Modul

Produktion + Steuerungstechnik Grundlagen

Zusammenfassung
Wintersemester 05/06

Basierend auf dem Skript von Prof. Jürg Habegger

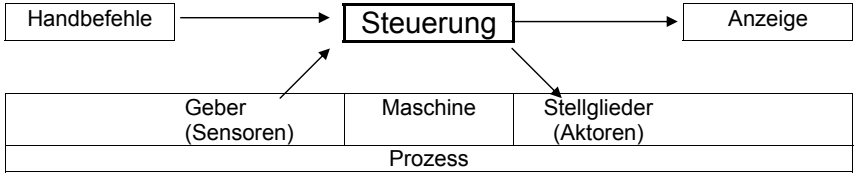
Inhaltsverzeichnis

Kapitel Nummerierung gemäss Skript. Fettgedruckte Zahlen am Anfang jedes Kapitel entspricht der Seitenzahl des Skriptes.

1. Einleitung

1.1 Einordnung

4



1.2.1 Steuern

5

Das Steuern, die Steuerung, ist der Vorgang in deinem System, bei dem eine oder mehrere Grössen als Eingangsgrössen andere Grössen als Ausgangsgrössen aufgrund der dem System eigentümlichen Gesetzmässigkeit beeinflussen.

- Offener Wirkungsweg
- Geschlossener Wirkungsweg

1.2.2 Regeln

7

Das Regeln ist ein Vorgang, bei dem die zu regeln de Grösse fortlaufend erfasst und so beeinflusst wird, dass sie sich der gewünschten Grösse angleicht.

- Geschlossener Wirkungskreis
- Regelgrösse wird mit Führungsgrösse verglichen und um Stellgrösse angepasst

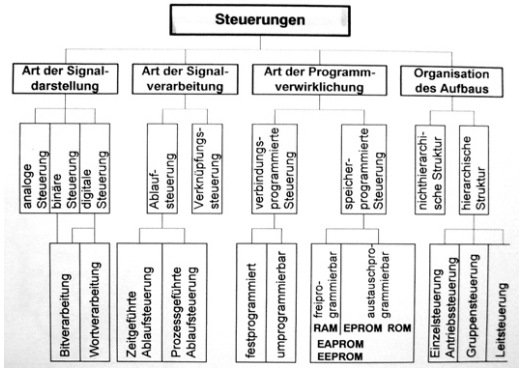
1.2.3 Übertragungsglieder

9

- Statisches Verhalten
- Dynamisches Verhalten

1.3 Steuerungstechnik

10



1.3.1 Signaldarstellung

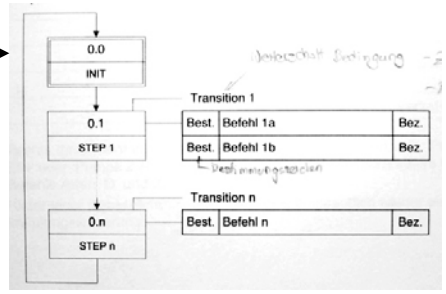
11

- Analog = entsprechend (Kurvenscheiben, Ventile)
- Binär = zwei Zustände (0&1)
- Digital = mehrere binäre Signale (Mikroprozessoren)

1.3.2 Signalverarbeitung

13

- Synchron = auf ein Taktsignal
- Asynchron = unmittelbar
- Verknüpfungssteuerung (and, or, not)
- Ablaufsteuerung (n-te Schritt wenn (n-1)te Schritt abgeschlossen /NS, S, R, D)
  - o Zeitgeführt
  - o Prozessgeführt



1.3.3 Programmverwirklichung

18

Programm bedeutet die die Gesamtheit aller Anweisungen und Vereinbarungen für die Signalverarbeitung, durch die eine zu steuernde Anlage beeinflusst wird.

- Verbindungsprogrammierte Steuerung (Verbindung durch fest verlegte Drähte oder Bahnen)
- Speicherprogrammierbare Steuerung (RAM, ROM usw.)

1.3.4 Struktur

20

- Nicht hierarchisch
- Hierarchisch
  - Einzelsteuerebene (viele kleine Daten, kurze Lebensdauer)
  - Gruppensteuerebene
  - Leitersteuerebene (Grosse Daten, lange Lebensdauer)

Unter Leiten versteht man die Gesamtheit aller Massnahmen, die bewirken, dass der gewünschte Prozessverlauf erreicht wird. Dabei ist meist auch eine Mitwirkung von Menschen vorgesehen.

- Messen, Zählen, Steuern, Regeln, Optimieren, Überwachen, Schützen, Auswerten, Anzeigen, Melden, Aufzeichnen, Protokollieren, Eingreifen, Stellen, Datenerfassen, Dateneingeben, Datenverarbeiten, Datenübertragen, Datenausgeben

2. Signale

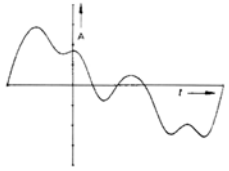
22

Unter einem Signal versteht man die physikalische Darstellung von Nachrichten oder Daten. Dargestellt durch einen Wertverlauf, die Amplitude ist von der Zeit t abhängig.

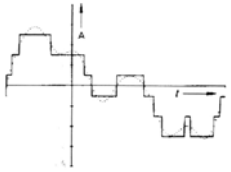
2.1 Definition analoge und digitale Signale

22

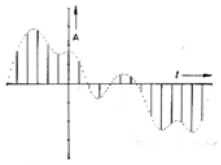
- Wert- und Zeitkontinuierlich = analoges Signal



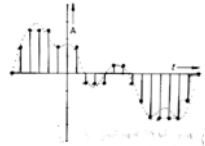
- Wertdiskret und Zeitkontinuierlich (LED-Anzeige der Stereoanlage)



- Wertkontinuierlich und Zeitdiskret



- Wert- und Zeitdiskret = digitales Signal



2.2 Diskrete, digitale und Binäre Signale

25

- Diskret (endliche Anzahl von Werten)
- Digital (ganzzahliges Vielfaches der Grundeinheit)
- Binär (einparametrisches digitales Signal)

2.x Aliasing

Abtastfrequenz f_a Signalfrequenz f_s Niquistfrequenz $f_n = \frac{f_a}{2}$

Aliasingfrequenz $f_a = |((f_s + f_N) \bmod f_a) - f_N|$

- Verhinderung von Aliasing
  - Abtasttheorem von Shannon einhalten
    theoretisch $f_a \geq 2 \cdot f_s$
    praktisch $f_a \geq 5 - 20 \cdot f_s$
  - Tiefpassfilter (Frequenz"glättung" / Grenzfrequenz $f_g \leq \frac{f_a}{2}$)

3. Datenübertragung und Datenspeicherung

27

- Parallel (8 Bits gleichzeitig / hohe Übertragungsgeschw., höhere Kosten)
- Seriell (8 Bit-Folgen nacheinander / langsamer, billiger)

3.1.3 Geschwindigkeit

28

- 1 Baud = 1 Bit pro Sekunde (USB2.0: 1.5 – 480 Mbit/s)

3.2 Programmspeicher

28

- Speicherung durch Lochen von Karten oder Streifen (Lochkarte, Lochstreifen)
- Speicherung durch stabilen magnetischen Fluss (Magnetkern, Magnetband, HDD, Floppy)
- Speicherung durch stabile Strom- oder Spannungsverteilung (Halbleiterspeicher)

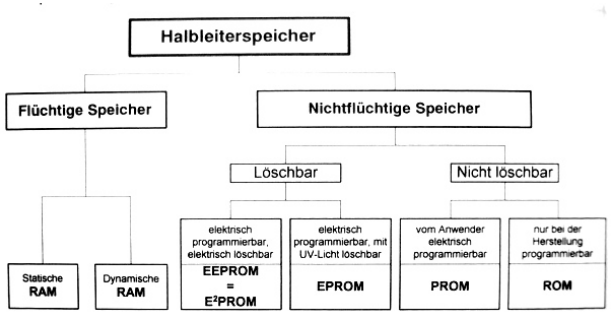
3.2.1 Kriterien für Daten- und Programmspeicher

29

- Speicherdichte, Speicherkapazität, Art des Zugriffs und Zugriffszeit, Adressierung, Energieverbrauch

3.2.2 Speichertypen

30



- Schreib-Lese-Speicher (Flüchtige Speicher)
  - RAM: Random Access Memory ; Speicher mit wahlfreiem Zugriff
  - Statische RAMs Speicherung durch Transistorzelle (grosse Abmessung, geringe Störanfälligkeit und Energieverbrauch)
  - Dynamische RAMs (geringe Abmessung, refresh-Zyklus)
- Nur-Lese-Speicher (Festwertspeicher)
  - ROM; Read Only Memory
  - Beliebig oft gelesen, nicht mehr verändert
  - Programmspeicher von festliegenden Abläufen
- Programmierbare Festwertspeicher
  - PROM; Programmable Read Only Memory
  - Einmalig beschreibbar, danach nicht mehr veränderbar
- Löschbare Festwertspeicher
  - EPROM; Erasable Programmable Read Only Memory
  - Durch UV-Licht löschbar (nur insgesamt) anschliessend neu beschrieben
- Elektrisch löschbare Festwertspeicher
  - EEPROM; Electrically Erasable Programmable Read Only Memory
  - Durch elektrischen Impuls insgesamt gelöscht

- Andere
  - EAROM; Electrically Alterable ROM
    - Inhalt Wort für Wort löschbar
  - NOVRAM; NO Volatile RAM

4. Codes

34

Ein Code bildet die Zeichen eines Zeichenvorrates auf die Zeichen eines zweiten Zeichenvorrates ab.
Nur Zeichen 0 oder 1. Kombination mehrerer Zeichen nennt man Wort
Mit der Wortlänge n können $N = 2^n$ verschiedene Kombinationen von 0 und 1 gebildet werden.

4.1 Zahlensysteme

- Dezimalsystem
  - 10 Ziffern (0-9)
  - Potenz zur Basis 10
  - Bsp.: $496 = 4*10^2 + 9*10^1 + 6*10^0$
- Dualsystem
  - Nur Ziffern 0 und 1
  - Potenz zur Basis 2
  - Bsp.: $1001 = 1*2^3 + 0*2^2 + 0*2^1 + 1*2^0$
- Oktalsystem
  - 8 Ziffern (0-7)
  - Potenz zur Basis 8
  - Bsp.: $352 = 3*8^2 + 5*8^1 + 2*8^0$
- Hexadezimalsystem
  - 16 Ziffern (0-9 & A-F)
  - Potenz zur Basis 16
  - Bsp.: $B3 = B*16^1 + 3*16^0$

4.2 Minimalcodes

38

Werden alle N Codewörter in einem Code benützt, so spricht man von einem Minimalcode, sonst von einem redundanten Code.

- Dualcode
- Gray-Code (ähnlich wie Dual, aber einschrittig, d.h. nur eine Ziffer ändert zum nächsten Wort. Eingesetzt in Abtastsystemen. Ist nicht bewertbar.)

4.3 Tetradische Codes (Codes aus vier Einheiten/Bits)

39

- BCD-Code; Binary Coded Decimals (8-4-2-1-Code)
  - Jede Dezimalzahl wird durch ihr duales Äquivalent dargestellt
  - 10-15 werden nicht genutzt: Pseudo-Tetraden
  - Redundant, kein Minimalcode, bewertbar
- Selbstkomplementäre Codes (Aiken, Exzess-3)
- Glixon-Code
  - Ähnlich Gray-Code (Ziffer 9 anders Codiert damit 9 auf 1 einschrittig)
  - Kein Minimalcode, nicht bewertbar
- Code mit Prüfbit (8-4-2-1-0-Code)
  - Parity-Checkbit zur Erkennbarkeit von Fehlern
  - Redundant, bewertbar

4.4 Fehlererkennende Codes (m-aus-n-Codes)

42

Es Werden Codewörter benützt, bei denen von n Stellen die Zahl der Einsen gleich m ist.

- 1-aus-10 Code
- 2-aus-7 Code
- 2-aus-5 Code
- 3-aus-5 Code: Lorenzcode
- EAN-Code

Für die Redundanz eines Codes gelten folgende Definitionen:

- Absolute Redundanz $R = \log_2 \frac{Anzahl - aller - möglichen - Bitkombinationen}{Anzahl - ausgenützter - Bitkombinationen}$
- Relative Redundanz $r = \frac{R}{n} \cdot 100\%$

4.5 Alfanumerische Codes

46

Zum Codieren von alphanumerischen Zeichen, d.h. Buchstaben, Ziffern, Satz- und Sonderzeichen.

- CCITT Nr.5 oder ASCII-Code (ISO 7-Bit Code)
  - 0-31 Steuerzeichen
  - 32-127 alphanumerisch

5. Digitale Schaltungstechnik

48

5.1 Logische Grundfunktionen

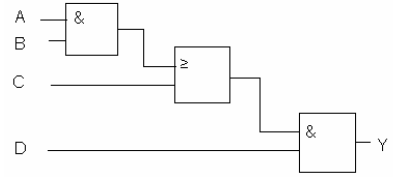
- Darstellung einer binären Funktion durch Algebraischen Ausdruck, Wahrheitstabelle, Signalschaltbild, Venn-Diagramm, Schaltermodell
- Grundoperationen UND (Konjunktion), ODER (Disjunktion), NICHT (Negation)
- Abgeleitete Formen NAND, NOR, ÄQUIVALENZ, ANTIVALENZ (Xor)

5.2 Schaltungsanalyse

54

Betrachtung der Kombinationen der Logikbausteine untereinander:

- Digitalschaltung



- Wahrheistabelle

Fall	D	C	B	A		Y
1	0	0	0	0		0
2	0	0	0	1		0
3	0	0	1	0		0

- Funktionsgleichung $[(A \wedge B) \vee C] \wedge D$

5.3 Schaltungsalgebra (Boolsche Algebra)

57

- Grundlegende Axiome
- Prioritäten der Grundoperationen
  1. Ausdrücke in Klammern
  2. Negation
  3. Konjunktion
  4. Disjunktion
- Rechenregeln mit Signalvariablen

$0 \wedge X = 0$ $\quad 0 \vee X = X$

$1 \wedge X = X$ $\quad 1 \vee X = 1$

$X \wedge X = X$ $\quad X \vee X = X$

$X \wedge \overline{X} = 0$ $\quad X \vee \overline{X} = 1$

5.3.4 Grundgesetzte

58

- Konjunktionen und Disjunktionen sind kommutativ: $A \vee B \vee C = C \vee A \vee B$
- Konjunktionen und Disjunktionen sind assoziativ: $(A \wedge B) \wedge C = A \wedge (B \wedge C)$
- Distributionsgesetzt kann angewendet werden $(A \wedge B) \vee (A \wedge C) = A \wedge (B \vee C)$
- De Morgan $\overline{A \wedge B \wedge C \wedge ..... \wedge Z} = \overline{A} \vee \overline{B} \vee \overline{C} \vee ..... \vee \overline{Z}$ $A \wedge B = \overline{\overline{A} \vee \overline{B}}$
- Shannon $F(A,B,C,....,Z,\wedge,\vee) = F(\overline{A},\overline{B},\overline{C},....,\overline{Z},\vee,\wedge)$

5.4 Schaltungssynthese

60

5.4.1 Aufbau von Verknüpfungsschaltungen

1. Beschreibung der Funktion der gesuchten Schaltung (Pflichtenheft)
2. Festlegung der Eingangs- und Ausgangsgrössen und der Bedeutung von 0 und 1
3. Erstellen der Wahrheitstabelle
4. Bestimmen der logischen Verknüpfungsschaltung
5. Vereinfachung und evtl. Umformung der Schaltung

5.4.2 Normalformen

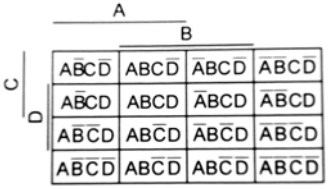
- ODER-Normalform (Disjunktive Normalform)
  Disjunktive Normalform wird jede disjunktive Verknüpfung von Vollkonjunktionen genannt. Vollkonjunktionen oder Minterme sind Konjunktionen, die sämtliche Eingangsvariablen, negiert oder nicht negiert, enthalten.
  $X = \overline{A} \wedge B \wedge C \vee A \wedge \overline{B} \wedge C \vee A \wedge B \wedge \overline{C} \vee A \wedge B \wedge C$
  Die ODER-Normalform gibt vollständig alle Schaltfunktionen wieder, die die gestellte Aufgabe lösen. Sie enthält die UND-Verknüpfungen aller vorkommenden Variablen für alle Fälle in denen das Ausgangssignal zu 1 wird.
- UND-Normalform (Konjunktive Normalform)
  Konjunktive Normalform wird jede konjunktive Verknüpfung von Volldisjunktionen genannt. Volldisjunktionen oder Maxterme sind Disjunktionen, die sämtliche Eingangsvariablen, negiert oder nicht negiert, enthalten.
  $\overline{X} = \overline{A} \wedge \overline{B} \wedge \overline{C} \vee \overline{A} \wedge \overline{B} \wedge C \vee \overline{A} \wedge B \wedge \overline{C} \vee A \wedge \overline{B} \wedge \overline{C}$

5.4.3.1 Vereinfachen von Schaltfunktionen mit Karnaugh

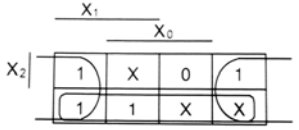
65

Im KV-Diagramm werden für alle Vollkonjunktionen, bei denen das Ausgangssignal 1 liefert eine 1 in das entsprechende Feld eingetragen.

- KV-Diagramm (für 4 Variablen)
  - Sind Vollkonjunktionen benachbart, so können sie zu Päckchen zusammengefasst werden
  - In einem Päckchen dürfen 2^n Variablen zusammengefasst werden



  - Ist ein Funktionswert nicht definiert wird ein X (don't care) gesetzt. Für das X kann im KV-Diagramm eine 0 oder 1 angenommen werden



5.5 Sequentielle Systeme

70

Bei der sequentiellen Logik wird das Ausgangssignal nicht nur vom Zustand der Eingangsvariablen, sondern auch von der Zeit beeinflusst. Diese zeitliche Abhängigkeit ist durch das innere Schaltwerk gegeben. Die Grundbausteine sind Flip-Flops (Speicherelemente)

- Speicher (Selbsthaltung)
  - s = setzten
  - r = rücksetzen
- dominierend löschend
  - r und s gleichzeitig 1, so bleibt Ausgang 0
- dominierend setzend
  - r und s gleichzeitig 1, so Ausgang 1

5.5.3 Typen von Flip-Flops

72

- Nicht-taktgesteuert (synchron)
  - Speicher-Flipflop (Latch-Flipflop)
- Taktgesteuert (asynchron)
  - Taktzustandsgesteuert
    - Auffang-Flipflops
  - Taktflankengesteuert
    - Einflanken
      - Einspeicher
      - Zweispeicher mit dynamischem Zwischenspeicher
    - Zweiflanken
      - Zweispeicher (Master-Slave)

5.5.3.1 Ungetaktete Flip-Flops

73

- NAND-Basis-Flipflop

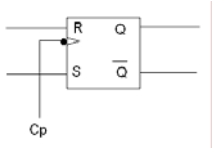
x1	x2	A1	A2
0	0	irregulär	
0	1	1	0
1	0	0	1
1	1	speichern	

- RS-Flipflop (Reset-Set) (durch invertieren der Eingänge des NAND-Basis)

5.5.3.2 Getaktete Flip-Flops

74

- Taktzustandgesteuert
  Die Eingänge werden nur freigegeben, wenn ein aktives Taktsignal anliegt (aktiv heisst in diesem Zusammenhang logisch 0 oder logisch 1).
  - Taktsignal C_p
  - Negativer aktiver Taktzustand durch Inversion
- Taktflankengesteuert
  Eingänge werden für eine kurze Zeit, bei einem Zustandswechsel des Taktsignals freigegeben.
  - Positiv-flankengesteuert (bei Wechsel von 0 auf 1)
  - Negativ-flankengesteuert (bei Wechsel von 1 auf 0)



5.5.4 Flip-Flops

77

Funktionstabelle, Wahrheitstabelle & Ansteuertabelle siehe Skript.

- Binäruntersetzer
  - Charakteristische Gleichung $Q_{n+1} = \overline{Q}_n$
  - Ändert auf jeden Taktimpuls den Zustand am Ausgang
- RS-Flip-Flop (Reset-Set-Flip-Flop)
  - Charakteristische Gleichung $Q_{n+1} = S_n \vee (Q_n \wedge \overline{R}_n)$
  - S setzten, R zurücksetzen, nie beide Eingänge auf 1
- T-Flip-Flop (Toggle-Flip-Flop)
  - Charakteristische Gleichung $Q_{n+1} = (T_n \wedge \overline{Q}_n) \vee (\overline{T}_n \wedge Q_n)$
  - Wenn T = 0 wird Ausgangssignal beibehalten. Wenn T = 1 wird Ausgangssignal invertiert
- D-Flip-Flop (Delay-Flip-Flop)
  - Charakteristische Gleichung: $Q_{n+1} = D_n$
  - Zustand am D-Eingang wird bei Taktimpuls weitergleitet
- JK-Flip-Flop (Universal-Flip-Flap)
  - Charakteristische Gleichung $Q_{n+1} = (J_n \wedge \overline{Q}_n) \vee (\overline{K}_n \wedge Q_n)$
  - Ähnlich RS-Flipflop, Zustand J=1 und K=1 ist jedoch erlaubt. Ausgang wird invertiert

5.5.5 Umwandlung von Flip-Flops

80

Durch die Beschaltung der Flip-Flop Eingänge kann jedes Flip-Flop eines bestimmten Typs in einen andern Typ umgewandelt werden.

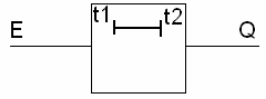
1. Erstellen der vollständigen Wahrheitstabelle des gesuchten Flip-Flops
2. Bestimmen der logischen Funktion für die Eingänge des gegebenen Flip-Flops
3. Mit KV-Diagramm minimalisieren, anschliessend die Funktionen für die Eingänge des gegebenen Flip-Flops durch die Variablen des gesuchten Flip-Flops ausdrücken

5.5.6 Logische Schaltungen mit einem Stabilen Zustand (Monoflop)

81

Flip-Flop wird durch Impuls in einen nichtstabilen Zustand gebracht. Nach Ablauf einer Zeit kehrt das Flip-Flop in den ursprünglichen Zustand zurück.
Eingesetzt in:

- Regeneration von Impulsen
- Erzeugen von Mehrfachimpulsen
- Verzögerung von Impulsen
  t_1 gibt an, um welche Zeit ansteigende Signalflanken verzögert werden
  t_2 gibt an, um welche Zeit abfallende Signalflanken verzögert werden

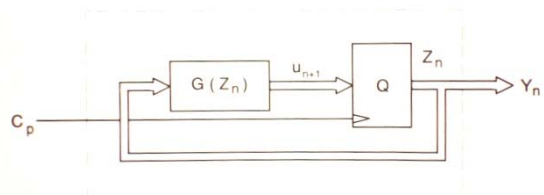


5.6 Synthese sequentieller Schaltwerke

83

5.6.1.1 Synchrongetaktete Schaltwerke ohne Eingangssignale

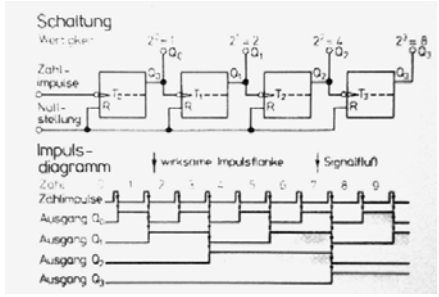
Schaltwerke ohne Eingangssignale sind Schalterwerke, die am Ausgang zyklisch einen bestimmten Code erzeugen. Sie werden allgemein als Zähler bezeichnet. Sie bestehen aus einem kombinatorischen Netzwerk und einem Mehrbitspeicher.



5.6.2 Zähler

88

- Asynchroner Dualzähler
  Es werden so viele T-Kippglieder benötigt wie der Zähler Binärstellen hat. Mit jedem Zählimpuls, beim Wechsel von 1 auf 0 am Eingang T_0, kippt der Ausgang Q_0 in die entgegen gesetzte Richtung. Es entsteht ein Impulszug mit genau halb so vielen Impulsen wie am Eingang T_0.



- Asynchrone 8-4-2-1 Zähldekade
  Ein BCD-Zähler mit dem 8-4-2-1-Code hat den gleichen Grundaufbau wie ein Dualzähler mit 4 Bits. Es muss nur dafür gesorgt werden, dass nach der Ziffer 9 wieder die Ziffer 0 entsteht und nicht die Bit-Kombination für die Ziffer 10. Dies wird durch Rücksetzen aller Kippglieder erreicht, wenn kurzzeitig die Bitkombination 1010 ansteht.

- Synchrone 8-4-2-1 Zähldekade (siehe Beispiel)
  Synchrone Zähler sind für Steuerungsaufgaben grundsätzlich besser geeignet als asynchrone Zähler, da die jeweils folgende Zahl mit Eintreffen des Taktimpulses sofort richtig ansteht.

5.6.1.2 Beispiel BCD-Zähler

84

Vorgehen:

1. Zeichnen des Zustandsdiagramms (Loop)
2. Aufstellen der Next-State-Tabelle (Z_n, Q_3, Q_2,… Z_n, Q_3, Q_2,…)
3. Wahl des einzusetzenden Speichertyps (Funktionstabelle, Ansteuertabelle aufstellen)
4. Aufstellen der Wahrheitstabelle für die Funktion U_{n+1} (Flip-Flop-Eingänge in Next-State-Tabelle einsetzten)
5. Bestimmung der minimalen Logik-Funktionen für $U_{n+1} \rightarrow U_{n+1} = G$ (KV-Diagramm)
6. Analyse der don't care Zustände (definierte X-Werte in Next-State-Tabelle einsetzten, Folgezustand herauslesen (Hauptloop – parasitärer Loop))
7. Zeichnen des Schemas

5.6.3 Entwurf einer Ablaufsteuerung

Vorgehen:

1. Aufstellen eines Zustandsdiagramms oder Funktionsplans
2. Ermittlung der Anzahl Zustände
3. Auswahl der Codierung für den Zustandsspeicher und Zeichnen des Zustandsspeichers
   a. Dualcode (Benötigt wenig Flip-Flops, Aufwand für Verknüpfungen ist jedoch am grössten)
   b. Taktstufen (1-aus-4-Code) (sehr übersichtlich, viele Flip-Flops da jeder Schritt/Takt eine Speicherstufe)

Anhang

Umfasst Vorlagen zu:

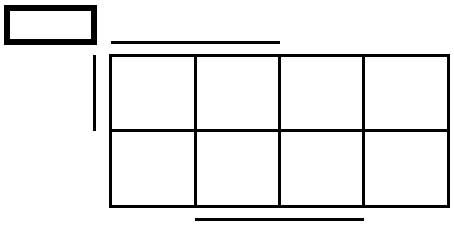
- Next-State-Tabelle
- KV-Diagramm 4x2
- KV-Diagramm 4x4

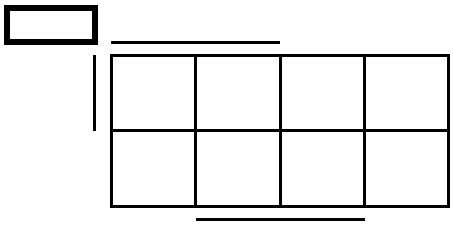
Next State Tabelle Aufgabe: _____

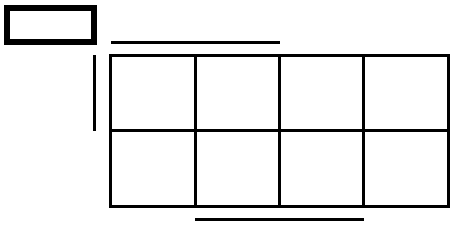
Zn					Zn+1												

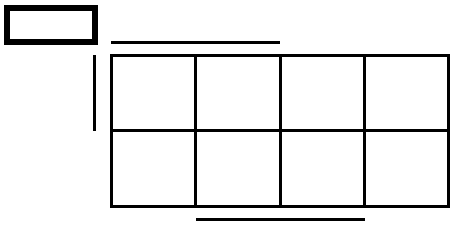
Datum:_____ Seite:_____

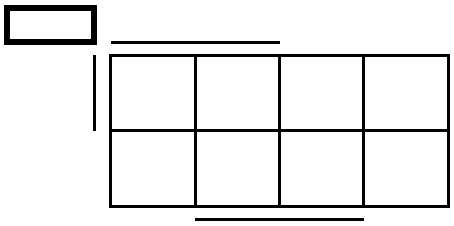
KV-Diagramm Aufgabe:_____

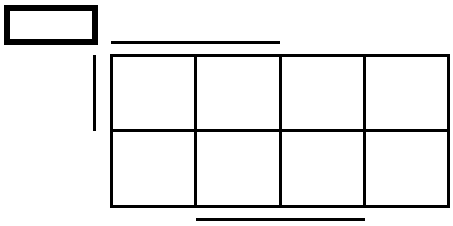


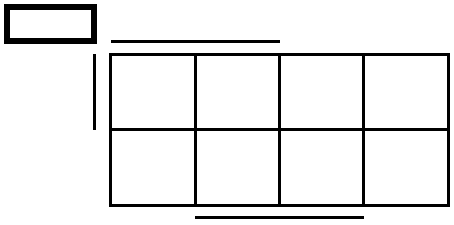


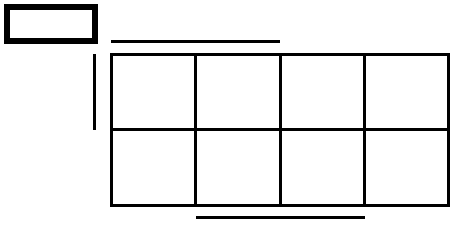






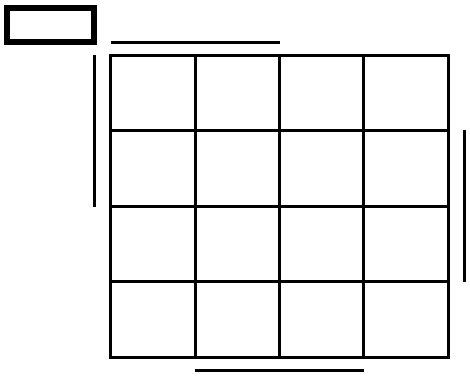


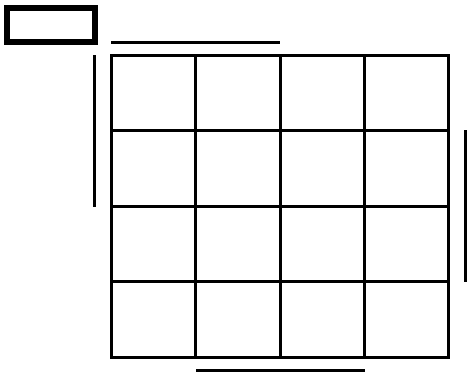


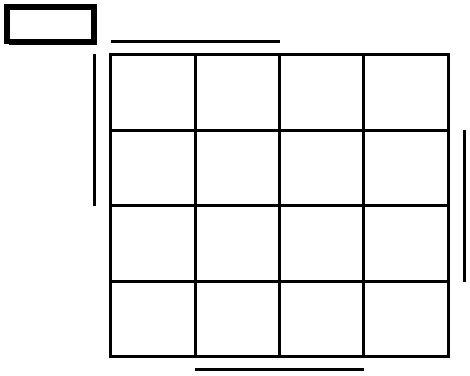


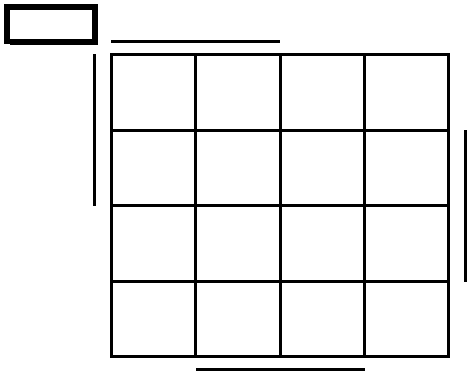
Datum:_____ Seite:_____

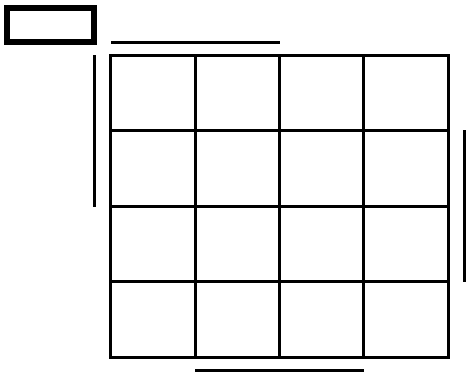
KV-Diagramm Aufgabe:_____

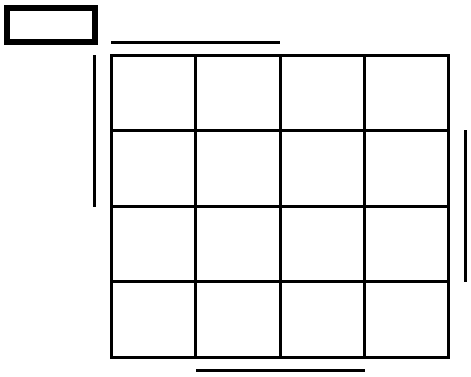


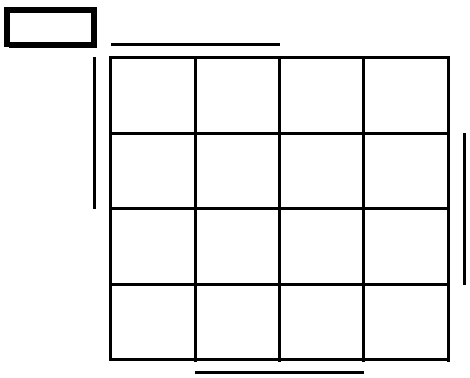


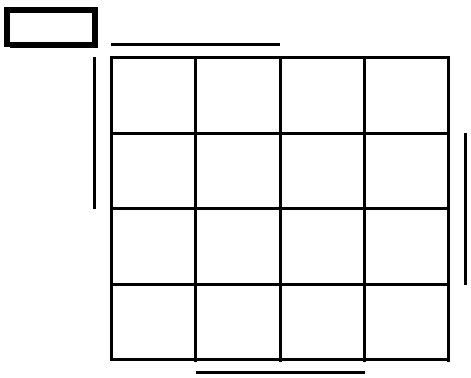












Datum:_____ Seite:_____