



Lijnen op het grafisch display

R. Koekoek

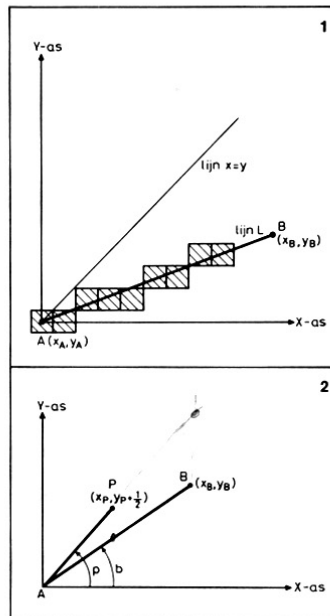
In een aantal artikelen wordt software besproken voor het grafisch display dat in Radio Bulletin is gepubliceerd. In dit artikel wordt een programma beschreven waarmee rechte lijnen op het grafisch display kunnen worden getekend. Alleen begin- en eindpunt van de lijn hoeven te worden ingevoerd, het programma tekent dan de lijn.

Achtergrond

Het grafisch display toont figuren door discrete punten aan of uit te zetten. Om een rechte lijn met een willekeurige hoek te tekenen moeten we daarom genoeg nemen met een zo goed mogelijke benadering met losse punten, zoals in afb. 1 getoond.

Het doel van het programma is lijnen door de computer te laten tekenen. Om eenvoudig te starten gaan we uit van een lijn die in de oorsprong begint (punt A) met de coördinaten X_a, Y_a en naar een punt B wordt getrokken met de coördinaten X_b, Y_b . Verder nemen we aan dat de hoek met de positieve x-as tussen de 0 en 45 graden ligt. Dit betekent dat $X_b > 0, Y_b > 0$ en $X_b > Y_b$.

Een lijn met de bovenstaande beperkingen kunnen we op het display tonen door naar rechts gaand vanuit de oorsprong punten wit te maken en, om de lijn te volgen, af en toe een punt omhoog te gaan. Het programma moet nu beslissen wanneer naar rechts en wanneer naar boven te gaan.



De volgende methode levert een goed resultaat, zoals in afb. 1 is te zien. We willen de ideale lijn L tussen A en B tekenen en beginnen door het punt in A aan te zetten. We gaan vervolgens een punt naar rechts. Als dit meer dan een halve lengte-eenheid (de afstand tussen twee punten op het display) van de ideale lijn is verwijderd, gaan we een punt omhoog. We herhalen dit proces tot we uiteindelijk in B aankomen.

Formule

We hebben een formule nodig om te kunnen bepalen of een punt wel of niet een halve lengte-eenheid boven de ideale lijn ligt, waarmee de computer gemakkelijk kan rekenen. Als we afb. 2 bekijken zien

we dat een punt P met coördinaten X_p, Y_p boven de lijn tussen A en B ligt als de hoeken p en b voldoen aan:

$$p > b$$

of, met de bekende definitie:

$$\tan(p) = \frac{X_p}{Y_p}$$

$$\tan(p) > \tan(b)$$

De beslissing of een punt een halve lengte-eenheid boven de ideale lijn ligt volgt nu uit:

$$\frac{Y_p + 1/2}{X_p} > \frac{Y_b}{X_b}$$

Deze vergelijking bevat delingen en dat is niet zo leuk voor de computer. Door beide zijden te vermenigvuldigen met $X_p \cdot X_b$ krijgen we:

$$(Y_p + 1/2) \cdot X_b > Y_b \cdot X_p$$

Alles naar links brengen en vermenigvuldigen met 2 levert:

$$2Y_p \cdot X_b + X_b - 2Y_b \cdot X_p > 0$$

Deze vergelijking bevat een aantal vermenigvuldigingen, wat een tijdrovende bezigheid is. De vergelijking is nog steeds niet erg geschikt. De linkerzijde noemen we nu T:

$$T = 2Y_p \cdot X_b + X_b - 2Y_b \cdot X_p$$

Voor het eerste punt op de lijn geldt dat $X_p = Y_p = 0$, zodat:

$$T = X_b \text{ (eerste punt)}$$

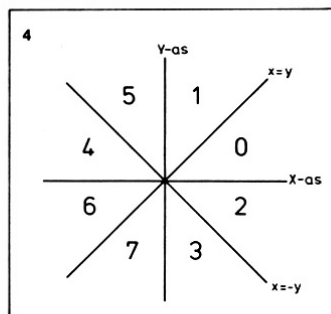
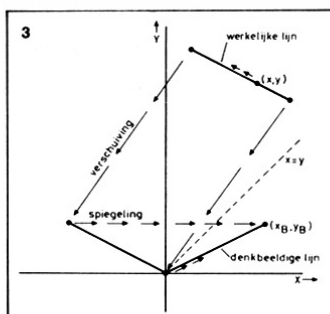
Voor het volgende punt zijn er twee mogelijkheden, namelijk dat we

Afb. 1 Benadering van een lijn met discrete punten op het grafisch display.

Afb. 2 Punt boven de ideale lijn met de hoeken p en b tussen x-as en lijnen vanuit oorsprong naar respectievelijk $(X_p, Y_p + 1/2)$ en eindpunt (X_b, Y_b) .



Grafisch display



een punt naar rechts gaan of een punt omhoog of met andere woorden:

1. X_p wordt met één opgehoogd, waardoor de T voor dit nieuwe punt (T_{nieuw}) uit de oude T (T_{oud}) kan worden berekend met:

$$T_{\text{nieuw}} = T_{\text{oud}} - 2Y_b$$

2. Y_p wordt met één opgehoogd, waardoor de T_{nieuw} volgt uit:

$$T_{\text{nieuw}} = T_{\text{oud}} + 2X_b$$

Door de T-waarde van het oude punt te onthouden kunnen we de T-waarde voor het nieuwe punt berekenen door optellen en aftrekken van de constanten $2 \cdot X_b$ en $2 \cdot Y_b$ zonder de coördinaten X_p , Y_p daarbij te betrekken.

Als T positief is, zal het punt met de coördinaten X_p , $Y_p + 1/2$ boven de ideale lijn liggen en is naar rechts gaan voldoende. Anders moeten we ook nog een punt omhoog gaan. Op

deze formule is het algoritme voor het trekken van lijnen gebaseerd. Het tekenen van een lijn gaat hiermee als volgt.

We beginnen in de oorsprong met het aanzetten van dat punt en gaan een punt naar rechts. Met de hier afgeleide formule beslissen we of we wel of niet ook een punt omhoog moeten gaan. De T-waarde en de coördinaten van elk getekend punt worden dan bijgewerkt en het punt getekend. We gaan hiermee door tot we het eindpunt hebben bereikt.

Willekeurige lijnen tekenen

Het algoritme dat we nu hebben ontwikkeld is alleen geschikt voor lijnen die in de oorsprong beginnen en een hoek tussen 0 en 45 graden met de positieve x-as maken.

Een willekeurige lijn kunnen we toch met dit eenvoudige algoritme te lijf. We gaan namelijk in een

coördinatenstelsel rekenen waarin de lijn aan de voorwaarden voldoet. Ten opzichte van het displaycoördinatenstelsel is dit reken-coördinatenstelsel gespiegeld en verschoven. Door het volgende punt in het reken-coördinatenstelsel te bepalen en daarna te vertalen naar het display-coördinatenstelsel is een algoritme voor willekeurige rechte lijnen gevonden.

In afb. 3 is een voorbeeld te zien van een dergelijke verschuiving en spiegeling om de lijn aan de voorwaarden te laten voldoen. In dit voorbeeld was na de verschuiving van het beginpunt naar de oorsprong een spiegeling voldoende.

Programma

In lijst 1 is de assemblerlisting van het programma „lijn” te vinden, in afb. 6 de opbouw in blokken van dit programma.

Het programma is opgezet als een subroutine.

Lijst 1

RL17N H3C0 81 6502 ASSEMBLER V2.0 PAGE 01

```

0010: *****
0020: *
0030: * LIJNEN OP HET GRAFISCH DISPLAY *
0040: *
0050: * R. KOEKOEK *
0060: *
0070: * COMPUTER BULLETIN *
0080: *
0090: *****
0100: ;
0110: 0400 RLJN ORG #0400 ;
0120: ;
0130: ; ZERO PAGE VARIABLEN :
0140: ;
0150: 0400 ZP #0090 ;
0160: 0400 XBEGIN * ZP +00 ; BEGIN PUNT LIJN
0170: 0400 YBEGIN * ZP +01 ;
0180: 0400 XEIND * ZP +02 ; EIND PUNT LIJN
0190: 0400 YEIND * ZP +03 ;
0200: 0400 XPUNT * ZP +04 ; HUIDIGE PUNT
0210: 0400 XPUNT * ZP +05 ;
0220: 0400 YPUNT * ZP +06 ;
0230: 0400 YPUNT * ZP +07 ;
0240: 0400 TL * ZP +08 ; T SOM
0250: 0400 TH * ZP +09 ;
0260: 0400 RICHT * ZP +0A ; RICHTING
0270: ;
0280: ; PIA ADRESSEN
0290: ;
0300: 0400 PIA * #1700 ;

```

```

0310: 0400 PRC * PIA +00 ; PIA A DATA
0320: 0400 PRD * PIA +01 ; PIA A DATA DIRECTION
0330: 0400 PRD * PIA +02 ; PIA B DATA
0340: 0400 PRD * PIA +03 ; PIA B DATA DIRECTION
0350: ;
0360: ; ROUTINE LIJN
0370: ;
0380: 0400 08 LIJN CLD ;
0390: 0401 A0 00 LDVH #00 ;
0400: 0402 04 95 STV XPUNT ; X,Y PUNT H := 0
0410: 0405 04 97 STV YPUNT ;
0420: 0407 04 99 STV TH ; T H := 0
0430: 0409 35 SEC ;
0440: 040A A5 92 LDA XEIND ; XPUNT := XEIND - XBEGIN
0450: 040C E5 90 SBC XBEGIN ;
0460: 040E 85 94 STA XPUNT ;
0470: 0410 80 00 BCS VBERG ; XPUNT NEGATIEF ?
0480: 0412 A9 01 LDVH #01 ; SPIEGELEN OM Y-AS
0490: 0414 E5 94 SBC XPUNT ;
0500: 0416 85 94 STA XPUNT ;
0510: 0418 A0 04 LDVH #04 ; SPIEGELEN ONTHOUDEN
0520: 041A A9 00 LDVH #00 ; XPUNT BEWAREN
0530: 041B 35 SEC ;
0540: 041C A5 93 LDA YEIND ; YPUNT := YEIND - YBEGIN
0550: 041E E5 91 SBC YBEGIN ;
0560: 0420 85 96 STA YPUNT ;
0570: 0422 80 00 BCS LZ ; YPUNT NEGATIEF ?
0580: 0424 A9 01 LDVH #01 ;
0590: 0426 E5 96 SBC YPUNT ; SPIEGELEN OM X-AS
0600: 0428 85 96 STA YPUNT ;
0610: 042A C8 INY ; SPIEGELING ONTHOUDEN
0620: 042B C8 INY ;
0630: 042C C5 94 CMP XPUNT ; XPUNT > YPUNT ?
0640: 042E 90 06 BCC LZ ;
0650: 0430 86 96 STX YPUNT ; XPUNT EN YPUNT VERWISSELEN
0660: 0432 85 94 STA XPUNT ;
0670: 0434 A9 00 LDVH #00 ;
0680: 0435 C8 INY ; VERWISSELING ONTHOUDEN
0690: 0436 B9 A3 04 LDRV TABEL ; HAAL RICHTING OP

```

Grafisch display

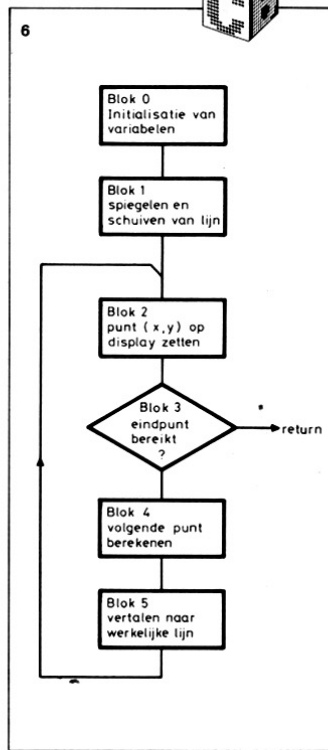
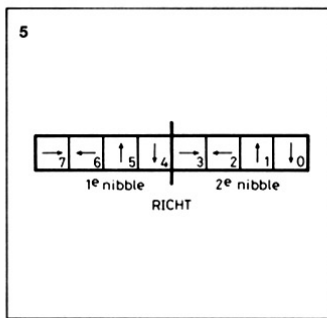
Afb. 3 Spiegeling en verschuiving om een willekeurige lijn om te vormen tot een lijn waarmee eenvoudig is te rekenen.

Afb. 4 Waarde in het Y-register die de spiegelingen aangeeft van lijnen tussen de aangegeven grenzen.

Afb. 5 Betekenis van de bits in de variabele RICHT.

Afb. 6 Opbouw van het programma in blokken.

Lijst 1 Listing van de subroutine „lijn”.



Het programma gebruikt de volgende variabelen:

$X_{\text{begin}}, Y_{\text{begin}}$ = coördinaten van het beginpunt van de lijn.

$X_{\text{eind}}, Y_{\text{eind}}$ = coördinaten van het eindpunt van de lijn.

$X_{\text{punt}}, Y_{\text{punt}}$ = het getal $2 \cdot X_b$.

Y_{punt}, T_H = het getal $2 \cdot Y_b$.

$T_L, T_H = T_{\text{som}}$ zoals boven beschreven.

RICHT = variabele waarin de spiegeling wordt onthouden.

X, Y = coördinaten van de werkelijke punt (registers).

Voor de X- en Y-coördinaten van begin- en eindpunt van de lijn zijn 8 bits getallen voldoende, het grafisch display is namelijk 256 bij 256 punten. De $T_{\text{som}}, X_{\text{punt}}$ en Y_{punt} nemen waarden tussen 0 en 65535 aan en daarom zijn daarvoor 16 bits getallen genomen.

In de blokken gebeurt het volgende:

Blok 0. Regel 380 – 420.

De initialisatie van variabelen gebeurt hier, zoals het nul maken van de meest significante bytes van $X_{\text{punt}}, Y_{\text{punt}}$ en T.

Het Y-register wordt ook nul gemaakt.

Blok 1. Regel 430 – 770.

Het doel van dit blok is het vertalen van de lijn in een lijn die in de oorsprong begint en een hoek tussen 0 en 45 graden met de positieve x-as maakt.

Begonnen wordt met schuiven: $X_{\text{punt}} (X_b, Y_b)$ van de lijn in het reken-coördinatenstelsel, wordt gelijkgesteld aan Eind – Begin.

Als de x-coördinaat van X_{punt} negatief is, moeten we om de Y-as spiegelen en onthouden dat in bit 2 van het Y-register. Als de Y-coördinaat negatief is, spiegelen we om de x-as en onthouden dat in bit 1 van het Y-register.

De hoek ligt nu tussen 0 en 90 graden en we zijn er bijna.

Als de Y-coördinaat groter is dan de x-coördinaat, moeten we om de lijn $x=y$ spiegelen om de hoek tussen 0 en 45 graden te krijgen. Deze spiegeling onthouden we in bit 0 van het Y-register.

Het Y-register bevat nu een getal

```

0700: 0439 85 9A      STA  RICHT ;
0710: 043B 06 96      RSL  YPUNT ; YPUNT := YPUNT * 2
0720: 043D 26 97      ROL  YPUNT ;
0730: 043F 86 98      STX  TL ; T := XPUNT
0740: 0441 06 94      RSL  XPUNT ; XPUNT := XPUNT * 2
0750: 0443 26 95      ROL  XPUNT ;
0760: 0445 96 90      LDY  YBEGIN ; BEGINPUNT OP DISPLAY ZETTEN
0770: 0447 94 91      LDY  YBEGIN ;
0780: ;
0790: ; PUNT X,Y OP DISPLAY ZETTEN
0800: ;
0810: 0449 80 02 17   WAIT LDA  PBD ; WACHT OP BLANKING
0820: 044C 10 FB      BPL  WAIT ; HOOG WORDT
0830: 044E 8E 00 17   STX  PBD ; X-COÖRDINAAT
0840: 0451 EE 02 17   INC  PBD ; 01 OP STUURUITGANG
0850: 0454 EE 02 17   INC  PBD ; 02 " "
0860: 0457 8C 00 17   STY  PBD ; Y-COÖRDINAAT
0870: 045A 99 08      LDYH  #8 ;
0880: 045C EE 02 17   INC  PBD ; 03 OP STUURUITGANG
0890: 045F 8D 02 17   STA  PBD ; INVERTEER COMMANDO
0900: 0462 E4 92      CPY  XEIND ; EINDPUNT BEREIKT ?
0910: 0464 D0 05      BNE  VERDER ;
0920: 0466 C4 93      CPY  YEIND ;
0930: 0468 D0 01      BNE  VERDER ;
0940: 046A 60        RTS ; EINDE LIJN IS BEREIKT
0950: ;
0960: ; VOLGENDE PUNT BEREKENEN
0970: ;
0980: 046B 38      VERDER SEC ;
0990: 046C 85 98      LDA  TL ; T := T - YPUNT
1000: 046E 85 96      SBC  YPUNT ; EEN PUNT NAAR RECHTS OVER LIJN
1010: 0470 85 96      STA  TL ;
1020: 0472 85 99      LDA  TH ;
1030: 0474 85 97      SBC  YPUNT ;
1040: 0476 85 99      STA  TH ;
1050: 0478 06        PHP ; BEWAAR SIGN BIT
1060: 0479 10 00      BPL  SKIP ; IS T NEGATIEF ?
1070: 047B 18        CLC ;
1080: 047C 85 98      LDA  TL ; T := T + XPUNT

```

```

1090: 047E 65 94      ADC  XPUNT ;
1100: 0480 85 98      STA  TL ;
1110: 0482 85 99      LDA  TH ;
1120: 0484 65 95      ADC  XPUNT ;
1130: 0486 85 99      STA  TH ;
1140: ;
1150: 0488 85 9A      JMP  SKIP ;
1160: 048A 28      BPL  RIGHT ; ALS NIET POSITIEF
1170: 048B 10 04      ASLA ;
1180: 048D 0A      ASLA ;
1190: 048E 0A      ASLA ;
1200: 048F 0A      ASLA ;
1210: 0490 0A      ASLA ;
1220: 0491 0A      ASLA ;
1230: 0492 90 01      BCC  LEFT ;
1240: 0494 E8      INX ; PLATS NAAR RECHTS
1250: 0495 0A      ASLA ;
1260: 0496 90 01      BCC  DOWN ;
1270: 0498 CA      DEX ; PLATS NAAR LINKS
1280: 0499 0A      ASLA ;
1290: 049A 10 01      BPL  UP ;
1300: 049C 86      DEY ; PLATS OMLAAG
1310: 049D 90 0A      BCC  WAIT ;
1320: 049F C8      INY ; PLATS OMHOOG
1330: 04A0 4C 49 04   JMP  WAIT ; ZET PUNT AAN
1340: ;
1350: ; TABEL MET RICHTING VAN LIJN
1360: ;
1370: 04A3 8A      TABEL = #8A ;
1380: 04A4 2A      = #2A ;
1390: 04A5 89      = #89 ;
1400: 04A6 19      = #19 ;
1410: 04A7 46      = #46 ;
1420: 04A8 26      = #26 ;
1430: 04A9 45      = #45 ;
1440: 04AA 15      = #15 ;
1450: ;
1460: ; EINDE LIJN

```



Grafisch display

Lijst 2 Voorbeeld van het gebruik van „lijn”

dat de richting van de werkelijke lijn aangeeft. In afb. 4 is aangegeven hoe dit getal samenhangt met de hoek met de positieve x-as. Met het Y-register als index wordt een getal uit TABEL opgehaald; de betekenis daarvan wordt straks duidelijk. Dit getal wordt opgeborgen in de variabele RICHT.

Omdat voor de berekening van T de waarden $2 \cdot X_b$ en $2 \cdot Y_b$ nodig zijn worden deze vast berekend door naar rechts schuiven van X_{punt} en Y_{punt} .

Het X- en Y-register krijgen de waarden van het beginpunt van de lijn. In het vervolg van de routine bevatten X- en Y-register altijd de coördinaten van het werkelijke aan de beurt zijnde punt.

Blok 2. Regel 810 – 890.

Hier wordt het punt op het display aangezet. Het X- en Y-register bevatten de coördinaten van het huidige punt op de lijn en met de stuurcodes van het grafisch display wordt dat punt op het display getekend. Hierbij is natuurlijk er van

uit gegaan dat de PIA's, waar het grafisch display op is aangesloten, al zijn geïnitieerd.

Blok 3. Regel 900 – 940.

De functie van dit blok is te beslissen of we de gehele lijn al hebben getekend. Als X- of Y-register gelijk is aan de x- of y-coördinaat van het eindpunt is het einde van de lijn bereikt en keren we uit de subroutine terug.

Blok 4. Regel 980 – 1130.

Het volgende punt op de lijn in het reken-coördinatenstelsel wordt nu berekend. Eerst gaan we een punt naar rechts en werken de T_{som} bij door $2 \cdot Y_b$ af te trekken. Als T daarvoor negatief wordt moeten we ook een punt omhoog gaan en brengen dat in rekening door bij T de waarde $2 \cdot X_b$ op te tellen. In het sign-bit onthouden we het al of niet omhoog gaan.

Blok 5. Regel 1150 – 1330.

De verplaatsing in het reken-coördinatenstelsel uit blok 4 moet nog worden vertaald in een verplaatsing over de lijn op het display.

Hierbij wordt de waarde van de variabele RICHT gebruikt. De waarde van RICHT was tot stand gekomen door de spiegelingen te onthouden en met die spiegelingen als index een constante uit een tabel op te halen. Als we over de lijn in het reken-coördinatenstelsel alleen naar rechts zijn gegaan gebruiken we het eerste nibble van RICHT en anders het tweede.

Wat zo een nibble voorstelt is in afb. 6 te zien. Een 1 op de plaats van een pijltje betekent dat het punt in de richting van dat pijltje

wordt verplaatst. De tabel is nu zo samengesteld dat elk nibble overeenkomt met de berekende spiegelingen.

De X- en Y-registers die de coördinaten van het punt op het display voorstellen worden met behulp van de nibble bijgewerkt om de verplaatsing over de reken-lijn te vertalen in een verplaatsing over de werkelijke lijn.

We springen nu naar blok 2 om het berekende punt te tonen en een eventueel volgend punt te berekenen.

Voorbeeld

Lijst 2 bevat een voorbeeld van het gebruik van de subroutine „lijn”. Daarbij worden lijnen getekend vanaf een punt naar de oorsprong (het middelpunt van het grafisch display). We beginnen in de rechter bovenhoek en schuiven punt voor punt naar links tot de linkerzijkant is bereikt. De subroutine „lijn” tekent een lijn door te invertieren wat een fraai interferentie patroon oplevert bij bijna overlappende lijnen.

In het voorbeeld worden eerst de PIA-registers geïnitieerd en het scherm schoongemaakt (regel 360 – 450).

Vervolgens worden de begin- en eind-coördinaten ingevuld en de lijnen getrokken waarbij het eindpunt naar links wordt geschoven.

De routine eindigt door naar de monitor te springen. Dit voorbeeld is op een KIM-1 geïmplementeerd, maar alleen de PIA-lokaties zijn daarvan afhankelijk.

Lijst 2

U00B8 HJC0 01 6502 ASSEMBLER V2.0 PAGE 01

```
0010: *****
0020: *
0030: * VOORBEELD VAN HET GEBRUIK VAN LIJN *
0040: *
0050: *****
0060:
0070:
0080: 0400 VOORS ORG #0400 ;
0090: ;
0100: ; ZERO PAGE GEBRUIK
0110: ;
0120: 0400 TEMP * #0000 ; TIJDELIJKE VARIABLE
0130: 0400 ZP * #0090 ;
0140: 0400 XBEGIN * ZP +00 ; BEGIN PUNT LIJN
0150: 0400 YBEGIN * ZP +01 ;
0160: 0400 XEIND * ZP +02 ; EIND PUNT LIJN
0170: 0400 YEIND * ZP +03 ;
0180: ;
0190: ; MONITOR ADRESSEN KIM
0200:
0210: 0400 SCANDS * #1F1F ; TOON DISPLAY
0220: 0400 MONIT * #1C22 ; MONITOR ENTRYPOINT
0230: ;
0240: ; PIA ADRESSEN
0250:
0260: 0400 PIA * #1700 ;
0270: 0400 PAD * PIA +00 ; PIA A DATA
```

```
0280: 0400 PADD * PIA +01 ; PIA A DATA DIRECTION
0290: 0400 PBD * PIA +02 ; PIA B DATA
0300: 0400 PBCD * PIA +03 ; PIA B DATA DIRECTION
0310: ;
0320: 0400 LIJN * #0400 ; SUBROUTINE LIJN
0330: ;
0340: ; VOORBEELD
0350: ;
0360: 0400 A2 FF EXAMP LDIXM #FF ; INITIALISEER I/O VAN DISPLAY
0370: 0402 0E 01 17 STX PADD ;
0380: 0405 A2 1F LDIXM #1F ;
0390: 0407 0E 03 17 STX PBD ;
0400: 040A 09 1C LDIXM #1C ; MARK SCHERM SCHOON
0410: 040C 05 00 STA TEMP ;
0420: 040E 00 02 17 STA PBD ;
0430: 0411 20 1F 1F CLS JSR SCANDS ; WACHT 2 MS
0440: 0414 C6 00 DEC TEMP ;
0450: 0416 D0 F9 BNE CLS ;
0460: ;
0470: ; TREK LIJN
0480: ;
0490: 0418 A2 00 LDIXM #00 ; BEGIN PUNT #00, #FF
0500: 041A 0E 02 17 STX PBD ;
0510: 041D 09 00 LDIXM #00 ;
0520: 041F 06 92 STX YEIND ;
0530: 0421 06 93 STX XEIND ; EIND PUNT #00, #00
0540: 0423 0A DEX ;
0550: 0425 06 91 STX YBEGIN ;
0560: 0427 05 90 STA XBEGIN ;
0570: 0429 20 00 04 LOOP JSR LIJN ; TREK LIJN
0580: 042B 06 92 INC XEIND ; XEIND := XEIND + 1
0590: 042D 00 F9 BNE LOOP ;
0600: 042F 4C 22 1C JMP MONIT ; GA DOOR TOT XEIND = 0
0610: ;
0620: ; EIND VOORBEELD
```