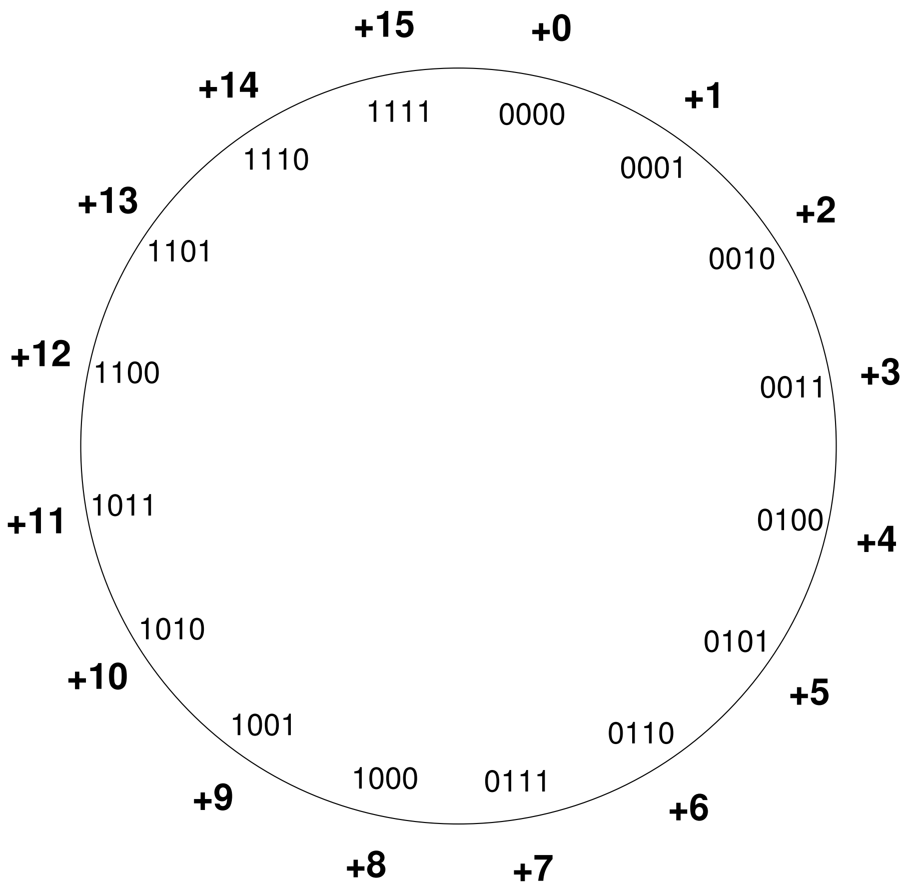
Wiederholung: Blatt 3, Aufgabe 1 – Zahlendarstellungen: Theorie

Vorzeichenlose Zahlendarstellung –



Wiederholung: Blatt 3, Aufgabe 1 – Zahlendarstellungen: Theorie

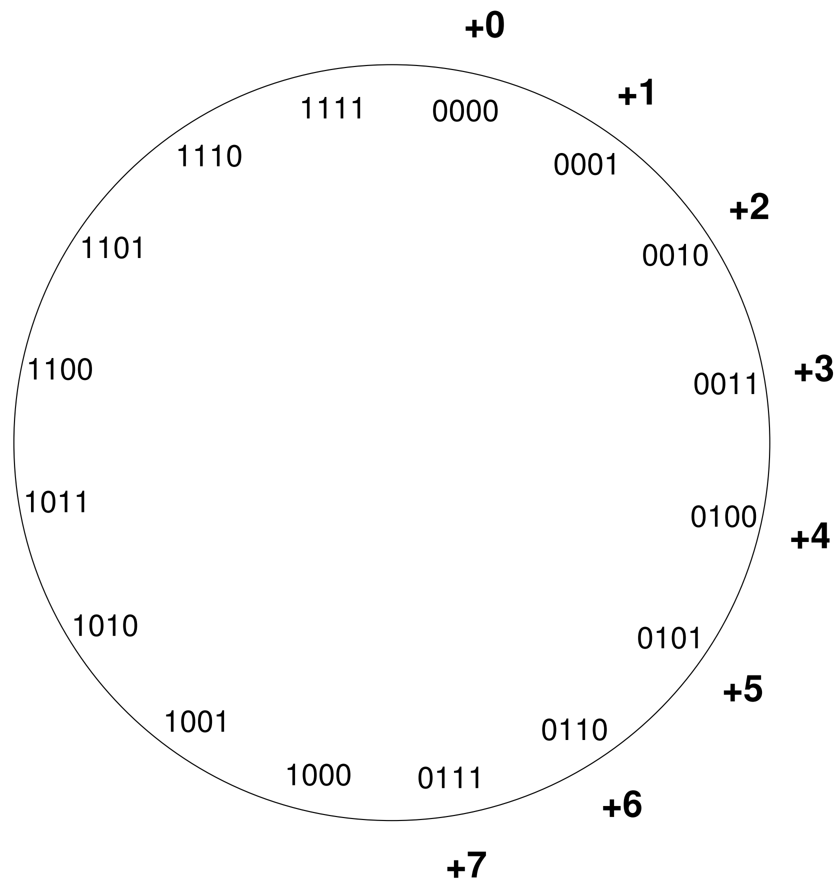
Vorzeichenlose Zahlendarstellung –



Wertebereich einer n bit breiten Zahl: $\left[0, 2^n - 1\right]$

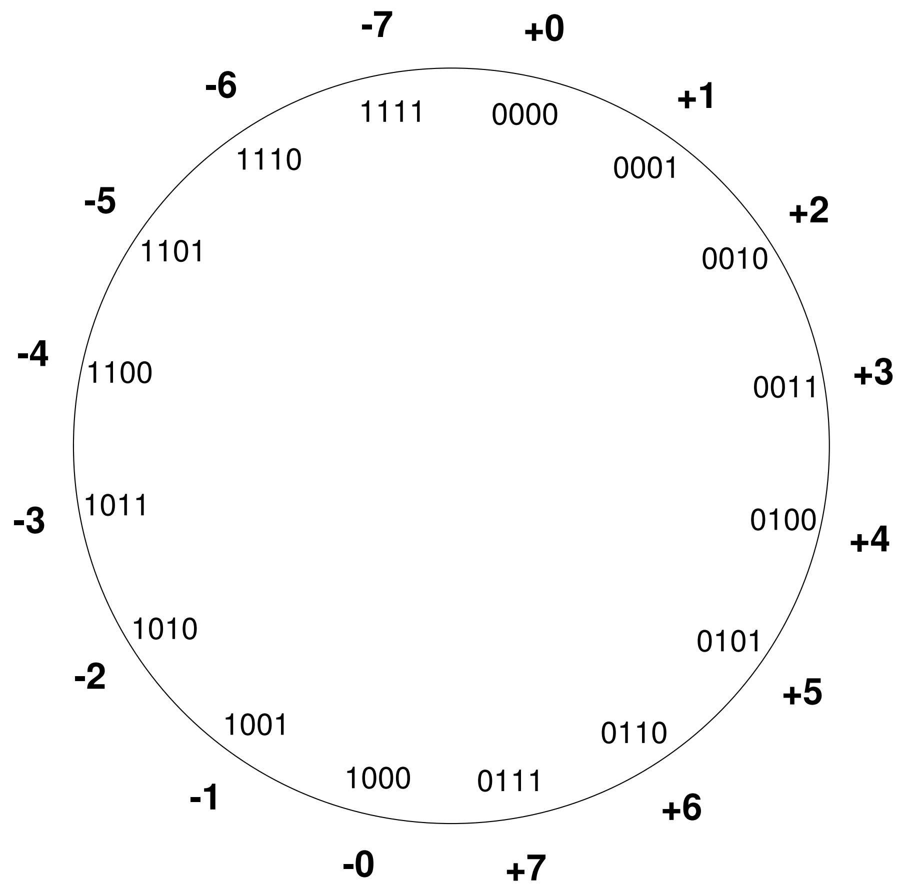
Wiederholung: Blatt 3, Aufgabe 1 – Zahlendarstellungen: Theorie

Vorzeichen-/Betragsdarstellung



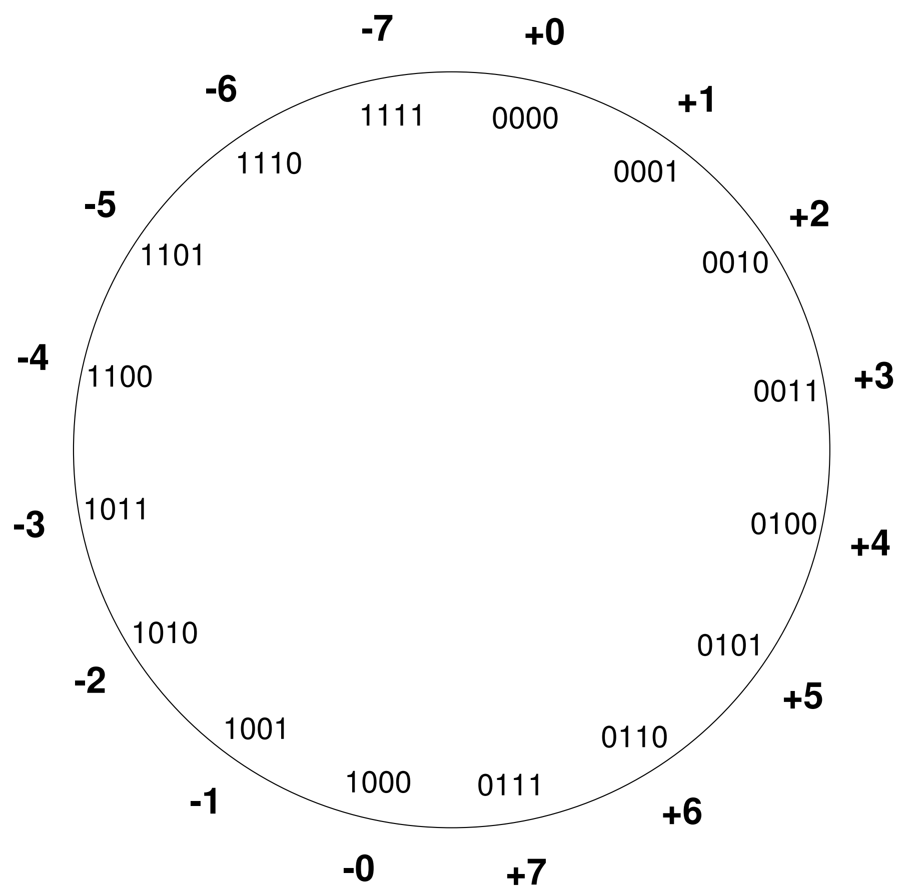
Wiederholung: Blatt 3, Aufgabe 1 – Zahlendarstellungen: Theorie

Vorzeichen-/Betragsdarstellung



Wiederholung: Blatt 3, Aufgabe 1 – Zahlendarstellungen: Theorie

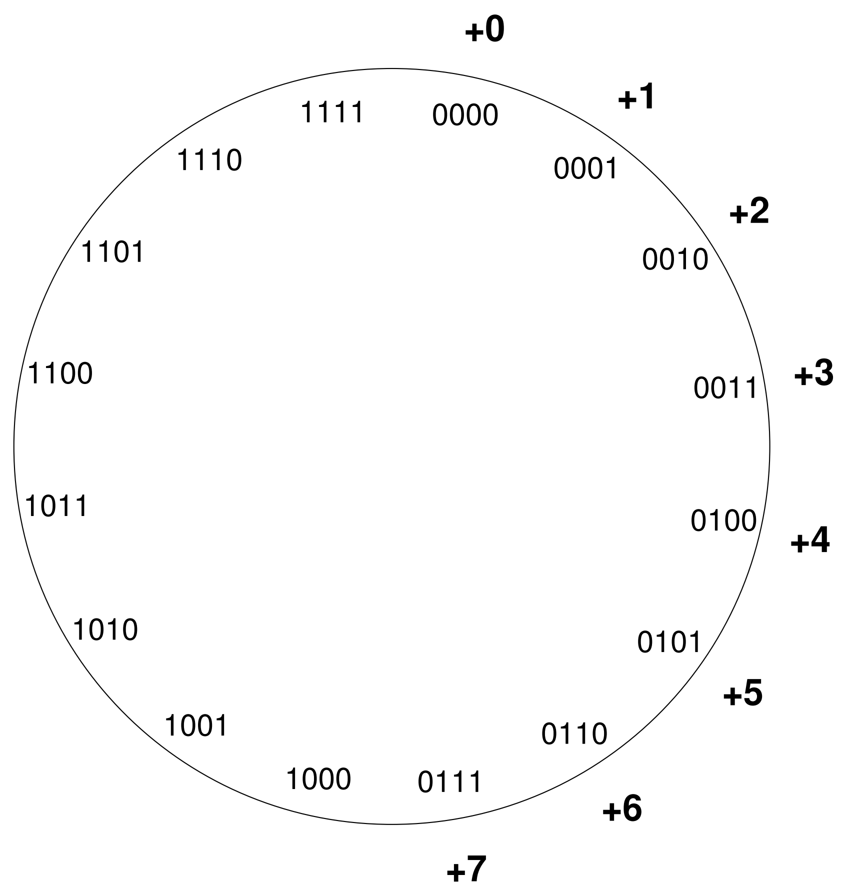
Vorzeichen-/Betragsdarstellung



Wertebereich einer n bit breiten Zahl: $\left[-2^{n-1}+1, 2^{n-1}-1\right]$

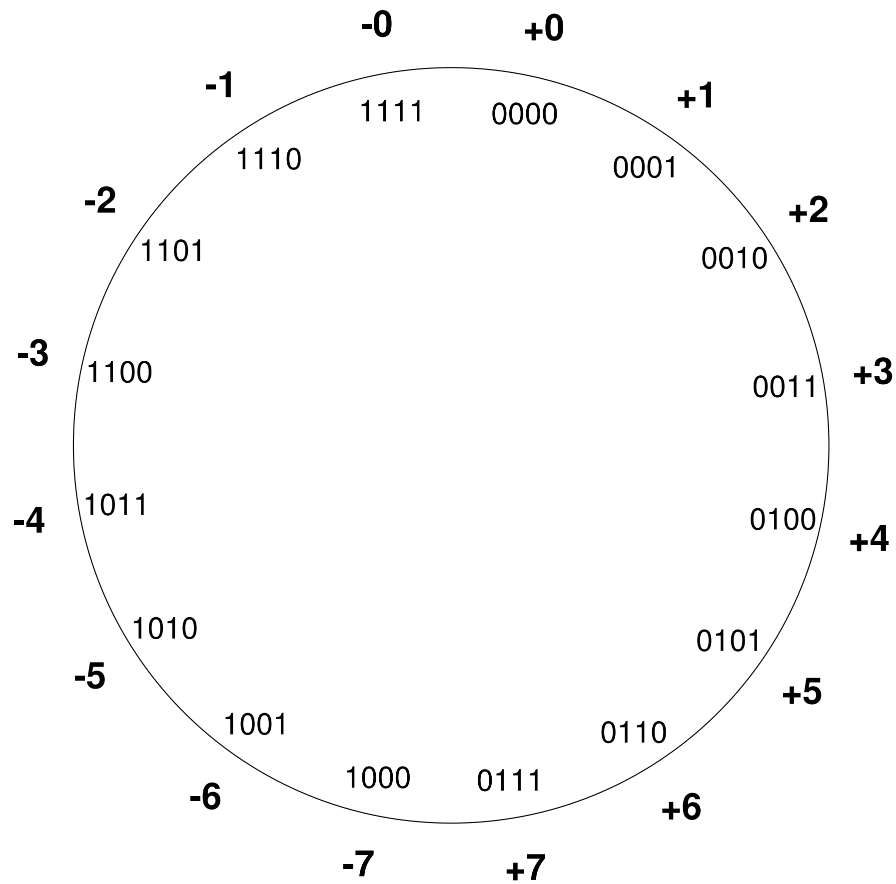
Wiederholung: Blatt 3, Aufgabe 1 – Zahlendarstellungen: Theorie

Einerkomplement



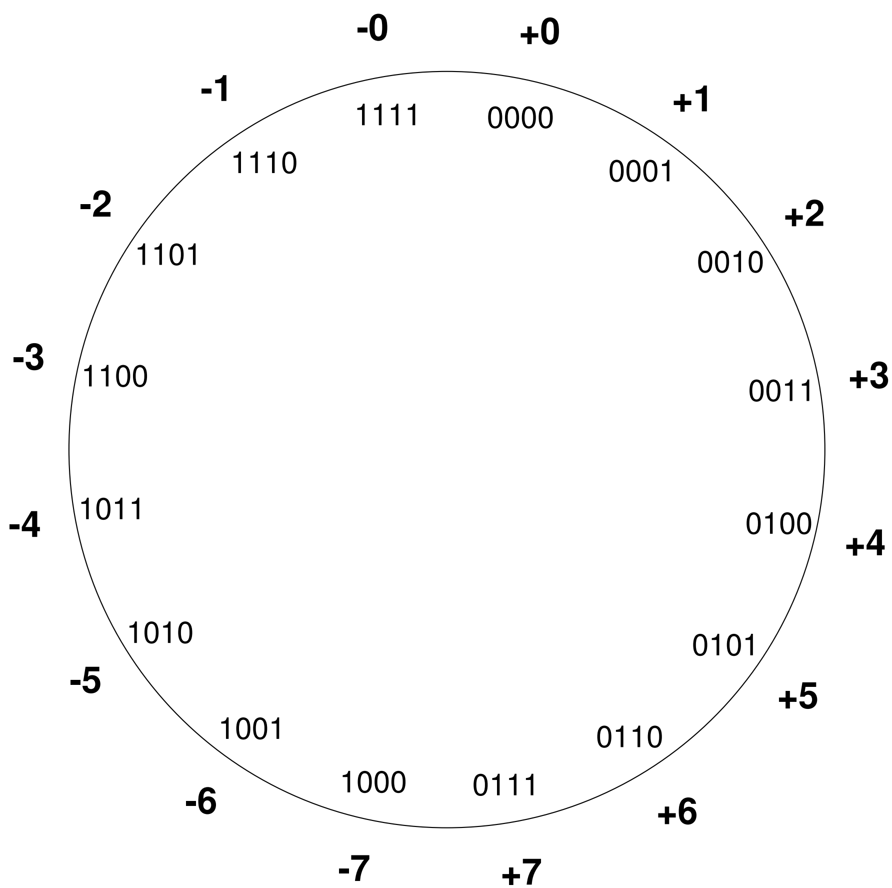
Wiederholung: Blatt 3, Aufgabe 1 – Zahlendarstellungen: Theorie

Einerkomplement



Wiederholung: Blatt 3, Aufgabe 1 – Zahlendarstellungen: Theorie

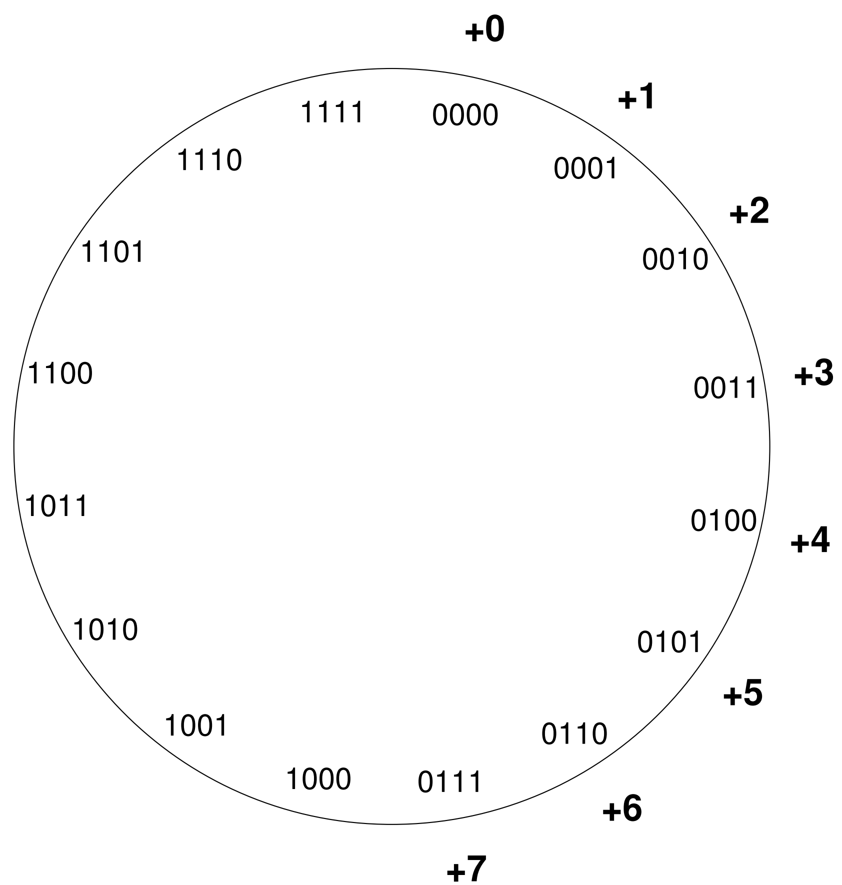
Einerkomplement



Wertebereich einer n bit breiten Zahl: $\left[-2^{n-1}+1, 2^{n-1}-1\right]$

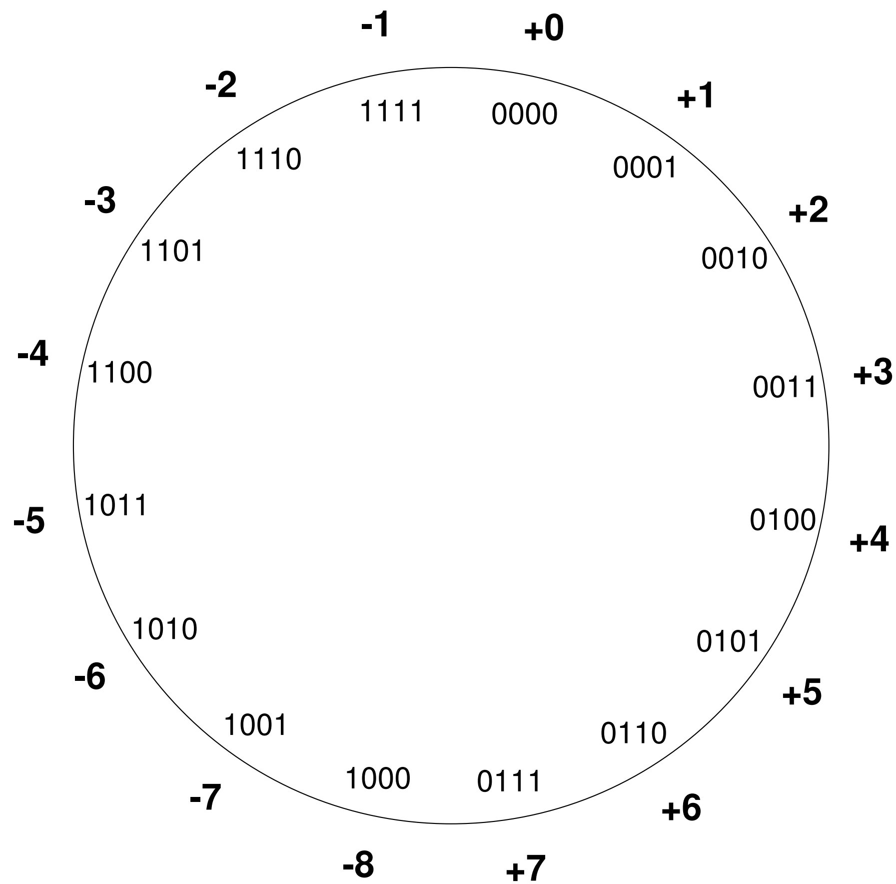
Wiederholung: Blatt 3, Aufgabe 1 – Zahlendarstellungen: Theorie

Zweierkomplement



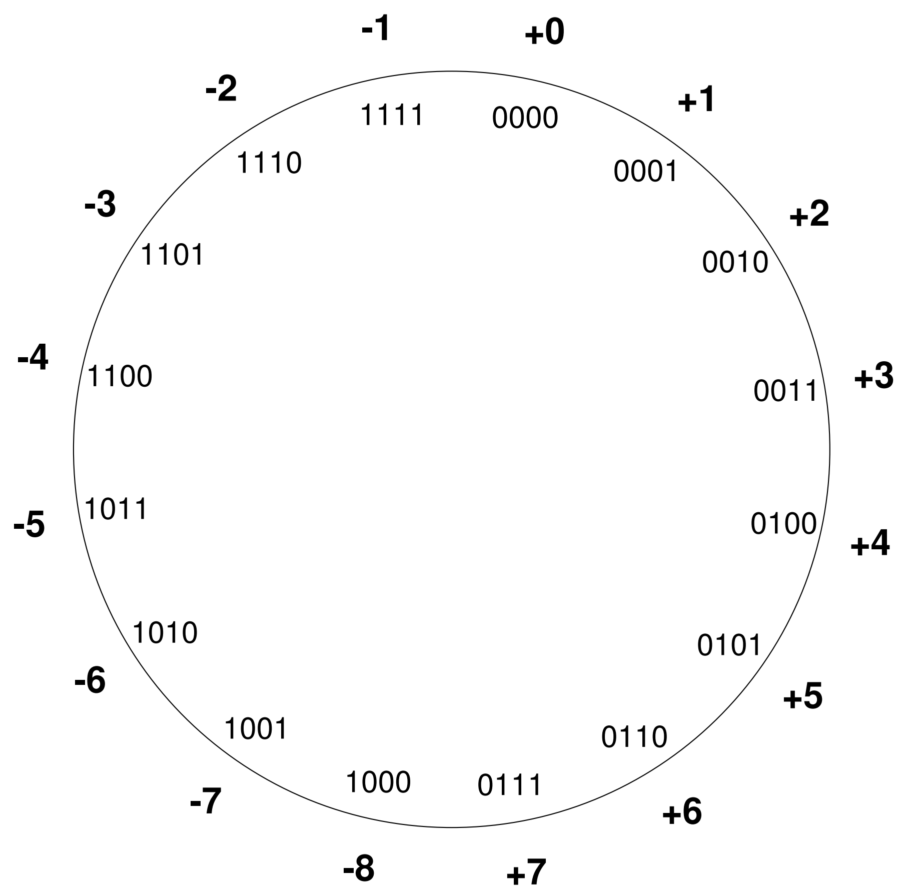
Wiederholung: Blatt 3, Aufgabe 1 – Zahlendarstellungen: Theorie

Zweierkomplement



Wiederholung: Blatt 3, Aufgabe 1 – Zahlendarstellungen: Theorie

Zweierkomplement



Wertebereich einer n bit breiten Zahl: $\left[-2^{n-1}, 2^{n-1} - 1\right]$

Wiederholung: Blatt 3, Aufgabe 1 – Zahlendarstellungen: Theorie

1er-Komplement

Im 1er-Komplement erhalten wir die „negative" Zahl $-a$ einer Zahl a, indem wir **jede einzelne Stelle negieren**.

$$a = (a_n \ldots a_0) \rightarrow -a = (\overline{a_n} \ \ldots \ \overline{a_0})$$

2er-Komplement

Im 2er-Komplement erhalten wir die „negative" Zahl $-a$ einer Zahl a, indem wir **die negative Zahl im 1er-Komplement bilden und** 1 **addieren.**

$$a = (a_n \ldots a_0) \rightarrow -a = (\overline{a_n} \ \ldots \ \overline{a_0}) + 1$$

Aufgabe 1 – Addierer/Subtrahierer

c) Erweitern Sie den RCA aus b) nun um eine Subktraktionsfunktion. Es soll $A - B$ berechnet werden, wenn der zusätzliche Steuereingang *sub* aktiv ist (ist *sub* inaktiv, soll weiterhin $A + B$ berechnet werden). Geben Sie jeweils eine Lösung an, die i) das 1er-Komplement und ii) das 2er-Komplement zur Berechnung nutzt.

d) Entwerfen Sie schließlich eine Komponente, die bestimmt, ob ein arithmetischer Überlauf vorliegt, und für beide Varianten aus c) verwendet werden kann.

Aufgabe 2 – Mehr-Operanden-Addierer

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 2 – Mehr-Operanden-Addierer

Entwerfen Sie ein Schaltnetz für die Addition von vier 4 Bit langen Summanden U, V, W und X, das beispielsweise für die Addition von Teilprodukten eines Multiplizierers benutzt werden könnte:

$$U + V + W + X = \begin{array}{ccccccc} & & & u_3 & u_2 & u_1 & u_0 \\ + & & & v_3 & v_2 & v_1 & v_0 \\ + & & & w_3 & w_2 & w_1 & w_0 \\ + & & & x_3 & x_2 & x_1 & x_0 \\ \hline = & s_5 & s_4 & s_3 & s_2 & s_1 & s_0 \end{array}$$

Verwenden Sie drei Ripple-Carry-Addierer, um zuerst $U + V$ und $W + X$ zu berechnen und anschließend die beiden Teilsummen zu addieren. Gehen Sie davon aus, dass nur die Volladdierer aus Aufgabe 2a) verwendet werden und annotieren Sie die Gatterverzögerungen an deren Ausgänge.

Aufgabe 3 – Arithmetik

design

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

Aufgabe 3 – Arithmetik

a) Multiplizieren Sie die beiden Binärzahlen $A = 0100110$ und $B = 0101$ durch Anwendung der Methode, die bei der Implementierung eines sequentiellen Multiplizierers zum Einsatz kommt. Geben Sie die einzelnen Schritte und das Ergebnis explizit an.

b) Dividieren Sie die Binärzahl $A = 0111101$ durch die Binärzahl $B = 0110$ mit dem Non-Restoring-Divisionsverfahren. Geben Sie die einzelnen Schritte sowie den Quotienten Q und Rest R explizit an.

Aufgabe 3 – Arithmetik: Arraymultiplizierer

Generelle Frage: Wie können wir zwei n/m-bit Binärzahlen $a = a_n ... a_0$, $b = b_m ... b_0$, $n, m \geqslant 1$ multiplizieren?

Aufgabe 3 – Arithmetik: Arraymultiplizierer

Generelle Frage: Wie können wir zwei n/m-bit Binärzahlen $a = a_n \dots a_0$, $b = b_m \dots b_0$, $n, m \geqslant 1$ multiplizieren?
Erste „naive“ Idee: Ansatz der „schriftlichen Multiplikation“

Aufgabe 3 – Arithmetik: Arraymultiplizierer

Generelle Frage: Wie können wir zwei n/m-bit Binärzahlen $a = a_n \ldots a_0$, $b = b_m \ldots b_0$, $n, m \geqslant 1$ multiplizieren?

Erste „naive“ Idee: Ansatz der „schriftlichen Multiplikation“

Um zwei Binärzahlen zu multiplizieren, gehen wir genauso wie bei der schriftlichen Multiplikation vor.

Aufgabe 3 – Arithmetik: Arraymultiplizierer

Generelle Frage: Wie können wir zwei n/m-bit Binärzahlen $a = a_n \ldots a_0$, $b = b_m \ldots b_0$, $n, m \geqslant 1$ multiplizieren?

Erste „naive“ Idee: Ansatz der „schriftlichen Multiplikation“

Um zwei Binärzahlen zu multiplizieren, gehen wir genauso wie bei der schriftlichen Multiplikation vor.

Sei im Beispiel $n = 4$ und $m = 2$:

$$
\begin{array}{ccccccccc}
 & a_4 & a_3 & a_2 & a_1 & a_0 & \times & b_2 & b_1 & b_0 \\
\hline
 & & & & & a_4 b_0 & a_3 b_0 & a_2 b_0 & a_1 b_0 & a_0 b_0 \\
 & & & & a_4 b_1 & a_3 b_1 & a_2 b_1 & a_1 b_1 & a_0 b_1 & \\
+ & & & a_4 b_2 & a_3 b_2 & a_2 b_2 & a_1 b_2 & a_0 b_2 & & \\
\hline\hline
 & & p_7 & p_6 & p_5 & p_4 & p_3 & p_2 & p_1 & p_0
\end{array}
$$

Aufgabe 3 – Arithmetik: Arraymultiplizierer

Generelle Frage: Wie können wir zwei n/m-bit Binärzahlen $a = a_n \ldots a_0$, $b = b_m \ldots b_0$, $n, m \geqslant 1$ multiplizieren?

Erste „naive“ Idee: Ansatz der „schriftlichen Multiplikation“

Um zwei Binärzahlen zu multiplizieren, gehen wir genauso wie bei der schriftlichen Multiplikation vor.

Sei im Beispiel $n = 4$ und $m = 2$:

$$
\begin{array}{cccccccccc}
 & a_4 & a_3 & a_2 & a_1 & a_0 & \times & b_2 & b_1 & b_0 \\
\hline
 & & & & & a_4b_0 & a_3b_0 & a_2b_0 & a_1b_0 & a_0b_0 \\
 & & & & a_4b_1 & a_3b_1 & a_2b_1 & a_1b_1 & a_0b_1 & \\
+ & & & a_4b_2 & a_3b_2 & a_2b_2 & a_1b_2 & a_0b_2 & & \\
\hline\hline
 & & p_7 & p_6 & p_5 & p_4 & p_3 & p_2 & p_1 & p_0
\end{array}
$$

Eine jeder dieser konjunktiven Verknüfungen lässt sich „leicht“ ausrechnen ($\rightsquigarrow$ UND-Gatter)

Aufgabe 3 – Arithmetik: Arraymultiplizierer

Generelle Frage: Wie können wir zwei n/m-bit Binärzahlen $a = a_n \ldots a_0$, $b = b_m \ldots b_0$, $n, m \geqslant 1$ multiplizieren?

Erste „naive“ Idee: Ansatz der „schriftlichen Multiplikation“

Um zwei Binärzahlen zu multiplizieren, gehen wir genauso wie bei der schriftlichen Multiplikation vor.

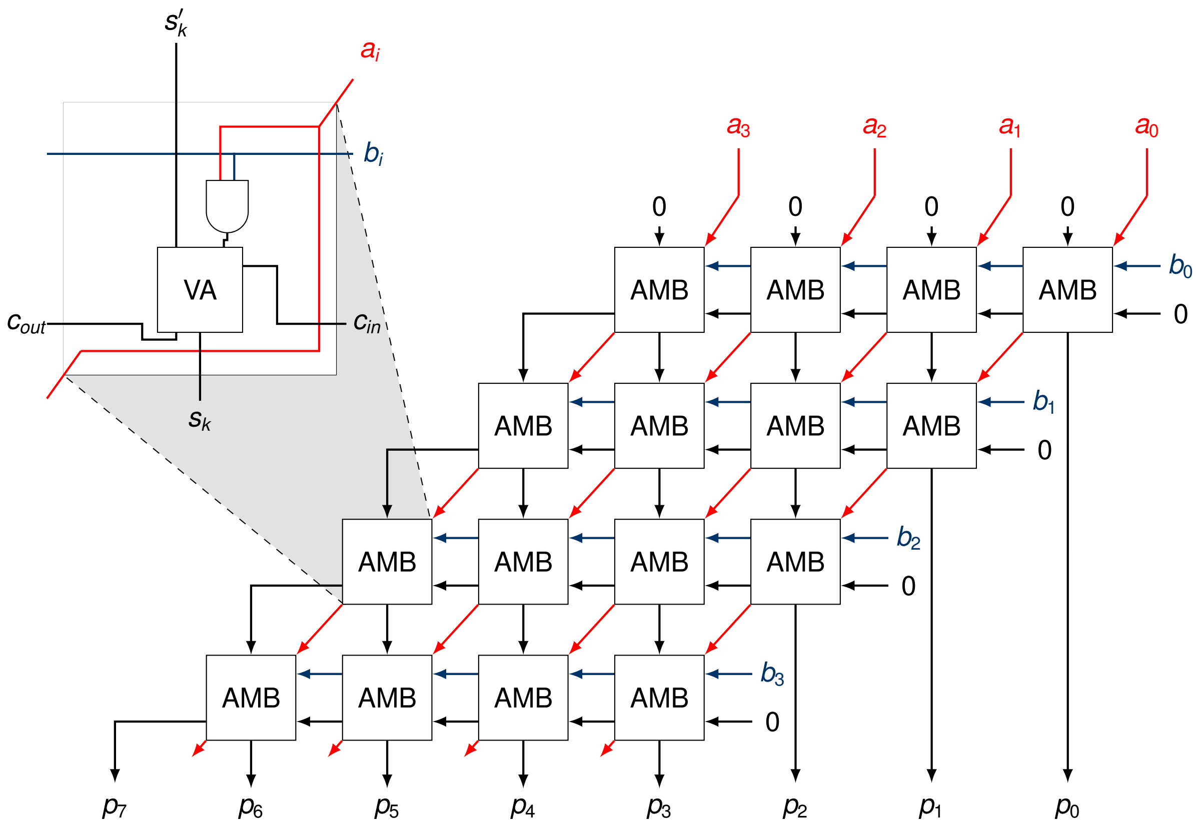
Sei im Beispiel $n = 4$ und $m = 2$:

$$
\begin{array}{cccccccccc}
 & a_4 & a_3 & a_2 & a_1 & a_0 & \times & b_2 & b_1 & b_0 \\
\hline
 & & & & & a_4 b_0 & a_3 b_0 & a_2 b_0 & a_1 b_0 & a_0 b_0 \\
 & & & & a_4 b_1 & a_3 b_1 & a_2 b_1 & a_1 b_1 & \boxed{a_0 b_1} & \\
+ & & & a_4 b_2 & a_3 b_2 & a_2 b_2 & a_1 b_2 & a_0 b_2 & & \\
\hline\hline
 & p_7 & p_6 & p_5 & p_4 & p_3 & p_2 & p_1 & p_0 &
\end{array}
$$

Eine jeder dieser konjunktiven Verknüfungen lässt sich „leicht“ ausrechnen ($\rightsquigarrow$ UND-Gatter)

Aufgabe 3 – Arithmetik: Arraymultiplizierer

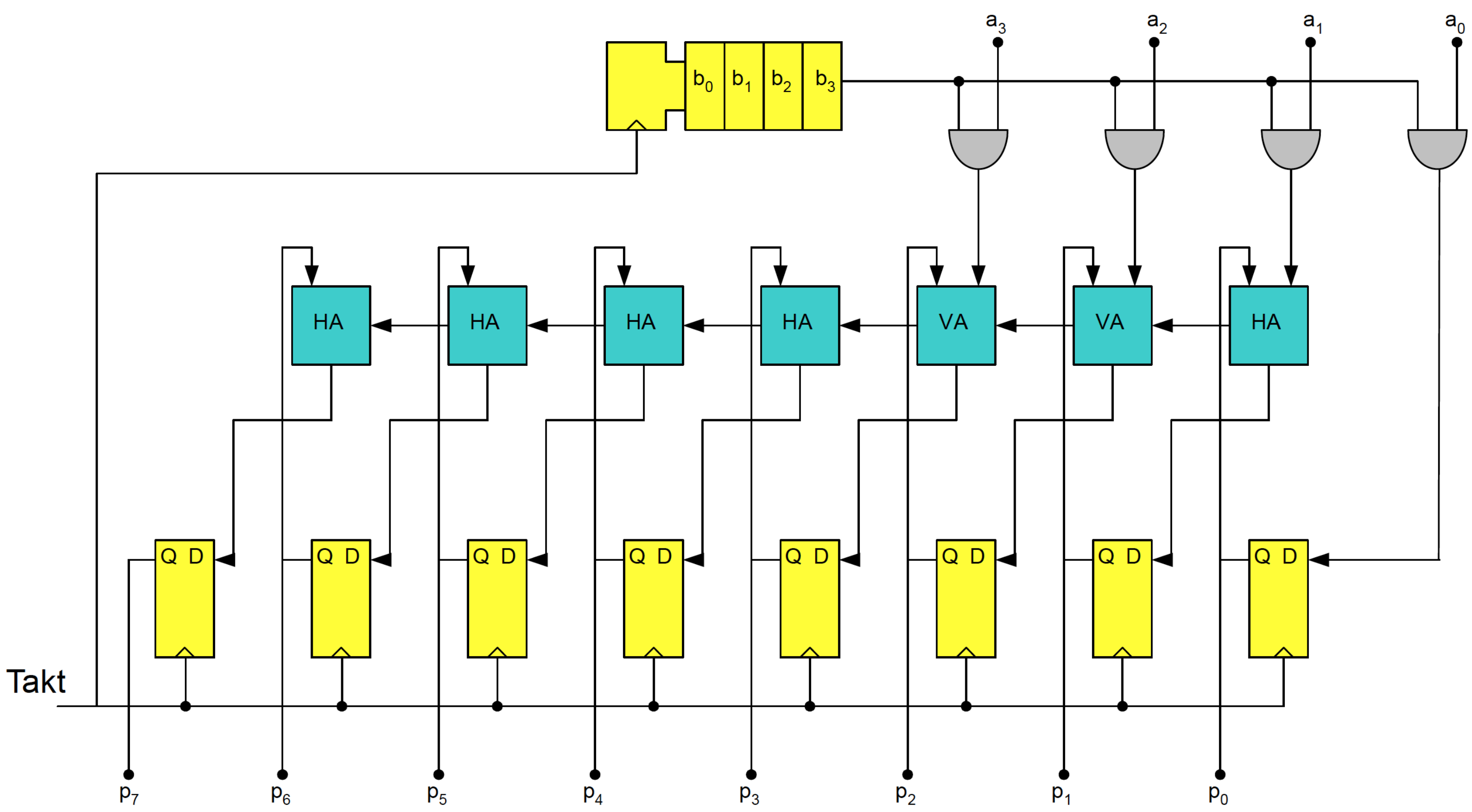
Man nennt einen solchen Multiplizierer, der auf diese Art multipliziert, „Arraymultiplizierer“. Sein Gatterschaltbild sieht wie folgt aus:



Problem: hoher Hardwareaufwand und sehr lange Laufzeit ($(3 \cdot n - 2) \cdot \tau$) mit τ als Latenz eines Gatters

Aufgabe 3 – Arithmetik: Sequentielle Multiplikation

Wir wollen nun den Hardwareaufwand verringern. Mit ein bisschen Überlegung gelangt man zu folgendem Schaltbild:



Aufgabe 3 – Arithmetik

a) Multiplizieren Sie die beiden Binärzahlen $A = 0100110$ und $B = 0101$ durch Anwendung der Methode, die bei der Implementierung eines sequentiellen Multiplizierers zum Einsatz kommt. Geben Sie die einzelnen Schritte und das Ergebnis explizit an.

b) Dividieren Sie die Binärzahl $A = 0111101$ durch die Binärzahl $B = 0110$ mit dem Non-Restoring-Divisionsverfahren. Geben Sie die einzelnen Schritte sowie den Quotienten Q und Rest R explizit an.