LÆREBOG
I
KODNING FOR GIER

3. udgave

af

Chr. Andersen og Chr. Gram

REGNECENTRALEN, KØBENHAVN

September 1962

INDHOLDSFORTEGNELSE.

FORORD.

I august 1961 udsendtes den første lærebog i kodning for GIER i duplikeret form; den blev revideret i løbet af vinteren, således at anden udgave (stadig duplikeret), dateret februar 1962, kunne udsendes i løbet af foråret.

Af praktiske grunde har vi valgt at udsende tredie udgave i to (eller flere) hæfter, og nærværende første hæfte er næsten identisk med de første 7 kapitler i 2.udgave. Vi har dog rettet alle de fejl, vi har fundet i 2.udgave, og desuden er eksempelsamlingen i kapitel 7 omredigeret og udvidet.

Som det fremgår af indholdsfortegnelsen indeholder dette første hæfte en almindelig oversigt over cifferregnemaskinernes egenskaber og virkemåde, en beskrivelse af GIERs struktur, den enkelte ordres opbygning, en fuldstændig operationsliste samt et kapitel med eksempler og øvelser i kodning.

Alt hvad der angår indlæsning, udlæsning, biblioteksekvenser og hjælpeprogrammer er derimod blevet henvist til hæfte 2, som vi håber at kunne udsende i november, og mens notationen i hæfte 1 følger de fælles konventioner for indlæseprogrammerne OLGA-DIG og SLIP, vil hæfte 2 blive mere SLIP-orienteret.

Vi skylder tak til mange for den hjælp vi har modtaget under udarbejdelsen, men først og fremmest vil vi gerne takke Bj.Svejgaard, thi dennes altid beredvillige medvirken har været en nødvendig forudsætning for denne bog. Dernæst vil vi gerne takke H.Isaksson, L.Hansson, T.Krarup og P.Mondrup, som ligeså beredvilligt har deltaget i diskussioner og som har foreslået mange forbedringer.

Chr.Andersen Christian Gram

1. CIFFERREGNEMASKINER.

1.1 Indledning.

Inden vi går igang med at gennemgå elektronregnemaskinen GIER, har vi fundet det rimeligt at give en oversigt over visse fælles træk ved de elektroniske cifferregnemaskiner [1]. Yderligere vil vi omtale forskellige emner, der har tilknytning til disse maskiner.

Oversigten, der gives i dette kapitel, er på ingen måde fuldstændig - bl.a. kommer vi ikke ind på spørgsmål af teknisk karakter - men den omtale, de enkelte ting har fået, er forhåbentlig tilstrækkelig til at sikre, at læsere uden forkundskab til cifferregnemaskiner alligevel kan tilegne sig indholdet af resten af bogen.

I kapitel 2 gennemgås GIER-maskinens opbygning og funktion, og kapitel 3 og de følgende kapitler er helliget alle de oplysninger, som brugeren (koderen) må have til sin rådighed, for at kunne skrive programmer til GIER.

---

1) Den store familie af analogregnemaskiner vil slet ikke blive omtalt i denne bog.

1.2 Struktur.

Langt de fleste af de i dag eksisterende regnemaskiner har den samme grundlæggende, logiske struktur. Enhver maskine har en aritmetisk enhed, en lagerenhed, en centralenhed og visse ydre enheder.

Den aritmetiske enhed har visse træk fælles med den moderne bord - regnemaskine, idet den kan udføre de 4 regningsarter, samt har plads til nogle få tal; disse lagerpladser i den aritmetiske enhed kaldes registre. På en bord - regnemaskine er det tilstrækkeligt med nogle få registre i maskinen, idet man nedskriver mellemresultater og lignende på papir, for ved senere brug atter at sætte dem ind i maskinen. En sådan fremgangsmåde kræver, at maskinen må stå og vente adskillige sekunder for hver udregning, og da elektronregnemaskiner er kostbare, er fremgangsmåden komplet ubrugelig her.

Derfor er elektronregnemaskinen forsynet med et lager, hvor den selv kan gemme mellemresultater (og lageret træder så i stedet for papiret). Den tid, det tager maskinen "at skrive et tal" i lageret eller "at læse et tal" fra dette, er af størrelsesordenen 10^{-4} sekund, og udregningerne i den aritmetiske enhed foregår med samme hastighed. Det er denne store hastighed, der gør det økonomisk at lade større beregninger udføre på elektronregnemaskiner.

Lageret er inddelt i felter eller skuffer, som kaldes celler. En celle består af en række fysiske elementer, der hver har et antal "stabile tilstande". I de fleste maskiner har hvert element kun to tilstande, og elementerne kaldes da binære (og den pågældende maskine siges at arbejde i det binære talsystem eller 2-tal-systemet). Hyppigt består hver celle af 40 binære elementer, og der bliver i så fald 2^{40} forskellige muligheder for "indholdet" i en celle. Et sådant celleindhold vil vi kalde en cifferkombination – svarende til, at indholdet af et enkelt element kaldes et ciffer eller en bit. De to mulige cifre betegnes med 0 og 1 (se afsnit 1.5). Maskinen vil hyppigt opfatte en cifferkombination som et (reelt) tal, og hver celle kan således rumme ét tal. Hver celle har et nummer, en adresse, og maskinen er bygget, så den bl.a. kan udføre operationer af typen "lagr det tal i celle nr. dit" samt af typen "læs tallet i celle nr. dat og send det til aritmetisk enhed".

For at maskinen skal kunne løse større opgaver, må den have et stort lager med plads til mange tal; men et stort lager, der skal være "lige så hurtigt" som den aritmetiske enhed, er meget dyrt. Mange maskiner er derfor, af økonomiske grunde, forsynet med et mindre, meget hurtigt arbejdende lager, samt et meget stort lager, som det er væsentlig langsommere både at skrive i og at læse fra. Man må så søge at dirigere beregningen sådan, at de tal, der skal bruges meget, fortrinsvis anbringes i det hurtige lager, mens tal, der kun bruges af og til under beregningen, forvises til dét langsomme lager.

Den centrale enhed styrer beregningens gang i overensstemmelse med givne instrukser, og den centrale enhed skal have detailleret besked på, hvad der skal udføres, samt i hvilken orden tingene skal ske. Maskinen kan ikke opfinde nye operationer eller metoder; den kan nok i visse situationer vælge mellem forskellige muligheder, men samtlige muligheder må på forhånd være nøje gennemtænkt og planlagt af opgavestilleren.

De ydre enheder er regnemaskinens kontakt med omverdenen, og det er igennem disse enheder, maskinen modtager oplysninger og afleverer resultater. Dette udstyrs ydeevne varierer meget fra maskine til maskine, og kun lidt kan siges generelt. Dog kan kun yderst få af de i dag eksisterende maskiner læse hånd- eller maskinskrevne tal eller bogstaver. Det almindelige princip er, at alle oplysninger til maskinen skal overføres på hulkort eller på papirkodebånd (dette sker ved hjælp af særlige skrivemaskiner), hvorefter maskinens læseenhed aflæser kortene henholdsvis kodebåndet.

Et hulkort er et rektangulært stykke pap, der er inddelt i et antal kolonner (som regel 80), og hver kolonne har 10 felter, der er nummereret 0,1,...,9, samt to ekstra felter, der benyttes til kontrolhulning. Et hul i et felt repræsenterer det tilsvarende ciffer; f.eks. kan tallet 15 skrives med et hul i felt 1 i 1.søjle og et hul i felt 5 i 2.søjle. Desuden kan kombination af 2 huller i samme søjle repræsentere bogstaver.

Principperne for brug af papirkodebånd ("hulstrimmel") er noget lignende: I rækker på tværs af båndets længde anbringes visse hulkombinationer, og hver hulkombination repræsenterer så et bestemt tegn (ciffer eller bogstav eller eventuelt typografisk tegn).

Selve indlæsningen foregår da, ved at hulkort eller kodebånd føres gennem læseenheden, som kan afføle disse kombinationer, og svarende til, hvad der læses, sende bestemte elektriske impulser videre til selve regnemaskinen.

Udlæsning kan ske på en elektrisk skrivemaskine, som er i direkte forbindelse med regnemaskinen. En almindelig skrivemaskine er imidlertid meget langsomt arbejdende i forhold til elektronregnemaskinen, og meget hurtige skrivemaskiner (som skriver en hel linie ad gangen) er meget dyre. Langt de fleste regnemaskiner er derfor forsynet med udlæseenheder, der skriver på hulkort eller papirkodebånd; thi sådant apparatur er rimeligt i pris og ca. 10 gange hurtigere end en normal skrivemaskine. Endnu en fordel, som ikke er uvæsentlig, er at ved denne fremgangsmåde kan resultater, der er udlæst på hulkort eller kodebånd, umiddelbart indlæses i maskinen igen, hvis de skal bruges i andre beregninger.

Sker udlæsningen til hulkort eller hulstrimmel, må man bagefter benytte hjælpemidler (kodebåndsstyrede skrivemaskiner og lign.) til at oversætte resultaterne til "mennesketal", men disse hjælpemidler er væsentlig billigere i brug end selve regnemaskinen.

1.3 Anvendelse. Et simpelt eksempel på kodning.

En betingelse for at udnytte elektronregnemaskinens store operationshastighed er, at den i en vis forstand skal være selvstyrende. Det skal forstås sådan, at en maskine på forhånd skal have besked om alt, hvad der skal ske i løbet af en beregning. Maskinen skal have en række ordrer, og den udfører så disse i den orden, hvori de er opskrevet. Da den kun kan udføre yderst elementære operationer som addition, subtraktion, multiplikation eller division af to tal, transport-operationer (som ovenfor nævnt) samt visse mere administrative operationer, må enhver opgave, som skal løses af en elektronregnemaskine, opløses helt til bunds og stykkes sammen af sådanne elementaroperationer, der skal udføres i en bestemt rækkefølge. Ved kodning eller programmering forstås netop denne omskrivning af en given opgave til en kæde af elementaroperationer, som maskinen kan udføre; maskinerne kaldes programstyrede, fordi en beregning styres af maskinen selv i overensstemmelse med det på forhånd kodede program.

F.eks. kan en regnemaskine ikke uden videre udføre følgende opgave:

(1) Løs ligningen $22x + 32 = 479$,

fordi den skal have nøjagtig besked om, hvordan sagen skal gribes an. Det hjælper betydeligt, hvis vi formulerer opgaven på følgende måde:

(2) Beregn $x = \frac{479 - 32}{22}$.

Parat til maskinbehandling er den først, når vi tænker os tallene 22, 32 og 479 anbragt i lageret, f.eks. i celle nr. 101, 102 og 103, og derefter giver maskinen følgende kæde af ordrer (følgende program):

"læs tallet i celle 103; send det til aritmetisk enhed"
"læs tallet i celle 102; træk det fra det, der står i aritmetisk enhed"
(3) "læs tallet i celle 101; dividér det op i det, der står i aritmetisk enhed"
"skriv på skrivemaskine det tal, der står i aritmetisk enhed" [1]
"stop"

---

1) Mens de øvrige ordrer svarer nøje til de virkelige forhold, er denne skriveordre en alvorlig forenkling af sandheden. Der skulle på dette sted i programmet i virkeligheden stå en hel række ordrer, som skulle dirigere antallet af cifre og decimaler etc.

Disse ordrer skal maskinen på forhånd have lagret i denne rækkefølge, og hertil bruges det samme lager som til talopbevaring. Ved mange maskiner er det sådan, at der i hver celle kan rummes enten et tal eller en ordre. Det er bekvemt at bruge det samme lager til begge dele, for det første fordi det gør maskinen mere flexibel i brug - ved nogle opgaver bruger man megen lagerplads til talmaterialet og kun lidt til ordrer og ved andre opgaver omvendt. For det andet betyder det, at maskinen kan operere ("regne") på sine egne ordrer, d.v.s. at maskinen selv kan ændre beregningsgangen under løsningen af en opgave. Dette kan dog kun ske i overensstemmelse med en af koderen fastlagt plan.

I hver celle i lageret kan der, som vi skal vise i det følgende, kun opbevares cifferkombinationer, og hvis en celles indhold sendes til den aritmetiske enhed, vil det blive opfattet og behandlet som et tal; men hvis den samme celles indhold sendes til kontrolenheden, vil det blive opfattet som en ordre, og denne vil derefter blive udført. Maskinen kan altså ikke selv kende forskel på, om en celle indeholder et tal eller en ordre, og derfor må koderen vælge og derefter holde nøje regnskab med, hvilke dele af lageret der bruges til tal, og hvilke der bruges til ordrer.

Arbejdsgangen for at få en maskine til at løse den ovenfor nævnte opgave er nu følgende:

1) Først "hulles" på en strimmel eller på hulkort de 5 ordrer og de 3 tal, der skal bruges. Man kunne her tænke sig at hulle præcis de cifferkombinationer, som ordrerne i (3) skal lagres som; men da mange elektronregnemaskiner benytter 2-tal-systemet, ville det betyde, at hver ordre og hvert sædvanligt decimalt tal først skulle oversættes eller omregnes til en stribe af nuller og ettaller

(og rent bortset fra besværet hermed ville det være næsten umuligt at gøre det rigtigt og at læse korrektur på det). Man har derfor ved næsten alle maskiner et særligt indlæseprogram (der allerførst anbringes i maskinen), som læser hulstrimler eller hulkort, der er hullet med mere behagelige konventioner (f.eks. skrives tal decimalt), og dette indlæseprogram oversætter så oplysningerne på hulstrimlen (eller hulkortene) til maskinens interne sprog. Hvad der skal hulles i stedet for f.eks. ordrerne i (3), kan man finde i operationslisten for maskinen; operationslisten beskriver for det første hvilke operationer maskinen overhovedet kan udføre, og for det andet fortæller den, hvad der skal hulles for hver enkelt ordre.

Man ville i dette tilfælde måske finde, at ordrerne (3) skulle hulles som

```
        FLYT  103
        SUB   102
(4)     DIV   101
        SKRIV
        STOP
```

idet man tager vidtgående mnemotekniske hensyn ved valget af det "ydre" sprog, og derefter blot opbygger et indlæseprogram i overensstemmelse hermed.

Foruden ordrerne skal tallene 22, 32 og 479 hulles (som decimale tal) og indlæses ved hjælp af indlæseprogrammet, som omregner tallene til maskinens talsystem (hvis dette ikke er 10-tal-systemet).

2) Efter at ordrerne er indlæst f.eks. til celle nr.1,2,...,5 og tallene til celle nr. 101, 102 og 103, startes maskinen. Herved aflæser centralenheden først indholdet i celle 1, tolker det

som en ordre, og udfører denne (i dette tilfælde føres altså tallet i celle 103 til den aritmetiske enhed); derefter aflæser centralenheden celle 2, tolker indholdet som en ordre, og udfører denne, o.s.v. indtil centralenheden tolker indholdet i celle 5 som en stopordre og standser maskinen.

Det er et typisk træk ved langt de fleste regnemaskiner, at ordrerne automatisk udføres i den orden, hvori de er anbragt i lageret. Nu er man imidlertid ofte interesseret i at lade maskinen gentage en beregning mange gange (f.eks. med nye data hver gang), altså at lade maskinen vende tilbage til en af de ordrer, der allerede er udført. Til dette formål - og i det hele taget til at bryde den naturlige rækkefølge af ordrerne - anvendes såkaldte <u>hopordrer</u>. Disse ordrer er rent administrative, idet udførelsen af en hopordre ikke bevirker nogen beregning, men <u>kun</u> at den næste ordre hentes fra et nyt, nærmere specificeret sted i lageret.

<u>Eksempel 1.1.</u>

Til illustration af brugen af hopordrer vil vi tage ovenstående kode (4). I denne kode er ordren SKRIV en grov modifikation af sandheden, idet ingen maskine kan nøjes med én ordre (ved en talskrivning), fordi man på en eller anden måde skal angive hvormange decimaler, der skal medtages, hvordan fortegnet skal skrives, skrivning af decimalkommaet på rette plads, m.v. (samt, hvis maskinen internt regner i 2-tal-systemet, foretage en omregning til decimal repræsentation). Da imidlertid en udskrivning af resultater er en funktion, der går igen i næsten alle opgaver, koder man én gang for alle et program, der besørger dette, og lagrer programmet et bestemt sted i maskinens lager, f.eks. fra celle 1000 og fremefter. Ønsker man nu som i kode (4) at skrive et resultat ud, anbringer man blot på SKRIV-ordrens plads en hopordre, som f.eks. kan hedde

HOP 1000.

Denne ordre bevirker blot, at efter ordren DIV 101 er udført, fortsætter maskinen med ordrerne i celle 1000, 1001, ..., indtil resultatet er skrevet færdigt på skrivemaskinen. Dette faste skriveprogram vil man ofte lade slutte med endnu en hopordre, der får maskinen til at "vende tilbage", hvor den kom fra, og derfor i vort eksempel fortsætte med ordren STOP i celle 5; d.v.s. at maskinen stopper.

1.4 Anvendelse. Biblioteksekvenser.

Det i eksempel 1.1 omtalte skriveprogram er et eksempel på en biblioteksekvens, d.v.s. et program eller et stykke kode, der ligger klar (og er gennemprøvet!), og som kan udføre en eller anden hyppigt tilbagevendende opgave.

De vigtigste biblioteksekvenser ved enhver maskine er et indlæse-program, der som omtalt i afsnit 1.3 kan oversætte fra vort ydre ordresprog og fra vort decimale sprog til maskinens interne repræsentation, samt et skrive-program, et udlæse-program til skrivning af resultater. Disse sekvenser bruges så hyppigt, at man ofte vil vælge at have dem stående fast et bestemt sted i lageret.

Som andre eksempler på biblioteksekvenser kan nævnes en sekvens der kan finde kvadratroden af et opgivet tal, en sekvens der kan finde cosinus til en given vinkel, en sekvens der kan sortere en talmængde i lageret efter størrelse o.s.v.

Disse sekvenser kan slet ikke lagres fast i maskinen alle sammen på grund af den begrænsede lagerkapacitet; men det væsentlige er, at der findes en god beskrivelse af bibliotekssekvensens nøjagtige funktion m.v., samt at selve koden findes let tilgængelig. Den, der f.eks. skal bruge kvadratrod-sekvensen, behøver da blot at indkopiere den én gang kodede sekvens på et passende sted i sit eget program.

Bemærkning: Et af kendetegnene på et godt regnecenter er, at der findes mange velbeskrevne og lettilgængelige bibliotekssekvenser, da det letter kodearbejdet meget, at mange "standard-funktioner" kan udføres ved hjælp af bibliotekssekvenser.

1.5 2-tal-systemet.

Da mange maskiner internt benytter 2-tal-systemet, fordi det er det nemmeste i elektronisk henseende (idet mange elektroniske komponenter er bistabile, det vil sige har 2 stabile tilstande), vil vi i dette afsnit trænge lidt ind i 2-tal-systemets mærkværdigheder og studere talrepræsentation i dette system.

1.5.1 Fast kommaplacering.

Ethvert positivt reelt tal kan skrives som en (endelig eller uendelig) binær-brøk, d.v.s. som en sum af positive og negative potenser af 2, hvor hver potens har koefficienten 0 eller 1; idet positionssystemet bruges på tilsvarende [illegible]brøker, skrives binærbrøker, så de ligner decimalbr[illegible]rene 0 og 1 forekommer.

Eksempel 1.2.

I de følgende eksempler på reelle tals [illegible]præsentation) i decimalsystemet og i 2-tal-systemet [illegible] decimale repræsentation til venstre og den binære i [illegible]

0	0
1	1
2	10
3	11
-4	-100
-7	-111
8	1000
11.5	1011.1
1.75	1.11
0.4	$0.01\overline{1001}10\ldots$

Man kan danne ganske enkle regler for omregning begge veje mellem de to repræsentationer.

I eksempel 1.2 repræsenteres negative tal ved deres absolutte værdier forsynet med et minustegn; man kunne bruge det samme system i en elektronregnemaskine, idet man blot reserverede en speciel bit (indholdet, informationsmængden, i en bistabil komponent) til at vise fortegnet. Imidlertid bliver visse dele af 2-tals-aritmetikken simplere, når man lader et negativt tal repræsenteres ved et komplement, nemlig ved det positive tal, der fås ved til det givne negative tal at addere en fast valgt potens af 2 (stor nok til at man får positive repræsentanter for alle de betragtede negative tal).

I mange maskiner, deriblandt også GIER, er det talområde, maskinen er bygget til at behandle, intervallet fra −1 til +1. Alle tal i maskinen repræsenteres med et fast antal binære cifre − for eksempel 40 − og maskinens indbyggede aritmetiske operationer behandler disse tal sådan, at det første ciffer i tallet behandles som et ciffer lige foran kommaet, og alle de øvrige cifre behandles som cifre efter kommaet i en binærbrøks-udvikling. Et tal med første ciffer lig 1 opfattes af maskinen som det negative tal, der fås ved at subtrahere 2 fra værdien af binærbrøken.

Eksempel 1.3.

Lad en maskine have plads til 40 cifre, og lad maskinens talrepræsentation være den ovennævnte.

Står der i en celle	behandles det af maskinen som tallet (binært)		(decimalt)
0111......11	0.11.....1	=	$1-2^{-39} \doteq 1$
0110......0C	0.110....0	=	0.75
0100......00	0.100....0	=	0.5
00010.....00	0.0010...0	=	0.125
0000......01	0.000...01	=	$2^{-39} \doteq 1.82 \cdot 10^{-12}$
000.......00	0.00.....0	=	0
111.......11	1.11.....1 −10	=	$-2^{-39} \doteq -1.82 \cdot 10^{-12}$
11110.....00	1.1110...0 −10	=	−0.125
1100......00	1.100....0 −10	=	−0.5
10100.....00	1.010....0 −10	=	−0.75
100.......01	1.000...01 −10	=	$-1 + 2^{-39}$
1000......00	1.000....0 −10	=	−1

Maskinens talområde bliver således alle tal med 39 binære cifre efter kommaet i intervallet $-1 \leq x < 1$.

Årsagen til, at man har valgt netop denne kommaplacering og denne repræsentation af negative tal er, at dele af aritmetikken bliver simplere ved disse valg end ellers.

F.eks. vil det være den samme proces, der skal udføres ved en addition, uanset om de tal, der indgår i denne, er negative eller positive.

Multiplikationen bliver ligeledes særlig simpel, idet produktet af to maskintal (d.v.s. tal i intervallet $-1 \leq x < 1$) igen bliver et maskintal – undtagen hvis begge faktorer er -1.

Uanset hvilken talrepræsentation man vælger, vil man ved addition og subtraktion (og iøvrigt også ved division) risikere at sprænge maskinens kapacitet, d.v.s. få et resultat, der ligger uden for det tilladte interval.

Eksempel 1.4.

Lader vi den i eksempel 1.3 omtalte maskine addere tallene 0.75 og 0.75, udfører den følgende regning:

```
  01100.....0
+ 01100.....0
  -----------
  11000.....0
```

og opfatter resultatet som tallet -0.5.

Skal den addere tallene -0.75 og -0.5, får den

```
  10100.....0
+ 11000.....0
  -----------
  01100.....0 ,
```

altså resultatet $0.75 = -1.25 + 2$. Den automatiske menteoverføring, hvor en mente på første plads forsvinder ud i den tomme luft, bevirker således, at maskinen regner modulo 2, d.v.s. at resultater uden for intervallet $-1 \leqq x < 1$ bringes ind i dette interval ved addition eller subtraktion af 2.

Af eksempel 1.4 ses, hvorledes man risikerer, at en maskine afleverer et galt resultat, hvis man sprænger dens kapacitet. Man kan naturligvis (og bør i mange tilfælde) planlægge sine beregninger sådan, at det ikke sker, men mange maskiner er dog udstyret med en facilitet, der gør det let at redde situationen i tilfælde som de i eksemplet nævnte:

I lighed med bordregnemaskiner har de elektroniske regnemaskiner større cifferkapacitet i selve regneregistrene end i resten af maskinen, idet der i disse registre er plads til to cifre foran kommaet (og samme antal cifre efter kommaet som ellers). Er det forreste ciffer her lig 0, behandles registrets indhold som et positivt tal med binærbrøkens værdi; men hvis det forreste ciffer er lig 1, behandles indholdet som det negative tal, der fås ved at subtrahere 4 fra værdien af binærbrøken.

Eksempel 1.5.

Lad en maskine have et regneregister med 41 positioner, hvor talrepræsentationen er den ovennævnte.

Står der i registeret	behandles det af maskinen som tallet (binært)			(decimalt)
01110.......0	1.11		=	1.75
01100.......0	1.1		=	1.5
01000.......0	1		=	1
00110.......0	0.11		=	0.75
000010......0	0.001		=	0.125
111110......0	11.111	−100	=	−0.125
11010.......0	11.01	−100	=	−0.75
11000.......0	11	−100	=	−1
10100.......0	10.1	−100	=	−1.5
10010.......0	10.01	−100	=	−1.75
10000.......0	10	−100	=	−2

Talområdet bliver således intervallet $-2 \leqq x < 2$.

Skal vi nu udføre de samme additioner som i eksempel 1.4, blot med 2 cifre foran kommaet, fås

(binært)	(decimalt)	(binært)	(decimalt)
00.1100.....0	= 0.75	11.010.....0	= −0.75
+ 00.1100.....0	= 0.75	+ 11.100.....0	= −0.5
01.1000.....0	= 1.5	10.110.....0	= −1.25

altså de rigtige resultater.

Det fremgår af eksempel 1.5 (og kan iøvrigt let bevises), at maskintal, altså tal i intervallet $-1 \leqq x < 1$, netop er de tal, der har cifrene 00 eller cifrene 11 foran kommaet i regneregisteret; står der 01 foran kommaet, ligger tallet i intervallet $1 \leqq x < 2$, og står der endelig 10 foran kommaet, ligger tallet i intervallet $-2 \leqq x < -1$.

Skal maskinen addere to tal, der hentes i lageret (med 1 ciffer foran kommaet), skal således de to tals fortegnscifre hver for sig dubleres, og derpå udføres additionen med 2 cifre foran kommaet.

Det er nu særdeles nemt af resultatet at se, om dette er et maskintal eller ej: Hvis de to cifre foran kommaet er ens, er resultatet atter et maskintal, og er de forskellige, ligger resultatet uden for intervallet $-1 \leqq x < 1$ (i dette tilfælde siger vi, at der er opstået <u>overløb</u> eller spild).

Hvis resultatet er et maskintal, kan det eventuelt gemmes i en lagercelle til senere brug, og det fremgår af ovenstående udvikling, at det er nok at gemme 1 ciffer foran kommaet samt alle cifrene efter kommaet; det ekstra ciffer foran kommaet i regne-registeret "glemmes" derefter.

<u>Eksempel 1.6.</u>

Vi bemærker, at tilføjer vi flere cifre foran kommaet – således at tallene noteres med f.eks. h cifre foran kommaet kan talområdet udvides tilsvarende: Med den vedtægt, at tal, hvis forreste ciffer er lig 0, opfattes som positive, mens tal med forreste ciffer lig 1 opfattes som negative, nemlig som binærbrøkens værdi minus 2^h, bliver talområdet intervallet $-2^{h-1} \leqq x < 2^{h-1}$. Tal i det grundlæggende interval $-1 \leqq x < 1$ er også her karakteriseret ved at cifrene foran kommaet er ens, lutter nuller for positive tal og lutter ettaller for negative tal.

<u>1.5.2 Flydende kommaplacering.</u>

Ved mange af de opgaver, der ønskes løst af elektronregnemaskiner, kan man ved passende planlægning klare sig med et talområde, som det i forrige afsnit beskrevne, f.eks. ved at anvende passende skalafaktorer på den datamængde, der indgår i beregningen.

Det afgørende for, om en sådan metode kan anvendes, er at den største og den mindste absolutværdi ikke afviger mere fra hinanden end maskinens cifferkapacitet kan bære. Hvis en maskine f.eks. kan lagre tal med 40 binære cifre, må forholdet mellem den største absolutværdi og den mindste relevante absolutværdi (forskellig fra 0) ikke overstige ca. 2^{39}, thi ellers vil enten de største tal overskride talområdet, eller også vil de mindste tal blive behandlet som nuller.

Ønsker man at løse en opgave, hvor dette krav til talmaterialet ikke kan opfyldes, vil det være bekvemt at lade maskinen opfatte hvert eneste tal som bestående af en taldel samt en exponentdel. Lad atter lagercellerne i en maskine bestå af 40 bits. Da kan maskinen f.eks. være bygget sådan, at den (ved visse bestemte ordrer) "læser" de første 30 bits som en taldel og de sidste 10 bits som en 2-tals-exponent, således at indholdet i en lagercelle behandles som et tal, hvis værdi er talværdien af de første 30 bits multipliceret med 2 opløftet til den exponent, der findes i de sidste 10 positioner.

En meget anvendt talrepræsentation for taldel og exponentdel er følgende:

a) De 30 bits, der er reserveret til taldelen, opfattes som en positiv eller negativ binærbrøk på nøjagtig samme måde, som beskrevet i afsnit 1.5 med komma mellem første og andet ciffer.

b) De 10 bits, der er reserveret til exponenten, opfattes som et heltal (skrevet i 2-tal-systemet); hvis det forreste ciffer er lig 0, opfattes exponenten som dette positive heltal; hvis det forreste ciffer er lig 1, opfattes exponenten som det negative heltal, der fås ved fra den positive værdi at subtrahere $2^{10} = 1024$.

Lad taldelen være z og lad exponenten være v. Indholdet i cellen behandles da som tallet

$$y = z \cdot 2^v,$$

og vi siger, at tallet er skrevet på flydende form eller som flydende tal.

Det talområde, der på denne måde står til rådighed, bliver, idet vi kun ser på absolutværdien,

$$y = 0$$
$$2^{-29} \cdot 2^{-512} \leqq y < 1 \cdot 2^{511}.$$

hvilket er en enorm forøgelse i forhold til det i afsnit 1.5.1 nævnte. Til gengæld er nøjagtigheden blevet mindre i den forstand, at hvert tal kun "huskes" med 30 cifre mod normalt 40 cifre, men den pris betaler man ofte gerne for at få talområdet udvidet.

Hvis man forlanger et større talområde, kan dette opnås på bekostning af nøjagtigheden, idet man kan reservere 11 cifre til eksponenten og nøjes med 29 til taldelen. Talområdet vil da i det væsentlige være

$$2^{-1024} < y < 2^{1023}$$

og hvis der helt generelt er afsat x cifre til exponenten, vil talområdet (idet vi ser på absolutværdien) i det væsentlige være

$$2^{-2^{x-1}} < y < 2^{2^{x-1}}.$$

For at udnytte de 30 bits i taldelen så godt som muligt, kan man forlange, at taldelen altid skal være normaliseret, d.v.s. at exponenten skal vælges sådan, at taldelens absolutværdi ligger mellem 0.5 og 1. Dette kan gøres for alle tal undtagen nul, idet man ved at udskille passende 2-potenser kan presse alle taldele (eller rettere absolutværdier) indenfor en bestemt binade, d.v.s. et bestemt interval af formen

$$2^a \leqq z \leqq 2^{a+1}.$$

Herved opnås mod en forholdsvis ringe indskrænkning af talområdet, at (løst sagt) alle de 30 cifre i taldelen indeholder betydende cifre. Det kan i den forbindelse nævnes, at de fleste af de maskiner, der kan behandle flydende tal, automatisk normaliserer resultatet af en aritmetisk operation (og indretter exponenten derefter), inden det afleveres.

2. GIERs OPBYGNING

2.1 Indledning.

Af hensyn til senere referencer er der i dette afsnit samlet en del oplysninger om GIER-maskinens opbygning og aritmetik. Det følgende bliver tung læsning og kan delvis overspringes i første omgang, blot man senere vender tilbage hertil, når man får brug for disse oplysninger.

2.2 Ferritlageret.

GIERs arbejdslager er et ferritlager med 1024 celler, og hver celle består af 42 bits, som vi nummererer fra 0 til 41 inclusive; de første 40, bits nr. 0 – 39, kan bruges til tallagring på to forskellige måder:

1) Ved regning med fast komma fungerer maskinens indbyggede aritmetiske operationer, som om kommaet er placeret mellem position nr. 0 og position nr. 1; tal med 0 i position 0 opfattes som tal i intervallet $0 \leqq x < 1$, mens tal med 1 i position 0 opfattes som tal i intervallet $-1 \leqq x < 0$, nemlig som værdien af binærbrøken (et tal

mellem 1 og 2) minus 2. Denne talrepræsentation kan karakteriseres ved, at maskinen regner modulo 2 i intervallet $-1 \leqq x < 1$ med 39 binære cifre efter kommaet.

2) Ved regning med <u>flydende komma</u> tænkes ethvert tal y skrevet på formen

$$y = z \cdot 2^{v}$$

hvor enten $1 \leqq z < 2$

eller $z = 0$

eller $-2 \leqq z < -1$ 1)

og hvor z er en binær brøk, mens v er et heltal. De indbyggede operationer fungerer således, at de 10 første bits, pos. 0 – 9, opfattes som exponenten v og resten som taldelen z. Hvis bit nr.0 er 0, opfattes exponenten som et positivt heltal, det vil sige $0 \leqq v \leqq 511$, og hvis der står 1 i pos. 0 som et negativt heltal i intervallet $-512 \leqq v < 0$, idet der regnes modulo 1024 i intervallet $-512 \leqq v \leqq 511$. I taldelen placeres kommaet mellem pos. 11 og 12, mens pos. 10 er fortegnsvisende. Position 10 og 11 bliver som følge af z's definition næsten altid forskellige: Er taldelen z positiv, står der 01 i pos. 10 – 11, og er z negativ, står der 10; kun taldelen 0 har 0 i begge positioner.

Taldelen står i pos. 10 – 39 og repræsenteres altså med 28 binære cifre efter kommaet.

---

1) Det er altid muligt at vælge z således, og derefter er exponenten v entydigt bestemt på nær i tilfældet $z = 0$.

Det talområde, der kan behandles med flydende regning i GIER, bliver således

$$-2\cdot 2^{511} \leqq y \leqq -(1 + 2^{-28})\cdot 2^{-512}$$

$$y = 0$$

$$1\cdot 2^{-512} \leqq y \leqq (2 - 2^{-28})\cdot 2^{511}$$

eller i sædvanlig decimalnotering

$$\text{ca. } 7.458\cdot 10^{-155} < \text{abs}(y) < \text{ca. } 1.341\cdot 10^{154} \text{ . } ^{1)}$$

De to ekstra bits i hver celle bruges til en eventuel <u>mærkning</u> af tal i lageret. De fleste tal vil (sikkert) ingen mærkning have, og det vil sige, at pos. 40 og 41 er nulstillede, men ethvert tal kan enten

a-mærkes ved at sætte 10 i pos. 40-41,

b-mærkes ved at sætte 01 i pos. 40-41 eller

c-mærkes ved at sætte 11 i pos. 40-41.

---

1) Bemærk: Ved flydende regning i GIER kan tallet 0 behandles på to måder: Enten altid med taldel og exponentdel lig 0 eller med taldelen 0 og den exponent, der er fremkommet under regningerne; i sidstnævnte tilfælde vil f.eks. additionen

$$0\cdot 2^{300} + 1.5\cdot 2^{225}$$

give resultatet $0\cdot 2^{300}$. Om de indbyggede flydende operationer bruger den ene eller den anden repræsentation, afhænger af en kontakt i GIER, således at brugeren kan vælge mellem disse to repræsentationer.

Eksempel 2.1

Indeholder en celle bitkombinationen

0	1	2	3	9	10	11	12	13	39	40	41
0	1	1	0	0	0	0	0	0	0	0	1

vil denne ved regning med fast komma blive "læst" som tallet 0.75 og ved flydende regning som nul, i begge tilfælde b-mærket.

Eksempel 2.2.

Indeholder en celle bitkombinationen

0	1	2	3	9	10	11	12	13	39	40	41
1	1	1	1	1	0	1	1	0	0	0	0

vil den ved regning med fast komma blive læst som tallet

$$-(2^{-10} + 2^{-12}) \doteq -0.00122$$

og ved flydende regning som

$$1.5 \cdot 2^{-1} = 0.75 ,$$

begge uden mærkning.

Eksempel 2.3.

Indeholder en celle bitkombinationen

0	1	2	3	9	10	11	12	13		39	40	41
0	0	0	0	0	0	0	0	0	0	1	1	1

vil den ved regning med fast komma blive læst som det c-mærkede tal

$$2^{-39} \doteq 1.82 \cdot 10^{-12}$$

mens den ved flydende regning er meningsløs, fordi taldelen ikke tilhører den "tilladte" talmængde.

I praksis vil man ofte mærke det første eller det sidste tal (eventuelt begge) i en talgruppe i lageret. Se eksemplerne i kapitel 7.

I hver celle kan i stedet for et tal lagres en helordsordre eller to halvordsordrer; her bruges den ene mærkebit til at skelne mellem disse to tilfælde, mens den anden mærkebit ved de aritmetiske grundoperationer angiver, om der skal regnes med fast eller flydende kommaplacering. Vedrørende ordrernes repræsentation i cellen henvises til afsnit 4.9.

2.3 Tromlelageret.

Udover ferritlageret har GIER et magnetisk tromlelåger med 320 kanaler à 40 celler, opbygget ligesom cellerne i ferritlageret. Overførsel af data til og fra tromlen sker kanalvis med 40 celler ad gangen, men da overførslen foregår uden om den aritmetiske enhed, kan der udføres beregninger simultant med overførslen. Det er dog en forudsætning ved transport fra tromle til ferritlager, at ingen af de 40 ferritceller, der læses til, berøres af eller indgår i den simultane beregning; tromletransporten foregår nemlig på følgende måde: Ved en forudgående ordre (en såkaldt VK-ordre) er nummeret på den aktuelle tromlekanal valgt (og anbragt i et særligt register, tk-registret). Ved en læseordre, en såkaldt LK-ordre, påbegyndes overførslen fra den valgte kanal til et afsnit på 40 celler i ferritlageret (angivet ved adressen i LK-ordren). Tromlen roterer med konstant hastighed, og læsningen fra tromlekanalen påbegyndes øjeblikkeligt, uanset hvilken kanalcelle der befinder sig ud for læsemekanismen. GIER beregner da automatisk, hvortil i ferritlagerafsnittet det læste skal transporteres og fortsætter således, indtil hele kanalen er læst [1]. Samtidig hermed fortsættes udførelsen af de ordrer, der følger efter LK-ordren. Overførsel den anden vej foregår på helt tilsvarende måde.

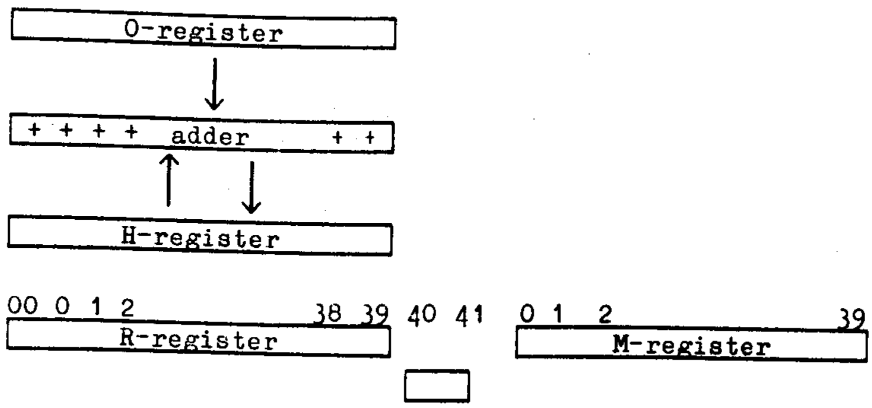
2.4 Aritmetisk enhed.

Det centrale i den aritmetiske enhed er adderen med de tilhørende, styrende 41-bits-registre, O-registeret og H-registeret. Her har H-registeret størst interesse, fordi resultatet af en aritmetisk operation (og i øvrigt også af en adresseberegning) lagres her, inden det føres videre i maskinen. H-registeret (og O-registeret) har til forskel fra lagercellerne ingen mærkebits, men har en ekstra foranstillet position nr. 00. Herved opnås, at regningerne udføres med to cifre foran kommaet (se afsnit 2.5).

H-registeret og flere andre i den aritmetiske enhed, er imidlertid ikke "tilgængelige" gennem programmeringen, hvor man benytter resultatregisteret R, der har 41 talbits ligesom H-registeret og derudover 2 mærkebits ligesom lagercellerne. Desuden anvender man multiplikatorregisteret M, der har 40 talbits ligesom lagercellerne men ingen mærkebits; brugen af disse registre forklares nærmere i næste afsnit.

---

1) I mange maskiner, deriblandt DASK, påbegyndes overførslen først, når celle nr. 1 i tromlekanalen er ud for læsehovedet. Man risikerer da, at tromlen skal dreje næsten 2 omgange, før læsningen er tilendebragt.



2.5 Aritmetikken.

2.5.1 Addition og subtraktion med fast komma.

Før addition eller subtraktion skal den ene operand anbringes i R-registeret; herved dubleres operandens fortegn, idet pos. 0 og 00 i R får samme indhold. Når den anden operand hentes fra lageret, dubleres også dennes fortegn, og selve operationen udføres med 2 cifre foran kommaet (og 39 cifre efter kommaet).

Dette sker i H-registeret, hvorefter resultatet flyttes til R-registeret. Da begge operanderne ligger i intervallet $-1 \leq x < 1$, vil resultatet ligge i intervallet $-2 \leq z < 2$, og derfor stå korrekt i R med 2 cifre foran kommaet, regnet modulo 4 i intervallet $-2 \leq z < 2$ (se afsnit 1 om 2-tal-systemet). Ligger resultatet i intervallet $-1 \leq z < 1$, vil bits nr. 00 og 0 være ens (11 for negative tal og 00 ellers), mens de er forskellige, hvis resultatet falder uden for dette interval (01 for resultater ≥ 1 og 10 for resultater < -1); i sidste tilfælde taler vi om overløb eller spild, fordi resultatet ikke uden videre kan gemmes i en lagercelle.

Information om overløb lagres i et specielt overløbsregister i maskinen (se afsnit 2.6). Overløbssituationen er dog under kontrol, idet GIER har en operation (et højre-skift), der flytter alle bits i R en position til højre (samt bevarer pos. 00 og foretager afrunding i pos. 39).

Udføres denne operation efter en addition eller en subtraktion, vil pos. 0-39 i R altid (uanset om der var overløb eller ej) indeholde den halve sum eller differens (på nær eventuel afrundingsfejl), og dette tal kan lagres og behandles på sædvanlig måde. Vi bemærker, at M-registeret ikke benyttes ved addition og subtraktion med fast komma.

2.5.2 Multiplikation med fast komma.

Det eksakte resultat af en multiplikation af to tal med 39 cifre efter kommaet er et tal med 78 cifre efter kommaet, og da man undertiden har brug for alle 78 cifre, men særdeles ofte dog kun er interesseret i de første 39, har GIER to indbyggede multiplikationsoperationer: kort multiplikation og lang multiplikation.

For begge operationers vedkommende gælder, at den ene faktor på forhånd anbringes i M-registeret. I selve multiplikationsoperationerne hentes den anden faktor fra en lagercelle. Vi behandler derefter de to operationer hver for sig.

Ved kort multiplikation udregnes produktet, hvorefter det afrundes til 39 binære cifre efter kommaet (idet der forhøjes, hvis produktets bit nr.40 er 1); derefter adderes indholdet af R-registeret til det afrundede produkt [1]. Operationen kaldes akkumulerende multiplikation.

---

1) Ønskes selve produktet, kan R-registeret nulstilles på forhånd.

Det fundne resultat anbringes i R-registeret, mens den første faktor stadig står i M-registeret. Hvis ikke begge faktorerne er -1, er produktet igen et tal i intervallet $-1 \leqq z < 1$, men ved den afsluttende addition kan der forekomme overløb på nøjagtig samme måde, som beskrevet i afsnit 2.5.1.

Er begge faktorer -1, giver multiplikationen det binære tal 01.00.....0, som derefter adderes til R-registerets indhold. Er dette sidste et negativt tal (mellem -1 og 0), giver additionen det korrekte svar uden overløb. Hvis tallet i R-registeret derimod ligger i intervallet $0 \leqq z < 1$, giver additionen det korrekte svar i R-registeret (med to positioner foran kommaet), men med overløb. Vi kan altså ikke lagre selve resultatet i en celle.

For kort multiplikation gælder da:

Såfremt der ikke er overløb i R-registeret før multiplikationen, vil R-registeret efter operationens udførelse indeholde det rigtige resultat – eventuelt med overløb – i hvilket tilfælde situationen kan bringes under kontrol med et højreskift, som ved almindelig addition eller subtraktion.

Ved lang multiplikation udregnes produktet med 2 cifre foran og 78 cifre efter kommaet; dertil adderes R-registerets indhold multipliceret med 2^{-39} (d.v.s. forskudt 39 pladser til højre). Resultatet placeres med de 2 cifre foran kommaet og de første 39 cifre efter kommaet i R-registeret og med de 39 sidste cifre i pos. 1-39 i M-registeret. Samtidig nulstilles pos. 0 i M. I dette tilfælde bliver resultatet altid korrekt, og såfremt der ikke er overløb i R før operationen, kan overløb kun forekomme, hvis begge faktorer er -1.

Eksempel 2.4.

Lad der i R-registeret stå tallet 0.5 og i M-registeret tallet $+ 1 - 2^{-39}$.

R	M
`0 │ 0 1 0 0      0 │ `	`0 1 1      1 1`

Multipliceres nu kort med tallet +0.5, bliver resultatet på grund af afrundingen +1, og R og M ser således ud:

R	M
`0 │ 1 0 0      0 │ `	`0 1 1      1 1`

Hvis vi i stedet benytter lang multiplikation med +0.5, bliver resultatet +0.5

R	M
`0 │ 0 1 0 0      0 │ `	`0 0 0      0 0`

idet selve multiplikationen giver $0.5 - 2^{-40}$, og derefter adderes $0.5 \cdot 2^{-39} = 2^{-40}$.

2.5.3 Division med fast komma.

Svarende til de to former for multiplikation har GIER en kort og en lang division: Ved lang division fungerer indholdet i det lange register bestående af R samt pos. 1–39 i M som dividend, og ved kort division fungerer indholdet i R (eller rettere indholdet i R efterfulgt af 39 nuller) som dividend. De to former for division foregår i øvrigt ens, således at dividenden på forhånd skal være anbragt i registeret (henholdsvis det lange R,M og det korte R), hvorefter divisor hentes fra lageret i selve divisionsoperationen. Forudsat at kvotienten q ligger i intervallet $-2 \leq q < 2$, udføres divisionen på følgende måde:

1) Positiv divisor d: Maskinen beregner den kvotient, der hører sammen med resten r i intervallet $0 \leqq r < d \cdot 2^{-39}$.

2) Negativ divisor d: Maskinen beregner den kvotient, der hører sammen med resten r i intervallet $d \cdot 2^{-39} \leqq r < 0$.

Derefter anbringes den fundne kvotient i R-registeret, og den tilhørende rest, multipliceret med 2^{39}, anbringes i M-registeret.

Hvis kvotienten ved division med fast komma ligger udenfor intervallet $-2 \leqq q < 2$, vil divisionsoperationerne nok blive udført, men de fremkomne tal i maskinen vil ikke være kvotient og rest. Det må således betragtes som en alvorlig fejl at benytte division med fast komma i de tilfælde, hvor man ikke er sikker på, at kvotienten ligger mellem -2 og 2.

Eksempel 2.5.

Hvis der efter en division ikke er overløb, ligger kvotienten q i intervallet $-1 \leqq q < 1$.

Er der overløb, og har man på forhånd sikret sig, at divisionen er "lovlig", ligger kvotienten i $-2 \leqq q < -1$ eller i $1 \leqq q < 2$; man kan altså, ved at foretage et højreskift i R-registeret, regne videre med den halve kvotient.

Eksempel 2.6.

Hvis divisionen (undtagelsesvis) går op, fås den rigtige kvotient kun, når divisor er positiv; thi hvis divisor er negativ, er den numerisk mindste rest med negativt fortegn divisor selv multipliceret med 2^{-39}, og kvotienten indrettes da efter denne rest.

Lad f.eks. dividenden $9 \cdot 2^{-78}$ stå i det lange register.

Divideres dette med $3 \cdot 2^{-39}$ ved lang division, vil efter divisionsoperationen kvotienten $3 \cdot 2^{-39}$ stå i R og resten 0 i M-registeret. Divideres derimod $-9 \cdot 2^{-78}$ med $-3 \cdot 2^{-39}$, vil (efter divisionsoperationen) kvotienten $2 \cdot 2^{-39}$ stå i R, og i M står tallet $-3 \cdot 2^{-39}$, svarende til, at divisionsresten er $-3 \cdot 2^{-78}$. Det kan derfor undertiden være farligt at benytte negativ divisor.

2.5.4 Flydende operationer.

Ved de indbyggede operationer med flydende kommaplacering fungerer R- og M-registrene som ét register, hvor taldelen z af et flydende tal altid anbringes i pos. 10-39 i R, og samtidig sættes bits nr. 00-9 identisk med bit nr. 10, således at taldelens fortegn kan aflæses i pos. 00. Exponenten v af et flydende tal anbringes i pos. 0-9 i M-registeret svarende til, at exponenterne står i pos. 0-9 i lagercellerne.

R-registeret, suppleret med pos. 0-9 i M-registeret, kaldes <u>det flydende resultatregister RF</u> og er det eneste aritmetiske register, der er tilgængeligt for koderen ved flydende regning. Resten af M-registeret benyttes internt under enhver flydende operation.

Ved alle fire regningsarter skal den ene operand på forhånd anbringes i RF-registeret. I selve operationen hentes den anden operand i lageret, og resultatet anbringes i RF-registeret.

Desuden bruges pos. 10-19 i M til, ved addition og subtraktion, at lagre oplysning om eventuelt ciffertab: Ved subtraktion af to flydende tal med samme exponent og omtrent samme taldel (eller ved addition af omtrent modsatte tal), får resultatet meget lille absolut værdi. Da taldel og exponent imidlertid indrettes sådan, at absolutværdien af taldelen ligger mellem 1 og 2, bliver taldelen suppleret med en række betydningsløse nuller (for at udfylde pos. 10–39); antallet af disse (eller anderledes sagt: det antal af venstreskift, der er nødvendigt for at normere den resulterende taldel) anbringes i pos. 10–19 i M-registeret som et heltal med enhed i pos. 19 [1].

Er der overløb efter en flydende operation, d.v.s. ligger resultatet uden for det flydende talområde (se afsnit 2.2), opgiver GIER, idet den hopper til celle 0, hvor der bør stå en stopordre eller et indhop i et kontrolprogram.

2.6 Central enhed.

I den centrale enhed findes et stort antal registre, men man behøver dog ikke at kende dem alle for at forstå GIERs virkemåde eller bruge GIER; vi vil her nævne de vigtigste registre.

Funktionsregisteret F er et 42-bits register, der i pos. 10–41 indeholder den aktuelle ordre på nær adressetallet under dennes udførelse. Efter et normalt stop (trykknapstyret) indeholder F_{10-41} dog den næste ordre, der skal udføres. (Efter et programmeret stop indeholder F selve stopordren). Pos. 0–9 udgør det

1) Ved division må absolutværdien af kvotienten mellem taldelene ligge mellem 0.5 og 2. Det vil sige, at der højst mistes et binært ciffer, og hverken her eller ved multiplikation gemmes der information om dette, men pos. 10–39 i M nulstilles før enhver flydende operation, således at informationen om ciffertab må gemmes eller benyttes umiddelbart efter den operation, hvor ciffertabet opstår.

såkaldte p-register eller indeksregister. Indholdet i dette bestemmes af koderen gennem visse specielle operationer, og det indgår i beregningen af indeksmærkede adresser. Omkring F grupperer sig en række 10-bits registre, der benyttes ved adresseberegningerne:

Ordretælleren r1, der under udførelsen af en ordre og ved normalt stop indeholder adressen på den næste ordre, der skal udføres.

Adresseregisteret r2, der indeholder adressen på den celle i ferritlageret, som den centrale enhed opererer på. (Det er dette registers indhold, der indgår i relative adresser, som omtales i afsnit 3.4.3).

Sekvensregisteret s1 indeholder normalt den adresse, hvorfra der sidst er udført et sekvenshop (se operationslisten); denne adresse indgår i beregningen af sekvensmærkede adresser (se afsnit 3.4.4). Registeret kan benyttes af koderen, idet man både kan gemme dets indhold og sætte et nyt indhold deri. Endnu et register s2 fungerer som hjælperegister for adresseberegningen, bl.a. når man bruger adresser, der er både sekvensmærkede og parentesmærkede (adresseberegningen omtales i afsnittene 3.4 - 3.6 og 4.2).

Tromlekanalregisteret tk indeholder nummeret på den kanal, som ferritlageret står i forbindelse med og styrer således, i hvilken kanal skrivning på (og læsning fra) tromlen foregår. Indholdet i tk kan ændres gennem en VK-operation, der "vælger en ny kanal".

Tromleadresseregisteret ta bruges under tromletransporter til at styre adressevalget i det pågældende ferritlagerafsnit (se afsnit 2.3), men er i øvrigt uden interesse for koderen. Både tk og ta er 10-bits registre.

Indikatorregisteret in er en samling af 12 1-bitregistre, der kan bruges uafhængigt af hinanden til at lagre information om overløb, nulsituation, fortegn og mærkning til senere brug. Anvendelsen af dette vigtige register omtales nærmere i afsnit 4.6 og følgende.

Overløbsregisteret O er et 1-bit register (en flip-flop), der under en aritmetisk operation sættes lig 1, hvis der er overløb i resultatet og lig 0 ellers. Indholdet af O kan ved en indikatoroperation (se kapitel 5) overføres til indikatoren til senere brug, men forbliver iøvrigt uændret indtil næste aritmetiske operation (heri medregnet talforskydning og cyklisk forskydning i R), da kun en sådan berører O-registeret. Vi bemærker, at O-registeret stilles i overensstemmelse med overløbssituationen i H-registeret, ikke i R-registeret; det gør dog kun en forskel for to operationers vedkommende (det drejer sig om addition og subtraktion i lagercelle, AC- og SC-operationerne, hvor det således bliver overløbssituationen for facit, der registreres).

Et 10-bits ydre enheds register by indeholder numrene på de ydre enheder, der er tilsluttet, og registeret styrer således, hvilke ydre enheder (se afsnit 2.7), der indlæses fra eller skrives på. Indholdet i by kan ændres gennem en VY-operation, der "vælger en ny ydre enhed".

2.7 Ydre enheder.

Til indlæsning har GIER en dielektrisk 8-huls strimmellæser, der fører et tegn ad gangen til en celle og til R-registeret; hastigheden er ca. 500 tegn/sek. Derudover er der en skrivemaskine, hvorfra man på samme måde kan indlæse et tegn ad gangen.

Til udlæsning har GIER en 8-huls perforator, der huller et tegn ad gangen med en hastighed af ca. 150 tegn/sek.; den samme skrivemaskine som ovenfor kan også bruges til udlæsning, og hastigheden er her ca. 8 tegn/sek.

2.8 Manøvrebord.

Fotografiet bagest i 2.hæfte viser GIERs manøvrebord, der består af to rækker af kontrollamper med tilhørende trykknapper, en tændingsnøgle, to sæt start-stop-knapper og en volumenkontrol for højttaler. Betydningen og virkningen af de enkelte lamper og knapper omtales i et senere kapitel i forbindelse med en almindelig "kørselsvejledning".

3. ORDRENS OPBYGNING I.

3.1 Indledning.

I dette kapitel indleder vi behandlingen af den ydre ordrekode, hvorved vi forstår den kode (eller det program), der skal nedskrives af brugeren af GIER; notationen er afhængig af hvilket indlæseprogram, der bruges. Den i det følgende benyttede notation er i overensstemmelse med de fælles konventioner for indlæseprogrammerne OLGA-DIG og SLIP. (Disse programmer omtales i 2.hæfte).
I første omgang bliver det udelukkende opbygningen af den enkelte ordre, der vil blive omtalt, og omtalen indskrænkes til de simpleste typer af ordrer, nemlig de såkaldte halvordsordrer.

3.2 Ordrens bestanddele.

En ordre indeholder altid en angivelse af hvilken grundoperation, der skal udføres, samt en adressedel.

I det simpleste tilfælde indeholder ordren ikke flere bestanddele. Som nævnt i kapitel 2 skal ordrer anbringes i celler i maskinens lager, og vi vil derfor i flæng tale om adressedelen i en ordre og i en celle.

3.3 Grundoperation.

Svarende til de 57 grundoperationer, som GIER kan udføre, har man valgt 57 kombinationer af to bogstaver. Ønsker man en bestemt grundoperation udført, skrives den tilhørende bogstavkombination som første del af ordren. De to bogstaver, der hører til en operation, er ofte hentet fra de ord, der beskriver operationens virkning. Således benyttes bogstaverne AR til angivelse af grundoperationen "Adder til Resultatregisteret". Som et andet eksempel anfører vi, at bogstaverne SC benyttes til angivelse af operationen "Subtraher fra Celle". I kapitel 5 findes en liste over samtlige 57 grundoperationer og de bogstavkombinationer, der skal benyttes som kodetegn for de forskellige operationer.

Vi bemærker i øvrigt, at beskrivelserne "Adder til Resultatregisteret" og "Subtraher fra Celle" er alt for upræcise. I kapitel 5 gennemgår vi de 57 grundoperationer, idet vi for hver enkelt operation gør rede for, hvad der sker i GIER, når maskinen udfører en ordre, der indledes med kodebogstaverne for den pågældende operation.

3.4 Adressedel.

Adressedelen i en ordre kan angives på flere forskellige måder, nemlig som henholdsvis absolut adresse, indeksmærket adresse, relativ adresse og sekvensmærket adresse.

3.4.1 Absolut adresse.

En absolut adresse angives ved et heltal h i intervallet

$-512 \leqq h \leqq 1023$, og virkningen er for de fleste operationers vedkommende, at h – taget modulo 1024 – opfattes som adresse på operanden. h kaldes adressetallet.

Eksempel 3.1.

Adressen 714 vil i ordren AR 714 bevirke, at det tal, der står i celle 714 adderes til indholdet af resultatregisteret.

Eksempel 3.2.

Adressen −15 vil i ordren MK −15 bevirke, at det tal, der står i celle (−15 + 1024) d.v.s. tallet i celle 1009, bliver multipliceret med indholdet af multiplikatorregisteret, hvorefter indholdet af resultatregisteret adderes til produktet.

3.4.2 Indeksmærket adresse.

En indeksmærket adresse noteres ved hjælp af bogstavet p efterfulgt af et heltal h i intervallet $-512 \leqq h \leqq 1023$. Vi bemærker, at positive heltal skal skrives med fortegn. Virkningen af en indeksmærket adresse er for de fleste operationers vedkommende, at summen af indholdet i indeksregisteret og adressetallet h, taget modulo 1024, opfattes som adressen på operanden. Indeksregisterets indhold er altid et heltal i intervallet $0 \leqq p \leqq 1023$.

Eksempel 3.3.

Lad indholdet af indeksregisteret være 42. Da vil adressen p + 24 i ordren SR p + 24 bevirke, at tallet i celle 66 subtraheres fra indholdet af resultatregisteret.

Eksempel 3.4.

Lad indholdet af indeksregisteret være 127. Ordren SC p-212 vil da først bevirke, at $127 - 212 + 1024 = 939$ opfattes som adresse på operanden, hvorefter indholdet af resultatregisteret subtraheres fra indholdet af celle 939.

3.4.3 Relativ adresse.

En relativ adresse noteres ved hjælp af bogstavet r efterfulgt af et heltal h i intervallet $-512 \leqq h \leqq 1023$. Vi bemærker, at positive heltal skal skrives med fortegn. Lad n være nummeret på den celle, der indeholder ordren med den relative adresse. Virkningen af denne relative adresse vil da normalt være, at summen $n + h$, taget modulo 1024, opfattes som adressen på operanden.

Eksempel 3.5.

Lad den aktuelle ordre være AR $r - 7$, og lad denne ordre stå i celle 72. Da vil $72 - 7 = 65$ blive opfattet som adresse på operanden, d.v.s. at tallet i celle 65 adderes til resultatregisteret.

Eksempel 3.6.

I celle 973 står ordren AC $r + 566$. Ved gennemløbet af denne ordre vil $973 + 566 - 1024 = 515$ blive opfattet som adresse på operanden, d.v.s. at tallet i resultatregisteret adderes til tallet i celle 515.

3.4.4 Sekvensmærket adresse.

En sekvensmærket adresse noteres ved hjælp af bogstavet s efterfulgt af et heltal h i intervallet $-512 \leqq h \leqq 1023$. Vi bemærker, at positive heltal skal skrives med fortegn. Virkningen af en sekvensmærket adresse er da, at indholdet af sekvensregisteret adderes til tallet h, hvorefter summen, taget modulo 1024, opfattes som adressen på operanden [1]. Sekvensregisterets indhold er altid et heltal i intervallet $0 \leqq s \leqq 1023$. Brugen og nytten af denne adresseringsform omtales senere.

3.5 Parentesmærkning af adresser.

I praksis vil man ofte have adressen på den celle, der indeholder en operand, stående som adressedel i en anden celle. Man har da mulighed for at få fat i operanden på en meget simpel måde, der blot består i at skrive en parentes om adressedelen i en ordre. Fremgangsmåden illustreres ved hjælp af de følgende eksempler.

Eksempel 3.7.

Lad den ønskede operand stå i celle 715, og lad tallet 715 stå som adresse i celle 43. Skriver vi nu en ordre med adressedelen (43), vil parentesen bevirke, at tallet 43 ikke opfat-

1) Skal operandadressen netop være indholdet af sekvensregisteret, skrives enten s eller s + 0, der vil give samme virkning. Det tilsvarende gælder for relativ og for indeksmærket adresse.

tes som adressen på operanden, men som tegn på, at operandens adresse findes i celle 43. Adressedelen (43) vil i dette tilfælde have helt samme virkning som adressedelen 715.

Eksempel 3.8.

Lad ordren SR (r − 6) stå i celle 862. Først dannes 862 − 6 = 856, og på grund af parentesen opfattes tallet 856 som adressen på den celle, hvor operandens adresse er lagret. Såfremt adressen i celle 856 er 14, vil operanden blive hentet i celle 14.

Eksempel 3.9.

Lad indholdet af indeksregisteret p være 38 og lad en ordre have formen MK (p + 10). Først dannes 38 + 10 = 48, der på grund af parentesen opfattes som adressen på den celle, der indeholder adressen på operanden. Såfremt adressedelen i celle 48 er r + 2, vil operanden blive hentet fra celle 50.

Eksempel 3.10.

Parentes-mærkningen er rekursiv, d.v.s. at såfremt den adresse, man finder frem til ved hjælp af en parentesmærket adressedel, selv er parentesmærket, vil der indgå endnu et mellemled, før adressen på operanden er fundet. Vi taler i så fald om en parenteskæde.

Lad adressedelen i den aktuelle ordre være (139) og lad adressedelen i celle 139 være (20). Endelig antager vi, at adressedelen i celle 20 er 888. I så fald vil den oprindelige adressedel (139) bevirke, at operanden hentes i celle 888.

NB: Er en sådan parenteskæde lukket, d.v.s. henviser den til "sig selv", vil ordren aldrig blive udført, fordi GIER vil blive ved med at søge efter den endelige adresse!

3.6 Den modificerede adresse.

Eksempel 3.9 viser, at ved en parentesmærket ordre beregnes den endelige adresse på grundlag af en helt "ny" adressedel med den dergældende r-værdi. Det fundne tal vil vi kalde den modificerede adresse (i kapitel 4 vil det vise sig, at det ikke behøver at være den resulterende adresse):

Den modificerede adresse defineres som det heltal h i intervallet $0 \leqq h \leqq 1023$, der beregnes af GIER på grundlag af bestanddelene i adressedelen i sidste led af parenteskæden. For alle de ordrer, der omtales i dette kapitel, er den modificerede adresse det samme som den resulterende adresse, d.v.s. det tal, der viser, fra hvilken celle operanden skal hentes.

Eksempel 3.11.

Lad ordren AR $r-48$ stå i celle 20. Den resulterende adresse er da $20-48+1024 = 996$.

Eksempel 3.12.

Lad indholdet af indeksregisteret p være 900, og lad indholdet af celle 276 være AR 27. Ordren MK $(p+400)$ har da den resulterende adresse 27 (stående i celle 276).

3.7 S-variant af grundoperation.

Efter de to bogstaver, der benyttes til angivelse af en grundoperation, kan man tilføje bogstavet S. I så fald vil resultatregisteret blive nulstillet før grundoperationen udføres [1]. Man siger også, at indholdet af resultatregisteret bliver slettet før operationen udføres. Valget af bogstavet S skyldes denne talemåde.

Eksempel 3.13.

Ordren ARS 439 vil bevirke, at resultatregisteret nulstilles, hvorefter indholdet af celle 439 adderes til resultatregisteret. Virkningen af ordren kan kortere beskrives som en overførsel af indholdet af celle 439 til resultatregisteret.

3.8 F-variant af grundoperation.

Efter de to bogstaver, der benyttes til angivelse af en grundoperation, kan man tilføje bogstavet F. Dette har dog kun betydning for de aritmetiske operationer. For disse operationers vedkommende vil udregningen foregå som regning med tal på flydende form, når ordren er F-mærket. Virkningen af andre operationer ændres ikke ved F-mærkning.

Eksempel 3.14.

Ordren SRF 45 vil bevirke, at subtraktionen finder sted med indholdet af celle 45 og indholdet af resultatregisteret opfattet som tal på flydende form.

---

1) d.v.s. at pos. 00-39 i R-registeret nulstilles, mens mærkebits (nr. 40-41) er uændrede.

F-varianten kan benyttes samtidig med S-varianten, og her bevirker S-varianten, at hele det flydende resultatregister RF, der består af R-registeret [1] samt pos. 0-9 i M-registeret, nulstilles. Den rækkefølge, hvori S og F skrives, er ligegyldig.

Eksempel 3.15.

Ordren ARSF 137 vil bevirke, at tallet i celle 137, opfattet som tal på flydende form, overføres til resultatregisteret RF.

Som afslutning bringer vi et skema, der viser de mulige bestanddele af en halvordsordre:

	S	F	
Altid 2 bogstaver. 57 forskellige muligheder, der alle er nævnt i kapitel 5	S eller F eller både S og F kan udelades		Altid en adressedel. 8 forskellige muligheder: h, p+h, r+h, s+h, (h), (p+h), (r+h) eller (s+h), hvor h står for et heltal i intervallet $-512 \leq h \leq 1023$, medens bogstaverne p, r, s skal skrives.

En halvordsordre kan lagres, således at den fylder en halvcelle i lageret. To halvordsordrer kan lagres i samme helcelle. Benyttes denne lagringsform, er der et specielt forhold, der må nævnes, nemlig at en F-mærkning af en halvordsordre også bevirker, at den anden halvordsordre i samme helcelle bliver F-mærket.

1) På nær mærkebits.

4. ORDRENS OPBYGNING II.

4.1 Indledning.

Der er intet i vejen for at opbygge programmer ved hjælp af halvordsordrer, men normalt vil det være en stor fordel at udnytte muligheden for at udbygge halvordsordrer til helordsordrer. Denne udbygning kan finde sted på flere forskellige måder, nemlig ved tilføjelse af et tælletal, eller ved hjælp af X-mærkning, V-mærkning eller D-mærkning. Eventuelt kan flere af disse (eller alle disse) muligheder benyttes samtidig.

Ved siden af disse udvidelser af ordren er der endnu mulighed for at udvide ved tilføjelse af en indikatordel. Dette forhold omtales til sidst.

4.2 Tælletal.

Tælletallet i en ordre angives ved hjælp af et heltal, h, i intervallet $-512 \leq h \leq 1023$. Tælletallet skal noteres sidst i ordren. Vi bemærker, at positive heltal skal skrives med fortegn. Virkningen af tælletallet er, at dette adderes til den modificerede adresse inden ordren udføres. Vi fremhæver, at såfremt adressen ikke er parentesmærket, ændres den aktuelle ordre, idet dens adressetal bliver summen af tælletallet og det tidligere adressetal. Er adressen parentesmærket, sker ændringen i den ordre, der indeholder sidste led i parenteskæden (det er netop denne ordres adressedel, der benyttes ved beregningen af den modificerede adresse). Vi henviser i øvrigt til de følgende eksempler.

Eksempel 4.1.

Ordren AR 814 +2 vil bevirke, at ordren, inden den udføres, ændres til AR 816 +2, hvorefter indholdet af celle 816 adderes til indholdet af resultatregisteret. I praksis benyttes ordrer med tælletal normalt kun, når disse ordrer skal gennemløbes flere gange. Ved næste udførelse af den her nævnte ordre vil denne ændres til AR 818 +2, hvorefter det er indholdet af celle 818, der adderes til indholdet af resultatregisteret. Det er således, at en enkelt ordre med tælletal ved gentagen anvendelse kan bevirke, at summen af en hel række af tal bliver dannet.

Eksempel 4.2.

Lad ordren SRF r + 50 -1 stå i celle 124. Første gang ordren udføres, ændres den til SRF r + 49 -1, og tallet i celle 173 subtraheres fra indholdet i resultatregisteret. På grund af F-mærkningen foregår regningen med flydende tal. Anden gang ordren udføres, ændres den til SRF r + 48 -1, og tallet i celle 172 subtraheres o.s.v.

Eksempel 4.3.

Lad adressedelen i celle 289 være 837. Ordren MK (289) +1 vil da ved første gennemløb bevirke, at adressedelen i celle 289 øges med 1 til 838, hvorefter indholdet af celle 838 opfattes som multiplikand. (Såfremt der står to halvordsordrer i celle 289, er der to adressedele i denne. I så fald er det kun adressen i første halvordsordre, der ændres). Næste gang ordren gennemløbes, øges adressedelen i 289 igen (til 839) o.s.v. Vi fremhæver, at selve ordren MK (289) +1 ikke ændres, men adresseændringen finder sted i celle 289. Yderligere bemærker vi, at såfremt adressen i 289 igen er parentesmærket, vil tællingen først foregå "i næste led". Helt almindeligt kan vi fastslå, at adresseændringen foregår i sidste led i en parenteskæde.

Eksempel 4.4.

Lad indholdet af indeksregisteret p være 72. Ordren AR (p + 7) +2 vil da - før grundoperationen udføres - bevirke, at adressen i celle 79 øges med 2 hver gang ordren AR (p + 7) +2 gennemløbes (forudsat at adressen i celle 79 ikke er parentesmærket).

Eksempel 4.5.

Lad celle 34 og 35 indeholde ordrerne

34: AR (35) +1
35: SC 218 -2

Den første ordre udføres som AR 219; samtidig ændres den anden ordre til SC 219 -2, hvorefter den udføres som SC 217 og samtidig ændres til SC 217 -2.

Det bemærkes, at virkningen af enkelte specielle halvordsordrer hænger sammen med, at GIER altid giver en halvordsordre tælletallet 0 (se operationslisten i kapitel 5).

1.3 X-variant af grundoperation.

Efter adressedelen kan man tilføje bogstavet X. I så fald vil indholdet af resultatregisteret og multiplikatorregisteret blive ombyttet efter at grundoperationen er udført. Det vil sige, at indholdet i pos. 0-39 i de to registre ombyttes, hvorefter R's bit nr. 00 sættes lig (den nye) bit nr. 0. Vi siger, at indholdene af de to registre "krydses". (Bogstavet X symboliserer et kryds).

Eksempel 4.6.

Ordren SR 445 X vil bevirke, at indholdet af celle 445 subtraheres fra indholdet af resultatregisteret, hvorefter indholdet af resultat- og multiplikatorregistrene bytter plads.

.4 V-variant af grundoperation.

Efter adressedelen kan man tilføje bogstavet V. V-mærkningen af en ordre bevirker slet ingen ændring af denne; men V-mærkningen vil medføre, at GIER springer den følgende helcelle over.

Eksempel 4.7.

Lad ordren ARS 705 V stå i celle 212. Efter at denne ordre er udført, fortsætter GIER med ordren i celle 214. Den mellemliggende celle 213 kan f.eks. benyttes til opbevaring af mellemresultater.

Eksempel 4.8.

En V-modificeret ordre skal altid stå i en helcelle, og den næste helcelle overspringes, uanset om den indeholder en helordsordre eller to halvordsordrer (eller et tal).

4.5 D-variant af grundoperation.

Efter adressedelen kan man tilføje bogstavet D. Virkningen af D-mærkningen afhænger af, hvilken grundoperation der er på tale. I operationslisten beskrives D-varianten for hver enkelt operation for sig. Almindeligt gælder, at den resulterende adresse i en D-mærket ordre vil blive benyttet som operand i regningerne (i modsætning til det normale, at adressen viser, hvor operanden befinder sig).

Eksempel 4.9.

Ordren AR 309 D bevirker, at selve tallet 309 bliver opfattet som operand, således at 309 vil blive adderet til adressedelen i resultatregisteret (positionerne 0-9), mens positionerne 10-39 i R-registeret er uændrede.

Eksempel 4.10.

Er adressedelen i en D-mærket ordre parentesmærket, viser adressedelen hen til en celle, hvis adressedel vil blive opfattet som operand. Lad ordren i celle 444 have adressedelen 615. Ordren AR (444) D +7 vil da bevirke, at tallet 622 vil blive adderet til adressedelen i resultatregisteret (desuden vil adressedelen i celle 444 nu være 622), thi først udføres adressemodifikationen (tællingen) på sædvanlig måde, og derefter udføres den D-modificerede operation.

X-, V- og D-varianterne kan benyttes samtidig (eller man kan benytte to af disse varianter i samme ordre). Den rækkefølge, i hvilken bogstaverne X, V og D noteres, spiller ingen rolle.

Eksempel 4.11.

Ordren ARS 42 XVD +1 tænkes lagret i celle 100. Udførelsen af ordren vil bevirke, at følgende ting sker:

1) Ordren ændres til ARS 43 XVD +1
2) Resultatregisteret nulstilles
3) Tallet 43 adderes til adressedelen i resultatregisteret
4) Resultatregisteret og multiplikatorregisteret krydses
5) Næste ordre hentes i celle 102.

Følgende skema viser de forskellige muligheder for opbygning af en ordre. Indikatordelen omtales på de følgende sider.

	S	F		X	V	D		
Altid 2 bogstaver	kan udelades		adressedel				indikatordel	tælletal

kan helt eller delvis udelades; findes blot én af disse bestanddele, lagres ordren i en helcelle.

S og F kan ombyttes. X, V og D kan skrives i vilkårlig rækkefølge.

4.6 Ordre med indikatordel.

Den sidste mulighed for at udvide en ordre består i at tilføje en indikatordel. Inden vi gennemgår de mange forskellige – og ofte særdeles nyttige – anvendelser af indikatoren, vil vi imidlertid fremhæve, at man i mange tilfælde kan lave udmærkede programmer uden at bruge indikatoren. Det kan derfor anbefales, at man sætter sig grundigt ind i operationslisten og gennemregner de første eksempler fra kapitel 7, inden man studerer listen over indikatordele og deres virkninger i kapitel 6. Før vi omtaler udseendet af en indikatordel, skal vi minde om det såkaldte indikatorregister.

4.6.1 Indikatorregister.

Indikatorregisteret, der er en del af GIERs kontrolenhed (se kapitel 2), består af 12 1-bit registre (12 flip-flops), der hver kan bruges til at lagre information om en vis tilstand eller et vist resultat til senere brug i administrationen af et program.

OA OB	TA TB	PA PB QA QB RA RB	KA KB
overløb eller nul i R	fortegn	a- og b-mærkning	kan kun indstilles fra manøv-rebordet.

Figuren viser de navne, de enkelte registre har fået, og teksten antyder, hvilken information der kan gemmes i de enkelte dele.

4.6.2 Indikatordel.

Indikatordelen består af en indikatoroperation og en indikatoradresse[1]. Begge bestanddele angives ved hjælp af store bogstaver. Indikatordelen skal (sammen med X, V og D varianter) noteres efter adressedelen og før et eventuelt tælletal.

En ordre kan højst indeholde én indikatordel.

4.6.3 Indikatoroperation.

Indikatoroperationen, der skal stå først i indikatordelen, skrives med et af de fire bogstaver I, M, N og L.

Her bevirker I, at en oplysning indiceres, d.v.s. lagres i indikatoren til senere brug. Hvilken oplysning, der skal indiceres, bestemmes af indikatoradressen (se næste afsnit).

M bevirker (ved nogle af grundoperationerne) en vis mærkning af operandcellen, d.v.s. den celle i lageret, som den resulterende adresse henviser til. Hvilken mærkning, der sker, bestemmes af indikatoradressen.

---

1) I et enkelt tilfælde skrives ingen indikatoradresse. Se listen over indikatordele, kapitel 6.

N og L bevirker, at den aktuelle ordre gøres betinget, d.v.s. at grundoperationen kun udføres, hvis en bestemt betingelse (enten i indikatoren eller i den aritmetiske enhed) er opfyldt. Igen fortæller indikatoradressen, hvilken betingelse det drejer sig om.

4.6.4 Indikatoradresse.

Indikatoradressen noteres som et eller to bogstaver umiddelbart efter indikatoroperationen. Som indikatoradresse kan benyttes navnene på de tolv registre i indikatoren OA, ..., KB, og hver af disse indikatoradresser henviser til det tilsvarende register. Indikatoradresserne OC, TC, ..., KC henviser henholdsvis til de to O-bits, til de to T-bits etc. i indikatoren. Endelig henviser indikatoradresserne A, B og C til en eller to mærkebits i operandcellen eller i R-registeret.

Den detaillerede virkning af alle de mulige indikatordele er beskrevet i listen over indikatordele, afsnit 6.1.

4.7 Intern ordrerepræsentation.

En helordsordre eller to halvordsordrer kan som omtalt stå i en celle i lageret. Det kan lejlighedsvis være af betydning for brugeren af GIER at vide, hvordan ordrens (eller ordrernes) forskellige bestanddele fordeler sig på de ialt 42 bits, der tilsammen udgør en celle. Yderligere kan det være nødvendigt for koderen at kende de værdier af de 42 bits, der svarer til en bestemt ordre. Vi afslutter derfor disse kapitler om ordrens opbygning med en oversigt over den såkaldte interne ordrerepræsentation.

4.7.1 Transformation fra den ydre ordrekode til den indre ordrerepræsentation.

Under indlæsningen af en ordre, der er nedskrevet i overensstemmelse med de foran givne anvisninger, sker der en transformation af ordren, inden denne lagres som en kombination af 1'er og 0'er i de 42 positioner i en celle. Denne transformation udføres af indlæseprogrammet (OLGA-DIG eller SLIP), der i forvejen må være lagret i maskinen. Indlæseprogrammet behandler de enkelte bestanddele af en ordre hver for sig. Det følgende skema viser, hvilke bits der benyttes til lagring af de enkelte dele af en helordsordre.

4.7.2 Helordsordrens interne repræsentation.

0..9	10..19	20..25	26	27	28 29	30	31	32	33 34	35 36 37	38 39	40	41
adressetal	tælletal	grundoperation	S	()	r, s og p	X	V	D	indikator-operation	K Z O T P Q R	A B C	O (nul)	F
										indikator-adresse			

De enkelte bits udfyldes i overensstemmelse med følgende regler:

Pos. 0-9 indeholder adressedelen skrevet som heltal i det binære talsystem.

Pos. 10-19 indeholder tælletallet skrevet som heltal i det binære talsystem.

Pos. 20-25 benyttes til angivelse af, hvilken grundoperation ordren indeholder. De seks bits tillader egentlig anvendelsen af $2^6 = 64$ forskellige grundoperationer. Som omtalt er kun 57 af disse muligheder udnyttet. En oversigt over sammenhængen mellem grundoperationer og bitkombinationer findes i kapitel 5.

Pos. 26 har værdien 1, når ordren er S-mærket, ellers værdien 0.

Pos. 27 har værdien 1, når ordren er parentesmærket, ellers værdien 0.

Pos. 28 og 29 udnyttes på følgende måde:

Adresse	Indhold i 28	29
absolut	0	0
relativ	1	0
indeksmærket	1	1
sekvensmærket	0	1

Pos. 30 har værdien 1, når ordren er X-mærket, ellers værdien 0.

Pos. 31 har værdien 1, når ordren er V-mærket, ellers værdien 0.

Pos. 32 har værdien 1, når ordren er D-mærket, ellers værdien 0.

Pos. 33 og 34 udnyttes til angivelse af indikatoroperation således:

Indikatoroperation	Indhold i 33	34
ingen indik.oper. eller I	0	0
M	0	1
N	1	0
L	1	1

Pos. 35, 36, 37, 38 og 39 udnyttes til angivelse af indikatoradres-

se således:

Første del af indikatoradresse	Indhold i 35	36	37
ingenting	0	0	0
K	0	0	1
Z	0	1	0
O	0	1	1
T	1	0	0
P	1	0	1
Q	1	1	0
R	1	1	1

Anden del af indikatoradresse	Indhold i 38	39
ingenting	0	0
A	1	0
B	0	1
C	1	1

Pos. 40 Ved en helordsordre skal denne bit have værdien 0.

Pos. 41 har værdien 1, når ordren er F-mærket, ellers værdien 0.

4.7.3 To halvordsordrers interne repræsentation.

0...9	10..19	20..25	26	27	28 29	30..35	36	37	38 39	40	41
adressedel af 1.ordre	adressedel af 2.ordre	grundoperation af 1.ordre	S af 1.ordre	( ) af 1.ordre	r, s og p af 1.ordre	grundoperation af 2.ordre	S af 2.ordre	( ) af 2.ordre	r, s og p af 2.ordre	1	F

Af det foregående fremgår, hvordan de enkelte positioner udfyldes. Vi bemærker specielt, at pos. 40 skal have værdien 1 i dette tilfælde, hvor der er lagret to halvordsordrer i cellen. Som tidligere omtalt vil en F-mærkning af en af de to ordrer virke på begge ordrerne.

Ved hjælp af et par eksempler viser vi betydningen af, at brugeren af GIER er fortrolig med den indre ordrerepræsentation.

Eksempel 4.12.

På et vist trin af en beregning ønsker man at ændre adressen i en bestemt ordre. Dette kan gøres ved hjælp af en anden ordre med grundoperationen GA (Gem Adressetal). Her er det imidlertid afgørende at vide, at dette ikke ændrer en eventuel parentesmærkning, relativmærkning m.v. Endvidere vil operationen kun have den ønskede virkning anvendt på en helordsordre eller på den første af to halvordsordrer.

Eksempel 4.13.

Ønsker man f.eks. at X-mærke en ordre, der står i en celle, kan dette gøres ved at indsætte 1 i position 30 i den pågældende celle. Grundoperationen AB (adder logisk) er velegnet til dette formål.

Til slut fremhæver vi, at den indre ordreopbygning er karakteristisk for maskinen, idet dennes kredsløb netop er opbygget i relation til den indre ordreopbygning. Den ydre ordrekode kan derimod vælges på mange forskellige måder, idet man blot for hvert valg af denne må indrette et specielt indlæseprogram.

5. OPERATIONSLISTE.

5.1 Indledning.

I dette afsnit gennemgås de 57 grundoperationer i GIER, én efter én, og i afsnit 5.3 findes en "brugsanvisning" til selve operationslisten, der findes i afsnit 5.4. Men først vil vi omtale, hvordan en kompliceret GIER-ordre udføres; desuden angiver vi her nogle omtrentlige operationstider.

I kapitel 6 findes en fuldstændig liste over de mulige indikatordele og disses virkninger.

5.2 Ordrens udførelse.

En GIER-ordre kan have mange bestanddele (som eksempel på en sammensat ordre kan tages SCS (r − 18) XV IOA +4). I nedenstående skema er angivet den orden, hvori disse dele udføres samt den tid, udførelsen tager (tidsenheden er 1 μs (mikrosekund) = 10^{-6} sekund).

Rækkefølge	Varighed
1. Indikatorbetingelse (N- og L-operation) undersøges.:	
Bet.ikke opfyldt: Næste ordre påbeg. efter	15 μs
Bet.opfyldt eller ingen N- og L-operation	ingen ekstra tid
2. Beregning af resulterende adresse, incl.r-, s- eller p-mærkning, men excl. tælletal og parentesmærkning	27 μs
Tælletal ≠ 0 : Tiden øges med	+ 9 μs
Parentesmærkning, ikke s-mærkning: Tiden øges pr.led med	+ 12 μs
Parentesmærkning og s-mærkning : - - - - -	+ 26 μs
3. Eventuel S-variant udføres	ingen ekstra tid
4. Grundoperation (eller dennes F-variant eller D-variant) udføres, herunder eventuel registrering af mærkning og overløb	afhænger af grundoperationen
5. Evt. indikatoroperation (I- eller M-operation) og evt. V-variant udføres	ingen ekstra tid
6. Evt. X-variant udføres: Tiden øges med	+ 4 μs
7. Evt. IK-indikatoroperation udføres: Tiden øges med	+ 6 μs

Den tid, det tager at udføre selve grundoperationen, kan i mange tilfælde kun opgives som et gennemsnit, fordi de numeriske forhold påvirker operationstiden: At multiplicere med cifferet 0 er hurtigere end at multiplicere med cifferet 1, og den samlede operationstid for en multiplikation afhænger således af, hvor mange ettaller og hvor mange nuller, der er i multiplikatoren. Ligeledes er D-varianten af en operation i nogle tilfælde hurtigere end grundoperationen. Med disse forbehold giver vi i nedenstående skema en oversigt over den tid, punkt 4 tager for de forskellige grundoperationer.

	Grundoperation µs	F-variant µs
QQ PS-PP VK	ca. 2	
GR-GM-GA-GT-GP-GS-GI-GK PM-PA-PT BS-BT-UD-XR LK-SK (se nedenfor) HV-HH-HK	ca. 9	kun GR = ca. 9
IS-IT-NS-NT-PI MB HS-VY	ca. 16	
AR-AC-AN-SR-SC-SN MT-CA-NC HR	ca. 22	ca. 66 +2.2·exponent diff. +2.2·antal norm.skift
AB-CM	ca. 27	
TK-TL-NK-NL-CK CL	ca.20+ 2.2 pr.skift 4.4 pr.skift	ca.30 + 2.2 pr.skift
MK-ML	ca. 155 (i middel)	ca. 140 (i middel)
DK-DL	ca. 240 (i middel)	ca. 190 (i middel)

LK og SK afsluttes først efter 20 millisek., men efter 9 µs kan GIER udføre andre ordrer simultant med tromletransporten; dog forsinkes udførelsen af disse andre ordrer med ialt 1 millisek. mens tromletransporten foregår.

En simpel addition eller subtraktion uden parentesmærket adresse og uden tælling tager således ca. 50 µs. En simpel multiplikation varer gennemsnitlig 180 µs og en simpel division varer gennemsnitlig 270 µs. En flydende multiplikation eller division er lidt hurtigere end den tilsvarende operation med fast komma, og en flydende addition eller subtraktion er ca. dobbelt så langsom som den tilsvarende "fast-komma"-operation.

5.3 Forklaring til operationslisten.

På de følgende sider gennemgås virkningerne af grundoperationerne i detailler. I beskrivelsen benyttes nogle standardudtryk med ALGOL-lignende betegnelser, hvis betydning vi vil fastlægge således:

c — betegner den resulterende adresse, som den beregnes ud fra adressedel med eventuel r-, s- eller p-mærkning, parentesmærkning og eventuelt tælletal. Den celle, som c henviser til, benævnes operandcellen. c opfattes, når det henviser til en lagercelle, som et tal i intervallet $0 \leqq c \leqq 1023$, men i mange andre tilfælde (f.eks. når tallet c selv indgår i en aritmetisk operation) som et heltal i intervallet $-512 \leqq c \leqq 511$.

celle[c] — betegner indholdet af celle c exclusive pos. 40-41 opfattet som maskintal, altså et tal i intervallet $-1 \leqq$ celle[c] < 1.

flydcelle[c] betegner indholdet af celle c opfattet som flydende tal.

adressetal[c], tælletal[c] betegner indholdene af henholdsvis pos. 0-9 og pos. 10-19 i celle c opfattet som heltal.

oper[c], voper[c], hoper[c] betegner indholdene af henholdsvis hele operationsdelen, 1.operationsdel og 2.operationsdel i celle c.

pos[i,c] — betegner indholdet af pos.i i celle c opfattet som logisk (boole'sk) variabel [1].

---

1) Her og i det følgende tolkes cifrene 0 og 1 som henholdsvis false og true.

R	betegner indholdet af resultatregisterets pos. 00-39 opfattet som maskintal. I visse forbindelser omfatter det kun pos. 0-39, men dette er da anført. Endelig benyttes R undertiden i teksten som betegnelse for selve registeret.
RF	betegner indholdet af resultatregisterets pos. 00-39 samt multiplikatorregisterets pos. 0-9 opfattet som flydende tal.
RM	betegner indholdet af det lange register bestående af R samt pos. 1-39 i M.
Radr, Rtæl	betegner indholdet af henholdsvis adressepositionerne (pos. 0-9) og tællepositionerne i R-registeret.
R00, Rpos[i]	betegner indholdet af en enkelt position i R-registeret opfattet som boole'sk variabel.
M, Madr, Mtæl, Mpos[i]	er de ganske tilsvarende betegnelser for M-registeret. Enkelte steder bruges tilsvarende betegnelser også for indholdet af regneregisteret H og for indholdet af funktionsregisteret F.
s1	betegner indholdet af sekvensregisteret.
r1	betegner indholdet af ordretælleren, som for helordsordrer og højre halvordsordrer øges med 1 allerede under den indledende adresseberegning.
r	betegner den aktuelle ordres adresse.
p	betegner indholdet af indexregisteret.
indikator	betegner indholdet af indikatoren på nær KA og KB.
indik[j]	betegner indholdet af en enkelt position i indikatoren opfattet som boole'sk variabel. Nummereringen er sådan, at indik[0] = OA, indik[9] = RB, indik[10] = KA og indik[11] = KB.

Foruden den resulterende adresse c benyttes følgende adressebetegnelser:

D-adressen eller Dadr. betyder adressen på den celle der indeholder det sidste led i parenteskæden, d.v.s. indeholder den adressedel, der virkelig benyttes ved udregning af den resulterende adresse.

Den modificerede adresse betegner den resulterende adresse på nær tælletallet. For statiske operationer er den resulterende og den modificerede adresse det samme tal.

statisk betyder at tælletallet <u>ikke</u> indgår i beregningen af den resulterende adresse for den pågældende grundoperation. De fleste operationer er imidlertid <u>tællende</u>, hvilket betyder at tælletallet indgår i adresseberegningen som omtalt i kapitel 4, og dette er ikke angivet explicit i operationslisten.

+ i rubrikken "Registr.M." betyder, at mærkningen registreres, d.v.s. at indholdet i pos. 40-41 i celle c (operand-cellen) overføres til $R_{40,41}$. Er rubrikken tom, er $R_{40,41}$ uændrede.

+ i rubrikken "Registr.O." betyder, at <u>overløbssituationen i H-registeret</u> registreres i overløbsregisteret O. Er rubrikken tom, er O-registeret uændret.

+ i rubrikken "Mærkeoper." betyder, at ordren kan forsynes med en mærkeoperation (en M-indikatoroperation), hvormed indholdet i pos. 40-41 i celle c kan ændres. Er rubrikken tom, kan mærkningen i celle c ikke ænd-

res (men en absolut mærkeoperation vil stadig ét-
stille $R_{40,41}$).

\+ i rubrikken "Ind. af tegn" betyder, at forsynes ordren med indikatordel for fortegns-indicering (ITA, ITB eller ITC), vil resultatets fortegn i H-registeret blive indiceret. Er rubrikken tom, vil indikatordelene ITA, ITB og ITC stadig indicere fortegnet fra H-registeret, men det vil være fortegnet for den resulterende adresse eller en anden mærkværdig størrelse.

De forskellige varianter af grundoperationerne, F-, S-, D-, X- og V-varianterne, er omtalt i operationslisten efter følgende principper:

1. F-varianter er omtalt i de tilfælde, hvor disse har speciel betydning. Såfremt F-varianten ikke er medtaget, betyder dette, at det er ligegyldigt, om operationen er F-mærket eller ej.
2. S- og V-varianter omtales slet ikke, da de altid virker som beskrevet i kapitlerne 3 og 4, og X-varianten er kun omtalt i de få tilfælde, hvor virkningen ikke udelukkende er den sædvanlige ombytning af R og M.
3. D-varianten er omtalt ved alle de operationer, hvor adressen er relevant.

Den adresseberegning, der indleder enhver ordres udførelse, kan i ALGOL (dog uden alle erklæringerne) beskrives således:

```
Ny ordre: r2:= r1;
          s2:= s1;
   if halvord then begin
      if ¬ højrehalvord then begin Fvoper:= voper[r2];
                                   Hadr:= adressetal[r2];
                                   r modif:= pos[28,r2];
                                   s modif:= pos[29,r2];
                                   parentes:= pos[27,r2];
                                   højrehalvord:= true;
                                   end venstrehalvord;
                        else begin Fvoper:= hoper[r2];
                                   Hadr:= tælletal[r2];
                                   r modif:= pos[38,r2];
                                   s modif:= pos[39,r2];
                                   parentes:= pos[37,r2];
                                   højrehalvord:= false;
                                   r1:= r1 + 1 end højrehalvord;

                        Ftæl:= 0;
                        Stop halvord: comment: Her stoppes ved
                        tryk på NORMAL STOP; end halvord;
                 else begin Foper:= oper[r2];
                            Hadr:= adressetal[r2];
                            Ftæl:= tælletal[r2];
                            r modif:= pos[28,r2];
                            s modif:= pos[29,r2];
                            parentes:= pos[27,r2];
                            r1:= r1 + 1;
                            højrehalvord:= false;
                            Stop helord: comment: Her stoppes ved
                            tryk på NORMAL STOP;
                            if indikatorbetingelse ej opfyldt then
                            go to Ny ordre end helord;

Adresseberegn: if parentes then go to Modif;
               if statisk operation then go to Modif;
               adressetal[r2]:= Hadr:= Hadr + Ftæl;
Modif: if s modif ∧ ¬ (r modif) then begin Hadr:= Hadr + s2;
                                          s2:= tælletal[s2] end s modif;
       if r modif ∧ ¬ (s modif) then Hadr:= Hadr + r2;
       if r modif ∧ s modif      then Hadr:= Hadr + p;
       if parentes then begin r2:= Hadr;
                              Hadr:= adressetal[r2];
                              r modif:= pos[28,r2];
                              s modif:= pos[29,r2];
                              parentes:= pos[27,r2];
                              go to Adresseberegn end parentes;
```

Herefter påbegyndes udførelsen af selve operationen, og den <u>resulterende adresse c</u> er netop c = Hadr, mens <u>D-adressen</u> er Dadr = r2.

Hvor tælletallet benyttes under grundoperationens udførelse, tages det fra F-registerets tællepositioner Ftæl.

Betegnelse	Virkning	Registr. M.	Registr. O.	Mærke-oper.	Ind. af tegn	Specielle bemærkninger
AR	celle[c] adderes til R og resultatet anbringes i R: R:=R+celle[c];	+	+		+	
AR D	$c \cdot 2^{-9}$ adderes til R og resultatet anbringes i R: R:=R+c×2⋏(-9);		+		+	c opfattes som et heltal i intervallet $-512 \leq c \leq 511$.
AR F	flydcelle[c] adderes til RF og resultatet anbringes i RF, mens M's tælledel indeholder antallet af skift for resultatet: RF:=RF+flydcelle[c]; Mtæl:=antal skift;	+			+	D-varianten må <u>ikke</u> anvendes. Ved overløb hoppes til celle 0 med adressen på næste ordre i Radr. (Hvis den aktuelle ordre er en venstre halvordsordre, hoppes til højre halvdel af celle 0; hvis den aktuelle ordre er en højre halvordsordre, hoppes til venstre halvdel af celle 0; i begge tilfælde er Radr = r1).
AN	Absolutværdien af celle[c] adderes til R og resultatet anbringes i R: R:=R+abs(celle[c]);	+	+		+	Operanden −1 behandles korrekt, d.v.s. absolutværdien af −1 er +1 (med overløb).
AN D	Absolutværdien af $c \cdot 2^{-9}$ adderes til R og resultatet anbringes i R: R:=R+abs(c×2⋏(-9));		+		+	c opfattes som et heltal i intervallet $-512 \leq c \leq 511$.
AN F	Absolutværdien af flydcelle[c] adderes til RF og resultatet anbringes i RF, mens M's tælledel indeholder antallet af skift for resultatet: RF:=RF+abs(flydcelle[c]); Mtæl:= antal skift;	+			+	D-varianten må <u>ikke</u> anvendes. Ved overløb sker det samme som ved ARF.
AC	R adderes til celle[c] og resultatet anbringes i celle c: celle[c]:=H:=celle[c]+R;	+	+	+	+	R-registeret er uændret. Vi fremhæver at ACF har samme virkning som AC, d.v.s. at der <u>ikke</u> sker en korrekt addition af de flydende tal.
AC D	Radr. adderes til ordrens eventuelle tælletal og denne sum adderes til adressetallet i Dadr: adressetal[Dadr]:=Hadr:=c+Radr;		+	+	+	

Beteg-nelse	Virkning	Registr. M.	Registr. O.	Mær-ke-oper.	Ind. af tegn	Specielle bemærkninger
SR	celle[c] subtraheres fra R og resultatet anbringes i R: R:=R-celle[c];	+	+		+	Operanden -1 behandles korrekt, d.v.s. subtraktion af -1 udføres som addition af +1.
SR D	$c \cdot 2^{-9}$ subtraheres fra R og resultatet anbringes i R: R:=R-c×2↑(-9);		+		+	c opfattes som et heltal i intervallet $-512 \leq c \leq 511$.
SR F	flydcelle[c] subtraheres fra RF og resultatet anbringes i RF, mens M's tælledel indeholder antallet af skift for resultatet: RF:=RF-flydcelle[c]; Mtæl:=antal skift;	+			+	D-varianten må <u>ikke</u> anvendes. Ved overløb hoppes til celle 0 som ved ARF.
SN	Absolutværdien af celle[c] subtraheres fra R og resultatet anbringes i R: R:=R-abs(celle[c]);	+	+		+	Operanden -1 behandles korrekt, d.v.s. absolutværdien af -1 er +1 (med overløb).
SN D	Absolutværdien af $c \cdot 2^{-9}$ subtraheres fra R og resultatet anbringes i R: R:=R-abs(c×2↑(-9));		+		+	c opfattes som et heltal i intervallet $-512 \leq c \leq 511$.
SN F	Absolutværdien af flydcelle[c] subtraheres fra RF og resultatet anbringes i RF, mens M's tælledel indeholder antallet af skift for resultatet: RF:=RF-abs(flydcelle[c]); Mtæl:= antal skift;	+			+	D-varianten må <u>ikke</u> anvendes. Ved overløb hoppes til celle 0 som ved ARF.
SC	R subtraheres fra celle[c] og resultatet anbringes i celle c: celle[c]:=H:=celle[c]-R;	+	+	+	+	R-registeret er uændret. Vi fremhæver at SCF har samme virkning som SC, d.v.s. at der <u>ikke</u> sker en korrekt subtraktion af de flydende tal.
SC D	Radr subtraheres fra adressetallet i Dadr. Ordrens eventuelle tælletal adderes til resultatet: adressetal[Dadr]:=Hadr:=c-Radr;					

Betegnelse	Virkning	Registr. M.	Registr. O.	Mærke-oper.	Ind. af tegn	Specielle bemærkninger
MK	M multipliceres med celle[c] hvorefter produktet adderes til R. Resultatet kommer i R: R:=R+M×celle[c]+2↑(-40);	+	+		+	Multiplikationen siges at være akkumulerende. Resultatet er afrundet. Indholdet af M ændres ikke.
MK D	M multipliceres med $c \cdot 2^{-9}$ hvorefter produktet adderes til R. Resultatet kommer i R: R:=R+M×c×2↑(-9)+2↑(-40);		+		+	Som ved MK. Iøvrigt vil c blive opfattet som et heltal i intervallet $-512 \leq c \leq 511$.
MK F	RF multipliceres med flydcelle[c] og produktet kommer i RF: RF:=RF×flydcelle[c];	+			+	D-varianten må ikke benyttes. Multiplikationen er ikke akkumulerende. Ved overløb hoppes til celle 0 som ved ARF.
ML og ML F	M multipliceres med celle[c] hvorefter produktet adderes til $R \cdot 2^{-39}$. Resultatet kommer i RM: RM:=R×2↑(-39)+M×celle[c]; Mpos[0]:=0;	+	+		+	Multiplikationen udføres uden afrunding. Resultatet har 78 binaler.
ML D	M multipliceres med $c \cdot 2^{-9}$ hvorefter produktet adderes til $R \cdot 2^{-39}$. Resultatet kommer i RM: RM:=R×2↑(-39)+M×c×2↑(-9); Mpos[0]:=0;		+		+	Resultatet har 78 binaler.
MT	R multipliceres med +1 eller med −1 eftersom pos[0,c] er 0 eller 1. Resultatet i R: R:=if pos[0,c]=1 then -R else R;	+	+		+	Står der et maskintal i c multipliceres R med dettes fortegn. Står der et flydende tal i c multipliceres R med fortegnet for dettes exponent.
MT D	$c \cdot 2^{-9}$ opfattes som maskintal og R multipliceres med c's fortegn: R:= if H00=1 then -R else R;		+		+	

Betegnelse	Virkning	Registr. M.	Registr. O.	Mærke-oper.	Ind. af tegn	Specielle bemærkninger

DK	R divideres med celle[c]; resultatet anbringes i R og resten – multipliceret med 2^{39} – anbringes i M. Hvis abs(R) > abs(celle[c]) registreres der overløb:
	if 2×abs(R)<abs(celle[c]) then
	begin
	q:=entier(R/celle[c]×2↑39)×
	2↑(-39);
	M:=(R-q×celle[c])×2↑39;
	R:=q;
	if celle[c]<0 ∧ R/celle[c]×2↑39=
	entier(R/celle[c]×2↑39)
	then begin
	R:=q-2↑(-39);
	M:=celle[c] end
	end
	else overløb:= true;
		+	+		+	Bør kun benyttes såfremt 2abs(R)<abs(celle[c]). Hvis abs(R)>abs(celle[c]) kan man – ved et højreskift – få den halve kvotient i R.
						Resten vil have divisors fortegn.
						Se i øvrigt afsnit 2.5.3.

DK D	R divideres med $c \cdot 2^{-9}$; resultatet anbringes i R og resten – multipliceret med 2^{39} – anbringes i M:
	if 2×abs(R)<abs(c×2↑(-9)) then
	begin
	q:=entier(R/c×2↑48)×2↑(-39);
	M:=(R-q×c×2↑(-9))×2↑39;
	R:=q;
	if c<0∧R/c×2↑48=entier(R/c×2↑48)
	then begin
	R:=q-2↑(-39);
	M:=c×2↑(-9) end
	end
	else overløb:= true;
			+		+	Bør kun benyttes såfremt 2abs(R) < abs $(c \cdot 2^{-9})$.
						c opfattes som et heltal i intervallet $-512 \leq c \leq 511$.
						Se iøvrigt bemærkningen til DK.

DK F	RF divideres med flydcelle[c]; resultatet kommer i RF; resten går tabt: RF:=RF/flydcelle[c];	+			+	D-varianten må ikke benyttes. Ved overløb hoppes til celle 0 som ved ARF.
DL	RM divideres med celle[c]; resultatet anbringes i R og resten – multipliceret med 2^{39} – anbringes i M. Hvis abs(RM) > abs(celle[c]) registreres der overløb: ALGOL-beskrivelse som ved DK, idet R erstattes af RM undtagen i sætningerne R:=q og R:=q-2↑(-39).	+	+		+	Bør kun benyttes såfremt 2abs(RM) < abs (celle[c]). Se iøvrigt bemærkningen til DK.
DL D	RM divideres med $c \cdot 2^{-9}$, iøvrigt som DK D blot med RM i stedet for R.		+		+	Bør kun benyttes såfremt 2·abs(RM)<abs $(c \cdot 2^{-9})$. c opfattes som et heltal i intervallet $-512 \leq c \leq 511$. Se iøvrigt bemærkningen til DK.

Betegnelse	Virkning	Registr. M.	Registr. O.	Mærke-oper.	Ind. af tegn	Specielle bemærkninger

NK	R normaliseres, d.v.s. skiftes indtil der fremkommer et tal, N, der opfylder følgende: $-1 \leqq N < -\frac{1}{2}$ eller $N = 0$ eller $\frac{1}{2} \leqq N < 1$. Normaliseringsexponenten (antallet af skift med modsat fortegn) anbringes som adressetal i celle c:
	exponent:=0;
	if R $\neq$ 0 then
	begin if R00 $\neq$ Rpos[0] then
	  begin for i:=39 step −1 until 1
	    do Rpos[i]:=Rpos[i−1];
	  Rpos[0]:=R00;
	  exponent:=1 end
	 else
	 E: if Rpos[0]=Rpos[1] then
	  begin
	    R:=2×R;
	    exponent:=exponent−1;
	    go to E end
	end;
	adressetal[c]:=exponent;
				+	+	Normaliseringsexponenten er kun positiv (+1), såfremt der er overløb i R inden operationens udførelse. Der foretages ingen afrunding ved dette højre skift.

NK D	Som NK med den ændring, at normaliseringsexponenten anbringes i den celle, hvis adresse er D-adressen.			+	+	

NK F	Først normaliseres R, hvorefter det normaliserede tal forskydes 10 pladser til højre. Normaliseringsexponenten minus 1 adderes til c og denne sum anbringes i Madr.(se bemærkningen).
	ALGOL-beskrivelse: Som ved NK idet den sidste sætning erstattes af:
	for i:=39 step −1 until 10 do
	  Rpos[i]:=Rpos[i−10];
	for i:=1 step 1 until 9 do
	  Rpos[i]:=Rpos[0];
	Madr:=c + exponent − 1;
					+	D-modifikationen må ikke anvendes.
						Ordren NK F 0 vil bevirke, at et maskintal i R omdannes til et flydende tal i RF.

NL	RM normaliseres, idet M_0 ikke indgår i skiftene: Mpos[0]:=0. I øvrigt som ved NK.			+	+	
NL D	Som NL, dog anbringes normaliseringsexponenten i den celle, hvis adresse er D-adressen.			+	+	

Betegnelse	Virkning	Registr. M.	Registr. O.	Mærke-oper.	Ind. af tegn	Specielle bemærkninger

TK	R forskydes c pladser til venstre hvis c > 0 og −c pladser til højre hvis c < 0:
	if c ≥ 0 then
	for k:=1 step 1 until c do
	begin R00:=Rpos[0];
	  for i:=0 step 1 until 38 do
	    Rpos[i]:=Rpos[i+1];
	  Rpos[39]:=0
	  end
	else
	begin q:=Rpos[40+c];
	  for k:=1 step 1 until −c do
	  begin
	    for i:=39 step −1 until 1 do
	      Rpos[i]:=Rpos[i−1];
	    Rpos[0]:=R00
	  end;
	  R:=R+q×2↑(−39) end;
			+		+	Såfremt $R \cdot 2^c$ ligger i intervallet $-2 \leq x < 2$ vil det være dette tal, der står i R efter operationen.
						Ved højreskift foretages afrunding til sidst.
						Overløb registreres til sidst, efter selve skifteoperationen.

TK D	R forskydes så mange pladser til venstre (eller til højre) som Dadr angiver. Iøvrigt som TK.		+		+	Normalt uden interesse.
TK F	R forskydes (c+Madr+1) pladser til venstre hvis (c+Madr+1) ≥ 0, og −(c+Madr+1) pladser til højre hvis (c+Madr+1) < 0.		+		+	D-modifikation må ikke anvendes. Ordren TK F 10 vil bevirke, at et flydende tal i RF omdannes til maskintal i R (hvis tallet ligger mellem −2 og 2). Overløb registreres til sidst, efter selve skifteoperationen.
TL	RM forskydes, idet dog M_0 ikke indgår: Mpos[0]:=0. I øvrigt som TK, men uden afrunding ved højre skift.		+		+	
TL D	Som TK D, men med RM i stedet for R.		+		+	Normalt uden interesse.

Betegnelse	Virkning	Registr. M.	Registr. O.	Mærke-oper.	Ind. af tegn	Specielle bemærkninger

CK	R forskydes cyklisk c pladser til venstre hvis c > 0 og -c pladser til højre hvis c < 0:
	R00:=0;
	if c ≥ 0 then
	for k:=1 step 1 until c do
	begin a:=Rpos[0];
	for i:=0 step 1 until 38 do
	Rpos[i]:=Rpos[i+1];
	Rpos[39]:=a end
	else
	for k:=1 step 1 until -c do
	begin a:=Rpos[39];
	for i:=39 step -1 until 1 do
	Rpos[i]:=Rpos[i-1];
	Rpos[0]:=a
	end;
			+			Pos.00 i R-registeret indgår ikke i skiftet, men nulstilles.

CK D	R forskydes cyklisk så mange pladser til venstre (eller til højre) som Dadr angiver. Iøvrigt som CK.		+			Normalt uden interesse.
CL	RM forskydes cyklisk c pladser til venstre hvis c > 0 og -c pladser til højre hvis c < 0. M_0 indgår ikke: Mpos[0]:=0. Iøvrigt som CK.		+			Pos.00 i R-registeret og pos.0 i M-registeret indgår ikke i skiftet, men nulstilles.
CL D	Som CK D, men med RM i stedet for R.		+			Normalt uden interesse.

Betegnelse	Virkning	Registr. M.	Registr. O.	Mærke-oper.	Ind. af tegn	Specielle bemærkninger
AB	celle[c] adderes logisk til R og summen anbringes i R: for i:=0 step 1 until 39 do Rpos[i]:=Rpos[i] ∨ pos[i,c]; R00:=R00 ∨ pos[0,c];	+			+	X-varianten har en speciel virkning i denne og den følgende ordre og beskrives særskilt.
AB D	c adderes logisk til R og summen anbringes i R: for i:=0 step 1 until 9 do Rpos[i]:=Rpos[i] ∨ Hpos[i]; R00:=R00 ∨ H00;				+	
AB X	Efter udførelsen af AB anbringes R i M-registeret og ¬R i R-registeret: for i:=0 step 1 until 39 do begin Mpos[i]:=Rpos[i] ∨ pos[i,c]; Rpos[i]:=¬Mpos[i] end; R00:=¬(R00 ∨ pos[0,c]);	+			+	

AB DX	Efter udførelsen af AB D anbringes R i M-registeret og ¬R i R-registeret:
	for i:=0 step 1 until 9 do
	Rpos[i]:=Rpos[i] ∨ Hpos[i];
	for i:=0 step 1 until 39 do
	begin Mpos[i]:=Rpos[i];
	Rpos[i]:=¬ Rpos[i] end;
	R00:=¬(R00 ∨ H00);
					+	

Betegnelse	Virkning	Registr. M.	Registr. O.	Mærke-oper.	Ind. af tegn	Specielle bemærkninger
MB	celle[c] multipliceres logisk med R og produktet anbringes i R: for i:=0 step 1 until 39 do Rpos[i]:=Rpos[i] ∧ pos[i,c]; R00:=R00 ∧ pos[0,c];	+			+	X-varianten har en speciel virkning i denne og den følgende ordre og beskrives særskilt.

MB D	c multipliceres logisk med R og produktet anbringes i R:
	for i:=0 step 1 until 9 do
	Rpos[i]:=Rpos[i] ∧ Hpos[i];
	R00:=R00 ∧ H00;
	for i:=10 step 1 until 39 do
	Rpos[i]:=0;
					+	

MB X	Efter udførelse af MB anbringes R i M-registeret og nonækvivalens af opr. R og celle c anbringes i R-registeret:
	for i:=0 step 1 until 39 do
	begin Mpos[i]:=Rpos[i] ∧ pos[i,c];
	Rpos[i]:=¬(Rpos[i] ≡ pos[i,c])
	end;
	R00:=¬(R00 = pos[0,c]);
		+			+	

MB XD	Efter udførelse af MB D anbringes R i M-registeret og nonækvivalens af opr. R og c anbringes i R-registeret:
	for i:=0 step 1 until 9 do
	begin Mpos[i]:=Rpos[i] ∧ Hpos[i];
	Rpos[i]:=¬(Rpos[i] ≡ Hpos[i])
	end;
	R00:=¬(R00 ≡ H00);
	for i:=10 step 1 until 39 do
	Mpos[i]:=0;
					+	Pos.10-39 i R er uændrede efter operationen.

Betegnelse	Virkning	Registr. M.	Registr. O.	Mærke-oper.	Ind. af tegn	Specielle bemærkninger
PM	Indholdet af celle[c] anbringes i M, d.v.s. M:=celle[c];	+			+	
PM D	$c \cdot 2^{-9}$ anbringes i M_{0-9} mens resten af M nulstilles, d.v.s. M:=c×2↑(-9);				+	
PI	Den modificerede adresse placeres i indikatoren (på nær KA og KB) med ordrens tælletal som maske, idet de positioner i indikatoren, der er angivet ved ettaller i ordrens tælletal (binært), *ikke* ændres: *for* i:=0 *step* 1 *until* 9 *do* indik[i]:=(indik[i] ∧ Ftæl[i]) ∨ (Hpos[i] ∧ ¬Ftæl[i]);					Ordren er *statisk*. Står PI som halvordsordre, opfattes tælletallet som nul, d.v.s. at hele den modificerede adresse overføres til indikatoren.
PI D	Som ovenfor idet D-adressen træder i stedet for den modificerede adresse.					Ordren er *statisk*, Normalt uden interesse.
PP	c placeres i p-registeret: p:=c;					
PP D	D-adressen anbringes i p-registeret: p:=Dadr;					Normalt uden interesse.
PS	c placeres i sekvensregisteret: s1:=c;					
PS D	D-adressen anbringes i sekvensregisteret: s1:=Dadr;					Normalt uden interesse.

Betegnelse	Virkning	Registr. M.	Registr. O.	Mærke-oper.	Ind. af tegn	Specielle bemærkninger
PA	Ordrens tælletal anbringes i positionerne 0-9 af celle c, mens resten af cellen er uforandret: adressetal[c]:=Ftæl;					Ordren er statisk. Hvis PA-ordren er en halvordsordre, virker den som en helordsordre med tælletal 0.
PA D	Ordrens tælletal anbringes i pos. 0-9 i den celle, D-adressen angiver: adressetal[Dadr]:=Ftæl;					Ordren er statisk.
PT	Ordrens tælletal anbringes i pos. 10-19 af celle c, mens resten af cellen er uændret: tælletal[c]:=Ftæl;					Ordren er statisk. Hvis PT-ordren er en halvordsordre, virker den som en helordsordre med tælletal 0.
PT D	Ordrens tælletal anbringes i pos. 10-19 af den celle, D-adressen angiver: tælletal[Dadr]:=Ftæl;					Ordren er statisk.

Betegnelse	Virkning	Registr. M.	Registr. O.	Mærkeoper.	Ind. af tegn	Specielle bemærkninger
GR	R lagres i celle c: celle[c]:=R;			+		R00 irrelevant. Ved en absolut mærkeoperation etstilles R_{40} og R_{41} (som ellers er uændrede).
GR D	Adressetal og tælletal i R lagres i de tilsvarende positioner i den celle, der angives ved D-adressen, mens resten af cellen er uændret: adressetal[Dadr]:=Radr; tælletal[Dadr]:=Rtæl;			+		Ved en absolut mærkeoperation etstilles R_{40} og R_{41} (som ellers er uændrede). R00 er irrelevant.
GR F	RF lagres i celle c: flydcelle[c]:=RF;			+		R_{00-9} er irrelevant. Ved en absolut mærkeoperation etstilles R_{40} og R_{41} (som ellers er uændrede). D-varianten må <u>ikke</u> benyttes.
GM	M lagres i celle c: celle[c]:=M;			+		Ved en absolut mærkeoperation etstilles R_{40} og R_{41} (som ellers er uændrede).
GM D	Adressetal og tælletal i M lagres i de tilsvarende positioner i den celle, der angives ved D-adressen, mens resten af cellen er uændret: adressetal[Dadr]:=Madr; tælletal[Dadr]:=Mtæl;			+		Ved en absolut mærkeoperation etstilles R_{40} og R_{41} (som ellers er uændrede).

Beteg-nelse	Virkning	Registr. M.	Registr. O.	Mær-ke-oper.	Ind. af tegn	Specielle bemærkninger
GA	Adressetallet i R lagres i pos.0-9 i celle c, mens resten af cellen er uændret: adressetal[c]:=Radr;			+		R00 er irrelevant. Ved en absolut mærkeoperation etstilles R_{40} og R_{41} (som ellers er uændrede).
GA D	Adressetallet i R lagres i pos.0-9 i den celle, der angives ved D-adressen, mens resten af denne celle er uændret: adressetal[Dadr]:=Radr;			+		R00 er irrelevant. Ved en absolut mærkeoperation etstilles R_{40} og R_{41} (som ellers er uændrede).
GT	Tælletallet i R lagres i pos.10-19 i celle c, mens resten af cellen er uændret: tælletal[c]:=Rtæl;			+		Ved en absolut mærkeoperation etstil- R_{40} og R_{41} (som ellers er uændrede).
GT D	Tælletallet i R lagres i pos.10-19 i den celle, der angives ved D-adressen, mens resten af denne celle er uændret: tælletal[Dadr]:=Rtæl;			+		Ved en absolut mærkeoperation etstilles R_{40} og R_{41} (som ellers er uændrede).

Betegnelse	Virkning	Registr. M.	Registr. O.	Mærkeoper.	Ind. af tegn	Specielle bemærkninger
GP	p-registerets indhold lagres i pos. 0-9 i celle c, mens resten af cellen er uforandret: adressetal[c]:=p;			+		Ved en absolut mærkeoperation etstilles R_{40} og R_{41} (som ellers er uændrede).
GP D	p-registerets indhold lagres i pos. 0-9 i den celle, der angives ved D-adressen, mens resten af cellen er uforandret: adressetal[Dadr]:=p;			+		Ved en absolut mærkeoperation etstilles R_{40} og R_{41} (som ellers er uændrede).
GS	Sekvensregisterets indhold lagres i pos. 0-9 i celle c, mens resten af cellen er uforandret: adressetal[c]:=s1;			+		Ved en absolut mærkeoperation etstilles R_{40} og R_{41} (som ellers er uændrede).
GS D	Sekvensregisterets indhold lagres i pos. 0-9 i den celle, der angives ved D-adressen, mens resten af cellen er uændret: adressetal[Dadr]:=s1;			+		Ved en absolut mærkeoperation etstilles R_{40} og R_{41} (som ellers er uændrede).
GI	Indikatorens indhold (på nær KA og KB) lagres i pos. 0-9 i celle c, mens resten af cellen er uændret: adressetal[c]:=indikator;			+		Ved en absolut mærkeoperation etstilles R_{40} og R_{41} (som ellers er uændrede).
GI D	Indikatorens indhold (på nær KA og KB) lagres i pos. 0-9 i den celle, der angives ved D-adressen, mens resten af denne celle er uændret: adressetal[Dadr]:=indikator;			+		Ved en absolut mærkeoperation etstilles R_{40} og R_{41} (som ellers er uændrede).
GK	Indholdet af by-registeret samt kanalregisteret lagres i adresse- og tællepositionerne i celle c, mens resten af cellen er uændret: adressetal[c]:=by; tælletal[c]:=tk;			+		Ved en absolut mærkeoperation etstilles R_{40} og R_{41} (som ellers er uændrede).
GK D	Som ovenfor, idet lagringen sker i den celle, der angives ved D-adressen: adressetal[Dadr]:=by; tælletal[Dadr]:=tk;			+		Ved en absolut mærkeoperation etstilles R_{40} og R_{41} (som ellers er uændrede).

Betegnelse	Virkning	Registr. M.	Registr. O.	Mærke-oper.	Ind. af tegn	Specielle bemærkninger

IS	IS-ordren har kun virkning, hvis s indgår i adresseberegningen for den umiddelbart følgende ordre og kun første gang s indgår. Virkningen vil da være, at den resulterende adresse i IS-ordren anvendes som s-værdi ved adresseberegningen i den følgende ordre uanset sekvensregisterets øjeblikkelige indhold (og dette er uændret efter operationen):
	s2:=c;
	<u>comment</u>: Denne sætning erstatter sætningen
	  s2:=s1
	i adresseberegningen for den umiddelbart følgende ordre;
						Den umiddelbart følgende ordre bør (af hensyn til overvågningsprogrammer) ikke være en hopordre eller en V-variant.
						  Indicering er virkningsløs. X- og V-variant i IS-ordren er virkningsløse.

IS D	Ordrens D-adresse anvendes som s-værdi i den umiddelbart følgende ordre. I øvrigt som IS-ordren.					Se under IS-ordren.
NS	Ordrens resulterende adresse med modsat fortegn anvendes som s-værdi i den umiddelbart følgende ordre. I øvrigt som IS-ordren: s2:=-c; <u>comment</u>: Denne sætning erstatter sætningen s2:=s1 i adresseberegningen for den umiddelbart følgende ordre;					Se under IS-ordren.
NS D	Ordrens D-adresse med modsat fortegn anvendes som s-værdi i den umiddelbart følgende ordre. I øvrigt som NS-ordren.					Se under IS-ordren.

Betegnelse	Virkning	Registr. M.	Registr. O.	Mærkeoper.	Ind. af tegn	Specielle bemærkninger
IT	Ordrens resulterende adresse anvendes som tælletal i den umiddelbart følgende ordre uanset denne ordres eventuelle tælletal (som er uændret efter operationen): Ftæl:=c; <u>comment</u>: Denne sætning erstatter sætningerne Ftæl:=0 og Ftæl:=tælletal[r2] i adresseberegningen for en umiddelbart følgende ordre;					Den umiddelbart følgende ordre bør (af hensyn til overvågningsprogrammer) ikke være en hopordre eller en V-variant. X- og V-variant i IT-ordren er virkningsløse. Indicering er virkningsløs. Hvis IT-ordren er en venstre halvordsordre, bør den tilhørende højre halvordsordre være statisk eller parentesmærket; i modsat fald ændres IT-ordrens adressetal.
IT D	Ordrens D-adresse anvendes som tælletal i den umiddelbart følgende ordre. I øvrigt som IT-ordren.					Se under IT-ordren.
NT	Ordrens resulterende adresse med modsat fortegn anvendes som tælletal i den umiddelbart følgende ordre. I øvrigt som IT-ordren: Ftæl:=-c; <u>comment</u>: Denne sætning erstatter sætningerne Ftæl:=0 og Ftæl:=tælletal[r2] i adresseberegningen for den umiddelbart følgende ordre;					Se under IT-ordren.
NT D	Ordrens D-adresse med modsat fortegn anvendes som tælletal i den umiddelbart følgende ordre. I øvrigt som NT-ordren.					Se under IT-ordren.

Betegnelse	Virkning	Registr. M.	Registr. O.	Mærke-oper.	Ind. af tegn	Specielle bemærkninger

BT	Idet ordrens resulterende adresse c og dens tælletal t begge opfattes som tal i intervallet −512 ≦ x ≦ 511, udføres den ordre (eller de ordrer), der findes til og med den følgende højre halvcelle, kun såfremt c > t. Ellers overspringes denne ordre (eller disse ordrer):
	if c ≦ Ftæl then
	begin højrehalvord:= false;
	r1:=r1+1 end;
						For en halvordsordre sættes tælletallet t=0.
						Hvis BT-ordren er V-mærket, gælder den beskrevne virkning ordren (ordrerne) i celle r+2, idet cellen umiddelbart efter BT-ordren da altid overspringes.

BT D	Ordren har en betingende virkning nøjagtig som BT, blot sammenlignes D-adressen med tælletallet:
	if Dadr ≦ Ftæl then
	begin højrehalvord:= false;
	r1:=r1+1 end;
						V-variant: Som ved BT.

BS	Idet ordrens modificerede adresse m og dens tælletal t begge opfattes som tal i intervallet −512 ≦ x ≦ 511, udføres den ordre (eller de ordrer), der findes til og med den følgende højre halvcelle, kun såfremt m > t. Ellers overspringes denne ordre (eller disse ordrer):
	if c ≦ Ftæl then
	begin højrehalvord:= false;
	r1:=r1+1 end;
						Ordren er statisk.
						For en halvordsordre sættes tælletallet t=0.
						V-variant: Som ved BT.

BS D	Ordren har en betingende virkning nøjagtig som BS, blot sammenlignes D-adressen med tælletallet:
	if Dadr ≦ Ftæl then
	begin højrehalvord:= false;
	r1:=r1+1 end;
						Ordren er statisk.
						V-variant: Som ved BT.

Betegnelse	Virkning	Registr. M.	Registr. O.	Mærke-oper.	Ind. af tegn	Specielle bemærkninger
CA	Den ordre (eller de ordrer), der findes til og med den følgende højre halvcelle, udføres kun, når den resulterende adresse er lig med adressetallet i R. Ellers overspringes denne ordre (eller disse ordrer): __if__ c ≠ Radr __then__ __begin__ højrehalvord:= __false__; r1:=r1+1 __end__;					c beregnes med 11 bits i pos.00-9 af H-registeret, og derpå sammenlignes disse 11 bits med de tilsvarende bits i R. V-variant: Som ved BT.
CA D	Samme virkning som CA, idet dog D-adressen sammenlignes med adressetallet i R: __if__ Dadr ≠ Radr __then__ __begin__ højrehalvord:= __false__; r1:=r1+1 __end__;					V-variant: Som ved BT.
NC	Den ordre (eller de ordrer), der findes til og med den følgende højre halvcelle, udføres kun når den resulterende adresse er forskellig fra adressetallet i R. Ellers overspringes denne ordre (eller disse ordrer): __if__ c = Radr __then__ __begin__ højrehalvord:= __false__; r1:=r1+1 __end__;					c beregnes som ved CA-ordren. V-variant: Som ved BT.
NC D	Samme virkning som NC, idet dog D-adressen sammenlignes med adressetallet i R: __if__ Dadr =Radr __then__ __begin__ højrehalvord:= __false__; r1:=r1+1 __end__;					V-variant: Som ved BT.

Betegnelse	Virkning	Registr. M.	Registr. O.	Mærke-oper.	Ind. af tegn	Specielle bemærkninger

CM	Den ordre (eller de ordrer), der findes til og med den følgende højre halvcelle, udføres kun når indholdet i celle c <u>ikke</u> er identisk med R i de positioner, hvor de tilsvarende positioner i M indeholder ettaller:
	
	coinc:= <u>true</u>;
	<u>for</u> i:=0 <u>step</u> 1 <u>until</u> 39 <u>do</u>
	  coinc:=coinc ∧
	    (pos[i,c] ∧ Mpos[i]
	    ≡ Rpos[i] ∧ Mpos[i]);
	<u>if</u> ¬coinc <u>then</u>
	<u>begin</u> højrehalvord:= <u>false</u>;
	      r1:=r1+1 <u>end</u>;
		+				R00 indgår ikke i coincidens-undersøgelsen.
						V-variant: Som ved BT.

CM D	Samme virkning som CM, idet dog selve den resulterende adresse c efterfulgt af 30 nuller sammenlignes med R (med M som maske).					R00 indgår ikke i coincidens-undersøgelsen. V-variant: Som ved BT.

Betegnelse	Virkning	Registr. M.	Registr. O.	Mærke-oper.	Ind. af tegn	Specielle bemærkninger
HV	Næste ordre hentes i celle c; er der to halvordsordrer i celle c, udføres først den venstre: højrehalvord:= <u>false</u>; r1:=c;					Er ordren en V-variant, hentes næste ordre i celle c+1.
HV D	Næste ordre hentes i den celle, der angives ved D-adressen. I øvrigt som HV-ordren.					
HH	Næste ordre hentes i celle c; er der to halvordsordrer i celle c, udføres kun den højre: højrehalvord:= <u>true</u>; r1:=c;					Er ordren en V-variant, hentes næste ordre i celle c+1.
HH D	Næste ordre hentes i den celle, der angives ved D-adressen. I øvrigt som HH-ordren.					

Betegnelse	Virkning	Registr. M.	Registr. O.	Mærke-oper.	Ind. af tegn	Specielle bemærkninger

HS	Næste ordre hentes i den celle, der angives ved den modificerede adresse c; indeholder denne celle to halvordsordrer, udføres først den venstre. HS-ordrens adresse lagres i sekvensregisteret, og hvis det er en helordsordre, lagres i dennes tællepositioner det tidligere indhold i sekvensregisteret:
	`if ¬ halvord then`
	`  tælletal[r]:=s1;`
	`s1:=r;`
	`højrehalvord:= false;`
	`r1:=c;`
						Ordren er statisk. Er ordren en V-variant, hentes næste ordre i celle c+1. Benyttes flere undersekvenser "inden i hinanden", bør HS-ordrerne altid være helordsordrer, undtagen i den yderste blok.

HS D	Samme virkning som HS. Blot hentes næste ordre i den celle, der angives ved D-adressen.					Ordren er statisk.

HR	Næste ordre hentes i celle c. Indeholder denne celle to halvordsordrer, udføres først den venstre. Desuden overføres tælletallet fra celle s til sekvensregisteret:
	`s1:=tælletal[s1];`
	`højrehalvord:= false;`
	`r1:=c;`
						Er ordren en V-variant, hentes næste ordre i celle c+1. Benyttes flere undersekvenser "inden i hinanden", må der benyttes et HR-hop svarende til hvert HS-hop, hvis sekvensmekanismen skal fungere korrekt.

HR D	Samme virkning som HR, men næste ordre hentes i den celle, der angives ved D-adressen.					
HK	Hvis der ikke er en tromletransport i gang, har ordren samme virkning som HS. Ellers har ordren ingen virkning (dog udføres altid en eventuel S-variant).					Ordren er statisk.
HK D	Hvis der ikke er en tromletransport i gang, har ordren samme virkning som HS D. Ellers har ordren ingen virkning (dog udføres altid en eventuel S-variant).					Ordren er statisk.

Betegnelse	Virkning	Registr. M.	Registr. O.	Mærke-oper.	Ind. af tegn	Specielle bemærkninger
VK	Den tromlekanal, der angives ved den resulterende adresse, tilkobles ferritlageret (men der sker ingen overførsel): tk:=c;					VK-ordren udføres først, når en eventuel igangværende tromletransport er afsluttet.
VK D	Den tromlekanal, der angives ved D-adressen, tilkobles ferritlageret (men der sker ingen overførsel).					Se under VK.
LK	Den i forvejen tilkoblede tromlekanal overføres til ferritlageret i 40 konsekutive celler, således at kanalens første ord lagres i celle c. R og M berøres ikke.					LK-ordren udføres først, når en eventuel igangværende tromletransport er afsluttet.
LK D	Samme virkning som LK, idet blot D-adressen træder i stedet for den resulterende adresse c.					Se under LK.
SK	På den i forvejen tilkoblede tromlekanal lagres indholdet af 40 konsekutive celler fra ferritlageret; den første af disse celler er celle c, der lagres som det første ord på tromlekanalen. R og M berøres ikke.					SK-ordren udføres først, når en eventuel igangværende tromletransport er afsluttet. Visse kanaler er låset for skrivning (se hæfte 2). En SK-ordre, der refererer til en af disse kanaler, negligeres.
SK D	Samme virkning som SK, idet blot D-adressen træder i stedet for den resulterende adresse c.					Se under SK.

Betegnelse	Virkning	Registr. M.	Registr. O.	Mærke-oper.	Ind. af tegn	Specielle bemærkninger
VY	Den modificerede adresse anbringes i by-registeret med ordrens tælletal som maske, idet de positioner, der er angivet ved ettaller i ordrens tælletal (binært), ikke ændres: __for__ i:=0 __step__ 1 __until__ 9 __do__ $by[i] := (by[i] \wedge Ftæl[i]) \vee (Hpos[i] \wedge \neg Ftæl[i])$;					Ordren er __statisk__. Står VY-ordren som halvordsordre, opfattes tælletallet som 0, d.v.s. at hele den modificerede adresse overføres til by-registeret.
LY	Et tegn læses fra en ydre enhed til adressepositionerne i R og i celle c således, at tegnets bitkombination (bortset fra paritetsmærket) anbringes i pos. 3-9. Pos. 0-2 (samt R00) nulstilles, og pos. 10-39 er uændrede. by-registerets pos. 7-9 bestemmer, fra hvilken ydre enhed der indlæses: by_{7-9} = 000: strimmellæser by_{7-9} = 001: skrivemaskine De øvrige kombinationer er ikke tilsluttet endnu (juni 1962).					Ved indlæsning fra skrivemaskine overføres tegnet direkte fra skrivemaskinen til cellen og R. Ved indlæsning fra strimmel bevirker LY, at et tegn overføres fra bufferregisteret bl til cellen og til R, og derefter læses det næste tegn fra strimlen til bl; hvis der her er paritetsfejl, stopper GIER, og man har da mulighed for at rette i bl-registeret. I bl-registeret står paritetsmærket i position 2.
SY	De 7 bageste cifre i den resulterende adresse c (pos. 3-9) udskrives i overensstemmelse med flexowriterkoden (uden hensyn til paritetsmærke). Ved udlæsning til strimmel sættes derefter paritetsmærke på rette plads. De 7 bageste cifre i c skiftes 10 positioner til højre og adderes til indholdet i celle 1023 i pos. 0-19. by-registerets pos. 4-5 bestemmer, hvilke ydre enheder der skrives på: by[5] = 1: skrivemaskine by[4] = 1: perforator.					I celle 1023, pos. 0-19, dannes summen af de udlæste "tegn", og indholdet kan derfor bruges som checksum for udskrift. Udskriftsenhederne kan tilsluttes uafhængigt af hinanden og af indlæseenheder.

Betegnelse	Virkning	Registr. M.	Registr. O.	Mærke-oper.	Ind. af tegn	Specielle bemærkninger
QQ	Ingen grundoperation, men alle ønskede modifikationer (adresseberegning, X-variant etc.) udføres.					Da talværdien for QQ er 0, bevirker ordren QQ 0 under indlæsningen nulstilling af en celle (eller af et halvord). Adressen er irrelevant for ordrens udførelse.
ZQ	Efter at have udført hele adresseberegningen og en eventuel S-variant stopper GIER. En eventuel igangværende tromletransport afsluttes dog før stoppet. Ved tryk på NORMAL START udføres eventuel indicering, X-variant og V-variant, hvorefter GIER fortsætter med næste ordre (eller den næste igen, hvis ZQ-ordren er forsynet med et V).					Når GIER er stoppet lyser M_4-lampen som tegn på, at ZQ-ordren ikke er helt fuldført. Adressen i ZQ-ordren er irrelevant, men står i H_{0-9} efter stoppet.
XR	Indholdet af R og M ombyttes (idet R00 ikke medtages): H:=M; M:=R; <u>comment</u>: exclusive R00; R:=H;				+	Adressen i XR-ordren er irrelevant. Det fortegn, der kan indiceres, er det resulterende fortegn i R, d.v.s. fortegnet for indholdet i M <u>før</u> operationen.

Betegnelse	Virkning	Registr. M.	Registr. O.	Mærke-oper.	Ind. af tegn	Specielle bemærkninger
UD	En af de ordrer, der står i celle c, udføres med den dergældende r-værdi og (hvis s-mekanismen benyttes korrekt) med den dergældende s-værdi. Af nedenstående skema aflæses virkningerne af samspillet mellem placeringen af UD-ordren og af ordren (ordrerne) i celle c. Er ordren i celle c en satellitordre (IS, NS, IT eller NT), udføres hele satellitkæden, d.v.s. satellitordren (evt. ordrerne) samt den ordre, der påvirkes af den sidste satellitordre. Samtidig øges r tilsvarende. (Er den sidste ordre i satellitkæden en hopordre, udføres denne).					X- og V-varianter i UD-ordren er virkningsløse.
UD D	Samme virkning som UD, idet c overalt erstattes med D-adressen.					Normalt uden interesse. X- og V-varianter i UD-ordren er virkningsløse.

<table>
<tr><th rowspan="2">I celle c</th><th colspan="3">UD-ordren i celle r er et</th></tr>
<tr><th>helord</th><th>venstre halvord</th><th>højre halvord</th></tr>
<tr><td><u>To halvord</u></td><td></td><td></td><td></td></tr>
<tr><td>Ingen hopordrer</td><td>Højre halvord i celle c udføres, og derefter ordren i celle r+1.</td><td>Venstre halvord i celle c udføres, og derefter højre halvord i celle r.</td><td>Højre halvord i celle c udføres, og derefter ordren i celle r+1.</td></tr>
<tr><td>Hopordre i venstre halvord</td><td>Res. adresse af venstre halvord lagres som adressetal i celle r. Dernæst udføres venstre eller højre halvord i celle r-1 eftersom hopordren er HV (HS, HR, HK) eller HH.</td><td>Som når UD-ordren er et helord.</td><td>Som når der ingen hopordrer er i celle c.</td></tr>
<tr><td>Hopordre i højre halvord</td><td>Res. adresse af højre halvord lagres som adressetal i celle r. Dernæst udføres venstre eller højre halvord i celle r-1 eftersom hopordren er HV (HS, HR, HK) eller HH.</td><td>Som når der ingen hopordrer er i celle c.</td><td>Res. adresse af højre halvord lagres som adressetal i celle r. Dernæst udføres venstre eller højre halvord i celle r-1 eftersom hopordren er HV (HS, HR, HK) eller HH.</td></tr>
<tr><td><u>Et helord</u></td><td colspan="3"></td></tr>
<tr><td>Ikke hopordre, uden V</td><td colspan="3">Ordren i celle c udføres, og derefter ordren i celle r+1.</td></tr>
<tr><td>Ikke hopordre, med V</td><td colspan="3">Ordren i celle c udføres, og derefter ordren i celle r+2.</td></tr>
<tr><td>Hopordre</td><td colspan="3">Res. adresse af ordren i celle c lagres som adressetal i celle r. Dernæst udføres venstre eller højre halvord i celle r-1 eftersom hopordren er HV (HS, HR, HK) eller HH.</td></tr>
</table>

6. INDIKATORDELE.

På de følgende sider findes en liste over indikatordelene samt disses virkninger. I rubrikken "Specielle bemærkninger" har vi bl.a. noteret visse forhold, som man især skal være opmærksom på.

Betegnelse	Virkning	Specielle bemærkninger
IOA	Registeret[1] OA får samme indhold som overløbsregisteret har efter grundoperationens udførelse: OA:= <u>if</u> overløb <u>then</u> 1 <u>else</u> 0;	Overløb <u>registreres</u> ved de <u>aritmetiske</u> grundoperationer (heri indbefattet skifteoperationer) og disses varianter med undtagelse af F-varianten. Det vil sige, at overløbsregisteret O bliver 1-stillet, hvis der er overløb i resultatet i H-registeret, og nul-stillet ellers. Ved alle øvrige grund-operationer og varianter er O-registeret uændret.
IOB	Registeret OB får samme indhold som O-registeret har efter grundoperationens udførelse: OB:= <u>if</u> overløb <u>then</u> 1 <u>else</u> 0;	
IOC	Registrene OA og OB får begge samme indhold som O-registeret har efter grundoperationens udførelse: OA:=OB:= <u>if</u> overløb <u>then</u> 1 <u>else</u> 0;	
IZA	Registeret OA 1-stilles, hvis indholdet i R-registeret er 0 efter grundoperationens udførelse. Ellers nulstilles OA: OA:= <u>if</u> R=0 <u>then</u> 1 <u>else</u> 0;	R00 er her medregnet. Indicering af nulsituationen i R-registeret kan ske i forbindelse med enhver operation. Indiceringen sker <u>inden</u> en eventuel X-variants udførelse.
IZB	Registeret OB 1-stilles, hvis indholdet i R-registeret er 0 efter grundoperationens udførelse. Ellers nulstilles OB: OB:= <u>if</u> R=0 <u>then</u> 1 <u>else</u> 0;	
IZC	Registrene OA og OB 1-stilles begge, hvis indholdet i R-registeret er 0 efter grundoperationens udførelse. Ellers nulstilles OA og OB: OA:=OB:= <u>if</u> R=0 <u>then</u> 1 <u>else</u> 0;	

1) Betegnelsen register benyttes i dette afsnit blandt andet om en enkelt position i indikatoren, der har 12 positioner (12 flip-flops).

Betegnelse	Virkning	Specielle bemærkninger
ITA	TA-registeret etstilles, hvis indholdet i H-registeret er negativt efter grundoperationens udførelse. Ellers nulstilles TA: TA:=H00;	Ved de aritmetiske grundoperationer (samt skifteoperationer og enkelte andre) er fortegnet i H-registeret det samme som resultatets fortegn, og det er således resultatets fortegn, der indiceres. Det er nødvendigt at indicere tegnet i den ordre, hvor det "opstår", da H-registeret anvendes i hver eneste operation. Fortegnet aflæses i pos.00 og er således korrekt selvom der er opstået overløb. Bemærk iøvrigt forskellen fra indikatordelene LT og NT, hvor fortegnet undersøges i R-registeret.
ITB	TB-registeret etstilles, hvis indholdet i H-registeret er negativt efter grundoperationens udførelse. Ellers nulstilles TB: TB:=H00;	
ITC	Registrene TA og TB etstilles begge, hvis indholdet i H-registeret er negativt efter grundoperationens udførelse. Ellers nulstilles TA og TB: TA:=TB:=H00;	
IPA	PA-registeret får samme indhold som R_{40} har efter grundoperationens udførelse: PA:=Rpos[40];	Operandcellens mærkning registreres, d.v.s. overføres til $R_{40,41}$ ved følgende grundoperationer: AR, AN, AC, SR, SN, SC, MK, ML, MT, DK, DL, AB, MB, PM, CM samt disses S-, F-, X- og V-varianter, men ikke D-varianterne.
IPB	PB-registeret får samme indhold som R_{41} har efter grundoperationens udførelse: PB:=Rpos[41];	
IPC	Registrene PA og PB får samme indhold som henholdsvis R_{40} og R_{41} har efter grundoperationens udførelse: PA:=Rpos[40]; PB:=Rpos[41];	
IQA, IQB, ..., IRC	har tilsvarende virkninger for Q- og R-registrene.	
IK, IKA, IKB, IKC	Har alle samme virkning, nemlig at indholdet af p-registeret og af indikatoren ombyttes efter grundoperationens udførelse.	Indholdet i KA og KB indgår ikke i ombytningen og forbliver uændret.

Betegnelse	Virkning	Specielle bemærkninger
M	Operandcellens mærkebits nulstilles begge. R-registerets mærkebits etstilles begge: pos[40,c]:=pos[41,c]:=0; Rpos[40]:=Rpos[41]:=1;	Bruges normalt kun i forbindelse med grundoperationerne AC, SC, GR, GM, GA, GT, GI, GP, GS, GK, NK og NL samt disses varianter. I forbindelse med de øvrige grundoperationer vil operandcellens mærkebits ikke ændres, men R-registerets mærkebits bliver stadig etstillet. M, MA, MB og MC betegnes som absolutte mærkeoperationer.
MA	Operandcellens mærkebits sættes lig 10. R-registerets mærkebits etstilles begge: pos[40,c]:=1; pos[41,c]:=0; Rpos[40]:=Rpos[41]:=1;	
MB	Operandcellens mærkebits sættes lig 01. R-registerets mærkebits etstilles begge: pos[40,c]:=0; pos[41,c]:=1; Rpos[40]:=Rpos[41]:=1;	
MC	Operandcellens mærkebits sættes lig 11. R-registerets mærkebits etstilles begge: pos[40,c]:=pos[41,c]:=1; Rpos[40]:=Rpos[41]:=1;	
MOA	Pos. 40 i operandcellen får samme indhold som OA, mens pos. 41 nulstilles. R-registerets mærkebits er uændrede: pos[40,c]:=OA; pos[41,c]:=0;	De øvrige registre i indikatoren inklusive KA og KB kan på helt samme måde som OA og OB bruges ved mærkning af en celle. R-registerets mærkebits er altid uændrede. Se iøvrigt bemærkningerne under M. MOA, MOB, MOC, MTA, ..., MKC betegnes som indikatorafhængige mærkeoperationer.
MOB	Pos. 41 i operandcellen får samme indhold som OB, mens pos. 40 nulstilles. R-registerets mærkebits er uændrede: pos[40,c]:=0; pos[41,c]:=OB;	
MOC	Pos. 40-41 i operandcellen får samme indhold som henholdsvis OA og OB. R-registerets mærkebits er uændrede: pos[40,c]:=OA; pos[41,c]:=OB;	

Overløbsbetinget
Nulbetinget
Fortegnsbetinget

Betegnelse	Virkning	Specielle bemærkninger
LO (NO)	Den aktuelle ordre udføres kun, hvis indholdet i overløbsregisteret er 1 (eller 0).	Uafhængige af indikatorindholdet.
LOA (NOA)	Den aktuelle ordre udføres kun, hvis indholdet i OA er 1 (eller 0).	
LOB (NOB)	Den aktuelle ordre udføres kun, hvis indholdet i OB er 1 (eller 0).	
LOC (NOC)	Den aktuelle ordre udføres kun, hvis både OA og OB indeholder 1 (eller 0).	
LZ (NZ)	Den aktuelle ordre udføres kun, hvis indholdet i R er lig 0 (forskelligt fra 0).	R00 er her medregnet. Uafhængige af indikatorindholdet.
LZA, LZB, LZC	Samme virkning som henholdsvis LOA, LOB og LOC.	
NZA, NZB, NZC	Samme virkning som henholdsvis NOA, NOB og NOC.	
LT (NT)	Den aktuelle ordre udføres kun. hvis R00=1 (eller R00=0).	Bemærk forskellen fra fortegnsindiceringen. hvor fortegnet tages fra H00. Uafhængige af indikatorindholdet.
LTA (NTA)	Den aktuelle ordre udføres kun, hvis TA=1 (eller TA=0).	
LTB (NTB)	Den aktuelle ordre udføres kun, hvis TB=1 (eller TB=0).	
LTC (NTC)	Den aktuelle ordre udføres kun, hvis TA =TB=1 (eller TA=TB=0).	

Mærkebetinget
K-betinget

Betegnelse	Virkning	Specielle bemærkninger
LA (NA)	Den aktuelle ordre udføres kun, hvis $R_{40}=1$ (eller $R_{40}=0$).	Uafhængige af indikatorindholdet.
LB (NB)	Den aktuelle ordre udføres kun, hvis $R_{41}=1$ (eller $R_{41}=0$).	
LC (NC)	Den aktuelle ordre udføres kun, hvis $R_{40}=R_{41}=1$ (eller $R_{40}=R_{41}=0$).	
LPA (NPA)	Den aktuelle ordre udføres kun, hvis PA=1 (eller PA=0).	
LPB (NPB)	Den aktuelle ordre udføres kun, hvis PB=1 (eller PB=0).	
LPC (NPC)	Den aktuelle ordre udføres kun, hvis PA=PB=1 (eller PA=PB=0).	
LQA, NQA, ..., NRC	har tilsvarende betingende virkninger	
LKA (NKA)	Den aktuelle ordre udføres kun, hvis KA=1 (eller KA=0).	Indholdet i KA og KB kan kun ændres ved hjælp af trykknapper på manøvrebordet (se hæfte 2).
LKB (NKB)	Den aktuelle ordre udføres kun, hvis KB=1 (eller KB=0).	
LKC (NKC)	Den aktuelle ordre udføres kun, hvis KA=KB=1 (eller KA=KB=0).	

7. EKSEMPLER OG ØVELSER.

7.1 Indledning.

Dette kapitel består af en samling eksempler på de enkelte ordrers virkning samt en række eksempler på kodning af forskellige problemer. I tilknytning til eksemplerne bringes der en række øvelsesopgaver.

I afsnit 7.2 findes eksempler med ordrer uden indikatordele, og i afsnit 7.3 behandler vi fortrinsvis ordrer med indikatordele. Afsnit 7.4 indeholder en del eksempler på kodning af mindre eller større problemer samt en del opgaver. Fremstillingen er i alle afsnit lagt således til rette, at det problem, der behandles i et eksempel, først beskrives, hvorefter problemet løses ved angivelse af et program. Disse programmer indeholder normalt kun få eller slet ingen kommentarer, idet formålet med eksemplerne er at gøre læseren fortrolig med operationslisten. Kun ordren UH optræder slet ikke i dette kapitel, men den er ret speciel og vil først blive nævnt i hæfte 2.

Vi gør i øvrigt læseren opmærksom på det forhold, at de stillede problemer hyppigt vil kunne løses på mange andre måder, end det her er vist.

7.2 Ordrer uden indikatordele.

7.2.1 Eksempler på addition, subtraktion, lagring samt anvendelse af stopordrer.

Eksempel 7.1.

Maskintallene x, y, z, v er lagrede i henholdsvis celle 100, 101, 102 og 103. Det antages, at summen $s = x + y + z + v$ er et maskintal. s skal lagres i celle 104. Derefter skal maskinen stoppes.

Program:

```
[m+0]  ARS  100, AR 101
[m+1]  AR   102, AR 103
[m+2]  GR   104, ZQ   0
```

Øvelse 7.2. Hvordan skal programmet i eksempel 7.1 ændres, hvis tallene x, y, z og v er flydende tal?

Øvelse 7.3. Bliver resultatet i eksempel 7.1 korrekt, hvis $x = 0{,}75$, $y = 0{,}75$, $z = -0{,}50$ og $v = -0{,}25$? (I så fald er mellemresultatet $x + y$ ikke et maskintal.).

Øvelse 7.4. Der skal udarbejdes et program, der danner $|x| + |y| + |z| + |v|$ i celle 105. (Samme lagring som i eksempel 7.1, idet vi antager, at resultatet igen er et maskintal.)

Øvelse 7.5. Som øvelse 7.4, idet tallene nu antages at være på flydende form.

Øvelse 7.6. $x - |y| - z + |v|$ skal lagres i celle 106. (Lagring som før.)

Eksempel 7.7.

Heltallene a1, a2 og a3 tænkes anbragt med enhed i pos. 9 (d.v.s. tallene står i adressepositionerne) i henholdsvis celle 201, 202 og 203. Positionerne 10-39 i de tre celler er nulstillede.

Tallet a1 − a2 + a3 + 13 skal lagres med enhed i position 9 i celle 200.

Program:

```
[m+0]  ARS 201, SR 202
[m+1]  AR  203
[m+2]  AR   13 D
[m+3]  GR  200, ZQ   0
```

Øvelse 7.8. Hvad er betingelsen for, at man virkelig får det ønskede resultat i eksempel 7.7?

Øvelse 7.9. Lav et program, der danner $-|a1| + |a2| - |a3| - 32$. (Forudsætninger som i eksempel 7.7.)

Øvelse 7.10. Lav et program, der lagrer tallene 1, 2, 3, 4 og 5 i cellerne 901, 902, 903, 904 og 905. Tallene skal lagres med enhed i pos. 9.

Eksempel 7.11.

I celle 201-203 står der helordsordrer med adressetallene a1, a2 og a3. Ingen af ordrerne er parentesmærkede. Tallet a1 − a2 + a3 + 13 skal lagres med enhed i pos. 0 i celle 200.

Program:

```
[m+0]  ARS (201) D
[m+1]  SR  (202) D
[m+2]  AR  (203) D
[m+3]  AR    13  D
[m+4]  GR   200, ZQ 0
```

Er blot en af ordrerne i celle 201–203 parentesmærket, fungerer ovenstående program ikke korrekt (hvorfor?). Opgavens løsning vil da kræve brug af endnu ikke gennemgåede operationer.

Eksempel 7.12.

Maskintallene a og b i henholdsvis R og celle 444. Summen a + b skal dannes i celle 444:

```
[m+0]  AC 444, ZQ 0
```

Eksempel 7.13.

De flydende tal a og b står i henholdsvis RF og i celle 444. Summen skal dannes i celle 444:

```
[m+0]  ARF 444, GR 444  [Hvorfor ikke blot ACF 444, ZQ 0?]
[m+1]  ZQ    0
```

Øvelse 7.14. Tallene a og b står som i eksempel 7.12. Dan

1) a - b i celle 444,

2) b - a i celle 444.

7.2.2 Eksempler på multiplikationer og divisioner. Placering i M-registeret.

Eksempel 7.15.

Maskintallene x, y, z og v står i henholdsvis celle 300, 301, 302 og 303. Størrelsen $a = xy(z+v)$ skal udregnes og lagres i celle 304.

Program 1:

```
[m+0]  ARS 302,  AR  303
[m+1]  GR  304,  PM  304
[m+2]  MKS 301,  GR  304
[m+3]  PM  304,  MKS 300
[m+4]  GR  304,  ZQ    0
```

Ved anvendelse af "krydsvarianten" kan vi løse opgaven ved hjælp af færre ordrer.

Program 2:

```
[m+0]  ARS 302
[m+1]  AR  303      X
[m+2]  MKS 301      X
[m+3]  MKS 300,  GR 304
[m+4]  ZQ    0
```

Vi bemærker, at programmet her fylder lige så meget som det første program. Opgaven kan løses ved hjælp af et lidt kortere program:

Program 3:

```
[m+0]  ARS 302,  AR  303
[m+1]  XR    0,  MKS 301
[m+2]  XR    0,  MKS 300
[m+3]  GR  304,  ZQ    0
```

Når vi har angivet forskellige løsninger skyldes det kun, at vi har ønsket at vise nogle forskellige typer af ordrer, og ikke, at vi vil opfordre læseren til at ændre programmer for at spare et par celler. Det er en tidkrævende sport at kode med det mindst mulige antal ordrer.

Øvelse 7.16. Samme problem som i eksempel 7.15, men med flydende tal i stedet for maskintal.

Øvelse 7.17. Hvad bliver indholdet af celle 304 i eksempel 7.15 efter udførelsen af program 1, hvis S-mærkningen i [m+2] og [m+3] udelades?

Øvelse 7.18. Idet vi forudsætter samme lagring som i eksempel 7.15, skal størrelsen $a = (x-y)(v-z)$ dannes og lagres i celle 304.

Eksempel 7.19.

Lad maskintallene a og b stå i henholdsvis M og R. Ordren MK c D vil da bevirke, at $ac \times 2^{-9} + b$ kommer i R, såfremt $-512 \leqq c \leqq 511$.

Øvelse 7.20. Lad a og b i eksempel 7.19 være henholdsvis 0.3 og 0.2. Hvad bevirker ordren MK 896 D?

Eksempel 7.21.

Heltallene a, b og c står i henholdsvis celle 400, 401 og 402 med enhed i pos. 39. Det antages, at $0 \leqq ab+c < 2^{39}$. ab + c skal lagres i celle 403 med enhed i pos. 39.

Program:

```
[m+0]  PM 400,  ARS 402
[m+1]  ML 401,  GM  403
[m+2]  ZQ   0
```

Øvelse 7.22. Hvorfor virker programmet i eksempel 7.21 ikke, hvis kravet $0 \leqq ab + c < 2^{39}$ ikke er opfyldt?

Eksempel 7.23.

De flydende tal a, b og c står i henholdsvis celle 400, 401 og 402. Størrelsen ab + c skal lagres i celle 403.

Program:

```
[m+0]  ARSF 400,  MKF 401
[m+1]  ARF  402,  GRF 403
[m+2]  ZQ     0
```

Vi bemærker, at ét F i hver af cellerne [m+0] og [m+1] er tilstrækkeligt.

Eksempel 7.24.

Maskintallene a, b og c står i henholdsvis celle 12, 13 og 14. Størrelsen ac/b skal lagres i celle 11.

Program 1:

```
[m+0]  PM 12,  MKS 14
[m+1]  DK 13,  GR  11    [Krav: |a × c| < |b|]
[m+2]  ZQ  0
```

Program 2:

```
[m+0]  ARS 12,  DK  13     [Krav: |a| < |b|]
[m+1]  XR   0,  MKS 14
[m+2]  GR  11,  ZQ   0
```

Øvelse 7.25. På grund af afrundingsfejl regner maskinen ikke korrekt. Skriv et program, der regner nøjagtigere end begge de ovenstående (benyt lange operationer).

Øvelse 7.26. Samme opgave som i eksempel 7.24, men med flydende tal i stedet for maskintal.

Øvelse 7.27. Samme opgave som i eksempel 7.24, men med heltal (lagrede med enhed i pos. 39) i stedet for maskintal.

Eksempel 7.28.

Maskintallene a og b står i henholdsvis celle 100 og 101, $\text{sign}(a) \times b$ ønskes lagret i celle 102. ($\text{sign}(a) = 1$ for $a \geqq 0$ og -1 for $a < 0$).

Program:

```
[m+0]  ARS 101,  MT  100
[m+1]  GR  102,  ZQ    0
```

I de følgende øvelser antages det, at tallene x, y, z og v står i henholdsvis celle 32, 33, 34 og 35.

Øvelse 7.29. Størrelsen $(x+y)/(z+v)$ skal lagres i celle 31. (x, y, z og v maskintal.)

Øvelse 7.30. Størrelsen $\frac{x}{2} + \frac{y}{3} - \frac{|z|}{4} + \frac{|v|}{5}$ skal lagres i celle 30. (x, y, z og v maskintal.)

Øvelse 7.31. Øvelse 7.29 med flydende tal.

Øvelse 7.32. Øvelse 7.30 med flydende tal.

7.2.3 Eksempler på normaliseringer og talforskydninger.

Eksempel 7.33.

Maskintallet a, der står i celle 42, ønskes normaliseret. Normaliseringsexponenten skal gemmes i celle 43, og det normaliserede tal skal gemmes i celle 44.

Program:

[m+0] ARS 42, NK 43

[m+1] GR 44, ZQ 0

Eksempel 7.34.

Maskintallet x, der står i celle 1000, skal omdannes til flydende tal og igen anbringes i celle 1000.

Program:

[m+0] ARS 1000

[m+1] NKF 0, GRF 1000

[m+2] ZQ 0

Øvelse 7.35. Hvorfor kan ordrerne i eksempel 7.34 ikke pakkes sammen således, at de kun fylder to helceller?

Øvelse 7.36. Hvilken virkning har det for programmet i eksempel 7.34, at 1) det første F i celle [m+1] udelades? 2) det andet F udelades? 3) både det første og det andet F udelades?

Eksempel 7.37.

Maskintallet a står i celle 702. Det antages, at $2^3 \times a$ ligeledes er et maskintal. Dette tal skal anbringes i celle 701.

Program:

[m+0] ARS 702, TK 3

[m+1] GR 701, ZQ 0

Eksempel 7.38.

Det flydende tal b, der står i celle 98, antages at ligge i intervallet $-1 \leqq b < 1$. Tallet skal omdannes til maskintal og lagres i celle 99.

Program:

[m+0] ARFS 98, TKF 10

[m+1] GR 99, ZQ 0

Øvelse 7.39. Maskintallene a og b står i henholdsvis celle 100 og 101. Kvotienten $\frac{a}{b}$ skal udregnes på formen $c \times 2^d$, hvor c er et normaliseret tal og d et heltal. c og d skal lagres i henholdsvis celle 102 og 103.

Eksempler på cykliske forskydninger findes i afsnit 7.4.

7.2.4 Anvendelse af de logiske operationer.

Eksempel 7.40.

Ordren AB 27 bevirker, at indholdet af celle 27 adderes logisk til indholdet af R. Lad indholdene før operationens udførelse være

```
                0                39
celle 27    | 1 0 . . 1 0 1 0 1 1 |

R         |0| 0 0 . . 0 0 1 1 0 0 |
```

Efter ordrens udførelse får vi da

```
          00                     39
R         |1| 1 0 . . 1 0 1 1 1 1 |
```

idet der kommer ettaller i de positioner, hvor der er et ettal mindst et af stederne.

Eksempel 7.41.

Ordren MB 27 bevirker, at indholdet af celle 27 multipliceres logisk med indholdet af R. Lad indholdene i R og i celle 27 være

```
                 0                39
celle 27        |1 0 . . 1 0 1 0 1 1|

R             |1|0 0 . . 1 0 1 1 0 0|
```

Efter ordrens udførelse er da

```
             00                   39
R             |1|0 0 . . 1 0 1 0 0 0|
```

idet der kun kommer ettaller i de positioner, hvor der er ettaller begge steder.

Eksempel 7.42.

X-mærkning af ordrer med operationsdele AB eller MB har virkninger, der afviger fra den sædvanlige ombytning af R og M. Lad indholdene i R og i celle 27 være

```
                 0                39
celle 27        |1 0 . . 1 0 1 0 1 1|

R             |0|0 0 . . 0 0 1 1 0 0|
```

Ordren AB 27 X vil da bevirke, at R og M får følgende indhold:

```
             00                   39
R             |0|0 1 . . 0 1 0 0 0 0|

M               |1 0 . . 1 0 1 1 1 1|
```

Det nye indhold i R har ettaller, hvor indholdet i M har nuller og omvendt nuller, hvor indholdet i M har ettaller.

Ordren MB 27 X vil bevirke, at R og M får følgende indhold:

```
       00                     39
R     |1|1 0 . . 1 0 0 1 1 1|

M        |0 0 . . 0 0 1 0 0 0|
```

Det nye indhold i R har ettaller, hvor det oprindelige indhold i R og i celle 27 har forskellig værdi og nuller i de øvrige positioner.

Vi fremhæver, at de logiske operationer udføres på 41 positioner med R_{00-39} som den ene operand og celleindholdet med dubleret fortegn som den anden operand.

__Øvelse 7.43.__ Lad R have ettaller i pos. 00, pos. 1, pos. 3, ..., pos. 39 og nuller i pos. 0, pos. 2, pos. 4, ..., pos. 38. Lad endvidere celle 100 have ettaller i pos. 0, pos. 2, pos. 4, ..., pos. 38 og nuller i pos. 1, pos. 3, ..., pos. 39. Hvad står der i R efter udførelsen af operationerne:

1) AB 100
2) MB 100
3) AB 100 X
4) MB 100 X

__Øvelse 7.44.__ Lad R have samme indhold som i øvelse 7.43. Hvad står der i R efter udførelsen af operationerne:

1) AB 5 D
2) MB 5 D
3) AB 5 DX
4) MB 5 DX

7.2.5 Eksempler på placeringer og lagringer.

Eksempel 7.45.

Ved hjælp af ordren PM 307 D bliver tallet 307 placeret i M med enhed i pos. 9. Resten af M nulstilles.

Eksempel 7.46.

Ved hjælp af ordren PP 47 bliver tallet 47 placeret i p-registeret.

Eksempel 7.47.

Ved hjælp af ordren PS 133 bliver tallet 133 placeret i s-registeret.

Eksempel 7.48.

Ordren PA 36 +14 bevirker, at tælletallet +14 placeres med enhed i pos. 9, d.v.s. som adressetal, i celle 36.

Eksempel 7.49.

Ordren PT 236 +44 bevirker, at tælletallet +44 placeres med enhed i pos. 19, d.v.s. som tælletal, i celle 236.

Øvelse 7.50. Skriv et program, der placerer tallene 7, 9 og 13 i henholdsvis pos. 0–9 af M-registeret, p-registeret og s-registeret.

Øvelse 7.51. Skriv et program, der placerer tallene 22 og −22 med enhed i henholdsvis pos. 9 og pos. 19 i celle 7.

Eksempel 7.52.

Ordren PA r+42 D +17 tænkes at stå i celle 278. Ordren vil da ændre sig selv til PA r+17 D+17.

Eksempel 7.53.

Ordren PA (r+42) D +17, der tænkes at stå i celle 278, bevirker, at tallet 17 placeres med enhed i pos. 9, d.v.s. som adressetal, i celle 320. (Hvilken forudsætning er her benyttet?)

Øvelse 7.54. I cellerne 25, 26 og 27 står der henholdsvis PT r+1 D -48, PT (r+1) D -86 og ZQ 0. Hvad står der i de tre celler efter et gennemløb af programmet?

Eksempel 7.55.

Lad adressetal og tælletal i R være henholdsvis 7 og 20. Ordren

GR p+408 D +12

vil da ændre sig selv til GR p+7 D +20.

Øvelse 7.56. Ordren i eksempel 7.55 ændres til GR (p+408) D +12. Tallet 1012 står i p-registeret. Hvilken virkning har ordren nu?

Eksempel 7.57.

Lad adressetal og tælletal i M være henholdsvis 207 og -413. Ordren

GM s+4 D +11

vil da ændre sig selv til GM s+207 D -413.

Øvelse 7.58. Hvilken virkning har følgende program:

[m+0] PM r+7 -6

[m+1] GM (r−1) D -49

[m+2] ZQ 0

Eksempel 7.59.

Lad adressetal og tælletal i R være henholdsvis 39 og 93. Ordren

GA r+4 D -17

vil da ændre sig selv til GA r+39 D −17. Ordren GT p-411 Ḋ −373 vil ændre sig selv til GT p-411 D +93.

Øvelse 7.60. Hvilken virkning har følgende program:

[m+0] ARS r−4 +7

[m+1] GA (r+1) D

[m+2] GT (r+2) D

[m+3] PP (r+1) +32

[m+4] ZQ 4 +5

Eksempel 7.61.

Lad p-registerets indhold være 378. Ordren GP p+3 vil da bevirke, at tallet 378 lagres i pos. 0-9 i celle 381. Ordren GP p+3 D vil ændre sig selv til

GP p+378 D.

Eksempel 7.62.

Lad s-registerets indhold være 788. Ordren GS 47 vil da bevirke, at tallet 788 lagres i pos. 0-9 i celle 47. Ordren GS 47 D vil ændre sig selv til GS 788 D.

Øvelse 7.63. Hvilken virkning har følgende program:

```
[m+0]  PP 42, PS -21
[m+1]  GP s+1, GS p+7
[m+2]  GS 42 D
[m+3]  GP 19 D
[m+4]  GS (r-3) D
[m+5]  GP (r-2) D
[m+6]  ZQ 0
```

Eksempler på operationer, der vedrører indikatoren, findes i de følgende afsnit.

7.2.6 Eksempler på satellitordrer.

Eksempel 7.64.

Lad s-registeret indeholde tallet 25. Vi betragter følgende programmer:

Program 1:	Program 2:
[m+0] IS 20, ARS s+2	[m+0] IS 20, ARS 400
[m+1] AR 400, ZQ 0	[m+1] AR s+2, ZQ 0

I program 1 følger der en s-mærket ordre umiddelbart efter IS-ordren. Virkningen af program 1 vil være, at tallene i cellerne 22 og 400 adderes. I program 2 er IS-ordren virkningsløs. s-registeret forbliver uændret i begge tilfælde.

Øvelse 7.65. Hvad bliver virkningen af følgende program, når indholdet i s-registeret er 200:

```
[m+0]  IS 50, ARS s+10
[m+1]  AR s+10, ZQ 0
```

Øvelse 7.66. Hvad bliver virkningen af følgende program, der står i celle 10–12:

[10] PS 2, IS 11

[11] ARS (s+1)

[12] AR s+1, ZQ 0

Eksempel 7.67.

I celle 37 står ordren NS r+82 D +167. Ordren vil bevirke, at tallet −37 benyttes som s-værdi i ordren (den første ordre) i celle 38.

Eksempel 7.68.

Vi betragter følgende program:

[m+0] IT 4 +3

[m+1] ARS 100 +20

[m+2] ZQ 0

Den resulterènde adresse i IT-ordren er 7, der benyttes som tælletal i den følgende ordre. Programmet vil bevirke, at indholdet af celle 107 overføres til R.

Øvelse 7.69. Hvad står der i cellerne [m+0], [m+1] og [m+2] i eksempel 7.68 efter et gennemløb af programmet?

Eksempel 7.70.

I celle 888 står ordren NT s−14 D −77. Ordren vil bevirke, at tallet −888 benyttes som tælletal i ordren (den første ordre) i celle 889.

Øvelse 7.71. Hvad bliver virkningen af følgende program, der står fra celle 100 og frem:

    [100] IT r+1 D +20
    [101] IT r-1   +40
    [102] SRS r-2   +60

Eksempel 7.72. Eksempel på IT-ordrens virkning.

Operationen IT bør kun bruges i en venstre halvordsordre, hvis den højre halvordsordre er statisk [1], thi ellers er virkningen ødelæggende: Tællingen (d.v.s. adresseændringen) sker i selve IT-ordrens adresse (nemlig i helcellens pos.0-9). Skrives f.eks. ordrerne

    IT 4, AR r+17

udføres dette som AR r+21, men efter udførelsen er ordrerne ændret til

    IT 21, AR r+17

Endnu værre kan det gå, hvis IT-ordren er parentesmærket: Ordrerne

    [m+0] GR 3 X
    [m+1] IT (r-1), AR r+4

udføres som GR 3 X, og derefter AR r+7, men derefter er ordrernes udseende ændret til

    [m+0] GR 3 X
    [m+1] IT (r+7), AR r+4

---

1) At en operation er tællende får for halvordsordrer kun betydning, hvis den foregående ordre er en IT-ordre; thi ellers sættes automatisk tælledelen lig 0.

fordi tællingen i additionsordren stadig sker i denne helcelles adressepositioner 0-9 og d.v.s. i IT-ordrens adresse.

Er derimod additionsordren parentesmærket, sker tællingen på normal måde i den resulterende adressedel; f.eks. udføres

[m+0] GR 3 X

[m+1] IT (r-1), AR (r+1)

[m+2] MK 5

som GR 3 X, derefter AR 8 og endelig MK 8, idet tællingen foregår i denne sidste ordre.

Står IT-operationen i en højre halvordsordre eller i en helordsordre, påvirker den udførelsen af ordren (eller den venstre halvordsordre) i næste celle på mere "normal" måde, idet tællingen nu foregår i dennes adressepositioner 0-9, der netop indeholder ordrens adressetal.

Hvis der endelig efter IT-ordren står en <u>statisk</u> ordre, opstår der slet ingen placeringsproblem, fordi der ikke foregår nogen ændring i ordrecellen.

<u>Eksempel 7.73. Eksempel på IS-ordrens virkning.</u>

Som fremhævet i operationslisten har IS-ordren <u>kun</u> virkning på den umiddelbart følgende ordres adressedel, dog også på den resulterende adressedel i en parenteskæde. For eksempel udføres ordrerne

[m+0] IS 13, AR (r+1)

[m+1] ZQ s-1

som AR 12 uafhængigt af s-registerets indhold. Derimod virker IS-ordren kun ved den første s-mærkning i en parenteskæde.

Således vil ordrerne

[14] SRS s-2, AR r+25

[15] IS 13, AR (s+1)

bevirke, at den effektive adresse i den sidste ordre beregnes ud fra den første adressedel i celle 14, altså s-2, og her indsættes for s tælledelen i celle 13, idet den sidste ordres adressedel med både s og parentes starter hele sekvensmekanismen (jfr. adresseberegningen side 66).

7.2.7 Betingende ordrer.

Eksempel 7.74.

Ordren BT p+37 +19 bevirker, at den ordre eller de ordrer, der findes til og med den følgende højre halvcelle, kun udføres såfremt p har et sådant indhold, at $p + 37 + 19 > 19$, idet $p + 37 + 19$ opfattes som et tal i intervallet $-512 \leqq x \leqq 511$. Desuden ændres BT-ordren til BT p+58 +19.

Øvelse 7.75. Undersøg virkningen af ordren i eksempel 7.74 for de forskellige mulige værdier af p.

Eksempel 7.76.

Ordren BS p+37.+19 bevirker, at den ordre (eller de ordrer), der findes til og med den følgende højre halvcelle, kun udføres såfremt p har en sådan værdi, at $p + 37 > 19$, idet p+37 opfattes som et tal i intervallet $-512 \leqq x \leqq 511$.

Øvelse 7.77. Undersøg virkningen af ordren i eksempel 7.76 for de forskellige mulige værdier af p.

Eksempel 7.78.

Ordren BS s−37 D +50 har kun virkning, når den er placeret i en celle, hvis adresse er større end 50.

Øvelse 7.79. Ordren i eksempel 7.78 ændres ved, at der tilføjes en parentes omkring s−37. Hvornår har ordren nu virkning?

Eksempel 7.80.

Ordren CA 42 +28 bevirker, at den ordre (eller de ordrer), der findes til og med den følgende højre halvcelle, kun udføres når adressetallet i R er 70. Desuden ændres CA-ordren til CA 70 +28.

Øvelse 7.81. Lad adressetallet i R være 70. Hvornår har ordren CA 42 D +28 virkning?

Eksempel 7.82.

Ordren NC r−19 +39 har kun virkning, når adressetallet i R netop er r+20. (Dog ændres adressetallet i NC-ordren under alle omstændigheder.)

Øvelse 7.83. Hvilken virkning har følgende program:

```
[m+0]  ARS r+1 V
[m+1]  ARS (s+37) D +114
[m+2]  CA  −19 X +56
[m+3]  MKS 37 D
[m+4]  ZQ  0
```

Øvelse 7.84. Lad adressetallet i R være 403. Hvornår har ordren NC p+42 DVX −413 virkning?

Eksempel 7.85.

Lad der i M-registeret stå 1-taller i pos. 10-19 og nuller iøvrigt. Ordren

CM 708

bevirker da, at indholdet i pos. 10-19 (tælletallet) i celle 708 sammenlignes med indholdet i de tilsvarende positioner i R-registeret. Såfremt der er overensstemmelse, overspringes den ordre (eller de ordrer), der findes til og med den følgende højre halvcelle.

7.2.8 Eksempler på hopordrer.

Eksempel 7.86.

Kodestumpen

```
[m+0]  AR 100,  GR 100
[m+1]  HV r-1
```

vil bevirke, at både additionsordren og lagringsordren gentages og gentages indtil GIER stoppes manuelt. Kodestumpen

```
[m+0]  AR 100,  GR 100
[m+1]  HH r-1
```

vil derimod bevirke, at kun lagringsordren gentages og gentages indtil GIER stoppes.

En effektiv stop-lås kan laves ved hjælp af de to ordrer

ZQ 0, HV r

hvor GIER vil stoppe igen så snart man trykker på NORMAL START.

Eksempel 7.87.

Skal fire maskintal i celle 100–103 adderes og summen lagres i celle 104, kan dette gøres med følgende program (som forudsætter at R er nulstillet i forvejen):

```
→ [m+0]  AR 99 +1
  [m+1]  BT  3 -1   [næste ordre udføres kun de første 3 gange]
└ [m+2]  HV r-2     [hop tilbage til additionsordren]
  [m+3]  GR 104, ZQ 0
```

I dette tilfælde, hvor den celle, hvortil der hoppes, indeholder en helordsordre, kunne vi lige så godt bruge en HH-ordre: Programmet

```
→ [m+0]  AR 99  +1
  [m+1]  BT  3  -1
└ [m+2]  HH r-2
  [m+3]  GR 104, ZQ 0
```

vil have samme virkning. (Se iøvrigt eksempel 122 ff der viser mange andre måder at kode en sådan beregning på.)

Eksempel 7.88.

Lad ordren HS r+12 stå i celle 440 og lad indholdet i s-registeret være 67. Ordren bevirker da, at næste ordre hentes i celle 452. Desuden gemmes indholdet af s-registeret, d.v.s. 67, i ordrens tælledel (dette sker kun når HS-ordren er en helordsordre). Endelig sættes adressen på HS-ordren, der 440, i s-registeret.

Efter gennemløbet af ordren står der således HS r+12 +67 i celle 440 og s-registeret indeholder som nævnt tallet 440.

Eksempel 7.89.

Vi betragter ordren HR s+1 i tilknytning til det foregående eksempel. Da s-registeret indeholder tallet 440, bliver den resulterende adresse 441. Ordren bevirker da, at næste ordre hentes i celle 441. Desuden undersøges det, om celle 440 indeholder en helordsordre. Når dette er tilfældet, overføres tælletallet i denne, her 67, til s-registeret.

Øvelse 7.90. Hvad bevirker følgende program, der tænkes lagret fra celle 20 og frem, når s-registeret indeholder tallet 2:

```
[20]  HS r+3
[21]  HS r+2
[22]  HR r+2
[23]  HR r-3 +1
[24]  ZQ   0
```

Eksempel 7.91.

En HK-ordre har samme virkning som en tilsvarende HS-ordre, såfremt der ikke er en tromletransport i gang. Ellers er HK-ordren uden virkning. (Dog vil en HKS-ordre altid slette R-registeret.)

Øvelse 7.92. Fra en ordre i celle 42 ønsker man, at GIER skal hoppe til celle 100; i denne celle står der en helordsordre uden parentesmærkning. Der er ingen tromletransport i gang.

Hvor mange af følgende ordrer vil kunne benyttes i celle 42?

HV 100	HV 100 D	HV (100) D
HH 100	HH 100 D	HH (100) D
HS 100	HS 100 D	HS (100) D
HR 100	HR 100 D	HR (100) D
HK 100	HK 100 D	HK (100) D

7.2.9 Eksempler på tromleordrer og tilkobling af ydre enheder.

Eksempel 7.93.

Vi betragter følgende program:

```
[m+0] VK 17,  LK 440
[m+1] VK 18,  LK 480
[m+2] VK 19,  SK 440
[m+3] VK 20,  SK 480
[m+4] ZQ  0
```

Virkningen af programmet vil være, at indholdet af kanal 17 (d.v.s. af de 40 celler på kanalen) overføres til kanal 19, og at indholdet af kanal 18 overføres til kanal 20.

Eksempel 7.94.

Skal summen af det første element på kanal 200 og det første element på kanal 201 lagres i celle 1015 (altsammen flydende tal), kan man kode således:

```
[m+0] VK  200, LK   974 [kanal 200 læses til celle 974-1013]
[m+1] VK  201, LK   975 [kanal 201 læses til celle 975-1014]
[m+2] VK    0, ARSF 974
[m+3] ARF 975, GR  1015 [addition og lagring]
[m+4] ZQ    0
```

Her sikrer ordren VK 0 at kanaloverførslen er afsluttet før behandlingen (additionen) af elementerne påbegyndes, og dette er nødvendigt.

Øvelse 7.95. Tallet a står i celle 12 på kanal 42 og tallet b i celle 27 på kanal 45. Tallet c = $|a| - |b|$ skal lagres i celle 5 på kanal 71. (Tallene a og b antages at være maskintal. Beregningen af c må ikke påbegyndes, medens der foregår tromletransport.)

Eksempel 7.96.

Inden indlæsning til maskinen eller udlæsning fra maskinen kan finde sted, må den (eller de) ydre enheder, man ønsker at benytte, være tilsluttet GIER. Der er følgende otte muligheder for tilslutning af apparatur (+ betyder tilsluttet, ellers ikke tilsluttet):

Indlæsning		Udlæsning		Kan gøres med ordren
Strimmel-læser	Skrive-maskine	Per-forator	Skrive-maskine	
+				VY 0
	+			VY 1
+			+	VY 16
	+		+	VY 17
+		+		VY 32
	+	+		VY 33
+		+	+	VY 48
	+	+	+	VY 49

De i skemaet nævnte VY ordrer fastlægger indlæse- og udlæsesituationen fuldstændigt, men derudover kan man med en VY-ordre med et tælletal som maske tilkoble et bestemt indlæse-apparatur uafhængigt af og uden at ændre udlæse situationen eller omvendt. For eksempel vil ordren

VY 0 +48

bevirke, at strimmellæseren tilkobles uden at påvirke udlæsesituationen. Ordren

VY 32 +7

tilkobler tilsvarende perforatoren til udlæsning uden at påvirke indlæsesituationen.

Eksempel 7.97.

Kernen i et primitivt program til at kopiere en indlæst strimmel kan se således ud:

[m+0] VY 32, LY r+1
[m+1] SY 0, HH r-1

(men programmet bør udstyres med lidt administration for at kunne starte og stoppe på fornuftig vis).

I øvrigt henviser vi til hæfte 2, hvor LY og SY omtales i forbindelse med indlæse- og udlæse-programmerne.

7.3 Ordrer, der benytter indikatoren.

I afsnittene 7.3 og 7.4 omtaler vi i forbindelse med indikatoren flere af de operationer, der ikke er benyttet i de foregående afsnit.

7.3.1 Eksempler på placeringer og lagringer.

Eksempel 7.98.

Vi betragter ordren PI 20 +3 og begynder med at omskrive den modificerede adresse, her 20, og tælletal, her 3, til det binære talsystem. Tallene skrives endvidere med enhed i pos.9.

pos.	0	1	2	3	4	5	6	7	8	9
20 =	0	0	0	0	0	1	0	1	0	0
3 =	0	0	0	0	0	0	0	0	1	1

Virkningen af ordren vil da være, at bit 8 og 9 i indikatoren <u>ikke</u> ændres (da 3 har ettaller i de tilsvarende positioner) medens bit 0, 1, 2, 3, 4, 5, 6, 7 antager værdierne 00000101.

<u>Eksempel 7.99.</u>

Ordren GI 37 +88 bevirker, at indikatorens indhold (på nær KA og KB) lagres i pos. 0-9 i celle 125.

<u>Øvelse 7.100.</u> Find virkningen af følgende program, der står fra celle 37. s-registeret indeholder tallet 900.

```
[37] PI s+179 +33
[38] GI s+179 D +33
[39] ZQ  0
```

<u>7.3.2 Eksempler på ordrer med indikatordele.</u>

<u>Eksempel 7.101. Eksempel på indicering af overløb.</u>

Ordren AR 37 IOB vil primært bevirke, at tallet i celle 37 adderes til indholdet af R-registeret. Såfremt summen, s, ligger udenfor intervallet $-1 \leqq x < 1$, vil der på grund af ordrens indikatordel blive indiceret overløb i OB, d.v.s. at OB:=1. Der vil da også være overløb i R-registeret, og den korrekte halve sum kan dannes i R ved hjælp af ordren TK -1.

Ordren AC 37 IOB vil primært bevirke, at summen s = celle [37] + R dannes i H-registeret hvorfra bits 0-39 overføres til celle 37. Såfremt s ligger i intervallet -1 $\leq$ s < 1, vil indikatordelen bevirke, at OB:=0; i modsat fald sættes OB:=1, og indholdet i celle 37 er galt. Situationen kan kun reddes med stort besvær, fordi det oprindelige indhold i celle 37 er gået tabt.

Ordren HV 10 IOB vil primært bevirke et hop til den første halvordsordre (eller helordsordren) i celle 10. Hvorvidt der bliver indiceret overløb eller ikke-overløb afhænger af resultatet af den nærmeste foregående aritmetiske operation.

Eksempel 7.102. Eksempel på indicering af nulsituation.

Ordren SNS 682 IZC +2 vil primært bevirke, at den absolutte værdi af tallet i celle 684 med negativt fortegn overføres til R-registeret. Hvis R er lig 0, sættes OA:= OB:=1, og hvis det er forskelligt fra 0, sættes OA:= OB:= 0.

Eksempel 7.103. Eksempel på indicering af nulsituation.

Lad M = 0.125. Ordren GRS 37 X IZA vil da primært bevirke, at R:= celle[37]:= 0; derefter indiceres nulsituationen således, at OA:=1. Endelig ombyttes R og M.

Eksempel 7.104. Eksempel på indicering af tegn.

Ordren MK 10 ITA bevirker primært, at M celle[10] adderes til R og anbringes i R. Derefter indiceres resultatets fortegn: Er resultatet negativt, sættes TA:= 1, og ellers sættes TA:= 0. Dette sker uanset om der er overløb eller ej.

Ordren ARS 10 X ITA bevirker primært, at celle 10 føres til R-registeret; derefter indiceres fortegnet for dette tal. Endelig føres tallet til M (og M's indhold til R).

Øvelse 7.105. Beskriv virkningen af ordren XR 10 ITC +2.

Eksempel 7.106. Eksempel på indicering af mærkning.

Ordren SR 11 IRB bevirker primært, at celle 11 subtraheres fra R, samt at mærkningen i celle 11 registreres, d.v.s. overføres til mærkebits i R; derefter bevirker indikatordelen IRB, at b-mærkningen indiceres, altså at RB:= Rpos[41]:= pos[41,11].

Ordren SR 11 D IRB bevirker primært, at tallet $11\cdot 2^{-9}$ subtraheres fra tallet i R; herved ændres R's mærkebits ikke, og indikatordelen IRB bevirker som før, at R_{41} overføres til RB (uden nogen forbindelse med mærkningen i celle 11).

Eksempel 7.107. Eksempel på mærkning.

Ordren GR 12 MB +1 bevirker, at tallet i R lagres i celle 13 samt at bit nr. 40-41 sættes lig 01 i celle 13 og lig 11 i R.

Ordren GR 12 D MB +1 bevirker, at indholdet i R_{0-19} lagres i de tilsvarende positioner i den celle, hvor ordren står. I denne celle sættes derefter bit nr. 40-41 lig 01 (og R_{40-41}:= 11). Lad indholdet i R_{0-9} være 16, og lad indholdet i R_{10-19} være -2; da bliver ordren ændret til GRF 16 D MB -2.

Øvelse 7.108. Beskriv virkningen af ordren GT r+3 D MA +1.

Eksempel 7.109. Eksempel på mærkning.

Ordren GR 12 MTA bevirker, at tallet i R lagres i celle 12, samt at bit nr. 40 i cellen sættes lig TA, medens bit nr. 41 nulstilles. R_{40} og R_{41} er uændrede.

Eksempel 7.110. Eksempel på mærkning.

Ordren AC 100 MQC bevirker primært, at indholdet i celle 100 forøges med tallet i R-registeret, og at mærkningen i celle 100 registreres, altså overføres til R's mærkebits. Derefter bevirker indikatordelen MQC, at de to mærkebits i celle 100 får samme indhold som QA, QB, og under denne mærkningsoperation berøres R's mærkebits ikke, således at disse, efter at hele ordren er udført, indeholder den oprindelige mærkning i celle 100.

Eksempel 7.111. Eksempel på betinget hop.

Lad den aktuelle ordre være HV r-1 LT. Hvis fortegnet i R-registeret er negativt (d.v.s. hvis bit nr. 00 er 1), udføres hopordren, og GIER tager derefter fat på den første halvordsordre eller helordsordren i den foregående celle. Hvis fortegnet er positivt, er ordren "blind", og GIER udfører den næste ordre.

Eksempel 7.112. Eksempel på betinget ordre.

Lad den aktuelle ordre være GR p+98 NZB -1. Hvis OB = 0, udføres lagring af R-registerets indhold i celle p+97, og ordrens adressedel ændres til p+97. Hvis OB = 1, er ordren blind, og der foregår hverken lagring eller adresseændring.

Eksempel 7.113. Eksempel på betinget stop.

Lad den aktuelle ordre være ZQ 0 LKA. Hvis operatøren på forhånd har sat KA = 1, stopper GIER (og går ved tryk på NORMAL START videre med næste ordre, uafhængigt af adressen). I modsat fald udfører GIER straks den næste ordre.

Eksempel 7.114. Eksempel på betinget hop.

Lad den aktuelle ordre være HH 17 V NPB +2. Hvis PB = 0, hopper GIER til "højre halvcelle" nr. 20 og tager fat på den anden halvordsordre eller helordsordren i celle 20; hopordren er desuden ændret til HH 19 V NPB +2.

Hvis PB = 1, forbliver ordren uændret, og GIER tager fat på ordren i den umiddelbart følgende celle (idet heller ikke V-modifikationen er virksom).

Øvelse 7.115. Beskriv virkningen af ordren HV s-3 D LRC -3.

7.4 Eksempler på programmer.

Eksempel 7.116. Eksempel på ombytning.

Tallene i celle 102 og 103 ønskes ombyttet. Det kan gøres således:

```
[m+0]  ARS 102,  PM 103  [R:= x, M:= y]
[m+1]  GR  103,  GM 102  [lagring]
[m+2]  ZQ    0           [stop]
```

Bemærk, at denne kode fungerer lige godt, hvad enten tallene er maskintal eller flydende tal (eller en hvilken som helst anden talform). Cellernes mærkebits ændres <u>ikke</u> af ovenstående ordrer.

<u>Øvelse 7.117.</u> a) Idet maskintallene x, y, z og v er lagret i cellerne 800, 801, 802 og 803, skal man beregne (x+y)(z−v) og lagre resultatet i celle 799. Optræder der overløb i mellemresultater eller facit, ønskes hop til celle 0. b) Den samme beregning skal foretages, idet x, y, z og v er flydende tal.

<u>Øvelse 7.118.</u> a) Lad a, b og c være tre maskintal, der er lagret i cellerne 50, 51 og 52. Idet det antages, at c $\neq$ 0 samt at abs(a) $\leqq$ abs(c), skal ab/c lagres i cellen umiddelbart før den første ordre-celle i programmet.

b) Lad tre maskintal a1, b1 og c1 være lagret som før. Hvis a1 × b1/c1 er et maskintal, skal dette lagres som før; i modsat fald ønskes hop til celle 0, hvis c1 $\neq$ 0, og hop til celle 10, hvis c1 = 0.

c) Samme opgave som under b), idet a1, b1 og c1 er flydende tal.

<u>Øvelse 7.119.</u> Lad heltallene x og y være lagret i celle 47 og 48 med enhed i pos. 39. I de samme celler ønskes lagret: den heltallige kvotient x/y i celle 47 og den tilhørende rest i celle 48; resten skal være $\geqq$ 0.

Eksempel 7.120. Eksempel på adresseberegning.

I adressedelene af celle 859 og 865 står heltallene N og J. NJ-1 ønskes indsat som adressetal i celle s+2:

[m+0]	PM	(859) D	
[m+1]	MKS	(865) D	[N × J står med enhed i pos. 18 i R]
[m+2]	TK	9	[N × J med enhed i pos. 9 i R]
[m+3]	SR	1 D	[N × J − 1 i R_{0-1}]
[m+4]	GA	s+2, ZQ 0	[N × J − 1 gemmes i celle s+2]

Bemærk, at det er nødvendigt at beregningen er D-modificeret for at undgå, at eventuelt indhold i de øvrige positioner i cellerne 859 og 865 påvirker adresseberegningen. Hvis man ved, at pos. 10-18 i de to celler er nulstillede (hvis f.eks. tælletallene er 0), kan de to første ordrer i ovenstående erstattes af to halvcelleordrer

PM 859, MKS 865,

fordi indholdet i resten af de to celler ikke kan spolere multiplikationen af adressedelene.

Øvelse 7.121. a) I adressedelene af ordrerne i celle s-3, s-2 og s-1 står heltallene A, B og C. Der ønskes en kode (afsluttet med en stopordre), som i helcellen umiddelbart efter stopordren indsætter A + 4B - C som adressetal.

b) Det samme problem ønskes løst, når man tillige ved, at tælletallene i celle s-3, s-2 og s-1 alle er 0.

Eksempel 7.122. Eksempel på løkkeregning.

I mange programmer kan ēn bestemt programdel gentages et stort antal gange, idet de samme operationer skal udføres på mange tal. For at kunne benytte den samme programdel skal adresserne i nogle af ordrerne blot ændres for hver gentagelse, og dette kan ske ved hjælp af tælletal.

Desuden skal koden indeholde en mekanisme, der holder regnskab med, hvor mange gange programdelen (løkken) er gennemløbet, og dette kan i GIER gøres på mange måder.

Endelig indledes programmet som regel med nogle ordrer, der retablerer alle ordrer, der ændres (med tælletal o.s.v.) under gennemløbet. Årsagen hertil er, at hele programmet ofte skal benyttes flere gange efter hinanden (f.eks. med forskellige datasamlinger).

Lad f.eks. 50 maskintal være lagret i celle 200 og de følgende celler; man skal anbringe summen af tallenes absolutte værdier i M-registeret.

1.Udkast:

```
   [m+0] PAS r+1 +199    [retabler adressetallet 199]
┌→ [m+1] AN  199 +1      [adder tallene absolut]
│  [m+2] BS (r-1) +248   [de næste to ordrer udføres først
│                         når 199 + j > 248]
│  [m+3] XR 0, HH r+1    [ombyt R og M, hop til stop]
└─ [m+4] HV r-3, ZQ  0   [hop tilbage de første 49 gange, stop]
   [m+5] HV r-5          [ved tryk på NORMAL START udføres hele
                          programmet endnu engang]
```

Følgende fremgangsmåde er lidt kortere i kodelængde, men regnetiden er omtrent den samme, og additionen af tallene sker bagfra:

2.Udkast:

```
   [m+0] PAS r+1 +250       [retabler adressetallet 250]
┌→ [m+1] AN  250 -1         [adder tallene absolut]
│  [m+2] BS  (r-1) X +200   [næste ordre udføres, så længe
│                             250 - j > 200]
└─ [m+3] HV  r-2 X          [hop tilbage de første 49 gange]
   [m+4] ZQ  0, HV r-4      [stop, begynd forfra ved tryk på
                             NORMAL START]
```

De to X-modifikationer er overflødige de første 49 gange, hvor den foreløbige sum blot flyttes til M og tilbage igen, men den 50. gang udføres kun den første X-variant, hvorved den endelige sum placeres i M.

3.Udkast:

```
   [m+0] PA  r+3
   [m+1] PAS r+1 +199
┌→ [m+2] AN   199 +1
│  [m+3] IT  0    +1
└─ [m+4] BS 50,   HV r-2
   [m+5] XR  0,   ZQ   0
   [m+6] HV  r-6
```

I de to sidste udkast er regnetiden omtrent den samme, idet programmets centrale del, nemlig den løkke der skal gennemløbes 50 gange, er nogenlunde ens i de to tilfælde.

Ved at benytte mærkning i forbindelse med en indikatordel kan man imidlertid danne en løkke på kun 2 ordrer; idet vi antager, at det sidste tal i talgruppen er a-mærket (og at ingen af de øvrige er det) fås:

4.Udkast:

```
[m+0] PAS r+1 +199 [retabler adressetallet 199]
→ [m+1] AN  199 +1   [adder tallene absolut. Mærkning re-
                      gistreres]
  [m+2] HV  r-1 NA   [hop tilbage, sålænge der ikke er
                      a-mærkning]
[m+3] XR  0,   ZQ 0 [flyt summen til M, stop]
[m+4] r-4            [gentag hele programmet ved tryk på
                      NORMAL START]
```

Bemærk: Mens de tre første udkast kun fungerer korrekt, hvis talmængden netop består af 50 tal, kan det fjerde udkast bruges til at summere en talmængde af vilkårlig længde (forudsat, at den lagres fra celle 200 og fremefter, samt at kun det sidste tal er a-mærket).

En ofte nyttig tællemekanisme kan dannes ved at benytte p-registeret således:

5.Udkast:

```
[m+0] PPS 0, AN p+200
[m+1] PP  p+1, IT p
[m+2] BS  50 , HH r-2
[m+3] XR   0 , ZQ  0
[m+4] HV  r-4
```

Blot for at vise GIERs mangfoldige muligheder vil vi vise endnu to metoder for den samme opgave.

6.Udkast:

```
    [m+0] PAS r+1 +199
┌─> [m+1] AN  199 X +1
│   [m+2] ARS r-1, NC 249
└── [m+3] HV  r-2 X
    [m+4] ZQ  0, HV r-4
```

7.Udkast:

```
    [m+0] PA  r+3 +49
    [m+1] PAS r+1 +199
┌─> [m+2] AN  199 X +1
│   [m+3] BT  49 -1
└── [m+4] HV  r-2 X
    [m+5] ZQ  0, HV r-5
```

Den sidste metode, hvor tællingen foregår i BT-ordren, har vist sig meget nyttig. Metoden kan nemt generaliseres, idet en løkke på f.eks. 20 celler, der skal gennemløbes K+1 gange, ser således ud:

```
    .....
    PA r+20 +K
┌─> .....
│
│   BT K -1
└── HV r-20
```

Øvelse 7.123. a) N maskintal er lagret i cellerne 300, 301, ... Skriv en kode, der anbringer det største af disse tal i R, idet N står som adressetal i celle 299.

b) Samme opgave, idet det vides, at kun det sidste tal er c-mærket. Betingelsen bør udformes ved hjælp af indikatordelen LPC, ikke NPC (eller tilsvarende). Hvorfor?

Eksempel 7.124. Eksempel på løkkeregning.

Maskintallene x_0, x_1, ..., x_{100} er lagrede i cellerne 100-200. Kun celle 200 er a-mærket. Σa_i ønskes dannet i celle 250.

Vi vil løse opgaven ved at transformere maskintallene til tal på flydende form og udføre summationen ved anvendelse af flydende regning:

```
    [m+0]  PAS r+1 +99
┌─→ [m+1]  ARS 99 +1          [Maskintal i R]
│   [m+2]  NKF 0, GRF (r-1)  [Flydende tal tilbage]
└── [m+3]  HV  r-2 NA         [Alle tal er nu på flydende form]
    [m+4]  PAS r+1 +99
┌─→ [m+5]  ARF 99 +1          [Summationen foretages]
└── [m+6]  HV r-1 NA
    [m+7]  GRF 250, ZQ 0
```

Øvelse 7.125. Samme opgave som i 7.124, men nu må der kun anvendes en løkke i programmet.

<u>Øvelse 7.126.</u> I cellerne 100 til 150 er der lagret 51 tal, der alle er på flydende form. Kun celle 150 er b-mærket. De tal, der kan transformeres til maskintal, skal transformeres og lagres i deres oprindelige celler, og disse celler a-mærkes.

<u>Eksempel 7.127. Eksempel på sortering.</u>

Maskintallene a_0, a_1, ..., a_{100} står i cellerne 200, 201, ..., 300, og kun celle 300 er a-mærket. Adressen på det største af disse tal skal anbringes som adressedel i celle 400:

[m+0]	PA r+5 +200	[retabler adressen i SR-ordren]
[m+1]	ARS 200, GR r+2	[sæt a_0 i arbejdscellen]
[m+2]	PA 400 V +200	[sæt foreløbig adressetallet 200 i celle 400]
[m+3]	QQ 0	[arb. celle]
[m+4]	ARS r-1	[R:= arb.celle]
[m+5]	SR 200 IPA +1	[R:= arb.celle - a_i, indicer a-mærke]
[m+6]	HV r+3 NT	[hop frem på R $\geqq$ 0]
[m+7]	ARS (r-2), GR r-4	[sæt nyt a_i i arb.celle]
[m+8]	ARS r-3, GA 400	[sæt adr. (a_i) i celle 400]
[m+9]	HV r-5 NPA	[gentag indtil første a-mærke]
[m+10]	ZQ 0	

<u>Eksempel 7.128. Eksempel på tromleoperationer.</u>

Tromlekanal 37 og 38 indeholder henholdsvis maskintallene a_0, a_1, ..., a_{39} og b_0, b_1, ..., b_{39}, mens maskintallene

c_0, c_1, ..., c_{39} er lagret fra celle 50 og fremefter; kun celle 89 er a-mærket. Tallene $d_0 = a_0b_0 + c_0$, $d_1 = a_1b_1 + c_1$, ..., $d_{39} = a_{39}b_{39} + c_{39}$ skal lagres på kanal 39:

```
   [m+0] VK  37, LK 90
   [m+1] VK  38, LK 130       [ de to kanaler overføres til
                                lageret]
   [m+2] VK  39, PP 0         [tromletransport afsluttes og
                               tælling startes]
┌→ [m+3] PP p+1, PM p+89      [tælling i p-register, M:= a_i]
│  [m+4] ARS p+49 IQA         [R:= C_i, a-mærkning indiceres]
│  [m+5] MK p+129, GR p+89    [d_i lagres i cellerne 90, 91, ...]
└─ [m+6] HV r-3 NQA           [gentag indtil første a-mærke]
   [m+7] SK 90, ZQ 0          [d_i flyttes til kanal 39]
```

Vi fremhæver, at placeringen af ordren VK 39, før cellerne 130–169 inklusive benyttes, er afgørende.

<u>Øvelse 7.129.</u> Samme opgave som i 7.128 med den ændring, at hverken p-registeret eller mærkning af cellerne må benyttes.

<u>Eksempel 7.130. Kvadratrodsuddragning.</u>

Det følgende program vil ved indhop i celle [m+0] opfatte indholdet af R som et maskintal og uddrage kvadratroden af dette tal, såfremt det er positivt eller nul. Ved udhop står kvadratroden i R. Ved indhop i celle [m+12] opfattes indholdet af RF som et flydende tal. Er dette positivt eller nul uddrages kvadratroden, der afleveres i RF før udhoppet fra celle [m+19].

Beregningen af kvadratroden y af det ikke-negative maskintal x udføres ved iteration efter formlen

$$y_{n+1} = \tfrac{1}{2}(y_n + \frac{x}{y_n})$$

$$y_0 = x \cdot 2^u$$

hvor u vælges, således at

$$1/4 \leqq x \cdot 2^{2u} < 1$$

Det flydende tal kan skrives $p \cdot 2^q$. Hvis q er lige, beregnes $\sqrt{\frac{p}{4}} \cdot 2^{\frac{(q+2)}{2}}$, hvis q er ulige, beregnes $\sqrt{\frac{p}{2}} \cdot 2^{\frac{(q+1)}{2}}$.

```
Maskintal -> [m+0]  PA r+8 +4
             [m+1]  GR r+9, NK r+1
             [m+2]  SRS 0 D
             [m+3]  TK -11, GT r+1
             [m+4]  ARS r+6, TK 0
          -> [m+5]  GR r+6, ARS r+5
          |  [m+6]  DK r+5, AR r+5
          |  [m+7]  TK -1, IT -1
          -- [m+8]  BT  4, HV r-3  [Iterationen udføres 5 gange]
             [m+9]  HR s+1         [Udhop for maskintal]
             [m+10] QQ 0
             [m+11] QQ 0
Flydende
tal       -> [m+12] GR r-2 X
             [m+13] AR 2 D
             [m+14] TK -1, GA r+5
             [m+15] TK 10, CK -19
             [m+16] GT r+1, ARS r-6
             [m+17] TK 9, TK 1
             [m+18] HS r-18
             [m+19] NKF 0, HR s+1 [Udhop for flydende tal]
```

Øvelse 7.131. Skriv kommentarer til ordrerne i eksempel 7.130.

Øvelse 7.132. Programmet i eksempel 7.130 skal ændres, således at der sker udhop til celle s+1, hvis det givne tal er negativt og ellers udhop til celle s+2. Hvis det opgivne tal er nul, skal udhoppet ske straks.

Øvelse 7.133. Lav et program, der er specielt indrettet til at uddrage kvadratroden af flydende tal.

Eksempel 7.134. Beregning af n!.

Det følgende program udregner n! = 1·2·3·.....·n flydende i celle s+1, når n står i adressedelen af celle s-1.

Program:

```
      [m+0]  ARS (s-1) DV +1
      [m+1]  QQ 0 +256          [1 flydende]
  --> [m+2]  ARSF r-1, GRF s+1
 |    [m+3]  ARS (s-1) D -1
 |    [m+4]  HR s+2 LZ
 |    [m+5]  NKF 9, MKF s+1
  --- [m+6]  HH r-4
```

Øvelse 7.135. Skriv kommentarer til ordrerne i eksempel 7.134.

Eksempel 7.136. Brug af undersekvenser.

Programmer, der kan løse hyppigt forekommende problemer, udfærdiges normalt som subroutiner (bibliotekssekvenser, standardsekvenser). Som eksempler på, hvordan sådanne subroutiner kan udformes, nævner vi eksempel 7.130 (kvadratrod) og eksempel

7.134 (n!). For sådanne sekvenser gælder der to hovedkrav: 1) Sekvenserne skal kunne virke uanset hvor de anbringes i lageret. 2) Brugeren skal kunne hoppe til sekvenserne fra alle steder i sit program, og sekvenserne skal automatisk sørge for korrekt tilbagehop. Det første krav opfyldes ved anvendelse af relativmærkede adresser i sekvensen, og det andet ved anvendelse af HS- og HR-hop.

Vi viser samspillet mellem disse sidstnævnte hop i henholdsvis <u>hovedprogrammet</u>, d.v.s. brugerens eget program og i undersekvensen, der f.eks. kan være sekvensen for n! (side 141). Vi antager, at koderen i sit program har brug for 12!.

Hovedsekvens, lagret fra 100	Undersekvens, lagret fra 800
[100] ...	[800] ...
⋮	⋮
[117] PA 0 D +12	[804] HR s+2 LZ
[118] HS 800	...
[119] QQ 0	
[120] ...	

Ordren i celle [118] bevirker:

1) Indholdet af s-registeret placeres som tælletal i 118.

2) s-registeret får derefter indholdet 118.

3) Næste ordre hentes i celle 800.

Undersekvensen bevirker nu, at 12! udregnes og lagres i celle s+1, d.v.s. i celle 119, og ordren i celle 804 bevirker derefter:

1) Næste ordre hentes i celle s+2, d.v.s. i celle 120.

2) Tælledelen i celle s, d.v.s. i celle 118, overføres til s-registeret, der nu har fået sit "gamle" indhold.

Øvelse 7.137. I en hovedsekvens har man brug for $\sqrt{13!}$. Anbring kvadratrodssekvensen fra 700 og n!-sekvensen fra 725. Skitser et stykke af hovedsekvensen og undersøg de forskellige nødvendige hopordrer.

Øvelse 7.138. Øvelse med undersekvens. Tallet 0.05 er lagret i celle 799 (som maskintal), og i celle 850 ff er der lagret en undersekvens, der beregner kvadratroden af det tal, der står i R-registeret; undersekvensen slutter (med den forlangte kvadratrod i R-registeret) med ordren HR s+1. Skriv en kode, der lagrer

sqrt (0), sqrt (0.05), sqrt (0.1), ..., sqrt (0.95)

i cellerne 800, 801, ...

Øvelse 7.139. Lav to programmer, der kan udregne størrelsen $\binom{n}{r} = \frac{n(n-1)\ ...\ (n-r+1)}{1\cdot 2\cdot .....\cdot r}$ flydende i celle s+1, når n og r står i adressedelene af henholdsvis celle s-2 og celle s-1:

a) Ved at benytte programmet fra eksempel 7.134 som undersekvens.

b) Ved at foretage udregningen i følgende orden

$\frac{n}{r} \cdot \frac{n-1}{r-1} \cdot ..... \cdot \frac{n-r+1}{1}$.

Ved den første metode overskrides maskinens talområde allerede for moderate værdier af n, medens den anden metode kan bruges for væsentlig større værdier.

Øvelse 7.140. I R's adressepositioner findes et tal mellem 1 og 5. Lav et program der bevirker, at der hoppes til

celle	100	hvis	Radr	er	1,
-	120	-	-	-	2,
-	140	-	-	-	3,
-	160	-	-	-	4,
-	180	-	-	-	5.

Øvelse 7.141. Fire heltal a, b, c og d er anbragt i celle 100 med enhed i henholdsvis pos. 9, pos. 19, pos. 29 og pos. 39. Tallene ønskes anbragt i henholdsvis celle 101, 102, 103 og 104 med enhed i pos. 39.

Øvelse 7.142. Lagring som i øvelse 7.141. Dan ac + bd i R med enhed i pos. 39.

Øvelse 7.143. Hundrede maskintal er lagrede fra celle 100. Mærk de celler, der indeholder positive tal, med et a, og de celler, der indeholder nul, med et b.

Øvelse 7.144. Anbring det største af tallene A og B i celle 5, når A er det mindste af tallene i celle 1 og celle 2, og B er det mindste af tallene i celle 3 og celle 4.

Øvelse 7.145. Maskintallene a og b er lagrede i cellerne 100 og 101. Dan $-|a|/(|a|+|b|)$ i R. Facit skal være normaliseret, og normaliseringsexponenten skal stå i M_{0-9}.

Øvelse 7.146. Samme problem som i øvelse 7.145, idet facit nu ønskes fundet på flydende form.

Øvelse 7.147. To flydende tal, x og y, er lagrede i cellerne 20 og 21. $|x| > |y|$. Dan $\frac{y}{x}$ som maskintal i celle 22.

Øvelse 7.148. En samling maskintal er lagrede med første i celle 200. Nogle af tallene er a-mærket. Det sidste er b-mærket (og eventuelt også a-mærket). Dan summen af de a-mærkede tal i celle 199.

Øvelse 7.149. En samling maskintal er lagrede fra celle 100. Kun det sidste er b-mærket. Tallene skal behandles på følgende måde: Negative tal halveres og positive tal fordobles. Det forudsættes, at der ikke optræder overløb.

Øvelse 7.150. Samme opgave som øvelse 7.149, men med flydende tal.

Øvelse 7.151. Sæt 1 i alle bits i R-registeret uden anvendelse af indlæste konstanter. (Det kan gøres med et program, der fylder 2 helceller inklusive stop.)

Eksempel 7.152.

Maskintallene a_0, b_0, a_1, b_1, ..., a_n, b_n er lagrede fra celle 400. Kun b_n er a-mærket. $\Sigma a_i b_i$ skal dannes i R.

Program:

```
   [m+0]  PAS  r+1 +399
→  [m+1]  PM   0 +1
   [m+2]  MK   (r-1) +1
   [m+3]  HV   r-2 NA
   [m+4]  ZQ   0
```

Øvelse 7.153. Samme opgave som eksempel 7.152, men med flydende tal.

Øvelse 7.154. Samme opgave som i eksempel 7.152, men produktsummen skal lagres uafkortet i R,M.

Øvelse 7.155. Lagring som i eksempel 7.152. Dan $\Sigma(b_i)^2$ i celle 399 og $\Sigma(a_i)^2$ i celle 398.

Øvelse 7.156. Samme opgave som i øvelse 7.155, men med flydende tal.

Øvelse 7.157. Maskintallene a_0, a_1, a_2, ..., a_n er lagrede fra celle 100. a_n er b-mærket. Tallet x står i R. I celle 99 skal lagres $y = a_n x^n + a_{n-1} x^{n-1} + \ldots\ldots + a_1 x + a_0$. Benyt algoritmen $y = (\ldots((a_n x + a_{n-1})x + a_{n-2})x + \ldots\ldots + a_0)$.

Eksempel 7.158.

Lad indholdet af R være

	00	0								39
R =	1	1	0	1	0	. . .	1	1	0	1

Ordren CK 3 vil da bevirke, at R får indholdet

	00	0										39
R =	0	0	. . .	1	1	0	1	1	0	1		

Øvelse 7.159. Lad indholdet af RM være

00	0	R			38	39		0	1	M	38	39
0	0	1	0	. . .	1	0		1	0	. . .	1	1

Hvad bliver indholdet af R,M efter udførelsen af ordren CL −1.

Eksempel 7.160. Eksempel på simpel indlæsning.

Til ethvert af de symboler (d.v.s. bogstaver, cifre eller tegn), der kan indlæses eller udlæses svarer der et heltal. Omvendt svarer der til ethvert heltal i intervallet $0 \leqq x \leqq 64$ normalt to symboler, nemlig et i Lower Case og et i Upper Case. Hvilket af de to tegn det drejer sig om på et givet tidspunkt afhænger af det sidst anvendte casetegn.

Tallene 49, 50, 51 svarer til henholdsvis a, b og c i Lower Case og henholdsvis A, B og C i Upper Case.

Vi vil nu indlæse et tegn fra strimmellæseren. Er det et A (evt.a), et B (evt.b) eller et C(evt.c), skal der hoppes til henholdsvis celle 100, celle 120 og celle 140. I modsat fald skal der indlæses et nyt tegn, der skal behandles på samme måde o.s.v.

Program:

```
   [m+0]  VY  0
→  [m+1]  LYS 0 D  [Et tegn indlæses til R og til denne celle]
   [m+2]  CA  49, HV 100
   [m+3]  CA  50, HV 120
   [m+4]  CA  51, HV 140
   [m+5]  HV  r-4
```

Eksempel 7.161. Eksempel på udlæsning.

Der skal udarbejdes et program, der dirigerer skrivemaskinen til ny linie, "skriver" SPACE ti gange og endelig skriver 3/7 1962.

Program:

```
[m+0] VY 16, SY 64  [64 er tegnet for CR (linieskift)
                      i både Upper og Lower Case]
[m+1] PA r+1 +10
[m+2] BT 10 -1
[m+3] SY  0, HV r-1 [0 er tegnet for SPACE i både Upper
                      og Lower Case]
[m+4] SY 58, SY 3   [Lower Case, skriv cifferet 3]
[m+5] SY 60, SY 3   [Upper Case, skriv /]
[m+6] SY 58, SY 7
[m+7] SY  0, SY 1
[m+8] SY  9, SY 6
[m+9] SY 16, ZQ 0   [16 er tegnet for 0]
```

Øvelse 7.162. Indlæs tyve tegn (bogstaver eller cifre) fra strimmelen og skriv tegnene ud på skrivemaskine i modsat rækkefølge med SPACE mellem de enkelte tegn. Det forudsættes, at alle tegn er i samme case.

Øvelse 7.163. Samme opgave som i øvelse 7.162, men nu kan der forekomme caseskift mellem tegnene, svarende til at der forekommer både store og små bogstaver.

Eksempel 7.164. Eksempel på overløb.

Maskintallene a_0, a_1, ..., a_{100} er lagrede i cellerne 100 til 200. Kun celle 200 er a-mærket. Summen $s = \Sigma a_i$ skal dannes i celle 201; hvis der opstår overløb under summationen, skal der foretages de nødvendige højreskift.

Program:

```
   [m+0]  PA  r+4 +0
   [m+1]  PAS r+2 +99
┌→ [m+2]  GR  201
│  [m+3]  ARS 0 IPA +1
│  [m+4]  TK  0, AR 201
│  [m+5]  QQ  (r-1) LO -1
│  [m+6]  TK  -1 LO
└─ [m+7]  HV  r-5 NPA
   [m+8]  GR  201, ZQ 0
```

Såfremt der kommer overløb under regningerne, vil programmet sørge for at skifte såvel den dannede delsum som de følgende addender til højre. Det endelige resultat af programmets arbejde bliver således ikke s, men $s \cdot 2^{-q}$. Exponenten $-q$ findes som adressetal i celle [m+4].

Øvelse 7.165. Øvelse med tromleoperationer og boole'ske operationer. I 2000 bevoksninger undersøges det for 40 plantearters vedkommende, hvorvidt de forekommer eller ikke forekommer. Plantearterne nummereres fra 0 til 39, og idet forekomst registreres som et 1-tal og ikke forekomst som et nul, kan hver lagercelle netop rumme oplysningerne om en bevoksning. Den indsamlede informationsmængde lagres nu i 2000 celler på tromlen, nemlig i 50 konsekutive kanaler begyndende med kanal nr. 60. Idet den sidste celle i den sidste kanal er (som den eneste) b-mærket, ønskes en kode, der

a) i celle 400 sættes nuller udfor de plantearter, der slet ikke forekommer i de 2000 bevoksninger; i resten af positionerne skal der sættes 1-taller.

b) i celle 401 sættes 1-taller udfor de plantearter, der forekommer i alle de 2000 bevoksninger; i resten af positionerne skal der sættes nuller.

c) i celle 402 lagres antallet af bevoksninger, hvor i hvert fald plantearterne nr. 0, 1, 2, ..., 10 forekommer; antallet lagres som heltal med enhed i pos. 39.

Øvelse 7.166. Samme problem som i den foregående øvelse, idet a- eller b-mærkning ikke bruges.

8. OVERSIGTER.

8.1 Grundoperationernes talværdier.

I nedenstående tabel er grundoperationerne ordnet efter deres decimale talværdi, d.v.s. indholdet i pos. 20-25 (henholdsvis 30-35) opfattet som heltal.

Talværdi	Operation
0	QQ
1 16	ZQ
2 32	AR
3 48	SR
4 64	AN
5 80	SN
6 96	AC
7 112	SC
8 128	MB
9 144	AB
10 160	MT
11 176	MK
12 192	ML
13 208	DK
14 224	DL
15 240	NK
16 256	NL
17 272	HR
18 288	TL
19 304	CK
20 320	CL
21 336	GR
22 352	GA
23 368	GT
24 384	TK
25 400	CA
26 416	GM
27 432	PM
28 448	XR
29 464	GI
30 480	PS
31 496	PP

Talværdi	Operation
32 512	PA
33 528	PT
34 544	HK
35 560	PI
36 576	IS
37 592	IT
38 608	CM
39 624	BT
40 640	NS
41 656	NT
42 672	GP
43 688	NC
44 704	
45 720	
46 736	
47 752	
48 768	
49 784	BS
50 800	HS
51 816	VY
52 832	LK
53 848	SK
54 864	GK
55 880	VK
56 896	HV
57 912	
58 928	SY
59 944	LY
60 960	HH
61 976	GS
62 992	
63 1008	UD

Som det fremgår af tabellen er 7 af de 64 mulige kombinationer endnu uudnyttede.

8.2 Indikatoroperationer og lignende.

Overløb registreres efter operationerne:

AR - AN - AC
SR - SN - SC
MK - ML - MT
DK - DL
TK - TL - CK - CL

} samt disses varianter.

O-registeret er uændret efter alle øvrige operationer.

Fortegn kan indiceres efter operationerne:

AR - AN - AC - AB
SR - SN - SC
MK - ML - MT - MB
DK - DL
NK - NL - TK - TL
XR

} samt disses varianter.

Enhver operation kan udstyres med ITA, ITB eller ITC, men kun ved ovennævnte operationer indiceres fortegnet for det aritmetiske resultat.

Mærkning registreres ved operationerne:

AR - AN - AC - AB
SR - SN - SC
MK - ML - MT - MB
DK - DL
PM
CM

} samt disses S-, F-, X- og V-varianter.

R's mærkebits er uændrede ved D-varianten af ovenstående operationer samt ved alle øvrige operationer.

Mærkning kan udføres ved operationerne:

AC - SC
GR - GM - GA - GT
NK - NL

} samt disses varianter.

Ved en absolut mærkning sættes samtidig $R_{40,41}$:= 11, og efter en indikatorafhængig mærkning er $R_{40,41}$ uændrede.

8.3 Korrespondancen på Flexowriter mellem trykte symboler og hulkombinationer.

Symbol Lower Case	Symbol Upper Case	Strimmel
a	A	`, oo  .  o,`
b	B	`, oo  . o ,`
c	C	`, ooo . oo,`
d	D	`, oo  .o  ,`
e	E	`, ooo .o o,`
f	F	`, ooo .oo ,`
g	G	`, oo  .ooo,`
h	H	`, oo o.   ,`
i	I	`, oooo.  o,`
j	J	`, o o .  o,`
k	K	`, o o . o ,`
l	L	`, o   . oo,`
m	M	`, o o .o  ,`
n	N	`, o   .o o,`
o	O	`, o   .oo ,`
p	P	`, o o .ooo,`
q	Q	`, o oo.   ,`
r	R	`, o  o.  o,`
s	S	`,  oo . o ,`
t	T	`,  o  . oo,`
u	U	`,  oo .o  ,`
v	V	`,  o  .o o,`

Symbol Lower Case	Symbol Upper Case	Strimmel
w	W	`,  o  .oo ,`
x	X	`,  oo .ooo,`
y	Y	`,  ooo.   ,`
z	Z	`,  o o.  o,`
æ	Æ	`, ooo .   ,`
ø	Ø	`, o oo. oo,`
0	∧	`,  o  .   ,`
1	∨	`,     .  o,`
2	×	`,     . o ,`
3	/	`,   o . oo,`
4	=	`,     .o  ,`
5	;	`,   o .o o,`
6	[	`,   o .oo ,`
7	]	`,     .ooo,`
8	(	`,    o.   ,`
9	)	`,   oo.  o,`
,	10	`,  ooo. oo,`
.	:	`, oo o. oo,`
−	+	`, o   .   ,`
<	>	`,  oo .  o,`
_	|	`,    o.oo ,`

Følgende hulkombinationer svarer til samme typografiske symbol i Lower og i Upper Case:

Typogr. symbol	Strimmel
Car.Return	`,o   .   ,`
Mellemslag	`,   o .   ,`
Tabulator	`,  ooo.oo ,`
Skift til LC	`, oooo. o ,`
Skift til UC	`, oooo.o  ,`
Tape Feed	`, oooo.ooo,`

Typogr. symbol	Strimmel
Stop Code	`,    o. oo,`
Punch Off	`,  o o.ooo,`
Punch On	`, o  o.o  ,`
Punch Adress	`,o   .   ,`
Aux Code	`,    o.o  ,`

Nøglen for tegnene _ og | fører ikke vognen frem. Nøglerne Punch Adress og Aux Code fungerer kun, når de benyttes samtidigt med en anden nøgle, og på strimlen perforeres da de tilsvarende hulkombinationer oven i hinanden.

8.4 Korrespondancen mellem talværdier og typografiske symboler.

Nedenstående tabel viser sammenhængen mellem de typografiske symboler og talværdierne i GIER. For ethvert symbol viser tabellen således det tal, a) som ved indlæsning af symbolet (med en LY-ordre) anbringes i pos. 3-9 af R og cellen, b) som skal benyttes som adresse i en SY-ordre ved udlæsning af symbolet (smlgn. operationslisten, ydre enheder).

Talværdi	Symbol Lower Case	Symbol Upper Case
0	Mellemslag	
1	1	∨
2	2	×
3	3	/
4	4	=
5	5	;
6	6	[
7	7	]
8	8	(
9	9	)
10	ubenyttet	
11	Stop Code	
12	ubenyttet	
13	å	Å
14	_	|
15	ubenyttet	
16	0	∧
17	<	>
18	s	S
19	t	T
20	u	U
21	v	V
22	w	W
23	x	X
24	y	Y
25	z	Z
26	ubenyttet	
27	,	$_{10}$
28	ubenyttet	
29	rødt farveb.	
30	Tabulator	
31	Punch Off	

Talværdi	Symbol Lower Case	Symbol Upper Case
32	−	+
33	j	J
34	k	K
35	l	L
36	m	M
37	n	N
38	o	O
39	p	P
40	q	Q
41	r	R
42	ubenyttet	
43	ø	Ø
44	Punch On	
45	ubenyttet	
46	ubenyttet	
47	ubenyttet	
48	æ	Æ
49	a	A
50	b	B
51	c	C
52	d	D
53	e	E
54	f	F
55	g	G
56	h	H
57	i	I
58	Lower Case	
59	.	:
60	Upper Case	
61	ubenyttet	
62	sort farveb.	
63	Tape Feed	
64	Car.Return	

Symbolerne å, Å, rødt farvebånd og sort farvebånd findes kun på skrivemaskinen, ikke på flexowriter'en, og derfor har f.eks. ordrerne SY 13, SY 29 og SY 62 kun virkning ved udlæsning på skrivemaskinen.

STIKORDSREGISTER.

(Op) henviser til beskrivelser i operationslisten.