MINIMIERUNG UND SCHALTUNGSENTWURF

Vorgehensweise

Die Vorgehensweise besteht aus einer allgemeinen Ausführung von vier Schritten:

1. Problemanalyse:

   Identifikation der Eingabe- und Ausgabevariablen.

2. Wahrheitstabelle:

   Systematische Erfassung aller Einkombinationen und Ausgabekombinationen.

3. Boolesche Funktion:

   Ableitung der Schaltfunktion aus der Wahrheitstabelle.

4. Technische Realisierung:

   Umsetzung der Funktion in eine physikalische Schaltung mit Logikgattern.

Normalformen boolescher Funktionen

Wortschatz

- Literal: Variable oder ihre Negation.
- Minterm: Vollständige Konjunktion aller Literale also die UND-Verknüpfung.

$$\prod_{i=0}^{n-1} \sim x_i$$

wobei $\sim x_i$ für x_i oder $\overline{x_i}$ steht

- Maxterm: Vollständige Disjunktion aller Literale also die ODER-Verknüpfung.

$$\sum_{i=0}^{n-1} \sim x_i$$

wobei $\sim x_i$ für x_i oder $\overline{x_i}$ steht

- Disjunktive Normalform oder DNF: besteht aus der Disjunktion von Mintermen.
- Konjunktive Normalform oder KNF: besteht aus der Konjunktion von Maxtermen.

Ausgezeichnete Normalformen

Ausgezeichnete Normalformen sind kanonische Formen:

- Ausgezeichnete, DNF oder ADNF: Vollständige Summe aller Minterme
- Ausgezeichnete, KNF oder AKNF: Vollständiges Produkt aller Maxterme

Minimale Normalformen

Minimale Normalformen sind optimierte Darstellungen:

- MDNF: Kürzeste mögliche DNF
- MKNF: Kürzeste mögliche KNF

Strategische Auswahl

Die Wahl der Normalform hängt von der Verteilung der Funktionswerte ab:

- Bei vielen 0en ist die AKNF effizienter

- Bei vielen 1en ist die ADNF vorteilhafter
- Don't-Care-Terme ermöglichen zusätzliche Optimierungsmöglichkeiten

Verallgemeinerung von De Morgan

Theorem

Sei $(\mathscr{L}, \wedge, \vee, \neg, 0, 1)$ eine Boolesche Algebra und seien $a_1, a_2, \ldots, a_n \in \mathscr{L}$. Dann gelten:

$$\neg(a_1 \wedge a_2 \wedge \cdots \wedge a_n) = \neg a_1 \vee \neg a_2 \vee \cdots \vee \neg a_n$$
$$\neg(a_1 \vee a_2 \vee \cdots \vee a_n) = \neg a_1 \wedge \neg a_2 \wedge \cdots \wedge \neg a_n$$

In kompakter Schreibweise:

$$\neg\left(\bigwedge_{i=1}^{n} a_i\right) = \bigvee_{i=1}^{n} \neg a_i \quad \text{und} \quad \neg\left(\bigvee_{i=1}^{n} a_i\right) = \bigwedge_{i=1}^{n} \neg a_i$$

Beweis

Wir beweisen die erste Gleichung mittels vollständiger Induktion über n. Der Beweis der zweiten Gleichung verläuft analog.

Für $n = 2$ erhalten wir die klassischen De Morgan'schen Gesetze:

$$\neg(a_1 \wedge a_2) = \neg a_1 \vee \neg a_2$$

Dies ist ein Axiom beziehungsweise eine grundlegende Eigenschaft jeder Booleschen Algebra.

Die Behauptung gelte für ein $n = k \geq 2$, das heißt:

$$\neg\left(\bigwedge_{i=1}^{k} a_i\right) = \bigvee_{i=1}^{k} \neg a_i$$

Wir betrachten den Ausdruck für $k + 1$ Elemente:

$$
\begin{aligned}
\neg \left(\bigwedge_{i=1}^{k+1} a_i \right) &= \neg \left(\left(\bigwedge_{i=1}^{k} a_i \right) \wedge a_{k+1} \right) \\
&= \neg \left(\bigwedge_{i=1}^{k} a_i \right) \vee \neg a_{k+1} \\
&= \left(\bigvee_{i=1}^{k} \neg a_i \right) \vee \neg a_{k+1} \\
&= \bigvee_{i=1}^{k+1} \neg a_i
\end{aligned}
$$

Damit ist die erste Gleichung für alle $n \geq 2$ bewiesen.

Der Beweis der zweiten Gleichung erfolgt durch duale Argumentation oder durch Anwendung der ersten Gleichung auf die negierten Elemente.

$\square$

Erzeugung der Normalformen

Jede vorgegebene boolesche Funktion (Wertetabelle, Algebraische Funktion, Schaltbild) kann eindeutig durch eine ADN oder eine AKN dargestellt werden.

Karnaugh-Veitch-Diagramme, KV-Diagramme

KV-Diagramme bieten eine visuelle Methode zur Minimierung für 3 bis 4 Variable. Die Nachbarschaftsbeziehung im Diagramm entspricht der logischen Nachbarschaft im Wahrheitsfeld.

Quine-McCluskey-Algorithmus

Dieses systematische Verfahren ist für beliebig viele Variablen geeignet und algorithmisch implementierbar. Der Algorithmus ist wie folgt:

1. Codierung

   Minterme werden als Binärzahlen dargestellt

2. <u>Gruppierung</u>

   Terme werden nach Anzahl der Einsen sortiert

3. <u>Verschmelzung</u>

   Benachbarte Terme werden kombiniert wobei die Differenz genau ein Bit ist

4. <u>Primimplikanten-Identifikation</u>

   Nicht kombinierbare Terme werden als Primimplikanten markiert

5. <u>Essenzielle Auswahl</u>

   Primimplikanten, die eindeutige 1en abdecken, werden ausgewählt

6. <u>Minimale Überdeckung</u>

   Restliche Abdeckung mit minimaler Anzahl an Primimplikanten