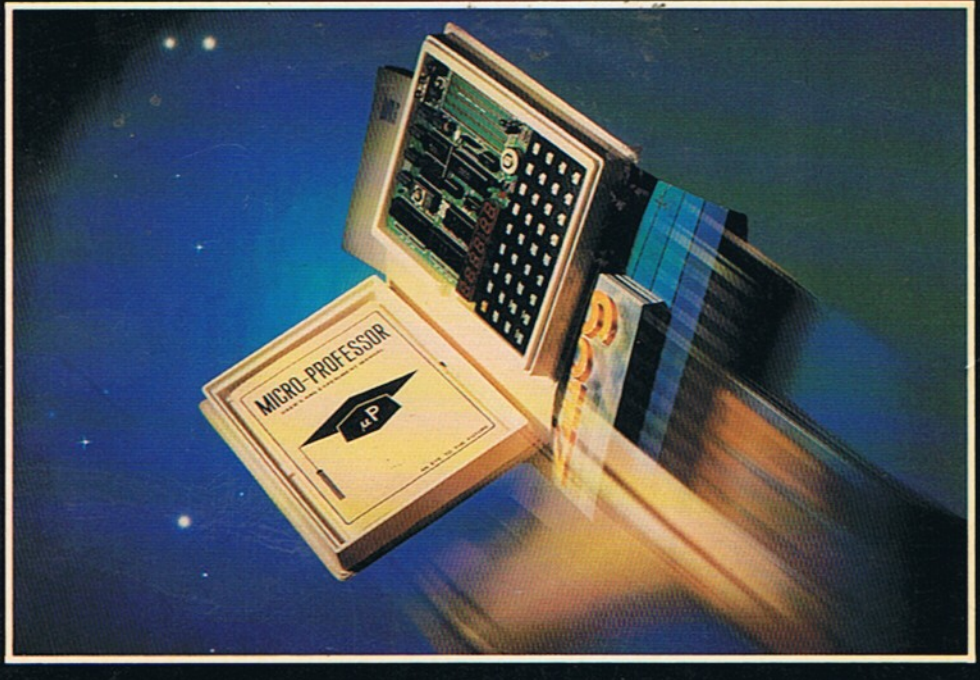
R. Lingier

de microprocessor Z-80 stap voor stap



MAKLU
MAARTEN KLUWER'S
INTERNATIONALE UITGEVERSONDERNEMING
Antwerpen - Apeldoorn

Roland Lingier

DE MICROPROCESSOR Z-80
STAP VOOR STAP

MAKLU
MAARTEN KLUWER'S
INTERNATIONALE UITGEVERSONDERNEMING
Antwerpen-Apeldoorn

C.I.P.-GEGEVENS

Lingier, Roland

De micro-processor Z-80 stap voor stap / Roland Lingier. — Antwerpen - Apeldoorn ; M. Kluwer's Internationale Uitgeversonderneming, 1985 — 230 p. : ill.
ISBN 90-6215-131-0
Doelgroep : informatici
SISO 664.2 UDC 681 UGI A650

Onderwerpen : Z-80 (Microprocessor)

D 1985/1997/10
boekhandelsrubriek 16

INHOUD

24. Interrupts 190

VOORWOORD

De opkomst van de microprocessor heeft het mogelijk gemaakt computers te fabriceren voor lage prijzen. De zeer snelle verspreiding van dit nieuwe instrument doet ook de vraag naar informatie over het gebruik ervan toenemen. De uitgevers reageren hier graag op met een uitgebreid aanbod op het gebied van software. Ook worden door allerlei instanties cursussen over software gegeven, die veel succes hebben. De reden waarom men zulke cursussen volgt is eenvoudig : ze bieden een veelbelovende toekomst zonder dat er veel gespecialiseerde voorkennis van de cursist verwacht wordt. Bijna iedereen kan met software beginnen en in korte tijd een betrekkelijk verrassend resultaat behalen.
De kennis over de werking van de computer verspreidt zich niet zo snel als die over de programmatuur. Veel minder boeken gaan over de hardware. De weinige die wel over die hardware gaan, zijn zelden op de praktijk gericht, wat voor de lezer die ervaring op dit gebied op wil doen, toch noodzakelijk is. Een theoretische benadering blijft te abstract. Door de grote hoeveelheid nieuwe stof gaan de met veel moeite opgedane begrippen snel vervagen. Daarbij komt nog dat een flinke voorkennis van elektronica noodzakelijk is, waardoor niet iedereen met deze studies kan starten.
Het sturen en regelen van allerlei toestellen kan door middel van een microcomputer gebeuren, maar vaak kan dit ook al met alleen een microprocessor en enkele chips. Is een zeer hoge snelheid vereist, dan is de microprocessor-opstelling zelfs in het voordeel ten opzichte van de veel duurdere microcomputer die in een hogere taal geprogrammeerd wordt.
Wie eens verder wil gaan dan het bedienen van zijn eigen microcomputer heeft niet genoeg aan zijn kennis van een programmeertaal. Tussen het toestel en de microcomputer moet een schakeling komen die zowel aan de microcomputer als aan het te controleren toestel aangepast moet zijn. Een dergelijke schakeling is moeilijk te krijgen, en het vereist bovendien een zekere kennis van hardware om hem in gebruik te nemen.

De werking van een moderne micro- of minicomputer berust helemaal op het principe van de werking van de microprocessor en enkele aanverwante chips. Vandaar dat de studie over microprocessors als vanzelf inzicht geeft in de hardware van computers.

De microprocessor is een stuk dode hardware zolang hij geen programma aangeboden krijgt. Deze programma's moeten in machinetaal geschreven zijn. Het schrijven in machinetaal is bijna niet te leren zonder kennis van hardware. Daarom gaat de studie van een microprocessor zowel over software als over hardware. Programma's moeten de eigenschappen van de hardware aan het licht brengen en omgekeerd.
De bedoeling van dit boek is dan ook de lezer ertoe te brengen, om door middel van proeven op de Micro-Professor de microprocessor onder de knie te krijgen. Van de lezer wordt verondersteld dat hij kennis van digitale techniek heeft. Enige praktische ervaring met dit deel van de elektronica is belangrijk in de hoofdstukken waar de tussenschakelingen (interface) besproken en getest worden.

Wie het principe van de werking van de microprocessor kent, weet wat 3-state-poorten zijn, en wie een RAM of ROM kan aansluiten kan direct beginnen met hoofdstuk 4.
Hoofdstuk 5 maakt de lezer vertrouwd met het gebruikte toestel, de Micro-Professor. Na het intoetsen van enkele kleine, niet al te ingewikkelde programma's, wennen de vingers aan het toetsenbord en wordt duidelijk wat op de display verschijnt.
Vanaf hoofdstuk 6 worden alle eigenschappen van de microprocessor verklaard, zoveel mogelijk aan de hand van kleine programma's waarmee de theorie toegepast kan worden. Daarmee heeft de lezer er dan ook telkens een voorbeeld van hoe de eigenschap benut moet worden.

R. Lingier
Leraar aan het
Stedelijk Hoger Technisch Instituut
in Oostende

1. GEHEUGENS

1.A. Three-state-uitgangen

De uitgang van een gewone TTL-poort is altijd hoog of laag. De twee uitgangstransistoren zijn nooit samen in geleiding, maar ze zijn ook nooit samen geblokkeerd.
Dit veroorzaakt problemen als verschillende signalen elk op hun beurt naar eenzelfde punt moeten worden gebracht.
Zo zal het signaal U in figuur 1.A.1 slechts naar A kunnen worden overgebracht als de signalen V en W voortdurend hoog blijven. Als V en/of W laag wordt kan de toestand op A verschillend zijn van die op U. Bovendien worden de uitgangen zwaar belast, in verhouding tot het aantal parallel geschakelde uitgangen.
Als een AND-poort en een OR-poort, zoals in figuur 1.A.2, aanwezig zijn, worden de eigenschappen al beter. Doordat de control-ingangen C2 en C3 laag gehouden worden kunnen de signalen V en W geen invloed uitoefenen op A, ook niet als deze van toestand veranderen.
Door de control-ingangen C1, C2 of C3 hoog te maken kan één van de drie signalen U, V of W naar de uitgang A overgebracht worden. Het aantal ingangen op de OR-poort beperkt het uitbreiden van de schakeling voor het selecteren van meer dan drie signalen.
Een andere mogelijkheid biedt het gebruik van three-state-poorten. Een three-state-poort heeft buiten zijn in- en uitgang ook nog een control-ingang.
Met een hoog op de control-ingang werk de three-state als een gewone TTL-poort. Een laag op de control-ingang maakt dat de beide uitgangstransistoren van de three-state-poort blokkeren. De uitgang Q is dan noch met de nul noch met de +5V verbonden. Deze toestand wordt dan ook als *high impedance state* (hoge impedantie-toestand) aangeduid.
Een three-state-poort kan ook gezien worden als een standaard-TTL-poort, waarbij met de uitgang een contact in serie geschakeld staat die gestuurd wordt door de control-ingang. Alleen als de control-ingang C het contact sluit, komt de informatie naar buiten op de uitgang Q. In het andere geval is de klem Q niet met het inwendige van de poort verbonden.
Het selecteren van signalen met three-state-poorten is heel eenvoudig, zoals figuur 1.A.4. laat zien. Van de ingang die zijn signaal naar A moet overbrengen, hoeft alleen de control-ingang hoog gehouden te worden. Het uitbreiden voor de selectie van meer dan drie signalen is onbeperkt mogelijk.
Met three-state-poorten kunnen nu ook signalen in beide richtingen overgebracht worden. Figuur 1.A.5 toont, dat een signaal van A naar B overgebracht kan worden als C1 hoog is en C2 laag. Als C2 hoog, en C1 laag is, kan een signaal van B naar A getransporteerd worden.

Taak. Teken een schakeling voor signaaloverdracht in beide richtingen, waarbij de enige ingang C de richting bepaalt.

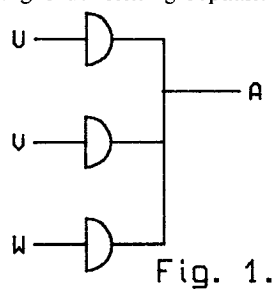


Fig. 1.A.1

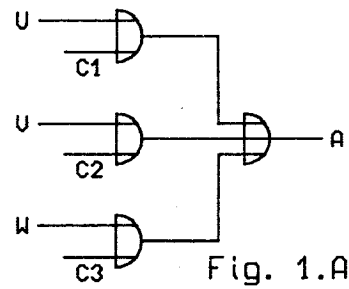
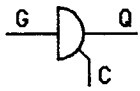


Fig. 1.A.2



C	G	Q
1	0	0
1	1	1
0	0	HI
0	1	HI

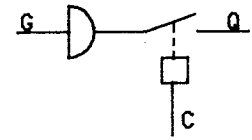


Fig. 1.A.3

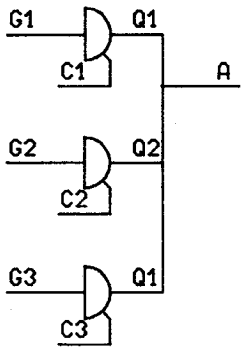
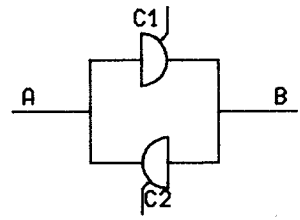


Fig. 1.A.4



C1	C2	richting
1	0	A naar B
0	1	B naar A

Fig. 1.A.5

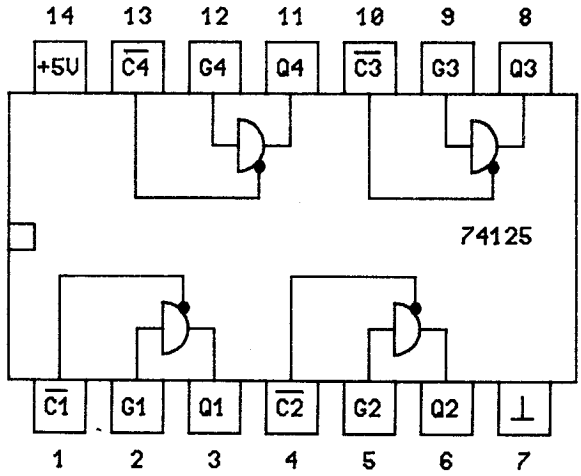
1.B. Three state I.C.'s

1.B.1. Poorten met 3-state-uitgang

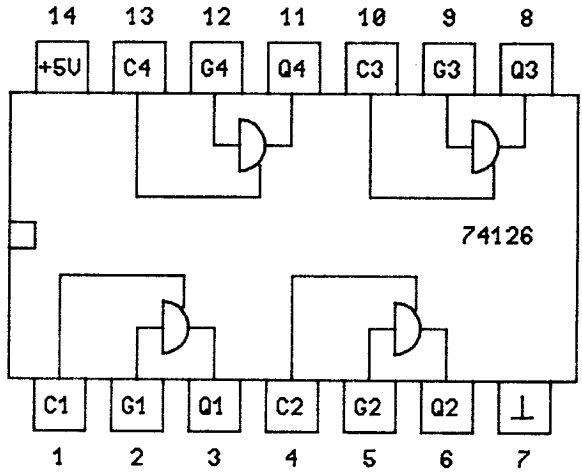
Deze komen ook voor in LS- en S-versie. De elektronische eigenschappen van deze poorten komen overeen met die van de normale TTL-poorten.

74126
Bevat vier onafhankelijke niet-omkerende poorten met three-state-uitgang. De control-ingang C moet hoog zijn om het signaal op de uitgang te krijgen. Als C laag is, is de uitgang Q in zijn high impedance state.

74125
Hier moet de control-ingang laag zijn om een uitgangssignaal te krijgen. Een hoog op de C-ingang brengt de poort in zijn high impedance state.



C	G	Q
0	0	0
0	1	1
1	0	HI
1	1	HI



C	G	Q
0	0	HI
0	1	HI
1	0	0
1	1	1

1.B.2. Hex bus drivers

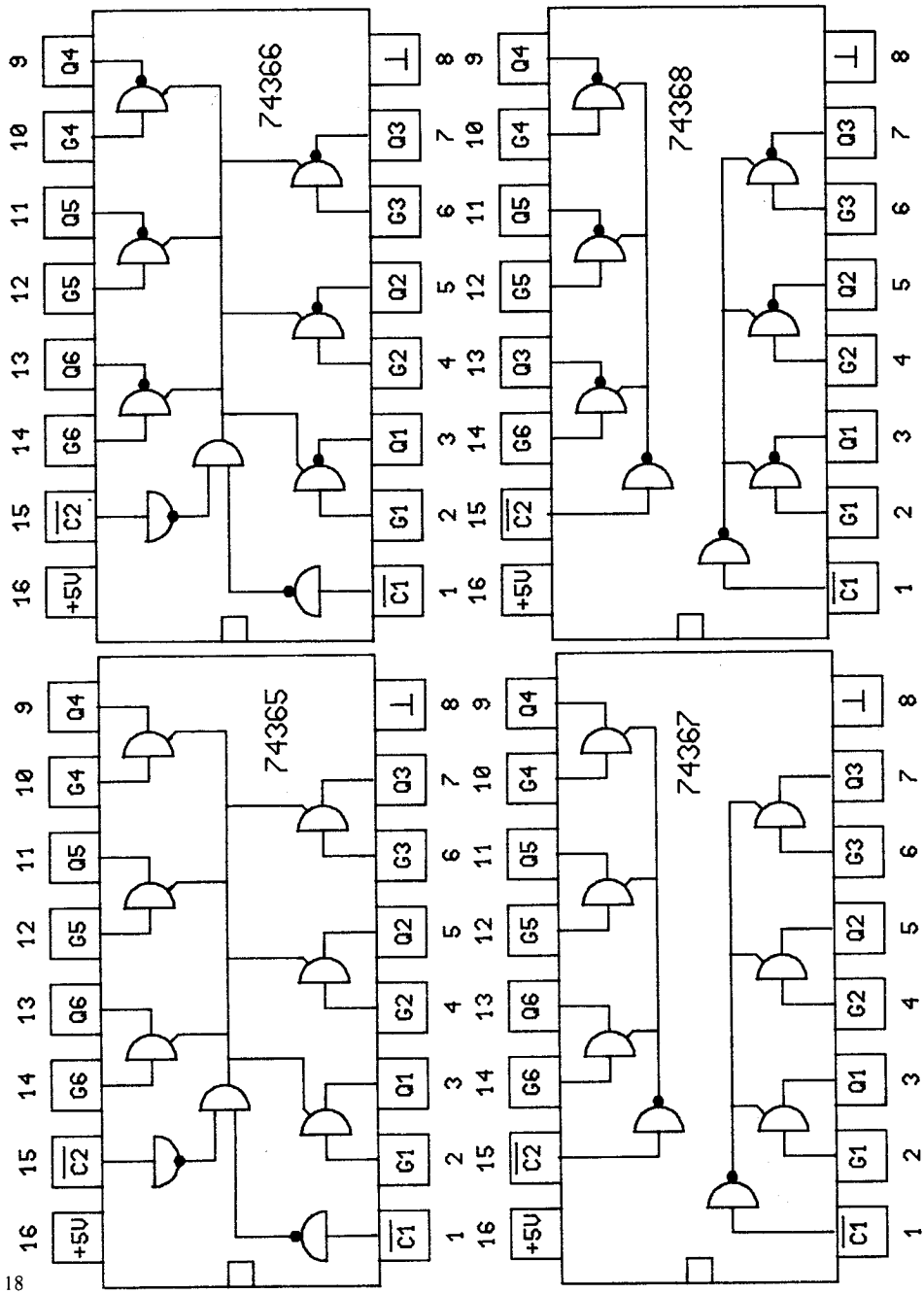
Deze bevatten zes poorten met three-state-uitgang. Andere elektronische eigenschappen als bij gewone poorten.
Komen ook voor in LS-versie.

74365
Een gemeenschappelijke control-lijn voor de zes poorten. Alleen signalen op de uitgangen als de control-ingangen C1' en C2' beide laag zijn. Anders zijn alle uitgangen in hun high impedance state.

74366
Als de 74365 ; maar het signaal aangelegd op de G-ingang komt omgekeerd op de Q-uitgang als de beide C-ingangen laag zijn.

74367
Een control-ingang heeft hier slechts invloed op drie poorten. Met nul op C1' komen signalen op de uitgangen Q1, Q2 en Q3.

74368
Als de 74367, maar het signaal uit de uitgang is het omgekeerde van dat op de ingang.



1.B.3. Octal buffers en line drivers

Naast de LS-uitvoering bestaat ook de S-versie.
Zij bevat acht three state buffers, waarvan de belangrijkste elektronische eigenschappen zijn :

		LS	S
IOH	High-level output current	-15 mA	-15 mA
IOL	Low-level output current	24 mA	64 mA
IIH	High-level input current	20 microA	50 microA
IIL	Low-level input current	-0.2 mA	-0.4 mA

Alle ingangen zijn van het Schmitt-trigger-type met een hysteresisch van 400 mV. Door hun grote belastbaarheid kunnen ze gebruikt worden als line driver van langere lijnen die een niet te verwaarlozen capaciteit hebben. Hun Schmitt-trigger-ingang maakt ze ook geschikt als line receiver. Ruis die op de lijn is ontstaan wordt door de Schmitt-trigger weggewerkt. Bij lange lijnen kunnen tussen driver en receiver nog één of meer repeaters ingeschakeld worden voor het herstellen van de signalen (fig. 1.B.1.).

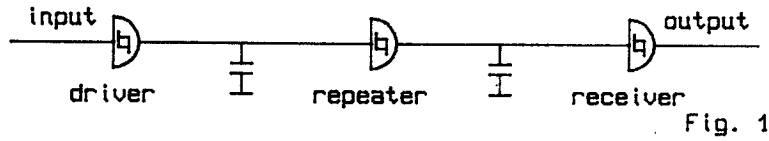
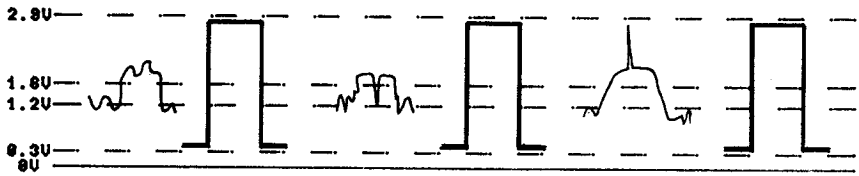


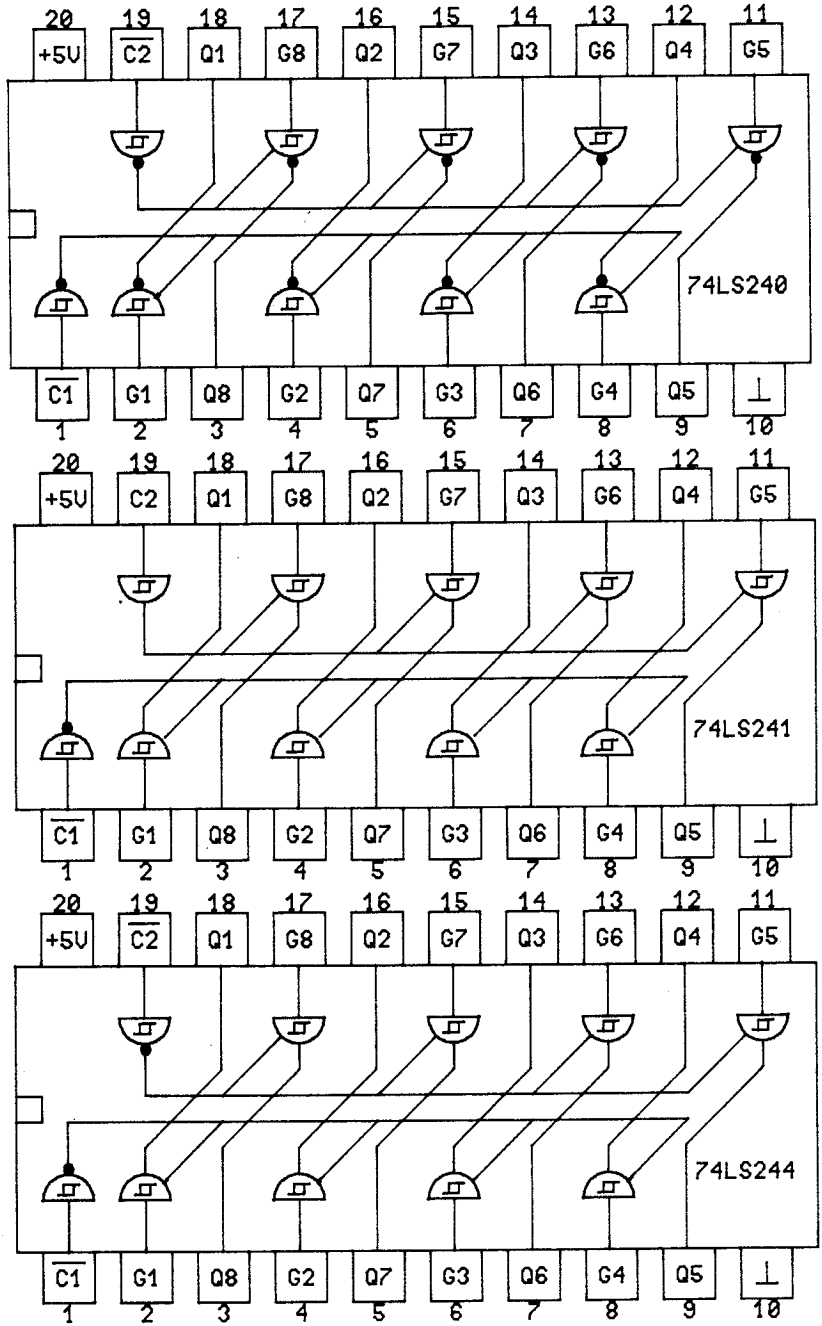
Fig. 1.B.1



74LS240
Iedere control-ingang beheerst vier buffers. De uitgangssignalen zijn de omgekeerde van de ingangssignalen.

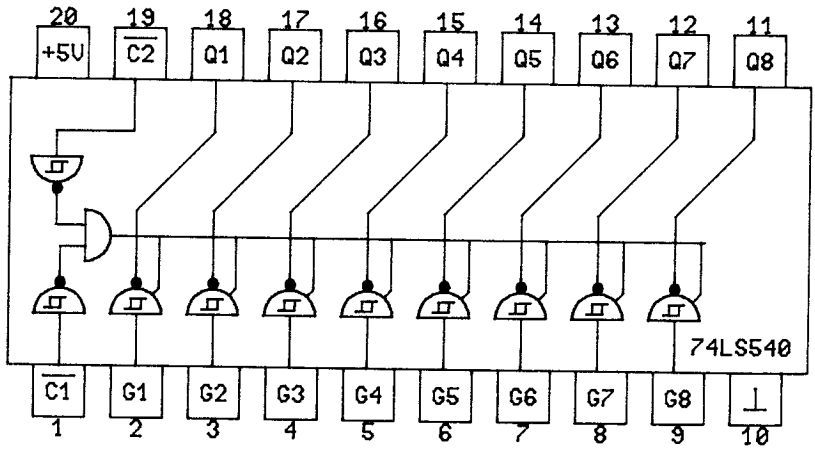
74LS241
De signalen worden niet omgekeerd door de buffers. De control-ingang C1' moet laag zijn om de buffers ervan actief te maken. Daarentegen moet de control-ingang C2 hoog zijn om de buffers daarvan in dezelfde toestand te brengen.

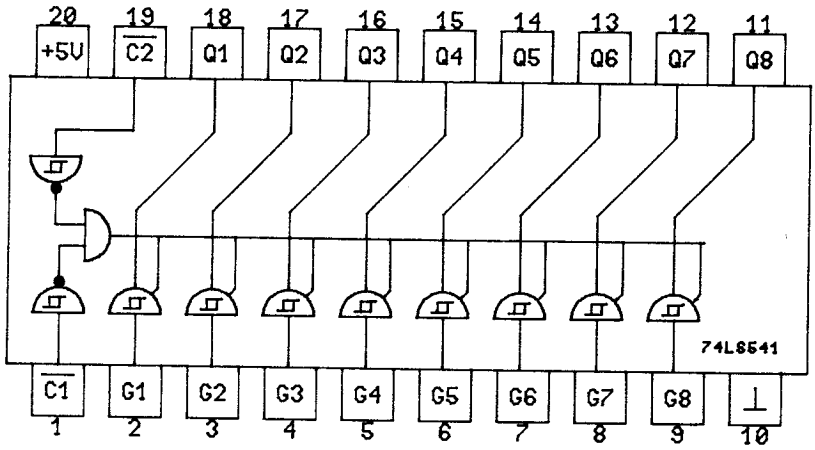
74LS244
Als de 74LS240, maar de buffers keren de signalen niet om.


20 19 18 17 16 15 14 13 12 11
+5V C2 Q1 G8 Q2 G7 Q3 G6 Q4 G5
74LS240
C1 G1 Q8 G2 Q7 G3 Q6 G4 Q5 ⊥
1 2 3 4 5 6 7 8 9 10
20 19 18 17 16 15 14 13 12 11
+5V C2 Q1 G8 Q2 G7 Q3 G6 Q4 G5
74LS241
C1 G1 Q8 G2 Q7 G3 Q6 G4 Q5 ⊥
1 2 3 4 5 6 7 8 9 10
20 19 18 17 16 15 14 13 12 11
+5V C2 Q1 G8 Q2 G7 Q3 G6 Q4 G5
74LS244
C1 G1 Q8 G2 Q7 G3 Q6 G4 Q5 ⊥
1 2 3 4 5 6 7 8 9 10

74LS540
De penbezetting is anders, en meestal handiger voor de lay-out van de print.
De beide control-ingangen moeten laag zijn omdat de acht buffers hun signaal anders zouden doorgeven aan de uitgangen.

74LS541
Als de 74LS540, maar niet-omkerende buffers.



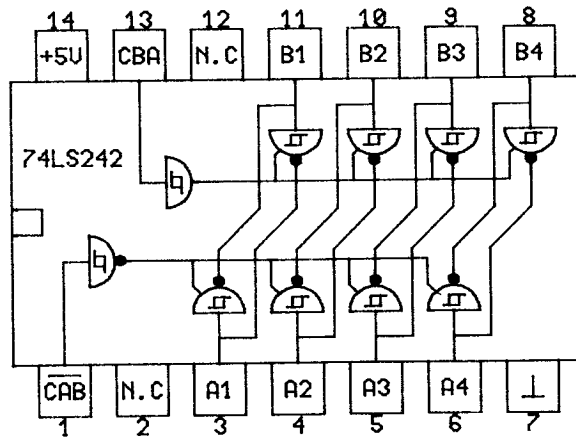


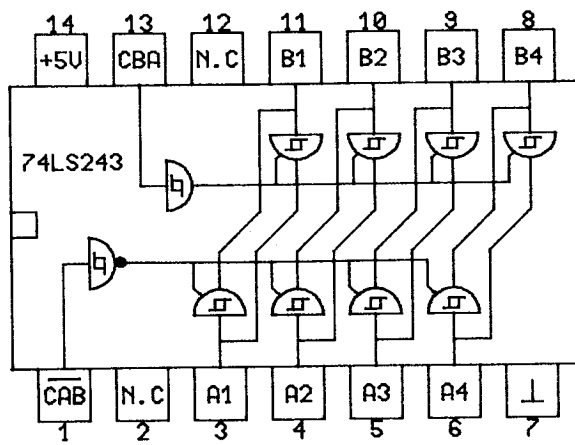
1.B.4. Quadruple bus transceivers

Deze kunnen zowel vier signalen uitzenden (transmitter), als vier signalen ontvangen (receiver). Worden daarom ook *transceivers* genoemd.
Elektronische eigenschappen als de 74LS240.

74LS242
Als beide control-ingangen laag zijn, worden de signalen van de A-klemmen omgekeerd en naar de B-klemmen doorgegeven. Zijn de beide control-ingangen hoog, dan worden de signalen van B omgekeerd en naar de A-klemmen gebracht.
Als CBA laag is en CAB' hoog, staan alle buffers in hun high impedance state. De omgekeerde toestand, CBA hoog en CAB' laag, maakt de transceiver op hetzelfde ogenblik in beide richtingen actief. Daardoor kunnen oscillaties optreden die in staat zijn de transceiver uit te schakelen.

74LS243
Als de 74LS242, maar met niet-omkerende buffers.





1.B.5. Octal bus transceivers

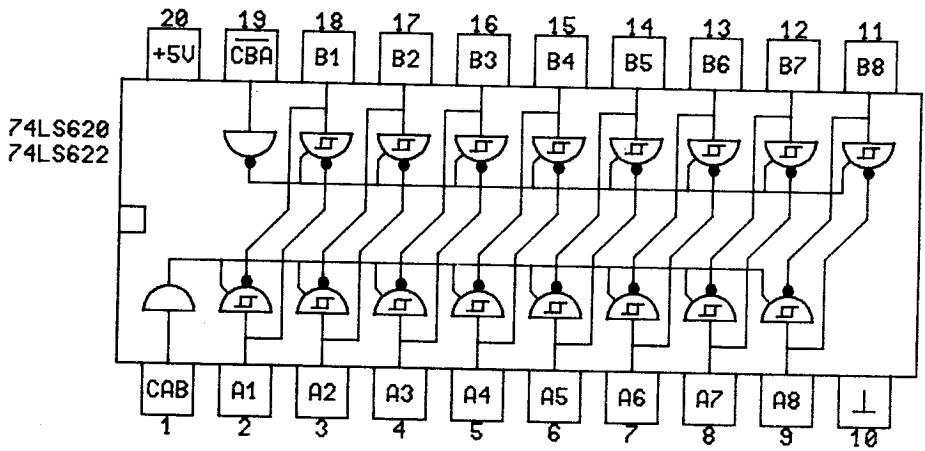
Elektronische eigenschappen als de 74LS242.

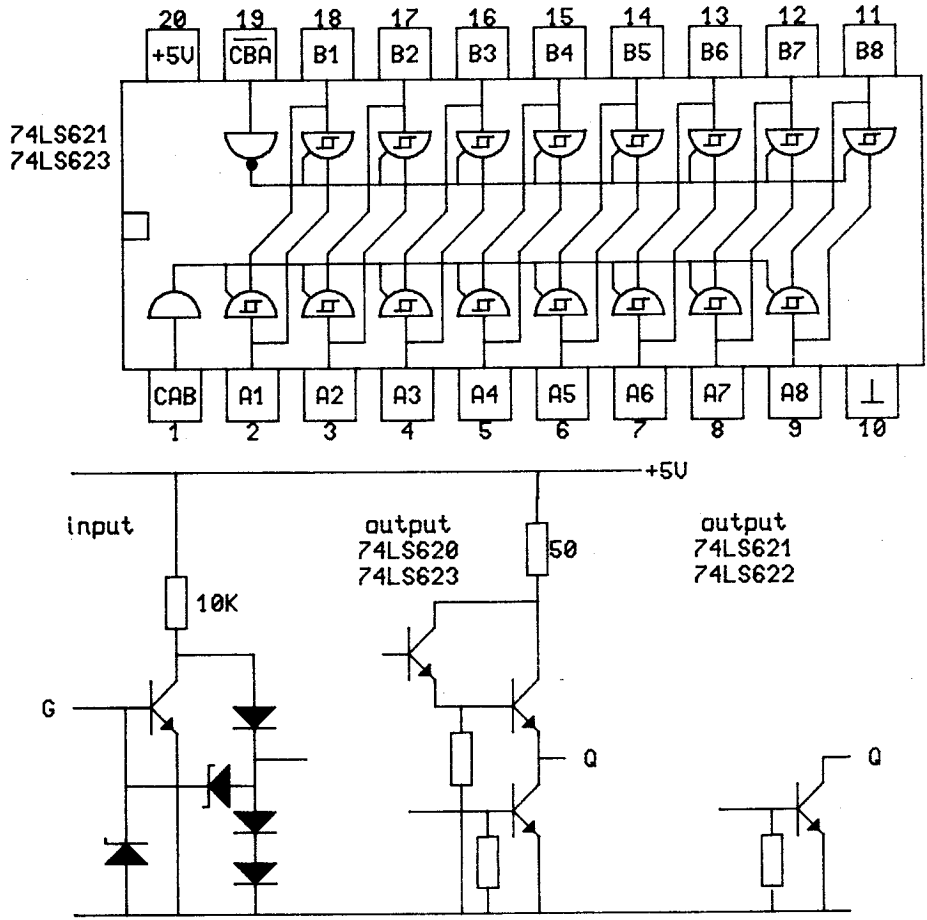
74LS620
Als CBA' en CAB beide laag zijn, worden de signalen van B omgekeerd en naar A doorgegeven. Als beide control-ingangen hoog zijn gebeurt dit in tegenovergestelde richting, eveneens met omkeren van de signalen.
Door CBA' hoog en CAB laag te houden komen alle buffers in hun high-impedance-toestand, en is de transceiver aan beide zijden uitgeschakeld. De omgekeerde toestand op de controlingangen maakt alle buffers actief, waardoor ook hier oscillaties kunnen optreden die de transceiver uitschakelen.

74LS622
Als de 74LS620, maar met open collector-uitgangen.

74LS623
Als de 74LS620, maar met niet-omkerende three state buffers.

74LS621
Als de 74LS623, maar met niet-omkerende open collector-uitgangen.





74LS640
De control-lijnen worden gestuurd door AND-poorten. Om een control-lijn hoog te krijgen moet in elk geval de E'-ingang laag zijn. Welke control-lijn dan hoog wordt, is afhankelijk van het niveau op de DIR-ingang. Een hoog op DIR keert de signalen op de A-klemmen om en brengt ze naar de B-klemmen. Als DIR laag is gebeurt dit in omgekeerde richting.
De ingang E' (Enable) bepaalt of signalen overgebracht moeten worden. Het niveau op de ingang DIR (DIRection) bepaalt de richting.
Hier kunnen de twee rijen buffers nooit beide actief zijn. Beschadiging door verkeerde signalen op de control-ingangen is hier niet mogelijk.

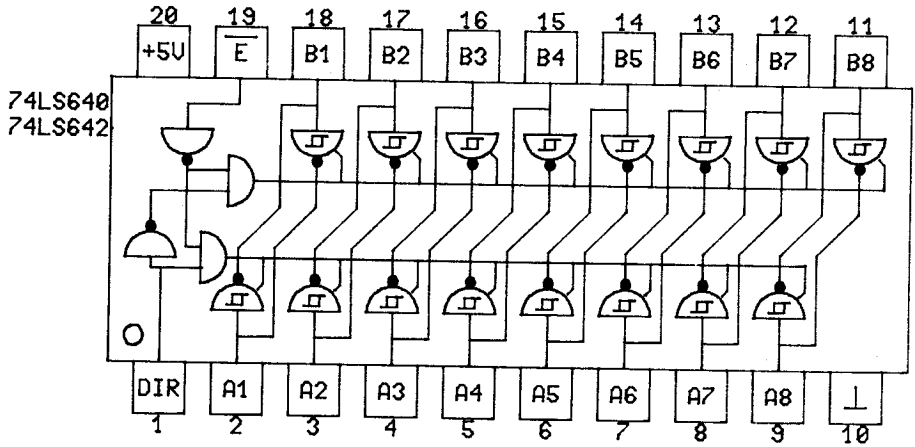
74LS645
Als de 74LS640, maar met niet-omkerende three state buffers.

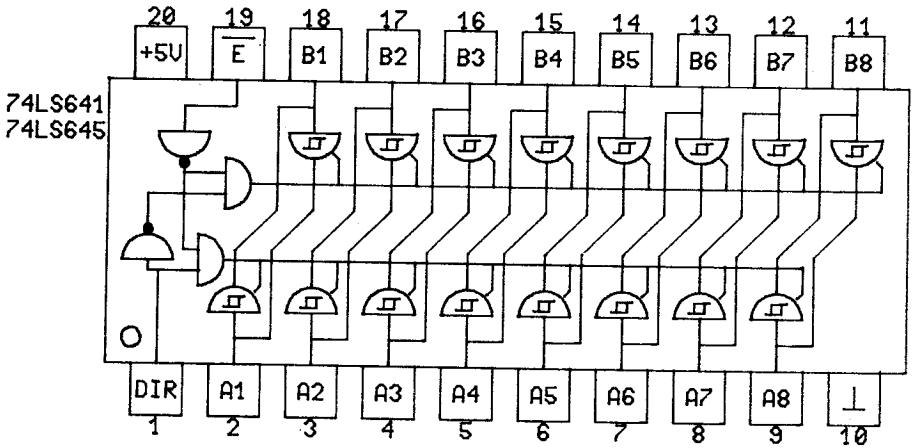
74LS643
Als de vorige, maar omkerend van A naar B en niet-omkerend van B naar A. Three-state-uitgangen.

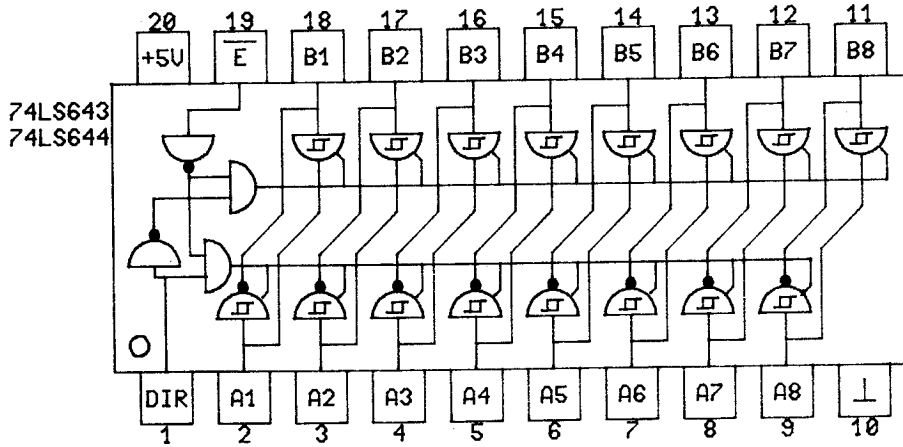
74LS642
Als de 74LS640 (omkerend), maar met open collector-uitgangen.

74LS641
Niet-omkerend met open collector-uitgangen.

74LS644
Omkerend en niet-omkerend met open collector-uitgangen.





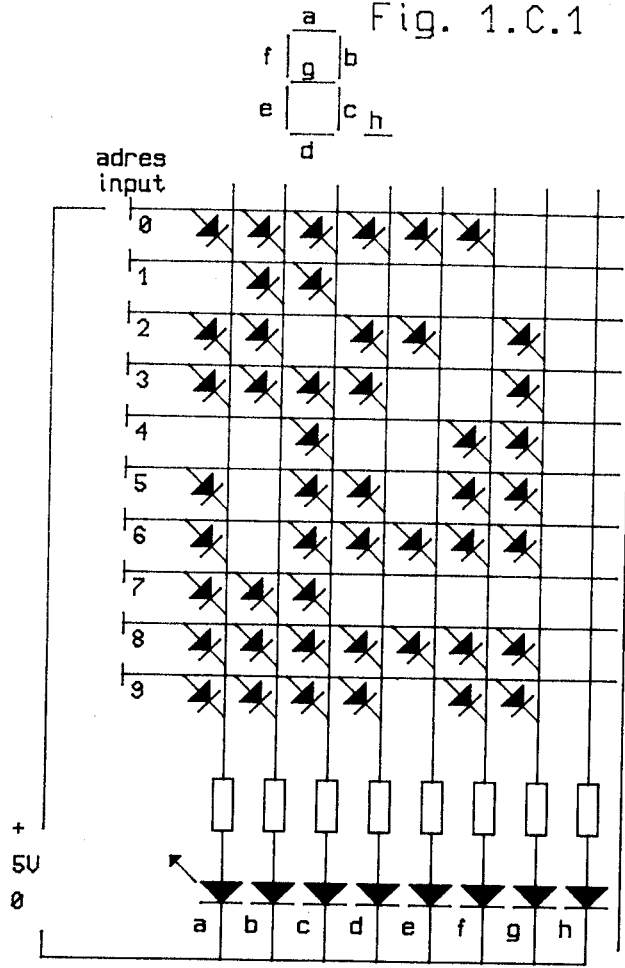


1.C. Het Read Only Memory (ROM)

De bekende diodematrix voor het sturen van een 7 segments-display, zoals weergegeven in figuur 1.C.1., is een eenvoudig voorbeeld van een geheugen. De geheugeninhoud wordt bepaald door de dioden, en kan van buitenaf niet gewijzigd worden. Door een van de ingangen met de positief te verbinden, wordt de inhoud van de geactiveerde lijn door de display weergegeven. Daar de geheugeninhoud alleen gelezen kan worden, spreekt men van een *Read Only Memory,* afgekort als ROM.
De diodematrix heeft tien lijnen of adresseerbare lokaties. Iedere lokatie kan acht bits onthouden en heeft haar eigen adres. Hier van 0 tot 9, dus decimaal.
Daar altijd maar één lokatie tegelijkertijd moet en mag worden gelezen, is ook binaire adressering mogelijk. Dit kan worden bereikt door voor de diodematrix een 4 - line - to - 10 - line decoder te schakelen, zoals in figuur 1.C.2. Het aantal adres-ingangen is daardoor van 10 naar 4 gereduceerd, waarbij dan nog niet alle adresseermogelijkheden benut zijn.
De inhoud van een ROM kan in een tabel weergegeven worden. Voor de diodematrix met binaire adressering wordt de tabel zoals in figuur 1.C.3. Daar zijn A0 tot A3 de adres-ingangen. De uitgangen worden aangeduid met D0 tot D7 als afkorting voor *Data* (informatie, gegevens).

Taak 1. Pas de diodematrix uit figuur 1.C.1. aan, zodat bij alle cijfers ook segment h brandt.
Taak 2. De display moet ook de letters A, B, C, D, E en F kunnen weergeven als op de binaire adres-ingangen een waarde groter dan 9 wordt aangelegd. Teken de schema's die overeenkomen met die uit figuur 1.C.1. en figuur 1.C.2.

Fig. 1.C.1



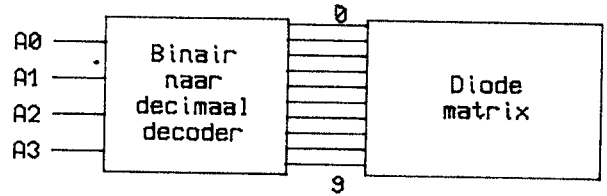


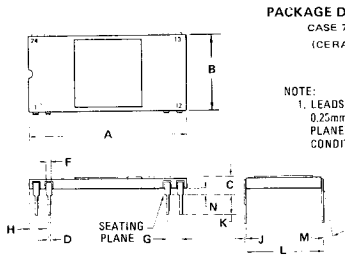
Fig. 1.C.2

Fig. 1.C.3

	address inputs				outputs							
	A3	A2	A1	A0	D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	1	1	1	1	1	1	0
1	0	0	0	1	0	0	1	1	0	0	0	0
2	0	0	1	0	0	1	1	0	1	1	0	1
3	0	0	1	1								
4												
5												
6												
7												
8												
9												

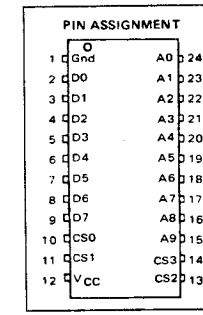
1.D. De ROM 6830

Figuur 1.D.1 toont de behuizing en de pin-indeling. Fig.1.D.1



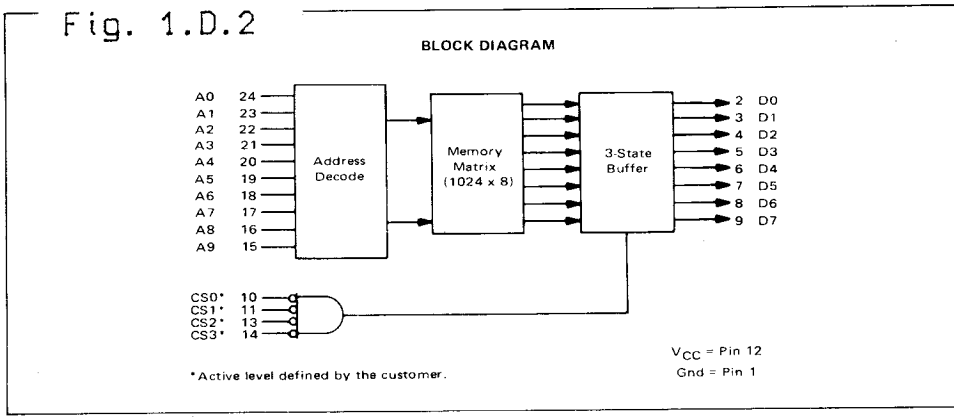
See Page 165 for Plastic Package dimensions.

DIM	MILLIMETERS MIN	MILLIMETERS MAX	INCHES MIN	INCHES MAX
A	29.97	30.99	1.180	1.220
B	14.88	15.62	0.585	0.615
C	3.05	4.19	0.120	0.165
D	0.38	0.53	0.015	0.021
F	0.76	1.40	0.030	0.055
G	2.54 BSC		0.100 BSC	
H	0.76	1.78	0.030	0.070
J	0.20	0.30	0.008	0.012
K	2.54	4.19	0.100	0.165
L	14.88	15.37	0.585	0.605
M	-	10°	-	10°
N	0.51	1.52	0.020	0.060



Het blokdiagram in figuur 1.D.2 toont dat er tien adres-ingangen zijn (A0 tot A9), waarmee 1024 lokaties kunnen worden geselecteerd.

Fig. 1.D.2



Iedere lokatie is 8 bits (of 1 byte) breed, zodat er ook acht data-uitgangen zijn (D0 tot D7). Voor iedere data-uitgang is er telkens een 3-state-poort geschakeld. Deze acht line drivers kunnen samen door één AND-poort met vier ingangen uit hun high impedance state gehaald worden. De vier ingangen worden de *chip select inputs* genoemd. Alleen als deze vier inputs alle in hun actieve toestand zijn, komt de inhoud van de geadresseerde lokatie op de data-uitgangen. Is één of meer van de chip select inputs niet op zijn actief niveau, dan zijn de acht data-uitgangen van de geheugenmatrix uitgeschakeld.
De 6830 heeft slechts één voedingsspanning van 5V nodig, en is TTL-compatible (verenigbaar, overeenstemmend, bruikbaar met). Dit is ook te zien in de DC operation conditions en characteristics. In deze tabellen is te zien dat de in- en uitgangsspanningen van de 6830 wel dezelfde zijn als die bij TTL-poorten, maar niet de in- en uitgangsstromen. Zo is de IOL (low level output current) slechts 1.6mA, zodat de 6830 slechts met één standaard TTL-ingang mag worden belast.
De inhoud van de ROM-lokaties wordt bij de fabricage aangebracht volgens de aanwijzingen van de klant, en kan nadien niet meer gewijzigd worden. Ook bij welk niveau de chip select inputs actief zijn, mag door de klant opgegeven worden.

Zelfs bij een gewone poort als de 7404 duurt het 10 nsec voordat de uitgang een ingangsverandering volgt (1 nsec = 1/1000.000.000 sec). De vertragingen die optreden bij een 6830 zijn te volgen in figuur 1.D.3.

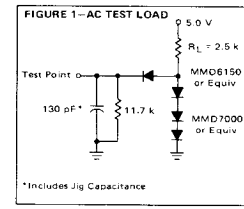
Fig. 1.D.3

AC OPERATING CONDITIONS AND CHARACTERISTICS

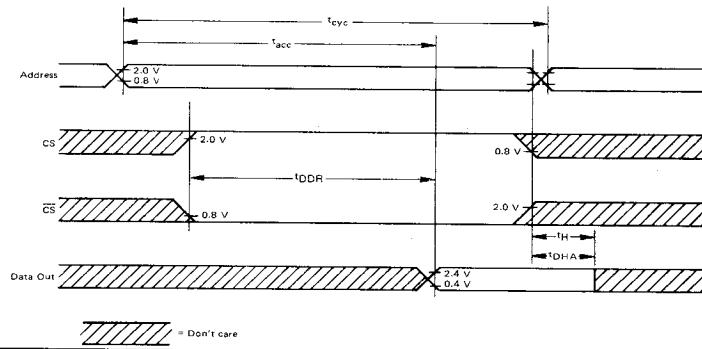
(Full operating voltage and temperature unless otherwise noted.)

(All timing with $t_r = t_f = 20$ ns, Load of Figure 1)

Characteristic	Symbol	Min	Max	Unit
Cycle Time	t_{cyc}	500	–	ns
Access Time	t_{acc}	–	500	ns
Data Delay Time (Read)	t_{DDR}	–	300	ns
Data Hold from Address	t_{DHA}	10	–	ns
Data Hold from Deselection	t_H	10	150	ns



TIMING DIAGRAM



Als een geldig adres op de adres-ingangen geplaatst wordt en de chip select inputs op hun actief niveau staan, duurt het 500 nsec voordat de data op de dataklemmen verschijnen. Deze tijd wordt de *access* (toegang, bereikbaarheid) *time (tacc)* genoemd. Alhoewel deze tijd voor menselijke begrippen enorm kort is, moet in sommige ontwerpen met deze vertraging rekening gehouden worden. De andere tijden die zijn opgegeven in het timing diagram, zijn minder belangrijk. Zo is tDDR de tijd die nodig is om de 3-state-poorten uit hun high impedance state te halen. Daar deze iets hoger is dan de access time, mogen de chip-select-ingangen iets later gestuurd worden dan de adres-ingangen, zonder dat de access time groter wordt.
De tijden tDHA en tH geven aan, hoe lang de data nog op de D-klemmen beschikbaar blijven nadat het adres of de chip-select-signalen verwijderd zijn.
In systemen waarbij meer dan 1024 byte ROM nodig is, kunnen twee of meer 6830 op dezelfde data- en adreslijnen aangesloten worden zoals in figuur 1.D.4. Met de chip-select-ingangen wordt bepaald welke van de twee ROM's gelezen zal worden.

A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	Hexa	Dec.	
0	0	0	0	0	0	0	0	0	0	0	000	0	
0	1	1	1	1	1	1	1	1	1	1	3FF	1023	ROM 1
1	0	0	0	0	0	0	0	0	0	0	400	1024	
1	1	1	1	1	1	1	1	1	1	1	7FF	2047	ROM 2

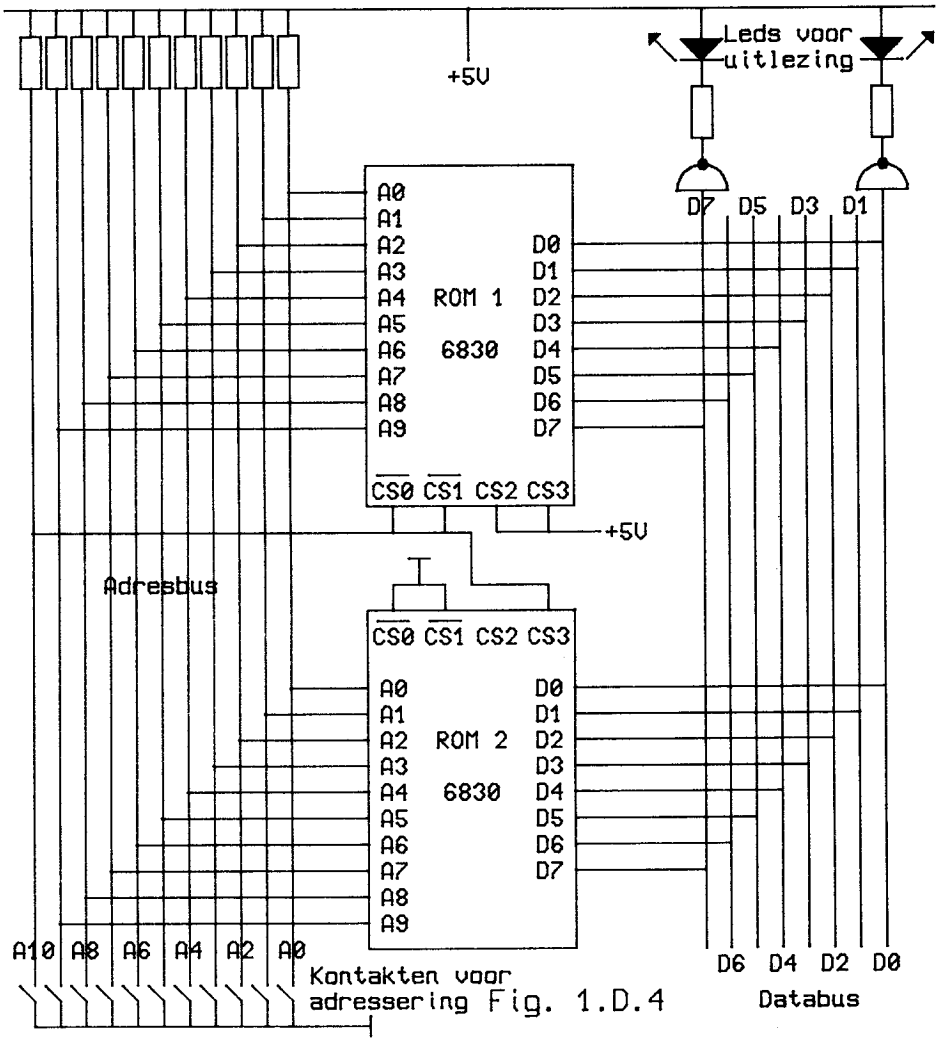


Fig. 1.D.4

Zolang A10 laag is, kan met de adreslijnen A0 tot A9 iedere lokatie van ROM 1 bereikt worden. Dit komt overeen met de adressen 0 tot 1023 decimaal.
De ROM 2 kan slechts gelezen worden als A10 hoog is ; dit is op de adressen 1024 tot 2047 decimaal.

Taak 1. Breid figuur 1.D.4 uit met een derde ROM 6830. De adressen daarvan moeten volgen op die van ROM 2.

Taak 2. Zoek in data books een andere ROM op. Bestudeer de eigenschappen daarvan en teken er een vervangingsschema mee voor figuur 1.D.4.

Taak 3. Pas het schema uit figuur 1.D.4. zodanig aan, dat om de seconde de inhoud van de volgende ROM-lokatie op de leds zichtbaar is. Behalve het bedienen van een druknop voor het starten moet alles automatisch verlopen.

1.E. Het Programmable Read Only Memory (PROM)

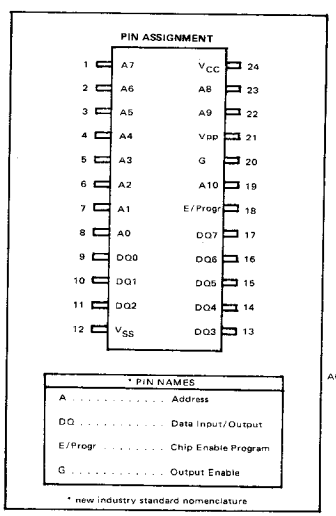
De inhoud van een ROM wordt aangebracht tijdens de fabricage, in opdracht van de klant. Uiteraard is dit alleen rendabel bij zeer grote aantallen (minimum 1000).
Bij een PROM staat op ieder kruispunt van de geheugenmatrix een diode met een smeltdraadje in serie. Met een speciaal toestel, een PROM programmer, kan elk gewenst smeltdraadje door een hoge spanning van een bepaalde duur doorgesmolten worden.
Als de PROM programmer weet welke inhoud aan de PROM moet worden gegeven, duurt het maar enkele minuten om de nodige smeltdraadjes door te branden.
Na deze bewerking kan de inhoud van de PROM niet meer gewijzigd worden. Indien een fout ontdekt wordt, moet met een nieuwe PROM de bewerking herhaald worden.

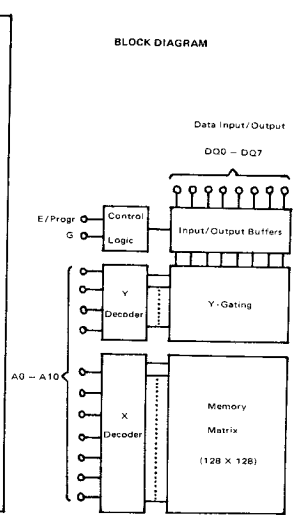
1.F. Het Erasable Programmable Read Only Memory (EPROM)

De EPROM heeft op ieder kruispunt van zijn matrix een halfgeleiderschakeling die opgeladen kan worden. Deze is zodanig uitgevoerd dat de lading gedurende tien jaar behouden blijft. De chip zelf is slechts afgedicht met een venstertje, dat bij normaal gebruik afgedicht moet zijn, daar de lading verloren gaat onder invloed van UV-licht.
Een niet-geladen schakeling op een kruispunt komt overeen met een diode bij een ROM. Ook hier is weer een speciaal toestel nodig, een EPROM programmer, die ook door hogere spanningspulsen de schakelingen een lading kan geven.
Een felle UV-belichting van 5 tot 10 minuten doet alle ladingen leeglopen. Nadien is dezelfde EPROM opnieuw programmeerbaar. Meestal is dat proces zo'n honderd maal herhaalbaar.
De prijs van een EPROM is hoger dan die van een PROM.
De meeste EPROM's hebben dan ook een pin-equivalente PROM en/of ROM.

1.G. De EPROM 2716

Pin-indeling, blokdiagram en belangrijkste kenmerken in figuur 1.G.1. **Fig. 1.G.1.**





PIN NUMBER	24	21	20	18	12	9 – 11, 13 – 17	Mode
	V_{CC}	Vpp	G	E/Progr	V_{SS}	DQ	
	V_{CC}	V_{CC}	V_{IL}	V_{IL}	V_{SS}	Data Out	Read
	V_{CC}	V_{CC}	V_{IH}	Don't care	V_{SS}	Hi Z	Output Disable
	V_{CC}	V_{CC}	Don't care	V_{IH}	V_{SS}	Hi Z	Standby
	V_{CC}	V_{HP}	V_{IH}	Pulsed V_{IL} to V_{IH}	V_{SS}	Data In	Program
	V_{CC}	V_{HP}	V_{IL}	V_{IL}	V_{SS}	Data out	Program verify
	V_{CC}	V_{HP}	V_{IH}	V_{IL}	V_{SS}	Hi Z	Program Inhibit

De 2716 heeft 2048 lokaties van acht bits, en daarmee ook acht datapinnen en elf adres-ingangen. De overige aansluitpinnen zijn ingangen die als volgt aangesloten moeten worden.

Vpp : Alleen nodig bij het programmeren van de EPROM. In een schakeling wordt de EPROM alleen maar gelezen, en moet deze pin met de +5V verbonden zijn.

G' : Een hoog op deze ingang brengt de three-state-poorten in hun high impedance state. Moet laag zijn om de EPROM te lezen.

E'/Progr Een hoog op deze ingang brengt eveneens de three-state-poorten in hun high impedance state, maar doet ook het stroomverbruik van normaal 55 mA dalen tot 10 mA.
Het nadeel is dat deze klem trager reageert dan G'.
Ook deze ingang moet mét G' laag zijn om de EPROM te lezen.
Deze ingang wordt ook gebruikt tijdens het programmeren van de EPROM.

In figuur 1.G.2 is een EPROM 2716 aangesloten, en bereikbaar op de adressen 0 tot 2047.

Taak. Teken een opstelling met twee EPROM's 2716 en twee ROM's 68308.

1.H. Het Random Access Memory (RAM)

De RAM is een geheugen dat even gemakkelijk beschreven als gelezen kan worden. Met het signaal op de pin W' of R/W' kan tussen lezen en schrijven gekozen worden.
Het lezen van de inhoud van een lokatie uit de RAM gebeurt net als bij een ROM, maar daarbij moet W' hoog zijn.
Om een lokatie in de RAM te beschrijven moet(en) de CS-ingang(en) op het actieve niveau gebracht worden, het juiste adres op de adres-ingangen aanwezig zijn, de W'-ingang laag zijn, en op de data pinnen de nieuwe te schrijven binaire waarde staan.
Op ieder punt van de geheugenmatrix bevindt zich bij de RAM een bistabiele multivib (flip flop), die in de schrijftoestand (write) kan geset of gereset worden.
In de leestoestand (read) kan de stand van de flip flop's uit de geadresseerde lokatie gelezen worden.
Daar data getransporteerd moet(en) worden van en naar de geheugenmatrix, bevinden zich tussen matrix en data pinnen three-state-poorten in twee richtingen. Het is de W'-klem die de richting van de data-overdracht bepaalt, en de datapinnen met de in- of uitgangen van de flip flop's van de geadresseerde lokatie verbindt.
Door het uitschakelen van de voedingsspanning vergeten de flip flop's hun stand, en gaat de informatie die in de RAM staat verloren. Daarom wordt RAM ook een *vluchtig geheugen (volatile memory)* genoemd.
Daar voor iedere geheugencel veel meer componenten nodig zijn dan bij een ROM, is de opslagcapaciteit per chip veel kleiner dan bij een ROM of PROM. De prijs per bit is uiteraard hoger.

A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	Hexa	Dec.	
0	0	0	0	0	0	0	0	0	0	0	0	000	0	EPROM
0	1	1	1	1	1	1	1	1	1	1	1	7FF	2047	
1	0	0	0	0	0	0	0	0	0	0	0	800	2048	ROM
1	0	1	1	1	1	1	1	1	1	1	1	BFF	3071	

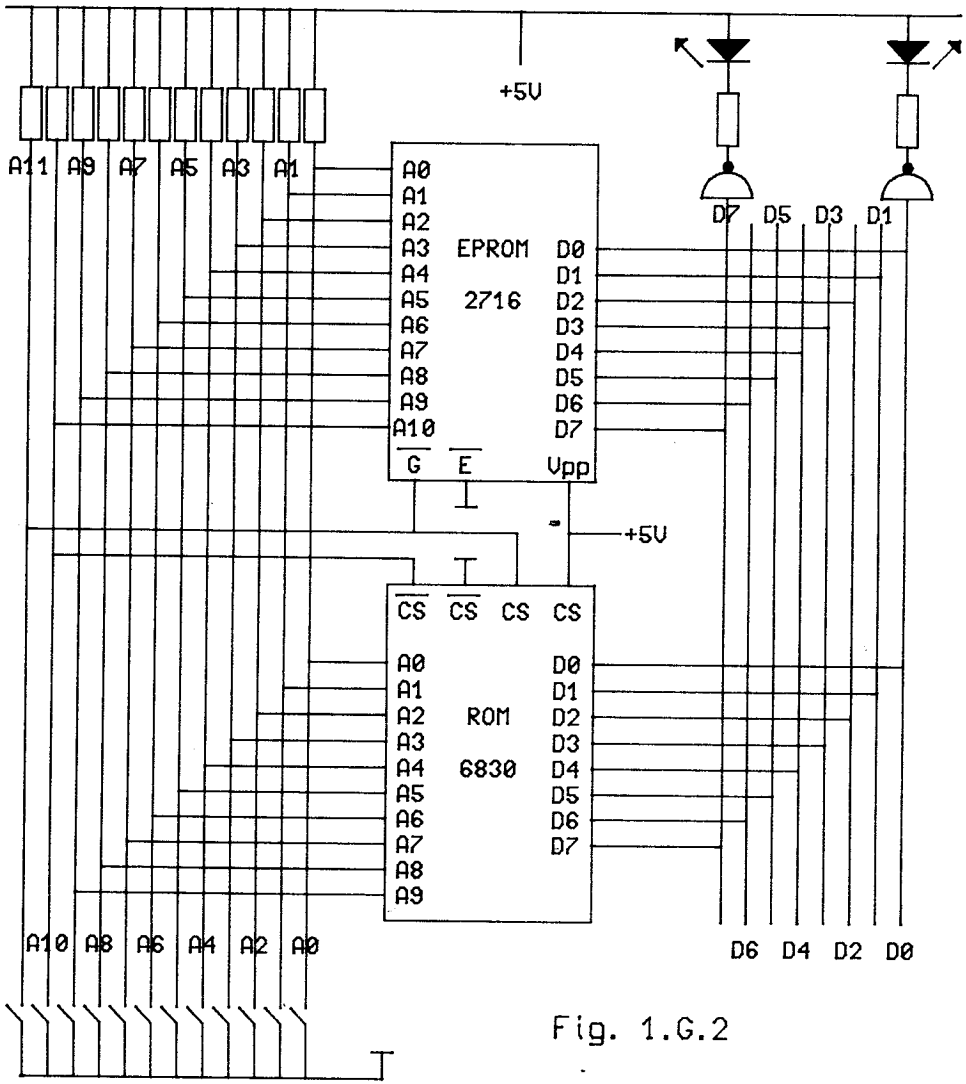


Fig. 1.G.2

1.I. De RAM 6810

Zie voor de data sheets van deze 8-bit RAM met 128 lokaties de figuren 1.I.1 tot 1.I.4. Er zijn 7 adres-ingangen, 8 dataklemmen, een W'-ingang en 6 selectingangen.

Fig. 1.I.1

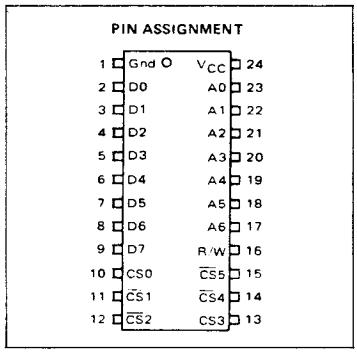
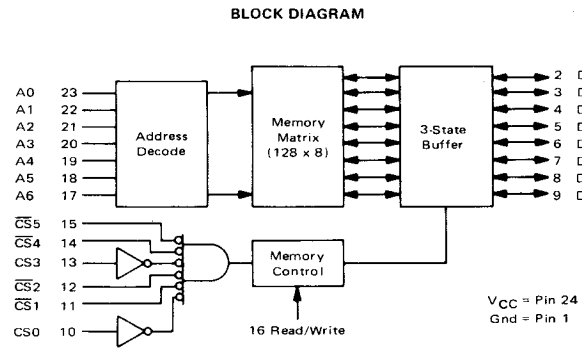


Fig. 1.I.2



AC OPERATING CONDITIONS AND CHARACTERISTICS

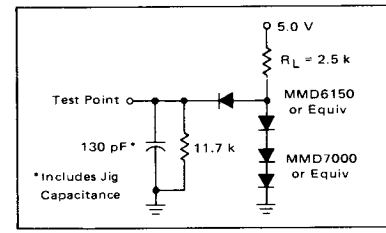
(Full operating voltage and temperature unless otherwise noted.)

Fig. 1.I.3

AC TEST CONDITIONS

Condition	Value
Input Pulse Levels	0.8 V to 2.0 V
Input Rise and Fall Times	20 ns
Output Load	See Figure 1

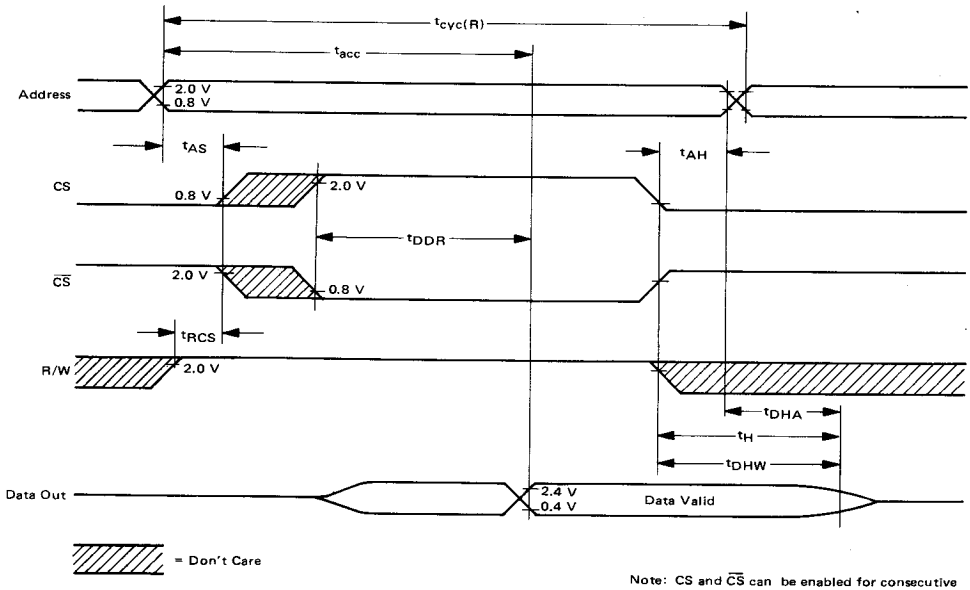
FIGURE 1 – AC TEST LOAD



READ CYCLE

Characteristic	Symbol	MCM6810AL Min	MCM6810AL Max	MCM6810AL1 Min	MCM6810AL1 Max	Unit
Read Cycle Time	$t_{cyc(R)}$	450	–	350	–	ns
Access Time	t_{acc}	–	450	–	350	ns
Address Setup Time	t_{AS}	20	–	20	–	ns
Address Hold Time	t_{AH}	0	–	0	–	ns
Data Delay Time (Read)	t_{DDR}	–	230	–	180	ns
Read to Select Delay Time	t_{RCS}	0	–	0	–	ns
Data Hold from Address	t_{DHA}	10	–	10	–	ns
Output Hold Time	t_H	10	–	10	–	ns
Data Hold from Write	t_{DHW}	10	80	10	60	ns

READ CYCLE TIMING



Note: CS and $\overline{CS}$ can be enabled for consecutive read cycles provided R/W remains at V_{IH}.

WRITE CYCLE

Characteristic	Symbol	MCM6810AL		MCM6810AL1		Unit
		Min	Max	Min	Max	
Write Cycle Time	$t_{cyc(W)}$	450	–	350	–	ns
Address Setup Time	t_{AS}	20	–	20	–	ns
Address Hold Time	t_{AH}	0	–	0	–	ns
Chip Select Pulse Width	t_{CS}	300	–	250	–	ns
Write to Chip Select Delay Time	t_{WCS}	0	–	0	–	ns
Data Setup Time (Write)	t_{DSW}	190	–	150	–	ns
Input Hold Time	t_H	10	–	10	–	ns

WRITE CYCLE TIMING

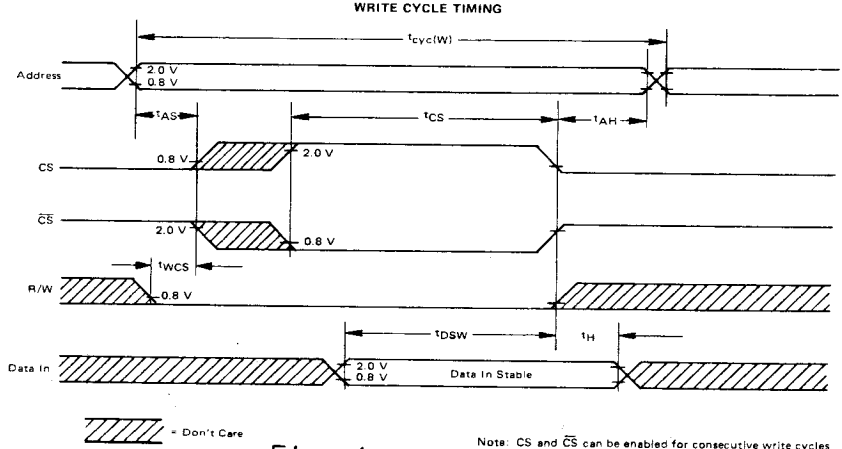


Fig. 1.I.4

Note: CS and $\overline{CS}$ can be enabled for consecutive write cycles provided R/W is strobed to V_{IH} before or coincident with the Address change, and remains high for time t_{AS}.

A7	A6	A5	A4	A3	A2	A1	A0	Hexa	Dec.	
0	0	0	0	0	0	0	0	00	0	RAM 1
0	1	1	1	1	1	1	1	7F	127	
1	0	0	0	0	0	0	0	80	128	RAM 2
1	1	1	1	1	1	1	1	FF	255	

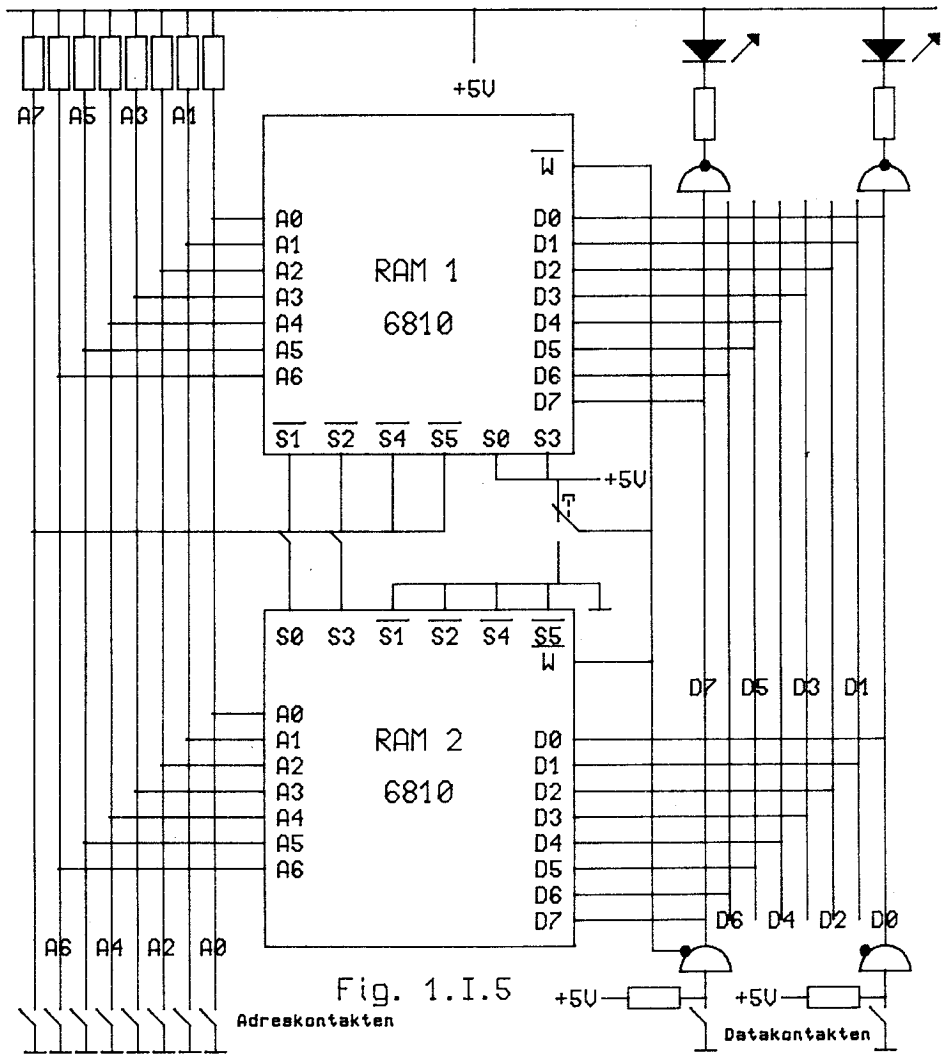


Fig. 1.I.5

Als voeding volstaat een enkele + 5V. In- en uitgangen zijn TTL-compatible. De access time is 450 nsec. De tijd die nodig is voor het beschrijven van een lokatie, bedraagt eveneens 450 nsec, en wordt de *write cycle time tcyc (W)* (schrijfcyclustijd) genoemd.
In figuur 1.I.5 is een opstelling weergegeven met twee RAM's 6810. De adres-ingangen en de selectingangen zijn zodanig aangesloten dat de RAM's te bereiken zijn op de adressen zoals weergegeven in de bijgetekende tabel.
Voor het uitlezen van de RAM's moet de drukknop in de getekende stand staan. Het beschrijven van een lokatie begint met het selecteren van de gewenste lokatie met behulp van de adrescontacten. De leds tonen dan de inhoud van deze lokatie. Dan worden de datacontacten volgens de nieuwe waarde gesteld. Wordt de drukknop nu bediend, dan wordt de W'-lijn laag, en de three-state-poorten brengen de signalen van de datacontacten op de databus. De leds duiden de stand van de datacontacten aan, en de nieuwe waarde wordt in de geadresseerde lokatie geschreven.
Als de drukknop weer losgelaten wordt, komen de RAM's in hun leestoestand en blokkeren de three states. De leds blijven hetzelfde aanduiden, wat bewijst dat de nieuwe waarde in de lokatie geschreven staat.
De three states zijn nodig om in de leesstand de datacontacten van de databus uit te schakelen. Een hoog signaal, dat geleverd wordt door een RAM-lokatie, zou anders door een gesloten datacontact laag gemaakt worden.

Taak 1. Kies de juiste I.C. om tussen de datacontacten en de databus te schakelen in figuur 1.I.5.

Taak 2. Ontwerp een opstelling met 1 ROM en 1 RAM. Als een ROM-lokatie is geselecteerd door de adrescontacten, mogen de three states niet in geleiding komen als op de write-drukknop wordt gedrukt.

1.J. De RAM 2114

Deze RAM heeft 1024 lokaties, maar iedere lokatie kan slechts vier bits bevatten.
Waar de databus 8 bit breed is, moeten twee RAM's 2114 gebruikt worden. Deze worden dan op dezelfde adreslijnen aangesloten, maar elk op een andere helft van de databus zoals in figuur 1.J.1.

Taak. Er bestaan ook RAM's waarvan iedere lokatie slechts één bit kan bevatten. Zoek het nummer van zo'n I.C. op in een data book, en stel daarmee een vervangschema voor die uit figuur 1.J.2. op.

A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	Hexa	Dec.
		0	0	0	0	0	0	0	0	0	0		
		1	1	1	1	1	1	1	1	1	1		

Vul de tabel verder aan.

Op welke adressen is het RAM bereikbaar ?

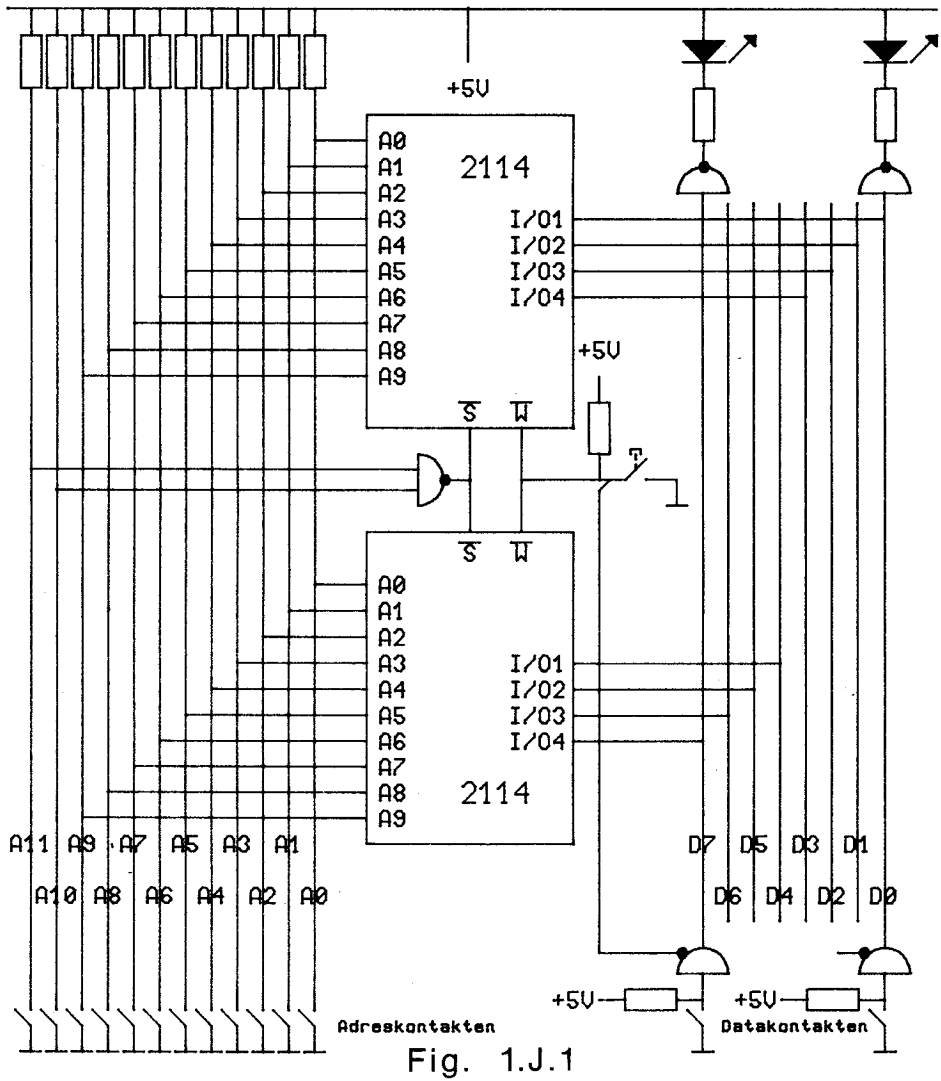


Fig. 1.J.1

1.K. De dynamische RAM

De besproken RAM, waarvan de geheugencellen uit flip flop's bestaan, wordt *statische RAM* genoemd.
Bij een dynamische RAM bestaat iedere geheugencel uit slechts één transistor. In de schrijfstand brengt een 1 op de data-ingang een lading op de basis die de transistor in geleiding brengt. Deze lading blijft slechts enkele milliseconden, en vloeit dan weg.
Daarom moet iedere lading ververst worden voordat hij helemaal is weggevloeid, wat ongeveer om de 2 msec moet gebeuren door een afzonderlijke elektronische schakeling. Deze

bijkomende schakeling wordt een *memory controller for dynamic RAM* genoemd, en komt voor in I.C. onder het nummer 3480. Sommige microprocessors (bijvoorbeeld de Z-80) hebben een ingebouwde memory controller.
De dynamische RAM is sneller, verbruikt minder en is goedkoper per bit dan de statische RAM. Bovendien is het aantal bits per chip beduidend groter, waardoor minder I.C.'s nodig zijn om een groot geheugen op te bouwen.
Een veel gebruikte dynamische RAM is de 4116. Er zijn 16384 lokaties van één bit. De 4116 vraagt wel drie voedingsspanningen.
Om een 8 bit-geheugen op te bouwen zijn er acht 4116's nodig, maar dan zijn er ook 16384 lokaties van 8 bit ter beschikking.
De ingangen CAS' en RAS' worden gestuurd door de memory controller voor het verversen.
Het nadeel van de bijkomende schakeling voor refreshing maakt de dynamische RAM niet interessant wanneer slechts een beperkt geheugen nodig is. Bij toestellen die een grote RAM nodig hebben (bv. de computer, minstens 48K RAM) wordt meestal een dynamische RAM gebruikt.
Er bestaan ook pseudostatische RAM's. Dit zijn dynamische RAM's met ingebouwde memory controller. Het aansluiten is even eenvoudig als bij de statische RAM. Hun eigenschappen liggen tussen die van de statische en die van de dynamische RAM.

2. ANDERS REKENEN

2.A. Het hexadecimaal stelsel

Omdat het binair stelsel voor de mens zeer moeilijk te gebruiken is, wordt bij microprocessors veelal het hexadecimaal stelsel gebruikt.
Een binaire waarde wordt vanaf de meest rechtse bit verdeeld in groepjes van 4 bits. Ieder groepje stelt een waarde voor van 0 tot 15. De tien laagste waarden worden voorgesteld door de decimale cijfers 0 tot 9, en de volgende 6 respectievelijk door A, B, C, D, E en F.

Binair	Hexa	Decim.
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	A	10
1011	B	11
1100	C	12
1101	D	13
1110	E	14
1111	F	15

Als in het zestientallig stelsel een teken een rij naar links opschuift, wordt zijn waarde met 15 vermenigvuldigd.
Voor het omzetten van decimaal naar hexa en omgekeerd, kan de volgende tabel gebruikt worden.

Hexa kolom 4		Hexa kolom 3		Hexa kolom 2		Hexa kolom 1	
Hexa	Decim.	Hexa	Decim.	Hexa	Decim.	Hexa	Decim.
0	0	0	0	0	0	0	0
1	4096	1	256	1	16	1	1
2	8192	2	512	2	32	2	2
3	12288	3	768	3	48	3	3
4	16384	4	1024	4	64	4	4
5	20480	5	1280	5	80	5	5
6	24576	6	1536	6	96	6	6
7	28672	7	1792	7	112	7	7
8	32768	8	2048	8	128	8	8
9	36864	9	2304	9	144	9	9
A	40960	A	2560	A	160	A	10
B	45056	B	2816	B	176	B	11
C	49152	C	3072	C	192	C	12

D	53248	D	3328	D	208	D	13
E	54344	E	3584	E	224	E	14
F	61440	F	3840	F	240	F	15

Als voorbeeld worden enkele binaire waarden omgezet in hexa en in decimaal.

Binair	Hexa	Decimaal
0011 1011	3B	48+11=59
0111 1111	7F	112+15=127
1000 0000	80	128+0=128
1111 1111	FF	240+15=255
0100 0000 0000	400	1024+0+0=1024
0111 1111 1111	7FF	1792+240+15=2047
1000 0000 0000	800	2048+0+0=2048
1111 1111 1111	FFF	3840+240+15=4095
0001 0000 0000 0000	1000	4096+0+0+0=4096
0011 1111 1111 1111	3FFF	12288+3840+240+15=16383
1000 0000 0000 0100	8004	32768+0+0+4=32772
1010 0100 0000 0000	A400	40960+1024+0+0=41984
1010 0111 1010 0111	A7A7	40960+1792+160+7=42919
1111 1000 0000 0000	F800	61440+2048+0+0=63488
1111 1111 1111 1111	FFFF	61440+3840+240+15=65535

Voor het omzetten van decimaal naar hexa kan dezelfde tabel gebruikt worden.

Voorbeeld : decimaal 15000 omzetten in hexa.

```
 15000
-12288      3 uit kolon 4
------
  2712
 -2560        A uit kolon 3
 -----
   152
  -144          9 uit kolon 2
  ----
     8            8 uit kolon 1
```

15000 decimaal is 3A98 hexa.

Het optellen en aftrekken gaat in hexa net als in decimaal. Wel moet er voortdurend rekening mee gehouden worden dat het grondgetal 16 is.
Kennis van hexadecimaal optellen en aftrekken is noodzakelijk voor het programmeren in machinetaal.
Om duidelijk aan te geven of een getal een decimale waarde of een hexawaarde voorstelt, wordt voor een hexawaarde een dollarteken geplaatst of een letter H erachter.
Voer de volgende bewerkingen uit om te wennen aan het hexastelsel. Iedere bewerking is ook vertaald in decimaal. Als proef kan de hexasom in decimaal omgezet worden.

```
  3H      3          3H      3          3H      3
+5H     +5         +9H     +9         +CH     +12
 ---     ---        ---     ---        ---     ---

  3H      3         25H     37         39H     57
+FH     +15        +73H    +115       +CAH    +202
 ---     ---        ----    ----       ----    ----

 3AH     58         A4H    164         78H    120
+CAH    +202       +C5H    +92        -35H    -53
 ----    ----       ----    ---        ----    ----

 CFH    207         B2H    178        100H    256
-AAH    -170        -9FH   -159       -1BH    -27
 ----    ----       ----    ----       ----    ---
```

Negatieve getallen kunnen binair weergegeven worden volgens verschillende methoden. De meest gebruikte en handigste wordt *two's complement* genoemd. Daar bij het verklaren van deze methode vaak verkeerde begrippen ontstaan, kan het best een vergelijking worden gemaakt uit de sfeer van de techniek.
Stelt u zich een wiel voor, waarvan de omtrek verdeeld is in 256 gelijke delen. De delen worden genummerd van 0 tot 255. Er is ook een vaste wijzer die naargelang de stand van het wiel, een der 256 getallen aanduidt, alle positief.
Moeten ook negatieve getallen aangeduid kunnen worden, dan is er aanpassing nodig. Als het wiel in de stand 0 staat, en één deel verdraaid wordt in tegengestelde richting, komt het wiel in de stand -1. In dit vak staat reeds het getal 255, waarnaast nu ook -1 geschreven wordt. Op deze manier kan in dezelfde richting verdergegaan worden, en komen naast de getallen 254, 253, 252 de waarden -2, -3, -4, en zo verder tot de helft van het wiel. De andere helft van het wiel blijft dienen voor het aanwijzen van positieve getallen.
Op de buitenomtrek van het wiel staan nu twee schalen getekend, die er uitgevouwen uitzien als in figuur 2.A.1.
De bovenste schaal loopt van 0 tot 255 en wordt gebruikt als er alleen met positieve waarden wordt gewerkt.
Als ook negatieve getallen moeten worden aangeduid is alleen de onderste schaal geldig. Zijn bereik gaat van 0 tot 127 voor positieve waarden, en van -1 tot -128 voor negatieve.

De buitenomtrek van een tweede wiel wordt eveneens in 256 gelijke delen verdeeld. De nummering gebeurt nu binair, waarbij slechts twee karakters ter beschikking staan, en geen minteken, zoals in figuur 1.A.2.
Moeten alleen positieve getallen aangeduid worden, dan is het bereik van 0000 0000 (00H) tot 111 111 (FFH). Als zowel positieve als negatieve getallen aangeduid moeten worden, dan is het positief bereik beperkt van 0000 0000 (00H) tot 0111 1111 (7FH).
Het negatief gebied ligt vanaf de binaire waarde 1111 1111 (FFH) tot 1000 0000 (80H). De binaire waarde 1111 1111 stelt dan -1 decimaal voor en 1000 0000 komt overeen met -128 decimaal.
Alle binaire waarden vanaf 1000 0000 (80H) tot 1111 1111 (FFH) kunnen zowel een positieve als een negatieve waarde voorstellen.
Daarom moet men voor een eerste wiskundige bewerking beslissen of er alleen met positieve waarden (unsigned) gewerkt zal worden, of dat er ook negatieve waarden gebruikt zullen worden (signed).

Deze stelling moet dan bij alle volgende bewerkingen voortdurend aangehouden worden.
Kenmerkend voor het beschreven two's-completement-systeem is, dat van alle positieve getallen de meest linkse bit een 0 is, en die van de negatieve getallen een 1.
De negatieve waarde van een binair getal kan worden gevonden door eerst iedere bit van het getal om te keren, en er daarna 1 bij te tellen.

Voorbeeld

```
0000 0010    dec.+2                  0111 1110    dec.+126

1111 1101    omgekeerd               1000 0001    omgekeerd
       +1    een bijtellen                  +1    een bijtellen
---------                            ---------
1111 1110    dec.-2                  1000 0010    dec.-126
```

De negatieve waarde van een getal dat in hexa staat, kan nog eenvoudiger gevonden worden door het getal af te trekken van 100H.

Voorbeeld

```
 02H  =  dec.+2                       7EH  =  dec.+126

100H                                 100H
-02H                                 -7EH
----                                 ----
 FEH  =  dec.-2                       82H  =  dec.-126
```

Het two's complement system heeft het grote voordeel dat positieve en negatieve waarden zonder problemen opgeteld kunnen worden. Indien daarbij de verkregen som drie digits bevat, zijn slechts de rechtse twee geldig.

Voorbeeld

```
 7FH  =  dec.+127                     7DH  =  dec.+125
+FEH  =  dec.  -2                     81H  =  dec.-127
----          ----                   ----          ----
17DH                                  FEH  =  dec.  -2
 7DH  =  dec.+125
```

Het bereik van het two's complement system met 8 bits loopt van − 12 dec. tot + 127 dec. Als de som van een bewerking buiten dit bereik komt, is zij ongeldig, en wordt van *overflow* gesproken.

Voorbeeld

```
 7EH  =  dec.+126                     82H  =  dec.-126
+02H  =  dec.  +2                    +FDH  =  dec.  -3
----                                 ----
 80H  =  dec.-128 overflow           17FH
                                      7FH  =  dec.+127 overflow
```

Taak 1. Teken het ontwikkelde wiel voor two's complement met 4 bits. Wat zijn de bereiken? Maak een bewerking waarbij overflow optreedt.
Taak 2. Idem voor 16 bits.

2.B. De adder

De adder is een elektronische schakeling die 2 bits bij elkaar optelt, en voorgesteld wordt zoals in figuur 2.B.1. De bij elkaar op te tellen bits moeten worden aangeboden op de ingangen A en B. Het resultaat komt ter beschikking op de uitgangen F en C. De uitgang C is de overdracht of carry.

2.C. De full adder

Als twee binaire waarden van elk 4 bits (A3 A2 A1 A0 B3 B2 B2 B0) bij elkaar opgeteld moeten worden, kunnen de bits A0 en B0 met een adder verwerkt worden. Bij de bits A1 en B1 moet ook de carry C0 meegeteld worden, zodat een schakeling nodig is zoals in figuur 2.C.1. die een *full adder* genoemd wordt.
De bits A2 en B2 moeten bij elkaar opgeteld worden met de carry C1, waarvoor weer een full adder nodig is. Ook bij A3 en B3 moet C2 meegeteld worden, weer door middel van een full adder.
Een full adder kan ook als adder (ook *half adder* genoemd) gebruikt worden, als zijn C0-ingang laag gehouden wordt.
De adder en de 3 full adders (of 4 full adders) kunnen samen in een I.C. ondergebracht zijn (bv. 7483) zoals voorgesteld in figuur 2.C.2.

2.D. De ALU

Een Arithmetic and Logic Unit voor 4 bits kan eveneens 2 binaire waarden van 4 bits bij elkaar optellen, maar kan ze ook van elkaar aftrekken. Ook logische bewerkingen met de ingangswaarden als AND, OR, NAND, NOR, NOT enz. kan de ALU uitvoeren.
Welke bewerking de ALU moet uitvoeren, kan bepaald worden met de selectie-ingangen S3 tot S0.
Figuur 2.D.1 toont het symbool van een 4-bits ALU. Naast de uitgang C4 (carry) zijn ook nog de uitgangen N (Negatief), Z (Zero) en V (oVerflow) beschikbaar.
Uitgang N wordt hoog als het resultaat van de uitgevoerde bewerking negatief is. Hier, met 4 bits, is dit het geval als F4 hoog is.

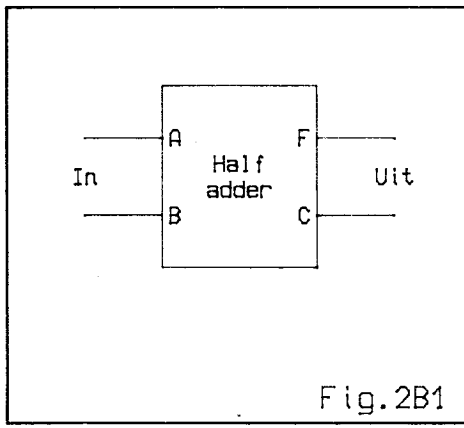


Fig.2B1

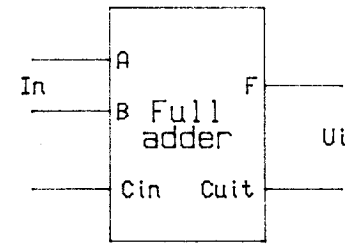


Fig.2C1

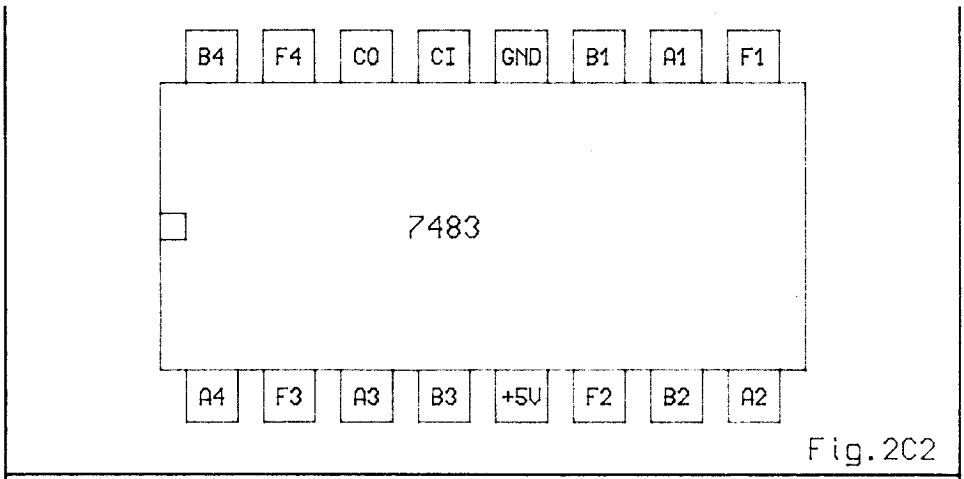


Fig.2C2

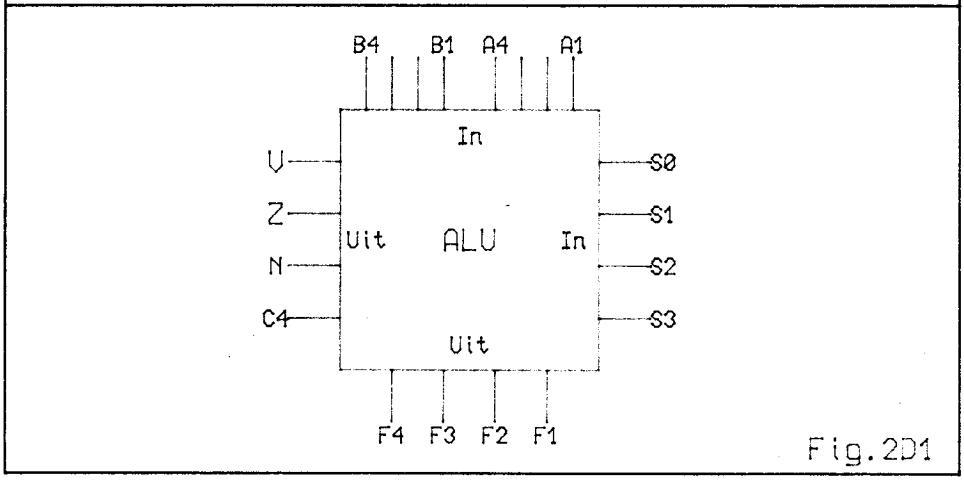


Fig.2D1

Uitgang Z wordt hoog als het resultaat nul is, dus als de uitgangen F4, F3, F2 en F1 alle nul zijn.
De uitgang V wordt hoog als het resultaat groter is dan +7 dec. of kleiner dan −8 dec. Alleen als men met signed numbers werkt, moet men na iedere bewerking de V-uitgang nagaan. Als het om unsigned numbers gaat, hoeft dit niet.
Onderstaande tabel toont de invloed van de selectie-ingangen bij een fictieve ALU, telkens aangevuld met een voorbeeld.

S3	S2	S1	S0	F=	Functie	F4	F3	F2	F1	C	N	Z	V
0	0	0	0	A plus B	Optellen	1	0	0	0	0	1	0	1
0	0	0	1	A min B	Aftrekken	0	1	1	0	0	0	0	0
0	0	1	0	A	Transport	0	1	1	1	0	0	0	0
0	0	1	1	B	Transport	0	0	0	1	0	0	0	0
0	1	0	0	A plus 1	Increment	1	0	0	0	0	1	0	1
0	1	0	1	A min 1	Decrement	0	1	1	0	0	0	0	0
0	1	1	0	B plus 1	Increment	0	0	1	0	0	0	0	0
0	1	1	1	B min 1	Decrement	0	0	0	0	0	0	1	0
1	0	0	0	-A	Negate	1	0	0	1	0	1	0	0
1	0	0	1	-B	Negate	1	1	1	1	0	1	0	0
1	0	1	0	A'	NOT	1	0	0	0	-	-	0	-
1	0	1	1	B'	NOT	1	1	1	0	-	-	0	-
1	1	0	0	A + B	OR	0	1	1	1	-	-	0	-
1	1	0	1	A + B	NOR	1	0	0	0	-	-	0	-
1	1	1	0	A * B	AND	0	0	0	1	-	-	0	-
1	1	1	1	A * B	NAND	1	1	1	0	-	-	0	-

Voorbeeld met
A = 0 1 1 1
B = 0 0 0 1

3. DE HYPOTHETISCHE MICROPROCESSOR

3.A. Inleiding

De werking van een microprocessor (ook *Central Processing Unit* genoemd) is beslist niet eenvoudig. Daarom is ter kennismaking voor een zeer eenvoudige, niet-bestaande (hypothetische) microprocessor gekozen, die de naam *Hypo* gekregen heeft.
Wat in Fig. 3.G.2. binnen het kader ligt, is het inwendige van Hypo.
Voor de overzichtelijkheid zijn de mogelijkheden van Hypo zeer beperkt gehouden. Zo heeft hij slechts vier adreslijnen en evenveel datalijnen. Deze bundels lijnen worden de *adresbus* en de *databus* genoemd.
Via de databus en de adresbus is Hypo met een RAM verbonden, die ook maar 4 bits breed is.

3.B. Accu A

Heeft de eigenschappen van een RAM-lokatie, en kan door 3-state-poorten die in twee richtingen leiden, verbonden worden met de databus. Eenzelfde verbinding is er tussen accu A en de ALU.
Accu A kan bijgevolg een 4-bit waarde leveren aan of krijgen van de databus. Dezelfde mogelijkheid bestaat voor dataverkeer tussen A en de ALU.

3.C. Register B

Heeft ongeveer dezelfde bouw en eigenschappen als accu A. Register B wordt minder gebruikt dan accu A. Zo komt het resultaat van bewerkingen die zijn uitgevoerd door de ALU, altijd in A terecht, nooit in register B. Daardoor wordt B ook wel een *hulpregister* genoemd, en heet A de *accu* (accumulator = verzamelaar).

3.D. De ALU

Accu A en register B kunnen hun inhoud op de ingangen van de ALU brengen. Naargelang de bewerkingswijze waarop de ALU is ingesteld, zal deze de waarden van A en/of B verwerken. Het resultaat van de bewerking wordt door de ALU altijd in accu A geschreven.

3.E. Het instructieregister

Welke bewerking Hypo moet uitvoeren, wordt hem kenbaar gemaakt door een code, de bewerkingscode of operatiecode, naar het instructieregister te brengen. De operatiecode (afgekort tot opcode) is door de gebruiker in de RAM geschreven in de vorm van een 4-bit-waarde. De opcodes die in het instructieregister komen, worden door de instructie-decoder verwerkt tot verschillende signalen. Deze signalen stellen de ALU in op de gewenste bewerkingswijze (optellen, aftrekken, omkeren), verbinden accu A en/of register B met de databus of de ALU, enz.. Daardoor zal Hypo de bewerking die overeenkomt met de opgegeven opcode, uitvoeren.

3.F. De Program Counter

Dit is een 4 bit-teller, die wordt verhoogd telkens wanneer een bewerking is uitgevoerd. De program counter brengt zijn inhoud op de databus, en bepaalt daarmee welke RAM-lokatie bereikbaar is.
Alle gebeurtenissen in Hypo geschieden één voor één in een tempo dat aangegeven wordt door klokpulsen. Deze pulsen worden opgewekt door een klokpulsgenerator die niet is weergegeven in figuur 3.G.2.

3.G. Werking van de CPU Hypo.

Stel dat Hypo de som van 3 en 5 moet berekenen. Het is duidelijk dat hier de bewerking ADD zal moeten worden uitgevoerd door de ALU.
Daarvoor moet eerst de waarde 3 naar A gebracht worden, en daarna de waarde 5 naar B. Dan pas kan de ALU de som maken die hij automatisch in accu A zal plaatsen. Nu moet de som nog vanuit A naar een vrije lokatie in de RAM gebracht worden, waar de gebruiker het getal kan lezen.
Een beter overzicht in de volgorde der bewerkingen wordt verkregen door het opstellen van een flowchart. In een flowchart wordt iedere bewerking voorgesteld door middel van een meetkundige figuur.
Het totaal van de bewerkingen die nodig zijn om een bepaalde taak uit te voeren, wordt een *programma* genoemd.
Begin en einde van een programma worden aangeduid met een ovaal.
Het brengen van waarden in de CPU (input) en het halen van waarden uit de CPU (output) wordt door een parallellogram aangegeven.
De bewerkingen die in de CPU zelf gebeuren worden door een rechthoek voorgesteld.
In de figuren wordt het gebeurde op eenvoudige wijze verklaard.
Om verklaringen bij bewerkingen kort te houden, en voor iedereen begrijpelijk te maken, heeft iedere bewerking haar afgekorte omschrijving (Mnemonic). Zo wordt het laden van A met de waarde 3 aangegeven met LD A,3. Het maken van de som door ADD A,B. Eerst komt de soort bewerking, daarna de bestemming, gevolgd door een komma en de bron.
In figuur 3.G.1. is de flowchart weergegeven met in de figuren de uitgebreide omschrijving en naast de figuren de mnemonics.
De figuren 3.G.2 tot 3.G.16 tonen, hoe stap voor stap het programma door Hypo wordt afgewerkt om de som van 3 en 5 in de RAM te krijgen.
Figuur 3.G.2. geeft de begintoestand van het programma weer, zoals dat zich in de RAM bevindt. Adres 0 bevat de opcode voor LA A,x. De waarde die in A geladen moet worden, staat direct achter de opcode, hier in lokatie 1. Op adres 2 staat de opcode voor LD B,x, direct gevolgd door de te verwerken waarde (ook *operand* genoemd) op adres 3.
De lokatie met adres 4 bevat de opcode die Hypo beveelt, de som te maken van A en B. Bij deze opcode hoort geen operand zodat op adres 5 weer een opcode volgt die Hypo beveelt, de som naar de RAM te brengen.

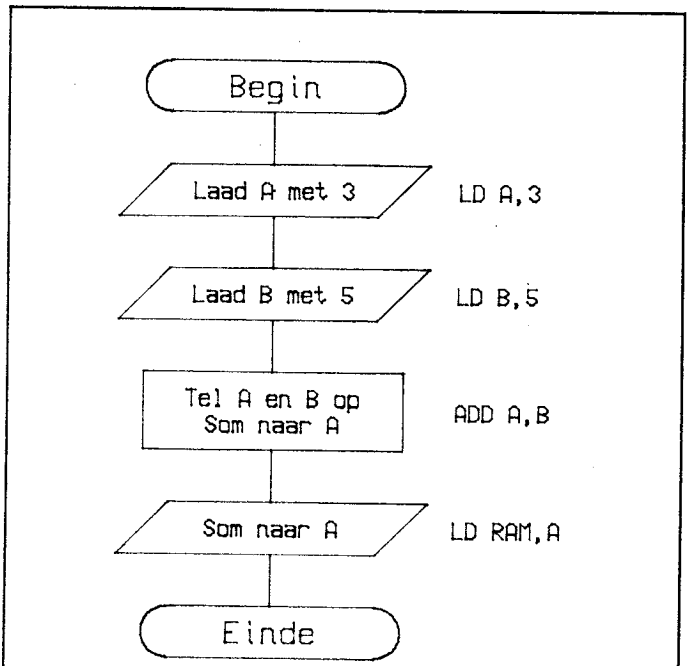


Fig.3G1

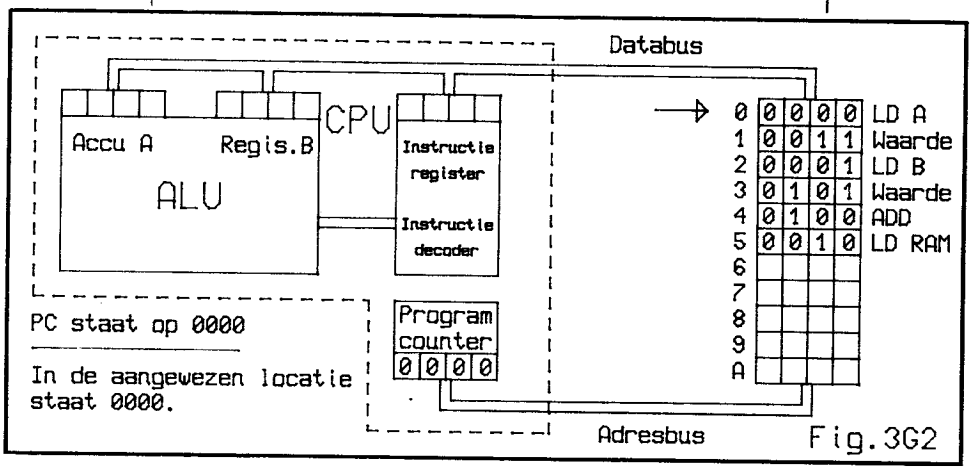


Fig.3G2

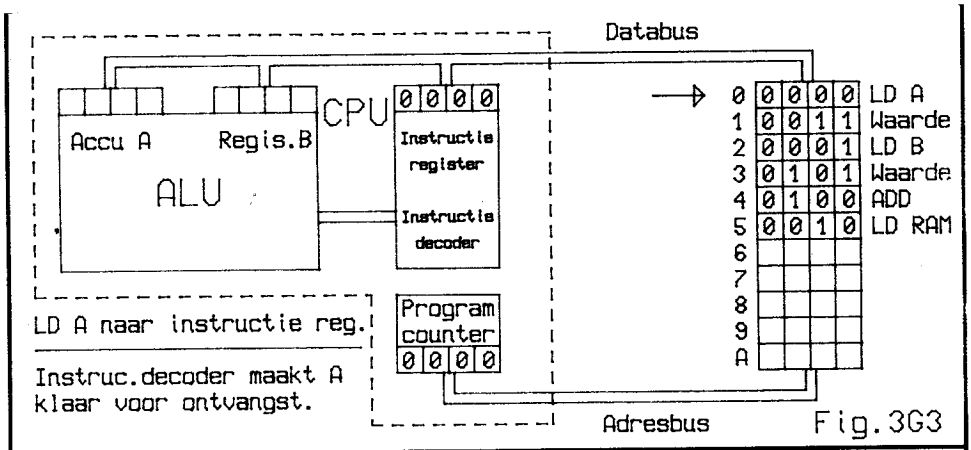


Fig.3G3

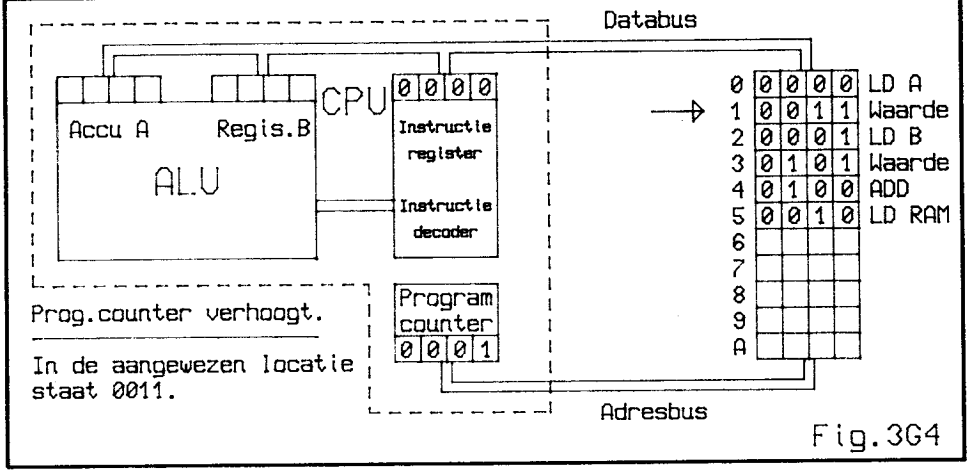


Fig.3G4

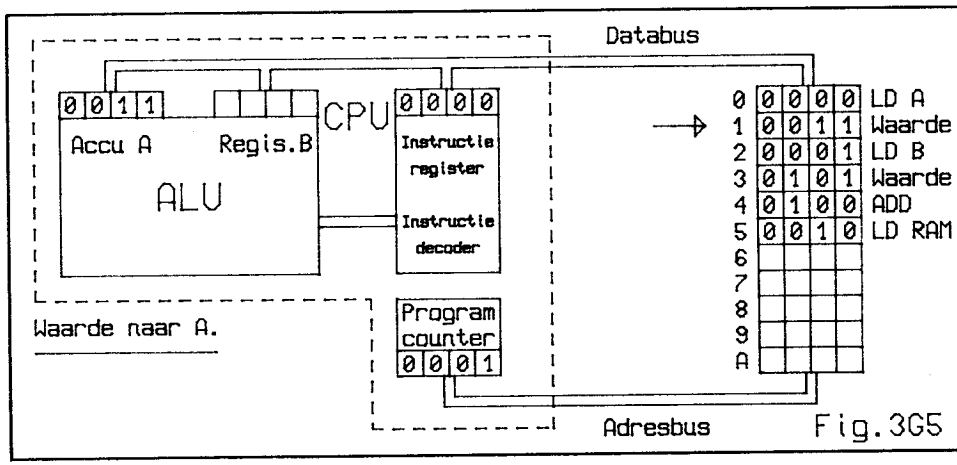


Fig.3G5

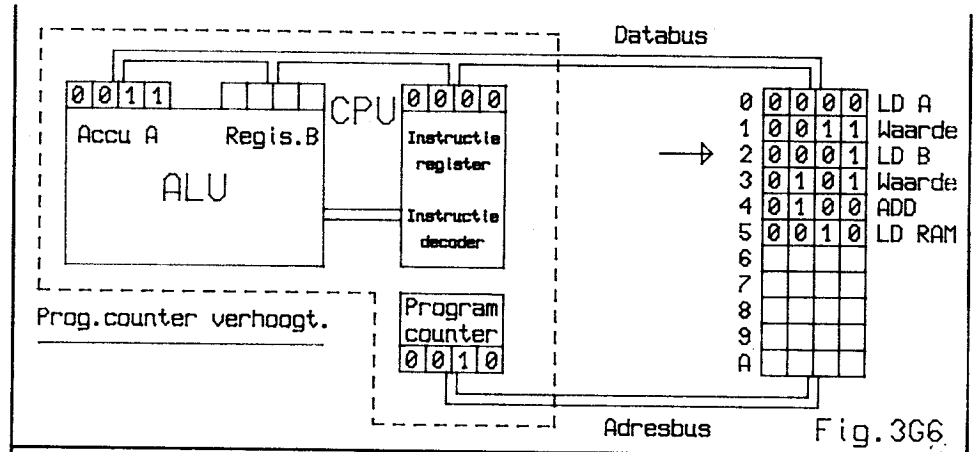


Fig.3G6

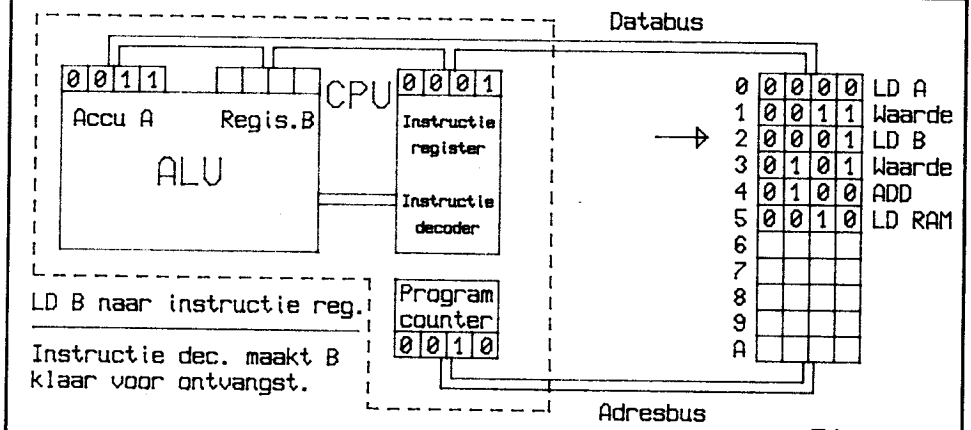


Fig.3G7

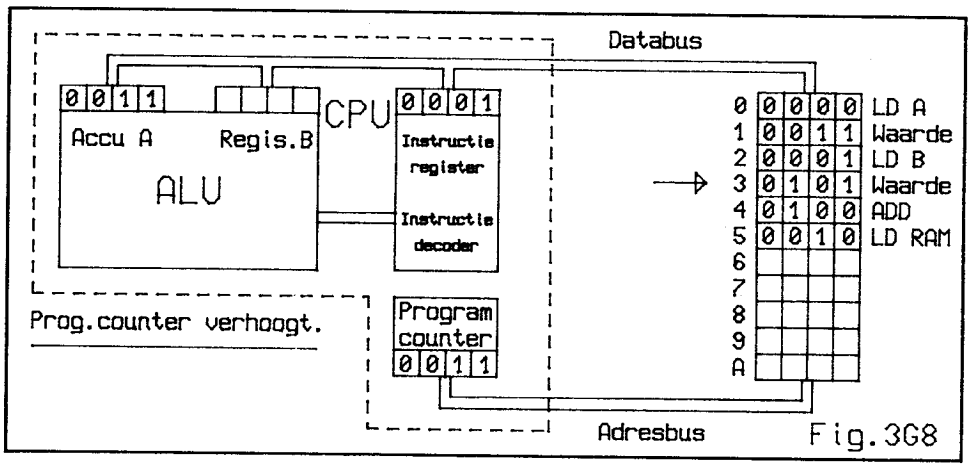


Fig.3G8

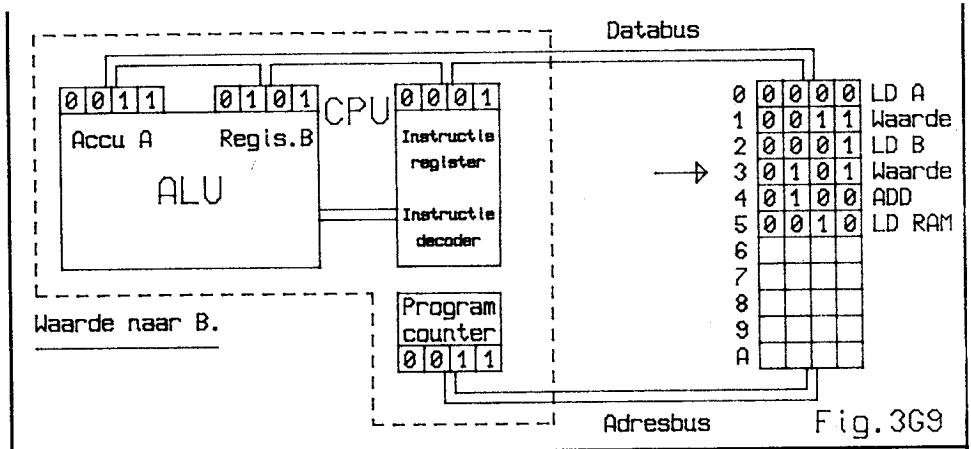


Fig.3G9

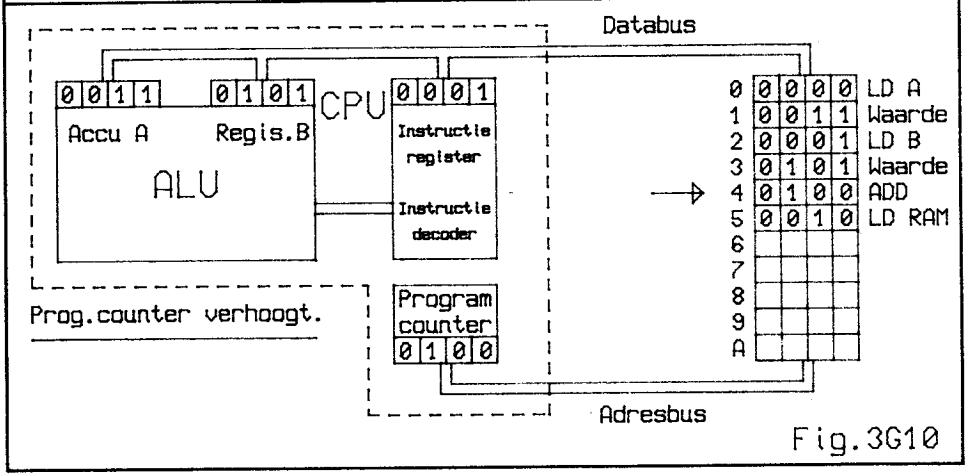


Fig.3G10

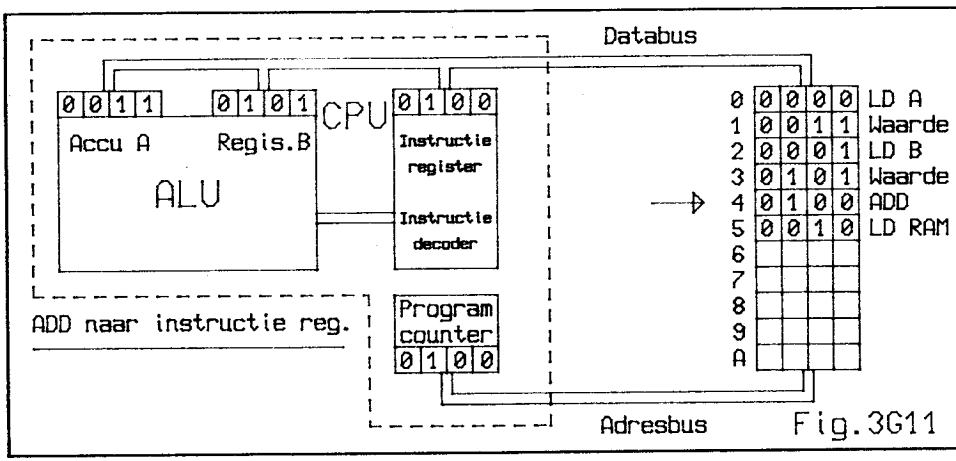


Fig.3G11

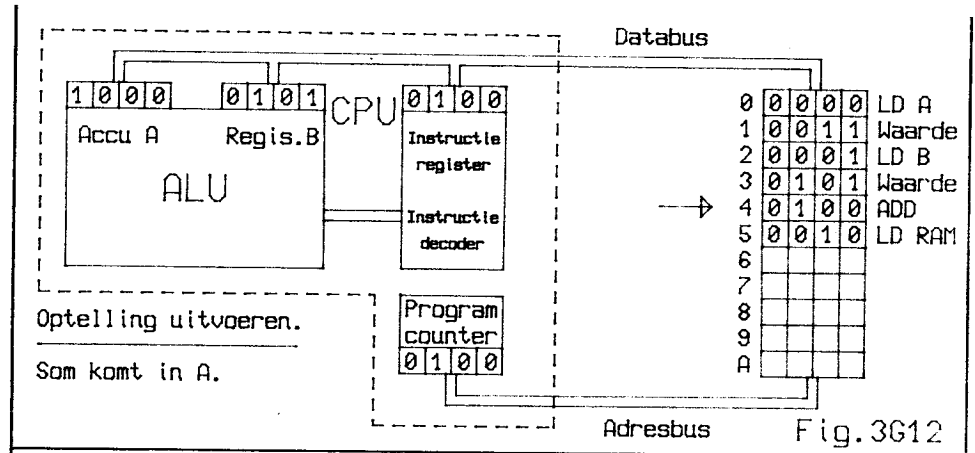


Fig.3G12

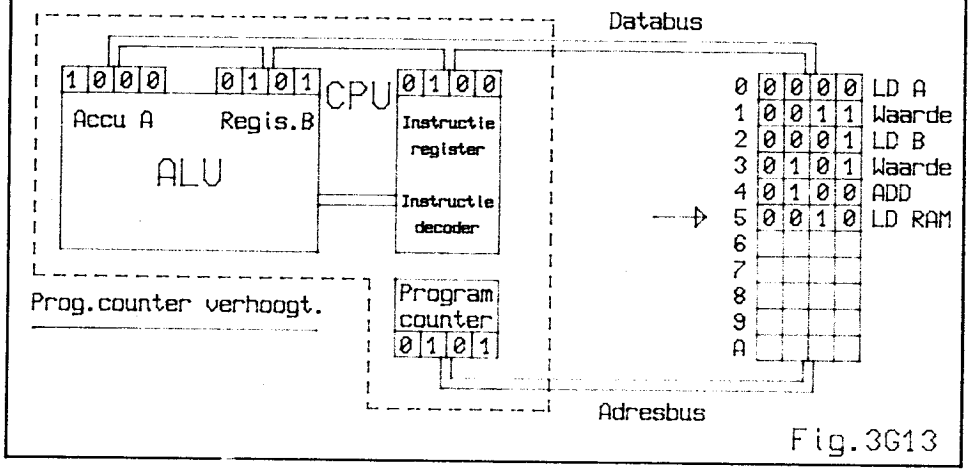


Fig.3G13

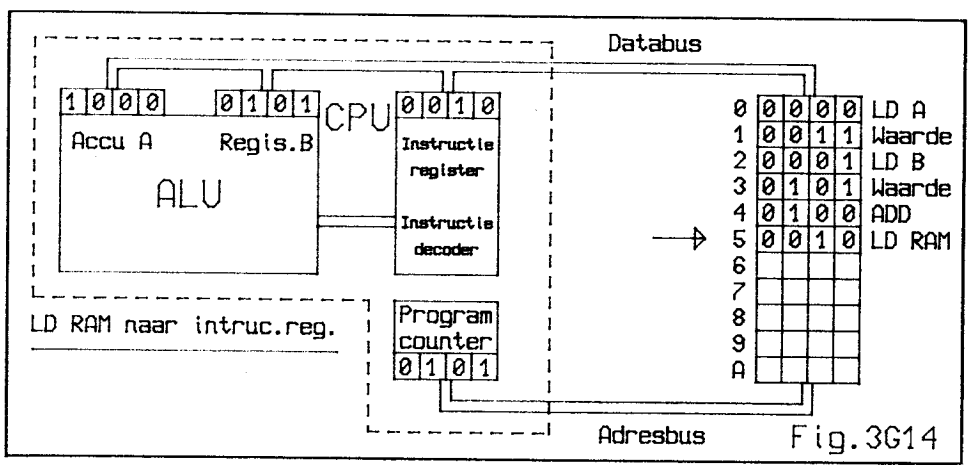


Fig.3G14

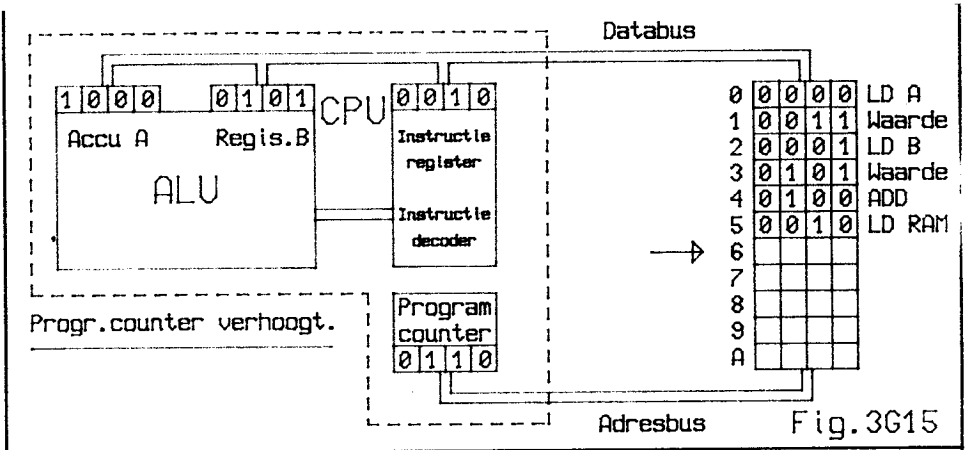


Fig.3G15

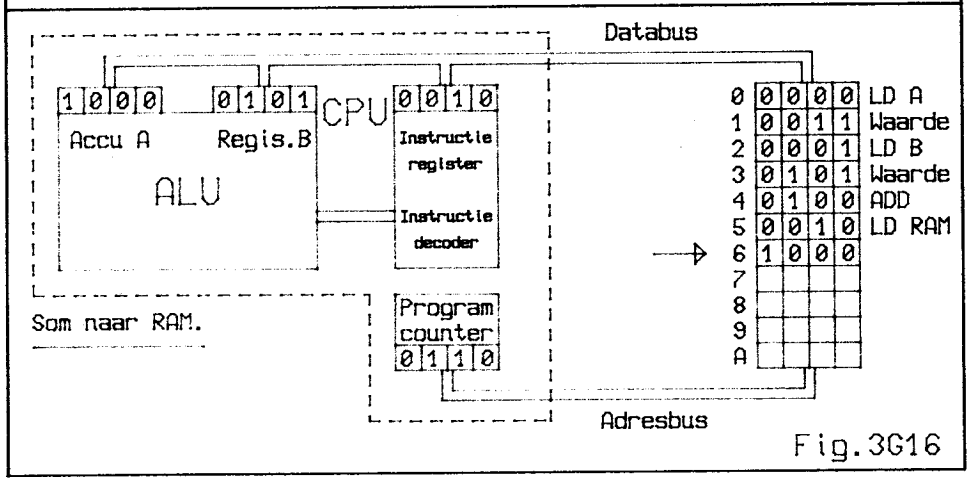


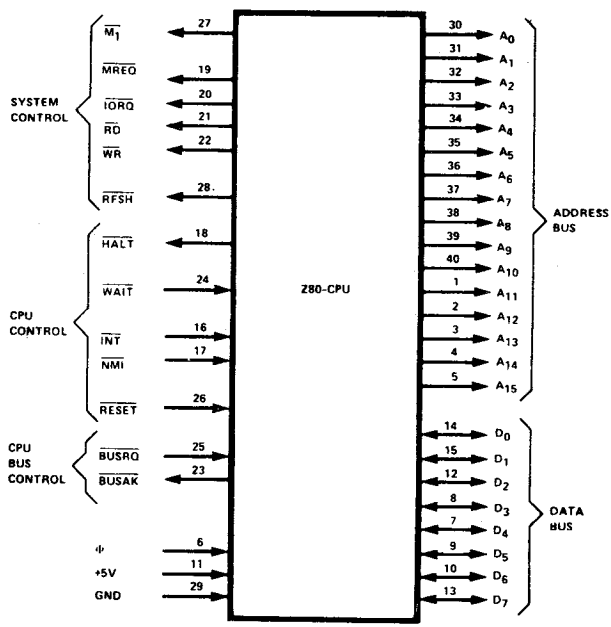
Fig.3G16

4. DE MICROPROCESSOR Z-80

4.A. Het blokdiagram

Dit geeft een overzicht van de inhoud en de architectuur van een CPU. In een blokdiagram moet te zien zijn, welke registers de CPU bevat en hoe breed (aantal bits) ze zijn, hoeveel datalijnen, adreslijnen en controllijnen er aanwezig zijn enz.
Alleen de kringen en registers die voor de gebruiker van belang zijn worden weergegeven. In werkelijkheid bevat de CPU nog meer kringen en registers, maar daar ze alleen een inwendige taak vervullen worden ze niet weergegeven en besproken.
Figuur 4.A.1 toont het blokdiagram van de Z-80 en figuur 4.A.2 de aansluitingen van de Z-80. De in- en uitgangen van de Z-80 zijn op TTL-niveau. De Z-80 heeft inwendig een schakeling voor het verversen van dynamische RAM. De Micro-Professor heeft alleen statische RAM, en maakt daarom geen gebruik van de uitgang RFSH'.

Fig. 4.A.2



Z80 CPU PIN-OUTS

In de Z-80 is geen klokgenerator ingebouwd, maar een enkel signaal op ingang I volstaat. De maximum frequentie is 2.5 Mhz, maar er is ook een snellere versie die tot 4 Mhz loopt. Bij de Micro-Professor wekt een eenvoudige generator een frequentie van 3.58 Mhz op, die door een tweedeler op 1.79 Mhz wordt gebracht, en op ingang I wordt aangesloten (zie sheet 1 schema). Een enkele voeding van +5V volstaat, die geleverd wordt door een stabilisatie IC 7805 (sheet 4 schema Micro-Professor). De genoemde sheets zijn te vinden op de laatste 4 pagina's van Hoofdstuk 10.

4.B. De databus

De 8 transceivers maken de verbinding tussen het inwendige van de Z-80 en de databus via de klemmen D7-D0 (fig. 4.A.1).

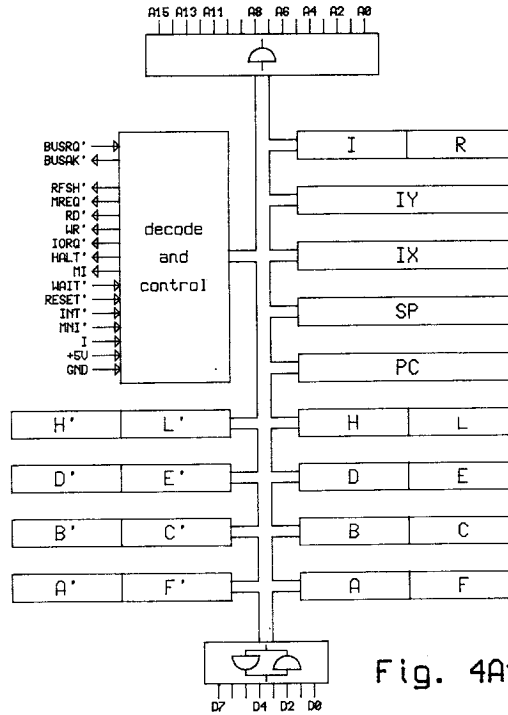


Fig. 4A1

Op de databus zijn de geheugens aangesloten (sheet 1). Zo ook de 8255 (sheet 2) voor het sturen van de display en het toetsenbord en de PIO en een timer counter (sheet 3). Via de bovenste connector kan op de databus afgetakt worden (sheet 4).

4.C. De adresbus

De Z-80 kan een adresbus sturen die 16 bit breed is, en is daarmee in staat om 65536 adressen aan te wijzen. Klemmen A15-A0, de adresbuffers, zijn three states, waardoor de Z-80 van de adresbus afgeschakeld kan worden.
De lijnen A0-A11 zijn op de adres-ingangen van de geheugens aangesloten. Via de lijnen A12-A15 worden, na decodering, de *chip-select-ingangen* gestuurd. Ook de adresbus is via de bovenste connector bereikbaar.

4.D. De Accu (A)

De Z-80 heeft een accu van 8 bit breed. De accu heeft meer mogelijkheden dan de andere registers, en wordt bij de meeste bewerkingen betrokken. Ook komt het resultaat van de meeste wiskundige en logische bewerkingen in A terecht.
De ALU, die zeer veel mogelijkheden heeft, is ook 8 bit breed. Deze staat niet in het blok-

diagram, aangezien het een inwendig register is dat niet rechtstreeks door de gebruiker benaderd wordt.

4.E. De Program Counter (PC)

De PC is hier een 16 bit register, waarvan de inhoud op verschillende manieren veranderd kan worden. De inhoud van de Program Counter wordt doorgegeven aan de adresbus.

4.F. General purpose registers (B, C, D, E, H, L)

Deze registers voor algemeen gebruik zijn 8 bit breed. Hun mogelijkheden liggen ergens tussen die van de accu en van RAM-lokaties, en zullen vooral duidelijk worden bij de studie van de instructies.
Deze registers kunnen ook per paar gebruikt worden, als 16 bit register BC, DE en HL. Daarin kunnen dan 16 bit bewerkingen gebeuren of een volledig 16 bit adres plaatsvinden.

4.G. Flag register (F)

Een 8 bit register, waarvan er 6 als flag werken ; de stand van deze 6 wordt bepaald door het resultaat van de bewerkingen. Alle instructies beïnvloeden niet iedere flag. Voor zekerheid moet de instruction set geraadpleegd worden.

Bit 0 C Carry flag. Is de carry van bit 7 uit accu A.

Bit 1 N Substract flag. Wordt hoog als een aftrekking wordt verricht. De meeste andere instructies maken flag N laag. Alleen van belang bij BCD-bewerkingen.

Bit 2 P/V Vervult twee functies.
Bij instructies voor wiskundige bewerkingen wordt deze flag hoog bij oVerflow.
Bij logische bewerkingen wordt deze bit gebruikt als parity flag. Deze wordt dan hoog als het aantal hoge bits in een resultaat even is. Een oneven aantal hoge bits in een resultaat maakt de P/V flag laag.

Bit 3 Wordt niet gebruikt.

Bit 4 H Half carry flag. Wordt hoog als er tijdens een wiskundige bewerking een carry is van bit 3 naar bit 4.
Alleen van belang bij BCD-bewerkingen.

Bit 5 Wordt niet gebruikt.

Bit 6 Z Zero flag. Wordt hoog als het resultaat van een bewerking 00 is, of als een 00 verplaatst wordt.

Bit 7 S Sign flag. Wordt hoog als het resultaat negatief is. Alleen van belang als in two's complement gewerkt wordt, of om te controleren of bit 7 van een resultaat hoog is.

Enkele instructies behandelen het F-register samen met A. In die zin vormen A en F een paar.

4.H. Prime registers (A' F' B' C' D' H' L')

Naast de bekende registers A, F, B, C, D, E, H, L, heeft de Z-80 nog eens dezelfde reeks registers, die aangeduid worden met A', F', B', C', D', H', L'.
De prime registers kunnen hun inhoud alleen verwisselen met de inhoud van de overeenkomstige non-prime registers.
Voor het maken van een tussenbewerking kan de inhoud van de non-prime registers, die niet verloren mag gaan, eerst in de prime registers worden geplaatst. Nadat de tussenbewerking is gemaakt en het resultaat is opgeborgen, kan de originele inhoud van de non-prime registers teruggehaald worden uit de prime registers.

4.I. Ingang BUSRQ'

Als ingang BUSRQ' laag wordt gemaakt, komen de adresbuffers, de databuffers en de controluitgangen in hun high impedance state. De Z-80 bevestigt dat deze toestand is ingetreden, door de uitgang BUSAK' laag te maken. In deze toestand kan de RAM rechtstreeks geladen worden (DMA Direct Memory Access), bijvoorbeeld door een bedieningspaneel met schakelaars.
Er zijn ook nog andere manieren om de RAM te laden, waarbij de CPU niet uitgeschakeld wordt, en zelf bij het laden optreedt.
De Micro-Professor past deze laatste methode toe, en maakt doordoor geen gebruik van de klemmen BUSREQ' en BUSAK' (sheet 1).

4.J. Uitgang MREQ'

Dit is een 3-state-uitgang die laag wordt, als een geldig adres door de Z-80 op de adresbus wordt geplaatst.
MREQ' is aangesloten op de decoder voor de chip selects, en verhindert dat een geheugen data levert zolang geen geldig adres op de adresbus staat.

4.K. Uitgang WR'

Deze wordt laag met de MREQ', als de Z-80 in het geheugen wil schrijven. De WR' is aangesloten op de W'-ingang van de RAM (sheet 1). WR' kan ook samen met IOREQ' laag worden.

4.L. Uitgang RD'

Wordt laag samen met MREQ' en IOREQ', als de Z-80 een leesopdracht uitvoert.
Wordt bij de Micro-Professor niet gebruikt voor het geheugen, wel voor de 8255 (sheet 2), de PIO en CTC (sheet 3).

4.M. Uitgang IORQ'

De Z-80 kan ook data op de databus plaatsen die niet bestemd zijn voor het geheugen, maar voor de randtoestellen. In zo'n geval wordt MREQ' niet laag, maar wel IORQ'.
Bij de Micro-Professor zijn de randtoestellen de 8255, en in optie de PIO (U10) en de counter timer (U11) (sheets 2 en 3).
IOREQ' is samen met A6 en A7 aangesloten op een decoder (U9b).
De chip selects van de 8255, de PIO en de timer counter kunnen alleen laag worden op het ogenblik dat IORQ' ook laag is.
Naargelang er uitgevoerd wordt naar, of ingevoerd wordt van de randtoestellen, zal RE' of WR' samen met IORQ' laag zijn.

4.N. Uitgang M1'

Iedere eerste byte van een instructie is de opcode. Gedurende de tijd dat de Z-80 de opcode leest en decodeert (fetch cycle), is M1' laag. Randtoestellen weten door dat signaal dat ze op dat ogenblik niet zullen worden aangesproken.

4.O. Uitgang HALT'

De instructie HALT, met opcode 76, brengt de Z-80 in een passieve toestand en maakt dit duidelijk aan de buitenwereld door zijn uitgang HALT' laag te maken.
Bij de Micro-Professor is op deze uitgang, via een transistor, een rode led aangesloten. Deze HALT-led licht op als het signaal HALT' laag wordt (sheet 1).

4.P. Overige registers en signalen

De indexregisters IX en IY, de stack pointer en het I (Interrupt)- en R(Refresh)-register kunnen beter besproken worden nadat enige ervaring in het programmeren is opgedaan. Ook het doel van de nog niet besproken aansluitklemmen RST', INT' en NMI' wordt later behandeld.

5. HET GEBRUIK VAN DE MICRO-PROFESSOR

5.A. Adresbezetting

0000 07FF	EPROM Monitor
0800 0FFF	Uitbreiding EPROM
1000 17FF	Niet in gebruik.
1800	RAM ter beschikking van de gebruiker.
	Top Stack gebruiker
1F9E	Bodem
1F9F 1FFF	RAM gebruikt door monitor
2000 2FFF	Uitbreiding voor RAM of EPROM
3000 FFFF	Niet in gebruik

In de 2 kbyte van 0000 tot 07FF staat het programma *Monitor.*
Dit is het programma dat het toestel klaar maakt voor gebruik en dan de tekst 'uPF——1' op de display brengt. De monitor tast ook het toetsenbord af, stuurt de display, maakt het mogelijk de inhoud van de registers en de geheugenlokaties te lezen en te wijzigen, en nog veel meer.
Vanaf adres 1800H tot 1F9E staat iets minder dan 2 kbyte RAM ter beschikking van de gebruiker. Normaal schrijft de gebruiker hier zijn programma's in het lagere adresdeel. Het gebied met de hogere adressen wordt gebruikt als stack.
Het RAM-gedeelte van 1F9F tot 1FFF wordt gebruikt door de monitor. Voorlopig kan de gebruiker het beste uit dat gebied blijven.

5.B. Toets RESET

Na het inschakelen van de netspanning wordt het toestel bestuurd door de monitor, die als teken dat hij klaar voor gebruik is de tekst 'uPF——1' op de display brengt.
Vanuit een andere toestand kan altijd weer naar deze uitgangspositie teruggekeerd worden met de toets RESET.

5.C. Geheugeninhoud lezen

Om de inhoud van een geheugenlokatie te kunnen lezen, moet men eerst het adres opgeven.
Druk eerst op de toets ADDR. Op de display verschijnen toevallige karakters. De vier linkse stellen een adres voor, en de twee rechtse de inhoud (data) van het vermelde adres.
In het adresveld zijn ook de punten toegelicht, wat erop wijst dat het adres gewijzigd kan worden, door invoer via de (witte) hexatoetsen.
Zodra het gewenste adres in het adresveld staat, verschijnt de inhoud van dit adres in het dataveld.
Om de inhoud van het volgende adres te lezen, is het voldoende, op de toets + te drukken. Het adres wordt met één verhoogd en in het dataveld wordt de inhoud van het verhoogde adres getoond.
Het vorige adres wordt bereikt met de toets −.
Na de bediening van de toets + of de toets −, zijn de punten in het adresveld verdwenen, wat erop wijst dat geen nieuw adres ingevoerd kan worden.
Eventueel opnieuw beginnen met de toets ADDR.

5.D. RAM laden

Adres op de display brengen zoals in 5.C. Daarna op de toets DATA drukken. De punten in het dataveld lichten op, wat erop wijst dat data ingevoerd kunnen worden via de hexatoetsen.
De volgende lokatie is weer bereikbaar met de toets +, waarna onmiddellijk de gewenste data ingevoerd kunnen worden.
Indien gepoogd wordt data te schrijven op een adres waar geen RAM staat, verandert het dataveld niet. Ook dooft dan de display zolang een hexatoets ingedrukt is.

Taak. Voer op de volgende adressen de bijvermelde data in. Controleer nadien nog eens de inhoud.

Adres	Data
1800H	3E
1801	12
1802	06
1803	13
1804	80
1805	32
1806	0F
1807	18
1808	76

5.E. Programma uitvoeren

De reeks hexa-waarden die zijn opgegeven in de vorige taak, vormen een eenvoudig programma, dat de waarden 12 en 13 optelt, en de som in lokatie 180F plaatst.
Een op deze manier geschreven programma is onoverzichtelijk en niet gemakkelijk te begrijpen ; daarom kan een programma beter als volgt geschreven worden.

Adres	Mach. taal	Mnemonic	Verklaring.
1800	3E 12	LD A,12	Laadt A met de waarde 12.
1802	06 13	LD B,13	Laadt B met de waarde 13.
1804	80	ADD A,B	Telt bij A de inhoud van B op
1805	32 0F18	LD (180F),A	Plaatst inhoud A in loc.180F.
1808	76	HALT	Gaat over naar wachttoestand.

Nadat het programma (de hexa waarden in de kolom Mach. taal) in de RAM geladen is, en nog eens gecontroleerd, kan het uitgevoerd worden.
Breng daarvoor eerst het startadres van het programma in het adresveld met de toets ADDR en de hexatoetsen. Druk daarna op de toets GO.
Het programma wordt uitgevoerd, met als zichtbaar gevolg het oplichten van de led HALT, en het doven van de display.
De laatste waarde in het programma is 76, de opcode voor HALT.
Deze doet de HALT led oplichten, en stopt de Z-80. De display wordt niet meer door de Z-80 gestuurd, en dooft.
Met de toets RESET kan de Micro-Professor naar de starttoestand teruggebracht worden. Van daaruit kan de RAM-lokatie 180F gecontroleerd worden, waar de som moet staan.

5.F. Toets MONI

Het drukken op de toets RESET brengt de Micro-Professor terug in de monitor mode, en geeft daarbij de registers hun oorspronkelijke inhoud terug.
De toets MONI onderbreekt ook het programma en doet het terugkeren in de monitor mode, maar zonder opnieuw opstarten, zodat de registers niet met hun oorspronkelijke waarde herladen worden. Ze bevatten dan de waarde van het ogenblik van de onderbreking.
Als het programma nog eens uitgevoerd wordt, kan nu ook met de toets MONI naar de monitor mode teruggekeerd worden. De display toont nu in zijn adresveld 1809, wat de inhoud is van de program counter. Deze waarde is heel logisch, aangezien de instructie HALT op adres 1808 reeds uitgevoerd is.

5.G. Toets REG (REGister)

Deze maakt het mogelijk de inhoud van de registers te lezen, en indien gewenst te wijzigen.
Druk vanuit de monitor mode eerst op de toets REG, waardoor in het adresveld de karakters 'rEg—' verschijnen. Als nu op toets 0 wordt gedrukt, verschijnt in het dataveld 'AF'. De linkerhelft van het adresveld toont de inhoud van A, en de rechterhelft de inhoud van F.
Met de toetsen 1 tot 9 kan op dezelfde manier de inhoud van de registers BC, DE, HL, AF', BC', DE', IX en IY opgevraagd worden. Bij de prime registers is het rechterpunt in het dataveld opgelicht.

Om de inhoud te wijzigen van een register dat op de display staat, moet eerst op de toets DATA gedrukt worden. De punten van de displays 3 en 4 lichten op, als teken dat deze waarden met de hexa-toetsen gewijzigd kunnen worden. Met de toets + worden de punten naar de displays 1 en 2 verschoven, waardoor de inhoud van het andere register gewijzigd kan worden. Als nogmaals de toets + bediend wordt, komen volgende registers aan de beurt. Met de toets − kunnen vorige registers bereikt worden.

5.H. Toets PC (Program Counter)

Toont de stand van de PC in het adresveld. Door het drukken op de toets RESET wordt ook de PC met 1800 geladen, het laagste adres in de RAM. Als daarna op toets PC wordt gedrukt, zal de display dan ook 1800 tonen.
Voor het starten van een programma dat begint op adres 1800, volstaat het na RESET op de toetsen PC en GO te drukken.

Taak. Start het programma uit 5.E. op de zojuist besproken manier. Kom terug naar de monitor mode door de toets MONI. Wijzig het getoonde adres met de toets ADDR. Druk daarna op de toets PC.

5.I. Toets STEP (single STEP)

Een iets groter programma is zelden direct juist. Met de toets STEP kan een programma-instructie per instructie uitgevoerd worden. Bij iedere stap kunnen de registers en de RAM-lokatie gecontroleerd worden, waardoor de reden van het foutief verloop van het programma ontdekt kan worden. De instructie op het adres die de PC aanduidt, is nog niet uitgevoerd. De toets STEP kan alleen gebruikt worden als de display een adres toont.
Laad het programma uit 5.E. in de RAM, en laad de registers A en B met 00. Druk daarna op RESET en PC zodat de PC het begin van het programma aanwijst. Telkens wanneer nu op STEP wordt gedrukt, wordt één instructie van het programma uitgevoerd. Na iedere stap kunnen zowel de RAM als de registers gecontroleerd en eventueel gewijzigd worden.

5.J. Een breakpoint plaatsen

Als er moeilijkheden worden ondervonden midden in een groot programma, zou het erg tijdrovend zijn om in de single step mode bij het probleempunt te komen. Daarom is ook in de mogelijkheid voorzien, een breakpoint te plaatsen. Het programma kan dan normaal gestart worden met de toets GO, maar wordt onderbroken als de instructie is uitgevoerd waar het breakpoint is geplaatst. Vanaf deze plaats kan dan in de single step mode verdergewerkt worden.
Een breakpoint wordt geplaatst door het gewenste adres op de display te brengen, en op de toets SBR (Set BReakpoint) te drukken. Alle punten op de display lichten dan op als teken dat het breakpoint is geplaatst.
Een breakpoint kan verwijderd worden met de toets CBR (Clear BReakpoint).
Alleen in de RAM kan een breakpoint geplaatst worden.
Een breakpoint moet altijd geplaatst worden op het adres waar de eerste byte van een instructie staat. In het programma uit 5.E. kunnen bijgevolg alleen breakpoints geplaatst worden op de adressen 1800, 1802, 1804, 1805 en 1808.

Slechts één kan er tegelijkertijd in een programma staan. Het plaatsen van een tweede breakpoint wist meteen de eerste uit.

Taak. Plaats in het programma uit 5.E. een breakpoint zodat de uitvoering onderbroken wordt voordat de optelling wordt uitgevoerd.

5.K. Toets MOVE

Deze maakt het mogelijk, een copy van een programma op een andere plaats in de RAM te schrijven.
Na het drukken op de toets MOVE verschijnt op de rechter-display de letter S (Start), waarmee de Micro-Professor om het startadres vraagt. Nadat dit ingevoerd is, moet op de toets + gedrukt worden, waardoor op de rechter-display de letter E (End) verschijnt. Nu moet het laatste adres van het programma ingetoetst worden, en weer de toets + bediend worden. Op de rechter-display verschijnt nu de letter D (Destination), waarop het beginadres van de bestemmingsplaats ingetoetst moet worden. Druk vervolgens op de toets GO, waardoor de copy wordt gemaakt.
Om het programma uit 5.E. te kopiëren in de RAM vanaf adres 1900 moeten de volgende toetsen bediend worden :

MOVE 1800 + 1808 + 1900 GO

Vergewist u zich er wel van dat op de bestemmingsplaats geen belangrijke data staan die door de copy overschreven zullen worden, of dat de copy niet in de stack wordt geplaatst, waardoor de Micro-Professor onbestuurbaar wordt.

5.L. Toets DEL (DELete)

Bij het ontwikkelen van grote programma's wordt soms vastgesteld dat bepaalde instructies in het programma overbodig, of zelfs hinderlijk zijn. Om ze te verwijderen moet het programma vanaf de overbodige instructies herschreven worden.
Bij de Micro-Professor is dit niet nodig als van de toets DEL gebruik gemaakt wordt.
Als op de toets DEL wordt gedrukt terwijl de display een adres toont, worden al de data van de volgende adressen tot 1DFF één adres lager geplaatst. Daarmee zijn dan ook de data op het getoonde adres verwijderd. Op adres 1DFF wordt 00 geschreven.
Stel dat in de RAM het programma uit 5.E. staat, en dat daaruit de instructie ADD (opcode 80) op adres 1804 verwijderd moet worden. Breng daarvoor het adres 1804 op de display met de toets ADDR, en druk daarna op de toets DEL. Het programma moet na deze bewerking er als volgt uitzien :

Adres	Mach. taal	Mnemonic	Verklaring.
1800	3E 12	LD A,12	Laad A met 12
1802	06 13	LD B,13	Laad B met 13
1804	32 0F18	LD (180F),A	Plaats in 180F de inhoud van A
1807	76	HALT	

Uiteraard kunnen er alleen uit de RAM instructies verwijderd worden, en dan nog alleen van adres 1800 tot 1DFF.

5.M. Toets INS (INSert)

Het inlassen van nieuwe instructies in een programma is vaker nodig dan het verwijderen. Met de toets INS kunnen instructies ingelast worden, zonder dat de rest van het programma herschreven moet worden.
Breng daarvoor het hoogste adres dat onveranderd moet blijven op de display, druk daarna op de toets INS. Het adres wordt verhoogd met één, en nieuwe data kunnen ingetoetst worden. Alle volgende data worden één plaats hoger geschoven.
Om de ingekorte versie van programma 5.K., nl. programma 5.L., weer in zijn oorspronkelijke vorm te krijgen, moeten de volgende toetsen bediend worden :

RESET ADDR 1803 INS 80

5.N. Overige toetsen

Zijn momenteel nog van geen belang, en worden te zijner tijd verklaard.

6. ADRESSEERWIJZEN

6.A. Inleiding

In het programma uit 5.E. staat de te verwerken waarde (operand) direct in het programma na de opcode. Vaak staat de te verwerken waarde ergens in een lokatie van het geheugen. In zo'n geval moet in de plaats van de te verwerken waarde het adres opgegeven worden waar de te verwerken waarde staat.
Meestal is het ook gewenst, dat het resultaat van een bewerking op een bepaalde plaats in het geheugen gedeponeerd wordt.
In beide gevallen moet het adres van de lokatie opgegeven worden bij de opcode die aanduidt wat moet gebeuren.

Het adres kan op verschillende manieren opgegeven worden, daarom spreekt men ook van verschillende adresseerwijzen.
Ontwerp een programma altijd in onderstaande volgorde :

1. Schrijf van een programma(deel) eerst alle mnemonics, en voeg er de eventuele verklaringen aan toe.
2. Vertaal de mnemonics in machinetaal met behulp van de instructie set tabel. Vergeet niet de te verwerken waarden die in het geheugen moeten staan, te noteren.
3. Plaats het beginadres van iedere instructie in de eerste kolom.

Iedere mnemonic bestaat uit een opcode-deel dat de soort bewerking aangeeft. Bij veel mnemonics moet dit aangevuld worden met een operand-deel. Dit geeft de bron of de bestemming aan, of beide. In dit laatste geval komt de bestemming eerst, gevolgd door een komma en de bron. Indien de bron of bestemming een adres is, moet dit tussen haakjes staan.

Voorbeelden

LD A,10	Laad A met 10
LD B,C	Laad B met C
LD HL,1234	Laad HL met 1234
LD A,(1234)	Laad A met inhoud lokatie 1234
LD (2345),A	Laad lokatie 2345 met A

6.B. Extended addressing

Bij extended addressing is altijd een geheugenlokatie betrokken. Deze lokatie is de bron waaruit de waarde opgehaald moet worden, of de bestemming waar een waarde heen moet. Het adres van deze lokatie moet volledig (extended) opgegeven worden in de instructie, waar vooraan de opcode (1 of 2 bytes) staat, gevolgd door het adres waarvan de 2 bytes in omgekeerde volgorde moeten staan.

Voorbeelden

Mnemonic	Mach. taal	Verklaring
LD A,(1234)	3A 3412	Laad A met inhoud locatie 1234
LD (2345),A	32 4523	Laad locatie 2345 met A

Met deze voorbeelden kan reeds een eenvoudig programma geschreven worden, dat een waarde uit een lokatie overbrengt naar een andere lokatie.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3A 1810	PGB1	LD A,(1810)	Laad A met inh.loc.1810
1803	32 1118		LD (1811),A	Laad loc.1811 met A.
1806	76		HALT	
1810	47	BRON		
1811	00	BEST		

Nadat het programma uitgevoerd is, moet in lokatie 1811 eveneens de waarde 47 staan.
De opcode wordt gevonden in de instructie set tabel, in de kolom of lijn 'Ext.Addr.' naargelang het adres bron of bestemming is.
De instructieset is verdeeld in groepen om het opzoeken te vergemakkelijken. Hier moet in de 8 bit load group gezocht worden.
In de bijkomende kolom 'Label' mag een voor de lijn passende korte naam genoteerd worden, om gemakkelijker naar die lijn te kunnen verwijzen. Vaak wordt op de eerste lijn van deze kolom de naam van het programma vermeld.

Taak. Schrijf een programma vanaf adres 1810 dat de waarde 00 die in lokatie 1800 staat, overschrijft in de lokaties 1801, 1802, 1803 en 1804.

6.C. Immediate addressing

Bij deze adresseerwijze wordt geen waarde uit het geheugen gehaald, maar uit het programma zelf. De waarde die moet worden verwerkt of geladen, staat in de instructie achter de opcode.
Eenvoudige voorbeelden van immediate addressing zijn de LD r,n-instructies.
Deze laden in register r het 8 bit integer (geheel getal) n.
Register r is een der registers A, B, C, D, E, H of L. De waarde n is een hexa-waarde van 00 tot FF.
Naargelang welk register geladen moet worden, is de opcode uiteraard verschillend. Na de opcode volgt de operand, die de waarde is die in het gekozen register moet worden geladen. Aangezien de operand hier geen adres is, mag hij in de mnemonic niet tussen haakjes staan.

Vul onderstaande tabel aan met behulp van de instructie set tabel.

Opcode	Operand	Mnemonic
3E	12	LD A,12
06	34	LD B,34
..	56	LD C,56
..	..	LD D,78
..	..	LD E,90
..	..	LD H,AB
..	..	LD L,CD

Vul het volgende programma aan. Voer het uit en controleer de registers.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	.. ..	P6C1	LD A,11	.....
180.	.. ..		LD B,22	.....
180.	.. ..		LD C,33	.....
180.	.. ..		LD D,44	.....
180.	.. ..		LD L,55	.....
180.	..		HALT	.....

Taak. Schrijf een programma dat de registers A, B, C, D, E, H en L met de waarde 00 laadt.

6.D. Implied addressing

Bij deze adresseerwijze bestaat de instructie alleen uit een opcode. De opcode zegt niet alleen welke bewerking uitgevoerd moet worden, maar ook met welke registers.
Eenvoudige voorbeelden van implied addressing zijn de LD r,s-instructies. Deze laden het register r met de inhoud van register s (source).
Voor r en s komen de volgende registers in aanmerking : A, B, C, D, E, H en L.
Laat het volgende programma uitvoeren in de single step mode. Controleer na iedere stap de inhoud van de betrokken registers.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 11	P6D1	LD A,11	.....
1802	06 22		LD B,22	.....
1804	0E 33		LD C,33	.....
1806	57		LD D,A	.....
1807	58		LD E,B	.....
1808	61		LD H,C	.....
1809	6C		LD L,H	.....
180A	76		HALT	.....

Taak. Schrijf een zo kort mogelijk programma dat de registers A tot L met 00 laadt.

Bij de Z-80 zijn er enkele opcodes die uit 2 bytes bestaan.
Als de eerste byte van de opcode de uit te voeren bewerking aangeeft, en de tweede byte alleen verschilt naargelang het register waarop de bewerking moet worden uitgevoerd, wordt soms ook van *register addressing* gesproken.
Worden echter de twee bytes samen als één opcode beschouwd, dan kunnen deze instructies ook als implied addressing gerangschikt worden. (Voorbeelden in de Rotate and Shift group.)

6.E. Extended immediate addressing

De operand bij immediate addressing is een 8 bit integer. Bij extended immediate addressing is de operand een 16 bit integer.
Deze adresseerwijze wordt o.a. toegepast om een 16 bit integer in de 16 bit registers IX, IY of SP te laden.

De opcode kan weer 1 of 2 bytes groot zijn. De operand bestaat altijd uit 2 bytes, die in omgekeerde volgorde in de instructie moeten worden opgenomen.

Gebruiksvoorbeeld

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	DD21 3412	P6E1	LD IX,1234	Laad IX met 1234
1804	FD21 7856		LD IY,5678	Laad IY met 5678
1808	76		HALT	

Twee 8 bit registers kunnen als één registerpaar door één enkele instructie geladen worden.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1810	01 2301	P6E2	LD BC,0123	Laad registerpaar BC
1813	11 6745		LD DE,4567	Laad registerpaar DE
1816	21 AB89		LD HL,89AB	Laad registerpaar HL
1809	76		HALT	

6.F. Register indirect addressing

Hier staat het adres waarvandaan de waarde moet komen of waarheen zij moet gaan, niet in de instructie ; het moet vooraf in het registerpaar HL geladen worden. De instructie bestaat dan ook enkel uit een opcode.
Hetzelfde resultaat als bij P6B1 kan nu ook als volgt verkregen worden.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	21 1018	P6F1	LD HL,1810	.....
1803	7E		LD A,(HL)	Met inh.aangew.door HL
1804	23		INC HL	Verhoog HL met 1
1805	77		LD (HL),A	Laad loc.aangew.door HL
1806	76		HALT	
1810	47	BRON		
1811	00	BEST		

De nog onbekende instructie INC HL verhoogt de inhoud van het registerpaar HL met 1 tot 1811.
Door de instructie LD (HL),n wordt de aangewezen lokatie met de integer n geladen.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	21 1018	P6F2	LD HL,1810	.....
1803	36 00		LD (HL),00	Laad 00 in locatie (HL)
1805	23		INC HL	.....
1806	36 01		LD (HL),01	Laad 01 in locatie (HL)
1808	23		INC HL	.....
1809	36 02		LD (HL),02	.....

180B	76		HALT	.....
1810	..			Bestemming.
1811	..			.....
1812	..			.....

Een lokatie kan ook aangewezen worden door registerpaar BC of DE, maar alleen om A te laden of op te bergen. Dit gaat met de instructies LD A,(BC), LD A,(DE), LD (BC),A en LD (DE),A.

6.G. Indexed addressing

Heeft grote overeenkomst met reg.ind.addr., maar hier moet het adres in register IX of IY staan. De opcode bestaat hier altijd uit twee bytes en een derde byte wordt het *displacement* genoemd. Als voor het displacement de waarde 00 genomen wordt, komt deze adresseerwijze helemaal overeen met reg.ind.addr..
Hetzelfde resultaat als in P6F1 wordt verkregen met P6G1, die ook dezelfde opbouw heeft.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	DD21 1018	P6G1	LD IX,1810	....
1804	DD7E 00		LD A,(IX)	Met inh.aangew.door IX
1807	DD23		INC IX	Verhoog inhoud IX met 1
1809	DD77 00		LD (IX),A	Laad loc.aangew.door IX
180C	76		HALT	
1810	47	BRON		
1811	00	BEST		

De nieuwe instructie INC IX komt overeen met INC HL, en verhoogt hier IX tot 1811.
Daar er twee indexregisters ter beschikking zijn, kan hetzelfde resultaat ook als volgt worden verkregen :

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	DD21 1018	P6G2	LD IX,1810	.....
1804	FD21 1118		LD IY,1811	.....
1808	DD7E 00		LD A,(IX)	.....
180B	FD77 00		LD (IY),A	.....
180E	76		HALT	
1810	47	BRON		
1811	00	BEST		

Voer beide programma's uit, en controleer telkens het resultaat en de inhoud van de gebruikte registers.

Als gebruik gemaakt wordt van het displacement, wijst de som van het indexregister en het displacement de lokatie aan. Het displacement moet een two's-complement-waarde zijn, waardoor iedere lokatie in het gebied vanaf IX-128dec.(d=80) tot IX+127dec.(d=7F) bereikt kan worden.
Als gebruiksvoorbeeld geven we het volgende programma, dat de lokaties 1800, 1880 en 18FF met 00 laadt, zonder het indexregister te wijzigen. Let erop dat het programma begint op adres 1900.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1900	DD21 8018	P6G3	LD IX,1880	.....
1904	3E 00		LD A,00	.....
1906	DD77 00		LD (IX),A	.....
1909	DD77 7F		LD (IX+d),A	Adres = 1880+7F=18FF
190C	DD77 80		LD (IX+d),A	Adres = 1880+80=1800
190F	76		HALT	
1800	47	BEST1		
1880	47	BEST2		
18FF	47	BEST3		

Taak. Schrijf een programma dat de lokaties 1900, 1910, 1920 en 1930 op 00 stelt door gebruik te maken van het indexregister IY.

Overeenkomstig de instructie LD (HL),n bestaan ook de instructies LD (IX+d),n en LD (IY+d),n die een integer in de aangewezen lokatie laden.

6.H. Overige adresseerwijzen

Relative addressing, bit addressing en modified page zero addressing zijn meer gespecialiseerde adresseerwijzen, en minder van algemeen belang. Ze worden te zijner tijd verder besproken.

6.I. Extended addressing (16 bits)

De inhoud van een registerpaar kan ook met één instructie in twee RAM-lokaties opgeborgen worden. Na de opcode moet het volledig adres van de laagste lokatie opgegeven worden.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	21 3412	P6I1	LD HL,1234	....
1803	22 1018		LD (1810),HL	Laad HL in 1810-1811 11
1806	76		HALT	....
1810	..			.....
1811	..			.....

Na uitvoering staat in 1810 de LSB (Least Significant Byte) 34 , en in 1811 de MSB (Most Significant Byte) 12 .

Op dezelfde manier kan ook de inhoud van de registerparen BC , DE , SP en van de 16 bit registers IX en IY in de RAM opgeborgen worden.

Vul de volgende tabel aan :

Mach. taal	Mnemonic	Verklaring
22 1018	LD (1810),HL	L naar 1810 H naar 1811
ED43	LD (1820),BC	.....
....	LD (1921),DE	.....
....	LD (1899),SP	.....
....	LD (182F),IX	.....
....	LD (18FF),IY	.....

Omgekeerd kan ook de inhoud van twee opeenvolgende lokaties in een registerpaar of een 16 bit register geladen worden. Ook hier moet alleen het adres van de laagste lokatie opgegeven worden.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	21 3412	P612	LD HL,1234	.....
1803	22 1018		LD (1810),HL	.....
1806	ED4B 1018		LD BC,(1810)	(1810)→C , (1811)→B
180A	76		HALT	.....
1810	..			.....
1811	..			.....

Het programma P612 brengt eerst de inhoud van HL naar 1810 en 1811, en laadt dan registerpaar B met de inhoud van dezelfde lokaties.
Na uitvoering is PC met dezelfde waarde geladen als HL, wat erop wijst dat de 16 bit waarde opgehaald wordt in dezelfde volgorde als bij het opbergen.
Ook de andere registerparen kunnen worden geladen met de inhoud van twee opeenvolgende lokaties.

Vul aan :

Mach. taal	Mnemonic	Verklaring
ED4B 1018	LD BC,(1810)	(1810) naar C , (1811) naar B
....	LD DE,(1820)	.....
..	LD HL,(0000)	.....
..	LD SP,(FFFF)	.....
DD22 9010	.....	.....
FD22 0018	.....	.....

Programma P612 geeft tegelijkertijd ook een methode aan om de inhoud van een registerpaar over te brengen naar een ander registerpaar.
Voor het overbrengen van de inhoud van HL, IX of IY moet van deze methode geen gebruik gemaakt worden, omdat daarvoor de instructies LD SP,HL, LD SP,IX en LD SP,IY ter beschikking staan.
Het gebruik van deze instructies is op het ogenblik nog niet aan te raden, daar de monitor de SP benut. Het wijzigen van de SP kan de monitor in de war brengen.

7. 8 BIT ARITHMETIC GROUP

7.A. ADD A,n

Telt bij A de 8 bit integer n op. Na deze bewerking staat de som in register A.
Immediate addressing. Opcode C6.
Naargelang het resultaat van de bewerking wordt het Flag register F als volgt beïnvloed :

S	wordt hoog als het resultaat negatief is, anders laag
Z	wordt hoog als het resultaat 00 is, anders laag
H	wordt hoog als er een carry is van bit 3, anders laag
P/V	wordt hoog bij oVerflow, anders laag
N	wordt laag
C	wordt hoog als er een carry is van bit 7, anders laag.

bit	7	6	5	4	3	2	1	0
Flag register	S	Z	.	H	.	P/V	N	C
waarde	80	40	20	10	8	4	2	1

De bits 3 en 5 in het Flag register worden niet gebruikt.
Overflow treedt op als in two's complement gewerkt wordt, en de som van twee positieve getallen groter is dan 7F. Ook als de som van twee negatieve getallen kleiner is dan 80 (– 128 dec.).

De volgende programma's zijn er vooral om de invloed van de instructie ADD A,r op de flags te bestuderen.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 12	P7A1	LD A,12	.....
1802	C6 13		ADD A,13	Tel bij A 13 op.
1804	76		HALT	.....

Na uitvoering kan de inhoud van A en F gecontroleerd worden door de volgende toetsen te bedienen : MONI, REG en AF. In register A wordt dan de som 25 gevonden en in F de waarde 20. Van register F is hier alleen bit 5, die niet als flag geldt, hoog. Al de flags zijn laag na het uitvoeren van P7A1.
Niet altijd is de vertaling van de hexa-waarde uit F naar een binair equivalent, dat aangeeft welke flags hoog zijn, zo eenvoudig. Hexa-waarden waarin meerdere bits hoog zijn laten zich gemakkelijk vertalen, en verhogen de kans op fouten.
Om dit te vermijden is de Micro-Professor uitgerust met de toetsen SZ.H en .PNC . Ze brengen elk een helft van het Flag register in binaire vorm op de display. Toets SZ.H toont de toestand van de flags SZ.H of de Hoge helft van F. Met toets .PNC wordt de toestand van de flags .PNC of de Lage helft van F getoond.
De toetsen SZ.H' en .PNC' geven de binaire toestand van het priemregister F' .
In P7A2 wordt bij de 90 die in A staat, 10 opgeteld. De beide getallen zijn positieve waarden, er wordt niet in two's complement gewerkt.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 90	P7A2	LD A,90	.....
1802	C6 10		ADD A,10	.....
1804	76		HALT	.....

Na uitvoering moet in A de waarde A0 staan. De S flag (Sign neg) is hier wel hoog, maar onbelangrijk omdat niet in two's complement wordt gewerkt.
P7A3 laadt eerst 90 in A, en telt er dan 80 bij op. Beide getallen zijn unsigned numbers.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 90	P7A3	LD A,90	.....
1802	C6 80		ADD A,80	.....
1804	76		HALT	.....

De som is 110, waardoor in A 10 komt en de flag C hoog wordt.
Ook P/V is hoog (oVerflow), maar weer onbelangrijk omdat niet in two's complement gewerkt wordt.
Tel bij de waarde 7F die in A komt, nog 02 op.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 7F	P7A4	LD A,7F	.....
1802	C6 02		ADD A,02	.....
1802	76		HALT	.....

Na uitvoering staat in A de som 81. De S flag (Sign neg) is hoog, wat betekent dat het resultaat een negatieve waarde is als in two's complement wordt gewerkt.
Ook flag V is hoog, die daarmee op oVerflow wijst als in two's complement wordt gewerkt. Alleen als 7F en 02 als unsigned numbers zijn bedoeld moet met de stand van beide flags geen rekening gehouden worden.
De flag H, die hier ook hoog is, is nu nog onbelangrijk en wordt later behandeld.
Het volgende programma is een uitbreiding van P7A4, en brengt de som naar lokatie 180F.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 7F	P7A5	LD A,7F	.....
1802	C6 02		ADD A,02	.....
1804	32 0F18		LD (180F),A	.....
1807	76		HALT	.....
180F	..	BEST		.....

Na uitvoering zijn de flags in dezelfde toestand als na P7A4, hoewel na de optelling nog een instructie werd uitgevoerd. Dit wijst erop dat de LD-instructie de flags niet heeft beïnvloed, wat ook vastgesteld kan worden in de instructie set tabel.

7.B. ADD A,r

Telt bij A de inhoud van register r op.
De flags worden beïnvloed als bij 7.A. Implied addressing.
Vul de volgende tabel aan, met behulp van de instructie set tabel.

Mach. taal	Mnemonic	Verklaring
87	ADD A,A	Telt bij A de inhoud van A op.
80	ADD A,B	Telt bij A de inhoud van B op.
..	ADD A,C	.....
..	ADD A,D	.....
..	ADD A,E	.....
84	.....	.....
85	.....	.....

De instructie ADD A,A vermenigvuldigt de inhoud van A met 2.
Vul het volgende programma aan.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	.. ..	P7B1	LD A,12	.....
180.	.. ..		LD B,55	.....
180.	..		ADD A,B	.....
180.	76		....	.....

Vul de volgende tabel in. Voer daarna P7B1 uit met de waarden uit de tabel, om zo de juistheid van uw voorspellingen te kunnen nagaan.

A	B	SOM	F	S	Z	.	H	.	P	N	C	Verklaring
29	39	..	..	.	.	.	.	.	.	.	.	.....
20	90	..	..	.	.	.	.	.	.	.	.	.....
10	F0	..	..	.	.	.	.	.	.	.	.	.....
40	40	..	..	.	.	.	.	.	.	.	.	.....
82	F0	..	..	.	.	.	.	.	.	.	.	.....
F0	F0	..	..	.	.	.	.	.	.	.	.	.....
02	FE	..	..	.	.	.	.	.	.	.	.	.....

Taak. Schrijf een programma dat de som berekent van de waarden die in H en L staan, en daarna het resultaat in lokatie 1830 plaatst.

7.C. SUB A,n

Trekt van A de 8 bit integer n af.
Immediate addressing. Opcode D6.
Flags als in 7.A. met uitzondering van :

C wordt hoog als er een borrow (lening) nodig is, anders laag
H wordt hoog als er geen borrow is van bit 4, anders laag
N wordt hoog na iedere aftrekking, dus ook hier.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 47	P7C1	LD A,47	.....
1802	D6 12		SUB 12	Trek 12 af van A.
1804	76		HALT	.....

Na uitvoering is enkel flag N hoog, als teken dat de laatste bewerking een aftrekking was. Als n groter is dan de inhoud van A, zal flag C (hier eigenlijk borrow) hoog zijn.
In P7C2 is de instructie HALT vervangen door de instructie RST 38.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 47	P7C2	LD A,47	.....
1802	D6 48		SUB A,48	.....
1804	FF		RST 38	Naar monitor.

De Micro-Professor komt na het uitvoeren van P7C2 niet meer in de HALT-toestand, maar geeft de controle terug aan de monitor. Het bedienen van de toets MONI is daardoor overbodig geworden.
Vul eerst de volgende tabel in, en vergelijk daarna met de praktijk :

A	B	Versch.	F	S	Z	.	H	.	P	N	C	Verklaring
47	00	..	..	.	.	.	.	.	.	.	.	.....
47	47	..	..	.	.	.	.	.	.	.	.	.....
47	48	..	..	.	.	.	.	.	.	.	.	.....
FF	0	..	..	.	.	.	.	.	.	.	.	.....
82	03	..	..	.	.	.	.	.	.	.	.	.....
F0	FF	..	..	.	.	.	.	.	.	.	.	.....
FF	FE	..	..	.	.	.	.	.	.	.	.	.....

7.D. ADD A,(HL)

Telt bij A de inhoud van de lokatie op die door HL aangewezen wordt.
Flags als bij 7.A. Register indirect addressing.
Opcode 86.
Als gebruiksvoorbeeld schrijft P7D1 de som van de waarden M en N die in 1820 en 1821 staan, in lokatie 1822.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	21 2018	P7D1	LD HL,1820	.....
1803	7E		LD A,(HL)	.....
1804	23		INC HL	.....
1805	86		ADD A,(HL)	Tel bij A de inh.v.1821
1806	23		INC HL	.....
1807	77		LD (HL),A	.....
1808	FF		RST 38	.....
1820	12	M		.....
1821	13	N		.....
1822	00	M+N		.....

7.E. ADD A,(IX + d), ADD A,(IY + d)

Telt bij A de inhoud van de lokatie die door IX + d of IY + d aangewezen wordt.
Flags als bij 7.A. Indexed addressing.

Mach. taal	Mnemonic	Verklaring
DD86 00	ADD A,(IX)	.....
DD86 10	ADD A,(IX+d)	.....
FD86 00	.....	.....
FD86 FF	.....	.....

Hetzelfde resultaat als met P7E1 kan nu ook als volgt verkregen worden :

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	DD21 2018	P7E1	LD IX,1820	.....
1804	DD7E 00		LD A,(IX)	.....
1807	DD86 01		LD A,(IX+01)	.....
180A	DD77 02		LD (IX+02),A	.....
180D	FF		RST 38	.....
1820	12	M		.....
1821	47	N		.....
1822	00	N+M		.....

7.F. SUB A,s

Trek van A de inhoud van s af. Deze kan een register zijn, of een lokatie aangewezen door HL, IX of IY.
Flags als 7.C.
Met de instructie SUB A,A kan register A op 00 gesteld worden.

Taak 1. Schrijf een zo kort mogelijk programma dat de registers A, B, C, D, E, H en L met 00 laadt.

Onderstaand programma schrijft de som van de waarden M en N in de lokatie 1822, en de beide verschillen in de lokaties 1823 en 1824. De nog niet besproken instructie DEC HL (DECrement), verlaagt de inhoud van registerpaar HL met één.
Vul het programma aan en voer het uit.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	21 2018	P7F1	LD HL,1820	.....
1803	..		LD A,(HL)	Laad A met M
18..	..		INC HL	Verhoog HL tot 1821
18..	..		LD B,(HL)	Laad B met N
18..	..		ADD A,B	Maak som M+N
18..	..		INC HL	Verhoog HL tot 1822
18..	..		LD (HL),A	M+N naar 1822
18..	..		DEC HL	Verlaag HL tot 1821

```
18..  ..                DEC HL          Verlaag HL tot 1820
18..  ..                LD A,(HL)       Laad A met M
18..  ..                SUB A,B         Maak verschil M-N
18..  ..                INC HL          Verhoog HL tot 1821
18..  ..                INC HL          Verhoog HL tot 1822
18..  ..                INC HL          Verhoog HL tot 1823
18..  ..                LD (HL),A       M-N naar 1823
18..  ..                DEC HL          Verlaag HL tot 1822
18..  ..                DEC HL          Verlaag HL tot 1821
18..  ..                DEC HL          Verlaag HL tot 1820
18..  ..                LD B,(HL)       Laad B met M
18..  ..                INC HL          Verhoog HL tot 1821
18..  ..                LD A,(HL)       Laad A met N
18..  ..                SUB A,B         Maak verschil N-M
18..  ..                INC HL          Verhoog HL tot 1822
18..  ..                INC HL          Verhoog HL tot 1823
18..  ..                INC HL          Verhoog HL tot 1824
18..  ..                LD (HL),A       N-M naar 1824
18..  ..                RST 38          ....................

1820  47        M                       ....................
1821  13        N                       ....................
1822  00        M+N                     ....................
1823  00        M-N                     ....................
1824  00        N-M                     ....................
```

Taak 2. Probeer van P7F1 een kortere versie te schrijven.

7.G. CP A,s

Vergelijkt de inhoud van A met de operand s. Daarbij kan s een integer zijn (immediate addr.), de inhoud van een register (implied addr.), of de inhoud van een lokatie aangewezen door HL (register indirect addr.) of door IX of IY (indexed addr.).
Flags als bij 7.A.
De vergelijking gebeurt door de bewerking A-s. Volgens het verschil worden de flags gesteld, maar de inhoud van A en s blijft onveranderd.
De instructie CP A,A vergelijkt de inhoud van A met de inhoud van A. Zij kan gebruikt worden om de flags te stellen volgens de inhoud van A. Bijvoorbeeld na een LD A,(HL) die de flags niet beïnvloedt.

7.H. INC s

Verhoogt de inhoud van een register, of de inhoud van een lokatie aangewezen door HL, IX of IY met één.
Verwar INC s niet met de instructie INC HL of INC IX, die de inhoud van het registerpaar of 16 bit register verhoogt.
Vul de volgende tabel aan :

Mach. taal	Mnemonic	Verklaring
3C	INC A	Verhoog de inhoud van A met 1.
Ø4	INC B	.....
..	INC C	.....
..	INC D	.....
..	INC E	.....
..	INC H	.....
..	INC L	.....
..	INC (HL)	Verhoog inh.locatie aangew.door HL
.... ..	INC (IX+d)	Verhoog inh.locatie aangew.door IX+d
.... ..	... .	Verhoog inh.locatie aangew.door IY+d

Vul in om alle misverstanden te vermijden :

H	L	Mach. taal	H	L
Ø1	8Ø	INC HL	..	..
Ø1	8Ø	INC L	..	..
Ø1	8Ø	INC H	..	..
Ø1	FF	INC L	..	..
Ø1	FF	INC HL	..	..
FF	FF	INC L	..	..
FF	FF	INC HL	..	..

Vergelijk de voorspellingen met de praktijk.

7.I. DEC s

Verlaagt de inhoud van een register of de inhoud van een lokatie aangewezen door HL, IX of IY met één.
Flags als bij 7.C.

7.J. ADC A,s

Telt bij A de carry en de operand s. De operand s kan een 8 bit integer zijn, de inhoud van een register, of de inhoud van een lokatie aangewezen door HL, IX of IY.
Flags als bij 7.A.
In één byte of 8 bits, kunnen hele getallen bewaard worden van 00 tot FF (255dec.). Daardoor kan een programma als P7B1 ook geen grotere waarden verwerken.
Waarden tot FFFF (65535dec) kunnen bewaard worden in twee bytes, net als adressen in een registerpaar of in een 16 bit register, en worden *double precision values* genoemd.
Het optellen van twee double-precision-waarden kan gebeuren door eerst de beide LSB's bij elkaar op te tellen en daarna de beide MSB's.

Voorbeeld

```
MSB LSB
 12 34
+56 78
======
 68 AC
```

Als beide optellingen gemaakt worden door een ADD-instructie, is de juistheid van het resultaat niet altijd gegarandeerd.

Voorbeeld

```
MSB LSB
 12 94
+56 98
======
 68 2C in plaats van 692C
```

Hier ontstaat bij het optellen van de LSB een carry (C hoog), waarmee geen rekening gehouden wordt als de som gemaakt wordt van de MSB's.
Worden eerst de LSB's opgeteld met de ADD-instructie, en daarna de som gemaakt van de MSB's door de ADC-instructie, dan wordt een eventuele carry van de LSB's bij de MSB' opgeteld.
Programma P7I1 telt de waarden 1294 en 5698 bij elkaar op, en plaatst de som in registerpaar BC.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 94	P7I1	LD A,94	.....
1802	C6 96		ADD A,98	.....
1804	4F		LD C,A	.....
1805	3E 12		LD A,12	.....
1807	CE 56		ADC A,56	Tel bij A de C en 56.
1809	47		LD B,A	.....
180A	FF		RST 38	.....

Door de immediate addressing is P7I1 eenvoudig, en gemakkelijk te volgen.
In de praktijk moeten meestal waarden opgeteld worden die ergens in de RAM staan. Ook de som moet meestal in de RAM komen.
In P7I2 wordt HL gebruikt om aan te wijzen, waar de waarden staan en waar de double-precision-som moet komen.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	21 2018	P7I2	LD HL,1820	.....
1803	..		LD A,(HL)	LSB M naar A
1804	..		INC HL	Verhoog HL tot 1821
18..	..		ADD A,(HL)	LSB M + LSB N
18..	..		INC HL	Verhoog HL tot 1822
18..	..		LD (HL),A	Som naar LSB S
18..	..		INC HL	Verhoog HL tot 1823
18..	..		LD A,(HL)	MSB N naar A
18..	..		INC HL	Verhoog HL tot 1824
18..	..		ADC A,(HL)	MSB M + MSB N + C
18..	..		INC HL	Verhoog HL tot 1825
18..	..		LD (HL),A	Som naar MSB S
18..	..		RST 38	Naar monitor
1820	94	LSB M		.....
1821	98	LSB N		.....
1822	00	LSB S		.....
1823	12	MSB M		.....
1824	56	MSB N		.....
1825	00	MSB S		.....

De operands staan hier net achter het programma in de RAM, maar ieder zestal opeenvolgende RAM-lokaties is geschikt. Wel moet de volgorde strikt in acht genomen worden.

LSB opteltal
LSB opteller
LSB som
MSB opteltal
MSB opteller
MSB som

Een groot nadeel van deze volgorde is dat de LSB en de MSB van de som niet in opeenvolgende lokaties staan. Daardoor kan de som niet rechtstreeks met één instructie (bv. LD HL,(nn)) in een registerpaar geladen worden. Hetzelfde geldt voor het opteltal en de opteller. Door het aanwijzen te laten gebeuren door IX of IY kan de opstelling verbeterd worden.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	....	P7I3	LD IX,1820	.....
18..	.... ..		LD A,(IX)	LSB M naar A
18..	.... ..		ADD A,(IX+2)	LSB M + LSB N
18..	.... ..		LD (IX+4),A	Som naar LSB S
18..	.... ..		LD A,(IX+1)	MSB M naar A
18..	.... ..		ADC A,(IX+3)	MSB M + MSB N
18..	.... ..		LD (IX+5),A	Som naar MSB S
18..	..		... ..	.....
1820	94	LSB M		.....
1821	12	MSB M		.....
1822	98	LSB N		.....
1823	56	MSB N		.....
1824	00	LSB S		.....
1825	00	LSB S		.....

Ook met double precision kan in two's complement gewerkt worden.
Het rekenbereik gaat dan van 8000 tot 7FFF (−32768 tot 32767 dec of 1000 0000 0000 0000 bin. tot 0111 1111 1111 1111 bin.).
Om te weten of het resultaat positief of negatief is, moet slechts naar de flag S (Sign neg) gekeken worden nadat de bewerking met de MSB's is uitgevoerd. De flag S na de bewerking met de LSB's is zonder betekenis. Daar is vooral de flag C van belang. Hetzelfde geldt voor de flag P/V.

Taak. Schrijf een programma dat de som maakt van de multiple-precision-waarden 12345678 en 1ABCDEF die in de RAM staan. De som moet in buurlokaties komen met de LSB op het laagste adres.

7.K. SBC A,s

Trekt van A de operand s en de carry af. Weer kan de operand s een integer zijn, of de inhoud van een register, of de inhoud van een lokatie die door HL, IX of IY aangewezen wordt. Flags als bij 7.C.

Bij het aftrekken van double-precision-waarden ontstaan dezelfde problemen als bij het optellen. Alleen als er bij het aftrekken ook rekening gehouden wordt met de carry van het lagere verschil is er zekerheid over de juistheid van het resultaat.

Taak. Pas de programma's uit 7.I. aan voor aftrekkingen.

7.L. AND A,s

Met iedere bit van A wordt een AND-bewerking uitgevoerd met de overeenkomstige bit van de operand, en het resultaat teruggeplaatst in A.
De operand s kan weer een integer zijn, of de inhoud van een register, of de inhoud van een lokatie die is aangewezen door HL, IX of IY.

Voorbeeld

```
A voor bewerking        1001 0100 = 94
Operand                 0111 1110 = 7E
                        =========
A na AND A,7E           0011 0100 = 14
```

Te controleren met navolgend programma.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 94	P7K1	LD A,94	.....
1802	E6 7E		AND A,7E	.....
1804	FF		RST 38	.....

Een toepassing van deze instructie is het laag maken van een deel van de inhoud van A (mask out). Daarmee kan één van de twee digits die in A staan op nul gesteld worden.

Voorbeeld

```
A voor bewerking        0101 1010 = 5A
B                       0000 1111 = 0F
                        =========
A na AND A,B            0000 1010 = 0A
```

In programma :

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 5A	P7K2	LD A,5A	.....
1802	06 0F		LD B,0F	.....
1804	A0		AND A,B	.....
1805	FF		RST 38	.....

Na uitvoering staat in A nog 0A. De linker-digit is op nul gesteld. Ook de rechter-digit kan alleen op nul gesteld worden met de operand F0.
De instructie AND A,A verandert de inhoud van A niet, maar stelt wel de flags.
Daar AND geen wiskundige, maar een logische bewerking is, kan er ook geen oVerflow optreden.
De flag P/V wordt bij logische bewerkingen hoog als er een even aantal bits hoog is (parity

even) in het resultaat. Bij bepaalde methoden van data-overdracht is dit een noodzakelijke informatie.
Na uitvoering van P7K2 moet de P/V flag hoog zijn omdat in het resultaat twee bits hoog zijn.

Taak 1. Pas de inhoud van A in P7K2 aan, zodat de P/V flag laag wordt.

Flags

S	wordt hoog als bit 7 van het resultaat hoog is, anders laag
Z	wordt hoog als alle bits in het resultaat laag zijn, anders laag
H	wordt hoog
P/V	wordt hoog bij pariteit van het resultaat, anders laag
N	wordt laag
C	wordt laag.

Taak 2. Schrijf een programma dat de oneven waarden die eventueel in de registers B, C, D, E, H en L staat even maakt door ze met één te verlagen.

7.M. OR A,s

Met iedere bit van A wordt een OR-bewerking uitgevoerd met de overeenkomstige bit van de operand s, en het resultaat weer geplaatst in A.
Operands en flags als in 7.K.

Voorbeeld

```
A voor bewerking        1001 0100 = 94
Operand n               0110 1110 = 6E
                        =========
A na OR A,n             1111 1110 = FE
```

Te controleren met :

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 94	P7L1	LD A,94	.....
1802	F6 6E		OR A,6E	.....
1804	FF		RST 38	.....

Met de OR A,s-instructie kunnen de gewenste bits in A naar hoog geforceerd worden. Zo zal na een OR A,0F de rechter-digit in A zeker F zijn, wat zijn oorspronkelijke waarde ook was.

Voorbeeld

```
A voor bewerking        0101 1010 = 5A
C                       0000 1111 = 0F
                        =========
A na OR A,C             0101 1111 = 5F
```

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 5A	P7L2	LD A,5A	.....
1802	0E 0F		LD C,0F	.....
1804	B1		OR A,C	.....
1805	FF		RST 38	.....

Na uitvoering staat in A 5F, de rechter-digit is tot F geforceerd.

Met de OR-instructie kan ook gecontroleerd worden, of een registerpaar of twee buurlokaties met 0000 geladen zijn.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	ED4B 1018	P7L3	LD BC,(1810)	.....
1804	78		LD A,B	.....
1805	B1		OR A,C	.....
1806	FF		RST 38	.....
1810	..	M		.....
1811	..	N		.....

Na uitvoering zal de flag Z alleen hoog zijn, als zowel in 1810 als in 1811 00 staat.
Bij uitvoering van P7L3 gaat de inhoud van A verloren. Verschillende methoden staan ter beschikking om de inhoud van A te bewaren ; maar daarover later.

Taak. Als taak 2 uit 7.L, maar de oneven waarden moeten verhoogd worden.

7.N. XOR A,s

Met iedere bit van A wordt een exclusieve OR-bewerking uitgevoerd met de overeenkomstige bit van de operand s, en het resultaat in A teruggeplaatst.
Operand en flags als in 7.K.

Voorbeeld

```
A voor bewerking        1001 0100 = 94
n                       0110 1110 = 6E
                        =========
A na XOR A,n            1111 1010 = FA
```

Aan te tonen met :

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 94	P7M1	LD A,94	.....
1802	EE 6E		XOR A,6E	.....
1804	FF		RST 38	.....

Met dè instructie XOR A,FF worden alle bits van A omgekeerd.

```
A voor bewerking        1001 0100 = 94
Operand                 1111 1111 = FF
                        ==========
A na XOR A,FF           0110 1011 = 6B
```

Ook enkele bits van A kunnen omgekeerd worden zonder de overige te wijzigen.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 94	P7M2	LD A,94	.....
1802	EE 0F		XOR A,0F	.....
1804	FF		RST 38	.....

De instructie XOR A,A stelt A op 00.

8. 16 BIT ARITHMETIC GROUP

8.A. ADD HL,ss

Telt bij HL de inhoud van registerpaar BC, DE, HL of SP op.

Flags

C wordt hoog als er een carry is uit bit 15, anders laag
N wordt laag
H onbekend
S, Z en P/V worden niet beïnvloed, daardoor kan niet vastgesteld worden of het resultaat negatief of nul is, of dat er overflow is opgetreden.

Vul aan :

Opcode	Mnemonic	Verklaring
09	ADD HL,BC	Telt bij HL de inhoud van BC.
..	ADD HL,DE	.....
..	ADD HL,HL	.....
..	ADD HL,SP	.....

Met deze instructies wordt het optellen van double-precision-waarden eenvoudiger. Hetzelfde resultaat als in P7I3 kan nu ook als volgt worden verkregen :

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	2A 2018	P8A1	LD HL,(1820)	.....
1803	ED4B 2218		LD BC,(1822)	.....
1807	09		ADD HL,BC	Double prec. som in HL.
1808	22 2418		LD (1824),HL	.....
180B	FF		RST 38	.....
1820	94	LSB M		.....
1821	12	MSB M		.....
1822	98	LSB N		.....
1823	56	MSB N		.....
1824	00	LSB S		.....
1825	00	MSB S		.....

Ook om registerpaar HL met een 8 bit integer te verhogen kan van deze instructies gebruik gemaakt worden.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	21 F01F	P8A2	LD HL,1FF0	Geef HL een inhoud,
1803	11 0040		LD DE,0040	die met 40 moet
1806	19		ADD HL,DE	verhoogd worden.
1807	FF		RST 38	.....

Na uitvoering moet in HL de waarde 2030 staan.
Met de instructie ADD HL,HL wordt de inhoud van HL met 2 vermenigvuldigd.

Taak 1. Schrijf een programma dat de inhoud van HL met 10 vermenigvuldigt.
Taak 2. Idem, maar maal 0A.

8.B. ADD IX,rr

Telt bij X de inhoud van registerpaar BC, DE, SP of IX op.
Flags en gebruik als in 8.A.

Vul aan :

Opcode	Mnemonic	Verklaring
DDØ9	ADD IX,BC	Tel bij IX de inhoud van BC op.
....	... .	.....
....	... .	.....
....	... .	.....

8.C. ADD IY,rr

Telt bij IY de inhoud van registerpaar BC, DE, SP of IY op.
Flags en gebruik als in 8.A.

Opcode	Mnemonic	Verklaring
....	... .	.....
....	... .	.....
....	... .	.....
FD29	ADD IY,IY	Vermenigvuldigt de inhoud van IY met 2.

8.D. ADC HL,ss

Telt bij HL de carry en de inhoud van registerpaar BC, DE, HL of SP op.

Er worden meer flags gesteld dan bij ADD HL,ss :

S wordt hoog als het resultaat negatief is, anders laag
Z wordt hoog als het resultaat nul is, anders laag
H onbekend
P/V wordt hoog bij overflow, anders laag
N wordt laag
C wordt hoog als er een carry is van bit 15, anders laag.

Vul aan :

Opcode	Mnemonic	Verklaring
ED4A	ADC HL,BC	Tel bij HL de inhoud van BC en de Car.op.
....	ADC HL,DE	.....
....	ADC HL,HL	.....
....	ADC HL,SP	.....

Met deze instructies kunnen nu waarden van 32 bit (multiple precision) vrij eenvoudig opgeteld worden.
Programma P8D1 telt de waarden 89ABCDEF en 12345678 bij elkaar op en plaatst de som in de RAM vanaf 1820.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	21 EFCD	P8D1	LD HL,CDEF	Lager deel opteltal.
1803	01 7856		LD BC,5678	Lager deel opteller.
1806	09		ADD HL,BC	Lager deel som in HL.
1807	22 2018		LD (1820),HL	Naar 1820 en 1821.
180A	21 AB89		LD HL,89AB	Hoger deel opteltal.
180D	01 3412		LD BC,1234	Hoger deel opteller.
1810	ED 4A		ADC HL,BC	Hoger deel som in HL.
1812	22 2218		LD (1822),HL	Naar 1822 en 1823.
1815	FF		RST 38	.....
1820	00	LSB S		.....
1821	00			.....
1822	00			.....
1823	00	MSB S		.....

Maak zelf de som, om het resultaat van P8D1 te kunnen controleren.

Taak. Schrijf een programma dat dezelfde waarden die nu in RAM staan, optelt en de som in de registerparen BC en DE plaatst.

8.E. SBC HL,ss

Trekt van de inhoud van HL de carry en de inhoud van registerpaar BC, DE, HL of SP af.
Flags als bij ADC HL,ss, met uitzondering van flag N die hoog wordt.
Kan gebruikt worden voor multiple-precision-aftrekkingen.
Wegens het ontbreken van een instructie voor 16 bit aftrekkingen zonder carry, kan SBC HL,ss ook gebruikt worden voor double-precision-aftrekkingen. Wel moet dan vooraf de carry laag gemaakt worden.

Vul aan :

Opcode	Mnemonic	Verklaring
ED42	SBC HL,BC	Trekt van HL carry en inhoud BC af.
....	... ..,..	.....
....	... ..,..	.....
....	... ..,..	.....

Met de instructie SBC HL,HL kan het registerpaar HL op 0000 gesteld worden. Wel op voorwaarde dat de carry laag is, anders wordt HL geladen met FFFF.

9. JUMP GROUP

9.A. JP nn

Laadt de operand nn in de program counter, die met deze nieuwe inhoud wijst naar de volgende instructie die moet worden uitgevoerd.
Kan ook geïnterpreteerd worden als 'Jump naar adres nn'.
Flags worden niet beïnvloed. Immediate extended addressing.
De eerste byte van de operand moet weer de lower order byte van het adres zijn.

Mach.taal	Mnemonic	Verklaring
C3 1018	JP 1810	Jump naar adres 1810 en werk daar verder.

Door de JP nn-instructie is het niet meer noodzakelijk dat een programma als één blok in opeenvolgende lokaties staat.
Het kan ook zo :

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 15	P9A1	LD A,15	.....
1802	06 03		LD B,03	.....
1804	80		ADD A,B	.....
1805	C3 1018		JP 1810	Ook duidelijk isJP AM10
1810	26 00	AM10	LD H,00	.....
1812	6F		LD L,A	.....
1813	29		ADD HL,HL	HL * 2
1814	29		ADD HL,HL	HL * 4
1815	29		ADD HL,HL	HL * 8
1816	29		ADD HL,HL	HL * 10
1817	FF		RST 38	.....

Nadat de som is gemaakt van A en B, overbrugt de JP 1810 de niet-gebruikte lokaties. In het tweede programmadeel met label AM10 wordt de inhoud van A met 10 vermenigvuldigd, en komt het produkt in HL.
Ook andere programma's, die moeten eindigen met de inhoud van A maal 10 te vermenigvuldigen, kunnen van AM10 gebruik maken door een JP 1810 instructie.

Taak 1. Voer PIA1 uit in de single step mode, en let vooral op de program counter.
Taak 2. Als P9A1, maar de waarden die bij elkaar moeten worden opgeteld staan in de lokaties 1810 en 1811.

De instructie JP nn wordt altijd uitgevoerd, er behoeft aan geen enkele voorwaarde voldaan te worden. Daarom wordt het ook een *unconditional jump* genoemd.

9.B. JP cc,nn

Alleen als aan de voorwaarde cc voldaan is, wordt de operand nn in de program counter

geladen, die daarmee de volgende instructie aanwijst die moet worden uitgevoerd.
Indien de voorwaarde cc niet aanwezig is, wordt de jump niet uitgevoerd en naar de volgende instructie overgegaan.
De voorwaarde cc is de stand van een der flags.
Met de opcode kan gekozen worden uit 8 verschillende voorwaarden.

Mach. taal	Mnemonic	Verklaring
CA 5018	JP Z,1850	Jump naar 1850 als Z=1.
C2 0019	JP NZ,1900	Jump naar 1900 als Z=0.
.. ....	JP C,0624	Jump naar 0624 als C=1.
.. ....	JP NC,05FE	Jump naar 05FE als C=0.
.. ....	JP M,1800	Jump naar 1800 als S=1 (Min).
.. ....	JP P,1850	Jump naar 1850 als S=0 (Plus).
.. ....	JP PE,0689	Jump naar 0689 als P/V=1(Parity Even)
.. ....	JP PO,18FF	Jump naar 18FF als P/V=0(parity Odd).

Als P9A1 twee getallen optelt waarvan de som groter is dan FF, wordt een foutief produkt verkregen, omdat geen rekening gehouden wordt met de carry.
Dit gebeurt wel in P9B1 die de bij elkaar op te tellen waarden uit de lokaties 1820 en 1821 haalt, en het produkt op 1822 en 1823 plaatst.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3A 2018	P9B1	LD A,(1820)	.....
1803	47		LD B,A	.....
1804	3A 2118		LD A,(1821)	.....
1807	80		ADD A,B	.....
1808	26 00		LD H,00	.....
180A	D2 0E		JP NC,180E	Jump als C=0 naar GEEN
180D	24		INC H	.....
180E	6F	GEEN	LD L,A	.....
180F	29		ADD HL,HL	.....
1810	29		ADD HL,HL	.....
1811	29		ADD HL,HL	.....
1812	29		ADD HL,HL	.....
1813	22 2218		LD (1822),HL	.....
1816	FF		RST 38	.....
1820	80	M		.....
1821	90	N		.....
1822	00	NM10L		.....
1823	00	NM10H		.....

Indien er bij ADD A,B een carry ontstaat, wordt de JP NC,GEEN niet uitgevoerd. Register H wordt dan door INC H op 01 gebracht, waardoor in HL de juiste som staat.
Ontstaat er geen carry, dan zal JP NC,GEEN over de instructie INC H heenspringen. Register H blijft op 00 staan.

9.C. JR e

Telt de integer e bij de inhoud van de program counter. De volgende instructie die uitgevoerd zal worden, is die welke door de nieuwe inhoud van program counter aangewezen wordt.

Dit is ook een jump-instructie, maar het bestemmingsadres wordt hier niet opgegeven, wel de relatieve verplaatsing (displacement) die gemaakt moet worden.
Daardoor wordt er ook van *relative addressing* gesproken.
Het programma P9A1 kan nu ook als volgt geschreven worden :

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 15	P9C1	LD A,15	.....
1802	06 03		LD B,03	.....
1804	80		ADD A,B	.....
1805	18 19		JR AM10	Jump relatief naar AM10
1807				
1820	26 00	AM10	LD H,00	Programmadeel A Maal 10
1822	6F		LD L,A	.....
1823	29		ADD HL,HL	.....
1824	29		ADD HL,HL	.....
1825	29		ADD HL,HL	.....
1826	29		ADD HL,HL	.....
1827	FF		RST 38	.....

Op adres 1805 staat de operand 18 voor Jump Relatief, en op 1806 het displacement dat moet worden uitgevoerd om aan de volgende instructie te komen.
Als de CPU de JR e gelezen heeft, staat de PC op 1807.
Daarbij wordt dan het displacement 19 geteld zodat de program counter op 1820 komt te staan, waar de instructie LD H,00 uitgevoerd zal worden.
Het displacement bij een relatieve jump naar voren wordt berekend door van het bestemmingsadres het adres van de JR e, verhoogd met 2, af te trekken (natuurlijk in hexadecimaal).

```
 1820 = bestemmingsadres
-1807 = adres JR e verhoogd met 2
=====
 0019 = verschil
   19 = displacement
```

Taak 1. Laat P9C1 uitvoeren in de single step mode, en let vooral op de program counter.

Bij het berekenen van het displacement worden vaak fouten gemaakt. De single step mode geeft de gelegenheid om de juistheid van de relatieve jump na te gaan.

Taak 2. Schrijf een programma vanaf adres 18A0, dat eerst de som maakt van 17 en 28, en dat daarna overspringt naar adres 1900, waar nog eens 17 bijgeteld moet worden.

Ook een relatieve jump achteruit is mogelijk. Daarvoor moet het displacement een negatieve waarde (two's complement) zijn.
P9C2 is een aanvulling bij P9C1. Het verschil tussen twee waarden wordt berekend, waarna de JR e doet overspringen naar label AM10 die de inhoud van A met 10 vermenigvuldigt.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1830	3E 28	P9C2	LD A,28	.....
1832	06 12		LD B,12	.....
1834	90		SUB A,B	.....
1835	18 E9		JR AM10	.....
1837				

Het berekenen van het displacement voor een relatieve jump achteruit, is in principe hetzelfde als voor een voorwaartse sprong.

```
 1820 = bestemmingsadres
-1837 = adres JR e + 2
=====
 FFE9 = verschil
   E9 = displacement
```

Daar het displacement in slechts één byte staat en two's complement wordt toegepast, is de grootste positieve waarde 7F en de kleinste (negatieve) 80.
Het relatieve-jump-bereik is daardoor $7F+2=81$ (129 dec) vooruit en $80+2=7E$ ($-128+2=126$ dec.) achteruit.
De voordelen van de relatieve jump zijn minder geheugengebruik (2 byte i.p.v. 3), maar vooral de mogelijkheid om het programma op een andere plaats in het geheugen te laten werken (relocable), omdat er geen absoluut adres wordt opgegeven.
Zo kan met de toets MOVE het programma P9A1 wel verder in de RAM geplaatst worden, maar zal daar niet werken door de JP 1820 in het programma.
Daarentegen is P9C1 wel verplaatsbaar, 'relocabel' : deze zal na een MOVE naar een andere plaats in de RAM daar even goed functioneren, door de relatieve jump.
Een nadeel van de JR e is het beperkte jump-bereik, en de kans op fouten bij het berekenen van het displacement.

9.D. Toets RELA

Bij het berekenen van het displacement worden vaak fouten gemaakt, vooral bij een achterwaartse sprong.
In de ROM van de Micro-Professor staat een programma dat de CPU deze bewerking laat maken.
Voor het berekenen van het displacement uit P9C1 met de Micro-Professor moet als volgt tewerk gegaan worden.
Druk op toets RELA (RELAtief). In het dataveld verschijnt $-S$, waarmee om het Source-adres gevraagd wordt. Ook de punten in het adresveld zijn hoog, wat erop wijst dat het adres van de JP e-instructie kan worden ingetoetst. Daarna moet toets + bediend worden, en verschijnt in het dataveld $-d$ (destination). Nu moet het bestemmingsadres ingetoetst worden, en daarna toets G ingedrukt worden.
Het relatief adres wordt nu automatisch achter de opcode in de RAM geschreven. De display toont het adres en het geplaatste displacement.
Voor het berekenen van het displacement uit P9C2 moeten de toetsen als volgt bediend worden :

RELA 1835 + 1820 GO

met als resultaat 1836 E9 op de display en het displacement E9 in lokatie 1836.

9.E. JR c,e

Alleen als aan de voorwaarde c is voldaan, wordt de integer e bij de inhoud van PC opgeteld.
Anders wordt gewoon naar de volgende instructie overgegaan.
Bij JR c,e kan slechts uit 4 voorwaarden c gekozen worden.

Opcode	Mnemonic	Verklaring
20	JR NZ,e	Jump relatief als Z=0.
28	JR Z,e	Jump relatief als Z=1.
30	JR NC,e	Jump relatief als C=0.
38	JR C,e	Jump relatief als C=1.

Het displacement e wordt op dezelfde manier berekend als bij de JR e-instructie.

Taak. Vervang in P9B1 de instructie JP NC,180E door een relatieve voorwaardelijke sprong.

9.F. Programma-lussen

Om een probleem op te lossen, moet soms een instructie of een groep instructies een bepaald aantal keren uitgevoerd worden. Als voorbeeld de vier opeenvolgende ADD HL,HL in P9A1. Vier zelfde instructies na elkaar kunnen misschien nog wel verantwoord zijn, maar de grens ligt zeker niet veel hoger.
Neem aan dat de lokaties 1900 tot 195F alle met 00 geladen moeten worden.
Een programma in de volgende vorm zou het wel doen, maar wordt zeer lang en eentonig :

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	21 0019	P9F1	LD HL,1900	.....
1803	36 00		LD (HL),00	Laad loc.1900 met 00
1805	23		INC HL	Verhoog HL tot 1901
1806	36 00		LD (HL),00	Laad loc.1901 met 00
1808	23		INC HL	Verhoog HL tot 1902
enz.				
....	23		INC HL	Verhoog HL tot 195F
....	36 00		LD (HL),00	Laad loc.195F met 00
....	FF		RST 38	Naar monitor

Een programma met een lus, die de beide instructies 60 maal uitvoert is veel korter.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	21 0019	P9F2	LD HL,1900	.....
1803	06 60		LD B,60	.....
1805	36 00	LUS	LD (HL),00	.....
1807	23		INC HL	.....
1808	05		DEC B	Lusteller
1809	20 FA		JR NZ,LUS	Terug naar lus als B>0
180B	FF		RST 38	.....

Na uitvoering van P9F2 moeten alle lokaties van 1900 tot 195F met 00 geladen zijn.
Het register B wordt gebruikt als lus-teller, en hier vooraf met 60 geladen. Telkens wanneer de lus doorlopen wordt, wordt B met één verlaagd. Zolang B nog niet 0 is, maakt de relatieve jump dat de lus nog eens doorlopen wordt.
Als P tot 00 verlaagd is, wordt flag Z = 1, waardoor niet meer voldaan wordt aan de voorwaarde voor de relatieve jump. Er wordt dan overgegaan naar de volgende instructie, die hier het programma beëindigt.
Met P9F2 kunnen tot max. 100 lokaties met 00 geladen worden. Daarvoor moet de lus-teller, register B, met 00 geladen worden.

Taak. Schrijf een programma dat de lokaties van 1900 tot 19FF laadt met de waarde van 00 tot FF.

9.G. Grotere lussen

Als een lus meer dan 100 maal doorlopen moet worden, kan een registerpaar als lus-teller gekozen worden. Bij het verlagen van een registerpaar worden echter de flags niet gesteld. Daardoor moeten bijkomende instructies controleren of het registerpaar op 0000 gekomen is, en de flag Z hoog maken.
Programma P9G1 demonstreert hoe dit kan. Vanaf 1820 worden 300 lokaties op 00 gesteld.

Adres	Mach. taal	Label	Mnemonic	Verklaring.
1800	21 2018	P9G1	LD HL,1820	.....
1803	01 0003		LD BC,0300	Laad lusteller.
1806	36 00	LUS	LD (HL),00	.....
1808	23		INC HL	.....
1809	0B		DEC BC	.....
180A	78		LD A,B	.....
180B	B1		OR A,C	.....
180C	20 F8		JR NZ,LUS	.....
180E	FF		RST 38	.....

9.H. DJNZ e (Decrement B, Jump if Not Zero)

Dit is een instructie die DEC B en JR NZ,e beide uitvoert. Eerst wordt register B met één verlaagd en als het niet nul is, wordt het displacement e bij de program counter opgeteld, en wordt overgegaan naar de instructie die de program counter aanwijst.
Wordt register B wel nul, dan wordt naar de instructie overgegaan die volgt op DJNZ e.
Het programma P9F2 kan nu met DJNZ e ingekort worden.

Adres	Mach. taal	Label	Mnemonic	Verklaring.
1800	21 0019	P9H1	LD HL,1900	.....
1803	06 60		LD B,60	.....
1805	36 11	LUS	LD (HL),11	.....
1807	23		INC HL	.....
1808	10 FB		DJNZ LUS	DEC.B Jump NZ naar LUS
180A	FF		RST 38	.....

Om het resultaat van het programma te herkennen worden de lokaties met 11 geladen.
Met DJNZ e kan de lus maximaal 100 maal doorlopen worden.

9.I. Opeenvolgende lussen

Stel dat het gewenst is dat in het gebied van 1900 tot 195F op alle even adressen 02 geladen wordt, en op alle oneven 01.
Een mogelijke oplossing is eerst een lus laten uitvoeren die al de even adressen laadt, en daarna met een tweede lus al de oneven adressen laten laden.

Adres	Mach. taal	Label	Mnemonic	Verklaring.
1800	21 0019	P9I1	LD HL,1900	Begin lus pare locat.
1803	06 30		LD B,30	.....
1805	36 02	LUSP	LD (HL),02	.....
1807	23		INC HL	.....
1808	23		INC HL	.....
1809	10 FA		DJNZ LUSP	.....
180B	21 0119		LD HL,1901	Begin onpare adressen
180E	06 30		LD B,30	.....
1810	36 01	LUSO	LD (HL),01	.....
1812	23		INC HL	.....
1813	23		INC HL	.....
1814	10 FA		DJNZ LUSO	.....
1816	FF		RST 38	.....

De twee lussen staan hier gewoon na elkaar, de ene heeft geen invloed op de andere.

Taak. Probeer hetzelfde resultaat met een enkele lus te verkrijgen.

9.J. In elkaar genestelde lussen

Stel dat de lokaties vanaf 1900 to 190F met 00 geladen moeten worden, van 1910 tot 191F met 01,.... en van 1950 tot 195F met 05. Een mogelijkheid zou zijn zes bijna gelijke lussen na elkaar te laten uitvoeren.
Veel korter wordt het programma als de lus die de 10 lokaties laadt (binnenlus) opgenomen wordt in een buitenlus, die telt hoeveel maal de binnenlus volledig wordt uitgevoerd.

Adres	Mach. taal	Label	Mnemonic	Verklaring.
1800	21 0019	P9J1	LD HL,1900	.....
1803	3E FF		LD A,FF	A laden met -1
1805	0E 06		LD C,06	Buitenlusteller laden.
1807	3C	BUIL	INC A	Begin buitenlus.
1808	06 10		LD B,10	Binnenlusteller laden.
180A	77	BINL	LD (HL),A	Begin binnenlus.
180B	23		INC HL	.....
180C	10 FC		DJNZ BINL	Einde binnenlus.
180E	0D		DEC C	.....
180F	20 F6		JR NZ,BUIL	Einde buitenlus.
1811	FF		RST 38	.....

Telkens wanneer de binnenlus volledig is afgeteld, wordt register C verlaagd, en als het niet nul is, wordt teruggesprongen naar het begin van de buitenlus. Daar wordt A met 1 verhoogd, en B opnieuw met 10 geladen. De binnenlus zal daardoor de 10 volgende lokaties met een waarde die 1 hoger is, laden. Zodra de buitenlus zes maal is doorlopen (en de binnenlus 60 maal), is C tot 00 verlaagd en eindigt het programma. Met figuur F9.J.1 wordt dit nog eens verduidelijkt.
Bij de start van het programma moet register A met FF geladen worden, om na de eerste INC A op 00 te staan.

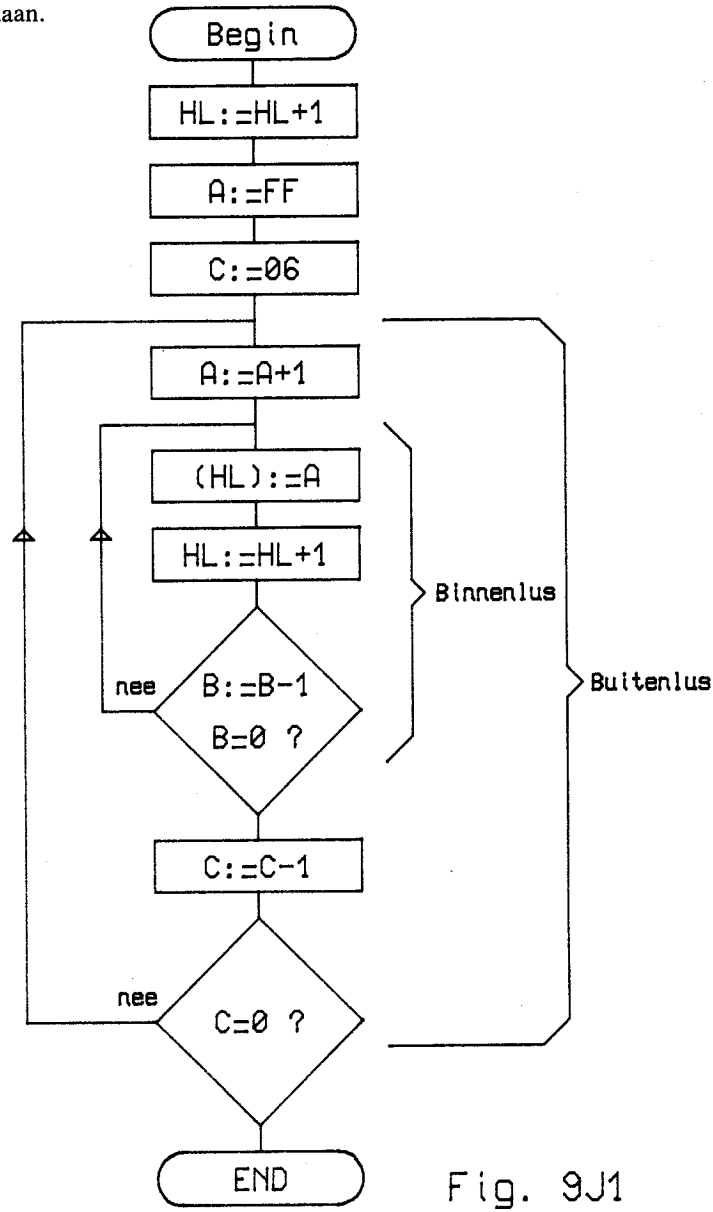


Fig. 9J1

9.K. Lus verlagen na vergelijking

Een lus heeft niet noodzakelijk een lus-teller nodig die zegt wanneer de lus verlaten moet worden. Dit sein kan ook door andere omstandigheden opgewekt worden. Bijvoorbeeld, als HL een bepaalde waarde heeft bereikt.
Stel dat de inhoud van de lokaties 1900 tot 195F met 1 moet worden verhoogd.

Adres	Mach. taal	Label	Mnemonic	Verklaring.
1800	21 0019	P9K1	LD HL,1900	.....
1803	3E 60		LD A,60	.....
1805	34	LUS	INC (HL)	Verhoog inhoud locatie
1806	23		INC HL	Wijs volgende loc.aan.
1807	BD		CP A,L	Is L=60 ?
1808	20 FB		JR NZ,LUS	Neen, dan naar LUS.
180A	FF		RST 38	.....
1900	00			
1901	01			
1902	02			
195E	5E			
195F	5F			

De instructie INC (HL) verhoogt de inhoud van de aangewezen lokatie, waarna INC HL het registerpaar verhoogt. Met CP A,L wordt nagegaan of L = 60 is, en als dat niet het geval is wordt de lus weer opgenomen.
Daar hier alleen register L gecontroleerd moet worden is het programma vrij eenvoudig.
Iets moeilijker wordt het als de inhoud van de lokaties van 1830 tot 193F verhoogd moet worden, daar nu zowel H als L gecontroleerd moet worden.

Adres	Mach. taal	Label	Mnemonic	Verklaring.
1800	21 3018	P9K1	LD HL,1830	.....
1803	3E 40	LUS2	LD A,40	Eindwaarde L+1.
1805	34	LUS1	INC (HL)	.....
1806	23		INC HL	.....
1807	BD		CP A,L	L voorbij 3F ?
1808	20 FB		JR NZ,LUS1	Neen, dan verder doen.
180A	3E 19		LD A,19	Eindwaarde H.
180C	BC		CP A,H	Is H=19 ?
180D	20 F4		JR NZ,LUS2	Neen, dan naar LUS2.
180F	FF		RST 38	.....
1830	30			
1831	31			
1832	32			
1840	40			
193E	3E			
193F	3F			

9.L. Lus verlaten na datacontrole

Als een bepaalde waarde onder normale omstandigheden niet tot de data zal behoren, kan deze waarde gebruikt worden als teken dat de lus verlaten moet worden.
Stel dat de inhoud van een blok vanaf adres 1820 tot aan de lokatie die 00 bevat, overgebracht moet worden naar het RAM-gebied vanaf 1900.

Adres	Mach. taal	Label	Mnemonic	Verklaring.
1800	21 2018	P9L1	LD HL,1820	Begin bron.
1803	11 0019		LD DE,1900	Begin bestemming.
1806	7E	LUS	LD A,(HL)	Van bron
1807	12		LD (DE),A	Naar bestemming.
1808	23		INC HL	Wijs volgende bron loc
1809	13		INC DE	Wijs volgende best.loc
180A	A7		AND A,A	Stel flags.
180B	20 F9		JR NZ,LUS	Terug als A>0.
180D	FF		RST 38	.....
1820	20			
1821	21			
1822	22			
182F	2F			
1830	00	TERM		

De instructie AND A,A is nodig, omdat geen enkele van de voorgaande instructies de flags stelt naar de inhoud van A. Zijn de overgebrachte data 00, dan zal flag Z hoog zijn, en de lus worden verlaten.
Deze methode is alleen bruikbaar als in de over te brengen data nooit de waarde 00 voorkomt. Een voordeel is dat de lengte van het over te brengen blok niet afhankelijk is van het programma, maar van de plaats waar de waarde 00 (terminator of stopper) staat.

9.M. JP (HL)

Doet onvoorwaardelijk overspringen naar het adres dat in registerpaar HL staat.
Een werkelijk nuttig programma met JP (HL) is nu nog moeilijk. Daarom wordt, alleen om aan te tonen hoe JP (HL) gebruikt moet worden, P9A1 herschreven met JP (HL).

Adres	Mach. taal	Label	Mnemonic	Verklaring.
1800	3E 15	P9L1	LD A,15	.....
1802	06 03		LD B,03	.....
1804	80		ADD A,B	.....
1805	21 1818		LD HL,1810	Bestemmingsadres in HL
1808	E9		JP (HL)	Naar adr.dat in HL sta
1810	26 00	AM10	LD H,00	.....
1812	6F		LD L,A	.....
1813	29		ADD HL,HL	.....
1814	29		ADD HL,HL	.....
1815	29		ADD HL,HL	.....

1816 1817	29 FF		ADD HL,HL RST 38	....

9.N. JP (IX + d), JP (IY + d)

Doet onvoorwaardelijk overspringen naar het adres dat IX + d of IY + d aanwijst.

10. HET STAPELGEHEUGEN

10.A. De stack pointer

Dit is een 16 bit register, net als de indexregisters. De stack pointer kan eveneens een lokatie aanwijzen, als hij vooraf met een adres geladen wordt. Het laden van de SP kan immediate of extended gebeuren. Daarbij komt dan nog de mogelijkheid, de inhoud van HL, IX of IY naar de SP over te brengen.

Mach. taal	Mnemonic	Verklaring.
31 9F1F	LD SP,1F9F	Laad SP met 1F9F
.. ....	LD SP,(1FD0)	Laad SP met inhoud van 1FD0 en 1FD1.
..	LD SP,HL	Laad SP met de inhoud van HL.
....	LD SP,IX	Laad SP met .....
....	LD SP,IY	.....

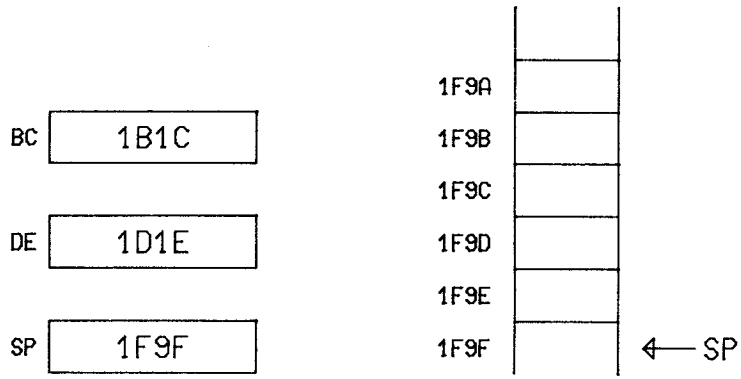
Door de eigenschappen van de SP is het mogelijk, de inhoud van alle registers in de RAM op te bergen, zonder dat een adres opgegeven wordt. Het deel van de RAM dat daardoor bezet wordt, heet de *stack*.
Als de opdracht wordt gegeven om de inhoud van een register op de stack te plaatsen, wordt eerst de SP 1 lager. Daarna wordt de inhoud van het register naar de door de SP aangewezen lokatie overgebracht.
Moet nog een register op de stack geplaatst worden, dan zal weer eerst de SP 1 lager worden, en de waarde naar de nieuw aangewezen lokatie overgebracht worden.
Bij de Z-80 kunnen de registers niet afzonderlijk op de stack geplaatst worden ; dit moet altijd per paar gebeuren, en wel in de voorgeschreven volgorde.

10.B. PUSH qq, PUSH IX en PUSH IY

Verlaagt de SP met 1, en laadt de high order byte van het registerpaar in de geheugenplaats die nu door de SP wordt aangewezen. Daarna wordt de SP nog eens met 1 verlaagd, en de low order byte van het registerpaar op het adres geladen dat de SP aanwijst.

Mach. taal	Mnemonic	Verklaring
F5	PUSH AF	Drukt het registerpaar AF op de stack
..	PUSH BC	.....
..	PUSH DE	.....
..	PUSH HL	.....
....	PUSH IX	Drukt register IX op de stack.
....	PUSH IY	.....

Het volgend programma laadt de registerparen BC en DE, en drukt daarna beide registerparen op de stack, Figuur 10.B.1 verduidelijkt, hoe de registers op de stack worden geplaatst.



Na PUSH BC

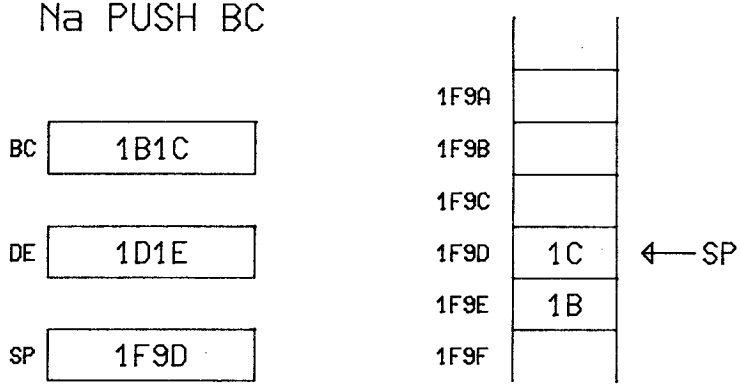
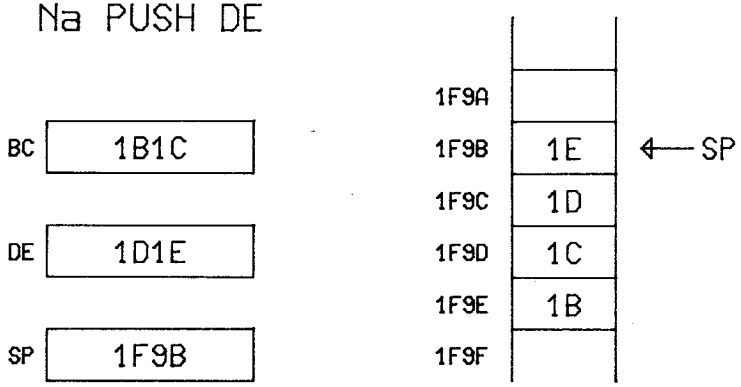


Fig. 10B1

Na PUSH DE



Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	31 9F1F	P10B1	LD SP,1F9F	Wijs de stack aan.
1803	01 1C1B		LD BC,1B1C	.....
1806	11 1E1D		LD DE,1D1E	.....
1809	C5		PUSH BC	B naar 1F9E,C naar 1F9D
180A	D5		PUSH DE	.....
180B	FF		RST 38	.....

Let erop dat de SP geladen moet worden met het beginadres + 1 van de stack.
Na het uitvoeren van P10B1 kan nagegaan worden of het resultaat overeenkomt met figuur 10.B.1.

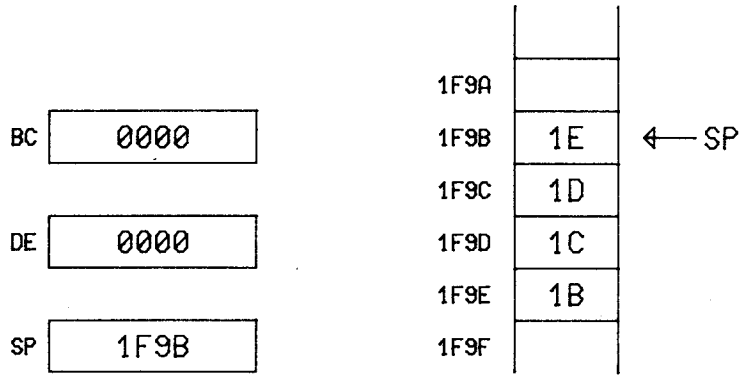
10.C. POP qq, POP IX en POP IY

Laadt de inhoud van de lokatie die de SP aanwijst in de low order portion van het betrokken registerpaar, en verhoogt dan de SP met 1. Daarna wordt de inhoud van de lokatie die de SP aanwijst, in de high order portion van hetzelfde registerpaar geladen, en de SP weer met 1 verhoogd.
Met andere woorden : het registerpaar wordt geladen met de inhoud van de twee bovenste lagen van de stack.

Mach. taal	Mnemonic	Verklaring
F1	POP AF	Laad registerpaar AF vanaf de stack.
..	POP BC	.....
..	POP DE	.....
..	POP HL	.....
....	POP IX	Laad register IX vanaf de stack.
....	POP IY	.....

Met P10C1 wordt de werking van de stack gedemonstreerd.
Nadat BC en DE op de stack zijn geplaatst, worden ze met 0000 geladen. Door de instructies POP DE en POP BC worden ze met hun originele inhoud uit de stack geladen.
Figuur 10.C.1 toont de toestand na de bijgekomen instructies.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	31 9F1F	P10C1	LD SP,1F9F	.....
1803	01 1C1B		LD BC,1B1C	.....
1806	11 1E1D		LD DE,1D1E	.....
1809	C5		PUSH BC	.....
180A	D5		PUSH DE	.....
180B	01 0000		LD BC,0000	Stel BC op nul
180E	11 0000		LD DE,0000	.....
1811	D1		POP DE	Haal DE uit de stack
1812	C1		POP BC	.....
1813	FF		RST 38	.....



Na POP DE

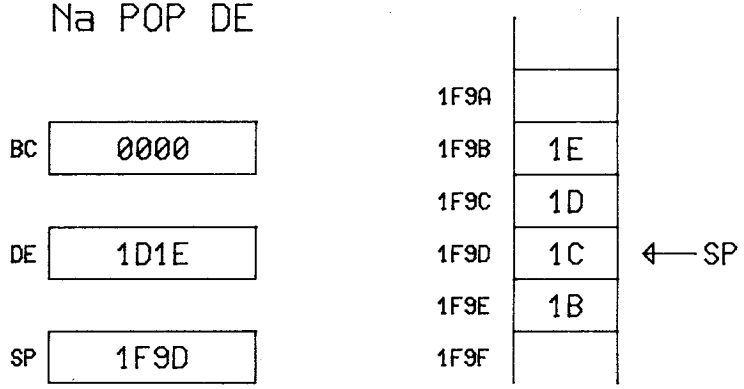
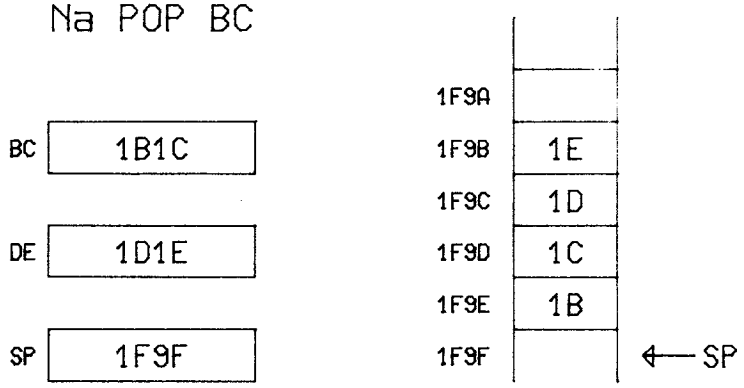


Fig. 10C1

Na POP BC



Na het uitvoeren van P10C1 moet de toestand overeenkomen met die uit figuur 10.C.1. Het ophalen uit de stack moet gebeuren in een volgorde die de omgekeerde is van die van het plaatsen, om de oorspronkelijke waarden in dezelfde registers terug te krijgen.

10.D. De plaats voor de stack

RAM is aanwezig op de adressen van 1800 tot 1FFF. Daarvan wordt het deel van 1F9F tot 1FFF door de monitor gebruikt.
Als stack is iedere plaats in de RAM bruikbaar, maar daar de stack aangroeit naar de lagere adressen, kan men de stack het beste op het hoogst beschikbare adres beginnen. Dan is de kans dat het user program en de user stack in elkaar groeien, het kleinst. Daar bij een PUSH de SP eerst met 1 verlaagd wordt, moet de SP geladen worden met 1F9F.
Ook de monitor maakt gebruik van de mogelijkheden die de stack pointer biedt, maar gebruikt daarvoor wel een afzonderlijke stack, de *system stack* genoemd, die begint op 1FAE. De monitor doet de system stack echter nooit verder aangroeien dan adres 1F9F (zie figuur 10.D.1).
Het adresgebied van 1FBC tot 1FDD wordt de *registerbuffer* genoemd. Daarin kan de inhoud van alle registers bewaard worden, als een user program onderbroken wordt, en de monitor de CPU gaat gebruiken. Dit is het geval als het user progam bij een breakpoint komt, of wanneer het programma stap voor stap wordt uitgevoerd. Voordat het user program dan voortgezet wordt, worden al de registers weer geladen met de waarden uit de registerbuffer (zie instructies vanaf 02B5 tot 02CB).
Na het inschakelen van de spanning of het drukken op de toets RESET laadt de monitor de lokaties 1FD0 en 1FD1 (USERSP) met 1F9F (zie instructies van 0054 tot 0059), en de lokaties 1FDC en 1FDD (USERPC) met 1800 (zie instructies 0017 tot 0023).
Bij het starten van een user program worden door het bedienen van de toets GO ook alle registers geladen met de inhoud van de registerbuffer (zie instructies vanaf 02B5 tot 02CB). Dit betekent dat de stack pointer geladen wordt met de waarde 1F9F, en de gebruiker in zijn programma de stack pointer niet moet laden (tenzij hij de stack op een andere plaats wenst). Op grond van deze kennis kan besloten worden dat de eerste instructie in de programma's P10B1 en P10C1 overbodig is.
Bij gebruik van het stapelgeheugen moet men er altijd op letten dat alles wat geplaatst wordt, ook weer wordt weggehaald.
Natuurlijk mag ook niet meer weggehaald worden dan er geplaatst werd, want dan komt men in de system stack terecht.
Telkens wanneer een user program onderbroken wordt, controleert de monitor of de USERSP niet naar de system stack wijst, en meldt dit eventueel op de display met SYS-SP. In P1OD1 is opzettelijk zo'n fout aangebracht. Laat het uitvoeren, en controleer de inhoud van de stack pointer.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	01 2C2B	P10D1	LD BC,2B2C	.....
1803	C5		PUSH BC	.....
1804	01 0000		LD BC,0000	.....
1807	C1		POP BC	.....
1808	D1		POP DE	.....
1809	FF		RST 38	.....

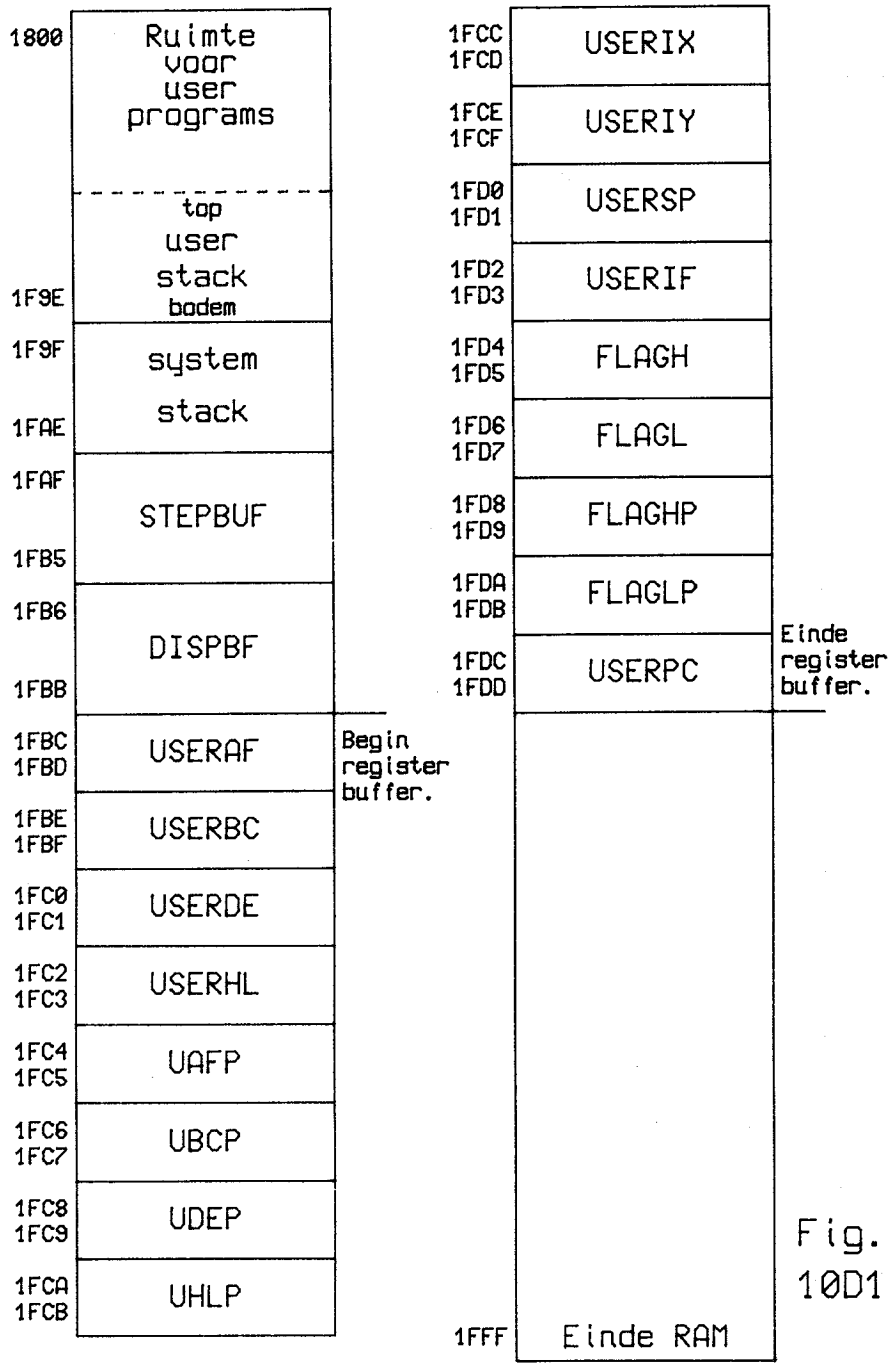

1800
Ruimte voor user programs
top user stack bodem
1F9E
1F9F
system stack
1FAE
1FAF
STEPBUF
1FB5
1FB6
DISPBF
1FBB
1FBC 1FBD
USERAF
Begin register buffer.
1FBE 1FBF
USERBC
1FC0 1FC1
USERDE
1FC2 1FC3
USERHL
1FC4 1FC5
UAFP
1FC6 1FC7
UBCP
1FC8 1FC9
UDEP
1FCA 1FCB
UHLP
1FCC 1FCD
USERIX
1FCE 1FCF
USERIY
1FD0 1FD1
USERSP
1FD2 1FD3
USERIF
1FD4 1FD5
FLAGH
1FD6 1FD7
FLAGL
1FD8 1FD9
FLAGHP
1FDA 1FDB
FLAGLP
1FDC 1FDD
USERPC
Einde register buffer.
1FFF
Einde RAM

Fig. 10D1

10.E. EX AF,AF'

De Z-80 heeft ook nog het registerpaar AF'. De enige bewerking die ermee kan worden uitgevoerd, is het uitwisselen van de inhoud ervan met die van AF.
Ook het laden van AF' kan alleen door eerst A en F te laden, en daarna de instructie EX AF,AF' uit te voeren.
Opcode 08. Er worden geen flags beïnvloed.
Het registerpaar AF' kan ook gebruikt worden om de inhoud van AF tijdelijk te bewaren. Ex AF,AF' is sneller dan PUSH AF, maar heeft niet de mogelijkheid om AF meerdere keren na elkaar op te slaan.

10.F. EXX

Ook de registerparen BC, DE en HL hebben hun uitwisselregister, die aangeduid worden met BC', DE' en HL'.
De instructie EX ruilt de inhouden van de registerparen BC, DE en HL met die van BC', DE' en HL'.
Opcode D9. Er worden geen flags beïnvloed.

10.G. EX DE,HL

Met deze instructie wordt de inhoud van register DE geruild met de inhoud van register HL.
Opcode EB. Er worden geen flags beïnvloed.
Kan ook gebrukt worden om de inhoud van DE of HL tijdelijk te bewaren als het andere registerpaar nog vrij is.

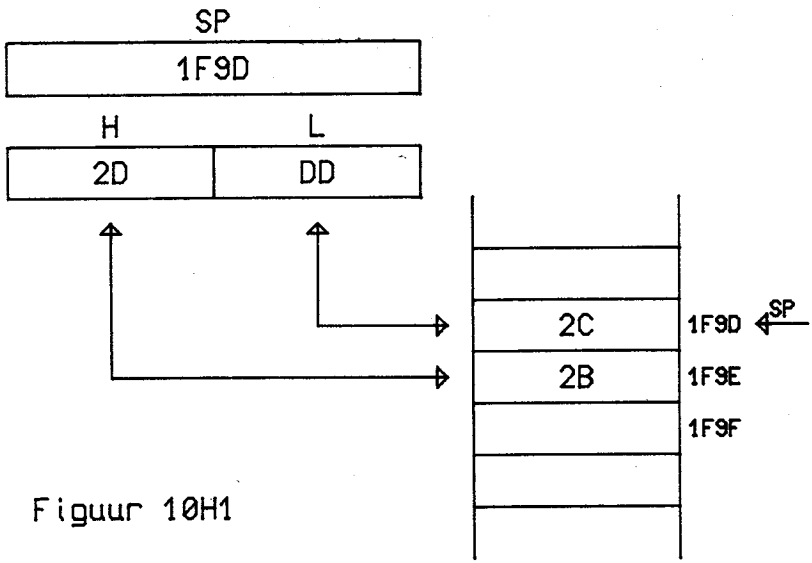
10.H. EX (SP),HL

Ruilt de inhoud van register L met de inhoud van de lokatie die de SP aanwijst, en register H met de inhoud van de lokatie op adres SP + 1. Met andere woorden : ruilt HL met de top van de stack. De SP wordt daarbij niet veranderd.
Opcode E3. Er worden geen flags beïnvloed.
Het gebeuren bij EX (SP),HL is grafisch weergegeven in figuur 10.H.I.

10.I. EX (SP),IX, EX (SP),IY

Ruilt de top van de stack met IX of IY.
Opcode DDE3 en FDE3. Er worden geen flags beïnvloed.
Het is nu nog niet mogelijk om met deze instructies zinvolle en eenvoudige programma's op te bouwen. Daarom heeft P10I1 slechts als doel, hun gebruik te demonstreren. Voorspel vooraf de inhoud van de registers als P10I1 uitgevoerd zal zijn. Laat desnoods het programma stap voor stap uitvoeren, en controleer telkens de registers.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 1A	P10I1	LD A,1A	Laad A
1802	A7		AND A	Laad F
1803	08		EX AF,AF'	Laad AF'
1804	3E 2A		LD A,2A	Herlaad A
1806	A7		AND A	Herlaad F
1807	01 1C1B		LD BC,1B1C	Laad BC
180A	11 1E1D		LD DE,1D1E	Laad DE
180D	21 DD1D		LD HL,1DDD	Laad HL
1810	D9		EXX	Laad BC' DE' HL'
1811	01 2C2B		LD BC,2B2C	Herlaad BC
1814	11 2E2D		LD DE,2D2E	Herlaad DE
1817	21 DD2D		LD HL,2DDD	Herlaad HL
181A	DD21 EE1E		LD IX,1EEE	Laad IX
181E	FD21 FF1F		LD IY,1FFF	Laad IY
1822	C5		PUSH BC	BC op stack
1823	E3		EX (SP),HL	Ruil stacktop met HL
1824	EB		EX DE,HL	Ruil DE met HL
1825	DDE3		EX (SP),IX	Ruil stacktop met IX
1827	FDE3		EX (SP),IY	Ruil stacktop met IY
1829	C1		POP BC	Herlaad BC uit stack
182A	76		HALT	



Figuur 10H1

11. DATA OUTPUT

11.A. Inleiding

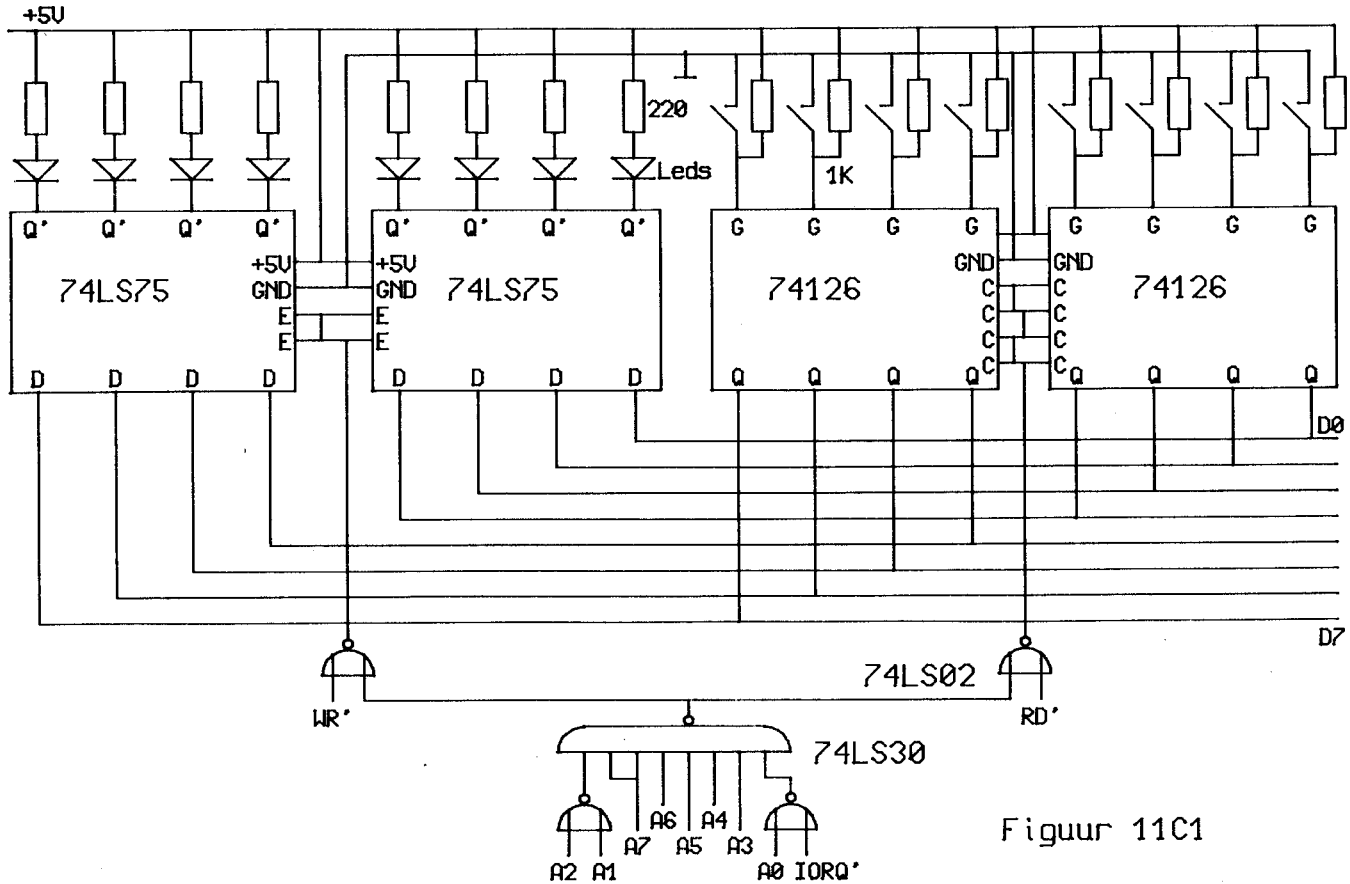
Het gebruik van de Micro-Professor begint pas interessant te worden als een uitwendig toestel gestuurd kan worden.
Zo'n randtoestel (peripheral device) kan in zijn eenvoudigste vorm een groep van acht leds zijn.
De Z-80 kan dan waarden naar de leds brengen via een aangepaste tussenschakeling (interface).

11.B. OUT (n),A

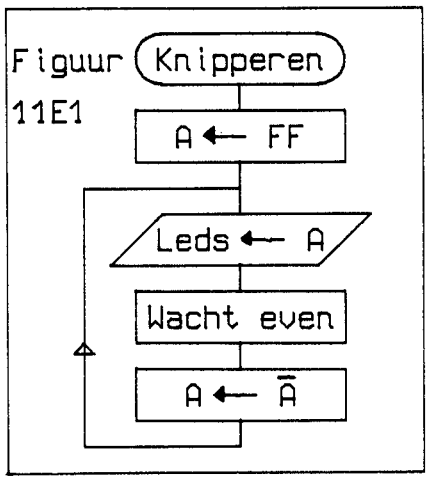
Deze instructie brengt op de adreslijnen A7-A0 de waarden (het poortadres), en op de databus de inhoud van A. Tegelijkertijd worden de uitgangen IORQ' en WR' laag.
De opcode is D3, en moet gevolgd worden door het poortadres.
Voorbeeld : D3 02 brengt de inhoud van A op de databus, en 02 op de adreslijnen A7-A0.
De flags worden door OUT (n),A niet beïnvloed.

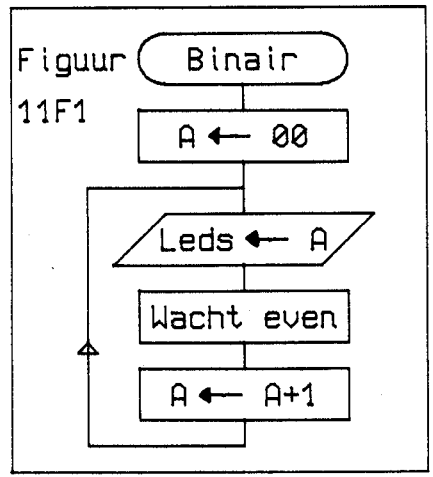
11.C. De interface

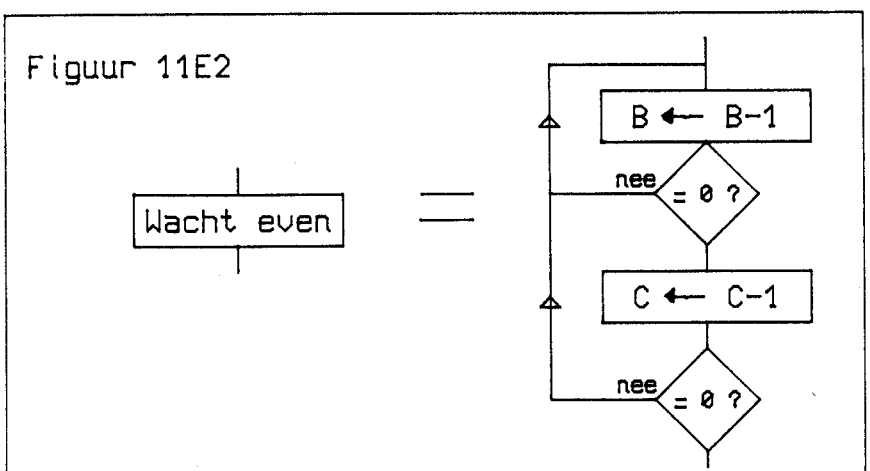
De inhoud van A staat slechts een heel kort ogenblik op de databus. Daarom moeten deze signalen aan 8 D-flip flop's toegevoerd worden.
De uitgang van een D-flip flop volgt de ingang D, zolang zijn ingang Enable hoog is. Op het ogenblik dat de ingang Enable laag wordt, blijft de uitgang Q zijn toestand behouden.
Van het schema uit figuur 11.C.1 wordt nu slechts de linkerhelft met de leds besproken.
De ingangen Enable van de beide 74LS75 worden gestuurd door een poort. Als de ingangen Enable kort hoog worden gemaakt, geven de leds de toestand van de databus weer op het ogenblik dat de ingangen E weer laag werden.
De ingangen Enable worden hoog als WR' en de uitgang van de 74LS30 beide laag zijn. Daarvoor moeten IORQ', A0, A1 en A2 laag zijn, en A3, A4, A5, A6 en A7 hoog.
Aan al deze voorwaarden kan alleen maar voldaan worden door de instructie OUT (F8),A. Het poortadres van de leds is hier F8.
Hoewel op het ogenblik alleen het deel met de leds nodig is, kan de interface toch het beste volledig gebouwd worden. De rechterhelft met de schakelaars is enkele hoofdstukken verder toch nodig.
De bouw gaat het eenvoudigst op een stukje print, die via een lintkabel verbonden wordt met de bovenste connector (Z-80 CPU BUS) van de Micro-Professor. Daar is echter de +5V niet beschikbaar. De interface kan door een afzonderlijk 5V-bron gevoed worden, maar het is ook mogelijk de niet-gebruikte klem 11 van P1 met de +5V van de Micro-Prof te verbinden.

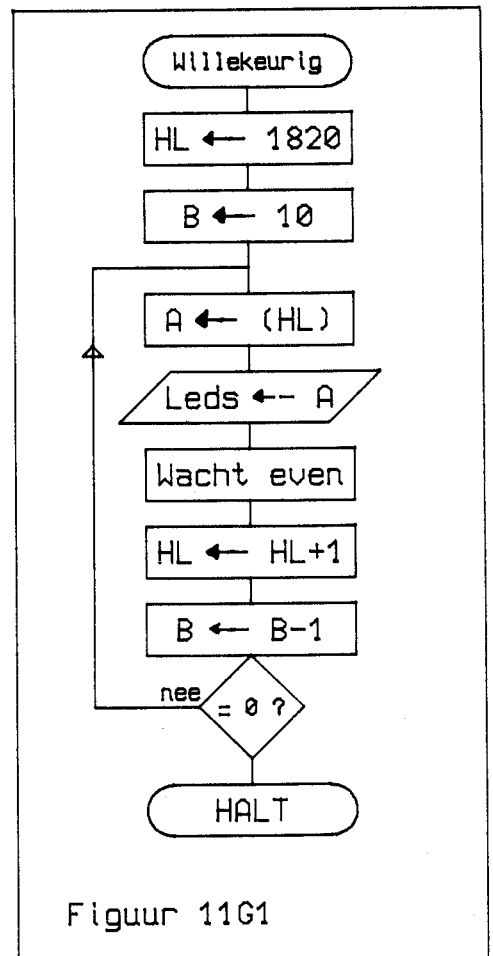

+5V
220
Leds
1K
74LS75
74LS75
74126
74126
Q*
D
G
Q
+5V
GND
E
C
D0
D7
74LS02
WR*
RD*
74LS30
A2
A1
A7
A6
A5
A4
A3
A0
IORQ*

Figuur 11C1


Figuur 11E1
Knipperen
A ← FF
Leds ← A
Wacht even
A ← Ā


Figuur 11F1
Binair
A ← 00
Leds ← A
Wacht even
A ← A+1


Figuur 11E2
Wacht even
=
B ← B-1
nee
= 0 ?
C ← C-1
nee
= 0 ?


Willekeurig
HL ← 1820
B ← 10
A ← (HL)
Leds ← A
Wacht even
HL ← HL+1
B ← B-1
nee
= 0 ?
HALT
Figuur 11G1

11.D. Het testen van de interface

Eerst moet er zekerheid zijn dat de interface behoorlijk functioneert, voordat met de studie van het sturen van de leds begonnen wordt. Dit kan het best met het volgende programma, dat zo eenvoudig mogelijk is gehouden, om de kans op software-fouten uit te sluiten.
De waarde die in lokatie 1801 staat, wordt door de instructie OUT (F8),A naar de leds overgebracht.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E FF	P11D1	LD A,FF	Waarde voor leds.
1802	D3 F8		OUT (F8),A	Naar leds.
1804	FF		RST 38	Monitor.

Bij het uitvoeren van de instructie OUT (F8),A staat een kort ogenblik F8 op de adreslijnen A7-A0, en FF op de databus. Tegelijkertijd zijn ook de uitgangen WR' en IORQ' van de Z-80 laag. Daardoor zijn de ingangen Enable van de beide 74LS75 even hoog, en gaan alle leds oplichten. Zodra de OUT (F8),A uitgevoerd is, verdwijnt de informatie van de bussen. De ingangen Enable zijn weer laag, maar alle leds blijven branden door de geheugeneigenschappen van de D-flip flop's.
Als de inhoud van lokatie 1801 vervangen wordt door 00, moeten alle leds doven als het gewijzigd programma uitgevoerd wordt.
Deze blijven dan uit tot een volgend programma hun toestand wijzigt.
Als deze proeven het gewenste resultaat leveren, is de kans groot dat de interface in orde is. Voor de zekerheid wordt de inhoud van lokatie 1801 nog eens gewijzigd tot 55, en het programma opnieuw uitgevoerd. Alle even leds (de meest rechtse is led 0) moeten nu gaan oplichten, en de oneven gedoofd blijven.
Met AA in lokatie 1801 moet het omgekeerde effect verkregen worden.
Als dit allemaal klopt, is er ook zekerheid dat er geen ongewenste verbindingen zijn tussen naast elkaar liggende kringen.

Taak 1. Laat alleen de vier rechtse leds oplichten.
Taak 2. Alleen de buitenste leds moeten oplichten.

11.E. Leds laten knipperen

De leds op de interface zijn de eenvoudigste verbruikers (toestellen) die door de Micro-Professor gestuurd kunnen worden.
De stroom die door de leds vloeit kan ook een triac (via een optokoppeling) of een relais (via een transistor) sturen, die dan grotere verbruikers (ook op 220V) als lampengroepen, weerstanden, motoren, elektrokleppen, stappenmotoren enz., kunnen sturen. Het in- en uitschakelen van deze verbruikers gebeurt dan eenvoudig door de juiste waarde naar de betrokken poort over te brengen.
Een eenvoudig voorbeeld is het, met korte tussenpozen, in- en uitschakelen van alle acht verbruikers (hier leds).

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E FF	P11E1	LD A,FF	FF naar register A
1802	D3 F8	LEDS	OUT (F8),A	Naar leds.

1804	10 FE	TIJD	DJNZ TIJD	Wacht
1806	0D		DEC C	hier
1807	20 FB		JR NZ,TIJD	even.
1809	2F		CPL	Bits van A omkeren.
180A	18 F6		JR LEDS	Herhaal.

Eerst de verklaring van de instructie CPL (ComPLement). Deze keert iedere bit van register A om. Iedere bit die hoog is wordt laag, en omgekeerd.
Het deel van adres 1804 tot 1808 bestaat uit twee in elkaar genestelde lussen die alleen de registers B en C aftellen. De CPU is daar ongeveer een halve seconde mee bezig, zodat een even lange wachttijd het gevolg is.
Na deze tijd worden alle bits van register A omgekeerd, en teruggesprongen naar de label LEDS, waar de gewijzigde inhoud van register A naar de leds wordt gebracht.
Het programma blijft in een gesloten lus lopen, waardoor de leds blijven knipperen.
De werking van het programma is ook gemakkelijk te volgen in de flowchart van figuur 11.E.1. Het overbrengen van de data naar de leds is voorgesteld door een trapezium. Dit is het gebruikelijke symbool voor een bewerking die data naar een randtoestel overbrengt. De rechthoeken stellen bewerkingen binnenin de CPU voor.
Een wachttijd kan ook veroorzaakt worden door slechts één register af te tellen. De tijd is dan uiteraard veel korter.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E FF	P11E2	LD A,FF	
1802	D3 F8	LEDS	OUT (F8),A	Data naar leds.
1804	10 FE	TIJD	DJNZ TIJD	Wacht even.
1806	2F		CPL	Alle bits omkeren.
1807	18 F9		JR LEDS	Herhaal.

Bij het beproeven van P11E2 blijven alle leds oplichten, maar wel zwakker. Het programma is niet fout, de leds flikkeren wel, maar in een veel hoger tempo dat niet meer met het oog te volgen is. Vandaar dat ze ook minder sterk oplichten. Ze zijn ook de helft van de tijd gedoofd.

11.F. Leds binair verhogen

Een van de grote voordelen van het sturen van toestellen (hier de leds) door een CPU is, dat een heel ander resultaat verkregen kan worden zonder dat één elektrische verbinding gewijzigd wordt. Het programma bepaalt het resultaat, een ander programma stuurt het randtoestel op een andere manier.
Voer P11F1 in, en probeer het. De flowchart is te vinden in figuur 11.F.1.

Adres	Mach. taal	Label	Mnemonic	Verklaring.
1800	97	P11F1	SUB A	Stel reg. A op nul.
1801	D3 F8	LEDS	OUT (F8),A	Naar leds.

1803	10 FE	TIJD	DJNZ TIJD	Wacht
1805	0D		DEC C	hier
1806	20 FB		JR NZ,TIJD	even.
1808	3C		INC A	Verhoog A met 1.
1809	18 F6		JR LEDS	Herhaal.

Het programma verhoogt de leds binair. Aan de oplichtende leds is te zien hoeveel maal de lus al doorlopen is geweest.
Als aangenomen wordt dat een verhoging gebeurt om de halve sec, dan kan ook de tijd dat het programma in uitvoering is van de leds afgelezen worden.
De wachttijd tussen iedere verhoging kan hier ook weer verkleind worden door slechts één register af te tellen. Het programma hoeft daarvoor niet herschreven te worden. Het is voldoende, de instructies DEC C en JR NZ,TIJD te vervangen door drie instructies NOP (No OPeration) met opcode 00.
Bij het uitvoeren van dit gewijzigd programma reageren de vier rechter-leds als deze uit P11E2. De vier linker-leds knipperen zichtbaar, maar niet met dezelfde frequentie. Door het binair verhogen knippert iedere led met een frequentie die de helft is van die van de led rechts van hem. Hiermee wordt ook bewezen dat de leds bij P11E2 wel degelijk knipperen.
Op de interface zijn nu acht verschillende frequenties beschikbaar. Ze kunnen samen gewijzigd worden door de wachttijd te veranderen.

Taak. Verlaag de leds binair in een laag tempo.

11.G. Willekeurige data toevoeren

In de vorige programma's worden de data die naar de leds worden gebracht, in de CPU opgewekt door een logische of wiskundige bewerking.
Als tussen verschillende waarden die naar de leds moeten worden gebracht geen logisch of wiskundig verband bestaat, moeten deze data vooraf in het geheugen worden geladen. Het programma moet ze daar ophalen, en in het gewenste tempo naar de leds brengen.
Onderstaand programma is daar een voorbeeld van. in figuur 11.G.1 is de flowchart weergegeven.

Adres	Mach. taal	Label	Mnemonic	Verklaring.
1800	21 2018	P11G1	LD HL,1820	Begin tabel.
1803	06 10		LD B,10	16 byte naar leds.
1805	7E	HAAL	LD A,(HL)	Haal uit tabel.
1806	D3 F8		OUT (F8),A	Naar leds.
1808	1B	TIJD	DEC DE	Wacht
1809	7A		LD A,D	hier
180A	B3		OR E	
180B	20 FB		JR NZ,TIJD	even.
180D	23		INC HL	Volgende byte aanwijz.
180E	10 F5		DJNZ HAAL	Herhaal tot B=0.
1810	76		HALT	

```
1820   00   01   02   04      Tabel
1824   08   10   20   40      met
1828   80   FF   00   FF      data.
182C   E7   C3   81   00
```

Voor de wachttijd wordt hier het registerpaar DE afgeteld, daar het register B gebruikt wordt voor het tellen van de bytes.

12. SUBROUTINES

12.A. Inleiding

Stel dat een optelling van twee binaire getallen op de leds gedemonstreerd moet worden. Daarbij moet eerst het opteltal op de leds komen, daarna de opteller en vervolgens de som. Als slot moeten alle leds doven.
Om de gelegenheid te scheppen om de binaire waarde op de leds te lezen, moet na iedere waarde een wachttijd in acht genomen worden.
Dit kan bereikt worden met het volgende programma.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	11 0000	P12A1	LD DE,0000	Instellen wachttijd.
1803	3E 11		LD A,11	Opteltal.
1805	D3 F8		OUT (F8),A	Naar leds.
1807	15	TIJD1	DEC D	Wacht
1808	20 FD		JR NZ,TIJD1	hier
180A	1D		DEC E	even.
180B	20 FA		JR NZ,TIJD1	
180D	47		LD B,A	Bewaar opteltal.
180E	3E 22		LD A,22	Opteller.
1810	D3 F8		OUT (F8),A	Naar leds.
1812	15	TIJD2	DEC D	Wacht
1813	20 FD		JR NZ,TIJD2	hier
1815	1D		DEC E	even.
1816	20 FA		JR NZ,TIJD2	
1818	80		ADD B	Maak som.
1819	D3 F8		OUT (F8),A	Naar leds.
181B	15	TIJD3	DEC D	Wacht
181C	20 FD		JR NZ,TIJD3	hier
181E	1D		DEC E	even.
181F	20 FA		JR NZ,TIJD3	
1821	97		SUB A	Maak A=0
1822	D3 F8		OUT (F8),A	Naar leds.
1824	FF		RST 38	Monitor.

Voor het opwekken van de wachttijd wordt hier het registerpaar DE gebruikt, daar register B al gebruikt wordt voor het maken van de optelling.
Het programma is betrekkelijk lang, vooral door de drie gelijke delen voor het opwekken van de wachttijden.
Het herhalende programmadeel kan ook nog veel groter zijn, en is soms tientallen keren in eenzelfde programma nodig. Een niet zo ingewikkeld programma zou met deze werkwijze zeer lang en eentonig zijn.

Zo'n zelfde programmadeel dat meerdere malen nodig is, kan ook afzonderlijk als een soort onderprogramma (subroutine) geschreven worden. De subroutine staat dan niet meer in het programma zelf, maar mag op iedere plaats in de RAM of de ROM geschreven staan.
In het hoofdprogramma mag dan op een willekeurige plaats het bevel gegeven worden om de subroutine uit te voeren. In de instructie die de subroutine oproept, moet het adres opgegeven zijn waar de subroutine begint. Na het uitvoeren van de subroutine wordt het hoofdprogramma voortgezet met de instructie die volgt op die welke de subroutine opgeroepen heeft.

12.B. CALL nn RET

De instructie CALL nn roept de subroutine op die begint op adres nn. Van adres nn staat ook weer de LSB eerst en dan de MSB.
Voorbeeld

1827 CD 0019 CALL 1900 Roep subroutine op die begint op adres 1900.

Als de CPU deze instructie gelezen heeft, is zijn program counter met 3 verhoogd. De instructie CALL doet dan niets anders dan de inhoud van PC op de stack drukken, en daarna nn in de PC laden. Daardoor wordt de uitvoering voortgezet op de plaats waar de subroutine begint.
Iedere subroutine moet altijd afgesloten worden met de instructie RET (RETurn). Deze instructie laadt de PC met de top van de stack. Daardoor wordt na RET het hoofdprogramma voortgezet op het adres na de CALL nn. Daar staat ook de eerste instructie die nog uitgevoerd moet worden.
Het programma uit P12A kan nu als volgt herschreven worden :

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	11 0000	P12B1	LD DE,0000	Instellen wachttijd.
1803	3E 11		LD A,11	Opteltal.
1805	D3 F8		OUT (F8),A	Naar leds.
1807	CD 3018		CALL TIJD1	Wacht even.
180A	47		LD B,A	Bewaar opteltal.
180B	3E 22		LD A,22	Opteller.
180D	D3 F8		OUT (F8),A	Naar leds.
180F	CD 3018		CALL TIJD1	Wacht even.
1812	80		ADD B	Maak som.
1813	D3 F8		OUT (F8),A	Naar leds.
1815	CD 3018		CALL TIJD1	Wacht even.
1818	97		SUB A	Maak A=0 .
1819	D3 F8		OUT (F8),A	Doof leds.
181B	FF		RST 38	Monitor.
1830	15	TIJD1	DEC D	Subroutine
1831	20 FD		JR NZ,TIJD1	
1833	1D		DEC E	
1834	20 FA		JR NZ,TIJD1	
1836	C9		RET	Terug naar hoofdprogr.

Taak. Verkort het programma P12B1 door het uitbreiden van de subroutine.

Het gebruik van subroutines maakt een programma niet alleen korter, maar ook overzichtelijker.
In eenzelfde programma mogen ook meerdere subroutines gebruikt worden. Een subroutine mag op zijn beurt ook weer een andere subroutine oproepen.
Bij het gebruik van subroutines moet er altijd rekening mee gehouden worden dat de subroutines de inhoud van bepaalde registers te niet kan doen. Gebruik daarom nooit een subroutine waarvan de eigenschappen niet volledig bekend zijn.
Van vreemde subroutines moet men weten, welke registers met een bepaalde waarde geladen moeten zijn voordat de subroutine opgeroepen wordt. Ook moet bekend zijn, in welke registers welk resultaat komt en van welke registers de inhoud te niet gedaan wordt.

12.C. CALL cc,nn

Nadat de CPU de instructie CALL cc,nn gelezen heeft, en daarvoor de PC met 3 heeft verhoogd, controleert hij de voorwaarde cc. Indien de voorwaarde cc niet aanwezig is, wordt gewoon de volgende instructie uitgevoerd, en indien wel, dan wordt naar de subroutine op adres nn overgegaan, net als in 12.B.
De voorwaarde waaraan voldaan moet worden, is de stand van een van de flags uit het register F.

Opcode	Mnemonic	Subroutine wordt uitgevoerd als:
C4 nn	CALL NZ,nn	Z = 0
CC nn	CALL Z,nn	Z = 1
D4 nn	CALL NC,nn	C = 0
DC nn	CALL C,nn	C = 1
E4 nn	CALL PO,nn	P/V = 0 (som hoge bits is onpaar)
EC nn	CALL PE,nn	P/V = 1 (hoge bits paar of overflow)
F4 nn	CALL P,nn	S = 0 (Plus)
FC nn	CALL M,nn	S = 1 (Min)

12.D. RET cc

Betekent eveneens de terugkeer vanuit een subroutine naar het hoofdprogramma, maar als de voorwaarde cc aanwezig is. Is dit niet het geval, dan wordt overgegaan naar de volgende instructie van de subroutine.

Opcode	Mnemonic	Doet terugkeren naar het hoofdprogr.als:
C0	RET NZ	Z = 0
C8	RET Z	Z = 1
D0	RET NC	C = 0
D8	RET C	C = 1
E0	RET PO	P/V = 0 (som hoge bits is onpaar)
E8	RET PE	P/V = 1 (hoge bits paar of overflow)
F0	RET P	S = 0 (Plus)
F8	RET M	S = 1 (Min)

De subroutine TIJD1 uit P12B1 kan nu ook als volgt geschreven worden :

Adres	Mach. taal	Label	Mnemonic	Verklaring
1830	15	TIJD2	DEC D	
1831	20 FD		JR NZ,TIJD2	
1833	1D		DEC E	
1834	C8		RET Z	
1835	18 F9		JR TIJD2	

13. UITVOERINGSTIJD

13.A. Inleiding

Op de Micro-Professor is een oscillator aanwezig die een frequentie van 3.579545 MHz opwekt. Deze wordt eerst door 2 gedeeld, en dan aan de Z-80 aangeboden, zodat deze op een frequentie van 1.78977 MHz werkt.
De tijd die nodig is om een bepaalde instructie uit te voeren staat vermeld in de instructietabel, en wordt opgegeven in T-states. Een T-state duurt één periode van de klokfrequentie.
Eén periode duurt 1/1 789 770 sec, of 0,000 000 558 731 sec, dit is .558 731 microsec.
Enkele voorbeelden :

Instructie	T-states.	
OUT (n),A	11	
SUB A,r	4	
LD A,n	7	
LD A,r	4	
INC (IX)	23	
JP nn	10	
JR e	12	
JR Z,e	12	als voorwaarde aanwezig is
JR Z,e	7	als voorwaarde niet aanwezig is

In P13A1 lichten de leds op het einde van de eerste OUT (F8),A-instructie op, en doven op het einde van de tweede OUT (F8),A. De leds zijn dan opgelicht tijdens het uitvoeren van de instructies SUB A en OUT (F8),A. Dit is gedurende 4 + 11 = 15 perioden, of 15 × 0.558 731 = 8.380 970 microsec.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E FF	P13A1	LD A,FF	
1802	D3 F8		OUT (F8),A	Onsteek leds.
1804	97		SUB A	4 perioden
1805	D3 F8		OUT (F8),A	11 perioden
1807	FF		RST 38	Monitor

Deze tijd is zo kort, dat het trage menselijke oog het oplichten niet waarneemt. Wie er niet van overtuigd is dat de leds even door P13A1 oplichten, kan het programma stap voor stap uitvoeren.
Om de oplichtingstijd te verlengen kunnen tussen de instructies SUB A en OUT (F8),A nog andere instructies geschoven worden. Van deze wordt geen prestatie verwacht. Ze zijn er alleen omdat de CPU enige tijd nodig heeft om ze uit te voeren.
Bij de keuze van deze instructies moet erop gelet worden, dat hun uitvoering nergens een nadelige invloed heeft (bv. INC A).
Een veilige tussenschuif-instructie is NOP, maar deze houdt de CPU ook maar vier perioden bezig.
Om de leds gedurende 1 msec (1/1000 sec) te laten oplichten, moet een vertraging verkregen worden van 1000/0.558731 = 1789.77 perioden. Daarvoor moeten er (1789 − 15)/4 = 443 NOP-instructies tussen SUB A en OUT (F8),A komen.

Een oplossing die minder geheugenplaats inneemt, en vlugger ingetoets is, wordt geboden door P13A2.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	06 00	P13A2	LD B,00	
1802	3E FF		LD A,FF	
1804	D3 F8		OUT (F8),A	
1806	97		SUB A	4
1807	10 FE	TIJD1	DJNZ TIJD1	255*13=3315 +8 = 3323
1809	D3 F8		OUT (F8),A	11
180B	FF		RST 38	----
				3338

De leds lichten hier op gedurende 3338 × 0.558731 = 1865 microsec, of 1,865 msec, wat al voor het oog waarneembaar is.

13.B. Berekening van een vertragingstijd

Om een oplichtingstijd van 1 msec te verkrijgen moet B met een kleinere waarde geladen worden.
Een msec komt overeen met 1000/0.558731 = 1790 perioden. Daarvan zijn er reeds 4 van SUB A, 11 van OUT (F8),A en 8 van DJNZ (als B=0) vast. De DJNZ (bij B>0) die 13 perioden duurt, moet de CPU nu nog 1790−(4+11+8) = 1767 perioden bezig houden. Daarvoor moet DJNZ (B>0) 1767/13 = 135.923 (of 135 met rest 12) keer uitgevoerd worden. Een afronding naar 136 is procentueel maar een heel kleine fout. Om DJNZ (bij B>0) 136 maal te laten uitvoeren moet B vooraf met 137 dec. of 87 Hex. geladen worden.
Om de leds juist 1 msec te laten oplichten, moet B met 136 dec. geladen worden. Het tekort van 12 perioden kan dan opgevuld worden door het inlassen van drie NOP-instructies.

14. DATA INPUT

14.A. Inleiding

In hoofdstuk 11 werd beschreven hoe data naar een randtoestel gebracht kunnen worden. Vaak moeten ook bepaalde toestanden van een randtoestel bij de Micro-Professor bediend zijn. De controle van een randtoestel is het eenvoudigst als deze zijn toestanden als digitale signalen aflevert. Zo kan een elektromotorisch bewogen afsluitkraan voorzien zijn van twee contacten. Het ene contact is ingedrukt als de kraan helemaal open staat, het andere als de kraan helemaal dicht is. Deze twee eindcontacten kunnen nu digitale signalen leveren die door de Z-80 kunnen worden ingevoerd. Met een inleesinstructie kan de Z-80 acht verschillende signalen binnenhalen.
Voor het controleren van analoge waarden (spanning, temperatuur, weerstand) moet eerst een analoog-naar-digitaal-convertor tussengeschakeld worden.
Voor proeven op het invoeren van data, is het randtoestel uit figuur 11.C.1 voorzien van acht schakelaars, die aan de Micro-Prof acht digitale signalen kunnen leveren.

14.B. IN A,(n)

Deze instructie brengt op de adreslijnen A7-A0 de waarde n (het poortadres), en maakt daarbij de uitgangen IORQ' en RD' laag. De waarde die op de databus staat wordt daarna in A geladen.
De opcode is DB, en moet gevolgd worden door het poortadres.

Voorbeeld : DB F8 IN A,(F8)

brengt de waarde F8 op de adreslijnen A7-A0, maakt de uitgangen IORQ en WR laag, en brengt de waarde die op de databus staat in A.
De flags worden door IN A,(n) niet beïnvloed.
Daar met een 8 bit adres gewerkt wordt, kunnen op de Z-80 256 verschillende input-poorten aangesloten worden, die elk afzonderlijk gelezen kunnen worden.

14.C. De input interface

De acht schakelaars mogen natuurlijk niet permanent op de databus aangesloten zijn, daar ze deze voortdurend zouden beïnvloeden.
De Z-80 en andere randtoestellen zouden daardoor de databus niet meer kunnen stellen.
Daarom zijn in de interface van figuur 11.C.1 de schakelaars via three-state-poorten met de databus verbonden. De three states brengen de signalen van de schakelaars op de databus over als het adres F8 op de adreslijnen A7-A0 staat, en tegelijkertijd de signalen IORQ' en WR' laag zijn. Deze toestand wordt verkregen door de instructie IN A,(F8). Het randtoestel reageert op deze instructie door de signalen van de schakelaars op de databus te brengen, die dan in A geladen worden.

14.D. Het testen van de input interface

Voordat met de studie van het binnenhalen van data begonnen kan worden, moet eerst de input interface getest worden.
Het eenvoudige testprogramma P14D1 geeft tevens een goed inzicht hoe de input interface functioneert, en hoe ermee gewerkt moet worden.
Het testprogramma leest met de eerste instructie de toestand van de schakelaars en brengt deze in A. De tweede instructie brengt de inhoud van A terug naar de leds. De laatste instructie vormt een lus, die zorgt dat de twee vorige instructies herhaald uitgevoerd worden.
Tijdens het lopen van het testprogramma moeten de leds de toestand van de schakelaars weergeven. Als de schakelaars van stand veranderen, moeten de leds dit volgen.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	DB F8	P14D1	IN A,(F8)	Laad schakelaars in A
1802	D3 F8		OUT (F8),A	A naar leds
1804	18 FA		JR P14D1	Sluit lus

14.E. Verwerking van input data

Data invoeren en zonder meer weer uitvoeren naar de leds heeft weinig zin. Data worden ingevoerd om gecontroleerd of verwerkt te worden. Eventueel kan er een data-uitvoer op volgen.
Stel dat van de binaire waarde die is voorgesteld door de schakelaars, de two's complement op de leds zichtbaar gemaakt moet worden.
Daarvoor moet eerst de binaire waarde van de schakelaars binnengehaald worden, door de CPU in de two's complement worden omgezet en daarna aan de leds worden doorgegeven.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	DB F8	P14E1	IN A,(F8)	Laad schakelaars in A
1802	ED44		NEG	2's complement
1804	D3 F8		OUT (F8),A	Naar leds
1806	18 F8		JR P14E1	Sluit lus

15. HET OPWEKKEN VAN GELUID

15.A. De 8255

Ook *PPI (Programmable Peripheral Interface)* genoemd ; is een programmeerbare IC voor het sturen van randtoestellen (sheet 2).
De 8255 heeft drie 8 bit poorten (PA7-PA, PB7-PB0, PC7-PC0), die als ingang én als uitgang ingesteld kunnen worden.
Bij de Micro-Professor wordt poort A (PA7-PA0) als 8 bit ingang gebruikt. PA5-PA0 voor het aftasten van het toetsenbord. Ingang PA6 controleert de user key, en via PA7 worden de signalen van de cassetterecorder binnengehaald.
De poorten B en C worden gebruikt om signalen naar buiten te brengen. De klemmen PB7-PB0 samen met PC5-PC0 sturen de led displays. Klem PC6 levert het signaal BREAK', dat later besproken zal worden, maar nu altijd hoog moet blijven.
Klem PC7 kan een signaal leveren aan de cassetterecorder. Als deze uitgang laag wordt, komt transistor Q2 in geleiding waardoor de groene led (TONE) gaat oplichten. Tegelijkertijd wordt ook de speaker geschakeld.
De 8255 wordt pas actief als zijn klem CS' laag is.
Bij de Micro-Professor is deze klem aangesloten op een uitgang van de 74LS139, die laag wordt als de adreslijnen A7 en A6 beide laag zijn met het signaal IORQ'.
Poort A komt in verbinding met de databus, als CS' laag is evenals de adreslijnen A1 en A0. Is CS' laag terwijl A1 laag en A0 hoog is, dan ontstaat de verbinding tussen poort B en de databus. Poort C verbindt zich met de databus als CS' en A0 laag zijn terwijl A1 hoog is.
Tenslotte blijft nog de mogelijkheid om CS' omlaag te brengen als A1 en A0 beide hoog zijn. In deze toestand kan een byte in het control register van de 8255 geschreven worden.
In tabel gebracht wordt dit :

A7	A6	A5	A4	A3	A2	A1	A0	Poortadres	Verbinding met:
0	0	X	X	X	X	0	0	00	Poort A
0	0	X	X	X	X	0	1	01	Poort B
0	0	X	X	X	X	1	0	02	Poort C
0	0	X	X	X	X	1	1	03	Control register

Het controlregister bepaalt welke poorten er als ingang of uitgang functioneren. Om poort A als ingang en de poorten B en C als uitgangen in te stellen moet naar het controlregister de waarde 90H gebracht worden.
Uiteraard moet dit gebeuren voordat een van de poorten benaderd wordt. Deze taak moet niet door de gebruiker uitgevoerd worden, daar de monitor dit werk doet bij het opstarten, en telkens als er op de toets RESET gedrukt wordt.
Dit is gemakkelijk te volgen in het monitor-programma vanaf adres 0004H. Daar is ook te zien dat de monitor vanaf adres 0008H de poortuitgangen PC7 en PC6 hoog maakt, zodat de groene led dooft, en het signaal BREAK' hoog wordt.
Voor het adresseren van de 8255 worden de adreslijnen A5-A2 niet gebruikt. Door deze onvolledige adressering zijn de poorten van de 8255 ook nog op andere adressen bereikbaar. Zo kan poort C ook nog benaderd worden op de adressen 00 0001 10 (06), 00 0010 10 (0A), 00 0011 10 (0E), 00 0100 10 (12), 00 0101 10 (16) enz.

Daarmee moet rekening gehouden worden bij het ontwerpen van een bijkomende interface zoals die in figuur 12.C.1.
De uitgangen van de 8255 (hier PB7-PB0 en PC7-PC0) zijn slechts belastbaar met 1mA, maar hebben wel geheugen-eigenschappen zoals de uitgangen in figuur 12.C.1.

15.B. Schakeling van de groene led

Om de groene led te laten oplichten, hoeft alleen PC7 laag gemaakt te worden. Hierbij moet de uitgang PC6, signaal BREAK', hoog blijven.

```
Adres   Mach. taal   Label   Mnemonic        Verklaring
======  ===========  ======  ==============  ========================
1800    3E 40        P15B1   LD A,40         Bit 7 laag, bit 6 hoog
1802    D3 02                OUT (02),A      Naar poort C
1804    76                   HALT
```

Het programma P15B1 wordt afgesloten met de instructie HALT, opdat de instructie RST 38 een deel van het monitorprogramma uitvoert, en de led weer dooft.
Het afsluiten met HALT doet ook de rode led oplichten. Voor het oog gelijktijdig met de groene led. Door het inlassen van een tijdsvertraging kan echter aangetoond worden, dat beide leds niet gelijktijdig oplichten.

```
Adres   Mach. taal   Label   Mnemonic        Verklaring
======  ===========  ======  ==============  ========================
1800    3E 40        P15B2   LD A,40
1802    D3 02        LEDSP   OUT (02),A
1804    01 0000              LD BC,0000      Waarde voor tijdsvertr.
1807    10 FE        TIJD    DJNZ            Wacht
1809    0D                   DEC C                 hier
180A    20 FB                JR NZ,TIJD                  even.
180C    76                   HALT
```

De groene led kan nu ook aan het knipperen worden gebracht door na het tijdsverloop de toestand van bit 7 uit A om te keren, en terug te keren naar label LEDSP (LED SPeaker).

```
Adres   Mach. taal   Label   Mnemonic        Verklaring
======  ===========  ======  ==============  ========================
1800    3E 40        P15B3   LD A,40
1802    D3 02        LEDSP   OUT (02),A
1804    01 0000              LD BC,0000
1807    10 FE        TIJD    DJNZ,TIJD       Wacht
1809    0D                   DEC C                 hier
180A    20 FB                JR NZ,TIJD                  even.
180C    EE 80                XOR 80          Keer enkel bit 7 om.
180E    18 F2                JR LEDSP        Herhaal
```

Het omkeren van bit 7 uit A gebeurt door XOR 80. Daarbij wordt alleen bit 7 omgekeerd, en blijft bit 6 altijd hoog.

15.C. Het opwekken van een toon

Bij het lopen van P15B3 is het omschakelen van de led ook telkens hoorbaar via de speaker. Om een toon op te wekken is het voldoende de frequentie op te drijven, wat bereikt wordt door de vertragingstijd te verkorten.
Daardoor worden meerdere instructies uit P15B3 overbodig, en het programma eenvoudiger.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 40	P15C1	LD A,40	
1802	D3 02	LEDSP	OUT (02),A	
1804	10 FE	TIJD	DJNZ,TIJD	Wacht even.
1806	EE 80		XOR 80	Bit 7 omkeren
1808	18 F8		JR LEDSP	Herhalen

De opgewekte toon kan verhoogd worden door B telkens vanaf een kleinere waarde te laten aftellen. De lagere startwaarde kan het eenvoudigst uit register C gehaald worden, die dan ook vooraf geladen moet worden.
Het programma P15C2 bevat dan ook maar twee instructies méér.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 40	P15C2	LD A,40	
1802	0E 50		LD C,50	Bepaalt de periode
1804	D3 02	LEDSP	OUT (02),A	
1806	41		LD B,C	
1807	10 FE	TIJD	DJNZ TIJD	Wacht even.
1809	EE 80		XOR 80	Keer bit 7 om.
180B	18 F7		JR LEDSP	Herhaal

De inhoud van lokatie 1803 bepaalt nu de toonhoogte ; hoe kleiner de waarde, hoe hoger de frequentie.

15.D. Toon met een bepaalde tijdsduur

Om een toon van een bepaalde tijdsduur te laten ontstaan, kan het aantal opgewekte pulsen geteld worden.
In programma P15D1 is de toonhoogte ook afhankelijk van de inhoud van C, en wordt het aantal perioden bepaald door de inhoud van HL.
De instructie ADD HL,HL verdubbelt de inhoud van HL. Dit is nodig, omdat voor iedere opgewekte periode de lus tweemaal doorlopen wordt, en HL ook tweemaal verlaagd wordt. Daardoor worden net zoveel perioden opgewekt als het aantal dat in HL vermeld staat.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	0E 80	P15D1	LD C,80	Duur halve periode
1802	21 0003		LD HL,0300	Aantal perioden
1805	29		ADD HL,HL	HL maal 2
1806	11 0100		LD DE,0001	Decrement voor HL
1809	3E 40		LD A,40	Enkel bit 6 hoog.
180B	D3 02	LEDSP	OUT (02),A	Naar speaker.

180D	41		LD B,C	Duur halve periode in B
180E	10 FE	TIJD	DJNZ TIJD	Wacht even.
1810	EE 80		XOR 80	Keer enkel bit 7 om.
1812	ED52		SBC HL,DE	Decrement HL
1814	20 F5		JR NZ,LEDSP	Herhaal tot HL=0
1816	76		HALT	

Een met P15D1 overeenkomende subroutine staat al in de monitor-EPROM geschreven, onder de naam TONE, vanaf adres 05E4.
Voordat deze subroutine opgeroepen wordt, moet C geladen zijn met een waarde die de duur van een halve periode zal bepalen, en HL met het aantal op te wekken perioden.
Hetzelfde resultaat als met P15D1 kan nu ook verkregen worden met :

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	0E 80	P15D2	LD C,80	Duur halve periode
1802	21 0003		LD HL,0300	Aantal perioden
1805	CD E405		CALL TONE	
1808	76		HALT	

De periodeduur die afhankelijk is van C, kan berekend worden met de volgende formule :

$T = 2 \times (44 + C \times 13) \times .56$ in microsec.

Register HL moet het aantal op te wekken perioden bevatten dat niet groter mag zijn dan 8000H.
Als een frequentie van 1 kHz moet worden opgewekt, kan een beroep gedaan worden op subroutine TONE1K die start op adres 05DE.
Voordat TONE1K opgeroepen wordt, hoeft alleen HL geladen te worden met het aantal op te wekken perioden (van 1kHz).

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	21 0010	P15D3	LD HL,1000	Aantal perioden
1803	CD DE05		CALL TONE1K	
1806	76		HALT	

Programma P15D3 wekt 1000 (4096 dec.) perioden op die elk een duur hebben van 1 msec. De toon is dan ook 4096 msec hoorbaar, of 4,096 sec.
Op adres 05E2 start de subroutine TONE2K, die een frequentie van 2 kHz voortbrengt. Door in P15D3 het adres van TONE2K te plaatsen zal een toon van 2 kHz opgewekt worden, maar nu gedurende 2.048 sec.

15.E. Meertonig geluid

Het welbekende geluid van een politiewagen bestaat uit twee tonen. Eén van ongeveer 265 Hz, de andere van rond de 352 Hz. Beide worden .75 sec aangehouden.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	0E 00	P15E1	LD C,0	Voor 265 Hz
1802	21 C000		LD HL,00C0	Dec. 192 perioden.
1805	CD E405		CALL TONE	Opwekken lage toon
1808	0E C0		LD C,C0	Voor 352 Hz
180A	21 0001		LD HL,0100	Dec. 256 perioden.
180D	CD E405		CALL TONE	Opwekken hoge toon
1810	18 EE		JR P15E1	Herhaal.

Voor meer dan twee tonen moet achtereenvolgens subroutine TONE opgeroepen worden, nadat telkens C en HL met de berekende waarden geladen zijn.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	0E 00	P15E2	LD C,0	256 decimaal
1802	21 C000		LD HL,00C0	Dec. 192 perioden
1805	CD E405		CALL TONE	Opwekken lage toon
1808	0E C0		LD C,C0	
180A	21 0001		LD HL,0100	
180D	CD E405		CALL TONE	Opwekken midden toon
1810	0E A0		LD C,A0	
1812	21 0002		LD HL,0200	
1815	CD E405		CALL TONE	Opwekken hoge toon
1818	18 E6		JR P15E2	Herhaal

15.F. Onderbroken toon

Wordt verkregen door een toon van een bepaalde duur op te wekken, en deze te laten volgen door een stille tijd. De stille tijd kan een gevolg zijn van een vertragingstijdroutine.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	0E A2	P15F1	LD C,A2	Instellen frequentie
1802	21 0001		LD HL,0100	Aantal perioden
1805	CD E405		CALL TONE	Opwekken toon
1808	01 0000		LD BC,0000	Instellen stille tijd
180B	10 FE	STIL	DJNZ STIL	Wacht
180D	0D		DEC C	hier
180E	20 FB		JR NZ,STIL	even.
1810	18 EE		JR P15F1	Herhaal

15.G. Het opwekken van een omschakelend signaal

Een omschakelend signaal is een meertonig signaal waarvan iedere frequentie zo kort aangehouden wordt, dat zij niet meer afzonderlijk waargenomen wordt.

Als voorbeeld worden acht perioden van 320 Hz ($8 \times 3.121 = 25$ msec) opgewekt, direct gevolgd door twaalf perioden van 480 Hz ($12 \times 2.087 = 25$ msec), waarna de cyclus van voren af aan begint.

(Opgelet voor het beginadres !)

Adres	Mach. taal	Label	Mnemonic	Verklaring
1803	0E D3	P15G1	LD C,D3	F=320 Hz , T=3.121 msec
1805	21 0800		LD HL,0008	8*3.121 = 25 msec.
1808	CD E405		CALL TONE	
180B	0E 8C		LD C,8C	F=480 Hz , T=2.087 msec
180D	21 0C00		LD HL,000C	12*2.087 = 25 msec.
1810	CD E405		CALL TONE	
1813	18 EB		JR P15G1	Herhaal

Om zo'n signaal slechts gedurende een bepaalde tijd op te wekken kan geteld worden, hoeveel maal de cyclus doorlopen wordt.
In P15G1 duurt één cyclus 50 msec. Als na twintig cyclussen het programma onderbroken wordt, zal het signaal 1 sec hoorbaar zijn.
Voor het tellen van het aantal cyclussen kan A gebruikt worden. Daar ook de subroutine TONE register A benut, moet A voor de cyclus in A' geplaatst worden, en teruggehaald worden na iedere cyclus.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 14	P15G2	LD A,14	20 maal
1802	08	BEL	EX AF,AF'	Bewaar A in A'
1803	0E D3		LD C,D3	
1805	21 0800		LD HL,0008	
1808	CD E405		CALL TONE	
180B	0E 8C		LD C,8C	
180D	21 0C00		LD HL,000C	
1810	CD E405		CALL TONE	
1813	08		EX AF,AF'	Haal A terug
1814	3D		DEC A	
1815	20 EB		JR NZ,BEL	Herhaal tot A=0
1817	76		HALT	

Misschien herkent u dit signaal wel als het belgeluid van een elektronische telefoon. Deze herhaalt het signaal echter met een stille tussentijd van 2 sec. Ook dit is te bereiken door een stille tijd van 2 sec in het programma op te nemen.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 14	P15G3	LD A,14	
1802	08		EX AF,AF'	
1815	20 EB		JR NZ,BEL	
1817	01 50C3		LD BC,C350	Voor 2 sec.
181A	E3		EX (SP),HL	Dummy
181B	E3		EX (SP),HL	Dummy
181C	EDA1		CPI	BC-1
181E	EA 1A18		JP PE,TIJD	Herhaal tot BC=0
1821	18 DD		JR P15G3	Herbegin

16. DE DISPLAY

16.A. Een enkele display sturen

De display van de Micro-Professor werkt volgens het multiplexed systeem.
Een schema aan de hand waarvan de werking van de display gevolgd kan worden, is weergegeven in figuur 16.A.1. Van iedere digit zijn alleen de segmenten a en b getekend. De overige segmenten zijn op dezelfde manier aangesloten op de uitgangen PB0-PB2 en PB5-PB7. De gemeenschappelijke katode van iedere digit is verbonden via een driver (75492), die zorgt voor de nodige stroom, aan de uitgangen PC0-PC5.
Om alleen het segment a van de rechter-digit (digit 0) te doen oplichten, moeten PB3 en PC0 beide hoog zijn. Daarbij mag niet vergeten worden dat PC6 en PC7 altijd hoog moeten blijven. Daarom moet naar poort 01 (poort B of SEG7) de waarde 08 gebracht worden, en naar poort 02 (poort C of DIGIT) de waarde C1.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 08	P16A1	LD A,08	Segment a
1802	D3 01		OUT (SEG7),A	Maak a hoog
1804	3E C1		LD A,C1	Kies digit 0
1806	D3 02		OUT (DIGIT),A	Maak K0 laag.
1808	76		HALT	

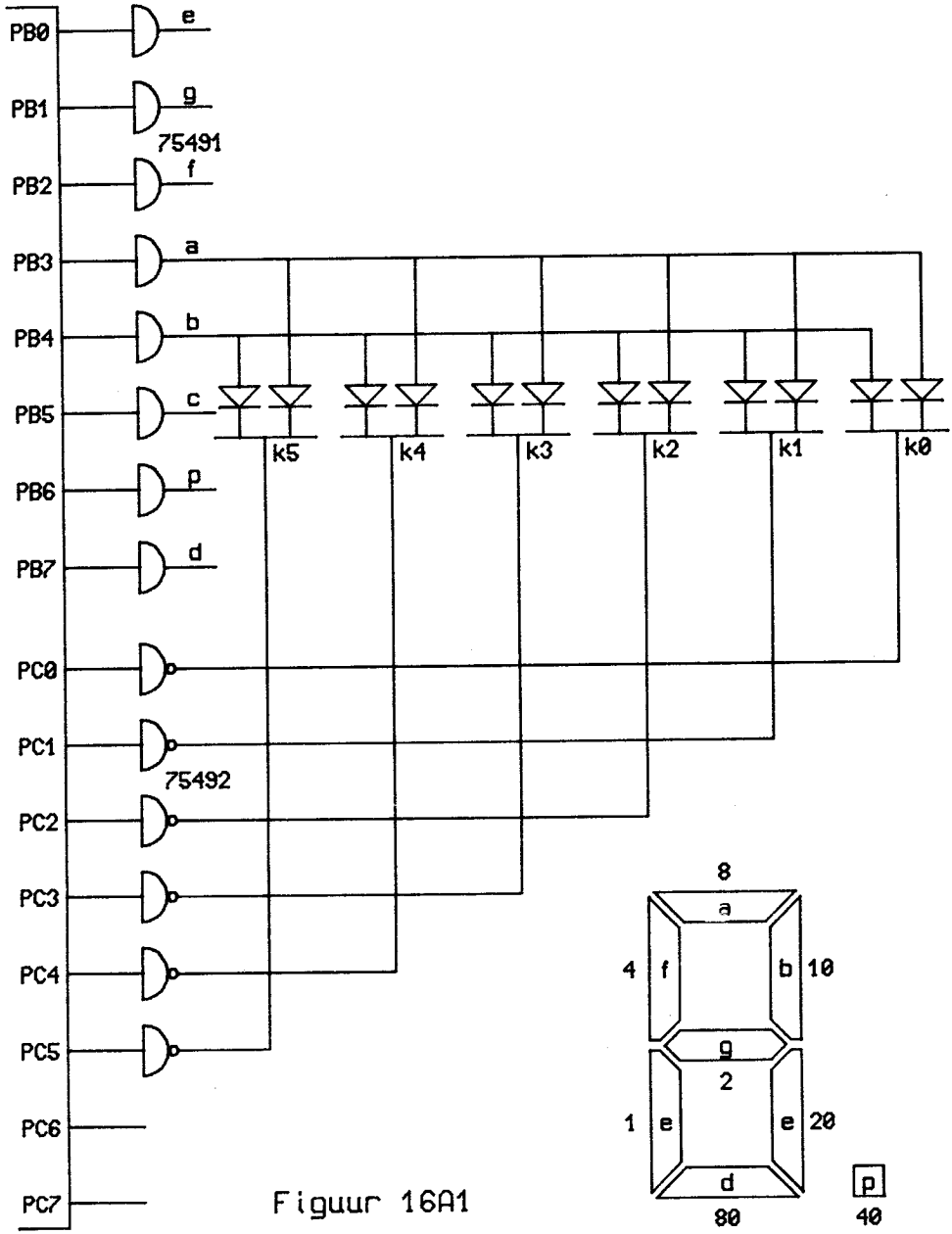
Bij de instructie HALT stopt de Z-80. Door de geheugen-eigenschappen van de 8255 blijven PB3 en PC0 hoog. Het segment a van digit 0 blijft branden.
Om cijfer 7 op digit 5 te krijgen, moeten PB3, PB4, PB5 en PC5 alle hoog zijn. Naar poort SEG7 moet daarvoor de waarde 8 + 10 + 20 = 38 gebracht worden, en naar poort DIGIT de waarde 80 + 40 + 20 = E0.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 38	P16A2	LD A,38	Patroon 7
1802	D3 01		OUT (SEG7),A	Maak a, b en c hoog
1804	3E E0		LD A,E0	Digit 5
1806	D3 02		OUT (DIGIT),A	Maak k5 laag
1808	76		HALT	

In figuur 16.A.1 zijn bij de segmenten ook waarden genoteerd. Door de som te maken van de segmenten die moeten oplichten, wordt het patroon voor het karakter verkregen.

Voorbeelden :

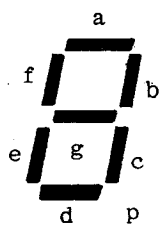
Het patroon voor karakter 2 is : 8 + 10 + 2 + 1 + 80 = 9B
H is : 4 + 1 + 2 + 10 + 20 = 37
A. is : 1 + 4 + 8 + 10 + 20 + 2 + 40 = 7F


PB0
PB1
PB2
PB3
PB4
PB5
PB6
PB7
PC0
PC1
PC2
PC3
PC4
PC5
PC6
PC7
e
g
75491
f
a
b
c
p
d
75492
k5
k4
k3
k2
k1
k0
8
a
4
f
b
10
g
2
1
e
e
20
d
80
p
40

Figuur 16A1

Figuur 16.A.2 toont hoe alle karakters weergegeven kunnen worden met een 7 segment display, en de bijbehorende patronen.
Programma P16A3 brengt, met tussenschakeling van een tijdsvertraging, alle patronen van 00 tot FF naar de segmentpoort.
Daardoor worden al de mogelijke karakters met een 7 segment display getoond. Naar de digitpoort wordt de waarde C1 gebracht, waardoor PC0 hoog blijft, en digit 0 wordt geselecteerd.

A. Display format, position-code and internal-code



7	6	5	4	3	2	1	0
d	p	c	b	a	f	g	e

DISPLAY FORMAT :

CODE	BD	30	9B	BA	36	AE	AF	38	BF	BE	3F	A7	8D	B3
DATA	0	1	2	3	4	5	6	7	8	9	A	B	C	D
DISP	0	1	2	3	4	5	6	7	8	9	A	b	C	d
CODE	8F	0F	AD	37	89	B1	97	85	2B	23	A3	1F	3E	03
DATA	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
DISP	E	F	G	H	I	J	K	L	M	n	o	P	q	r
CODE	A6	87	B5	B7	A9	07	B6	8A	83	A2	32	02	C0	00
DATA	S	T	U	V	W	X	Y	Z	(	)	+	-	,	
DISP	S	T	U	V	W	X	Y	Z	(	)	+	-	,	

Fig 16.A.2

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E C1	P16A3	LD A,C1	Kies digit 0
1802	D3 02		OUT (DIGIT),A	Maak k0 laag
1804	97		SUB A	Stel A op nul
1805	D3 01	LUS	OUT (SEG7),A	Naar segmenten
1807	3C		INC A	
1808	10 FE	TIJD	DJNZ TIJD	Wacht
180A	0D		DEC C	hier
180B	20 FB		JR NZ,TIJD	even.
180D	18 F6		JR LUS	Volgend patroon

16.B. Alle digits samen sturen

Als het patroon 38 naar de segmentpoort is gebracht, en C1 naar de digit-poort, staat het karakter 7 op digit 0. Door de instructie die C2 naar de digit-poort brengt verspringt het karakter 7 van digit 0 naar digit 1.
Een programma dat achtereenvolgens de waarden C1 (1100 0001), C2 (1100 0010), C4 (1100 0100, C8 (1100 1000), D0 (1101 0000) en E0 (1110 0000) naar de digit-poort brengt, doet hetzelfde karakter beurtelings op de zes digits verschijnen.
Programma P16B1 doet dit met een vaste en grote tijdsvertraging.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 38	P16B1	LD A,38	Patroon voor 7
1802	D3 01		OUT (SEG7),A	Naar segmenten
1804	3E C1	NUL	LD A,C1	Kies digit nul
1806	D3 02	LUS	OUT (DIGIT),A	Maak gem.katode laag.
1808	0E 00		LD C,00	Grootste vertraging
180A	10 FE	TIJD	DJNZ TIJD	Wacht
180C	0D		DEC C	hier
180D	20 FB		JR NZ,TIJD	even.
180F	E6 3F		AND 3F	Maak bit 7 en 6 laag
1811	87		ADD A,A	Maak volgende bit hoog
1812	FE 40		CP A,40	Bit 6 hoog ?
1814	28 EE		JR Z,NUL	Herbegin dan bij NUL
1816	F6 C0		OR C0	Maak b6 en b7 weer hoog
1818	18 EC		JR LUS	Volgende digit

De tijd dat een karakter op dezelfde digit blijft staan, is afhankelijk van de inhoud van lokatie 1809. Door de inhoud van deze lokatie in meerdere étappen te verkleinen, kan het principe van multiplexed display gedemonstreerd worden.

Taak 1. Voer P16B1 meerdere malen uit, maar telkens met de inhoud van 1809 de helft kleiner.

Om iedere digit een ander karakter te laten tonen, moet met het omschakelen van digit ook een ander patroon naar de segmenten gebracht worden. Daarvoor moeten de patronen van de te tonen karakters ergens in het geheugen ter beschikking staan, zodat ze op het juiste moment naar de segmenten gebracht kunnen worden.

Zoiets gebeurt in P16B2, de patronen staan in de lokaties 1820 tot 1825, en worden door aanwijzing van HL op het juiste ogenblik opgehaald, en naar de segmenten gebracht.
Een vertragingsroutine laat het omschakelen langzaam gebeuren, zodat de werking gevolgd kan worden.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	21 2018	P16B2	LD HL,1820	Wijs tabel patronen aan
1803	16 06		LD D,06	Zes digits aftellen
1805	1E C1		LD E,C1	Begin met digit 0
1807	7E	LUS	LD A,(HL)	Haal patroon
1808	D3 01		OUT (SEG7),A	Naar segmenten
180A	7B		LD A,E	Haal nummer digit
180B	D3 02		OUT (DIGIT),A	Maak een katode laag
180D	0E 00		LD C,00	Grootste vertraging
180F	10 FE	TIJD	DJNZ TIJD	Wacht
1811	0D		DEC C	hier
1812	20 FB		JR NZ,TIJD	even.
1814	E6 3F		AND 3F	Maak bit 7 en 6 laag
1816	87		ADD A	Maak volgende bit hoog
1817	F6 C0		OR C0	Maak 7 en 6 weer hoog
1819	5F		LD E,A	Bewaar nieuw nummer dig
181A	23		INC HL	Wijs volgend patroon
181B	15		DEC D	Tel digit af
181C	20 E9		JR NZ,LUS	Terug voor volgende dig
181E	18 E0		JR P16B2	Herbegin
1820	BD			Tabel
1821	30			met
1822	9B			patronen.
1823	BA			
1824	36			
1825	AE			

Om alle karakters samen te zien moet alleen de snelheid opgevoerd worden. Dit doel kan ook bereikt worden door de volgende instructies uit het P16B2 te verwijderen : LD C,00 DEC C en JR NZ,TIJD.
Let erop, dat in de tabel de patronen in omgekeerde volgorde moeten staan.
In de EPROM staat vanaf adres 07F0 een tabel (SEGTAB) met de patronen voor de karakters 0 tot F. Door in P16B2 de lokaties 1801 en 1802 te laden met F0 en 07 haalt het programma de patronen uit deze tabel.
Ook de patronen van berichten die door de Micro-Professor op de display gebracht moeten worden, staan in de EPROM (079F-07FF), en kunnen met P16B2 op de display gebracht worden. In deze tabel verdient het deel BLANK aandacht ; daarmee kan de display gewist worden.

Taak 2. Laat uw naam, of een deel ervan, op de display verschijnen.

16.C. Subroutine SCAN1

Net als P16B2 brengt de subroutine SCAN1 de aangewezen patronen naar de display. Wel moet bij gebruik van SCAN1 de plaats waar de patronen staan, aangewezen worden door

register IX. Ook hier moet IX wijzen naar de lokatie waar het patroon voor de rechter-digit staat, en moet in lokatie X+5 het patroon voor digit 5 staan. Subroutine SCAN1 begint op adres 0624. De subroutine SCAN1 brengt slechts eenmaal de zes patronen naar de digits, wat ongeveer 10 msec (9.97 msec) duurt. Om de karakters voortdurend zichtbaar te houden, moet SCAN1 dan ook herhaald opgeroepen worden.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	DD21 2018	P16C1	LD IX,1820	Tabel patronen aanwijz.
1804	CD 2406	LUS	CALL SCAN1	Patronen naar digits
1807	18 FB		JR LUS	Herhalen
1820	30			Tabel
1821	00			met
1822	23			patronen.
1823	3F			
1824	8D			
1825	AE			

De subroutine SCAN1 tast tegelijkertijd ook het toetsenbord af.
Hoe het toetsenbord gecontroleerd wordt, zal afzonderlijk bestudeerd worden. Daarom wordt hier aan deze eigenschap van SCAN1 geen nadere aandacht besteed.
Om de tekst slechts 1 sec op de display te houden, moet SCAN1 honderd maal (10 msec × 100 = 1 sec) opgeroepen worden. Register B is bijzonder geschikt om deze telling bij te houden.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	DD21 2018	P16C2	LD IX,1820	Tabel patronen aanwijz.
1804	06 64		LD B,64	Honderd maal.
1806	CD 2406	LUS	CALL SCAN1	Patronen naar digits
1809	10 FB		DJNZ	Aftellen
180B	76		HALT	

Nu is het ook niet moeilijk meer om de tekst met een frequentie van 1 Hz te laten knipperen (0.5 sec aan, 0.5 sec uit).

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	DD21 2018	P16C3	LD IX,TEKST	Adres tabel TEKST
1804	06 32		LD B,32	50 maal
1806	CD 2406	LUS1	CALL SCAN1	Tekst naar digits
1809	10 FB		DJNZ LUS1	Aftellen en herhalen
180B	DD21 A507		LD IX,BLANK	Adres tabel BLANK
180F	06 32		LD B,32	50 maal of 0.5 sec
1811	CD 2406	LUS2	CALL SCAN1	Blank digits
1814	10 FB		DJNZ LUS2	Aftellen en herhalen
1816	18 E8		JR P16C3	Herbegin
1820	30			Tabel
1821	00			met
1822	23			patronen.

Hier wordt gebruik gemaakt van de patronentabel BLANK uit de EPROM om de digits te wissen.

Taak. Na 10 sec moet het programma overschakelen en alleen een alarmtoon opwekken.

Het programma P16C3 bestaat uit twee helften die alleen verschillen in de lading van IX. Als de inhoud van IX met een ander adres geruild kan worden, is de helft van het programma voldoende. De eenvoudigste manier daarvoor is de inhoud van IX ruilen met de top van de stack. Wel moet dan eerst het adres van BLANK op de top van de stack geplaatst worden.

ADRES	Mach. taal	Label	Mnemonic	Verklaring
1800	21 A507	P16C4	LD HL,BLANK	Adres tabel BLANK
1803	E5		PUSH HL	Naar top van de stack
1804	DD21 2018		LD IX,TEKST	Adres tabel TEKST
1808	DD E3	RUIL	EX (SP),IX	Adressen ruilen
180A	06 32		LD B,32	50 maal, 0.5 sec
180C	CD 2406	LUS	CALL SCAN1	Patronen naar digits
180F	10 FB		DJNZ LUS	Aftellen en herhalen
1811	18 F5		JR RUIL	Omschakelen

Als dit programma nog verder uitgebreid wordt, mag niet vergeten worden dat er al een waarde op de stack staat, en dat deze er afgehaald moet worden als dit programmadeel niet meer gebruikt wordt.
Ook mag niets anders op de stack geplaatst worden zolang dit programmadeel in gebruik is.

16.D. Subroutine SCAN

De subroutine SCAN1 brengt slechts eenmaal de patronen naar de digits, waardoor SCAN1 altijd in een lus moet lopen.
Subroutine SCAN (adres 05FE) blijft voortdurend de patronen naar de digits brengen, tot op een toets wordt gedrukt. Daarom moet SCAN niet in een lus opgenomen worden. Ook bij SCAN moet IX wijzen naar de lokatie van het patroon voor digit 0 staat.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	DD21 2018	P16D1	LD IX,TEKST	Adres tabel TEKST
1804	CD FE 05		CALL SCAN	Herhalend naar digits
1807	76		HALT	tot een toets ingedrukt
1820	00			Tabel
1821	23			TEKST
1822	3F			
1823	8D			
1824	AE			
1825	00			

De verschillen tussen SCAN1 en SCAN zijn het grootst bij het aftasten van het toetsenbord. Ze worden daarom in een volgend hoofdstuk verder besproken.

Taak. Enkele toetsen onderbreken de subroutine SCAN niet. Welke, en waarom ?

16.E. Van vier bits naar één patroon

Stel dat twee waarden, bijvoorbeeld 3 en 4, bij elkaar opgeteld moeten worden, en het resultaat op de display moet verschijnen.
Het maken van de som is uiteraard geen probleem meer, maar het resultaat 7 kan niet zomaar naar de display gebracht woren.
De subroutine SCAN moet het patroon van het resultaat hebben om de som zichtbaar te maken. Daarvoor moet de som die als vier bits in A staat, omgevormd worden tot een patroon.
Ter vereenvoudiging wordt hier aangenomen dat de som nooit groter zal zijn dan F, zodat maar één patroon geleverd moet worden.
In de tabel SEGTAB staan vanaf adres 07F0 de patronen van de karakters 0 tot F in volgorde. Zo is het patroon van 7 te vinden op adres 07F0 + 7 = 07F7.
Door op dezelfde manier het adres te berekenen waar het overeenkomstige patroon staat, kan het juiste patroon daar opgehaald, en naar een buffer (DISBF) overgebracht worden.
Als IX nu ook nog naar de DISBF wijst, moet alleen SCAN opgeroepen worden om het resultaat zichtbaar te maken.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 03	P16E1	LD A,03	03 in A
1802	C6 04		ADD A,04	07 in A
1804	C6 F0		ADD A,F0	F7 in A
1806	6F		LD L,A	F7 in L
1807	26 07		LD H,07	07F7 in HL
1809	7E		LD A,(HL)	Patroon in A
180A	DD21 3018		LD IX,(DISBF0)	IX wijst naar DISBF0
180E	DD77 00		LD (IX),A	Patroon in DISBF0
1811	CD FE05		CALL SCAN	Naar digit 0
1814	76		HALT	
1830	00	DISBF0		
1831	00	DISBF1		
1832	00	DISBF2		
1833	00	DISBF3		
1834	00	DISBF4		
1835	00	DISBF5		

Taak. Pas het programma aan, zodat van een som die groter is dan F toch de juiste waarde van de vier laagste bits op de display komt.

16.F. Subroutine HEX7

Vormt de waarde van de vier laagste bits uit A om in het overeenkomstige patroon. Subroutine HEX7 begint op adres 0689, en kan gebruikt worden om hetzelfde resultaat als van P16E1 te verkrijgen.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 05	P16F1	LD A,05	05 in A
1802	C6 07		ADD A,07	Som in A
1804	CD 8906		CALL HEX7	Patroon in A
1807	DD21 3018		LD IX,DISBF0	Adres DISBF0 in X
180B	DD77 00		LD (IX),A	Patroon in BISBF0
180E	CD FE05		CALL SCAN	Som naar digit 0
1811	76		HALT	
1830	00	DISBF0		
1831	00	DISBF1		
1835	00	DISBF5		

Subroutine HEX7 levert alleen het patroon van de waarde die in de vier laagste bits staat, en houdt geen rekening met wat in de vier andere bits staat.

16.G. Van één byte naar twee patronen

Om de volledige inhoud van A op de display te krijgen, moeten ook de vier hoogste bits tot een patroon omgevormd worden.
Eerst moet weer het patroon van de vier laagste bits met HEX7 gevormd worden, en in DISBF0 worden geplaatst.
Voor het verkrijgen van het patroon van de vier hoogste bits, moet de originele inhoud van A, vier bits naar rechts geschoven worden.
Met HEX7 kan dan ook weer het patroon verkregen worden, dat in dit geval naar DISBF1 gebracht moet worden.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 12	P16G1	LD A,12	12 in A
1802	C6 25		ADD A,25	37 in A
1804	21 3018		LD HL,DISBF0	HL wijst naar DISBF0
1807	F5		PUSH AF	Bewaar AF
1808	CD 8906		CALL HEX7	Patroon van 7 in A
180B	77		LD (HL),A	Patroon naar DISBF0
180C	23		INC HL	Wijs naar DISBF1
180D	F1		POP A	37 in A
180E	0F		RRCA	
180F	0F		RRCA	
1810	0F		RRCA	
1811	0F		RRCA	73 in A
1812	CD 8906		CALL HEX7	Patroon van 3 in A
1815	77		LD (HL),A	Patroon naar DISBF1
1816	DD21 3018		LD IX,1830	IX wijst naar DISBF0
181A	CD FE05		CALL SCAN	Patronen naar displ.
181D	76		HALT	

1830	00	DISBF0		
1835	00	DISBF5		

16.H. Subroutine HEX7SG

Begint op adres 0678, en plaatst van de waarde die in A staat de twee overeenkomstige patronen in DISBF0 en BISBF1. Voordat HEX7SG opgeroepen wordt moet HL geladen zijn met DISBF0. Na HEX7SG is HL met 2 verhoogd, en wijst dan naar de volgende vrije DISBF. Het resultaat van een bewerking kan nu door gebruik van HEX7SG gemakkelijker op de display gekregen worden.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 98	P16H1	LD A,98	98 in A
1802	D6 53		SUB A,53	45 in A
1804	21 3018		LD HL,DISBF0	HL wijst naar DISBF0
1807	CD 7806		CALL HEX7SG	Patronen naar DISBF01
180A	DD21 3018		LD IX,DISBF0	IX wijst naar DISBF0
180E	CD FE05		CALL SCAN	Patronen naar digits
1811	76		HALT	
1830	00	DISBF0		

Daar HL met 2 verhoogd wordt, zullen bij een tweede maal oproepen van HEX7SG de patronen in DISBF2 en DISBF3 geplaatst worden.
Daardoor kan nu ook de inhoud van een registerpaar op de display gebracht worden.
In P16H2 wordt daarvan gebruik gemaakt door de som van twee double-precision-waarden (16 bits) op de display te tonen.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	21 3412	P16H2	LD HL,1234	Opteltal
1803	01 4523		LD BC, 2345	Opteller
1806	09		ADD HL,BC	Som in HL
1807	EB		EX HL,DE	Som in DE
1808	21 3018		LD HL,DISBF0	HL wijst naar DISBF0
180B	7B		LD A,E	LSB van som in A
180C	CD 7806		CALL HEX7SG	Patronen in DISBF0-1
180F	7A		LD A,D	MSB van som in A
1810	CD 7806		CALL HEX7SG	Patronen in DISBF2-3
1813	DD21 3018		LD IX,DISBF0	IX wijst naar DISBF0
1817	CD FE 05		CALL SCAN	Patronen naar digits
181A	76		HALT	
1830	00	DISBF0		

Taak 1. De som van twee double-precision-waarden kan ook uit vijf digits bestaan. Pas het vorige programma aan, zodat ook de vijfde digit zichtbaar wordt.

Taak 2. Bereken de som van 123321 en 321123, en laat het resultaat op de display verschijnen.

17. HET TOETSENBORD

17.A. Inleiding

Een toetsenbord kan ook gebouwd worden volgens de principes van figuur 11.C.1. Iedere toets moet dan een afzonderlijke input-lijn van een poort krijgen. Om te controleren of een toets ingedrukt is, zou een eenvoudig programma voldoende zijn. Wel zouden er dan ook 32 input-lijnen nodig zijn om een toetsenbord als dat van de Micro-Professor aan te sluiten.
Dit is een veel voorkomende toestand. Eenvoudige software gaat vaak samen met uitgebreide en kostbare hardware.
Dikwijls is er ook een oplossing in de andere richting. Eenvoudige goedkope hardware, die wel door uitgebreide software moet worden gestuurd.
Daar het systeem in meerdere exemplaren gebouwd zal worden, worden de kosten voor het ontwikkelen van het programma snel gecompenseerd door de goedkope hardware.
Daarom zijn de meeste toetsenborden aangesloten in matrix-vorm.
Daarbij staat iedere toets (maakcontact) op een kruispunt van de matrix.
Bij de Micro-Professor bestaat de matrix uit zes kolommen (verticaal) en zes lijnen (horizontaal), zodat er plaats is voor 36 toetsen (zie sheet 2). Vier plaatsen op de matrix zijn niet gebruikt.
De lijnen van de matrix zijn aangesloten op de klemmen PA0-PA5 van de 8255, die ingesteld worden als ingangen. De kolommen worden gestuurd door de als uitgang ingestelde klemmen PC0-PC5, die ook al gebruikt worden voor de display.
De hardware is hier dus heel beperkt, slechts zes ingangslijnen zijn nodig. Ook het toetsenbord zelf is eenvoudiger, slechts twaalf aansluitklemmen voor 32 (mogelijk 36) toetsen.
Een programma dat controleert of een toets op de matrix aangesloten is, en welke, is minder eenvoudig. Om de werking goed begrijpelijk te maken zal deze weer in een later stadium verklaard worden.

17.B. Het aftasten van de rechterkolom

Hier zal alleen gecontroleerd worden, of een van de toetsen van de rechterkolom (toetsen 3, 7, B en F) gesloten is. Alleen de hardware die getekend is in figuur 17.B.1 is hierbij belangrijk. Alhoewel op de rechterkolom (kolom 0) slechts vier toetsen aangesloten zijn, controleert P17B1 toch de zes kruispunten. Later is het principe dan ook bruikbaar voor de overige kolommen.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	0E 00	P17B1	LD C,00	Toetsteller.
1802	3E FE		LD A,FE	1111 1110
1804	D3 02		OUT (DIGIT),A	Maak kolom 0 laag.
1806	06 06		LD B,06	6 lijnen
1808	DB 00		IN A,(KIN)	Lees lijnen (PA0-PA7).
180A	1F	LUS	RRA	Schuif bit 0 in carry.
180B	30 05		JR NC,STOP	STOP als carry=0
180D	0C		INC C	Verhoog toetsenteller.
180E	10 FA		DJNZ LUS	Terug voor de 6 bits.
1810	18 EE		JR P17B1	Herbegin.
1812	F7	STOP	RST 30	Naar monitor.

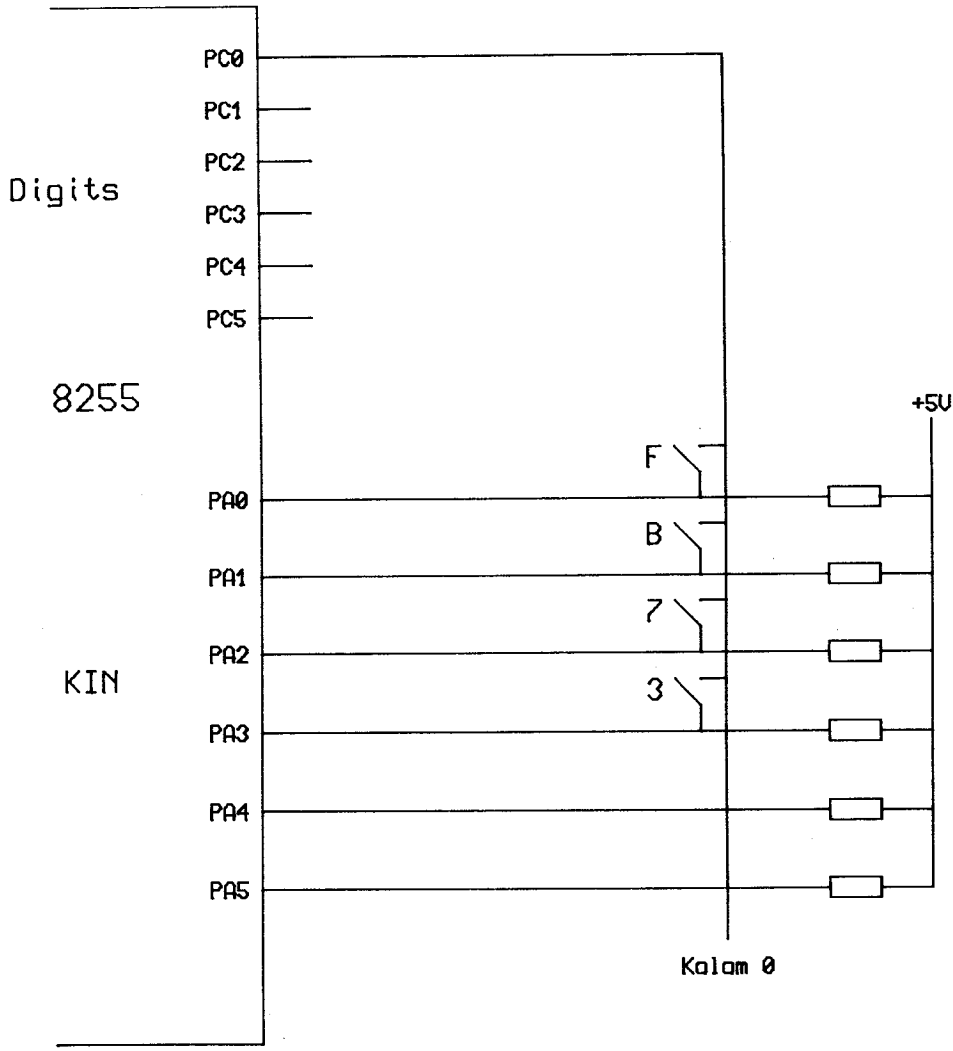


Fig. 17B1

Eerst wordt de toetsenteller op 00 gesteld, en kolom 0 laag gemaakt. Nadat B geladen is met 06 worden de lijnen gelezen.
Als toets B ingedrukt is, komt daardoor in A de waarde BF (1111 1011). Door RRA wordt A = X111 1101 en de carry 1. Er wordt niet overgegaan naar STOP.
Nu wordt de toetsenteller C met 1 verhoogd, wat aanduidt dat reeds 1 toets gecontroleerd is. Door de DJNZ wordt overgegaan naar RRA, waarna A = 1X11 1110 en carry = 1. De toetsenteller wordt daardoor weer verhoogd met 1 tot 02, waarna weer naar RRA wordt teruggekeerd. Deze maakt A = 11X1 1111 en carry = 0. De JR NC,STOP reageert nu wel, zodat overgegaan wordt naar STOP, waar de RST 30 wordt uitgevoerd.
ReSTart 30 betekent eveneens terugkeer naar de monitor, maar zonder dat de inhoud van de

registers gewijzigd wordt. Daardoor kan met de toetsen REG en BC nagegaan worden, welke waarde in register C staat. Als toets B was ingedrukt moet daar de waarde 02 staan.
Naargelang welke toets van kolom 0 werd ingedrukt, zal in register C een van de volgende waarden staan, die de *position-code* van de toets wordt genoemd.

Toets	Position-code
3	00
7	01
B	02
F	03

Moesten ook op de onderste twee kruispunten van kolom 0 contacten staan, dan zouden deze de position-code 04 en 05 in register C brengen.
Als er geen toets van kolom 0 is ingedrukt zal, nadat de zes rechter-bits uit A in de carry zijn gecontroleerd, door DJNZ het programma hervat worden. Dit, tot een van de toetsen 3, 7, B of F wordt ingedrukt.

17.C. Het aftasten van alle kolommen

Voor het aftasten van het volledige toetsenbord (sheet 2), kan begonnen worden met het controleren van de toetsen op kolom 0. Als daar geen gesloten contact gevonden wordt, moet kolom 1 laag gemaakt worden, waarna op dezelfde manier nagegaan wordt of daar een contact gesloten is. Daarbij wordt telkens ook weer een toetsenteller met 1 verhoogd.
Als in kolom 1 een gesloten contact gevonden wordt, keert het programma eveneens terug naar de monitor. Daar moet dan in register C een van de waarden 06, 07, 08 of 09 gevonden worden. De waarden 0A en 0B kunnen ook hier niet voorkomen, omdat er weer slechts vier contacten aanwezig zijn.
Indien er geen gesloten contact gevonden wordt in kolom 1, moet kolom 2 laag gemaakt worden, en op gesloten contacten worden gecontroleerd. Indien de test negatief is, moeten de kolommen 3, 4 en 5 op dezelfde manier afgetast worden.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	0E 00	P17C1	LD C,0	Toetsteller.
1802	1E FE		LD E,FE	1111 1110
1804	26 06		LD H,06	Aantal kolommen.
1806	7B	KOL	LD A,E	Maak lopende
1807	D3 02		OUT (DIGIT),A	kolom laag.
1809	06 06		LD B,06	6 lijnen.
180B	DB 00		IN A,(KIN)	Lees lijnen.
180D	1F	LUS	RRA	Schuif bit 0 in carry.
180E	30 0A		JR NC,STOP	STOP als carry = 0.
1810	0C		INC C	Verhoog toetsteller.
1811	10 FA		DJNZ LUS	6 bits aftellen.
1813	CB 03		RLC E	Maak volgende bit laag
1815	25		DEC H	Tel aantal kolommen af
1816	20 EE		JR NZ,KOL	Volgende kolom.
1818	18 E6		JR P17C1	Herbegin.
181A	F7	STOP	RST 30	Naar monitor.

Bij het starten van P17C1 moet de toets GO zo kort mogelijk aangeslagen worden. Anders vindt P17C1 deze toets gesloten, en keert hij terug naar de monitor.
Nieuw ten opzichte van P17B1 is het gebruik van register H, dat zorgt dat na het controleren van kolom 5 het programma opnieuw begint.
In het begin van het programma wordt van register E alleen bit 0 laag gemaakt, die bepaalt dat met kolom 0 begonnen zal worden.
Telkens als een kolom gecontroleerd is, wordt door RLC E de lage bit één plaats naar links geschoven, waarmee de volgende kolom aangewezen wordt.
Met P17C1 kan nu de position-code van iedere toets in register C gebracht worden. De position-code van alle toetsen is te vinden in 17.D.

17.D. Omvorming van de position-code

De position-code van een hexatoets stemt niet overeen met de waarde van de toets zelf. In de EPROM staat vanaf 077B een tabel KEYTAB, met de waarden van de hexatoetsen in volgorde van hun position-code. Op basis van KEYTAB kan P17C1 uitgebreid worden om de position-code om te zetten in de waarde van de hexatoets. In KEYTAB staan ook waarden voor de niet-hexatoetsen, zodat ook hun position-code kan worden omgevormd.
De waarde die wordt verkregen na de omvorming wordt de *key internal code* genoemd, en is na uitvoering van het uitgebreide programma te vinden in register A.

Adres	Mach. taal	Label	Mnemonic	Verklaring
181A	21 7B07	STOP	LD HL,KEYTAB	Uitbreiding P17C1
181D	79		LD A,C	Position-code in A
181E	85		ADD A,L	Verhoog register L met
181F	6F		LD L,A	de position-code.
1820	7E		LD A,(HL)	Internal-key-code in A
1821	F7		RST 30	Naar monitor.

Toets	Position-code	Internal-code
0	12	00
1	0C	01
2	06	02
3	00	03
4	13	04
5	0D	05
6	07	06
7	01	07
8	14	08
9	0E	09
A	08	0A
B	02	0B
C	15	0C
D	0F	0D
E	09	0E
F	03	0F
+	21	10
-	1F	11
GO	16	12

STEP	10	13
DATA	20	14
SBR	1E	15
INS	22	16
DEL	1C	17
PG	19	18
ADDR	1B	19
CBR	18	1A
REG	1A	1B
MOVE	23	1C
RELA	1D	1D
TPWR	17	1E
TPRD	11	1F

17.E. Het aftasten met SCAN1

De subroutine SCAN1 brengt niet alleen de inhoud van DISBF eenmaal naar de display, maar tast ook eenmaal het volledige toetsenbord af. Het aftasten gebeurt volgens hetzelfde principe als in P17C1.
Voordat SCAN1 opgeroepen wordt, moet IX wijzen naar DISBF0. SCAN1 brengt dan eenmaal de inhoud van DISBF naar de display, en tast ook eenmaal het volledige toetsenbord af. Daarna eindigt de subroutine met de carry hoog, als er geen toets gesloten was. Indien er wel een contact gesloten was, dan is de carry laag en staat in register A de position-code van de ingedrukte toets.
Indien er meer dan één contact gesloten was, is het de position-code van de toets met de hoogste position-code die in register A staat.
Daar het aftasten slechts eenmaal gebeurt door SCAN1, moet de subroutine in een lus opgenomen worden.
Het programma P17E1 wordt onderbroken als er een toets wordt ingedrukt. In register A is dan de position-code te vinden.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	DD21 3018	P17E1	LD IX,1830	DISBF0
1804	CD 2406		CALL SCAN1	
1807	38 FB		JR C,P17E1	Als geen toets i is.,
1809	F7		RST 30	Naar monitor.
1830	00	DISBF0		
1831	23			
1832	3F			
1833	8D			
1834	AE			
1835	00	DISBF5		

De position-code van de ingedrukte toets kan ook gemakkelijk in een lokatie (KEYBF0) opgeborgen worden.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	21 5018	P17E2	LD HL,1850	KEYBF0
1803	DD21 3018	LUS	LD IX,1830	DISBF0
1807	CD 2406		CALL SCAN1	
180A	38 F7		JR C,LUS	Wacht tot toets in.
180C	77		LD (HL),A	Opbergen in KEYBF0
180D	F7		RST 30	Naar monitor.
1830	00 23 3F 8D AE 00		DISBF0-DISBF5	
1850	..		KEYBF0	

Na het succes met P17E2 is het volgende programma een poging om de code van achtereenvolgens ingedrukte toetsen keurig na elkaar in KEYBF te laden.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	21 5018	P17E3	LD HL,1850	KEYBF0
1803	06 40		LD B,40	40 codes opbergen
1805	C5	VLGCO	PUSH BC	Bewaar B tijdelijk
1806	DD21 3018	LUS	LD IX,1830	DISBF0
180A	CD 2406		CALL SCAN1	
180D	38 F7		JR C,LUS	Terug als geen key in
180F	77		LD (HL),A	Opbergen in KEYBF
1810	23		INC HL	Volgende plaats aanw.
1811	C1		POP BC	Haal B terug
1812	10 F1		DJNZ VLGCO	Volgende code
1814	F7		RST 30	Naar monitor
1830	..	DISBF0		
1835	..	DISBF5		
1850	..	KEYBF0		
1851	..	KEYBF1		

Register B telt hoeveel codes reeds opgeslagen zijn, en beëindigt het programma nadat 40 codes zijn geplaatst. Daar SCAN1 de inhoud van B te niet doet, moet B in de stack bewaard worden.
Bij het starten van het programma moet heel kort op de toets GO gedrukt worden. Druk achtereenvolgens op de toetsen 3, 7, B, F, 2, 6, A enz., zodat de codes nadien gemakkelijk te herkennen zijn in KEYBF.
De poging om 40 (64 dec.) toetsen in te drukken zal wel mislukken. Het programma komt, na het indrukken van slechts enkele toetsen, terug naar de monitor. Een controle in KEYBF vanaf adres 1850 leert, dat de position-codes van de bediende toetsen wel aanwezig zijn, maar van elk een hele reeks.
Een grondiger studie van P17E3 maakt dat verklaarbaar. Het aftasten van het toetsenbord door SCAN1 gaat zo snel, dat gedurende de tijd dat een toets ingedrukt blijft, de lus VLGCO (VoLGende COde) meerdere malen doorlopen wordt, en zo ook meerdere lokaties van KEYBF met dezelfde code laadt.
Ook als P17E3 gestart wordt doordat u zeer lang op de toets GO drukt, wordt naar de moni-

tor teruggekeerd. In KEYBF staat dan 40 maal de position-code 16 van de toets GO. Wat weer gemakkelijk verklaarbaar is.
Hieruit blijkt dat met SCAN1 slechts de position-code van een enkele toets opgehaald kan worden.

17.F. Het aftasten met SCAN

Subroutine SCAN brengt de inhoud van de DISBF naar de display totdat er een toets wordt ingedrukt. Dit houdt in dat ondertussen ook het toetsenbord wordt afgetast. Zodra een gesloten contact wordt ontdekt, eindigt de subroutine SCAN. In register A staat dan de internal code van de ingedrukte toets.
Dit is een eerste verbetering t.o.v. SCAN1. Als nu bijvoorbeeld op toets 3 wordt gedrukt, komt in register A de waarde 03.
Een tweede voordeel is, dat SCAN niet in een lus moet worden opgenomen om één code op te halen, wat het programma vereenvoudigt.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	DD21 3018	P17F1	LD IX,1830	DISBFØ
1804	CD FE05		CALL SCAN	
1807	F7		RST 30	Naar monitor

Een derde voordeel kan ondervonden worden bij het starten van programma P17F1. De toets GO mag zeer lang ingedrukt worden. De subroutine SCAN start wel, wat bewezen wordt door de tekst die op de display komt, maar het aftasten begint pas als er geen enkele toets meer ingedrukt is. Dus na het loslaten van toets G0.
Om de internal code in een lokatie te laden, moet rekening gehouden worden met het feit dat SCAN ook HL te niet doet.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	21 5018	P17F2	LD HL,1850	KEYBFØ
1803	DD21 3018		LD IX,1830	DISBFØ
1807	E5		PUSH HL	Bewaar HL tijdelijk
1808	CD FE05		CALL SCAN	Haal internal code
180B	E1		POP HL	Haal HL terug
180C	77		LD (HL),A	Int.code naar locatie
180D	F7		RST 30	Monitor

De bedoeling van P17E3 kan hier nu gerealiseerd worden. Het programma P17F3 is ingesteld om de internal code van tien toetsen in KEYBF te laden.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	21 5018	P17F3	LD HL,1850	KEYBFØ
1803	DD21 3018		LD IX,1830	DISBFØ
1807	06 10		LD B,10	16 codes ophalen
1809	E5	VLGCO	PUSH HL	Bewaar HL tijdelijk
180A	C5		PUSH BC	Bewaar B tijdelijk
180B	CD FE05		CALL SCAN	Haal 1 int.code
180E	C1		POP BC	Haal B terug

180F	E1		POP HL	Haal HL terug
1810	77		LD (HL),A	Code naar KEYBF
1811	23		INC HL	Wijs volgende plaats
1812	10 F5		DJNZ VLGCO	Terug voor volgende
1814	F7		RST 30	Monitor

Let er op dat bij SCAN; DISBFO slechts eenmaal aangewezen moet worden, zodat de instructie LD IX,1830 niet in de lus moet worden opgenomen.
Het resultaat van P17F3 in KEYBF toont ook aan, dat geen enkele code dubbel werd geplaatst, zelfs niet als er zeer lang op de toetsen werd gedrukt.
Subroutine SCAN elimineert ook de contactdender die optreedt bij de mechanische contacten van het toetsenbord.
De internal code van de ingedrukte toets kan ook rechtstreeks op de display gebracht worden. Hiervoor moet gebruik worden gemaakt van de subroutine HEX7SG, die de waarde in register A omvormt in twee patronen, en deze in de lokatie plaatst die HL aanwijst.
Daar de patronen in DISBFO en DISBF1 moeten komen, moet HL telkens met 1830 herladen worden. Anderzijds eist ook SCAN dat DISBF0 aangewezen wordt door IX.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	DD21 3018	P17F4	LD IX,1830	DISBF0
1804	CD FE05		CALL SCAN	Haal internal code
1807	21 3018		LD HL,1830	DISBF0
180A	CD 7806		CALL HEX7SG	Int.code naar 2 patr.
180D	18 F5		JR LUS	Herhaal

Het volgende programma is een uitbreiding van P17F4. De internal code van de ingedrukte toetsen wordt eveneens op de display getoond, maar bovendien in KEYBF geladen. Daar is het niet het programma dat bepaalt hoeveel codes geplaatst worden. De gebruiker kan op ieder ogenblik naar de monitor terugkeren, door op de toets MOVE te drukken.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	DD21 3018	P17E5	LD IX,1830	DISBF0
1804	11 5018		LD DE,1850	KEYBF0
1807	CD FE05	LUS	CALL SCAN	Haal internal code
180A	FE 1C		CP 1C	MOVE ?
180C	28 0A		JR Z,STOP	Dan naar STOP
180E	12		LD (DE),A	Code naar KEYBF
180F	13		INC DE	Volgende plaats KEYBF
1810	21 3018		LD HL,1830	DISBF0
1813	CD 7806		CALL HEX7SG	Van code naar 2 patro.
1816	18 EF		JR LUS	Herhaal
1818	F7	STOP	RST 30	Naar monitor

18. BLOCK TRANSFER AND SEARCH GROUP

18.A. Block transfer

Stel dat P15E1 in de RAM staat, maar daar plaats moet maken voor een nog te schrijven programma. P15E1 blijft nodig, maar moet nu beginnen vanaf adres 1860. In plaats van het programma te herschrijven, kan er ook een programma ontwikkeld worden dat P15E1 overbrengt (transfer) naar de gewenste plaats. Daarbij kan dan HL het begin van de bron aanwijzen, en DE het begin van de bestemming. In B kan de lengte van het over te brengen blok staan.

Bronprogramma op zijn oorspronkelijke plaats :

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	0E 00	P15E1	LD C,0	Voor 265 Hz.
1802	21 C000		LD HL,00C0	Dec. 192 perioden.
1805	CD E405		CALL TONE	Opwekken lage toon.
1808	0E C0		LD C,C0	Voor 325 Hz.
180A	21 0001		LD HL,0100	Dec. 256 perioden.
180D	CD E405		CALL TONE	Opwekken hoge toon.
1810	18 EE		JR P15E1	Herhaal.

Programma voor het overplaatsen :

Adres	Mach. taal	Label	Mnemonic	Verklaring
1900	21 0018	P18A1	LD HL,1800	Begin bron.
1903	11 6018		LD DE,1860	Begin bestemming.
1906	06 12		LD B,12	Aantal byte in P15E1.
1908	7E	TRANS	LD A,(HL)	Haal byte uit bron.
1909	12		LD (DE),A	Breng byte naar bestem
190A	23		INC HL	Volgende byte bron.
190B	13		INC DE	Volgende plaats bestem
190C	05		DEC B	Aftellen.
190D	20 F9		JR NZ,TRANS	Herhaal tot B=0.
190F	F7		RST 30	Naar monitor.

Na het uitvoeren an P18A1 staat P15E1 tweemaal in de RAM. Zoals voordien vanaf adres 1800, en nu ook vanaf adres 1860. Het overgeplaatste programma werkt evengoed als het originele op adres 1800. Dit is niet bij alle overgeplaatste programma's het geval. Programma's die een of meer JP nn-instructies bevatten, die laten overspringen naar een absoluut adres binnen het programma, zullen niet werken als het bronprogramma verwijderd is.
Met P15E1 is dit niet het geval omdat het programma alleen JR n-instructies bevat. Zulke programma's worden, zoals we eerder zagen, relocable (verplaatsbaar) genoemd.
P18A1 kan een blok overbrengen dat uit maximaal 256 dec. byte bestaat. Daarvoor moet register B met 00 geladen worden.
Voor het overbrengen van grotere blokken zou ook register C ingeschakeld moeten worden voor het aftellen.

18.B. LDIR (LoaD Increment and Repeat)

Met deze ene instructie kan een blok overgebracht worden als HL naar het laagste adres van de bron wijst, en als DE het laagste adres van de bestemming aanduidt. De lengte van het blok moet daarbij in BC staan.
De instructie LDIR doet evenveel als het deel van 1908 tot 190E uit P18A1.
LDIR telt niet register B af, maar registerpaar BC. Daardoor is de lengte van het blok niet meer beperkt.
De instructie LDIR verhoogt telkens HL en DE en verlaagt BC, tot BC = 0.
Na het uitvoeren van LDIR zijn de flags H, P/V en N laag.
Hetzelfde resultaat als met P18A1 kan nu met een korter programma verkregen worden.

Adres	Mach. taal	Label	Instructie	Verklaring
1900	21 0018	P18B1	LD HL,1800	Begin bron.
1903	11 8018		LD DE,1880	Begin bestemming.
1906	01 1200		LD BC,0012	Aantal byte in blok.
1909	EDB0		LDIR	Block transfer.
190B	F7		RST 30	Naar monitor.

18.C. LDI (LoaD and Increment)

Als LDIR, maar niet uit zichzelf herhalend. De byte die is aangewezen door de HL wordt overgebracht naar de aangewezen lokatie. Daarna worden HL en DE verhoogd en BC verlaagd.
De flags H en N worden laag. Flag S en Z veranderen niet. Flag P/V blijft hoog zolang BC niet nul is.
Het voordeel van LDI is dat nog andere instructies in de lus kunnen worden opgenomen.
Programma P1801 heeft als taak, vanuit het blok op 1830 tot 183F alleen de bytes van de even adressen over te brengen naar opeenvolgende lokaties vanaf 1850.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	21 3018	P18C1	LD HL,1830	Bron.
1803	11 5018		LD DE,1850	Bestemming.
1806	01 0800		LD BC,0008	Aantal.
1809	EDA0	TRANS	LDI	Overbrengen.
180B	23		INC HL	1 bronadres overlaten.
180C	EA 0918		JP PE,TRANS	Herhaal tot BC=0
180F	F7		RST 30	Naar monitor.
1830	00	BRON		
1831	01			
1832	02			
1833	03			
183F	0F			
1850	FF	BESTEM		
1851	FF			
1858	FF			

In de TRANSfer-lus is hier nog eens een extra INC HL opgenomen, zodat HL altijd naar even adressen verwijst,
De JP PE,TRANS laat herhalen zolang BC groter is dan nul.

18.D. Overlapping

Als het bronadres en het bestemmingsadres verder uit elkaar liggen dan de lengte van het over te brengen blok zijn er geen problemen met LDIR en LDI.
Programma P18D1 heeft als taak, een blok tien lokaties verder te plaatsen.

Adres	Mach. taal	Label	Mnemonic	Verklaring.
1800	21 5018	P18D1	LD HL,1850	Begin bron.
1803	11 6018		LD DE,1860	Begin bestemming.
1806	01 1800		LD BC,0018	Lengte blok.
1809	EDB0		LDIR	Transfer.
180B	F7		RST 30	Naar monitor.
1850	50	BRON		
1851	51			
1852	52			
1866	66			
1867	67			

Daar het laagste bestemmingsadres binnen het oorspronkelijke blok ligt, wordt reeds bij het overbrengen van de eerste byte het oorspronkelijk blok te niet gedaan.
Het verlies aan informatie door overlapping bij transfer kan vermeden worden door de transfer in omgekeerde volgorde uit te voeren. Dus eerst de laatste byte uit het blok overbrengen, dan de voorlaatste enz., zoals in figuur 18.D.I.
Bij het overplaatsen van een blok met LDIR of LDI naar een lager adres, treedt er geen beschadiging op.

Adres	Mach. taal	Label	Mnemonic	Verklaring.
1800	21 5018	P18D2	LD HL,1850	Begin bron.
1803	11 4018		LD DE,1840	Begin bestemming.
1806	01 1800		LD BC,0018	Lengte blok.
1809	ED B8		LDIR	Transfer.
180B	F7		RST 30	Naar Monitor.

Hoewel het oude blok overschreven wordt door het nieuwe, heeft dit laatste dezelfde inhoud als het oorspronkelijk blok.

18.E. LDDR (LoaD Decrement and Repeat)

Brengt het blok over waarvan HL het hoogste adres van de bron aanwijst, en DE het hoogste adres van de bestemming. De lengte van het blok moet weer in BC staan.
Na het overbrengen van één byte worden HL, DE en BC verlaagd, waarna de volgende byte overgebracht wordt. Dit, totdat BC = 0.

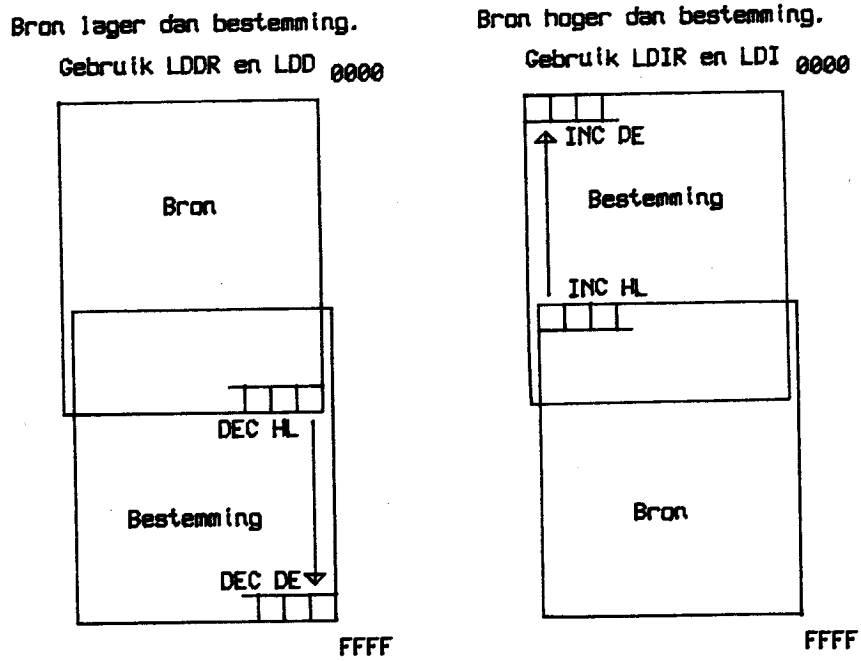


Fig.18D1

De opcode voor LDDR is EDB8. Na het uitvoeren van LDDR zijn de flags H, P/V en N laag. In geval van overlapping kan zonder beschadiging een blok overgebracht worden naar een hoger adresgedeelte.
Het is een goede gewoonte, LDDR toe te passen voor een transfer naar een hoger adres, en LDIR voor een transfer naar een lager adres. Er hoeft dan nooit nagegaan te worden of er overlapping optreedt.

18.F. LDD (LoaD and Decrement)

Net als LDDR, maar zonder automatisch herhalen, zodat nog andere instructies in de lus kunnen worden opgenomen.
Opcode EA8. Flags H en N worden laag. Flag P/V blijft hoog zolang BC niet tot 0 verminderd is.

18.G. CPIR (ComPare Increment and Repeat)

Vergelijkt de inhoud van A met inhoud van de geheugenlokatie die is aangewezen door HL. Indien A niet gelijk is aan (HL), wordt HL verhoogd en BC verlaagd, waarna de instructie wordt hervat. En zo verder tot BC = 0 of A = (HL), waarna de instructie eindigt.

De flags zijn dan :

S Hoog als het resultaat van de laatste vergelijking negatief was, anders laag
Z Hoog als A = (HL), anders laag
H Hoog als er geen borrow was van bit 4, anders laag
P/V Hoog als BC > 0, anders laag
N Hoog
C Onveranderd.

De instructie wordt gebruikt om een waarde op te zoeken in een blok. Dat de waarde gevonden is na het uitvoeren van CPIR, is te zien aan de flag Z die dan hoog is. Het adres waar de gezochte waarde staat, is HL – 1.
Als het blok helemaal doorzocht is zonder resultaat, eindigt ook de instructie, maar dan zijn de flags Z en P/V laag, en staat BC op 0000. Staat de gezochte waarde in de laatste lokatie van het blok, dan is Z wel hoog, maar P/V laag.
Programma P18G1 zoekt uit of de waarde C9 in het monitorprogramma voorkomt.

Adres	Mach. taal	Label	Mnemonic	Verklaring.
1800	21 0000	P18G1	LD HL,0000	Begin monitor program.
1803	01 FF07		LD BC,07FF	Lengte monitor program
1806	3E C9		LD A,C9	Te zoeken waarde.
1808	EDB1		CPIR	ComPare Increment Rep.
180A	F7		RST 30	Naar monitor.

Na het uitvoeren van het programma moet eerste naar flag Z gekeken worden. Indien deze hoog is, kan het adres waar de gezochte waarde staat, gevonden worden door van HL 1 af te trekken.

Taak 1. Zoek uit of in de monitor de instructies JR NC, JR C, RET C of RET PE toegepast worden.
Controleer het resultaat in de Monitor program source listing.

Als in de monitor een instructie toegepast wordt waarvan de opcode uit twee byte bestaat, kan dit met P18G1 niet nagegaan worden. Wel met P18G2, die een uitbreiding bevat.

Adres	Mach. taal	Label	Mnemonic	Verklaring.
1800	21 0000	P18G2	LD HL,0000	Begin monitor
1803	01 FF07		LD BC,07FF	Lengte monitor.
1806	3EED	LUS	LD A,ED	Eerste byte dub.opcode
1808	EDB1		CPIR	Zoek eerste byte.
180A	20 03		JR NZ,STOP	Als 1ste niet aanwezig
180C	3E 21		LD A,21	Tweede byte dub.opcode
180E	BE		CP (HL)	In volgende locatie ?
180F	20 F5		JR NZ,LUS	Neen, dan herhalen.
1811	F7	STOP	RST 30	Naar monitor.

Volgens hetzelfde principe kan nu ook een programma geschreven worden dat een woord van drie, vier of meer byte opzoekt.

Taak 2. Wijzig P18G1 zodat het woord (van twee byte) opgezocht wordt dat in het registerpaar DE staat.

18.H. CPDR (ComPare Decrement and Repeat)

Net als CPIR, maar HL wordt telkens verlaagd, waardoor het zoeken in omgekeerde zin gebeurt. De lokatie waar de gevonden waarde staat is hier HL + 1. Flags als bij CPIR.
Programma P18H1 doorzoekt eveneens het monitorprogramma, maar nu in omgekeerde richting.

Adres	Mach. taal	Label	Mnemonic	Verklaring.
1820	21 FF07	P18H1	LD HL,01FF	Hoogste adres.
1823	01 FF07		LD BC,01FF	Lengte blok.
1826	3E D9		LD A,D9	EXX
1828	EDB9		CPDR	Zoek.
182A	F7		RST 30	Naar monitor.

Taak. Pas P18G2 aan voor zoeken in omgekeerde richting.

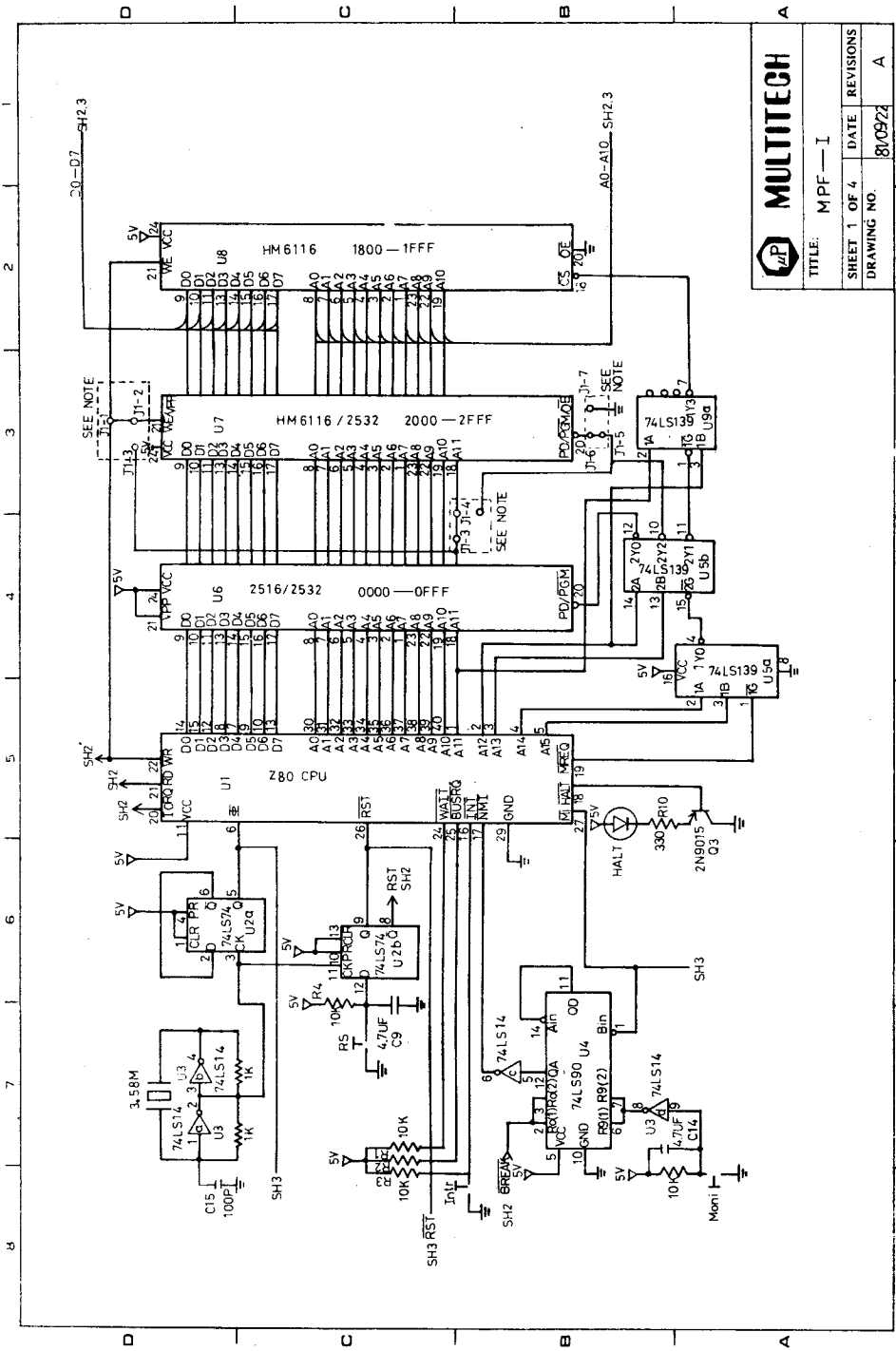
18.I. CPI (ComPare and Increment)

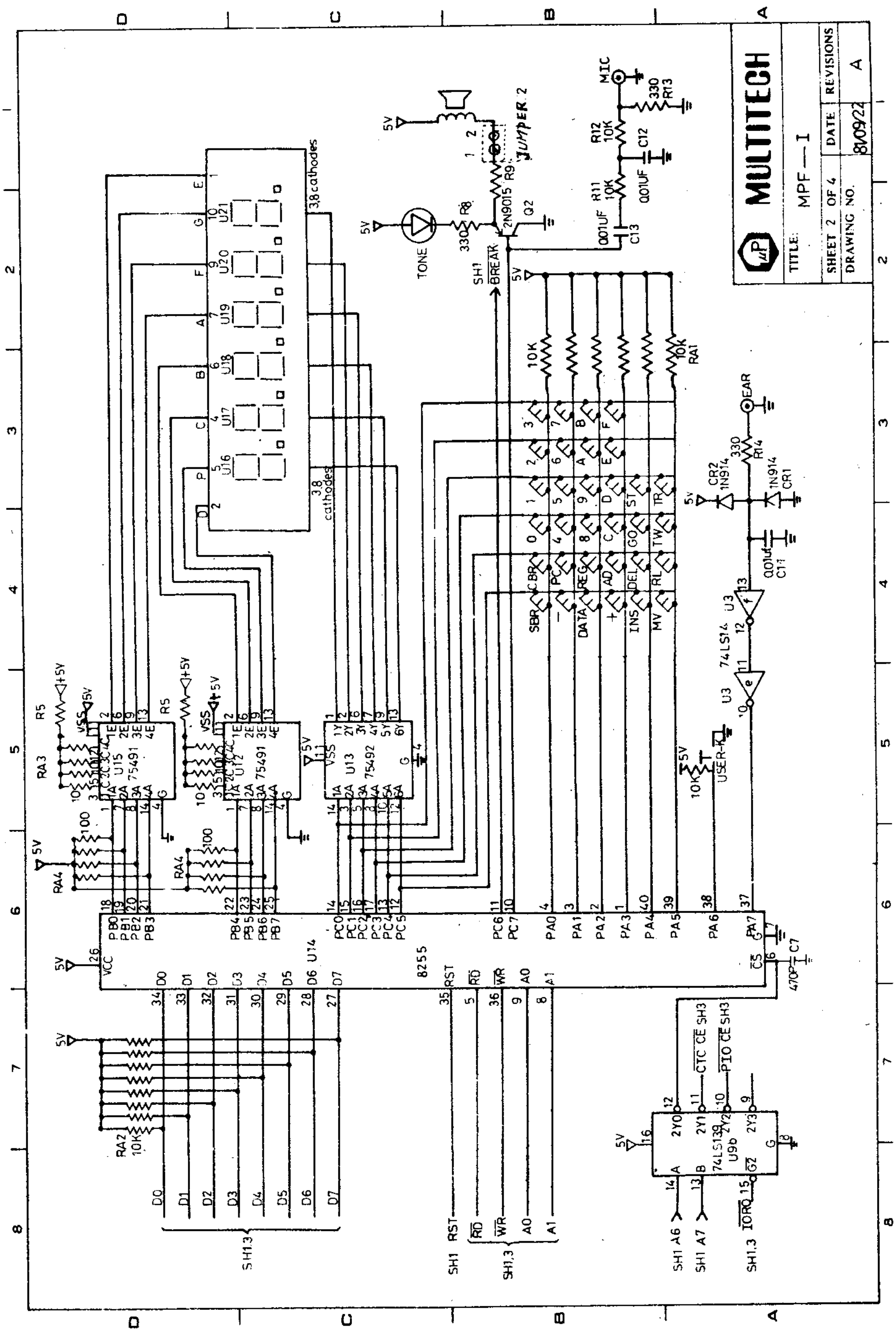
Net als CPIR, maar zonder dat de instructie hervat wordt.
Na iedere vergelijking kunnen nu nog andere instructies uitgevoerd worden.
Flags als bij CPIR.
Programma P18I1 dient alleen als voorbeeld hoe CPI toegepast dient te worden. Het programma zoekt uit of een C9 voorkomt op een even adres. Dit wordt bereikt door na iedere vergelijking met CPI het registerpaar HL nog eens extra te verhogen, waardoor alle oneven adressen overblijven.

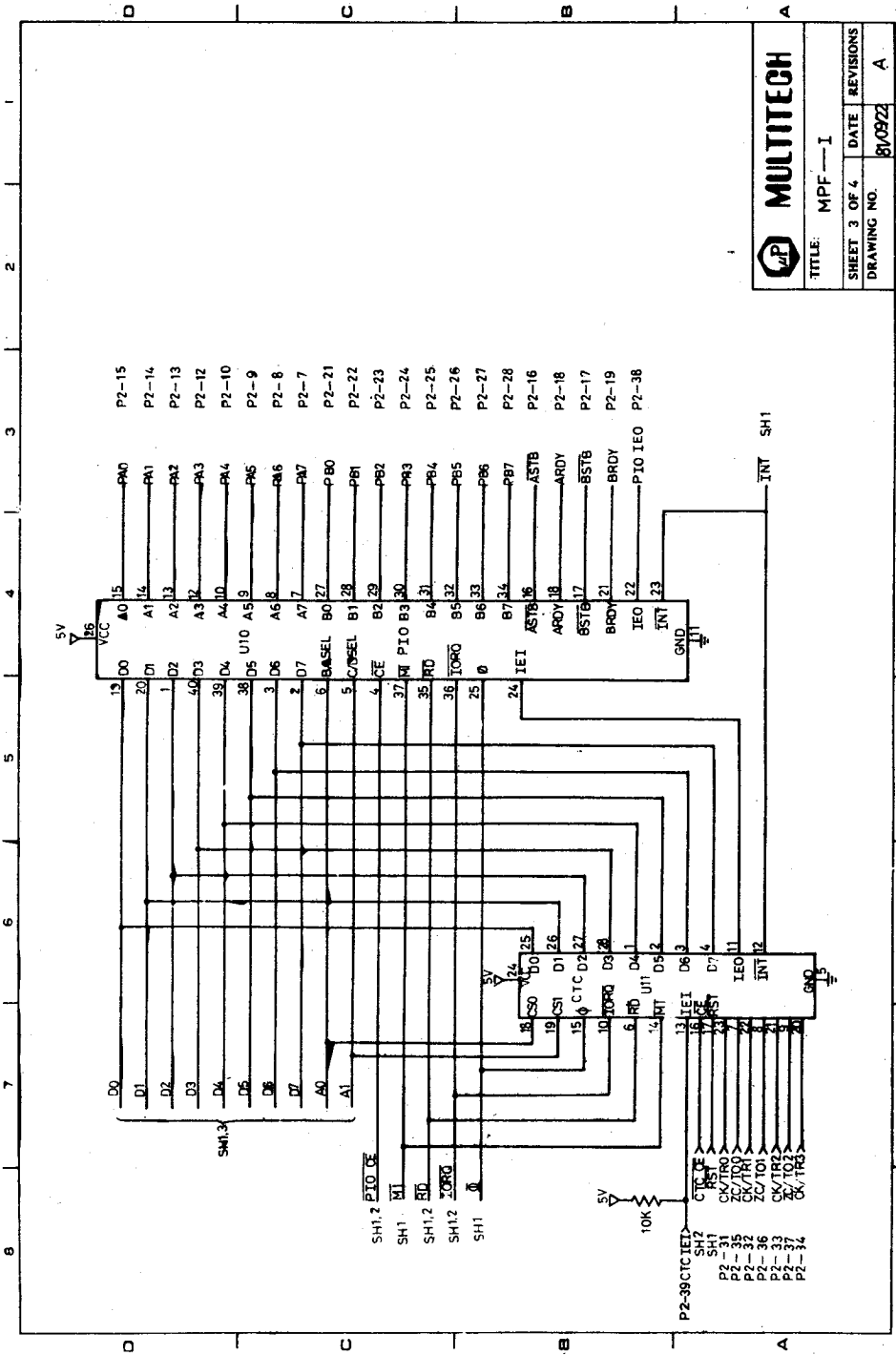
Adres	Mach. taal	Label	Mnemonic	Verklaring.
1800	21 0000	P18I1	LD HL,0000	Begin blok.
1803	01 0004		LD BC,0400	Halve lengte blok.
1806	3E C9		LD A,C9	Te zoeken.
1808	EDA1	LUS	CPI	Compare and Increm. HL
180A	28 04		JR Z,STOP	Indien gevonden.
180C	23		INC HL	Verhoog HL extra.
180D	EA 0818		JP PE,LUS	Zolang P/V hoog is.
1810	F7		RST 30	Naar monitor.

18.J. CPD (ComPare Decrement)

Net als CPI, maar voor het doorzoeken van een blok in omgekeerde zin.


MULTITECH
TITLE: MPF — I
SHEET 1 OF 4
DRAWING NO.
DATE
REVISIONS
A
8/09/22
HM6116 1800 — 1FFF
HM6116 / 2532 2000 — 2FFF
2516/2532 0000 — 0FFF
Z80 CPU
74LS139
74LS74
74LS14
74LS90
SEE NOTE
HALT
Moni
Init
3.58M


MULTITECH
TITLE: MPF — I
SHEET 2 OF 4
DRAWING NO.
DATE
81/09/22
REVISIONS
A

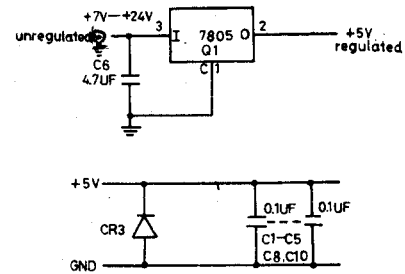

MULTITECH
TITLE: MPF—I
SHEET 3 OF 4
DRAWING NO.
DATE
81/09/22
REVISIONS
A
U10
PIO
U11
CTC
10K
5V
GND
P2-15
P2-14
P2-13
P2-12
P2-10
P2-9
P2-8
P2-7
P2-21
P2-22
P2-23
P2-24
P2-25
P2-26
P2-27
P2-28
P2-16
P2-18
P2-17
P2-19
P2-38
PIO IEO
SH1
P2-39CTC IEI

P1 PIN FUNCTION

PIN NO	SIGNAL	PIN NO	SIGNAL
1	A11	21	A10
2	A12	22	A9
3	A13	23	A8
4	A14	24	A7
5	A15	25	A6
6	CLK	26	A5
7	D4	27	A4
8	D3	28	A3
9	D5	29	A2
10	D6	30	A1
11	NC	31	A0
12	D2	32	GND
13	D7	33	$\overline{RFSH}$
14	D0	34	$\overline{M1}$
15	D1	35	$\overline{RESET}$
16	$\overline{INT}$	36	$\overline{BUSRQ}$
17	$\overline{NMI}$	37	$\overline{WAIT}$
18	$\overline{HALT}$	38	$\overline{BUSAK}$
19	$\overline{MREQ}$	39	$\overline{WR}$
20	$\overline{IORQ}$	40	$\overline{RD}$

P2 PIN FUNCTION

PIN NO	SIGNAL	PIN NO	SIGNAL
1	NC	21	PB0
2	NC	22	PB1
3	NC	23	PB2
4	NC	24	PB3
5	NC	25	PB4
6	NC	26	PB5
7	PA7	27	PB6
8	PA6	28	PB7
9	PA5	29	NC
10	PA4	30	GND
11	GND	31	CK/TR0
12	PA3	32	CK/TR1
13	PA2	33	CK/TR2
14	PA1	34	CK/TR3
15	PA0	35	ZC/TO0
16	$\overline{ASTB}$	36	ZC/TO1
17	$\overline{BSTB}$	37	ZC/TO2
18	ARDY	38	PIO IEO
19	BRDY	39	CTC IEI
20	NC	40	NC



NOTE: 1. U7 is optional, it may be 2516. 2716 .2532, 2732, 6116

2. Jumper 1 is used for selecting certain memory type

Memory Type	Memory Address	Cuts	Jumpers
2516 2716	2000-27FF	NIL	NIL
2532	2000-2FFF	NIL	NIL
2732	2000-2FFF	PIN1 TO PIN2 PIN3 TO PIN4 PIN5 TO PIN6	PIN2 TO PIN3 PIN4 TO PIN5 PIN6 TO PIN7
6116	2000-27FF	PIN3 TOPIN4	PIN4 TO PIN5

MULTITECH

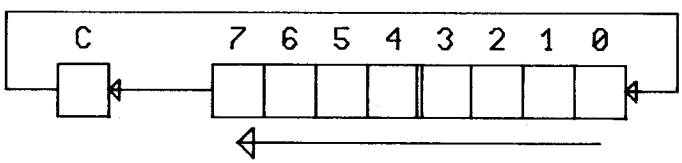
TITLE: PF—I

SHEET 4 OF 4	DATE	REVISIONS
DRAWING NO.	81/09/22	A

19. ROTATE AND SHIFT GROUP

19.A. RLA (Rotate Left A)

Alle bits van A worden één plaats naar links geschoven. Bit 7 komt in de carry, terwijl de carry in bit 0 schuift. Zie figuur 19.A.1.



RLA
RLs

Fig. 19A1

Opcode : 17.
De flags S, Z en P/V worden niet beïnvloed. Flag H en N worden laag.
Met de schakelaars-leds interface kan de werking van de rotate- en shift-instructies zichtbaar gemaakt worden.
Het programma P19A1 toont op de leds hoe RLA de bits in A laat roteren.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	97	P19A1	SUB A	Maak carry laag.
1801	3E 01		LD A,01	Maak bit 0 hoog.
1803	D3 F8	LEDS	OUT (F8),A	Naar leds.
1805	10 FE	TIJD	DJNZ TIJD	Wacht hier even.
1807	0D		DEC C	
1808	20 FB		JR NZ,TIJD	
180A	17		RLA	Rotate Left Accu
180B	18 F6		JR LEDS	Herhaal.

Het resultaat van P19A1 is, dat één opgelichte led naar links schuift. Vooraan in het programma wordt eerst de carry laag gemaakt door SUB A. Zonder deze instructie is er een kans dat er twee opgelichte leds schuiven, als toevallig de carry hoog is bij het starten van het programma.
Let erop, dat op een bepaald ogenblik geen enkele led brandt.
Dit is het moment dat de carry hoog is, en alle bits in A laag zijn.
Door A met een andere waarde te laden kan iedere willekeurige combinatie van opgelichte leds naar links schuiven.

Taak 1. Laat drie naast elkaar liggende leds naar links schuiven.
Taak 2. Laat één gedoofde led naar links schuiven.
Taak 3. Met de schakelaars moet in te stellen zijn, welke opgelichte leds naar links schuiven.

Als RLA negen maal uitgevoerd wordt, is de inhoud van A en carry weer als voorheen. Wel is iedere bit dan even in de carry geweest, en kon er geen bewerking uitgevoerd worden naargelang de carry hoog was of niet.
Hierdoor is het nu mogelijk een programma te schrijven, dat telt hoeveel bits van een byte hoog zijn.
Als voorbeeld telt P19A2 hoeveel schakelaars er hoog staan, en toont het resultaat in binaire vorm op de leds.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	06 09	P19A2	LD B,9	Lusteller.
1802	97		SUB A	Maak carry laag.
1803	4F		LD C,A	Bit-teller op nul.
1804	DB F8		IN A,(F8)	Schakelaars in A.
1806	17	ROT	RLA	Schuif bit 7 in carry.
1807	30 01		JR NC,AFTEL	
1809	0C		INC C	Verhoog bit-teller.
180A	10 FA	AFTEL	DJNZ ROT	Herhaal tot B=0.
180C	79		LD A,C	Resultaat in A.
180D	D3 F8		OUT (F8),A	Resultaat naar leds.
180F	18 EF		JR P19A2	Herhaal.

Taak 4. Als P19A2, maar het resultaat moet op de display komen in plaats van op de leds.

19.B. RL s (Rotate Left)

Maakt dezelfde bewerking als RLA, maar met een van de registers of een RAM-lokatie die aangewezen wordt door HL, XI of IY.

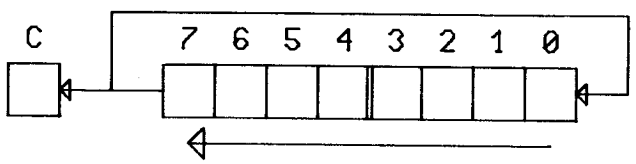
Opcode	Mnemonic
CB17	RL A
CB10	RL B
CB11	RL C
CB12	RL D
CB13	RL E
CB14	RL H
CB15	RL L
CB16	RL (HL)
DDCB d 16	RL (IX+d)
FDCB d 16	RL (IY+d)

Let erop dat de instructies RL (IX+d) en RL (IY+d) uit vier byte bestaan.
Opvallend is, dat hier ook RL A voorkomt. De invloed op A is wel dezelfde als van RLA (uit 19.A.), maar de flags reageren anders.
De invloed van RL n op de flags is :

S Hoog als het resultaat negatief is, anders laag
Z Hoog als het resultaat nul is, anders laag
H Laag
P/V Hoog als het aantal enen in het resultaat even is, anders laag
N Laag
C Hoog als bit 7 van de operand hoog was voor de bewerking.

19.C. RLCA (Rotate Left Circular A)

Alle bits van A worden één plaats naar links geschoven. Bit 7 komt in de carry, maar ook in bit 0. Zie figuur 19.C.1.



RLCA

RLCs

Fig. 19C1

Opcode : 07.
De flags S, Z en P/V worden niet beïnvloed. Flags N en V worden laag.
Na acht instructies RLCA is de inhoud van A weer zoals voorheen. De carry is dan zoals bit 0 van A.
De werking van RLCA kan met P19A1 gedemonstreerd worden, als de RLA vervangen wordt door RLCA. Daar bij RLCA de carry niet meecirculeert, is het niet nodig dat de carry vooraf op 0 wordt gesteld. Het programma kan daardoor iets vereenvoudigd worden.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 01	P19C1	LD A,01	
1802	D3 F8	LEDS	OUT (F8),A	Naar leds.
1804	10 FE	TIJD	DJNZ TIJD	Wacht
1806	0D		DEC C	hier
1807	20 FB		JR NZ,TIJD	even.
1809	07		RLCA	Rotate Left Circuular
180A	18 F6		JR LEDS	Herhaal.

Na vier instructies RLCA zijn de nibbles (1/2 byte) in A van plaats verwisseld. Deze eigenschap kan toegepast worden om twee digits die in A staan, van plaats te doen wisselen.
Programma P19C2 maakt gebruik van de subroutines HEX7SG en SCAN1 om deze eigenschap op de display te demonstreren.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 12	P19C2	LD A,12	Getal in A.
1802	F5	LUS	PUSH AF	Bewaar getal.
1803	21 3018		LD HL,1830	DISBUF0 aanwijzen.
1806	CD 7806		CALL 0678	HEX7SG Patr.in DISBF01
1809	DD21 3018		LD IX,1830	Wijs DISBF0 aan.
180D	CD 2406	TOON	CALL 0624	SCAN1 Patr.naar displ.
1810	10 FB		DJNZ TOON	Wacht hier even.
1812	F1		POP AF	Haal A terug.
1813	07		RLCA	Verwissel
1814	07		RLCA	de
1815	07		RLCA	2
1816	07		RLCA	digits.
1817	18 E9		JR LUS	Herhaal.

19.D. RLC s (Rotate Left Circular)

Net als RLCA, maar met een van de registers, of een RAM-lokatie die aangewezen wordt door HL, IX of IY.
Flags als bij RL s.

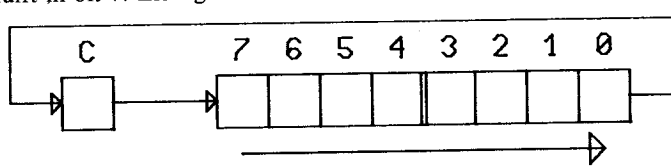
Opcode	Mnemonic
CB07	RLC A
CB00	RLC B
CB01	RLC C
CB02	RLC D
CB03	RLC E
CB04	RLC H
CB05	RLC L
CB06	RLC (HL)
DDCB d 06	RLC (IX+d)
FDCB d 06	RLC (IY+d)

Als toepassingsvoorbeeld het programma P19D1, dat van de waarden in het blok van 1830 tot 183F de digits van plaats doet wisselen.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	21 3018	P19D1	LD HL,1830	Begin blok aanwijzen.
1803	06 10		LD B,10	Lusteller.
1805	CB06	LUS	RLC (HL)	Verwissel
1807	CB06		RLC (HL)	digits
1809	CB06		RLC (HL)	van
180B	CB06		RLC (HL)	plaats.
180D	23		INC HL	Volgende loc.aanwijzen.
180E	10 F5		DJNZ LUS	Herhaal tot einde blok.
1810	F7		RST 30	Naar monitor.
1830	30	BLOK		
1831	31			
1832	32			
183F	3F			

19.E. RRA (Rotate Right A)

Alle bits van A worden één plaats naar rechts geschoven. Bit 0 komt in de carry , en de carry schuift in bit 7. Zie figuur 19.E.I.



RRA

RRs

Fig. 19E1

Opcode : 1F
Flags als bij RLA.

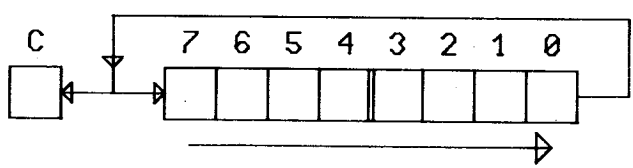
19.F. RR s (Rotate Right)

Als RRA, maar met één van de registers, of een RAM-lokatie die wordt aangewezen door HL, IX of IY.
Flags als bij RL s.

Opcode	Mnemonic
CB1F	RR A
CB18	RR B
CB19	RR C
CB1A	RR D
CB1B	RR E
CB1C	RR H
CB1D	RR L
CB1E	RR (HL)
DDCB d 1E	RR (IX+d)
FDCB d 1E	RR (IY+d)

19.G. RRCA (Rotate Richt Circular A)

Alle bits in A worden één plaats naar rechts geschoven. Bit 0 komt in bit 7, maar ook in de carry. Zie figuur 19.G.1.



RRCA

RRCs

Fig. 19G1

Opcode : 0F
Flags als bij RLCA.

19.H. RRC s (Rotate Right Circular)

Als RRCA, maar met een van de registers, of een RAM-lokatie die wordt aangewezen door HL, IX of IY.
Flags als bij RLC s.

Opcode	Mnemonic
CB0F	RRC A
CB08	RRC B
CB09	RRC C
CB0A	RRC D
CB0B	RRC E
CB0C	RRC H
CB0D	RRC L
CB0E	RRC (HL)
DDCB d 0E	RRC (IX+d)
FDCB d 0E	RRC (IY+d)

19.I. SLA s (Shift Left Arithmetic)

Schuift iedere bit van een van de registers, of van een RAM-lokatie die wordt aangewezen door HL, IX of IY, één plaats naar links. Bit 7 komt in de carry. In bit 0 schuift een laag. Zie figuur 19.I.1.
Flags als bij RL s.

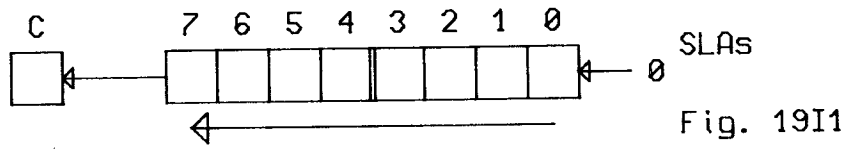


Fig. 19I1

Opcode	Mnemonic
CB27	SLA A
CB20	SLA B
CB21	SLA C
CB22	SLA D
CB23	SLA E
CB24	SLA H
CB25	SLA L
CB26	SLA (HL)
DDCB d 26	SLA (IX+d)
FDCB d 26	SLA (IY+d)

Programma P19I1 laadt 03 in A, en toont deze waarde op de display. Daarna wordt herhaald de instructie SLA A uitgevoerd en de nieuwe inhoud van A naar de display gebracht.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 03	P19I1	LD A,03	Startwaarde in A.
1802	F5	LUS	PUSH AF	Bewaar A.
1803	21 3018		LD HL,1830	DISBF0 aanwijzen.
1806	CD 7806		CALL 0678	HEX7SG Patr.in DISBF0
1809	DD21 3018		LD IX,1830	DISBF0 aanwijzen.
1800	CD 2406	TOON	CALL 0624	Patronen naar displa
1810	10 FE		DJNZ TOON	Wacht even.
1812	F1		POP AF	Haal A terug.
1813	CB 27		SLA A	Shift Left Arithmetic
1815	18 EB		JR LUS	Herhaal.
1830	00	DISBF0		
1831	00	DISBF1		
1835	00	DISBF5		

Op de display verschijnen achtereenvolgens de hexa-waarden uit de volgende tabel, waarin ook de overeenkomstige decimale en binaire waarden zijn weergegeven.

Hexa	Decimaal	C 7654 3210
03	3	0 0000 0011
06	6	0 0000 0110
0C	12	0 0000 1100
18	24	0 0001 1000
30	48	0 0011 0000
60	96	0 0110 0000
C0	192	0 1100 0000
80	128	1 1000 0000
00	0	1 0000 0000

In de tabel valt het direct op, dat A door iedere SLA A verdubbeld wordt. Proeven met andere beginwaarden tonen aan, dat dit geen toeval is maar regel.
Met SLA kan dus rechtstreeks vermenigvuldigd worden met 2, met 4, met 8 enz.. Het schuiven moet wel beëindigd worden zodra de carry hoog is, anders is het produkt fout.
Vermenigvuldigen met andere getallen kan door een combinatie van links schuiven en optellen.
Programma P1912 toont als voorbeeld, hoe 23 (35 dec.) met 6 vermenigvuldigd kan worden. Het produkt komt op de display.

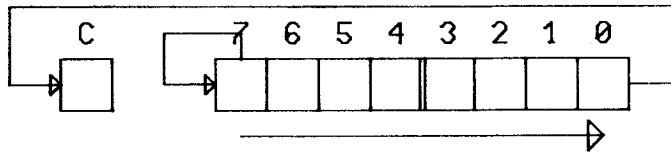
Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 23	P1912	LD A,23	Vermenigvuldigtal in A
1802	CB27		SLA A	23 * 2 in A
1804	47		LD B,A	23 * 2 in B
1805	CB27		SLA A	23 * 4 in A
1807	80		ADD A,B	23 * 6 in A
1808	21 3018		LD HL,1830	DISPF0 aanwijzen.
180B	CD 7806		CALL 0678 .	HEX7SG Patr.in DISBF01
180E	DD21 3018		LD IX,1830	DISBF0 aanwijzen.
1812	CD FE05		CALL 05FE	SCAN Patr.naar display
1815	F7		RST 30	Naar monitor.
1830	00	DISBF0		

Het programma controleert de carry niet, zodat bij een ander (groter) vermenigvuldigtal een foutief produkt getoond kan worden.

Taak. Pas P1912 aan zodat een foutief produkt niet op de display getoond wordt.

19.J. SRA s (Shift Right Arithmetic)

Schuift iedere bit van een van de registers, of van een lokatie die wordt aangewezen door HL, IX of IY, één plaats naar rechts. Bit 0 komt in de carry. Bit 7 schuift in bit 6, maar bit 7 behoudt zijn toestand. Zie figuur 19.J.1.
Flags als bij RL s.



SRAs

Fig. 19J1

Opcode	Mnemonic
CB2F	SRA A
CB28	SRA B
CB29	SRA C
CB2A	SRA D
CB2B	SRA E
CB2C	SRA H
CB2D	SRA L
CB2E	SRA (HL)
DDCB d 2E	SRA (IX+d)
FDCB d 2E	SRA (IY+d)

Het programma P19I1 is aangepast tot P19J1 om de eigenschappen van SRA A te demonstreren.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 40	P19J1	LD A,40	Startwaarde in A.
1802	F5	LUS	PUSH AF	Bewaar A.
1803	21 3018		LD HL,1830	DISBF0 aanwijzen.
1806	CD 7806		CALL 0678	HEX7SG.
1809	DD21 3018		LD IX,1830	DISBF0 aanwijzen.
180D	CD 2406	TOON	CALL 0624	SCAN1
1810	10 FE		DJNZ TOON	Wacht even.
1812	F1		POP AF	Haal A terug.
1813	CB2F		SRA A	Shift Right Arithmetic
1815	18 EB		JR LUS	Herhaal.
1830	00	DISBF0		

En geeft de volgende waarden op de display :

Hexa	Decimaal	7654 3210 C
40	64	0100 0000 0
20	32	0010 0000 0
10	16	0001 0000 0
08	8	0000 1000 0
04	4	0000 0100 0
02	2	0000 0010 0
01	1	0000 0001 0
00	0	0000 0000 1

Waaruit zou kunnen worden afgeleid dat SRA door 2 deelt.

Met de startwaarde 48 krijgen we de volgende tabel :

Hexa	Decimaal	7654 3210 C
48	72	0100 1000 0
24	36	0010 0100 0
12	18	0001 0010 0
09	9	0000 1001 0
04	4	0000 0100 1
02	2	0000 0010 0
01	1	0000 0001 0
00	0	0000 0000 1

Opmerkelijk is hier de deling 9/2, die als resultaat 4 oplevert en ook de carry hoog maakt ; daarmee wordt de rest van de deling aangegeven.

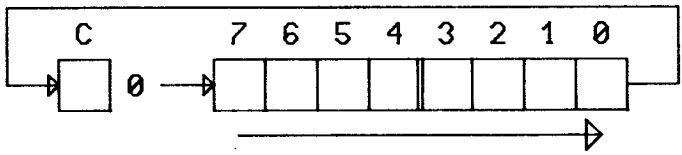
Het is leerrijk, met de startwaarde 80 de tabel op te stellen.

Hexa	Decimaal	7654 3210 C	2'complement
80	128	1000 0000 0	-128
C0	192	1100 0000 0	- 64
E0	226	1110 0000 0	- 32
F0	240	1111 0000 0	- 16
F8	248	1111 1000 0	- 8
FC	252	1111 1100 0	- 4
FE	254	1111 1110 0	- 2
FF	255	1111 1111 0	- 1

Waaruit geconcludeerd kan worden, dat positieve waarden groter dan 7F niet met SRA gedeeld kunnen worden, maar dat SLA bijzonder geschikt is om two's complement-waarden te delen, zonder dat hun teken verloren gaat.

19.K. SRL s (Shift Right Logical)

Schuift iedere bit van een van de registers, of van een lokatie die wordt aangewezen door HL, IX of IY, één plaats naar rechts. Bit 0 komt in de carry. In bit 7 schuift een laag. Zie figuur 19.K.1.
Flags als bij RL s.



SRL s

Fig. 19K1

Opcode	Mnemonics
CB3F	SRL A
CB38	SRL B
CB39	SRL C
CB3A	SRL D
CB3B	SRL E
CB3C	SRL H

CB3D	SRL L
CB3E	SRL (HL)
DDCB d 3E	SRL (IX+d)
FDCB d 3E	SRL (IY+d)

Om de invloed van SRL s op een waarde te bestuderen, kan programma P19J1 gebruikt worden, als daar de SRA A vervangen wordt door SRL A.
Met de startwaarde 80 komt dan op de display :

Hexa	Decimaal	7654 3210 C
80	128	1000 0000 X
40	64	0100 0000 0
20	32	0010 0000 0
10	16	0001 0000 0
08	8	0000 1000 0
04	4	0000 0100 0
02	2	0000 0010 0
01	1	0000 0001 0
00	0	0000 0000 1

De tabel toont dat ook SRL s door 2 deelt. De startwaarde is groter dan 7F, maar het quotiënt is toch een positief getal. Ook andere startwaarden groter dan 7F, leveren telkens een positief quotiënt op.
Met de startwaarde C8 krijgen we :

Hexa	Decimaal	7654 3210 C
C8	200	1100 1000 X
64	100	0110 0100 0
32	50	0011 0010 0
19	25	0001 1001 0
0C	12	0000 1100 1
06	6	0000 0110 0
03	3	0000 0011 0
01	1	0000 0001 1
00	0	0000 0000 1

Opnieuw is te zien, dat de carry hoog wordt als bij de deling een rest ontstaat.
Uit het bovenstaande kan worden afgeleid, dat SRL s geschikt is om een waarde zonder teken (unsigned number) door 2 te delen.

19.L. RLD (Rotate Left Digit)

Zie figuur 19.L.1.

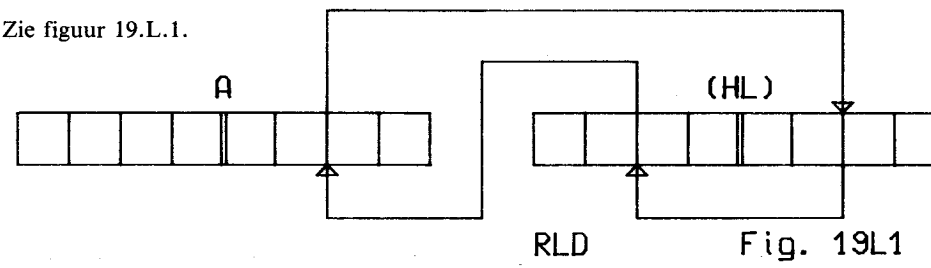


Fig. 19L1

In één byte zijn acht bits. Een halve byte of vier bits wordt ook wel een *nibble* genoemd. De instructie RLD schuift de rechter-nibble uit A naar de rechterhelft van de lokatie die wordt aangewezen door HL. De rechter-nibble van deze lokatie schuift in de linkerhelft, en deze op zijn beurt terug in de rechterhelft van A.
Opcode ED6F.
Flags :

S	Hoog als de inhoud van A negatief is, anders laag
Z	Hoog als na de bewerking de inhoud van A nul is, anders laag
H	Wordt laag
P/V	Hoog als na de bewerking een even aantal bits hoog is in A, anders laag
N	Wordt laag
C	Wordt niet beïnvloed.

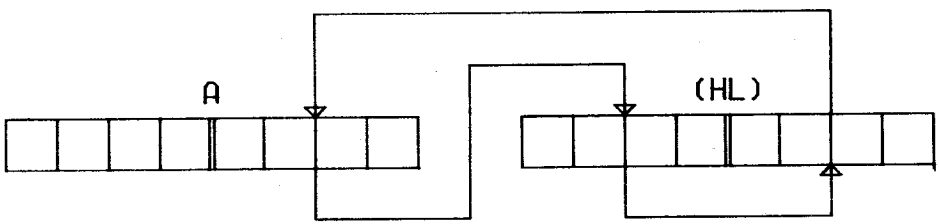
Voor een toepassing met RLD moet eerst aan een vorig programma herinnerd worden. Door P17F2 werd met behulp van de subroutine SCAN de internal code van een toets geladen in een lokatie die werd aangewezen door HL. Het drukken op toets 5 brengt 05 in lokatie 1850. Om nu de waarde van twee na elkaar ingedrukte toetsen in eenzelfde lokatie te krijgen, kan de instructie RLD gebruikt worden. Zo moet na het drukken op de toetsen 3 en 5 in lokatie 1850 de waarde 35 staan.

Adres	Mach. taal	Label	Mnemonics	Verklaring
1800	21 5018	P19L1	LD HL,1850	KEYBF0
1803	DD21 3018		LD IX,1830	DISBF0
1807	E5		PUSH HL	
1808	CD FE05		CALL 05FE	SCAN Eerste toets.
180B	E1		POP HL	
180C	ED6F		RLD	Eerste digit in (HL)
180E	E5		PUSH HL	
180F	CD FE05		CALL 05FE	SCAN Tweede toets.
1812	E1		POP HL	
1813	ED6F		RLD	Tweede digit in (HL)
1815	F7		RST 30	Naar monitor.

Wat het programma doet als op de toetsen 3 en 5 wordt gedrukt, is grafisch weergegeven in figuur 19.L.2. Het resultaat is 35 in de lokatie die is aangewezen door HL.

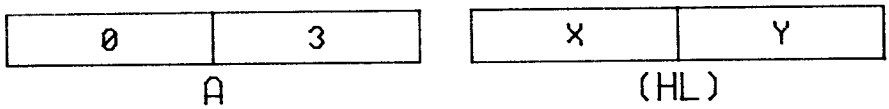
19.M. RRD (Rotate Right Digit)

Zie figuur 19.M.1.

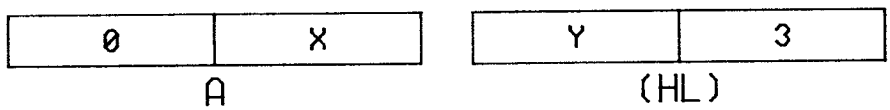


RRD Fig. 19M1

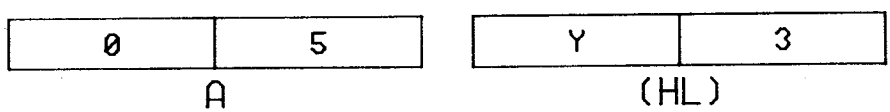
Na drukken op toets 3



Na eerste RLD



Na drukken op toets 5



Na tweede RLD

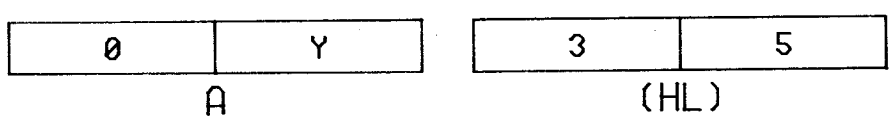


Fig. 19L2

De rechter-nibble van A schuift in de linkerhelft van de lokatie die is aangewezen door HL. De linkerhelft van (HL) schuift in de rechterhelft, en deze in de rechterhelft van A.
Opcode ED67.
Flags als bij RLD.
Als in P19L1 de instructies RLD vervangen worden door RRD, zal na het drukken op de toetsen 3 en 5 in lokatie (HL) de waarde 53 staan. Figuur 19.M.2 toont aan hoe dat gebeurt.

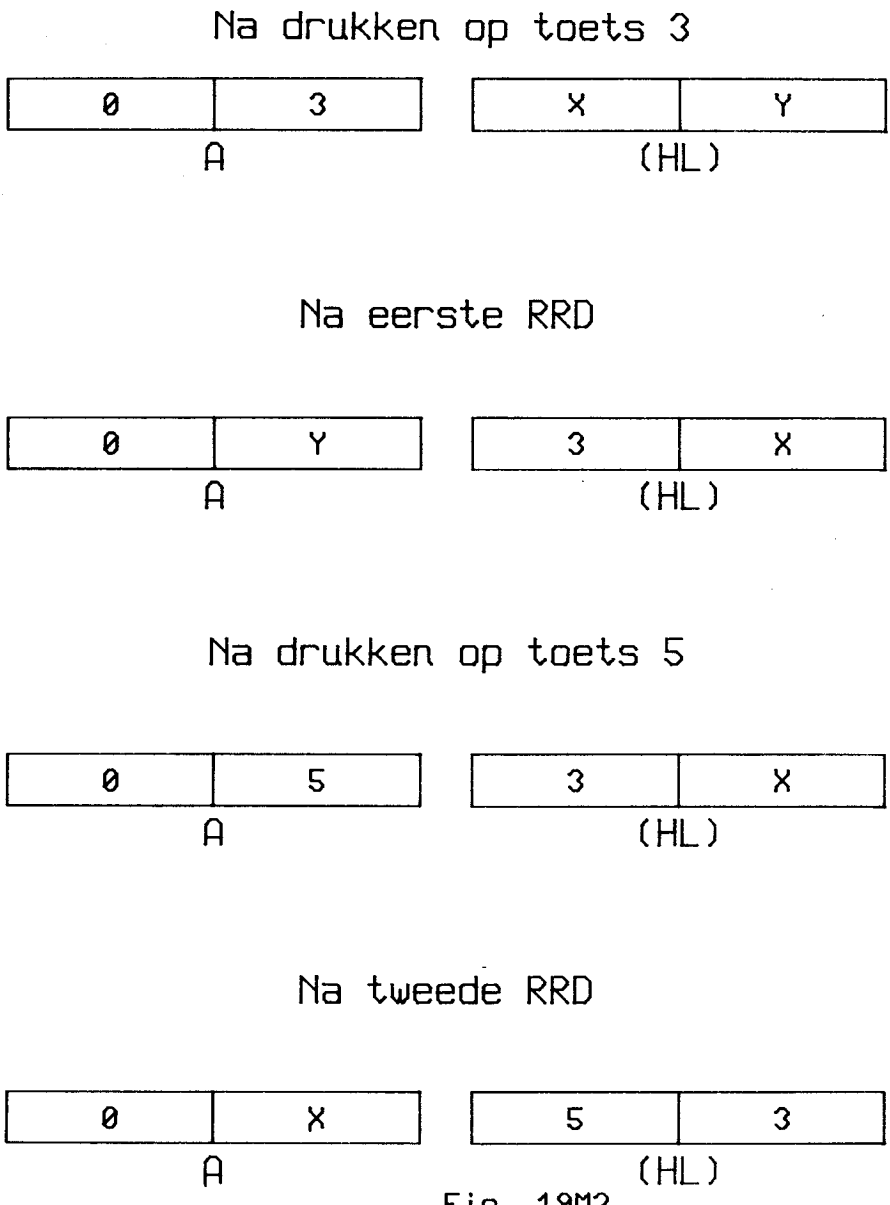


Fig. 19M2

19.N. 16 bit shift-bewerkingen

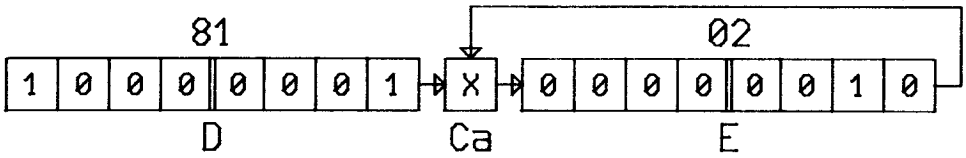
Hoewel de Z-80 geen 16 bit shift- of rotate-instructies heeft, is het toch mogelijk om een shift-bewerking op een registerpaar uit te voeren.
Zo kan een SRL op het registerpaar DE verkregen worden door een combinatie van SR D en RR E. Figuur 19.N.1 verduidelijkt hoe de 16 bit SRL verkregen wordt. Bit 0 die uit E in de carry schuift, heeft geen invloed, daar de volgende SR D zijn bit 0 over die in de carry schuift. Met een iets uitgebreider programma kan de SRL DE gedemonstreerd worden.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	11 0080	P19N1	LD DE,8000	Startwaarde.
1803	21 3018	LUS	LD HL,1830	DISBF0
1806	7B		LD A,E	
1807	CD 7806		CALL 0678	HEX7SG
180A	7A		LD A,D	
180B	CD 7806		CALL 0678	HEX7SG
180E	DD21 3018		LD IX,1830	DISBF0
1812	CD 2406	TOON	CALL 0624	SCAN1 Patr.naar displ.
1815	10 FB		DJNZ TOON	Wacht even.
1817	CB3A		SRL D	Shift Right Logic D
1819	CB1B		RR E	Rotate Right E
181B	18 E6		JR LUS	Herhaal.
1830	00	DISBF0		
1835	00	DISBF5		

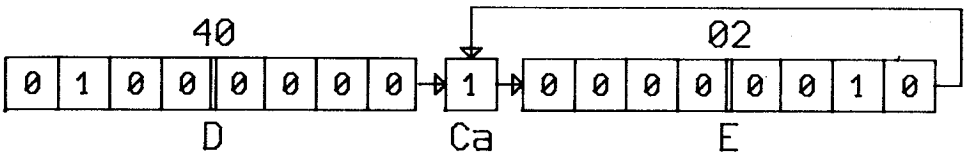
De resultaten van het programma bewijzen de goede werking van de combinatie.

Taak. Onderzoek welke andere 16 bit of rotate-bewerkingen kunnen worden verkregen door combinaties.

Na LD DE,8102



Na SR D



Na RR E

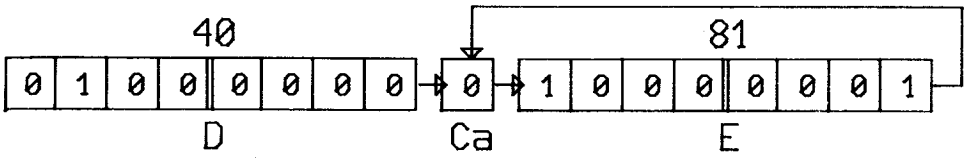


Fig. 19N1

20. BIT SET, RESET AND TEST GROUP

20.A. SET b,s

Maakt de opgegeven bit b van een van de registers, of van een RAM-lokatie aangewezen door HL, IX of IY, hoog.
Flags worden daarbij niet beïnvloed.
Als de aangewezen bit al hoog is, blijft hij hoog.
De ASCII-code is een code voor het weergeven van cijfers, letters en leestekens. De Micro-Prof maakt zelf van deze 7 bit code geen gebruik, maar de meeste printers wel. Als de Micro-Prof een tekst naar zo'n printer wil doorgeven, moet deze in ASCII in de RAM staan.
Het verschil tussen de ASCII-waarde van een hoofdletter en die van dezelfde kleine letter, is dat van deze laatste bit 5 hoog is.

Hoofdletter A	ASCII = 41	binair 0100 0001
Kleine letter a	ASCII = 61	binair 0110 0001
Hoofdletter Z	ASCII = 5A	binair 0101 1010
Kleine letter z	ASCII = 7A	binair 0111 1010

Daar in het geheugen ergens de ASCII-waarde van een hoofdletter staat, kan deze door SET b,s gemakkelijk omgevormd worden tot een kleine letter.
Zo staat vanaf adres 1810 een reeks ASCII-waarden van hoofd- en kleine letters door elkaar. Programma P20A1 heeft als taak de hoofdletters om te vormen tot kleine letters.

Adres	Mach. taal	Label	Mnemonic	Verklaring.
1800	21 1810	P20A1	LD HL,1810	Begin blok.
1803	06 0D		LD B,0D	Lusteller laden.
1805	CBEE	LUS	SET 5,(HL)	Maak bit 5 (HL) hoog.
1807	23		INC HL	Volgende aanwijzen.
1808	10 FB		DJNZ LUS	Herhaal tot einde blok
180A	F7		RST 30	Naar monitor.
1810	4C	TEKST		
1811	65			
1812	72			
1813	65			
1814	6E			
1815	44			
1816	6F			
1817	6F			
1818	72			
1819	44			
181A	6F			
181B	65			
181C	6E			

20.B. RES b,s (RESet)

Maakt de opgegeven bit b van een van de registers, of van een RAM-lokatie aangewezen door HL, IX of IY, laag.
Flags worden daarbij niet beïnvloed.
Als de aangewezen bit al laag is, blijft hij laag.
Van dezelfde tekst uit P20A1 kunnen nu, met RES b,s, alle kleine letters omgevormd worden tot hoofdletters.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	DD21 1018	P20B1	LD IX,1810	Begin blok.
1804	06 0D		LD B,0D	Lengte blok.
1806	DDCB 00 AE	LUS	RES 5,(IX+d)	Maak bit (IX+d) laag.
180A	DD23		INC IX	Volgende aanwijzen.
180C	10 F8		DJNZ LUS	Herhaal tot einde blok
180E	F7		RST 30	Naar monitor.
1810	4C	TEKST		
1811	65			
1812	72			
1813	65			
1814	6E			
1815	44			
1816	6F			
1817	6F			
1818	72			
1819	44			
181A	6F			
181B	65			
181C	6E			

Hier wordt IX gebruikt om de RAM-lokatie aan te wijzen.

Taak. De tekst uit P20B1 bestaat uit drie woorden. Plaats tussen de woorden een spatie. Schrijf een programma dat alle kleine letters in hoofdletters omvormt, maar let wel op de spaties.

20.C. BIT b,s (test BIT)

Test, of de opgegeven bit b van een van de registers, of van een lokatie aangewezen door HL, IX of IY, hoog of laag is.
Aan de bit zelf wordt niets veranderd, maar flag Z neemt een toestand aan die de omgekeerde is van die van de geteste bit.

Flags :

S	Onbekend
Z	Hoog als de onderzochte bit laag is, anders laag
H	Hoog
P/V	Onbekend
N	Laag
C	Wordt niet beïnvloed.

Programma P20C1 telt door middel van BIT b,s hoeveel hoofdletters in de tekst voorkomen.

Adres	Mach. taal	Label	Mnemonic	Verklaring.
1800	21 1018	P20C1	LD HL,1810	Begin tekst.
1803	06 0D		LD B,0D	Lengte tekst.
1805	97		SUB A	Stel A op nul.
1806	CB6E	LUS	BIT 5,(HL)	Hoofdletter ?
1808	20 01		JR NZ,KLEIN	Neen, dan naar KLEIN.
180A	3C		INC A	Tel 1 op in A.
180B	23	KLEIN	INC HL	Wijs volgende aan.
180C	10 F8		DJNZ LUS	Herhaal tot einde blok
180E	F7		RST 30	Naar monitor.
1810	4C	TEKST		
1811	65			
1812	72			
1813	65			
1814	6E			
1815	44			
1816	6F			
1817	6F			
1818	72			
1819	44			
181A	6F			
181B	65			
181C	6E			

Na uitvoering moet in A het aantal hoofdletters staan dat in de tekst voorkomt.

Taak. Wijzig het programma zodat in A het aantal hoofdletters staat, en in C het aantal kleine letters.

21. INPUT AND OUTPUT GROUP

21.A. Memory mapped output

Data naar een randtoestel sturen gaat het eenvoudigst via de poorten van de Z-80, zoals besproken in hoofdstuk 11.
Een randtoestel dat data moet ontvangen, kan ook aangesloten worden als een RAM van één byte groot. De interface uit figuur 11.C.1 moet dan uitgebreid worden om 16 adreslijnen te controleren en signaal IORQ' moet vervangen worden door signaal MREQ'.
Data overbrengen naar zo'n memory mapped randtoestel gaat dan niet meer met OUT (n),A. Wel kan dit met al de instructies die data overbrengen naar een RAM-lokatie.
Voorbeelden : LD (nn),A, LD (HL),A, LD (HL),n en LD (IX+d),A.

21.B. Memory mapped input

Als aan het schema uit figuur 11.C.1 de voorgestelde wijzigingen aangebracht zijn, kunnen ook geen data meer ingelezen worden (van de schakelaars) met de instructie IN A,(F8). Ook hier komen dan weer de LoaD-instructies in aanmerking, die normaal gebruikt worden om data te lezen uit een RAM- of ROM-lokatie.
Voorbeelden : LD A,(nn), LD B,(HL) en LD E,(IY+d).

Door het nadeel van memory mapped IN/OUT dat 16 adreslijnen gecontroleerd moeten worden, loopt de verbinding met de randtoestellen meestal via de poorten. Waar meer dan 256 randtoestellen van elk 8 bit gestuurd moeten worden, kan memory mapped IN/OUT de aanvulling verzorgen.
Buiten de reeds bestudeerde instructies OUT (n),A en IN A,(n) voor de poorten, is er bij de Z-80 nog een hele reeks krachtige instructies beschikbaar, die nog besproken moeten worden. Het voordeel van de vele LD-instructies bij memory mapped IN/OUT wordt daardoor afgezwakt.

21.C. OUT (C),r

Naar de poort waarvan het adres in register C staat, wordt de inhoud van register r overgebracht.

Opcode	Mnemonic	Verklaring
ED79	OUT (C),A	Naar poort aangewezen door C de inhoud A
ED41	OUT (C),B	Naar poort aangewezen door C de inhoud B
ED49	OUT (C),C	Naar poort aangewezen door C de inhoud C
ED51	OUT (C),D	Naar poort aangewezen door C de .....
ED59	OUT (C),E	.....
ED61	OUT (C),H	.....
ED69	OUT (C),L	.....

Flags worden daarbij niet beïnvloed.

Waarden uit de registers B, D, E, H en L kunnen nu rechtstreeks naar de poorten gebracht worden, zonder de inhoud van A te beïnvloeden.
Deze instructie is vooral van nut als meerdere output-poorten aangesloten zijn. Eenzelfde instructie OUT (C),r, opgenomen in een lus, kan dan data naar de verschillende poorten brengen als in de lus ook C gewijzigd wordt.

21.D. IN r,(C)

Laadt in register r de data die geleverd worden door de poort waarvan het adres in C staat. De flags worden hier wel beïnvloed.

S Hoog als de ingevoerde waarde negatief is, anders laag
Z Hoog als de ingevoerde waarde 0 is, anders laag
P/V Hoog als van de ingevoerde waarde het aantal hoge bits even is, anders laag
N Wordt laag.

Opcode	Mnemonic	Verklaring
ED78	IN A,(C)	Laadt in A de data van de poort die door
ED40	IN B,(C)	C aangewezen wordt.
ED48	IN C,(C)	
ED50	IN D,(C)	Laadt in D de data van de poort die..enz
ED58	IN E,(C)	Laadt in E de data .....
ED60	IN H,(C)	Laadt in H de .....
ED68	IN L,(C)	Laadt in L.....
ED70	IN (C)	Stelt enkel de flags naargelang de data
		geleverd door de poort aangewezen door C

Voor het gebruik van het volgend programma moet het leds-schakelaars randtoestel (uit figuur 11.C.1) aangesloten zijn. Zodra het programma gestart wordt, laden alle registers zich met de door de schakelaars geleverde data.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	0E F8	P21D1	LD C,F8	Poortadres.
1802	ED78		IN A,(C)	Input naar A.
1804	ED40		IN B,(C)	Naar B.
1806	ED50		IN D,(C)	Naar D.
1808	ED58		IN E,(C)	
180A	ED60		IN H,(C)	
180C	ED68		IN L,(C)	
180E	ED48		IN C,(C)	Data van poort naar C
1810	F7		RST 30	Monitor.

Let erop dat register C als laatste geladen wordt. De reden zal wel duidelijk zijn.

21.E. Onderdrukking van contactdender

Door de snelheid waarmee P21D1 uitgevoerd wordt, zal de gebruiker geen gelegenheid krijgen om de schakelaars van stand te veranderen, om zo andere waarden in de verschillende registers te laden. Daarvoor zou tussen iedere IN (C),r een tijdsvertraging moeten komen. De Micro-

Prof zou dan wel moeten aanduiden wanneer de tijdsvertraging verlopen is, en de schakelaars zouden opnieuw versteld mogen worden. De tijdsvertraging zou zo groot gekozen moeten worden, dat de gebruiker de gelegenheid heeft om rustig de acht schakelaars te bedienen. Als dan slechts één schakelaar omgelegd moet worden, gaat er veel tijd verloren.
Een betere methode is dan ook, dat de gebruiker een signaal kan geven als de schakelaars opnieuw ingesteld zijn, en er nieuwe data ter beschikking staan.
Dit gebeurt in P21E1, waar op de user key gedrukt moet worden als de schakelaars gelezen mogen worden.
De user key is aangesloten op de als ingang geprogrammeerde klem PA6 van de 8255, die laag gemaakt wordt door het drukken op die toets.
Een probleem daarbij is de contactdender die optreedt bij het sluiten en het openen van het contact. Door de snelheid van het programma zouden toch weer alle registers met dezelfde data geladen worden.
In P21E1 wordt de user key gecontroleerd door de subroutine USERK, die ook de contactdender onderdrukt. De subroutine keert terug naar het hoofdprogramma als het contact weer openstaat. Dus na het loslaten van de toets.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	0E F8	P21E1	LD C,F8	Poortadres.
1802	CD 5018		CALL 1850	USERK
1805	ED78		IN A,(C)	Laad reg. A
1807	CD 5018		CALL 1850	USERK
180A	ED40		IN B,(C)	Laad reg. B
180C	CD 5018		CALL 1850	USERK
180F	ED50		IN D,(C)	Laad reg. D
1811	CD 5018		CALL 1850	USERK
1814	ED58		IN E,(C)	Laad reg. E
1816	CD 5018		CALL 1850	USERK
1819	ED60		IN H,(C)	Laad reg. H
181B	CD 5018		CALL 1850	USERK
181E	ED68		IN L,(C)	Laad reg. L
1820	CD 5018		CALL 1850	USERK
1823	ED48		IN C,(C)	Laad reg. C
1825	F7		RST 30	Monitor.
1850	F5	USERK	PUSH AF	
1851	C5		PUSH BC	
1852	DB 00	SLUIT	IN A,(00)	Wacht
1854	CB77		BIT 6,A	op sluiten
1856	20 FA		JR NZ,SLUIT	van kontakt.
1858	06 00	LAAG	LD B,00	Instellen lusteller.
185A	DB 00	HOOG	IN A,(00)	Is het kontakt
185C	CB77		BIT 6,A	gesloten ?
185E	28 F8		JR Z,LAAG	Ja, dan herinstellen.
1860	10 F8		DJNZ HOOG	Neen, dan aftellen.
1862	C1		POP BC	
1863	F1		POP AF	
1864	C9		RET	

De subroutine USERK blijft in de lus SLUIT lopen tot het contact sluit. Daarna wordt de lusteller ingesteld op dec. 256, en het contact opnieuw gecontroleerd. Zolang het contact gesloten blijft, wordt de lus-teller telkens opnieuw ingesteld.

Wordt een open contact vastgesteld, dan wordt de lus-teller met 1 verlaagd, natuurlijk zonder het opnieuw instellen van de lus-teller.
Als nu door dender, bij het loslaten van de toets, het contact weer even sluit, dan wordt de lus-teller weer op 256 dec. ingesteld.
Alleen nadat 256 maal na elkaar een open contact is vastgesteld, is B tot 00 verlaagd, en wordt de subroutine afgesloten.

Taak 1. Schrijf een programma dat de data van de schakelaars leest, en deze in de RAM vanaf adres 1900 plaatst. De invoer moet eindigen als tweemaal na elkaar dezelfde waarde wordt aangeboden. Daarna moet het ingevoerde blok, byte per byte, op de leds getoond worden.

Bij alarminstallaties moet vooral gecontroleerd worden, of alle bits van een ingangspoort laag blijven. Daarvoor is de nogal onbekende instructie IN (C), die alleen de flags stelt zonder een register te laden, heel geschikt.
Programma P21E2 controleert of alle schakelaars laag blijven. Zodra een schakelaar wordt omgezet, komt het programma in het alarmdeel, waar alleen de met de schakelaar overeenkomende led oplicht. Natuurlijk kunnen in het alarmdeel nog andere acties bijgeprogrammeerd worden.

Adres	Mach. taal	Label	Mnemonic	Verklaring.
1800	0E F8	P21E2	LD C,F8	Poortadres schakelaars
1802	97		SUB A	Stel A op nul
1803	ED79		OUT (C),A	om de leds te doven.
1805	ED70	LUS	IN (C)	Stel flags.
1807	28 FC		JR Z,LUS	Terug zolang Z hoog is
1809	ED78		IN A,(C)	Lees schakelaars in A.
180B	ED79		OUT (C),A	Naar leds.
180D	F7		RST 30	Monitor

Taak 2. Laat de led(s) van de schakelaar(s) die hoog staan, flikkeren.
Taak 3. Bij alarm moet ook een geluidsignaal opgewekt worden.

21.F. OUTI (OUT and Increment)

Brengt naar de poort, aangewezen door register C, de waarde uit de lokatie die aangewezen wordt door HL. Daarna wordt HL met 1 verhoogd, en B met 1 verlaagd.
Flags.
Z Wordt hoog als B voor de instructie 1 was, anders laag
N Wordt hoog.

Deze instructie is bijzonder geschikt om een blok data (maximum 256 bytes) naar een poort over te brengen.
Programma P21F1 brengt 10 (dec. 16) waarden uit de RAM naar de leds. Zonder oponthoud zou dit veel te snel gebeuren om waarneembaar te zijn. Daarom wordt in de lus een wachttijd opgenomen, zodat iedere waarde ongeveer 1 sec. zichtbaar blijft.

Adres	Mach. taal	Label	Mnemonic	Verklaring.
1800	0E F8	P21F1	LD C,F8	Poortadres leds.
1802	06 10		LD B,10	Lengte blok.
1804	21 3018		LD HL,1830	Begin blok.
1807	EDA3	LUS	OUTI	
1809	F5		PUSH AF	Bewaar flags.
180A	1B	TIJD	DEC DE	Blijf
180B	7A		LD A,D	hier
180C	B3		OR E	even
180D	20 FB		JR NZ,TIJD	wachten.
180F	F1		POP AF	Haal flags terug.
1810	20 F5		JR NZ,LUS	Herhaal tot B=0.
1812	F7		RST 30	Monitor.

Een printer heeft ook enige tijd nodig om de aangeboden data te verwerken. Daardoor kan P21F1 ook gebruikt worden om een tekst die in de RAM staat, naar een printer te brengen die aangesloten is op een poort. De vertragingstijd mag daarbij wel korter gekozen worden, maar er moet toch rekening mee gehouden worden, dat bij printers sommige taken langer duren dan andere.
Meestal leveren printers ook een signaal als de opgedragen taak uitgevoerd is, en nieuwe data aangeboden mogen worden. Als men van deze eigenschap gebruik wil maken, is er ook een aangepast programma nodig.

21.G. INI (IN and Increment)

Laadt de lokatie die wordt aangewezen door HL, met de waarde geleverd door de poort waarvan het adres in register C staat. Daarna wordt HL met 1 verhoogd en B met 1 verlaagd. Flags als bij 21.F. Deze instructie is bijzonder geschikt om een reeks waarden (max. 256 dec.), geleverd door een randtoestel (bv. ponsbandlezer) dat is aangesloten op een poort, in de RAM te laden.
Met P21G1 kunnen tien waarden van het randtoestel (hier de schakelaars) gelezen worden, die vanaf adres 1840 na elkaar in de RAM geladen worden.
Om de gebruiker de gelegenheid te geven de schakelaars in te stellen, is de subroutine SCAN in de lus van INI opgenomen. Deze subroutine blijft wachten tot een toets van het keyboard wordt ingedrukt, en brengt ondertussen de patronen uit DISBF op de display.
Telkens als de schakelaars ingesteld zijn op een nieuwe waarde, moet dus op een toets van het keyboard gedrukt worden.
Daar de subroutine SCAN de registers B en HL te niet doet, moeten ze tijdelijk op de stack bewaard worden.

Adres	Mach. taal	Label	Mnemonic	Verklaring.
1800	0E F8	P21G1	LD C,F8	Poortadres.
1802	06 10		LD B,10	Aantal invoeren.
1804	21 4018		LD HL,1840	Begin buffer.
1807	DD21 3018		LD IX,1830	DISBF0
180B	C5	LUS	PUSH BC	Bewaar BC
180C	E5		PUSH HL	Bewaar HL

```
1800  CD FE05          CALL 05FE     SCAN
1810  E1               POP HL
1811  C1               POP BC
1812  ED42             INI
1814  20 F5            JR NZ,LUS     Herhaal tot B=0.
1816  F7               RST 30        Naar monitor.

1830  00        DISBF0
1831  00
1832  37
1833  AE
1834  B5
1835  1F

1840  00        INPBF
```

Met één instructie meer kan op de leds getoond worden hoeveel waarden nog ingevoerd moeten worden.

```
1812  ED42             INI
1814  ED41             OUT (C),B     Naar leds.
1816  20 F3            JR NZ,LUS     Herhaal tot B=0.
1818  F7               RST 30        Monitor.
```

21.H. OUTD (OUT and Decrement)

Als OUTI, maar HL wordt met 1 verlaagd.
Met deze instructie kan een blok in omgekeerde volgorde naar het randtoestel gebracht worden.

Taak 1. Pas P21F1 aan, zodat de waarden uit de RAM in omgekeerde volgorde naar de leds worden gebracht.
Taak 2. Voer met de inmiddels verworven kennis taak 1 uit 21.E opnieuw uit.
Taak 3. Breid P21G1 uit, zodat op de displays 1 en 0 telkens de ingevoerde waarde verschijnt.

21.I. IND (IN and Decrement)

Als INI, maar HL wordt met 1 verlaagd.

Taak. Wijzig P21G1 om de invloed van IND aan te tonen.

21.J. INIR (IN Increment and Repeat)

Als INI, maar met de instructie JR NZ,... ingebouwd.
Met deze instructie kan een aantal waarden van een randtoestel (als deze ze snel genoeg kan leveren) zeer snel in de RAM geladen worden. Ook wordt een instructie uitgespaard. Een nadeel is, dat nu geen andere instructie of subroutine in de lus opgenomen kan worden.
Alleen als voorbeeld hoe de instructie gebruikt moet worden, laadt P21J1 100 (dec. 256) inputwaarden in de buffer.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	0E F8	P21J1	LD C,F8	Poortadres.
1802	06 00		LD B,00	Aantal invoeren.
1804	21 4018		LD HL,1840	Begin buffer.
1807	ED B2		INIR	
1809	F7		RST 30	Monitor.

Na uitvoering staat van adres 1840 tot 193F dezelfde waarde omdat het programma al uitgevoerd is voordat ook maar één schakelaar omgezet kan worden.

21.K. INDR (IN Decrement and Repeat)

Als IND, maar met ingebouwde JR NZ,...
In P21J1 kan INIR vervangen worden door INDR, maar het resultaat zal hetzelfde zijn. Wel wordt de lokatie in omgekeerde volgorde geladen.

21.L. OTIR (OuT Increment and Repeat)

Als OUTI, maar met ingebouwde JR NZ,...

21.M. OTDR (OuT Decrement and Repeat)

Als OUTD, maar met ingebouwde JR NZ,...

22. RESTART

22.A. Inleiding

Met de ReSTart-instructie die slechts één byte klein is, kunnen eveneens subroutines opgeroepen worden.
Deze mogelijkheid is echter beperkt tot subroutines die beginnen op de adressen 0000, 0008, 0010, 0018, 0020, 0028, 0030 en 0038.
Net als bij de CALL nn-instructie wordt bij RST eerst het lopende adres uit de PC op de stack geplaatst. Daarna wordt, naargelang de opcode, een van de bovenvermelde adressen in de PC geladen en de daar aanwezige instructie uitgevoerd.

Opcode	Mnemonic
C7	RST 00
CF	RST 08
D7	RST 10
DF	RST 18
E7	RST 20
EF	RST 28
F7	RST 30
FF	RST 38

Op de vermelde adressen zou dan telkens een subroutine kunnen beginnen, maar daar zich tussen ieder startadres slechts acht lokaties bevinden, is daar geen plaats genoeg voor.
Een mogelijkheid is, in dit gebied slechts enkele kleine subroutines te schrijven.
Voorbeeld : vanaf 0000 tot 001F, een tweede van 0020 tot 0037 en een grotere vanaf 0038. In dit geval mogen alleen de instructies RST 00, RST 20 en RST 38 gebruikt worden.
Om elke RST-instructie aan een behoorlijke subroutine te koppelen, kan op ieder RST-adres een JP nn geschreven worden, die laat overspringen naar de eigenlijke subroutine verderop in het geheugen.
De Micro-Professor maakt alleen gebruik van de volgende RST-instructies.

22.B. RST 00

Op adres 0000 begint de RESET routine van de Micro-Professor die :
— de 8255 programmeert,
— de led TONE dooft, en het signaal BREAK' hoog maakt,
— USERPC (uit reg. buffer) met 1000 of 1800 laadt naargelang waar de RAM begint,
— USERIF (uit reg. buffer) met 0000 laadt,
— USERSP (uit reg. buffer) met 1F9F laadt,
— en daarna het toetsenbord aftast.
RESET wordt uitgevoerd door het drukken op de toets RESET, en ook direct na het inschakelen van de spanning.
De instructie RST 00 kan ook gebruikt worden om een user program af te sluiten, maar heeft andere eigenschappen dan RST 30 die gewoonlijk als afsluiter wordt toegepast.
Om deze eigenschappen aan te tonen eerst een eenvoudig programma dat afgesloten wordt met RST 30.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 44	P22B1	LD A,44	
1802	01 4444		LD BC,4444	
1805	11 4444		LD DE,4444	
1808	21 4444		LD HL,4444	
180B	F7		RST 30	

Na uitvoering van P22B1 toont de display 180C XX. Met de toets REG is in al de betrokken registers de waarde 44 te vinden. De PC staat op 180C.
Hetzelfde programma wordt nu afgesloten met RST 00. Al de operands worden vervangen door 55, om het effect van het programma te kunnen nagaan.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	3E 55	P22B2	LD A,55	
1802	01 5555		LD BC,5555	
1805	11 5555		LD DE,5555	
1808	21 5555		LD HL,5555	
180B	C7		RST 00	

Na uitvoering verschijnt nu op de display uPF--1, net als na het drukken op de toets RESET. De PC staat op 1800, en als de registers A, BC, DE en HL nagekeken worden door middel van de toets REG, staat daar nog 44 in plaats van 55.
Dit komt omdat RST 00 de inhoud van de registers niet naar de registerbuffer (zie blz. 105) brengt. Met toets REG worden niet de registers zelf gelezen (dat is onmogelijk), maar de registerbuffer. Door het afsluiten van een user program met RST 00, is het niet meer mogelijk de inhoud van de registers te lezen.
Bij het inschakelen van de spanning (power-up) wordt automatisch RESET uitgevoerd, maar aangevuld met de subroutine INI (power-up INItialization) met startadres 03C1.
Het verschil is ook merkbaar op de display. Het drukken op toets RESET (alleen RESET routine) laat uPF--1 in één keer op de display komen. Bij power-up (RESET + INI) schuift dezelfde tekst van rechts op de display.

Hoe beslist de RESET routine of INI uitgevoerd moet worden of niet ? Daarvoor is de RAM-lokatie 1FE5 (POWERUP) genoemd) gereserveerd. Als daarin niet de waarde A5 staat, roept RESET de subroutine INI op. Een deeltaak van INI is, in de lokatie POWERUP de waarde A5 plaatsen. Daardoor zal INI pas opgeroepen worden na het inschakelen van de spanning, als in POWERUP een willekeurige waarde staat. Nadien zal de routine RESET daar de waarde A5 vinden, en INI niet laten uitvoeren.
De beslissing voor het oproepen van INI wordt genomen van adres 000F tot 0016. Het laden van de waarde A5 (PWCODE) in de lokatie POWERUP gebeurt van 03D3 tot 03D7 en 06B3 tot 06B5.
De frequentie van het BEEP-signaal, dat opgewekt wordt als een toets wordt ingedrukt, wordt bepaald door de inhoud van RAM-lokatie 1FF1 (FBEEP) ; de tijdsduur door de inhoud van de lokaties 1FF2 en 1FF3 (TBEEP). Het BEEP-signaal wordt alleen maar opgewekt als in lokatie 1FF0 (BEEPSET) de waarde 55 staat.
Het is een deeltaak van INI, deze lokaties met de juiste waarde te laden, wat gebeurt van 06B6 tot 06C7.
Deze kennis maakt het mogelijk om de monitor het zwijgen op te leggen bij het intoetsen, door een andere waarde in 1FF0 (BEEPSET) te laden.

De frequentie of/en de tijdsduur van het signaal kan nu eveneens gewijzigd worden, natuurlijk op voorwaarde dat in BEEPSET de waarde 55 staat.
De gewijzigde toestanden blijven behouden, ook nadat op de toets RESET is gedrukt, omdat INI niet meer uitgevoerd wordt.
Terugkeren naar de normale toestand kan door even de spanning te onderbreken, zodat INI weer uitgevoerd wordt die BEEPSET, BFEEP en TBEEP weer met hun normale waarde laadt. Natuurlijk gaat bij deze bewerking een eventueel programma in de RAM verloren.
Terugkeren naar de normale toestand zonder verlies van een programma kan door de inhoud van lokaties 1FE5 (POWERUP) te wijzigen en op de toets RESET te drukken. De RESET routine vindt dan een andere waarde dan A5 in POWERUP en laat INI uitvoeren, zodat BEEPSET, FBEEP en TBEEP opnieuw geladen worden.

Taak. Breid PA7F3 uit zodat bij het intoetsen van de tien waarden ook een geluid hoorbaar is, maar met een andere frequentie en duur. Nadat het programma beëindigd is moet BEEP weer zoals voorheen zijn.
Gebruik de subroutine BEEP na de subroutine SCAN.

22.C. RST 28

Deze subroutine wordt alleen gebruikt door de monitor voor het behandelen van een breakpoint. Voorlopig is hij niet van belang voor de gebruiker.

22.D. RST 30

Programmeert de 8255 (deze kan door het user program gewijzigd zijn) opnieuw.
Maakt het signaal BREAK' weer hoog, en dooft de led TONE.
Plaatst de inhoud van alle registers in de registerbuffer.
Controleert of de user SP naar een niet-RAM-lokatie wijst. Indien wel, dan wordt de melding ERR-SP op de display gebracht.
Gaat ook na of de user SP niet naar het gebied van de system stack wijst, om dan het bericht SYS-SP op de display te brengen.
Keert terug naar de monitor indien deze fouten niet voorkomen, en brengt de user PC met de bijbehorende data op de display.
Daar deze routine veel te groot is voor de acht lokaties tussen 0030 en 0037 begint hij vanaf 0066. Op 0030 en 0031 staat alleen een JR naar 0066.
Voor vele programma's is de instructie RST 30 de meest geschikte terminator.

22.E. RST 38

Haalt het adres op dat in de RAM-lokaties AFEE en AFEF (IM1AD) staat, en springt dan over naar dit adres.
Tijdens de power-up initialization is in IM1AD de waarde 0066 geplaatst. Daardoor heeft RST 38 hier hetzelfde effect als RST 30.
Door in IM1AD het adres van een door de gebruiker geschreven en veel toegepaste subroutine te plaatsen, kan deze opgeroepen worden met RST 38.
Door deze mogelijkheid kan P21E1 korter geschreven worden.

Adres	Mach. taal	Label	Mnemonic	Verklaring
1800	21 5018	P22E1	LD HL,1850	User subroutine.
1803	22 EE1F		LD (1FEE),HL	IM1AD
1806	0E F8		LD C,F8	Poortadres.
1808	FF		RST 38	USERK
1809	ED78		IN A,(C)	Laad reg. A
180B	FF		RST 38	
180C	ED40		IN B,(C)	
180E	FF		RST 38	
180F	ED50		IN D,(C)	
1811	FF		RST 38	
1812	ED58		IN E,(C)	
1814	FF		RST 38	
1815	ED60		IN H,(C)	
1817	FF		RST 38	
1818	ED68		IN L,(C)	
181A	FF		RST 38	
181B	ED48		IN C,(C)	
181D	F7		RST 30	
1850	F5	USERK	PUSH AF	
1851	C5		PUSH BC	
1852	DB 00	SLUIT	IN A,(00)	Wacht
1854	CB77		BIT 6,A	op sluiten
1856	20 FA		JR NZ,SLUIT	van kontakt.
1858	06 00	LAAG	LD B,00	Instellen lusteller.
185A	DB 00	HOOG	IN A,(00)	Is het kontakt
185C	CB77		BIT 6,A	gesloten ?
185E	28 F8		JR Z,LAAG	Ja, dan herinstellen.
1860	10 F8		DJNZ HOOG	Neen, dan aftellen.
1862	C1		POP BC	
1863	F1		POP AF	
1864	C9		RET	Terug.

23. DE CASSETTERECORDER

23.A. Inleiding

De Micro-Professor is voorzien van de nodige soft- en hardware om data uit de RAM, door middel van een normale cassetterecorder, op een gewone audio-cassette te plaatsen.
De cassetterecorder wordt via een passende kabel (verkrijgbaar in de elektronicawinkel) op de aansluitingen AER (AERphone) en MIC (MICrophone) aangesloten. 't Meest geschikt zijn cassettes met een goede kwaliteit, maar met een korte speelduur. Zelfs op een cassette van 2 × 15 min. is er plaats voor tientallen programma's. Een recorder met telwerk vergemakkelijkt het zoeken.
Een programma of een ander blok data dat op een cassette is geplaatst, wordt een *file* (rij, reeks) genoemd.
De Micro-Prof heeft de mogelijkheid om iedere file een eigen naam te geven, die vooraan de file op de band geschreven wordt.
Zowel tijdens het lezen als het schrijven op de cassette is het signaal akoestisch en visueel waar te nemen via de speaker (luidspreker) en de groene led (TONE).
Gebruik geen cassettes waar al muziek of gesprekken op staan. Alleen nieuwe of gewiste cassettes.
Bijna alle cassettes hebben een aanloopstrook waar niets op geschreven kan worden.

23.B. Data schrijven op de cassette

Om alle misverstanden te voorkomen zal de werkwijze voor het overbrengen van een programma naar de cassette door middel van voorbeelden gegeven worden.
Schrijf programma P16D1 in de RAM. Test het programma om er zeker van te zijn dat het goed ingevoerd is.
Sluit de recorder aan en plaats een lege cassette. Stel deze in tot net voorbij de aanloopstrook. Druk op de toets TAPE WR (TAPE WRite). Op de display verschijnt X.X.X.X.-F. Daarmee vraagt de Micro-Prof om de naam van de file. Toets voor dit programma 16D1 in, en druk op de toets +. Nu verschijnt X.X.X.X.-S. Hier moet nu het startadres van het programma ingetoetst worden, dus 1800, waarna weer op de toets + gedrukt moet worden. De display toont nu X.X.X.X.-E en vraagt daarmee naar het eindadres van het weg te schrijven blok. Programma P16D1 eindigt wel op adres 1807, maar ook de tabel met de tekst moet bewaard worden. Daarom wordt als eindadres 1825 ingetoetst.
Druk nu de toetsen RECORD en PLAY van de recorder in, en daarna de toets GO. De display dooft, maar de groene led en de speaker geven het signaal weer dat naar de recorder wordt gebracht. Na enkele seconden stopt het geluid, en komt op de display het eindadres van het overgebrachte blok data. Laat de recorder nog even doordraaien om zo een lege ruimte tussen de files te krijgen.
De cassette staat daarmee op de goede plaats om een volgende file te schrijven.
Ga nu op dezelfde manier te werk om de programma's P11F1 en P15F1 na file 17D1 te schrijven.
Iedere file bevat niet alleen het aangewezen blok data, maar ook de filename, het begin- en het eindadres van het blok en nog een waarde, *checksum* genoemd. De checksum bestaat uit de laatste twee digits van de som van alle bytes uit het blok.

23.C. Data lezen van de cassette

Om een file van de cassette te lezen hoeft de Micro-Prof alleen de filenaam te kennen. Waar in de RAM de gelezen data moeten komen wordt van de cassette gelezen, daar de file ook het begin- en het eindadres van het blok bevat.
De eerste file op de cassette is het eenvoudigst op te laden, dus hier file 16D1.
Schakel eerst de spanning van de Micro-Prof even uit zodat alle programma's uit de RAM verdwenen zijn. Spoel de cassette volledig terug. Plaats het volume van de recorder op maximum.
Druk op de toets TAPE RD (TAPE ReaD). Op de display verschijnt X.X.X.X.-F, waarmee naar de filenaam gevraagd wordt. De gewenste file heeft 16D1, wat nu ingetoets moet worden. Druk daarna op de toets GO, waardoor zes punten op de display verschijnen. Nu de toets PLAY van de recorder indrukken. Zodra de recorder iets leest wordt het doorgegeven aan de Micro-Prof, die het via de speaker en de groene led kenbaar maakt.
Na enkele toonveranderingen moet op de display de filenaam 16D1 verschijnen gedurende ruim 1 sec. Daarmee toont de Micro-Prof dat hij het begin van een file gevonden heeft. Daar het hier de gevraagde file is wordt deze in de RAM geladen. Het laden van de file wordt gekenmerkt door zes '—' tekens op de display. Zodra de file helemaal is geladen, verschijnt het laatste adres van het geladen blok op de display. Stop nu onmiddellijk de recorder. Test het opgeladen programma.

Als een file helemaal gelezen is, maakt de Micro-Prof weer de som van alle bytes van het blok, en vergelijkt deze met de checksum. Alleen als de som en de gelezen checksum gelijk zijn laat de Micro-Prof het laatste adres van het gelezen blok op de display verschijnen. Anders volgt de melding -Err , waarmee de gebruiker gewaarschuwd wordt dat in het opgeladen blok een of meerdere fouten aanwezig zijn.
Als een file die niet helemaal vooraan op de cassette staat, opgeladen moet worden, is de Micro-Prof in staat deze aan de hand van de filenaam op te zoeken.
Om P15F1 op te laden kan als volgt te werk gegaan worden :
Cassette helemaal terugspoelen.
Toets TAPE RD indrukken en 15F1 antwoorden. Op toets GO drukken. De toets PLAY van de recorder indrukken. Na enkele seconden verschijnt 16D1 op de display, maar wordt niet vervangen door de zes '—' tekens. Wel verschijnen weer de zes punten, wat erop wijst dat het zoeken voortgezet wordt. Na heel wat toonveranderingen komt de filename 11F1 op de display, die weer vervangen wordt door de zes punten. Even later komt 15F1, de gezochte file, op de display die nu wel vervangen wordt door de zes mintekens. De file wordt opgeladen. Nadat het einde van de file bereikt is komt het laatste adres van het programma op de display, waarna de recorder gestopt mag worden.
Ook hier kan de melding -Err verschijnen als de oplading niet perfect verlopen is.

Taak. De drie files die op de cassette aanwezig zijn moeten alle drie in de RAM komen. Het gebruik van de toets MOVE zal wel nodig zijn.

23.D. De hard- en software voor de recorder

Een byte overbrengen naar een randtoestel als de leds-schakelaarsopstelling gebeurt parallel. Dit betekent dat de acht bits tegelijkertijd overgebracht worden. Naar de cassetterecorder is dit niet mogelijk, omdat deze slechts één enkele ingangslijn heeft. Daarom moeten van elke byte de acht bits één voor één doorgegeven worden. Dit wordt *seriële overdracht* genoemd.

Bij de Micro-Prof worden deze bits naar buiten gebracht via de uitgang PC7 van de 8255. Dit signaal, dat tevens op de groene led en de speaker komt, gaat via enkele condensatoren en weerstanden (voor vormaanpassing) naar de uitgang MIC.
Voor de recorder is dit signaal net als een signaal van een gewone microfoon.
Het signaal dat de recorder levert in de stand PLAY komt in de Micro-Prof via de ingang EAR. De weerstand en de twee dioden zorgen voor het afzwakken van te sterke signalen. Twee smith-trigger-poorten verbeteren het signaal, dat via de ingang PA7 van de 8255 binnengehaald wordt.

De eenvoud van de hardware staat in tegenstelling met de moeilijkheidsgraad van de software. Een bit die laag is wordt uitgezonden als 8 perioden van 2 kHz gevolgd door 2 perioden van 1 kHz. Dit duurt 8 × 0.5 msec + 4 × 1 msec = 6 msec. Zie figuur 23.D.1.

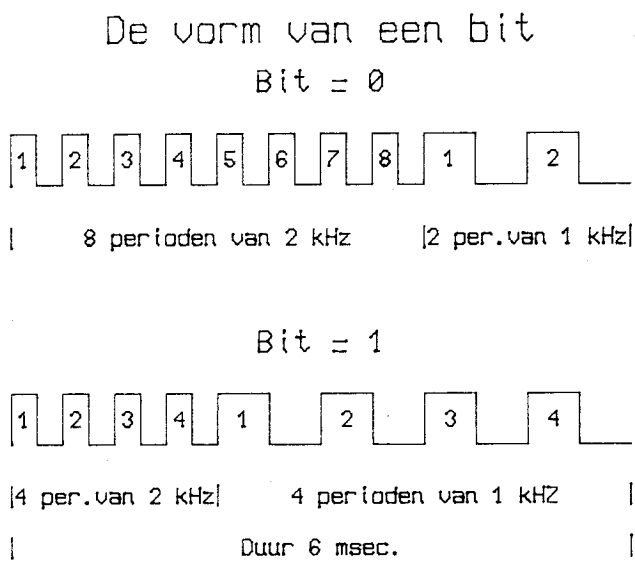


Fig. 23D1

Een hoge bit komt naar buiten als 4 perioden van 2 kHz gevolgd door 4 perioden van 1 kHz. De duur bedraagt 4×0.5 msec + 4×1 msec = 6 msec, dus even lang als een 0 bit.
Voordat bit 0 van een byte uitgezonden wordt, komt eerst een startbit, en na bit 7 volgt een stopbit. Beide in de vorm zoals net besproken, maar de startbit is altijd laag en de stopbit altijd hoog. Zie figuur 23.D.2.

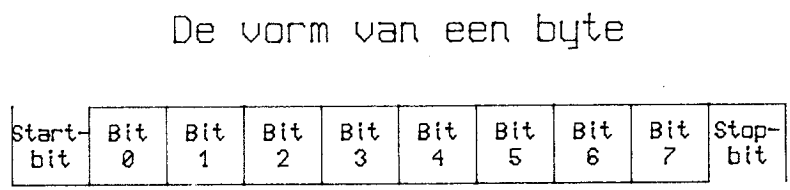


Fig. 23D2

Het uitzenden van 1 byte duurt dan ook (8+2) × 6 msec = 60 msec.
In 1 sec kunnen bijgevolg 1000/60 = 16.6 byte uitgezonden worden.
De routine GWT (Go Write Tape) die een blok naar de recorder laat overbrengen begint op adres 0324. Nadat alle parameters ingeladen zijn worden deze gecontroleerd (is startadres < eindadres ?) waarna de checksum berekend wordt.
Het uitzenden begint met een signaal van 1 kHz gedurende 4 sec, die het *leading synchro signal* genoemd wordt. Daarna wordt de filename, het startadres en het eindadres uitgezonden, die elk twee byte lang zijn. Ook de checksum volgt, maar deze is maar één byte groot. Vervolgens komt het middle synchro signal dat een frequentie heeft van 2 kHz en 2 sec duurt. Pas dan wordt het datablok uitgezonden, elke byte bit na bit met telkens een start- en een stopbit.
Dit alles wordt afgesloten met een tail synchro signal van 2 kHz gedurende 2 sec. Zie figuur 23.D.3.

De vorm van een file

Leader synchro signal	File-name	Start address	End address	Che. sum	Middle syn.sign	Data blok	Tail syn.sign
1 kHz 4 sec	2 byte	2 byte	2 byte	1 byte	2 kHz 2 sec	Afhankelijke lengte	2 kHz 2 sec

Fig. 23D3

Daarvoor maakt de routine GWT gebruik van verschillende, niet zo eenvoudige subroutines : o.a. SUM1, TONE1K, TONE2K, TAPEOUT en GETPTR.
Het binnenhalen van een file gebeurt door de routine GRP (Go Read Tape) met als startadres 035A.
Uit GRP kan worden geleerd dat het voldoende is als er 1000 van 4000 perioden waaruit het leading synchro signal bestaat, gelezen worden. Het hindert dus niet als de drie eerste sec van een file niet ontvangen worden. Ook niet als er gestart wordt in het midden van een vorige file. Daardoor is het niet nodig dat de tape nauwkeurig ingesteld wordt om een file in te laden.

24. INTERRUPTS

24.A. Inleiding

Een programma in uitvoering kan bij de Z-80 van buitenaf op vier verschillende manieren onderbroken worden.
Programma-onderbreking met de RESET-toets is al bekend. In het schema op sheet 1 is te zien, dat het RESET-contact via een hoekgetriggerde D-flip flop de ingang RESET' laag maakt.
Een programma kan eveneens onderbroken worden door de ingang BUSRQ' laag te maken. Dit gebeurt dan op het einde van de lopende machinecyclus, zelfs als dit niet het einde van de instructie is. De Z-80 brengt dan zijn databus, adresbus en controlbus in de high impedance state en maakt deze toestand naar buiten bekend door zijn uitgang BUSAK' (klem 23) laag te maken.
In deze toestand kan een uitwendig toestel rechtstreeks over het geheugen beschikken (DMA Direct Memory Access) om snel data uit te wisselen. DMA wordt alleen gebruikt bij uitgebreide systemen waar snel en vaak veel data overgebracht moeten worden. De Micro-Prof maakt van deze mogelijkheid geen gebruik. De ingang BUSRQ' ligt vast op een hoog niveau en de uitgang BUSAK' is nergens mee verbonden.
Ook de ingangen NMI' en INT' kunnen een programma onderbreken.
Door de ingang NMI' laag te maken onderbreekt men het programma onvoorwaardelijk als de lopende instructie is afgewerkt.
Met de ingang INT' is het iets ingewikkelder. De Z-80 beschikt ook nog over een Interrupt Flip Flop die met 'IFF' wordt aangeduid. De IFF kan met de instructies DI (Disable Interrupt opcode F3) en EI (Enable Interrupt opcode FB) gesteld worden.
Als de ingang INT' laag gemaakt wordt, zal het programma onderbroken worden nadat de lopende instructie is afgewerkt, maar alleen op voorwaarde dat de IFF in de stand 'enable' staat.
Daardoor kan een programma zelf bepalen in welk gedeelte het niet onderbroken wenst te worden.
Het gebeuren na het onderbreken van een programma door INT' is afhankelijk van de ingestelde Interrupt Mode (0, 1 of 2). Deze kan worden gekozen met de instructies IM 0 (ED46), IM 1 (ED56) en IM 2 (ED5E).
Met een CALL nn wordt een programma (synchroom) onderbroken van binnenuit om een subroutine uit te voeren. Met NMI' en INT' wordt een programma (asynchroon) van buitenaf onderbroken voor het uitvoeren van een interrupt servicing routine.

24.B. RESET'

Door het laag worden van de ingang RESET' veroorzaakt de Z-80 zelf de volgende gebeurtenissen :
Het interrupt-register I wordt op 00 gesteld. Het doel van register I wordt later besproken.
Ook het refresh-register R wordt met 00 geladen. Het R-register is alleen van belang als dynamische RAM is aangesloten.
De IFF wordt in de stand 'disable' gebracht.
De interrupt mode 0 wordt ingesteld.

De PC wordt met 0000 geladen.
Zolang RESET' laag blijft zijn de adresbus en de databus in hun high impedance state. Wordt RESET' weer hoog, dan krijgt de Z-80 de vrije loop, en daar de PC op 0000 staat wordt daar de programma-uitvoering gestart.
Bij de Micro-Prof staat op 0000 de RESET-routine, die de opstelling op gang brengt.
Bij het inschakelen van de spanning, is de condensator C9 nog ontladen. De eerste klokpuls maakt RESET' laag. Na een korte tijd is de condensator voldoende geladen om de D-ingang hoog te maken. De eerstvolgende dalende flank van de oscillator maakt RESET' hoog. Zoals gezegd start dan de routine RESET op adres 0000.
Met de RESET-toets kan ook de D-ingang van de flip flop laag gemaakt worden, waardoor bij de eerstvolgende dalende flank RESET' laag wordt.
De bedoeling van de D-flip flop is niet om contactdender weg te werken (dit gebeurt ook niet), maar voornamelijk het omgekeerde signaal RESET te krijgen voor de 8255.
Waar een Z-80-opstelling gebruikt wordt voor het sturen van een uitwendig proces (bv. lift, verkeerslichten) kan het systeem automatisch in werking komen bij het inschakelen van de spanning, indien het als bij de Micro-Prof de RESET'-ingang automatisch even laag gehouden wordt. Wel moet het programma in de EPROM dan beginnen op adres 0000.

24.C. Interrupt mode 1

Als in deze mode INT' laag wordt terwijl IFF in enable staat, wordt het programma na afwerking van de lopende instructie onderbroken, en voert de Z-80 zelf navolgende bewerkingen uit :
De IFF wordt in disable gezet.
De PC wordt op de stack geplaatst en opnieuw geladen met 0038.
Op dit adres moet nu de interrupt servicing routine beginnen of een jump-instructie naar deze routine, die dan ook uitgevoerd wordt.
Een subroutine moet afgesloten worden met een RET-instructie (RETurn from subroutine), en een interrupt routine met de instructie RETI (RETurn from Interrupt) met als opcode ED4D. De instructie RETI haalt een adres van de stack en plaatst dit in de PC. Daardoor wordt het hoofdprogramma voortgezet op de plaats waar het onderbroken werd.
Het schema op sheet 1 toont, dat de ingang INT' laag gemaakt worden kan met de drukknop INTR op het toetsenbord.
Voor proeven en demonstraties zijn programma's gekozen met langzame effecten, zodat ze goed te volgen zijn met de trage menselijke zintuigen.
Zo laat P24C1 een opgelichte led (van het leds-schakelaars randtoestel) in kringloop naar links schuiven.

Adres	Mach. taal	Label	Menmonic	Verklaring
1800	0E F8	P24C1	LD C,F8	Poortadres.
1802	16 01		LD D,01	Startwaarde.
1804	ED51	LEDS	OUT (C),D	Naar leds.
1806	2B	TIJD	DEC HL	Wacht
1807	7C		LD A,H	hier
1808	B5		OR L	even.
1809	20 FB		JR NZ,TIJD	
180B	CB02		RLC D	Schuif naar links.
180D	18 F5		JR LEDS	Herhaal.

Bij uitvoering van P24C1 kan vastgesteld worden dat de toets INTR niet de minste invloed heeft. Zo vreemd is dit niet, daar bij het opstarten (of het drukken op toets RESET) de IFF in disable is gezet. Daardoor wordt een interrupt-aanvraag natuurlijk geweigerd.
Het programma kan zelf de IFF in enable zetten om zo een interrupt te laten aanvaarden. Aangezien hier de mode 1 bestudeerd wordt, en door het opstarten mode 0 is ingesteld, moet ook nog de instructie IM 1 vooraan in het programma opgenomen worden.
Programma P24C2 is als P24C1, maar met de besproken uitbreiding. Met de toets IN kan P24C1 uitgebreid worden tot P24C2.

Adres	Mach. taal	Label	Menmonic	Verklaring
1800	ED 56	P24C2	IM 1	Stel Int.mode 1 in.
1802	FB		EI	Maak int. mogelijk.
1803	0E F8		LD C,F8	Poortadres.
1805	16 01		LD D,01	Startwaarde.
1807	ED51	LEDS	OUT (C),D	Naar leds.
1809	2B	TIJD	DEC HL	Wacht
180A	7C		LD A,H	hier
180B	B5		OR L	even.
180C	20 FB		JR NZ,TIJD	
180E	CB 02		RLC D	Schuif naar links.
1810	18 F5		JR LEDS	Herhaal.

Programma P24C2 kan nu wel onderbroken worden met de toets INT. De PC wordt dan geladen met 0038, maar bij de Micro-Prof staan daar de instructies die de PC opnieuw laden met het adres dat in de lokaties 1FEE en 1FEF (IM1AD) staat. In 22.E. hebben we gezien dat daar het adres 0066 staat, zodat de uitvoering op deze plaats voortgezet wordt. Daar begint de routine die ook gebruikt wordt door RST 30 (zie 22.C).
Het indrukken van toets INT onderbreekt het programma, waarbij de Micro-Prof naar de monitor terugkeert zonder de inhoud der registers te verliezen.

Dit is natuurlijk nog niet de typische toepassing van een interrupt. Het programma wordt wel onderbroken, maar er wordt niet overgegaan naar een interrupt routine en daarna teruggekeerd naar het hoofdprogramma.
Om dit alles te kunnen volgen is een, weer zeer langzame, interrupt routine gekozen die de leds eenmaal naar rechts doet vollopen. Deze is geschreven vanaf adres 1830. Dit adres moet nu in de lokaties IM1AD geschreven worden, wat vooraan in P24C3 gebeurt. Als na de interrupt routine het hoofdprogramma voortgezet wordt kan dit alleen perfect verlopen, als de gebruikte registers weer dezelfde inhoud hebben als voor de interrupt. Daarop is de eerste en voorlaatste instructie van de interrupt routine EXX. De waarden van het hoofdprogramma worden tijdens de interrupt routine in de priemregisters bewaard.

Adres	Mach. taal	Label	Menmonic	Verklaring
1800	21 3018	P24C3	LD HL,1830	Adres interrupt rout.
1803	22 EE1F		LD (1FEE),HL	naar IM1AD
1806	ED56		IM 1	Kies Int.mode 1.
1808	FB		EI	Enable Interrupt
1809	0E F8		LD C,F8	Poortadres leds.
180B	16 01		LD D,01	Startwaarde.
180D	ED51	LEDSH	OUT (C),D	Naar leds.
180F	2B	TIJDH	DEC HL	Wacht

```
1810  7C          |        LD A,H       |      hier
1811  B5          |        OR L         |           even.
1812  20 FB       |        JR NZ,TIJDH  |
1814  CB02        |        RLC D        | Schuif naar links.
1816  18 F5       |        JR LEDSH     | Herhaal.


1830  D9          | I24C3  EXX          | Naar priemregisters.
1831  9F          |        SUB A        | Clear carry.
1832  0E F8       |        LD C,F8      | Poortadres leds.
1834  16 80       |        LD D,80      | Startwaarde.
1836  ED51        | LEDSI  OUT (C),D    | Naar leds.
1838  2B          | TIJDI  DEC HL       | Wacht
1839  7C          |        LD A,H       |      hier
183A  B5          |        OR L         |           even.
183B  20 FB       |        JR NZ,TIJDI  |
183D  CB2A        |        SRA D        | Vollopen naar rechts.
183F  30 F5       |        JR NC,LEDSI  | Herhaal tot Car.= 1.
1841  D9          |        EXX          | Van priemregisters.
1842  ED4D        |        RETI         | Terug naar hoofdprogr
```

Let erop, welke led opgelicht is op het ogenblik dat de onderbreking aangevraagd wordt door op toets INTR te drukken.
Als de interrupt routine uitgevoerd is en alle leds branden, moet het schuiven naar links voortgezet worden met de volgende led.
Het testen van P24C3 leidt tot de conclusie, dat de interrupt routine slechts eenmaal uitgevoerd kan worden. Ook na het nog eens starten van het programma blijkt opnieuw dat slechts één interrupt aanvaard wordt.
Hoewel onverwacht, is dit effect heel normaal. Zodra de interrupt routine aanvaard wordt schakelt IFF in disable, zodat een volgende interrupt geweigerd wordt.
Door de IFF weer in enable te zetten net voordat de interrupt routine verlaten wordt, zal een volgende aanvraag wel worden aanvaard.
De interrupt routine hoeft slechts weinig te worden aangepast.

```
1842  FB          |        EI           | Enable interrupt.
1843  ED4D        |        RETI         | Terug naar hoofdprogr
```

Als de toets INT langdurig ingedrukt wordt, zal na het beëindigen van de interrupt routine deze hervat worden zonder dat naar het hoofdprogramma teruggekeerd wordt. Daaruit is af te leiden, dat de ingang INT' reageert op het laag zijn en niet op een dalende flank.
Hoewel het hoofdprogramma nu onbeperkt onderbroken kan worden, heeft tijdens de interrupt routine zelf de toets INTR geen invloed. Om ook de interrupt routine zelf nog eens te kunnen onderbreken moet de IFF eerder in enable gezet worden.
Dit is gedaan in de interrupt routine I24C4, die samen met het hoofdprogramma P24C3 gebruikt kan worden.

```
Adres | Mach. taal | Label | Menmonic      | Verklaring
======|============|=======|===============|====================
1830  | D5         | I24C4 | PUSH DE       | Bewaar DE op stack.
1831  | 9F         |       | SUB A         | Clear carry.
1832  | 16 80      |       | LD D,80       | Startwaarde.
1834  | ED51       | LEDSI | OUT (C),D     | Naar leds.
```

```
1836  2B       TIJDI  DEC HL      Wacht
1837  7C              LD A,H            hier
1838  B5              OR L                    even.
1839  20 FB           JR NZ,TIJDI
183B  FB              EI          Enable interrupt.
183C  CB2A            SRA D       Vollopen naar rechts.
183E  30 F4           JR NC,LEDSI Herhaal tot vol.
1840  D1              POP DE      DE terug van stack.
1841  ED4D            RETI        Terug naar hoofdprog.
```

De registers uit het hoofdprogramma kunnen nu niet meer bewaard worden in de priem-registers, omdat ze door een interrupt tijdens de interrupt routine te niet gedaan worden. Daarom wordt hier bij iedere interrupt het registerpaar DE op de stack geplaatst, waar ook een onderbroken interrupt routine op de onderbrekingsplaats wordt voortgezet.
De instructie EI mag niet voor de tijdsvertraging geplaatst worden, anders zou bij het drukken op de toets INT een zeer groot aantal interrupts aanvaard worden. Daarbij zou telkens DE op de stack geplaatst worden, totdat deze aangroeit tot in de interrupt routine zelf.

Taak 1. Als een onderbreking aangevraagd wordt tijdens de interrupt routine moet een tweede interrupt routine zorgen, dat eenmaal leeglopen naar links wordt uitgevoerd.

In de vorige programma's is de Z-80 meer dan 99 % van de tijd bezig met tijdsvertragings-lussen. Om nu ook tekst op de display te laten verschijnen moet in de vertragings-lussen de subroutine SCAN1 opgenomen worden. Deze subroutine zal slechts enkele tientallen malen opgeroepen moeten worden om dezelfde tijdsvertraging te krijgen. Voor het aftellen is dan ook een enkel register voldoende die zelfs met een lagere waarde geladen moet worden. Daarmee is dan ook de mogelijkheid aanwezig om de tijdsvertraging voor hoofdprogramma en interrupt routine verschillend te maken.

```
Adres  Mach. taal   Label  Menmonic        Verklaring
=====  ===========  =====  ==============  ======================
1800   21 3018      P24C5  LD HL,1830      Adres inter.routine.
1803   22 EE1F             LD 1FEE,HL      Naar IM1AD.
1806   DD21 5018           LD IX,1850      DISBF0.
180A   ED56                IM 1            Instellen Int.mode 1.
180C   0E F8               LD C,F8         Poortadres.
180E   16 01               LD D,01         Startwaarde.
1810   06 40        LUSH   LD B,40         Tijdinstelling.
1812   ED51                OUT (C),D       Naar leds.
1814   F3           TIJDH  DI              Disable Interrupt.
1815   CD 2406             CALL 0624       SCAN1.
1818   FB                  EI              Enable Interrupt.
1819   10 F9               DJNZ TIJDH      Herhaal tot B = 0.
181B   CB02                RLC D           Rotate Left Circ. D.
181D   18 F1                JR LUSH        Blijf herhalen.


1830   D5           I24C5  PUSH DE         Bewaar DE op stack.
1831   97                  SUB A           Clear carry.
1832   16 80               LD D,80         Startwaarde.
1834   06 20        LUSI   LD B,20         Tijdinstelling.
1836   ED51                OUT (C),D       Naar leds.
1838   CD 2406      TIJDI  CALL 0624       SCAN1.
```

183B	10 FB		DJNZ TIJDI	Herhaal tot B = 0.
183D	CB2A		SRA D	Shift Right Arith.D
183F	30 F3		JR NC,LUSI	Herhaal tot vol.
1841	D1		POP DE	Haal DE van stack.
1842	06 20		LD B,20	Tijdinstelling.
1844	FB		EI	Enable interrupt.
1845	ED4D		RETI	Terug naar hoofdpro.
1850	BD	DISBF0		
1851	30			
1852	9B			
1853	BA			
1854	36			
1855	AE			

De subroutine SCAN1 is een niet-eenvoudige en uitgebreide routine die alle registers gebruikt, en bovendien de registers uitwisselt met de priemregisters. Als zo'n routine onderbroken wordt is het zeer moeilijk, alle registers hun oorspronkelijke waarde terug te geven na de interrupt routine. Daarom is het niet aan te raden SCAN1 te onderbreken. Met een instructie DI net vóór SCAN1 en een EI direct na SCAN1 wordt het onderbreken uitgesteld tot deze subroutine volledig afgehandeld is.
De tekst op de display kan ook aangepast worden, alnaargelang het hoofdprogramma of de interrupt routine loopt.

Adres	Mach. taal	Label	Menmonic	Verklaring
1830	DDE5	I24C6	PUSH IX	Bewaar adres tekst H
1832	DD21 5618		LD IX,1856	Adres tekst InTErU
1836	D5		PUSH DE	Bewaar DE op stack.
1837	97		SUB A	Clear carry.
1838	16 80		LD D,80	Startwaarde.
183A	06 20	LUSI	LD B,20	Tijdsinstelling.
183C	ED51		OUT (C),D	Naar leds.
183E	CD 2406	TIJDI	CALL 0624	SCAN1.
1841	10 FB		DJNZ TIJDI	Herhaal tot B = 0.
1843	CB2A		SRA D	Shift Right Arithm.D
1845	30 F3		JR NC,LUSI	Herhaal tot vol.
1847	D1		POP DE	Haal DE van stack.
1848	06 20		LD B,20	Tijdinstelling.
184A	DDE1		POP IX	Haal adr.tekst HooFdP
184C	FB		EI	Maak inter. mogelijk.
184D	ED4D		RETI	Terug naar hoofdprogr
1850	1F	P		
1851	B3	d		
1852	0F	F		
1853	A3	o		
1854	A3	o		
1855	37	H		
1856	B5	U		
1857	03	r		
1858	8F	E		
1859	07	t		
185A	23	n		
185B	30	I		

Door de interrupt-mogelijkheden kan het systeem (schijnbaar) meerdere taken tegelijkertijd uitvoeren. Een voorbeeld zal aantonen dat dit niet zo moeilijk is, maar dat toch altijd onverwachte problemen opduiken die een oplossing vragen.
Stel dat de Micro-Prof een reclameverlichting moet sturen (hier de leds-schakelaars interface) maar ook moet tellen, hoeveel maal een contact gesloten wordt en dit moet weergeven op de display. Het contact kan er een zijn van een mechanisch toestel dat telkens, wanneer een bewerking uitgevoerd is een contact sluit (voorbeeld : fotokopieermachine). Het contact moet dan telkens INT' even laag maken. Voor het testen van het programma kan de toets INT dit contact vervangen.
Als hoofdprogramma kan P24C5 dienst doen. Voor het bijhouden van de telling wordt lokatie 184F gekozen, net boven de display-buffer. De nieuwe interrupt routine moet dan telkens lokatie 184F met 1 verhogen en de nieuwe waarde omvormen tot twee patronen die naar DISBF1 en DISBF0 moeten worden gebracht. Dit laatste kan worden toevertrouwd aan de subroutine HEX7SG. Deze subroutine vereist wel dat de waarde in A staat en dat HL naar DISBF0 wijst.

Adres	Mach. taal	Label	Menmonic	Verklaring
1830	21 4F18	I24C7	LD HL,184F	Adres teller.
1833	34		INC (HL)	Verhoog teller.
1834	7E		LD A,(HL)	Teller in A.
1835	23		INC HL	Wijs DISBF0 aan.
1836	CD 7806		CALL 0678	HEX7SG.
1839	ED4D		RETI	Terug naar hoofdprog.
184F	00	TELLER		
1850	00	DISBF0		
1851	00			
1852	00			
1853	00			
1854	00			
1855	00			

Bij het testen van het programma met de nieuwe interrupt routine komen, zoals meestal bij het ontwerpen, onvolmaaktheden aan het licht. De leds schuiven wel en de display wordt verhoogd als op INT gedrukt wordt, maar niet, zoals gewenst, één voor één.
Zolang de toets INTR ingedrukt blijft, wordt de display in een hoog tempo verhoogd. Zelfs bij de kortst mogelijke aanslag is de verhoging groter dan 1.
Als de fout ontdekt is, is de verklaring gemakkelijk te geven. Na het terugkeren in het hoofdprogramma wordt de IFF weer in enable geplaatst, en als INT' nog laag is wordt weer teruggekeerd naar de interrupt routine waar de display weer verhoogd wordt.
Een eenvoudige oplossing is, de interrupt routine iets langer te laten duren, zodat het contact al open is als naar het hoofdprogramma teruggekeerd wordt. In de bijgevoegde tijdsvertraging kan het beste weer de subroutine SCAN1 opgenomen worden zodat de display gevoed blijft.

Adres	Mach. taal	Label	Menmonic	Verklaring
1830	C5	I24C8	PUSH BC	Bewaar B op stack.
1831	21 4F18		LD HL,184F	Adres TELLER.
1834	34		INC (HL)	Verhoog TELLER.
1835	7E		LD A,(HL)	TELLER in A.
1836	23		INC HL	Adres DISBF0.

```
1837  CD 7806          CALL 0678   HEX7SG.
183A  06 10            LD B,10     Tijdinstelling.
183C  CD 2406    TIJDI CALL 0624   SCAN1.
183F  10 FB            DJNZ TIJDI  Herhaal tot B = 0.
1841  C1               POP BC      Haal B van stack.
1842  ED4D             RETI        Terug naar hoofdprog.

184F  00         TELLER
1850  00         DISBF0
1851  00
1852  00
1853  00
1854  00
1855  00
```

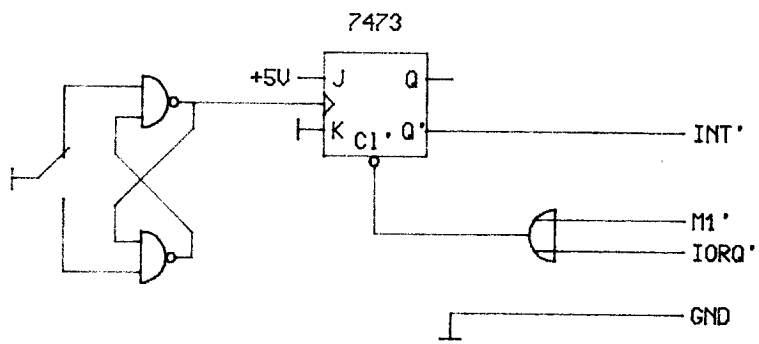
Daar 124C8 register B op 00 brengt, zou bij het terugkeren in het hoofdprogramma de eerstvolgende tijdsvertraging voor de leds zeer lang kunnen duren. Daarom wordt het registerpaar BC op de stack bewaard tijdens de interrupt routine en teruggehaald, voordat naar het hoofdprogramma wordt teruggekeerd.
Als INT' laag gemaakt wordt door een mechanisch bediend contact moet in 124C8 de tijdsvertraging ingesteld worden naar de duur, dat het contact gesloten blijft.

Taak 2. Breid 124C8 uit voor het tellen tot FFFF.

24.D. Interrupt acknowledge (bevestiging)

De oplossing met 124C8 kan niet perfect genoemd worden. De display wordt alsmaar verhoogd als het contact langer gesloten blijft dan de vertragingstijd uit de interrupt routine. Dus geen verhoging per contactbediening.
Als een interruptaanvraag aanvaard wordt, deelt de Z-80 dat aan de buitenwereld mee door heel kort de uitgangen IORQ' en M1' beide laag te maken. Bij I/O-bewerkingen zijn deze twee uitgangen nooit beide laag.
Met deze eigenschap en wat hardware zoals in figuur 24.D.1 is een betere oplossing te verkrijgen.

Fig. 24D1



De bevelpuls moet nu wel dendervrij zijn, wat bereikt wordt met een wisselcontact en een R-S flip flop. De dalende flank van deze puls maakt de uitgang Q' van de hoekgetriggerde JK-flip flop laag en daarmee ook de ingang INT' van de Micro-Prof. Als de IFF in enable staat aanvaardt de Z-80 de interrupt en bevestigt de aanvraag door M1' en IORQ' beide laag te maken. Dit doet ook even de RESET'-ingang van de JK-flip flop laag worden die reset en daarmee de ingang INT' weer hoog maakt.
Hoelang het contact ook gesloten blijft, op INT' komt altijd een zeer korte puls die de interrupt routine ook maar een enkele maal zal laten uitvoeren. De eenvoudige interrupt routine 124C7 is hier voldoende om een perfecte verhoging van de display per toetsbediening te krijgen.

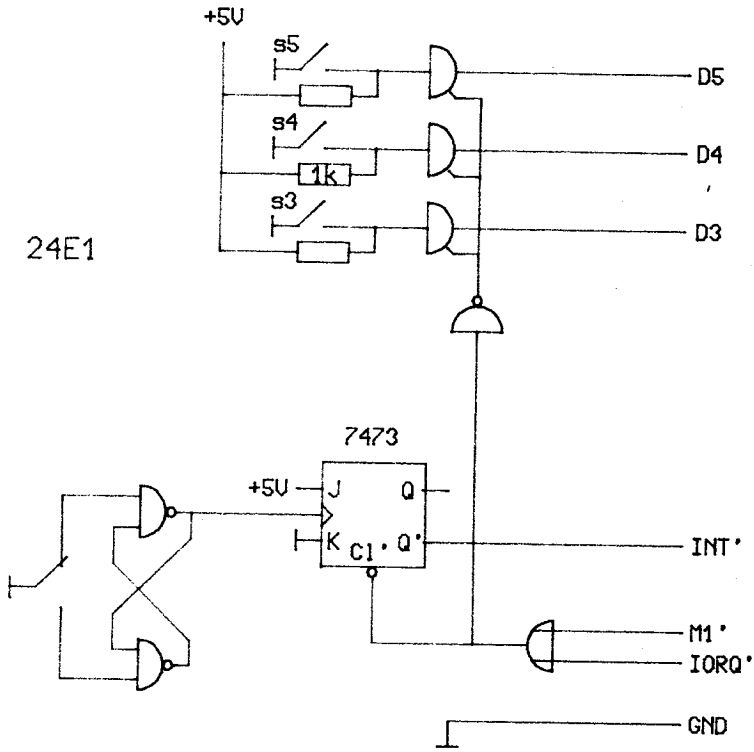
24.E. Interrupt mode 0

Bij het opstarten stelt de Z-80 zelf de interrupt mode 0 in.
Vanuit mode 1 of mode 2 kan naar interrupt mode 0 overgegaan worden door de instructie IM 0 (ED46).
Als IFF in enable staat en INT' laag wordt, werkt de Z-80 de lopende instructie af. Net zoals bij interrupt mode 1 wordt dan de IFF in disable geplaatst en de uitgangen M1' en IORQ' beide even laag gemaakt ter bevestiging, dat de onderbrekingsaanvraag ontvangen is.
De volgende gebeurtenis is wel heel bijzonder. De data die op de databus staan worden als een opcode rechtstreeks in het instructieregister gebracht en uitgevoerd. Met een voorbeeld wordt dit duidelijk gemaakt.
Laad interrupt routine 124C8 en P24C5 in de RAM en sluit het leds-schakelaarsrandtoestel aan. In P24C5 moet de instructie IM 1 vervangen worden door IM 0 (ED 46). Het hoofdprogramma heeft natuurlijk hetzelfde effect op de leds. Als op INTR gedrukt wordt, wordt het hoofdprogramma onderbroken zoals boven beschreven is. Op dit ogenblik is niets verbonden met de databus, alle datalijnen staan hoog waardoor in het instructieregister de waarde FF komt. Dit is de opcode van RST 38, die ook uitgevoerd wordt. Het drukken op INTR heeft in dit geval hetzelfde effect als een RST 38. Daarbij wordt eerst de PC op de stack geplaatst en dan opnieuw geladen met 0038, vanwaar de uitvoering wordt voortgezet.
Door van buitenaf een andere waarde op de databus te plaatsen op het ogenblik dat deze in het instructieregister geladen wordt, kan zo de opcode van een andere instructie (meestal ook een ReSTart) in het instructieregister komen. Deze andere RST-instructie kan dan ook een tweede interrupt routine laten uitvoeren. Daarmee ontstaat de mogelijkheid, van buitenaf te bepalen welke routine moet worden uitgevoerd als de ingang INT' laag wordt.
Om de hardware die dit mogelijk moet maken te ontwerpen, moet eerst aandacht besteed worden aan de opcodes van de acht RST-instructies.

Mnemonic	Opcode in hexa	Opcode in binair Bits 76 543 210
RST 00	C7	11 000 111
RST 08	CF	11 001 111
RST 10	D7	11 010 111
RST 18	DF	11 011 111
RST 20	E7	11 100 111
RST 28	EF	11 101 111
RST 30	F7	11 110 111
RST 38	FF	11 111 111

De tabel toont duidelijk dat bij de acht opcodes de bits 7, 6, 2, 1 en 0 altijd hoog zijn, en alleen de bits 5, 4 en 3 verschillen naargelang de instructie.
Daaruit volgt dat de hardware alleen de bits 5, 4 en 3 van de databus moet kunnen beïnvloeden om de acht opcodes in het instructieregister te kunnen drukken. Het schema daarvoor is weergegeven in figuur 24.E.1.

Fig. 24E1



Het onderste deel van de schakeling komt overeen met F24D1 voor interrupt acknowledge. Als na een interrupt-aanvraag de Z-80 de ontvangst bevestigt door M1' en IORQ' laag te maken, wordt de JK-flip flop teruggezet (INT' wordt weer hoog). Tegelijkertijd worden de drie three states actief, en brengen de stand van de schakelaars over op de databus. Daardoor wordt een van de acht opcodes (afhankelijk van de stand van de schakelaars) in het instructieregister geladen en uitgevoerd.
Voor het testen van deze eigenschappen kan de proefschakeling uit figuur 24.E.1 aangesloten worden op de bovenste connector in een afzonderlijke 5V-bron, maar vergeet niet dat de min ook met de GND-klem op de connector verbonden moet worden.
Als software is P24E1 voldoende, een vereenvoudigde versie van P24C5 en I24C7. In het hoofdprogramma is de instructie voor het sturen van de leds weggelaten, en instructie IM 1 vervangen door IN 0. Het hoofdprogramma heeft hier alleen als taak de inhoud van DISBF naar de displays te brengen. De interrupt routine moet alleen de TELLER (lokatie 184F) verhogen.

```
Adres  Mach. taal  Label  Menmonic        Verklaring
=====  ==========  =====  ==============  ======================
1800   21 3018     P24E1  LD HL,1830      Adres intr.routine.
1803   22 EE1F            LD 1FEE,HL      Naar IM1AD.
1806   DD21 5018          LD IX,1850      Adres DISBF0.
180A   ED46                IM 0            Interrupt mode 0.
180C   F3          LUS    DI
180D   CD 2406             CALL 0624       SCAN1.
1810   FB                  EI
1811   18 F9               JR LUS          Herhaal.


1830   21 4F18     I24E1  LD HL,184F      Adres TELLER.
1833   34                 INC (HL)        Verhoog TELLER.
1834   7E                 LD A,(HL)       TELLER in A.
1835   23                 INC HL          Wijs naar DISBF0.
1836   CD 7806            CALL 0678       HEX7SG.
1839   ED4D               RETI            Terug naar hoofdprog.

184F   00          TELLER
1850   00          DISBF0
1851   00

1855   00
```

Met de drie schakelaars open veroorzaakt het bedienen van de drukknop een onderbreking die in het instructieregister de opcode FF laadt.
De RST 38 wordt uitgevoerd en de display met 1 verhoogd.
Met drie gesloten schakelaars komt na het bedienen van de drukknop de opcode C7 (1100 0111) in het instructieregister en wordt de RST 00 uitgevoerd. Bij de Micro-Prof begint op adres 0000 de RESET-routine, zodat hetzelfde effect verkregen wordt als door het drukken op de RESET-toets. Op de display verschijn uPF--1.
Als S5 en S4 gesloten zouden zijn, zou CF in het instructieregister gedrukt worden met als gevolg een RST08. Daar bij de Micro-Prof op adres 0008 geen routine begint, heeft het geen zin deze proef uit te voeren.
Op adres 0030 begint wel een afzonderlijke routine. Deze kan van buitenaf opgeroepen worden als alleen S3 gesloten is. De bevestiging van de Z-80 drukt dan F7 in het instructieregister dat een RST30 laat uitvoeren. Het gevolg van deze routine is bekend : onderbreking van het programma met behoud van de inhoud van de registers.
Bij systemen waarin op ieder restart-adres een andere routine begint (of naar een andere routine wordt verwezen) kunnen van buitenaf acht verschillende interrupt routines gekozen worden voor de onderbreking. Op zichzelf is dit niet zo belangrijk, maar daarmee ontstaat de mogelijkheid, acht verschillende toestellen elk hun eigen interrupt routine te laten aanvragen. Door de in de Micro-Prof aanwezige EPROM is dit hier niet mogelijk.

24.F. Interrupt mode 2

Dit is de krachtigste mode, waarbij meerdere toestellen elk een onderbreking kunnen aanvragen om hun eigen interrupt routine te laten uitvoeren. Daarvoor moet in de RAM een tabel aangelegd worden met de startadressen (interrupt vectors) van de verschillende interrupt routines. Een voorbeeld van zo'n interrupt-vector-tabel zal dit verduidelijken.

Stel, dat een viertal interrupt routines gebruikt worden die beginnen op de adressen 1880, 18A0, 1900 en 1940. De interrupt-vector-tabel kan er dan uitzien als in figuur 24.F.1.

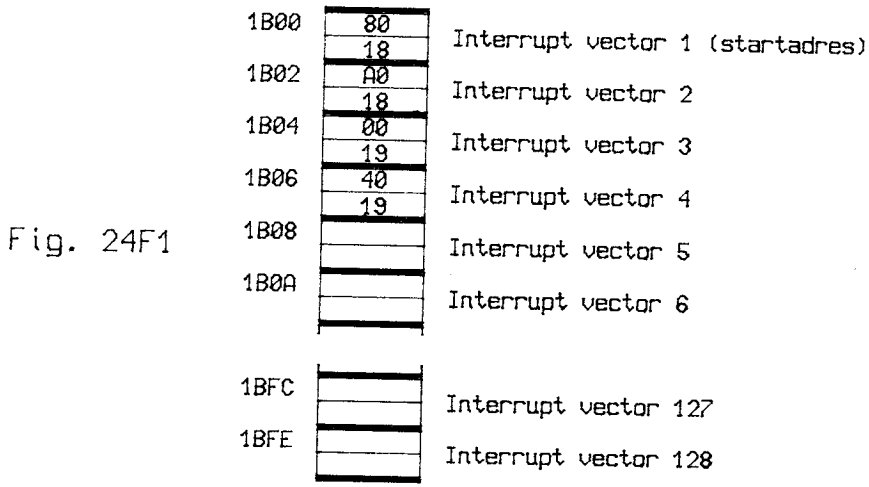


Fig. 24F1

De tabel mag op iedere willekeurige plaats in de RAM geschreven worden. De enige beperking is, dat de tabel binnen eenzelfde page (bladzijde) blijft. Een page is dit deel van de RAM waarvan de eerste twee digits van de adressen dezelfde zijn. Zo loopt page 18 van 1800 tot 18FF, page 1B van 1B00 tot 1BFF.
Daar iedere vector twee byte in beslag neemt is er plaats voor 128 vectors in de tabel, en kunnen evenveel randtoestellen hun eigen interrupt routine aanvragen. Als er minder dan 128 vectors zijn hoeven deze niet noodzakelijk na elkaar of vooraan in de page te staan.
Om de Z-80 te laten weten waar de interrupt-vector-tabel staat, moet het nummer van de page in het interrupt-register staan. Voor de tabel uit figuur 24.F.1 moet dus in register I de waarde 1B geladen worden.
Een toestel dat een interrupt aanvraagt, en daarbij interrupt routine 4 wil laten uitvoeren, moet daarvoor INT' laag maken en daarbij de waarde 06 op de databus plaatsen. De Z-80 weet daardoor dat de vector op adres 1B06 staat. Als IFF dan in enable staat, reageert de Z-80 door eerst IFF in disable te plaatsen en M1' en IORQ' even laag te maken. Nu leest de Z-80 de waarde op de databus, plaatst de PC op de stack en laadt deze opnieuw met de inhoud van de lokaties 1B06 en 1B07. Nu staat in de PC het adres 1940 (het startadres van de gewenste interrupt routine), vanwaar de uitvoering wordt voortgezet.
Als start zal hard- en software ontworpen worden voor een programma met slechts één interrupt routine die begint op adres 1830 (normaal zou hiervoor mode 1 voldoen). Eerst moet bepaald worden waar de vector in de interrupt-vector-tabel geplaatst zal worden. Figuur 24.F.2 toont de gekozen page, maar nu met de LSB van de adressen in binaire vorm. Dit is ook de waarde die het randtoestel op de databus moet plaatsen bij een onderbrekingsaanvraag. Een normale menselijke neiging is de vector van de interrupt routine vooraan in de tabel te plaatsen, dus op de adressen 1B00 en 1B01. In dit geval moet het randtoestel alle acht bits van de databus laag maken, en zijn er bijgevolg ook acht three-state-poorten naar de databus nodig.
Wordt daarentegen de enige vector achteraan in de page geplaatst, dan moet alleen bit 0 laag gemaakt worden door het randtoestel, waarvoor slechts één three-state-poort nodig is.
De nodige hardware beperkt zich tot die uit figuur 24.F.3.

Te leveren byte Bit 7654 3210	Adres
0000 0000	1B00
0000 0010	1B02
0000 0100	1B04
1111 0000	1BF0
1111 0010	1BF2
1111 0100	1BF4
1111 0110	1BF6
1111 1000	1BF8
1111 1010	1BFA
1111 1100	1BFC
1111 1110	1BFE

Fig. 24F2

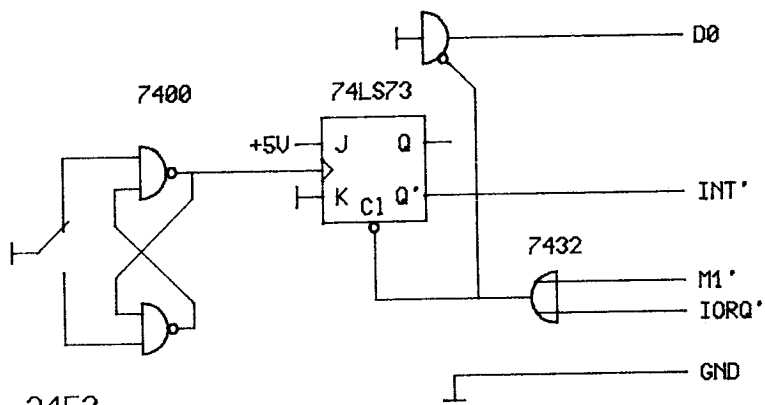


Fig. 24F3

Om ook weer de display telkens met 1 te verhogen bij een interrupt moet P24E1 aangepast worden voor gebruik bij mode 2.

Adres	Mach. taal	Label	Menmonic	Verklaring
1800	21 3018	P24F1	LD HL,1830	Adres interrupt rout.
1803	22 FE1B		LD 1BFE,HL	Naar int.vector 128d.
1806	DD21 5018		LD IX,1850	Adres DISBF0.
180A	ED5E		IM 2	Interrupt mode 2.
180C	3E 1B		LD A,1B	MSB int.vector tabel.
180E	ED47		LD I,A	Naar Interrupt regis.
1810	F3	LUS	DI	
1811	CD 2406		CALL 0624	SCAN1.
1814	FB		EI	
1815	18 F9		JR LUS	Blijf herhalen.
1830	21 4F18	I24F1	LD HL,184F	Adres TELLER.
1833	34		INC (HL)	Verhoog TELLER.
1834	7E		LD A,(HL)	TELLER in A.
1835	23		INC HL	Wijs naar DISBF0.
1836	CD 7806		CALL 0678	HEX7SG.
1839	ED4D		RETI	Terug naar hoofdprog.
184F	00	TELLER		
1850	00	DISBF0		
1851	00			
1855	00			

Om van buitenaf uit meerdere interrupt routines te kunnen kiezen is de hardware van figuur 24.F.4 nodig. Met de twee schakelaars kan uit vier verschillende interrupt routines gekozen worden waarvan de vector moet staan op de adressen 1BF8, 1BFA, 1BFC en 1BFE. Naargelang de stand van de twee schakelaars zal door het bedienen van de drukknop de display met 1 worden verhoogd, met 1 worden verlaagd, met 2 verhoogd of op nul gesteld worden.

Adres	Mach. taal	Label	Menmonic	Verklaring
1800	21 3018	P24F2	LD HL,1830	Adres int.rout.+1.
1803	22 FE1B		LD 1BFE,HL	Naar int.vector 128d.
1806	21 6018		LD HL,1860	Adres int.rout.-1.
1809	22 FC1B		LD 1BFC,HL	Naar int.vector 127d.
180C	21 7018		LD HL,1870	Adres int.rout.+2.
180F	22 FA1B		LD 1BFA,HL	Naar int.vector 126d.
1812	21 8018		LD HL,1880	Adres int.rout.op 0.
1815	22 F81B		LD 1BF8,HL	Naar int.vector 125d.
1818	DD21 5018		LD IX,1850	Adres DISBF0.
181C	ED5E		IM 2	Interrupt mode 2.
181E	3E 1B		LD A,1B	Page.
1820	ED47		LD I,A	Naar interrupt regis.
1822	F3	LUS	DI	
1823	CD 2406		CALL 0624	SCAN1.
1826	FB		EI	
1827	18 F9		JR LUS	Blijf herhalen.
1830	21 4F18	I24F20	LD HL,184F	Adres TELLER.
1833	34		INC (HL)	Verhoog TELLER.

```
1834  7E           LD A,(HL)     TELLER in A.
1835  23           INC HL        Wijs naar DISBF0.
1836  CD 7806      CALL 0678     HEX7SG.
1839  ED4D         RETI          Terug naar hoofdprog.

184F  00     TELLER
1850  00     DISBF0
1851  00

1855  00

1860  21 4F18  I24F21  LD HL,184F   Adres TELLER.
1863  35               DEC (HL)     Verlaag TELLER.
1864  7E               LD A,(HL)    TELLER in A.
1865  23               INC HL       Wijs naar DISBF0.
1866  CD 7806          CALL 0678    HEX7SG.
1869  ED4D             RETI         Terug naar hoofdprog.

1870  21 4F18  I24F22  LD HL,184F   Adres TELLER.
1873  34               INC (HL)     Verhoog TELLER.
1874  34               INC (HL)     Verhoog TELLER.
1875  7E               LD A,(HL)    TELLER in A.
1876  23               INC HL       TELLER in A.
1877  CD 7806          CALL 0678    HEX7SG.
187A  ED4D             RETI         Terug naar hoofdprog.

1880  21 4F18  I24F23  LD HL,184F   Adres TELLER.
1883  36 00            LD (HL),00   TELLER op nul.
1885  7E               LD A,(HL)    TELLER in A.
1886  23               INC HL       Wijs naar DISBF0.
1887  CD 7806          CALL 0678    HEX7SG.
188A  ED4D             RETI         Terug naar hoofdprog.
```

Dit van buitenaf kiezen welke routine uitgevoerd zal worden, demonstreert met eenvoudige hardware hoe de Z-80 de gewenste routine vindt.
Van meer praktisch belang is, dat meerdere randtoestellen een interrupt kunnen aanvragen en dat daarbij de passende interrupt routine wordt uitgevoerd. Natuurlijk moet ieder toestel dan een interrupt-signaal kunnen leveren. In figuur 24.F.5 zijn er vier randtoestellen die alleen bestaan uit een drukknop met contactdender-onderdrukking. De bedoeling van de drukknoppen is de nummerieke waarde op de display te beïnvloeden. Iets in de zin van een scorebord voor sportwedstrijden, of voor het automatisch tellen van verschillende soorten voorwerpen.
Hier moet :
— drukknop b0 de display met 1 verhogen,
— drukknop b1 de display met 1 verlagen,
— drukknop b2 de display met 2 verhogen,
— drukknop b3 de display op 00 stellen.
Als software is hiervoor P24F2 bruikbaar met de interrupt routines I24F20, I24F21, I24F22 en I24F23.
Het IC 74148 is een 8 naar 3 encoder. De invloed van zijn ingangen 0 tot 7 op de uitgangen C, B, A en EO' is te zien in navolgende tabel.

ingang laag	C	B	A	OE'
geen	H	H	H	H
0	H	H	H	L
1	H	H	L	L
2	H	L	H	L
3	H	L	L	L
4	L	H	H	L
5	L	H	L	L
6	L	L	H	L
7	L	L	L	L

Het indrukken van drukknop b0 maakt ingang 0 laag. De uitgangen C, B en A zijn hoog maar OE' wordt laag en triggert de JK-flip flop. Daar wordt uitgang Q' laag en daarmee INT'. Het hoofdprogramma P24F2 en de Z-80 maakt M1' en IORQ' beide laag. De JK-flip flop wordt teruggezet, de 74367 wordt actief en maakt daarmee alle datalijnen hoog, uitgezonderd D0. De Z-80 leest op de databus de waarde FE, en weet daardoor dat de vector van de gewenste routine op adres 1BFE staat. Daar heeft het hoofdprogramma het adres 1830 geschreven, waardoor de routine uitgevoerd wordt die de display met 1 verhoogt.
Bij het indrukken van de drukknoppen b1, b2 en b3 gebeurt hetzelfde, maar dan komen op de databus de waarden FC, FA en F8, waardoor dan ook telkens een andere routine wordt uitgevoerd.

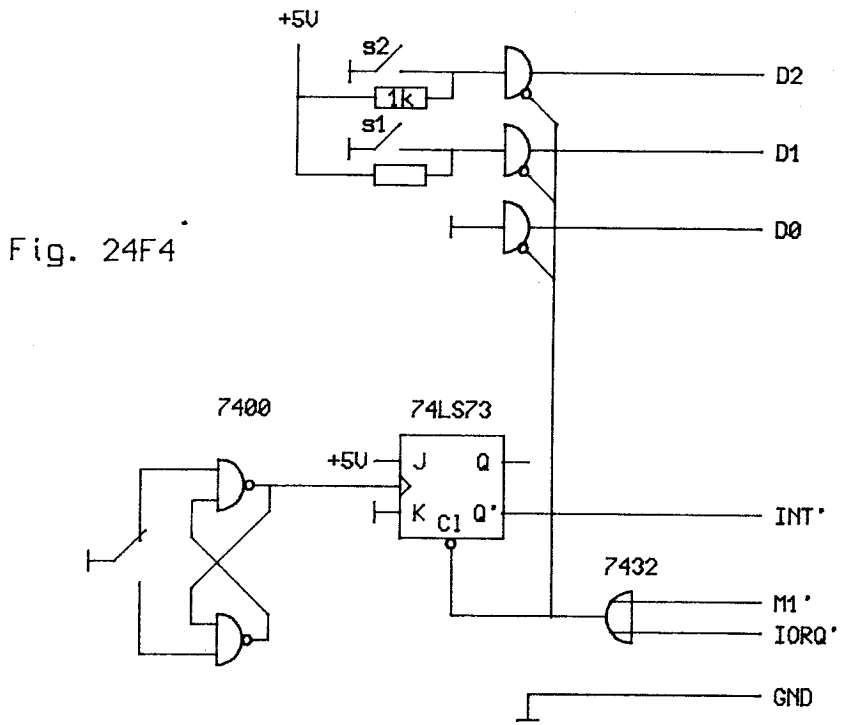


Fig. 24F4

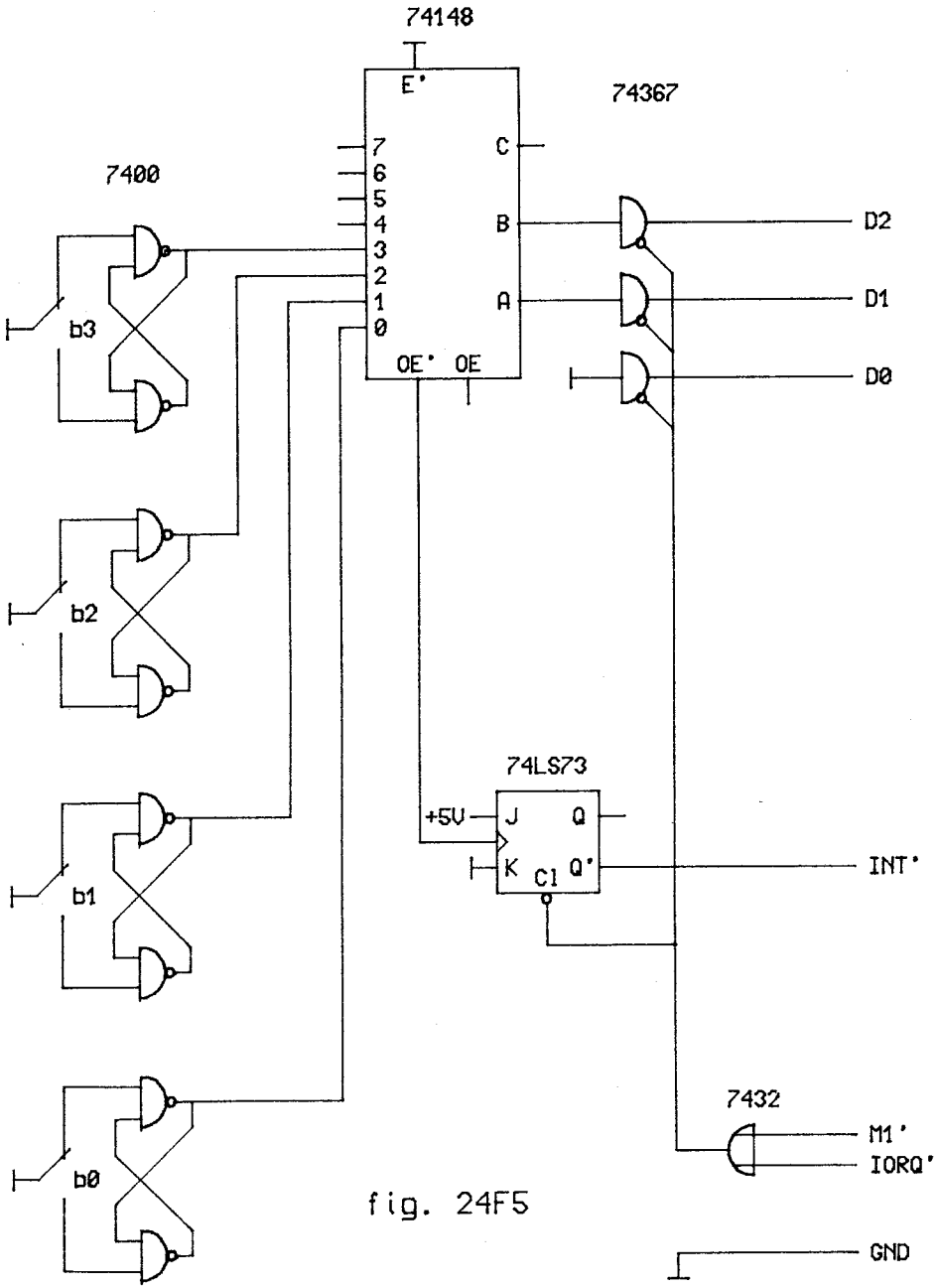

74148
E*
74367
7400
7
6
5
4
3
2
1
0
C
B
A
D2
D1
D0
b3
b2
b1
b0
OE*
OE
74LS73
+5V
J
K
Q
Q*
C1
INT*
7432
M1*
IORQ*
GND

fig. 24F5

Op de ingangen 4 tot 7 kunnen nog vier bijkomende drukknoppen aangesloten worden. Wel moet dan ook uitgang C via een poort van de 74367 op de datalijn D3 aangesloten worden. Met de gepaste encoder (128 naar 7) kan het systeem verder uitgebreid worden zodat 128 randtoestellen elk een onderbreking kunnen aanvragen om hun eigen routine te laten uitvoeren.

Taak 1. Ontwerp een scorebord voor voetbal.
Op de displays 0 en 1 moet de score voor de thuisploeg verschijnen en op de displays 2 en 3 die voor de bezoekers.
De displays 4 en 5 geven het aantal gespeelde minuten weer.
Drukknop b0 verhoogt de score voor de thuisploeg en drukknop b1 die van de bezoekers. Met drukknop b2 wordt de tijd op 00 gesteld bij de start van de tweede speelhelft.
Een uitwendige schakeling moet het signaal leveren voor het verhogen van de minuten, en afkomstig zijn van de netfrequentie.

Taak 2. Ontwerp hard- en software voor een opstelling die het bedrag moet berekenen van een aantal ingevoerde muntstukken. Neem aan dat vier soorten muntstukken aangeboden worden. Elk aangeboden muntstuk sluit eenmaal een contact overeenkomend met de waarde van hun muntstuk. Na iedere invoer moet het totale bedrag van de ingevoerde stukken getoond worden. Een vijfde contact stelt de display op nul.

24.G. NMI (NonMaskable Interrupt)

Bij een dalende flank op de ingang NMI' werkt de Z-80 de lopende instructie af en zet IFF in disable. De PC wordt op de stack gedrukt en opnieuw geladen met adres 0066 waar de interrupt routine moet staan. Na het uitvoeren van de NMI routine wordt het onderbroken programmadeel voortgezet waar het onderbroken werd. Een dalende flank op NMI' laat altijd de routine op adres 0066 uitvoeren. De stand van de IFF heeft daarop geen invloed. De NMI'-ingang heeft daardoor totale voorrang op de INT'-ingang.
Bij de Micro-Prof start op adres 0066 de NMI routine, die alle registers in de registerbuffer drukt (zie figuur 10.D.1), controleert of de stack pointer naar een toegelaten RAM-deel wijst en daarna het toetsenbord gaat aftasten.
De drukknop MONI is via twee NOT-poorten en de tienteller 74LS90 aangesloten op de NMI'-ingang.
Door het sluiten van MONI worden de ingangen R9 van de 74LS90 hoog. Deze ingangen hebben totale voorrang op alle andere en stellen de tienteller op 9 (1001). Daarbij is uitgang QA hoog en wordt NMI' laag. De NMI-routine wordt uitgevoerd.
Terwijl MONI openstaat stelt het signaal BREAK', dat normal hoog is, de 74LS90 op 0 (0000).
Als een breakpoint ontmoet wordt, moet het programma onderbroken worden net als met de MONI-toets. De breakpoint routine brengt daarvoor de waarde 80 (1000 0000) naar poort C van de 8255, waardoor de uitgang BREAK' laag wordt. Dit gebeurt door de instructies op de adressen 0048 tot 0053. Daar is ook te zien dat eerst nog enkele instructies moeten worden uitgevoerd, voordat de interrupt mag optreden. Deze vertraging wordt verkregen door de 74LS90. Het laag worden van BREAK' maakt de vijfteller in de 74LS90 vrij, die begint te tellen. Na vijf pulsen van M1' wordt QA hoog en NMI' laag. Pas dan wordt een interrupt aange-

vraagd waardoor de Micro-Prof in dezelfde toestand komt als na het drukken op de toets MONI.
Daar de ingang NMI' door de Micro-Prof gebruikt wordt, en niet ter beschikking staat van de gebruiker, wordt hier niet verder ingegaan op deze manier van onderbreken.