

X1

N.V. ELECTROLOGICA

KORTE ALGEMENE BESCHRIJVING

van de

elektronische rekenmachine X 1

(EL-3)

N. V. ELECTROLOGICA

's-Gravenhage

1958

Een woord vooraf

Deze brochure beschrijft de X-1 met de bijbehorende apparatuur, alsmede de mogelijkheden er van. Tevens geeft zij een beknopte uiteenzetting van hetgeen zich in de machine afspeelt, zonder op zuiver technische kwesties in te gaan. Zij richt zich voornamelijk tot die lezers, die op grond van wetenschappelijke belangstelling een indruk van de interne processen wensen te verkrijgen.

Wie zich voorlopig slechts een algemeen overzicht van de X-1 en haar prestaties wenst te verschaffen, kan zich de bestudering van deze brochure vergemakkelijken door de gedeelten in kleine letters geheel of gedeeltelijk over te slaan. Hij behoeft niet te vrezen, daardoor de samenhang uit het oog te verliezen.

In het algemeen kan worden gesteld, dat voor de programmeur zowel als de operateur kennis van de (hier niet in zijn gehele omvang omschreven) opdrachtencode en de bedieningsaanwijzingen voldoende is. Hij behoeft dan ook niet op de hoogte te zijn van de interne processen om zijn taak naar behoren te kunnen vervullen. In het bijzonder geldt dit voor de uiteenzettingen over de invoer via ponsband en de synchronisatie met de externe apparatuur (hoofdstuk VII en VIII). Het is in eerste instantie voldoende de aanwijzingen en hun gevolgen te kennen.

Het zou echter onjuist zijn uit het bovenstaande de conclusie te trekken, dat nauwkeurige kennis van de interne processen ten enenmale overbodig zou zijn. Integendeel, hoe beter men de machine kent, des te beter zal men de door haar geboden mogelijkheden weten uit te buiten.

N. V. ELECTROLOGICA.

I. Inleiding.

De X-1 is een snelle elektronische rekenautomaat, die zowel voor administratief als voor wetenschappelijk werk zeer geschikt is.

Hij bestaat in hoofdzaak uit drie onderdelen: de basismachine, het geheugen en de in- en uitvoerorganen.

De basismachine, die in principe voor elke installatie dezelfde is, bevat het arithmetisch orgaan, de besturingsapparatuur, het bedieningspaneel, etc.

Het geheugen heeft een omvang, die tot een zeker maximum voor elke installatie gekozen kan worden en daardoor aangepast kan zijn aan de behoefte van de gebruiker.

Evenzo kan per installatie, natuurlijk binnen zekere grenzen, gekozen worden, welke in- en uitvoerapparatuur aan de machine gekoppeld wordt.

(Wie begint met bescheiden accessoires, kan later het geheugen of de aangesloten apparatuur uitbreiden.)

Dat de X-1 bijzonder geschikt is voor wetenschappelijk zowel als voor administratief werk, dankt hij, behalve aan zijn grote flexibiliteit, aan de hoge rekensnelheid. Voor administratief werk komt de snelheid onder andere tot uiting in de mogelijkheid, de X-1 gelijktijdig b.v. verschillende ponskaart-apparaten te laten behandelen.

II. Beschrijving van de onderdelen.

a. De basismachine.

De basismachine bevat het arithmetisch orgaan, de verschillende interne registers, de besturingsapparatuur etc. De fysieke gedaante is ongeveer die van een normaal schrijfbureau; op een gedeelte van het blad en verder op een opstaand paneeltje daarachter bevinden zich de verschillende knoppen en schakelaars voor de bediening en een aantal indicatielampjes.

De circuits, welke in de elektronische onderdelen gebruikt worden, zijn alle volledig getransistoriseerd, zodat het totaal opgenomen vermogen voor een rekenautomaat van dit kaliber bijzonder laag is (enkele honderden Watts). De anders noodzakelijke kostbare en vaak volumieuze koelingsapparatuur is daarmee geheel vervallen.

Intern opereert het arithmetisch orgaan op getallen in het tweetallig stelsel met vaste komma. Alle transporten, benevens optellingen en aftrekkingen, worden parallel uitgevoerd, d.w.z. alle cijfers van het getal worden simultaan behandeld. Met uitzondering van de vermenigvuldiging en de deling, die elk 500 mms duren, is daarmee voor alle andere opdrachten een operatietijd, ruimschoots onder de 100 mms bereikt (variërend van 0,000032 sec. tot 0,000084 sec.).

De "natuurlijke" informatie-eenheid van de X-1 wordt een woord genoemd en omvat 27 bits (= binaire cijfers); woorden kennen hoofdzakelijk twee interpretaties, n.l. als getal en als opdracht. Als getal beschouwd bestaat een woord uit 1 tekenbit, gevolgd door 26 binaire cijfers van het getal. Als opdracht beschouwd omschrijven de 27 bits van een woord een door de X-1 uit te voeren handeling. Wij kunnen ze daarbij verdelen in twee groepen: de linkse 12 (het functiegedeelte) omschrijven volgens bepaalde regels de aard van de operatie, de rechtse 15 (het numerieke of adresgedeelte) kunnen dienen voor de specificatie van het getal, waarmee de operatie wordt uitgevoerd.

Het arithmetisch orgaan bevat drie rekenregisters, n.l. de registers A en S, elk van 27 bits en het B-register van 16 bits. Behalve dat het B-register, doordat het korter is, niet zulke grote getallen kan herbergen als het A- en het S-register, zijn deze drie registers, wat betreft hun accumulator-functie gelijkwaardig: in deze drie registers kan worden opgeteld en afgetrokken, hun inhoud kan bij een getal in het geheugen worden opgeteld of daarvan afgetrokken, etc.

Bij de vermenigvuldigingen en delingen, die zich altijd afspelen in de registers A en S, hebben deze registers een verschillende functie. Het B-register heeft als bijzonderheid, dat zijn inhoud na extractie van een opdracht uit het geheugen (zie volgende alinea's), doch vóór de uitvoering ervan, bij het adresgedeelte van deze opdracht kan worden opgeteld.

Men noemt het B-register dan ook wel: adresmodificatieregister. De indicatie of deze optelling al dan niet dient te geschieden, bevindt zich in het functiegedeelte van de opdracht zelf. De capaciteit van het B-register van 16 bits vloeit voort uit de omstandigheid, dat dit adresincrement beide tekens kan hebben, zodat er behalve 15 adresbits ook nog een tekenbit aanwezig is.

Bij de bespreking van de opdrachtencode zal dit alles nader toegelicht worden.

Nog twee registers hebben voor de programmeur enige betekenis, n.l. het opdrachtregister OR en de opdrachtteller T. (De inhoud van de vijf genoemde registers kan zichtbaar gemaakt worden door middel van indicatielampjes op het controlepaneel. Wat er zich in de X-1 afspeelt kan men, indien gewenst, stap voor stap volgen.)

Voordat een opdracht uitgevoerd kan worden, moet hij in het opdrachtre-gister OR zijn geplaatst, omdat hier de "analyse" van de bits van de op-dracht plaats vindt; tijdens de gehele uitvoering van de opdracht blijft de-ze in OR bewaard. Omdat een opdracht een volledig woord beslaat, is er in OR ruimte voor 27 bits. De opdracht, die ter uitvoering in OR geplaatst werd, kwam uit het geheugen. De woorden in het geheugen zijn genummerd en de opdrachtteller T bevat steeds het nummer van het woord, dat als volgende opdracht van het geheugen naar OR getransporteerd zal worden. De opdrachtteller T is een register van 15 bits.

Tenslotte zijn er nog drie registertjes met elk een capaciteit van 1 bit, die in het volgende ter sprake zullen komen: het conditieregister, het laatste-teken-register en het overloopregister.

b. Het geheugen.

Het geheugen van de X-1 biedt maximaal plaats aan $2^{15} = 32768$ woorden, genummerd van 0 t/m 32767; deze nummers worden adressen der geheugen-plaatsen genoemd, omdat zij gebruikt worden om naar de woorden in het geheugen te refereren. Het geheugen kan woordsgewijs informatie ontvan-gen of afleveren: in het eerste geval zeggen wij, dat er een woord in het geheugen geschreven wordt, in het andere geval, dat er een woord gelezen wordt. Bij iedere lees- of schrijf-opdracht wordt behalve de operatie ook het adres opgegeven van de betrokken geheugenplaats. Als er op een geheu-genplaats een woord geschreven wordt, geschiedt dit met vernietiging van de vorige inhoud; bij de leesoperatie blijft, tenzij anders aangegeven, de geheugeninhoud ongestoord.

Het geheugen van de X-1 bestaat uit twee gedeelten, die de namen "levend geheugen" en "dood geheugen" dragen. Beide zijn opgebouwd uit ferriet-kernen en hebben dezelfde accestijd voor leesoperaties. Het verschil is, dat terwijl in het levende geheugen door de machine ook geschreven kan wor-den, dit bij het dode geheugen niet mogelijk is. Bij het levende geheugen wordt de informatie vastgelegd door de magnetische toestand der ferriet-kernen, in het dode geheugen daarentegen zijn de gegevens vastgelegd door een permanente bedrading. Het voordeel van het dode geheugen ligt in de relatief lage prijs: het is daarom bij uitstek geschikt voor het bewaren van standaard-programma's van allerlei soort, zoals programma's voor de in- en uitvoer, arithmetische subroutines en interpretatieve subroutine-aggregaten en desgewenst zelfs standaard-hoofdprogramma's.

De opdrachtcodes van de X-1 is een zgn. één-adres-code, d.w.z. elke opdracht refereert hoogstens naar één woord in het geheugen: welk woord dat dan is, wordt bepaald door het adresgedeelte van de opdracht. Dat in het opdrachtwoord 15 bits ter beschikking staan voor het adresgedeelte is de achtergrond van de eerder gedane mededeling, dat het geheugen (levend en dood samen) maximaal 2^{15} woorden omvat.

Het levend geheugen wordt afgeleverd in blokken van 512 woorden, welke geassembleerd worden tot kasten van 4096 woorden; elke kast heeft zijn eigen volledig onafhankelijke selectie-apparatuur, zodat bij vergroting van het geheugen de betrouwbaarheid niet afneemt.

Het dode geheugen wordt geleverd in blokken van 64 woorden, welke geas-sembleerd worden tot plugbare eenheden van 4096 woorden.

Dat het geheugen zijn informatie correct opgeeft, wordt gegarandeerd door een pariteitscontrole: aan elk woord in het geheugen is een 28ste bit toegevoegd, dat altijd een dusdanige waarde heeft, dat het totale aantal énen in het aldus aangevulde woord oneven is. Dit oneven zijn wordt steeds door de basismachine gecontroleerd.

c. In- en uitvoerorganen.

Ter verwerking van ponsbanden en het uittypen van gegevens kan de machine uitgevoerd worden met een bandlezer (150 symbolen/sec.), een bandponser (25 symbolen/sec.), beide voor vijf-gats ponsband en een door de machine bediende elektrische schrijfmachine (10 symbolen/sec.). Omdat deze apparaten voor massale in- en uitvoer van gegevens te langzaam zijn, kunnen ook ponskaartenmachines aangeschakeld worden. Als eerste mogelijkheid vermelden wij de reproductrices.

Van een reproductrice kunnen zowel leesbaan als ponsbaan door de X-1 bediend worden; de kaarttoevoer in beide banen kan onafhankelijk geregeld worden. De snelheid per baan is ca. 7200 kaarten per uur.

Voorts komt in aanmerking als invoerorgaan een snelle sorteermachine (42000 kaarten per uur): de X-1 kan de gegevens van de passerende kaarten opnemen en tevens het sorteermechanisme besturen.

Tenslotte vermelden wij de tabulatrice; het leesstation hiervan kan 9000 kaarten per uur lezen en kan als invoerorgaan gebruikt worden; het drukmechanisme van de tabulatrice kan direct door de X-1 bestuurd worden.

De communicatie tussen de basismachine en ponskaartenmachines vindt plaats door tussenkomst van zogenaamde buffergeheugens. Deze maken het mogelijk, dat de X-1 gedurende het kaartlezen normaal verder rekent. Dank zij enige speciale voorzieningen is de X-1 in staat verschillende ponskaartapparaten gelijktijdig te bedienen, ook wanneer deze onderling niet gesynchroniseerd zijn.

Tenslotte vermelden wij, dat het lezen van kaarten, evenals het ponsen, automatisch volledig gecontroleerd wordt.

III. De opdrachtcode.

Zoals reeds gezegd, kunnen de binaire cijfers van het opdrachtwoord gesplitst worden in functiegedeelte en adresgedeelte (ook wel numeriek gedeelte genaamd). In overeenstemming met deze indeling noteert de programmeur de opdrachten in principe eveneens als "functie", gevolgd door "adres". In de volgende beschrijving van de opdrachtcode (een opsomming van de voorkomende functies!) zullen wij voorlopig het adresgedeelte met de letter n aanduiden. Met (n) wordt bedoeld: "de inhoud van adres n"; evenzo met (A): "de inhoud van het A-register" etc. Ter beschrijving van de operaties zullen wij gebruik maken van het zgn. gerichte gelijkteken: $\Rightarrow$ (lees "vervangt"). Voor elk der drie registers bestaat het volgende achttal opdrachten (hierin is R = A, S of B):

0R n	$(R) + (n) \Rightarrow (R)$
1R n	$(R) - (n) \Rightarrow (R)$
2R n	$+ (n) \Rightarrow (R)$
3R n	$- (n) \Rightarrow (R)$
4R n	$(n) + (R) \Rightarrow (n)$
5R n	$(n) - (R) \Rightarrow (n)$
6R n	$+ (R) \Rightarrow (n)$
7R n	$- (R) \Rightarrow (n)$

0R & 1R Optelling in.

Het getal op geheugenplaats n wordt, al of niet na tekenwisseling opgeteld bij het getal, dat zich in het register R bevindt; het antwoord wordt in dat register opgeborgen. De oorspronkelijke inhoud van dat register gaat dus verloren, terwijl de inhoud van de geheugenplaats n ongewijzigd blijft.

2R & 3R Transport in.

Het getal op geheugenplaats n wordt, al of niet na tekenwisseling in het register R geplaatst. De oorspronkelijke inhoud van het betrokken register gaat dus verloren, terwijl de inhoud van geheugenplaats n ongewijzigd blijft.

4R & 5R Optelling uit.

Het getal in het register R wordt, al of niet na tekenwisseling, opgeteld bij het getal op geheugenplaats n, het resultaat wordt opgeborgen op adres n. De oorspronkelijke inhoud van geheugenplaats n gaat dus verloren, terwijl de inhoud van het register onaangetaast blijft.

6R & 7R Transport uit.

De inhoud van het register R vervangt (al of niet na tekenwisseling) de inhoud van adres n. De oorspronkelijke inhoud van dit adres gaat dus verloren, de inhoud van het register blijft ongewijzigd.

In het tweede viertal opdrachten zijn de rollen van register en geheugenplaats dus juist verwisseld. Wij merken op, dat bij additie dus altijd het ene getal in het geheugen, het andere in een register staat. Dit betekent, dat wij een geheugenplaats als "werkruimte" moeten gebruiken, als wij registerinhouden bij elkaar op willen tellen. Zo zouden wij b.v. de totale operatie

$$(S) - (A) + (B) \Rightarrow (S)$$

kunnen opbouwen met het volgende rijtje opdrachten:

$$\left. \begin{array}{l} 6B \text{ n} \quad \text{d.w.z.} \quad (B) \Rightarrow (n) \\ 5A \text{ n} \quad (n) - (A) \Rightarrow (n) \\ 0S \text{ n} \quad (S) + (n) \Rightarrow (S) \end{array} \right\} \text{ Samen: } (B) - (A) \Rightarrow (n)$$

Als volgend voorbeeldje: stel gegeven, dat $(n) = x$ en $(m) = y$. Gevraagd, om de woorden op adressen m en n te verwisselen, maar de ene bovendien van teken te veranderen, precieser: $-y \Rightarrow (n)$ en $x \Rightarrow (m)$. Het volgende rijtje opdrachten verwezenlijkt deze substitutie:

2A n	d.w.z.	$x \Rightarrow (A)$
2S m		$y \Rightarrow (S)$
7S n		$-y \Rightarrow (n)$
6A m		$+x \Rightarrow (m)$

Het (kortere) B-register is in dit laatste voorbeeldje niet gebruikt; (B) is dus ongewijzigd gebleven.

Uit de zojuist gegeven voorbeeldjes blijkt, dat een getal in het geheugen zowel positief als negatief kan zijn, van teken gewisseld kan worden, etc. Voor een goed begrip beschrijven wij nu de getalrepresentatie nader. Daartoe nummeren wij de cijfers in een woord

$d_{26} \ d_{25} \dots\dots\dots d_0$

Het meest linkse cijfer d_{26} is het zgn. tekencijfer; $d_{26} = 0$ voor alle positieve getallen.

Het grootste positieve getal, dat in een woord gerepresenteerd kan worden is $67108863 = 2^{26} - 1$; binair: $d_{26} = 0$, $d_{25} = d_{24} = \dots\dots\dots = d_0 = 1$.

Om een getal van teken te doen wisselen, worden alle cijfers van het woord geïnverteerd, d.w.z. nullen worden enen en enen worden nullen. Negatieve getallen beginnen dus met $d_{26} = 1$. B.v.

+ 25 : 00 00000 00000 00000 00000 11001
- 25 : 11 11111 11111 11111 11111 00110.

Dit heeft tot gevolg, dat er twee representaties zijn voor het getal nul, n.l. + 0 (alle cijfers = 0) en - 0 (alle cijfers = 1). De - 0 is preferent; de opteller levert slechts + 0 als antwoord af als beide toegevoerde addenda = + 0 zijn. Aftrekking wordt uitgevoerd door het betrokken getal geïnverteerd op te tellen.

Voor de registers A & S bestaan de volgende zgn. logische operaties (met $R = A$ of S).

0LR n	$+ (n) \nabla (R) \Rightarrow (R)$
1LR n	$- (n) \nabla (R) \Rightarrow (R)$
2LR n	$+ (n) \wedge (R) \Rightarrow (R)$
3LR n	$- (n) \wedge (R) \Rightarrow (R)$

Opm.: De logische operaties bestaan alleen voor A- en S-register, niet voor het B-register; bovendien bestaan zij uitsluitend als "in-opdrachten".

0LR & 1LR Logische optelling.

Bij de opdracht 0LR (dus 0LA of 0LS) wordt uit de inhoud van adres n en die van het betrokken register een resultaat gevormd, dat een 1 heeft in alle cijferposities, waarin (n) en (R) verschillende cijfers hebben; in alle andere cijferposities heeft het resultaat een cijfer 0. Het aldus gevormde resultaat vervangt tenslotte de oorspronkelijke inhoud van het betrokken register R , die dus verloren gaat; de inhoud van geheugenplaats n blijft onveranderd. De opdrachten 1LA en 1LS werken met het geïnverteerde woord $- (n)$ in plaats van (n) .

2LR & 3LR Logische vermenigvuldiging.

Bij de opdracht 2LR (dus 2LA of 2LS) wordt uit de inhoud van adres n en die van het betrokken register een resultaat gevormd, dat een 1 heeft in alle cijferposities waarin zowel (n) als (R) een 1 hebben; in alle andere cijferposities heeft het resultaat een cijfer 0. Dit resultaat vervangt de oorspronkelijke inhoud van het register R , die dus verloren gaat, terwijl de inhoud van geheugenplaats n onveranderd blijft. De opdrachten 3LA en 3LS werken met het geïnverteerde woord $- (n)$ in plaats van (n) .

De code kent vier vermenigvuldigingen:

0X n	$[A] + [n] \cdot [S] \Rightarrow [AS]$
1X n	$[A] - [n] \cdot [S] \Rightarrow [AS]$
2X n	$+ [n] \cdot [S] \Rightarrow [AS]$
3X n	$- [n] \cdot [S] \Rightarrow [AS]$

De plaats van de komma (d.w.z. de plaats van de eenheid) speelt bij optellingen amper een rol, zolang de komma in beide addenda maar op dezelfde plaats staat. Bij vermenigvuldiging en deling ligt dit anders; wij gebruiken daarom vierkante haken [], om expliciet aan te geven, dat de inhoud, waarop deze haken betrekking hebben, opgevat moet worden als geheel getal.

Het symbool [AS] wordt gebruikt om het dubbel-lengte gehele getal aan te geven, waarvan de meest significante cijfers in A, de minst significante in het S-register staan.

Het symbool [AS] is alleen van toepassing als de tekens van (A) en (S) gelijk zijn; de numerieke waarde is gegeven door $[AS] = [A] \cdot 2^{26} + [S]$.

Bij de vermenigvuldigingen wordt dus de (al of niet van teken gewisselde) inhoud van adres n vermenigvuldigd met de inhoud van het S-register. Bij de additieve versies 0X en 1X wordt dit dubbel-lengte produkt aan de minst significante zijde vermeerderd met de oorspronkelijke inhoud van het A-register; bij de versies 2X en 3X blijft deze vermeerdering achterwege. Van het resultaat komt het meest significante stuk in A, het minst significante in S. De tekens van A en S zijn uiteindelijk gelijk aan het teken van het resultaat. De oorspronkelijke inhoud van A en S gaat verloren, de inhoud van geheugenplaats n blijft onveranderd.

De andere gebruikelijke plaats, waar de komma in een woord geïnterpreteerd wordt is tussen d_{26} en d_{25} ; het woord stelt dan een breuk voor met absolute waarde kleiner dan 1. Deze interpretatie wordt met accolades aangegeven. Kennelijk geldt

$$\{n\} = [n] \cdot 2^{-26}$$

Evenzo voeren wij voor het dubbel-lengte getal (AS) nog twee interpretaties in, n.l. als gemengd getal

$$[AS] = [AS] \cdot 2^{-26} = [A] + \{S\},$$

dus met de komma tussen beide registers in, en als lange breuk

$$\{AS\} = [AS] \cdot 2^{-52} = \{A\} + \{S\} \cdot 2^{-26},$$

met de komma voor in het meest significante register.

Uit de omschrijving b.v.

$$+ [n] \cdot [S] \Rightarrow [AS]$$

van de opdracht 2X n vinden wij na vermenigvuldiging met 2^{-26} , resp. 2^{-52} , de aequivalente omschrijvingen

$$+ [n] \cdot \{S\} = \{n\} \cdot [S] \Rightarrow [AS]$$

en $\{n\} \cdot \{S\} \Rightarrow \{AS\}$

In woorden: "vermenigvuldiging van "breuk x geheel getal" geeft een produkt met het geheel gedeelte in het A-register en het breukgedeelte in het S-register", en

"vermenigvuldiging van "breuk x breuk" geeft een dubbel-lengte breuk als antwoord, met het meest significante gedeelte in A". In het S-register staan, indien de breuken $\{n\}$ en $\{S\}$ afgerond waren, een aantal onverantwoorde cijfers.

De code kent vier delingen, n.l.

0D n	[AS] : +[n], rest $\Rightarrow$ [A], quotiënt $\Rightarrow$ [S]
1D n	[AS] : -[n], " , "
2D n	[A]. 2^{26} : +[n], " , "
3D n	[A]. 2^{26} : -[n], " , "

De deelopdrachten 2D en 3D beginnen met in S een 0 te plaatsen en wel de nul met het teken van A. Daarna is hun operatie verder gelijk aan 0D en 1D, die slechts uitgevoerd mogen worden als teken (A) = teken (S). (Anders is immers het symbool [AS] niet van toepassing.) Het dubbel-lengte getal [AS] wordt nu gedeeld door $\pm[n]$, het quotiënt komt in S, de rest blijft in het A-register achter. De rest heeft het teken van het deeltal en is in absolute waarde kleiner dan de deler. De oorspronkelijke inhoud van A en S gaat verloren, de inhoud van adres n blijft onveranderd.

Afgezien van de rest is de deling 2D n, na deling door 2^{26} zinvoller gedefinieerd als $[A] : [n] = \{A\} : \{n\} \Rightarrow \{S\}$.

Het quotiënt is dan niet afgerond.

De vier delingen geven slechts dan een correct antwoord als het quotiënt de capaciteit van een woord niet overschrijft.

In één opzicht is de beschrijving van de hierboven genoemde opdrachten onvolledig geweest; bij elke opdracht is n.l. inbegrepen de vervanging

$$(T) + 1 \Rightarrow (T),$$

d.w.z. de inhoud van de opdrachtsteller T wordt tijdens de uitvoering van elke opdracht met 1 verhoogd.

Aangezien, zoals reeds werd vermeld, (T) steeds gelijk is aan het adres van de geheugenplaats waaruit de volgende uit te voeren opdracht gelezen zal worden, betekent dit, dat de opdrachten uitgevoerd worden in de volgorde, waarin ze in het geheugen geborgen staan. Dit zolang wij ons beperken tot bovengenoemde opdrachten. Er bestaan n.l. opdrachten, die de machine van deze normale volgorde doen afwijken. Zij worden aangeduid als sprongopdrachten; hun functie bestaat uit het wijzigen van (T). Het zijn:

0T n	$(T) + (n) \Rightarrow (T)$	
1T n	$(T) - (n) \Rightarrow (T)$	
2T n	$+ (n) \Rightarrow (T)$	
4T n m	$(m) - 1 \Rightarrow (m); n \Rightarrow (T)$	$0 \leq m \leq 7$
6T n m	$(T) \Rightarrow (m+8); n \Rightarrow (T)$	$0 \leq m \leq 15$

0T & 1T De additieve sprongopdrachten.

Men dient het symbool (T), dat links van het vervangingssymbool staat, op te vatten als "de inhoud van T na de optelling $(T) + 1 \Rightarrow (T)$ ".

Deze optelling wordt al uitgevoerd, voordat de opdracht in OR als sprongopdracht geïdentificeerd is.

Speciaal: als $(n) = 0$, zijn beide opdrachten zonder effect. Zij fungeren dan als skip. d.w.z. zonder effect in het geheugen of de registers gaat de besturing over naar de volgende opdracht. Als $(n) = -1$, dan is de opdracht 0T n een zgn. dynamische stop.

2T De normale sprong.

Met de vervanging $(n) \Rightarrow (T)$ wordt bedoeld, dat de minst significante vijftien cijfers van het woord (n) in de opdrachtsteller overgenomen worden. Hierna is de sprongopdracht als zodanig voltooid en de volgende opdracht wordt uit het geheugen gelezen van het adres, dat door (T) is vastgelegd.

4T De tellende sprongopdrachten.

Hier wordt het adres gevolgd door een index m, die lopen kan van 0 t/m 7. De 4T opdracht is een sprongopdracht, omdat zijn adresgedeelte in T wordt overgenomen. (N.B.: Het is direct het getal n en niet de inhoud van adres n.) Bovendien wordt de inhoud van adres m met 1 verminderd. In dit verband wordt deze inhoud gewoonlijk aangeduid met τ_m ; de telling kan beschreven worden met: $\tau_m - 1 \Rightarrow \tau_m$.

De ware betekenis van deze opdrachten komt pas tot uiting, als wij ten volle gebruik maken van enige andere faciliteiten; bij de bespreking van de zgn. varianten der opdrachten komen wij hierop terug.

6T De subroutinesprong.

Ook de subroutinesprongen worden van een index m voorzien, die hier loopt van 0 t/m 15. Evenals bij de tellende sprongopdrachten wordt hier de substitutie in T uitgevoerd door direct het adresgedeelte n, en niet (n), in T over te nemen. Voordat dit echter geschiedt, wordt de inhoud van T, nadat de normale verhoging met 1 al heeft plaats gehad, overgenomen in geheugenplaats $m+8$. In dit verband wordt deze inhoud gewoonlijk met λ_m aangeduid. (De 16 λ 's beslaan dus de adressen 8 t/m 23.) De oorspronkelijke waarde van de betrokken λ_m gaat daarbij verloren. Deze opdrachten heten subroutinesprongen, omdat in λ_m het adres is vastgelegd van de opdracht, die aan de beurt geweest zou zijn, ware het niet, dat de 6T-opdracht als sprong de normale opeenvolging verstoord had.

De subroutinesprong "redt" dus de momentane inhoud van T en brengt deze in veiligheid in de vijftien minst significante plaatsen van λ_m . De reddende functie van de subroutinesprong is uitgebreider: de resterende bits van λ_m worden gevuld op grond van de momentane inhoud van enkele andere registertjes. Specificatie hiervan stellen wij uit, totdat deze registertjes besproken zijn.

Om der wille van de volledigheid vermelden wij de binaire representatie der genoemde opdrachten. In het opdrachtwoord zijn de 15 minst significante cijfers $d_{14} \dots d_0$ gereserveerd voor het numeriek gedeelte n. De gegeven "cijfer-letter"-namen der opdrachten bepalen de inhoud van de zes meest significante bits $d_{26} \dots d_{21}$ van het opdrachtwoord. Het getal, door deze zes binalen gevormd, kan lopen van 0 t/m 63; de waarde ervan wordt gevonden door bij het functiecijfer een door de functieletter bepaald aantal malen 8 op te tellen. Dit aantal malen is gegeven in onderstaand tabelletje:

voor A : 0	voor LS: $3\frac{1}{2}$
" S : 1	" B : 4
" X : 2	" T : 5
" LA: $2\frac{1}{2}$	" Y : 6
" D : 3	" Z : 7

later te bespreken

De opdracht 2D begint binair dus met 011010, de opdracht 3LA met 010111, etc. Bij tellende sprong en subroutinesprong (4T en 6T) is functiecijfer + 5 x 8 even, dus volgens bovenstaande regel $d_{21} = 0$. Bij deze opdrachten heeft echter d_{21} een afwijkende betekenis: de index m wordt in $d_{21} \dots d_{18}$ vastgelegd. Precieser: als de binalen van m zijn $b_3 b_2 b_1 b_0$, dan is $d_{21} = b_2$, $d_{20} = b_1$, $d_{19} = b_0$, $d_{18} = b_3$.

De meest significante bit b_3 (die bij de tellende sprong niet voorkomt, aangezien $0 \leq m \leq 7$) is dus achteraan gezet.

De resterende opdrachten worden kortweg onder de naam "communicatie-opdrachten" samengevat. Zij dragen een bijzonder karakter en zullen later besproken worden.

IV. De varianten.

De in het vorige hoofdstuk gegeven opdrachtbeschrijvingen zijn van kracht in het zgn. "normale geval". De opdrachten bestaan n.l. in verschillende versies. In drie opzichten kunnen de opdrachten n.l. gemodificeerd worden; deze varianten der opdrachten beheersen ruwweg "de adresverandering", "de conditie-reactie" en "de conditie-zetting".

a. De adresverandering.

De variant der adresverandering kent vier gevallen. Het normale geval is het beschrevene, de drie andere mogelijkheden worden aangegeven door achter het adres één van de letters A, B en C te plaatsen.

A ("Absoluut".)

De adresverandering is hier een verandering in de interpretatie van de adreescijfers: het numeriek gedeelte n van de opdracht wordt hierbij niet als adres, doch als getal opgevat. Men denkt daartoe de 15 cijfers van n aan de meest significante kantaangevuld met 12 nullen. Steeds als in de opdrachtbeschrijving stond "de inhoud van geheugenplaats n", moet men in het geval "Absoluut" daarvoor lezen "het getal n".

Zo is b.v. van de opdracht

2S n A, de functie: $n \Rightarrow (S)$,

in tegenstelling tot de normale opdracht

2S n, met de functie $(n) \Rightarrow (S)$.

Deze opdrachten maken het mogelijk, kleine constanten meteen in het adresgedeelte van de opdracht vast te leggen. Zij besparen daardoor geheugenruimte. Bovendien zijn ze over het algemeen sneller dan de normale versie: na het aanhalen van de opdracht is ter verkrijging van de operand in het geval "Absoluut" geen tweede geheugencontact meer nodig. Wij merken op:

- 1e. dat de Absoluut-versie zinloos is bij alle uit-opdrachten (4A t/m 7A, 4S t/m 7S en 4B t/m 7B);
- 2e. dat bij de opdrachten 4T en 6T (tellende, resp. subroutine-sprongen), blijkens de gegeven beschrijving het adres altijd "Absoluut" gebruikt wordt. Bij deze opdrachten is de variant der adresverandering niet van toepassing, het adres mag niet door A (noch B, noch C) gevolgd worden. In plaats daarvan komt de verplichte index m.

B & C ("B-correctie" & "C-correctie").

In de gevallen van B- of C-correctie wordt vóór het uitvoeren van de aangehaalde opdracht de inhoud van het B-register bij het adresgedeelte ervan opgeteld. Bij de uitvoering van de opdracht wordt dit nieuwe adres normaal (d.w.z. niet Absoluut) gebruikt. De beide adres-correcties verschillen hierin, dat bij B-correctie de betrokken opdracht ongewijzigd in het geheugen blijft staan, terwijl bij C-correctie het nieuw gevormde adres ook in het geheugen het oorspronkelijke adres vervangt. Dus $f n C$ wordt $f n + (B) C$.

Tenslotte kan in dit verband nog opgemerkt worden, dat bij sommige communicatie-opdrachten het gebruik van B- of C-correctie tot vreemde resultaten aanleiding kan geven, omdat daar de adreescijfers in feite functiebetekenis hebben: door B-correctie kan zo'n opdracht volslagen van karakter veranderen.

b. De conditie-reactie.

De variant der conditie-reactie kent vier mogelijkheden. In het vorige hoofdstuk is het normale geval beschreven; in de andere gevallen zet men vóór het functiegedeelte één van de letters U, Y of N. De laatste twee hebben de naam aan deze variant gegeven, want zij maken de opdracht reagerend op de zgn. "conditie". Dit is een afzonderlijk intern geheugenelementje, waarin één binair cijfer wordt bewaard. Dit ene binaire cijfer is dus = 0 of = 1; deze twee toestanden duiden wij aan door "Conditie is bevestigend, resp. ontkennend".

Y ("Yes-conditional").

In dit geval wordt de opdracht slechts uitgevoerd, indien de conditie bevestigend is, anders wordt de opdracht geskipt.

N ("No-conditional").

In dit geval wordt de opdracht slechts uitgevoerd, indien de conditie ontkennend is, anders wordt de opdracht geskipt. (Door de N reageert de opdracht, vergeleken bij Y, dus invers op de conditie.)

Wij vestigen er nogmaals de aandacht op, dat een geskipte opdracht niets in de registers, noch in het geheugen wijzigt; als b.v. een conditionele C-gecorrigeerde opdracht geskipt wordt, wordt de C-correctie niet uitgevoerd, de opdracht blijft ditmaal onveranderd in het geheugen staan.

U ("Undisturbed")

Deze variant betekent, dat het bedoelde resultaat wel ergens in de X-1 gevormd wordt, maar dat het transport naar de uiteindelijke bestemming onderdrukt wordt. Het resultaat van de bewerking gaat dus volledig verloren, terwijl de oorspronkelijke gegevens alle bewaard blijven. Het is duidelijk, dat deze versie op zich zelf zinloos zou zijn, ware het niet, dat het gevormde resultaat wel gebruikt kan worden om de conditie te zetten. (Zie onder.)

De variant "Undisturbed" bestaat niet voor vermenigvuldigingen, delingen en de later te behandelen schuif- en normeeropdrachten. (In het algemeen die opdrachten, waarbij de registers intensief gebruikt worden om het resultaat te vormen.)

Voor de T-opdrachten (sprongopdrachten) heeft de U afwijkende betekenis gekregen: n.l. "Voer de sprongopdracht uit, als de overloopindicatie = 1 en herzet deze dan op 0; anders skip". De overloopindicatie is een apart geheugenelement, dat één bit bevat (vergelijkbaar dus met de conditie). In dit element wordt een 1 neergezet, wanneer door optelling van twee positieve getallen door capaciteitsoverschrijding een negatief antwoord ontstaat of wanneer door optelling van twee negatieve getallen een positief antwoord ontstaat. (In sommige gevallen kan een capaciteitsoverschrijding helemaal geen kwaad, zodat het geen zin heeft de machine bij elke capaciteitsoverschrijding te laten stoppen. Soms zal men echter in dit geval speciale maatregelen willen treffen: de U-sprongen bieden hiertoe een mogelijkheid.)

c. De conditiezetting.

In principe beheerst de laatste variant, of de opdracht de conditie zal zetten. Zoals de opdrachten in het vorige hoofdstuk beschreven zijn, laten zij de conditie onveranderd. Willen wij, dat de conditie op grond van het resultaat der operatie gezet wordt, dan noteren wij helemaal achter aan de opdracht een van de letters P, Z of E.

Omdat de twee standen van de conditie "bevestigend" resp. "ontkennend" genoemd worden, formuleren wij het zetten van een conditie als het hierin vastleggen van het antwoord op een vraag. De mogelijkheden zijn:

P ("Positive?")

In dit geval wordt in de conditie vastgelegd het antwoord op de vraag "Is het resultaat $\geq + 0$?"

Z ("Zero?")

In dit geval wordt in de conditie vastgelegd het antwoord op de vraag "Is het resultaat = 0?"

E ("Equal signs?")

In dit geval wordt in de conditie vastgelegd het antwoord op de vraag "Is het teken van het resultaat gelijk aan het teken van het resultaat van de vorige conditezettende opdracht?"

Om de laatste vraagstelling mogelijk te maken, bestaat er een speciaal "Laatste-teken-registertje", dat evenals conditie en overloopindicatie één bit bevat. De inhoud van deze drie registertjes is permanent zichtbaar op indicatielampjes op het controlepaneel.

Van de genoemde opdrachten verdienen in dit verband vermenigvuldigingen, delingen en sprongopdrachten aparte vermelding: vermenigvuldigingen en delingen, omdat wij voor hen moeten preciseren, dat onder "het resultaat" verstaan moet worden de uiteindelijke inhoud van het A-register, en de sprongopdrachten, omdat op hen de boven gegeven omschrijving niet van toepassing is.

Bij de subroutine-sprongen (6T) komt P, Z of E niet voor. Bij de tellende sprongen (4T) kan P, Z of E wel voorkomen, hoewel zij de conditie onaangestast laten! Zoals in het vorige hoofdstuk de tellende sprongen beschreven zijn, zouden de uitvoering van de telling: $\tau_m - 1 \Rightarrow \tau_m$ en van de sprong: $n \Rightarrow (0T)$ onlosmakelijk aan elkaar verbonden zijn. Door achter de label m echter een P, Z of E te noteren, wordt het al of niet uitvoeren van de sprong afhankelijk gesteld van de nieuwe waarde van τ_m . En wel

P ("Spring als τ_m positief is")

De sprong wordt slechts uitgevoerd, als de nieuwe $\tau_m > 0$ is.

Z ("Spring als τ_m nul is")

De sprong wordt slechts uitgevoerd, als de nieuwe $\tau_m = 0$ is.

E ("Spring extra keer")

De sprong wordt slechts uitgevoerd, als de nieuwe $\tau_m \geq 0$ is.

(De sprong wordt hier - in geval van herhaald aantreffen van de opdracht - één keer vaker uitgevoerd dan in geval P; vandaar de naam: extra keer.)

Een tellende sprongopdracht kan conditioneel zijn op de conditie (Y of N) of op de overflow-indicatie (U); als aan de betrokken voorwaarde niet voldaan is, wordt de opdracht geskipt, dat wil dus zeggen er wordt niet geteld en niet gesprongen. Wordt de opdracht niet geskipt, dan wordt in elk geval geteld; de opdracht geldt dan voor uitgevoerd (bij de U-versie wordt de overflow-indicatie gecleard!), ook als na de telling blijkt, dat het effectueren van de sprong achterwege dient te blijven.

Bij de andere sprongen (0T, 1T en 2T) bestaat in deze variant alleen de versie P: de sprong wordt dan een zgn. herstellende sprong. Voor het ogenblik volstaan wij met een gedeeltelijke beschrijving.

De opdracht "2T n" transporteert de minst significante 15 bits van (n) in de opdrachtsteller. De opdracht "2T n P" doet dit ook, maar verricht enige nevenfuncties op grond van de meer significante cijfers van (n).

Wij vermelden voorlopig slechts de verwerking van vier van deze bits.

(n):	bij opdrachten 0, 1, 2T n P.
$d_{26} = 0$	Laat de X-1, wat betreft de sprongopdracht, doorwerken.
$\quad = 1$	Stop de X-1 na uitvoering van de sprong.
$d_{18} = 0$	Zet de conditie bevestigend.
$\quad = 1$	Zet de conditie ontkennend.
$d_{17} = 0$	Zet laatste tekenregister positief
$\quad = 1$	Zet laatste tekenregister negatief.
$d_{16} = 0$	Zet 0 in de overflow-indicatie (geen overflow).
$\quad = 1$	Zet 1 in de overflow-indicatie (wel overflow).

Bij de subroutinesprongen vinden de inverse operaties plaats: behalve dat (T) in de bijbehorende λ_m gered wordt, worden dus bovendien van λ_m de cijfers d_{18} , d_{17} en d_{16} bepaald door conditie, laatste teken en overflow-indicatie. Het is hiermee mogelijk geworden een subroutine aan te roepen, die na afloop de besturing weer naar het hoofdprogramma terug verwijst met wat betreft conditie etc. herstel van de status quo op het moment van aanroep.

Indien de subroutinesprong conditioneel was op overflow (dus de U-variant van de sprongopdracht), dan wordt in de bijbehorende λ_m de overflow-indicatie al op nul gezet overgenomen. Als aan het einde van de subroutine de besturing herstellend terugspringt, wordt dus (OF) = 0.

V. De opdrachtnotatie.

a. De adresnotatie.

Het begrip voor de adresnotatie wordt vergemakkelijkt als men zich voorstelt, dat het geheugen is ingedeeld in paragrafen, deze weer in pagina's en elke pagina tenslotte in (32) opeenvolgende regels.

Dienovereenkomstig wordt een adres omschreven door:

1. het regelnummer,
2. het paginanummer,
3. de paragraaf-aanduiding.

1. De regels zijn genummerd van 0 t/m 31, met het gevolg, dat het regelnummer steeds kleiner is dan 32. Op regel 31 van een pagina volgt regel 0 van de volgende pagina. De standaard-programmavellen voor de X-1 zijn op deze notatie volledig afgestemd en bevatten dan ook 32 regels, van boven naar beneden genummerd van 0 t/m 31.

2. Telkens tot en met 17 pagina's, naar gelang van hun aantal doorlopend genummerd met getallen van 0 - 16, vormen (bij gelijkluidende paragraafletters!) een paragraaf.

3. De paragrafen worden niet met cijfers doch met letters (paragraafletters) aangeduid.

Een adres wordt genoteerd in de volgorde:

regelnummer (0 - 31) - paragraafletters - paginanummer (0 - 16).

Uit deze symbolen leidt de X-1 tijdens de invoer de interne binaire representatie van het adres af. Hiertoe telt hij het regelnummer, vermeerderd met het twee-en-dertigvoud van het paginanummer, op bij het beginadres - d.w.z. het adres van regel 0 van pagina 0 - van de betrokken paragraaf. Hieruit volgt, dat wij b.v. regel 3 van een pagina desgewenst zouden kunnen beschrijven als regel 35 van de vorige pagina.

De paragraafletters zijn in eerste instantie twee letters en de programmeur heeft voor de paragraafnamen de keuze uit 169 mogelijkheden, omdat hij zowel voor de eerste als voor de tweede letter een keuze mag doen uit de volgende dertien letters:

Z, E, F, H, K, L, R, S, T, W, U, Y en N.

De paragraafnamen, die met de letter Z beginnen, worden gereserveerd voor de standaardbanden van subroutines; wij komen hier later op terug.

Een adres wordt genoteerd in drie kolommen, van links naar rechts:

- 1e. een kolom voor het regelnummer,
- 2e. een kolom voor de eerste paragraafletter,
- 3e. een kolom voor de tweede paragraafletter, gevolgd door het paginanummer.

Opmerking:

Het vermelden van regelnummer en paginanummer is verplicht, ook wanneer een van deze nummers = 0 is.

Het heeft zin, om de paragrafen, die met dezelfde eerste paragraafletter worden aangeduid, te beschouwen als paragrafen uit een zelfde hoofdstuk. Indien n.l. de eerste paragraafletter gelijk is aan de eerste paragraafletter van het voorafgaande adres, dan mag de eerste letter worden weggelaten! Voor de eerste letter is juist een aparte kolom gereserveerd, om het weggelaten duidelijk zichtbaar te maken: het openlaten in deze kolom fungeert als "aanhalingstekens".

De eerste paragraafletter hoeft dus alleen genoteerd te worden, als deze wisselt, en men kan zich veel schrijf- en ponswerk besparen, als men deze wisselingen zoveel mogelijk vermijdt. Nu is ook het begrip "hoofdstuk" duidelijk: als men paragrafen, die het meeste onderling naar elkaar verwijzen, onderbrengt in een zelfde hoofdstuk, kan men een meer beknopte en overzichtelijke notatie voor het programma verkrijgen.

Met de genoemde paragraafletters worden paragrafen aangeduid, waarvan het beginadres, zoals wij later zullen zien, willekeurig kiesbaar is. Deze beginadressen fungeren dus als vrij kiesbare "nulpunten", ten opzichte waarvan de programmeur zijn adressen kan nummeren. Er bestaan eveneens twee vaste nulpunten, die elk met één letter (nl. X resp. D) worden aangeduid. Deze letters worden "onafhankelijk van hoofdstuk" verwerkt, hun gebruik geldt niet als een wisseling van de eerste paragraafletter in de zin als boven beschreven. De X en de D worden daarom dan ook in de laatste adreskolom op de plaats van de tweede paragraafletter geschreven. Samen met het er op volgende paginanummer, dat na deze letters lopen mag van 0 t/m 31. De X verwijst naar adres 0, het begin van het (levende) geheugen, de D verwijst naar een vast punt in het dode geheugen.

b. De opdracht.

De standaard-programmavellen bevatten voor de notatie van opdrachten 7 kolommen, ten behoeve van de beschrijving nu genummerd van 0 t/m 6. (Links van deze kolommen staat de nummering van 0 t/m 31, rechts is ruimte voor toelichting, notities etc.)

Zij zijn in volgorde gereserveerd voor:

kolom 0: "Conditiereactie" (U, Y of N komt, indien aanwezig, in deze kolom),

kolom 1: "Functiegedeelte", d.w.z. functiecijfer, gevolgd door functieletter(s),

kolom 2: "Regelnummer",

kolom 3: 1ste paragraafletter (meestal oningevuld) adres

kolom 4: 2de paragraafletter, gevolgd door paginanummer.

kolom 5: "Adresverandering" (A, B of C komt, indien aanwezig, in deze kolom. Bij de opdrachten 4T en 6T, waar deze letters immers uitgesloten zijn, komt in deze kolom de index m (≤ 7 resp. ≤ 15).)

kolom 6: "Conditiezetting" (P, Z of E komt, indien aanwezig, in deze kolom).

Samenvattend: tussen de dikke strepen staan functie en adres, daar buiten is ruimte voor de varianten.

Opmerking:

Bij een groot gedeelte der "absolute" opdrachten zal het adresgedeelte fungeren als gewoon getal, waarbij men geenszins er in geïnteresseerd is een aantal malen 32 in een paginanummer af te splitsen! In de beschreven notatie zou het (decimale) getal op de plaats van het regelnummer komen, gevolgd door $\{X0 \mid A\}$. (D.w.z. X omdat het "beginadres", dat bij het regelnummer opgeteld zal worden = 0 moet zijn, dan een 0 voor het 32-voud en tenslotte, in de volgende kolom, de A, die Absoluut aangeeft). Deze combinatie komt zo vaak voor, dat hiervoor een speciale verkorting is ingevoerd: in dit geval mag de X0 in kolom 4 weggelaten worden.

Vermeld is hoe de index m drie of vier bits in het opdrachtwoord bepaalt, rest ons te vermelden, hoe het opdrachtwoord van de varianten afhangt, als deze door letters bepaald zijn;

kolom 5:	$d_{20}:d_{19}$:	kolom 6:	$d_{18}:d_{17}$:	kolom 0:	$d_{16}:d_{15}$:
oningeruuld	0 0	oningeruuld	0 0	oningeruuld	0 0
A	0 1	P	0 1	U	0 1
B	1 0	Z	1 0	Y	1 0
C	1 1	E	1 1	N	1 1

VI. De communicatie-opdrachten.

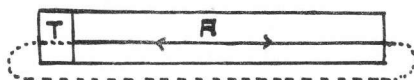
Met de verzamelnaam "communicatie-opdrachten" worden de opdrachten aangeduid met de bijzonderheid, dat het adresgedeelte van de opdracht nimmer verwijst naar een geheugenplaats, maar (volgens een of andere codering) medebepalend is voor de aard van de operatie. Zij kunnen met de functieletters Y en Z genoteerd worden. Voor een gedeelte van deze opdrachten zou dit evenwel leiden tot een tamelijk onoverzichtelijke notatie. Om hieraan tegemoet te komen is de speciale functieletter P ingevoerd.

De symbolen, die na de P in de kolommen voor het adresgedeelte genoteerd worden, hebben een geheel andere betekenis. De paragrafenindeling is niet van toepassing; deze "adressen" veroorzaken nimmer een wisseling van eerste paragraafletter, zoals dit bij de adresnotatie is beschreven. Om hieraan te herinneren, wordt de overeenkomstige kolom bij deze opdrachten altijd oningevuld gelaten.

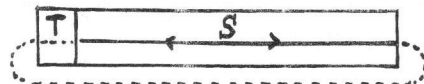
Schuifopdrachten.

In vier zgn. "circuits" kan geschoven worden. Het zijn:

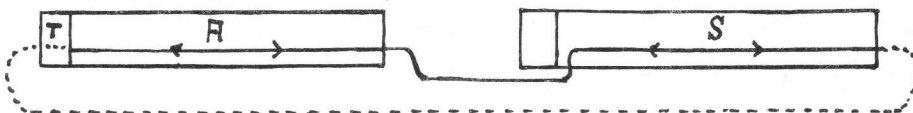
Circuit AA
(27 bits)



Circuit SS
(27 bits)



Circuit AS
(53 bits)



Circuit SA
(54 bits)



Het tekencijfer van het eerstgenoemde register fungeert als "teker cijfer van het circuit"; in de schema's is dit met T aangegeven. Voorts merken wij op, dat het tekencijfer van S niet in het circuit AS is opgenomen.

Zoals de pijltjes in de schema's al aanduiden, kunnen de cijfers in een circuit zowel naar rechts als naar links geschoven worden.

Dit kan nu op twee manieren, "rondschiiven" en "schoonschiiven".

Het rondschiiven behoeft nauwelijks verdere explicatie: de cijfers van het circuit worden cyclisch rondgeschoven over een gegeven aantal plaatsen, hetzij rechtsom, hetzij linksom. Bij het schoonschiiven wordt het circuit als het ware opgeknijpt bij het tekencijfer T van het circuit. (Het gedeelte van de route, dat in de schema's gestippeld is aangegeven, komt dan te vervallen.)

Het tekencijfer T blijft bij schoonschiiven ongemoeid; de uitgeschreven cijfers gaan verloren, de vrijkomende plaatsen worden aangevuld met copieën van T.)

Het type van de schuif wordt op het programma in kolom 1 aangegeven, en wel:

- 0P Schuif rond, naar links,
- 1P Schuif rond, naar rechts,
- 2P Schuif schoon, naar links,
- 3P Schuif schoon, naar rechts.

In kolom 2 noteren wij het aantal plaatsen n, waarover geschoven wordt ($0 \leq n \leq 31$).

In kolom 3 wordt niets ingevuld, in kolom 4 wordt met twee letters het circuit aangeduid.

Ter illustratie:

0P	3	AA		$8[A] \cdot \text{Mod. } (2^{27} - 1) \Rightarrow [A]$
2P	3	AA		$8[A] \cdot \text{Mod. } (2^{26}) \Rightarrow [A]$
3P	1	AS		$\frac{1}{2}(AS) \Rightarrow (AS) \text{ (niet afgerond)}$
0P	0	SS	B	$[S] \cdot 2[B] \cdot \text{Mod. } (2^{27} - 1) \Rightarrow [S] \text{ (mits } 0 \leq [B] \leq 31)$

Bij schuifopdrachten bestaat de U-versie niet. De A-versie bestaat evenmin, de interpretatie van de bits in het adresgedeelte ligt bij deze opdrachten immers al vast. De B- en C-versie bestaan wel, maar moeten met de nodige voorzichtigheid gebruikt worden: als de resulterende n bijvoorbeeld ≥ 32 zou worden, gaat de oorspronkelijke schuifopdracht in een andere communicatie-opdracht over! De P-, Z- en E-versies bestaan alle, met dien verstande, dat onder het resultaat verstaan moet worden de uiteindelijke inhoud van het A-register bij de circuits AA en AS en van het S-register bij de circuits SS en SA.

De registertransporten.

Er is een volledig stel opdrachten, waarmee de inhoud van één van de registers A, S of B, al of niet na tekenwisseling overgenomen kan worden in, hetzij een ander, hetzij hetzelfde register. De kolommen 2 en 3 blijven hierbij oningevuld. De algemene gedaante van deze opdrachten is, met R, $R^* = A, S$ of B:

4P	RR*	$(R) \Rightarrow (R^*)$
5P	RR	$-(R) \Rightarrow (R^*)$

Bij deze opdrachten kunnen de varianten van conditiesettingen reactie volledig gebruikt worden. Voorbeelden:

4P	AS	$(A) \Rightarrow (S)$
5P	BB	$-(B) \Rightarrow (B)$
4P	AA	$(A) \geq +0?$
5P	SS	$(S) \leq -0? \text{ (of } -(S) \geq +0?)$
4P	AB	$(A) \Rightarrow (B) = 0?$

De normeeropdrachten.

Ook bij de normeeropdrachten blijven kolommen 2 en 3 oningevuld. Wat bij de schuifopdrachten over de varianten gezegd is, is ook hier van toepassing; B- en C-correctie zijn hier alleen minder actueel. Normeeropdrachten zijn nl. schuifopdrachten naar links. Het aantal plaatsen, waarover geschoven wordt, is echter niet in de opdracht gegeven: de cijfers in het circuit worden zover naar links geschoven, totdat de meest-significante bit van de tekenbit verschilt. (M.a.w. over het maximum aantal plaatsen, waarbij schoon- en rondschiiven nog geen verschil maakt!) Dit aantal wordt na afloop van de normeeropdracht in B geplaatst.

De normeeropdrachten bestaan alleen voor de circuits AA, SS en AS. Het maximum aantal plaatsen waarover bij normering geschoven wordt, is 26 voor de circuits AA en SS, = 52 voor het circuit AS. Dit aantal wordt alleen bereikt, als de inhoud van het circuit = ± 0 is. In de code worden deze opdrachten genoteerd:

6P	AA	normeer (A)
6P	SS	normeer (S)
6P	AS	normeer (AS)

De stopopdracht.

7P		
----	--	--

Deze opdracht stopt de X-1; hij bestaat in Y- en N-versie.

De interne codering van de schuif- en normeeropdrachten is van links naar rechts als volgt:

- $d_{26} = d_{25} = 1$
- $d_{24} = 0$ voor AA en AS; = 1 voor SS en SA
- $d_{23} = d_{22} = 1$
- $d_{21} = 0$ voor OP, 2P en 6P; = 1 voor 1P en 3P
- $d_{20} \text{ t/m } d_{15} = \text{bepaald door varianten}$
- $d_{14} = d_{13} = \dots, d_8 = 0$
- $d_7 = 0$ voor OP, 1P, 2P en 3P; = 1 voor 6P
- $d_6 = 0$ voor AA en SS; = 1 voor AS en SA
- $d_5 = 0$ voor rondschiiven; = 1 voor schoonschiiven en normeren
- $d_4 \dots d_0 = \text{binair representatie van } n \text{ bij schuifopdrachten.}$
- $\quad \quad \quad = 0 \text{ bij normeeropdrachten.}$

Bij de registertransporten is de codering gegeven door:

$d_{26} = d_{25} = 1$
 $d_{24} = 0$
 $d_{23} = d_{22} = 1$
 $d_{21} = 0$ voor 4P; $= 1$ voor 5P
 d_{20} t/m d_{15} = bepaald door de varianten
 d_{14} ... d_0 = binaire representatie van het getal $256 + 32 p + q$,
 als p, q = 0 voor A } waarbij p betrekking heeft op het "bronregister" (eerste letter) en q op
 = 1 voor S } het bestemmingsregister (tweede letter).
 = 2 voor B }

De representatie van de stopopdracht 7P is

$d_{26} = d_{25} = 1$;
 $d_{24} = 0$
 $d_{23} = d_{22} = d_{21} = 1$
 d_{20} t/m d_{15} = bepaald door de varianten
 $d_{14} = 1$
 $d_{13} = d_{12} = \dots = d_0 = 0$.

De snelle vermenigvuldigingen met 10.

Ten behoeve van een snelle conversie van tien- naar tweetallig en omgekeerd stelsel zijn twee speciale vermenigvuldigopdrachten ingevoerd. Het zijn

$\begin{array}{|l|l|} \hline 6Z:32: \\ \hline 6Z:33: \\ \hline \end{array} \begin{array}{|l|} \hline XX \\ \hline XX \\ \hline \end{array} \begin{array}{l} 10[S] \Rightarrow [AS] \text{ (of } 10\{S\} \Rightarrow \{AS\}) \\ 10[S] \Rightarrow [S] \text{ (onder voorbehoud van capaciteitsoverschrijding)} \end{array}$

Bij de eerste opdracht gaat de oorspronkelijke inhoud van A verloren, bij de tweede blijft deze onaangetaast. De eerste opdracht is speciaal van belang bij de uitvoer, de tweede bij de invoer. Wat bij de laatste opdracht aan de meest significante kant S zou verlaten, gaat verloren; beide opdrachten beschouwen wat betreft de conditiesetting de uiteindelijke inhoud van het S-register als resultaat.

De numerieke waarde van het adresgedeelte dezer opdrachten is 1024, resp. 1025 (dus 2^{10} , resp. $2^{10} + 1$).

De tijdsduur van beide opdrachten is $64 \mu\text{sec.}$ (de normale vermenigvuldigingen vergen $500 \mu\text{sec.}$).

De "echte" communicatie-opdrachten.

De nu volgende opsomming van opdrachten voor bandlezen, bandponsen en typen is volledigheidshalve in deze beschrijving opgenomen. De meeste programmeurs zullen deze opdrachten nooit in hun programma neer behoeven te schrijven, omdat zij van standaard communicatieprogramma's gebruik kunnen maken. (Deze verzorgen dan o.a. tevens de vereiste controles en de conversies van tien- naar tweetallig stelsel en vice versa.)

De bandleesopdrachten.

De gebruikte ponsband heeft vijf gaatjes per rij; de twee meest significante bevinden zich aan de ene kant van het transportgaatje en de drie minst significante aan de andere. De bandleesopdrachten beschouwen de uitgang van de bandlezer als een getal van vijf cijfers - gaatje = 1 en ongeponste positie = 0 - dat aan de meest significante zijde aangevuld wordt met 22 nullen. Dit getal kan in A of S opgeteld of afgetrokken worden, of positief dan wel negatief, naar een van beide registers getransporteerd worden. Dit geschiedt door de opdrachten

$\begin{array}{ l l } \hline 0Y:1: \\ \hline 1Y:1: \\ \hline 2Y:1: \\ \hline 3Y:1: \\ \hline 0Z:1: \\ \hline 1Z:1: \\ \hline 2Z:1: \\ \hline 3Z:1: \\ \hline \end{array} \begin{array}{ l } \hline XP \\ \hline XP \\ \hline XP \\ \hline XP \\ \hline XP \\ \hline XP \\ \hline XP \\ \hline \end{array}$	$(A) + (BL) \Rightarrow (A)$ en step band een rijtje op $(A) - (BL) \Rightarrow (A)$ " " " " " " $+ (BL) \Rightarrow (A)$ " " " " " " $- (BL) \Rightarrow (A)$ " " " " " " $(S) + (BL) \Rightarrow (S)$ " " " " " " $(S) - (BL) \Rightarrow (S)$ " " " " " " $+ (BL) \Rightarrow (S)$ " " " " " " $- (BL) \Rightarrow (S)$ " " " " " "
--	---

De bandponsopdrachten.

De vijf minst significante cijfers van het A- of S-register kunnen, al dan niet na inversie, geponst worden; zij worden naar de zgn. "typ-ponsrelais" TP gestuurd. Het geschiedt door de opdrachten

6Y	1	:	:	XP	+	(A)	⇒	(TP)	en pons en step op
7Y	1	:	:	XP	-	(A)	⇒	(TP)	" " " " "
6Z	1	:	:	XP	+	(S)	⇒	(TP)	" " " " "
7Z	1	:	:	XP	-	(S)	⇒	(TP)	" " " " "

De typopdrachten.

De werking van de typopdracht is volledig vergelijkbaar met die van de bandponsopdracht. Nu echter bepalen de minst significante zes cijfers van A of S, welk symbool getypt wordt. Ook hier wordt gebruik gemaakt van de zgn. typ-ponsrelais TP. Het geschiedt door de opdrachten

6Y	2	:	:	XP	+	(A)	⇒	(TP)	en typ
7Y	2	:	:	XP	-	(A)	⇒	(TP)	en typ
6Z	2	:	:	XP	+	(S)	⇒	(TP)	en typ
7Z	2	:	:	XP	-	(S)	⇒	(TP)	en typ

Het gebruik van zes binalen maakt het mogelijk om zowel cijfers als letters te typen. Ook de "hoofdlettertoets" HL kan bediend worden; deze is houdend, en wordt door de "kleine-lettertoets" kl teruggezet. Welk symbool getypt wordt, is in de onderstaande tabel opgegeven.

(TP)	kl	HL	(TP)	kl	HL	(TP)	kl	HL
0	0	$\frac{3}{4}$	16	/	:	32	l	L
1	1	$\frac{1}{4}$	17	'	"	33	m	M
2	2	$\frac{1}{2}$	18	HL		34	n	N
3	3	£	19	kl		35	o	O
4	4	\$	20			36	p	P
5	5	%	21	a	A	37	q	Q
6	6	f	22	b	B	38	r	R
7	7	&	23	c	C	39	s	S
8	8	(	24	d	D	40	t	T
9	9	)	25	e	E	41	u	U
10		Tab.	26	f	F	42	v	V
11		TWNR x)	27	g	G	43	w	W
12	-		28	h	H	44	x	X
13	+	=	29	i	I	45	y	Y
14	.	;	30	j	J	46	z	Z
15	,	?	31	k	K	47	'	`
						48	^	..

x) TWNR = Terug Wagen, Nieuwe Regel.

56 t/m 63 spatie

De terugleesopdrachten.

Deze opdrachten hebben ten doel een zo volledig mogelijke controle op het typen van resultaten mogelijk te maken. De decimaal of letter welke getypt wordt, kan, in gedecodeerde vorm, van de typrelais TR teruggevoerd worden naar A- of S-register. Door tijdens het typen uit de teruggelezen symbolen het oorspronkelijke woord opnieuw op te bouwen, controleert men niet alleen het typen zelf, doch ook de eventuele deconversie van binair naar tientallig stelsel. Het teruglezen geschiedt door de volgende opdrachten. (Het symbool (TR) is een getal van zes cijfers, dat aan de linkerzijde met 21 nullen aangevuld moet worden.)

0Y	2	:	XP	(A) + (TR) $\Rightarrow$ (A)
1Y	2	:	XP	(A) - (TR) $\Rightarrow$ (A)
2Y	2	:	XP	+ (TR) $\Rightarrow$ (A)
3Y	2	:	XP	- (TR) $\Rightarrow$ (A)
0Z	2	:	XP	(S) + (TR) $\Rightarrow$ (S)
1Z	2	:	XP	(S) - (TR) $\Rightarrow$ (S)
2Z	2	:	XP	+ (TR) $\Rightarrow$ (S)
3Z	2	:	XP	- (TR) $\Rightarrow$ (S)

Voor de communicatie-opdrachten met "adres" 1 XP is het numeriek gedegelte van de opdracht in de interne representatie = $2^9 + 1$, voor die met adres 2 XP is dit = $2^9 + 2$. De functiecijfers worden normaal door de Y of Z en de letters der varianten bepaald. Versie A bestaat niet, versies B en C zijn weinig actueel.

De timing der communicatie opdrachten.

Zoals gezegd, kan de bandlezer per seconde circa 150 symbolen verwerken, de bandponser 25 en de schrijfmachine 10.

Als nude X-1 probeert te lezen, te ponsen of te typen, dan kunnen er zich twee gevallen voordoen. Is er sinds de laatste soortgelijke opdracht zoveel tijd verstreken, dat de betrokken apparatuur weer tot stilstand is gekomen, dan wordt de communicatie-opdracht direct uitgevoerd en de X-1 kan onmiddellijk doorrekenen. Is echter de benodigde apparatuur nog bezet wegens het afwerken van een vorige communicatie-opdracht, dan wacht de X-1 automatisch, totdat het moment gekomen is, waarop de communicatie-opdracht wel uitgevoerd kan worden, en vervolgt daarna pas zijn programma. De programmeur heeft zich bij het gebruik van deze opdrachten niet te bekommeren om de relatieve timing van de apparaten enerzijds en de X-1 anderzijds.

Dat na het gehoorzamen van een communicatie-opdracht de X-1 onmiddellijk door kan rekenen, terwijl de feitelijke voltooiing der operatie nog geruime tijd (nl. 7,40 of 100 m.sec.) vergt, is mogelijk, dankzij het feit dat deze voltooiingen door externe controle-apparatuur geregeld worden. De bandlezer heeft daartoe zijn eigen controle-apparatuur, terwijl bandponser en schrijfmachine van dezelfde controle-apparatuur gebruik maken. Dit heeft tot gevolg, dat de blokkade der bandleesopdrachten enerzijds geheel onafhankelijk is van die van de pons- en typopdrachten anderzijds. Preciese omschrijving der blokkade is als volgt.

Na het accepteren van een bandleesopdracht worden gedurende 1/150ste seconde geen nieuwe bandleesopdrachten geaccepteerd.

Na het accepteren van een ponsopdracht worden gedurende 1/25ste seconde geen nieuwe pons-, typ- of terugleesopdrachten geaccepteerd.

Na het accepteren van een typopdracht wordt gedurende 1/10de seconde geen nieuwe pons-, typ- of terugleesopdracht geaccepteerd.

(Als de typopdracht een Tab-signaal naar de schrijfmachine heeft gestuurd, duurt deze blokkade langer, na een TWNR-signaal veel langer. In deze twee gevallen moet de wagen van de schrijfmachine nl. een grotere afstand afleggen. De tijdsduur van deze blokkades zal instelbaar zijn.)

Ontmoet de X-1 een communicatie-opdracht, die wegens een blokkade niet onmiddellijk geaccepteerd kan worden, dan wacht de X-1 zoals gezegd automatisch, totdat de blokkade opgeheven is en voert dan de opdracht pas uit. Omdat gedurende deze tijd de X-1 werkloos zou staan, is een systeem ontworpen, waarmee het mogelijk is deze wachttijden te vermijden: tijdens de blokkade rekent de X-1 door aan een of ander programma; zodra de blokkade is opgeheven, krijgt de X-1 een "seintje" dat nu de volgende communicatie-opdracht zonder wachttijd uitgevoerd kan worden. Zo'n "seintje" heet een zgn. ingreepsignaal; dit zal later uitvoerig beschreven worden.

VII. Invoer via ponsband.

a. Algemeen overzicht van de functies van het bandleesprogramma.

Om informatie in de machine in te voeren, kan men deze in een geschikte code op band ponsen. Voor opdrachten is deze code gelijk aan de reeds beschreven opdrachtnotatie.

Op het toetsenbord van de X-1 bandponser bevinden zich dus alle gebruikte symbolen. In onderstaande tabel is weergegeven, welke symbolen voorkomen op de toetsen van de ponser; bijna alle toetsen zijn dubbel benoemd. In het tabelletje staat steeds het nevensymbool rechts van het hoofdsymbool. Het laatste loopt van 0 t/m 31 en is gelijk aan de bandwaarde der bijbehorende ponsing

0	O	8	16	P	24	S
1	I	9	17	Z	25	T
2	J	10	18	E	26	W
3	G	11	19	F	27	U
4	M	12	20	H	28	Y
5	Q	13	21	K	29	N
6	V	14	22	L	30	D
7		15	23	R	31	X

Als een geponste band onder de foto-electrische bandlezer ligt, kan hij gelezen worden door een bandleesprogramma. Een dergelijk bandleesprogramma heeft een omvangrijke "vertalende" functie. Immers, de programmeur heeft de opdrachten van zijn programma genoteerd in een code, die voor hem convenient is; daarna zijn de in te voeren opdrachten van zijn programmavellen zonder meer geponst en het is nu aan het bandleesprogramma om uit de symbolen op de band - ook wel "pentades" genoemd - steeds het binaire woord te construeren, dat intern de bedoelde opdracht representeert. Wordt de ponsband daarentegen gebruikt om (decimaal geponste!) getallen in te voeren, dan moet het bandleesprogramma de vereiste conversies van tien- naar tweetalig stelsel uitvoeren. Al deze verrichtingen vatten wij samen onder de term assemblage, waaronder dus verstaan wordt het "rekenwerk" in de ruimste zin, dat nodig is om de binaire cijferrijen te construeren uit groepjes symbolen, die de programmeur (volgens een of andere code) heeft neergeschreven.

Behalve dat het bandleesprogramma de pentades aldus tot woorden "assembleert", is het vervolgens eveneens in staat, deze woorden te "verwerken". De twee verwerkingsmogelijkheden zijn "schrijven" en "vergelijken". Als de verwerkingscyclus "schrijft", worden alle opgebouwde woorden op de daarvoor bestemde adressen in het geheugen ingevuld; dan vindt dus de werkelijke invoer van informatie naar het geheugen plaats. Als de verwerkingscyclus "vergelijkt", wordt elk door de assemblage opgebouwd woord vergeleken met de inhoud van het ermee corresponderende adres; in geval van discrepantie stopt de machine en hiermee is een van de methoden gegeven om de invoer van gegevens aan controle te onderwerpen. Andere toepassingen van vergelijkend bandlezen zullen wij later ontmoeten.

Behalve de pentadegroepen, waaruit de in te voeren woorden geassembleerd worden, komen er op de ponsband nog andere pentadecombinaties voor, de zgn. directieven. Zij geven op een of andere wijze een nadere specificatie van de beide verrichtingen (assemblage en verwerking) van het bandleesprogramma en vallen dan ook hoofdzakelijk in twee groepen uiteen. De eerste groep omschrijft de assemblage nader. Hieronder vallen de zgn. soortspecificaties, die vastleggen, volgens welke regels de assemblage de woorden uit de pentades op moet bouwen. Verschillende soorten zijn b.v. opdrachten en getallen. De soortspecificatie geeft dus eigenlijk aan, in welke "taal" (code) tot nader order de in te voeren informatie op de band is geponst.

Het aantal verschillende soorten kan altijd uitgebreid worden; in aanmerking komen b.v. getallen van dubbele lengte, getallen met drijvende komma etc.

De tweede groep directieven, met als belangrijkste de zgn. adresindicatie en controlewisseling, heeft betrekking op de verwerkingscyclus van het invoerprogramma.

De adresindicatie beheerst, op welke geheugenplaatsen de gevormde binaire woorden geschreven worden of met de inhoud van welke geheugenplaatsen zij vergeleken worden; het tweede directief bestuurt of er geschreven dan wel vergeleken wordt. Met de adresindicatie wordt aangegeven in welk adres het eerstvolgende op te bouwen woord moet (komen te) staan; tenzij anders aangegeven - door een nieuwe adresindicatie b.v.! - hebben opeenvolgende woorden op de band betrekking op opeenvolgende adressen in het geheugen. Als men dus - wat meestal het geval is - een aaneensluitend stuk in het geheugen met informatie wil vullen, hoeft men alleen de bestemming van het eerste woord van het rijtje met een adresindicatie expliciet aan te geven.

Bij de adresnotatie is beschreven, hoe adressen genummerd worden ten opzichte van het begin van paragrafen. Deze beginadressen worden bij de opbouw der adressen verwerkt als additieve parameters; vóórdat dit echter kan gebeuren, moeten deze parameters wel aan de machine zijn meegedeeld. Dit geschiedt met behulp van speciale directieven, de zgn. "parameterbepalingen"; zij vormen samen de zgn. "voorponsing", die zich altijd aan het begin van de band bevindt.

Uit het voorgaande is op te maken, dat het werkend bandleesprogramma tot dusver vijf soorten informatie bijhoudt:

- 1e. de (laatst gegeven) soortspecificatie,
- 2e. de waarde der (gespecificeerde) parameters,
- 3e. het adres, dat aan de beurt is gevuld, dan wel vergeleken te worden (dit wordt na elk verwerkt woord met één verhoogd),
- 4e. of de verwerkingscyclus schrijft, dan wel vergelijkt,
- 5e. de laatste expliciet vermelde eerste paragraafletter; (deze eerste paragraafletter mag in geval van herhaling immers weggelaten worden; in de X-1 moet dan bekend zijn, welke eerste paragraafletter er is weggelaten!)

Tenslotte vermelden wij, dat eventuele extra pentades X door het bandleesprogramma worden overgeslagen aan het begin van informatie-eenheden (directieven, opdrachten, getallen etc.). De X (bandwaarde = 31) komt overeen met de volledig geponste pentade en kan als "Erase" gebruikt worden. Dit stelt de ponser in staat om tijdens het ponsen gemerkte fouten onmiddellijk te corrigeren: beginnend bij de eerste pentade van de mislukte informatie-eenheid overponst hij alles, wat er al van geponst was, met X'en en ponst de eenheid opnieuw.

In overeenstemming hiermede zijn de codes zo gekozen, dat geen enkele informatie-eenheid ooit met een X begint.

b. Voorponsingen.

Wij resumeren uit het voorgaande:

Ten behoeve van de adresnotatie wordt het geheugen ingedeeld in paragrafen (van maximaal 17 aaneensluitende pagina's van 32 woorden elk). Elke paragraaf wordt door twee paragraafletters aangegeven; paragrafen met gemeenschappelijke eerste paragraafletter vormen samen een hoofdstuk. Op welke geheugenplaatsen nieuwe paragrafen beginnen, mag de programmeur vrij kiezen.

Het is verstandig voor de beginregels van paragrafen een 32-voud te kiezen; men bereikt daarmee, dat de vijf minst significante binalen van het adres gelijk zijn aan het regelnummer. Als het een langere paragraaf betreft is het verstandig voor het beginadres zo mogelijk een veelvoud van een hogere macht van twee te kiezen. Op deze wijze vergemakkelijkt men de inspectie bij de machine aanzienlijk,

Om aan de machine het beginadres van een paragraaf mede te delen, gebruikt men een speciaal directief, de zgn. parameterbepaling. Deze bestaat uit:

- 1e. in kolom 0 de letters DP, gevolgd door
- 2e. in kolom 1 de paragraafletters, waaraan de nieuwe waarde toegekend wordt (beide letters verplicht!), gevolgd door
- 3e. de nieuwe waarde, als adres (d.w.z. regelnummer, paragraafletter(s) en paginanummer). Hierbij mag slechts van paragraafletters gebruik gemaakt worden, waarvoor eerder al een parameterbepaling is ingelezen! In de eerste parameterbepaling op de band zal dus met de letter X (of D) het adres zijn vastgelegd!

De paragrafenindeling is ingevoerd om tegemoet te komen aan de behoeften van de programmeur. Slechts heel eenvoudige programma's nl. worden als één geheel geconcipeerd en van de eerste tot de laatste opdracht in volgorde opgeschreven. Alras wordt een berekening liever uitgesplitst in onderdelen, die elk voor zich een tamelijk isoleerbare functie hebben. Het is van groot belang, dat deze onderdelen goed gekozen worden; hoe duidelijker geïsoleerd de functie van deze onderdelen is, des te overzichtelijker is het programma. De normale praktijk is nu, dat men elk onderdeel een eigen paragraaf geeft, aangegeven door eigen paragraafletters. Deze verschillende paragraafletters maken ten eerste het programma voor het oog "sprekender"; ten tweede is het nu mogelijk om aan het programmeren van de ene paragraaf te beginnen, terwijl er nog één of meer andere paragrafen onvoltooid zijn. Van een onvoltooid paragraaf weet men nl. nog niet hoeveel ruimte hij in het geheugen zal beslaan, maar dankzij het feit, dat de nulpunten der paragrafen willekeurig kiesbaar zijn, kan men deze keuze uitstellen, totdat alle paragrafen voltooid zijn. Dan is de lengte van de paragrafen bekend en men kan het geheugen naar believen indelen. Welke paragrafen men onderbrengt in eenzelfde hoofdstuk, wordt alleen gesuggereerd door de logische samenhang der paragrafen en heeft niets te maken met de uiteindelijke plaatsing der paragrafen in het geheugen. Paragrafen uit één hoofdstuk hoeven, in tegenstelling tot pagina's uit een paragraaf, niet aansluitend in het geheugen te staan.

Het rijtje parameterbepalingen, dat voor elk programma de geheugenindeling vastlegt, wordt de generale voorponsing genoemd. Is dit bandje ingelezen, dan zijn de beginpunten der paragrafen vastgelegd en de rest van de band kan worden ingelezen.

Een tweede toepassing van de vrije plaatsing der paragrafen is de wijze, waarop standaardsubroutines in een programma geïncorporeerd worden. Terwijl de programmeur hierbij gebruik maakt van de standaardbanden - althans van een mechanisch vervaardigde kopie - heeft hij toch de volledige vrijheid bij de beslissing in welke geheugenplaatsen hij de betrokken standaardsubroutines wil opbergen.

Een standaardsubroutine is meestal één paragraaf en aan de gebruiker wordt overgelaten, waar hij deze paragraaf in het geheugen wil laten beginnen. Op het standaardbandje van de subroutine is de eigen paragraaf met een bepaalde paragraafnaam aangeduid. Het enige, wat de programmeur vooraan het standaardbandje hoeft toe te voegen, is een zgn. specifieke voorponsing, waarin aan de betrokken parameter de gewenste waarde wordt gegeven. Alle subroutines zijn als regel geponst in termen van dezelfde paragraafletters (nl. die met een Z als eerste letter); een dergelijke paragraafnaam kan dus in een programma, dat van een aantal subroutines gebruik maakt, enige malen met verschillende betekenis op de band voorkomen. Tengevolge hiervan zijn de lettercombinaties beginnend met een Z niet geschikt voor referentie naar een willekeurig adres in het geheugen en mogen zij niet in de generale voorponsing gedefinieerd worden. De gebruikelijke taktiek is, dat de paragrafen der standaardsubroutines in feite dubbel benoemd zijn: een paragraafnaam, beginnend met een Z en een benaming, waarvan de betekenis is vastgelegd in de generale voorponsing. De eerste benaming fungeert alleen als middel om van de standaardband gebruik te kunnen maken. Welke paragraafnaam uit de Z-groep hiervoor precies gebruikt wordt, hoeft de programmeur zich alleen te realiseren bij de opstelling van de specifieke voorponsing voor deze subroutines; deze nieuwe definitie van de betrokken parameterwaarde, alsmede elke referentie naar deze subroutine in het hoofdprogramma, geschiedt in termen van een paragraafnaam, die vastgelegd is in de generale voorponsing. Tevens bereikt men, dat op deze manier de uiteindelijke plaats van een subroutine gewijzigd kan worden, door alleen maar de generale voorponsing te wijzigen.

c. Soortspecificaties.

De meest gevarieerde groep directieven is die der soortspecificaties. Zij delen mede, volgens welke code de nu volgende informatie-eenheden tot nader order (d.w.z. tot de volgende soortspecificatie) geponst zullen zijn.

De structuur van het bandleesprogramma maakt het zonder veel omslag mogelijk - ook naderhand - nieuwe soortspecificaties toe te voegen. Tevens is met de soortspecificatie bereikt, dat nieuwe codes onafhankelijk van de reeds bestaande gekozen kunnen worden; dit is een van de belangrijkste aspecten van de flexibiliteit van de organisatie voor de X-1.

Wij laten hier vier soortspecificaties volgen. De soortspecificatie voor getallen van dubbele lengte of met drijvende komma beschouwen wij als onderdeel van de bijbehorende interpretatieve programma's en deze vallen buiten het bestek van deze beschrijving.

DO "Skip blank tape".

Het directief DO duidt aan, dat vooralsnog onbekend is, volgens welke code de assembleroutine de symbolen op de band zal interpreteren. Dit ontbreken van een "echte" soortspecificatie wordt dus ook als soortspecificatie behandeld.

Als het directief DO gelezen is, skipt de machine nullen en pentades X, totdat een D gevonden wordt. Ontmoet het bandleesprogramma een pentade $\neq$ D, dan stopt de machine ter indicatie van een fout, nl. het ontbreken van een soortspecificatie (die, als alle directieven, met de letter D begint).

Als de nieuwe D het inleidingssymbool is van een niet soortspecificerend directief - zoals b.v. de parameterbepaling of de adresindicatie (zie onder) - dan gaat na verwerking van dit directief het bandleesprogramma rustig door blank tape (en X) te skippen, totdat tenslotte een (echte) soortspecificatie gelezen wordt. Aan het blank skippen is dan voorlopig een einde gekomen.

Als men de X-1 normaal start in het bandleesprogramma, om een nieuwe band in te lezen, wordt de soortspecificatie automatisch ingesteld op "Skip blank tape" als boven omschreven.

DI "Lees opdrachten" (Read Instructions).

Na deze soortspecificatie leest het bandleesprogramma tot nader order (d.w.z. tot een nieuwe soortspecificatie) uitsluitend opdrachten en directieven.

Opdrachten worden, precies zoals ze genoteerd worden, van links naar rechts geponst; voor oningevulde kolommen wordt niets geponst. Aan het begin van elke opdracht worden eventuele extra pentades X door het bandleesprogramma geskipt.

Het paginanummer, evenals de index m bij de speciale sprongopdrachten, moet in één aanslag geponst worden. Het regelnummer mag, indien $\neq 31$, in één aanslag geponst worden. Verder: als men tussen regelnummer en paginanaam een minteken inlast, wordt het regelnummer negatief in rekening gebracht. (Het is dus een minteken met terugwerkende kracht.) Het totale adres mag hierbij niet negatief worden; deze beperking is niet essentieel voor de speciale toepassing waarvoor deze faciliteit is ingelast. Bij de opdrachten met functieletter P (OP t/m 3P) is dit minteken niet toegestaan.

DN "Lees getallen" (Read Numbers).

Na de soortspecificatie DN leest het bandleesprogramma tot nader order uitsluitend directieven en getallen van enkele lengte. Aan het begin van elk getal (d.w.z. vóór het teken) worden pentades X geskipt; voor de ponsing der getallen wordt alleen gebruik gemaakt van de toetsen voor de cijfers van 0 t/m 9, het plus- en minteken en de punt.

Na DN is het assemblageprogramma in staat de conversie van tien- naar tweetallig stelsel te verzorgen voor gehele getallen en echte breuken. Alle getallen beginnen met een teken (+ of -). Het teken moet altijd vermeld worden en mag dus niet b.v. worden weggelaten bij positieve getallen; op de band fungeren de tekens nl. tevens als scheiding tussen de decimale cijfers van opeenvolgende getallen.

Gehele getallen bestaan uit het teken, gevolgd door de decimale cijfers; non-significante nullen aan het begin mogen weggelaten worden. In absolute waarde moet het getal uit de aard der zaak kleiner zijn dan $2^{26} = 6710\ 8864$.

Bij breuken komt na het teken eerst de punt (komma) en vervolgens de decimale cijfers; nonsignificante nullen aan het einde mogen verder weggelaten worden. In verband met de woordlengte van de X-1 is het, hoewel toegestaan, tamelijk zinloos de breuken in meer dan acht decimalen achter de komma in te voeren.

Tussen teken en komma mag een willekeurig aantal nullen ingelast worden. Wij vestigen er de aandacht op, dat na DN breuken en gehele getallen elkaar zonder speciale directieven mogen afwisselen. Hier worden dus twee verschillende interpretaties van het binaire woord door één soortspecificatie bestreken; de keuze tussen deze twee mogelijkheden wordt steeds bepaald door het al of niet optreden van de komma.

DB "Lees binair woord".

Binaire woorden worden door de X-1 zelf uitgeponst. Wij volstaan hier met de vermelding, dat aan de 27 bits van een woord bij het uitponsen een pariteitsbit toegevoegd wordt. Deze pariteit wordt door het bandleesprogramma gecontroleerd.

Een veel voorkomende toepassing van het ponsen van binaire band is het uitponsen van de inhoud van een stuk van het geheugen. Zodra de X-1 een dergelijke band heeft geponst, leest men deze band vergelijkend. Daarbij wordt gecontroleerd, of de X-1 de inhoud van het geheugen correct heeft uitgeponst. Tevens wordt daardoor gecontroleerd, of bij het uitponsen steeds het correcte pariteitscijfer is toegevoegd.

d. Directieven voor de verwerkingscyclus.

Vijf directieven hebben een besturende invloed op de verwerkingscyclus van het bandleesprogramma.

DA Adresindicatie.

Het directief voor de adresindicatie dient om de geheugenplaats voor het eerstvolgende woord vast te leggen. De letters DA dienen daartoe gevolgd te worden door het betroffen adres, d.w.z. regelnummer, paragraafletter(s) en paginanummer.

DX Skip aantal adressen.

Na het directief DA hebben de woorden in de volgorde, waarin ze door de assembleroutine worden afgeleverd, betrekking op successieve adressen. Omdat het vaak voorkomt, dat men een paar plaatsen open wil laten - omdat ze door het programma worden ingevuld - is speciaal hiervoor het directief "Skip aantal plaatsen" ingevoerd. De letters DX moeten daartoe gevolgd worden door het aantal plaatsen, dat door de verwerkingscyclus overgeslagen moet worden. Dit aantal is maximaal 31 en moet met één aanslag geponst worden.

DC Controlewisseling.

Een geprogrammeerde wissel legt vast, of de verwerkingscyclus schrijft, dan wel vergelijkt. Door het directief DC kan de stand van deze wissel omgezet worden: was de verwerkingscyclus aan het schrijven, dan gaat hij nu vergelijken en omgekeerd.

De normale manier, waarop programma geponst wordt, is de volgende. Na een stuk blank tape pons men een pagina programma. Dit begint met minstens twee directieven (adresindicatie en soortspecificatie) en wordt afgesloten door DC DO en een stukje blank tape. Onmiddellijk hierachter pons men een compleet duplicaat. Of men geen ponsfout heeft gemaakt, controleert men visueel door eerste en tweede helft over elkaar te leggen en samen tegen het licht te houden. Als een dergelijk zgn. "zelfcontroleerend bandje" ingelezen wordt, dan wordt met de eerste helft de informatie in het geheugen gebracht, met de tweede helft wordt het inlezen gecontroleerd. Dankzij de herhaling van het directief DC blijft tenslotte de verwerkingscyclus "schrijvend" achter, gereed om het volgende zelfcontroleerende stuk te lezen.

Het bandleesprogramma kan op twee wijzen gestart worden, nl. schrijvend, zowel als vergelijkend. Het eerste is de normale manier; als men echter het bandleesprogramma vergelijkend start, is het directief DC zonder effect! Dit is gedaan met het oog op de volgende toepassing. Een gevreesde programmafout is, dat door een "verdwaalde" schrijfo opdracht iets van het programma overschreven wordt. Vrees men dit tijdens het testen, dan leest men de band opnieuw in, maar start vergelijkend. De verwerkingscyclus gaat dan nooit over naar de schrijvende toestand.

DS Stop.

Het directief DS indiceert het einde van de band; het bandleesprogramma houdt hierdoor op te werken. Als het bandleesprogramma door de operateur gestart is, stopt de X-1, nadat het directief DS gelezen is.

DE Exit.

Het directief "Exit" bestaat uit de letters DE, gevolgd door een adres. De besturing verlaat dan het bandleesprogramma en springt naar het aangegeven adres. Met dit directief kan men dus via de band een programma starten.

e. Opmerkingen over het gebruik der directieven.

De bovenbeschreven directieven brengen alle een speciale wijziging in de "toestand" van het bandleesprogramma, en ook alleen maar de vermelde wijziging! In alle andere opzichten laten zij de toestand van het bandleesprogramma ongewijzigd! Soortspecificaties veranderen dus niets aan de instelling van de verwerkingscyclus, de adresindicatie laat de soortspecificatie en de wisselstand ongewijzigd etc. Een gevolg hiervan is, dat de volgorde van soortspecificatie en adresindicatie aan het begin van een band willekeurig is

Wat betreft de zelfcontrolerende banden tenslotte nog het volgende. Het is verstandig om elk zelfcontrolerend bandje, afgezien van de benodigde generale voorponsing, met de nodige directieven te maken tot een verder volledig afgerond geheel. Dit kost weliswaar enige extra symbolen, maar men omzeilt hiermee een mogelijkheid tot het maken van vergissingen. Dit impliceert tevens, dat men zich aan moet wennen om boven aan elke pagina in de adres-indicatie de eerste paragraafletter altijd expliciet te vermelden. Men sluit zodoende tevens aan het normale gebruik van aanhangstekens nauwer aan. Een en ander komt de overzichtelijkheid van het programma en de hanteerbaarheid van de band niet weinig ten goede.

VIII. Synchronisatie met externe apparatuur.

Allerlei apparaten voor in- en uitvoer van gegevens kunnen in principe aan de X-1 gekoppeld worden. Behalve bandlezer, bandponser en schrijfmachine noemen wij ponskaartenapparatuur zoals reproductrices, een sorteermachine en een tabelleermachine, alsmede een snelle printer en apparatuur voor magnetische banden.

In het voorafgaande zijn de opdrachten met betrekking tot bandlezer, bandponser en schrijfmachine uitvoerig beschreven. Analooq zullen er voor eventueel andere aangekoppelde apparaten passende communicatie-opdrachten in de code van de X-1 opgenomen worden; zij vallen echter buiten het bestek van deze algemene beschrijving. Wel wijden wij hier onze aandacht aan de oplossing van een algemeen probleem, dat zich bij koppeling van de X-1 met externe apparatuur voordoet, nl. dat der synchronisatie.

Immers, de zojuist genoemde opdrachten (die deel zullen uitmaken van een zgn. communicatieprogramma) stellen de X-1 wel in staat om contact op te nemen met een dergelijk apparaat, maar veelal zal de constructie daarvan met zich meebrengen, dat zo'n communicatieprogramma niet op elk willekeurig moment uitgevoerd kan worden! Wij hebben b.v. al gezien, dat na de uitvoering van een typopdracht er minstens 100 m.sec. verstreken moet zijn - TAB en TWRN uitgezonderd, waarna deze periode langer duurt - voordat de volgende typopdracht uitgevoerd kan worden. Bij een ponskaartenapparaat, waardoor in een vast ritme kaarten verwerkt kunnen worden, is "het eigen tijdsbesef" nog geprononceerder en dit heeft de wenselijkheid ten gevolge (een wenselijkheid, die zo dringend kan zijn, dat wij soms gerust van een eis mogen spreken), dat het moment, waarop de X-1 het onderhavige communicatieprogramma uitgaat voeren, niet zozeer door de X-1, als wel door het externe apparaat bepaald zal worden. (Voor de schrijfmachine is dit b.v. het moment, waarop de volgende typopdracht geaccepteerd kan worden, voor een reproductrice b.v. is dit het moment, waarop zojuist een kaart de leesborstels gepasseerd is, etc.)

Om de X-1 in staat te stellen aan een specifiek communicatieprogramma binnen een extern bepaald tijdsbestek te beginnen, is de zgn. ingreep geschapen. De ingreep bestaat in grote trekken hieruit, dat de X-1, zodra een dergelijk tijdsbestek aanbreekt, het programma dat uitgevoerd wordt zo snel mogelijk onderbreekt, door middel van een zgn. ingreepprogramma het betrokken apparaat "verzorgt" en na afloop het tijdelijk onderbroken programma vervolgt, alsof er niets gebeurd was.

Wij kunnen ons de ingreep zo voorstellen, dat na voltooiing van een opdracht niet normaal de volgende opdracht, maar in plaats daarvan een speciale subroutinesprong in het opdrachtregister wordt geplaatst. Doordat de tussengelaste opdracht een subroutinesprong is, wordt de momentane inhoud van de opdrachtsteller en de conditie etc. in de overeenkomstige Λ_m gered; de tussengevoegde subroutinesprong verwijst de besturing naar het begin van het ingreepprogramma. Dit zal in het algemeen beginnen, de momentane inhoud van de registers A, S en B op speciaal hiervoor gereserveerde geheugenplaatsen in veiligheid te brengen. Daarna gaat het ingreepprogramma de gewenste informatietransporten van X-1 naar apparaat en/of vice versa uitvoeren; na afloop van het ingreepprogramma wordt met behulp van de in veiligheid gebrachte gegevens de status quo hersteld en de onderbroken berekening wordt voortgezet.

Met andere woorden: zodra er urgenter werk te doen is, wordt de X-1 automatisch even "uitgeleend" aan een ingreepprogramma, dat dan dit "haastkarweitje" onmiddellijk uitvoert. Vergeleken bij oplossingen zonder ingreep wordt hierdoor het nuttig effect van de X-1 enorm verhoogd, niet in de laatste plaats, doordat het nu mogelijk is de X-1 samen te laten werken met een aantal, onderling gesynchroniseerde apparaten.

In het laatste geval kan men zich de wenselijkheid voorstellen, dat verschillende apparaten (eventueel) hun eigen ingreep bewerkstelligen, waarbij sommige ingrepen preferent boven andere plaats moeten kunnen vinden. Een (klein) ingreepprogramma, dat in een kort tijdsbestek moet plaats vinden, mag b.v. niet onderbroken worden door een (langdurige) ingreep, die (waarschijnlijk absoluut gesproken) minder haast heeft; andersom moet dan wel mogelijk zijn.

interruptie permit

Welke ingrepen op elk ogenblik zijn toegestaan en welke niet, wordt aangegeven door de zgn. "luistervergunning"; deze kan door opdrachten gewijzigd worden: ingrepen vinden dus automatisch plaats, maar slechts voorzover het programma dit toestaat!

Door m.b.v. de luistervergunning een ingreep tijdelijk te verbieden, wordt het effectueren van deze ingreep uitgesteld, totdat hij na passende wijziging van de luistervergunning toelaatbaar wordt. Om dit uitstel mogelijk te maken, veroorzaken apparaten (of onderdelen daarvan) die "verzorging" behoeven, niet direct een ingreep, maar zenden zij slechts een houdend zgn. "ingreep-sig-naal" naar de X-1. Wanneer dit ingreepsig-naal de ingreep tengevolge heeft, zal dus van de luistervergunning afhangen. Er zijn voorzieningen getroffen voor hoogstens zeven verschillende ingrepen. Hierbij is het mogelijk, dat verschillende ingreepsignalen dezelfde ingreep tengevolge hebben. Alle ingreepsignalen, die dezelfde ingreep effectueren, vormen samen een zgn. klasse. In de luistervergunning is per klasse vastgelegd, of de ingreepsignalen "gehoord" worden of niet, m.a.w. de luistervergunning kan bepaalde ingrepen verhinderen. De klassen zijn genummerd van 1 t/m 7; de ingrepen van klasse 7 kunnen niet onderdrukt worden (zie onder) en de luistervergunning omvat daarom zes bits.

Wat betreft de ingreepsignalen is de aanwezigheid van één uit de klasse al voldoende om de ingreep te bewerkstelligen. Het ingreepprogramma nu moet kunnen beschikken over het gegeven welke ingreepsignalen uit eigen klasse aanwezig zijn. Hiertoe is per klasse een speciale opdracht ingebouwd, die het zgn. klassewoord uitleest: elk ingreepsig-naal uit die klasse correspondeert met een bepaalde bit uit het klassewoord, waarin de aan- of afwezigheid van dit ingreepsig-naal is vastgelegd. Zodra de aanwezige ingreepsignalen door het uitlezen van het klassewoord tot het rekenorgaan zijn doorgedrongen, houden zij op te bestaan. (Er is hierbij van uitgegaan, dat een klasse voorlopig niet meer dan 2⁷ ingreepsignalen zal omvatten. Zo kan b.v. een reproductrice gelijktijdig vier verschillende ingreepsignalen naar de X-1 zenden, nl. wanneer zojuist een kaart de lees- dan wel de controleborstels van de lees- dan wel de ponsbaan gepasseerd is.)

Tenslotte bestaat er een methode om alle ingrepen te verhinderen, zonder daarbij de luistervergunning te wijzigen. De X-1 kent nl. twee toestanden, "doof" resp. "horend". Boven is de toestand beschreven bij horende X-1; als de X-1 doof is, wordt er ongeacht de luistervergunning geen enkele ingreep verwezenlijkt. Overgang van doof naar horend en omgekeerd kan door speciale opdrachten bewerkstelligd worden. Tevens wordt de X-1 automatisch doof, zodra een ingreep heeft plaats gevonden.

Met behulp van de luistervergunning wordt de hiërarchie der ingrepen geregeld. Het doof en horend kunnen zijn is uit andere noodzaak geboren. Met het invoeren van de ingreep is nl. een volledig nieuw element ingelast: de werkelijke volgorde waarin de X-1 de verschillende handelingen uitvoert, is nl. niet meer door het programma alleen gegeven. Om het programma desondanks in staat te stellen, zijn functie naar behoren te vervullen, is het gewenst, dat de onzekerheid over het al of niet optreden van de ingreep over bepaalde "kritieke stukjes" programma uitgebannen kan worden; dit is door doof zijn mogelijk geworden en wel, zonder de luistervergunning te verstoren. Doofheid zal zich als regel slechts over enkele opdrachten uitstrekken.

Nu kan de volledige reddende functie van de subroutinesprong beschreven worden. In λ_m wordt nl. van links naar rechts vastgelegd

d_{26}	stopbit	= 1 wel	stoppen bij herstellende sprong
		= 0 niet	
d_{25}	bit luistervergunning klasse 6	}	= 1 als ingreep toegestaan, en
$\vdots$			
d_{20}	bit luistervergunning klasse 1		
			= 0 indien ingreep verboden is.
d_{19}	doof-horend indicatie	= 1 als horend	
		= 0 als doof	
d_{18}	(C) = Inhoud conditieregister		
d_{17}	(LTR) = Inhoud Laatste Tekenregister		
d_{16}	(OF) = Inhoud overflow indicatie		
d_{15}	= 0 (ongebruikt)		
d_{14}	}		= (T), Inhoud opdrachtsteller T op het moment van de ingreep
$\vdots$			
$\vdots$			
d_0			

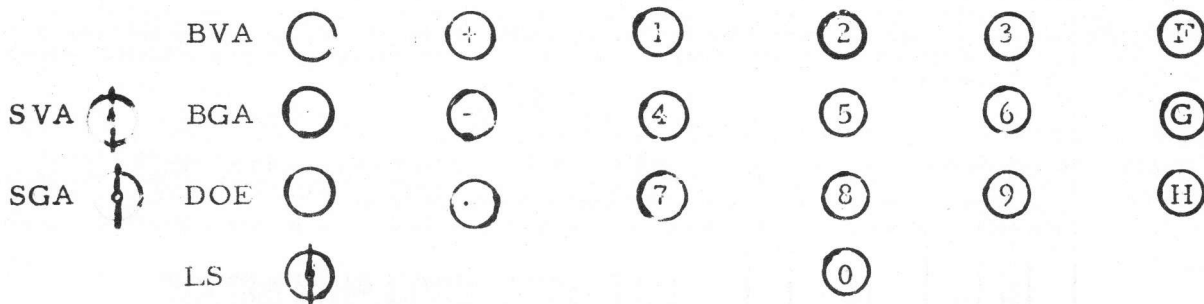
Opmerking:

d_{26} van λ_m wordt = 1 als de subroutinesprong uitgevoerd wordt, terwijl SVA ("STOP VOLGENDE OPDRACHT" zie hierna) aanstaat.

enige knopje
andere

IX. Het bedieningspaneel van de X-1.

Op het vrijwel verticale gedeelte achter het tafelblad van de X-1 bevinden zich een aantal schakelaars en indicatielichtjes. Op het tafelblad zelf bevindt zich een paneeltje met negentien druktoetsen en drie schakelaars:



De twee schakelaars, die zich links op dit paneeltje bevinden, beheersen, hoe de machine gestopt wordt.

SVA: "Stop Volgend Adres". Wanneer deze schakelaar bij werkende machine in wordt gezet (d.w.z. van de operateur af), voltooit de machine de opdracht, waaraan zij bezig is en stopt vervolgens. De opdrachtteller bevat dan het adres van de volgende opdracht. Deze schakelaar wordt o.a. gebruikt, om de machine instantaan te stoppen. Als men de machine start, terwijl SVA instaat, stopt zij na voltooiing van de eerste opdracht.

Indien een subroutinesprong wordt uitgevoerd, terwijl SVA in staat, wordt d_{26} (= het zgn. stopbit) van de bijbehorende Δ gelijk 1 gemaakt.

SGA: "Stop Gekozen Adres". Met deze schakelaar kan men de machine op een bepaald punt in het programma stoppen. Dit punt legt men vast met de vijftien schakelaars van het zgn. stopadres. Wanneer de schakelaar SGA in staat, stopt de X-1 zodra in het programma de opdracht in het stopadres is uitgevoerd; weer bevat de opdrachtteller het adres van de volgende opdracht.

De vijftien schakelaars van het stopadres bevinden zich rechts aan de bovenkant van het verticale paneel achter het tafelblad; men kan er een adres binair in vastleggen; schakelaar naar beneden = 0, naar boven = 1.

Het starten van de machine geschiedt met druktoetsen.

BVA: "Begin Volgend Adres". Wanneer men deze knop indrukt en weer loslaat, gaat de machine werken, te beginnen bij de opdracht, waarvan het adres op dat ogenblik in de opdrachtteller staat. Indien SVA uit staat, gaat de machine dus gewoon verder (dit heet "doorstarten"). Staat SVA in, dan stopt de machine weer onmiddellijk na de uitvoering van deze opdracht. BVA wordt dus zowel gebruikt om het gestopte programma door te starten, als ook om een (stukje) programma opdracht voor opdracht uit te voeren.

BGA: "Begin Gekozen Adres". Met deze knop bereikt men, dat de machine een programma gaat uitvoeren, te beginnen bij de opdracht, waarvan het adres is vastgelegd in de vijftien schakelaars van het zgn. beginadres. Men kan zich voorstellen, dat de startknop BGA hetzelfde doet als BVA, na eerst het beginadres in de opdrachtteller overgenomen te hebben.

De vijftien schakelaars van het beginadres bevinden zich links aan de bovenkant van het verticale paneel. Adressen worden hier binair ingezet, juist als bij het stopadres.

DOE: Hieronder bevindt zich de drukknop DOE. Deze knop dient in hoofdzaak voor testdoeleinden. Als men deze knop heeft ingedrukt voert de X-1 de opdracht uit, die gespecificeerd staat in de zevenentwintig zgn. woordschakelaars; zij bevinden zich onderaan het verticale paneel. Staat SVA in, dan wordt de opdracht uit de woordschakelaars éénmaal uitgevoerd; staat SVA uit, dan wordt die opdracht bij herhaling uitgevoerd, totdat SVA in wordt gezet. Mits deze opdracht geen gehoorzaamde sprongopdracht is, wordt door de knop DOE de inhoud van de opdrachtteller niet aangestart.

(De zgn. "Laatste Schakelaar" LS, die zich onder de knop DOE bevindt, is voor de programmeur niet van belang: het omzetten van deze schakelaar is de handeling, waarmee het aanzetten van de X-1 voltooid wordt.)

De zestien resterende drukknoppen op dit paneeltje worden samengevat onder de naam "Handregister". Bij werkende X-1 heeft het indrukken van deze knoppen geen effect, bij stilstaande X-1 starten zij de machine door middel van de ingreep van klasse 7 (zie boven). Zij worden gebruikt om de X-1 service-handelingen te laten verrichten. De gewenste handelingen worden slechts uitgevoerd als SVA uit staat.

Op het verticale paneel zijn reeds genoemd de schakelaars van het beginadres, die van het stopadres en de woordschakelaars. Elk van deze drie groepen schakelaars correspondeert bovendien met een bepaald woord in het dode geheugen:

(1D0) = beginadres
(2D0) = stopadres } aan de meest significante kant aangevuld met twaalf nullen.
(3D0) = consolewoord (d.w.z. inhoud van de woordschakelaars).

Deze drie woorden passen, hoewel zij variabel zijn, toch in het dode geheugen: zij kunnen nl. door het programma wel gelezen, maar niet gewijzigd worden! Het grote belang van de uitleesbaarheid van deze woorden is daarin gelegen, dat zij de programmeur in staat stellen programma's te construeren, waarvan de loop "extern" beïnvloed kan worden.

De rest van het verticale paneel is bestemd voor display en wel vindt men hier allereerst twee rijen van 28 resp. 27 lampjes, waarop men met behulp van twee draaischakelaars de inhoud van een aantal registers zichtbaar kan maken. De registers zijn

OR, U en M (bovenste rij lampjes)

A, S, B en T (onderste rij lampjes)

Van deze registers behoeven U en M nadere uitleg.

Het register U is het centrale register van de X-1, dat tot nog toe nergens besproken hoefde te worden, omdat het voor de programmeur onbereikbaar is. Algemeen kan men zeggen, dat het arithmetisch resultaat van elke opdracht na afloop van deze opdracht in het U-register staat. Wanneer nu (U) op de lampjes zichtbaar gemaakt wordt, kan men dus, als een programma stap voor stap doorlopen wordt, de resultaten van de bewerkingen volgen, onafhankelijk van het feit of deze in A, S, B of een geheugenplaats worden geborgen. Ook zuivere transporten gaan via het U-register.

Wat een opdracht in U achterlaat, kan men afleiden, als men bedenkt dat conditiesettende opdrachten na afloop van de bewerking (U) analyseren.

Het M-register dient als in- en uitgang van het geheugen, en is de enige plaats waarop men het pariteitsbit zien kan.

Er zij hier gewezen op de mogelijkheid een getal uit het geheugen zichtbaar te maken, zonder de inhoud van een van de werkregisters aan te raken. Men zet daartoe de U 2A-opdracht in de woordschakelaars en drukt op de knop DOE. Het uitgelezen woord uit het geheugen verschijnt daarbij in M en U.

Tenslotte vindt men op dit paneel nog enkele afzonderlijke lampjes, waarop de inhoud van het conditieregister, laatste tekenregister en overflowindicatie zichtbaar zijn. Als de machine gestopt is wegens het falen van de pariteitscontrole, geven lichtjes aan of dit een "fout getal" dan wel een "foute opdracht" betrof.

Zoals de lezer zal begrijpen, worden de indicatielampjes niet gebruikt, zolang de X-1 en het programma feilloos zijn. Bij werkende X-1 is er aan de lampjes ook heel weinig te zien: de inhoud van de registers wisselt veel te snel. Alleen aan (het meer significante gedeelte van) de opdrachtteller zal het geoefend oog

misschien iets kunnen zien over het verloop van het rekenproces. Pas als de X-1 gestopt is, komen de indicatielampjes tot hun recht: bij de localisering van fouten en vergissingen zijn zij van onschatbare waarde.