142.0/86D 213

ERKLÄRUNG

Ich erkläre hiermit, die Arbeit selbständig verfaßt und andere als die angegebenen Hilfsmittel nicht benutzt zu haben.

Ilmenau, 21. Januar 1986

Simeon

142.0/86D 213

Technische Hochschule Ilmenau
Sektion Informationstechnik und theoretische Elektrotechnik

Thema der Diplomarbeit:

Programm zur Konvertierung des symbolischen Layouts in das Datenformat des physikalischen Layouts

Signatur: 142.0/86D 213

T H E S E N

1. Die vorliegende Diplomarbeit bietet eine Algorithmenkonzeption zur Konvertierung der Daten des symbolischen Layouts eines CMOS Gate-Array-Schaltkreises in das Format des physikalischen Layouts, wie es vom Schaltkreishersteller verwendet wird.

2. Die programmtechnische Realisierung des Konvertierungsalgorithmus' ist für einen Rechner der KULON-Serie in der Sprache FORTRAN-77 entwickelt worden.

3. Die Konvertierung des symbolischen Layouts ist unter Einsatz eines modifizierten Left-Edge-Algorithmus' durchgeführt worden.

4. Dem Nutzer stehen 2 Tasks zur Verfügung: RASTER (etwa 26K) zur Eingabe der Rasterabstände des physikalischen Layouts und SYLAY (etwa 54K) für die eigentliche Konvertierung, die nacheinander gestartet werden müssen.

5. Ausgangsdateien sind die Datenfiles der symbolischen Leitbahn- und Kontaktfensterebenen.

6. Der Zeichengraph wird in die spezifische Dateistruktur DS11 umgewandelt.

7. Das Konvertierungsprogramm SYLAY besteht aus 19 Quellmodulen. Die jeweiligen Objektmodule sind in einer Library zusammengefaßt.

8. Eine Besonderheit des vorliegenden Programms ist, daß die anzugebenden Leitzug- und Kontaktfensterbreiten im physikalischen Layout gerade Zahlen sind.

9. Weitere Entwicklungsmöglichkeiten sind gegeben.

TECHNISCHE HOCHSCHULE ILMENAU

Sektion Informationstechnik u. Theoretische Elektrotechnik

DIPLOM - THEMA

für cand.ing. Plamen Simeonov

geb. am: 25. 9. 1960

Fachrichtung: Informationstechnik

Wissenschaftsbereich: Schaltungstechnik und elektron. Meßtechnik

Verantwortl. Hochschullehrer: Prof. Dr. sc. Scarbata

Betreuender wiss. Mitarbeiter: Dr. A. Müller

Themen-Nr.: 142.0/86 D 213

Kurzthema: Programm zur Konvertierung des symbolischen Layouts in das Datenformat des physikalischen Layouts

Konvertierungsprogramm der Daten des symbolischen Layouts eines Gate-Array-Schaltkreises in das Format des physikalischen Layouts.

Ausgegeben am: 2. 9. 85

Ilmenau, den 18. 11. 1985

Abgabetermin: 21. 1. 86

Sektionsdirektor

Prof. Dr.-Ing. W. Kienast

Bereichsleiter

i.V. Thomas

Prof. Dr.-Ing. W. Kemnitz

SG 149/042/77/142.3

142.0/86D 213

Hiermit möchte ich mich bei Herrn Prof. Dr. sc. techn. G. Scarbata und Herrn Dr. A. Müller, unter deren Anleitung die vorliegende Diplomarbeit entstanden ist, für Ihre sachkundige Betreuung und Unterstützung bedanken.

Inhaltsverzeichnis Seite

1. Einleitung

Für den Entwurf einer integrierten Schaltung bis hin zu ihrer Fertigung benötigt man einen sehr großen Aufwand. Dieser ist nur dann gerechtfertigt, wenn auch große Mengen dieser integrierten Schaltung abgesetzt werden können. Da die Komplexität eines Systems das Anwendungsfeld begrenzt (d. h. je höher der Integrationsgrad eines Chips ist, um so spezieller einsetzbar ist es im allgemeinen), ergibt sich die Notwendigkeit für die VLSI-Technik, technologisch eine allzu starke Individualisierung zu vermeiden. Dieses Problem ist bisher auf verschiedene Weise gelöst worden - z. B. Individualisierung von Festwertspeichern durch Programmierung beim Anwender (PROM, EPROM, EEPROM), Individualisierung von unspezifischer Hardware durch Speicherprogrammierung (/uP) usw.

In den letzten Jahren hat sich eine sehr erfolgreiche Methode unter dem Namen "Masterslice" bzw. "Gate Array" durchgesetzt, wobei der Anwender sehr rasch ein hochintegriertes Chip aus logischen Schaltungen erhält. Der Hersteller entwirft und produziert zunächst einheitliche Chips, die mit einer Vielzahl von elementaren Schaltungen (Transistoren bzw. Gates) in einer Matrixstruktur (Array) bedeckt sind, wobei die Strukturierung der Leitungsebenen noch nicht durchgeführt ist. Der Anwender kann dann eine seinen Forderungen entsprechende Verdrahtung entwerfen bzw. ein Verbindungsnetz so in eine vorbereitete Maskenzeichnung eintragen, daß die miteinander verbundenen elementaren logischen Schaltungen die gewünschte Gesamtschaltung ergeben. Die wesentlichen Vorteile dieser Technik sind:

- Der Anwender verrät nicht sein "know-how" an den Hersteller
- Er erhält eine maßgeschneiderte hochintegrierte Schaltung zu einem angebrachten Preis
- Kurze Wartezeit auf die fertigen Chips
- Hohe Zuverlässigkeit der hergestellten Schaltkreise
- Begünstigt Kleinmengenproduktion

Die vorliegende Diplomarbeit ist ein Teil des Layoutentwurfs eines GA-Schaltkreises. Das symbolische Layout der Leitbahnen wird in ei physikalisches Layout konvertiert.

59012 VV Freiberg Ag 307/82 III/15/4 2391/1 1182 250,0 T/A 7434

2. Übersicht

Ein Vorzug, der besonders für den Gate-Array-Erfolg spricht, ist die reguläre Struktur, die für die Layout-Automatisierung geeignet ist (s. Anhang 4, Abb. 1).

Ein Gate Array besteht aus logischen Zellen (NANDs und NORs), die innerhalb des Chips in Zeilen und Spalten angeordnet sind. Jede Zeile besitzt elektrische Anschlußklemmenpaare auf den oberen und unteren Seiten (s. Anhang 4, Abb. 2). Die von I/O-Zellen umgebene Fläche außerhalb dieser Zellen ist das Verdrahtungsfeld, das in horizontalen und vertikalen Kanälen aufgeteilt worden ist. Jeder Kanal besitzt eine Anzahl von Spuren, auf welche die Leitbahnen gelegt sind. Die Spurenanzahl wird als Kapazität des Kanals bezeichnet. Abb. 3 (Anhang 4) stellt eine interne Kanalstruktur dar.

Gate Arrays lassen sich durch ihre Strukturiertheit algorithmisch besser bearbeiten und ermöglichen dadurch einen schnellen Entwurfsprozeß. Die beschriebene Technik, die häufig auch als "Uncommitted Logic Array" (ULA) bezeichnet wird, ist in einer Vielzahl von Layout-Konzepten entwickelt worden (/3/ - /10/). Mit Hilfe heuristischer Algorithmen werden rechnergestützte Methoden zum automatischen Entwurf ausgearbeitet, die bestimmte Bezugskriterien wie Entwurfsregeln, Globalzellengrenzen oder die benutzte Chipfläche minimieren.

Der Entwurf eines Gate Arrays besteht im wesentlichen aus 2 Hauptschritten:

- Plazierung (placement) bzw. Einordnung (ordering) der hierarchischen Strukturen innerhalb eines Kristalls und
- Verdrahtung (routing) der Mikrozellen in diesen Strukturen.

Die erste Phase eines Gate Array Layouts ist die physische Plazierung der Elemente einer Makrolibrary in einem geordneten Muster, das die effektive Nutzung der Routingfläche ermöglicht. Der iterative Austausch der Elemente ist der Kern dieser Phase. Das Hauptkriterium des Gate Array Layouts ist die Neuverteilung oder das Vermeiden der Engpässe (bottlenecks).

Die zweite Phase des Gate Array Layouts ist die globale Verdrahtung (global routing). Der Chip wird zunächst durch ein grobes Gitter in Globalzellen (global cells) zwecks Reduzierung der Komplexität zerlegt (s. Anhang 4, Abb. 4). Es folgt das Generieren der Globalrouten für alle Signalnetze über das grobe Gitter (/15/, /17/). Die Lage der Globalzellengrenzen ist entscheidend für die Ausführung der Verdrahtung, denn verschiedene Gate Array Architekturen verlangen verschiedenes Implementieren der Globalzellen. Wenn die Plazierung und die globale Verdrahtung abgeschlossen sind, wird eine detaillierte Verdrahtung ausgeführt, die jedem Signal eine physische Spur zuordnet (Vertical Assignment). Anschließend erfolgt die sogenannte Maze Routing, die alle übrigen Subnetze verbindet. Gelegentlich kann auch eine iterative Wiederverdrahtung (Rerouting) zwecks Verbesserung der lokalen Dichte vorkommen.

Eine ausführliche Diskussion der Arbeitsphasen der Gate Array Layouts und der damit verbundenen Probleme ist in /16/ vorgestellt. Es muß erwähnt werden, daß z. Z. kein universelles Modell für die automatische Entwicklung eines Gate Array Layouts existiert. Die konventionellen Methoden zur Lösung bestimmter Probleme (z. B. die Einordnung der Verdrahtung von Signalnetzen oder das Vermeiden von Engpässen wie Moore-, Lee- oder Line-Search-Algorithmen (/22/, /23/, /24/)) haben im allgemeinen begrenzte Anwendungsmöglichkeiten. Neue Ideen und Techniken treten immer wieder auf (/12/ - /19/). Diese Probleme liegen aber weit über dem Themenkreis der vorliegenden Arbeit und sollten hier nur überblicksmäßig dargestellt werden.

3. Problemstellung

Die Leitbahn- und Kontaktfenstermasken bei der Herstellung eines GA-Schaltkreises werden vom Kunden entwickelt. Zuerst wird die Chipfläche in mehrere Teile zerlegt. Der folgende Schritt ist die Aufstellung des symbolischen Layouts, der meistens kombiniert interaktiv und automatisch durchgeführt wird. Jedes Chip wird einzeln verarbeitet. Die dort befindlichen Strukturen (Transistoren und Kontaktflächen) werden als ASCII-Zeichen auf einer Textseite dargestellt. Das Raster des physischen Layouts, das die nichtregulären Abstände zwischen den einzelnen Zeichen angibt, wird separat eingegeben und abgespeichert (s. Anhang 2). Danach befassen sich spe-

zielle Routing Programme mit der Verdrahtung der jeweiligen Strukturen. Anschließend werden alle Textseiten zu einer Leiterbahnmaske zusammengesetzt.

Das Thema dieser Arbeit ist die Konvertierung des symbolischen Layouts in das Datenformat des physikalischen Layouts, das für die automatische Fertigung der Leitbahnmasken im Pattern-Generator benutzt wird.

Der Entwurf eines CMOS GA-Schaltkreises erfolgt technologisch in drei Ebenen. Zunächst erstellt man die Grundmodulzellen auf dem Master. Danach werden die Kontaktfenster auf einer anderen Ebene aufgebaut. Die eigentlichen Verbindungen zwischen den einzelnen Elementen finden im letzten Schritt - dem Durchziehen der Leiterbahnen - statt. Ihre Breite ist bei einem CMOS Gate Array kleiner als die Kontaktfensterbreite. Dort, wo die Leiterbahnen über Kontaktfenster verlaufen, werden die letzten aufgeblendet und die Leiterzüge verbreitet.

Das symbolische Layout der Leiterbahnebene stellt die funktionellen Verbindungen zwischen den Modulzellen dar. Es wird recordweise (in Records jeweils 40 Byte) eingelesen, wobei die einzelnen Records die Elemente einer Matrixstruktur sind (s. Anhang 4, Abb. 5).

Spezielle Zeichen auf dieser Textseite stellen den Verlauf der Leitbahnen dar. Z. B. bedeutet das '!'-Zeichen vertikale Lage, das '-'-Zeichen horitzontale Lage, die '/'- und '\'-Zeichen weisen auf eine schräge Lage hin, und das '+'-Zeichen ist für Richtungsänderung gesetzt (s. Anhang 1). Anfang und Ende eines Leiterzuges werden durch '$'-Zeichen gekennzeichnet (Anhang 4, Abb. 6).

Die Konvertierung dieses symbolischen Layouts in ein physikalisches Layout wird unter Einsatz eines modifizierten Left-Edge-Algorythmus durchgeführt. Der Zeichengraph wird in eine spezifische Dateistruktur - dem sogenannten DS11-Format - umgewandelt (s. Anhang 3).

4. Beschreibung des Konvertierungsalgorithmus'

Eine symbolische Leitbahn wird als ein verzweigter Graph dargestellt (s. Anhang 4, Abb. 6-a). Um sie zu einem abgeschlossenen Polygonzug in der physikalischen Layoutebene zu überführen, wird der folgende Algorithmus angewandt:

Phase 1

Schritt 1: Überführung der symbolischen Leitbahn- und Kontaktfensterrecordmatrizen RM in die Recordmatrizen RM' (s. Anhang 1 und Anhang 4, Abb. 5)

Schritt 2: Die symbolische Leitbahnrecordmatrix wird zeilenweise nach einem Knick- oder Anfangssymbol ('+' oder '§') durchsucht. Wenn kein solches Zeichen gefunden wird, ist die Konvertierung abgeschlossen. (Ende)

Schritt 3: Merken der Koordinaten (l, k) des Anfangssymbols in der symbolischen Leitbahnzeichenmatrix (s. Anhang 1)

Schritt 4: Nachbarschaftsuntersuchung zwecks Bestimmung des Linienzugverlaufs. Beginn der Suche an der Stelle (l, k-1) in Richtung gegen den Uhrzeigersinn.

Bemerkung: Falls das Anfangssymbol ein Knickpunkt ist (s. Anhang 4, Abb. 6) wird zunächst der untenliegende Zweig gewählt. Der rechtsliegende Zweig wird erst im Schritt 32 durchlaufen.

Schritt 5: Transformation der Symbolkoordinaten (l, k) in absolute Layoutkoordinaten (x,y) mittels der Rasterdateien DDX und DDY (s. Anhang 2)

Frage 6: Überprüfen der Kontaktfensterebene (symbolische Kontaktfensterrecordmatrix) nach einem an der Symbolstelle (l,k) liegenden Kontaktfenster

(6)

Schritt 7: In einem Stack werden die Koordinaten der links im Bezug auf die Richtung stehenden Knickpunkte (left-edge) des physikalischen Leitbahnlayouts entsprechend der Nachbarschaftskonfiguration abgespeichert, wobei die Kontaktfenster (falls es ein solches gibt (s. Frage 6)) und Leitbahnbreiten berücksichtigt werden.

Schritt 8: Übergang zum nächsten Symbol in Leitbahnrichtung. Die logische Variable HIN (für Hinlauf auf dem Linienzug) wird .TRUE. gesetzt.

Phase 2:

Frage 9: Überprüfen des Symbols und Verzweigen entsprechend seiner Art zur Frage 10 (Zweigende : '$'), bzw. Schritt 19 (Strecke: '-', '|', '/' oder '\') oder Schritt 24 (Knick: '+')

Frage 1Ø (Zweigenede): Überprüfen der Koordinaten des jeweiligen Symbols nach einem Abschluß des Polygonzuges (s. Schritt 3). Falls ein solches Ende gefunden wird, geht man zum Schritt 15.

Schritt 11: Wiederholung von Schritt 5

Frage 12: Wiederholung von Frage 6

Schritt 13: Im Stack werden die Koordinaten der um das Symbol liegenden Knickpunkte des physikalischen Leitbahnlayouts abgespeichert, wobei die Breite der Kontaktfenster und der Leitbahn berücksichtigt wird.

Schritt 14: Löschen des aktuellen '$'-Symbols und Übergang zum vorhergehenden Symbol. Die logische Variable HIN wird .FALSE. gesetzt (Rücklauf). Anschließend - Übergang zur Frage 9

Schritt 15: (Polygonzugende): siehe Schritt 5

Frage 16: siehe Frage 6

59012 VV Freiberg Ag 307/82 III/15/4 2391/1 1182 250,0 T/A 7434

Schritt 17: siehe Schritt 7

Schritt 18: Löschen des letzten Zeichens und Überführung der im Stack abgespeicherten Knickpunktkoordinaten des Symbolgraphen in das DS11-Format. Gehe zu Schritt 2

Phase 3:

Schritt 19 (Strecke): Nachbarschaftsuntersuchung zwecks Bestimmung des Linienverlaufs

Schritt 2Ø: siehe Schritt 5

Frage 21: siehe Frage 6

Schritt 22: siehe Schritt 7

Schritt 23: Übergang zum nächsten Symbol in der Leitbahnrichtung, wobei das aktuelle Symbol gelöscht wird, wenn es sich um einen Rückauf auf der Strecke handelt (d. h. HIN=.FALSE.). Anschließend Übergang zur Frage 9

Schritt 24 (Knick): Bestimmung der Anzahl der Nachbarsymbole

Frage 25: Ist die Nachbaranzahl größer als 1, gehe zu Schritt 34

Frage 26: Überprüfen der logischen Variable HIN. Ist HIN=.TRUE.: gehe zu Schritt 34

Frage 27: Überprüfen der Koordinaten des jeweiligen '+!-Symbols nach einem rechtsliegenden Zweig (s. Schritt 3 und Schritt 4 - Bemerkung)
Ist ein rechtsliegender Zweige vorhanden: gehe zu Schritt 28
Kein rechtsliegender Zweig: gehe zu Schritt 33

Schritt 28: siehe Schritt 5

Frage 29: siehe Frage 6

Schritt 3Ø: siehe Schritt 7

59012 VV Freiberg Ag 307/82 III/15/4 2391/1 1182 250,0 T/A 7434

Schritt 31: Das aktuelle '+'-Symbol wird mit einem '$'-Symbol überschrieben. Die Variable HIN für Hinlauf wird gleich .TRUE. gesetzt.

Schritt 32: Übergang zum folgenden Symbol - rechter Linienzweig. Anschließend gehe zu Frage 9

Schritt 33: Löschen des Knicksymbols '+'

Schritt 34: siehe Schritt 5

Frage 35: siehe Frage 6

Schritt 36: siehe Schritt 7

Schritt 37: Übergang zum folgenden Symbol.
Die logische Variable HIN wird gleich .TRUE. gesetzt.
Anschließend gehe zu Frage 9

Der Programmablaufplan im Anhang 5 veranschaulicht den Verlauf des Algorithmus'.

5. Programmablauf

Die programmtechnische Realisierung des Konvertierungsalgorithmus wurde an einem Rechner der KULON-Serie in der Sprache FORTRAN-77 durchgeführt. Es wurden 2 Tasks erstellt - RASTER (etwa 26K) zur Eingabe der Rasterabstände des physikalischen Layouts und SYLAY (etwa 54 K) für die eigentliche Konvertierung, die nacheinander gestartet werden müssen. Ausgangsdateien sind die Datenfiles der symbolischen Leitbahn- und Kontaktfensterebenen (SQWC: name.LAY bzw. KON: name.KON), die anschließend in der physikalischen Layoutdatei DSØ: DS11.DS (s. Anhang 3)gespeichert werden.

Als Hilfsmodule kann man die Tasks LAYEDI (etwa 77K) zum Editieren eines symbolischen Layouts und BILD (etwa 3Ø K) für die Displaykontrolle der DS11-Datei verwenden.

5.1. Rastereingabe

Das Programm RASTER ermöglicht die Eingabe der Abstände zwischen den einzelnen Rasterpunkten. Das erfolgt recordweise in Vektoren von jeweils 2Ø I*2-Elementen getrennt für die jeweilige Chiprichtung. Eine wiederholte Rasterbelegung ist auch möglich. Die maximale Anzahl der Rasterabstände in beiden Chiprichtungen ist 2ØØØ (1ØØ Records jeweils 4Ø Byte). Auf diese Weise werden die beiden Rasterfiles DDX: DX.VEK und DDY: DY.VEK erstellt. Eingabeabschluß in der jeweiligen Chiprichtung erfolgt durch Angabe von Ø als Abstandsmaß. Eine Korrektut der Eingabe ist nicht vorgesehen. Dafür muß man die alten Rasterfiles löschen und die Task neu starten.

5.2. Konvertierung

Das Konvertierungsprogramm SYLAY besteht aus 19 Quellemodulen. Die jeweiligen Objektmodule werden in einer Library zusammengefaßt.

Der Verlauf des Konvertierungsalgorithmus (s. Punkt 4) wird durch das Hauptprogramm SYLAY gestartet. Die übrigen 18 Unterprogramme stellen einzelne widerholbare Teilschritte dar.

59012 VV Freiberg Ag 337/82 III/15/4 2391/1 1182. 250 DT/A 7431

Hier werden einige wichtige Variable erläutert:

Dateien

SQWC, ARRAY - Direktzugriffsdateien für die symbolische Leitbahnebene (s. Anhang 1, Anhang 4; Abb. 5)

KON, WIND - Direktzugriffsdateien für die symbolische Kontaktfensterebene (s. Anhang 1, Anhang 4; Abb. 5)

DDX, DDY - Rasterfiles (s. Punkt 5.1.)

DSØ - Direktzugriffsfile für den physikalischen Layout im DS11-Format

Vektoren

Für die Verarbeitung des symbolischen Layouts werden folgende Vektoren definiert:

CHARACTER

WIN(3,4Ø) - Ein laufendes Fenster für die Nachbarschaftsuntersuchung der symbolischen Layoutstruktur

VEK(4Ø) - Ein Zeilenvektor für den direkten Zugriff auf einen beliebigen Record der Leitbahn- oder Kontaktfensterdatei

BUF(512) - Ein Puffer für die Arbeit mit DSØ

INTEGER*2

SCPUF(256) - Puffer für die Arbeit mit DSØ (SCPUF und BUF adressieren einen gemeinsamen Speicherbereich)

VE(2Ø) - Zeilenvektor für den direkten Zugriff auf einen beliebigen Record der Leitbahn- und Kontaktfensterdatei (VE und VEK adressieren einen gemeinsamen Speicherbereich)

ST1(2ØØ),ST2(2ØØ) - Vektoren für die Abspeicherung der Eckpunktkoordinaten des jeweiligen Linienzuggraphen im physikalischen Layout

59012 VV Freiberg Ag 307/82 III/15/4 2391/1 1182 · 250,0 T/A 7434

PR(3),AC(3),WK(3) - Vektoren für die Koordinaten des vorhergehenden, aktuellen und des nächsten Symbols beim Durchlauf des jeweiligen symbolischen Graphen (s. Anhang 1)

TOP(3) - Vektor für die Anfangskoordinaten des jeweiligen symbolischen Graphen

DX(2ØØØ),DY(2ØØØ) - Vektoren für die Rasterabstände in beiden Chiprichtungen des physikalischen Layouts. Sie werden beim Lesen der Dateien DDX und DDY gebildet.

einfache Variablen

LOGICAL

HIN - wird entsprechend der Laufrichtung des jeweiligen Graphenzweiges gesetzt (s. Punkt 4)

OW - bezeichnet das Vorhandensein eines Kontaktfensters

CHARACTER

PRE, ACT, NXT - Variablen für das vorhergehende, das aktuelle und das nächste Symbol beim Durchlauf des jeweiligen symbolischen Graphen

MER - Variable zum Ändern eines bestimmten Elementes des sybolischen Layouts (MER = '␣' oder MER = '$')

ZKF - Symbol für Kontaktfenster

LAGE - Variable zum Unterscheiden der jeweiligen Symbollage

INTEGER*2

II - aktueller Index von SCPUF
Q - aktueller Index von ST1 und ST2
PGN - Codezahl POLYGON
SOG - Codezahl START OF GROUP
EOG - Codezahl END OF GROUP
ANZ - Nachbaranzahl
A,B,C - aktuelle Koordinaten im symbolischen Layout (s. Anhang 1, Anhang 4; Abb. 5)

59012 VV Freiberg Ag 337/32 III/15/4 2351/1 1182 250 T/A 7431 +

X,Y - aktuelle Koordinaten im physikalischen Layout
N - Zeilenzahl
M - Spaltenzahl
LW - Leitzugbrreite
KW - Kontaktfensterbreite

Und hier folgt die kurze Beschreibung der einzelnen Programmodule:

KNVRT

Die Routine KNVRT transformiert die symbolischen Layoutkoordinaten (s. Anhang 1) in die physikalischen Layoutkoordinaten mittels der Files DDX und DDY.

PX, PY, MX, MY

Diese 4 Routinen aktualisieren die laufende Layoutkoordinate getrennt für die jeweilige Chiprichtung.

ACTU

Dieses Unterprogramm aktualisiert die physikalischen Layoutkoordinaten beim Durchlauf des jeweiligen Graphenzweiges. ACTU ruft die Subroutinen PX, PY, MX und MY auf.

RDVEK

Einlesen eines bestimmten Records des symbolischen Layouts in den Vektor VEK.

RDWIN

Einlesen eines Fensterabschnitts von dem symbolischen Layout in WIN (3,4Ø).

OPLI

Diese Routine liest einen bestimmten Record in den Vektor VEK ein, verändert dort ein vorgegebenes Zeichen und schreibt den Vektorinhalt wieder an die jeweilige Stelle in das symbolische Layoutfile.

TRNSF

Löschen des aktuellen Sybols, wenn HIN=.FALSE. (Rücklauf des Zweiges). Transformation der symbolischen Layoutkoordinaten beim Übergang zum nächsten Symbol (s. Anhang 1).

PR(i) := AC(i)
AC(i) := NX(i)
ACT := NXT

Dieses Programm ruft die Subroutine OPLI auf.

TS1

Testen der vertikalen Zeichenlage und bedingte Bestimmung der Koordinaten des nächsten Symbols mit anschließendem Erhöhen der Nachbaranzahl.

INCI

Erhöhen des aktuellen Index II des Puffers SCPUF, der mit den Polygonziffern gefüllt wird (s. Anhang 3). Wenn der Puffer voll ist, wird sein Inhalt in die DS11-Datei eingeschrieben und der Index II gleich 1 gesetzt.

POLY

Diese Routine kodiert ein Polygon in DS11-Format (s. Anhang 3). Die Eckpunktkoordinaten des Polygonzuges befinden sich in ST1 und ST2. POLY ruft die Subroutine INCI auf.

CLCYC

Suchen des nächstfolgenden Liniensymbols an einem Zweig entgegen dem Uhrzeigersinn und Bestimmen seiner Koordinaten. Dieses Programm ruft die Subroutine RDVEK auf.

CALANZ

Berechnen der Nachbaranzahl. CALANZ ruft TS1 und RDVEK auf.

NXTGER

Dieses Programm generiert die Koordinaten des nächsten Symbols beim Durchlauf einer Zeichenstrecke. Anschließend wird das Symbol eingelesen. Die Routine ruft RDVEK auf.

Die Unterprogramme GERADE und KNICK speichern die Eckpunktkoordinaten eines Polygonzuges im physikalischen Layout in die Vektoren ST1 und ST2 ab.

59612 VV Freiberg Ag 307/82 III/15/4 2391/1 1182 250,0 T/A 7434

GERADE

Dieses Unterprogramm befaßt sich mit dem Fall Symbol innerhalb einer Zeichenstrecke ('-', '/', '\' oder '!'). Zunächst werden die physikalischen Layoutkoordinaten aktualisiert (Aufruf von ACTU). Dann erfolgt die Abfrage, ob an derselben Stelle in der Kontaktfensterebene ein Kontaktfenster liegt (Unterprogramm RDWIN, aktuelles Symbol '-' oder '!'). Wenn ja, dann werden die jeweiligen Eckpunktkoordinaten des Polygonzuges in ST1 und ST2 abgespeichert. Anschließend werden die Unterpgogramme NXTGER und TRNSF aufgerufen.

KNICK

Dieses Programm verarbeitet den Fall Knick in dem symbolischen Graph ('+'-Zeichen). Je nach der Lage der Nachbarsymbole wird der jeweilige Knick unterschieden. Dann erfolgt die Abfrage nach Kontaktfenster. Die entsprechenden Eckpunktkoordinaten des Polygonzuges werden in ST1 und ST2 abgespeichert. Anschließend wird das Unterprogramm TRNSF aufgerufen. Andere Unterprogrammodule sind PX, PY, MX, PY, RDWIN und GERADE (für den T-förmigen Knick).

SYLAY (.MAIN!)

Das Hauptprogramm SYLAY beginnt mit der Initialisierung der verwendeten Dateien, Vektoren und einfachen Variablen. Dann wird die Namenangabe der zu verarbeitenden symbolischen Layoutdatei abgefragt, die vom Anwender eingegeben wird. (Die Kontaktfensterdatei hat denselben Namen). Als nächstes folgt die Abfrage nach Leitbahn- und Kontaktfensterbreite, die auch angegeben werden müssen. Dabei ist zu beachten, daß die Breiten ganze Zahlen sein müssen. Diese Zahlen werden (in dieser Version) in geraden Zahlen abgerundet, so daß eine symmetrische Layoutstruktur als Ergebnis entsteht (s. Anhang 4, Abb. 6). Weitere Erläuterungen sind im Punkt 6: "Entwicklungsmöglichkeiten" zu finden.

Anschließend erfolgt das Einlesen der Dateien DDX und DDY in die Vektoren DX und DY.

Erst dann folgt die Programmausführung nach dem Konvertierungsalgorithmus im Punkt 4.

6. Entwicklungsmöglichkeiten

Mit dem Abschluß dieser Arbeit sind die Probleme der Konvertierung einer symbolischen Layoutdatei in eine physikalische Layoutdatei nicht umfassend gelöst.

Eine Besonderheit des vorliegenden Programms SYLAY ist, daß die Leitzug- und Kontaktfensterbreiten gerade Zahlen sind. Die Ursachen dafür war die zunächst gestellte Forderung nach Symmetrie der CMOS-Layoutstruktur (s. Anhang 4, Abb. 6). Das erwies sich aber als nicht richtig bei der Herstellung der Leitbahnmasken im Pattern Generator. Ein weiterer Entwicklungsschritt ist also die Berücksichtigung der ungeradzahligen Leitzug- und Kontaktfensterbreiten, die zu einer unsymmetrischen Anordnung in der physikalischen Layoutebene führen. Dann entsteht das Problem der Veränderung des Konvertierungsalgorithmus' und der Erweiterung der Knickpunktvarianten. Was ist damit gemeint? Wenn z. B. die Leitzugbreite gleich 5 /um ist, werden nach dem Algorithmus im Punkt 4 beim Hinlauf längs eines Linienzweiges in der physikalischen Layoutebene 2 /um und beim Rücklauf —3 /um Linienzugbreite durchgezogen oder umgekehrt (3 /um beim Hinlauf und 2 /um beim Rücklauf). Außerdem müssen alle Knickpunktvarianten mit dazugehörigen Eckpunkten in Hin- und Rücklaufrichtung bei den vier möglichen Kombinationen zwischen Leitzug- und Kontaktfensterbreiten (gerade und ungerade Zahlen) extra abgespeichert werden.

Zur Zeit sind 32 Knickpunktvaianten (ausgenommen die Spitzen Winkel als nicht erlaubt) abgespeichert. Mit der Berücksichtigung der oben erwähnten Variationen muß man noch weitere 192 Knickpunktfälle hinzuaddieren. Damit wird die Speicherplatzanforderung deutlich erhoben. Außerdem muß man bei der Ausführung des Programms alle diese Fälle unterscheiden, was einen sehr großen Aufwand bedeutet.

Eine weitere Entwicklungsmöglichkeit, die eine Stufe höher liegt, ist die Grundidee der in /20/ dargelegten automatischen Traceierung mit veränderlicher Leitzugbreite.

Die Vorteile dieser Technik sind folgende:

1. Die Makrozellenkonturen im Master sind nicht durch das Vorhandensein eines Rasters begrenzt.

2. Die Leitzugbreite muß nicht ein Vielfaches des Rasterabstandes sein.
3. Das interaktive Layout läßt sich ohne weitere Modifikationen mit dem automatischen Layout kombinieren.
4. Die Programmausgangsdateien können direkt in dem Pattern-Generator eingegeben werden.

Die Technik der veränderlichen Leitzugbreite läßt sich sowohl bei der von uns angewendeten Rastertrassierung (grid routing) als auch bei den anderen 2 bekannten Trassierungsmethoden: Line Routing und Line Packing anwenden. Dafür muß man aber prinzipiell neue Algorithmen entwickelen, die direkt das physikalische Layout erstellen, ohne daß zunächst ein symbolisches Layout aufgebaut wird. Es müssen lediglich die Makrozellenkontakte mit einer Verbindungstabelle angegeben werden /21/, /22/. Mit dieser Ausgangsinformation können Algorithmen und Layout Systeme wie in /7/, /8/ und /10/ das physikalische Layout erstellen.

Es werden verschiedene Kanalmodelle (Channel Models) mit speziellen Makrozellenanordnungen (/12/, /13/, /27/) und hierarchische Mehrschritt-Trassierungsmethoden angeboten. Die Auswahl ist groß. Aber nicht alle Techniken lassen sich für VLSI anwenden. Es gibt solche, die nur für Leiterplatten, Hybrid- und LSI-Schaltkreise geeignet sind.

Wollen wir allerdings bei der Konvertierungstechnik der Rastertrassierung bleiben, dann existiert auch die Möglichkeit, den Symbolgraphen auf andere Weise zu behandeln. Sie besteht darin, daß zunächst der Symbolgraph in mehreren unverzweigten Teilgraphen zerlegt wird, wobei die ursprünglichen Zweigpunkte als Anfänge dieser Graphen betrachtet werden. Dann wird jede dieser Strecken durchgegangen und der entsprechende Polygonzug aufgestellt. Das kann auf 2 Arten durchgeführt werden: entweder in Hin- und Rücklaufrichtung (wie nach dem bereits erwähnten Algorithmus) oder nur in Hinlaufrichtung, wobei die Eckpunkte an der Knick- und Kontaktstelle von beiden Seiten des Linienzuges abgespeichert werden. Dann müssen die einzelnen Polygonzüge an den ursprünglichen Zweigstellen mit oder ohne Überlappung verbunden werden. Dabei muß berücksichtigt werden, daß die endgültigen Polygoneckpunkte nur einmal auftreten.

Es ist zu erwähnen, daß auch diese Konvertierungstechnik keine einfache Lösung des Problems der ungeradzahligen Leitzug- und Kontaktfensterbreiten bietet.

7. Zusammenfassung

Die vorliegende Diplomarbeit bietet eine Algorithmenkonzeption zur Konvertierung der Daten des symbolischen Layouts eines CMOS Gate-Array-Schaltkreises in das Format des physikalischen Layouts. Es wurde ein Programm in der Sprache FORTRAN-77 für einen Rechner der KULON-Serie entwickelt. Die Konvertierung des symbolischen Layouts wird unter Einsatz eines modifizierten Left-Edge-Algorithmus durchgeführt. Der Zeichengraph wird in eine spezifische Datenstruktur - dem sogenannten DS11-Format - umgewandelt. Es wurden 2 Tasks erstellt - RASTER (etwa 26K) zur Eingabe der Rasterabstände des physikalischen Layouts und SYLAY (etwa 54 K) für die eigentliche Konvertierung, die nacheinander gestartet werden müssen. Ausgangsdateien sind die Datenfiles der symbolischen Leitbahn- und Kontaktfensterebenen. Das Konvertierungsprogramm SYLAY besteht aus 19 Quellmodulen. Die jeweiligen Objektmodule werden in einer Library zusammengefaßt. Eine Einschränkung des vorgelegten Programms ist, daß die anzugebenden Leitzug- und Kontaktfensterbreiten im physikalischen Layout gerade Zahlen sein müssen. Weitere Entwicklungsmöglichkeiten sind gegeben.

Literaturverzeichnis

/1/ Design Automation of Digital Systems/M. A. Breuer. - Vol. 1 (1972). - Englewood Cliffs, NJ::Prentice Hall

/2/ Theory and Design of Switching Circuits/A. Friedman. - (1975). - Computer Science Press

/3/ Placement and Routing Program for Masterslice LSI's/R. Kamikawa, K.Kishida, A. Osawa, I. Yasuda, T. Chiba. - In: Proc. 13th DA Conf. - (1976). - pp. 245 - 250

/4/ A Minicomputer - Based System for Automatic LSI Layout/G. Persky, D. Deutsch, D. G. Schweikert. - In: I. Des. Automat. Fault-Tolerant Comput. - Vol. 1 (1977). - pp. 217 - 256

/5/ The Chip Layout Problem: An Automatic Wiring Procedure/ K. A. Chen, M. Feuer, K. Khokhani, N. Nan, S. Schmidt. - In: Proc. 14th DA Conf. - (June 1977). - pp. 298 - 302

/6/ The Chip Layout Problem: A Placement Procedure for LSI/ K. Khokhani, A. M. Patel. - In: Proc. 14th DA Conf. - (1977). - pp. 291 - 297

/7/ An Automated Layout System for Master-Slice LSI: MARC/ K. Ueda, Y. Sugiyama, K. Wada. - In: IEEE J. Solid-State Circuits. - Vol. SC-13 (1978). - pp. 716 - 721

/8/ Automatic Layout Algorithmus for Masterslice LSI/ H. Yoshizawa, H. Kawanishi, S. Goto, A. Kishimoto, Y. Fujinami, K. Kani. - In: Proc. IEEE ISCAS. - (1979). - pp. 470 - 473

/9/ Efficient Placement and Routing Technique for Masterslice LSI/ H. Shirashi, F. Hirose. - In: Proc. 17th DA Conf. - (1980). - pp. 470 - 473

/10/ CGAL - A Multi Technology Gate Array Layout System/ L. F. Todd et cal. In: Proc. 19th DA Conf. - (June 1982). - pp. 792 - 801

/11/ Global Router/J. Soukup. - In: Proc. 16th DA Conf. - (1979). - pp. 481 - 484

/12/ A Greedy Channel Router/R. L. Rivest, C. M. Fiduccia. - In: Proc. 19th DA Conf. - (1982). - pp. 418 - 424

/13/ Efficient Algorithmus for Channel Routing/T. Yoshimura, E. S. Kuh. - In: IEEE Trans. on Computer-Aided Design. - Vol. CAD-1 (1981). - pp. 25 - 35

/14/ An Efficient Variable Cost Maze Router/R. K. Korn. - In: Proc. 19th DA Conf. - (June 1982). - pp. 425 - 431

/15/ Hierarchical Wire Routing/M. Burstein, R. Pelavin. - In: IEEE Trans. on Computer-Aided Design. - Vol. CAD-2 (1983). - pp. 223 - 234

/16/ Routing Techniques for Gate Array/B. Ting, B. Tien. - In: IEEE Trans. on Computer-Aided Design. - Vol. CAD-2 (Oct. 1983). - pp. 301 - 312

/17/ A New Global Router for Gate Array LSI/ S. Tsukiyama, I. Hirada, M. Fukai, I. Shirakawa. - In: IEEE Trans. on Computer-Aided Design. - Vol. CAD-2 (Oct. 1983). - pp. 313 - 321

/18/ A New Symbolic Channel Router: YACR 2/J. Reed, A. Sangrovanni, M. Santomauro. - In: IEEE Trans. on Computer-Aided Design. - Vol. CAD-4 (Juli 1985). - pp. 208 - 219

/19/ Hierarchical VLSI-Routing: An Approximate Routing Procedure/ A. Patel, N. Soong, R. Korn. - In: IEEE Trans. on Computer-Aided Design. - Vol. CAD-4 (April 1985). - pp. 121 - 126

/20/ Automatic Variable-Width Routing for VLSI/H.-J. Rothermel, D. Mlynski. - In: IEEE Trans. on Computer-Aided Design. - Vol. CAD-2 (Oct. 1983). - pp. 271 - 284

/21/ A Method of Improving the Terminal Assignment in the Channel Routing for Gate Arrays/M. Teray. - In: IEEE Trans. on Computer-Aided Design. - Vol. CAD-4 (Juli 1985). - pp. 329-336

/22/ The Shortest Path Trough a Maze/E. F. Moore. - In: Bell Syst. Mono.- (1959). - no. 3523

/23/ An Algorithm for Path Connections and Its Applications/ C.Y. Lee. - In: IRE Trans. Electron. Comput.-Vol. EC-10

(Sept. 1961). - pp. 346 - 365

/24/ A Solution to the Line-Routing Problems on the Continuous Plane /D. W. Hightower. - In: Proc. 6th DA Workshop. - (1969). - pp. 1 - 24

/25/ Grundprobleme der Mikroelektronik/Wolfgang Hilberg.-(1982). - R. Oldenbourg Verlag München, Wien

/26/ Wire Routing Optimizing Channel Assignment/A. Hashimoto, J. Stevens. - In: Proc. 8th DA Workshop. - (1971), pp. 155 - 169

Anhang 1 Symbolisches Layout (Auszug)

Das symbolische Layout wird in vier Schritten erzeugt: Grundzellenentwurf, Masterentwurf, Kontaktfensterbelegung und Leitbahnführung. Ergebnisse der letzten zwei Schritte sind die Kontaktfenster- und Leitbahnebenen, welche die Ausgangsdateien für die Aufstellung eines physikalischen Gate Array Layouts darstellen. Sie sind als unformatierte Direktzugriffsfiles mit der Recordlänge 40 Byte organisiert. Die Zeichenbelegung sieht wie folgt aus:

1. Record

Byte	Belegung	Standard
1 - 2	Zeilenzahl m	I*2
3 - 4	Spaltenzahl n	I*2
5 - 10	Zeichen	
5	vertikal/Fenster	|/W
6	horizontal	-
7	steigend	/
8	fallend	\
9	Knickpunkt	+
10	Endpunkt	$
11	Kontaktfensterebene	.FALSE.
12 - 20	Kursorbewegung	
21 - 28	Schreibmodebedienung	
29 - 40	frei	

Die Bytes 12 - 40 werden bei der Konvertierung nicht benutzt. Sie spielen lediglich bei der Eingabe und Korrektur der Eingabedateien eine Rolle.

2. - 5. Record : frei

Die Anzahl der Recordspalten j wird durch die Formel j = (Spaltenzahl - 1)/40 + 1 berechnet.

6. - (j*m+5).Record

Die Belegung des Masters, der Kontaktfensterebene, der Leitbahnebene erfolgt in einer Zeichenmatrix wie folgt:

	1 2 3 ... 39 40	..K..	(j-1)*40+1...n-1 n...40 j
1	6. Record	.....	(j-1)*m + 6. Record
2	7. Record	.....	(j-1)*m + 7. Record
.	"		"
.	"		"
l	l+5. Record	.....	(j-1)*m + l. Record
.	"		"
.	"		"
m	m+5. Record	.....	j*m + 5. Record

Diese Zeichenmatrix.kann auch als eine Recordmatrix RM betrachtet werden (s. Anhang 4, Abb. 5a). Diese symbolische Recordmatrix wird im Schritt 1 des Konvertierungsalgorithmus in die Recordmatrix RM' überführt (s. Anhang 4, Abb. 5b). Die Lage jedes Zeichens wird eindeutig durch 3 Koordinaten bestimmt:

A - Y - Koordinate des Recors in RM': A = 1
B - X - Koordinate des Records in RM': B = INT((k-1)/4Ø + 1)
C - Lage des Zeichens im jeweiligen Record: C = K - (B-1)*4Ø/

Anhang 2 Rasterdateiorganisation

Ausgangsdateien für die Konvertierung eines symbolischen Layouts sind die beiden Kontaktfenster- und Leitbahnebenen (s. Anhang 1). Bevor das Konvertierungsprogramm SYLAY gestartet werden kann, muß die Eingabe der Zeichenabstände in beiden Chiprichtungen erfolgt sein. Dafür sorgt das Programm RASTER. Es werden zwei unformatierte Direktzugriffsdateien DDX und DDY mit der Recordlänge 4Ø Byte geschaffen. Ihre Eingabe erfolgt interaktiv recordweise in I*2-Vektoren von jeweils 2Ø Elementen. Die Zeichenabstände müssen positive ganze Zahlen ungleich Ø sein. Der Eingabeabschluß in der entsprechenden Chiprichtung erfolgt durch die Eingabe von Ø. Eine Duplizierung des Vektorinhaltes ist durch die Angabe von Recordnummern möglich. Es können maximal 1ØØ Records (d. h. 1ØØ Vektoren mit insgesamt 2ØØØ Elementen) eingegeben werden.

59012 VV Freiberg Ag 307/82 III/15/4 239'/1 1182 250,0 I/A7434

Anhang 3 Die Datenstruktur DS11 (Auszug)

Eine Layoutdatei besteht aus mehreren Gruppen, wobei jede Gruppe aus Elementen aufgebaut wird. Solche Elemente sind z. B. Gruppenaufrufe, Polygone, Text usw.

Ein Polygon ist ein geschlossener zweidimensionaler Polygonzug. Diesem Polygon ist ein Attribut zugeordnet, das die technologische Ebene beschreibt, der es im Layout zugeordnet ist.

Eine Gruppe fängt mit dem Gruppenkopf an, der globale Information enthält. Anschließend folgen die Datenblöcke mit der kontinuierlichen Darstellung der Elemente, die zur Gruppe gehören.

Die Gruppenaufrufe transformieren die jeweils aufgerufenen fertigen Gruppen an die vorgesehenen Stellen im Layout. Dadurch wird der IC-Aufbau topologisch hierarchisch beschrieben.

Bei der Aufstellung der konvertierten Layoutstruktur bestehen die Gruppen ausschließlich aus Polygonen. Gruppenaufrufe sind einfachheitshalber ausgeschlossen.

Die Beschreibung eines Elementes nach dem DS11-Format sieht folgendermaßen aus:

Wort 1 (16 Bit): - Identifizierung des Elements
- Elementausgabe
- Information über Datengenauigkeit

Bit 15: = Ø - Es treten nur INTEGERx2 Zahlen auf
= 1 - Es treten nur INTEGERx4 Zahlen auf

Bit 14 - 13: - unbelegt

Bit 12 - 8: - "36 = 1111Ø - Start of Group (SOG)
- "32 = 11Ø1Ø - End of Group (EOG)
- "4 = ØØ1ØØ - Polygon (PGN)

Bit 7 - Ø: - Für Bildelemente - Ausgabe der Layoutebene;
- Für SOG und EOG - ohne Bedeutung

Die Ebenenmarkierung kann von1 bis 255 laufen. Bildelemente mit der Ebenenbezeichnung = Ø sind Pseudoelemente und sie dienen einer zusätzlichen Segmentierung der DS-Datei.

Aus der vorgelegten Elementenbeschreibung folgt die für den betrachteten Fall ausgewählte Codierung (in Dezimalzahlen):

59612 VV Freiberg Ag 307/32 III/15/4 2391/1 1182 250,0 T/A 7434

SOG = 768Ø
PGN = 1Ø24 + Ebenennummer
EOG = 6656

Die Organisationselemente einer Leitbahnstruktur werden wie folgt beschrieben:

1. Start of Group (SOG):

Die Wörter 2 - 78 enthalten die globale Gruppeninformation:
Wort 2 - 37: Gruppenkommentar (72 Zeichen)
Wort 38 - 43: Name des Operators (12 Zeichen)
Wort 44 - 46: Name des Programms, welches die Datei zuletzt bearbeitet hat (6 Zeichen)
Wort 47 - 51: Datum des letzten Zugriffs zur Datei (10 Zeichen)
Wort 52: XMIN
Wort 53: XMAX
Wort 54: YMIN
Wort 55: YMAX
LAYOUTmaße

Die Wörter 56 - 78 werden für die Sortierung und für die Markierung der vorhandenen Ebenen in der Gruppe, der Meßeinheit, der Technologie und der Anzahl der Gruppenaufrufe benutzt.

Die übrigen Wörter stehen für beliebige Verwendung zur Verfügung.

2. End of Group

Das Element besteht nur aus einem Wort - die Elementbezeichnung. Es ist das letzte Element im letzten Satz einer Gruppe.

3. Polygon

Wort 2: x - Koordinate des Bezugspunktes
Wort 3: y - Koordinate des Bezugspunktes
Wort 4: Δx
Wort 5: Δy

usw. abwechselnd Δx und Δy. Die angegebenen Zahlen bestimmen paarweise die Strecken zwischen den einzelnen Knickpunkten.

Das Ende eines Polygons wird durch die Zahl 2**15-2 = 32768-2 markiert.

Eine Folge von Schrägen wird durch die Zahl 2**15-3 begonnen und durch 2**15-3 beendet, falls ein Δx folgt.

59012 VV Freiberg Ag 307/82 III/15/4 2391/1 1182. 250.0T/A7434

Wenn ein y folgt, wird die Schräge mit 2xxx15-4 abgeschlossen. Liegen Daten mit doppelter Genauigkeit vor, so sind die Zahlen entsprechend 2xxx31-2, 2xxx31-3 und 2xxx31-4.

59012 VV Eckelberg Ag 307/82 III/15/4 2321/1 1182 250,0 T/A 7434

Anhang 4

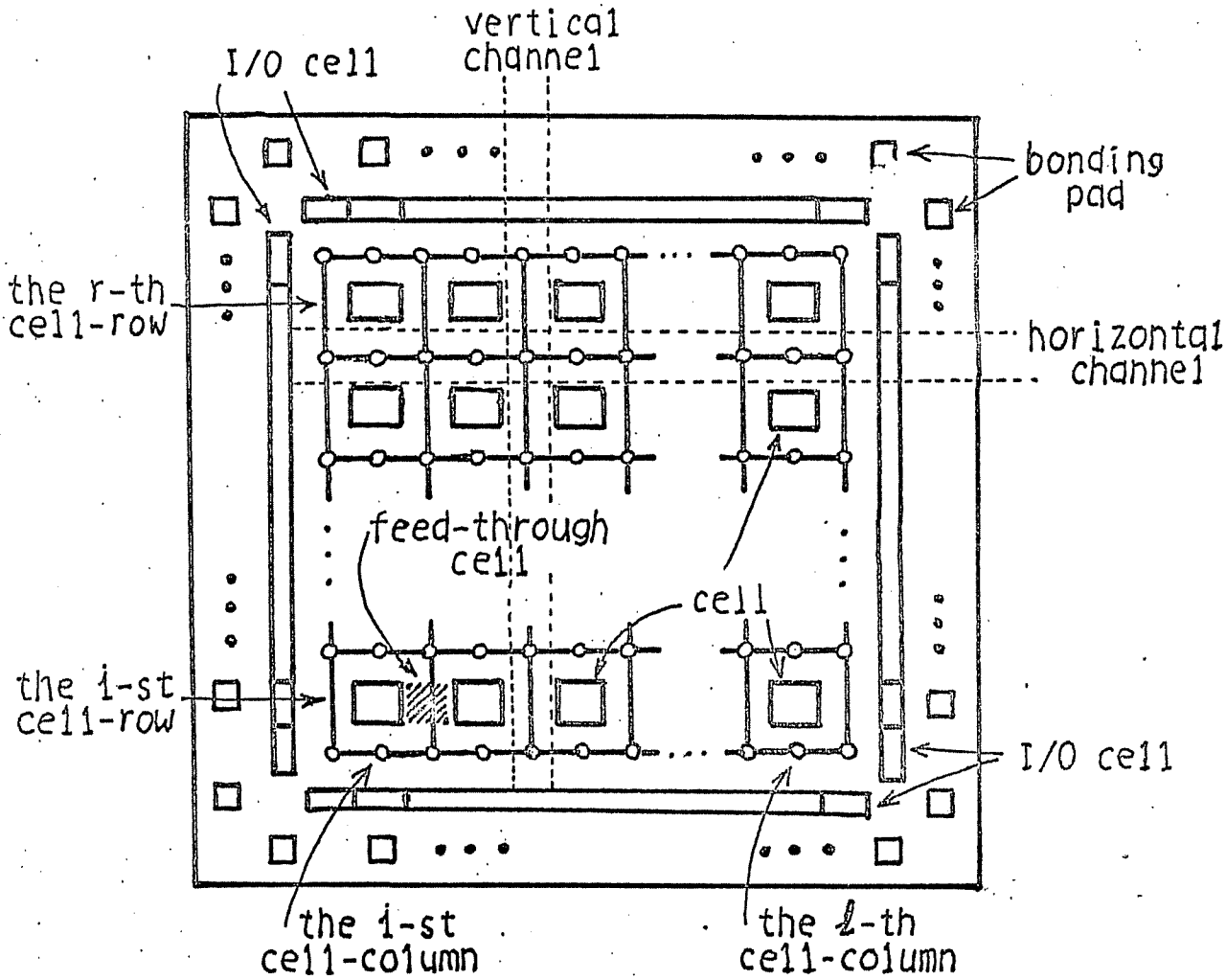


Abb.1 : Innere Struktur eines LSI-Gate-Array-Chips

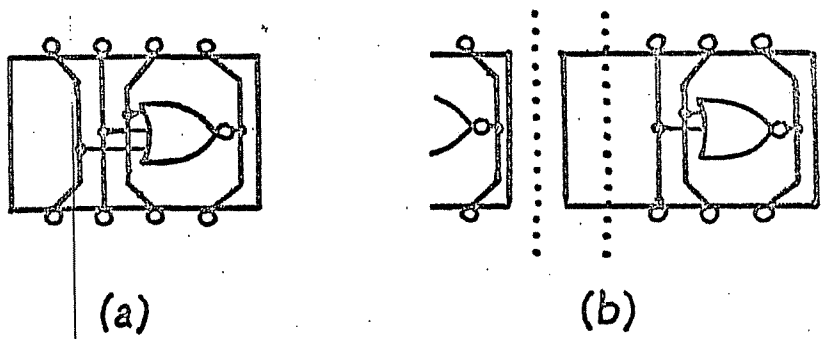


Abb. 2 : Zellen eines LSI-Gate-Array-Chips

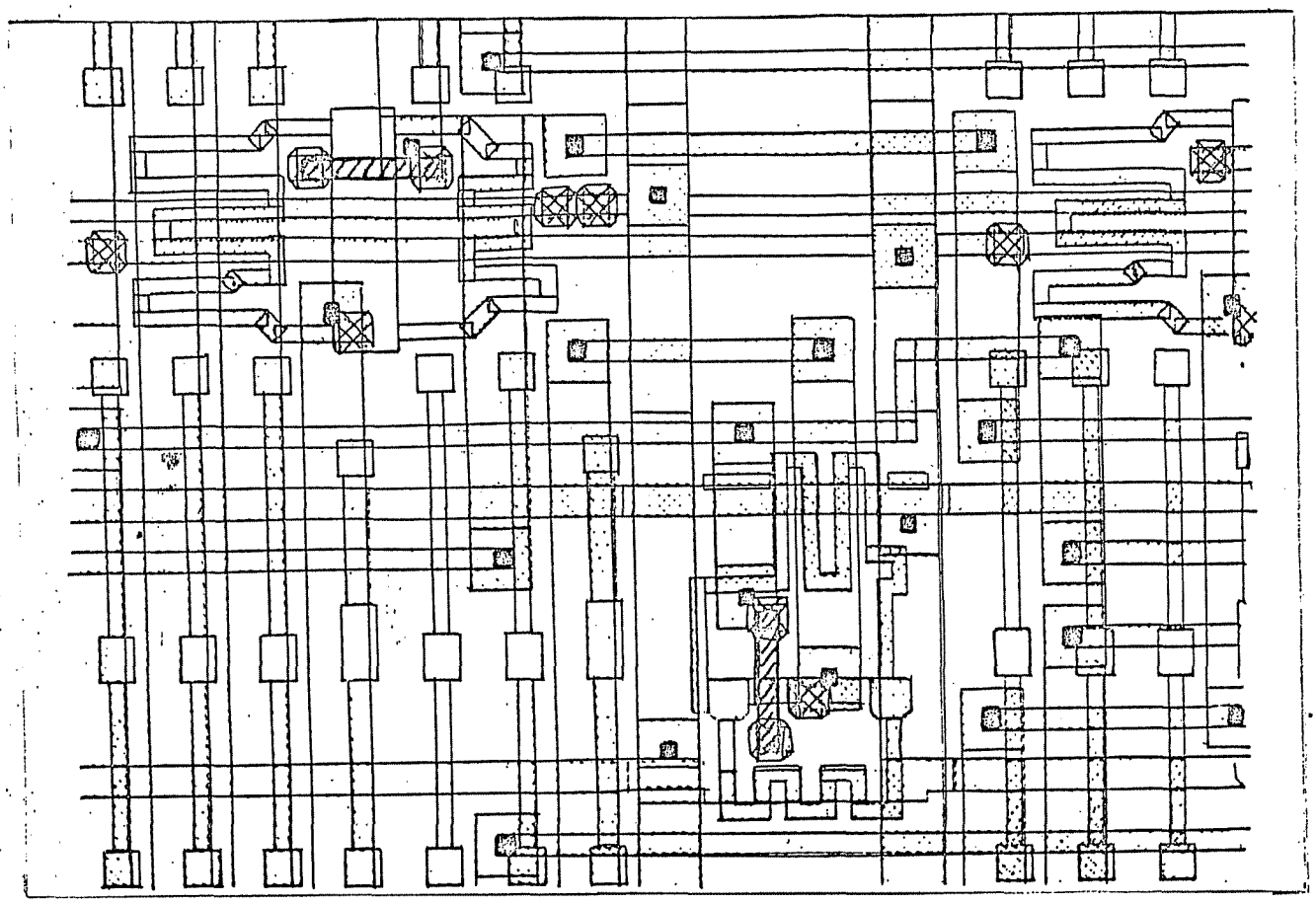


Abb. 3a. : Physikalisches Layout einer Gate-Array-Zelle

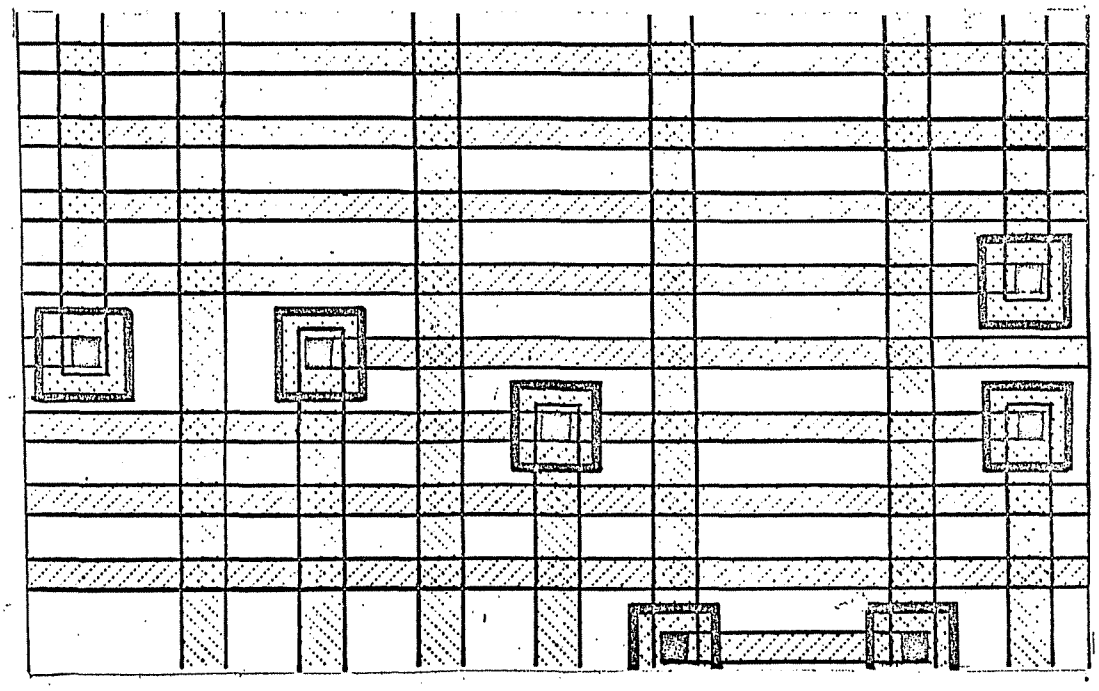
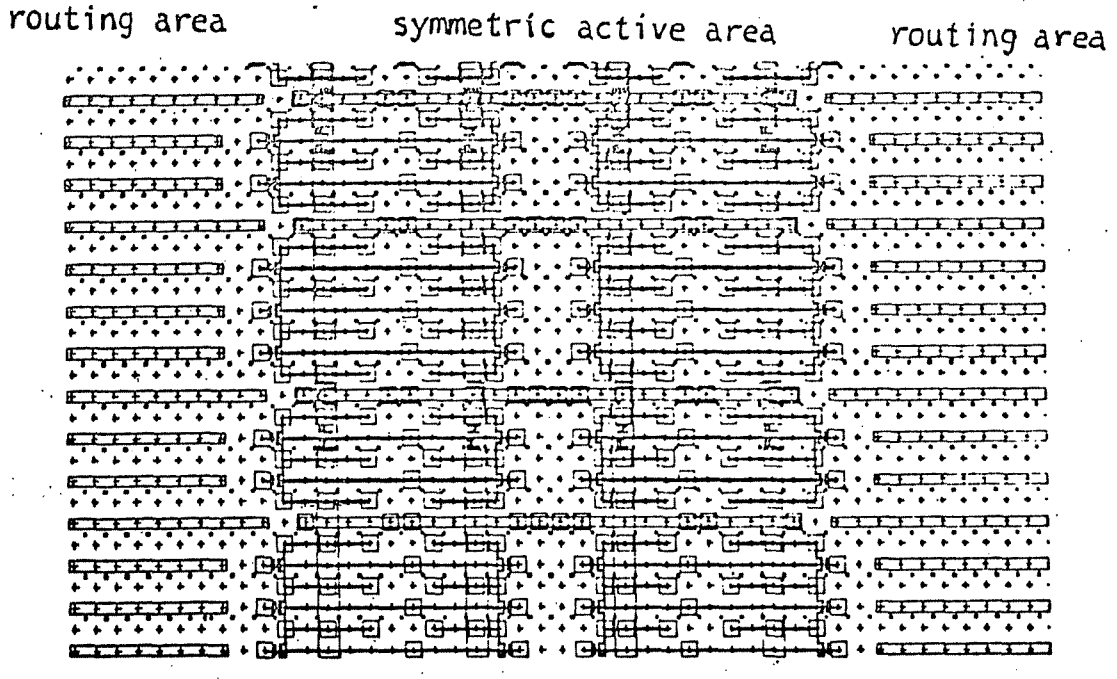
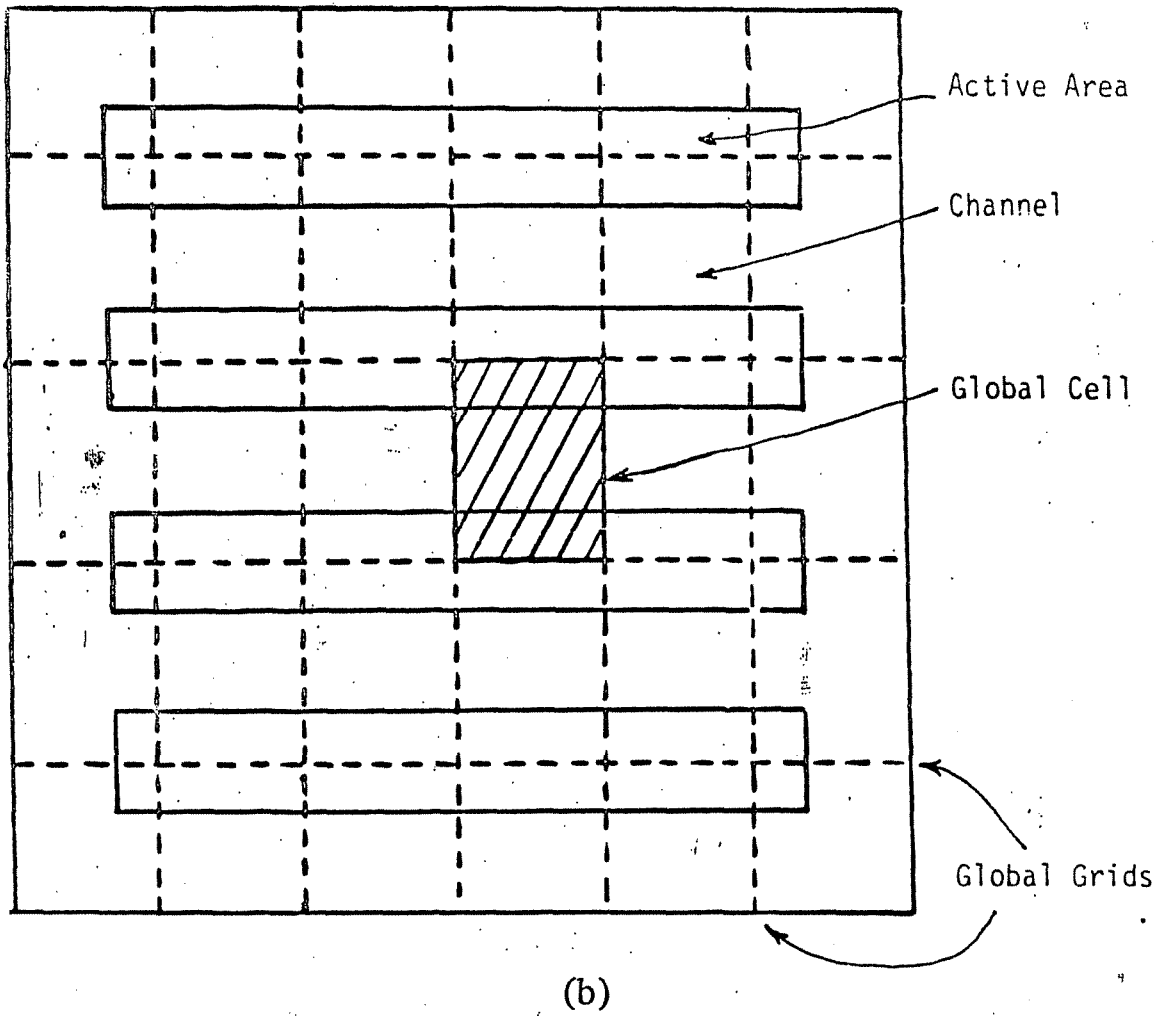


Abb. 3b. : Interne Kanalstruktur



(a)



(b)

Abb. 4. : a/ Symbolische Darstellung einer Gate-Array-Zelle
b/ Zerlegung eines Gate-Array-Chips in Globalzellen

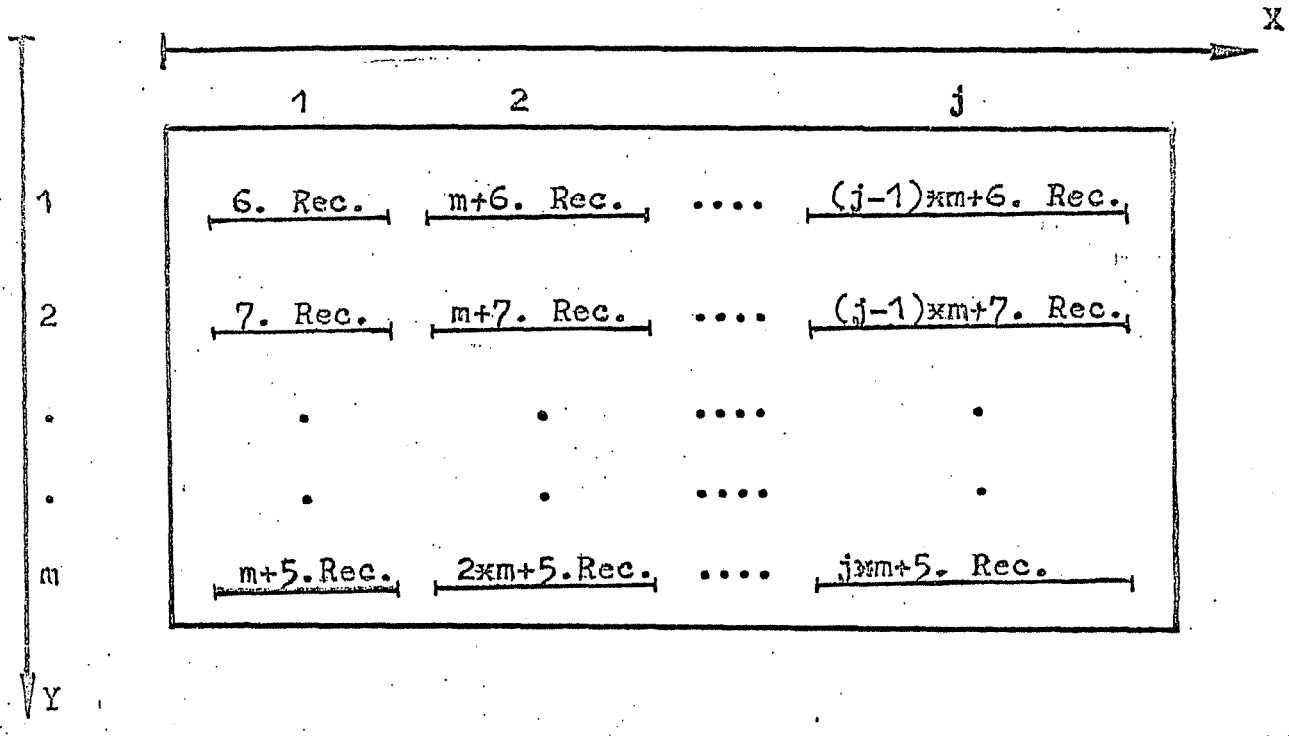


Abb. 5a.: Symbolische Recordmatrix RM (Zeilenzahl: m, Spaltenzahl: n, Anzahl der Recordspalten: j = (n-1)*4Ø+1)

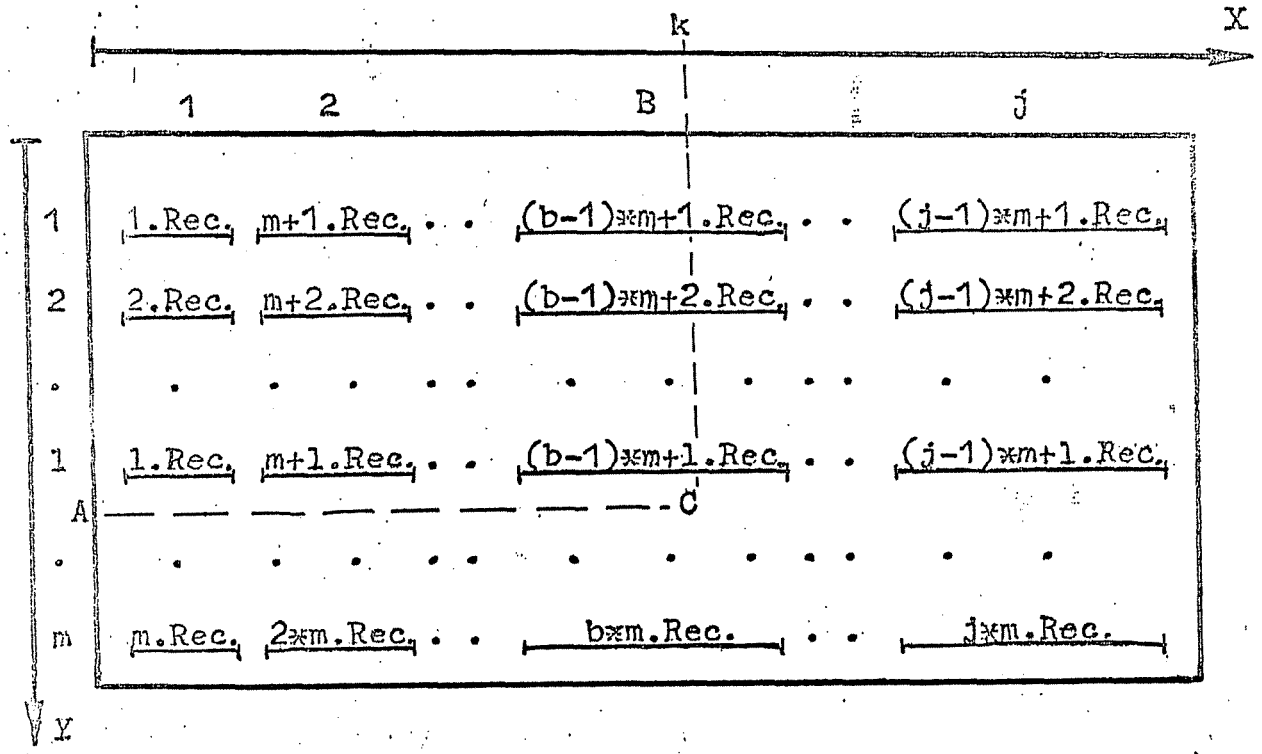
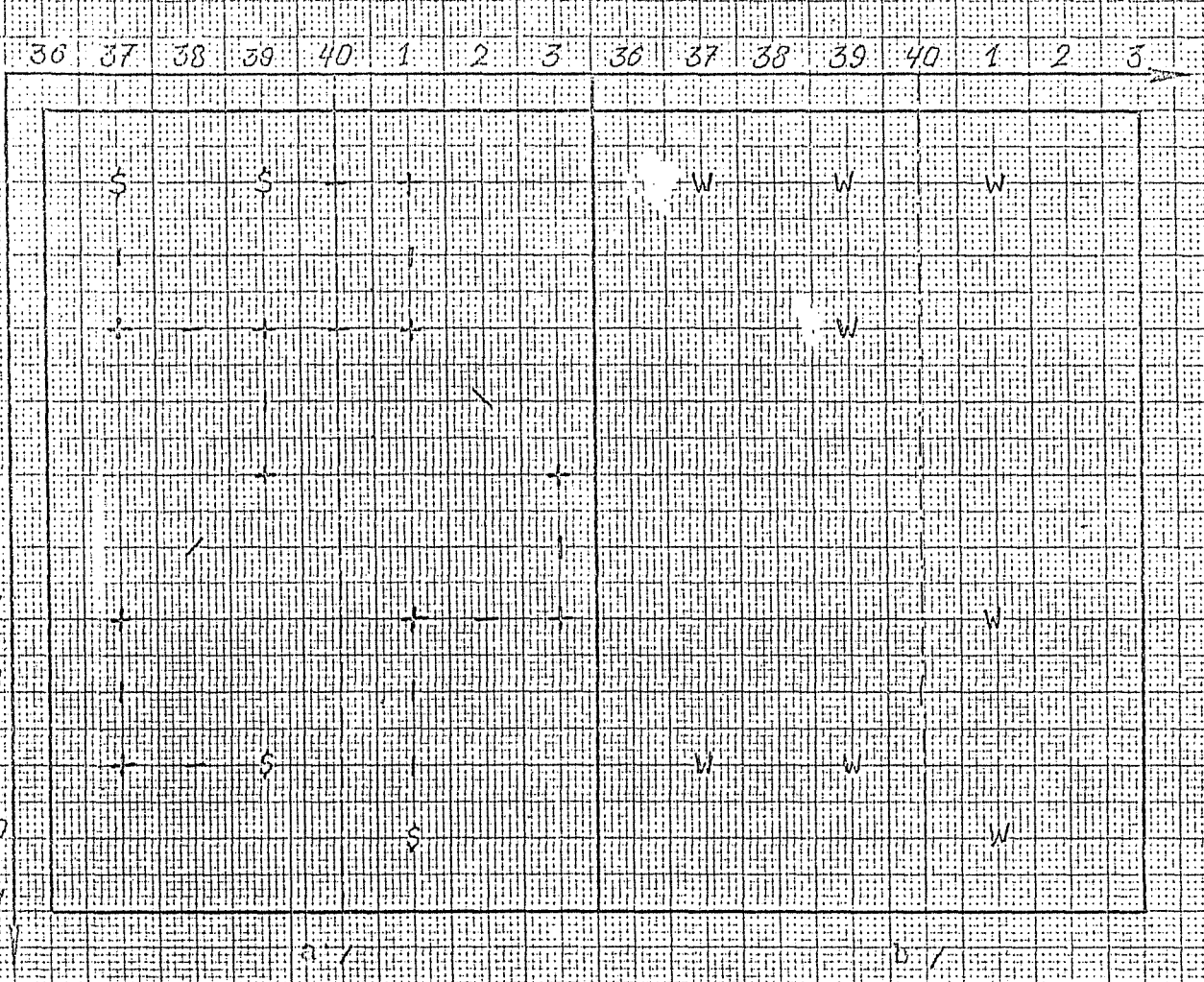


Abb. 5b.: Symbolische Recordmatrix RM'

Zeichenlage in der Matrix RM!: A = 1

B = (k-1)/4Ø + 1

C = k - (B-1)*4Ø



a / b /

Abb. 6.: Symbolische Leitbahn- /a/ und Kontaktlochebene /b/

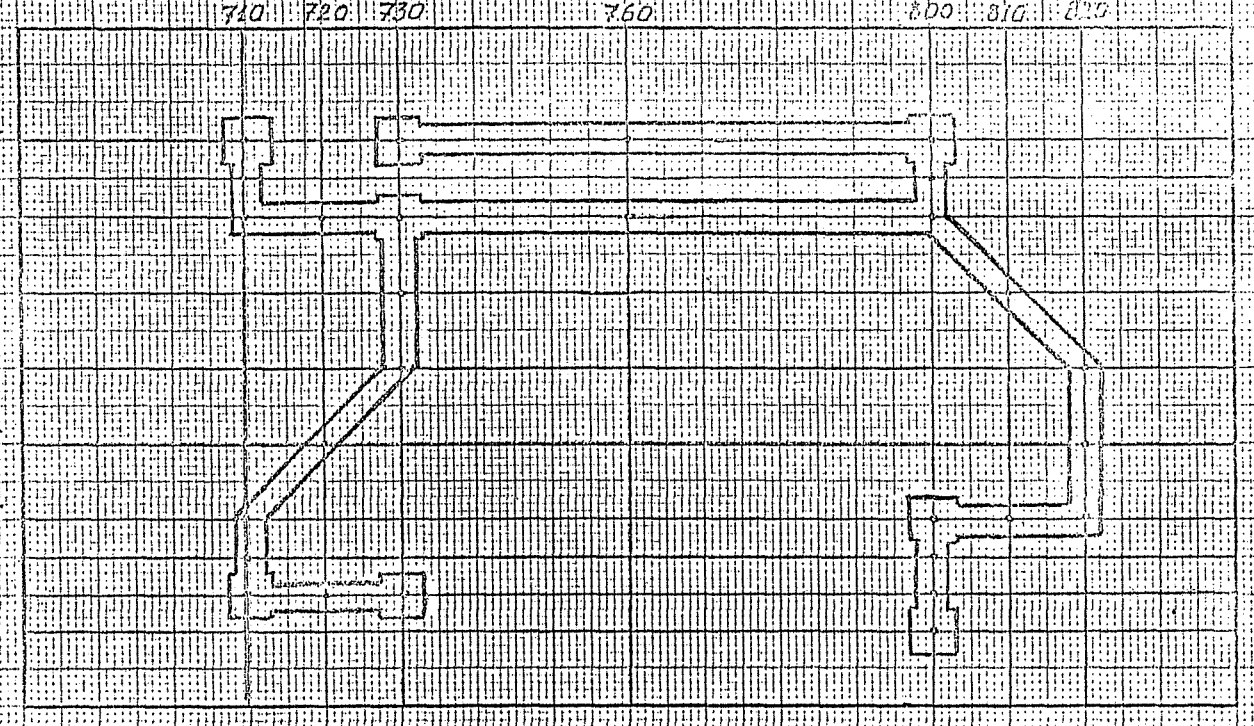


Abb. 7.: Typisches Layout aus Abb. 5.

Anhang 5

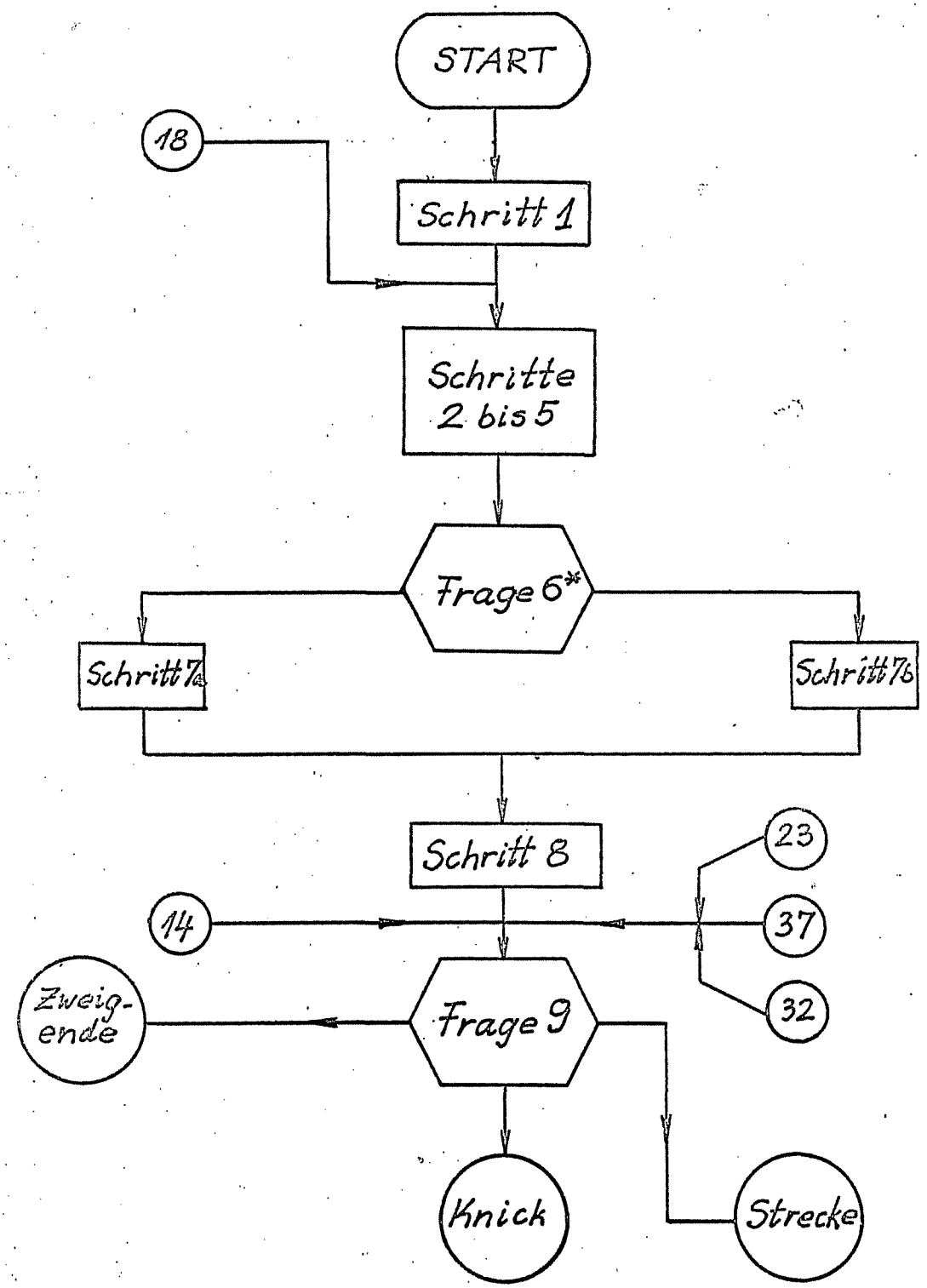
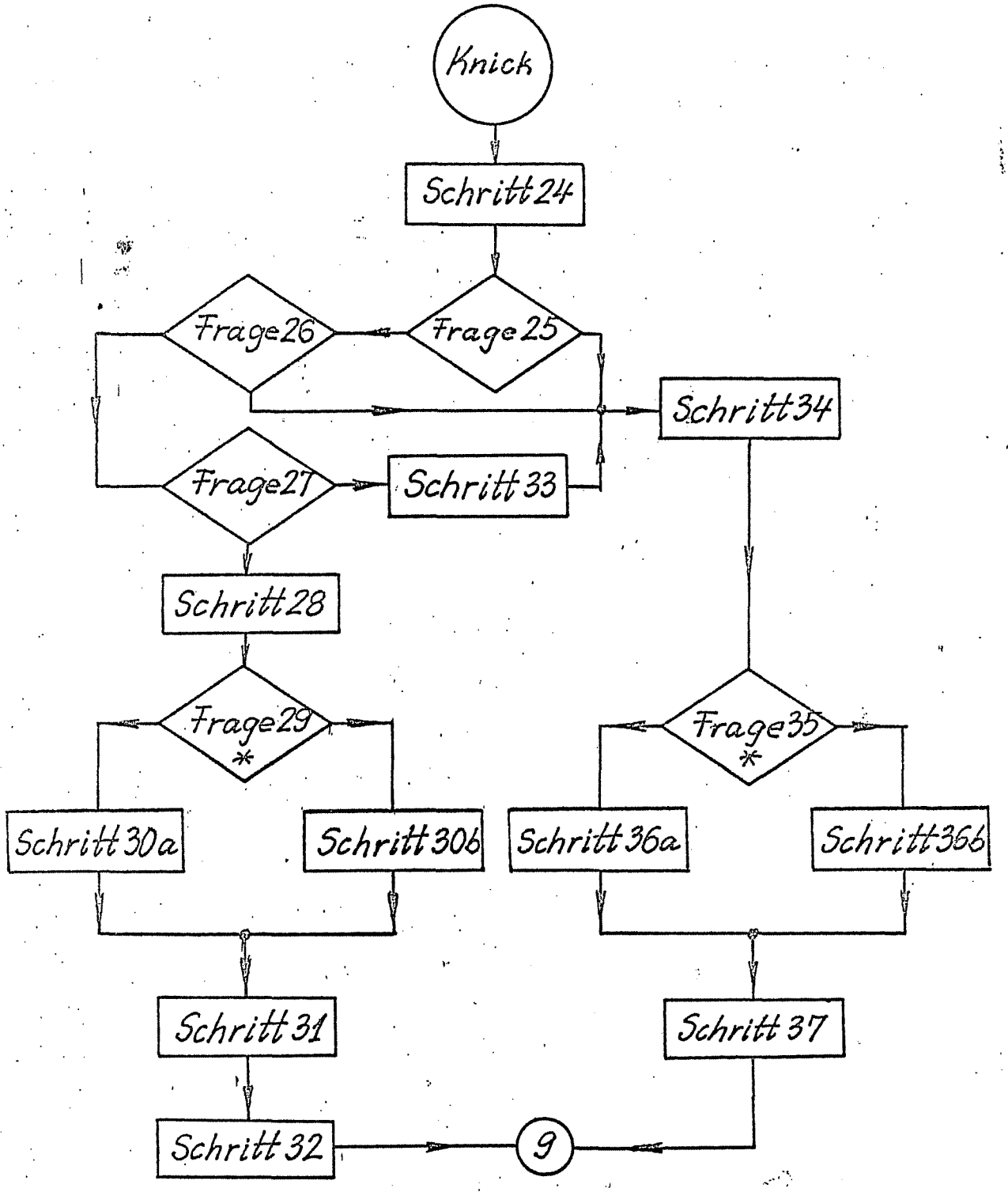
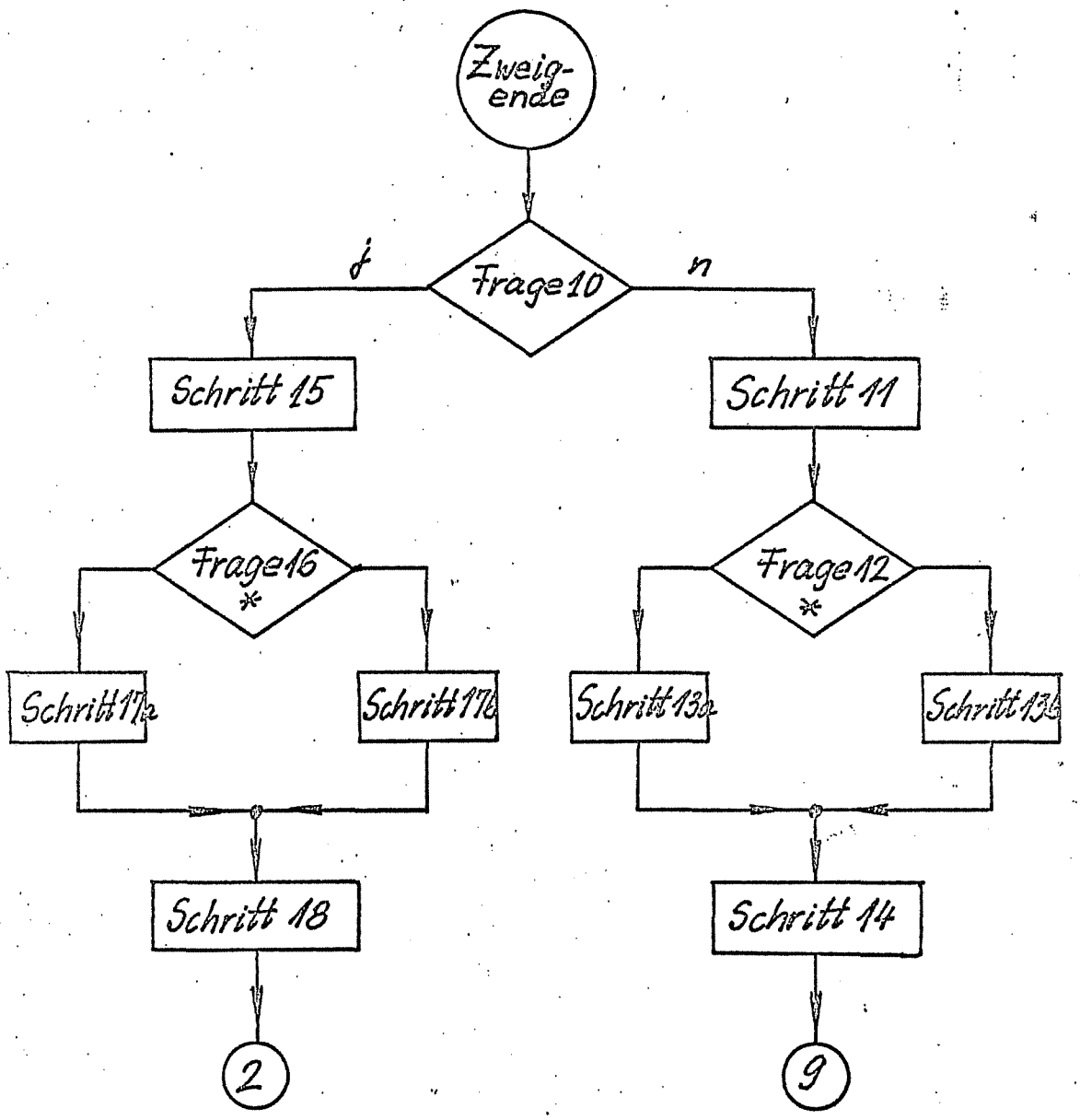


Abb. 1.1 : Programmablaufsplan zum Konvertierungsalgorithmus



* - Der nächste Schritt wird /a/ mit und /b/ ohne Kontaktfenster ausgeführt

Abb. 1.2. : Programmablaufsplan zum Konvertierungsalgorithmus



* - Der nächste Schritt wird /a/ mit und /b/ ohne Kontaktfenster ausgeführt

Abb. 1.3. : Programmablaufsplan zum Konvertierungsalgorithmus

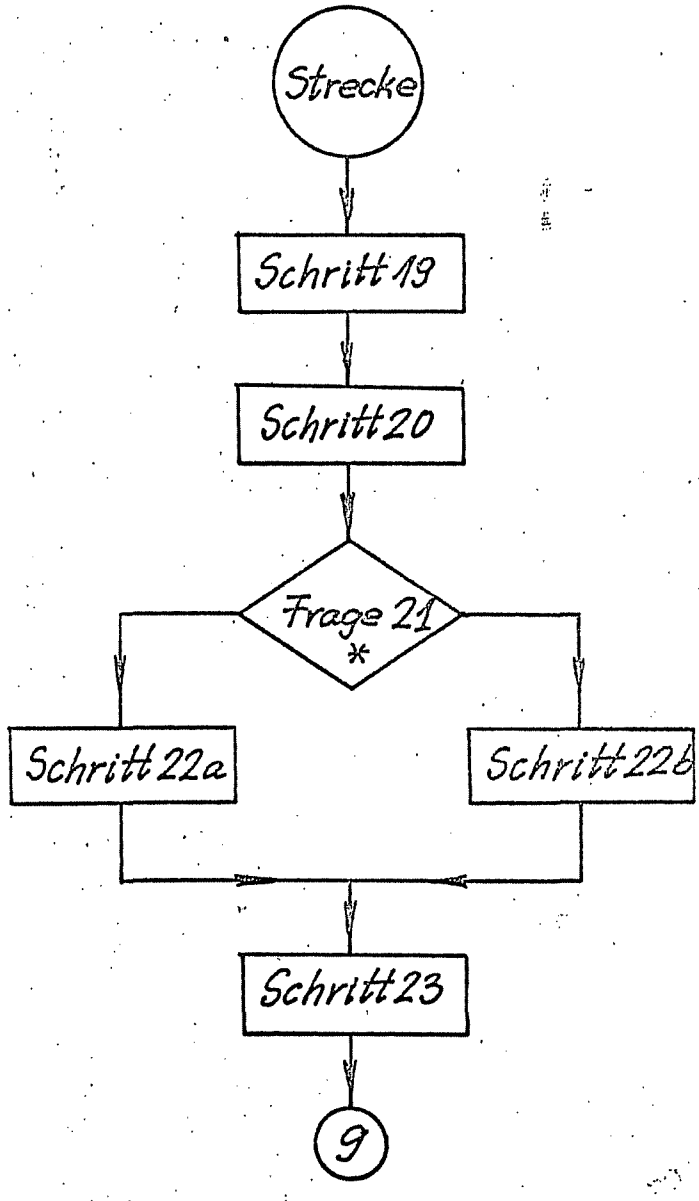


Abb. 1.4. : Programmablaufsplan zum Konvertierungsalgorithmus