Prof. Dr.-Ing. Jürgen Teich
Lehrstuhl für Informatik 12
(Hardware-Software-Co-Design)
Friedrich-Alexander-Universität Erlangen-Nürnberg

Klausur
Grundlagen der Technischen Informatik

19. Oktober 2020

Vorname	
Nachname	
Matrikelnummer	
Raum	Tentoria
Sitzplatz	

Aufgabe	1	2	3	4	5	Σ
Max. Punkte	16	16	16	16	16	80
Erreichte Punkte						
Note						

Organisatorische Hinweise

Bitte sorgfältig lesen und die Kenntnisnahme durch Unterschrift bestätigen

---

a) Bitte legen Sie Ihren Studentenausweis bereit.

b) Als Hilfsmittel sind nur Schreibmaterialien und ein beidseitig handbeschriebenes DIN A4-Blatt zugelassen.

c) Verwenden Sie nur dokumentenechte Stifte. Verwenden Sie weder Rot- noch Bleistifte und keine Korrekturroller (z. B. Tipp-Ex).

d) Sie können bei der Aufsicht zusätzliche Bearbeitungsblätter anfordern.

e) Unleserliches wird nicht bewertet.

f) Bei mehreren präsentierten Lösungen wird die Aufgabe nicht gewertet. Streichen Sie daher bei Angabe mehrerer Lösungsansätze die nicht zu bewertenden Lösungen durch.

g) Die Bearbeitungszeit beträgt 120 Minuten.

Erklärung

a) Im Falle einer während der Prüfung auftretenden Prüfungsunfähigkeit zeige ich dies sofort der Aufsicht an und befolge deren Anweisungen. Ich weiß, dass ich die volle Beweislast trage. Ich lasse mir das Formular des Prüfungsamts, das für diese Fälle vorgesehen ist, aushändigen und verfahre nach den dort niedergelegten Richtlinien.

b) Ich weiß, dass im Falle des Täuschungsversuchs oder der Benutzung unerlaubter Hilfsmittel („Unterschleif“) der Prüfungsausschuss die Entscheidung treffen kann, die betroffene Prüfungsleistung als mit „nicht ausreichend“ bewertet gelten zu lassen.

c) Die per E-Mail am 8. Oktober 2020 an mich gesandten speziellen COVID19-bedingten Hinweise zur Teilnahme an Klausuren in Präsenzform habe ich sorgfältig gelesen. Die darin festgelegten Hygiene- und sonstige Regeln befolge ich genau.

d) Ich habe die obigen Hinweise zur Kenntnis genommen.

Erlangen, den .....

Unterschrift

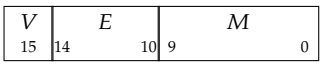
Aufgabe 1 (Zahlensysteme) (16 Punkte)

a) Wie heißen die Zahlensysteme zur Basis 8 und 16? (1 Punkt)

b) Wie lautet der Wertebereich einer vorzeichenlosen n-stelligen polyadischen Zahl zur Basis 16? (2 Punkte)

c) Konvertieren Sie die Zahl 103_{10} zur Basis 10 in eine vorzeichenlose Quintärzahl, d. h. in eine Zahl zur Basis 5. (3 Punkte)

d) In dieser Aufgabe soll mit 16-Bit Gleitkommazahlen gearbeitet werden. Diese werden analog zum IEEE-Format gebildet. Das Format der Gleitkommazahl sieht dabei wie folgend aus:
Vorzeichen (*1 Bit*), *Exponent* (*5 Bit*), *Mantisse* (*10 Bit*)



Führen Sie nun die Subtraktion $x - y$ der beiden in diesem Format dargestellten Gleitkommazahlen $x = 1\ 10101\ 0000001001$ und $y = 1\ 10001\ 1001010000$ aus, und geben Sie das Ergebnis im gleichen Format an. Zur Lösung sind stets alle Berechnungen und Lösungswege vollständig anzugeben. (5 Punkte)

e) In dieser Aufgabe wird ein 4-Bit Mikrocontroller betrachtet, welcher nur arithmetische Operationen für 4-Bit Zweierkomplementzahlen ausführen kann. Somit stehen Ihnen die arithmetischen Operationen op $\in \{+, -, =, \neq, >, \geq, <, \leq\}$, die logischen Verknüpfungen op $\in \{\wedge, \vee\}$, sowie die Negation $\neg x$ zur Verfügung. Die Operanden x und y sowie das Ergebnis $r = x \,\text{op}\, y$ sind 4-Bit Zweierkomplementzahlen. Ergebnisse für vier Beispiele der $>$-Operation sind im Folgenden gegeben.

Operation					Ergebnis
$S_2(1100_2) =$	-4_{10}	$>$	$S_2(1010_2) =$	-6_{10}	1
$S_2(1100_2) =$	-4_{10}	$>$	$S_2(0110_2) =$	6_{10}	0
$S_2(0111_2) =$	7_{10}	$>$	$S_2(0110_2) =$	6_{10}	1
$S_2(0111_2) =$	7_{10}	$>$	$S_2(1010_2) =$	-6_{10}	1

Da die $>$-Operation leider nur für 4-Bit Zweierkomplementzahlen ausgelegt ist, können vorzeichenlose 4-Bit Binärzahlen nicht korrekt verglichen werden. Im Folgenden sind die obigen vier Beispieloperationen nochmals aufgelistet, wenn deren Operanden nun vorzeichenlose 4-Bit Binärzahlen darstellen.

Operation					Ergebnis
$S(1100_2) =$	12_{10}	$>$	$S(1010_2) =$	10_{10}	1
$S(1100_2) =$	12_{10}	$>$	$S(0110_2) =$	6_{10}	0 korrekt wäre 1
$S(0111_2) =$	7_{10}	$>$	$S(0110_2) =$	6_{10}	1
$S(0111_2) =$	7_{10}	$>$	$S(1010_2) =$	10_{10}	1 korrekt wäre 0

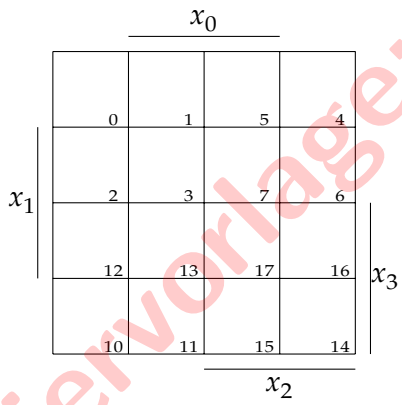
Geben Sie einen Test an, welcher zu **wahr** auswertet (1), wenn der Wert $S(x)$ der vorzeichenlosen 4-Bit Binärzahl x größer als der Wert $S(y)$ der vorzeichenlosen 4-Bit Binärzahl y ist. Dem Test stehen ausschließlich die 11 Operationen des 4-Bit Mikrocontrollers, Konstanten darstellbar als 4-Bit Zweierkomplementzahlen, die Operanden x und y, sowie beliebige Zwischenergebnisse zur Verfügung. (5 Punkte)

Aufgabe 2 (Minimierung von Schaltfunktionen) (16 Punkte)

Sie möchten eine Funktionseinheit für das IEEE 754 Half-Precision Gleitkommaformat mit Vorzeichenbit s, Exponentenbits e_i und Mantissenbits m_j entwickeln, die eine Spezialfunktion zum Abruf der Inversen $1/x$ mit $x \in \mathbb{N}, 0 \leq x \leq 15$ bereitstellen soll. Dazu nutzen Sie folgende Funktionstabelle:

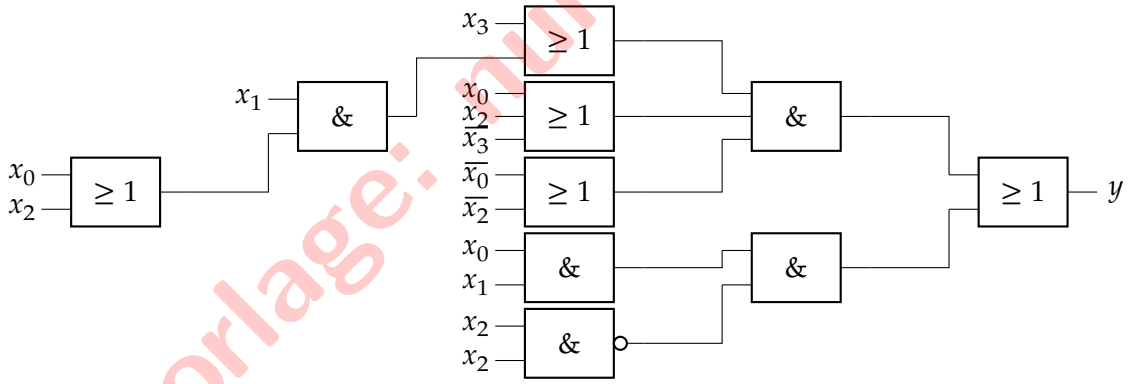
x	$1/x$		
$x_3x_2x_1x_0$	s	$e_4e_3e_2e_1e_0$	$m_1m_2m_3\,m_4m_5\,m_6m_7m_8m_9m_{10}$
0 0 0 0	0	1 1 1 1 1	0 0 0 0 0 0 0 0 0 0
0 0 0 1	0	0 1 1 1 1	0 0 0 0 0 0 0 0 0 0
0 0 1 0	0	0 1 1 1 0	0 0 0 0 0 0 0 0 0 0
0 0 1 1	0	0 1 1 0 1	0 1 0 1 0 1 0 1 0 1
0 1 0 0	0	0 1 1 0 1	0 0 0 0 0 0 0 0 0 0
0 1 0 1	0	0 1 1 0 0	1 0 0 1 1 0 0 1 1 0
0 1 1 0	0	0 1 1 0 0	0 1 0 1 0 1 0 1 0 1
0 1 1 1	0	0 1 1 0 0	0 0 1 1 0 0 0 0 0 0
1 0 0 0	0	0 1 1 0 0	0 0 0 0 0 0 0 0 0 0
1 0 0 1	0	0 1 0 1 1	1 1 0 0 0 1 1 1 0 0
1 0 1 0	0	0 1 0 1 1	1 0 0 1 1 0 0 1 1 0
1 0 1 1	0	0 1 0 1 1	0 1 1 1 0 1 0 0 0 1
1 1 0 0	0	0 1 0 1 1	0 1 0 1 0 1 0 1 0 1
1 1 0 1	0	0 1 0 1 1	0 0 1 1 1 0 1 1 0 0
1 1 1 0	0	0 1 0 1 1	0 0 1 0 0 1 0 0 1 0
1 1 1 1	0	0 1 0 1 1	0 0 0 1 0 0 0 1 0 0

a) Bestimmen Sie mittels des gegebenen Symmetriediagramms alle Prim- und Kernimplikate der Funktion $m_{10}(x_3, x_2, x_1, x_0)$, und geben Sie eine KMF für m_{10} an. (4 Punkte)



b) Bestimmen Sie für die Funktion $m_4(x_3, x_2, x_1, x_0)$ alle Primimplikanten mittels des Quine/McCluskey-Verfahrens, und geben Sie eine DNF für m_4 an. (4 Punkte)

c) Beweisen Sie: Folgendes Schaltnetz implementiert die Schaltfunktion $y = m_6(x_3, x_2, x_1, x_0)$. (4 Punkte)



d) Sei eine DMF für die Funktion $e_2(x_3, x_2, x_1, x_0)$ gegeben. Entwerfen Sie eine CMOS-Schaltung für e_2. Geben Sie das Pull-Down- und Pull-Up-Netzwerk an und vervollständigen Sie die gegebene Grafik. Dazu stehen Ihnen die Konstanten 0 und 1 sowie die Eingangssignale x_3, x_2, x_1, x_0 zur Verfügung, jedoch nicht deren Negation. (4 Punkte)

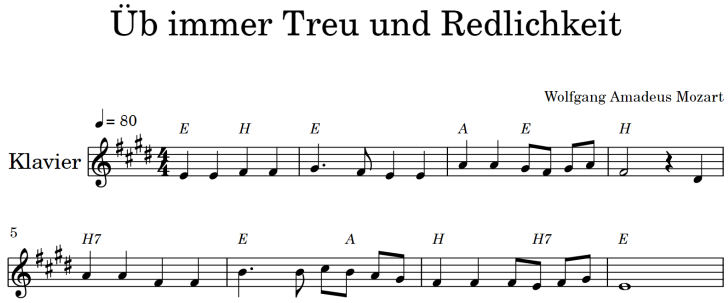
$DMF(e_2) = \overline{x_3} + x_3 \cdot \overline{x_2} \cdot \overline{x_1} \cdot \overline{x_0}$

Aufgabe 3 (Automaten) (16 Punkte)

a) Unter einer Kadenz (lat. *cadere* 'fallen, enden') versteht man in der Harmonielehre eine Akkordfolge, die häufig den Abschluss musikalischer Abschnitte indiziert, aber gleichwohl als zentraler Grundbaustein des harmonischen Gerüsts fungiert. Bei der harmonischen Analyse der Musik des 18. und 19. Jahrhunderts lassen sich verschiedenartige Kadenzen feststellen, die auf der Tonika (T), dem Akkord über der ersten Stufe der Skala (it. *scala* 'Leiter, Tonleiter'), der Subdominante (S), dem Akkord über der vierten Stufe, und der Dominante (D), dem Akkord über der fünften Stufe, beruhen. Die Stufe gibt für jede der Funktionen (T, D, S) den Grundton des korrespondierenden Akkordes an. Eine besondere Form des Akkordes ist der Septakkord, der mit einer 7 hinter dem Akkordnamen annotiert wird, was aber mit keiner Änderung der musikalischen Funktion einhergeht. Als authentische Kadenz bezeichnet man die Folge T – D – T, als plagale Kadenz benennt man die Sequenz T – S – T und als Vollkadenz wird schließlich die Folge T – S – D – T verstanden. Diese Kadenzarten liegen weiterhin auch dann vor, wenn einzelne Funktionen mehrere Takte hintereinander auftreten (so ist z.B. T – T – D – D – T eine valide authentische Kadenz). Im Folgenden soll ein Mealy-Automat erstellt werden, der als Eingabe die Funktion des Akkordes (T, S, D) des aktuellen Takts erhält und die identifizierte Kadenz (Authentisch, Plagal, Vollkadenz) ausgibt. Die erkannte Kadenz soll nur einmalig bei der Rückkehr in die Tonika ausgegeben werden. Ansonsten sei ‚?' auszugeben, um zu signalisieren, dass noch keine vollständige Kadenz erkannt wurde. Zu Beginn befindet sich der Automat im Zustand ‚Start'. Die abschließende Funktion einer Kadenz kann dabei gleichzeitig auch eine nachfolgende Kadenz eröffnen. Bei Akkordfolgen, die nicht den obigen Muster folgen (zum Beispiel ‚ST' oder ‚TDS'), soll in einen Fehlerzustand 'Error' übergegangen werden, der durch die Eingabe einer Tonika wieder verlassen werden kann. *Hinweis: eine minimale Umsetzung des Automaten kommt mit sechs Zuständen aus.*

1. Vervollständigen Sie den obig beschriebenen Zustandsautomaten. Sie dürfen die Zustände beliebig benennen. (4 Punkte)

2. Geben Sie die Ausgabe des Automaten für die Akkordsequenz des Stückes 'Üb immer Treu und Redlichkeit' von Wolgang Amadeus Mozart an. Die Akkorde sind hierbei über den Noten annotiert; der Automat erhalte als Eingabe die Funktion der Akkorde. Das Stück ist in E-Dur, welches die Skala E – Fis – Gis – A – H – Cis – Dis besitzt (E ist hiermit die erste Stufe, Fis die Zweite, usw.). Die Ausgabe '?' muss nicht angegeben werden. (2 Punkte)



3. Wie viele Zustände weist eine minimale Umsetzung des beschriebenen Mealy-Automaten als Moore-Automaten auf? (1 Punkt)

b) Im Folgenden ist die Automatentafel für einen weiteren Automaten gegeben, der Harmonien erkennt. Vervollständigen Sie die Automatentafel unter Verwendung von taktflankengesteuerten JK- bzw. T-Flipflops. (3 Punkte)

Zustandsname	Aktueller Zustand		Eingabe		Nachfolgezustand		Ansteuerfunktion		
	q_1	q_0	i_1	i_0	q_1'	q_0'	J_1	K_1	T_0
Tonika	0	0	0	0	0	0			
Tonika	0	0	0	1	0	1			
Tonika	0	0	1	0	1	0			
Tonika	0	0	1	1	-	-			
Subdominante	0	1	0	0	0	0			
Subdominante	0	1	0	1	0	1			
Subdominante	0	1	1	0	1	0			
Subdominante	0	1	1	1	-	-			
Dominante	1	0	0	0	0	0			
Dominante	1	0	0	1	0	1			
Dominante	1	0	1	0	1	0			
Dominante	1	0	1	1	-	-			
(Ungültig)	-	-	-	-	-	-			

c) Wie viele Flipflops sind minimal notwendig, um die Funktionalität des Automaten aus der Teilaufgabe 3b) umzusetzen? (1 Punkt)

d) Bei der Minimierung eines dritten Harmonie-Automaten, der aus einem Toggle-Flipflop und einem D-Flipflop besteht, ergeben sich die folgenden konjunktiven Minimalformen für T_1 und D_0:

$$KMF_{T_1} = (\overline{q_1} + q_0 + i_0)(\overline{q_0} + i_1), KMF_{D_0} = (i_0 + \overline{i_1})(q_0 + q_1)$$

Zeichnen Sie das Schaltwerk des Automaten, indem sie geeignet Logikgatter hinzufügen und verschalten. Sie dürfen die Flipflops wie aus der Vorlesung bekannt als Blackbox abstrahieren. (3 Punkte)

e) Zeichnen Sie das Schaltnetz eines Toggle-Latch. (2 Punkte)

Aufgabe 4 (Codierung und Rechnerarithmetik) (16 Punkte)

a) Geben Sie für die folgenden drei Codes jeweils an, ob es sich um einen Huffman-Code handelt. Falls ja, geben Sie den entsprechenden Codierungsbaum an. Falls nein, begründen Sie Ihre Antwort. (4 Punkte)

1. $C_1 = [00, 10, 01, 100, 101]$
2. $C_2 = [00, 01, 11, 100, 101]$
3. $C_3 = [00, 01, 111, 101, 100]$

b) Ein fairer Würfel, welcher die Würfelzahl w (mit $1 \leq w \leq 6$) zeigt, wird $n = 3$ Mal geworfen. Ordnen Sie die folgenden Nachrichten gemäß ihres Informationsgehalts von niedrig nach hoch: (1 Punkt)

A := „Der erste Wurf zeigt eine Zahl $w \leq 2$.“

B := „Die letzten zwei Würfe zeigen jeweils $w = 6$.“

C := „Es wird genau einmal $w = 1$ geworfen.“

$$I(\square) < I(\square) < I(\square)$$

c) Gegeben ist ein Code mit folgenden Codewörtern:

A	B	C	D	E
1111	1011	0110	1000	0000

1. Was ist die minimale Anzahl an zusätzlichen Bits, um welche der gegebene Code erweitert werden muss, so dass 1-Fehler erkennbar sind? (1 Punkt)

2. Wie viele Prüfstellen k sind minimal nötig, so dass 1-Fehler korrigiert werden können? Geben Sie für den gegebenen Code den entsprechenden Hamming-Code an. (3 Punkte)

d) Zur Fehlererkennung und Fehlerkorrektur in der Hardware werden oft Einheiten zur Überprüfung der Parität, welche als Paritätsgeneratoren bezeichnet werden, eingesetzt.

1. Zuerst soll ein Schaltnetz für einen Paritätsgenerator für 2-Bit Binärwörter ausschließlich mittels NOR-Gattern realisiert werden. Dieser soll bei gerader Parität einer Zahl (x_1, x_0) eine 1, sonst eine 0 zurückgeben.

(3 Punkte)

2. Entwerfen Sie nun, unter ausschließlicher Verwendung von Äquivalenzgattern mit zwei Eingängen einen Paritätsgenerator (gerade Parität) für 4-Bit Binärwörter (x_3, x_2, x_1, x_0).

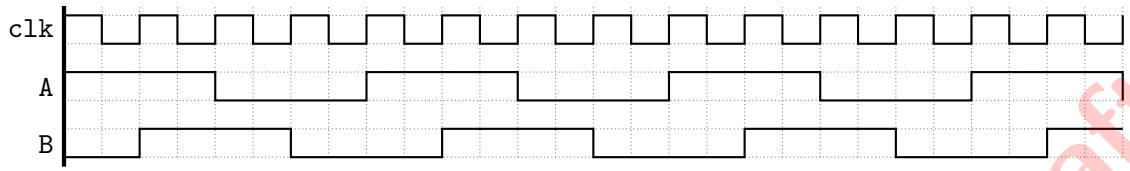
(1 Punkt)

e) Zeichnen Sie ein Schaltnetz eines 2-Bit-Addierers/Subtrahierers, der mit einem Bit sub gesteuert zwei Zweierkomplentzahlen $x = (x_1, x_0)$ und $y = (y_1, y_0)$ entweder addiert ($sub = 0$) oder y von x subtrahiert ($sub = 1$). Verwenden Sie dabei logische Gatter und gegebenenfalls Volladdierer. (2 Punkte)

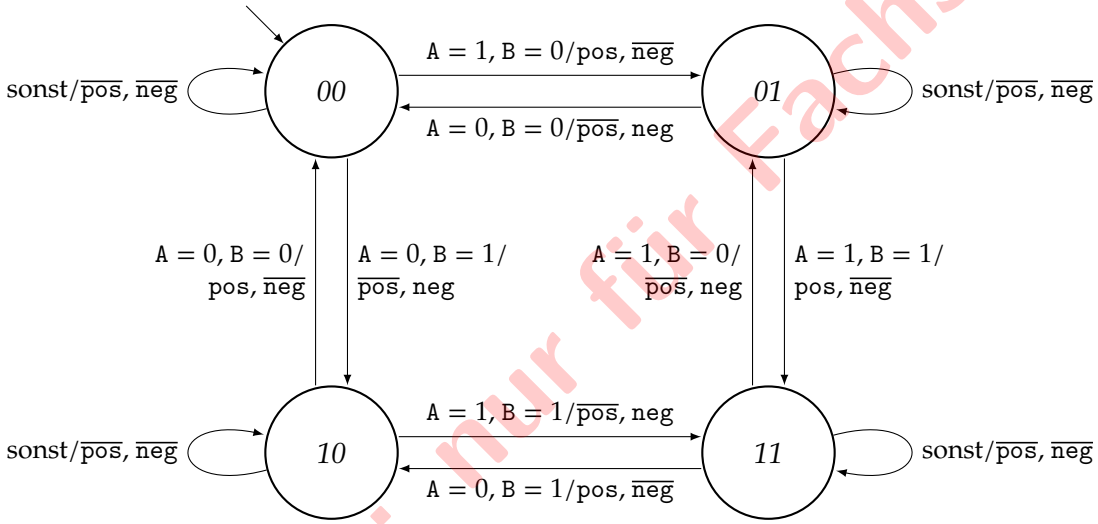
f) Erweitern Sie die Schaltung aus der vorherigen Aufgabe um einen Fehlererkennungsausgang f, der mit $f = 1$ signalisieren soll, dass das Ergebnis $s = (s_1, s_0)$ nicht korrekt ist, da ein Überlauf stattgefunden hat. (1 Punkt)

Aufgabe 5 (VHDL) (16 Punkte)

Inkrementalgeber erfassen Positions- und Winkeländerungen an beweglichen Bauelementen. Sie sind aufgebaut aus zwei Sensoren, die um 90 Grad phasenverschobene (in diesem Fall: um einen Takt verschobene) Signale `A` und `B` ausgeben. Folgendes Wellenformdiagramm illustriert diese Charakteristik beispielhaft:



Anhand dieser Verschiebung lässt sich die Richtung der Positions- oder Winkeländerung erkennen. Durch folgenden Graphen spezifizierter Automat realisiert diese Erkennung:



Der Automat hat die beiden Eingabesignale `A` und `B` sowie zwei Ausgabesignale: Das Signal `pos` kennzeichnet eine *positive* Änderung der Position oder des Winkels, das Signal `neg` eine *negative*.

Implementieren Sie diesen Automaten in VHDL als *synchrone* Schaltung, die sich *synchron* zurücksetzen lässt.

a) Vervollständigen Sie die folgende `entity`-Deklaration des Inkrementalgebers. (3 Punkte)

```
entity encoder is

end encoder;
```

b) Vervollständigen Sie den Rumpf der folgenden `architecture`. (9 Punkte)

```
architecture behavioral of encoder is
  type states is (s00, s01, s10, s11);
  signal state : states;
begin
```

```
end architecture;
```

c) Erklären Sie die Unterschiede zwischen Signalen und Variablen.

1. Geben Sie jeweils ein Beispiel für eine Zuweisung an ein Signal und an eine Variable. (1 Punkt)

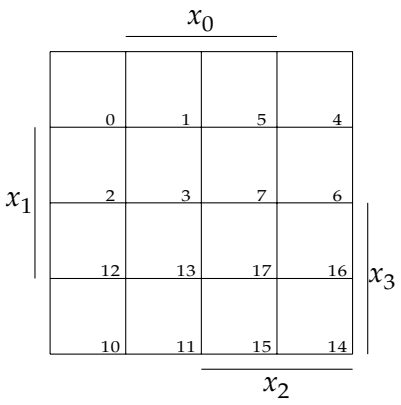
2. Wann übernimmt ein Signal den Wert einer Zuweisung innerhalb eines `process`? Wann eine Variable? (1 Punkt)

3. Wofür eignen sich Signale besser und wofür Variablen? (1 Punkt)

d) Wieso ist die Überprüfung der funktionalen Korrektheit beim Entwurf digitaler Schaltungen besonders wichtig? (1 Punkt)

Sie können das folgende Material zur Läsung der jeweiligen Aufgabe nutzen.
Streichen Sie alles, was nicht bewertet werden soll, eindeutig durch.

Aufgabe 2



GND

VCC

Aufgabe 3