



computer bulletin

een supplement van RB,
gewijd aan microprocessoren
en aanverwante onderwerpen

NIEUWS

Onze rubriek microgebeuren vindt u weer op pagina's 40, 45 en 46

SOFTWARE

Een praktische toepassing van de microcomputer vindt u in het artikel 'Morse decoding met KIM'. De geseinde tekst verschijnt op het display van de KIM. Het is evenwel ook mogelijk met wat wijzigingen dit programma te laten lopen op andere 6502 machines. (blz. 36)

GRAFISCH DISPLAY

Door ruimtegebrek moest vorige maand de bij het programma behorende symbol table komen te vervallen. Deze symbol table, plus een correctie op een drukfout in lijst 1, vindt u op blz. 39

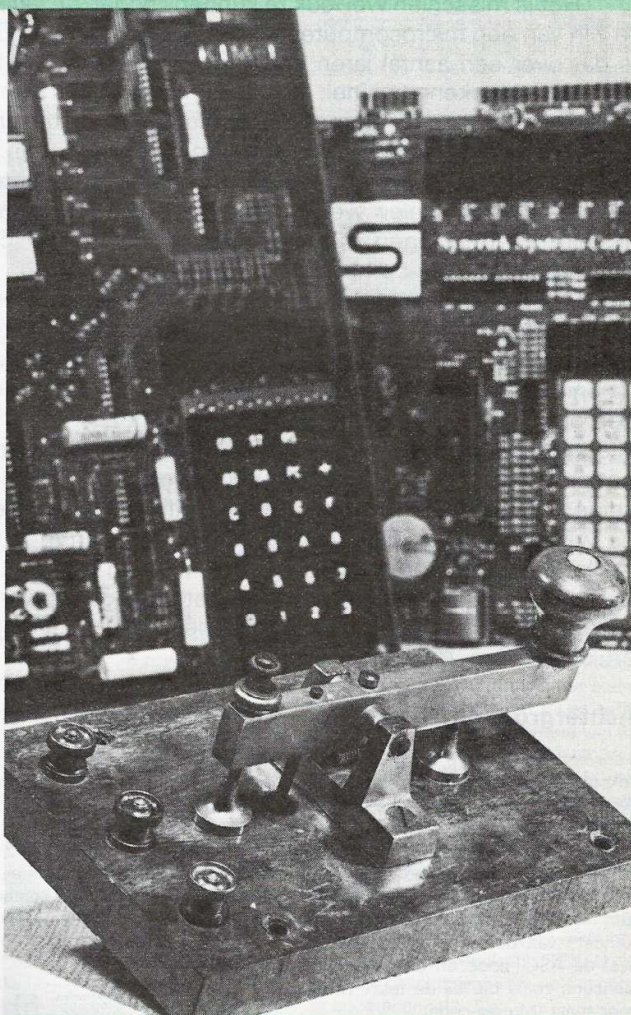
TEST

Deze maand komt het Heathkit H8 systeem aan de beurt. Zoals we van Heathkit gewend zijn géén compleet apparaat, maar een zéér goed gedocumenteerde bouwdoos. (blz. 41)

CURSUS

Het laatste deel van de cursus Basic kunt u vinden op blz. 47

Morsedecodering is voor de microcomputer een koud kunstje...





MORSE- DECODERING MET DE KIM

M. B. IMMERZEEL

Het is misschien wel leuk te kunnen zeggen de eigenaar te zijn van een microcomputer (als het zo doorgaat als nu dan is dat over een aantal jaren net zo gewoon als nu het bezit van een zakrekenmachine), nog leuker is het om ook programma's te bezitten en met het apparaat een aantal nuttige en vooral aangename uurtjes te kunnen doorbrengen. Het hiernavolgende programma dat voor de KIM is geschreven kan daartoe misschien wel bijdragen. Het schept de mogelijkheid om ontvangen morsesignalen als 'leesbaar schrift' zichtbaar te maken op het display van de KIM.

Hoewel met een zevensegmentdisplay lang niet alle lees- en lettertekens kunnen worden gevormd en dus voor deze tekens bepaalde symbolen moesten worden gevonden, is het na enig oefenen heel goed mogelijk de in morsecode verzonden berichten van het display te kunnen aflezen.

Niet alleen voor het weergeven van ontvangen berichten kan het programma worden gebruikt, maar ook bij het oefenen in seinen en bij het controleren van het seinschrift. Met hetgeen wat door het display wordt weergegeven kan men vaststellen welke fouten er bij het seinen worden gemaakt.

Een variant op dit programma, nl. het weergeven van het morseschrift met een telex, kan in een eventueel volgend artikel worden verwacht.

Achtergronden

Voor het overbrengen van geschreven teksten langs mechanisch-elektrische weg (telex) of via moderne verbindingswegen (teletext of viewdata bijv.) is het noodzakelijk een codesysteem toe te passen waarbij de schrijf- en leestekens elk door een apart codeteken worden weergegeven. De bekendste van deze codesystemen zijn wel de ASCII-code en de vijf-eenheden-code zoals die bij de telex wordt toegepast (Murray-code).

Het voordeel van deze systemen is dat alle codetekens even lang zijn en elk codeteken in een gelijk aantal, even

lange 'eenheden' worden verdeeld (bij telex vijf) die ook wel 'elementen' worden genoemd. Elk element kan dan 'stroomvoerend' of 'stroomloos' zijn, vergelijkbaar met de '1' en de '0' van het binaire systeem.

De specifieke kenmerken van dit systeem, nl. dat elk codeteken steeds uit een vast aantal elementen bestaat van gelijke lengte en die slechts '0' of '1'

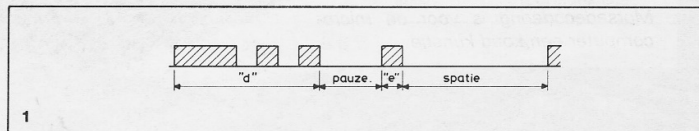
kunnen zijn, maken het voor de machine betrekkelijk gemakkelijk om uit de aangeboden codetekens de juiste schrifttekens te herkennen.

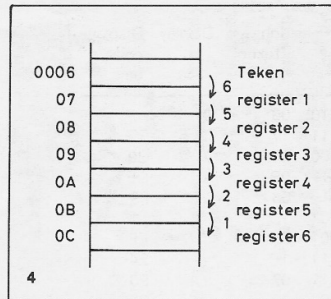
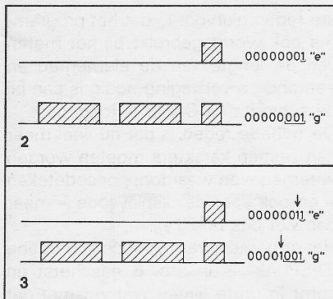
Geheel anders ligt het bij het morsecodesysteem. Mogelijk kan worden gesteld dat dit systeem eenvoudiger door de mens met zijn gehoor te analyseren is dan de hiervoor genoemde codes, voor de machine is het echter moeilijk te verteren dat de codetekens onregelmatig zijn voor wat betreft het aantal elementen (van één punt voor de 'e' tot zes punten voor het vergissingsteken) en voor wat betreft de lengte van de elementen. De strepen in deze codetekens zijn immers drie keer zo lang als de punten.

Het zijn niet alleen de punten en de strepen die het teken bepalen. Tussen de elementen van het teken komt een stilte voor die we hier verder 'rust' zullen noemen en waarvan de lengte gelijk aan die van een punt dient te zijn. Tussen de codetekens dient een langere pauze te worden aangehouden. Deze pauze moet even lang duren als een streep.

Een codeteken voor spatie wordt in het algemeen niet verzonden. Hiervoor wordt gewoon een langere pauze gebruikt die tenminste een lengte van vijf punten moet hebben (voorgeschreven is nu zeven punten, vroeger paste men vijf punten toe).

Samengevat ziet het woord 'de' in morsetekens eruit zoals in afb. 1. De 'd' is een streep met twee punten. De rusten zijn één punt lang en de streep drie punten. De 'e' is slechts één punt en wordt met een pauze met een leng-





te van drie punten gescheiden van de 'd'. Het volgende woord zal slechts beginnen na een pauze met een tijdsduur van zeven punten.

Een ander probleem dat om een oplossing vraagt is de seinsnelheid die voor iedere telegrafist weer anders ligt. Het is dan ook noodzakelijk om eerst de lengte van een punt te bepalen. Dit is slechts mogelijk nadat zowel een punt als een streep is ontvangen. Dit programma zal daarom het eerste schriftteken (of in enkele gevallen de eerste paar schrifttekens) niet of anders verminkt weergeven. Bezwaarlijk kan dat niet zijn. Elke uitzending wordt nl. voorafgegaan door een oproep die enkele keren wordt herhaald.

Als een seinschrift niet door een machine is opgewekt zal het in bijna alle gevallen er niet zo uitzien als is voorgeschreven. We zullen er rekening mee moeten houden dat een streep niet altijd drie punten lang is. Bij dit programma is de lengte van twee punten als grens genomen, d.w.z. dat elk element dat langer is dan twee punten als een streep wordt herkend. Dat betekent dan wel dat een seiner niet meer kan worden 'genomen' als hij punten seint die meer dan twee keer zo lang zijn als eerder geseinde punten.

Verder dient het programma rekening te houden met onregelmatigheden in de seinsnelheid. Het verhogen van de seinsnelheid geeft hier geen problemen. Het verlagen van de seinsnelheid moet in verband met het voorgaande echter zo zijn dat de lengte van de eerstvolgende punt in elk geval kleiner moet zijn dan twee maal de lengte van de voorgaande.

Om deze veranderingen van de seinsnelheid te kunnen blijven volgen is het noodzakelijk om steeds de lengte van de laatst ontvangen punt te blijven onthouden. Het kan echter voorkomen dat een ongecontroleerde sleutelbe-

weging van een minder ervaren seiner een zeer kort contact kan veroorzaken hetgeen dan wordt ervaren als een zeer korte punt. Duurt deze 'punt' korter dan de helft van de daarop volgende punten dan zullen alle volgende code-elementen voor strepen worden aangezien zodat de werking van het programma dan geheel in de war geraakt.

Deze storing kan worden voorkomen door de tijdsduur van elke ontvangen punt te middelen met die van de voorgaande. Op deze manier worden de uitschieters 'gladgestreken' en hebben onregelmatigheden minder invloed op de juiste werking.

Gesteld is reeds dat de seinsnelheid onbeperkt kan worden verhoogd, maar niet onbeperkt kan worden verlaagd. Des ondanks kan een spronggewijze snelheidsverandering redelijk worden opgevangen. Rekening houdend met het hiervoor genoemde 'middelen' kan dan ook een stap van bijv. 23 naar 16 woorden per minuut zonder bezwaar worden toegestaan. Regelmatige snelheidsverminderingen kunnen dan ook geen moeilijkheden opleveren.

Het display

Dit programma is gemaakt om de morsetekens als leestekens weer te geven op het KIM-display. De moeilijkheid hierbij is echter dat het slechts een zeven-segmentdisplay is en dus bepaalde letters en leestekens niet kan weergeven. Voor deze letters en leestekens (k;m;n;v;x;?;komma;punt en punt-komma) zijn daarom een aantal symbolen verzonden die zoveel mogelijk op de oorspronkelijke letters en tekens gelijken. Gebleken is dat na een betrekkelijk korte oefentijd te teksten, ondanks de vreemde symbolen, ook bij hoge seinsnelheden soepel zijn te lezen. Het display geeft de teksten weer zoals dat bij een lichtkrant plaatsvindt.

Een verschil is echter dat de tekst steeds één plaats verspringt.

De tekens verschijnen na elkaar op het meest rechtse display en verschuiven na ontvangst en vertaling van het volgende morseteken steeds één plaats op naar links. Een spatie wordt weergegeven door het betreffende display uit te laten. In lijst 1 zijn o.a. de gebruikte leestekens met de daarbij behorende morsetekens en displaysymbolen gegeven.

Voor het vertalen van de morsetekens wordt een punt gelezen als '1' en een streep als '0'. Ook voor de KIM is het nodig om over te gaan op codetekens die alle een gelijk aantal elementen bevatten. Hier is het aantal elementen: 8, nl. gelijk aan het aantal bits per geheugenplaats. Doordat alle lege geheugenplaatsen van een '0' zijn voorzien zouden, indien de morsetekens zonder meer in enen en nullen werden vertaald, fouten ontstaan. Er zou dan bijv. geen onderscheid meer zijn tussen een 'e' en een 'g' omdat de strepen ('0') en de lege geheugenplaatsen (ook '0') niet meer te onderscheiden zijn. In afb. 2 is weergegeven hoe een 'e' en een 'g' in de KIM zouden worden gelezen. Door de tekens vooraf te markeren met een '1' wordt deze fout voorkomen. In afb. 3 is weergegeven hoe de 'e' en de 'g' door de KIM worden vertaald. De markeringen zijn met een pijltje gemerkt.

In lijst 1 zijn onder de kolom 'TEKEN' de codetekens weergegeven waarin de KIM het morseteken vertaalt. Elk morseteken wordt dus in een binair getal vertaald waarvan de kolom 'teken Hex' de hexadecimale waarde weergeeft. Op het adres 0006 vinden we de geheugenplaats 'TEKEN'. Hierin worden na elkaar de elementen van het code-teken geschoven. Als het teken compleet is wordt het in het eerste van de zes registers overgebracht die worden gebruikt bij het monitorprogramma. Hiervoor zijn de geheugenplaatsen 0007 t/m 000C gereserveerd (afb. 4). De subroutine 'SCHUIF' wordt dan in werking gesteld waardoor, te beginnen bij register 5, de inhoud één plaats worden opgeschoven en zodoende het lichtkranteffect wordt bereikt. De geheugenplaats 'TEKEN' is nu weer beschikbaar voor het volgende codeteken (afb. 5).

Het was niet mogelijk van de subroutine 'SCANDS' van het KIM-monitorprogramma gebruik te maken. De eer-



SOFTWARE morse decodering

Lijst 1

Teken	Morse-code	Inh. 'teken' binair	Inh. 'teken' hex.	Display	Display-code hex.
A	· · · ·	00000110	06	8	F7
B	· · · · ·	00010111	17	8	FC
C	· · · · ·	000101001	15	8	B9
D	· · · · ·	00001011	0B	8	DE
E	·	00000011	03	8	F9
F	· · · ·	00011101	1D	8	F1
G	· · · ·	00001001	09	8	EF
H	· · · · ·	00011111	1F	8	F4
I	· ·	00000111	07	8	90
J	· · · · ·	00011000	18	8	8E
K	· · · ·	00001010	0A	8	F0
L	· · · · ·	00011011	1B	8	B8
M	· ·	00000100	04	8	B7
N	· ·	00000101	05	8	D4
O	· · · ·	00001000	08	8	DC
P	· · · · ·	00011001	19	8	F3
Q	· · · · ·	00010100	12	8	E7
R	· · · ·	00001101	0D	8	D0
S	· · · ·	00001111	0F	8	ED
T	·	00000010	02	8	F8
U	· · · ·	00001110	0E	8	9C
V	· · · · ·	00011110	1E	8	BE
W	· · · · ·	00001100	0C	8	FE
X	· · · · ·	00010110	16	8	F6
Y	· · · · ·	00010010	14	8	EE
Z	· · · · ·	00010011	13	8	DB
1	· · · · ·	00110000	30	8	86
2	· · · · ·	00111000	38	8	DB
3	· · · · ·	00111100	3C	8	CF
4	· · · · ·	00111110	3E	8	E6
5	· · · · ·	00111111	3F	8	ED
6	· · · · ·	00101111	2F	8	FD
7	· · · · ·	00100111	27	8	87
8	· · · · ·	00100011	23	8	FF
9	· · · · ·	00100001	21	8	EF
0	· · · · ·	00100000	20	8	BF
?	· · · · ·	01110011	73 (33)	8	A7
,	· · · · ·	01001100	4C (2B)	8	8C
/	· · · · ·	00101101	2D	8	D2
.	· · · · ·	01101010	6A (1A)	8	88
:	· · · · ·	01000111	47 (26)	8	89
—	· · · · ·	01011110	5E (3D)	8	C0
=	· · · · ·	00101110	2E	8	C—
spatie	· · · · ·	00110110	36	8	80
begin	· · · · ·	00101010	2A	8	D0
eind	· · · · ·	00110101	35	8	C4
sluiten	· · · · ·	01111010	7A (3A)	8	80
apostroph	· · · · ·	01100001	61 (11)	8	82
(	· · · · ·	01010010	52 (31)	8	8F
wacht	· · · · ·	00110111	37	8	C9
	· · · · ·	01010101	55 (34)	8	8D

Lijst 2

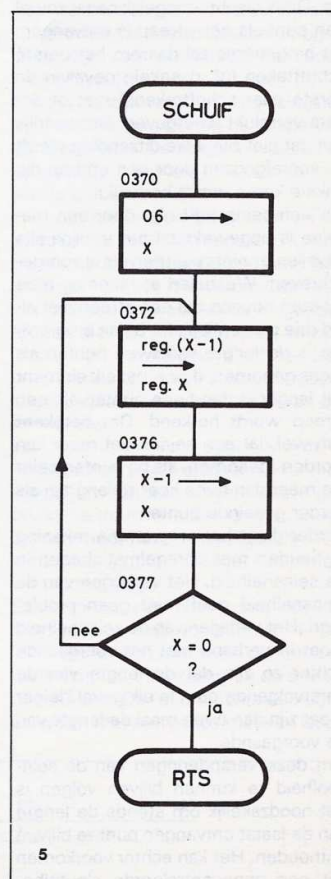
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0																
1		(A ₀)	Q	Z	Y	C	X	B	J	P	(.)	L	F	V	H	
2	0	9		8		(:)	7			Be	(.)	/	=	6		
3	1	(I)	(7)	(:)	E ₁	S _p	W _a	2		(S ₁)	3	(-)	4	5		
4																
5																
6		A ₀														
7											S ₁					

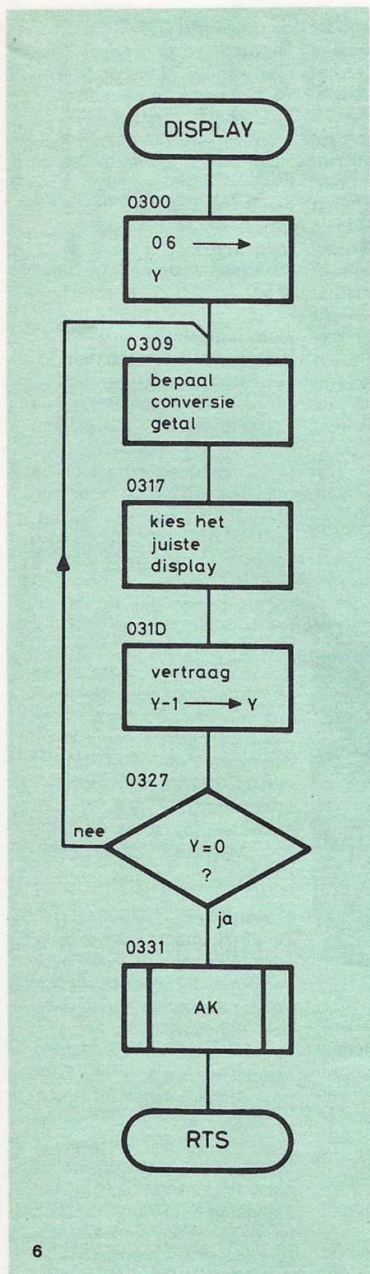
ste reden hiervoor is dat het programma ook wordt gebruikt bij het meten van de lengte van de elementen en een andere vertraging nodig is dan bij de subroutine CONVD is toegepast.

De tweede reden is dat nu veel meer dan zestien karakters moeten worden weergegeven waardoor per codeteken – en ook voor de displaycode – meer dan vier bits nodig zijn.

Het stroomdiagram voor de subroutine 'DISPLAY' is in afb. 6 geschetst en komt in grote lijnen overeen met dat van 'SCANDS'.

De kolom 'inhoud teken Hex' van lijst 1 geeft als laagste waarde Hex 02 voor de 'T' en heeft Hex 7A voor 'sluiten' als hoogste waarde (zie hiervoor ook lijst 2). Dit zou betekenen dat voor de tabel voor de displaycode bijna 128 geheugenplaatsen nodig zijn met meer dan 64 open plaatsen (er zijn 51 verschillende codetekens). De omvang van de





tabel kan worden beperkt door bepaalde tekens te 'verplaatsen'. Uit lijst 2 volgt dat hiervoor de tekens op rij 7 moeten worden verminderd met Hex 40, die op rij 6 met Hex 50 en op rij 5 en 4 met Hex 21. In lijst 1 zijn de nieuwe waarden tussen haakjes geplaatst. De displaycodetabel staat in de geheugenplaatsen 02C0-02FF.

(wordt vervolgd)



MORSE- DECODERING MET DE KIM

M.B. IMMERZEEL

Het hoofdprogramma

Het stroomdiagram van het hoofdprogramma vinden we in afb. 7. Het startadres is 0200 en de seinsleutel komt parallel aan één van de contacten van het toetsenbord (bijv. A17 en A18 van de applicatiesteker). Na het starten van het programma door op de toets 'go' te drukken worden eerst de in- en uitgangsbuffers geïnitieerd en bepaalde hulpregisters gereset. De displayregisters worden geladen met het codeteken voor '=' waarna het programma blijft wachten tot de toets 'go' wordt losgelaten. Na het loslaten wacht het programma op het eerste code-element. Bij deze laatste twee programmastappen wordt de displaysubroutine gebruikt zodat het display oplicht en '=====' laat zien. Direct na het indrukken van de seinsleutel, dus bij de start van een element, dooft het rechte display omdat 'teken' wordt geladen met het codeteken voor 'spatie' (3A) dat middels subroutine 'schuif' in register 1 komt. De programmastap 'reset teken' laadt 'TEKEN' met Hex 01 voor het markeren.

Voor het meten van de lengte van het element wordt eveneens gebruik gemaakt van de subroutine 'DISPLAY', die een aantal malen wordt doorlopen en tijdens elke cyclus de teller 'ELEMEN' verhoogt (afb. 8). Het displayprogramma moet daarvoor een bepaalde lengte hebben en de geheugenplaats 031E is voor de juiste vertraging daarom geladen met Hex FF.

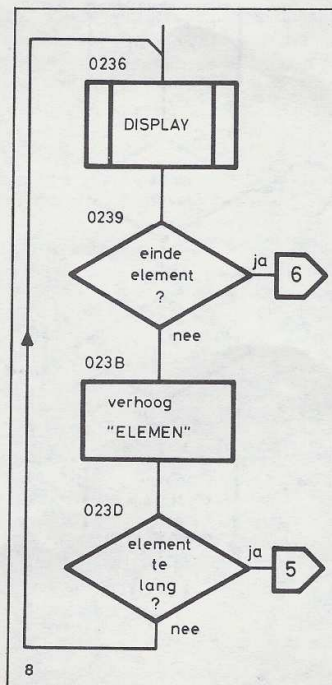
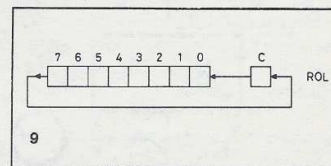
Bij hoge seinsnelheden (30 woorden per minuut en sneller) kan het beter zijn dit adres te laden met Hex 7F. De vertraging wordt daardoor kleiner zodat in een bepaalde tijd een groter aantal metingen plaatsvindt.

Wordt de seinsleutel te lang achtereenvolgend ingedrukt gehouden dan volgt er een

foutmelding (het achtste bit van 'elementen' wordt 1).

Na het eind van het element, aan het begin van de hierop volgende rust dus, wordt nagegaan of het element een punt of een streep was. Dit is na ontvangst van het eerste element uiteraard niet mogelijk. Pas na het ontvangen van twee elementen van verschillende lengte kan deze meting plaats vinden. De subroutine 'VERGELIJK' (zie later) zorgt er voor dat de lengte van een punt wordt berekend. De geheugenplaats 'LENGE' (0000) wordt dan gevuld met twee keer de lengte van een punt. De inhoud van 'ELEMEN'

wordt hiermee vergeleken. Is het element langer (streep) dan volgt CLC, anders SEC. De programmastap 'ROL teken' schuift nu respectievelijk een 0 of een 1 in het eerste bit van 'TEKEN', terwijl alle andere bits naar links opschuiven (afb. 9). Verder zal, indien het element een punt blijkt te zijn, subrou-

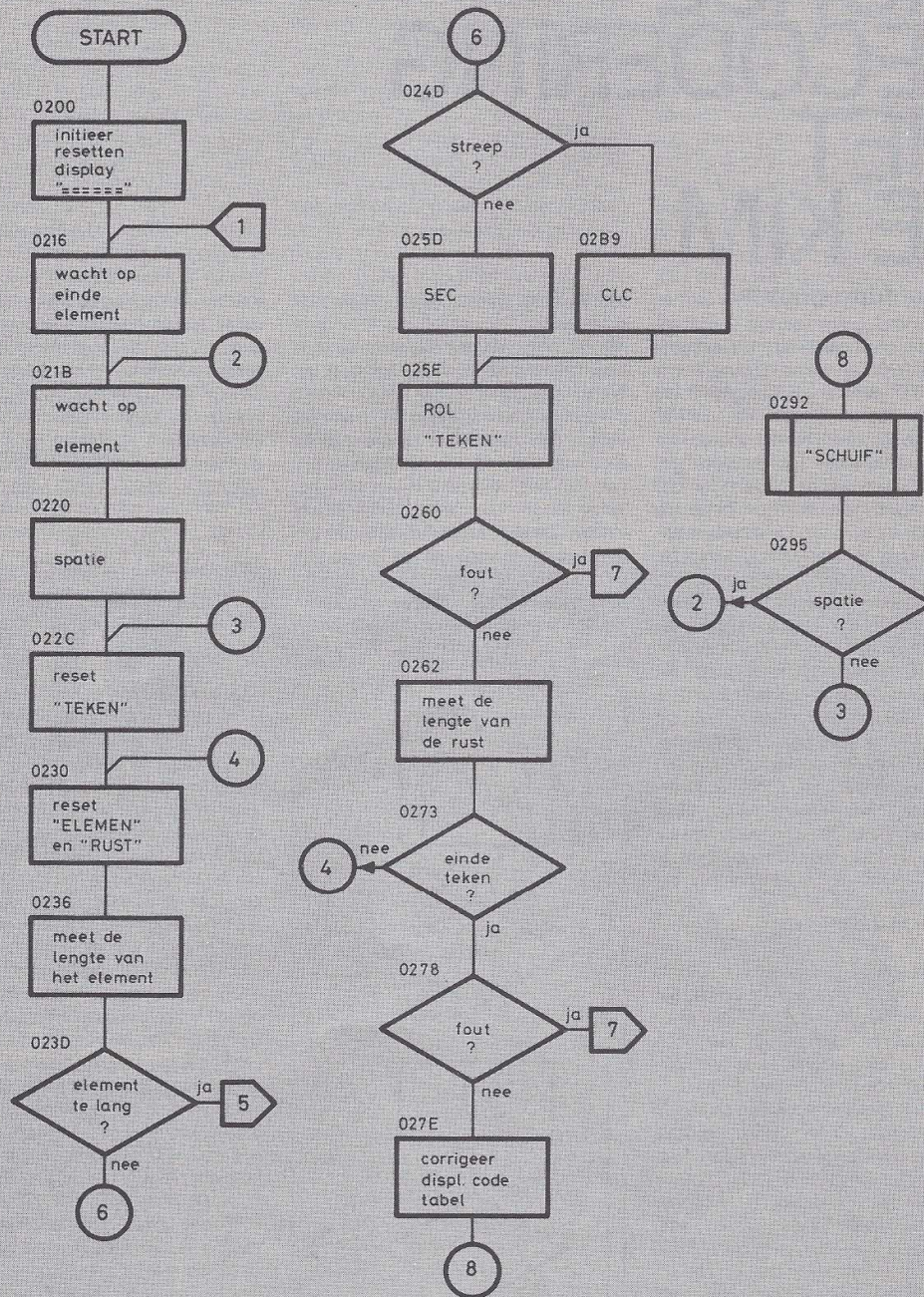


tine 'MIDDEL' de gemiddelde waarde berekenen van de nieuwe lengte van de punt met de voorgaande (afb. 10). Het grootste aantal elementen waaruit een morseteken kan bestaan is zes. Voegen we daar het markeringsteken bij dan zijn dus maximaal zeven bits nodig van de acht bits van geheugenplaats 'teken'. Komt het markeringsteken toch in de plaats van het achtste bit dan is er dus sprake van een fout en volgt er een foutmelding. Deze fout zal voorkomen bij telegrafisten die niet genoeg ruimte houden tussen de morsetekens en zo alles aan elkaar 'plakken'. De lengte van de rust wordt op overeenkomstige wijze bepaald als de lengte van het element. Het stroomdiagram is dan ook gelijk aan dat in afb. 8. Voor element en 'ELEMEN' moet nu echter 'RUST' worden gelezen. Als teller dient nu geheugenplaats 'RUST' op adres 0005.

In theorie is de lengte van de rust gelijk aan die van de punt. Beter is het echter daarmee niet te rekenen en steeds de lengte van een rust te middelen met de voorgaande, zoals dat ook met de punt gebeurt. Bij dit programma wordt deze methode dan ook toegepast, zij



SOFTWARE morse decoding



7



het dat bij het eerste teken, als de lengte van de rust nog niet bekend is, de lengte van de punt als norm wordt gesteld. In de praktijk blijkt dit goed te voldoen en garandeert dat binnen de kortst mogelijke tijd het eerste juiste karakter op het display verschijnt. De geheugenplaats 'LENTE' op adres 0003 wordt dan gevuld met twee maal de lengte van een rust. De inhoud van 'RUST' wordt hiermee vergeleken en het zal dan blijken of het werkelijk een rust is dan wel een pauze.

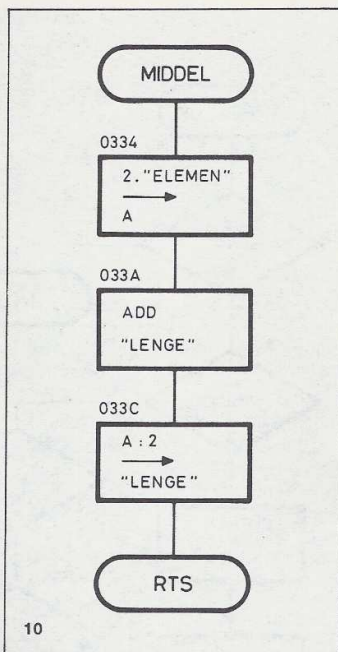
In het geval van een rust (einde teken is: neen) wordt na het resetten van 'ELEMEN' en 'RUST' overgegaan tot het meten van het nieuwe element dat nu is begonnen.

In het geval van een pauze is het code-teken compleet (einde teken). Evenals bij het element kan deze pauze erg lang worden aangehouden waardoor de teller 'RUST' gaat vollopen. Dit is echter geenszins als een fout aan te merken en in dit geval geeft de KIM automatisch een spatie.

De volgende programmastap bepaald of de telegrafist een foutmelding heeft gedaan (zes punten). Zo ja dan geeft het display 'FFFFFF'. In het andere geval wordt nagegaan of de inhoud van 'TEKEN' moet worden gewijzigd om de omvang van de displaycodetabel te beperken (lijst 2), waarna de subroutine 'SCHUIF' de inhoud van de display-registers opschuift en de inhoud van 'TEKEN' in het eerste displayregister brengt.

Wat de lengte van de pauze betreft zijn er nu nog twee mogelijkheden:

- De pauze duurt ongeveer drie keer zo lang als de rust. In dit geval wordt na het resetten van 'TEKEN', 'ELEMEN' en 'RUST' overgegaan tot het meten van het eerste element van het volgende codeteken dat nu al is begonnen.
- De pauze is langer (zeven maal zo



10

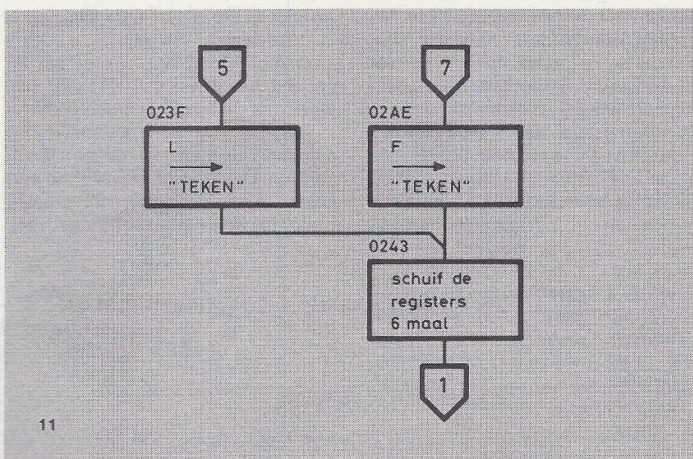
lang als de rust of veel langer). Dit betekent het eind van een woord. Het programma blijft nu eventueel wachten op het eerste element van het volgende codeteken en zal dan, na eerst een spatie te hebben gegeven resetten als onder a en de lengte van dat element gaan meten.

Als het goed is duurt de pauze tussen twee letters van een woord ongeveer

even lang als drie rusten. Deze pauze kan dan worden gevonden door te gaan vergelijken met vier maal de lengte van een rust. De meeste telegrafisten maken deze pauze echter te lang (uit angst te gaan 'plakken'?). Het gevolg zou kunnen zijn dat de KIM na elke letter een spatie geeft. Het is daarom beter te gaan vergelijken met zes maal rust. Hiertoe dient de subroutine 'LANG' op adres 038B, die kan worden aangeroepen door Hex 8B te plaatsen op adres 0298.

Voor hen die het seinen met de KIM willen beoefenen is het echter ook mogelijk te vergelijken met vier rusten ('kort') of vijf rusten ('midden') door respectievelijk Hex 7C of 80 op adres 0298 te plaatsen.

In het voorgaande was drie keer sprake van een foutmelding, één keer voor het te lang indrukken van de seinsleutel, dan voor een te groot aantal elementen in 'TEKEN' in daarna voor het seinen van het codeteken voor 'fout'. In het eerste geval laat het display 'LLLLLL' zien en in de andere gevallen



11



'FFFFFF'. Hiertoe wordt het betreffende codeteken in 'TEKEN' geladen en door het zes maal doorlopen van de subroutine 'SCHUIF' in alle zes de displayregisters gebracht. Het programma gaat nu terug naar 'wacht op einde element' (afb. 11).

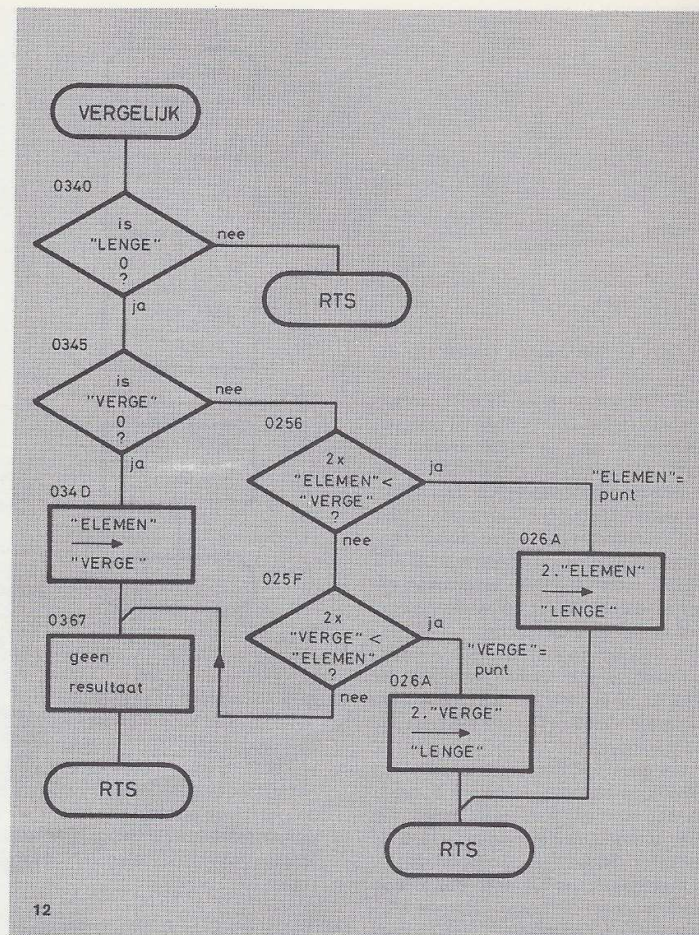
De subroutine 'vergelijk'

De subroutinevergelijk dient uitsluitend om bij het ontvangen van de eerste code-elementen te bepalen welk element een punt is. Is de lengte van een punt eenmaal bekend en 'LENGE' is geladen met een waarde die twee maal zo groot is dan doet deze subroutine geen dienst meer. Direct bij de start van de subroutine wordt dan ook vastgesteld of 'LENGE' geladen is ja of nee (afb. 12).

Na ontvangst van het eerste element zijn 'LENGE' en 'VERGE' nog leeg en wordt de inhoud van 'ELEMEN' in 'VERGE' (0001) geladen. Uiteraard is het nog niet bekend of dit een punt of een streep is. De inhoud van 'VERGE' wordt verder gebruikt om de lengte van het volgende element of volgende elementen hiermee te vergelijken. De programmastap 'geen resultaat' stelt vast dat de lengte van de punt nog niet bekend is waarna wordt teruggesprongen naar 'wacht op element'.

Na ontvangst van het tweede element is de situatie dus zo dat in 'VERGE' de lengte van het eerste element is vastgelegd en in 'ELEMEN' de lengte van de tweede. Er zijn nu vier mogelijkheden:

- | | | | |
|------------------|---|---|---|
| a. VERGE: punt | { | $2 \times \text{ELEMEN} > \text{VERGE}$ | $2 \times \text{VERGE} > \text{ELEMEN}$ |
| ELEMEN: punt | | | |
| b. VERGE: punt | { | $2 \times \text{ELEMEN} > \text{VERGE}$ | $2 \times \text{VERGE} < \text{ELEMEN}$ |
| ELEMEN: streep | | | |
| c. VERGE: streep | { | $2 \times \text{ELEMEN} < \text{VERGE}$ | |
| ELEMEN: punt | | | |
| d. VERGE: streep | { | $2 \times \text{ELEMEN} > \text{VERGE}$ | $2 \times \text{VERGE} > \text{ELEMEN}$ |
| ELEMEN: streep | | | |



Duidelijk is dat alleen in de situaties b en c de lengte van een punt wordt gevonden en 'LENGE' wordt gevuld met een waarde die overeenkomt met twee maal de lengte van een punt.

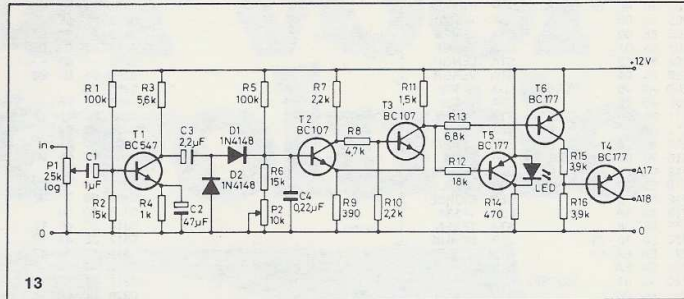
In de gevallen a en d wordt geen resultaat bereikt en wordt gewacht op een volgend element. Dit zal zich herhalen totdat zich de situatie b of c voordoet.

Nabespreking van het programma

Het gehele programma is in lijst 3 gegeven. De aandachtige beschouwer zal hierin een aantal schoonheidsfoutjes kunnen ontdekken. Het enige gevolg van deze foutjes is dat er een paar geheugenplaatsen meer gebruikt zijn dan strikt noodzakelijk was. Ik heb er dan ook geen voldoende reden in gezien om het gehele programma daarvoor te gaan overschrijven met de kans om hierbij fouten te maken die niet altijd even eenvoudig weer zijn op te sporen.

In het kort volgen hierbij nog enige belangrijke punten:

- Starten op adres 0200.
- Seinsleutel tussen de punten A17



13

- en A18 van de applicatiesteker.
- Na ontvangst van twee verschillende code-elementen (een punt en een streep) worden de karakters op het display zichtbaar.
- Bij een zeer langzame seiner laat het display 'LLLLLL' zien.
- Bij een 'plakker' of een foutmelding 'FFFFFF'.
- Een toename van de seinsnelheid wordt zonder meer geaccepteerd.

- Een sprongsgewijze vermindering kleiner dan 1/3 van de seinsnelheid ook.
- Bij grote seinsnelheden Hex 7F op adres 031E.
- Bij seinoefeningen Hex 7C op adres 0298.
- Stoorimpulsen (bijv. bij ontvangst van radiosignalen) kunnen het programma verstoren. Alle elementen worden dan als strepen geregistreerd.

streerd. In dat geval opnieuw het programma starten.

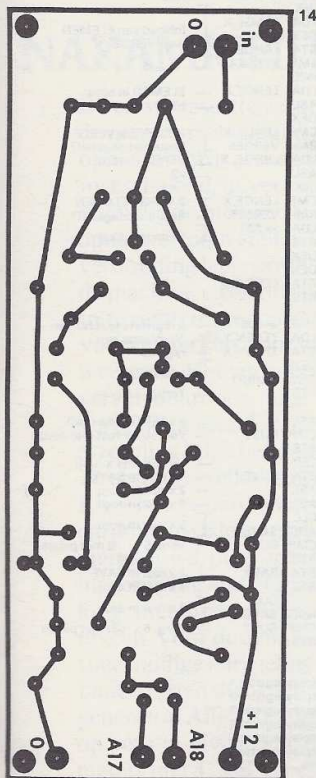
Hulpschakeling

Een schakeling waarmee signalen uit een ontvanger kunnen worden omgezet tot impulsen voor de KIM is geschilderd in afb. 13. De schakeling is gevoelig maar heeft een te lage ingangsimpedantie om rechtstreeks op de uitgang van een detector te kunnen worden aangesloten. Voor de gelijkspanning kan de 12 V worden gebruikt die wordt geleverd door het KIM voedingsapparaat. Met de potmeter P1 wordt de ingangsgoedigheid geregeld. Het signaal dat uit het laagfrequentgedeelte van de ontvanger wordt afgetakt (eventueel van de luidspreker-aansluiting) wordt door T1 versterkt en door de dioden D1 en D2 gelijkgericht. De transistoren T2 en T3 zijn als Smitt-trigger geschakeld. In rusttoestand is T3 geleidend en daardoor ook T5 en T6. De LED wordt door T5 kortgesloten en is dus uit. Door het geleiden van T6 is T4 gesperd (dit ervaart de KIM als '0'). De transistor T4 is op A17 en A18 van de applicatiesteker aangesloten (denk om de juiste polariteit, A17 is emitter).

Komt een voldoende groot signaal op de ingang dan gaat T2 geleiden. T6 spert waardoor T4 opengaat en A17 en A18 praktisch kortsluit. Dit wordt door de KIM ervaren als '1'. Ook T5 spert, zodat de LED die hiermee parallel is geschakeld, oplicht.

Voor het afregelen van P2 zorgt men dat er geen signaal aan de ingang aanwezig is. Potmeter P2 wordt nu zo ingesteld dat de LED net niet oplicht. Daarna wordt het signaal aan de ingang aangesloten en P1 opgedraaid totdat de LED de morsetekens door oplichten zichtbaar maakt.

In afb. 14 is de print getekend en in afb. 15 de componentenopstelling.



[illegible]