NGU-rapport nr. 86.128

PC som arbeidsstasjon til ionekromatograf

Norges geologiske undersøkelse

Leiv Eirikssons vei 39, Postboks 3006, 7001 Trondheim - Tlf. (07) 92 16 11
Oslokontor, Drammensveien 230, Oslo 2 - Tlf. (02) 50 25 00

Rapport nr. 86.128	ISSN 0800-3416	Åpen/[redacted]

Tittel:

PC som arbeidsstasjon til ionekromatograf.

Forfatter:	Oppdragsgiver:
Helge Iversen, Teknisk data	Geokjemisk avd., seksjon for kjemiske analyser

Fylke:	Kommune:
Kartbladnavn (M. 1:250 000)	Kartbladnr. og -navn (M. 1:50 000)
Forekomstens navn og koordinater:	Sidetall: 121 Pris: kr.150.00 Kartbilag:

Feltarbeid utført:	Rapportdato: 13.06.86	Prosjektnr.: 4.1.2086.21	Prosjektleder: B. Andreassen

Sammendrag:

Rapporten tar utgangspunkt i dagens rutiner i forbindelse med ionekromatografi. Den tar for seg svakheter, mangler og hvordan rutinene kan forbedres ved å ta i bruk EDB. Rapporten er stort sett EDB-teknisk.

Emneord		
	EDB	IONEKROMATOGRAFI
SYSTEMERING	KRAVSPESIFIKASJON	AVHANDLING

I N N H O L D

(**)

APPENDIX

A: ARBEIDS INSTRUKSJONER

- Litt om notasjon
- Oppstart av PCION
- Avslutting av PCION
- Operativsystem kommandoer
- Klargjøring av skriveren
- Klargjøring av ny arbeidsdiskett
- Klargjøring av ny systemdiskett
- Manuell BACKUP av arbeidsdisketter
- Manuell BACKUP av systemdisketten
- Kopiering av systemdisketten
- Justering av klokken
- Justering av datoen

B: DISKETT BRUK, KARTOTEK

- Håndtering av disketter
- Slitte disketter
- Systemdiskett
- Disketter til brukerfiler
- Arbeidsdiskett

C: INTERRUPT STYRT DATA LOGGING

D: DATABLAD TIL MM58167 RT-KLOKKEN

E: DATABLAD TIL 8259A INTERRUPT CONTROLLER

F: RUTINEBIBLIOTEK og PROCEDURER

INNLEDNING

Ved seksjon for kjemiske analyser ved NGU er det en ionekromatograf i drift stort sett daglig. En del sider ved arbeidet er tungvindte og lite rasjonelle. Spesiellt er lagring av metoder og analyseresultater samt rutinene for bearbeiding av målinger og overføring av data til NGU's sentrale data-anlegg, HP-3000, lite tilfredstillende. Ionekromatografen arbeider sammen med en integrator etter gitte metoder som lagres i integratoren. Dersom det blir strømbrudd eller kraftig tordenvær går metodene tapt, og metodene må tastes inn på nytt. Analyseresultatene må i dag tastes inn manuellt på HP-3000. Behovet for ionekromatografiske analyser er stort, og det ventes å øke. på denne bakgrunn ble det besluttet å legge rutinene over på EDB.

Denne rapporten utreder rutinene slik de er i dag, hva som ikke er tilfredstillende, mangler og en mulig løsning på problemene. Systembeskrivelsen er en kort oppsummering av rutinene i dag og hvordan det kan bli dersom datasystemet PCION benyttes. Kravspesifikasjon er en detaljert beskrivelse av hvordan PCION vil virke med hensyn på skjermbilder, utskrifter, filstruktur osv. Kravspesifikasjonen er laget i nært sammarbeid med prosjektleder Birger Th. Andreassen ved seksjon for kjemiske analyser.

Helge Iversen 12.06.86

(*'*)

SYSTEM BESKRIVELSE

IONEKROMATOGRAF, IONEKROMATOGRAFI

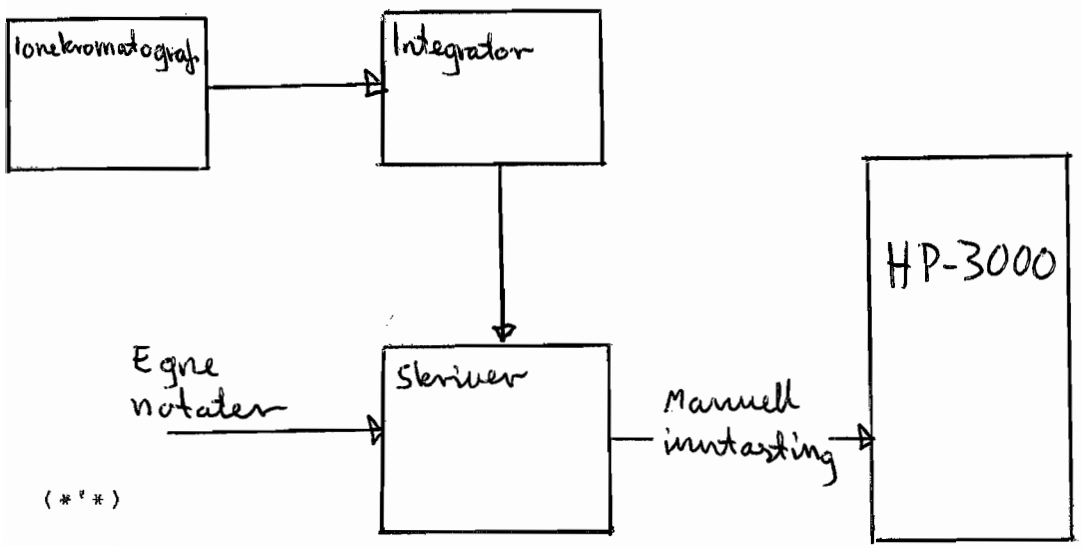
Ionekromatografi er en kjemisk analyseteknikk som benyttes til skille ulike ioner fra hverandre og å bestemme konsentrasjonen.

Prinsippet for teknikken er at de forsjellige ioner i en prøveløsning vandrer med forskjellig hastighet når de ved hjelp av en løsning som kalles eluent, vaskes igjennom en kolonne fyllt med en ionebytter. De ulike ioner kommer derfor ut av kolonnen i tur og orden med konsentrasjonsmaksima til tider som er bestemt ved kromatograferingsbetingelsene. Kolonne, eluent og gjennomstrømningshastighet er vesentlige faktorer i denne sammenheng. Utgående eluent-strøm overvåkes kontinuerlig av en detektor som gir konsentrasjonsavhengig signal, vanligvis elektrisk spenning. Ved at detektorens respons tegnes ut på papir som funksjon av tiden, fås et kromatogram. Ionene identifiseres ved toppenes plass på kromatogrammets tidsakse. Toppenes høyde eller areal angir ionenes konsentrasjon, idet forholdet mellom konsentrasjon og toppstørrelse (responsfaktoren), for de enkelte ion er bestemt på forhånd ved kjøring av standarder. Standarder er løsninger med kjent sammensetning. Disse kjøringer fastlegger også topptidene for de aktuelle ioner. Med topptid forstås tiden regnet fra det øyeblikk prøveveksleren aktiviseres for inntak av prøveløsning til maksimalt utslag. Denne er forsjellig fra retensjonstid som regnes fra det tidspunkt prøveløsningen injiseres ved kolonnens topp.

Det finnes mange typer detektorer. En vanlig og meget anvendelig detektor er et instrument for måling av elektrisk ledningsevne med mikrocelle. NGU's ionekromatograf er inntil videre kun utstyrt med ledningsevnedetektor. Eluentens ledningsevne-bidrag, dvs. bakgrunnen, reduseres til et minimum v.hj.a. en dempningskolonne montert mellom seperasjonskolonnen og ledningsevnecellen.

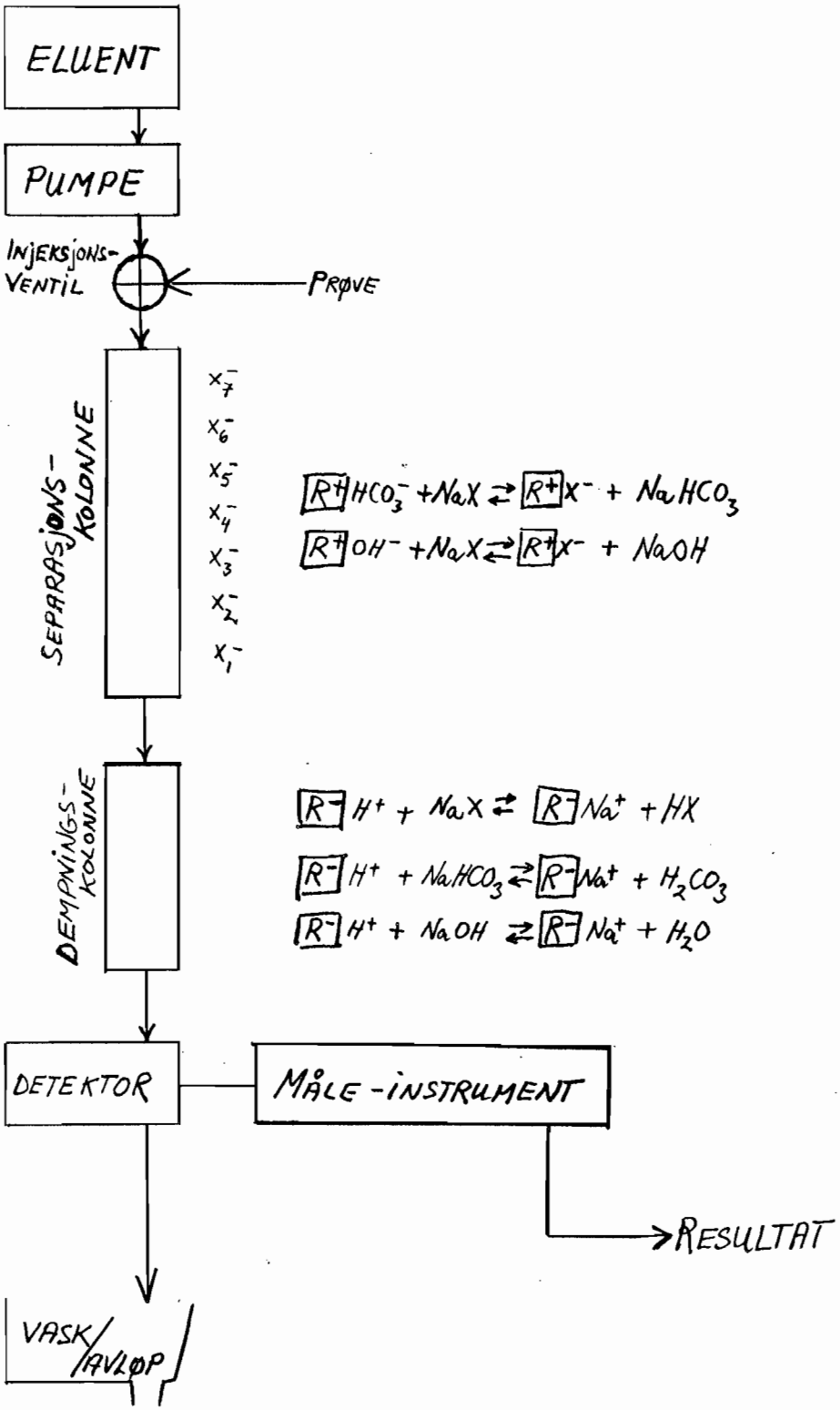
En skjematisk framstilling av ionekromatografen er vist på neste side. "Resultat" er en spenning som er koblet til integratoren. Kolonnene i ionekromatografen er utskiftbare, og det finnes en rekke typer til forskjellige analyseoppgaver.

SYSTEM I DAG



(*'*)

IONEKROMATOGRAFI
(ANIONER)



Integrator, Metoder

Integratoren er en regne-enhet tilsluttet ionekromatografen. Den arbeider etter gitte metoder (parametre) som tastes inn i integratoren. I integratoren er det plass til 9 metoder samtidig. Dersom det oppstår strømbrudd eller sterkt tordenvær må alle metodene tastes inn på nytt.

Presentasjon av resultater

Det er pr i dag koblet en printer/plotter til integratoren. Figuren på neste side viser et eksempel på to analyseresultater presentert på denne skriveren.

Vi ser at det er så høye konsentrasjoner at flere topper ligger utenfor figuren. Det er da umulig å se kurvens forløp, og i visse tilfeller må prøveløsningen tynnes ut og analyseres på nytt. Vi ser også at integratoren har utelatt å beregne konsentrasjonen for toppen med topptid = 6.81 i første resultat. Dette kan i de fleste tilfeller beregnes manuellt (11.8862).

Lagring av resultater

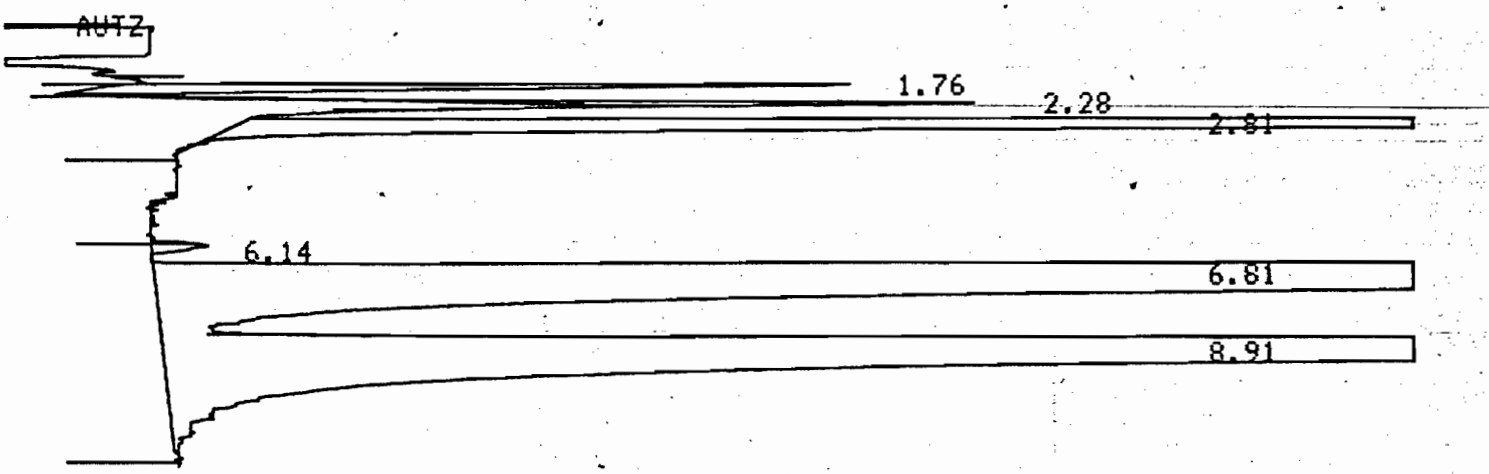
Skriveren benytter ruller av thermo-papir. Dette medfører at etter en dags arbeid ved ionekromatografen, har man resultatene på en flere meter lang papir-rull som ikke egner seg for lagring på noen måte. Papiret er ikke sidedelt slik at resultatene må oppbevares i ruller. Dessuten er papiret varmefølsomt og mørkner med tiden. Dersom papiret utsettes for derikte sollys vil teksten ganske raskt bli visket bort.

Brukerfil, Analyserapport

Når alle prøvene under ett oppdrag er analysert skal resultatene legges på brukerfil på HP-3000. Brukerfilene bearbeides av programvare og benyttes bl.a. til kartfremstilling. For å generere brukerfilen må alle resultatene tastes inn på HP-3000. På grunnlag av brukerfilen blir det generert en analyserapport. Analyserapporten kan ikke inneholde flere enn 8 forsjellige ionekonsentrasjoner for hver prøve p.g.a. plassmangel. Dette er en uheldig begrensning når det skal analyseres mer kompliserte løsninger.

(*'*)

IEEE NO	SAMPLE	METHOD	TIME	DATE
31	0005	METHOD1	11:32:28	10:03:86



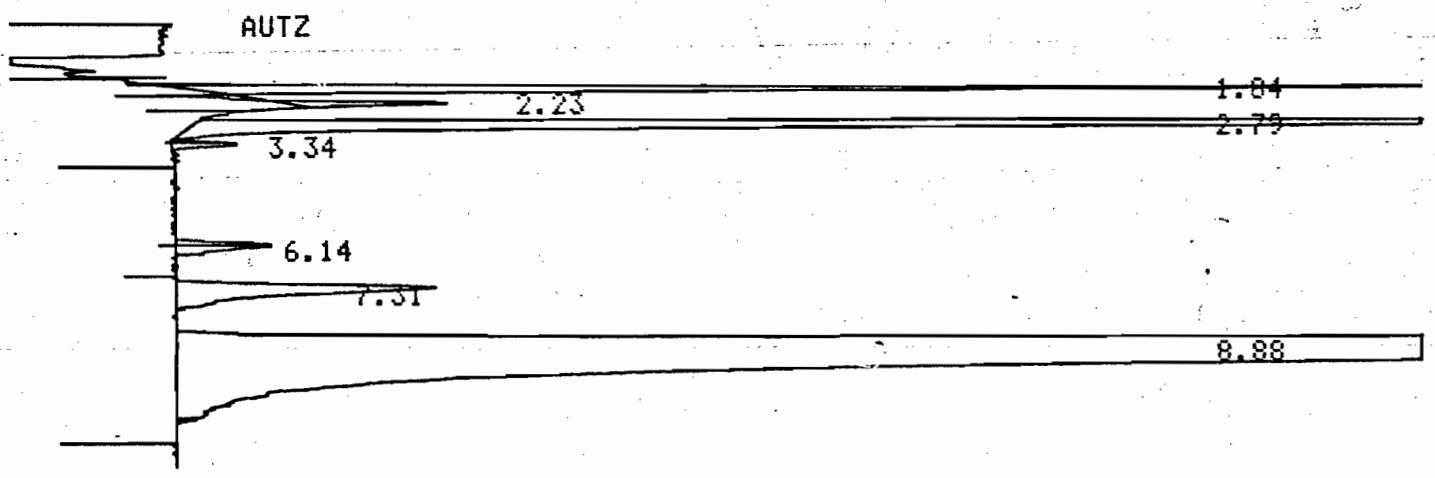
RUN TIME 12.00

EXTERNAL STANDARD METHOD USING AREA

TIME	HEIGHT	AREA		CONC
1.76	333	5617		
2.28	301	5480		
2.81	38809	253892		9.82603602
6.14	103	654	P	0.05810750
6.81	11171	165754	P	
8.91	15937	196335		10.69044075
TOTAL		627732		20.57458427

11.8862

IEEE NO	SAMPLE	METHOD	TIME	DATE
31	0010	METHOD1	12:35:06	10:03:86



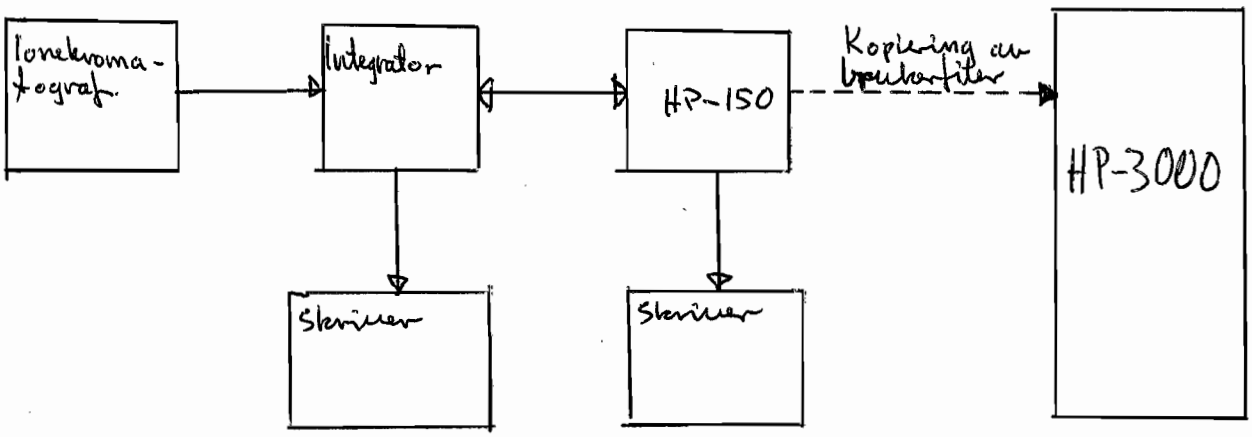
RUN TIME 12.00

EXTERNAL STANDARD METHOD USING AREA

TIME	HEIGHT	AREA		CONC
1.84	567	8476		0.22468180
2.23	104	2128		
2.79	5569	35710		1.38203545
3.34	53	502		0.04758293
6.14	161	1129	P	0.10031097
7.31	416	4077	P	0.29236167
8.88	15648	189144		10.29889080
TOTAL		241166		12.34586364

SYSTEM MED PCION

PCION er et datasystem som skal fjerne de svakheter analyseprosessen har i dag, og optimalisere utbyttet av analysearbeidet.



Integrator, Metoder

Når PCION slås på vil alle metodene ligge trykt i integratoren uansett om det har vært strømbrudd. Nye metoder kan tastes inn, gis navn og lagres på diskett. Opptil 9 metoder kan ligge i integratoren samtidig. Man kan få utlisting av alle metoder på disketten og integratoren.

Presentasjon av resultater

Analyseresultatene kontrolleres via PC'en. Kromatogrammet tegnes på skjermen slik at hele kurven vises. Kromatogrammet kan deretter skaleres og flyttes rundt på skjermen slik at hele kurven kan studeres. Dersom ønskelig kan kurven tegnes på skriver sammen med konsentrasjoner etc. på A4 format. Analyseresultatet må godkjennes eller forkastes. Analyse-resultatene lagres på disketter og kan senere brukes til utlisting av resultater eller til søking etter bestemte ioner. Søking etter ioner innebærer at om man f.eks. er interresert i fluor, da fås en liste over alle prøvene som inneholder fluor inkl. konsentrasjon, analysedato etc. Man kan videre få en liste over alle oppdrag m/opplysninger som er registrert på disketten. I kravspesifikasjonen er alle skjermbilder og utlistinger beskrevet i detalj.

Lagring av resultater

Alle godkjennte analyseresultat lagres på diskett. Se appendix B for nærmere beskrivelse av diskett-kartotek. Alle utskrifter skrives ut på A4-ark og kan lett oppbevares i permer.

Brukerfil, Analyserapport

Brukerfil og analyserapport genereres automatisk når tilhørende oppdrag avsluttes. Analyserapporten kan inneholde inntil 50 forsjellige ione-konsentrasjoner for hver prøve.

KRAV SPESIFIKASJON

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
PCION	PC som arbeidsstasjon til ionekromatograf	H Iversen DATO 3/86	1	PCION/1

MÅL

- Forenkle analysearbeid, metodeundersøkelser og rapportering.

- Lette den manuelle kontroll og bearbeiding av analysedataene via PC'en, og få til en mest mulig automatisk overføring av de aktuelle data til brukerfil på HP-3000.

- Gi mulighet for utlisting av analyseresultater samt søking etter enkelte ioner.

BESKRIVELSE

PCION er et datasystem som bl.a. skal forenkle prosessen fra analyse til analyserapport.
Ionekromatografen arbeider etter gitte metoder som ligger i integratoren. Disse metodene skal lagres på diskett via PC'en. Inntasting av en metode første gang vil fortsatt skje fra integratoren. På PC'ens disketter vil alle ananlyse-resultater og metoder bli lagret.
Kontroll og vurdering/bearbeiding skjer via PC'en. Under kontroll av resultatene vil det være mulig å få tegnet kurvene (i ønsket skala) på skriver sammen med resultatene.
Etter at en analyse er kontrollert (event. modifisert) og godtatt eller forkastet vil det ikke være mulig å hente fram selve kurven igjen, kun resultatet (p.g.a. plasskrav).
Når alle analysene tilhørende ett oppdrag er utført og kontrollert overføres de til HP-3000 on-line (i første omgang via diskett pga. støy på linjen).

BRUKERVENNLIGHET

PCION vil starte automatisk når PC'en slås på. Dialogen mellom brukere og systemet vil stort sett være menyorientert. Ledetekster og feilmeldinger skal være på Norsk. Alle meldinger vil bli skrevet ut på linje 1.

SIKKERHETSKRAV

Dersom data lagret på diskett tapes eller ødelegges skal gårsdagens sikkerhetskopi tas inn. Dette kan gjøres av brukerne. Spesiell adgangskontroll er ikke nødvendig.

TILGJENGELIGHET

Systemet kan brukes hele døgnet.

MASKINUTSTYR

På laboratoriet benyttes en PC (HP-150) med dobbel diskettstasjon og en HP ThinkJet skriver.
Ellers benyttes eksisterende utstyr: HP-3000 med skrivere, DIONEX IONEKROMATOGRAF, LDC/Milton Roy CI-10B integrator og en Milton Roy skriver. Denne skriveren vil være overflødig dersom systemet bygges ut til også å omfatte utskrift av selve metodeparametrene.

(*'*)

BENEVNINGER

På utlistinger/rapporter og skjermbilder kan benevningene variere, men på alle filer inkl. integrator benyttes følgende benevning.

Variabel	Benevning
Ionekonsentrasjon	Parts Pr Million (ppm)
Bestemmels grense	ppm
Tid	minutt (med desimaler i 1/100 minutt)
Høyde	ingen betydnig
Areal	ingen betydnig

For konsentrasjon benyttes følgende benevninger i skjermbilder og utskrifter.

ppt	= 1E-6 * ppm	
ppb	= 1E-3 * ppm	
ppm	= 1 ppm	(parts pr million)
%	= 1E+4 * ppm	

I de neste kapitler er benevning antydet med :

(a) xxxxbbb (kun i analyserapporten)
(b) xxxxx bbb

der bbb er benevning og x'ene er siffer.

Formatet på sifferene er:

(a) for analyserapporten x.xx
xx.x
xxx

(b) ellers x.xxx
xx.xx
xxx.x

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
		H Iversen		
PCION	Hovedmeny	DATO 4/86	2	HMENY

BESKRIVELSE

Når PC'en slås på vil integratoren bli lastet med de metodene den inneholdt sist den ble benyttet, dette for å være sikker på at et eventuellt strømbrudd ikke har slettet noen av metodene. Metodene vil bli plassert under de samme metodenummerene som sist.

Alle Ja/Nei spørsmål må besvares med følgende bokstaver (j,J,y,Y, n,N).

Retur til hovedmeny kan skje når som helst ved å trykke -1 i ett variabel-felt.

Skriveren vil bli initialisert med sidestørrelse, tegnstørrelse etc. Deretter vil følgende meny vises på skjermen.

```
H O V E D  -  M E N Y


        Metode manipulering ..................................... <M>
        Toppnavn, lagring/endring/sletting ...................... <T>

        Registrering av nytt oppdrag ............................ <R>
        Oppstart av ionekromatograf ............................. <O>
        Kontrollering av resultat ............................... <K>
        Avslutting av oppdrag ................................... <A>
        Sletting av brukerfil ................................... <S>

        List analyseresultat .................................... <L>
        List oppdrag ............................................ <P>
        List ioner .............................................. <I>

        Avslutte ................................................ <Q>
```

INNDATA Ett tegn som angir valg. Se detaljert rutinebeskrivelse i kap 2.

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
		H Iversen		
PCION	BACKUP	DATO 4/86	2	BACKUP

BESKRIVELSE

BACKUP anvendes for å ta sikkerhetskopi av system- og arbeidsdisketten. Denne rutinen utføres hver gang PCION avsluttes. Det tas kun BACKUP av en arbeidsdiskett i rutinen, dersom det er benyttet flere disketter må det tas manuell backup av de øvrige.

```
S I K K E R H E T S  -  K O P I E R I N G

1. Ta ut systemdisketten i venstre drive
2. Plasser ARBEIDS-BACKUP-disketten i venstre drive
3. Trykk 3 ganger RETURN

4. Ta ut begge diskettene
5. Plasser systemdisketten i venstre drive og
   SYSTEM-BACKUP-disketten i høyre drive
6. Trykk 3 ganger RETURN

7. Sett tilbake arbeids-disketten i høyre drive og trykk RETURN


 NØDUTGANG   tast -1 istedenfor RETURN
```

BRUKSFREKVENS 1 gang pr. dag.

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
PCION	Metode manipulering	H Iversen DATO 4/86	2	METODE

BESKRIVELSE

Denne rutinen benyttes til å utveksle metoder mellom integratoren og HP-150, og til utlisting av metoder og eventuellt plasseringen i integratoren.

```
!                                                                    !
!                                                                    !
!                                                                    !
!                  M E T O D E   M A N I P U L E R I N G             !
!                                                                    !
!                                                                    !
!                List alle metoder ..... <M>                         !
!                                                                    !
!                List metoder i integrator ..... <R>                 !
!                                                                    !
!                Legg metode på diskett ..... <D>                    !
!                                                                    !
!                Legg metode i integrator ..... <I>                  !
!                                                                    !
!                Slett metode på diskett ..... <S>                   !
!                                                                    !
!                Avslutt ..... <A>                                   !
!                                                                    !
!                                                                    !
!____________________________________________________________________!
```

Menyvalg S

INNDATA Metodenavn

Menyvalg M og R

INNDATA ingen

UTSKRIFT se 4/LISTMET

Menyvalg D og I

```
!                                                                    !
!                                                                    !
!                                                                    !
!    Metode nummer .....: 9                                          !
!                                                                    !
!    Metode navn .....: ANIONER I VANN                               !
!                                                                    !
!                                                                    !
!____________________________________________________________________!
```

INNDATA se 3/SYSDATA

UTDATA ingen

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
	Toppnavn	H Iversen		
PCION	lagring, endring etc.	DATO 4/86	2	TOPNAV

BESKRIVELSE

TOPNAV anvendes for å sette navn (kjemisk symbol) på aktuelle topper. Dette letter dokumentasjon av analyseresultatene, (automatisk navnsetting).

Under kolonne for topptid er følgende kommandoer tilgjengelig :

Ø = Linjen slettes,
-1 = Avslutt, topptabellen lagres
-2 = Avslutt, topptabellen lagres og listes ut på skriver

TOPP TID	TOPP TID TOL. %	ION
1.84	2	F'
2.74	1	Cl'
.	.	.
.	.	.
.	.	.
.	.	.
.	.	.
.	.	.

INNDATA se 3/SYSDATA

UTLISTING se 4/TOPPLST

BRUK

Ved trykk på RETURN vil kursoren bevege seg fra linje til linje nedover. Dersom en linje skal endres kan en skrive inn de nye verdiene. Ellers kan det under kolonne for topptid skrives:

Ø	=>	Linjen slettes,
-1	=>	Avslutt, topptid-tabellen lagres.
-2	=>	Avslutt, topptid-tabellen lagres og skrives ut på skriver.

Det kan maksimalt være 50 forsjellige topptider, dersom det er topp-intervaller som overlapper hverandre vil det første påtruffne intervall bli valgt.

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
		H Iversen		
PCION	Registrering av oppdrag	DATO 4/86	2	REGOPDR

BESKRIVELSE

REGOPDR anvendes til å registrere ett nytt oppdrag. Det kan ikke kjøres analyser under ett oppdrag som ikke er registrert. Når ett oppdrag registreres vil noen av de data som senere vil utgjøre brukerfilens hode bli lagt i oppdragstabellen, REG 7.

```
                R E G I S T R E R I N G   A V   N Y T T   O P P D R A G


                Oppdrags nummer .: 133/86
                Oppdragsgiver ...: NORSKE SKOG A/S

                Prosjekt nummer .: NGU1986A

                Prøvetype .......: Nedbørsvann

                Antall ionenavn .: 7

                                                TOPP TID
                ION           TOPP TID           TOL %
                ---------------------------------------
                F'              1.84               2
                Cl'             2.98               3
                Br'             6.14               1
                .                .                 .
                .                .                 .
```

BRUK

Navnene på ionene tastes inn. Dersom navnet er kjent for systemet vil topptid og toleranse automatisk bli utfyllt. Verdiene kan ikke endres her, se 2/TOPNAV. Hvis navnet er ukjent må topptid og toleranse oppgis.

INNDATA Se 3/SYSDATA

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
		H Iversen		
PCION	Start ionekromatograf	DATO 4/86	2	STION

BESKRIVELSE

STION anvendes til å starte ionekromatografen. Ledig diskplass i % fylles ut av systemet. Det er analytikers ansvar at det er plass på disketten til å ta imot rådata. Dersom disketten går full, vil analyse-resultatene gå tapt. Avhengig av om det kjøres standarder eller vanlig oppdrag, vil ett av følgende skjermbilder vises.

```
S T A R T   A V   I O N E K R O M A T O G R A F

     Ledig diskplass i % ............: 55

     Standard ? .....................: N

     Oppgragsnummer .................: 133/86

     Første prøvenummer .............: 1050
```

```
S T A R T   A V   I O N E K R O M A T O G R A F

     Ledig diskplass i % ............: 55

     Standard ? .....................: J

     Antall standarder (1..40).......: 3

     STD nummer          Fortynnings faktor
     ------------------------------------
     AS100               1
     AS100               10
     AS101               25
       .                 .
       .                 .
```

INNDATA se 3/SYSDATA

UTSKRIFT ingen

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
PCION	Kontroll av resultat	H Iversen DATO 4/86	2	KONTR/1

BESKRIVELSE

KONTR benyttes til å kontrollere/vurdere analyse resultater. Dersom analysen gjelder en standard vil den første linjen inneholde STD-nummer og fortynningsfaktor.

Analyseresultatene må kontrolleres i den rekkefølgene analysene ble utført, dette for å sikre at ingen analyser blir uteglemt. Dette skjer automatisk.

```
  Oppdrag : 133/86          NORSKE SKOG A/S
  Prøve   : 2500            Dato :  86.12.31            Kl :  12:45
 ____________________________________________________________________
0.0 !
    !
2.0 !
    !
4.0 !                    ( PLASS FOR KURVEN )
    !
6.0 !
    !
8.0 !
        Tid min : 0.0           Høyde min : 0
        Tid max : 8.0           Høyde max : 2.5E6        Skriver ? : N

   TOPP TID     ION              HØYDE          AREAL       KONS. (ppm)
   ---------------------------------------------------------------------
    1.23       F'                564564          64646          24.1234567
    2.34       CH3COO'       12345678901    12345678901    1234567.1234000
    .          .                 .              .              .
    .          .                 .              .              .
    .          .                 .              .              .
    .          .                 .              .              .
```

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
		H Iversen		
PCION	Kontroll av resultat	DATO 4/86	2	KONTR/2

BRUK

Tid min/max og Høyde min/max vil automatisk bli beregnet slik at hele kurven vises på skjermen.

Disse parametrene kan velges fritt slik at hvilket som helst område av kurven kan vises i ønsket skala.

Når kurven er tegnet på skjermen vil kursoren plasseres til venstre for første detalj-linje. Dersom det nå trykkes på mellomrom-tasten vil kursoren flytte seg til "tid min" og de fire kurve-parametrene kan endres, og kurven tegnes på nytt. Ved trykk på en hvilken som helst annen tast medfører det at kursoren plasseres under feltet ion. Feltene ion og konsentrasjon kan nå endres. Dersom en detalj-linje ønskes fjernet, tastes det -2 i feltet for konsentrasjon.
Kursoren vil nå flytte seg til neste detalj-linje og samme prosess gjenntas.

Når alle detalj-linjene er kontrollert vil systemet markere de ionene som oppdragsgiver har forespurt med en stjerne "*". Systemet vil kontrollere at alle forespurte ioner er representert, og hvis ikke det er tilfelle, kan analyseresultatet ikke godkjennes. Denne ekstra kontrollen garanterer at de riktige resultat overføres til brukerfil.

Når siste detaljlinje er kontrollert vil følgende meny vises på/over kurvens høyre side.

```
                                                    Godkjenne ... <G>
                                                    Forkaste .... <F>
                                                    Kurve param.. <K>
                                                    Neste analyse <N>
                                                    Avslutte .... <A>
```

Der "kurve param." innebærer mulighet for flere utskrifter av kurven til skriver.

INNDATA se 3/SYSDATA

UTSKRIFT se 4/KURVLST

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
		H Iversen		
PCION	Avslutting av oppdrag	DATO 4/86	2	AVSLOP/1

BESKRIVELSE

Når ett oppdrag avsluttes vil den endelige brukerfilen lages. Brukerfilens hode genereres på grunnlag av REG 7. Deretter legges alle analyseresultatene på filen, sortert. Denne filen blir kopiert over til HP-3000 der den blir flettet sammen med en koordinatfil som inneholder prøve_ nummer og koordinater. Disse data blir bl.a. brukt til kartfremstilling.

Systemet vil presentere alle ioner som oppdragsgiver har forespurt. Analytiker må da taste inn bestemmelses grensen for hvert ion.

På brukerfilen skal det kun være ett analyseresultat for hver prøve. Analyser som er tatt to eller flere ganger kan ikke automatisk overføres til brukerfil. Disse blir vist på skjermen sammen med en kode som må tastes inn, for på den måten å peke ut hvilket resultat som skal overføres til brukerfilen, se AVSLOP/2. Analyseresultatene sorteres før de legges på brukerfilen.

Det kan registreres flere analyseresultat etter at oppdraget er avsluttet. For å få disse resultat med på brukerfilen må oppdraget avsluttes igjen.

```
_________________________________________________________________________________
!                                                                               !
!                                                                               !
!          Oppdragsnummer ..........: 133/86                                    !
!          Prosjekt navn ...........: Norkalott prosjektet                      !
!          Analytiker ..............: Birger Th Andreassen                      !
!          Seksjons sjef ...........: Ola Normann                               !
!          Antall analyserapporter..: 2                                         !
!                                                                               !
!          Kommentarer :                                                        !
!          xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx         !
!          xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx         !
!          xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx         !
!          xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx         !
!          xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx         !
!          xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx         !
!                                                                               !
!          IONE NAVN     BESTEMMELSES GRENSE  (ppm)                             !
!          ----------------------------------------                             !
!          F'                 1e-03                                             !
!          Cl'                0.34                                              !
!          Br'                215                                               !
!           .                  .                                                !
!           .                  .                                                !
!_______________________________________________________________________________!
```

BRUK

Ione navnene blir skrevet på skjermen og bruker må da taste inn grensene. Når skjermbildet er ferdig utfyllt gis det melding om at det kan settes inn en ny diskett i venstre drive. Brukerfilen blir lagt på denne disketten. Når brukerfilen er ferdig gis det melding om å sette tilbake systemdisketten i venstre drive.

INNDATA se 3/SYSDATA

UTSKRIFT se 4/ANRAPP

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
PCION	Avslutting av oppdrag	H Iversen DATO 4/86	2	AVSLOP/2

BESKRIVELSE

Dersom det er tatt reanalyser, vil de forsjellige resultatene vises på skjermen som vist nedenfor. Dersom en prøve er analysert flere enn 6 ganger, vil kun de 6 siste vises, og det må velges blandt en av dem.

```
!                                                                              !
!                                                                              !
!              NB!  prøve nummer 1342 er analysert  3 ganger !                 !
!                                                                              !
! KODE            A          B          C                                      !
! ---------------------------------------------------------------------------- !
! Dato        86.12.30   86.12.30   86.12.31                                   !
! Kl             12:45      12:59       9:34                                   !
! F'         xxxxx bbb  xxxxx bbb  xxxxx bbb  .........  .........  .........  !
! Cl'        xxxxx bbb  xxxxx bbb  xxxxx bbb                                   !
!  .              .          .          .                                      !
!  .              .          .          .                                      !
!                                                                              !
!                                                                              !
!                                                                              !
!                                                                              !
!                                                                              !
!                                                                              !
!                                                                              !
!    Velg riktig analyseresultat, tast inn kode ..: B                          !
!______________________________________________________________________________!
```

INNDATA Ett tegn som angir kode.

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
		H Iversen		
PCION	Sletting av brukerfil	DATO 4/86	2	SLBRK

BESKRIVELSE

SLBRK benyttes til å slette en brukerfil fra disketten i venstre drive. Brukerfilen må være overført til HP-3000 ellers går den tapt.

En tapt brukerfil kan regenereres ved å "Avslutte oppdraget".

INNDATA Oppdrags nummer

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
PCION	List analyse resultat	H Iversen DATO 4/86	2	LIRES

BESKRIVELSE

LIRES benyttes til å få en utlisting av ønskede analyser på fil. Dersom det tastes RETURN på listfil blir utlistingen skrevet ut på skriver. Det kan ikke skrives ut analyseresultat som ikke er kontrollert og godkjent.

```
U T L I S T I N G   A V   A N A L Y S E   R E S U L T A T

        Kun standarder ..... <S>

        Kun oppdrags analyser ..... <O>

        Alle analyser ..... <A>


        Fra dato : 860422
        Til dato : 860431

        Oppdrags nummer : 133/86
        STD nummer .....: AS101

        Listfil .....: res_idag
```

BRUK

STD-nummer og oppdragsnummer er begrensninger i tillegg til fra/til-dato. Alle disse parametrene er valgfri (trenger ikke å oppgis).

INNDATA se 3/SYSDATA

UTSKRIFT se 4/LISTRES

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
		H Iversen		
PCION	List ioner	DATO 5/86	2	IONSØK

BESKRIVELSE

IONSØK anvendes til å søke etter ett bestemt ion. Det søkes enten blandt standardene eller under ett oppdrag. Oppdragsnummer eller STD-nummer må oppgis.

```
I O N E   S Ø K I N G

Fra dato : 860303

Til dato : 860631

Ion .....................: Cl'

Standard/Oppdrag (S/O) .: O

Oppdrags nummer .........: 133/86

(   STD nummer ..............: AS101    )
```

BRUK

Fradato,tildato og ion kan utelates. Dersom ion utelates søkes det etter navnløse topper.

INNDATA se 3/SYSDATA

UTSKRIFT se 4/IONLST

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
		H Iversen		
PCION	System data	DATO 4/86	3	SYSDATA/1

BESKRIVELSE

SYSDATA inneholder en beskrivelse av alle data som systemet lagrer på filer, pluss noen felter som bare brukes til dokumentasjon av utskrifter og rapporter.

Tegnforklaring :

T = Tastatur
B = Beregnes
i = Integrator
P = PC'en

x = forekomst
V = forekomst kun ved Vanlige prøver
S = forekomst kun ved Standarder

FELT	TYPE	LENGDE	KILDE	REGISTRE 1	2	3	4	5	6	7	8
Oppdrags_nr	A+N	6	T	V			x			x	
Analytiker	A	35	T								
Seksjons_sjef	A	35	T								
Bestem_grense	N	REAL	T				x				
Kommentarer	A+N	6*60	T								
Ant_ione_slag	N	INT	T								
Oppdrags_giv	A+N	45	T				x			x	
Prosjekt_nr	A+N	10	T				x			x	
Prosjekt_navn	A+N	35	T								
Prøve_type	A+N	45	T							x	
Metode_nr	N	INT	T					x			
Metode_navn	A+N	50	T					x			
*Met_f_navn	A+N	8	B					x			
Topp_tid_tab	N	REAL	T								x
Topp_t_tol_tab	N	INT	T								x
Analyse_dato	A	6	P	x	x	x					
Analyse_kl	A	4	P	x	x	x					
Tid_min	N	REAL	T								
Tid_max	N	REAL	T								
Høyde_min	N	REAL	T								
Høyde_max	N	REAL	T								
Prøve_nr	N	INT	T,B	V	x		x				
Ant_reg_topp	N	INT	B	x							
+Topp_tid	N	REAL	i	x	x	x					
+Høyde	N	REAL	i	x	x	x					
+Areal	N	REAL	i	x	x	x					
+Konsentrasjon	N	REAL	i,T	x	x	x	x				
+Ione_navn	A+N	9	B,T		x	x	x			x	x
STD_nummer	A+N	5	T	S		x					
Fortynn_fakt	N	INT	T	S		x					
Tid_koordinat	N	REAL	i	x							
Høyde_koordinat	N	REAL	i	x							
Metode_prog	BINÆR	VARIABEL	i						x		

* Metode-filnavn genereres ved tegnuttrekk av metodenavnet.

\+ Heretter vil THAK(I) være en forkortelse for :
Tid, Høyde, Areal, Konsentrasjon (og Ionenavn).

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
		H Iversen		
PCION	System data	DATO 5/86	3	SYSDATA/Z

Tegnforklaring :

n = Ø,1,2,,9. Filer kan ikke være større enn 64Kb. Det vil imidlertid være behov for langt større filer i enkelte tilfeller. For første fil er n = Ø, når denne filen er full opprettes en ny fil med samme navn, men med n = 1 osv.

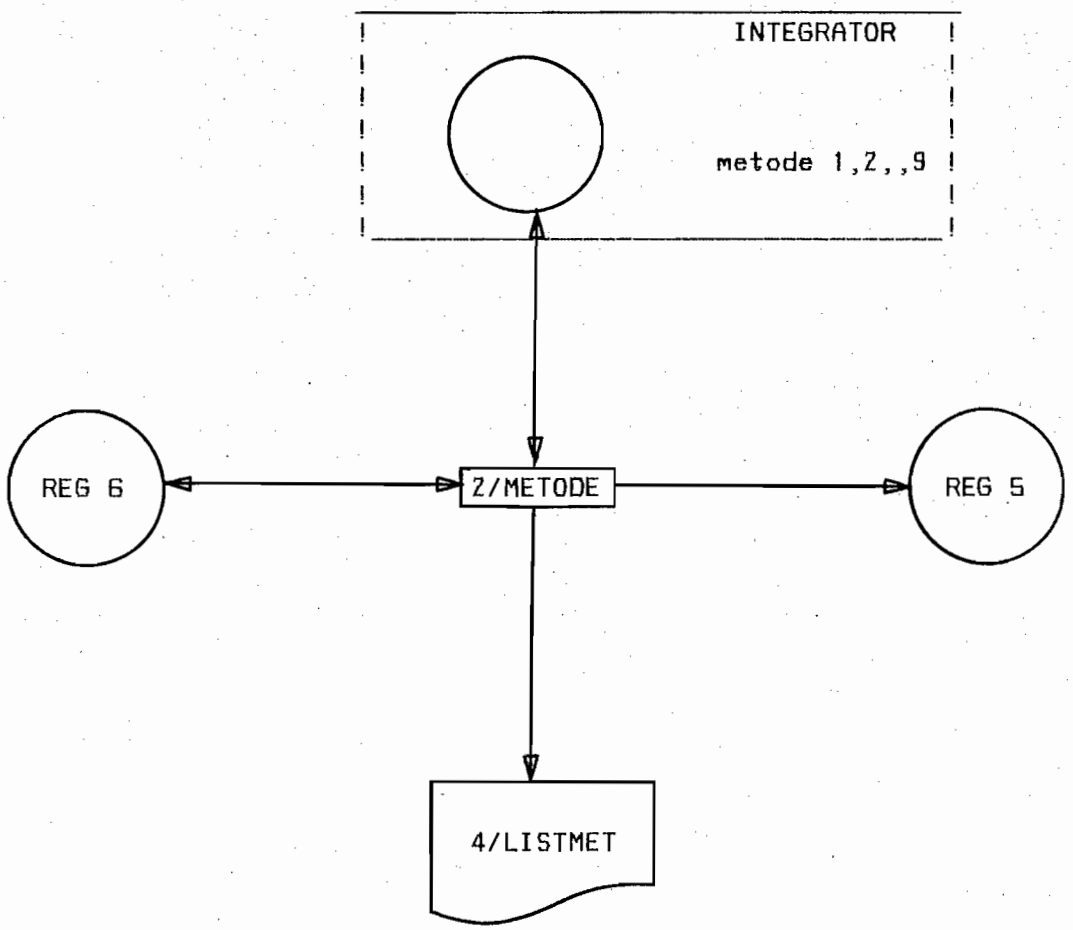
xxxxx = 13386 for oppdrag 133/86

bbbbbbbb = ekstrakt av metodenavn, se 3/REG5A

REGISTEROVERSIKT HP-15Ø

REGISTER	DRIVE	FIL-NAVN	INNHOLD
1	B	RAADATn.ION	Rådata hentet fra integrator
2	B	OPxxxxxn.RES	Kontrollerte analyseresultat. Det vil være en fil for hvert oppdrag. NB! ett oppdrag må ikke fordeles på flere disketter.
3	B	STANDARn.RES	Kontrollerte analyseresultat av standarder.
4	A	KAxxxxxn.BRK	Brukerfil som skal kopieres over til HP-3000.
5	A	METODE.TAB	Tabell over alle metoder samt plassering i integratoren.
6	A	bbbbbbbb.BIN	Inneholder en metode i binært format. Det vil være en fil for hver metode.
7	B	OPPDRAG.TAB	Tabell over alle oppdrag som har kontrollerte resultat på "denne" arbeidsdisketten. Brukes bl.a. 2/LIRES dersom oppdragsnummer ikke oppgis og til andre utlistinger.
8	A	IONENAVN.TAB	Tabell over ione_navn samt topptid og toleranse.

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
		H Iversen		
PCION	Informasjonsgraf,metode	DATO 5/86	3	INFMET

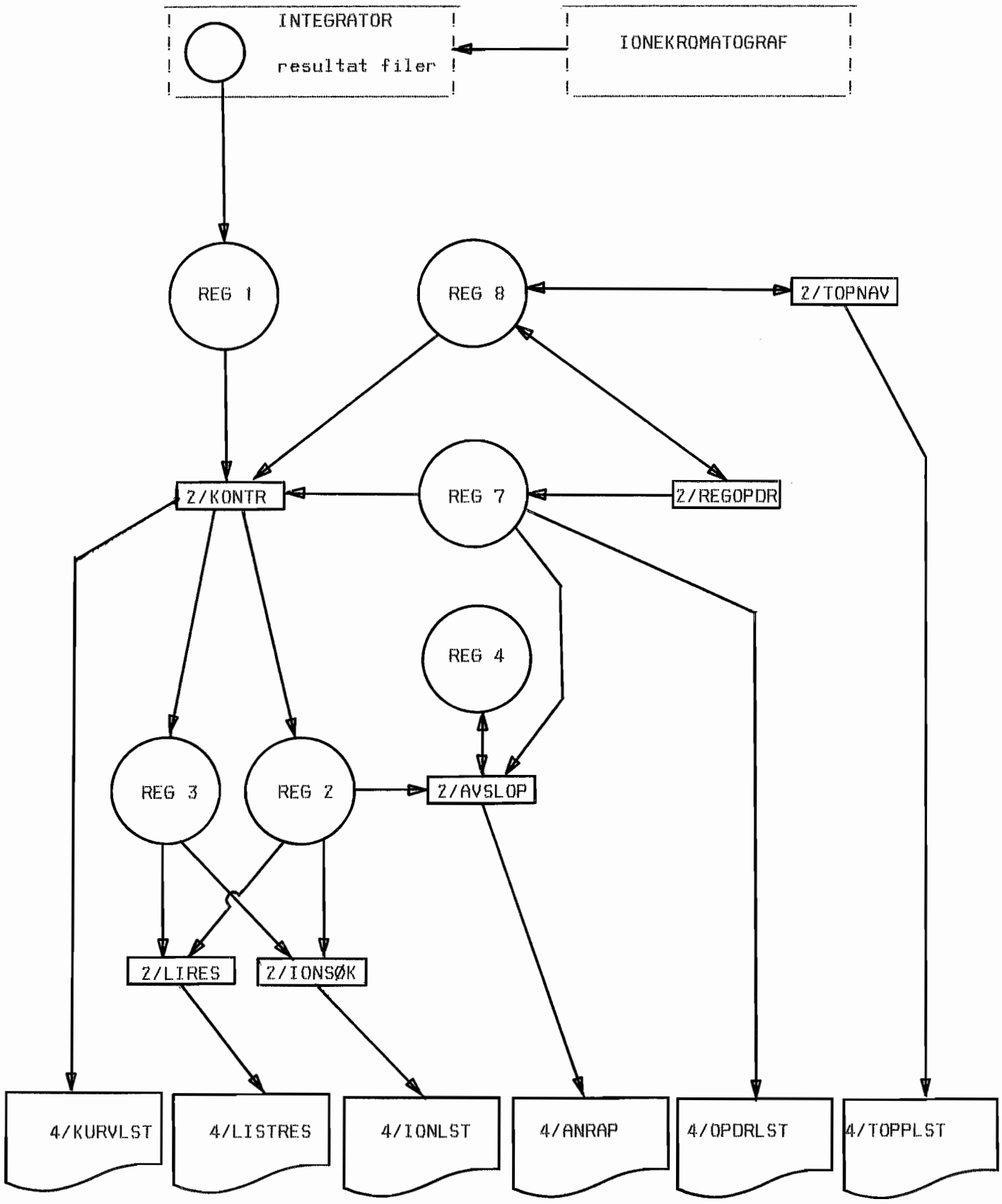


Z/METODE

Utveksler metoder mellom REG 6 og metodenr. 1,2,,9 i integratoren.
Vedlikeholder metode-index-tabellen REG 5.
Henter data fra REG 5 til 3/LISTMET.

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
PCION	Informasjonsgraf, data	H Iversen DATO 5/86	3	INFDATA/1



2/TOPNAV

Endrer,sletter og lagrer toppnavn med tilhørende topptid og toleranse på REG 8.

2/KONTR

Henter rådata fra REG 1, fletter det sammen med ionenavn fra REG 8 og kontrollerer med ionenavnene på REG 7.
Resultatet legges på REG 3 dersom det er en standard som er analysert, ellers legges det på REG 2.
Rådata som er kontrollert slettes fra REG 1.
Genererer utskrift 4/KURVLST.

2/REGOPDR

Henter og lagrer toppnavn med tilhørende topptid og toleranse på REG 8.
Registrerer oppdraget i oppdragstabellen REG 7.

2/AVSLOP

Henter oppdragsdata fra REG 7 og genererer brukerfilens hode, REG 4.
Henter analyseresultat fra REG 2, fjerner "duplikate" resultat og legger de aktuelle data sortert på økende prøvenummer på brukerfilen, REG 4.
Henter data fra REG 4 til 4/ANRAP, når REG 4 er ferdig.

2/LIRES og 2/IONSØK

Henter data fra REG 2 og REG 3 til hhv. 4/LISTRES og 4/IONLST

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
		H Iversen		
PCION	Format reg 1 drive B	DATO 5/86	3	REG1B

BESKRIVELSE

Registret består av en fil som inneholder rådata hentet fra integratoren.
Hver linje består av ett analyseresultat.
Når ett resultat er kontrollert vil det bli slettet fra denne fila.
Antall registrerte topper og antall koordinater er variable.

TEKSTFIL;

FORMAT

Vanlige analyser	Endring dersom standard
Oppdrags nummer	'S' + STD-nummer
Prøve nummer	Fortynnings faktor
Analyse dato	
Analyse kl	
Antall registrerte topper (n)	
THAK(1)	
THAK(2)	
.	
.	
THAK(n)	
Tid_koordinat(1)	
Høyde_koordinat(1)	
Tid_koordinat(2)	
Høyde_koordinat(2)	
.	
.	
Høyde_koordinat(siste)	
END OF LINE	

NB! Formatet på denne filen er ikke endelig. Det vil sannsynligvis være tilstrekkelig å lagre sample_intervall og Høyde_koordinater istedenfor tid- og høyde_koordinater.
Dessuten består ett kurvebilde av selve kurven pluss en basislinje som har en stor analytisk verdi, denne må også tas med.

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
		H Iversen		
PCION	Format reg 2 drive B	DATO 5/86	3	REG2B

BESKRIVELSE

Registret inneholder en fil for hvert oppdrag. Filen identifiseres ved oppdragsnummeret som er en del av filnavnet. Hver linje inneholder resultatet fra en analyse. Data vil aldri bli slettet fra registret. Registret vil med tiden ligge over flere disketter. Antall THAKI er variable.

TEKSTFIL

FORMAT

```
Prøve nummer
Analyse dato
Analyse kl
THAKI(1)
THAKI(2)
  .
  .
THAKI(siste)
END OF LINE
```

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
PCION	Format reg 3 drive B	H Iversen DATO 5/86	3	REG3B

BESKRIVELSE

Registret består av en fil. Hver linje inneholder resultat fra kjøring av en standard. Data vil aldri bli slettet fra registret. Registret vil med tiden ligge over flere disketter. Antall THAKI er variable.

TEKSTFIL

FORMAT

STD nummer
Fortynnings faktor
Analyse dato
Analyse kl
THAKI(1)
THAKI(2)
.
.
THAKI(siste)
END OF LINE

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
PCION	Format reg 4 drive A	H Iversen DATO 5/86	3	REG4A

BESKRIVELSE

Registret består av en fil for hvert oppdrag (brukerfil). Filen identifiseres ved oppdragsnummeret som er en del av filnavnet. Filen består av et hode på 5 linjer. Deretter følger resultater fra en analyse på hver linje. Når en brukerfil er kopiert over til HP-3000 må den slettes fra REG 4. Dette for å unngå unødig beslaglegging av diskplass. Antall ione_navn, bestem_gr og antall ionekonsentrasjoner er fast for en og samme brukerfil, men varierer fra brukerfil til brukerfil.

TEKSTFIL

FORMAT : HODET

```
Linje 1:   Oppdragsnummer          Prosjektnummer
Linje 2:   Oppdragsgiver
Linje 3:   'IONEKROMATOGRAF'
Linje 4:   Ione_navn(1)  Ione_navn(2)    ,,,,,,, Ione_navn(siste)
Linje 5:   Bestem_gr(1)  Bestem_gr(2)    ,,,,,,, Bestem_gr(siste)
```

FORMAT : DETALJLINER

```
Prøve nummer
Ione konsentrasjon(1)
Ione konsentrasjon(2)
  .
  .
Ione konsentrasjon(siste)
END OF LINE
```

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
		H Iversen		
PCION	Format reg 5 drive A	DATO 5/86	3	REG5A

BESKRIVELSE

Registret består av en fil. Filen er en tabell over alle metoder som er lagret på diskett samt filnavn og eventuell plassering i integratoren. Data for en metode ligger på en linje.

Filnavnet som en metode legges under beregnes slik:

1,2,3,5,7,11,17,30.BIN

der tallene foran typebetegnelsen angir bokstavnummer i metodenavnet.

Dersom det oppgis ett nytt metodenavn som vil få samme filnavn som et eksisterende, blir det gitt melding om det og metodenavnet må endres.

```
FILE OF RECORD
           INTEGER
           STRING(.50.)
           STRING(.8.)
        END
```

FORMAT

Metodenummer i integrator
Metodenavn
Filnavnet metode-koden ligger lagret under

Hvis en metode ikke er plassert i integrator, er metodenummer -1.

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
		H Iversen		
PCION	Format reg 6 drive A	DATO 5/86	3	REG6A

BESKRIVELSE

Registret består av en fil for hver metode.

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
		H Iversen		
PCION	Format reg 7 drive B	DATO 5/86	3	REG7A

BESKRIVELSE

Registret består av en fil og inneholder en tabell over registrerte oppdrag som har analyseresultater på disketten. Data blir aldri slettet fra dette registret. Ione navnene tilsvarer de ionene opppdragsgiver har forespurt. Antall ione-navn vil variere.

TEKSTFIL

FORMAT

```
Oppdragsnummer
Oppdragsgiver
Prosjekt nummer
Prøvetype
Ione_navn(1)
Ione_navn(2)
    .
    .
Ionenavn(siste)
```

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
		H Iversen		
PCION	Format reg 8 drive A	DATO 5/86	3	REG8A

BESKRIVELSE

Registret består av en fil. Filen inneholder en tabell over toppnavn som bl.a. Z/KONTR benytter seg av for å sette navn på ionene.

```
FILE OF RECORD
            REAL
            INTEGER
            STRING(.9.)
        END
```

FORMAT

Topptid
Topptid toleranse
Toppnavn (ionenavn)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
PCION	Listing av metoder	H Iversen DATO 5/86	4	LISTMET

BESKRIVELSE

Dette er en utskrift av metodene som er kjent for systemet.
For utlistinger av kun de metodene som ligger i integratoren vil bare de ni første detaljlinjene bli med på utskriften.

```
                                                                    Side        xx (1)
                          M E T O D E R                             Dato  xx.xx.xx (2)
#########################################################################################

     Metode nr.
    i integrator          Metode navn
    ------------------------------------------------------------------
          1      (3) xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
          2      (3) xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
          3          ledig
          4      (3) xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
          5                              .
          6                              .
          7                              .
          8                              .
          9                              .
                (4) xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                (4) xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
                                       .
                                       .
                                       .
```

(1) Sidetall, 1,2,,,

(2) Utskrifts dato.

(3) Variabelfeltet vil inneholde "ledig" dersom tilhørende metodenummer i integratoren ikke benyttes, ellers vil det inneholde metodenavnet.

(4) Resten av metodene som er lagret på diskett.

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
		H Iversen		
PCION	Oppdrags liste	DATO 5/86	4	OPDRLST

BESKRIVELSE

Dette er en liste over alle oppdrag som har analyseresultatene på denne arbeidsdisketten. Antall ionenavn vil variere og tar den plass den trenger. Oppdragene separeres med to blanke linjer.

```
!                                                                                !
!                                                          Side        xx (1)!
!                O P P D R A G S   L I S T E               Dato  xx.xx.xx (2)!
!################################################################################!
!                                                                                !
!   Oppdrag   xxxxxx (3)    xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx (5) !
!   Prosjekt  xxxxxxxxxx(4) xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx (6) !
!   xxxxxxxxx xxxxxxxxx xxxxxxxxx xxxxxxxxx xxxxxxxxx xxxxxxxxx xxxxxxxxx      !
!   (7) .     (7) .     (7) .     (7) .     (7) .     (7) .     (7) .          !
!       .         .         .         .         .         .         .          !
!                                                                              !
!                                                                              !
!   Oppdrag   xxxxxx (3)    xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx (5) !
!   Prosjekt  xxxxxxxxxx(4) xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx (6) !
!   xxxxxxxxx xxxxxxxxx xxxxxxxxx xxxxxxxxx xxxxxxxxx xxxxxxxxx xxxxxxxxx      !
!   (7) .     (7) .     (7) .     (7) .     (7) .     (7) .     (7) .          !
!       .         .         .         .         .         .         .          !
!                                                                              !
!______________________________________________________________________________!
```

(1) Sidetall, 1,2,,
(2) Utskrifts dato
(3) Oppdragsnummer
(4) Prosjekt nummer
(5) Oppdrags giver
(6) Prøvetype
(7) Ione navn

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
PCION	Utskrift av resultater	H Iversen DATO 5/86	4	LISTRES

BESKRIVELSE

Dette er en utlisting av analyseresultater se 2/LIRES. Resultatene grupperes slik at først kommer standardene deretter ett og ett oppdrag sortert på analysedato og klokkeslett.

```
                                                                    Side        xx (1)
            A N A L Y S E   R E S U L T A T E R                     Dato  xx.xx.xx (2)
################################################################################
                (3)                 (4)             (5)                (6)
      Oppdrag xxxxxx        Prøve xxxxx        Kl xx:xx       Dato  xx.xx.xx
      TOPP TID   ION                    HØYDE           AREAL      KONSENTR.
      ---------------------------------------------------------------------
      xxxxx      xxxxxxxxx      xxxxxxxxxxx     xxxxxxxxxxx      xxxxx bbb
      (7)        (8) .          (9) .           (10).            (11) .
        .            .              .               .                .
        .            .              .               .                .
```

Dersom analysen gjelder en standard vil (3) og (4) bli erstattet av:

```
                (12)                (13)
      STD nr. xxxxx   Fort.fakt. xxxxx
```

(1) Sidetall 1,2,,
(2) Utskrifts dato.
(3) Oppdragsnummer
(4) Prøvenummer
(5) Analyse tid
(6) Analyse dato
(7) Topptid
(8) Ionenavn
(9) Topphøyde
(10) Beregnet areal
(11) Konsentrasjon
(12) STD-nummer
(13) Fortynningsfaktor

VOLUM

Dersom det er plass, vil flere analyseresultat skrives ut på samme side. Analyseresultat som trenger flere enn 1 side (svært sjeldent) vil fortsette på neste side med ny heading (oppdrag, prøve etc.).

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
PCION	Utskrift av kurver	H Iversen DATO 5/86	4	KURVLST

BESKRIVELSE

Dette er en utskrift som kan fås ved kontrollering av analyseresultater. Utskriften inneholder det samme som 4/LISTRES pluss oppdragsgiver og en plotting av selve kurven inkl. kurveparametre.

```
                                                                    Utskrift nr xx (1)
                                                                    Dato  xx.xx.xx (2)

                  (3)       (14)
      Oppdrag : xxxxxx    xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
      Prøve   : xxxxx (4)       Dato  xx.xx.xx (6)       Kl  xx:xx (5)

       xxxxxxxxxx (16)                                    (17) xxxxxxxxxx
(15)  _____________________________________________________________________
xx.x  |
      |
xx.x  |
      |              (PLASS FOR KURVEN)
xx.x  |
      |
xx.x  |

      TOPP TID   ION                  HØYDE           AREAL      KONSENTR.
      ---------------------------------------------------------------------
      xxxxx      xxxxxxxxx      xxxxxxxxxxx      xxxxxxxxxxx      xxxxx bbb
      (7)        (8) .          (9) .            (10).            (11) .
        .            .              .                .                .
        .            .              .                .                .
```

Dersom analysen gjelder en standard vil (3) og (4) bli erstattet av:

```
      STD nr. : xxxxx (12)
      Fort.f. : xxxxx (13)
```

(1) Fortløpende nummerering av utskrifter tilhørende samme analyse.
(2)..(13) Se 4/LISTRES
(14) Oppdragsgiver
(15) Tids skala på kurven, går fra 'tid min' til 'tid max'.
(16) Høyde min
(17) Høyde max

VOLUM En analyse på hver side.

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
		H Iversen		
PCION	Analyse rapport	DATO 5/86	4	ANRAP/1

BESKRIVELSE

Analyse rapportens forside (Den vil muligens bli standardisert).

```
------------------------------------------------------------------------------
!                                                                            !
!     Norges geologiske undersøkelse,                         Side       1   !
!     Seksjon for kjemiske analyser                           Dato   xx.xx.xx (1)!
!                                                                            !
!                                                                            !
!                           A N A L Y S E   R A P P O R T                    !
!                                                                            !
!                             (2)                                            !
!      Prosjekt nummer  :  xxxxxxxxxx                                        !
!                             (3)                                            !
!                          xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx                !
!                             (4)                                            !
!      Oppdrags nummer  :  xxxxxx                                            !
!                             (5)                                            !
!      Oppdrags giver   :  xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx      !
!                             (6)                                            !
!      Prøve type       :  xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx      !
!                                                                            !
!      Instrument       :  IONEKROMATOGRAF                                   !
!                             (7)                                            !
!      Antall prøver    :  xxxxx                                             !
!                             (8)                                            !
!      Nummerert        :  xxxxx - xxxxx,  xxxxx - xxxxx,  xxxxx - xxxxx,    !
!                          xxxxx - xxxxx,        .               .           !
!                                .               .               .           !
!                                                                            !
!                                                                            !
!         (9)                                                                !
!         xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx       !
!         xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx       !
!         xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx       !
!         xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx       !
!         xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx       !
!         xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx       !
!                                                                            !
!                                                                            !
!                                                                            !
!                                                                            !
!                                                                            !
!   -----------------------------------   -----------------------------------!
!   xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx   xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx!
!   (10)                                  (11)                               !
!              Seksjons sjef                          Analytiker             !
!                                                                            !
!                                                                            !
!                                                                            !
!    Benevninger: ppt, ppb, ppm og % betyr for vannprøver og løsninger       !
!    hhv. ng/l, ug/l, mg/l og g/100ml.                                       !
!                                                                            !
!____________________________________________________________________________!
```

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
PCION	Analyse rapport	H Iversen DATO 5/86	4	ANRAP/2

(1) Utskriftsdato
(2) Prosjektnummer
(3) Eventuellt prosjektnavn
(4) Oppdrags nummer
(5) Oppdrags giver
(6) Prøvetype
(7) Antall prøver
(8) Prøvenummer fra - til, fra - til, etc.
(9) Eventuellt kommentarer
(10) Seksjonssjefens navn , sentrert
(11) Analytikers navn , sentrert

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
PCION	Analyse rapport	H Iversen DATO 5/86	4	ANRAP/3

BESKRIVELSE

Analyserapportens detaljlinjer. Det er plass til 7 forsjellige ioneslag på hvert ark. Dersom analyseresultatet inneholder flere enn 7 vil resultatet av en prøve være spredt over flere ark. Prøver på side 2A fortsetter på side 2B,2C osv. Logisk kan det betraktes som om hver side kan bestå av flere deler A,B,C osv. Er det mindre enn 8 forsjellige ionetyper på rapporten, vil ikke sidedelen skrives. Eks. side 2A => 2.

```
!                                                                        !
!                                                        Side  (1) xx x (2)!
!                                                        Dato  xx.xx.xx (3)!
!Prøve  (4)       (4)       (4)       (4)       (4)       (4)       (4)       !
!nr.    xxxxxxxxx xxxxxxxxx xxxxxxxxx xxxxxxxxx xxxxxxxxx xxxxxxxxx xxxxxxxxx !
!-------------------------------------------------------------------------- !
!xxxxx   xxxxbbb   xxxxbbb   <xxxxbbb  xxxxbbb   <xxxxbbb  xxxxbbb   xxxxbbb  !
! (5).   (6)  .    (6)  .    (7)  .    (6)  .    (7)  .    (6) .     (6) .    !
!    .       .         .         .         .         .         .         .    !
!    .       .         .         .         .         .         .         .    !
!                                                                        !
!                                                                        !
!                                                                        !
!                                                                        !
!________________________________________________________________________!
```

(1) Sidetall 2,3,,,,
(2) Side del A,B,,,, (kun dersom det er flere enn 7 ioner)
(3) Utskriftsdato
(4) Ione navn
(5) Prøve nummer
(6) Ione konsentrasjon
(7) Bestemmelsesgrense for dette ionet.

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
		H Iversen		
PCION	Utskrift av toppnavn	DATO 5/86	4	TOPPLST

BESKRIVELSE

Dette er en utlisting av alle toppnavn som er kjent for systemet. Disse navnene sammen med topptid og toleranse brukes av kontrollrutinen for å sette navn på konsentrasjonstopper registrert av integratoren.

```
R E G I S T R E R T E   T O P P N A V N          Dato  xx.xx.xx (1)

                                TOPP TID
          TOPP TID               TOL. %            ION
          ------------------------------------------
       (2) xx.xx              (3) xx          (4) xxxxxxxxx
```

(1) Utskrifts dato
(2) Topptid
(3) Topptid toleranse i prosent
(4) Ione navn (toppnavn)

VOLUM Mindre enn 50 detaljlinjer.

(*'*)

SYSTEM	NAVN	FORFATTER	KAPITTEL	KODE/SIDE
PCION	Ionelisting	H Iversen DATO 5/86	4	IONLST

BESKRIVELSE

Dette er en utlisting av de analyseresultatene som inneholder forespurte ioner, se 2/IONSØK. Første skjermbilde gjelder vanlige oppdrag, det andre gjelder standarder.

```
!                                                                        !
!                                                                        !
!      Ion       : xxxxxxxxx (1)                    Side        xx (3)!
!      Oppdrag   : xxxxxx (2)                       Dato  xx.xx.xx (4)!
!                                                                        !
!                                                                        !
!   Prøve        Dato          Kl          Høyde         Areal      Konsentr.  !
!   ------------------------------------------------------------------------  !
!   xxxxx      xx.xx.xx      xx:xx     xxxxxxxxxxx   xxxxxxxxxxx      xxxxx bbb  !
! (5) .        (6) .        (7).        (8)  .         (9) .          (10).      !
!     .            .           .           .             .              .       !
!     .            .           .           .             .              .       !
!                                                                        !
!________________________________________________________________________!
```

(1) Ionenavn
(2) Oppdrags nummer
(3) Sidetall
(4) Utskriftsdato
(5) Prøvenummer
(6) Analysedato
(7) Analyse kl
(8) Høyde
(9) Areal
(10) Ione konsentrasjon

```
!                                                                        !
!                                                                        !
!      Ion       : xxxxxxxxx (1)                    Side        xx (3)!
!      STD nr    : xxxxxx (11)                      Dato  xx.xx.xx (4)!
!                                                                        !
!                                                                        !
!   Fort.f.      Dato          Kl          Høyde         Areal      Konsentr.  !
!   ------------------------------------------------------------------------  !
!   xxxxx      xx.xx.xx      xx:xx     xxxxxxxxxxx   xxxxxxxxxxx      xxxxx bbb  !
! (12).        (6) .        (7).        (8)  .         (9) .          (10).      !
!     .            .           .           .             .              .       !
!     .            .           .           .             .              .       !
!                                                                        !
!________________________________________________________________________!
```

(11) STD nummer
(12) Fortynnings faktor

APPENDIX A

arbeids instruksjoner

I N N H O L D

(*'*)

LITT OM NOTASJON

"XXXXXXX"	= skriv det som står imellom hermetegn.
"XX" RETURN	= skriv det som står i mellom hermetegn og trykk på tasten RETURN
"XXX" "navn"	= skriv det som står imellom hermetegn, skriv en blank og tast "verdien" til navnet.

OPPSTART AV PCION

a) Når PC'en er slått av:

- Plasser systemdisketten i venstre drive og en formattert arbeidsdiskett i høyre drive.
- Slå på ionekromatografen, integratoren, begge skriverne og diskettstasjonen.
- Vent i ca. 5 sekunder.
- Slå på PC'en.

Dersom du får feilmeldingen " Load opsys failed...."

1. Trykk RETURN
2. Hold KONTROLL og SKIFT tastene nede samtidig som du trykker på OMSTART.

Dersom dette ikke nytter kontroller at du har plassert riktige disketter på riktig plass og prøv punktene 1 og 2 igjen.

b) Når det startes fra operativsystemet (opsys).

- Slå på ionekromatografen, skriverne osv.
- Plasser systemdisketten i venstre drive og en formattert arbeidsdiskett i høyre drive.
- Tast:

"A:" RETURN
"PCION" RETURN

AVSLUTTING AV PCION

PCION avsluttes fra hovedmenyen. Det er nå operativsystemet som presenterer seg med X>, der X = (A,B,,,,) og angir default diskdrive. A = venstre diskdrive, B = høyre. (DEFAULT = det/den som velges dersom ingen spesifiseres.)

(*'*)

OPERATIVSYSTEM KOMMANDOER

Fra opsys kan det utføres en rekke kommandoer. Her vises noen få eksempler. Se HP 150/MS-DOS User's Guide.

- Endring av default diskdrive: "X:" RETURN

  X angir diskdrive.

- Utlisting av filer: "DIR>LST" RETURN

  De filene som er på disketten i default diskdrive listes ut på skriver.

- Sletting av filer: "ERASE" "filnavn" RETURN

  Filen med navn "filnavn" vil bli slettet fra disketten i default diskdrive.

KLARGJØRING AV SKRIVEREN

PCION klargjører skriveren hver gang det startes. Dersom skriveren slås av og på igjen må den klargjøres på nytt. Dette gjøres ved å trykke på <LF> tasten på skriveren helt til skrivehodet er plassert på toppen av ett ark, dernest gå ut av PCION og start PCION igjen.

KLARGJØRING AV NY ARBEIDSDISKETT

Plasser orginaldisketten HP 150 SYS_MASTER i venstre drive.
Plasser den nye arbeidsdisketten i høyre drive.

Tast
"A:" RETURN
"FORMAT" RETURN

Ta på skjermen på feltet merket med "B",
Dersom du vil ha navn på disketten, skriv inn navnet og avslutt med RETURN. Hvis ikke, ta på feltet "ingen etikett".
Ta på feltet "start format".
Vent til formateringen er ferdig.
Ta på feltet "avslutt format"

Du er nå tilbake i opsys. Ta ut SYS_MASTER disketten og sett tilbake systemdisketten.

Disketten er nå ferdig og kan brukes til backup og arb.-diskett. Se app. B for navnsetting av diskettene. Husk å skrive navnet på disketten.

(*'*)

KLARGJØRING AV NY SYSTEMDISKETT

Plasser orginaldisketten HP 150 SYS_MASTER i venstre drive.
Plasser den nye systemdisketten i høyre drive.

Tast

"A:"	RETURN
"FORMAT"	RETURN

Ta på feltet på skjermen merket med "B".
Skriv "PCION_SYS" eller "PCIONH_BAC" alt etter hva disketten skal brukes til, avslutt med RETURN.
Ta på feltet "kopier opsys" slik at en stjerne vises i feltet
Ta på feltet "start format"
Vent til formatteringen er ferdig.
Ta på feltet "avslutt format".

Du er nå tilbake i opsys. Ta ut SYS_MASTER disketten og sett tilbake systemdisketten.

Tast

"B:"	RETURN
"REN CONFIG.SYS CONFIG.SAV"	RETURN
"COPY A:AUTOEXEC.BAT B:"	RETURN
"COPY A:PCION.COM B:"	RETURN

Disketten er nå ferdig til bruk som system-backup-diskett.

MANUELL BACKUP AV ARBEIDSDISKETTER

Plasser arbeids-backup-disketten i venstre drive (må være klargjort).
Plasser arbeids-disketten i høyre drive.
Tast:

"COPY B:*.* A:" RETURN

MANUELL BACKUP AV SYSTEMDISKETTEN

Plasser system-backup-disketten i høyre drive (må være klargjort).
Plasser systemdisketten i venstre drive.
Tast:

"COPY A:KA*.BRK B:"	RETURN
"COPY A:METODE.TAB B:"	RETURN
"COPY A:*.BIN B:"	RETURN
"COPY A:IONENAVN.TAB B:"	RETURN

(*'*)

KOPIERING AV SYSTEMDISKETTEN

1. Klargjør en ny system-diskett.
2. Bruk den som system-backup-diskett en gang.
   Disketten er nå klar til bruk som systemdiskett.
3. Kast den gamle slitte systemdisketten (for å unngå
   inkonsistente data ved bruk av feil systemdiskett)

JUSTERING AV KLOKKEN

Skriv "TIME" RETURN
Klokken skrives ut på skjermen.
Dersom det skal endres, skriv nytt klokkeslett og trykk RETURN,
hvis ikke, trykk RETURN.

JUSTERING AV DATOEN

Skriv "DATE" RETURN
Datoen skrives ut på skjermen.
Dersom datoen skal endres, skriv ny dato og trykk RETURN,
hvis ikke, trykk RETURN.

APPENDIX B

diskett bruk, kartotek

HÅNDTERING AV DISKETTER

Disketter er magnetiske media og trenger spesiell behandling.
De må ikke utsettes for direkte sollys eller sterk magnetisme.
Temperaturendring må ikke overstige 20 grader/time.
De må oppevares på et rent og tørt sted.
Gjør aldri forsøk på å rense eller vaske disketter.
Ta ardri på diskettens magnetbelegg, data kan ødelegges eller gå tapt.

SLITTE DISKETTER

Når en diskett har gått rundt i en diskettenhet ca. 1,5 millioner ganger begynner lyset foran på enheten å blinke, av og på, mens magnet-hodene lager en klikkende lyd. Det må nå tas kopi av disketten. Den gamle bør kastes. Dersom du fortsetter å bruke disketten vil du få lov til å bruke den til den ha gått totalt ca. 2 milloner ganger rundt. Etter dette får du ikke lov å skrive på disketten.

SYSTEMDISKETT

Diskettene gis navn slik:

Systemdiskett	: 'PCION_SYS'
BACKUP til systemdiskett	: 'PCION_BAC'

Etter hver gang PCION har vært i drift bør det tas BACUP av PCION_SYS til PCION_BAC. Diskettene vil da ha identiskt innhold. Når en av disse diskettene viser tegn til slitasje må det kopieres en ny systemdiskett. Den gamle bør kastes. En bør alltid ha en klargjort systemdiskett tilgjengelig.

DISKETTER TIL BRUKERFILER

Når oppdrag avsluttes må systemdisketten byttes ut med en diskett der brukerfilen kan legges. Denne disketten bør være tom slik at det ikke oppstår uklarheter om hva som er brukerfilen. Brukerfilen kan, fysisk sett, være delt over flere filer, se krav spesifikasjon kap. 3 SYSDATA/2. Disketten leveres for kopiering over til HP-3000. Disketten skal leveres tilbake. Når disketten skal klargjøres, brukes samme framgangsmåte som for arbeidsdisketter. Her trengs ikke navn på disketten, ta på feltet på skjermen merket "ingen etikett". Dersom disketten klargjøres hver gang før den skal brukes, Da er en sikker på at kun brukerfilen avleveres.

(*'*)

ARBEIDSDISKETT

Når en ny arbeidsdiskett skal klargjøres, gis den navn etter hva den skal brukes til, og når den ble tatt i bruk, slik :

Vanlige oppdrag	: 'OPDååmmddx'
Standarder	: 'STDååmmddx'
BACUP av OPD og STD	: 'BACååmmddx'

Der "ååmmdd" er datoen disketten ble klargjort, og "x" er en bokstav (A,B,,,,). Denne bokstaven er nødvendig for å kunne skille mellom flere disketter som klargjøres samme dag.

Samtidig som det klargjøres nye arbeidsdisketter til standarder eller oppdrag, må det for hver av dem, klargjøres en bacupdiskett med eksakt samme "ååmmddx". Det vil nå ikke være tvil om hvilke disketter som har hvilke bacupdisketter etc.

Årsaken til en så omstendig nummerering er at det til enhver til vil være flere (ca 2-6) arbeidsdisketter i bruk. Alle disse må også ha en backupdiskett. Det er behov for flere arbeidsdisketter samtidig fordi minst en brukes til standarder og, dersom det er store oppdrag må hvert oppdrag ha hver sin diskett.

Når en arbeidsdiskett er full, og alle oppdrag på den er avsluttet kan disketten tas vare på i et kartotek. Den tilhørende backupdisketten er det ikke lenger behov for, den kan klargjøres på nytt for bruk til hva som helst. Diskettene i kartoteket kan når som helst brukes til:

Utlisting av resultater
Utlisting av bestemte ioner
Utlisting av alle oppdrag på disketten
Generering av analyserapport og brukerfil

Når det ikke lenger er ønskelig å bevare disse opplysningene, kan disket-tene klargjøres på nytt (Husk å endre navnet på etiketten).

APPENDIX C

interrupt styrt datalogging

INTERRUPT STYRT DATA LOGGING

Slik det fremgår av kravspesifikasjonen er det ikke mulig å la PC'en hente analysedata fra integratoren samtidig som noen av de andre rutinene kjøres.

Systemet vil bli mer fleksibelt dersom det lar seg gjøre å gå inn i rutinen "Start av ionekromatograf", starte den, og kunne fortsette med andre rutiner uavhengig av om ionekromatografen er i drift.

Dette medfører følgende tillegg til kravspesifikasjon:

- Dersom det gjøres forsøk på å starte ionekromatografen når den er i gang, gis det melding.
- Det må være mulig å gi beskjed til interrupt-rutina at når den nåværende analysen er ferdig, så skal ionekromatografen stoppes.

Interrupt styrt data logging kan muligens gjøres på tre måter.

1. Integratoren sender interrupt til HP-150 hver gang en ny kurvehøyde er ferdig. Interruptrutina leser kurvehøyden og legger den i ett array (helst en lenket liste fordi antall koordinater kan variere enormt, og bli ganske mange).Når en analyse er ferdig (les integrator status), leses resultatene Topptid, Høyde Areal og konsentrasjon fra filer i integratoren. Dette sammen med kurvekoordinatene legges så på fil.

2. Disse kurvehøydene som integratoren sender (leses fra integrator) skal leses med jevne mellomrom. Hvor ofte integratoren har en ny høyde klar, avhenger av noen parametre som inngår i metoden som integratoren benytter. Dersom det er mulig, enten å bestemme seg for ett fast tids-intervall (lite dynamisk), eller at brukeren taster inn tids-intervallet når ionekromatografen strartes, kan dette tids-intervallet brukes for å få RT-klokka til å sende interrupt-signal når en ny kurvehøyde skal leses fra integratoren.

3. Dersom det er mulig å lese HELE kurven fra integratoren når den er ferdig analysert vil det være den enkleste løsning. Kurven ligger lagret etter at analyseringen er ferdig, men om den er komplett eller krympet må undersøkes når interface-kort er montert. Dersom kurven er krympet, vil skalering og translering av kurven på skjermen, gi galt resultat. Denne metoden må interrupt-styres av integratoren (dersom mulig).

(*'*)

NB! Det kan oppstå problemer når interrupt-rutina avbryter en disk-operasjon og bruker disken selv. Dette bør undersøkes nærmere. Se forøvrig HP 150 Technical Reference Manual og LDC Milton Roy CI-10B Integrator manualen.

EKSEMPEL på interrupt-rutine i Turbo Pascal:

Her er ett eksempel på en interrupt-rutine (inter_test) som aktiviseres når det registreres interruptsignal på pinne IR2 på 8259A Inter. Contr.

```
PROGRAM PCION(input,output);

   .
   .
   .
   .

PROCEDURE Inter_test;

BEGIN
   InLine ($50/$53/$51/$52/$57/$56/$06/$FB);  (* Push register etc. *)
      .
      .
      programkode
      .
      .
   InLine ($07/$5E/$5F/$5A/$59/$5B/$58/$CF);  ( Pop register og IRET *)
END;


   .
   .
   .
   .

BEGIN                        (***    P C I O N    ***)
   MemW(.$0000:$0108.) := Ofs(Inter_Test);   (* Offset intr. vektor *)
   MemW(.$0000:$010A.) := Cseg;              (* Base adr. ----"---- *)
   .
   .
   .
END.
```

Interrupt Vectors

Interrupts within the HP 150 are triggered by hardware attached to the 8088 microprocessor, software interrupt instructions performed by MS-DOS, software interrupt instructions from application (user) programs, or under some special circumstances, by the 8088 itself.

Every interrupt is assigned a type code that identifies it to the 8088. Interrupts are identified as "INT n" where n (the type code) is a number between 0 and 255 inclusive. The type code is used by the 8088 to calculate a location in the memory based interrupt vector table containing the four byte address of the interrupt routine. The interrupt vector for INT 0 is at address 00000H, the vector for INT 1 is at address 00004H, and so on. The interrupt vector table can contain up to 256 vectors, one for each interrupt type.

Each entry in the table is a doubleword pointer containing the address of the procedure that is to service interrupts of that type. The higher addressed word of the pointer contains the base address of the segment containing the procedure. The lower addressed word contains the procedure's offset from the beginning of the segment. Since each entry is four bytes long, the 8088 can calculate the location of the correct entry for a given interrupt type by simply multiplying the type by four. For more information on how the 8088 processor treats interrupts and the conditions which cause Intel-reserved interrupts, see the IAPX86/88, 186/188 User's Manual, Programmer's Reference, Intel Corporation, May 1983.

The 256 interrupt types are pre-allocated for the HP 150. The following table describes that allocation.

Starting Address	Interrupt Vector Description	Usage
003FCH	Type 255 Pointer - (**** not used ****)	Available
00200H	Type 128 Pointer - (**** not used ****)	
001FCH	Type 127 Pointer - (RESERVED)	Reserved for HP
00120H	Type 72 Pointer - (RESERVED)	
0011CH	Type 71 Pointer - HP 150 Hardware (IR7)	Reserved for and used by HP hardware.
00118H	Type 70 Pointer - HP 150 Hardware (IR6)	
00114H	Type 69 Pointer - HP 150 Hardware (IR5)	
00110H	Type 68 Pointer - HP 150 Hardware (IR4)	
0010CH	Type 67 Pointer - HP 150 Hardware (IR3)	
00108H	Type 66 Pointer - HP 150 Hardware (IR2)	
00104H	Type 65 Pointer - HP 150 Hardware (IR1)	
00100H	Type 64 Pointer - HP 150 Hardware (IR0)	
000FCH	Type 63 Pointer - (**** not used ****)	Available
000A8H	Type 42 Pointer - (**** not used ****)	
000A4H	Type 41 Pointer - MSDOS Interrupt 29	Reserved for MicroSoft
000A0H	Type 40 Pointer - MSDOS Interrupt 28	
0009CH	Type 39 Pointer - MSDOS Interrupt 27	
00098H	Type 38 Pointer - MSDOS Interrupt 26	
00094H	Type 37 Pointer - MSDOS Interrupt 25	
00090H	Type 36 Pointer - MSDOS Interrupt 24	
0008CH	Type 35 Pointer - MSDOS Interrupt 23	
00088H	Type 34 Pointer - MSDOS Interrupt 22	
00084H	Type 33 Pointer - MSDOS Interrupt 21	
00080H	Type 32 Pointer - MSDOS Interrupt 20	
0007CH	Type 31 Pointer - (**** not used ****)	Available
00014H	Type 5 Pointer . (**** not used ****)	Reserved for Intel
00010H	Type 4 Pointer - Intel Dedicated	
0000CH	Type 3 Pointer - Intel Dedicated	
00008H	Type 2 Pointer - Intel Dedicated	
00004H	Type 1 Pointer - Intel Dedicated	
00000H	Type 0 Pointer - Intel Dedicated	

MS-DOS INTERRUPTS

Interrupt 27H: Terminate But Stay Resident
Interrupt 26H: Absolute Disc Write
Interrupt 25H: Absolute Disc Read
Interrupt 24H: Fatal Error Abort Address
Interrupt 23H: CONTROL-C Exit Address
Interrupt 22H: Terminate Address
Interrupt 21H: Function Request
Interrupt 20H: Program Terminate

HP 150 HARDWARE INTERRUPTS

IR7: Real Time Clock (MM58167A)
IR6: Not Used (Tied High)
IR5: HPIB Controller (9914)
IR4: *Integral Printer,Accessory Slot NOCINT (Low Priority Open Collector)
IR3: Keyboard and Touchscreen
IR2: Not Used (Tied High)
IR1: *MPSC (Datacomm Controller),Accy. Slot NDCOCINT (High Priority O/C)
IR0: Video Controller (9007)

* These interrupts may be initiated from an accessory board by using the NOCINT and NDOCOINT open collector lines. Either may be used, IR1 has higher priority than IR4 (IR1 will be serviced prior to IR2-IR7). See Accessory Board Subsystem in the Hardware Subsystems section of this manual.

Firmware Variables

The firmware uses the RAM space between 00400H and 067FFH for jump vectors and other working data storage. See Section 6, "System Firmware" for more information.

BIOS and BDOS

The Basic Input/Output System (BIOS) and the Basic Disc Operating System (BDOS) are loaded from disc during initialization. BDOS is loaded following BIOS loading and initialization, and overlays the initialization portion of BIOS. The origin address of MS-DOS is BIOS revision dependent.

Disc Buffer Cache

The Disc Buffer Cache is an area used for buffering of disc data for all drives. It includes portions or all of the File Allocation Tables (depending upon cache size), the root directory, and data for non-sector oriented reads and writes. The size of the Disc Buffer Cache may be

APPENDIX D

Datablad for MM58167 RT-Clock

National Semiconductor

October 1981

MM58167A Microprocessor Real Time Clock

General Description

The MM58167A is a low threshold metal gate CMOS circuit that functions as a real time clock in bus oriented microprocessor systems. The device includes an addressable real time counter, 56 bits of RAM, and two interrupt outputs. A $\overline{\text{POWER DOWN}}$ input allows the chip to be disabled from the rest of the system for standby low power operation. The time base is a 32,768 Hz crystal oscillator.

Features

- Microprocessor compatible (8-bit data bus)
- 1/10,000 of a second through month counters
- 56 bits of RAM with comparator to compare the real time counter to the RAM data
- 2 INTERRUPT OUTPUTS with 8 possible interrupt signals
- $\overline{\text{POWER DOWN}}$ input that disables all inputs and outputs except for one of the interrupts
- Status bit to indicate rollover during a read
- 32,768 Hz crystal oscillator
- Four-year calendar (no leap year)
- 24-hour clock

Functional Description

Real Time Counter

The real time counter is divided into 4-bit digits with 2 digits being accessed during any read or write cycle. Each digit represents a BCD number and is defined in Table I. Any unused bits are held at a logical zero during a read and ignored during a write. An unused bit is any bit not necessary to provide a full BCD number. For example tens of hours cannot legally exceed the number 2, thus only 2 bits are necessary to define the tens of hours. The other 2 bits in the tens of hours digit are unused. The unused bits are designated in Table I as dashes.

The addressable portion of the counter is from 1/10,000 of a second to months. The counter itself is a ripple counter. The ripple delay is less than 60 μs above 4.0V and 300 μs at 2.0V.

RAM

56 bits of RAM are contained on-chip. These can be used for any necessary power down storage or as an alarm latch for comparison to the real time counter. The data in the RAM can be compared to the real time counter on a digit basis. The only digits that are not compared are the unit ten thousandths of seconds and tens of days of the week (these are unused in the real time counter). If the two most significant bits of any RAM digit are ones, then this RAM location will always compare.

The RAM is formatted the same as the real time counter, 4 bits per digit, 14 digits, however there are no unused bits. The unused bits in the real time counter will compare only to zeros in the RAM.

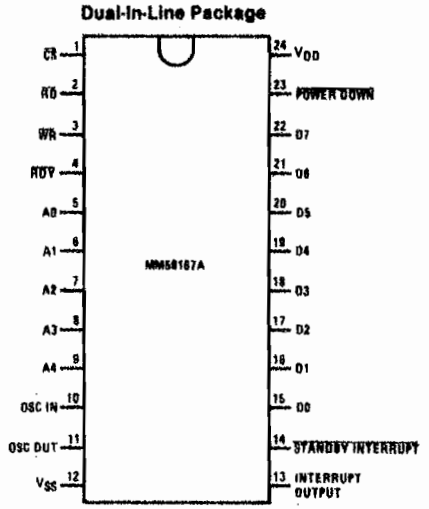
Interrupts and Comparator

There are two interrupt outputs. The first and most flexible is the INTERRUPT OUTPUT (a true high signal). This output can be programmed to provide 8 different output signals. They are: 10 Hz, 1 Hz, once per minute, once per hour, once a day, once a week, once a month, and when a RAM/real time counter comparison occurs. To enable the output a one is written into the interrupt control register at the bit location corresponding to the desired output frequency *(Figure 1)*. Once one or more bits have been set in the interrupt control register, the corresponding counter's rollover to its reset state will clock the interrupt status register and cause the interrupt output to go high. To reset the interrupt and to identify which frequency caused the interrupt, the interrupt status register is read. Reading this register places the contents of the status register on the data bus. The interrupting frequency will be identified by a one in the respective bit position. Removing the read will reset the interrupt.

The second interrupt is the $\overline{\text{STANDBY INTERRUPT}}$ (open drain output, active low). This interrupt occurs when enabled and when a RAM/real time counter comparison occurs. The $\overline{\text{STANDBY INTERRUPT}}$ is enabled by writing a one on the D0 line at address 16_H or disabled by writing a zero on the D0 line. This interrupt is not triggered by the edge of the compare signal, but rather by the level. Thus if the compare is enabled when the $\overline{\text{STANDBY INTERRUPT}}$ is enabled, the interrupt will turn on immediately.

Connection Diagram

Dual-In-Line Package



TOP VIEW

IM-B25M101/Printed in U.S.A

Absolute Maximum Ratings

Voltage at All Pins	V_{SS} − 0.3V to V_{DD} + 0.3V
Operating Temperature	− 40°C to 85°C
Storage Temperature	− 65°C to 150°C
V_{DD}-V_{SS}	6.0V
Lead Temperature (Soldering, 10 seconds)	300°C

Electrical Characteristics $V_{SS} = 0V$, $-40°C \le T_A \le 85°C$

Parameter	Conditions	Min	Max	Units
Supply Voltage				
V_{DD}	Outputs Enabled	4.0	5.5	V
V_{DD}	$\overline{\text{POWER DOWN}}$ Mode	2.0	5.5	V
Supply Current				
I_{DD}, Static	Outputs TRI-STATE* $f_{IN} = DC$, $V_{DD} = 5.5V$		10	μA
I_{DD}, Dynamic	Outputs TRI-STATE $f_{IN} = 32$ kHz, $V_{DD} = 5.5V$ $V_{IH} \ge V_{DD} - 0.3V$ $V_{IL} \le V_{SS} + 0.3V$		20	μA
I_{DD}, Dynamic	Outputs TRI-STATE $f_{IN} = 32$ kHz, $V_{DD} = 5.5V$ $V_{IH} = 2.0V$, $V_{IL} = 0.8V$		12	mA
Input Voltage				
Logical Low		0 0	0.8	V
Logical High		2.0	V_{DD}	V
Input Leakage Current	$V_{SS} \le V_{IN} \le V_{DD}$	− 1	1	μA
Output Impedance	I/O and INTERRUPT OUT			
Logical Low	$V_{DD} = 4.5V$, $I_{OL} = 1.6$ mA		0.4	V
Logical High	$V_{DD} = 4.5V$, $I_{OH} = -400$ μA	2.4		V
	$I_{OH} = -10$ μA	0.8 V_{DD}		V
TRI-STATE	$V_{SS} \le V_{OUT} \le V_{DD}$	− 1	1	V
Output Impedance	$\overline{\text{RDY}}$ and $\overline{\text{STANDBY}}$ $\overline{\text{INTERRUPT}}$			
Logical Low, Sink	$V_{DD} = 4.5V$, $I_{OL} = 1.6$ mA		0.4	V
Logical High, Leakage	$V_{OUT} \le V_{DD}$		10	μA

Functional Description (Continued)

TABLE I. REAL TIME COUNTER FORMAT

Counter Addressed		Units D0	D1	D2	D3	Max BCD Code	Tens D4	D5	D6	D7	Max BCD Code
1/10,000 of Seconds	(00_H)	-	-	-	-	0	D4	D5	D6	D7	9
Hundredths and Tenths Sec	(01_H)	D0	D1	D2	D3	9	D4	D5	D6	D7	9
Seconds	(02_H)	D0	D1	D2	D3	9	D4	D5	D6	-	5
Minutes	(03_H)	D0	D1	D2	D3	9	D4	D5	D6	-	5
Hours	(04_H)	D0	D1	D2	D3	9	D4	D5	-	-	2
Day of the Week	(05_H)	D0	D1	D2	-	7	-	-	-	-	0
Day of the Month	(06_H)	D0	D1	D2	D3	9	D4	D5	-	-	3
Month	(07_H)	D0	D1	D2	D3	9	D4	-	-	-	1

( ) indicates unused bits

TRI STATE® is a registered trademark of National Semiconductor Corp.

Functional Description (Continued)

TABLE II. ADDRESS CODES AND FUNCTIONS

A4	A3	A2	A1	A0	Function
0	0	0	0	0	Counter—Ten Thousandths of Seconds
0	0	0	0	1	Counter—Hundredths and Tenths of Seconds
0	0	0	1	0	Counter—Seconds
0	0	0	1	1	Counter—Minutes
0	0	1	0	0	Counter—Hours
0	0	1	0	1	Counter—Day of Week
0	0	1	1	0	Counter—Day of Month
0	0	1	1	1	Counter—Month
0	1	0	0	0	RAM—Ten Thousandths of Seconds
0	1	0	0	1	RAM—Hundredths and Tenths of Seconds
0	1	0	1	0	RAM—Seconds
0	1	0	1	1	RAM—Minutes
0	1	1	0	0	RAM—Hours
0	1	1	0	1	RAM—Day of Week
0	1	1	1	0	RAM—Day of Month
0	1	1	1	1	RAM—Months
1	0	0	0	0	Interrupt Status Register
1	0	0	0	1	Interrupt Control Register
1	0	0	1	0	Counter Reset } See Table III Below
1	0	0	1	1	RAM Reset } See Table III Below
1	0	1	0	0	Status Bit
1	0	1	0	1	GO Command
1	0	1	1	0	$\overline{\text{STANDBY INTERRUPT}}$
1	1	1	1	1	Test Mode

All others unused

TABLE III. COUNTER AND RAM RESET FORMAT

D0	D1	D2	D3	D4	D5	D6	D7	Counter or RAM Reset
1	0	0	0	0	0	0	0	Ten Thousandths of Seconds
0	1	0	0	0	0	0	0	Hundredths and Tenths of Seconds
0	0	1	0	0	0	0	0	Seconds
0	0	0	1	0	0	0	0	Minutes
0	0	0	0	1	0	0	0	Hours
0	0	0	0	0	1	0	0	Days of the Week
0	0	0	0	0	0	1	0	Days of the Month
0	0	0	0	0	0	0	1	Months

For counters reset address = 12_H

For RAM reset address = 13_H

Functional Description (Continued)

The comparator is a cascaded exclusive NOR. Its output is latched 61 μs after the rising edge of the 1 kHz clock signal (input to the ten thousandths of seconds counter). This allows the counter to ripple through before looking at the comparator. For operation at less than 4.0V, the thousandths of seconds counter should not be included in a compare because of the possibility of having a ripple delay greater than 61 μs. (For output timing see Interrupt Timing.)

Power Down Mode

The $\overline{\text{POWER DOWN}}$ input is essentially a second chip select. It disables all inputs and outputs except for the $\overline{\text{STANDBY INTERRUPT}}$. When this input is at a logical zero, the device will not respond to any external signals. It will, however, maintain timekeeping and turn on the $\overline{\text{STANDBY INTERRUPT}}$ if programmed to do so. (The programming must be done before the $\overline{\text{POWER DOWN}}$ input goes to a logical zero.) When switching V_{DD} to the standby or power down mode, the $\overline{\text{POWER DOWN}}$ input should go to a logical zero at least t μs before V_{DD} is switched. When switching V_{DD} all other inputs must remain between V_{SS} - 0.3V and V_{DD} + 0.3V. When restoring V_{DD} to the normal operating mode, it is necessary to insure that all other inputs are at valid levels before switching the $\overline{\text{POWER DOWN}}$ input back to a logical one. These precautions are necessary to insure that no data is lost or altered when changing to or from the power down mode.

Counter and RAM Resets; GO Command

The counters and RAM can be reset by writing the proper data at address 12_H or address 13_H respectively. The resets are divided up the same way as the counters and RAM are divided up when accessing them. The data written into the part will determine which set of 2 digits (1 digit for ten thousandths of seconds and days of the week) will be reset. The address will determine whether the 2 digits are in the RAM or in the real time counter. Each logical one on the data bus will cause 2 digits to be reset. Resetting the most significant used bit of any counter will clock the following counter.

A write pulse at address 15_H will reset the thousandths, hundredths, tenths, units, and tens of seconds counters. This GO command is used for precise starting of the clock. The data on the data bus is ignored during the write. If the seconds counter is at a value greater than 40 when the GO is issued, the minute counter will increment; otherwise the minute counter is unaffected. This command is not necessary to start the clock, but merely a convenient way to start precisely at a given minute. (See Table III for reset format.)

Status Bit

The status bit is provided to inform the user that the clock is in the process of rolling over when a counter is read. The 1 kHz clock into the thousandths of seconds counter has a pulse width of 61 μs. If a read of the real time counter (any digits) is done during this 61 μs period the status bit will be set. This tells the user that the clock is rippling through the real time counter. Because the clock is rippling, invalid data may be read from the counter. If the status bit is set following a counter read, the counter should be reread.

The status bit appears on D0 when address 14_H is read. All the other data lines will be zero. The bit is set when a logical one appears. This bit should be read every time a counter read or series of counter reads are done. The trailing edge of the read at address 14_H will reset the status bit.

Oscillator

The oscillator used is the standard Pierce oscillator. Externally only 2 capacitors and the crystal are required. For micropower crystals a resistor in series with the oscillator output may be necessary to insure the crystal is not overdriven. This resistor should be approximately 200 kΩ. The capacitor values should be typically 20 pF–25 pF. The crystal frequency is 32,768 Hz.

The oscillator input can be externally driven, if desired. In this case the output should be left floating and the input levels should be within 0.3V of the supplies.

A ground line or ground plane between pins 9 and 10 may be necessary to prevent interference of the oscillator by the A4 address.

Control Lines

The $\overline{\text{READ}}$, $\overline{\text{WRITE}}$, and $\overline{\text{CHIP SELECT}}$ signals are active low inputs. The $\overline{\text{READY}}$ signal is an open drain output. At the start of each read or write cycle the $\overline{\text{READY}}$ line (open drain) will pull low and will remain low until valid data from a chip read appears on the bus or data on the bus is latched in during a write. $\overline{\text{READ}}$ and $\overline{\text{WRITE}}$ must be accompanied by a $\overline{\text{CHIP SELECT}}$ (see *Figures 3 and 4* for read and write cycle timing).

Test Mode

The test mode is merely a mode for production testing. It allows the counters to count at a higher than normal rate. In this mode the 32 kHz oscillator input is connected directly to the ten thousandths of seconds counter. The chip select and write lines must be low and the address must be held at $1F_H$.

Functional Description (Continued)

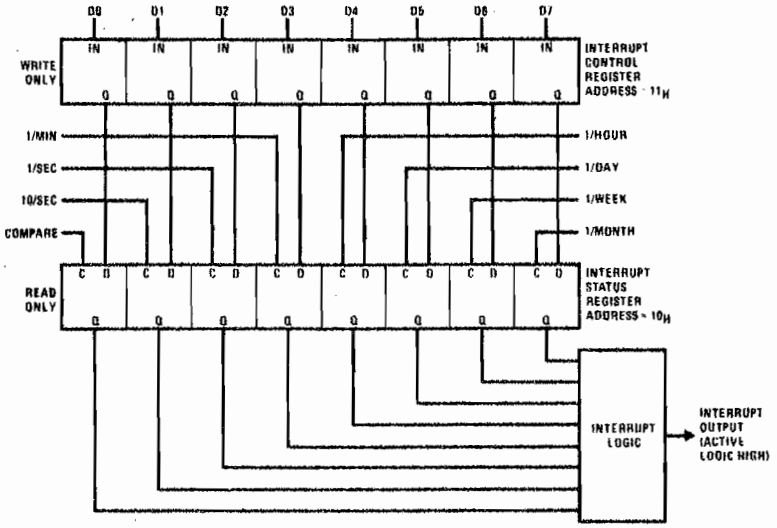


FIGURE 1. Interrupt Register Format

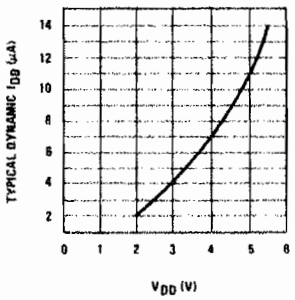


FIGURE 2. Typical Supply Current vs Supply Voltage

Interrupt Timing $-40°C \le T_A \le 85°C$, $4.5V \le V_{DD} \le 5.5V$, $V_{SS} = 0V$

	Parameter	Min	Max	Units
t_{INTON}	Status Register Clock to INTERRUPT OUTPUT (Pin 13) High (Note 1)		5	μs
t_{SBYON}	Compare Valid to $\overline{\text{STANDBY INTERRUPT}}$ (Pin 14) Low (Note 1)		5	μs
t_{INTOFF}	Trailing Edge of Status Register Read to INTERRUPT OUTPUT Low		5	μs
t_{SBYOFF}	Trailing Edge of Write Cycle (D0 = 0; Address = 16_H) to $\overline{\text{STANDBY INTERRUPT}}$ Off (High Impedance State)		5	μs

Note 1: The status register clocks are: the corresponding counter's rollover to its reset state or the compare becoming valid. The compare becomes valid 61 μs after the 1/10,000 of a second counter is clocked, if the real time counter data matches the RAM data.

Read Cycle Timing $-40°C \le T_A \le 85°C$, $4.5V \le V_{DD} \le 5.5V$, $V_{SS} = 0V$

	Parameter	Min	Max	Units
t_{AR}	Address Bus Valid to Read Strobe	100		ns
t_{CSR}	Chip Select to Read Strobe	0		ns
t_{RRY}	Read Strobe to Ready Strobe		150	ns
t_{RYD}	Ready Strobe to Data Valid		800	ns
t_{AD}	Address Bus Valid to Data Valid		1050	ns
t_{RH}	Data Hold Time From Trailing Edge of Read Strobe	0		ns
t_{HZ}	Trailing Edge of Read Strobe to TRI-STATE Mode		250	ns
t_{RYH}	Read Hold Time after Ready Strobe	0		ns
t_{RA}	Address Bus Hold Time from Trailing Edge of Read Strobe	50		ns

Write Cycle Timing $-40°C \le T_A \le 85°C$, $4.5V \le V_{DD} \le 5.5V$, $V_{SS} = 0V$

	Parameter	Min	Max	Units
t_{AW}	Address Valid to Write Strobe	100		ns
t_{CSW}	Chip Select to Write Strobe	0		ns
t_{DW}	Data Valid before Write Strobe	100		ns
t_{WRY}	Write Strobe to Ready Strobe		150	ns
t_{RY}	Ready Strobe Width		800	ns
t_{RYH}	Write Hold Time after Ready Strobe	0		ns
t_{WD}	Data Hold Time after Write Strobe	110		ns
t_{WA}	Address Hold Time after Write Strobe	50		ns

Data bus loading is 100 pF.
Ready output loading is 50 pF and 20 kΩ pull-up.
Input and output AC timing levels:
Logical one = 2.0V
Logical zero = 0.8V

Read and Write Cycle Timing Diagrams

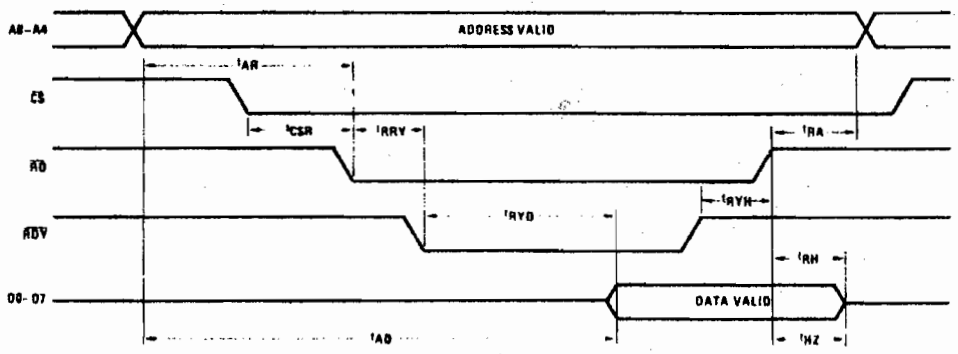


FIGURE 3. Read Cycle Timing

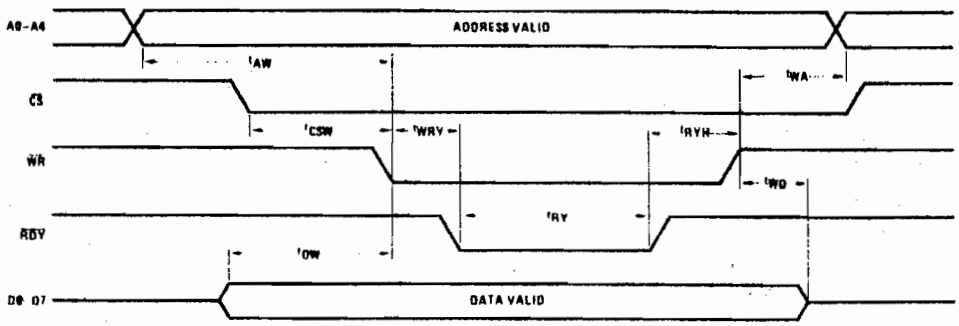
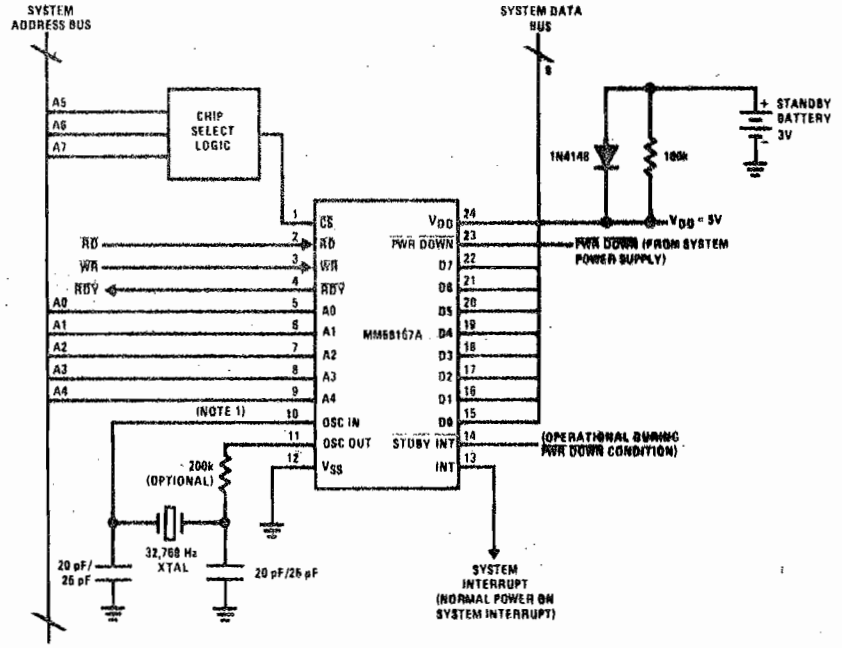


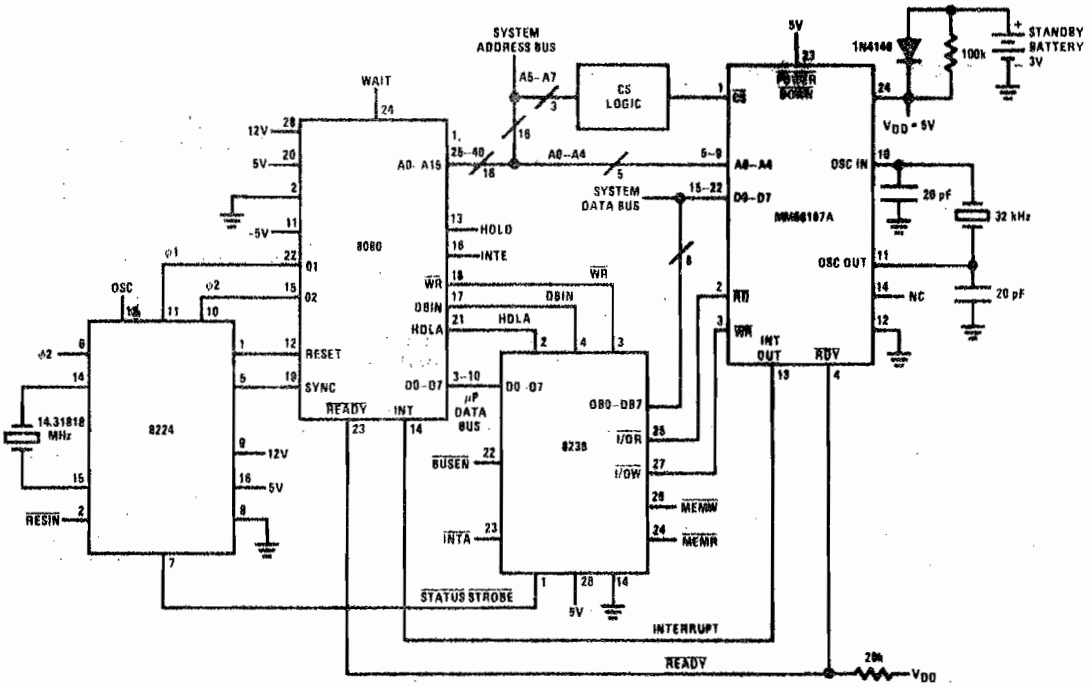
FIGURE 4. Write Cycle Timing

Typical Applications



Note 1: A ground line or ground plane guard trace should be included between pins 9 and 10 to insure the oscillator is not disturbed by the address line.

FIGURE 5. Typical Connection Diagram



Note 1: Must use 8238 or equivalent logic to insure advanced I/OW pulse; so that the ready output of the MM58167A is valid by the end of $\phi 2$ during the T2 microcycle.

Note 2: $t_{\phi 2} \geq t_{RS8080} + t_{DL8238} + t_{WRY48167}$.

FIGURE 6. 8080 System Interface with Battery Backup

Block Diagram

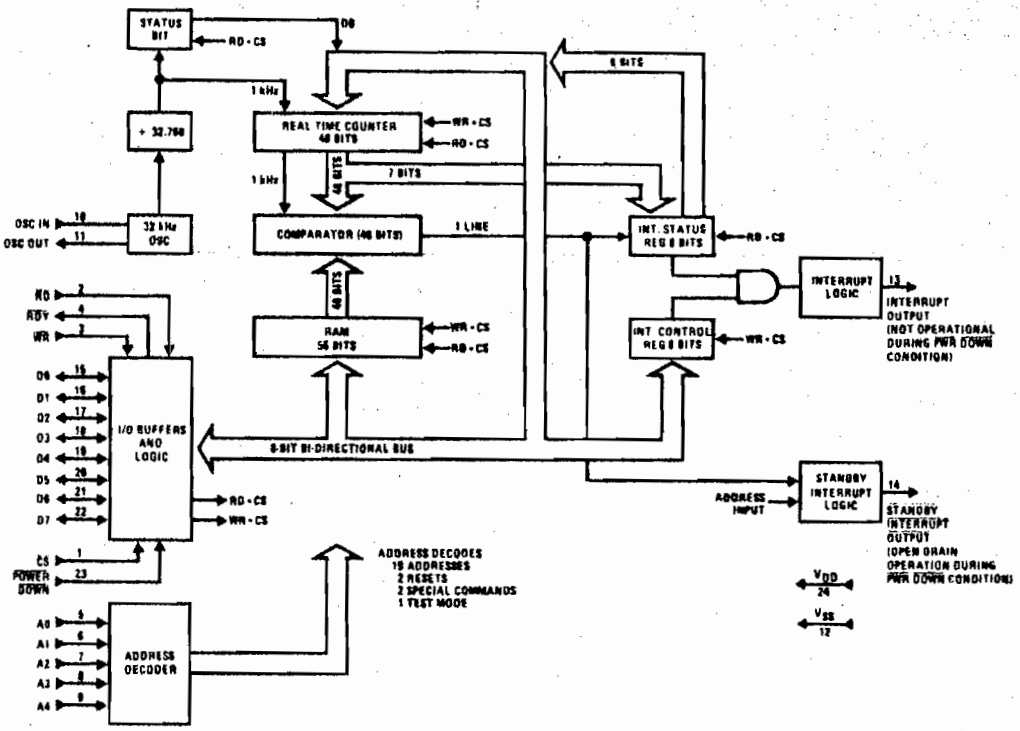
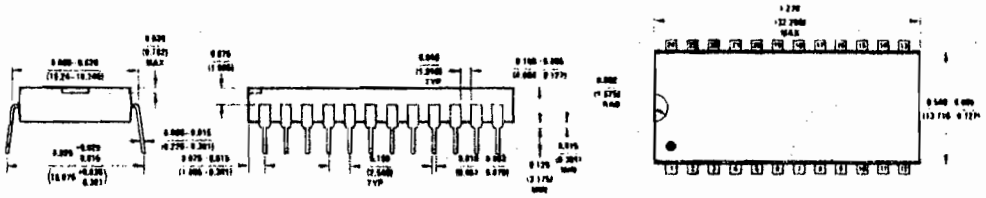


FIGURE 7

Physical Dimensions inches (millimeters)



Molded Dual-In-Line Package (N)
Order Number MM58167AN
NS Package Number N24A

LIFE SUPPORT POLICY

NATIONAL'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF NATIONAL SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

National Semiconductor Corporation
2900 Semiconductor Drive
Santa Clara, California 95051
Tel: (408) 721-5000
TWX: (910) 339-9240

National Semiconductor GmbH
Eisenheimerstrasse 63-65
8000 Munchen 21
West Germany
Tel: (089) 5706091
Telex: 522772

NS Japan K.K.
POB 2152 Shinjuku Center Building
1-25-1 Nishishinjuku, Shinjuku-ku
Tokyo 160, Japan
Tel: (03) 349-0811
Telex: J27390

National Semiconductor (Hong Kong) Ltd
[illegible]

NS Electronics Do Brasil
[illegible]

NS Electronics Pty. Ltd
[illegible]

APPENDIX E

Datablad for 8259A interrupt controller

8259A/8259A-2/8259A-8
PROGRAMMABLE INTERRUPT CONTROLLER

- iAPX 86, iAPX 88 Compatible
- MCS-80®, MCS-85® Compatible
- Eight-Level Priority Controller
- Expandable to 64 Levels
- Programmable Interrupt Modes
- Individual Request Mask Capability
- Single +5V Supply (No Clocks)
- 28-Pin Dual-In-Line Package
- Available In EXPRESS
  - Standard Temperature Range
  - Extended Temperature Range

The Intel® 8259A Programmable Interrupt Controller handles up to eight vectored priority interrupts for the CPU. It is cascadable for up to 64 vectored priority interrupts without additional circuitry. It is packaged in a 28-pin DIP, uses NMOS technology and requires a single +5V supply. Circuitry is static, requiring no clock input.

The 8259A is designed to minimize the software and real time overhead in handling multi-level priority interrupts. It has several modes, permitting optimization for a variety of system requirements.

The 8259A is fully upward compatible with the Intel® 8259. Software originally written for the 8259 will operate the 8259A in all 8259 equivalent modes (MCS-80/85, Non-Buffered, Edge Triggered).

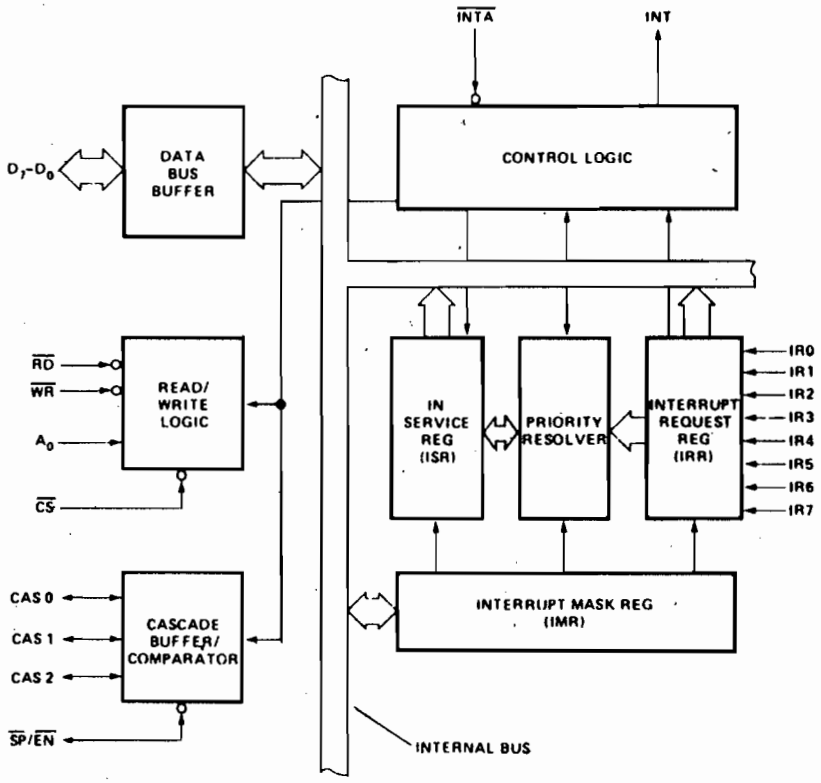


Figure 1. Block Diagram

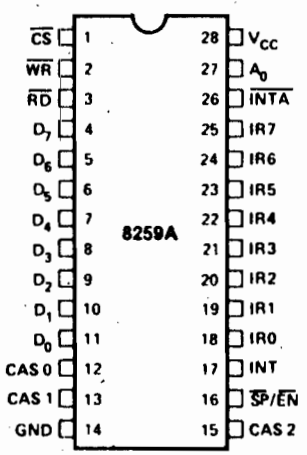


Figure 2. Pin Configuration

Table 1. Pin Description

Symbol	Pin No.	Type	Name and Function
V_{CC}	28	I	**Supply:** +5V Supply.
GND	14	I	**Ground.**
$\overline{CS}$	1	I	**Chip Select:** A low on this pin enables $\overline{RD}$ and $\overline{WR}$ communication between the CPU and the 8259A. INTA functions are independent of CS.
$\overline{WR}$	2	I	**Write:** A low on this pin when CS is low enables the 8259A to accept command words from the CPU.
$\overline{RD}$	3	I	**Read:** A low on this pin when CS is low enables the 8259A to release status onto the data bus for the CPU.
D_7–D_0	4–11	I/O	**Bidirectional Data Bus:** Control, status and interrupt-vector information is transferred via this bus.
CAS_0–CAS_2	12, 13, 15	I/O	**Cascade Lines:** The CAS lines form a private 8259A bus to control a multiple 8259A structure. These pins are outputs for a master 8259A and inputs for a slave 8259A.
$\overline{SP}/\overline{EN}$	16	I/O	**Slave Program/Enable Buffer:** This is a dual function pin. When in the Buffered Mode it can be used as an output to control buffer transceivers (EN). When not in the buffered mode it is used as an input to designate a master (SP = 1) or slave (SP = 0).
INT	17	O	**Interrupt:** This pin goes high whenever a valid interrupt request is asserted. It is used to interrupt the CPU, thus it is connected to the CPU's interrupt pin.
IR_0–IR_7	18–25	I	**Interrupt Requests:** Asynchronous inputs. An interrupt request is executed by raising an IR input (low to high), and holding it high until it is acknowledged (Edge Triggered Mode), or just by a high level on an IR input (Level Triggered Mode).
$\overline{INTA}$	26	I	**Interrupt Acknowledge:** This pin is used to enable 8259A interrupt-vector data onto the data bus by a sequence of interrupt acknowledge pulses issued by the CPU.
A_0	27	I	**A0 Address Line:** This pin acts in conjunction with the $\overline{CS}$, $\overline{WR}$, and $\overline{RD}$ pins. It is used by the 8259A to decipher various Command Words the CPU writes and status the CPU wishes to read. It is typically connected to the CPU A0 address line (A1 for iAPX 86, 88).

FUNCTIONAL DESCRIPTION

Interrupts in Microcomputer Systems

Microcomputer system design requires that I/O devices such as keyboards, displays, sensors and other components receive servicing in an efficient manner so that large amounts of the total system tasks can be assumed by the microcomputer with little or no effect on throughput.

The most common method of servicing such devices is the *Polled* approach. This is where the processor must test each device in sequence and in effect "ask" each one if it needs servicing. It is easy to see that a large portion of the main program is looping through this continuous polling cycle and that such a method would have a serious, detrimental effect on system throughput, thus limiting the tasks that could be assumed by the microcomputer and reducing the cost effectiveness of using such devices.

A more desirable method would be one that would allow the microprocessor to be executing its main program and only stop to service peripheral devices when it is told to do so by the device itself. In effect, the method would provide an external asynchronous input that would inform the processor that it should complete whatever instruction that is currently being executed and fetch a new routine that will service the requesting device. Once this servicing is complete, however, the processor would resume exactly where it left off.

This method is called *Interrupt*. It is easy to see that system throughput would drastically increase, and thus more tasks could be assumed by the microcomputer to further enhance its cost effectiveness.

The Programmable Interrupt Controller (PIC) functions as an overall manager in an Interrupt-Driven system environment. It accepts requests from the peripheral equipment, determines which of the incoming requests is of the highest importance (priority), ascertains whether the incoming request has a higher priority value than the level currently being serviced, and issues an interrupt to the CPU based on this determination.

Each peripheral device or structure usually has a special program or "routine" that is associated with its specific functional or operational requirements; this is referred to as a "service routine". The PIC, after issuing an interrupt to the CPU, must somehow input information into the CPU that can "point" the Program Counter to the service routine associated with the requesting device. This "pointer" is an address in a vectoring table and will often be referred to, in this document, as vectoring data.

The 8259A

The 8259A is a device specifically designed for use in real time, interrupt driven microcomputer systems. It manages eight levels or requests and has built-in features for expandability to other 8259A's (up to 64 levels). It is programmed by the system's software as an I/O peripheral. A selection of priority modes is available to the programmer so that the manner in which the requests are processed by the 8259A can be configured to match his system requirements. The priority modes can be changed or reconfigured dynamically at any time during the main program. This means that the complete interrupt structure can be defined as required, based on the total system environment.

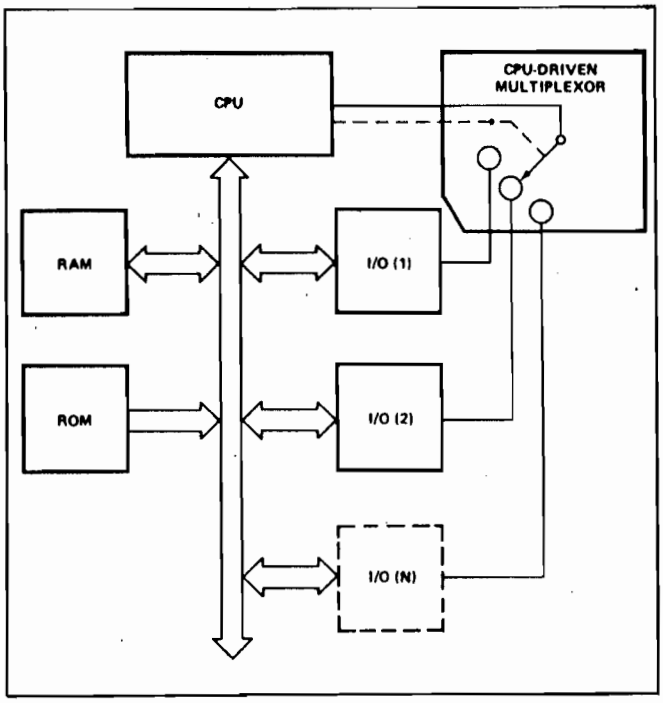


Figure 3a. Polled Method

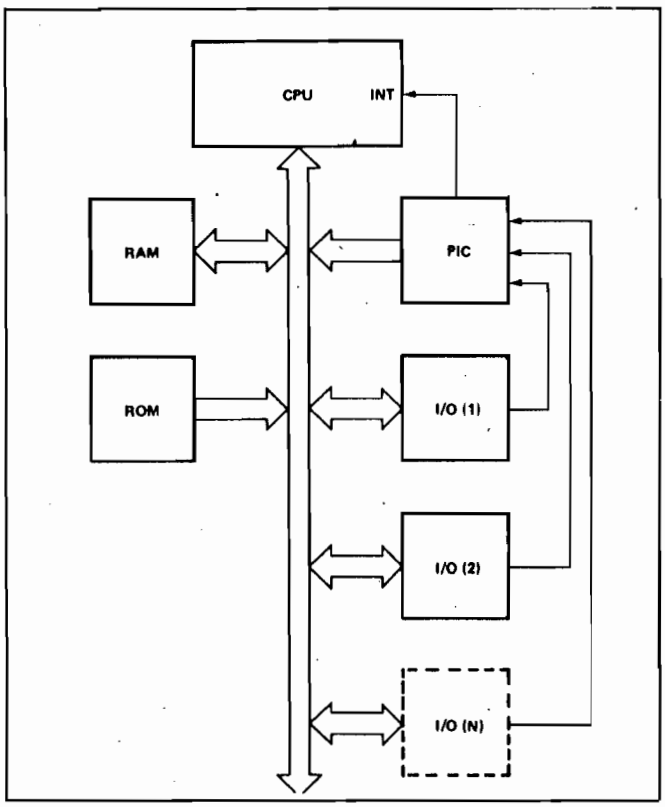


Figure 3b. Interrupt Method

INTERRUPT REQUEST REGISTER (IRR) AND IN-SERVICE REGISTER (ISR)

The interrupts at the IR input lines are handled by two registers in cascade, the Interrupt Request Register (IRR) and the In-Service Register (ISR). The IRR is used to store all the interrupt levels which are requesting service; and the ISR is used to store all the interrupt levels which are being serviced.

PRIORITY RESOLVER

This logic block determines the priorities of the bits set in the IRR. The highest priority is selected and strobed into the corresponding bit of the ISR during $\overline{INTA}$ pulse.

INTERRUPT MASK REGISTER (IMR)

The IMR stores the bits which mask the interrupt lines to be masked. The IMR operates on the IRR. Masking of a higher priority input will not affect the interrupt request lines of lower priority.

INT (INTERRUPT)

This output goes directly to the CPU interrupt input. The V_{OH} level on this line is designed to be fully compatible with the 8080A, 8085A and 8086 input levels.

$\overline{INTA}$ (INTERRUPT ACKNOWLEDGE)

$\overline{INTA}$ pulses will cause the 8259A to release vectoring information onto the data bus. The format of this data depends on the system mode (μPM) of the 8259A.

DATA BUS BUFFER

This 3-state, bidirectional 8-bit buffer is used to interface the 8259A to the system Data Bus. Control words and status information are transferred through the Data Bus Buffer.

READ/WRITE CONTROL LOGIC

The function of this block is to accept OUTput commands from the CPU. It contains the Initialization Command Word (ICW) registers and Operation Command Word (OCW) registers which store the various control formats for device operation. This function block also allows the status of the 8259A to be transferred onto the Data Bus.

$\overline{CS}$ (CHIP SELECT)

A LOW on this input enables the 8259A. No reading or writing of the chip will occur unless the device is selected.

$\overline{WR}$ (WRITE)

A LOW on this input enables the CPU to write control words (ICWs and OCWs) to the 8259A.

$\overline{RD}$ (READ)

A LOW on this input enables the 8259A to send the status of the Interrupt Request Register (IRR), In Service Register (ISR), the Interrupt Mask Register (IMR), or the Interrupt level onto the Data Bus.

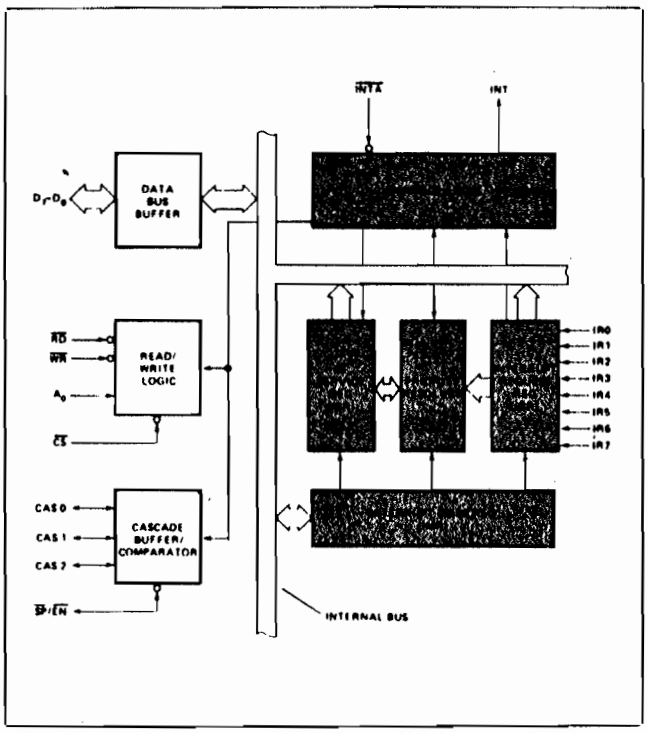


Figure 4a. 8259A Block Diagram

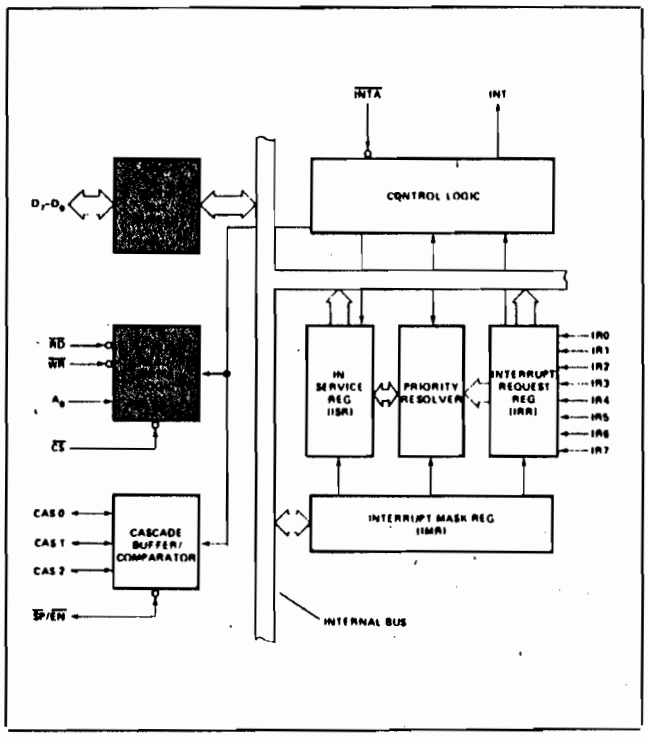


Figure 4b. 8259A Block Diagram

A_0

This input signal is used in conjunction with $\overline{WR}$ and $\overline{RD}$ signals to write commands into the various command registers, as well as reading the various status registers of the chip. This line can be tied directly to one of the address lines.

THE CASCADE BUFFER/COMPARATOR

This function block stores and compares the IDs of all 8259A's used in the system. The associated three I/O pins (CAS0-2) are outputs when the 8259A is used as a master and are inputs when the 8259A is used as a slave. As a master, the 8259A sends the ID of the interrupting slave device onto the CAS0-2 lines. The slave thus selected will send its preprogrammed subroutine address onto the Data Bus during the next one or two consecutive $\overline{INTA}$ pulses. (See section "Cascading the 8259A".)

INTERRUPT SEQUENCE

The powerful features of the 8259A in a microcomputer system are its programmability and the interrupt routine addressing capability. The latter allows direct or indirect jumping to the specific interrupt routine requested without any polling of the interrupting devices. The normal sequence of events during an interrupt depends on the type of CPU being used.

The events occur as follows in an MCS-80/85 system:

1. One or more of the INTERRUPT REQUEST lines (IR7-0) are raised high, setting the corresponding IRR bit(s).
2. The 8259A evaluates these requests, and sends an INT to the CPU, if appropriate.
3. The CPU acknowledges the INT and responds with an $\overline{INTA}$ pulse.
4. Upon receiving an $\overline{INTA}$ from the CPU group, the highest priority ISR bit is set, and the corresponding IRR bit is reset. The 8259A will also release a CALL instruction code (11001101) onto the 8-bit Data Bus through its D7-0 pins.
5. This CALL instruction will initiate two more $\overline{INTA}$ pulses to be sent to the 8259A from the CPU group.
6. These two $\overline{INTA}$ pulses allow the 8259A to release its preprogrammed subroutine address onto the Data Bus. The lower 8-bit address is released at the first $\overline{INTA}$ pulse and and the higher 8-bit address is released at the second $\overline{INTA}$ pulse.
7. This completes the 3-byte CALL instruction released by the 8259A. In the AEOI mode the ISR bit is reset at the end of the third $\overline{INTA}$ pulse. Otherwise, the ISR bit remains set until an appropriate EOI command is issued at the end of the interrupt sequence.

The events occurring in an iAPX 86 system are the same until step 4.

4. Upon receiving an $\overline{INTA}$ from the CPU group, the highest priority ISR bit is set and the corresponding IRR bit is reset. The 8259A does not drive the Data Bus during this cycle.
5. The iAPX 86/10 will initiate a second $\overline{INTA}$ pulse. During this pulse, the 8259A releases an 8-bit pointer onto the Data Bus where it is read by the CPU.
6. This completes the interrupt cycle. In the AEOI mode the ISR bit is reset at the end of the second $\overline{INTA}$ pulse. Otherwise, the ISR bit remains set until an appropriate EOI command is issued at the end of the interrupt subroutine.

If no interrupt request is present at step 4 of either sequence (i.e., the request was tod short in duration) the 8259A will issue an interrupt level 7. Both the vectoring bytes and the CAS lines will look like an interrupt level 7 was requested.

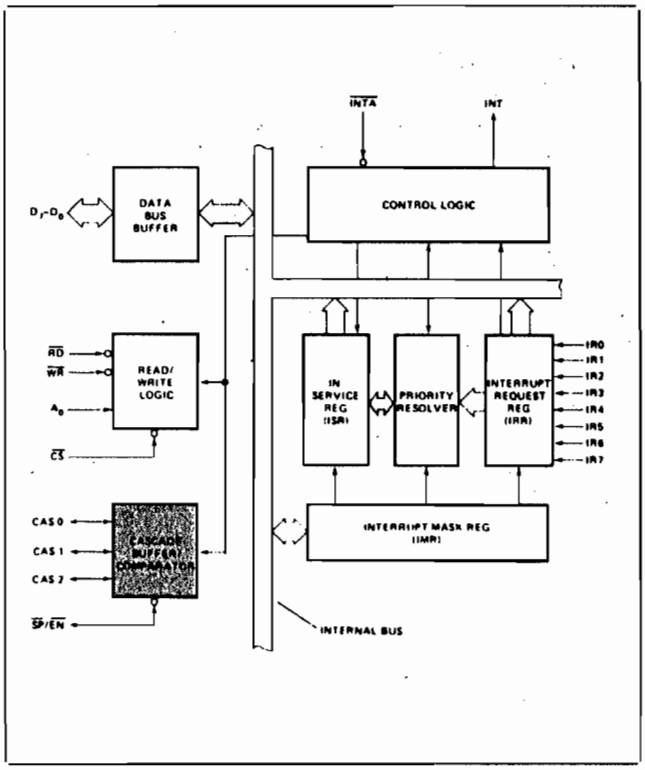


Figure 4c. 8259A Block Diagram

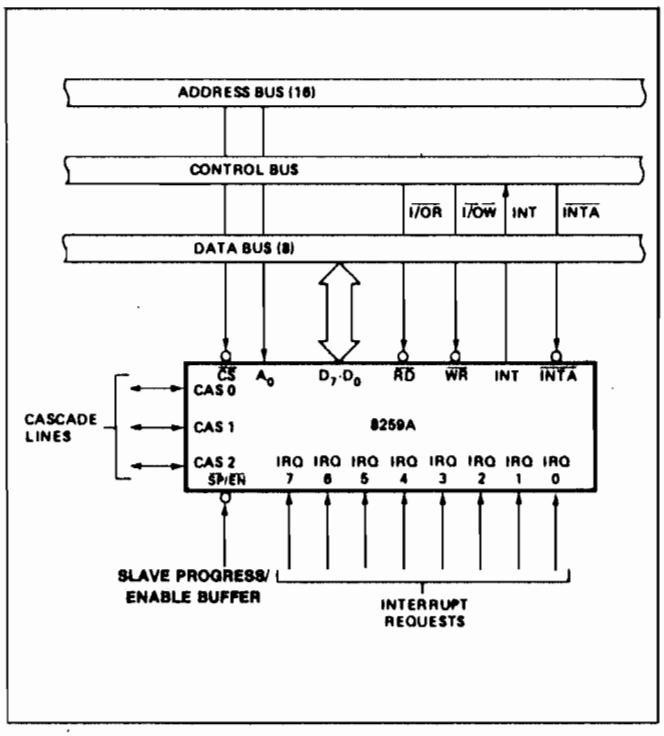


Figure 5. 8259A Interface to Standard System Bus

INTERRUPT SEQUENCE OUTPUTS

MCS-80®, MCS-85®

This sequence is timed by three $\overline{INTA}$ pulses. During the first $\overline{INTA}$ pulse the CALL opcode is enabled onto the data bus.

Content of First Interrupt Vector Byte

	D7	D6	D5	D4	D3	D2	D1	D0
CALL CODE	1	1	0	0	1	1	0	1

During the second $\overline{INTA}$ pulse the lower address of the appropriate service routine is enabled onto the data bus. When Interval = 4 bits A_5-A_7 are programmed, while A_0-A_4 are automatically inserted by the 8259A. When Interval = 8 only A_6 and A_7 are programmed, while A_0-A_5 are automatically inserted.

Content of Second Interrupt Vector Byte

IR	Interval = 4							
	D7	D6	D5	D4	D3	D2	D1	D0
7	A7	A6	A5	1	1	1	0	0
6	A7	A6	A5	1	1	0	0	0
5	A7	A6	A5	1	0	1	0	0
4	A7	A6	A5	1	0	0	0	0
3	A7	A6	A5	0	1	1	0	0
2	A7	A6	A5	0	1	0	0	0
1	A7	A6	A5	0	0	1	0	0
0	A7	A6	A5	0	0	0	0	0

IR	Interval = 8							
	D7	D6	D5	D4	D3	D2	D1	D0
7	A7	A6	1	1	1	0	0	0
6	A7	A6	1	1	0	0	0	0
5	A7	A6	1	0	1	0	0	0
4	A7	A6	1	0	0	0	0	0
3	A7	A6	0	1	1	0	0	0
2	A7	A6	0	1	0	0	0	0
1	A7	A6	0	0	1	0	0	0
0	A7	A6	0	0	0	0	0	0

During the third INTA pulse the higher address of the appropriate service routine, which was programmed as byte 2 of the initialization sequence ($A_8 - A_{15}$), is enabled onto the bus.

Content of Third Interrupt Vector Byte

D7	D6	D5	D4	D3	D2	D1	D0
A15	A14	A13	A12	A11	A10	A9	A8

iAPX 86, iAPX 88

iAPX 86 mode is similar to MCS-80 mode except that only two Interrupt Acknowledge cycles are issued by the processor and no CALL opcode is sent to the processor. The first interrupt acknowledge cycle is similar to that of MCS-80, 85 systems in that the 8259A uses it to internally freeze the state of the interrupts for priority resolution and as a master it issues the interrupt code on the cascade lines at the end of the INTA pulse. On this first cycle it does not issue any data to the processor and leaves its data bus buffers disabled. On the second interrupt acknowledge cycle in iAPX 86 mode the master (or slave if so programmed) will send a byte of data to the processor with the acknowledged interrupt code composed as follows (note the state of the ADI mode control is ignored and A_5–A_{11} are unused in iAPX 86 mode):

Content of Interrupt Vector Byte for iAPX 86 System Mode

	D7	D6	D5	D4	D3	D2	D1	D0
IR7	T7	T6	T5	T4	T3	1	1	1
IR6	T7	T6	T5	T4	T3	1	1	0
IR5	T7	T6	T5	T4	T3	1	0	1
IR4	T7	T6	T5	T4	T3	1	0	0
IR3	T7	T6	T5	T4	T3	0	1	1
IR2	T7	T6	T5	T4	T3	0	1	0
IR1	T7	T6	T5	T4	T3	0	0	1
IR0	T7	T6	T5	T4	T3	0	0	0

PROGRAMMING THE 8259A

The 8259A accepts two types of command words generated by the CPU:

1. *Initialization Command Words (ICWs):* Before normal operation can begin, each 8259A in the system must be brought to a starting point — by a sequence of 2 to 4 bytes timed by $\overline{WR}$ pulses.
2. *Operation Command Words (OCWs):* These are the command words which command the 8259A to operate in various interrupt modes. These modes are:
   a. Fully nested mode
   b. Rotating priority mode
   c. Special mask mode
   d. Polled mode

The OCWs can be written into the 8259A anytime after initialization.

INITIALIZATION COMMAND WORDS (ICWS)

GENERAL

Whenever a command is issued with A0 = 0 and D4 = 1, this is interpreted as Initialization Command Word 1 (ICW1). ICW1 starts the initialization sequence during which the following automatically occur.

a. The edge sense circuit is reset, which means that following initialization, an interrupt request (IR) input must make a low-to-high transition to generate an interrupt.
b. The Interrupt Mask Register is cleared.
c. IR7 input is assigned priority 7.
d. The slave mode address is set to 7.
e. Special Mask Mode is cleared and Status Read is set to IRR.
f. If IC4 = 0, then all functions selected in ICW4 are set to zero. (Non-Buffered mode*, no Auto-EOI, MCS-80, 85 system).

*Note: Master/Slave in ICW4 is only used in the buffered mode.

INITIALIZATION COMMAND WORDS 1 AND 2 (ICW1, ICW2)

A_5-A_{15}: *Page starting address of service routines.* In an MCS 80/85 system, the 8 request levels will generate CALLs to 8 locations equally spaced in memory. These can be programmed to be spaced at intervals of 4 or 8 memory locations, thus the 8 routines will occupy a page of 32 or 64 bytes, respectively.

The address format is 2 bytes long (A_0-A_{15}). When the routine interval is 4, A_0-A_4 are automatically inserted by the 8259A, while A_5-A_{15} are programmed externally. When the routine interval is 8, A_0-A_5 are automatically inserted by the 8259A, while A_6-A_{15} are programmed externally.

The 8-byte interval will maintain compatibility with current software, while the 4-byte interval is best for a compact jump table.

In an iAPX 86 system A_{15}-A_{11} are inserted in the five most significant bits of the vectoring byte and the 8259A sets the three least significant bits according to the interrupt level. A_{10}-A_5 are ignored and ADI (Address interval) has no effect.

LTIM: If LTIM = 1, then the 8259A will operate in the level interrupt mode. Edge detect logic on the interrupt inputs will be disabled.

ADI: CALL address interval. ADI = 1 then interval = 4; ADI = 0 then interval = 8.

SNGL: Single. Means that this is the only 8259A in the system. If SNGL = 1 no ICW3 will be issued.

IC4: If this bit is set — ICW4 has to be read. If ICW4 is not needed, set IC4 = 0.

INITIALIZATION COMMAND WORD 3 (ICW3)

This word is read only when there is more than one 8259A in the system and cascading is used, in which case SNGL = 0. It will load the 8-bit slave register. The functions of this register are:

a. In the master mode (either when SP = 1, or in buffered mode when M/S = 1 in ICW4) a "1" is set for each slave in the system. The master then will release byte 1 of the call sequence (for MCS-80/85 system) and will enable the corresponding slave to release bytes 2 and 3 (for iAPX 86 only byte 2) through the cascade lines.

b. In the slave mode (either when $\overline{SP}$ = 0, or if BUF = 1 and M/S = 0 in ICW4) bits 2-0 identify the slave. The slave compares its cascade input with these bits and, if they are equal, bytes 2 and 3 of the call sequence (or just byte 2 for iAPX 86 are released by it on the Data Bus.

INITIALIZATION COMMAND WORD 4 (ICW4)

SFNM: If SFNM = 1 the special fully nested mode is programmed.

BUF: If BUF = 1 the buffered mode is programmed. In buffered mode $\overline{SP}/\overline{EN}$ becomes an enable output and the master/slave determination is by M/S.

M/S: If buffered mode is selected: M/S = 1 means the 8259A is programmed to be a master, M/S = 0 means the 8259A is programmed to be a slave. If BUF = 0, M/S has no function.

AEOI: If AEOI = 1 the automatic end of interrupt mode is programmed.

μPM: Microprocessor mode: μPM = 0 sets the 8259A for MCS-80, 85 system operation, μPM = 1 sets the 8259A for iAPX 86 system operation.

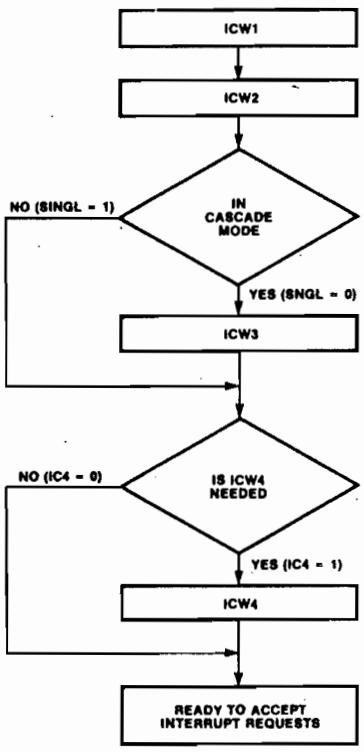
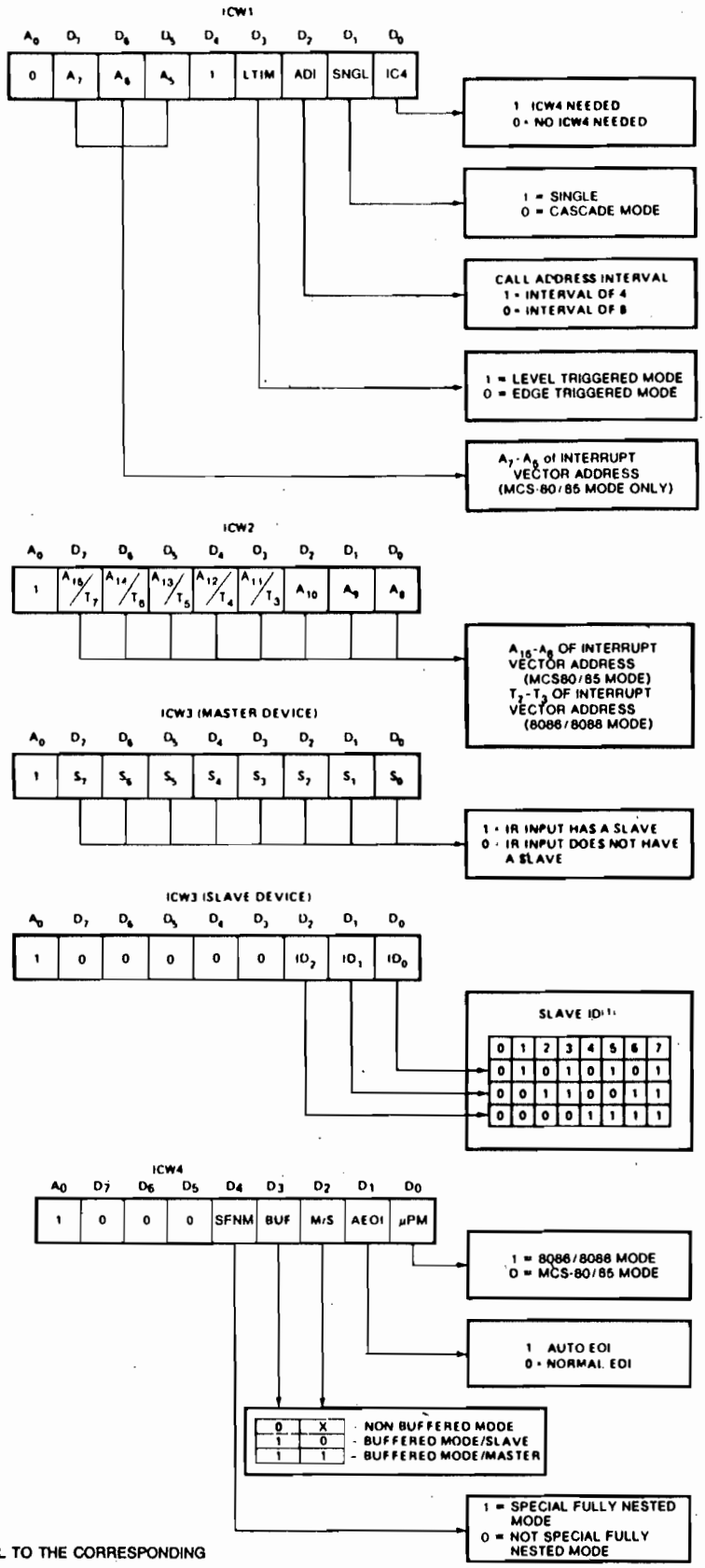


Figure 6. Initialization Sequence



NOTE 1: SLAVE ID IS EQUAL TO THE CORRESPONDING MASTER IR INPUT.

Figure 7. Initialization Command Word Format

OPERATION COMMAND WORDS (OCWs)

After the Initialization Command Words (ICWs) are programmed into the 8259A, the chip is ready to accept interrupt requests at its input lines. However, during the 8259A operation, a selection of algorithms can command the 8259A to operate in various modes through the Operation Command Words (OCWs).

OPERATION CONTROL WORDS (OCWs)

OCW1

A0	D7	D6	D5	D4	D3	D2	D1	D0
1	M7	M6	M5	M4	M3	M2	M1	M0

OCW2

0	R	SL	EOI	0	0	L2	L1	L0

OCW3

0	0	ESMM	SMM	0	1	P	RR	RIS

OPERATION CONTROL WORD 1 (OCW1)

OCW1 sets and clears the mask bits in the interrupt Mask Register (IMR). $M_7 - M_0$ represent the eight mask bits. M = 1 indicates the channel is masked (inhibited), M = 0 indicates the channel is enabled.

OPERATION CONTROL WORD 2 (OCW2)

R, SL, EOI — These three bits control the Rotate and End of Interrupt modes and combinations of the two. A chart of these combinations can be found on the Operation Command Word Format.

L_2, L_1, L_0—These bits determine the interrupt level acted upon when the SL bit is active.

OPERATION CONTROL WORD 3 (OCW3)

ESMM — Enable Special Mask Mode. When this bit is set to 1 it enables the SMM bit to set or reset the Special Mask Mode. When ESMM = 0 the SMM bit becomes a "don't care".

SMM — Special Mask Mode. If ESMM = 1 and SMM = 1 the 8259A will enter Special Mask Mode. If ESMM = 1 and SMM = 0 the 8259A will revert to normal mask mode. When ESMM = 0, SMM has no effect.

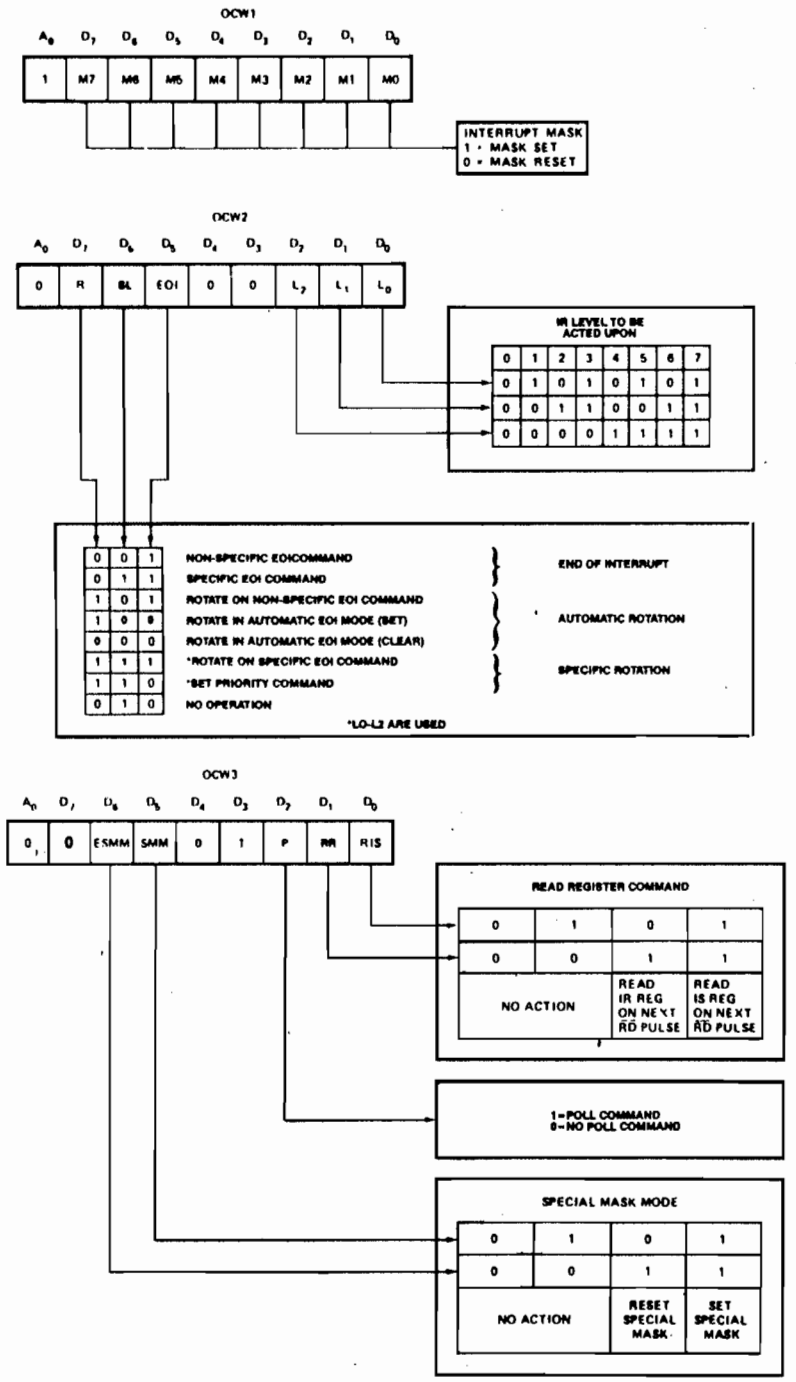


Figure 8. Operation Command Word Format

FULLY NESTED MODE

This mode is entered after initialization unless another mode is programmed. The interrupt requests are ordered in priority form 0 through 7 (0 highest). When an interrupt is acknowledged the highest priority request is determined and its vector placed on the bus. Additionally, a bit of the Interrupt Service register (ISO-7) is set. This bit remains set until the microprocessor issues an End of Interrupt (EOI) command immediately before returning from the service routine, or if AEOI (Automatic End of Interrupt) bit is set, until the trailing edge of the last INTA. While the IS bit is set, all further interrupts of the same or lower priority are inhibited, while higher levels will generate an interrupt (which will be acknowledged only if the microprocessor internal interrupt enable flip-flop has been re-enabled through software).

After the initialization sequence, IR0 has the highest priority and IR7 the lowest. Priorities can be changed, as will be explained, in the rotating priority mode.

END OF INTERRUPT (EOI)

The In Service (IS) bit can be reset either automatically following the trailing edge of the last in sequence $\overline{\text{INTA}}$ pulse (when AEOI bit in ICW1 is set) or by a command word that must be issued to the 8259A before returning from a service routine (EOI command). An EOI command must be issued twice if in the Cascade mode, once for the master and once for the corresponding slave.

There are two forms of EOI command: Specific and Non-Specific. When the 8259A is operated in modes which preserve the fully nested structure, it can determine which IS bit to reset on EOI. When a Non-Specific EOI command is issued the 8259A will automatically reset the highest IS bit of those that are set, since in the fully nested mode the highest IS level was necessarily the last level acknowledged and serviced. A non-specific EOI can be issued with OCW2 (EOI = 1, SL = 0, R = 0).

When a mode is used which may disturb the fully nested structure, the 8259A may no longer be able to determine the last level acknowledged. In this case a Specific End of Interrupt must be issued which includes as part of the command the IS level to be reset. A specific EOI can be issued with OCW2 (EOI = 1, SL = 1, R = 0, and LO-L2 is the binary level of the IS bit to be reset).

It should be noted that an IS bit that is masked by an IMR bit will not be cleared by a non-specific EOI if the 8259A is in the Special Mask Mode.

AUTOMATIC END OF INTERRUPT (AEOI) MODE

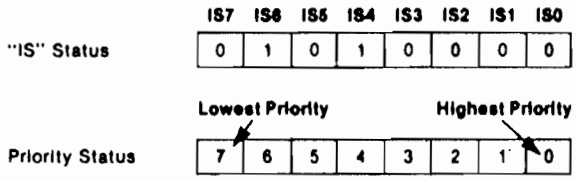
If AEOI = 1 in ICW4, then the 8259A will operate in AEOI mode continuously until reprogrammed by ICW4. In this mode the 8259A will automatically perform a non-specific EOI operation at the trailing edge of the last interrupt acknowledge pulse (third pulse in MCS-80/85, second in iAPX 86). Note that from a system standpoint, this mode should be used only when a nested multilevel interrupt structure is not required within a single 8259A.

The AEOI mode can only be used in a master 8259A and not a slave.

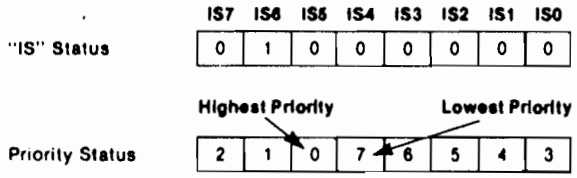
AUTOMATIC ROTATION (Equal Priority Devices)

In some applications there are a number of interrupting devices of equal priority. In this mode a device, after being serviced, receives the lowest priority, so a device requesting an interrupt will have to wait, in the worst case until each of 7 other devices are serviced at most *once*. For example, if the priority and "in service" status is:

Before Rotate (IR4 the highest priority requiring service)



After Rotate (IR4 was serviced, all other priorities rotated correspondingly)



There are two ways to accomplish Automatic Rotation using OCW2, the Rotation on Non-Specific EOI Command (R = 1, SL = 0, EOI = 1) and the Rotate in Automatic EOI Mode which is set by (R = 1, SL = 0, EOI = 0) and cleared by (R = 0, SL = 0, EOI = 0).

SPECIFIC ROTATION (Specific Priority)

The programmer can change priorities by programming the bottom priority and thus fixing all other priorities; i.e., if IR5 is programmed as the bottom priority device, then IR6 will have the highest one.

The Set Priority command is issued in OCW2 where: R = 1, SL = 1; LO–L2 is the binary priority level code of the bottom priority device.

Observe that in this mode internal status is updated by software control during OCW2. However, it is independent of the End of Interrupt (EOI) command (also executed by OCW2). Priority changes can be executed during an EOI command by using the Rotate on Specific EOI command in OCW2 (R = 1, SL = 1, EOI = 1 and LO–L2 = IR level to receive bottom priority).

INTERRUPT MASKS

Each Interrupt Request input can be masked individually by the Interrupt Mask Register (IMR) programmed through OCW1. Each bit in the IMR masks one interrupt channel if it is set (1). Bit 0 masks IR0, Bit 1 masks IR1 and so forth. Masking an IR channel does not affect the other channels operation.

SPECIAL MASK MODE

Some applications may require an interrupt service routine to dynamically alter the system priority structure during its execution under software control. For example, the routine may wish to inhibit lower priority requests for a portion of its execution but enable some of them for another portion.

The difficulty here is that if an Interrupt Request is acknowledged and an End of Interrupt command did not reset its IS bit (i.e., while executing a service routine), the 8259A would have inhibited all lower priority requests with no easy way for the routine to enable them

That is where the Special Mask Mode comes in. In the special Mask Mode, when a mask bit is set in OCW1, it inhibits further interrupts at that level *and enables* interrupts from *all other* levels (lower as well as higher) that are not masked.

Thus, any interrupts may be selectively enabled by loading the mask register.

The special Mask Mode is set by OCW3 where: SSMM = 1, SMM = 1, and cleared where SSMM = 1, SMM = 0.

POLL COMMAND

In this mode the INT output is not used or the microprocessor internal Interrupt Enable flip-flop is reset, disabling its interrupt input. Service to devices is achieved by software using a Poll command.

The Poll command is issued by setting P = "1" in OCW3. The 8259A treats the next $\overline{RD}$ pulse to the 8259A (i.e., $\overline{RD} = 0$, $\overline{CS} = 0$) as an interrupt acknowledge, sets the appropriate IS bit if there is a request, and reads the priority level. Interrupt is frozen from $\overline{WR}$ to $\overline{RD}$.

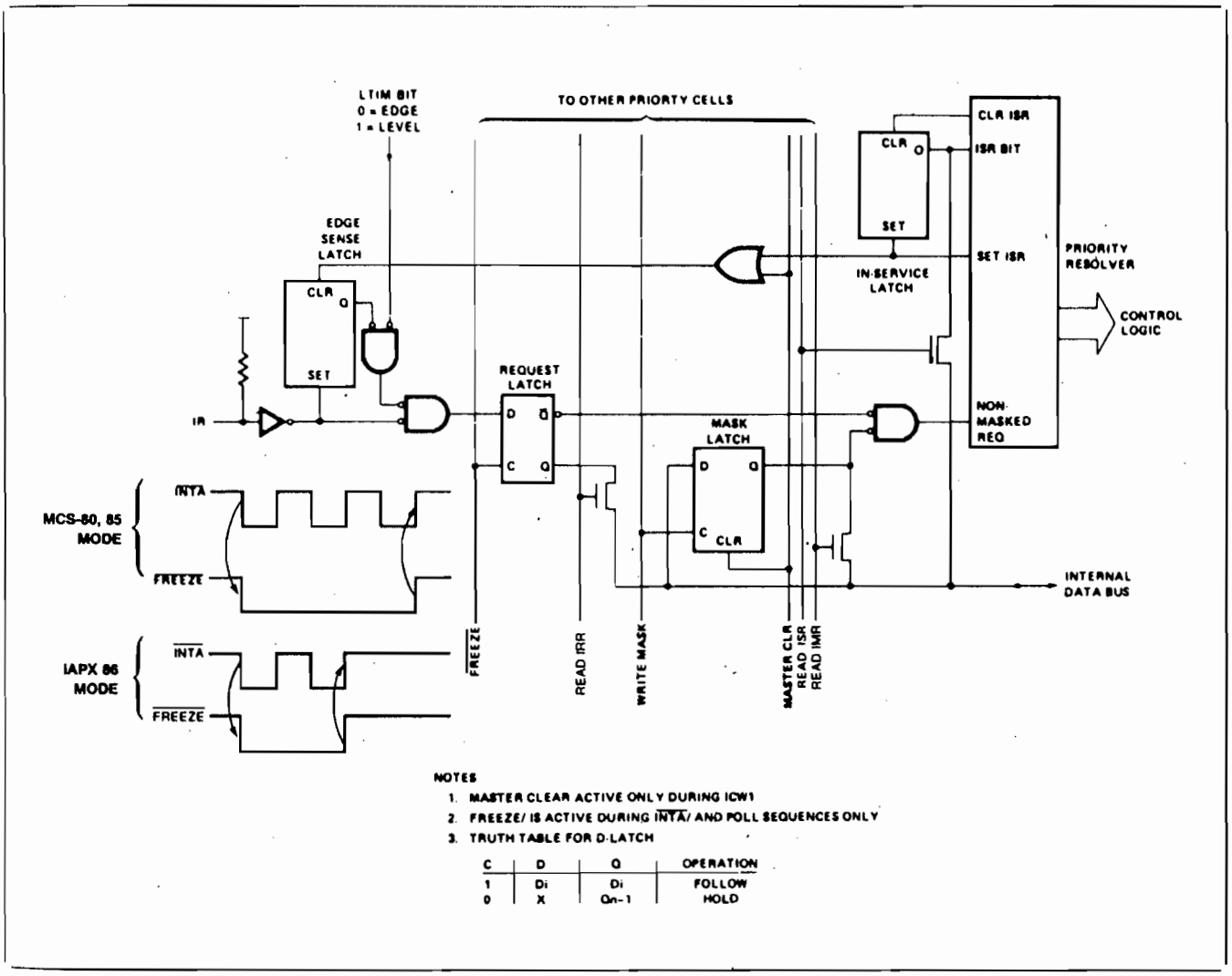
The word enabled onto the data bus during $\overline{RD}$ is:

D7	D6	D5	D4	D3	D2	D1	D0
I	—	—	—	—	W2	W1	W0

W0-W2: Binary code of the highest priority level requesting service.

I: Equal to a "1" if there is an interrupt.

This mode is useful if there is a routine command common to several levels so that the $\overline{INTA}$ sequence is not needed (saves ROM space). Another application is to use the poll mode to expand the number of priority levels to more than 64.



NOTES

1. MASTER CLEAR ACTIVE ONLY DURING ICW1
2. FREEZE/ IS ACTIVE DURING $\overline{INTA}$/ AND POLL SEQUENCES ONLY
3. TRUTH TABLE FOR D-LATCH

C	D	Q	OPERATION
1	Di	Di	FOLLOW
0	X	Qn-1	HOLD

Figure 9. Priority Cell—Simplified Logic Diagram

READING THE 8259A STATUS

The input status of several internal registers can be read to update the user information on the system. The following registers can be read via OCW3 (IRR and ISR or OCW1 [IMR]).

Interrupt Request Register (IRR): 8-bit register which contains the levels requesting an interrupt to be acknowledged. The highest request level is reset from the IRR when an interrupt is acknowledged. (Not affected by IMR.)

In-Service Register (ISR): 8-bit register which contains the priority levels that are being serviced. The ISR is updated when an End of Interrupt Command is issued.

Interrupt Mask Register: 8-bit register which contains the interrupt request lines which are masked.

The IRR can be read when, prior to the RD pulse, a Read Register Command is issued with OCW3 (RR = 1, RIS = 0.)

The ISR can be read when, prior to the RD pulse, a Read Register Command is issued with OCW3 (RR = 1, RIS = 1).

There is no need to write an OCW3 before every status read operation, as long as the status read corresponds with the previous one; i.e., the 8259A "remembers" whether the IRR or ISR has been previously selected by the OCW3. This is not true when poll is used.

After initialization the 8259A is set to IRR.

For reading the IMR, no OCW3 is needed. The output data bus will contain the IMR whenever $\overline{RD}$ is active and AO = 1 (OCW1).

Polling overrides status read when P = 1, RR = 1 in OCW3.

EDGE AND LEVEL TRIGGERED MODES

This mode is programmed using bit 3 in ICW1.

If LTIM = '0', an interrupt request will be recognized by a low to high transition on an IR input. The IR input can remain high without generating another interrupt.

If LTIM = '1', an interrupt request will be recognized by a 'high' level on IR Input, and there is no need for an edge detection. The interrupt request must be removed before the EOI command is issued or the CPU interrupt is enabled to prevent a second interrupt from occurring.

The priority cell diagram shows a conceptual circuit of the level sensitive and edge sensitive input circuitry of the 8259A. Be sure to note that the request latch is a transparent D type latch.

In both the edge and level triggered modes the IR inputs must remain high until after the falling edge of the first INTA. If the IR input goes low before this time a DEFAULT IR7 will occur when the CPU acknowledges the interrupt. This can be a useful safeguard for detecting interrupts caused by spurious noise glitches on the IR inputs. To implement this feature the IR7 routine is used for "clean up" simply executing a return instruction, thus ignoring the interrupt. If IR7 is needed for other purposes a default IR7 can still be detected by reading the ISR. A normal IR7 interrupt will set the corresponding ISR bit, a default IR7 won't. If a default IR7 routine occurs during a normal IR7 routine, however, the ISR will remain set. In this case it is necessary to keep track of whether or not the IR7 routine was previously entered. If another IR7 occurs it is a default.

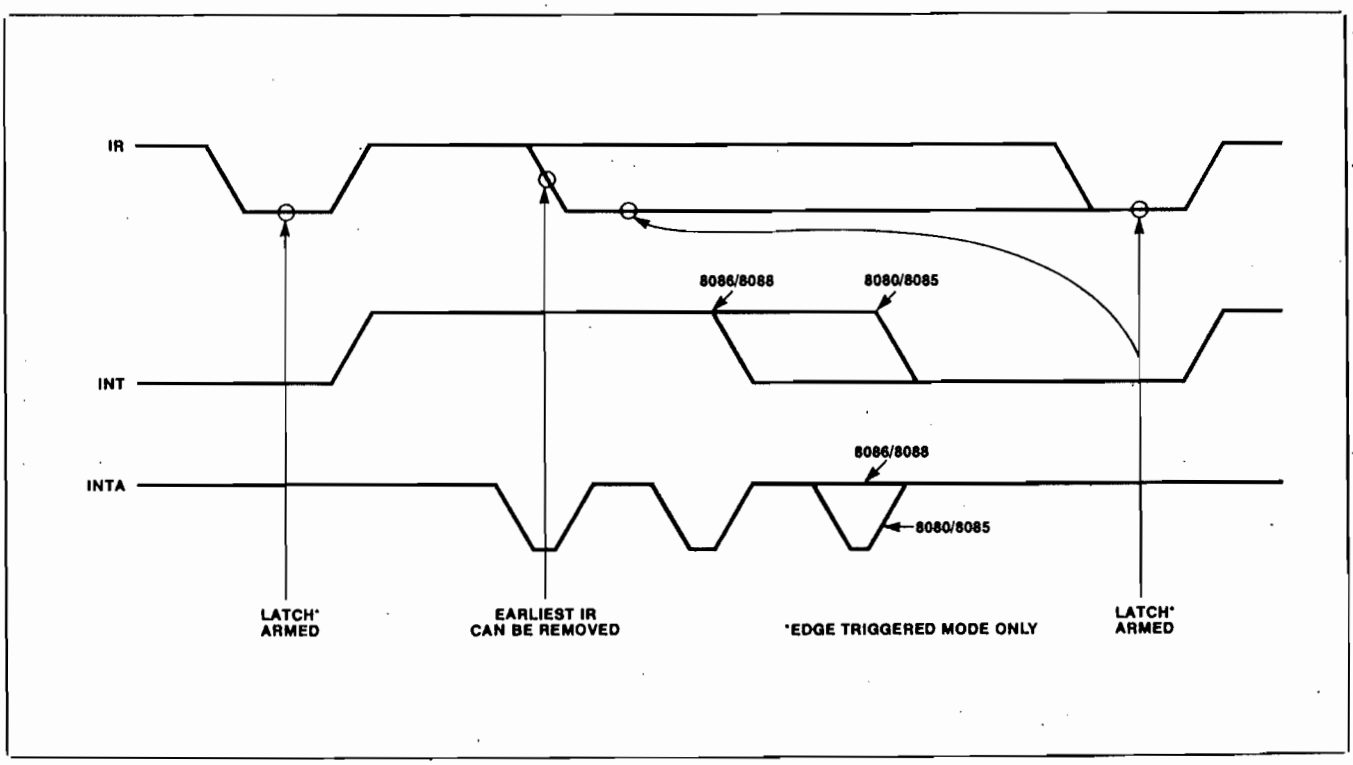


Figure 10. IR Triggering Timing Requirements

THE SPECIAL FULLY NESTED MODE

This mode will be used in the case of a big system where cascading is used, and the priority has to be conserved within each slave. In this case the fully nested mode will be programmed to the master (using ICW4). This mode is similar to the normal nested mode with the following exceptions:

a. When an interrupt request from a certain slave is in service this slave is not locked out from the master's priority logic and further interrupt requests from higher priority IR's within the slave will be recognized by the master and will initiate interrupts to the processor. (In the normal nested mode a slave is masked out when its request is in service and no higher requests from the same slave can be serviced.)

b. When exiting the Interrupt Service routine the software has to check whether the interrupt serviced was the only one from that slave. This is done by sending a non-specific End of Interrupt (EOI) command to the slave and then reading its In-Service register and checking for zero. If it is empty, a non-specific EOI can be sent to the master too. If not, no EOI should be sent.

BUFFERED MODE

When the 8259A is used in a large system where bus driving buffers are required on the data bus and the cascading mode is used, there exists the problem of enabling buffers.

The buffered mode will structure the 8259A to send an enable signal on $\overline{SP}/\overline{EN}$ to enable the buffers. In this mode, whenever the 8259A's data bus outputs are enabled, the $\overline{SP}/\overline{EN}$ output becomes active.

This modification forces the use of software programming to determine whether the 8259A is a master or a slave. Bit 3 in ICW4 programs the buffered mode, and bit 2 in ICW4 determines whether it is a master or a slave.

CASCADE MODE

The 8259A can be easily interconnected in a system of one master with up to eight slaves to handle up to 64 priority levels.

The master controls the slaves through the 3 line cascade bus. The cascade bus acts like chip selects to the slaves during the $\overline{INTA}$ sequence.

In a cascade configuration, the slave interrupt outputs are connected to the master interrupt request inputs. When a slave request line is activated and afterwards acknowledged, the master will enable the corresponding slave to release the device routine address during bytes 2 and 3 of INTA. (Byte 2 only for 8086/8088).

The cascade bus lines are normally low and will contain the slave address code from the trailing edge of the first INTA pulse to the trailing edge of the third pulse. Each 8259A in the system must follow a separate initialization sequence and can be programmed to work in a different mode. An EOI command must be issued twice: once for the master and once for the corresponding slave. An address decoder is required to activate the Chip Select (CS) input of each 8259A.

The cascade lines of the Master 8259A are activated only for slave inputs, non slave inputs leave the cascade line inactive (low).

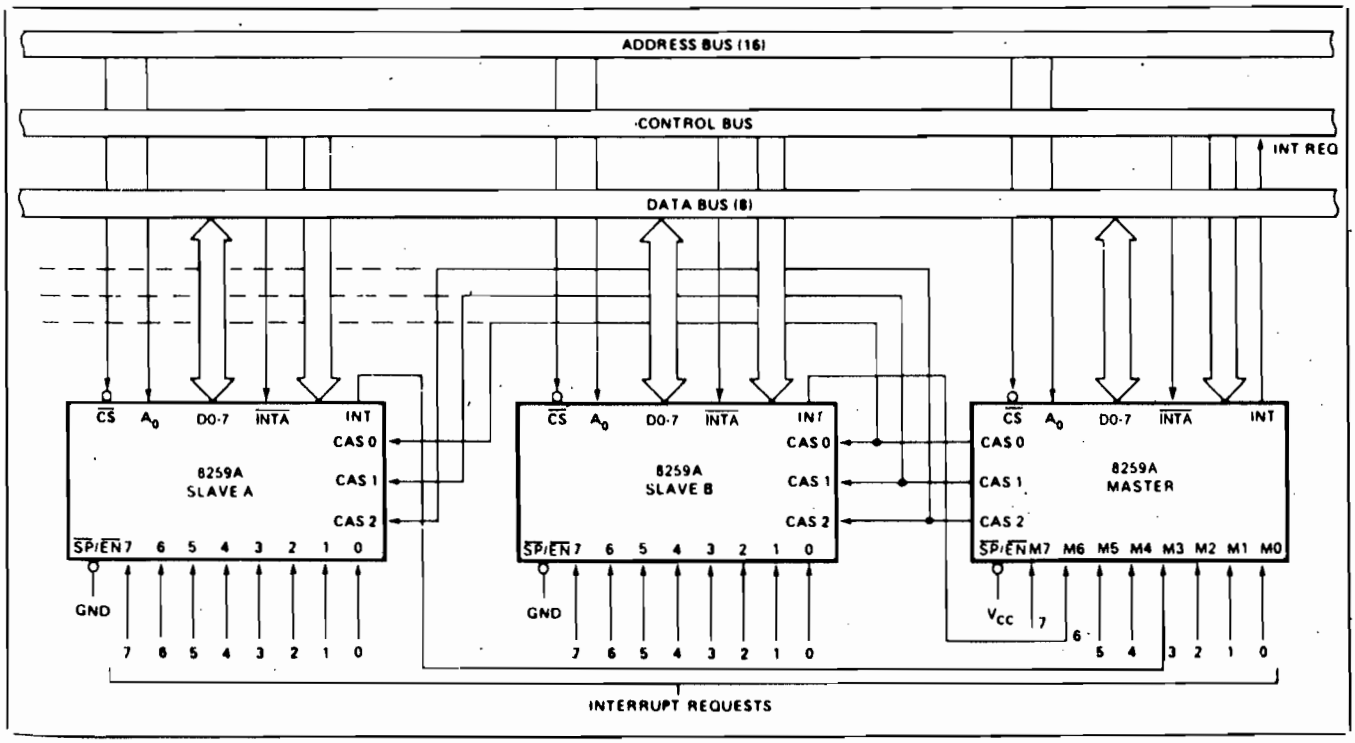


Figure 11. Cascading the 8259A

ABSOLUTE MAXIMUM RATINGS*

Ambient Temperature Under Bias ..... 0°C to 70°C
Storage Temperature ..... −65°C to +150°C
Voltage on Any Pin
with Respect to Ground ..... −0.5V to +7V
Power Dissipation ..... 1 Watt

NOTICE: Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied.

D.C. CHARACTERISTICS [T_A = 0°C to 70°C, V_{CC} = 5V ±5% (8259A-8), V_{CC} = 5V ±10% (8259A, 8259A-2)]

Symbol	Parameter	Min.	Max.	Units	Test Conditions
V_{IL}	Input Low Voltage	−0.5	0.8	V	
V_{IH}	Input High Voltage	2.0*	V_{CC} +0.5V	V	
V_{OL}	Output Low Voltage		0.45	V	I_{OL} = 2.2mA
V_{OH}	Output High Voltage	2.4		V	I_{OH} = −400μA
$V_{OH(INT)}$	Interrupt Output High Voltage	3.5		V	I_{OH} = −100μA
		2.4		V	I_{OH} = −400μA
I_{LI}	Input Load Current	−10	+10	μA	$0V \leq V_{IN} \leq V_{CC}$
I_{LOL}	Output Leakage Current	−10	+10	μA	$0.45V \leq V_{OUT} \leq V_{CC}$
I_{CC}	V_{CC} Supply Current		85	mA	
I_{LIR}	IR Input Load Current		−300	μA	V_{IN} = 0
			10	μA	$V_{IN} = V_{CC}$

*Note: For Extended Temperature EXPRESS V_{IH} = 2.3V.

CAPACITANCE (T_A = 25°C; V_{CC} = GND = 0V)

Symbol	Parameter	Min.	Typ.	Max.	Unit	Test Conditions
C_{IN}	Input Capacitance			10	pF	fc = 1 MHZ
$C_{I/O}$	I/O Capacitance			20	pF	Unmeasured pins returned to V_{SS}

A.C. CHARACTERISTICS [T_A = 0°C to 70°C, V_{CC} = 5V ±5% (8259A-8), V_{CC} = 5V ± 10% (8259A, 8259A-2)]

TIMING REQUIREMENTS

Symbol	Parameter	8259A-8		8259A		8259A-2		Units	Test Conditions
		Min.	Max.	Min.	Max.	Min.	Max.		
TAHRL	AO/$\overline{CS}$ Setup to $\overline{RD}$/$\overline{INTA}$↓	50		0		0		ns	
TRHAX	AO/$\overline{CS}$ Hold after $\overline{RD}$/$\overline{INTA}$↑	5		0		0		ns	
TRLRH	$\overline{RD}$ Pulse Width	420		235		160		ns	
TAHWL	AO/$\overline{CS}$ Setup to $\overline{WR}$↓	50		0		0		ns	
TWHAX	AO/$\overline{CS}$ Hold after $\overline{WR}$↑	20		0		0		ns	
TWLWH	$\overline{WR}$ Pulse Width	400		290		190		ns	
TDVWH	Data Setup to $\overline{WR}$↑	300		240		160		ns	
TWHDX	Data Hold after $\overline{WR}$↑	40		0		0		ns	
TJLJH	Interrupt Request Width (Low)	100		100		100		ns	See Note 1
TCVIAL	Cascade Setup to Second or Third $\overline{INTA}$↓ (Slave Only)	55		55		40		ns	
TRHRL	End of $\overline{RD}$ to next $\overline{RD}$ End of $\overline{INTA}$ to next $\overline{INTA}$ within an $\overline{INTA}$ sequence only	160		160		160		ns	
TWHWL	End of $\overline{WR}$ to next $\overline{WR}$	190		190		190		ns	

A.C. CHARACTERISTICS (Continued)

Symbol	Parameter	8259A-8		8259A		8259A-2		Units	Test Conditions
		Min.	Max.	Min.	Max.	Min.	Max.		
*TCHCL	End of Command to next Command (Not same command type)	500		500		500		ns	
	End of $\overline{INTA}$ sequence to next $\overline{INTA}$ sequence.								

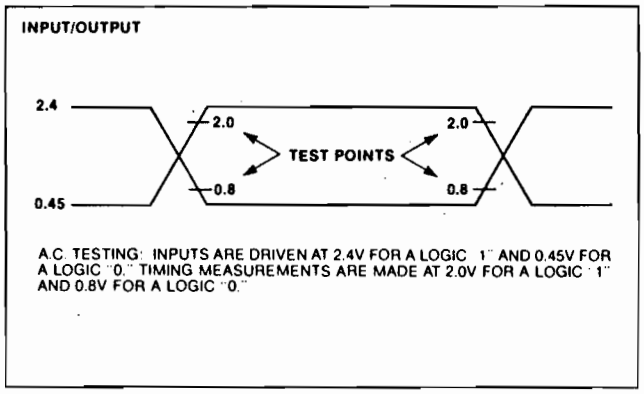
*Worst case timing for TCHCL in an actual microprocessor system is typically much greater than 500 ns (i.e. 8085A = 1.6μs, 8085A-2 = 1μs, 8086 = 1μs, 8086-2 = 625 ns)

NOTE: This is the low time required to clear the input latch in the edge triggered mode.

TIMING RESPONSES

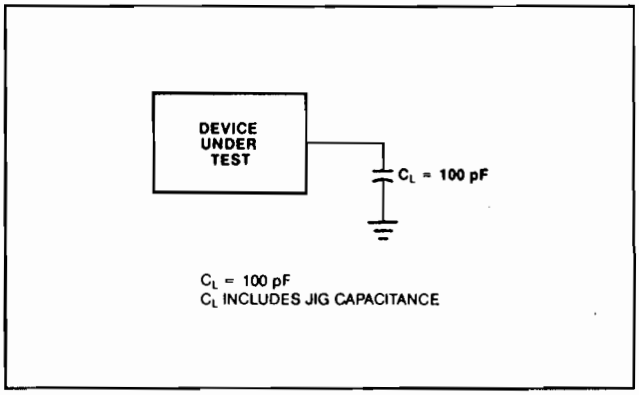
Symbol	Parameter	8259A-8		8259A		8259A-2		Units	Test Conditions
		Min.	Max.	Min.	Max.	Min.	Max.		
TRLDV	Data Valid from $\overline{RD}$/$\overline{INTA}$		300		200		120	ns	C of Data Bus = 100 pF
TRHDZ	Data Float after $\overline{RD}$/$\overline{INTA}$	10	200	10	100	10	85	ns	C of Data Bus Max text C = 100 pF Min. test C = 15 pF
TJHIH	Interrupt Output Delay		400		350		300	ns	
TIALCV	Cascade Valid from First $\overline{INTA}$ (Master Only)		565		565		360	ns	C_{INT} = 100 pF
TRLEL	Enable Active from $\overline{RD}$ or $\overline{INTA}$		160		125		100	ns	$C_{CASCADE}$ = 100 pF
TRHEH	Enable Inactive from $\overline{RD}$ or $\overline{INTA}$		325		150		150	ns	
TAHDV	Data Valid from Stable Address		350		200		200	ns	
TCVDV	Cascade Valid to Valid Data		300		300		200	ns	

A.C. TESTING INPUT, OUTPUT WAVEFORM



A.C. TESTING: INPUTS ARE DRIVEN AT 2.4V FOR A LOGIC "1" AND 0.45V FOR A LOGIC "0." TIMING MEASUREMENTS ARE MADE AT 2.0V FOR A LOGIC "1" AND 0.8V FOR A LOGIC "0."

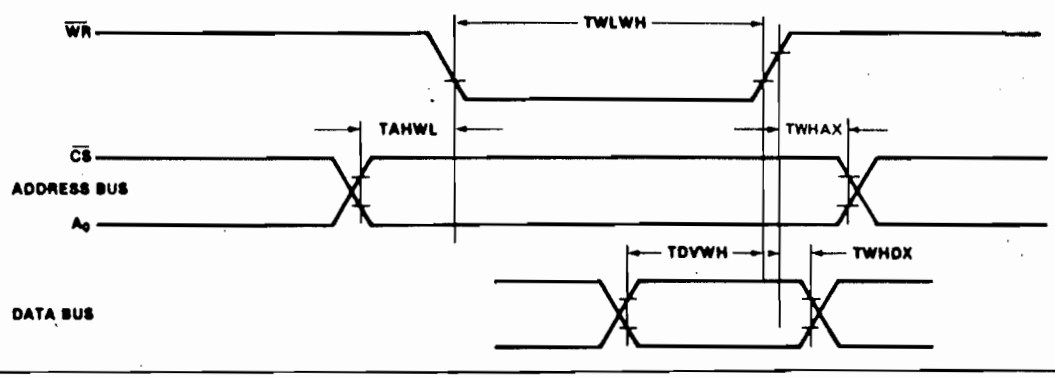
A.C. TESTING LOAD CIRCUIT



C_L = 100 pF
C_L INCLUDES JIG CAPACITANCE

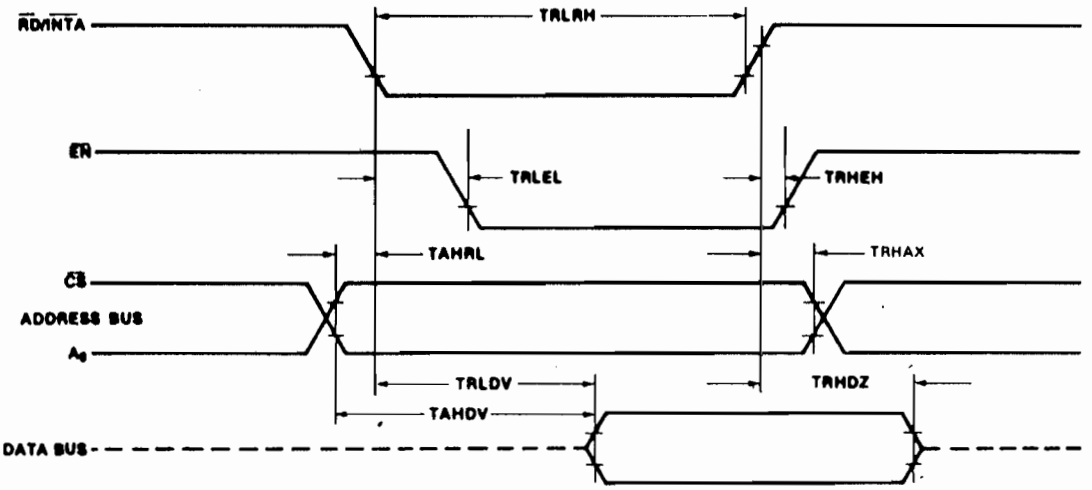
WAVEFORMS

WRITE

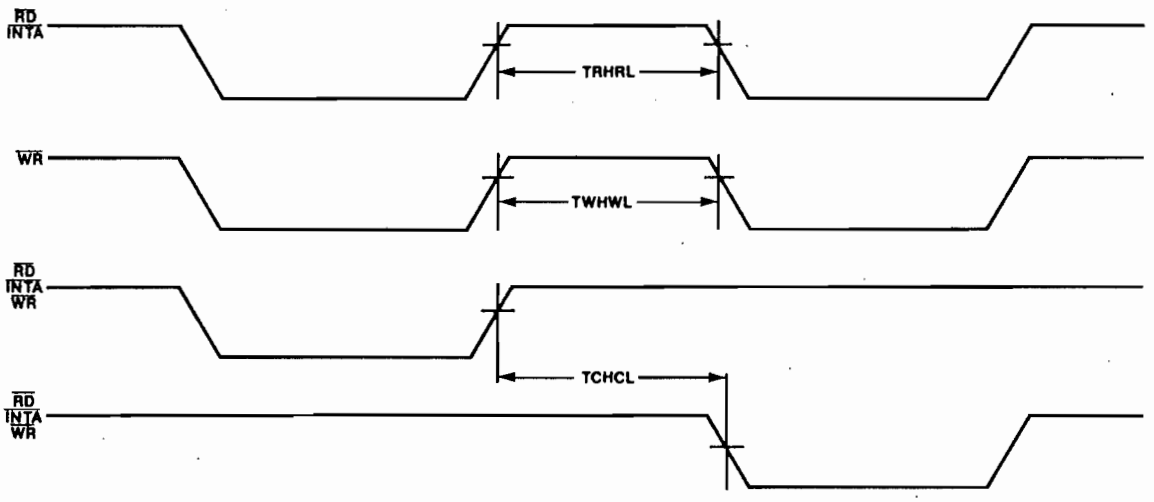


WAVEFORMS (Continued)

READ/INTA

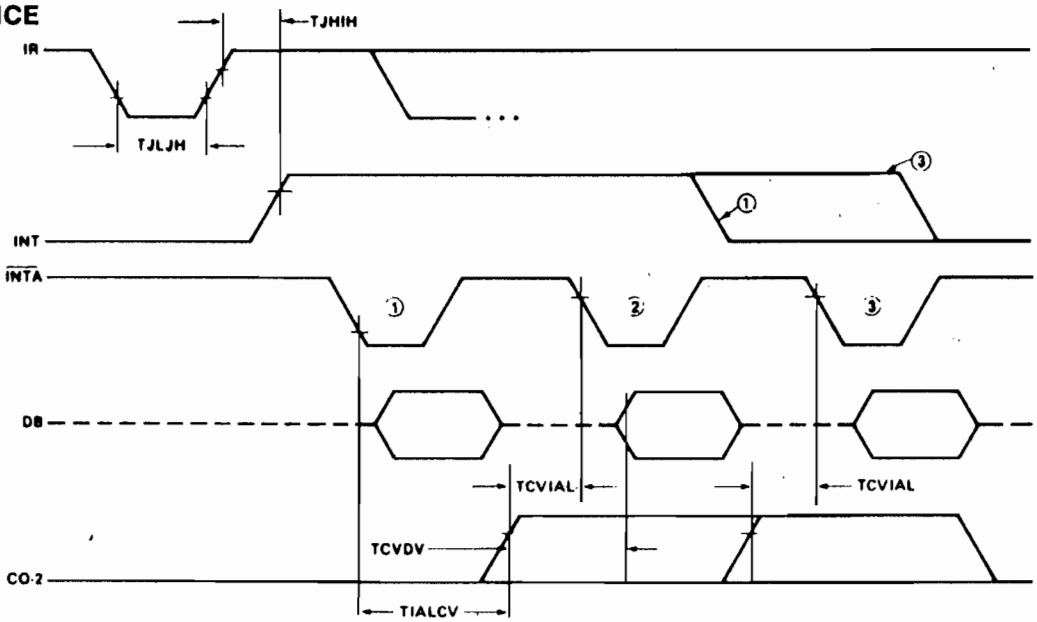


OTHER TIMING



WAVEFORMS (Continued)

INTA SEQUENCE



NOTES: Interrupt output must remain HIGH at least until leading edge of first INTA.
1. Cycle 1 in iAPX 86, iAPX 88 systems, the Data Bus is not active.

APPENDIX F

rutinebibliotek og procedurer

```
init_lst
turbo
```

```
PROGRAM INIT_LST;

(*************************************************************************

        INIT_LST klargjører skriveren, og startes automatisk
        dersom:

                - Det finnes en batch-fil med navn AUTOEXEC.BAT
                  som gjør ett kall til INIT_LST.

                - Det endres type på OP-SYS fila CONFIG.SYS til
                  f.eks. CONFIG.SAV


*************************************************************************)


BEGIN
  write(lst,#$1B'&k0S');          (*  tegnstørrelse på skriver  *)
  write(lst,#$1B'&l72P');         (*  side størrelse  A4        *)
  write(lst,#$1B'(s1B');          (*  hard skrift               *)
  write(lst,#12)                  (*  Form Feed                 *)
END.
```

```
PROGRAM LIST(INPUT,OUTPUT);

(*****************************************************************************

        LIST benyttes for å få en utskrift filer på skriveren, HP ThinkJet.
        Sidene får en heading med filnavn og sidenummer.
        Kopieringen kan brytes ved å taste 'q' eller 'Q'.

*****************************************************************************)

label 100;

var   fn      : string(.20.);
      f       : text;
      t       : char;
      spec    : set of char;
      i,side  : integer;


procedure heading;

BEGIN
   side := side + 1;
   write(lst,#$1B'&dD');            (* underline  *)
   writeln(lst,#$1B'&k1S');         (* store bokstaver *)
   write(lst,'FIL: ',fn);
   for i:= 1 to round(25-length(fn)) do write(lst,' ');
   writeln(lst,'SIDE : ',side:3);
   writeln(lst);
   write(lst,#$1B'&k0S');           (* normale bokstaver *)
   write(lst,#$1B'&d@');            (* abort underline *)
END;

(*'*)
```

```
begin
   side := 0;
   spec := [.'å','ø','æ','Å','Æ','Ø'.];
   write('Filnavn.....: ');
   readln(fn);
   for i:= 1 to length(fn) do
      fn[.i.] := upcase(fn[.i.]);
   assign(f,fn);
   (*$I-*)
   reset(f);
   (*$I+*)
   if ioresult <> 0
      then
         writeln('Får feil ved åpning av filen')
      else
         BEGIN
           heading;
           while not eof(f) do
           begin
              if keypressed
              then
                 BEGIN
                    read(kbd,t);
                    if t in [.'Q','q'.]
                    then goto 100;
                 END;
              read(f,t);
              IF T IN SPEC
                 THEN
                    CASE T OF
                       'Å'  : WRITE(LST,CHR(208));
                       'Ø'  : WRITE(LST,CHR(210));
                       'Æ'  : WRITE(LST,CHR(211));
                       'æ'  : WRITE(LST,CHR(215));
                       'ø'  : WRITE(LST,CHR(214));
                       'å'  : WRITE(LST,CHR(212));
                    END   (*  CASE  *)
                 else
                    WRITE(LST,T); (* på neste linje testes det på sideskift *)
              if t =''''
```

```
                then
                  BEGIN
                     read(f,t);
                     write(lst,t);
                     read(f,t);
                     write(lst,t);
                     write(lst,#12);   (*  FormFeed  *)
                     heading;
                  END;
            END;
100:        writeln(lst,#12);
        END;
END.
```

```
TYPE
     streng4  = string(.4.);
     streng5  = string(.5.);
     streng6  = string(.6.);
     streng8  = string(.8.);
     streng9  = string(.9.);
     streng12 = string(.12.);
     streng65 = string(.65.);
     streng80 = string(.80.);
     streng200= string(.200.);

     reg = record
             ax,bx,cx,dx,bp,si,ds,es,flag  : integer;
           END;

     topp_rec = record
                  topp_tid  : real;
                  toleranse : integer;
                  ion       : string(.9.);
                END;


VAR
     topp_tab                : array (.1..50.) of topp_rec;
```

```
PROCEDURE MELDING(TEKST : STRENG65);
(*************************************************************************

          MELDING brukes for å skrive all meldinger/feilmeldinger.
          Meldingen skrives på øverste linje.

*************************************************************************)
BEGIN
   write(^G);  gotoxy(5,1);
   write(tekst,', trykk RETURN');
   readln; gotoxy(1,1); clreol;
END;




Procedure LES_TALL( Var tall    : real;
                    Var status  : integer);
(*************************************************************************

          LES_TALL leser ett flyt-tall fra terminal. Både komma og punktum
          er lovlige desimaltegn. Blanke tegn fjernes.

          INNPARAMETRE :  ingen

          UTPARAMETRE  :  status,tall;

                          status      betydning
                          ------------------------------
                            1         ikke tastet noen ting
                            0         tastet godkjent tall
                           -1         uforståelig tall

*************************************************************************)
Var
     inndata,schrink      : string(.80.);
     i,kode               : integer;

BEGIN
   status := 0;
   read(inndata);
   schrink := '';
   for i:= 1 to length(inndata) do
      BEGIN
         if inndata(.i.) = ','
         then schrink := schrink + '.'
         else  if inndata(.i.) <> ' '
               then schrink := schrink + inndata(.i.);
      END;
   if length(schrink) = 0
   then status := 1
   else
      BEGIN
         Val(schrink,tall,kode);
         if kode <> 0
         then status := -1;
      END;
END;
(*'*)
```

```
PROCEDURE LES_REAL( Var tall   : real;
                    Var status : integer);

(*****************************************************************************

          LES_REAL leser ett flyt-tall fra terminal og gir
          eventuell feilmelding.

*****************************************************************************)

BEGIN
   les_tall(tall,status);
   if status = -1
   then
      melding('UFÅRSTÅELIG TALL');
END;




Procedure LES_INTEGER( Var tall, status  : integer);
(*****************************************************************************

        LES_INTEGER leser ett heltall fra terminal og gir
          eventuell feilmelding.

          INNPARAMETRE :  ingen

          UTPARAMETRE  :  status,tall


                          status        betydning
                          ------------------------------------
                            1           Det er ikke tastet noe
                            0           Det er tastet ett OK tall
                           -1           Uforståelig tall
                           -2           For stor tallverdi

*****************************************************************************)
Var
     temp      : real;

BEGIN
   les_tall(temp,status);    (* kan tastes inn eksponensiellt *)
   if status = 0
   then
      BEGIN
         if frac(temp) <> 0
         then status:= -1
         else  if abs(temp) >= maxint
               then status := -2;
      END;
   if status < 0
   then
      BEGIN
         case status of
            -1  : melding('Uforståelig heltall');
            -2  : melding('For stor tallverdi');
         end;
      END
   else  if status <> 1 then tall := round(temp);
END;
```

```
FUNCTION intern_dato : streng6;
(*****************************************************************************

          INTERN_DATO henter datoen fra RT-klokka og returnerer det
          på formen ååmmdd.

*****************************************************************************)

Var
     dag,mnd  : string(.2.);
     aar      : string(.4.);
     register : reg;


BEGIN
   register.ax := $2a shl 8;
   MsDos(register);
   str(register.cx,aar);
   str(register.dx mod 256,dag);
   str(register.dx shr 8,mnd);
   if length(mnd) = 1
   then
      mnd := '0' + mnd;
   if length(dag) = 1
   then
      dag := '0' + dag;
   intern_dato := aar(.3.) + aar(.4.) + mnd + dag;
END;



FUNCTION edit_dato( dato : streng6) : streng8;
(*****************************************************************************

          EDIT_DATO returnerer "dato" på formen "xx.xx.xx"

*****************************************************************************)
BEGIN
   edit_dato := copy(dato,1,2)+'.'+copy(dato,3,2)+'.'+copy(dato,5,2);
END;

(*'*)
```

```
FUNCTION intern_tid : streng4;
(*****************************************************************************

         INTERN_TID henter fram klokkeslettet fra RT-klokka og returnerer
         det på formen ttmm.

*****************************************************************************)

Var
     time,min    : string(.2.);
     register    : reg;
     ah,al       : byte;


BEGIN
   ah := $2c;
   register.ax := ah shl 8 + al;
   Intr($21,register);
   str(register.cx shr 8,time);
   str(register.cx mod 256,min);
   if length(time) = 1
   then
      time := '0' + time;
   if length(min) = 1
   then
      min := '0' + min;
   intern_tid := time + min;
END;




FUNCTION edit_tid( tid : streng4) : streng5;
(*****************************************************************************

         EDIT_TID returnerer "tid" på formen "tt:mm"

*****************************************************************************)
BEGIN
   edit_tid := copy(tid,1,2)+':'+copy(tid,3,2);
END;
```

```
PROCEDURE LINE ( X1,Y1,X2,Y2  : REAL);
(*************************************************************************

          Tegner en strek fra (X1,Y1) til (X2,Y2) på skjermen i fysiske
          skjermkoordinater. Endrer ikke aktuell penn-posisjon.

*************************************************************************)
VAR   X1S,X2S,Y1S,Y2S          : STRING(.3.);

BEGIN
   STR(ROUND(X1):1,X1S);
   STR(ROUND(X2):1,X2S);
   STR(ROUND(Y1):1,Y1S);
   STR(ROUND(Y2):1,Y2S);
   WRITE(#$1B'*pa'+X1S+','+Y1S+','+X2S+','+Y2S+'Z');
END;



PROCEDURE MOVETO(X,Y  : REAL );
(*************************************************************************

          Plasserer "pennen" på skjermen i posisjon (x,y).

*************************************************************************)
VAR   YS,XS           :STRING(.4.);

BEGIN
   STR(ROUND(X):1,XS);
   STR(ROUND(Y):1,YS);
   WRITE(#$1B'*pa'+XS+','+YS+'Z');
END;



PROCEDURE LINETO( X,Y : REAL);
(*************************************************************************

          Tegner en linje fra aktuell penn-posisjon til (x,y), ny
          penn-posisjon.

*************************************************************************)
VAR  XS,YS            : STRING(.4.);

BEGIN
   STR(ROUND(X):1,XS);
   STR(ROUND(Y):1,YS);
   WRITE(#$1B'*pb'+XS+','+YS+'Z');
END;



PROCEDURE INIT_GRAF;
(*************************************************************************

          Fåretar en grafisk reset og setter penmode til inverse video.

*************************************************************************)
BEGIN
   WRITE(#$1B'*wR');
   WRITE(#$1B'*m3A');
END;
(*'*)
```

```
PROCEDURE CLEAR_GRAF_MEM;
(*****************************************************************************

            Sletter alt fra grafisk RAM.

*****************************************************************************)
BEGIN
   WRITE(#$1B'*dA');
END;



PROCEDURE CURSOR_ON;
(*****************************************************************************

           Slår på kursoren

*****************************************************************************)
BEGIN
   write(#$1B'*dQ');
END;



PROCEDURE CURSOR_OFF;
(*****************************************************************************

           Slår av kursoren   (vil ikke virke)

*****************************************************************************)
BEGIN
   write(#$1B'*dR');
END;


PROCEDURE VIDEO_ON;
(*****************************************************************************

            Viser på skjermen det som er i alphanumerisk RAM.

*****************************************************************************)
BEGIN
   write(#$1B'*dE');
END;



PROCEDURE VIDEO_OFF;
(*****************************************************************************

           Viser ikke på skjermen det som er i alphanumerisk RAM

*****************************************************************************)
BEGIN
   write(#$1B'*dF');
END;
(*'*)
```

```
PROCEDURE S_DUMP( LINJE1,LINJE2 : integer);

(*****************************************************************************

          S_DUMP tar en kopi av grafisk RAM på HP-150 til HP ThinkJet-
          skriver. Det kopieres f.o.m pixel_linje LINJE1 t.o.m pixel-
          linje LINJE2. Grafisk RAM ligger i intervallet C0000H-C8000H,
          men kun en del av dette er direkte bilde, resten er atributter
          og programkode.
          Hver bytes må snus før den sendes til skriveren.

*****************************************************************************)

VAR    I,J,T,INN,UT : INTEGER;
       SKJERM       : ARRAY(.$0..$617F.) OF CHAR ABSOLUTE $C000:0000;


   FUNCTION POWER(A,B : INTEGER ) : INTEGER;
   VAR    M,I  : INTEGER;
   BEGIN
      IF B = 0
         THEN POWER := 1
         ELSE
            BEGIN
               M := A;
               FOR I:= 1 TO B-1 DO M:= M*A;
               POWER := M;
            END;
  END;


BEGIN    (*********************   S D U M P   ************************)
   WRITE(LST,#$1B'*r640S');                    (*  Oppløsning       *)
   WRITE(LST,#$1B'*rA');                       (*  Klargjør buffer  *)

   IF LINJE1 <   0 THEN LINJE1 :=   0;
   IF LINJE2 > 289 THEN LINJE2 := 289;

   FOR I:= LINJE1 TO LINJE2 DO
      BEGIN
         WRITE(LST,#$1B'*b64W');              (* 64 grafiske bytes *)
         for j:= 0 to 63 do
            BEGIN
               INN := ORD(SKJERM(.64*I + J.));     (* Hent 8 pixel *)
               UT  := 0;                          (* start snu byte *)
               FOR T:= 0 TO 7 DO
                  BEGIN
                     IF INN MOD 2 <> 0
                        THEN UT := UT + POWER(2,7-T);
                     INN := INN DIV 2;
                  END;                                (* ferdig snudd *)
               WRITE(LST,CHR(UT));              (* sendes til skriver *)
            END;
      END;
   WRITE(LST,#$1B'*rB');                             (* Tøm buffer  *)
END;
```

```
PROCEDURE  TEXT_SIZE( SIZE : integer);

(*************************************************************************

           TEXT_SIZE initialiserer skriveren med tekst_størrelse.
           1 er små bokstaver 4 er store. HP ThinkJet.

*************************************************************************)


VAR    streng     : string(.1.);


BEGIN
   IF size IN (.1..4.)
     THEN
        BEGIN
           CASE SIZE OF
                   1      : write(lst,#$1B'&k2S');
                   2      : write(lst,#$1B'&k0S');
                   3      : write(lst,#$1B'&k3S');
                   4      : write(lst,#$1B'&k1S');
            end;    (*  case  *)
        END;
END;




PROCEDURE  LST_ULINE_ON;
(*************************************************************************

           Sette på understreking på skriver, HP ThinkJet.

*************************************************************************)

BEGIN
   WRITE(LST,#$1B'&dD');
END;



PROCEDURE LST_ULINE_OFF;
(*************************************************************************

           Slår av understreking på skriver, HP ThinkJet.

*************************************************************************)

BEGIN
   WRITE(LST,#$1B'&d@')
END;


(*'*)
```

```
PROCEDURE LST_PAGE_SIZE( ANT_LINJER : INTEGER );
(*************************************************************************

          Setter sidestørrelsen på skriveren, HP ThinkJet, til
          ANT_LINJER. For å få passe FormFeed til HP ThinkJet-papir
          må sidestørrelsen settes til 72. Default = 66.

*************************************************************************)

VAR  STRENG : STRING(.2.);

BEGIN
   IF ANT_LINJER IN (.10..99.)
      THEN
         BEGIN
            STR(ANT_LINJER,STRENG);
            WRITE(LST,#$1B'&l'+streng+'P');
         END;
END;



PROCEDURE LST_TEXT_HARD;
(*************************************************************************

          Setter skriveren til å bruke mørkere skrift, HP ThinkJet.

*************************************************************************)

BEGIN
   WRITE(LST,#$1B'(s1B');
END;



PROCEDURE LST_TEXT_SOFT;
(*************************************************************************

           Setter skriveren tilbake til normal (lys) skrift, HP ThinkJet.

*************************************************************************)

BEGIN
   WRITE(LST,#$1B'(sØB');
END;


(*'*)
```

```
FUNCTION lst_streng( tekst : streng80) : streng80;

(*****************************************************************************

        LST_STRENG returnerer en tekststreng som kan skrives ut på
        skriveren. Denne funksjonen må benyttes dersom teksten inneholder
        æ,ø,å,Æ,Ø eller Å.

*****************************************************************************)

Var
     i                  : integer;
     norsk              : set of char;


BEGIN
   norsk := (.'æ','ø','å','Æ','Ø','Å'.);
   for i:= 1 to length(tekst) do
   BEGIN
      if tekst(.i.) in norsk
      then
         case tekst(.i.) of
              'æ'   : tekst(.i.) := chr(215);
              'ø'   : tekst(.i.) := chr(214);
              'å'   : tekst(.i.) := chr(212);
              'Æ'   : tekst(.i.) := chr(211);
              'Ø'   : tekst(.i.) := chr(210);
              'Å'   : tekst(.i.) := chr(208);
         END;
   END;
   lst_streng := tekst;
END;
```

```
FUNCTION FORMAT_KONS( kode : integer;
                      kons : real):streng12;

(*************************************************************************

        FORMAT_KONS returnerer en tekst-streng som inneholder konsen-
        trasjon og benevning.

        INNPARAMETRE   kode = format-type,
                       kons = konsentrasjon målt i ppm.

        RETUPARAMETER  format_kons = formatert tekst-streng


        Følgende benevninger brukes:

                       ppt = 1e-6 ppm
                       ppb = 1e-3 ppm
                       ppm
                        %  = 1e+4 ppm



        FORMAT
                         kode = 0          kode <> 0
                        ---------------------------------
                        " x.xxbbb"       " x.xxx bbb"
                        " xx.xbbb"       " xx.xx bbb"
                        " xxx bbb"       " xxx.x bbb"


                         der bbb er benevning.

*************************************************************************)


Var
     benevning              : string(.3.);
     temp                   : string(.9.);
     mellom                 : string(.1.);



(*'*)
```

```
BEGIN
   if kode = 0
   then
      BEGIN
         mellom := '';
         if abs(kons) >= 0.9995E3
         then
            BEGIN
               benevning := ' % ';
               kons := kons/1E4;
            END
         else
         if abs(kons) >= 0.9995
         then
            BEGIN
               benevning := 'ppm';
            END
         else
         if abs(kons) >= 0.9995E-3
         then
            BEGIN
               benevning := 'ppb';
               kons := kons*1E3;
            END
         else
            BEGIN
               benevning := 'ppt';
               kons := kons*1E6;
            END;

         if abs(kons) >= 99.95
            then
               BEGIN
                  str(kons:4:0,temp);
                  mellom := ' ';
               END
            else
            if abs(kons) >= 9.995
                then str(kons:5:1,temp)
                else str(kons:5:2,temp);
      END
(*'*)
```

```
   else                                    (*  kode <> 0  *)
      BEGIN
         mellom := ' ';
         if abs(kons) >= 0.99995E3
         then
            BEGIN
               benevning := ' % ';
               kons := kons/1E4;
            END
         else
         if abs(kons) >= 0.99995
         then
            BEGIN
               benevning := 'ppm';
            END
         else
         if abs(kons) >= 0.99995E-3
         then
            BEGIN
               benevning := 'ppb';
               kons := kons*1E3;
            END
         else
            BEGIN
               benevning := 'ppt';
               kons := kons*1E6;
            END;

         if abs(kons) >= 99.995
            then str(kons:6:1,temp)
            else
            if abs(kons) >= 9.9995
               then str(kons:6:2,temp)
               else str(kons:6:3,temp);
      END;

   FORMAT_KONS := TEMP + MELLOM + BENEVNING;
END;
```

```
PROCEDURE TOPNAV;

(*****************************************************************************

          DOKUMENTASJON se kravspesifikasjon kap. 2.

*****************************************************************************)

CONST
     toppfil : string(.16.) = 'A:IONENAVN.TAB';
     L1: string(.26.) = '                    TOPP TID';
     L2: string(.40.) = 'TOPP TID           TOL. %              ION';
     L3: string(.46.) = '----------------------------------------------';



VAR
     f                                    : file of topp_rec;
     temp                                 : topp_rec;
     ant_lest,status,teller,linje         : integer;
     skriver,ferdig                       : boolean;


PROCEDURE topnav_skjerm;

BEGIN
   clrscr;
   gotoxy(7,3);
   write('Under kolonne for topptid er følgende kommandoer');
   write(' tilgjengelig :');
   gotoxy(7,5);
   write(' Ø = Linjen slettes');
   gotoxy(7,6);
   write('-1 = Avslutt, topptabellen lagres');
   gotoxy(7,7);
   write('-2 = Avslutt, topptabellen lagres, og listes ut på skriver');
   gotoxy(17,9);
   write(L1);
   gotoxy(17,10);
   write(L2);
   gotoxy(17,11);
   write(L3);
END;


(*'*)
```

```
PROCEDURE les_inn_topper;

BEGIN
   ant_lest := 0;
   assign(f,toppfil);
   (*$I-*)
   reset(f);
   (*$I+*)
   if ioresult = 0
   then
      while not eof(f) do
      BEGIN
         ant_lest := ant_lest + 1;
         read(f,topp_tab(.ant_lest.));
      END;
   close(f);
END;


(*'*)
```

```
PROCEDURE endre_linje;

BEGIN
   if teller > ant_lest
   then
      BEGIN
         ant_lest := ant_lest + 1;
         with topp_tab(.teller.) do
            BEGIN
               topp_tid  := 0;
               toleranse := 5;
               ion := '';
            END;
      END;
   gotoxy(10,linje);
   write(temp.topp_tid:13:2,' ':14);
   topp_tab(.teller.).topp_tid := temp.topp_tid;
   if teller = ant_lest
   then
      BEGIN
         gotoxy(37,linje);
         write('5');
      END;
   repeat
      gotoxy(37,linje);
      les_integer(temp.toleranse,status);
   until status >= 0;
   if status = 0
   then
      topp_tab(.teller.).toleranse := abs(temp.toleranse);
   gotoxy(35,linje);
   write(topp_tab(.teller.).toleranse:5,' ':14);
   repeat
      gotoxy(54,linje);
      buflen := 9;
      read(temp.ion);
      if (length(temp.ion) = 0)  and (length(topp_tab(.teller.).ion) = 0)
      then
         melding('Ione navn må oppgis');
   until (length(temp.ion) > 0 ) or (length(topp_tab(.teller.).ion) > 0);
   if length(temp.ion) > 0
   then
      topp_tab(.teller.).ion := temp.ion;
   gotoxy(54,linje);
   write(topp_tab(.teller.).ion);
   clreol;
 END;
(*'*)
```

```
PROCEDURE kommando;

Var          i        : integer;

BEGIN
    CASE round(temp.topp_tid) of
         -1    :   ferdig  := true;
         -2    :   skriver := true;
          0    :   BEGIN
                      for i:= teller to ant_lest -1 do
                         topp_tab(.i.) := topp_tab(.i+1.);
                      if teller <= ant_lest
                      then
                         ant_lest := ant_lest - 1;
                      teller := teller - 1;
                      gotoxy(6,linje);
                      clreol;
                      write('slettet --->');
                      clreol;
                   END;
    END;     (*  CASE  *)
END;
(*$*)
```

```
PROCEDURE utfor_endring;

BEGIN
   gotoxy(1,11);
   teller := 0;
   while (teller < ant_lest) and (teller < 13 ) do
   BEGIN
      teller := teller + 1;
      with topp_tab(.teller.) do
      BEGIN
         writeln;
         write(topp_tid:22:2,toleranse:17,' ':14,ion);
      END;
   END;
   linje := 11;
   teller := 0;
   ferdig := false;
   skriver := false;
(*$*)
```

```
   while (teller < 50) and not (ferdig or skriver) do
   BEGIN
      teller := teller + 1;
      if linje < 24
      then linje := linje + 1
      else
         BEGIN
            gotoxy(1,12);
            DelLine;
            gotoxy(1,linje);
            if teller <= ant_lest
            then
               with topp_tab(.teller.) do
                  write(topp_tid:22:2,toleranse:17,' ':14,ion);
         END;
      repeat
         gotoxy(18,linje);
         les_real(temp.topp_tid,status);
      until (status >= 0) and (temp.topp_tid > -2.49);
      if status = 0
      then
         BEGIN
            if temp.topp_tid > 0
            then
               endre_linje
            else
               kommando;
         END
      else
         BEGIN
            if teller > ant_lest
            then
               BEGIN
                  teller := teller - 1;
                  gotoxy(1,linje);
                  clreol;
               END
            else
               BEGIN
                  gotoxy(10,linje);
                  write(topp_tab(.teller.).topp_tid:13:2,' ':14);
               END;
         END;
   END;
   if (ant_lest = 50) and not ( skriver or ferdig )
   then
      melding('Det er ikke plass til flere toppnavn');
   if (skriver or ferdig) and (ant_lest > 0)
   then
      BEGIN
         gotoxy(1,1);
         write('* LAGRES *');
      END;
END;
(*'*)
```

```
PROCEDURE sorter_topper;

Var i,j   : integer;

BEGIN
   for i:= 1 to ant_lest - 1 do
      for j:= i+1 to ant_lest do
         if topp_tab(.i.).topp_tid > topp_tab(.j.).topp_tid
         then
            BEGIN
               temp := topp_tab(.i.);
               topp_tab(.i.) := topp_tab(.j.);
               topp_tab(.j.) := temp;
            END;
END;



PROCEDURE skriv_til_fil;

Var
     i,j     : integer;

BEGIN
   rewrite(f);
   for i:= 1 to ant_lest do
      write(f,topp_tab(.i.));
   close(f);
END;



PROCEDURE TOPPLST;

Var      i    : integer;

BEGIN
   writeln(lst); writeln(lst);
   write(lst,'          R E G I S T R E R T E   T O P P N A V N');
   writeln(lst,'          Dato  ',edit_dato(intern_dato));
   for i:= 1 to 8 do write(lst,'###########');
   writeln(lst); writeln(lst); writeln(lst);
   writeln(lst,'                ',L1);
   writeln(lst,'                ',L2);
   writeln(lst,'                ',L3);
   for i:= 1 to ant_lest do
       with topp_tab(.i.) do
          writeln(lst,topp_tid:22:2,toleranse:16,' ':14,lst_streng(ion));
   write(lst,#12);
END;



BEGIN                (*     T O P N A V      *)
   video_off;
   topnav_skjerm;
   les_inn_topper;
   video_on;
   utfor_endring;
   sorter_topper;
   skriv_til_fil;
   if skriver and (ant_lest > 0)
   then
      TOPPLST;
```

```
PROCEDURE REGOPDR;

(*****************************************************************************

          DOKUMENTASJON se kravspesifikasjon kap. 2.

*****************************************************************************)

CONST
     toppfil_navn : string(.16.) = 'A:IONENAVN.TAB';
     oppdr_fil_navn: string(.16.) = 'OPPDRAG.TAB';

Var
     tegn                                   : char;
     topper                                 : file of topp_rec;
     oppdr_fil_ny,oppdr_fil_gml             : text;
     ant_reg_ion,lest_tol                   : integer;
     ion_tab                                : array (.1..51.) of string(.9.);
     opdr_nr                                : string(.6.);
     opdr_giv                               : string(.45.);
     prosjekt_nr                            : string(.10.);
     prove_type                             : string(.45.);
     ant_ion,status,
     linje,i,j,k,l,m                        : integer;
     tillegg,bryt                           : boolean;
     lest_tid                               : real;

PROCEDURE regopdr_skjerm;

BEGIN
   clrscr;
   gotoxy(12,3);
   write('R E G I S T R E R I N G   A V   N Y T T   O P P D R A G');
   gotoxy(12,6);
   write('Oppdrags nummer .:');
   gotoxy(12,7);
   write('Oppdragsgiver ...:');
   gotoxy(12,9);
   write('Prosjekt nummer .:');
   gotoxy(12,11);
   write('Prøvetype .......:');
   gotoxy(12,13);
   write('Antall ionenavn .:');
   gotoxy(43,15);
   write('TOPP TID');
   gotoxy(12,16);
   write('ION          TOPP TID          TOL. %');
   gotoxy(12,17);
   write('-----------------------------------------');
END;

(*'*)
```

```
PROCEDURE les_ionetopp_navn;

BEGIN
   ant_reg_ion := 0;
   assign(topper,toppfil_navn);
   (*$I-*)
   reset(topper);
   (*$I+*)
   if ioresult = 0
      then
         while not eof(topper) do
         BEGIN
            ant_reg_ion := ant_reg_ion + 1;
            read(topper,topp_tab(.ant_reg_ion.));
         END;
   close(topper);
END;



FUNCTION topptiden(ione_navnet : streng9) : real;

Var   p         : integer;
      temp      : real;

BEGIN
   p := 1;
   while topp_tab(.p.).ion <> ione_navnet do
      p := p + 1;
   topptiden := topp_tab(.p.).topp_tid;
END;



(*'*)
```

```
FUNCTION duplikat(lest_oppdr : streng6) : boolean;

Var
     f  : text;
     nr : string(.6.);


BEGIN
   duplikat := false;
   nr := '';
   assign(f,oppdr_fil_navn);
   (*$I-*)
   reset(f);
   (*$I+*)
   if ioresult = 0
   then
      BEGIN
         while (not eof(f)) and
               (copy(nr,1,length(lest_oppdr)) <> lest_oppdr) do
            readln(f,nr);
         if copy(nr,1,length(lest_oppdr)) = lest_oppdr
         then
            duplikat := true;
      END;
   close(f);
END;

(*'*)
```

```
PROCEDURE sok_etter_topp_tid;

BEGIN
   j := 1;
   while (ion_tab(.i.) <> topp_tab(.j.).ion) and (j < ant_reg_ion) do
      j := j + 1;
   if (ion_tab(.i.) = topp_tab(.j.).ion) and (ant_reg_ion > 0)
   then
      with topp_tab(.j.) do
      BEGIN
         gotoxy(20,linje);
         write(topp_tid:11:2,toleranse:18);
      END
   else
      BEGIN
         if ant_reg_ion = 50
         then
            BEGIN
               melding('Tabellen er full, toppnavn må slettes');
               bryt := true;
            END
         else
            BEGIN
               repeat
                  gotoxy(25,linje);
                  les_real(lest_tid,status);
                  gotoxy(20,linje);
                  clreol;
               until status = 0;
               if lest_tid = -1
               then
                  bryt := true
               else
(*'*)
```

```
                (* gjør plass til ny topptid og legg i tabell *)
                BEGIN
                   tillegg := true;
                   write(lest_tid:11:2);
                   j := 1;
                   while (topp_tab(.j.).topp_tid < lest_tid)
                      and (j <= ant_reg_ion) do
                      j := j + 1;
                   for k:= ant_reg_ion downto j do
                      topp_tab(.k+1.) := topp_tab(.k.);
                   ant_reg_ion := ant_reg_ion + 1;
                   topp_tab(.j.).ion := ion_tab(.i.);
                   topp_tab(.j.).topp_tid := lest_tid;
                   gotoxy(46,linje);
                   write('5');
                   repeat
                      gotoxy(46,linje);
                      les_integer(lest_tol,status);
                   until status >= 0;
                   if (status = 0) and (lest_tol = -1)
                   then
                      bryt := true
                   else
                   BEGIN
                      if status = 0
                         then
                            topp_tab(.j.).toleranse := abs(lest_tol)
                         else
                            topp_tab(.j.).toleranse := 5;
                      gotoxy(44,linje);
                      write(topp_tab(.j.).toleranse:5);
                      clreol;
                   END;
                END;
             END;
       END;
END;

(*'*)
```

```
PROCEDURE les_felter;

Var
     Flere_like : boolean;


BEGIN
   repeat
      repeat
         gotoxy(31,6);
         clreol;
         buflen:= 6;
         read(opdr_nr);
         if length(opdr_nr) <= 3
            then write(^G);
      until (length(opdr_nr) > 3) or (opdr_nr = '-1');
      Flere_like := duplikat(opdr_nr);
      if flere_like
      then
         melding('Oppdraget er registrert');
   until not Flere_like;
   if opdr_nr = '-1' then bryt := true else BEGIN
   gotoxy(31,7);
   buflen:= 45;
   read(opdr_giv);
   if opdr_giv = '-1' then bryt := true else BEGIN
   gotoxy(31,9);
   buflen:= 10;
   read(prosjekt_nr);
   if prosjekt_nr = '-1' then bryt := true else BEGIN
   gotoxy(31,11);
   buflen:= 45;
   read(prove_type);
   if prove_type = '-1' then bryt := true else BEGIN
   repeat
      gotoxy(31,13);
      les_integer(ant_ion,status);
   until (status = 0) and ((ant_ion > 0) or (ant_ion = -1));
   if ant_ion = -1 then bryt := true else BEGIN
   gotoxy(31,13);
   write(ant_ion:3);
   clreol;
(*'*)
```

```
   tillegg := false;
   linje := 17;
   i := 1;
   while (i <= ant_ion) and not bryt do
   BEGIN
      if linje < 24
      then linje := linje + 1
      else
         BEGIN
            gotoxy(1,19);
            DelLine;
         END;
      repeat
         gotoxy(12,linje);
         clreol;
         buflen:= 9;
         read(ion_tab(.i.));
      until length(ion_tab(.i.)) > 0;
      if ion_tab(.i.) = '-1'
      then
         bryt := true
      else
         BEGIN
            k := 1;
            while (k < i) and (ion_tab(.i.) <> ion_tab(.k.)) do
               k := k + 1;
            if k <> i
            then
               BEGIN
                  gotoxy(1,linje);
                  clreol;
                  melding('Ionet er registrert');
                  linje := linje - 1;
               END
            else
               BEGIN
                  sok_etter_topp_tid;
                  i := i + 1;
               END;
         END;
   END;
   END; END; END; END; END;
END;

(*'*)
```

```
BEGIN                          (*    R E G O P D R    *)
   video_off;
   bryt := false;
   regopdr_skjerm;
   les_ionetopp_navn;
   video_on;
   les_felter;
   if not bryt
   then
      BEGIN
         gotoxy(1,1);
         write('Oppdraget lagres');
         if tillegg
         then
            BEGIN
               rewrite(topper);
               for k:= 1 to ant_reg_ion do
                  write(topper,topp_tab(.k.));
               close(topper);
            END;
         for i:= 1 to ant_ion -1 do
            for j:= i+1 to ant_ion do
               if topptiden(ion_tab(.i.)) > topptiden(ion_tab(.j.))
               then
                  BEGIN
                     ion_tab(.51.) := ion_tab(.i.);
                     ion_tab(.i.) := ion_tab(.j.);
                     ion_tab(.j.) := ion_tab(.51.);
                  END;
(*'*)
```

```
        assign(oppdr_fil_ny,'OPPDRAG.TMP');
        rewrite(oppdr_fil_ny);
        assign(oppdr_fil_gml,oppdr_fil_navn);
        (*$I-*)
        reset(oppdr_fil_gml);
        (*$I+*)
        if ioresult = 0
        then
           BEGIN
              while not eof(oppdr_fil_gml) do
                 BEGIN
                    read(oppdr_fil_gml,tegn);
                    write(oppdr_fil_ny,tegn);
                 END;
              close(oppdr_fil_gml);
              erase(oppdr_fil_gml);
           END;
        write(oppdr_fil_ny,opdr_nr);
        if length(opdr_nr) < 6
           then write(oppdr_fil_ny,' ':6-length(opdr_nr));
        write(oppdr_fil_ny,opdr_giv);
        if length(opdr_giv) < 45
           then write(oppdr_fil_ny,' ':45-length(opdr_giv));
        write(oppdr_fil_ny,prosjekt_nr);
        if length(prosjekt_nr) < 10
           then write(oppdr_fil_ny,' ':10-length(prosjekt_nr));
        write(oppdr_fil_ny,prove_type);
        if length(prove_type) < 45
           then write(oppdr_fil_ny,' ':45-length(prove_type));
        for i:= 1 to ant_ion do
        BEGIN
           write(oppdr_fil_ny,ion_tab(.i.));
           if length(ion_tab(.i.)) < 9
              then write(oppdr_fil_ny,' ':9- length(ion_tab(.i.)));
        END;
        writeln(oppdr_fil_ny);
        close(oppdr_fil_ny);
        rename(oppdr_fil_ny,oppdr_fil_navn);
     END;
END;
```

```
PROCEDURE OPDRLST;

(*****************************************************************************

          DOKUMENTASJON se kravspesifikasjon kap. 4.

*****************************************************************************)

CONST
     oppdrags_fil     : string(.16.)  = 'OPPDRAG.TAB';

Var
     fil                    : text;
     opdr_nr                : string(.6.);
     ion                    : string(.9.);
     opdr_giv,pr_typ        : string(.45.);
     pro_nr                 : string(.10.);
     side,ant,ant_ion,i     : integer;


PROCEDURE overskrift;

BEGIN
   side := side + 1;
   ant := 0;
   if side <> 1
   then
      write(lst,#12);
   writeln(lst);
   writeln(lst,' ':63,'Side',side:10);
   write(lst,'                          O P P D R A G S - L I S T E');
   writeln(lst,'Dato  ':22,edit_dato(intern_dato));
   for i:= 1 to 8 do write(lst,'############');
   writeln(lst);
   writeln(lst);
END;

(*'*)
```

```
BEGIN                          (*   O P D R L S T   *)
   assign(fil,oppdrags_fil);
   (*$I-*)
   reset(fil);
   (*$I+*)
   if ioresult <> 0
   then
      melding('Det er ingen registrerte oppdrag på disketten')
   else
   BEGIN
      ant := 0;
      side := 0;
      while not eof(fil) do
      BEGIN
         if ant mod 11 = 0
         then
            overskrift;
         ant := ant + 1;
         read(fil,opdr_nr,opdr_giv,pro_nr,pr_typ);
         writeln(lst,'Oppdrag':11,opdr_nr:9,lst_streng(opdr_giv):53);
         write(lst,'Prosjekt':12,lst_streng(pro_nr):12);
         write(lst,lst_streng(pr_typ):49);
         ant_ion := 0;
         repeat
            if ant_ion mod 7 = 0
            then
               BEGIN
                  writeln(lst);
                  write(lst,'    ');
               END;
            ant_ion := ant_ion + 1;
            read(fil,ion);
            write(lst,lst_streng(ion):10);
         until eoln(fil);
         readln(fil);
         writeln(lst);
         writeln(lst);
         writeln(lst);
      END;
      close(fil);
      write(lst,#12);
   END;
END;
```

```
PROGRAM PCION(input,output);

(*****************************************************************************

        PCION er roten til programmet. Her presenteres hoved-menyen.

*****************************************************************************)

(*$I TYPE_VAR.INK *)   (* noen globale typer og variable *)

(*$I TID_DATO.LIB *)
(*$I LES_TALL.LIB *)
(*$I LST_FUNK.LIB *)
(*$I     GRAF.LIB *)
(*$I   TOPNAV.PAS *)
(*$I  REGOPDR.PAS *)
(*$I  OPDRLST.PAS *)

Var
     hm_valg                          : char;
     hoved_meny                       : set of char;


BEGIN
   hoved_meny := (.'B','M','T','R','O','K','A','S','L','P','I','Q'.);
   text_size(2);                      (* init skriver  *)
   lst_page_size(72);
   lst_text_hard;
   hm_valg := '-';

(* oppdat_integrator;  *)            (* laster integrator med metodene *)
(*'*)
```

```
   REPEAT
      if upcase(hm_valg) <> 'P'
      then
      BEGIN
         clrscr;
         video_off;
         gotoxy(11,4);
         write('                        H O V E D   M E N Y');
         gotoxy(11,7);
      write('Metode manipulering ..................................... <M>');
         gotoxy(11,8);
      write('Toppnavn, lagring/endring/sletting ...................... <T>');
         gotoxy(11,10);
      write('Registrering av nytt oppdrag ............................ <R>');
         gotoxy(11,11);
      write('Oppstart av ionekromatograf ............................. <O>');
         gotoxy(11,12);
      write('Kontrollering av resultat ............................... <K>');
         gotoxy(11,13);
      write('Avslutting av oppdrag ................................... <A>');
         gotoxy(11,14);
      write('Sletting av brukerfil ................................... <S>');
         gotoxy(11,16);
      write('List analyseresultater .................................. <L>');
         gotoxy(11,17);
      write('List oppdrag ............................................ <P>');
         gotoxy(11,18);
      write('List ioner .............................................. <I>');
         gotoxy(11,20);
      write('Avslutte ................................................ <Q>');
         video_on;
      end;
      gotoxy(1,1);


      REPEAT
         read(kbd,hm_valg);
         if not (upcase(hm_valg) in hoved_meny)
         then
            write(^G);
      UNTIL upcase(hm_valg) in hoved_meny;

      CASE upcase(hm_valg) OF
     (*    'B'     : Backup;
           'M'     : Metode;       *)
           'T'     : Topnav;
           'R'     : Regopdr;
     (*    'O'     : Stion;
           'K'     : Kontr;
           'A'     : Avslop;
           'S'     : Slbrk;
           'L'     : Lires;        *)
           'P'     : Opdrlst;
     (*    'I'     : Ionsok;       *)
      END;      (*  CASE  *)

   UNTIL  upcase(hm_valg) = 'Q';
END.
```