1. Ein Wort aus dem Zeichenvorrat $\{0,1\}$ ist definiert als $(0^m1^n)^m(0^n1^m)^n$. Ermitteln Sie das Wort für alle $0 \leq m \leq 2$ und $0 \leq n \leq 2$.

2. Finden Sie (analog zum vorigen Beispiel) möglichst kurze Definitionen folgender Wörter:

   01001001010101010
   00100101010111111
   01011010010111010
   001001000100101

3. Ermitteln Sie folgende Wörtermengen und jeweils die Anzahl der Wörter.

   $\{a \circ b \circ c \mid a = 1^m \wedge b = 0^n \wedge c = 1^{5-m-n}\}$
   $\{a \in \{0,1\}^5 \mid a \in \{0,1\}^m \circ 010 \circ \{0,1\}^n \wedge m,n \geq 0\}$
   $\{01,10\}^3 \cup \{010,101\}^2$
   $\{0^m1^n \mid n = \lfloor m/3 \rfloor \wedge 4 \leq m+n \leq 9\}$

4. Finden Sie intensionale Definitionen folgender Wörtermengen:

   $\{\square\square\square\square\square, \triangle\square\square\square\triangle, \square\triangle\square\triangle\square, \triangle\triangle\square\triangle\triangle, \square\square\triangle\square\square, \triangle\square\triangle\square\triangle, \square\triangle\triangle\triangle\square, \triangle\triangle\triangle\triangle\triangle\}$
   $\{00011, 00101, 01001, 10001, 00110, 01010, 10010, 01100, 10100, 11000\}$
   $\{aba, abc, aca, acb, bab, bac, bca, bcb, cab, cac, cba, cbc\}$

5. Eine Wörtermenge kann auch rekursiv definiert werden. Der Zeichenvorrat sei $\{0,1,\_\}$. S sei die kleinste Wörtermenge, die folgende Regeln erfüllt:

   (a) $\_1 \in S$
   (b) $1a0\_1a1 \in S$ für alle $a \in \{0,1\}^*$
   (c) $a1\_b0 \in S$, wenn $a\_b \in S$

   Ermitteln Sie die kürzesten acht Wörter aus S.

6. Gegeben sind die Zeichenvorräte A, B. Berechnen Sie allgemein:

   (a) die Anzahl der möglichen Codewörter $|A^n|$ der Länge n, sowie $|B^m|$.

   (b) die Anzahl der möglichen Codewörter mit *maximaler* Länge n: $|A^0 \cup \ldots \cup A^n|$.

   (c) die Anzahl der möglichen Codewörter $|(A \cup B)^{m+n}|$ mit gemeinsamem Zeichenvorrat.

   (d) die Anzahl der konkatenierten Codewörter $|A^n \circ B^m|$.

   Demonstrieren Sie die Ergebnisse am Beispiel $A = \{a, b, c\}$, $n = 2$, $B = \{1, 2\}$, $m = 3$.

7. Die Levenshteindistanz ist eine Verallgemeinerung der Hammingdistanz. Während bei der Hammingdistanz die Anzahl der Schritte gezählt wird, mit denen man durch Austausch eines Zeichens das eine Wort in das andere verwandelt, sind bei der Levenshteindistanz auch das Einfügen und Entfernen eines Zeichens (an beliebiger Stelle) erlaubt. Zum Beispiel ist wegen $0101 \to 101 \to 1010$ die Levenshteindistanz zwischen 0101 und 1010 gleich 2, während die Hammingdistanz gleich 4 ist. Bestimmen Sie die Hamming- und Levenshteindistanz der Wörter

   $$01010101 \qquad \text{und} \qquad 10110010\,.$$

8. Gegeben ist die Abbildung $c : \{0, \ldots, n\} \longrightarrow \{0,1,2,3\}^3$ mit

   $$c(k) = a_4 \circ a_3 \circ a_2\,, \quad \text{wobei} \quad a_p = k \mod p\,.$$

   Also z.B. $c(7) = 311$. Geben sie alle Codewörter $c(0), \ldots, c(10)$ an. Ist diese Abbildung für $n = 10$ ein decodierbarer Code? Und wie sieht es für $n = 20$ aus? Begründen Sie Ihre Antwort.

9. Geben Sie alle Codewörter des Codes $c : \{0, \ldots, 7\} \longrightarrow \{0,1\}^*$ an:

   $c(k) := c_3(k)$, wobei

   $c_m(k) := c_{m-1}(\lfloor \frac{k}{2} \rfloor) \circ u(k)$,

   $u(k) = 0$ wenn k gerade, $u(k) = 1$ wenn k ungerade,

   $c_0(k) = \lambda$ (das leere Wort).

10. Die *Decodierung* eines Codes $c : \mathbb{N} \longrightarrow \{0,1\}^*$ ist gegeben durch

    $$c^{-1}(a_n \ldots a_1 a_0) = \sum_{k=0}^{n} a_k b_k\,,$$

    wobei $b_0 = 1$ und $b_k = \lceil \frac{3}{2} b_{k-1} \rceil$. $\lceil\,\rceil$ steht für Aufrunden. So ist z.B. $c^{-1}(101) = 3 + 1 = 4$. Decodiere die Codewörter 10101010 und 101010101. Finde mindestens fünf mögliche Codewörter für $c(47)$.

11. (a) Erzeugen Sie einen *nicht-zyklischen* Graycode für 10 Symbole $\{A, B, C, D, E, F, G, H, I, J\}$ und geben Sie die verwendete Transitionssequenz an.
    (b) Finden Sie einen *zyklischen* Graycode für obige Symbole und geben Sie auch hier die Transitionssequenz an.
    (c) Lässt sich ein zyklischer Graycode für 9 Symbole finden? Begründen Sie Ihre Antwort.

12. Ein Code besteht aus drei Zeichen aus dem bekannten Alphabet $a, \ldots, z$, wobei korrekte Codewörter jene sind, in denen niemals zwei Mitlaute (Alphabet ohne a, e, i, o, u) zweimal direkt hintereinander stehen. Wieviele mögliche und wieviele gültige Codewörter gibt es?

13. Ein 2-aus-4-Code und ein 4-Bit-Code bestehend aus 3 Datenbits plus einem Parity-Bit werden zu einem Gesamtcode konkateniert. Berechne für den Gesamtcode:
    (a) die Anzahl der möglichen, der gültigen und der ungültigen Codewörter.
    (b) die Wahrscheinlichkeit, dass ein zufälliges Codewort ungültig ist unter der Annahme, dass alle Codewörter gleich wahrscheinlich sind.

14. Wir betrachten binäre Codewörter der Länge n, die mit einem *parity bit* zur Fehlererkennung versehen werden. Ein Codewort hat also dann die Länge $n+1$. Ein realistischer Fehlerfall sind *Burstfehler*, wobei ein *Burst* eine Anzahl b von aufeinanderfolgenden Bits kippt.
    (a) Für welche Burstlängen b genügt das parity bit zur Fehlererkennung?
    (b) Wieviele verschiedene Fehlerfälle gibt es für eine Burstlänge b auf einem Codewort?
    (c) Ein Burst habe die Wahrscheinlichkeit $p = \frac{1}{r^b}$, wobei $r > 1$ ist. Wie groß ist die Wahrscheinlichkeit, dass das Codewort von **einem** Burst verändert wird?

15. Wir codieren das Wort *REEDEREI* mit einem Binärcode für jeden Buchstaben.

    (a) Überlegen Sie einen Code mit minimaler fester Wortlänge. Wieviele Bits hat das codierte Wort?

    (b) Definieren Sie nun einen Code mit variabler Wortlänge, der zu einem kürzeren Codewort (im Vergleich zu (a)) führt. Was ist die mittlere Wortlänge Ihres Codes?

16. Gegeben ist der Zeichenvorrat $\{A, B, C, D, E, F, G, H\}$ mit der Codierung c:

Zeichen	Code	Zeichen	Code
A	011100	E	001010
B	111111	F	110101
C	110011	G	100000
D	110110	H	000111

    (a) Ermitteln Sie die minimale und maximale Hammingdistanz des Codes.

    (b) Zeichnen Sie den zugehörigen Codebaum.

    (c) Kürzen Sie die Codewörter von hinten her so weit, dass gerade noch die Fanobedingung erfüllt ist.

    (d) Zeichnen Sie den neuen Codebaum.

    (e) Decodieren Sie mit dem neuen Code die Nachricht 11101110110011100000000111010

17. Die Symbolsequenz $AAAAAABBBCCCCCCDDDDD$ mit der Länge 20 besitzt relative Häufigkeiten der Symbole, die als Symbolwahrscheinlichkeiten $P(A) = 0.3$, $P(B) = 0.15$, $P(C) = 0.3$, $P(D) = 0.25$ interpretiert werden können

    (a) Konstruieren Sie den Huffman-Code

    (b) Konstruieren Sie einen möglichst ungünstigen Code, der aber soweit gekürzt ist, dass gerade noch die Fano-Bedingung erfüllt ist, indem Sie im Huffman-Algorithmus immer die größten Wahrscheinlichkeiten statt der kleinsten zusammenfassen.

    Codieren Sie damit jeweils die Symbolsequenz und geben Sie die Länge des Bitstreams an.

18. Gegeben ist der Zeichenvorrat {P, Q, R, S, T, U, V} mit folgenden absoluten Häufigkeiten.

Zeichen	Häufigkeit	Zeichen	Häufigkeit
P	130	T	250
Q	170	U	80
R	20	V	190
S	160		

    (a) Konstruieren Sie den Huffman-Code.

    (b) Ermitteln Sie die mittlere Codelänge für den Huffman-Code sowie den kürzesten Code fixer Länge.

19. *Ein Bild sagt mehr als tausend Worte.* Überprüfen Sie diese Aussage unter der Verwendung des Informationsgehaltes. Betrachten Sie ein Grauwertbild mit $m \times n$ Bildpunkten und k Grauwerten pro Pixel und ein Vokabular von l Wörtern, wobei alle Graustufen bzw. Wörter mit jeweils gleicher Wahrscheinlichkeit auftreten. Geben Sie allgemein den Informationsgehalt des Bildes und einer Beschreibung aus w Wörtern an. Wie groß muss ein Bild mit 64 Graustufen sein, damit obige Aussage bei einem Vokabular von $l = 1024$ stimmt? Wieviele Graustufen muss ein Bild der Größe 80×60 aufweisen (bei gleichem Vokabular), um die Aussage richtig zu machen?

20. Gegeben ist der Zeichenvorrat A, B, C, D, E, folgende absolute Häufigkeiten und folgender Code.

Zeichen	Häufigkeit	Code
A	9	000
B	12	001
C	16	010
D	2	011
E	9	1

    Betrachten Sie die relativen Symbolhäufigkeiten als Symbolwahrscheinlichkeiten und berechnen Sie den mittleren Informationsgehalt der Quelle sowie die mittlere Codelänge, die Redundanz, die relative Redundanz und die Codeeffizienz **ohne Taschenrechner**. Die Ergebnisse sind maximal zweistellige Brüche.

21. Die Zeichen A und B haben Wahrscheinlichkeiten 0.1 und 0.9. Ermitteln Sie den Huffman-Code (trivial) und berechnen Sie mittlere Codelänge, Redundanz und relative Redundanz. Fassen Sie jeweils 2 bzw. 3 Symbole zusammen zu $\{AA, AB, BA, BB\}$, sowie für $\{AAA, AAB, \ldots, BBB\}$. Berechnen Sie jeweils die mittlere Codelänge L_2, L_3 des erweiterten Codes, die mittlere Codelänge pro Zeichen $L_1 = L_2/2$ bzw. $L_1 = L_3/3$, und daraus wieder die Redundanzen.

22. Der Manchester-Code kann interpretiert werden als Recodierung eines Bits in einen 1-aus-2-Code. Das entspricht einer relativen Redundanz von 50%. Die Gleichanteilsfreiheit bedeutet, dass in den neuen Codewörtern gleich viele 0 wie 1 vorkommen. Eine Verallgemeinerung wäre daher, n Bits in einen $m/2$-aus-m-Code zu recodieren. Wie groß muss n mindestens sein, um damit eine kleinere relative Redundanz zu erzielen? Geben Sie so einen Code an.

23. Die Zahl 5432 ist im Dezimalsystem gegeben. Wandeln Sie sie durch fortgesetztes Dividieren in eine Zahl zur Basis (a) 3, (b) 8, (c) 14 um.

24. Stellen Sie die folgenden Dezimalzahlen jeweils als Binär-, Oktal- und Hexadezimalzahlen dar:

$$7, 23, 64, 139\,.$$

25. Geben Sie eine Formel an, um für jede beliebige natürliche Zahl n die genaue Anzahl der Ziffern ihrer Dezimaldarstellung zu berechnen, sowie Formeln zur Berechnung der genauen Anzahl der Ziffern ihrer Binärdarstellung und ihrer Hexadezimaldarstellung.

    Verwenden Sie diese Formeln, um für sehr große natürliche Zahlen n abzuschätzen, um wie viel Prozent ihre Binär- bzw. Hexadezimaldarstellung näherungsweise länger bzw. kürzer als ihre Dezimaldarstellung ist (die Anzahl der Ziffern betreffend).

26. Wandeln Sie die Zahl -57 durch sukzessive Division in eine Zahl zur Basis -2 um. Überprüfen Sie das Ergebnis. *Hinweis:* Als Ziffern (also auch Divisionsrest) stehen 0 und 1 zur Verfügung, die Ziffer b_i hat das Gewicht $(-2)^i$.

27. Führen Sie die folgenden Berechnungen mit Binärzahlen der Länge 8 jeweils im 1-er Komplement und im 2-er Komplement durch. Ermitteln Sie Überträge und Überläufe aus der Binärdarstellung. Stellen Sie das Ergebnis wieder im Dezimalsystem dar.

$$\begin{aligned} 53 &- 105 \\ 82 &+ 97 \\ -23 &- 105 \end{aligned}$$

28. Wandeln Sie folgende Zahlen durch fortgesetztes Multiplizieren in die Binärdarstellung um:

    $a = 0.3125 \quad b = 0.1 \quad c = \frac{1}{3} \quad d = \frac{5}{11}$

29. Addieren Sie die positiven Zahlen $a = 1110_2$, $b = 1101_2$, $c = 0110_2$ binär folgendermaßen: Addieren Sie zuerst für jede Stelle $a_i + b_i + c_i = (e_i d_i)_2$. Addieren Sie dann die zwei Zahlen $d + 2e = (d_3 d_2 d_1 d_0)_2 + (e_3 e_2 e_1 e_0 0)_2$. Überprüfen Sie das Ergebnis durch dezimale Addition. Überlegen Sie, was diese Vorgangsweise für Vorteile haben könnte.

30. *Nachträglich zur Sustainability-Week:* Bei arithmetischen Operationen in Rechenwerken wird hauptsächlich bei Bit-Flips (von 0 zu 1 oder von 1 zu 0) Energie verbraucht. Die Berechnung von Summen- und Carry-Bits ($(c_{k+1} s_k)_2 = a_k + b_k + c_k$) passiert dabei nicht sequentiell, sondern in Zeitschritten t (Reaktionszeit der Schaltelemente), in denen die Bits ihren Wert mehrmals wechseln können: $(c_{k+1}(t) s_k(t))_2 = a_k + b_k + c_k(t-1)$. Bei $t = 0$ seien alle Bits 0: $s_7(0) \ldots s_0(0) = 00000000$, $c_8(0) \ldots c_1(0) = 00000000$. Addieren Sie auf diese Weise die zwei Zahlen $a = 00110110$ und $b = 11001011$ schrittweise für $t = 0, 1, 2, \ldots$, bis sich nichts mehr ändert. Wie viele Bit-Flips ergeben sich?

    Um das Klima zu entlasten, berechnen wir nun in einem ersten Taktzyklus nur die Carry-Bits, d.h.: $c_{k+1}(t) = 1$ wenn in $a_k, b_k, c_k(t-1)$ mehr als eine 1 vorkommt, sonst 0. Im zweiten Taktzyklus werden dann alle $s_k(1)$ gleichzeitig aus $a_k, b_k, c_k(7)$ berechnet. Wie viele Bit-Flips gibt es jetzt?

31. Berechnen Sie im Binärsystem:

$$1100.0011 \times 1001.1001$$
$$10101010 : 1001$$

32. Wandeln Sie die Dezimalzahlen 1.01, 1.02, 1.03 und 1.04 in das Binärsystem mit 7 Nachkommastellen um. Nachfolgende Stellen werden abgeschnitten, d.h. abgerundet. Wandeln Sie das Ergebnis danach wieder zurück in das Dezimalsystem und runden Sie auf zwei Nachkommastellen. Bekommen Sie immer die ursprüngliche Zahl heraus?

33. Wir wollen den Kehrwert von $0.7 \approx 0.10110011_2$ (auf acht binäre Nachkommastellen genau) berechnen. Und zwar so: Am Anfang steht der Bruch $1.Z/0.N = 1.00000000/0.10110011$. Wir multiplizieren jetzt Zähler ($1.Z$) und Nenner ($0.N$) mit $2 - 0.N$. Dadurch wird der Nenner näher an 1 gerückt, der Quotient bleibt aber gleich. $2 - 0.N$ kann ermittelt werden als $1.W$, wobei W = (-N) im 2-er Komplement. (Warum?) Wir beschränken uns dabei immer auf 8 Nachkommastellen. Das wird jetzt wiederholt, bis der Nenner 1 ist (eigtl. 0.11111111), dann muss nämlich der Zähler $1.Z \approx 1/0.7$ sein.

34. Stellen Sie folgende 32-Bit IEEE Floating Point Zahlen als Dezimalzahlen dar:

```
0 01000000 01010101010101010101010
1 11111111 00110111110000100010000
1 00000000 00000000000000000000000
0 00001101 00110011001000000000000
1 11111111 00000000000000000000000
0 00110110 11010011000000000000000
```

35. Entwerfen Sie ein Format für Gleitkommazahlen ähnlich dem IEEE Format, das die Zahlen 65408 und $-1.52587890625 \cdot 10^{-5}$ darstellen kann. Das Format soll so wenig Bits wie möglich besitzen. Ignorieren Sie Ausnahmen, d.h. Null muss nicht exakt darstellbar sein.

36. Zwei Zahlen x und y in der 32-Bit-IEEE-Darstellung sollen multipliziert werden.

    (a) Stellen Sie die Zahlen $x = -1.25$ und $y = 3.125$ im obigen Format dar.

    (b) Geben Sie allgemeine Gleichungen für das Vorzeichenbit $v_z = f(v_x, v_y)$, die Charakteristik $C_z = f(C_x, C_y, M_x, M_y)$, und die Mantisse $M_z = f(M_x, M_y)$ des Produkts z an (**Anm.:** Das Produkt z sei als darstellbar vorausgesetzt, Spezialfälle von x und y seien ausgeschlossen).

    (c) Überprüfen Sie Ihr Ergebnis mit Hilfe der Zahlen aus (a).

37. Geben Sie alle möglichen Belegungen einer Wahrheitstabelle für zwei Aussagenvariablen a und b an. Wie viele Zeilen hat die Wahrheitstabelle, und wie viele verschiedene Belegungen gibt es? Welche Werte würden sich für n Aussagenvariablen ergeben?

    Bestimmen Sie für jede dieser Belegungen eine passende Aussageform mit minimaler Anzahl von Junktoren, sowie die benötigte Anzahl von Junktoren. Die mehrfache Verwendung desselben Junktors zählt dabei auch mehrfach.

    Dazu sollen nur die drei Junktoren Negation, Konjunktion und Disjunktion verwendet werden. Alle Symbole aus der Menge $\{a, b, 0, 1, (,)\}$ sind ebenfalls erlaubt, gezählt werden nur die Junktoren.

    Für eine weitere Betrachtung dürfen zusätzlich auch die Junktoren Subjunktion, Bijunktion, exklusives Oder, NAND und NOR verwendet werden. Für welche Belegungen kann damit die Anzahl der Junktoren noch reduziert werden?

38. Zeigen Sie, dass die Verknüpfung von n Bits $x_1, x_2, ..., x_n$ mit XOR, dem exklusiven Oder, für $n \geq 2$ immer genau das Ergebnis für die gerade Parität ergibt. Also:

    $x_{n+1} = x_1 \oplus x_2 \oplus ... \oplus x_n$ ist genau die *even parity* dieser n bits.

39. Zeigen Sie die Assoziativität der Bijunktion, also $(a \leftrightarrow b) \leftrightarrow c \Leftrightarrow a \leftrightarrow (b \leftrightarrow c)$, einmal mittels Wahrheitstabelle, und einmal mittels Umformungen.

40. Zeigen Sie mittels Umformungen, ob die folgenden Aussageformen Tautologien oder Kontradiktionen sind:

    $ab(\bar{a} \vee \bar{b})$

    $a \vee b \vee \bar{a}\bar{b}$

    $(\bar{b} \rightarrow ac) \rightarrow ((\bar{a} \vee b \vee \bar{c}) \rightarrow b)$

    $(a \rightarrow bc) \leftrightarrow (a\bar{b} \vee a\bar{c})$

41. Drei Kinder Justus, Peter und Bob stellen sich in einer Reihe auf. Drei Aussagen a, b, c sind definiert als: a: „Justus steht neben Peter“, b: „Peter steht neben Bob“ und c: „Justus steht neben Bob“. Für welche Wahrheitswerte von a, b, c lässt sich eine Reihenfolge finden, die die drei Aussagen erfüllt? Erstellen Sie die Wahrheitstabelle für diese Erfüllbarkeit. Zeigen Sie, dass die Aussageform

    $$(a \vee bc)\overline{a(b \leftrightarrow c)}$$

    die selbe Wahrheitsfunktion darstellt.

42. Das Axiomenschema nach Huntington ist:

    (1) $a \vee b \Leftrightarrow b \vee a$
    (2) $a \wedge b \Leftrightarrow b \wedge a$
    (3) $a \wedge (b \vee c) \Leftrightarrow (a \wedge b) \vee (a \wedge c)$
    (4) $a \vee (b \wedge c) \Leftrightarrow (a \vee b) \wedge (a \vee c)$
    (5) $a \vee 0 \Leftrightarrow a$
    (6) $a \wedge 1 \Leftrightarrow a$
    (7) $a \vee \bar{a} \Leftrightarrow 1$
    (8) $a \wedge \bar{a} \Leftrightarrow 0$

    Leiten Sie für jede mögliche einfache Verknüpfung der Elemente von $B = \{0, 1\}$ durch $\wedge$, $\vee$, $\bar{}$ das Ergebnis her (also die Wahrheitstabellen von $\wedge$, $\vee$, $\bar{}$). Wenden Sie in jedem Schritt nur ein Axiom an, und geben Sie dessen Nummer an. So ist z.B. $\bar{0} \overset{5}{\Leftrightarrow} \bar{0} \vee 0 \overset{1}{\Leftrightarrow} 0 \vee \bar{0} \overset{7}{\Leftrightarrow} 1$. *Tipp:* Für $0 \wedge 0$ betrachten Sie $0 \wedge (0 \vee 1)$, für $1 \vee 1$ betrachten Sie $1 \vee (1 \wedge 0)$.

43. Gegeben sei eine Menge A von n Aussagevariablen.

    a. Wieviele verschiedene Konjunktionsterme können daraus gebildet werden?

    b. Wieviele obiger Konjunktionsterme sind Minterme?

    c. Wieviele verschiedene vollständige DNFen können mit den Variablen aus A gebildet werden?

44. Weisen Sie für die Verknüpfungen *AND*, *NOR* und *XOR* durch Umformen nach, dass die jeweils abgeleitete KNF äquivalent zur aus der Wahrheitstabelle abgeleiteten DNF ist.

45. Überprüfen Sie, ob die Terme $a\bar{c}$, $\bar{b}c$ und $\bar{a}\bar{b}\bar{c}$ Implikanten oder sogar Primimplikanten von

$$\alpha := \bar{a}\bar{b} \vee a\bar{b}c \vee \bar{a}c$$

    sind. Verwenden Sie *nicht* das Verfahren von Quine-McCluskey sondern Umformungen und/oder Einsetzen von Wahrheitswerten.

46. Finden Sie alle Aussagenvariablenbelegungen der Aussageform

$$f(a,b,c,d) \equiv (a \vee \bar{b})(\bar{b} \vee c)(\bar{c} \vee \bar{d})(d \vee \bar{a}),$$

    die die Aussageform wahr (**1**) machen, und zwar auf folgende Weise: Ermitteln Sie zuerst $f(\mathbf{1},b,c,d)$ und $f(\mathbf{0},b,c,d)$, und kürzen Sie so weit wie möglich Variablen oder Terme. Ermitteln Sie daraus z.B. $f(\mathbf{1},\mathbf{1},c,d)$, $f(\mathbf{1},\mathbf{0},c,d)$, usw., bis $f(\mathbf{0},\mathbf{0},\mathbf{0},\mathbf{0})$. Es ergibt sich ein Suchbaum. Bei geschickter Wahl der Variablen (z.B. $f(\mathbf{1},b,c,\mathbf{0})$) kann der Suchbaum frühzeitig gekürzt werden.

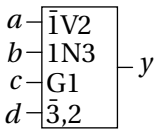
47. Vereinfachen Sie folgende Aussageformen mittels K-Diagramm, d.h. finden Sie die wesentlichen Primimplikanten (DMF).

    (a) $\overline{b\bar{c} \vee a\bar{b}\bar{d} \vee \bar{a}\bar{b}(d \to c)}$ (Beachten Sie die große Negation über der Aussageform!)

    (b) Wie (a), aber der Wahrheitswert sei egal, wenn $\bar{a}\bar{b}(d \to c)$ gilt.

48. Wie Beispiel 47, aber lösen Sie die Aufgabe jeweils mit dem Quine-McCluskey-Algorithmus.

49. (a) Welche Aussageform repräsentiert folgendes Schaltsymbol mit Abhängigkeitsnotation?



    (b) Definieren Sie ein Schaltsymbol zur Aussageform $y := (a \vee bd) \oplus (c\bar{d} \vee bd)$.

    Zeichnen Sie jeweils auch die Schaltung.

50. Konstruieren Sie eine Schaltung, die eine BCD-Ziffer $x = x_3x_2x_1x_0$ um eins erhöht (inkrementiert). Die Schaltung soll vier Ausgänge für die inkrementierte Ziffer $y = y_3y_2y_1y_0$ haben, sowie einen Ausgang c, der dann eins ist, wenn $x = 9$ ist, was $y = 10$ ergeben würde. In diesem Fall soll aber $y = 0$ ausgegeben werden. Falls x keine BCD-Ziffer darstellt ($x > 9$), ist es egal, was die Schaltung ausgibt.

    - Erstellen Sie die Wahrheitstabelle für alle Ausgänge und minimieren Sie die Schaltfunktionen mittels K-Diagramm.
    - Realisieren Sie die Schaltung mit OR-, AND-, und NOT-Gattern.

51. Konstruieren Sie eine Schaltung, die ermittelt, ob die 2-Bit-Zahl x kleiner oder gleich der 2-Bit-Zahl y ist. x und y seien dargestellt durch die Eingänge x_0, x_1, y_0, y_1.

    (a) Erstellen Sie die Wahrheitstabelle.

    (b) Minimieren Sie die Schaltfunktion mittels K-Diagramm.

    (c) Realisieren Sie die Schaltung mit OR-, AND-, und NOT-Gattern.

52. Wie Beispiel 51, konstruieren Sie aber zuerst eine Schaltung für die invertierte Funktion, also $x > y$, und ergänzen Sie sie am Ausgang mit einem NOT-Gatter.

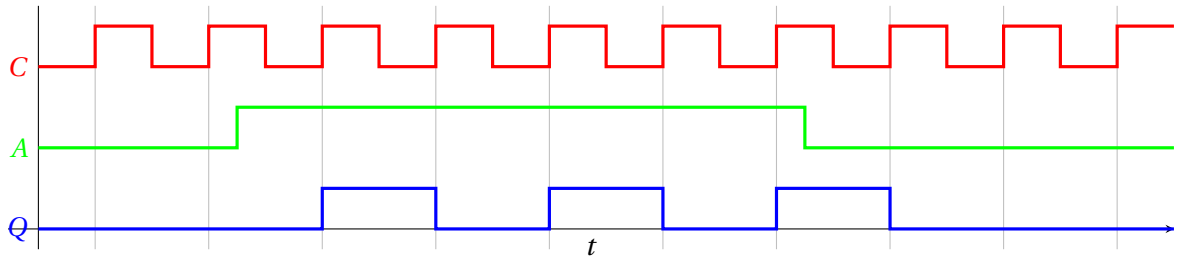
53. Ein (Crossbar-) Switch ist eine Schaltung, die n Eingänge mit n Ausgängen verbinden kann, wobei jeder Eingang mit genau einem Ausgang verbunden ist. Sie permutiert also die Eingänge.

    (a) Überlegen Sie sich die Schaltfunktionen für einen 2×2-Switch, der in Abhängigkeit einer Steuerleitung s die Eingänge x_0 und x_1 direkt oder vertauscht auf den Ausgängen y_0 und y_1 ausgibt. (Also $y_0 \Leftrightarrow x_0$, $y_1 \Leftrightarrow x_1$ wenn $\bar{s}$, und $y_0 \Leftrightarrow x_1$, $y_1 \Leftrightarrow x_0$ wenn s.)

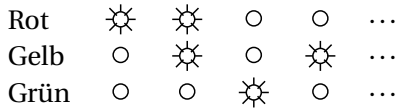
    (b) Versuchen Sie, einen 4×4-Switch aus 2×2-Switches zu konstruieren und dabei möglichst wenige 2×2-Switches zu verwenden. Ermitteln Sie die Stellungen (s) der Switches für die Permutation $0123 \mapsto 1302$.

54. Überlegen Sie sich eine Schaltung, die als Kernelement ein flankengesteuertes JK-Flipflop enthält: Eingänge sind ein Taktsignal C und sowie ein Steuereingang A. Der Ausgang Q soll folgendes Verhalten zeigen: Wenn A auf 0 gesetzt wird, soll auch Q konstant 0 werden. Wenn A auf 1 gesetzt wird, soll Q mit halber Frequenz des Taktsignals oszillieren, also immer zwischen 1 und 0 wechseln.

    Beispiel:

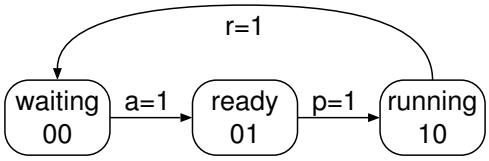


55. Implementieren Sie eine Ampelsteuerung (ohne Blinken) mit J-K-Flip-Flops. Die Ampel durchläuft zyklisch folgende Phasen:

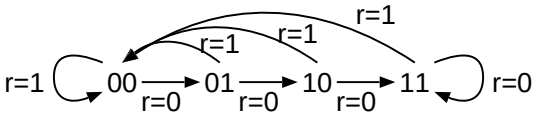


Der Zustand wird durch die drei Ausgänge Rot, Gelb und Grün repräsentiert, wobei 1 für hell und 0 für dunkel steht. Erstellen Sie die Transitionstabelle, ermitteln und vereinfachen Sie die logischen Ausdrücke für die Eingänge der FFs und geben Sie die gesamte Schaltung an.

56. Entwickeln Sie folgende Prozess-Zustands-Maschine mit JK-Flipflops. Der Zustand (waiting, ready, running) wird durch zwei Bits s_1, s_0 repräsentiert. Die Zustandsübergänge hängen von drei Eingängen ab: r . . . resource request, a . . . resource available, p . . . processor available. Wenn die Bedingung für einen Zustandsübergang nicht erfüllt ist, wird der Zustand beibehalten.



57. Entwerfen Sie einen gesättigten 2-Bit-Zähler mit Reseteingang r. Der Zähler hat folgende Zustandsübergänge:



Realisieren Sie dieses System mit JK-Flipflops. Minimieren Sie die Schaltfunktionen für die Eingänge der JK-Flipflops mit K-Diagrammen (wenn sinnvoll, aber mindestens für einen Eingang). Zeichnen Sie die Schaltung.

58. Entwickeln Sie einen 2×1-Bit Speicher mit JK-Flipflops. Die Schaltung hat einen Adresseingang a, der eines der zwei Bits auswählt, einen Daten-Input i, dessen Zustand in das durch a ausgewählte JK-Flipflop übernommen wird, wenn der Read-Write-Eingang w auf 1 ist. Der Ausgang o soll den Zustand des durch a ausgewählten Flipflops ausgeben.