Digitaltechnik & Rechnersysteme

Arithmetik II und Schaltwerke

Martin Kumm

Hochschule Fulda
University of Applied Sciences

AI | Angewandte Informatik

WiSe 2023/2024

Lehrevaluation

Angewandte Informatik

In dieser Woche startet die Evaluation dieser Veranstaltung.

Sie sollten eine E-Mail bekommen haben mit dem Link zur anonymen Evaluation.

Bitte nehmen Sie an dieser Teil und geben Sie mir Rückmeldung zu allem was Ihnen gefallen/nicht gefallen hat.

Die Ergebnisse werden ausführlich besprochen und fließen ggf. noch in diese Veranstaltung ein!

Probeklausuren online

Angewandte Informatik

Teil 1 der 1. Probeklausur ist nun online

Dieser kann als Selbsttest verwendet werden (Musterlösung ist online).

Bitte nicht selbst betrügen: Erst nach der eigenen Lösung in die Musterlösung schauen!

Die »Eigene Aufgabe«

Angewandte Informatik

Sie haben mit dem Aufgabenblatt 6 die Aufgabe eine »Eigene Aufgabe« zu erstellen und hochzuladen

Falls noch nicht geschehen bitte Aufgabenstellung hochladen.

Denken Sie an: Klare Aufgabenstellung (Was ist gegeben, was gesucht?) sowie alternativer Lösungsweg!

Wenn genügend Aufgaben zusammenkommen wird eine davon in der Klausur drankommen!

Was bisher geschah...

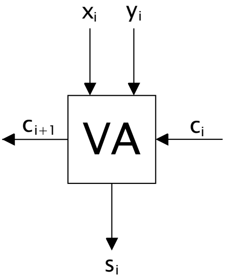
Angewandte Informatik

- Wrap-Up zu Darstellungsalternativen
- Don't cares
  - Vereinfachungsmöglichkeit im KV-Diagramm wenn Ausgabewert für bestimmte Eingabe egal (*don't care*) ist
- Spezielle Schaltnetze
  - Dekoder
  - Multiplexer (allgemein)
- Addition / Subtraktion
  - Rechenschritt einer Stelle mittels Volladdierer
  - Durch serielle Verschaltung ergibt sich Ripple-Carry Addierer

Volladdierer

Angewandte Informatik

Der Volladdierer (VA) abstrahiert die beiden Funktionen in einem Element



$$s_i = x_i \oplus y_i \oplus c_i$$

$$c_{i+1} = x_i y_i + x_i c_i + y_i c_i$$

Ohne Carry-In ($c_i = 0$) wird aus dem VA ein Halbaddierer (HA):

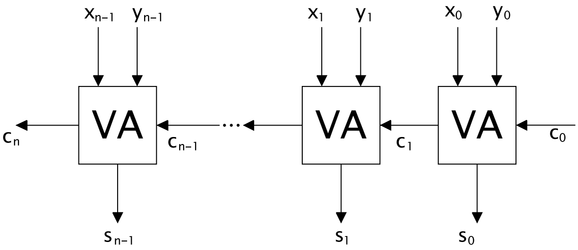
$$s = x_i \oplus y_i \text{ und } c_o = x_i y_i.$$

Ripple-Carry Addierer

Angewandte Informatik

Ripple-Carry Addierer

- Durch die Weiterschaltung der Überträge zu höherwertigen VAs gelangt man zum Ripple-Carry-Addierer (RCA).
- Direkte Umsetzung der »Papier und Bleistift« Methode.
- Der VA für das LSB (rechts) kann durch einen HA ersetzt werden, wenn kein Carry-Eingang nötig ist.



Grundlegende Subtraktion

Angewandte Informatik

Wie funktioniert die Subtraktion?

Die Subtraktion wird über eine Addition mit dem Zweierkomplement realisiert: $D = X - Y = X + (-Y)$

Beispiel: $700_{10} - 82_{10}$ ($X = 01010111100_2$, $Y = 1010010_2$):

$$
\begin{array}{rr}
\overline{Y} & 11110101101 \\
+1 & 1 \\
\hline
-Y = & 11110101110
\end{array}
$$

$$
\begin{array}{rr}
X & 01010111100 \\
+(-Y) & +11110101110 \\
\hline
= D & = 1|01001101010
\end{array}
$$

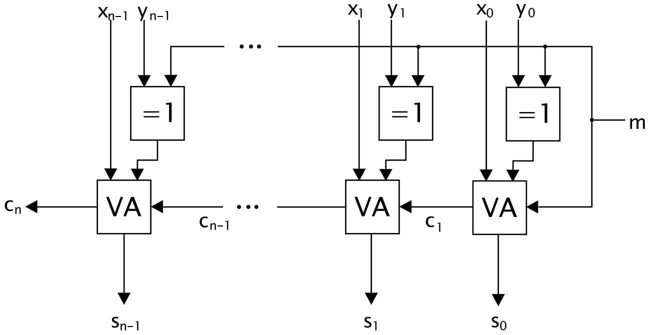
- Überträge werden im Zweierkomplement ignoriert!
- Alle Zahlen (auch Ergebnis) müssen gleiche Wortbreite haben!
- Alle Zahlen (auch Ergebnis) müssen auch darstellbar sein!

Ripple-Carry Addierer/Subtrahierer

Angewandte Informatik

Für die Subtraktion wird das Komplement gebildet und über das Carry-In eine Eins hinzuaddiert.

Ein Umschaltbarer Addierer/Subtrahierer lässt sich somit über zusätzliche XOR-Gatter realisieren ($m = 0$ Addieren, $m = 1$ Subtrahieren):



Inhalte

Multiplikation mit binären Zahlen

Die Multiplikation wird auf bitweise Multiplikation mit anschließender bitverschobener Summation reduziert:

$$P = X \times Y = \underbrace{\sum_{i=0}^{n-1} x_i 2^i}_{=X} \times Y$$

Beispiel $n = 4$: $P = x_0 2^0 Y + x_1 2^1 Y + x_2 2^2 Y + x_3 2^3 Y$

Beispiel: $0111_2 \times 1101_2 = 7_{10} \times 13_{10} = 91_{10} (= 1011011_2)$

$$\begin{aligned}
x_0 2^0 Y &= 1 \times 2^0 \times 13_{10} = & 1101_2 \\
x_1 2^1 Y &= 1 \times 2^1 \times 13_{10} = & +11010_2 \\
x_2 2^2 Y &= 1 \times 2^2 \times 13_{10} = & +110100_2 \\
x_3 2^3 Y &= 0 \times 2^3 \times 13_{10} = & +000000_2 \\
\hline
& & = 1011011_2
\end{aligned}$$

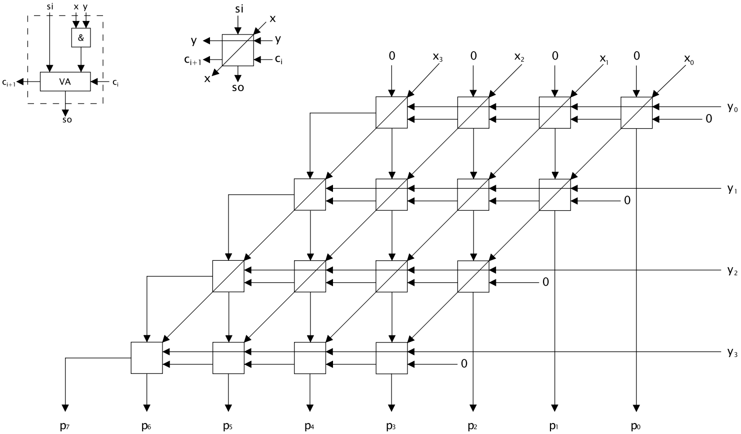
Vorlesungsaufgabe

Angewandte Informatik

Berechnen Sie das Ergebnis der Binären Multiplikation aus:

$9_{10} \times 5_{10} = 1001_2 \times 101_2 (= 45_{10} = 101101_2)$

Ripple Carry Array Multiplizierer



Moderne Multiplizierer

Angewandte Informatik

Der Ripple Carry Array Multiplizierer ist einer der einfachsten Multiplizierer aber leider auch der langsamste

Viele Logikstufen müssen hier verarbeitet werden um das Ergebnis zu erzeugen

In modernen Multiplizierern werden die Teilprodukte als auch deren Summation weitestgehend parallel berechnet

⇒ Bei Interesse sehen wir uns wieder im Modul »Computerarithmetik« im Master AI! 🙂

Arithmetisch-logische Einheit I

Angewandte Informatik

Die Arithmetisch-logische Einheit (engl. *Arithmetic Logic Unit*, ALU) ist die Recheneinheit einer CPU

Sie fasst die arithmetischen und logischen Operationen in einer Einheit zusammen

Arithmetische Operationen sind u.A.

- Addition, Subtraktion
- Vergleichsoperation
- Multiplikation
- Division

Logische Operationen sind u.A.

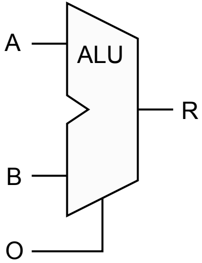
- UND, ODER, NICHT, XOR (Bitweise für ein ganzes Wort
- Bitverschiebungen nach rechts oder links

Arithmetisch-logische Einheit II

Die ALU ist rein kombinatorisch.

Die ALU hat i.d.R

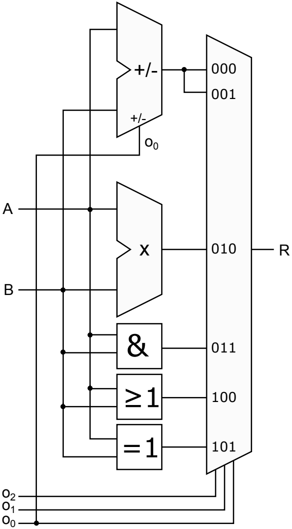
- zwei Operanden-Worte (A und B, je n Bit)
- ein Ergebnis-Wort (R, n Bit)
- ein Steuereingang zur Auswahl der Operation (O, m Bit)



Ggf. existieren noch weitere Ein-/Ausgaben wie z.B. Überträge

Aufbau einer ALU

Angewandte Informatik



o_2	o_1	o_0	Operation	R
0	0	0	$+$	$A + B$
0	0	1	$-$	$A - B$
0	1	0	$\times$	$A \times B$
0	1	1	UND	$A \wedge B$
1	0	0	ODER	$A \vee B$
1	0	1	XOR	$A \oplus B$
1	1	–	keine	–

Schaltwerke

Angewandte Informatik

Schaltnetze sind dadurch gekennzeichnet, dass zu einem bestimmten Zeitpunkt die Ausgänge **nur** von den Eingängen zu **diesem** Zeitpunkt abhängen.

Zeit spielt nur in Form von Verzögerungen (Schaltzeiten) eine Rolle.

Schaltnetze sind für die Erfassung von Abläufen, bei denen die **Vorgeschichte** eine Rolle spielt, nicht geeignet.

Schaltwerke

Zur Erfassung von Abläufen ist eine **Speicherung** der Vorgeschichte erforderlich.

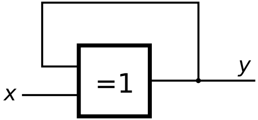
Ein einfaches Beispiel stellt ein Zähler dar, wobei die Anzahl der Zählereignisse als „Zählerstand" repräsentiert wird.

Solche Schaltungen bezeichnet man als **Schaltwerke** oder **sequentielle Schaltungen** oder **asynchrone Automaten**.

Die Speicherung wird durch das Einführen von Rückkopplungen realisiert.

Beispiel: XOR mit Rückkopplung

Beispiel: $y = x \oplus y$



Ausgangswert hängt nun von **vorherigem** Ausgangswert ab!
Wir nennen alten Ausgangswert y^t, den neuen $y^{t+\tau}$
(τ kann man sich als sehr kleine Zeitspanne vorstellen).

y^t	x	$y^{t+\tau}$	
0	0	0	← Ausgang bleibt stabil: $y^{t+\tau} = y^t = 0$
0	1	1	← Ausgang wechselt von 0 auf 1
1	0	1	← Ausgang bleibt stabil: $y^{t+\tau} = y^t = 1$
1	1	0	← Ausgang wechselt von 1 auf 0

Zustände und Zustandsvektor

Aufgrund der Historie kann sich das Schaltwerk in unterschiedlichen Zuständen befinden.

Variablen von denen der dieser Zustand beeinflusst wird werden als Zustandsvariablen q_i, $1 \leq i \leq k$ bezeichnet.

Die Menge der möglichen Zustände definiert einen Zustandsraum $Q = \{0, 1\}^k$.

Für ein Schaltwerk mit k Zustandsvariablen enthält Q damit $l = 2^k$ Elemente.

Zustandsübergangsverhalten

Das Zustandsübergangsverhalten beschreibt die Übergänge von einem Zustand $q^t \in Q$ in einen Folgezustand $q^{t+\tau} \in Q$:

$$q^{t+\tau} = G(q^t, x), \quad q^t, q^{t+\tau} \in Q$$

G beschreibt, wie sich zum Zeitpunkt $t + \tau$ der Zustand eines Schaltwerks $q^{t+\tau}$ aus dem Zustand zu einem zurückliegenden Zeitpunkt q^t und dem Eingangsvektor x berechnen lässt.

τ kann hierbei eine gedachte Zeitspanne > 0 sein oder konkret in Gatterlaufzeiten ausgedrückt werden.

G wird als Zustandsübergangsfunktion (ZÜF) bezeichnet.

Zustandsübergänge werden auch Transitionen genannt.

Berechnung der Ausgänge

Angewandte Informatik

Das Verhalten zum Zeitpunkt t wird beschrieben durch das Ausgangsverhalten (wie beim Schaltnetz) und das Zustandübergangsverhalten.

Ausgangsvektor zum Zeitpunkt t:

$$y = F(q^t, x), \; q^t \in Q$$

F wird als Ausgangsfunktion (AF) bezeichnet.

Beschreibung als Übergangstabelle

Zur Beschreibung des Verhaltens von Schaltwerken eignen sich die **Zustandsübergangstabelle** oder **Übergangstabelle**:

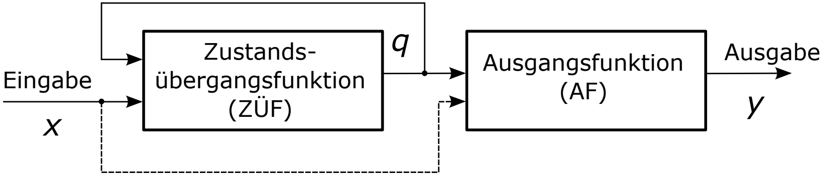
$q_{k-1}^t \ldots q_1^t \; q_0^t$	$x_{n-1}^t \ldots x_1^t \; x_0^t$	$q_{k-1}^{t+\tau} \ldots q_1^{t+\tau} \; q_0^{t+\tau}$	$y_{m-1}^t \ldots y_1^t \; y_0^t$
$00 \ldots 00$	$00 \ldots 00$		
$\vdots$	$\vdots$		
$11 \ldots 11$	$11 \ldots 11$		

k: Anzahl Zustandsbits, n: Anzahl Eingänge, m: Anzahl Ausgänge

In jedem möglichen Zustand werden sämtliche Eingangskombinationen betrachtet.

Darstellung eines allgemeinen Schaltwerks

Angewandte Informatik



Die ZÜF berechnet den Folge-Zustandsvektor $q^{t+\tau}$ in Abhängigkeit des aktuellen Zustandsvektors q^t und der aktuellen Eingabe x.

Die AF berechnet die Ausgabe y in Abhängigkeit des aktuellen Zustandsvektors q^t und der aktuellen Eingabe x.

Moore und Mealy Automaten

Angewandte Informatik

Üblicherweise werden zwei Typen von Automaten unterscheiden:

MEALY Automat: ist der allgemeine Fall

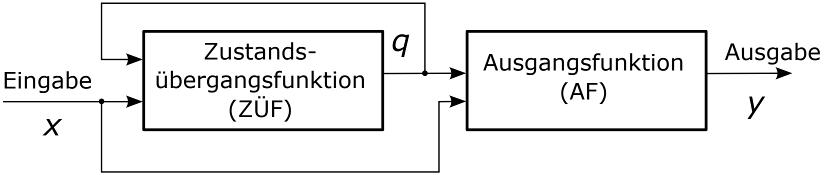
MOORE Automat: Einschränkung, dass die Ausgänge nur von den Zuständen abhängen:

$$y = F(q)$$

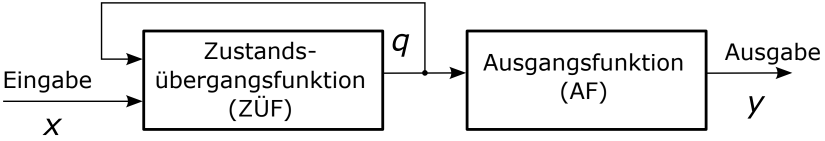
Moore und Mealy Automaten

Angewandte Informatik

MEALY Automat:



MOORE Automat:

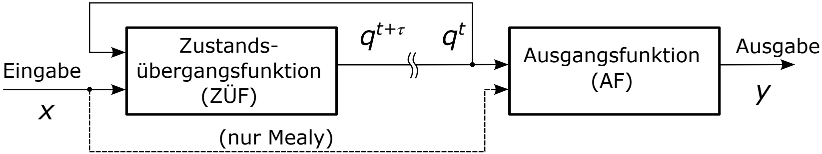


Schaltwerksanalyse

Angewandte Informatik

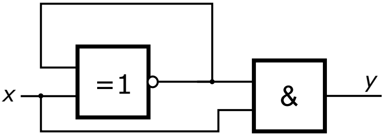
Vorgegeben ist ein Schaltbild.

Durch die **Analyse** soll auf die Funktion geschlossen werden.
Dazu werden Rückkopplungen so aufgetrennt, dass die Struktur eines Schaltnetzes entsteht (rückkopplungs**frei**).



Vorlesungsaufgabe

Analysieren Sie das folgende Schaltwerk:



Erstellen Sie die Zustandsübergangs- und Ausgangstabelle

Lösung Vorlesungsaufgabe

Stabile Zustände

Angewandte Informatik

Transitionen (Zustandsübergänge), bei denen (alle) q^t und $q^{t+\tau}$ übereinstimmen führen zu **stabilen Zuständen**.

D.h. für die Eingabekombination dieser Transition bleibt der Automat in seinem Zustand.

Vorlesungsaufgabe

Markieren Sie die Transitionen, welche zu stabilen Zuständen führen:

q^t	x	$q^{t+\tau}$	y
0	0	1	0
0	1	0	0
1	0	0	0
1	1	1	1

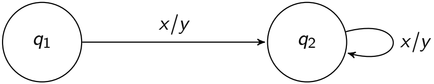
Beschreibung als Übergangsgraph

Angewandte Informatik

Übergangsgraph/**Zustandsübergangsgraph**: Den Knoten (Ecken) des Graphen werden die Zustände zugeordnet. Zustandsübergänge (Transitionen) entsprechen gerichteten Kanten (Pfeile).

Die **Knoten** werden durch Kreise dargestellt und erhalten als Beschriftung die Zustandsbezeichnung.

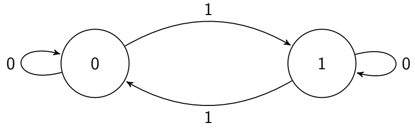
Die **Kanten** werden mit dem Eingangsvektor beschriftet, der den entsprechenden Übergang auslöst, sowie dem Ausgabevektor:



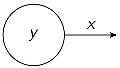
Beispiel: XOR mit Rückkopplung

AI Angewandte Informatik

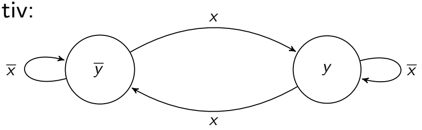
y^t	x	$y^{t+\tau}$	
0	0	0	← Ausgang bleibt stabil: $y^{t+\tau} = y^t = 0$
0	1	1	← Ausgang wechselt von 0 auf 1
1	0	1	← Ausgang bleibt stabil: $y^{t+\tau} = y^t = 1$
1	1	0	← Ausgang wechselt von 1 auf 0



Notation:

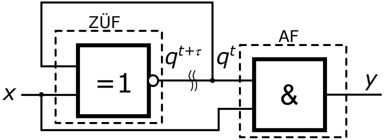


Alternativ:



Vorlesungsaufgabe

Erstellen Sie für das Schaltwerk den Zustandsübergangsgraph.



q^t	x	$q^{t+\tau}$	y
0	0	1	0
0	1	0	0
1	0	0	0
1	1	1	1

Notation: