

# Platinen-Layout mit Autorouter für ZX-Spectrum

Quelle: c't 1984, H8, S37...43; H9, S46...50; H10, S68...71

*Der nachfolgende Text ist ein von PS Mitte der 1980er Jahre bearbeiteter Auszug aus den Original-Artikeln. Dieser wurde damals zeitnah aus den vorliegenden Zeitschriften angefertigt und sollte als Grundlage zu eigenen Programmier-Arbeiten dienen.*

*Zu dem Beschreibungstext gehören mehrere Programm-Dateien:*

Autorouter BASIC-Programm.txt  
Autorouter Z80-AssemblerProgramm.txt  
Autorouter zusätzliche BASIC-Routinen.txt  
Autorouter AnzeigeErweiterungen.txt  
Autorouter PlacementErweiterungen.txt  
Autorouter ErweiterungenEntflechtung.txt  
Autorouter DruckGrafik.txt

Achtung!

*Da diese von teilweise schlecht oder kaum lesbaren Kopien stammen, können Übertragungs-Fehler nicht ausgeschlossen werden.*

Lee-Algorithmus in Maschinen-Sprache, Eingabe von Daten der zu verbindenden Punktepaare in BASIC

Eingabe der Daten und Platzieren der BE logisch getrennt, d.h. Erfassung in 3 Listen:

- *Bauteilliste:*  
pro BE wieviel Pins, welchen Abstand der Pins voneinander, wie groß ist das BE;  
evtl. Angaben pro IC-Typ, welche Pin Eingang bzw. Ausgang,  
welches Fan-Out (bzw. -in) zur Fehlerkorrektur
- *Components-Liste*  
Eintragung der BE mit Namen (z.B. "R1", "C1", "IC1" usw.);  
welches BE entspr. BE-Typ lt. BT-Liste  
für Orientierung des BE-Speicherplatzes vorsehen;
- *Connection-Liste*  
nach der Form "Bauteil i, PinNr; Bauteil j, PinNr" usw. eintragen,  
z.B. "IC1, 12; C5, 2 ..."

dann BE platzieren, nur bei optimaler Platzierung kann Autorouter optimal arbeiten, deshalb Manipulations- und Kontroll-Routinen

Zwei Möglichkeiten:

- Autoplacement -> kompliziert und rechenintensiv
  - Manuell-Placement
1. Placement-Routine (Positionierung und Orientierung manipulierbar -> besser für einfache Systeme geeignet)
  2. Darstellung auf dem Bildschirm
  3. Routine für die Darstellung der Verbindungen als Geraden
    - > vermeidbare Überschneidungen / Fehlpositionierungen erkennen
    - > Bauteilposition optimieren

## Reconnect-Routine

- > suchen von Verbindungsketten (Masseleitungen, Busleitungen)
- > 2. Schritt. äquivalente Kette mit kürzestmöglichen Verbindungen
- > evt. BE erneut positionieren usw.

Bevor der "Autorouter" aufgerufen wird, Leitungen nach Länge sortieren, zuerst die kurzen herstellen

Trotzdem ist es möglich, dass der "Autorouter" den Weg nicht findet, dann

*Manuell-Router* einsetzen, d.h.:

- > Geraden-Stücke interaktiv auf dem Bildschirm verschieben, um "Autorouter" Zugang zu den "isolierten" Punkten schaffen, danach wieder "Autorouter" aufrufen

**Autorouter-Routine** (siehe auch c't 84/2/57)

- Platine als Matrix mit 128x128 Elementen, 1 Element entspr. Quadrat mit 1/20 Zoll Kantenlänge
- Übergabe von Start- und Zielpunkt-Koordinaten einer Verbindung
- > Startpunkt-Koordinaten in "FIFOA"  
Untersuchung der Nachbarfelder der Matrix bis Zielpunkt gefunden:
- > Suchvorgang abbrechen

nicht gefunden:

- > freie Matrixpunkte mit Routecode versehen (siehe Bild) FIFOB
- > FIFOB wird zu FIFO A

für jeden Punkt aus FIFOA werden wieder die Nachbarfelder untersucht, d.h.

FIFOA entspr. Vorgänger-Wellenfront; FIFOB entspr. neue Wellenfront

- nicht alle Punkte in den Stack eintragen, nur "alte" und "neue" Wellenfront
- Routercode wird von 1...3 hochgezählt und beginnt dann wieder bei 1
- danach Weg wieder zurück zum Startpunkt durch zyklisches Vermindern des Routercodes
- > 2 Teile des Programmproblems: Weg zum Zielpunkt und wieder zurück

### **Programmteil 1**

verwaltet FIFOA und FIFOB, beide liegen im gemeinsamen Speicherbereich STKNEU ... STKMOD

Reg IX ist Zeiger für FIFOA

Reg IY ist Zeiger für FIFOB

Das Ende von FIFOA = Anfang von FIFOB, wird durch STKEND markiert (Bild "a")

Ist FIFOA abgearbeitet, zeigt IX auf Anfang von FIFOB, das jetzt anstelle FIFOA tritt

Reg IY zeigt auf den freien Bereich, wo Koordinaten der neuen Punkte abgelegt werden,

STKEND muss zur Markierung des Endes von FIFOA neu gesetzt werden (Bild "b")

Wenn IY obere Grenze vom FIFO-Bereich erreicht hat, dann auf Anfang (STKNEU), d.h.

IX = STKMOD, wenn ja: -> IX = STKNEU (Zeile 520...550)

IY = STKMOD, wenn ja: -> IY = STKNEU (Zeile 1360...1390)

Alte Eintragungen werden überschrieben, da nur noch nicht abgearbeitete Einträge auf FIFOA und neue auf FIFOB relevant sind.

Verhindert werden muss, dass neue Einträge auf FIFOB noch nicht abgearbeitete Einträge von FIFOA überschrieben werden:

IX = IY -> Fehlercode und Abbruch der Routine (Zeile 1400...1430, 1630...1660)

Wenn kein "Stackoverflow", so ist FIFOA irgendwann vollständig abgearbeitet, d.h.

IX = STKEND (Zeilen 560...580)

Letzte Bedingung ist:

IY = STKEND (Zeilen 590...620)

Ist FIFOA abgearbeitet und diese Bedingung wahr, so ist in FIFOB kein Element eingetragen worden, d.h. es gibt keinen "Ausweg", auch nicht zum Zielpunkt

-> Fehlercode und Routinen-Abbruch

### **Adressierung der Matrix**

Matrix 128x128 Elemente (Zeilen und Spalten 0...127)

XY-Adresse:  $XYADR = ((Y * xMAX) + X + BASIS)$

BASIS = Adr. des Matricelements (0:0)

xMAX = 128

x = X-Koordinate

y = Y-Koordinate

Da xMAX = 128 = 80H, wird Multiplizierung durch 7-fach Linksschieben des Operanden erreicht,

Beispiel:

$$\begin{aligned} 61 &= 3DH = 00111101B \\ * 128 &= 80H = 10000000B \\ + 115 &= 73H = 01110011B \\ 0001111010000000 &= 3DH sieben Mal nach links verschoben (16Bit-Wert) \\ +0000000001110011 &= 0001111011110011 \\ &1 \quad E \quad F \quad 3 \quad = 7923 = 62 * 128 + 115 \end{aligned}$$

Dafür sind folgende Operationen notwendig:

|       |                             |   |        |          |
|-------|-----------------------------|---|--------|----------|
| SLA E | Bit 7 rausschieben          | ) | D-Reg: | Y-Koord. |
| RR D  | Bit 0 ins Carry             | ) | E-Reg: | X-Koord. |
| RR E  | ... und in Bit 7 übernehmen | ) |        |          |

Damit ist relative Adresse berechnet und mit

LD HL, (BASIS)

ADD HL, DE die Absolutadresse (Zeilen 1180 ... 1220, bzw. 2350 ... 2390)

Nach Erreichen Zielpunkt Rückweg mit Codes markieren, die eindeutig die Bahnstücke beschreiben.

Für jedes Bahnstück muß Tupel < von, nach > bestimmt und codiert werden (siehe Tabelle 2).

Bild 9a: P(6,7) Zielpunkt, P(3,6) Startpunkt

P(6,6) erstes Bahnstück

Suche nach Punkt mit Routercode 3 ->

P(6,5)

Koord. Ist in STKEND gespeichert

(VON) = Richtung von P(6,6)

-> P(6,5) hier "oben"

dann Routercode 2 - P(6,4), Zugrichtung = (NACH)

damit liegt Bahnstück für P(6,5) fest, dessen Position in STKEND steht:

< VON, NACH > = < "oben", "oben" = (I)

P(6,5) wird mit dem passenden Code überschrieben,

die momentane Position P(6,4) -> STKEND, VON -> NACH überschreiben

Bild 9b: Suche Routercode 1 -> P(5,4), Zugrichtung "links" -> NACH

das Bahnstück für P(6,4) ist < VON, NACH > = < "oben", "links" >

Bild 9c: = (+)

Der Prozess wird beendet, wenn der Startpunkt erreicht ist (Bild 9d).

Algorithmus müsste beim 1. Punkt versagen, weil kein "Vorgänger" vorhanden, nicht jedoch im Programm, da in STKEND die Adresse des Punktes steht, welcher überschrieben werden soll, d.h. beim ersten Überschrieben zeigt STKEND auf nicht mehr benutzten FIFO-Bereich.

FILL aufgerufen in Zeile 430 ersetzt die Leerzeichen (ASCII = 32) durch Code 156.

FREE aufgerufen in Zeile 740 überschreibt alle Routercodes mit Code 156.

### **Besonderheiten beim "Spectrum"**

BASIC-Anweisung DIM A\$ (128, 128) erzeugt eine Matrix mit 128 Zeilen und 128 Spalten. Gelöscht und initialisiert wird die Matrix im BASIC-Teil (Tabelle 3)

Das Maschinen-Programm wird mit FN A (A\$, P\$) aufgerufen, in "FN" erfolgt dann der 'USR'-Aufruf.



## (2. Teil)

### **Spectrum-spezielle Befehle:**

DIM B\$(100,7): Feld von 100 Strings mit je 7 Charkteren,  
Strings werden mit 'Blanks' initialisiert  
H\$ = B\$(71,3): H\$ wird das 3. Zeichen des 71. Strings zugeordnet  
H\$ = B\$(71, 2TO5): H\$ wird der Teilstring des 71. Strings zugeordnet,  
bestehend aus 2 bis 5 Zeichen

### Listen

Bauteilbibliothek: Feld E\$(20,7) und Zeiger 'MXE', der auf den letzten Eintrag zeigt  
Jede Definition enthält 7 Einträge (Tab1), die immer  
den gleichen Variablen zugewiesen werden  
(Prog.-Zeilen 44-46, 5-6)  
-> nur Darstellung von R, C, IC (DIL) und Steckerleisten möglich,  
keine TO-Gehäuse etc. (Bild 1)

Components-Liste: Feld B\$(100,7) – Tab. 2  
Name und Nr. des zugehörigen BE-Bibl.-Eintrags  
Drehung und Positionierung des BE erst bei Positionierung  
(Bild 2, Def. der Drehung)

Verbindungsliste: Feld V\$(500,7) – Tab. 3)  
enthält alle Verbindungen des Schaltplans,  
bezieht sich auf Bauteilliste (Components-)

Listen werden mit den Routinen ab Zeile 9500 erfaßt, bzw. auf Kassette laden und davon  
holen (ab 9800)

### Positionieren

Struktogramm Bild 3 – 3 Blöcke

- Anzeige der BE und Verbindungen
- Auswahl eines BE und Anzeige der zugehörigen Daten  
(Name, Position, Drehung)

|                     |           |  |
|---------------------|-----------|--|
| Unterprogramm-Zeile | 5 ... 11  | gibt BE 'B\$(2)' auf Bildschirm aus<br>Daten werden den Listen entnommen   |
|                     | 15 ... 18 | Anzeige aller BE   |
|                     | 26 ... 31 | Anzeige der Verbindungen   |
|                     | 50 ... 80 | baut Anzeige auf, läßt BE blinken und<br>zeigt zugehörige Daten im unteren<br>Bildschirmbereich an<br>Auswahl der BE mit Cursor-Tasten |

Über Tastatur werden neue Koordinaten (bezogen auf PIN1) eingegeben (Zeile 1200 ... 1270)

Variablen-Abfrage: - unverändert (ENTER)  
- absolut geändert -> neue Pos. eingeben  
- relativ geändert -> z.B. x+5: BE 5 Schritte nach rechts

Gültigkeitsüberprüfung der Tastatur-Eingabe in den Zeilen 1100 ... 1170  
(Anzeige-Routinen sind Rechner-spezifisch!)

### Reconnect

Aufgabe: Ketten von Verbindungen aus der Verbindungsliste suchen und durch  
gleichwertige Ketten ersetzen

Hauptroutine (2000 ... 2031): Steuerung -> initialisieren der Felder 'K\$' und 'L\$'  
Zeiger 'HXV' auf letzten Eintrag in der  
Verbindungsliste V\$ setzen

'HXV' trennt innerhalb der Verbindungsliste die Teillisten:

- noch nicht bearbeitete Verbindungen
- bearbeitete Verbindungen

Alle Verbindungen vom 1. Eintrag bis einschließlich 'H\$ (HXV)' = noch nicht bearbeitet  
von 'H\$ (HXV + 1)' bis zum letzten Eintrag sind bearbeitet

Nach Initialisierung gibt Steuerung die Kontrolle an

- GETCON )
- RECON ) Bearbeiten von Verbindungsketten

Alle Verbindungen bearbeitet: -> 'HXV = 0'

GETCON (2032 ... 2095): erfasst die Anschlusspunkte mit Koordinaten,  
die zu einer Kette gehören  
Verbindungen aus 'V\$', die zu einer Kette gehören,  
werden aus der unbearbeiteten Teilliste gestrichen und  
in die Teilliste 'bearbeitete Verbindungen' übernommen

In GETCON wird benutzt:

- V\$ Verbindungsliste
- HXV Zeiger auf die letzte, noch nicht bearbeitete Verbindung
- L Zeiger auf V\$;  $1 \leq L \leq HXV$
- K\$ Punktliste für die Punkte einer Kette
- MXK Zeiger auf letzten Eintrag in K\$
- PTK Zeiger auf K\$;  $1 \leq PTK \leq MXK$

BECON (2200 ... 2350): bildet aus erfassten Punkten neue,  
möglichst kurze Verbindungen,  
die in V\$ zurück geschrieben

In BECON wird benutzt:

- V\$, HXV, K\$, MXK, wie bei GETCON und
- J Zeiger auf K\$;  $1 \leq J < MXK$
- L\$ Punktliste der schon verbundenen Punkte einer Kette
- MXL Zeiger auf den letzten Punkt in L\$
- I Zeiger auf L\$;  $1 \leq I \leq MXL$

Bild 4 zeigt Struktogramm der Reconnect-Routine.

Die RECON-Routine ist aufwendig, da zwei Sonderfälle zu berücksichtigen sind:

- Router kann einen Anschlusspunkt höchstens mit 4 weiteren verbinden  
Punkte in L\$, die schon mit 4 Punkten verbunden sind, werden deshalb aus  
der Liste gestrichen!
- Ketten sollen keine geschlossenen Kreise enthalten. Die Eigenschaft eines  
Kreises ist, dass eine Verbindung entfernt werden kann, ohne dass die  
Verbindung an allen Punkten unterbrochen ist

Reconnect-Routine ermittelt solche Verbindungen:

- Kreis – Kette -> GETCON Punktliste in pkt(px) zweimal
- > RECON Punkt px in K\$ und L\$, d.h. Verbindung (px, px)  
mit kürzest möglicher Länge (0)

Verbindung wird nicht nach V\$ zurück geschrieben, d.h. Anzahl der Verbindungen wird vermindert, 'V\$' wird verkürzt, 'MXV' angeglichen

Programmzeilen 2700 ... 2730 enthalten eine Bubble-Sort-Routine, die vorm Routen aufgerufen werden sollte, um Verbindungslisten aufsteigend nach der Länge zu sortieren. Abschluß ist die Menü-Steuerung, von der alle Routinen aufgerufen werden. Kaltstartadresse ist 9100, RAMTOP erniedrigt, Maschinen-Programm geladen und 'Charaktere' definiert.

### Reconnect-Routine - Erklärung

Bild 1a zeigt die Verbindungspunkte, die mit 2 Ketten verbunden sind.

Bild 1b zeigt zugehörige Verbindungskette, jedoch noch unbearbeitet, d.h. die Ketten sind noch nicht optimiert.

Steuerung setzt 'HXV' auf letztes Element von 'V\$' und übergibt an GETCON.

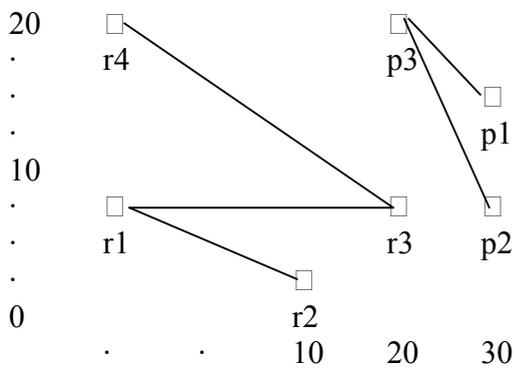


Bild 1a

von r1\_PIN1 nach r2\_PIN1  
 von r4\_PIN1 nach r3\_PIN1  
 von r1\_PIN1 nach r3\_PIN1  
 von p3\_PIN1 nach p1\_PIN1  
 von p2\_PIN1 nach p3\_PIN1

Bild 1b

In 'GETCON' - 'r1' der ersten Verbindung in Liste (in 'V\$') nach 'K\$' übernommen. Zeiger 'MXK' (zeigt immer auf die letzte Eintragung in 'K\$') wird auf ersten und z.Zt. letzten Eintrag in 'K\$' (r1) gesetzt. 'PTK' (Zeiger auf Punkt in 'K\$', nach dem in 'V\$' gesucht wird) wird auf 'r1' gesetzt.

| <u>Liste V\$</u>     | <u>K\$</u>     |
|----------------------|----------------|
| L> von r1 nach r2    | PTK > r1 < HXK |
| von r4 nach r3       | ...            |
| von r1 nach r3       | ...            |
| von p3 nach p1       | ...            |
| von p2 nach p3 < HZV |                |

Bild 2a

- > Suchvorgang nach Punkt 'K\$ (PTK)' in V\$
- gesucht wird in 'V\$' bis zum 'HXV' einschließlich Punkt 'r1'
- sofort in erster Verbindung gefunden
- 'r2' (korrespondierender Punkt zu 'r1' wird als neuer und letzter Punkt in 'K\$' eingetragen und 'MXV' erhöht (Bild 2b)

|    |                      |          |            |
|----|----------------------|----------|------------|
|    | <u>V\$</u>           |          | <u>K\$</u> |
| +> | von r1 nach r2       | PTK > r1 |            |
|    | von r4 nach r3       | →        | r2 < HXK   |
|    | von r1 nach r3       |          |            |
|    | von p3 nach p1       |          |            |
|    | von p2 nach p3 < HXV |          |            |

Bild 2b

Punkte 'r1' und 'r2' (Verbindung in 'V\$(L)') sind jetzt in 'K\$'.

Verbindung gilt als bearbeitet und wird mit letzter, noch nicht bearbeiteten Verbindung 'V\$(HXV)' ausgetauscht.

'HXV' wird um eine Position erniedrigt und trennt somit bearbeitete von unbearbeiteten Verbindungen (Bild 2c).

|    |                      |          |            |
|----|----------------------|----------|------------|
|    | <u>V\$</u>           |          | <u>K\$</u> |
| +> | von p2 nach p3       | PTK > r1 |            |
|    | von r4 nach r3       |          | r2 < HXK   |
|    | von r1 nach r3       |          |            |
|    | von p3 nach p1 < HXV |          |            |
|    | von r1 nach r2       |          |            |

Bild 2c

Ab einschließlich der vertauschten Verbindung wird Suche nach weiteren Verbindungen, die 'r1' enthalten, fortgesetzt (Bild 3a).

+> zeigt auf solche Verbindung, Punkt 'r3' wird in 'K\$' übernommen und 'MXK' erhöht (Bild 3b).

Beide Punkte (r1, r3) sind in 'K\$' erfasst.

-> Verbindung wird in Teilliste der bearbeiteten Verbindungen übernommen (Bild 3c).

|    |                      |          |            |
|----|----------------------|----------|------------|
|    | <u>V\$</u>           |          | <u>K\$</u> |
| +> | von p2 nach p3       | PTK > r1 |            |
|    | von r4 nach r3       |          | r2 < HXK   |
|    | von r1 nach r3       |          |            |
|    | von p3 nach p1 < HXV |          |            |
|    | von r1 nach r2       |          |            |

Bild 3a

|    |                      |          |            |
|----|----------------------|----------|------------|
|    | <u>V\$</u>           |          | <u>K\$</u> |
|    | von p2 nach p3       | PTK > r1 |            |
|    | von r4 nach r3       |          | r2         |
| +> | von r1 nach r3       | →        | r3 < HXK   |
|    | von p3 nach p1 < HXV |          |            |
|    | von r1 nach r2       |          |            |

Bild 3b

|    | <u>V\$</u>           | <u>K\$</u> |
|----|----------------------|------------|
|    | von p2 nach p3       | PTK > r1   |
|    | von r4 nach r3       | r2         |
| +> | von p3 nach p1 < HXV | r3 < HXK   |
| +> | von r1 nach r3       |            |
|    | von r1 nach r2       |            |

Bild 3b

Ab einschließlich der vertauschten Verbindungen wird Suche fortgesetzt. Die Liste der noch nicht bearbeiteten Verbindungen enthält aber keine Verbindungen mit dem Punkt 'r1' mehr. Dies wird erkannt durch '+ > HXV'. 'PTK' wird erhöht und nun 'r2' zu suchen, der in weiteren Verbindungen vorkommen kann.

### Teil 3

#### **Erweiterung der Placement-Routine** (Prog. 1)

'neue' Routine mit Kommandos nach folgender Tabelle:

- |      |                |   |  |
|------|----------------|---|--|
| 6, 7 | Cursor-Tasten  | -   | Auswahl des BE (Position, Drehung ändern)<br>das ausgewählte BE blinkt<br>BE-Daten (Name, Position) werden in Fußzeile angezeigt |
| b    | Break          |   |  |
| c    | Cursor-Mode    | (Kommandos nach Tabelle 3)  |  |
| d    | Drehung        | (0 ... 3 -> Drehung um 90° gegen den Uhrzeigersinn; Pin1)   |  |
| e    | BE-Definition, | z.B. Änderung der Gehäuseabmessung bei unterschiedlicher<br>PIN-Anzahl -> zusätzliche Bestätigung |  |
| o    | 'old'          | -> macht letzte Positions-Änderung rückgängig   |  |
| z    | 'zoom'         | -> Anzeigeformat  | 1) 128 x 128 Matrixpunkte<br>2) 128 x 88 Matrixpunkte  |

#### Erweiterung der Anzeige-Routine (Prog.2)

Folgende Befehle:

- |            |   |
|------------|---|
| 5 ... 8    | läßt die Platine unter dem Bildschirmfenster 'wandern' (Pan-Funktion) |
| -"- & CAPS | -"-, aber in größeren Schritten                                       |
| b          | break (-> Rückkehr ind Hauptmenü)                                     |

#### Manuell routen (Erweiterung der Placement-Routine)

- verschieben von schon geroutetem Geradenstück senkrecht, bzw. waagerecht
  - > zum Leiterbahnen begradigen ab blockierte Punkte, die beim Routen nicht verbunden werden konnten, freilegen
  - > Cursor auf zu verschiebendes Geradenstück, dann auf 'H'-Taste drücken (damit werden die Koordinaten gespeichert); dann Position markieren, wo Gerade hin soll, dann wieder 'H'-Taste drücken (2. Koordinatensatz übernehmen) und Routine zum Verschieben aufrufen

Im 'Cursormode' stehen folgende Programme zur Verfügung (Tabelle 3):

- |         |   |
|---------|---|
| 5 ... 8 | Cursor  |
| h / H   | Hold – Markierung der Geraden- und Ziel-Koordinaten |
| p / P   | Platine - zurück zur Anzeigeroutine                 |

Bei der 'Placement'-Routine gilt:

- |         |  |
|---------|--|
| 5 ... 8 | Cursor   |
| a       | Abbruch – Cursormode -> Placement-Routine<br>Änderungen werden <u>nicht</u> übernommen                   |
| e       | Ende - Cursormode .> Placement-Routine,<br>neue Position wird übernommen, Orientierung BE wird abgefragt |

Manuell-Router im "Programm 2" in den Zeilen 6000 ... 6680

(Bestimmung des Codes der Abschlußstücke der zu verschiebenden Geraden G und G'  
Fallentscheidung ab Zeile 6600)

Bild 1 zeigt die Wirkung und verwendete Variable, sowie deren Bedeutung.

Durch die Cursor-Routine der Platinenanzeige (5100 ... 5240) sind die Variablen K(1, 1), K(1,2) die Koordinaten des Geradenstücks der zu verschiebenden Geraden festgelegt.

Variable K(2, 1), K(2, 2) bestimmen die Koordinaten, zu der die Gerade verschoben werden soll. Gerade wird ab (K(1, 1), K(1, 2)) nach 'links' bis zu einem Lötspunkt oder Eckstück verfolgt.

Wird die Gerade durch Lötspunkt abgeschlossen, sind alle Eck-Codes bestimmbar (Fallunterscheidung ab Zeile 6600)

Anderenfalls ist die Ecke Abschluß. Angrenzende Gerade G wird ebenfalls bearbeitet (ab Zeile 6400). Aus den gewonnenen Daten erfolgt die Ableitung des Eck-Codes, die in R\$ eingetragen werden (Bild 2a).

Nächster Schritt: - Suche rechtes Anschlußstück ausgehend von linker Ecke der Geraden (Zeile 6180 ... 6200)  
- Löschung der Codes der Geradenstücke in der Matrix

Gleiches gilt für die Eck-Codes des rechten Abschluß (Bild 2b).

Sind die Eck-Codes erfasst -> Gerade verschieben  
-> G und G' verlängert, bzw. verkürzt und Eck-Codes eintragen (6250 ... 6340)

dann linke Abschlußgerade G' anpassen (Bild 2d)

Letzter Schritt (6350 ... 6370):

Eintragung der verschobenen Geraden (neu) in die Matrix (Bild 2e)

Falls mit 'Manuell-Router' blockierende Punkte freigelegt werden

-> Neuaufruf de Routers

Durch Erweiterung der Routine ("Programm 3") neue Befehle:

- N(eu) - löscht die Matrix und routet sämtliche Verbindungen neu
- R(est) - keine Löschen, sondern Ergänzen von Verbindung zu Punkten, die vorher blockiert waren (1841 ... 1870)
- A(lternativ) - kein Löschen der Matrix, sucht alternative Verbindungen (2400 ... 2670)

Wenn eine Verbindung, die zu einer Kette gehört, nicht herstellbar ist, entstehen automatisch zwei Teilketten.

- > Suche nach einer Verbindung, die die Teilketten an beliebiger Stelle miteinander verbindet. Dazu folgende Routine: (2500 ... 2560 ... 2570)
- > es werden beide Teilketten bestimmt (-> 'GETCON')
- > mit allen Punkt-Paaren der Teillisten Router aufrufen, um Verbindung herzustellen

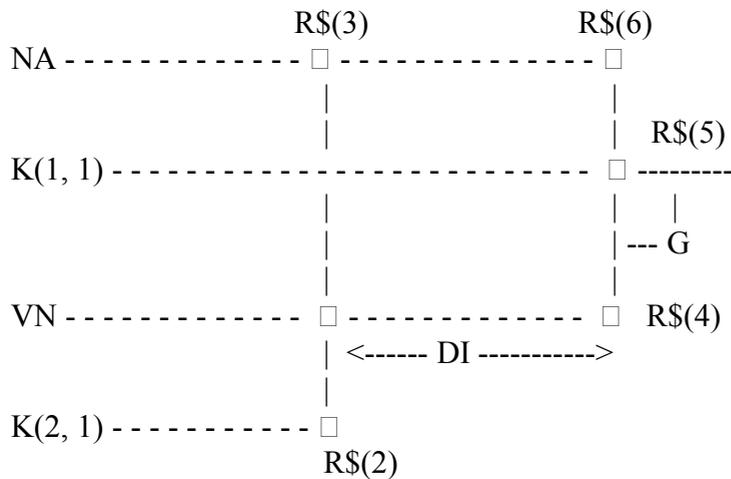
**Druckeroutine** (Programm 3 und 4)

Vorraussetzung: - frei definierbarer Zeichensatz  
- Proportionalschrift  
- einstellbarer Zeilenabstand

hier FX 80 (EPSON)

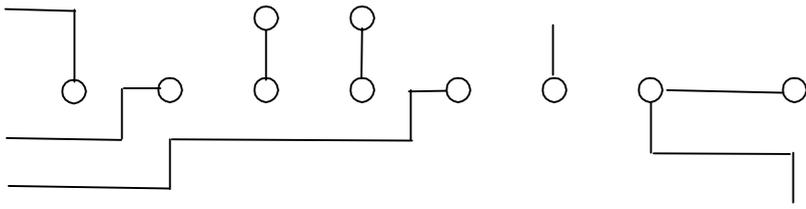
- Initialisierung Drucker
- Definition Grafikzeichen im Spectrum
- Reservierung Platz für Maschinen-Programm

Nach Init sind Zeichen für Geradenstücke und Ecken, sowie Lötunkte definiert. Jedes Zeichen ist im Proportional-Mode ca. 1/20 Zoll breit; Zeilenabstand ebenfalls (Zeile 310)  
 Ausgabe der Matrix Zeile für Zeile -> fertiges Layout.



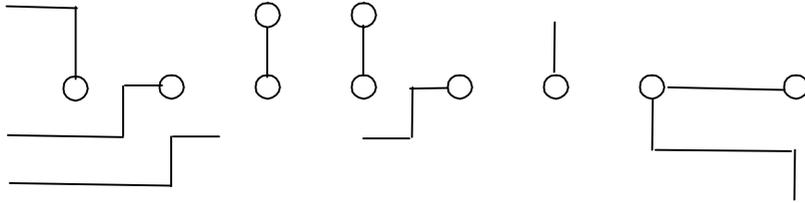
- VN (von)      Ordinate der zu verschiebenden Geraden
- NA (nach)    Ordinate, zu der die Gerade verschoben werden soll
- DI (Distanz) Länge des zu verschiebenden Geradenstücks
- R\$            Codes für Abschlußstücke
- RS(1), R\$(4) neuer Code für das linke, bzw. rechte Abschlußstück der zu verschiebenden Geraden
- R\$(2), R\$(5) neuer Code für die Abschlußstücke G, bzw. G'
- R\$(3), R\$(6) neuer Code für die Abschlußstücke der verschobenen Geraden

Bild 1: Variablen, die der 'Manuel Router' benutzt und ihre Bedeutung



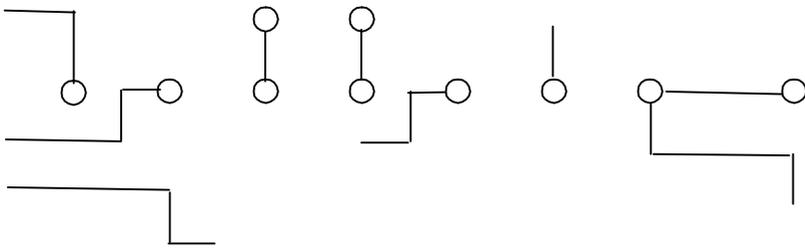
R\$ = “ ++ “

a)

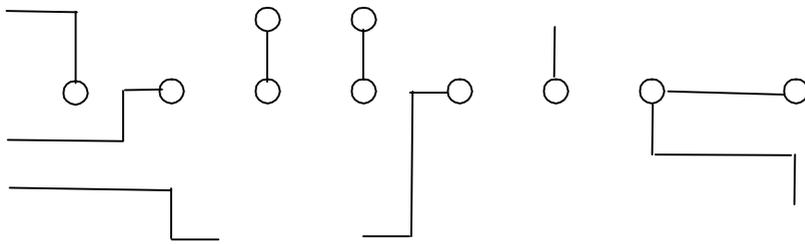


R\$ = „ ++ | ++ “

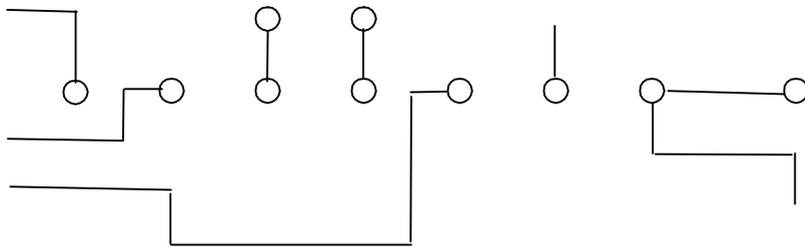
b)



c)



d)



e)

Bild 2: Phasen beim Verschieben einer Geraden

# Anhang

## Ablaufschema für den LAY-OUTER

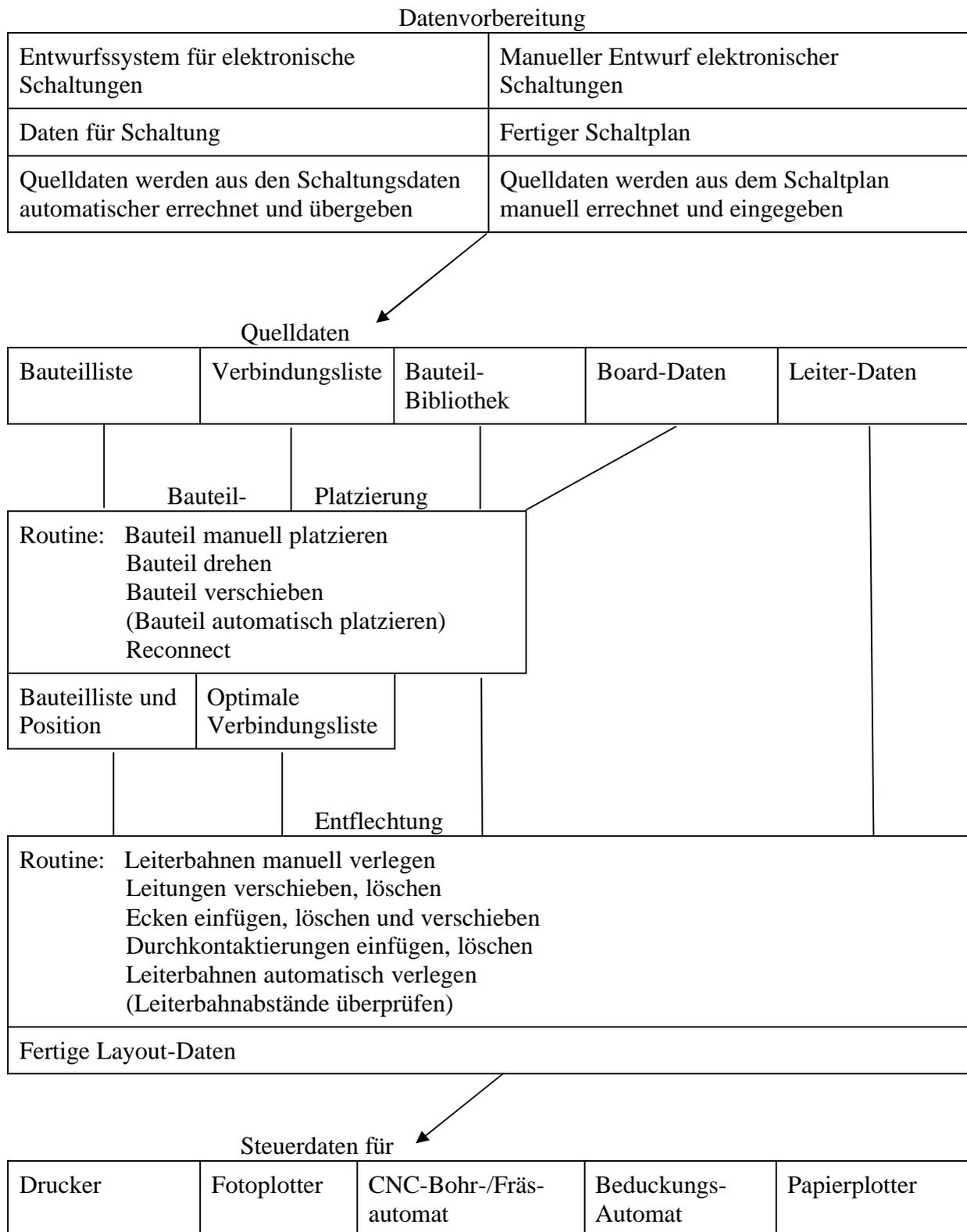
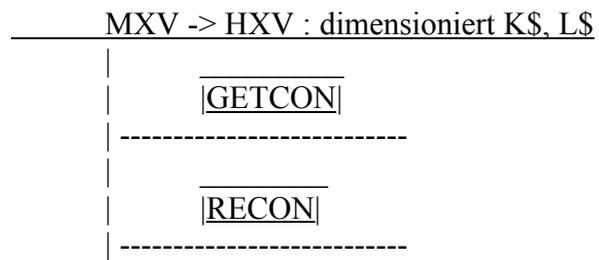
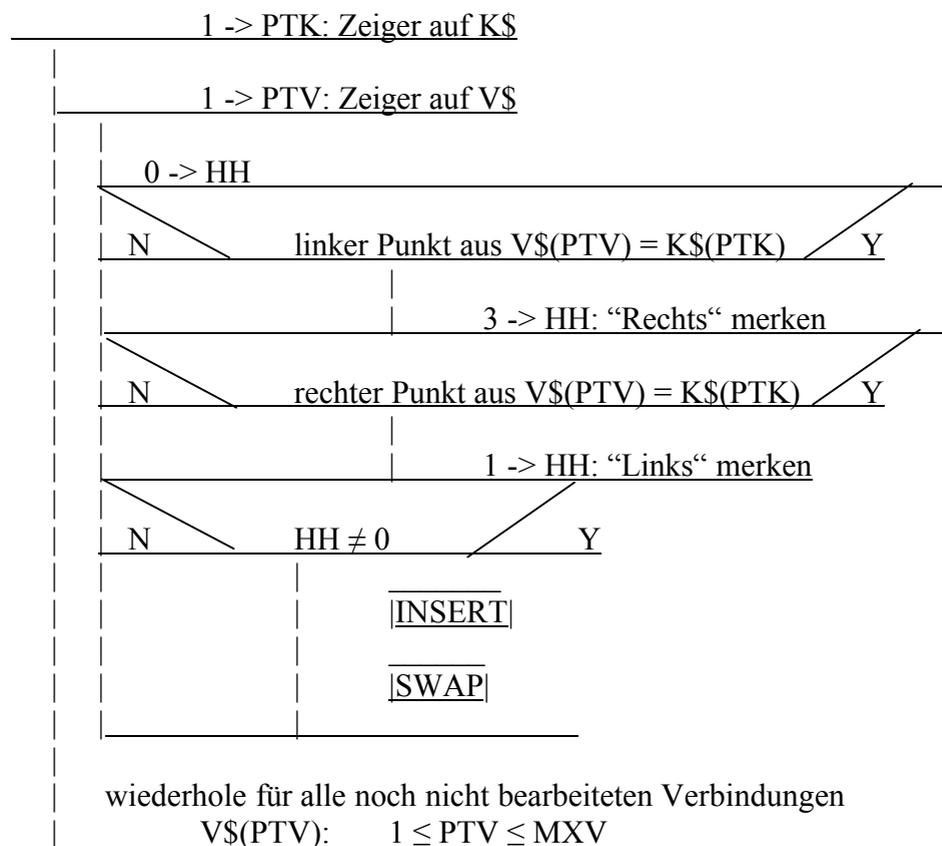


Bild 4: Struktogramm der Reconnect-Routine



bis HXV = 0 (bis alle Verbindungen abgearbeitet)

TITEL : GETCON



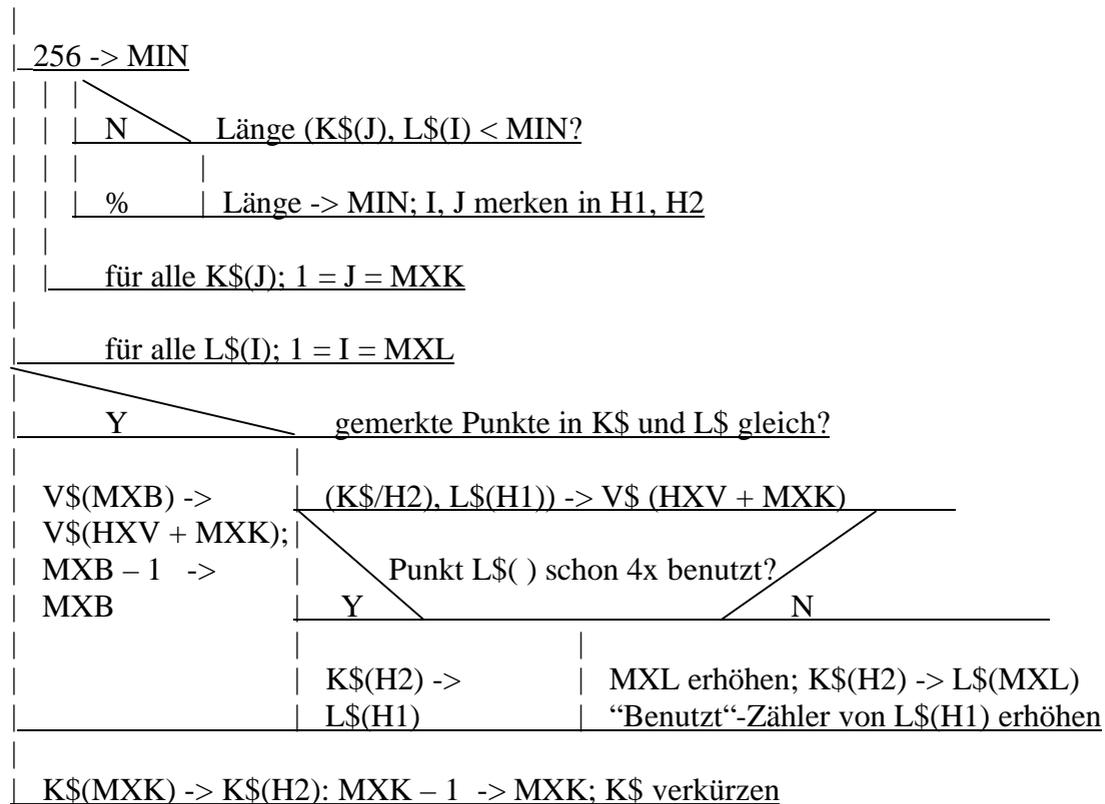
Wiederhole für alle Punkte K\$(PTK):

K\$(PTV): 1 ≤ PTK ≤ MXK

oder bis alle Verbindungen in V\$ abgearbeitet (HXV = 0)

TITEL : RECON

Letzter Punkt aus K\$ nach L\$; 1 -> MXL; MXK - 1 -> MXK



bis MXK = 0 (bis K\$ "leer" ist)

TITEL : INSERT

[GETPOS] Position des einzutragenden Punktes bestimmen

-----  
MXK erhöhen; Punkt unter K\$(MXK) eintragen  
-----

TITEL : SWAP

Erfasste Verbindung V\$(PTV) und letzte nicht

Erfasste Verbindung V\$(MXV) vertauschen

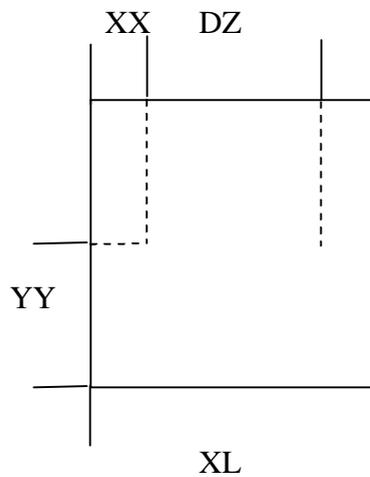
-----  
MXV - 1 -> MXV

Tabelle 1: Bauteilbibliothek

| Eintrag   | Variable | Bedeutung                                   |
|-----------|----------|---|
| E\$(J, 1) | HG       | Distanz zur nächsten PIN-Reihe in 1/20 Zoll |
| E\$(J, 2) | PN       | Anzahl der PINs in der ersten Reihe         |
| E\$(J, 3) | DZ       | Distanz zwischen den PINs in 1/20 Zoll      |
| E\$(J, 4) | XX       | X-Abstand von PIN1 zum Gehäuserand          |
| E\$(J, 5) | YY       | Y-Abstand von PIN1 zum Gehäuserand          |
| E\$(J, 6) | XL       | X-Länge des Gehäuses                        |
| E\$(J, 7) | YL       | Y-Länge des Gehäuses                        |

z.B. gültige BE-Angaben

- (1) HG = 0  
 PN = 2  
 DZ = 6  
 XX = 1  
 YY = 4  
 XL = 8  
 YL = 8



- (2) HG = 0  
 PN = 3  
 DZ = 2  
 XX = 1  
 YY = 1  
 XL = 6  
 YL = 2



- (3) HG = 6  
 PN = 8  
 DZ = 2  
 XX = 0  
 YY = 0  
 XL = 14  
 YL = 6



Tabelle 2: Bauteilliste

| Eintrag      | Bedeutung                                    |
|--------------|--|
| B\$(J, 1to3) | Name des Bauteils (z.B. IC1, oder R15)       |
| B\$(J, 4)    | Nummer des beschriebenen Bibliothekseintrags |
| (B\$(J, 5)   | (0 ... 3); Drehung des BE in 90°-Schritten   |
| (B\$(J, 6)   | X-Position des BE (bezogen auf PIN1)         |
| B\$(J, 7)    | Y-Position des BE (bezogen auf PIN1)         |

Bild 2: Definition der BE-Drehung

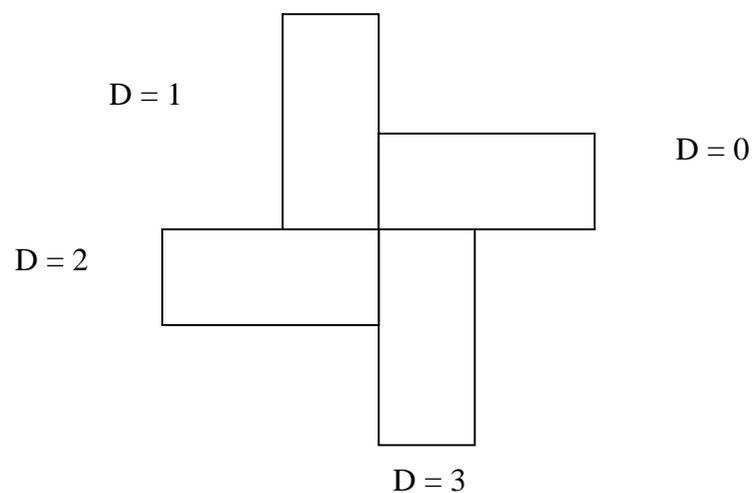


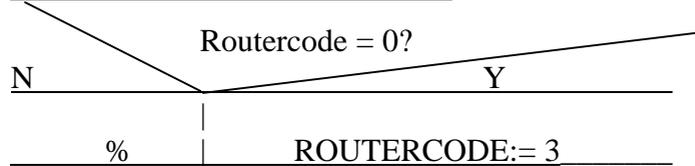
Tabelle 3: Verbindungsliste

| Eintrag   | Bedeutung  |
|-----------|--|
| V\$(J, 1) | Nr. des BE, von dem die Verbindung ausgeht         |
| V\$(J, 2) | Nr. des dazugehörenden PINs                        |
| V\$(J, 3) | Nr. des BE, zu dem die Verbindung führt            |
| V\$(J, 4) | Nr. des dazugehörenden PINs                        |
| V\$(J, 5) | Markierung, falls die Verbindung nicht herstellbar |
| V\$(J, 6) | unbelegt   |
| V\$(J, 7) | Länge der Verbindung                               |



TITLE : DECROT

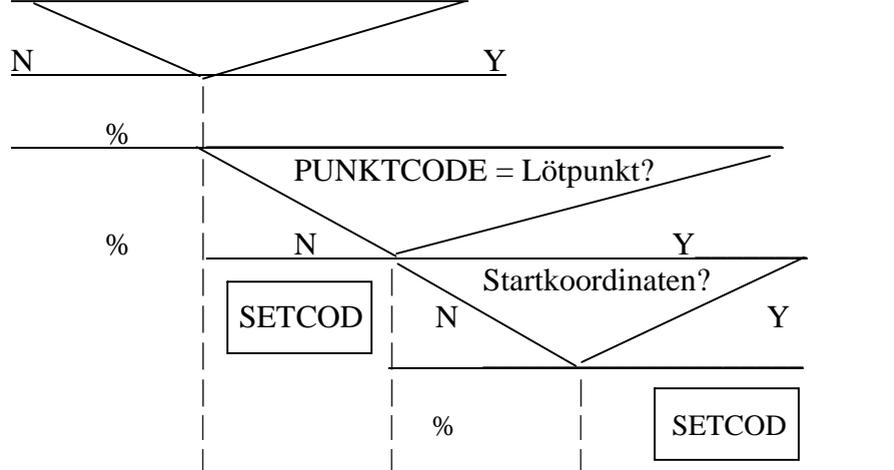
Routercode:= ROUTERCODE - 1



TITLE : PKTCOD

Berechne Pktadr: Pktadr -> POS

PUNKTCODE = ROUTERCODE?



TITLE : SETCON

< VON, ZUGRICH > -> (STKEND)

POS -> STKEND

ZUGRICH -> VON

BACK sucht den Weg vom Zielpunkt zum Startpunkt

DECROUT erniedrigt den ROUTERCODE um 1

Adresse des Punktes wird berechnet

Setzen des Codes (Bild 10a ... d)

Bild 7: zwei FIFOs im RAM

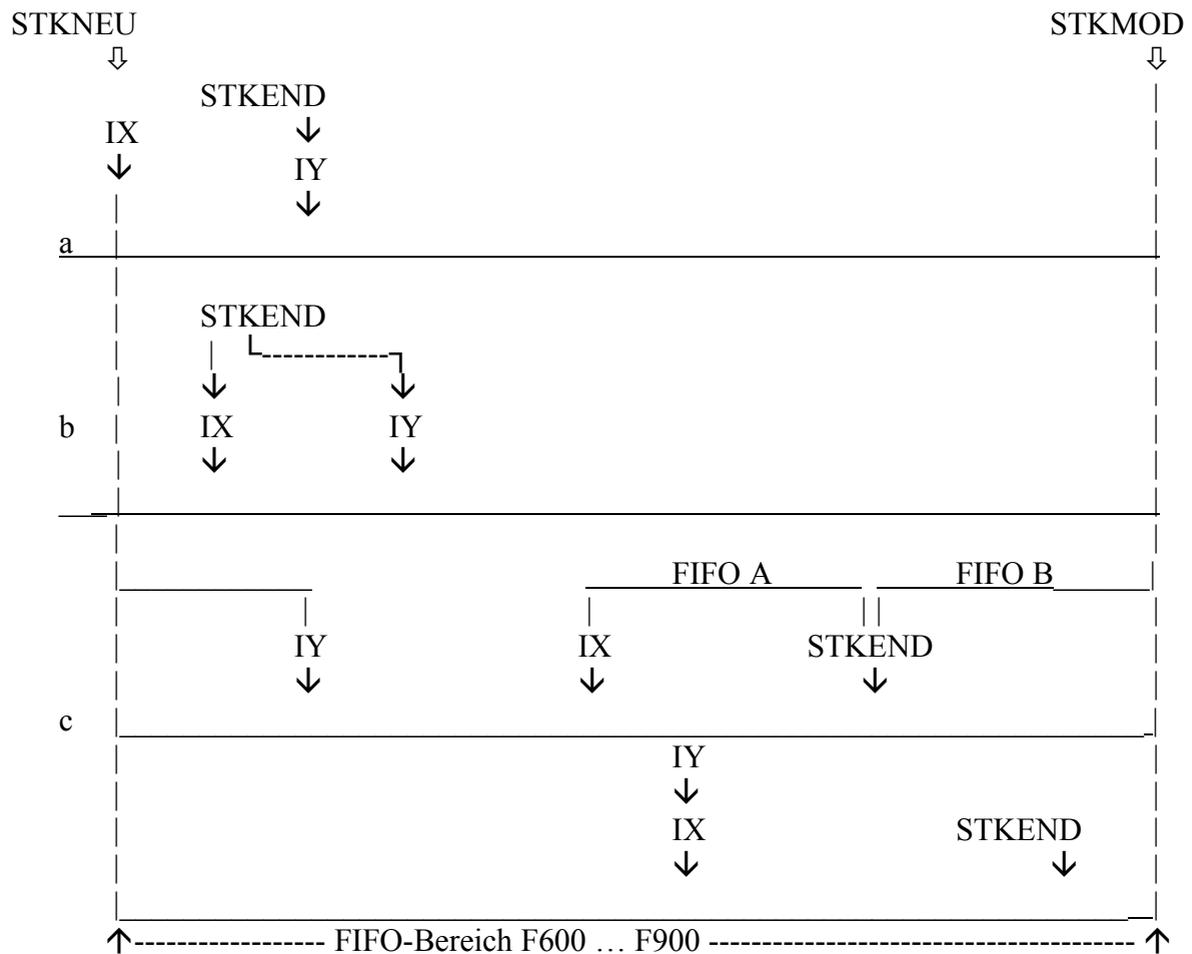


Tabelle 2

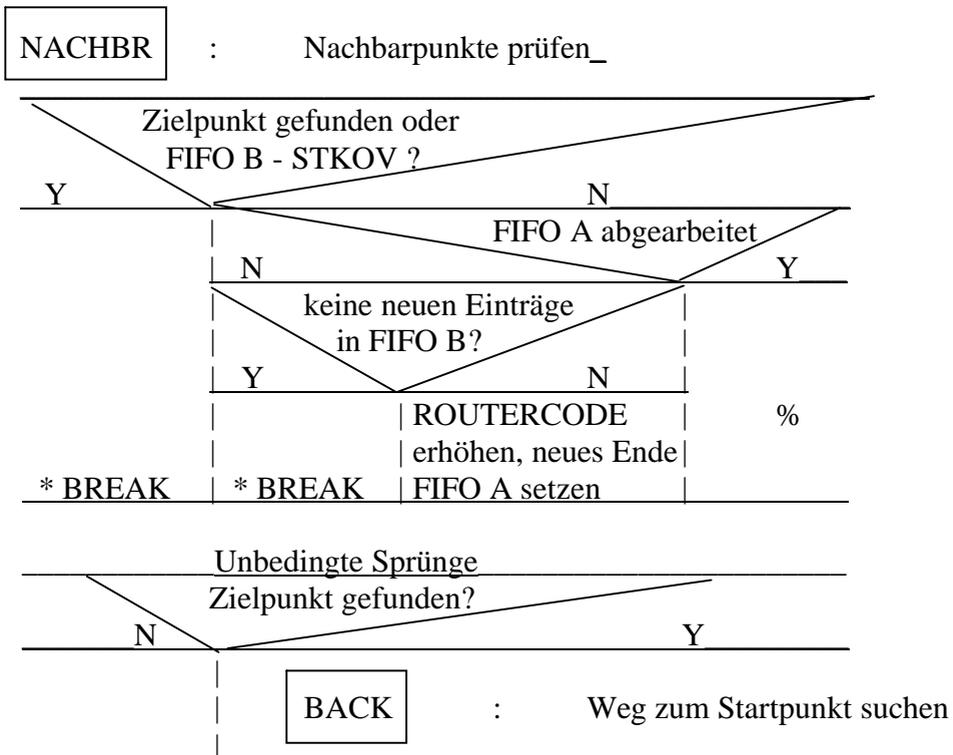
| Code | Symbol | Verwendung  |
|------|--------|-------------|
| 150  | ┌      | O/R; L/U    |
| 151  | —      | R/R; L/L    |
| 152  | └      | O/L; R/U    |
| 153  | └┘     | U/R, L/O    |
| 154  | ┘      | R/O: U/L    |
| 155  |        | O/O; U/U    |
| 156  | —      | Leerzeichen |
| 157  | α      | Lötunkt     |

TITLE : ROUTER

Parameter von BASIC holen

FIFO A / FIFO B initialisieren + Zielpunkt in FIFO A eintragen

Punkt aus FIFO A holen / FIFO A-Pointer erhöhen



restliche Routercodes löschen : zurück zum BASIC

Bild 8a: Struktogramm des Routers. Die Routine "NACHBR" ist in Bild 8b erläutert  
BREAK unterbricht die unbedingte Schleife

TITLE : NACHBR

Untersuche unteren Punkt

|        |
|--------|
| PKTCHK |
|--------|

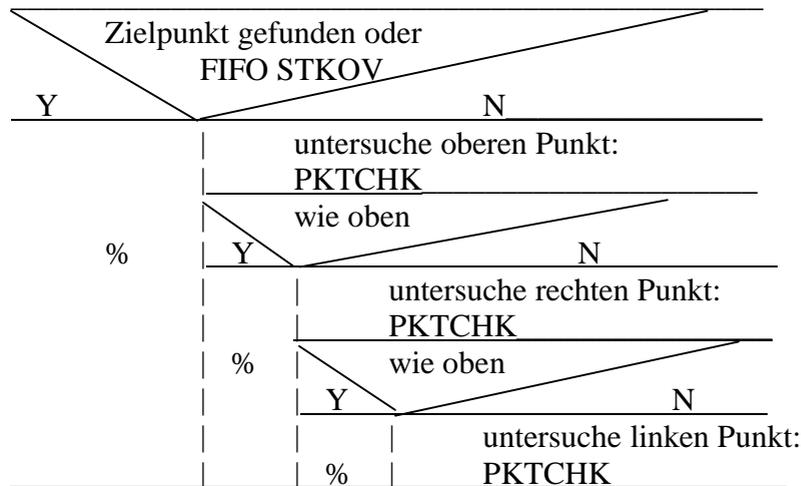


Bild 8b: NACHBR prüft die Punkte um die aktuelle Routerposition

TITLE : PKTCHK

Koordinaten in Adresse umwandeln

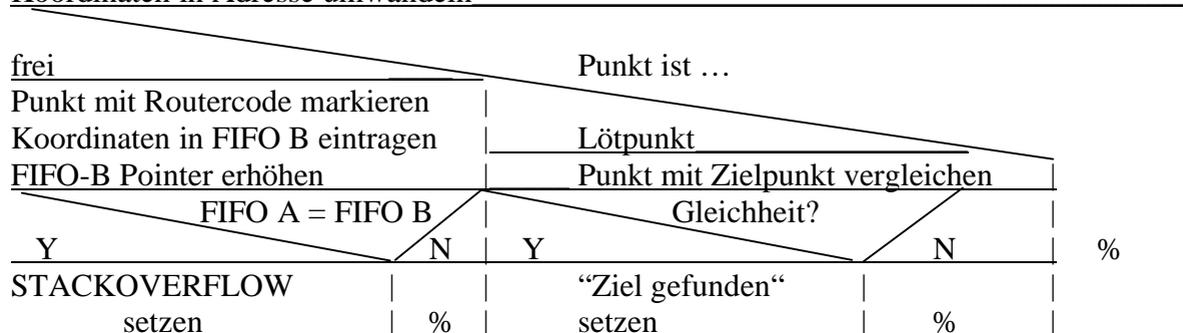


Bild 8c: PKTCHK prüft, ob Zielpunkt erreicht

Bild 9 : Ausschnitt aus der Matrix mit Start- und Zielpunkt

a)

|   | 0 | 1       | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---------|---|---|---|---|---|---|
| 0 |   | 2       | 1 | 3 | 1 | 2 |   |   |
| 1 | 2 | 1       | 3 | 2 | 3 | 1 | 2 |   |
| 2 | 1 | 3       | 2 | 1 | 2 | 3 | 1 | 2 |
| 3 | 3 | 2       | 1 | 3 | 1 | 2 | 3 | 1 |
| 4 | 2 | 1       | 3 | 2 | 3 | 1 | 2 | 3 |
| 5 | 1 | 3       | 2 | 1 | 2 | ∞ | 3 | 1 |
| 6 | 3 | 2       | 1 | ∞ | 1 |   |   | 2 |
| 7 | 1 | ∞-----+ |   |   |   |   |   | ∞ |

b)

|   | 0 | 1 | 2 | 3       | 4 | 5 | 6 | 7 |
|---|---|---|---|---------|---|---|---|---|
| 0 |   |   |   |         |   |   |   |   |
| 1 |   |   |   |         |   |   |   |   |
| 2 |   |   |   |         |   |   |   |   |
| 3 | 3 | 2 | 1 | 3       | 1 | 2 | 3 | 1 |
| 4 | 2 | 1 | 3 | 2       | 3 | 1 | 2 | 3 |
| 5 | 1 | 3 | 2 | 1       | 2 | ∞ |   | 1 |
| 6 | 3 | 2 | 1 | ∞       | 1 |   |   | 2 |
| 7 |   |   |   | ∞-----+ |   |   |   | ∞ |

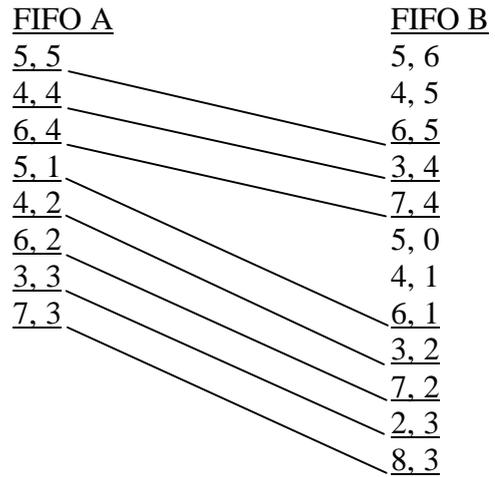
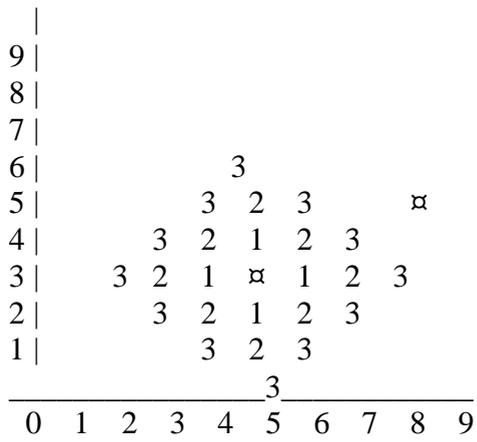
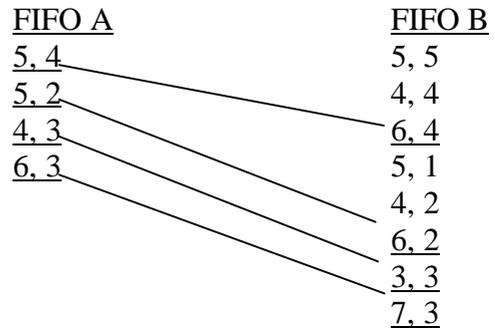
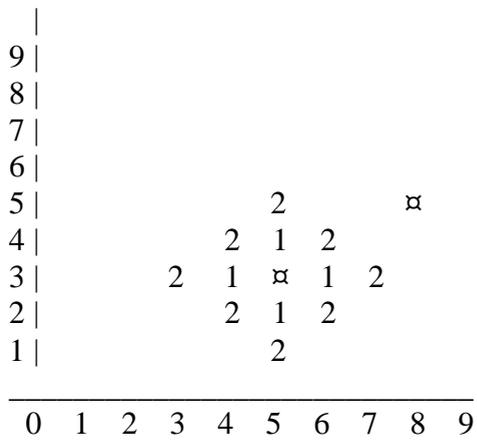
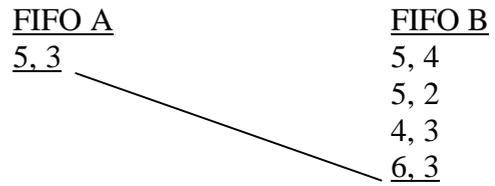
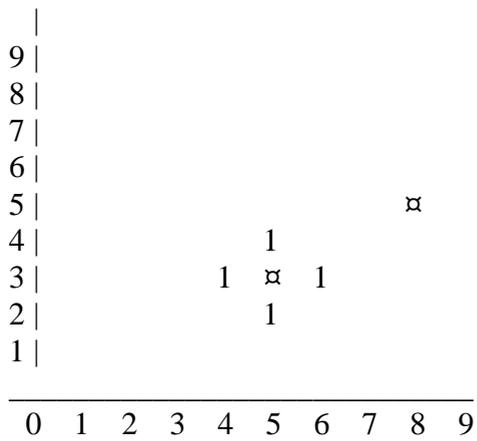
c)

|   | 0 | 1 | 2       | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---------|---|---|---|---|---|
| 0 |   |   |         |   |   |   |   |   |
| 1 |   |   |         |   |   |   |   |   |
| 2 |   |   |         |   |   |   |   |   |
| 3 | 3 | 2 | 1       | 3 | 1 | 2 | 3 | 1 |
| 4 | 2 | 1 | 3       | 2 | 3 | 1 | + | 3 |
| 5 | 1 | 3 | 2       | 1 | 2 | ∞ |   | 1 |
| 6 | 3 | 2 | 1       | ∞ | 1 |   |   | 2 |
| 7 |   |   | ∞-----+ |   |   |   |   | ∞ |

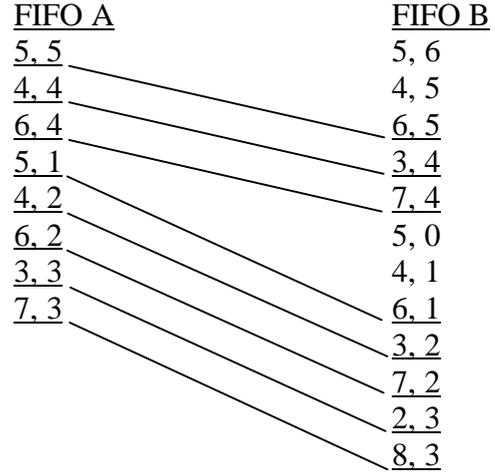
d)

|   | 0 | 1 | 2 | 3       | 4       | 5 | 6 | 7 |
|---|---|---|---|---------|---------|---|---|---|
| 0 |   |   |   |         |         |   |   |   |
| 1 |   |   |   |         |         |   |   |   |
| 2 |   |   |   |         |         |   |   |   |
| 3 | 3 | 2 | 1 | 3       | 1       | 2 | 3 | 1 |
| 4 | 2 | 1 | 3 | 2       | +-----+ |   |   | 3 |
| 5 | 1 | 3 | 2 | 1       |         | ∞ |   | 1 |
| 6 | 3 | 2 | 1 | ∞--+    |         |   |   | 2 |
| 7 |   |   |   | ∞-----+ |         |   |   | ∞ |

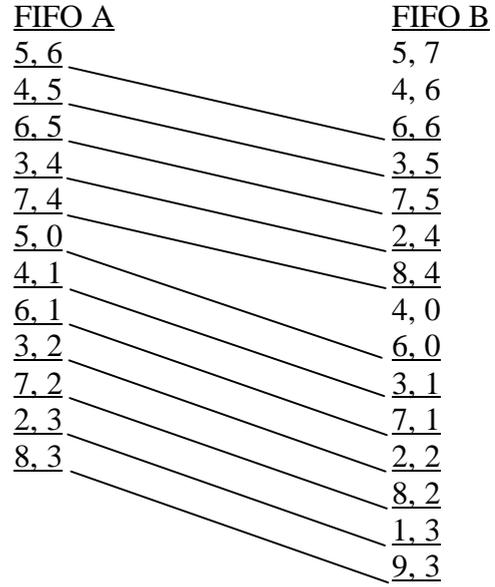
„Root“-Vorgang schematisch mit FIFO-Zuständen



|   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|
| 9 |   |   |   |   |   |   |   |   |   |   |
| 8 |   |   |   |   |   |   |   |   |   |   |
| 7 |   |   |   |   |   |   |   |   |   |   |
| 6 |   |   |   |   | 3 |   |   |   |   |   |
| 5 |   |   |   | 3 | 2 | 3 |   | ⊗ |   |   |
| 4 |   |   | 3 | 2 | 1 | 2 | 3 |   |   |   |
| 3 |   | 3 | 2 | 1 | ⊗ | 1 | 2 | 3 |   |   |
| 2 |   |   | 3 | 2 | 1 | 2 | 3 |   |   |   |
| 1 |   |   |   | 3 | 2 | 3 |   |   |   |   |
|   |   |   |   |   | 3 |   |   |   |   |   |
|   | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |



|   |   |   |   |   |   |   |   |   |   |   |
|---|---|---|---|---|---|---|---|---|---|---|
| 9 |   |   |   |   |   |   |   |   |   |   |
| 8 |   |   |   |   | 2 |   |   |   |   |   |
| 7 |   |   |   |   | 1 | 2 |   |   |   |   |
| 6 |   |   |   | 1 | 3 | 1 | 2 |   |   |   |
| 5 |   |   | 1 | 3 | 2 | 3 | 1 | ⊗ | 1 |   |
| 4 |   | 1 | 3 | 2 | 1 | 2 | 3 | 1 | 2 |   |
| 3 | 1 | 3 | 2 | 1 | ⊗ | 1 | 2 | 3 | 1 |   |
| 2 |   | 1 | 3 | 2 | 1 | 2 | 3 | 1 |   |   |
| 1 |   |   | 1 | 3 | 2 | 3 | 1 |   |   |   |
|   |   |   |   |   | 3 |   |   |   |   |   |
|   | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |



|   |   |   |   |   |   |     |    |   |   |   |
|---|---|---|---|---|---|-----|----|---|---|---|
| 9 |   |   |   |   |   |     |    |   |   |   |
| 8 |   |   |   |   |   |     |    |   |   |   |
| 7 |   |   |   |   | 1 |     |    |   |   |   |
| 6 |   |   |   | 1 | 3 | 1   |    |   |   |   |
| 5 |   |   | 1 | 3 | 2 | 3   | 1  | ⊗ |   |   |
| 4 |   | 1 | 3 | 2 | 1 | -2- | 3- | 1 |   |   |
| 3 | 1 | 3 | 2 | 1 | ⊗ | 1   | 2  | 3 | 1 |   |
| 2 |   | 1 | 3 | 2 | 1 | 2   | 3  | 1 |   |   |
| 1 |   |   | 1 | 3 | 2 | 3   | 1  |   |   |   |
|   |   |   |   |   | 3 |     |    |   |   |   |
|   | 0 | 1 | 2 | 3 | 4 | 5   | 6  | 7 | 8 | 9 |

