E L E M E N T Æ R T B R U G E R K U R S U S

TIL: GIER-installationerne ved Københavns Universitets Matematiske Institut.

ved: Ellen Banz og Klaus Hansen

Dette brugerkursus forudsætter:

1) elementært kendskab til datamaters opbygning

2) kendskab til GIER algol 4

3) en ubændig trang til at lære at læse RC-manual-engelsk og RC-manual-notation.

Kurset giver ingen oplysninger om afdelingens hullemateriel (DURA), her henvises til Peter Naurs kursus.

INDHOLDSFORTEGNELSE

INDHOLDSFORTEGNELSE fortsat.

INDHOLDSFORTEGNELSE fortsat.

FORORD

Første del af dette brugerkursus, materiel, er anden reviderede udgave. Første udgave udkom i februar 1970, med Anker Berg-Sonne som medforfatter.

Vi takker Krølle for hans sønderlemmende og inspirerende kritik af brugerkursets anden del, programmel.

Januar 1971

Klaus og Ellen.

0. INDLEDNING

For at maskineriet, med det usædvanligt store antal brugere kan fungere nogenlunde smertefrit, er det uhyre vigtigt at alle er hensynsfulde og rydder omhyggeligt op efter sig.

ROD VIRKER DEMORALISERENDE PÅ MENNESKER OG MASKINER.

Til hjælp for brugerne findes forskellige faciliteter:

1) OPERATØR.

Opholder sig om dagen i rum s04.
Træffes han ikke her, kan han (ligeledes om dagen) tilkaldes over telefonen ved at man drejer 0354 og venter til han svarer.

2) RESERVATIONSPROTOKOL.

Reservation af en datamat foretages via reservationsprotokollen, der ligger på skabet i gangen mellem dørene til rum s03 og rum s04.
Oplysninger om regler for reservation af datamaterne findes på opslag.
VIGTIGT: skriv fulde navn tydeligt ved en reservation, ellers risikerer du at blive slettet.

3) SMÅKØRSEL.

Småkørsel er beregnet til mindre kørsler (max. 5 min., hvis nogen venter). I bestillingsbogen er noteret: SMAAKØRSEL udfor de tider, der er reserveret hertil. Udenfor døren til rum s03 hænger en holder med ventemærker. Tag et nummer og stil op i køen.

4) INDLEVERING TIL OPERATØRKØRSEL.

Foregår i specielle kasser, der er anbragt i en reol på gangen. Det forudsættes at man vedlægger en forståeligt udfyldt kørselsinstruks, der også ligger i reolen eller kan fås hos operatøren.

5) HÅNDHULLER, SPLEJSEAPPARATUR, MANUALS, SYSTEMSTRIMLER o.l.

Findes i maskinrummene. Finder du ikke hvad du søger, henvend dig til operatøren.

6) PAPIRKURVE.

kan stilles under læseren, således at store strimler ikke behøver at ligge på gulvet.

7) PAPIRSÆKKE.

benyttes til at henlægge kasserede strimler og papirark.
Endnu engang:

RYD OP, UNDGÅ KAOS.

(se billede 1.1.)

1. MATERIEL

1.1 FEJLPROTOKOL

Det er vigtigt at operatøren får kendskab til enhver fejl, der viser sig ved maskineriet, for at han enten kan rette den eller om nødvendigt tilkalde en tekniker.

Derfor: Opstår der en fejl, tag fejlprotokollen, der ligger på skabet udenfor rum s03 og beskriv fejlen så udførligt som muligt. Husk at ledsage beskrivelsen med dato, klokkeslæt og dit navn. Det sidste er en hjælp for operatøren således at han eventuelt kan kontakte dig og få yderligere oplysninger.

1.2 KATASTROFEKNAP

I begge maskinrum findes på væggen en knap (med en rød lampe) der er mærket: katastrofeknap.

Knappen bevirker at al strøm til maskinerne afbrydes. Denne metode til at slukke for maskineriet medfører at sikringer og muligvis også lamper i Disk-filen springer. Operatør- og eventuelt teknikerhjælp er derfor nødvendig når strømmen igen skal tilsluttes.

Katastrofeknappen skal (og må kun) benyttes i tilfælde af brand og lignende.

1.3 MASKINRUMMENE (se billede 1.2)

Afdelingen er for tiden i besiddelse af to forskellige GIER-anlæg. GIER 1, det ældste anlæg er placeret i rum s03.

GIER 1 har som baggrundslager en tromle hvis kapacitet er 12800 ord. (Et ord eller en celle kan f.ex. indeholde et tal, (40 bit)). Af ydre enheder findes standardenhederne, nemlig en skrivemaskine, en strimmellæser (RC 2000) og en perforator. Iøvrigt kan der tilsluttes en plotter og en linieskriver. Disse sidste ydre enheder deles med GIER 2 anlægget, der er placeret i rum s01-02.

GIER 2s baggrundslager er en disk-file (pladelager) hvis kapacitet svarer til 60 tromler.
Desuden er der til GIER 2 tilsluttet en buffer (extra ferritlager til brug for arrays under kørsel med algoloversætteren ga4b). Derudover har GIER 2 samme ydre enheder som GIER 1.

Den nedenstående mere udførlige beskrivelse af maskineriet gælder, når intet andet bemærkes både GIER 1 og GIER 2.

1.4 GIER SKABET

1.4.1 almindelig beskrivelse

Indeholder styreenhed, regneenhed, ferritlager (hurtiglager - kapacitet på 1024 ord) og kontrolkredsløb for ydre enheder.

1.4.2 tænding-slukning

Ved tænding af GIER 1: 1) hovedkontakten bag døren til GIER 2 rummet skal være slået ned.
2) Nøglen på det store kontrolbord (se under punkt 1.9) drejes højre om.
3) Vent 5 min. Herefter kan du forsøge at benytte maskinerne. Mislykkes dette, tilkald operatøren.

Ved tænding af GIER 2: MÅ KUN TÆNDES AF OPERATØREN.

Ved slukning: Ingen af maskinerne må slukkes af brugerne. Se dog afsnittet om katastrofeknappen.

1.4.3 knapper-lamper

De eneste knapper og lamper i GIER skabet, det er nødvendigt at kende sidder øverst i den midterste del af skabet.

GIER 1: Skiftekontakten med positionerne mærket

fri spærret

bruges til at beskytte en del af baggrundslageret mod uønskede ændringer (se afsnit 2.1). Knappens normale stilling er spærret. De øvrige knappers funktion berører centrale processer og må ikke røres. (se billede 1.4)

1.4.4 yderligere bemærkninger

GIER 1 skabet indeholder udover de i punkt 1.4.1 nævnte enheder også baggrundslageret - en tromle.

1.5 BUFFER (Kun GIER 2)

1.5.1 almindelig beskrivelse

Bufferen indeholder et extra hurtiglager af samme type som ferritlageret. Den bruges kun ved kørsel af algol-programmer oversat med ga4b-oversætteren, (se afsnit 2.3.1). Ved sådanne kørsler bruges bufferens 4096 ord til lagring af arrays.

1.5.2 tænding-slukning

Ved tænding: 1) tryk på knappen mærket ON
2) sørg for at omskifteren står på LOAD ON.

Ved slukning: MÅ IKKE SLUKKES AF BRUGERNE.

1.5.3 knapper-lamper ingen af interesse.

1.5.4 yderligere bemærkninger

Hvis bufferen er slukket under kørsel af et program oversat med ga4b, vil den eneste reaktion være at alle array-elementers værdi forbliver -1 uanset hvilke operationer programmet udfører.

1.6 BAGGRUNDSLAGER

1.6.1 TROMLE (kun GIER 1)

1.6.1.1 almindelig beskrivelse

Baggrundslager for GIER 1, indeholder 320 blokke a 40 ord.

1.6.1.2 tænding-slukning

Tromlen kan kun tændes af en operatør og dette må først ske tidligst 2 timer efter afbrydelsen, derfor

AFBRYD ALDRIG TROMLEN.

1.6.1.3 knapper-lamper ingen af interesse.

1.6.1.4 yderligere bemærkninger

For at operatøren kan vide hvornår han kan tænde tromlen efter afbrydelse (i tilfælde af strømafbrydelse) vil det være en hjælp hvis du, ifald du har oplevet at tromlen standser, noterer tidspunktet i fejlprotokollen.
At tromlen er standset ses ved at den røde lampe på kassen på væggen bag GIER 1 skabet lyser.

1.6.2 DISK-FILE (Pladelager, kun GIER 2)

1.6.2.1 almindelig beskrivelse

Baggrundslager for GIER 2, svarer til 60 tromler (19200 blokke a 40 ord).
Disk-file betyder pladelager. På grund af sin komplicerede mekaniske opbygning er disk-filen meget følsom overfor forkert behandling, derfor: er du i tvivl om noget vedrørende denne så tilkald operatøren.

1.6.2.2 tænding-slukning

Tænding: 1) tryk på start-knappen

2) når lampen, mærket 0, lyser er disk-filen parat til brug.

NB: GÅR NOGET GALT, FORETAG DIG INTET ANDET END AT TILKALDE OPERATØREN.

Slukning: MÅ IKKE SLUKKES AF BRUGERNE.

1.6.2.3 knapper-lamper se under 1.6.2.2.

1.6.2.4 yderligere bemærkninger

Der må ikke skiftes Disk-kit (selve pladelageret) uden operatørhjælp.

1.7 LINIESKRIVER (lineprinter)

1.7.1 almindelig beskrivelse

Maximalhastighed er ca. 10 linier i sekundet. Kan tilsluttes GIER 1 eller GIER 2 ved hjælp af en trykknap.

1.7.2 tænding-slukning (se billede 1.5)

Tændes: 1) off-lampen skal lyse, ellers tændes kontakten på væggen.
2) on-knappen trykkes ind.
3) start-knappen trykkes ind
4) knappen mærket: 2 1 trykkes ind, så nummeret svarende til den maskine du ønsker at benytte, lyser..
5) angående udskiftning af papir, se punkt 1.7.4.

Slukkes: off-knappen trykkes ind.

1.7.3 knapper-lamper (se billede 1.5)

start: se under punkt 1.7.2.

stop: - - - - Lyser stop-lampen er printeren ikke klar - dvs. den kan ikke modtage output fra GIER, som derved ikke blive færdig med sin udskrift-ordre. Dette vil bevirke at du ikke kan komme i kontakt med GIER fra skrivemaskinen. Sørg enten for at starte linieskriveren eller hvis dette ikke kan lade sig gøre så tryk på power off.

test-print: trykker en linie M-er for hvert tryk på knappen.

6 line - 8 line: du kan vælge at få trykt 6 eller 8 linier pr. tomme.

top of form: skifter et vist antal linier frem, afhængig af styrestrimlen. Hvis styrestrimlen passer til papirformatet og papiret er sat korrekt i, vil tryk på knappen bevirke sideskift.

tractor index: unødvendig at kende.

stop on top of form: holdes denne knap inde, vil det virke som hvis man trykkede stop ved top of form.

print cycle: lyser hver gang en linie bliver trykt. lyser den vedvarende, er der noget galt med linieskriveren.

2 1 : se under punkt 1.7.2.

(blank): ubenyttet.

no paper: lyser lampen er det fordi der ikke er mere papir i linieskriveren.

paper low alert: printerpapiret er ved at slippe op, dvs. der trykkes på sidste side.

yoke open: lyser når yoken (se under 1.7.4.) er åben.

alarm status: lyser ved fejl.

1.7.4 yderligere bemærkninger

Angående papirskift i linieskriveren: Første gang skal det demonstreres af operatøren. Herefter kan følgende tjene som huskeseddel:

1) tryk på knappen mærket STOP.

2) skift styrestrimlen, således at den passer med det papirformat du ønsker at bruge.

3) tryk på knappen mærket TOP OF FORM.

4) luk låget på linieskriveren op og før skrivetromlen (yoke) væk fra papiret ved at trykke samtidig på de dertil indrettede kontakter.

5) tag det gamle papir ud og indsæt det nye, således at starten på en side sidder udfor hamrene på linieskriveren.

6) før skrivetromlen tilbage, luk låget.

7) tryk TEST PRINT efterfulgt af TOP OF FORM og undersøg om første linie på siden står som den skal.

8) tryk START.

NB: Opstår der problemer vil operatøren med glæde komme til hjælp.

1.8 PLOTTER

Ønsker du at benytte plotteren, som kan tilknyttes både GIER 1 og GIER 2 direkte, tilkald operatøren, som med stor fornøjelse vil demonstrere brugen af den.

1.9 MANØVREBORD

1.9.1 SKRIVEMASKINEN (for indlæsning: typewriter, for udlæsning: writer)

1.9.1.1almindelig beskrivelse

Skrivemaskinen benyttes af operatøren til ind- og udlæsning for GIER. Den kan modtage/skrive max. 10 tegn pr. sek.

1.9.1.2 knapper-lamper

GRØN LAMPE: lyser, når GIER venter på inddata fra skrivemaskinen.

Iøvrigt virker tasterne som på DURA-maskinerne, undtagen de der benyttes til at flytte margin. Disse vil muligvis blive beskrevet i en senere udgave af noterne, men kan på opfordring demonstreres af operatøren.

1.9.1.3 yderligere bemærkninger

Farvebånd skal udskiftes af operatøren. Der advares stærkt mod at benytte skrivemaskinen som ind- og udlæse medium for rettelser med edit og under kørsel med algol-programmer, da den ifølge sagens natur er meget langsom.

1.9.2 STRIMMELLÆSEREN (reader)

1.9.2.1 Almindelig beskrivelse (se billede 1.6)

Indlæse medium for GIER. Læser ca. 2000 tegn i sekundet ved maximalhastighed.

1.9.2.2. knapper

RESET: tryk på denne knap bevirker at den såkaldte buffer (et lille ferritlager placeret i læseren, med plads til 256 tegn) tømmes og såfremt der sidder en strimmel i læseren indlæses 100 - 200 tegn af denne til bufferen. Bemærk: herved overføres ingen information til GIER, tegnene gemmes blot, parat til indlæsning.

READ: tryk på knappen virker som ved RESET, bortset fra at den information der allerede står i bufferen ikke slettes, bufferen fyldes bare op.

SKIP: tryk på knappen bevirker at strimmel fremføres uden læsning, sålænge knappen holdes nede.

UP: løfter tryklaaget, der holder strimlen på plads.

1.9.2.3 yderligere bemærkninger

HUSK at lågen, der sørger for at strimmelrullen forbliver i læseren kun åbnes ved tryk på den lille pind, der sidder foroven til venstre for lågen. Tryklaaget, der holder strimlen på plads ved læsehovedet kan kun lukkes ned, såfremt lågen er lukket. (se billede 1.6)

Vil læseren ikke læse dine strimler, trænger den muligvis til at blive justeret. Dette må kun gøres af operatøren.

Det er nødvendigt at dine strimler starter med et ca. 20 cm. blankt stykke tape, for at du kan placere den i læseren.

1.9.3 PERFORATOREN (puncher)

1.9.3.1 almindelig beskrivelse

Perforatoren huller ved maximalhastighed 150 tegn pr. sek.

1.9.3.2 knapper

Fremføringsknappen: den lille sorte knap på højre side bevirker, når den skubbes frem mod dig selv, at perforatoren huller blank strimmel, dvs. kun fremføringshullet hulles. (se billede 1.7)

1.9.3.3 papirskift

Perforatoren forstoppes meget let. Det er derfor af stor vigtighed, at du gennemfører alle punkter i følgende beskrivelse omhyggeligt. (se billede 1.7)

1) skift strimmel i god tid, dvs. når det særligt farvede stykke af strimlen viser sig.

2) træk det lille stik ud.

3) læg mærke til hvordan den gamle rulle er sat i, inden du fjerner den.

4) Før den gamle strimmel fjernes skal du vippe fremføringsmekanismen ved at trykke på fingerknappen (se billede 1.7) samt løfte guillotinen.
Vær forsigtig med at fjerne resterne af en strimmel, undgå at lim eller andet kommer ind i perforatoren.
(Skal du skifte strimmel midt under en kørsel, tryk da på normal stop (se 1.9.5.) og efter papirskift normal-start.)

5) fjern ca. 1 meter af den nye strimmelrulle og klip derefter denne omhyggeligt til med en lang spids med afrundede hjørner.

6) lad papiret glide forsigtigt i uden at bruge vold, derved fjernes snavs fra forrige strimmel.

7) du må sikre dig at skuffen under perforatoren ikke trænger til at tømmes (ellers tøm den), da perforatoren kan ødelægges fuldstændigt såfremt konfetti ikke frit kan falde ud.

8) vip fremføringsmekanismen paa plads, og sæt stikket i.

1.9.3.4 yderligere bemærkninger

Perforatoren er sårbar, derfor benyt aldrig skarpe instrumenter, skruetrækkere e.l., men prøv med nænsom hånd og veltilklippet perforatorpapir, såfremt der er forstoppelse. Ellers tilkald operatøren.
Vil en DURA ikke læse dine strimler, skal perforatoren justeres. Tilkald operatøren eller noter det i fejlprotokollen.
Ved perforering af lange strimler må perforatoren af og til have en pause til afkøling. Tryk paa NORMAL-STOP - vent et par minutter - tryk på NORMAL-START. (se billede 1.10).

1.9.4 LILLE KONTROLBORD

1.9.4.1 almindelig beskrivelse

Det lille kontrolbord indeholder de vigtigste knapper til styring af maskinen samt lamper, der tilkendegiver maskinens tilstand.

1.9.4.2 tænding-slukning

Tændes og slukkes sammen med GIERs centralenhed.

1.9.4.3 knapper-lamper (se billede 1.8)

1) knapper:

KA og KB knapper: De to øverste knapper bruges til at tænde eller slukke KA-registeret. De to næstøverste knapper går tilsvarende på KB-registeret.

Reset-knappen: Ved tryk på denne knap standser maskinen øjeblikkeligt og lampen mærket klar tændes. Da der ved tryk på denne knap udløses uønskede bivirkninger bør denne knap benyttes mindst mulig.

HP-knappen: Ved tryk på denne knap afbrydes den operation som maskinen er i gang med og kørselsnummer, dato og udhopscellens nummer udskrives, hvorefter der ventes på information fra skrivemaskinen. (se afsnit 2.)
Reagerer maskinen ikke på den ovenfor beskrevne måde, kan trykket paa HP-knappen efterfølges af et mellemrumsslag på skrivemaskinen.
Skulle den forventede situation ikke opstå ved dette, trykkes Reset efterfulgt af HP.
(Bemærk om linieskriveren står i STOP-situation og er tilknyttet pågældende maskine.)

2) lamper:

KA: angiver ved lys om KA registret er tændt.

KB: angiver ved lys om KB registret er tændt. Inde fra et algol-program kan der testes på KB registrets tilstand ved hjælp af proceduren kbon. Denne får værdien:
true, hvis KB er tændt
false, - - - slukket.

klar: hvis en kørsel er standset ved tryk på Reset
Mikro-tempi stop (se det store kontrolbord)
Normal stop (- - - -)
lyser denne lampe.

YE: angiver ved lys om maskinen venter at en ydre enhed skal blive færdig med at læse eller skrive.

str.læs.parf. hvis denne lampe lyser er strimmellæseren standset ved et tegn p.g.a. paritetsfejl. Trykkes på normal start (se store kontrolbord) søger maskinen at rette tegnet og kører videre.

tromle fejl: er tændt hvis der ved læsning fra baggrundslageret er fundet en fejl. Tilkald operatøren.

HP spærret: er tændt hvis HP-knappen er blokeret. Et tryk på Reset slukker lampen.

i-løkke: er tændt hvis der opstår fejl i GIERs adressearitmetik. Tilkald operatøren.

1.9.4.4 yderligere bemærkninger

Under oversættelse af algol-programmer skal KA og KB være slukket. Forklaring på dette gives i =A Manual of GIER-algol 4= side 65.

GIER 2 s lille kontrolbord er udstyret med et krydsfelt, der gør det muligt at ændre valget af ydre udlæse-enheder manuelt. Søjlerne svarer (fra højre mod venstre) til: select(8), select (16), select(32) og select(64).
Rækkerne svarer (fra oven og ned) til: linieskriveren, perforatoren, skrivemaskinen og plotteren.
Hvis en knap er trykket ned svarer det til at select på værdien svarende til søjlen vil medføre udlæsning paa en ydre enhed, der svarer til rækken. Bemærk: flere knapper kan være trykket ned i samme række eller søjle. (se billede 1.9).

1.9.5 STORE KONTROLBORD

1.9.5.1 almindelig beskrivelse

Det store kontrolbord benyttes til direkte indgriben i GIERs registre og lagre. Dette kursus vil dog ikke beskrive hvordan. Kun 3 knapper er af interesse, da de benyttes til start og stop under kørsler.

1.9.5.2 tænding-slukning

Tændes og slukkes sammen med GIERs centralenhed.

1.9.5.3 knapper-lamper (se billede 1.10)

Normal stop: Denne knaps funktion er at standse maskinen under en kørsel, således at den kan startes igen på samme punkt ved tryk på:

Normal start.

Mikro-tempi stop: har samme virkning som Reset.

De andre knapper bør under ingen omstændigheder røres, da de alle kan bryde forstyrrende ind i maskinens kørsel.

1.9.5.4 yderligere bemærkninger

På GIER 1s store kontrolbord sidder den nøgle, der benyttes til at slukke og tænde centralenheden.

For yderligere oplysninger, se: =A Manual of GIER PROGRAMMING= volume 2.

2. PROGRAMMEL

HELP-3 SYSTEMET, GIERs BASISPROGRAMMEL.

Fra teknikernes side er GIER kun forsynet med meget primitive ind- og udlæsefunktioner. Konsekvensen af dette er, at det er ubetinget nødvendigt permanent i maskinen at have et overvågende program, der kan tage sig af de rutinemæssige ind- og udlæsefunktioner.

Et af mange mulige programmer (main help) er indlagt i help-3 systemet. Udover ovennævnte har dette program adskillige andre brugervenlige funktioner.

For at main help kan være i stand til at læse, fortolke og udføre kommandoer (se afsnit 2.2) benytter det sig af en indholdsfortegnelse kaldet kataloget. (I = A Manual of Help-3 = kaldes den catalog).

2.1 kataloget

2.1.1 indhold

Navnene i kataloget er hægtet sammen med en beskrivelse af det, navnet repræsenterer. Help-3 kender beskrivelse af følgende 5 typer:

1) indlæse-enheder, dvs. ydre enheder, hvorfra der kan læses. Normalt findes både strimmellæser (reader) og skrivemaskine (typewriter) beskrevet.

2) udlæse-enheder, dvs. ydre enheder, hvorpå der kan skrives. Normalt findes linieskriver (l), perforator (p) og skrivemaskine (writer) beskrevet.

3) binære programmer, dvs. områder der indeholder direkte udførbare programmer. Der findes et varierende antal beskrevet i kataloget. Exempler er standardprogrammerne algol, edit og ga4. De vigtigste er nærmere beskrevet i afsnit 2.3.

4) dataområder. Et dataområde er et område på baggrundslageret, hvori der står data på en eller anden form. Det er netop dataområder, der benyttes ved put og get i algol-programmer.

5) konstanter Normalt findes kun konstanten date (dags dato).

2.1.2 eksempel

På billede 2.1 er gengivet et eksempel på et katalog og en tilsvarende opdeling af baggrundslageret.

Den venstre del af tegningen er en skematisk gengivelse af baggrundslageret. Baggrundslageret er rent fysisk inddelt i blokke a 40 ord. En sådan blok kaldes en kanal. På tegningen er blokkene opstillet i rækkefølge (fra oven og ned) efter help-3s nummerering, (på GIER 1: 0-319 og på GIER 2: 0-19199).

Blok 0 indeholder det, der i main-help kaldes kanal 0 (se afsnit 2.2.1). De følgende blokke indeholder main-help (se afsnit 2.2) og catalog (se afsnit 2.1).

Til højre på tegningen er vist en forstørret, skematisk gengivelse af kataloget.

I dette tilfælde indeholder kataloget den ydre indlæse-enhed r og den ydre udlæse-enhed 1. Af standardprogrammer findes her algol, edit og ga4 og desuden det binære program limp. Eksempler på dataområder er sl13, mat4, matrixa, free og work.

At sl13 indeholder et oversat algol4 program, mat4 et algol4 program på textform og matrixa (som er reserveret fra et algol4 program med reserve) indeholder tal, kan ikke ses af tegningen.
I praksis vil der kun kunne skelnes mellem de 3 områders indhold ved den brug, man gør af dem.

2.1.3 faste områder, der er tilgængelige for brugerne

free, work og image.

Ved help-3 systemets fødsel bliver baggrundslageret delt i to hovedområder, nemlig området hvori standardprogrammerne ligger og resten, der er benævnet free.

free er den del af det oprindelige frie areal på baggrundslageret, der ifølge kataloget ikke i øjeblikket bruges til permanent lagring af information. Kataloget indeholder altid navnet free samt information om free s aktuelle størrelse. free er fuldt tilgængeligt for alle brugere, der dels kan benytte det som arbejdsareal, dels kan beslaglægge dele af det til lagring af f.ex. program eller data under passende brugerspecificeret navn. Ved reservation optages navnet i kataloget sammen med information om hvor megen plads det reserverede område optager.
free formindskes herved tilsvarende. (Se afsnit 2.3.3., res).

Ofte benyttes free i forbindelse med retteprogrammet edit:

edit, ofree<

Denne kommando medfører at den rettede streng (program, datastrimmel el.l.) lagres i free.
Katalog beskrivelsen af free påvirkes dog ikke, så efterfølges ovennævnte kommando ikke af en kommando som res,<navn>< bevarer free sin størrelse. Som følge heraf vil det lagrede ødelægges næste gang free benyttes som uddataområde.

work er et arbejdsområde på baggrundslageret, som er tilgængeligt for brugerne lige som free. Brugeren skal dog være opmærksom på at work desuden benyttes af standardprogrammerne algol og edit. algol benytter work som arbejdsplads under oversættelse af algol-programmer og slutter med at lægge det oversatte program i work. edit benytter work til midlertidigt lager for rettelser.
Derfor: sålænge et oversat program lagret i work benyttes, er det højst utilrådeligt at benytte sig af edit. Flyt derfor det oversatte program i sikkerhed ved hjælp af standardprogrammet move før edit benyttes, hvis det oversatte program skal bruges senere. (Se afsnit 2.3.5., move).

work er placeret forskelligt på GIER 1 og GIER 2. På GIER 1 er work placeret sammesteds som free og er af variabel størrelse, dvs. work optager så stor en del af free, som der i den aktuelle situation er brug for. Man kan derfor på GIER 1 komme ud for at ødelægge eventuel information i free under oversættelse af et algol-program. På GIER 2 er work derimod et selvstændigt område.

image er et område på baggrundslageret hvor main help gemmer et billede af hurtiglageret. I visse situationer vil et program kunne genstartes, hvis det under kørslen er blevet afbrudt ved tryk på HP-knappen. For nærmere oplysninger se =A Manual of Help-3= side 13 og 35.

image er ligesom work placeret forskelligt på GIER 1 og GIER 2, dvs. det er et selvstændigt område på GIER 2 medens det på GIER 1 er placeret i den sidste del af free.
På GIER 1 vil brugeren ofte modtage udskriften image efter tryk på HP-knappen eller efter kørsel af et algol-program. Dette skyldes at work og image optager den samme del af baggrundslageret.
Med udskriften image spørger main help om
1) man ønsker at gemme information nødvendig for genstart, hvorved indholdet af work ændres
   eller
2) man ønsker at work skal lades uforandret, hvorved genstart umuliggøres.

Brugeren kan nu taste sit svar på skrivemaskinen:

< : hurtiglageret (ferritlageret) gemmes i image, work ødelægges.

mellemslag: work lades uforandret.

alt andet : spørgsmålet (image) gentages.

Trykkes på HP-knappen mens et standardprogram (edit, algol e.l.) kører, vil main help ikke gemme hurtiglageret, så al information om det kørende program ødelægges.

2.2 MAIN-HELP

I help-3 findes 2 vigtige begreber:

1) help-3s løbende indlæse-medium

Help-3s løbende indlæse-medium er det sted, hvorfra help-3 henter sin næste kommando.
Løbende indlæse-medium kan være:

1) en ydre indlæse enhed
2) et dataområde.

I tilfælde 2 vil help-3 fortolke indholdet som værende på textform (se beskrivelsen i =A manual of Help-3=, side 10).

Løbende indlæse-medium er fra starten skrivemaskinen.

2) help-3s valgte udlæse-medium

Help-3s valgte udlæse-medium er den ydre enhed (udlæse enhed), hvortil standardprogrammerne sender deres uddata. Standard udlæse medium er fra starten forskelligt på forskellige installationer:

på GIER 1: skrivemaskinen (w)
på GIER 2: linieskriveren (1).

Help-3s standard udlæse-medium bør ikke forveksles med alarm out, som er den ydre enhed, alle fejludskrifter fra help-3 bliver udskrevet på.

2.2.1 HP-KNAP (se billede 1.8)

2.2.1.1 funktion

I help-3 systemet er HP-knappens funktion koblet sammen med main help. Ved tryk på knappen udløses følgende funktioner:

1) Det undersøges om main help er i orden.
2) Hvis det ikke er tilfældet skrives fejludskriften SUM (se afsnit 2.2.3, udskrifter fra main help).)
   Hvis alt er i orden udskrives dato og anden information, (se afsnit 1.9.4.3), og main help begynder at læse fra skrivemaskinen - løbende indlæse-medium.

2.2.1.2 brug

Ved hvert HP-tryk udløses forskellige aktioner, der tager nogle sekunder hver gang. Knappen bør kun bruges i to tilfælde:

1) Man vil afbryde et standardprogram (dog aldrig standardprogrammet compress) eller et algolprogram medens det kører.
2) Man aner ikke hvilken tilstand GIER står i og vil gerne i en veldefineret udgangssituation.

Det er unødvendigt at benytte knappen hver gang GIER har udført en kommando.

2.2.1.3 situationen efter HP-tryk

Efter et HP-tryk er:

help-3s løbende indlæse-medium: t (skrivemaskinen)

help-3s standard udlæse-medium: w (skrivemaskinen) på GIER 1
l (linieskriveren) på GIER 2,

og help-3 starter læsningen fra løbende indlæse-medium.

2.2.2 EN KOMMANDO TIL MAIN-HELP

En kommando (benævnes help-3 information i =A Manual of Help-3=) kan indeholde ændring af help-3s løbende ind/udlæse medier, kald af et standardprogram samt eventuelle parametre til standardprogrammet. Kommandoen afsluttes med et < (udtales hak). Main-help begynder først at fortolke og udføre en kommando, når < er læst.

2.2.2.1 fortolkning

Kommandoen fortolkes ved hjælp af kataloget (se afsnit 2.1). Startende med det første element i kommandoen, vil fortolkningen foregå således:

1) hvis navnet beskriver en udlæse-enhed vil denne enhed blive valgt som help-3s valgte udlæse-medium og main-help fortsætter fortolkningen af kommandoen.

2) hvis navnet beskriver en indlæse-enhed (eller et område på baggrundslageret, der indeholder text) vil enheden blive valgt som help-3s løbende indlæse-medium og fortolkningen af kommandoen fortsættes.

3) hvis navnet beskriver et standardprogram vil resten af kommandoen blive betragtet som parametre til dette program. Hvorledes disse parametre fortolkes afhænger nu af det pågældende program. Dette hentes frem fra baggrundslageret til hurtiglageret og overtager fortolkningen fra main-help.

4) hvis terminatoren < mødes vil main-help starte med at læse en kommando fra det sidst valgte løbende indlæse-medium.
Dette gælder dog ikke for standardprogrammerne run og edit, som altid vil vælge skrivemaskinen som løbende indlæse-medium.

5) hvis det navn, der mødes ikke findes beskrevet i kataloget vil main-help give en alarmmeddelelse.

2.2.2.2 en kommandos form (se billede 2.2 punkt 3.1)

Den tilladte form for en kommando anskueliggøres ved at sammenholde eksemplerne på billede 2.3 med det udsnit af =A Manual of Help-3= der findes på billede 2.2.

2.2.3 UDSKRIFTER FRA MAIN-HELP

Dette afsnit vil kun behandle udskrifter fra main-help, der er relevante for dette brugerkursus. Ved andre udskrifter henvises til beskrivelserne under de enkelte standardprogrammer samt til =A Manual of Help-3= kapitel 8, side 15.
Det bemærkes at disse udskrifter fremkommer på det udlæse-medium, der i help-3 kaldes alarm-out.

udskrift	farve	forklaring
1) annul	rødt	Den løbende linie af en kommando er blevet annulleret; dvs. at netop den linie, der skrives på er blevet udeladt. (se bemærkningen om symbolet å på billede 2.2, midten).
2) catalog	rødt	Kataloget er ødelagt. På GIER 2: underret operatøren. På GIER 1: læg strimmel mærket BASIC HELP i strimmellæseren og tast ≤ to gange.
3) full	rødt	En kommando er for lang eller et tal for stort.
4) image	sort	Se afsnit 2.1.2 under forklaringen af området image.
5) overflow	rødt	svarer til fejludskriften spill fra algol4. (Se =A Manual of GIER algol4=). Denne udskrift bør ikke forekomme under normale kørsler - UNDERRET OPERATØREN.
6) param	rødt	utilladelig rækkefølge af parametre i en kommando.
7) sum	rødt	ved kald af et hjælpeprogram viser en simpel kontrol, at området ikke er i overensstemmelse med beskrivelsen i kataloget. Situationen kan i mange tilfælde på GIER 1 reddes ved indlæsning af systemstrimmel =add<navn>= med r≤, (se afsnit 2.4) eller indlæsning af MAIN-HELP. Opstår fejlen på GIER 2, underret operatøren.
8) SUM	rødt	Main-help er ødelagt. På GIER 2: underret operatøren. På GIER 1: sæt skiftekontakten i skabet på fri (se afsnit 1.4.3). Indlæs strimmel mærket BASIC HELP ved at taste mellemslag to gange. Sæt skiftekontakten på spærret. Kørslen kan nu fortsættes.
9) syntax	rødt	Syntaktisk fejl i en kommando. Kommandoen udføres ikke.
10) undef	rødt	Et navn, der forekommer i kommandoen findes ikke beskrevet i kataloget.
11) kind	rødt	Et navn beskriver hverken en ydre enhed eller et område på baggrundslageret.

2.3 DE VIGTIGSTE STANDARDPROGRAMMER

Beskrivelserne af standardprogrammerne er ikke fuldstændige. En fuldstændig detaljeret beskrivelse findes i =A Manual of Help 3=. Oplysning om andre faste programmer kan fås hos operatøren.

2.3.1 algol (for yderligere oplysninger se =A Manual of GIER algol 4=

2.3.1.1 funktion

Standardprogrammet algol kalder en algol4 oversætter på en sådan måde, at oversætteren læser fra help-3s løbende indlæse-medium og fortolker det læste som en algol4-programtext. Hvis et indlæsemedium ikke er specificeret i kommandoen, vil algol altid starte indlæsningen fra strimmellæseren .

2.3.1.2 parametre (se =A Maunual of Help-3= side 25)

KALD: ... algol, <parameter liste> <

PARAMETRE: <parameter liste>::=<tom>|<parameter><parameter liste>

idet: <parameter>::=<navn på udlæse-medium>|
<navn på oversætter>|s|i|d|n|<tal>

BETYDNING AF PARAMETRENE:

<tom>: oversættelsen sker med oversætteren ga4 på GIER 1 og ga4b på GIER 2. Der genereres ingen udskrift af program og det oversatte algol4-program vil blive kørt med index check. (se =A Manual of GIER Algol4, side 78, index)

<navn på udlæse-medium>: skal være en ydre enhed. Mediet benyttes ved eventuel udskrift af program. Hvis valg af udlæse-medium udelades benyttes standard udlæse-medium.

<navn på oversætter>: skal beskrive navnet på en algol oversætter (f.ex. ga4) eller navnet på et indlæsemedium. (for transient oversætter, se afsnit 2.5.3 og =A Manual of Help-3= side 25). Udelades navnet bruges navnet ga4 på GIER 1 og ga4b på GIER 2. (se afsnit 2.3.1.4).

n: Der vil intet (no) index-check blive foretaget under køslen af programmet. (se =A Manual of GIER Algol4=, side 78).

d: Benyttes denne parameter vil oversættelsestiden for større algol programmer på GIER 2 i mange tilfælde blive væsentlig mindre.

<tal>: Hver <tal>te linie i programmet udskrives på standard eller valgte udlæse-medium. Hvis <tal>=0 eller udeladt gives ingen linieudskrift.

s og i: For oplysninger om disse specielle parametre henvises til =A Manual of Help-3=, side 25, idet de almindeligvis er uden interesse.

2.3.1.3 udskrifter fra algol

Hvis det indlæste algol-4 program er syntaktisk korrekt vil algol efter oversættelsen give meddelelsen

ok

på det valgte eller standard udlæse-medium (se afsnit 2.3.1.2), og main-help vil fortsætte med at læse fra help-3s løbende indlæse-medium.
Hvis det ikke er tilfældet vil skrivemaskinen blive valgt som løbende indlæse medium.

Fejludskrifter fra oversætteren vil blive udskrevet på alarm-out (se afsnit 2.2.2). Angående betydningen af disse henvises til =A Manual of GIER Algol4= (se indholdsfortegnelse under fejludskriftens navn).

2.3.1.4 bemærkninger

Efter oversættelsen ligger det oversatte algol4-program (på binær kode form, se =A Manual of GIER Programming=, vol.1) i området work.

Ved oversættelse med ga4b (f.ex. pip, algol, ga4b, 10<) vil det oversatte program blive forsynet med passende maskininstruktioner, således at bufferen (se afsnit 1.5) benyttes til lagring af arrayelementer. Ved oversættelse med ga4 benyttes bufferen ikke.

2.3.1.5 eksempler

1) algol<

Dette kald af algol vil bevirke at en strimmel vil blive læst, og fortolket som et algol4-program, og er det syntaktisk korrekt oversættes det med oversætteren ga4 på GIER 1 og ga4b på GIER 2.
Meddelelser om syntaktiske fejl vil blive udskrevet på help-3s standard udlæse-medium, dvs. skrivemaskinen på GIER 1 og linieskriveren på GIER 2.

2) free,algol,1,1<

Dette kald vil bevirke at indholdet af baggrundsområdet free vil blive fortolket som algol-program, idet free nu er valgt som help3s løbende indlæse-medium.
Parameteren 1 vælger udlæse-enhed for standardprogrammet algol. Denne kan dog udelades på GIER 2, da den der er standard.
Parameteren 1 vil forårsage at hver linie i algol-programmet vil blive udskrevet (og nummereret) på algols udlæse-enhed, i dette tilfælde linieskriveren.
Meddelelser om syntaktiske fejl vil dog blive udskrevet på help-3s valgte udlæse-medium (se eksempel 1).

3) pab5,w,algol,1,10<

Som man ser indeholder denne kommando to navne før kaldet af standardprogrammet algol.
Det første navn, pab5, ændrer help-3s løbende indlæse-medium til at være et område på baggrundslageret ved navn pab5.
Herfra vil algol altså læse.
Navnet w vælger skrivemaskinen (writer) som help-3s udlæse-medium, dvs. meddelelser om syntaktiske fejl vil blive udskrevet på skrivemaskinen.
Parametrene 1 og 10 vil bevirke at hver 10. linie af programmet vil blive udskrevet på linieskriveren.

4) GIER 2: algol,ga4,n<

Med strimmellæseren som indlæse-medium vil blive læst og i tilfælde af at programmet er syntaktisk korrekt, oversat et algol4-program, dog med den specielle version af algoloversætteren, der ikke benytter bufferen (se afsnit 2.3.1.4)
Programmet vil blive kørt uden index-check på array-elementer (n).

2.3.2 run

2.3.2.1 funktion

run er et lille standardprogram, der tager et dataområde og undersøger om det er et oversat algol4 program.
Er dette tilfældet starter run udførelsen af dette program.

2.3.2.2 parametre (se =A Manual of Help-3=, side 41)

KALD: run< og run,<navn><

PARAMETRE: <navn> er det dataområde, hvorfra run henter det oversatte algol4-program.
Er parameterlisten tom, henter run programmet i dataområdet work (se afsnit 2.3.1.4).

2.3.2.3 udskrifter fra run

Hvis dataområdet (enten work eller det der er angivet ved <navn> i parameterlisten) ikke indeholder et oversat algol4-program udskrives på main-helps alarm-out meddelelsen:

not present

2.3.2.4 bemærkninger

Efter udførelsen af algol4-programmet overgives kontrollen til main-help med skrivemaskinen som løbende indlæse-medium. Her skal dog undtages de tilfælde, hvor algol4-programmet løber i en uendelig løkke.

2.3.2.5 eksempler

1) run<

(se afsnit 2.3.2.2, parametre).

2) run,mb3<

Standardprogrammet run vil sørge for kørsel af det oversatte algolprogram, der ligger på baggrundslageret under navnet mb3.

2.3.3 edit

2.3.3.1 funktion

edit er et retteprogram, der kan benyttes til at rette textstrenge, dvs. en følge af symboler, der skal være repræsenteret i flexowriterkode (se appendix), f.ex. et algolprogram. En textstreng bør være afsluttet med en STOP CODE (se flexowriterkode, appendix).

edit virker i to faser:

1) edit læser en rettelsesliste fra løbende indlæse-medium og lagrer den i baggrundslagerområdet work. Som slut på fase 1 udskrives antal læste rettelser på skrivemaskinen.
Skal den textstreng, der ønskes rettet indlæses fra strimmellæseren skal der nu tastes et mellemslag på skrivemaskinen.

2) edit læser nu den textstreng, der skal rettes. Rettelserne udføres en for en under indlæsningen. Som slut på fase 2 udskrives det antal rettelser, edit har udført. Herefter venter maskinen på kommando til main-help.

2.3.3.2 parametre

KALD: edit,<indlæse-medium>,<udlæse-medium>,<andre parametre>

PARAMETRE:

<indlæse-medium>: i<navn>

<navn> beskriver det indlæse-medium hvorfra den textstreng der skal rettes hentes.
Er parameteren tom vælges strimmellæseren som indlæse-medium.

<udlæse-medium>: o<navn>

<navn> beskriver det udlæse-medium hvortil den rettede textstreng bliver sendt.
Er parameteren tom vælges perforatoren som udlæse-medium.

<andre parametre>: se =A Manual of Help-3= side 32 og 33.

2.3.3.3 rettelseslisten

rettelserne som indlæses af edit har form af enkeltrettelser, der skal komme i den rækkefølge, hvori de forekommer i den textstreng, der skal rettes.

En enkeltrettelse består af textstrenge (evt. tomme) adskilt med . :

<kopier til og med>.<indsæt>.<overspring til og med>.

Listen af rettelser afsluttes med et slutmærke (STOPCODE eller å). Da å således har speciel betydning for edit er det bekvemmest, dog ikke nødvendigt (se =A Manual of Help-3= side 33, <changes>) at undlade at benytte dette symbol i textstrenge. Bemærk at koden for linieskift og mellemslag er blinde symboler i strengene: <kopier til og med> og <overspring til og med>.

Bemærk endvidere at en rettelsesliste kan bestå udelukkende af slutmærke, hvilket vil medføre en direkte kopiering af textstrengen.

EKSEMPLER PÅ RETTELSESLISTER:

1)
```
                    .Jeg ..<slutmærke>
vil ændre textstrengen:
          gik mig over sø og land<slutmærke>
til:
          Jeg gik mig over sø og land<slutmærke>
```

2)
```
rettelseslisten:
                    <.
                    be.e.
                    (¢.<..
                    read..ø.
                    <slutmærke>
vil ændre textstrengen:
                    algol<
                    egin integer i,j;
                    select(6);
                    writetext(¢øllebrød¢);
                    i:= readøinteger;
                    ...
                    end t<
                    <slutmærke>
til:
                    algol<
                    begin integer i,j;
                    select(8);
                    writetext(¢<øllebrød¢);
                    i:= readinteger;
                    ...
                    end t<
                    <slutmærke>
```

2.3.3.4 udskrifter fra edit

meddelelser:
edit udskriver antal enkeltrettelser, det finder i rettelseslisten, <tal1> og efter udførelsen af rettelserne udskrives hvor mange enkeltrettelser, der er udført, <tal2>. Der kan nu opstå følgende situationer:

<tal1> = <tal2>: alle rettelserne i rettelseslisten er fundet og udført.

<tal1> > <tal2>: edit har kun kunnet fuldføre <tal2> enkeltrettelser i den tekststreng der skulle rettes.

<tal1> eller <tal2> udskrives ikke. Dette betyder at edit ikke har mødt et slutmærke. Situationen kan reddes ved tryk på mikrotempi-stop og normalstart på det store kontrolbord. (se billede 1.10).

specielle alarmudskrifter:

char: edit har mødt et symbol der ikke benyttes af flexowriterkoden. Symbolet overspringes og edit fortsætter.

full: længden af en tekststreng af typen <kopier til og med> eller <overspring til og med> overskrider 400 symboler, eller
rettelseslisten er for lang til at være i baggrundsområdet work.

overlap: forekommer i to tilfælde:
1) hvis den rettede tekststreng overskrider pladsen for det udlæse-medium, der er specificeret i parameterlisten. Dette forekommer naturligvis kun hvis udlæsemediet er et område på baggrundslageret.
2) hvis ind- og udlæse-medium er det samme område på baggrundslageret og indtext indhenter udtext.

parity: der er paritetsfejl i et symbol på rettelseslisten eller i den tekststreng der skal rettes. edit søger at rette symbolet og fortsætter. Samtidig udskrives de følgende par linier på skrivemaskinen.

termination: antal . i rettelseslisten er ikke et multiplum af 3, eller en enkeltrettelse er ikke gjort færdig.

2.3.3.5 eksempler

1) edit,ofree<

Rettelseslisten læses fra skrivemaskinen. Efter at have tastet et slutmærke, å, indlæses den textstreng, der skal rettes fra strimmellæseren. Den rettede textstreng lagres i free.

Det må indskærpes brugerne at undgå at benytte skrivemaskinen som indlæsemedium for rettelseslister, da dette optager GIER i unødvendig lang tid.

2) r,edit,iebslut,ofree<

Rettelseslisten læses fra strimmellæseren. Den textstreng, der skal rettes læses fra et område på baggrundslageret med navnet ebslut og den rettede textstreng udlæses til free.

3) ret,edit,imig<

Rettelseslisten læses fra et område på baggrundslageret med navnet ret. Den textstreng, der skal rettes læses fra et område med navnet mig og den rettede ttxtstreng udlæses på perforatoren.

4) r,edit,113,n<

Kan benyttes hvis en textstreng indeholder å, der ikke ønskes brugt som slutmærke.
Parametrene 113,n (lad flexowritersymbolet 13 = å være normalsymbol) vil medføre at å ikke betragtes som slutmærke.
Hvis man ønsker at slette et bestemt symbol i textstrengen benyttes parametrene

1<symbolværdi>,s

eller man ønsker et symbol skal betragtes som blindsymbol:

1,<symbolværdi>,b

BEMÆRK at det er tidsbesparende at lade sin rettelsesliste (på strimmel eller i et område på baggrundslageret) starte med kommandoen, således at det kun er nødvendigt at taste r< eller områdets navn efterfulgt af et < på skrivemaskinen.

2.3.4 res (se billede 2.4)

2.3.4.1 funktion

res er et standardprogram, der kan indføre beskrivelse af et nyt dataområde i kataloget. Dette dataområde vil samtidig optage et stykke af free (se afsnit 2.1.2) som derfor mindskes tilsvarende.

2.3.4.2 parametre (se =A Manual of Help 3= side 40)

KALD: res,<længde><bits><navn><

PARAMETRE:

<længde>: Hvis <længde> er et tal>0 vil det dataområde, der reserveres optage <længde> x 40 ord (en blok er på 40 ord) af dataområdet free.
Hvis <længde> er tom, vil res benytte en intern variabel (booked) som længde. Dette vil kun have mening hvis det antal blokke, man ønsker at reservere som dataområde i forvejen er overført til free ved hjælp af standardprogrammet edit eller move, idet booked herved vil blive sat til den rigtige størrelse.

<bits>: For nærmere oplysning om denne parameter henvises til =A Manual of Help 3= side 40, idet den almindeligvis vil være uden interesse og derfor tom.

<navn>: Her skrives det navn, under hvilket man ønsker at reservere sit dataområde. (Navnet kan indeholde bogstaver, cifre samt - og . , se =A Manual of Help 3= side 7 og side 50).

2.3.4.3 udskrifter fra res

1) full: kataloget er allerede fyldt op.

2) length: området, man vil reservere går udover den tilladte størrelse, eller free er for lille.

3) length≤0: området, man vil reservere har længde ≤ 0.

4) name: navnet er allerede i kataloget.

5) value: en numerisk parameter har en ikke-tilladt værdi.

2.3.4.4 bemærkninger

Har man fundet ud af hvordan res benyttes er det VIGTIGT at undersøge funktionen af standardprogrammet clear.

2.3.4.5 eksempler

1) res,thj1<

Dette kald vil bevirke at den information, der evt. er lagret i området free vil blive reserveret under navnet thj1, dvs. navnet thj1 vil blive optaget i kataloget sammen med information om områdets størrelse. Samtidig vil området free formindskes tilsvarende. Dette kald vil ofte blive benyttet, når man ved hjælp af edit har lagret en information i free og ønsker at bevare denne information, men ikke ved hvor mange blokke den fylder.

2) res,20,nb7m<

20 blokke a 40 ord vil blive taget fra free og reserveret, dvs. optaget i kataloget under navnet nb7m. Herefter kan man f.ex. med retteprogrammet edit lagre information direkte i nb7m, dvs. man kan bruge kommandoen: edit,onb7m<. Her må man dog i forvejen sikre sig at den information man ønsker at lagre kan være i nb7m. standardprogrammet list kan benyttes til at få information om et områdes størrelse (se afsnit 2.3.7) Desuden kan oplyses at 60 cm strimmel svarer til 240 tegn, 1 blok.

2.3.5 clear (se billede 2.5 og billede 2.6)

2.3.5.1 funktion

Standardprogrammet clear fjerner fra kataloget beskrivelsen af dataområdet med det navn, der angives som parameter, forudsat at det er et almindeligt reserveret område.

2.3.5.2 parametre (se =A Manual of Help 3= side 31)

KALD: clear, <navn><

PARAMETER, <navn>: navnet på det pågældende område, bemærk her kan kun stå eet navn.

2.3.5.3 udskrifter fra clear

1) name: navnet findes ikke beskrevet i kataloget.

2) no clear: navnet er free, work eller et ikke-reserveret område.

2.3.5.4 bemærkninger

HUSK altid at sørge for at områder, der ikke mere er aktuelle bliver slettet af kataloget med clear.

2.3.5.5 eksempel

clear,pf0<

Dette kald vil bevirke at navnet pf0 fjernes fra kataloget. Hvis pf0 er det sidste område, der er reserveret med res, vil free blive udvidet med den plads pf0 har optaget. Er dette ikke tilfældet vil der blot blive efterladt et hul i baggrundslageret. (se 2.3.7, compress).

2.3.6 move

2.3.6.1 funktion

move flytter information fra et område til et andet og sætter -booked-..

2.3.6.2 parametre (for yderligere oplysninger, se =A Manual of Help 3= side 36).

KALD: move,<navn1><navn2><

PARAMETRE: <navn1>: det dataområde, informationen flyttes fra.
<navn2>: det dataområde, informationen flyttes til.

2.3.6.3 udskrifter fra move

overlap: move undersøger først om det område, informationen skal flyttes til er stort nok til at indeholde denne. Er det ikke det fremkommer udskriften overlap og move træder ikke i funktion.

2.3.6.4 eksempel

move,work,free<

Den information, der ligger i work vil blive flyttet til free. Benyttes ofte når man ønsker at gemme et oversat algol-program, dvs. indeholder work et oversat algol-program og efterfølges ovennævnte kommando af kommandoen:

res,uf<

kan man herefter, når man ønsker at køre programmet blot taste:

run,uf<

2.3.7 check,list

2.3.7.1 funktion

check og list har næsten samme funktion. De undersøger om et givet navn findes i katalogett Hvis det specificeres i kommandoen kan disse programmer yderligere udskrive beskrivelser, som de findes i kataloget.

check har en funktion som list ikke har, idet check kan udføre en simpel kontrol af dataområder/programmer såfremt der er givet besked i beskrivelsen i kataloget af pågældende dataområde/program. (se =A Manual of Help 3= Appendix A)

2.3.7.2 parametre (se =A Manual of Help 3= side 30 og 35)

KALD: check, <form><navneliste><

list, <form><navneliste><

PARAMETRE: <form>
- tom: de pågældende beskrivelser i kataloget udskrives ikke.
- a: de pågældende beskrivelser i kataloget udskrives i fuldt omfang.
- n: kun type og navn udskrives.

<navneliste>
- tom: hele kataloget behandles af check/list, afhængig af <form> og for checks vedkommende desuden af kravene i beskrivelsen.
- <navne>: hvis det/de navne, der skrives her (adskilt med komma) forekommer i kataloget behandles det/de af check/list på samme måde som nævnt ovenfor.

2.3.7.3 udskrifter fra check/list

name: et af navnene i <navneliste> findes ikke beskrevet i kataloget.

sum: (sort) kun fra check: den simple kontrol viser fejl på et dataområde/program. check fortsætter.

2.3.7.4 bemærkninger

Udskriften sker på løbende udlæse-medium.

2.3.7.5 eksempler

1) w,list,a,protokoloversat,protokol<

vil medføre udskriften på skrivemaskinen:

```
0,   84, 4393, r
     protokoloversat
0,   92, 4478, r
     protokol
```

der skal fortolkes således:

- 0: angiver at navnene beskriver områder på baggrundslageret.
- 84 og 92: angiver hvor mange blokke a 40 ord områderne fylder.
- 4394 og 4478: angiver nummeret på den kanal hvor det pågældende område begynder.
- r: angiver at området er reserveret enten med standardprogrammet res eller med gier-proceduren reserve. Området kan derfor fjernes med standardprogrammet clear.

2) l,check,n,r,l,date<

vil medføre udskriften på linieskriveren:

```
4,   date
7,   r
6,   l
```

4 angiver at navnet date beskriver en konstant.
7 angiver at navnet r beskriver et indlæse-medium
6 angiver at navnet l beskriver et udlæse-medium.

3) check,a,ga4<

vil medføre udskriften på help-3s valgte udlæse-medium:

```
0,   171,  1226, p i s  0.0.0.0
     ga4,  10.194.194.0
```

Angående fortolkningen af de tre første tal, se eksempel 1.

- p angiver at områdets indhold er et help-3 program.
- i angiver at hurtiglageret er spærret medens programmet kører, dvs. hurtiglageret gemmes ikke ved tryk på HP-knap når dette program kører.
- s angiver at der sumkontrol af programmet.

0.0.0.0 normalt uden interesse for brugeren.
10.194.194.0 katalogets oplysning til main-help om hvorledes dette program sættes i funktion.

2.3.8 compress

2.3.8.1 funktion

compress flytter samtlige reserverede områder, således at -huller- efterladt af standardprogrammet clear forsvinder (se billede 2.7). Samtidig ajourføres kataloget of free øges tilsvarende. compress kan samtidig udskrive det ajourførte katalog, dog er den udskrevne beskrivelse af free den, der var i det gamle katalog.

2.3.8.2 parametre

KALD: compress,<form><

PARAMETER: <form> : se afsnit 2.3.7.2, parametre til check.

2.3.8.3 udskrifter

Når compress er færdig udskrives:

words in cat, free size = <m>,<n>

hvor

<m> angiver antallet af brugte ord i det ajourførte katalog

og

<n> angiver free s endelige størrelse.

2.3.8.4 bemærkninger

compress må ikke afbrydes medens det arbejder, idet kataloget ændres undervejs og derfor ikke behøver at svare til baggrundslagerets faktiske udseende. Af samme grund spærres HP-knappen under udførelse af compress.
compress findes ikke på GIER 2, idet operatøren dagligt foretager sletninger i kataloget i overensstemmelse med kørselsreglerne for GIER 2 og derefter udfører kald af compress. Ved akut eller kronisk pladsmangel bedes man henvende sig til operatøren, IKKE SLETTE SELV.
På GIER 1 er der ingen restriktioner for sletning af områder.

2.3.8.5 eksempler

compress<

Denne kommando udfører en simpel compress og udskriver på standard udlæse-mediet katalogets og free s størrelse.

compress,a<

udskriver desuden undervejs det ajourførte katalog.

2.3.9 Andre standardprogrammer.

Følgende afsnit er en oversigt over de øvrige standardprogrammer i HELP-3 systemet. Yderligere detaljerede oplysninger findes i =A Manual of Help-3=.

BININ	indlæser en binær strimmel, hullet af programmet binout, til et område på baggrundslageret.
BINOUT	huller et dataområde ud på binær form dvs. en stærkt koncentreret form, beregnet til hurtig indlæsning med binin eller det primitive indlæseprogram. I modsætning til edit kan alle dataområder hulles ud af binout.
EXIT	bruges ved genstart af programmer.
OUTPARAM	bruges i forbindelse med binout. Huller parameterlisten på tekstform.
PAIR	sammenligner to dataområder og udskriver forskellene.
PRINT	kan udskrive baggrundslageområder (og bufferen) på en givet form.
SET	indsætter en indgang i kataloget. Modsvarer res.
SLIP	GIERs assembler (Symbolic Language Input Program).
START	kan 1) sætte datoen i kataloget. 2) initiallisere baggrundslagerområder.

SYSTEMSTRIMLER GIER 1

NR	NAVN	INDHOLD
1	old → new	kanal 0 (help3). Bruges naar hjælp er i maskinen og overgang til help3 ønskes.
3*	Basic help 3	mainhelp, binin, exit, start, edit, set, res og clear.
4*	Basic help3 + slip, check, print binout, move	Som ovenfor + slip, check, list, setsum, compress, print, pair, outparam, binout, move.
5*	Gier Algol 4	ga4, move, check, run, copy, algol.
6	ga4 transient (passes)	} ga4. Under oversættelse tages de
7	do. GPA + pass1	} enkelte passager fra strimmellæseren.
8	add algol	algol.
9	add binout	binout, outparam.
10	add check	check, list, setsum, compress.
11	add slip	slip.
12	add print	print, pair.
13	add move	move.
14	add run	run.
15	add algol3	gier algol 3 oversætteren.
16	algol3 transient	} gier algol3 oversætteren.
17	algol3 D (passes)	} Virker som ga4 transient.
18	add run17, 17	17 oversætteren og run17.
19	add imp4	imp4-oversætteren.
20	head Kompud	head Kompud.
21	check bin	check bin.
22	punch head Kompud	punch head Kompud.
23	new → old	Kanal 0 (hjælp). Bruges naar help3 er i maskinen og overgang til hjælp ønskes.
24	plot+charset bin	plotter procedurerne og tegnsættet charset.

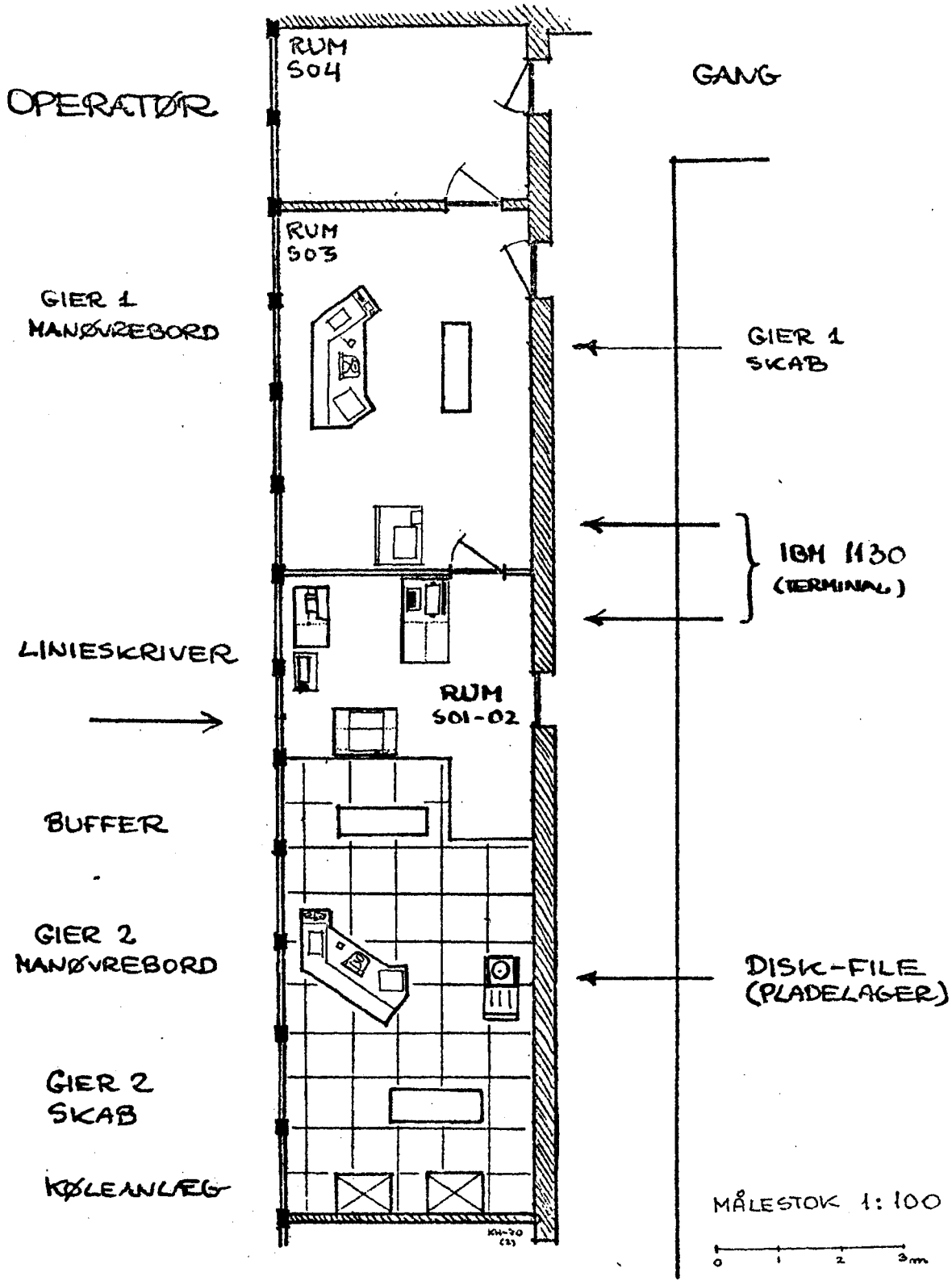
*: Kataloget initialiseres, d.v.s. kun indgangene som er nævnt i højre kolonne vil findes efter indlæsningen.

BILLEDE 1.1. DEN GODE BRUGER.

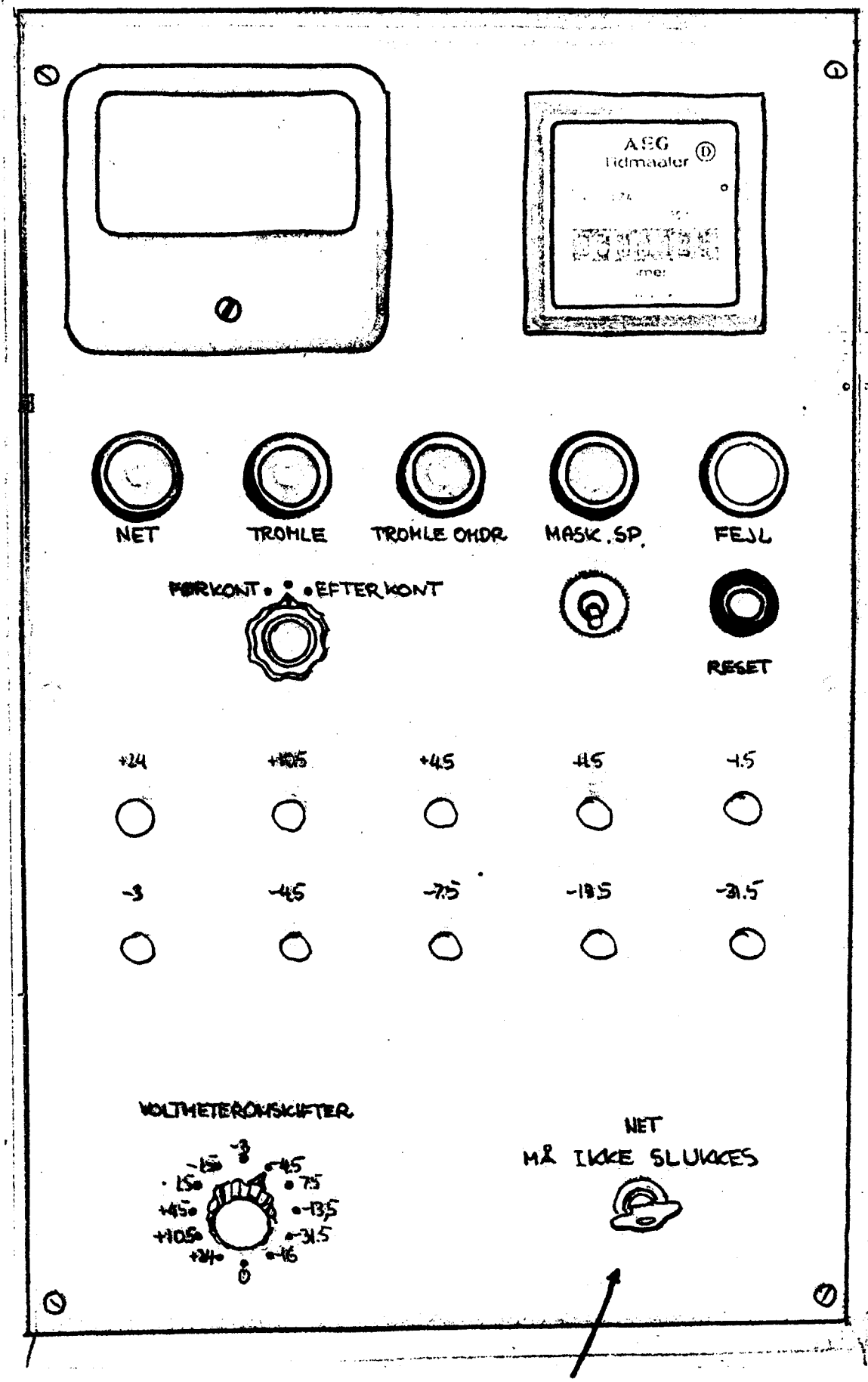
En glad og tilfreds bruger i en ryddelig maskinstue,
(hvor det er muligt at finde de systemstrimler, man skal bruge!)

BILLEDE 1.2

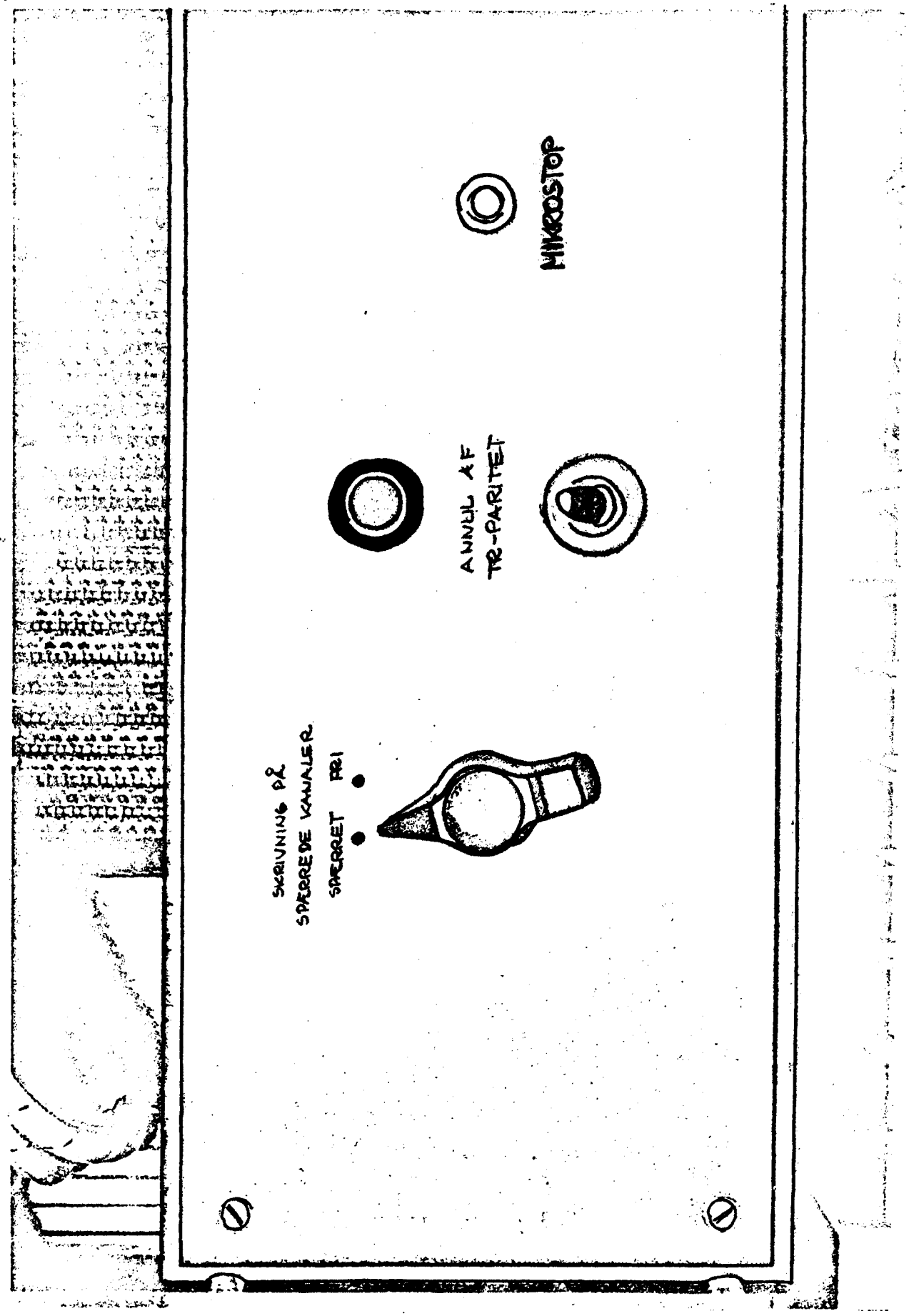
GRUNDPLAN OVER GIER-RUMMENE



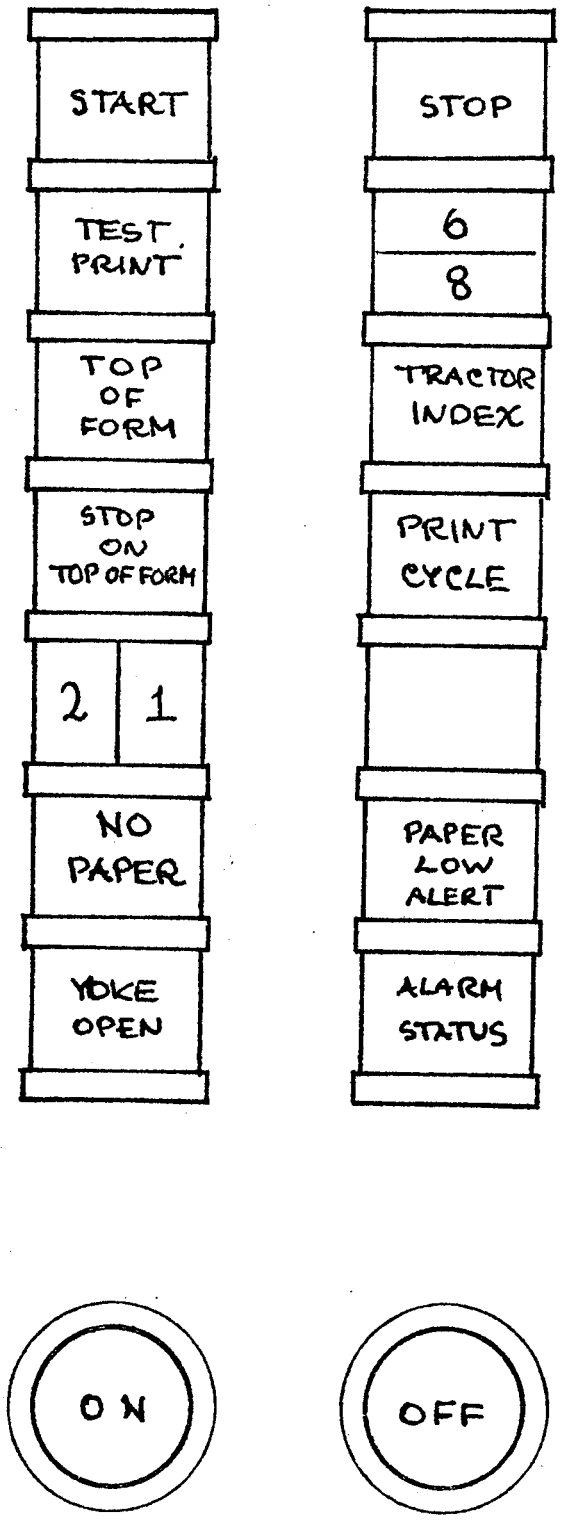
BILLEDE 1.3


AEG
Tidmaaler
NET
TROHLE
TROHLE OHDR
MASK. SP.
FEJL
FØRKONT
EFTERKONT
REGET
+24
+10.5
+4.5
-1.5
-4.5
-3
-4.5
-7.5
-13.5
-31.5
VOLTMETEROMSKIFTER
-3
-4.5
-7.5
-13.5
-31.5
-16
0
+24
+10.5
+4.5
-1.5
NET
MÅ IKKE SLUKKES
NØGLE TIL TÆNDING AF GIER 2.

BILLEDE 1.4 KNAPPER I GIER 1 - SKABET.

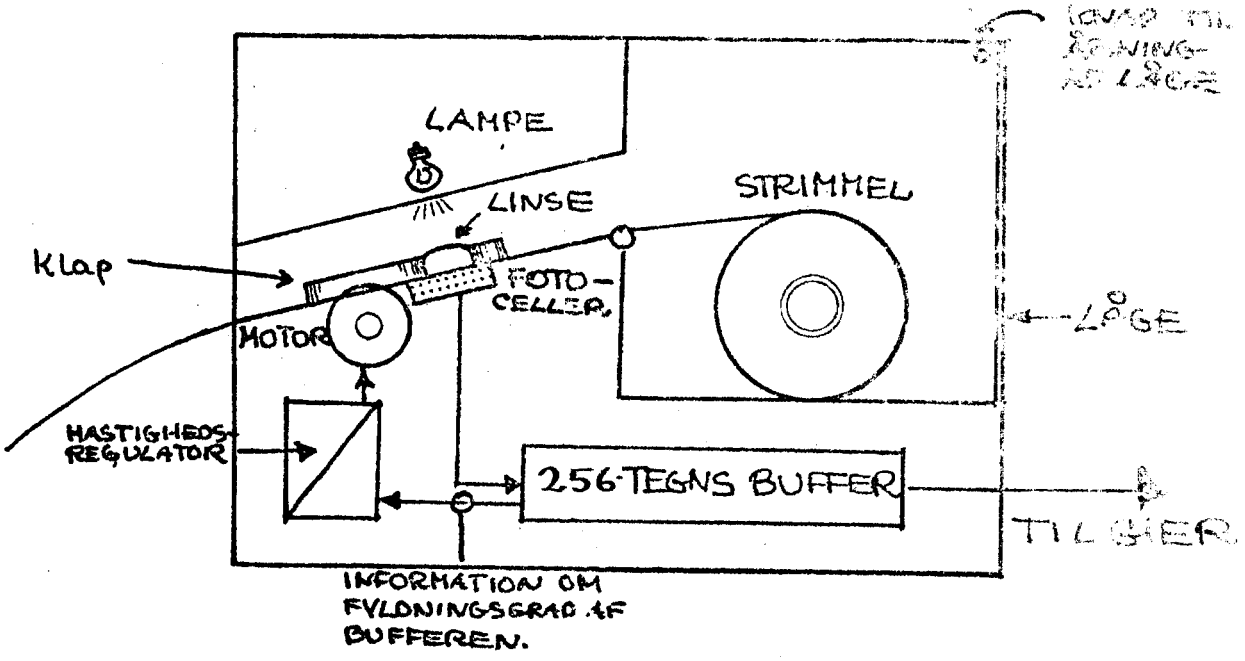


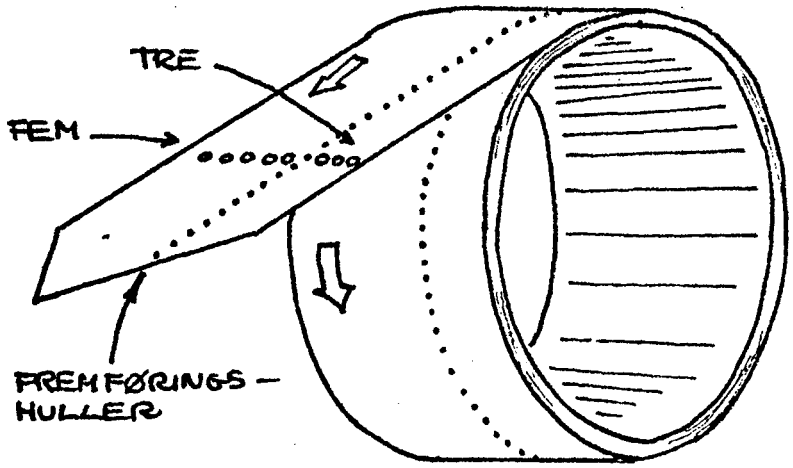
BILLEDE 1.5 KNAPPER OG LAMPER PÅ LINIEBKRIVEREN.



BILLEDE 1.6

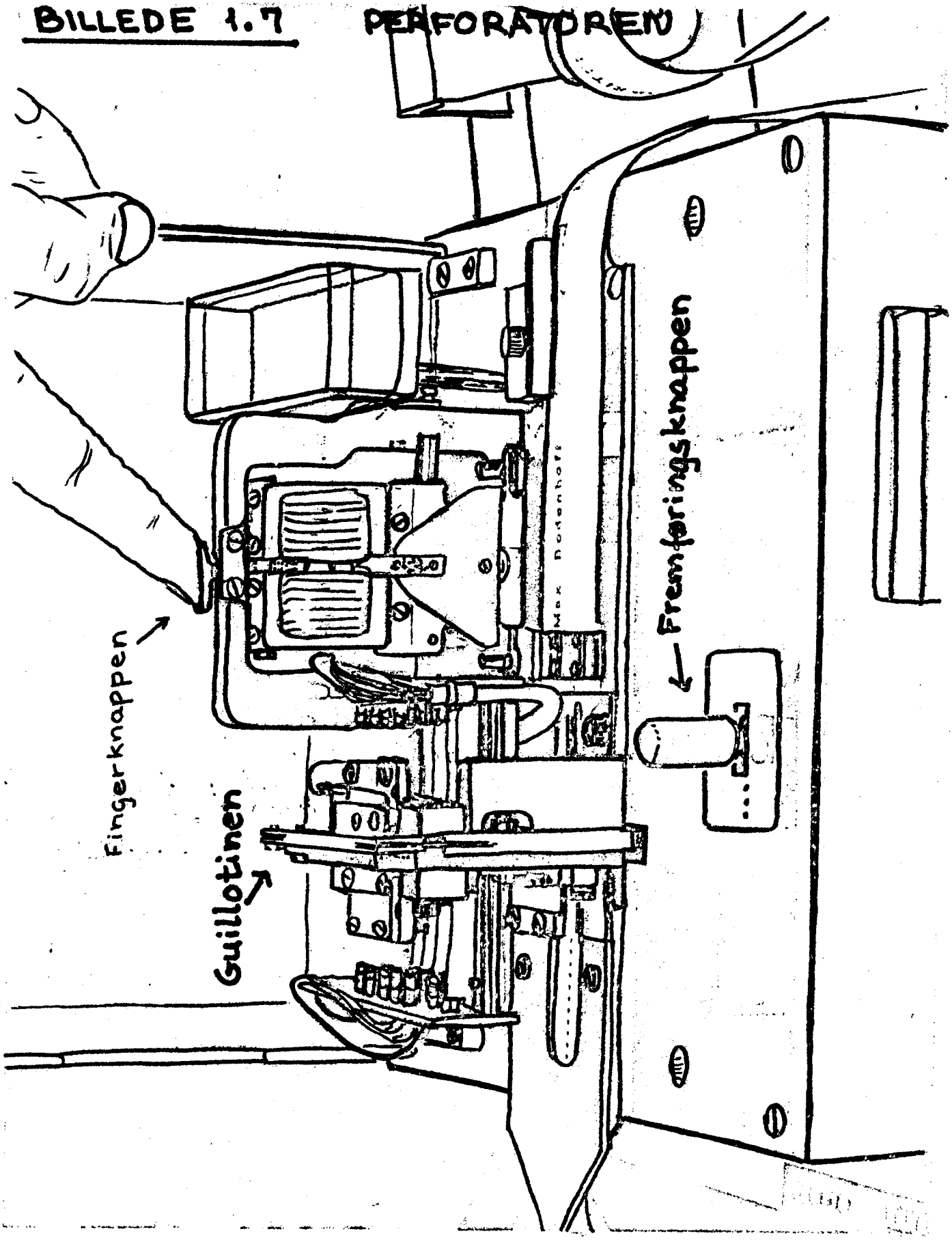
PRINCIPSKITSE AF STRIMMELLÆSER



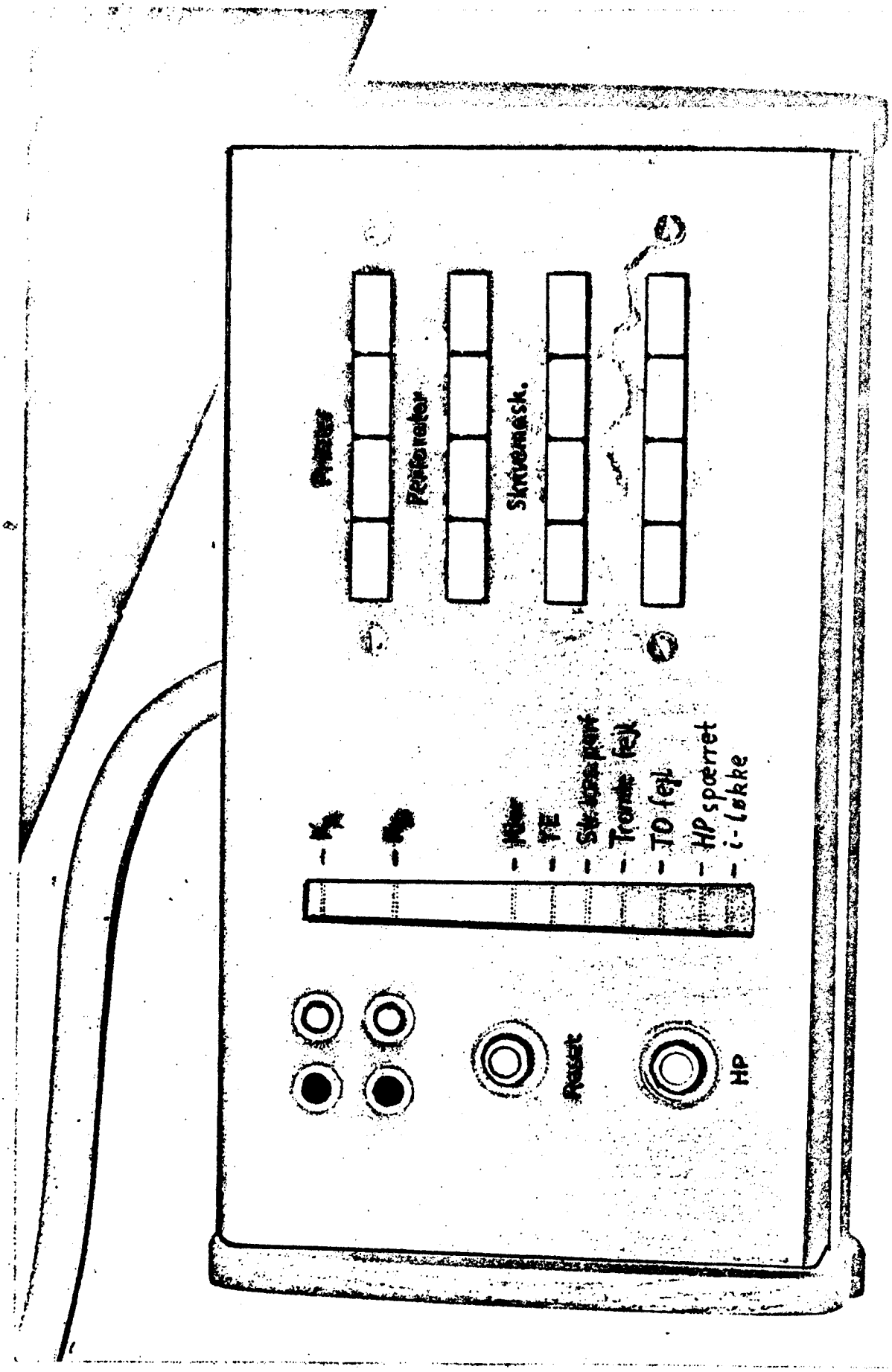


FOR AT STRIMMEL-LÆSEREN SKAL KUNNE LÆSE KORREKT, MÅ STRIMLERNE OPVIKLES PÅ DEN VISTE MÅDE.

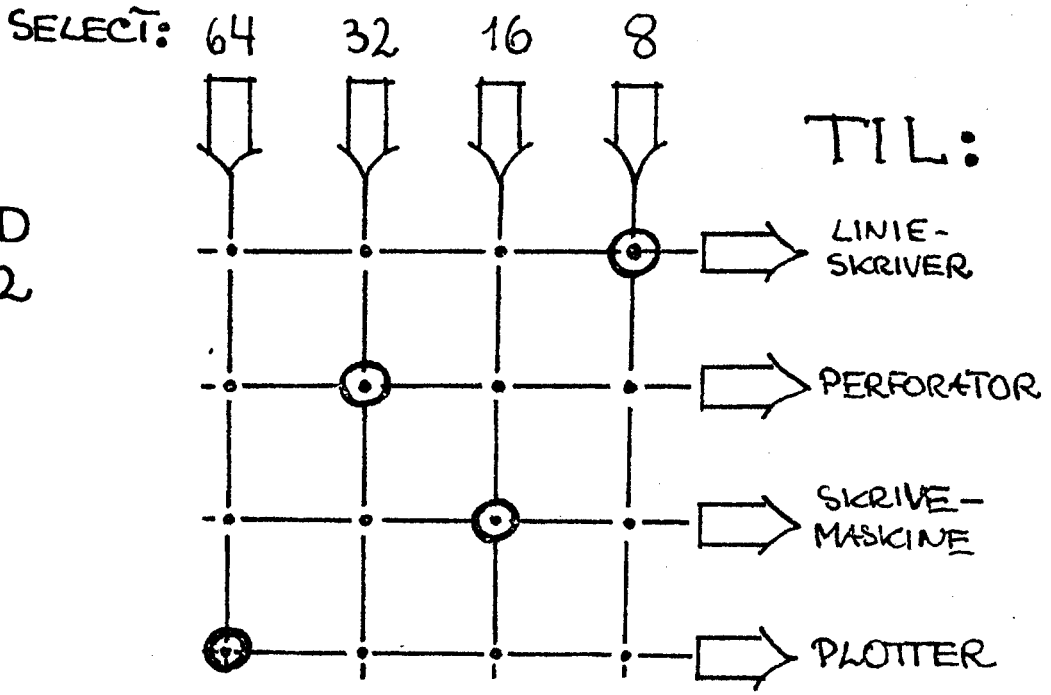
BILLEDE 1.7 PERFORATOREN



BILLEDE 1.8 LILLE KONTROLBORD GIER 2

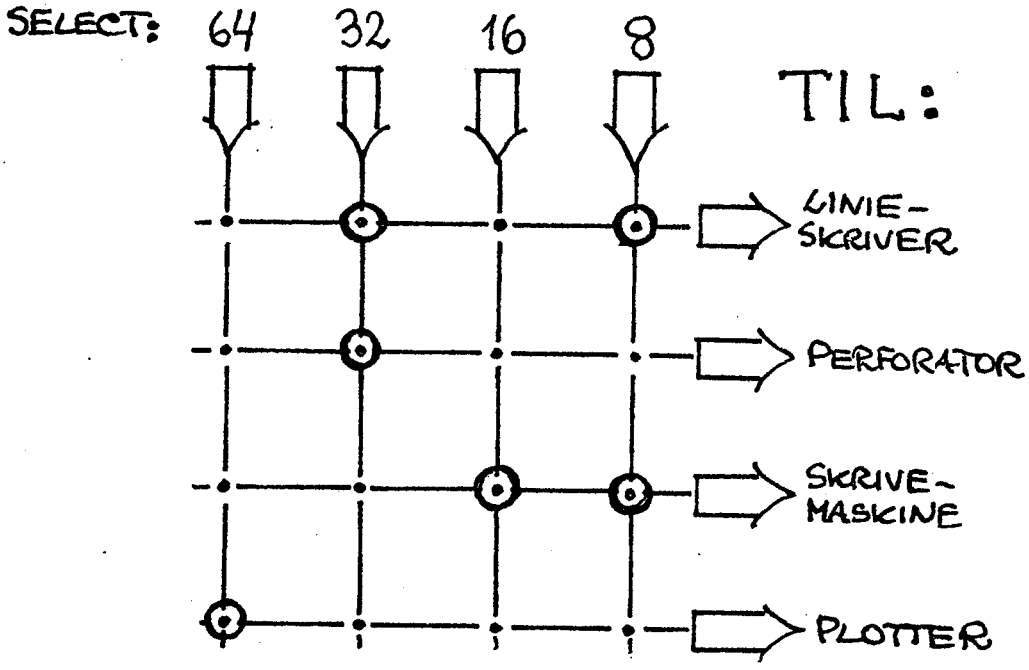


BILLEDE 1.9 KRYDSFELT



STANDARD PÅ GIER 2

FAST PÅ GIER 1

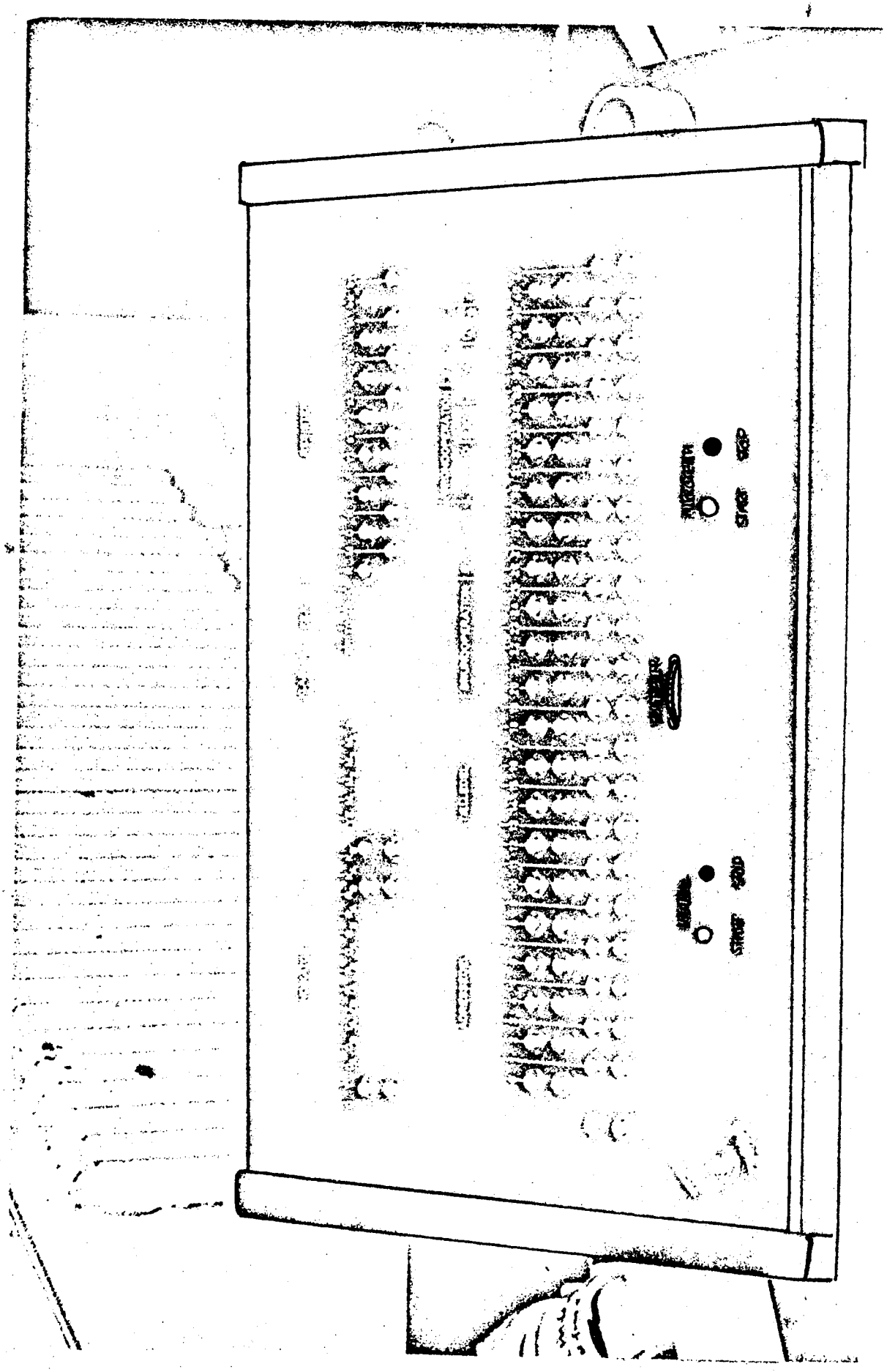


EN ANDEN MULIGHED PÅ GIER 2

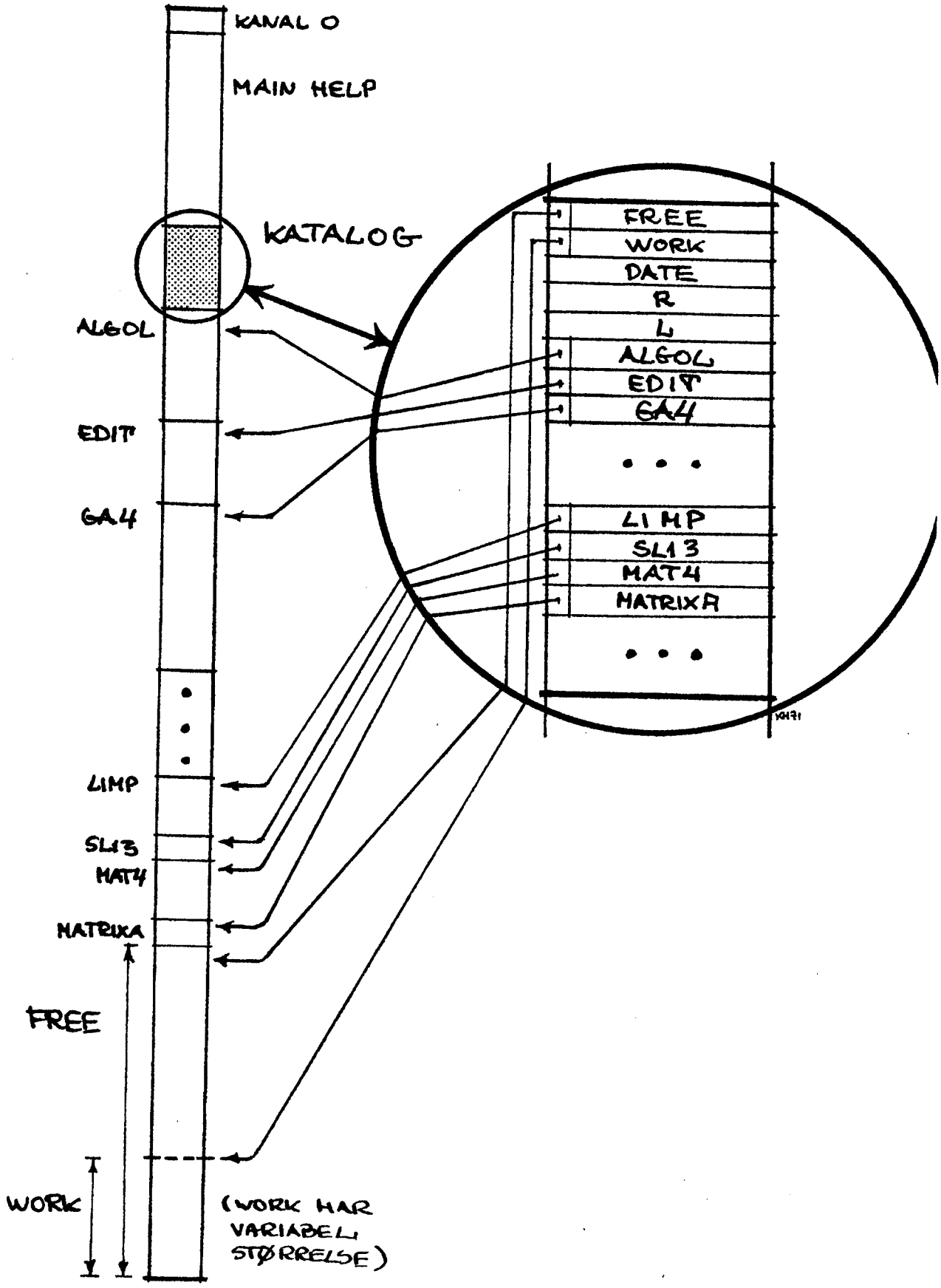
TEGNFORKLARING : ⊙ NEDTRYKKET

• IKKE NEDTRYKKET

BILLEDE 1.10 STORE KONTROLBORD



BILLEDE 2.1


KANAL 0
MAIN HELP
KATALOG
ALGOL
EDIT
GA4
LIMP
SL13
MAT4
MATRIXA
FREE
WORK
(WORK HAR VARIABEL STØRRELSE)
FREE
WORK
DATE
R
L
ALGOL
EDIT
GA4
LIMP
SL13
MAT4
MATRIXA

BILLEDE 2.2

3. HELP 3 INFORMATION.

3.1. External form:

<Help 3 information> ::= <list> < |<list> ≤ |<list> <text> < |
<list> <Help number> <

<list> ::= <empty>|<list> <separator>|<list> <single>|
<list> <text> <separator>|<list> <Help number> <separator>

<text> ::= <letter>|<text> <digit>|<text> <letter>|<text> .

<Help number> ::= <digit>|<Help number> <digit>|<Help number> .

<single> ::= <underlined digit>|<underlined letter>|_|.

<separator> ::= ,|<CR>

<digit> ::= 0|1|2|3|4|5|6|7|8|9

<letter> ::= a|b| ... |z|æ|ø|-

The symbol å will always delete the current line and select the typewriter as input unit.
The symbol <10> marks the end of a text string and will cause an alarm.
All symbols not mentioned above are blind. Thus all upper case symbols will be skipped.

Examples: r, <
sl 13, print, b, 105, ra p 0.10.0.39 <
≤

3.2. Internal form.

The Help 3 information is transformed to an internal word by word form easily described in Slip language:

<text> -> t <text>; (0-marked words) spaces and blind symbols are removed.
<number> -> b-marked word with point replaced by slash, e.g. 10..39 -> 10//39 b
<single> -> qq <value of the underlined symbol>, (a-marked), e.g. a -> qq 49,
≤ and < -> qqf,

The termination ≤ will immediately (without interpretation) cause a jump to the primitive input program on track 0. ≤ can be used to reestablish the system if the catalog is spoilt.

BILLEDE 2.3

```
 30   4.11.70  849
r,edit,ofree<
   2
   2
list,afree<
0,  73, 247,  x i d 4, 2, 41
    free

res,salk,ks15<
list,a,free,salk<
0,  71, 249,  x i d 4, 0, 41
    free

0,  2, 247,  r
    salk
    ks15

salk<
ok
run<
   .
   .
   .
image
p31   4.11.7   17
list,afree
wøå
annul
work,image<
0,  71, 249,  x i d 4, 0, 41
    free

0,  11, 309,  x r
    work

0,  26, 294
    image

run<

spill  4-14, 1022
image
p32   4.11.70  58
edit,iks15,o1<

å

clear,salk<
compress<
Words in cat, free size   80, 73
```

booked

her tastes space

PÅ GIER2

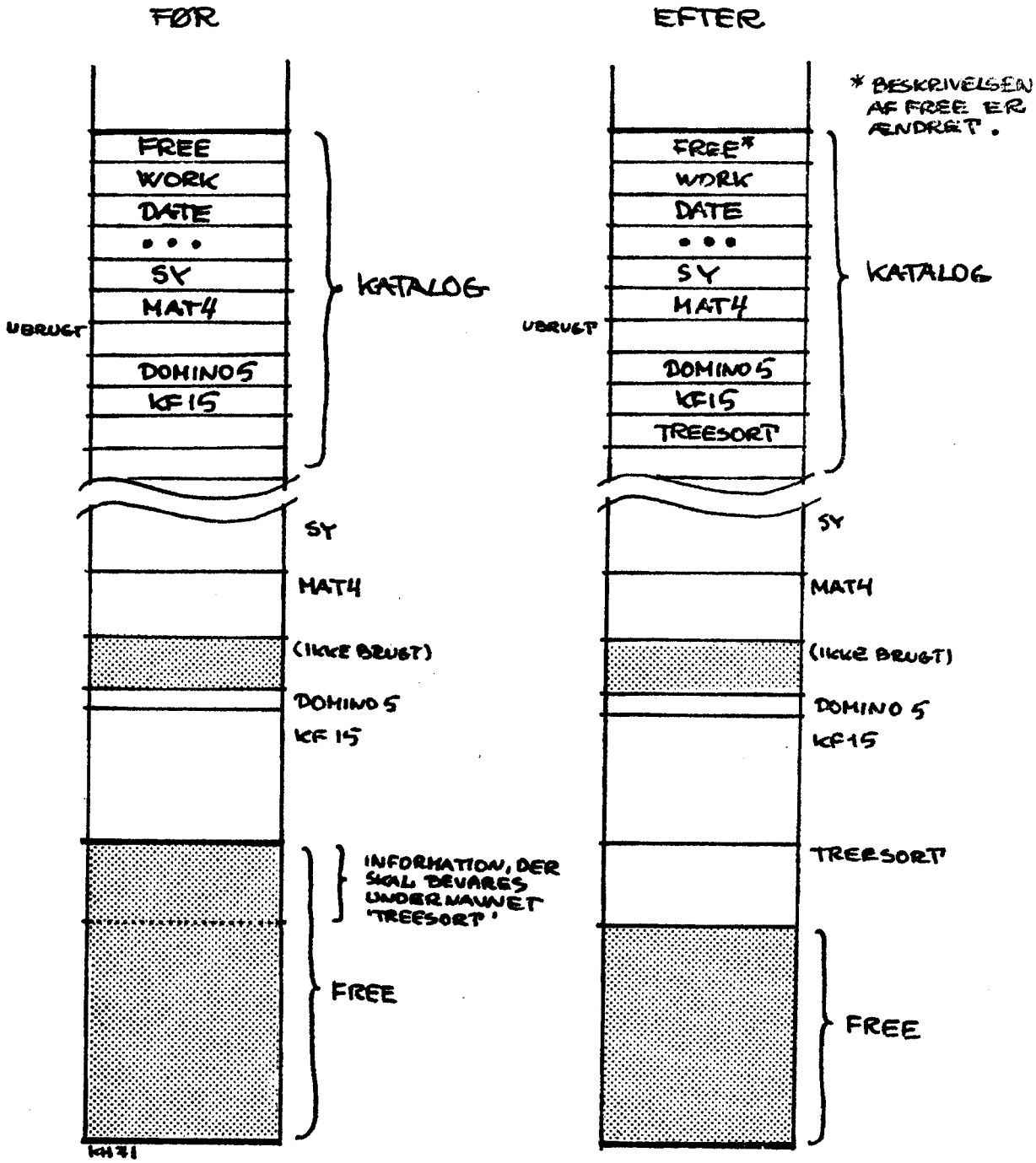
```
w,list,a
free,work,image<
0,  11191, 8009,  x i d 70, 0, 2880
    free

0,  123, 741,  x d 0, 802, 62
    work

0,  26, 864
    image
```

DER RYDDES OP TIL NÆSTE BRUGER

HELP-3 KALD: RES, TREESORT' <

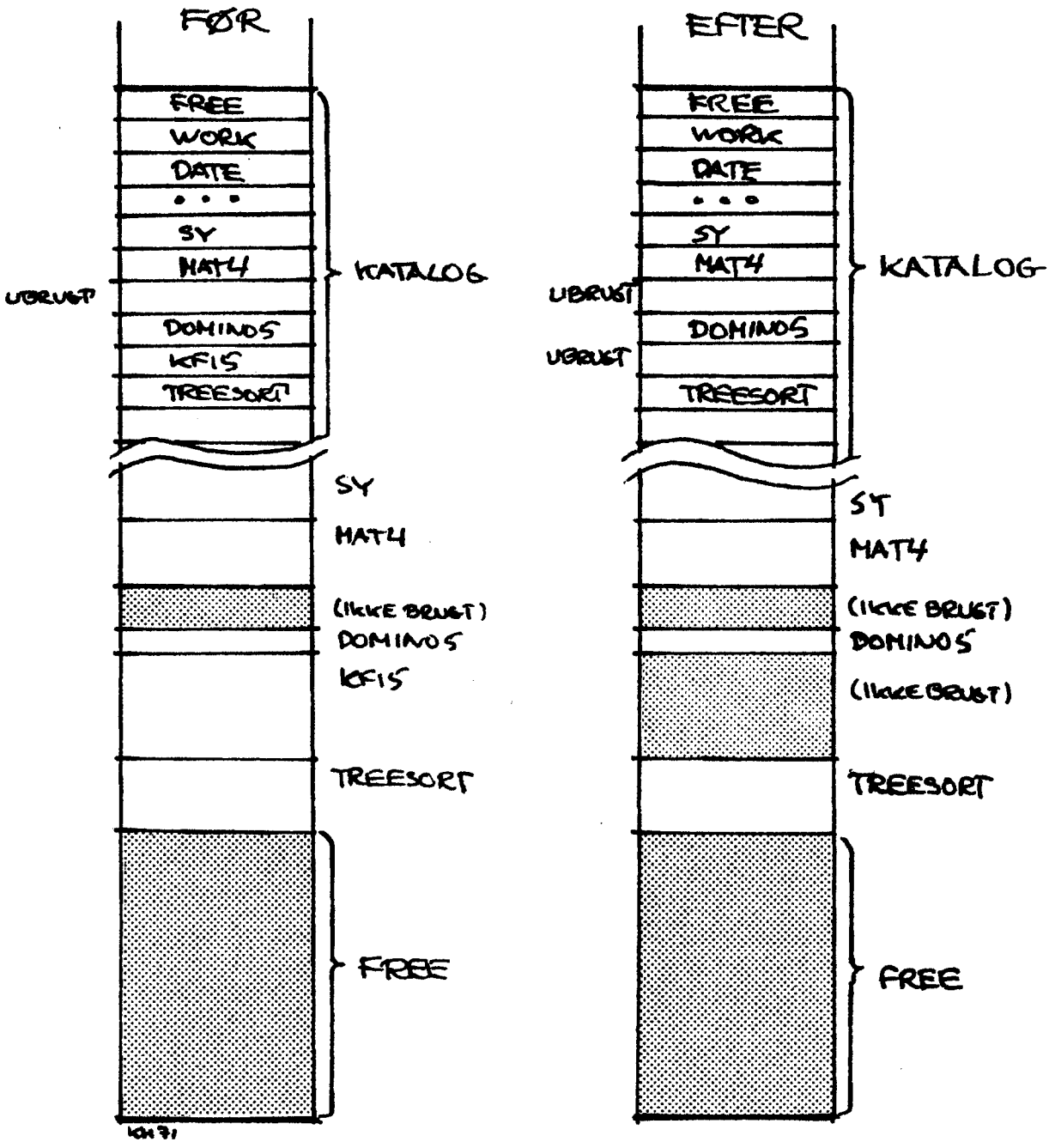
BILLEDE 2.4

RES



BILLEDE 2.5
CLEAR

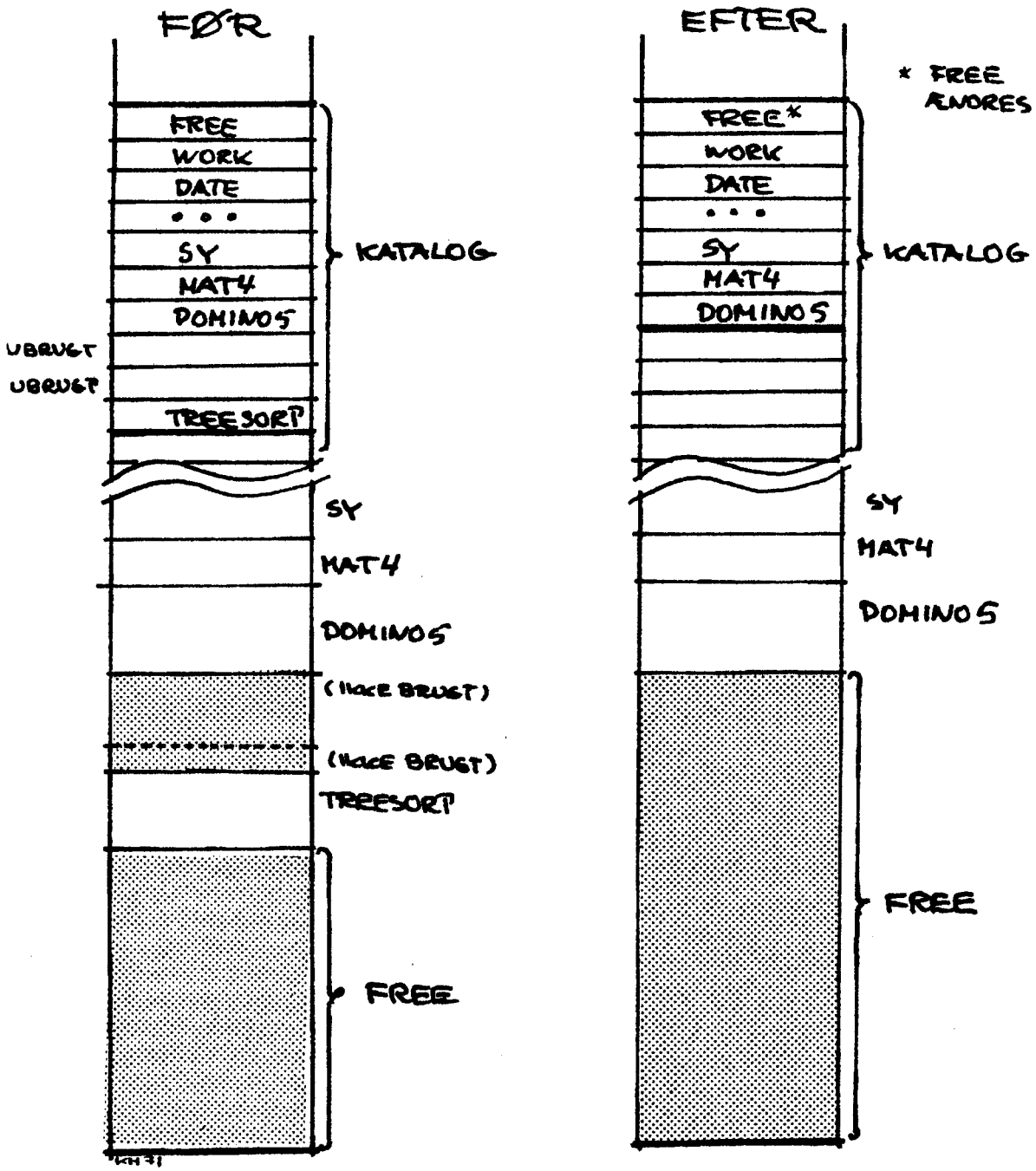
HELP-3 KALD: CLEAR, KF15 <



BRUGES CLEAR PÅ ET NAVN, DER BESKRIVER ET OMRÅDE, SOM IKKE STØDER OP TIL FREE, SLUTTES KUN INDGANGEN I KATALOGET, OG FREE ÆNDRES IKKE. (SMLGN. BILLEDE 2.6)

BILLEDE 2.6
CLEAR

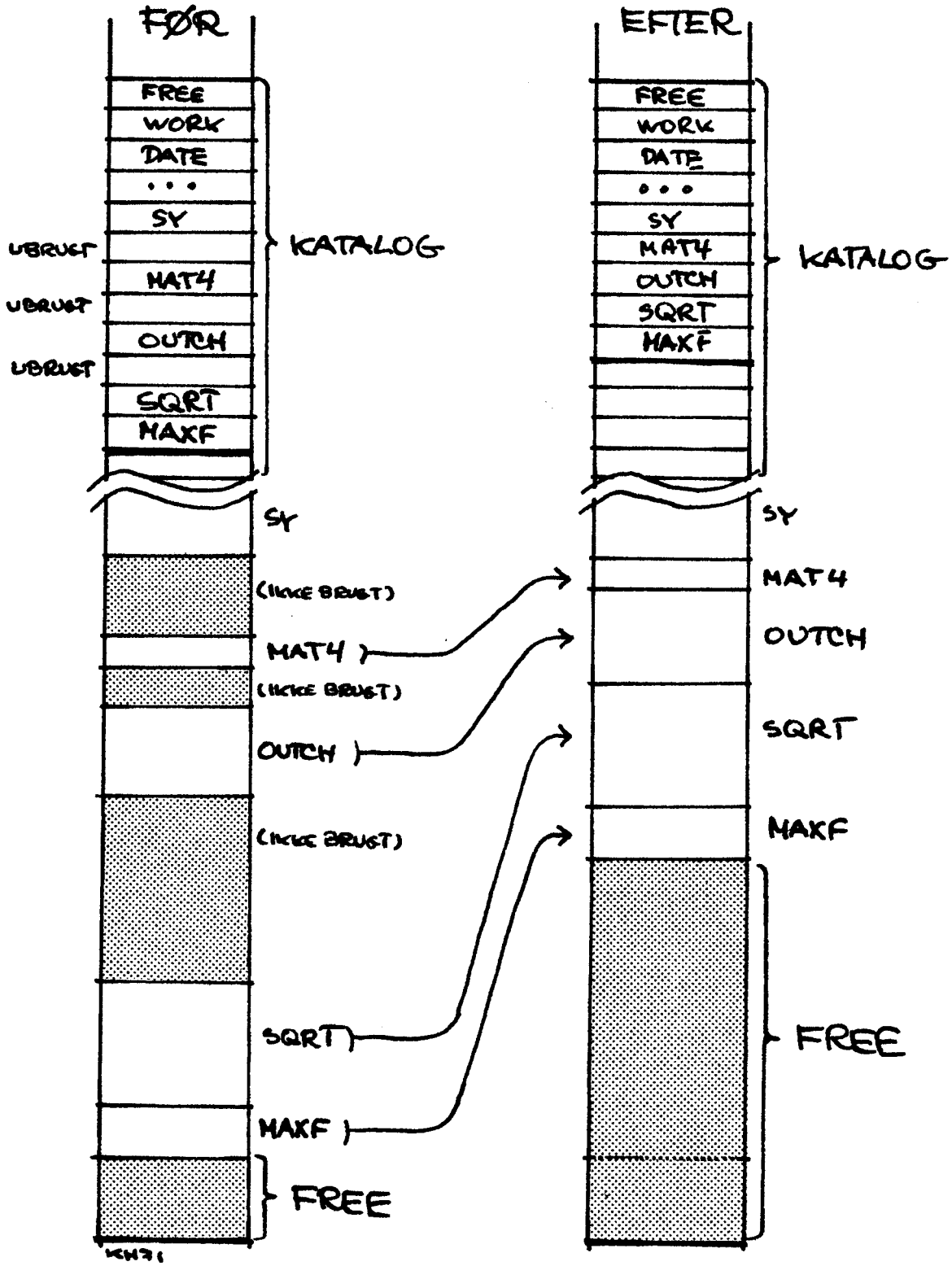
HELP-3 KALD: CLEAR, TREESORT <



BRUGES CLEAR PÅ ET NAVN, DER BESKRIVER ET OMRÅDE, DER STØDER OP TIL FREE, ØGES FREE MED DET FRIGIVNE OMRÅDE, OG DESUDEN MED TILSTØDENDE, TIDLIGERE FRIGIVNE OMRÅDER.

BILLEDE 2.7
COMPRESS

HELP 3 KALD: COMPRESS <



Forbindelsen mellem Gier Algol 4 og Algol 60. Grundsymbolerne i Gier Algol 4 stemmer næsten fuldstændig overens med grundsymbolerne i referencesproget. Der er dog mindre afvigelser for følgende symboler

referencesprog	↑	¬	␣	/	\	÷	⇒
Gier Algol	↑	-,	⊥	*	*	:	=>

Desuden er der i Gier Algol 4 tilføjet nogle nye grundsymboler nemlig abs entier round mod shift case of core code message copy finis 1 2 ... 40. De fleste af disse vil blive omtalt i det følgende, her skal kun nævnes message. Hvor man kan skrive comment i et algolprogram, kan man også skrive message. Forskellen er blot, at den text, som skrives mellem message og det først følgende semikolon, vil blive udskrevet på skrivemaskinen under oversættelsen. På denne måde kan programmøren give instruktioner til operatøren.

Tal. Variable og konstanter af type integer bliver behandlet som fastkomma-tal og er defineret i intervallet

$$-549755813888 = -2\uparrow 39 \leq \text{integer} \leq 2\uparrow 39-1 = 549755813887$$

Variable og konstanter af type real bliver behandlet som flydende tal, dvs på formen $r = z\times 2\uparrow v$, hvor

for $r \neq 0$	$1 \leq \text{abs}(z) < 2$,	$-512 \leq v < 512$
for $r = 0$	$z = 0$	$v = 0$

dvs for en variabel real, hvor real ≠ 0 haves

$$7.458_{10}-155 = 2\uparrow(-512) < \text{abs(real)} < 2\uparrow 512 = 1.341_{10}154$$

Variable af heltalstype vil indenfor definitionsområdet blive repræsenteret eksakt, mens variable af type real er repræsenteret med en relativ nøjagtighed på ca. $3_{10}-9$ svarende til 29 betydende binære cifre, eller 8-9 betydende decimale cifre. I aritmetiske udtryk med operander af blandet type vil heltal (dog så sent som muligt) blive omdannet til reelle tal.

Aritmetiske udtryk. I aritmetiske udtryk kan man foruden de sædvanlige aritmetiske operatorer (+,-,×,/,:,↑) benytte nogle nye operatorer, som alle har højere prioritet end de sædvanlige aritmetiske operatorer (undtagen mod som har prioritet som ×,/,:), nemlig

abs entier round integer real mod

abs og entier virker som de tilsvarende standard funktioner. round giver nærmeste heltal. integer og real har i almindelig-

hed kun mening, når de står foran en boolsk operand, og dennes bitmønster (se side 7) opfattes da som hhv et heltal eller et reelt tal. mod (modulo) har to operander, som begge skal være af heltalstype, og resultatet bliver heltalsresten ved heltals divisionen. Eksempler:

```
f:= absg+h-entier(roundJ/entier absk;
```

dette aritmetiske udtryk vil blive vurderet som om følgende paranteser var sat

```
f:= (absg)+h-(entier(roundJ)/(entier(absk));
```

a mod b er det samme som a-(a:b)×b

Case udtryk og case sætninger betegner en udvidelse af Algol 60. De er en naturlig generalisering af betingede udtryk og sætninger. I if then else konstruktionen har man et valg mellem to muligheder, ved case of konstruktionen kan man få et valg mellem n muligheder, hvor n er et heltal > 0. I princippet kan n være vilkårlig stor, men på Gier skal n < 35.
Lad en case konstruktion bestå af

case <aritmetisk udtryk> of

et case udtryk består så af en case konstruktion efterfulgt af en liste af udtryk, adskilt af komma og omsluttet af paranteser. Udtrykkene i listen skal være af samme type.

en case sætning består af en case konstruktion efterfulgt af en liste af sætninger adskilt med semikolon og omsluttet af begin end.

Virkningen af et case udtryk hhv case sætning er som følger. Først beregnes det aritmetiske udtryk mellem case og of. Hvis værdien af dette er ≤ 0 eller > antallet af listeelementer, kommer der fejludskrift, ellers udregnes det valgte udtryk hhv udføres den valgte sætning. Eksempler:

```
a:= 0.5+(case q-1 of (y,2,pip+19))↑2

case p of begin
a:= a×p; pip:= sin(y); begin p:= 4; a:= 2×p end end

goto case p of (label1,label2,label3)
```

Bitmønstre. En celle (også kaldet et ord) i Gier består af 40 bits + 2 mærkebits. De sidste er ikke tilgængelige i Gier Algol 4. De 40 bits, som hver kan have værdierne 0 eller 1, numereres fra 0 til 39. Af hensyn til den menneskelige læser er de 40 bits, som repræsenterer indholdet i en tilfældig celle, her samlet i grupper på fem

```
00110 11011 10001 00011 11110 00101 11001 10011
↑                                             ↑
bit0                                        bit39
```

Lad os antage, det er værdien af en variabel, som er lagret i denne celle. Det vil nu afhænge af denne variabels type, hvorledes indholdet i cellen skal behandles. Er den variable af type real, r = z×2↑v , vil de første 10 bits give eksponenten v og de sidste 30 mantissen z. Er typen integer angiver cellens indhold heltallet på binær form med enhed i position 39. Er typen boolean kan den variables sandhedsværdi aflæses i bit0, idet

bit0=0 svarer til false bit0=1 svarer til true

Udover at kunne repræsentere sandhedsværdierne true og false (hvor kun indholdet i bit0 udnyttes) kan boolske variable i Gier Algol 4 benyttes til at repræsentere vilkårlige bitmønstre. Sådanne bitmønstre kan opbygges ved hjælp af understregede positive heltal

```
b:= 5 3 5 6 3 1 2 2 10 1023 5 5 ;
```

I dette eksempel vil den boolske variabel b komme til at indeholde følgende bitmønster

00011 00110 00110 11111 11111 00101 00000 00000

Opbygningen sker fra venstre, de understregede heltal angiver længden af et delmønster, det efterfølgende heltal på binær form angiver mønstret. Cellen fyldes evt op med nuller til slut. Ønskes et delmønster helt fyldt op med ettaller, kan man efter det understregede tal som angiver delmønstrets længde skrive et m i stedet for et tal. Særligt simple bitmønstre kan dannes ved hjælp af true og false, idet disse bitmønstre er lig bitmønstrene 40 m og 40 0 hhv.

Man kan operere på bitmønstre med de sædvanlige logiske operatorer. Desuden er indført en ny operator shift, eksempel

```
b shift i
```

hvis virkning er, at bitmønstret som svarer til den boolske variabel b forskydes cyklisk i pladser til venstre for i > 0 og -i pladser til højre hvis i < 0. i skal være integer og b skal være boolean. Hvis i har værdien 8 og b indeholder samme bitmønster som i sidste eksempel, da vil a:= b shift i bevirke at den logiske variabel a kommer til at indeholde følgende bitmønster:

10001 10111 11111 11001 01000 00000 00000 11001

Eksempel: Find 2-talseksponenten, eksp2, for en variabel xyz, hvor xyz er af type real og xyz > 1. En løsning er

```
eksp2:= integer((boolean xyz ^ 10 1023) shift 10);
```

Fire nye operatorer, integer real boolean og string kan benyttes til at ændre typen af deres argument. Eksempel:

```
integer a, real b, boolean (c+d), string e[i]
```

Argumentet opfattes som et bitmønster på 40 bits, der ikke ændres af operatoren.

Reserverede_navne. I Gier Algol 4 er der 33 reserverede navne. De ni af disse er standardfunktionerne. Reserverede navne skal ikke erklæres i programmet. Listen ser således ud:

abs	kbon	sin
arctan	ln	sqrt
cancel	lyn	system
char	put	tracks transferred
checksum	readgeneral	us
cos	readinteger	where
entier	readreal	write
exp	readstring	writechar
get	reserve	writecr
gier	select	writeinteger
il	sign	writetext

Valg_af_input_og_ouput_enheder,_select. Valg af input enhed og output enhed foregår ved hjælp af standard proceduren select, som kræver een heltalsparameter. Eksempel:

```
select(8);
```

strimmellæser vælges som input enhed og linieskriver som output enhed.

Hver input og output enhed har tilknyttet en by-værdi (svarende til indholdet i et 10 bits register i Gier, by-registret). Disse by-værdier kan variere fra installation til installation. Nedenfor er givet en tabel over sammenhængen mellem by-værdier og input-output enheder, idet disse sidste også er beskrevet ved deres Hjælp 3 navne.

by-værdi	input-output enhed	
	navn	betydning
0	r	input strimmellæser
1	t	input skrivemaskine
3	r	input strimmellæser ispp
8	l	output linieskriver
16	w	output skrivemaskine
32	p	output perforator
64		output plotter

ispp betyder **ikke stop på paritetsfejl**. Tabellen gælder for Gier installationen på HCØ.

Ved et kald af select kan man vælge en vilkårlig kombination af input og output enheder ved simpelthen at addere by-værdierne for disse og benytte select med summen som parameter.

select er en integer procedure, hvis værdi er den aktuelle parameter svarende til forrige kald af select. Eksempler:

```
select(35);
```

herved vælges strimmellæser som input og perforator som output. Dette er det standard valg af ydre enheder, som har gyldighed i starten af et algolprogram, dvs før det første kald af select.

```
begin integer s, ... ;
 ...
SKRIV:  s:= select(17);
comment her vælges skrivemaskine til input og output;
 ...
select(s);
comment her retableres valg af enheder som før SKRIV;
 ...
end
```

For Gier datamater med trykknap manøvrebord til valg af outputenhed er det forudsat, at knapperne er trykket ned på standard måde, nemlig (HCØ):

output-enheder	by-værdier 64	32	16	8
linieskriver				XX XX XX
perforator		XX XX XX		
skrivemaskine			XX XX XX	
plotter	XX XX XX			

Standard output procedurer

writecr Proceduren writecr kaldes uden parametre og bevirker output af eet linieskift symbol (CR).

writechar Proceduren writechar har een parameter og bevirker output af det symbol, som svarer til talværdien af parameteren. Forbindelsen mellem symbol og talværdi er givet på sidste side. Eksempler:

```
writechar(16)  bevirker output af et nul
writechar(49)  bevirker output af et a
writechar(72)  bevirker output af et sideskift sym-
bol (benyttes i forbindelse med linieskriver).
```

writetext Proceduren writetext har een parameter og bevirker output af en streng af symboler. Parameteren kan f. eks. være en streng af algolsymboler omsluttet af ‡< og ‡>, eller en variabel med operatoren string foran. Eksempler:

```
writetext(¢<x= ¢);
writetext(if q then ¢<yes¢ else ¢<no ¢);
writetext(case n of (¢<henrik¢,¢<anders¢,¢<jens¢,
          ¢<thyge¢,¢<børge¢));
writetext(string b[1]);
writetext(a); her må a være en formel parameter spe-
          cificeret som string.
```

layout

I de outputprocedurer, som udskriver tal, skal den første parameter være et layout. Et typisk layout ser således ud

¢-ddd.dd¢

Layouttet er en model af tallet og angiver i dette tilfælde, at man ønsker output af et tal med 3 cifre foran . og to efter samt - tegn forrest, hvis tallet er negativt. Et layout omgives altid af ¢ og ¢. Imellem disse tegn kan skrives mellemrum (SPACE), +, -, p, d, ., 0_{10}. Tallet vil i almindelighed fylde lige så meget som layouttet angiver. Hvis tallet ikke har cifre til at udfylde layouttet, vil der blive fyldt op med mellemrum. Hvis tallet er for stort til layouttet, vil det blive skrevet ud med et nød-layout, hvilket bevirker at der bruges ekstra plads. Et layout er af type boolean og kan derfor assignes til en boolsk variabel. Eksempler på layout:

¢␣␣␣-d.ddd$_{10}$+d¢ , ¢-dddd¢ , ¢pddd.dd¢

- ␣ betyder mellemrum. Mellerum kan indlede et layout eller anbringes blandt cifre
- \- betyder at - bliver trykt, hvis tallet er negativt ellers trykkes mellemrum
- \+ betyder at fortegn altid trykkes
- d angiver cifre
- p betyder at ledende nuller i et tal trykkes.

Idet || benyttes til at angive udstrækning af det trykte tal, vil tallet 37.95321 med de ovenstående layout blive trykt:

| 3.795$_{10}$+1| , | 38| , |0037.95|

writeinteger

Proceduren writeinteger har to parametre. Den første er et layout, og den anden er et udtryk, som skal være af heltalstype. Proceduren bevirker output af heltallet i overensstemmelse med layouttet. writeinteger er ca dobbelt så hurtig som write (se nedenfor), men giver kun output af eet heltal pr kald. Desuden er der visse restriktioner på layouttet for writeinteger idet $_{10}$-potens ikke kan benyttes. Man kan få anbragt et decimalpunktum i heltallet. Eksempler:

```
writeinteger(¢-ddd.dd¢,5713);  udskriver  | 57.13|

b:= ¢pdd ddd ddd ddd¢;  writeinteger(b,x);  hvis x=
5713 fås udskriften  |000 000 005 713|.
```

write

Proceduren write kan kaldes med mindst to parametre, hvoraf den første skal være et layout. De følgende parametre skal være aritmetiske udtryk. Kald af proceduren bevirker output af værdierne af disse udtryk

efter hinanden i overensstemmelse med layouttet. Eksempler:

```
write(‡   -.ddd ddd ddd10-dd‡,a,b,c,d);
write(case i of(‡d‡,‡d.d‡,‡d.dd‡),s×t↑2);
```

Standard Input Procedurer

char er en standard variabel af type integer. Den vil blive brugt som første input symbol ved kald af read-procedurer. Når et sådant kald er udført, vil char indeholde værdien af det sidst indlæste symbol. char er i starten af et algolprogram initialiseret til 0 (bemærk talværdien 0 svarer til symbolet SPACE, se sidste side).

lyn er en integer procedure, som indlæser eet symbol, og hvis værdi er værdien af dette symbol (se sidste side Eksempel:

```
for i:=lyn while i‡11 do;
```

denne sætning scanner datastrimlen eller dataarealet indtil der kommer en STOPCODE.

readinteger er en integer procedure og har som værdi det næste heltal, som indlæses fra det valgte inputmedium. Eksempel:

```
n:= read integer;
```

readreal er en real procedure og har som værdi det næste reeltal, som indlæses fra det valgte inputmedium. Eksempler:

```
r:= read real;   Z[read integer]:= read real;
```

readgeneral er en integer procedure som kan indlæse tal og symboler mellem hinanden ind til et array. Proceduren kunne være erklæret som følger

```
integer procedure read general(R,SKIPANDEXIT,ELEMENT);
value SKIPANDEXIT; <type>array R; boolean SKIPANDEXIT;
integer ELEMENT;
```

Proceduren kaldes altså med tre parametre. Den første R, er navnet på det array man vil indlæse til. Hvert tal kommer til at fylde et element i R, og hvert symbol mellem tal kommer til at fylde et element i R. Det afhænger af arrayets type på hvilken form tallene læses ind. Hvis det er et boolean eller integer array læses tallene ind som med read integer. Hvis det er et real array læses de som flydende tal, dog bemærkes at titalseksponent i tallene ikke er tilladt.

Den tredje parameter, ELEMENT, skal være en variabel af heltalstype. Lad os tænke os en fortløbende nummerering af arrayet Rs elementer startende med 1 (fremkommmet ved at starte med at variere bageste indeks mellem dets nedre og øvre grænse, osv). ELEMENT skal ved kald af read general have værdien af nummeret på det sidst benyttede element af R. Ved start af indlæsning til et array vil ELEMENT derfor ofte være

sat lig 0. Indlæsningen starter til nummer (ELEMENT+1) af arrayets elementer. ELEMENT vil svare til indeks, hvis R er endimensional og erklæret som R[1:n].

Den anden parameter, SKIPANDEXIT, er et bitmønster bestående af 4 tibitsgrupper, som definerer 4 symboler. 2 skipsymboler og 2 exitsymboler. Når et af de to skipsymboler læses, bliver det ikke lagret, men overspringes. Når et af de to exitsymboler læses, standses indlæsningen, og exitsymbolet lagres ikke. Indlæsningen standses også, dersom arrayet bliver fyldt op. Bitmønstret for SKIPANDEXIT ser således ud:

10 SKIP1 10 SKIP2 10 EXIT1 10 EXIT2

De første 3 bits i hver tibitsgruppe kan bruges til at beskrive casesituationen for skip og exit symbolerne: 0 svarer til lower, 1 til upper, 2 til lower eller upper, og 3 til not used. Efter udførelse vil værdien af readgeneral fortælle noget om indlæsningens forløb

-1 betyder arrayet blev fyldt op
0 betyder exitsymbol blev læst
>0 betyder at symboler andre end tal er blevet lagret og værdien peger på det første symbol

Eksempel: Lad os vælge SKIP1 til komma, SKIP2 til CR, EXIT1 til ENDCODE, EXIT2 bruges ikke. Inputstrengen er

25.3, 27.5, 31.7
28.9, 11.5, -9.8,

lad alfa være array alfa[20:25], og lad koden være

```
p:=0;  rg:=readgeneral(alfa,10273276432712 33,p);
```

efter udførelsen af ovenstående sætninger vil alfa[20], ... , alfa[25] indeholde de ovenstående tal, p er 6, char er 27, og endelig vil rg have værdien -1.

readstring er en integer procedure som indlæser en vilkårlig streng af symboler til et array. Den kunne være erklære

```
integer procedure readstring(S,STARTANDEXIT,ELEMENT);
value STARTANDEXIT; <type>array S; boolean STARTANDEXIT
integer ELEMENT;
```

Der er 3 parametre. Den første er det array, der læses ind til. Den anden er et bitmønster bygget op efter tilsvarende konventioner som SKIPANDEXIT i read general Man kan her angive 4 symboler, START1, START2, EXIT1, og EXIT2. Alle symboler i inputstrengen overspringes indtil der læses et STARTsymbol, derefter indlæses symboler til arrayet indtil dette er fuldt, eller til der er læst et EXITsymbol. Den tredje parameter, ELEMENT er en heltalsvariabel, som før kaldet af readstring bør have antaget værdien 0. Eksempel:

```
n:=0;  readstring(A,10641064106410 64,n);
```

Herved indlæses en textstreng mellem to linieskift til arrayet A, idet symbolerne lagres 6 i hvert element af

Brug af Baggrundslager i Gier Algol 4

Der er mulighed for følgende baggrundslagre ved en Gier-installation:

a) tromlelager, som kan bestå af een eller tre tromler, som hver er delt op i 320 blokke à 40 ord.

b) bufferlager på 4096 ord.

c) pladelager (disk), som enten er knyttet til ferritlageret og da er inddelt i 19200 blokke på 40 ord, eller det er sat i forbindelse med buffer og det er da inddelt i 2030 blokke a 400 ord.

d) magnetbåndslager, som er i forbindelse med buffer med blokke på op til 4096 ord.

e) karrussel, som kommunikerer med buffer i blokke på 512 ord.

Information om maskinkonfigurationen ved en Gierinstallation kan fås ved hjælp af standardproceduren system.

Da den plads, som kan reserveres i ferritlageret til variable er ret begrænset (ca. 650 variable), og da det yderligere vil gøre udførelsen af et program langsommere, hvis hele variabelpladsen udnyttes, vil man ofte være henvist til at bruge baggrundslager. Set fra et algolprogram er baggrundslageret delt op i blokke, og programmet kan overføre data til og fra ferritlager i enheder på en blok. En samling blokke i følge kaldes et areal, og man kan navngive et areal og på den måde reservere det (se side 2). Gennem Gier Algol 4 har man adgang til Hjælp 3 kataloget og dermed til hele den automatiske administration af baggrundslageret.

I det følgende skal kort nævnes de standard procedurer, som bestyrer brugen af baggrundslageret i Gier Algol 4. Man bør særlig lægge mærke til, at man gennem Gier Algol 4 kun kan arbejde med navngivne arealer, hvis absolutte placering i baggrundslageret kan variere fra kørsel til kørsel, og som desuden i almindelighed er uden interesse for brugeren. Information om, hvor et reserveret areal befinder sig i baggrundslageret, vil blive anbragt i den aktuelle parameter svarende til den formelle parameter AREA, når man spørger efter sit navngivne areal med proceduren where. Når AREA har fået sin værdi, kan procedurerne put og get benyttes med bl. a. AREA som parameter.

reserve Denne procedure har to parametre og kunne være erklæret

```
integer procedure reserve(NAME,BLOCKS);
value BLOCKS; string NAME; integer BLOCKS;
```

Et kald af reserve vil bevirke et forsøg på at reservere et areal på BLOCKS blokke med navnet NAME. Hvis reservationen foretages, får reserve værdien 0. Eksempel

```
a:= reserve(*<atp*,m);
```

where Denne procedure har to parametre og kunne være erklæret

```
integer procedure where(NAME,AREA);
string NAME; integer AREA;
```

Virkningen af et kald af where er, at kataloget gennemsøges for at se om NAME er der, og hvis det er tilfældet vil en beskrivelse af dette areal blive anbragt i AREA. Kun hvis NAME fandtes i kataloget, får where værdien 0. Eksempel

```
check:= where(‡<atp‡, AREA);
```

cancel `integer procedure cancel(NAME); string NAME;`

Virkningen af et kald af cancel er, at kataloget gennemsøges for at se, om NAME er der, og hvis det er tilfældet slettes dette areal i kataloget, og cancel får værdien 0.

z Eksempel

```
cancel(‡<atp‡);
```

put
```
integer procedure put(A,AREA,PLACE);
value AREA, PLACE; integer AREA, PLACE; <type>array A;
```

Effekten af et kald af put er, at data bliver transporteret fra arrayet A i ferritlageret til den del af AREA, som PLACE angiver. Se videre under get.

get
```
integer procedure get(A,AREA,PLACE);
value AREA,PLACE; integer AREA, PLACE; <type>array A;
```

Effekten af et kald af get er, at data bliver transporteret fra den del af AREA, som PLACE angiver, til arrayet A. Værdien af AREA skal stamme fra et kald af where. PLACE skal være et positivt heltal, som peger på den første blok, som skal transporteres. Blokkene indenfor et areal er numereret 1,2,... Derfor haves

$$1 \leq \text{PLACE} \leq \text{antal blokke i arealet}$$

Arrayet A skal have mindst 40 elementer. Antallet af ord, som overføres, vil være lig antallet af elementer i A.

Elementerne i et array af dimension ≥ 2 bliver behandlet som en lineær sekvens efter de sædvanlige konventioner.

Data overført til baggrundslageret med et givet array ved et kald af put, kan overføres til ferritlageret igen ved et kald af get med det samme AREA og PLACE og med et array af samme størrelse som før. Eksempel

```
put(R,AREA,2);          get(R,AREA,2);
```

Det følgende eksempel på brug af baggrundslageret er et program, som skal indlæse to n×m matricer der antages at være så store, at de må lagres i baggrundslageret. De skal lagres under navnene matrix1 og matrix2. Efter indlæsning skal summatricen udregnes og lagres i baggrundslageret under navnet summatrix.

```
algol<
begin integer n,m;
procedure alarm(text); string text;
begin writecr; writetext(text); goto END end;
select(16); n:= readinteger; m:= readinteger;
     begin boolean SKIPANDEXIT;
     array A,B[1:if m>40 then m else 40];
     integer rækkelængde,blokantal,i,J,p,rg,FREE,MATRIX1,MATRIX2;
     rækkelængde:= (m+39):40; blokantal:= rækkelængde×n;

     comment man har valgt at lade dataoverførslen mellem tromle
     og ferritlager foregå rækkevis, derfor udregnes rækkelængden
     i hele blokke. Man kunne også have overført data blokvis
     (kanalvis). I så fald skulle A og B være erklæret A,B[1:40],
     blokantal skulle være lig (n×m+39):40, i de forsætninger
     hvor i er styrende variabel skulle n erstattes med blokantal,
     og i alle kald af put og get skulle 3.-parameter være lig 1.
     Ved summationen skulle m erstattes med 40 og efter sætningen
     B[j]:= A[j]+B[j] måtte man indsætte en test, således at man
     ikke summerer mere end n×m elementer;

     SKIPANDEXIT:= 1027103201026833;

     comment datastrimlen tænkes hullet, så de første tal er
     n og m. Derefter kommer de to matricers elementer rækkevis,
     således at der efter hver matrix (dvs efter n×m tal) er
     hullet en END CODE. SKIPANDEXITs værdi svarer til, at komma
     og vognretur (CR) kan benyttes som skilletegn mellem tal,
     og END CODE er exitsymbol;

     for J:=1,2 do
     begin if where(‡<free‡,FREE)‡0 then alarm(‡<free‡);
           for i:=1 step 1 until n do
           begin p:= 0; char:= 0;
                 rg:= readgeneral(A,SKIPANDEXIT,p);
                 put(A,FREE,(i-1)×rækkelængde+1)
           end i;
           if rg‡0 then alarm(‡<datafejl‡);
           p:= reserve(case j of(‡<matrix1‡,‡<matrix2‡),blokantal)
           if p‡0 then alarm(‡<matrixlagring mislykket‡);
     end J;
     where(‡<matrix1‡,MATRIX1); where(‡<matrix2‡,MATRIX2);

     comment nu er de to matricer lagret i arealerne med navne
     matrix1 og matrix2 og med arealbeskrivelser MATRIX1 og MATRIX

     where(‡<free‡,FREE);

     comment hver gang reserve er blevet benyttet ændres stør-
     relsen af free. Derfor må where kaldes igen, så FREE kan
     komme til at indeholde den aktuelle arealbeskrivelse af free;

     for i:=1 step 1 until n do
     begin
           get(A,MATRIX1,(i-1)×rækkelængde+1);
           get(B,MATRIX2,(i-1)×rækkelængde+1);
           for J:=1 step 1 until m do B[j]:= A[j]+B[j];
           put(B,FREE,(i-1)×rækkelængde+1)
     end;
     p:= reserve(‡<summatrix‡,blokantal);
     if p‡0 then alarm(‡<summatrix lagring mislykket‡);
     end indre blok;
END: if cancel(‡<matrix1‡)+cancel(‡<matrix2‡)+cancel(‡<summatrix‡)
           then writetext(‡<cancel‡);

comment hvis matricerne skal overleve i baggrundslageret til sener
brug for et andet program, bør den foregående sætning slettes;
end ‡<
```

Programmering

I det følgende er givet eksempler på forhold man bør tage hensyn til, når et program udarbejdes.

Numeriske hensyn. Bemærk at matematisk ækvivalente udtryk ikke i almindelighed er numerisk ækvivalente, således at væsentlige numeriske forbedringer ofte kan opnås ved simple omskrivninger. Eksempel: hvis n og N er store, næsten lige store heltal vil (N-n)/N være at foretrække for 1-n/N.

Undgå at danne differens mellem to variable, der kan være næsten lige store (ciffertab).

Hvis x eller y er real, da kan sætningen `if x=y then S1;` være meningsløs og bør erstattes med `if abs(x-y)<eps then S1;` eller `if abs(x-y)≤eps×(absx+absy) then S1;`

Bemærk også while konstruktionen i følgende

```
e:= 0;  i:= 0;
for u:=1, u×x/i while e+u≠e do
begin e:= e+u;  i:= i+1    end;
```

En for-sætning af formen

```
for t:=0 step 0.1 until 100 do
begin ... y:=f(t); ...     end;
```

er numerisk uheldig, da 0.1 ikke kan repræsenteres som en endelig binalbrøk. En bedre løsning er

```
for i:=0 step 1 until 1000 do
begin t:=i/10; ... y:=f(t); ... end;
```

Æstetiske hensyn. Generelt kan siges, at en kompliceret programstruktur ofte skyldes dårlig organisation, specielt kan der være tale om en uhensigtsmæssig datarepræsentation. Et program med mange goto sætninger er vanskeligt at læse. Man plejer i denne forbindelse at fremhæve to særlig hyppigt forekommende konstruktioner

```
if a<b then goto C else goto D;:
C: S1;
D: S2;
```

```
if x≥4 then goto A;
S1;
goto B;
A: S2;
B:
```

disse omskrives bedre til

```
if a<b then S1;  S2;
```

```
if x<4 then S1 else S2;
```

Effektivitets hensyn. Hvis man ønsker at få et Gier Algol 4 program til at køre hurtigere kan følgende anbefales.

Undgå indicerede variable hvor simple variable kan bruges.

Er programmet indkørt, kan den automatiske indekskontrol udelades (benyt n som parameter ved kald af algol).

Undgå call-by-name, dvs undersøg om procedureparametre af type integer, real og boolean ændres af proceduren. Gør de ikke det, eller har ændringerne kun betydning indenfor procedureblokken, bør parametrene være value-specificerede.

Undgå aritmetiske udtryk med variable af blandet type.

Benyt heltalsudtryk som indices.

I Gier Algol 4 kan en kort for-løkke, hvis styrede sætning er en sammensat sætning, bringes til at køre hurtigere ved som en kommentar efter det afsluttende end at skrive for (se manual):

```
for i:=1 step 1 until n do
begin   ...                 end for;
```

Er den til rådighed værende variabelplads i ferritlageret (ca 650 variable) næsten fuldt udnyttet, kan man undersøge, om en ændring af blokstrukturen kan nedsætte antallet af samtidigt levende variable, hvorved tid vil spares.

Generelt: undgå de mere komplicerede strukturer i algol til fordel for de simple, hvis køretid skal spares.

Tilsidst skal nævnes, at det i Gier Algol 4 er let at indlemme maskinkode i et algolprogram (se manual).

2.5.3 Hvordan køres algol-4 programmer i Help-3 systemet.

Som bekendt er GIER algol 4 programmer ikke direkte udførbare på GIER. Af denne grund findes der blandt standardprogrammerne i Help-3 systemet nogle oversættere (compilere) med navne som ga4 og lignende, som kan oversætte programmerne til en maskinorienteret form.

De følgende afsnit behandler oversættere og hvorledes forbindelsen mellem disse og Help-3 systemet er.

2.5.3.1 Oversættelse.

Oversætternes karakteristika er opsummeret i afsnit 2.5.3.3 . I dette afsnit beskrives deres fællestræk.

Inden en oversættelse bør KA og KB registrene nulstilles (se afsnit 1.9.4), idet der ellers kan forekomme mystiske udskrifter fra oversætteren.

En oversættelse igangsættes ved afgivelsen af en kommando til MAIN HELP, som indeholder et kald af standardprogrammet algol. Denne kommando kan bekvemt anbringes umiddelbart før det algol4-program, der skal oversættes:

```
algol< begin . . .  <
```

Den ovenfor afbildede strimmel kan indlæses og programmet oversættes blot ved tastning af r< .

En oversætter tager sine inddata fra Hjælps løbende indlæsemedium. UNDTAGELSE: hvis løbende indlæsemedium er skrivemaskinen (t) uden det udtrykkeligt er anglvet i kommandoen, vælges læseren (r) som indlæsemedium for oversætteren.

Er løbende indlæsemedium et dataområde, skal dette være på tekstform, for eksempel genereret af edit.

Oversætteren ignorerer alt indtil det første begin.

Oversætteren stopper indlæsningen, når blokstrukturen går op, dvs.

antal begin = antal end

og når antal ¢< og ¢ = antal ¢ .

Her er altså mulighed for mærkelige fejl, som kan have til årsag en syntaktisk forkert =parantesstruktur= et helt andet sted i programmet.
Eksempel:

```
begin
        .
        .
    writetext(¢< - - -  |);
        .
        .
end t<
```

Her vil programmet efter ¢< blive opfattet som parameter til writetext, og da oversætteren savner ¢ og end, vil den læse udover det end, der skulle have været det sidste.

Læg mærke til, at oversætteren kun læser til og med end . Semikolon er altså ikke nødvendigt efter det sidste end i programmet.
Da oversætteren imidlertid efter en vellykket oversættelse overlader kontrollen til MAIN HELP, der så læser videre umiddelbart efter programmet, bør der stå en passende kommando på dette sted , for eksempel t< .
Da MAIN HELP ignorerer alt i uppercase, vil et semikolon ikke skade efter end.

Bemærk at oversætteren bruger work til lagring af mellemresultater. Hvis work som på GIER1 ligger i free, vil indholdet af free efter en oversættelse altså ikke være intakt.

Når oversætteren er færdig, indtræffer en af følgende hændelser:

1) programmet var syntaktisk korrekt. Da udskrives teksten =ok=. Det oversatte program ligger klar til udførelse i work. MAIN HELP læser videre fra løbende indlæsemedium.

2) programmet var ikke syntaktisk korrekt. Udskrifter, der angiver fejlrammte konstruktioner, er da udskrevet undervejs på Hjælps udlæsemedium. De fejlramte sætninger smides væk af oversætteren. På alarmout udskrives =sorry=. MAIN HELP læser fra skrivemaskinen.

2.5.3.2 Copy.

Når man arbejder med store programmer kan det af hensyn til edit være bekvemt at have programmet i flere dele. Analogt kan en programdel, for eksempel en procedure eller plottersystemet til GIER, tænkes at bruges i mange forskellige programmer. Af disse grunde er GIER Algol4 oversætterne forsynet med en mulighed for at skifte indlæsemedium undervejs og senere vende tilbage igen til det tidligere valgte. Dette kræver dog, at kataloget er intakt.

Ønskes et sted i programmet (og dette kan ske mellem to vilkårlige algolsymboler) indkopieret den tekst, der står på et medium med et givet navn, skrives blot det pågældende sted

copy <navnet> <

Bemærk, at alle mellemrum fjernes, før oversætteren slår det op i kataloget.

Ønskes nu et sted, at oversætteren læser fra det forrige, valgte indlæsemedium, skrives blot

finis

Der må højest være 6 kald af copy ad gangen uden modsvarende finis .

Eksempel: i free ligger teksten

```
algol<
begin
      writetext( copy r<);
end t<
```

i r ligger en strimmel

```
↑<<dette er en prøve↑finis
```

Programmet vil nu blive oversat, som om det oprindelidt så sådan ud:

```
begin
      writetext(↑<<dette er en prøve↑);
end
```

2.5.3.3 Oversætterne.

OVERSÆTTER	GA4	GA4B	GA4P	TRANSIENT OVERSÆTTER
FINDES PÅ DISKETTE NR	1, 2	2	2	1
PLADSKRAV PÅ BAGGR. LAGER I KANALER	>171	>173	>173	>12
MAKSIMALE ANT. VARIABLE				
ALMINDELIGE	600	600	600	600
ARRAY		4096		
BRUGER BUFFER	NEJ	JA	JA	NEJ
BRUGES HVIS ANDET NAVN IKKE ANGIVES				
GIER 1	JA	-	-	-
GIE2	-	JA	-	-

GA4P BENYTTER BUFFEREN TIL LAGRING AF PROGRAM.

2.5.4 Litteratur.

[1] S. Lauesen (ed.) :
A Manual of Help 3.

[2] P. Naur (ed.) :
A Manual of GIER Algol 4.

[3] P. Naur (ed.) :
Revised Report on The Algoritmic Language Algol 60.

[4] A Manual of GIER Programming I-III

[5] S. Lauesen :
Datamatik.

Flexowriterkoden. Den følgende tabel giver den numeriske kode for flexowritersymbolerne og de specielle kontrolsymboler. Desuden er strimmelkoden svarende til den numeriske kode givet (binær repræsentation med ulige paritet). LC og UC står for lower case og upper case.

strimmel kode	tal kode	symbol LC	symbol UC
o .	0	SPACE	
. o	1	1	v
. o	2	2	×
o . oo	3	3	/
.o	4	4	=
o .o o	5	5	;
o .oo	6	6	[
.ooo	7	7	]
o.	8	8	(
oo. o	9	9	)
oo. o	10		
o. oo	11	STOPCODE	
oo.o	12	ENDCODE	
o.o o	13	å	Å
o.oo	14	_	|
oo.ooo	15		
o .	16	0	^
oo . o	17	<	>
oo . o	18	s	S
o . oo	19	t	T
oo .o	20	u	U
o .o o	21	v	V
o .oo	22	w	W
oo .ooo	23	x	X
ooo.	24	y	Y
o o. o	25	z	Z
o o. o	26		
ooo. oo	27	,	[10]
o o.o	28	CLEARCODE	
ooo.o o	29	REDRIBBON	
ooo.oo	30	TAB	
o o.ooo	31	PUNCHOFF	
.		NOHOLES	

strimmel kode	tal kode	symbol LC	symbol UC
o .	32	-	+
o o . o	33	J	J
o o . o	34	k	K
o . oo	35	l	L
o o .o	36	m	M
o .o o	37	n	N
o .oo	38	o	O
o o .ooo	39	p	P
o oo.	40	q	Q
o o. o	41	r	R
o o. o	42		
o oo. oo	43	ø	Ø
o o.o	44	PUNCHON	
o oo.o o	45		
o oo.oo	46		
o o.ooo	47		
ooo .	48	æ	Æ
oo . o	49	a	A
oo . o	50	b	B
ooo . oo	51	c	C
oo .o	52	d	D
ooo .o o	53	e	E
ooo .oo	54	f	F
oo .ooo	55	g	G
oo o.	56	h	H
oooo. o	57	i	I
oooo. o	58	LOWERCASE	
oo o. oo	59	.	:
oooo.o	60	UPPERCASE	
oo o.o o	61	SUMCODE	
oo o.oo	62	BLACKRIBBON	
oooo.ooo	63	TAPEFEED	
o .	64	CAR.RETURN	

Printersymboler. Printeren har ingen små bogstaver. Specielle symboler er: 12 i LC,UC: Ü, 15 i LC: % og UC: &, 44 i LC: ‡ og UC: '. Printerkontrolsymboler: CAR.RETURN (=64) giver almindeligt linieskift. Papirfremførsel iøvrigt fås ved til 64 at addere en eller flere af følgende værdier, 1, 2, 4, 8, som hver svarer til en kanal på styrestrimlen, kanal 2, 4, 6, 8 hhv. Virkningen vil være, at printeren laver linieskift indtil den møder et hul i den til symbolet svarende kanal på styrestrimlen. Karakteren 80 bevirker udskrift af en linie, men ingen papirfremførsel. TAB (=30) har special virkning på printeren idet det følgende symbol multipliceret med 2 vil blive brugt til at angive tabulatorpositionen. _ og | optager een position på printeren.