PROEFSTATION VOOR TUINBOUW ONDER GLAS TE NAALDWIJK

Bouwen van simulatieprogramma KASSIM

G. v. Steekelenburg

Intern rapport no.4
januari 1984

BOUWEN VAN KASSIM

Inhoud: Pag.

1. STRUCTUUR VAN KASSIM

1.1 Fasen tijdens het simuleren

Tijdens het simuleren kan men een aantal fasen onderscheiden, die ook in het simulatieprogramma zijn terug te vinden nl:

a. INFO: ophalen van informatie uit de verschillende blokken* (dimensies van arrays,namen van uitgangen enz.),
b. PARAMETERS: wijzigen van parameters (simulatietijden, outputparameters, parameters in model-/regelaarblokken, enz.),
c. INITIALISEREN van simulatie: in-/uitgangen en geheugenelementen krijgen een beginwaarde, starttijd wordt gezet, simulatieparameters getest enz.;
d. SIMULEREN: digitale en quasi-continue "simulatiedelen" (blokken*) worden doorlopen;
e. "UPDATING": tijdens een "interrupt" kunnen bijvoorbeeld parameters gewijzigd worden , daarna kan de simulatie voortgezet worden;
f. AFSLUITEN: tegenhanger van initialiseren bij beeindigen van simulatie (bijvoorbeeld afsluiten van files).

Alle blokken komen in iedere fase aan bod. In het flow-diagram in fig. 1.1-1 is het verband tussen deze fasen geillustreerd. In dit diagram zijn onder andere besturingsmogelijkheden als herstarten en stoppen niet opgenomen.

* Onder blokken worden in deze handleiding de subroutines REGELA, MODSIG en MODEL verstaan.

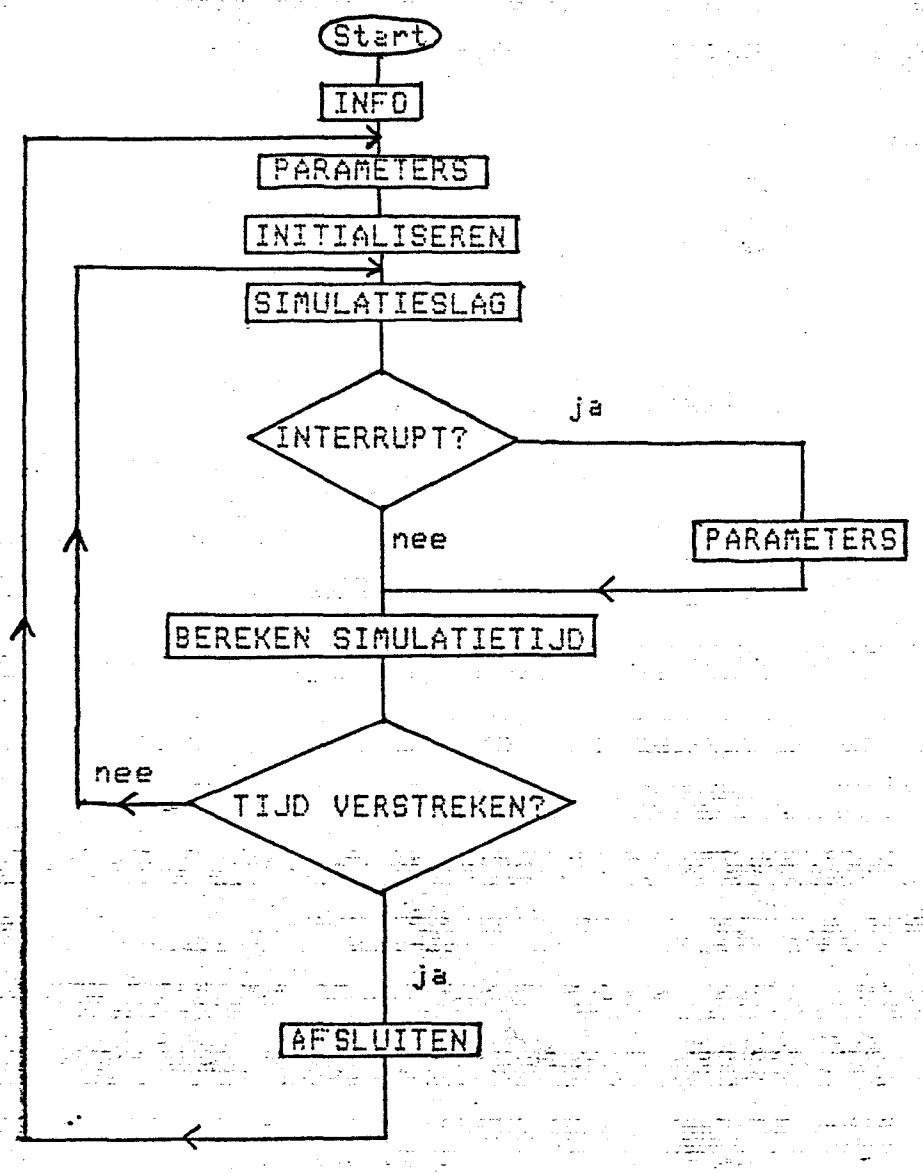


Fig. 1.1-1 Verband tussen de simulatiefasen

1.2 OPBOUW KASSIM MET BEHULP VAN SUBROUTINES

De opbouw van KASSIM met behulp van subroutines/funkties is in schema 1.2-1 aangegeven; basisroutines uit bibliotheek USERLIB worden hierin (bijna) niet genoemd.

Interpretatie van de boomstructuur:

- Er is slechts een opsomming van aangeroepen routines gegeven. De volgorde komt daarom niet noodzakelijkerwijs overeen met die in het programma en bovendien worden routines in iedere fase hoogstens eenmaal genoemd.
- Uiterst links komen de subroutines/funkties voor die aangeroepen worden in het hoofprogramma;
- deze routines roepen vaak weer andere routines aan, die dan tevens genoemd worden, waarbij het niveauverschil door inspringen wordt aangegeven.

- Routines op hetzelfde niveau beginnen in dezelfde kolom, of staan op dezelfde regel gescheiden door komma's.
- Onderniveaus van een routine worden slechts eenmaal genoemd, maar bij OUTPUT worden voor iedere fase de bijbehorende routines genoemd.

```
--- "informatiefase":
INFO
    OUTPUT
    BLOCKS
        MODSIG
        REGELA
        CONMOD
            MODEL
    TSTLIM
    PARLES
--- "parameterfase":
COMDIM,COMDEF,COMTST
SUBR
    BLOCKS
    OUTPUT
        COMDIM,COMDEF,COMTST
        PARAM
            PARSHO,CHANGE
        PARSHO
        PARN
            PARSHN,CHANGE
        HDTXT
        MMTAIL
        PREPAR
            EVHEAD
            OUTTST
            SCALL
            GRHEAD
                NAAMEV
            MMHEAD
    PARLES,PARFIL,PARWIS
    PARPRI
        COMDIM,COMDEF,COMTST
        HDTXT
        PARSHO
    PARAM
--- "initiatiefase":
PARTST
    PARAM
BLOCKS
OUTPUT
    PREPAR
RESET
    TTYATA (alleen onder FORTRAN 77)
        KBINTP
```

```
--- "simulatiefase":
BLOCKS
OUTPUT
    MEMO
        TABLE
            TBHEAD
            NAAMEV
            EVUIT
        GRAPH
            EVUIT
            GRPOS
TYDOP
    TTYDET,TTYATA (alleen onder FORTRAN 77)
    PARAM,CHANGE
--- "update-fase" (interruptie):
INTRUP
    TTYDET (alleen onder FORTRAN 77)
    COMDIM,COMDEF,COMTST
    SUBR
    RESET
--- "afsluitfase":
BLOCKS
OUTPUT
    MMTAIL
```

Schema 1.2-1 Programmastructuur.

Toelichting bij de verschillende routines:

Het merendeel van de routines is in bibliotheken ondergebracht, namelijk in:
SUBLIB: subroutines en functies speciaal voor PROGRAM KASSIM en SUBROUTINE OUTPUT,
SIMLIB: subroutines speciaal voor simulatiedoeleinden
USERLIB: de basissubroutines en -functies voor algemeen gebruik.
De routines uit USERLIB zijn elders beschreven.

INFO — berekent de beginpositie (pointers) in de diverse arrays voor de verschillende blokken, vervolgens wordt een parameterfile ingelezen (zie par. 1.3 en 1.4).
TSTLIM — test of arraygrenzen niet overschreden worden

COMDIM,COMDEF,COMTST dimensioneren, definieeren en testen van commando's (zie verder de beschrijvingen van USERLIB).

PARLES,PARPRI,PARFIL,PARWIS voeren handelingen met de parameters van het hele programma uit: resp. inlezen van schijf, printen (op terminal of schijf), wegschrijven op schijf, of wissen van parameterfiles.

SUBR roept, afhankelijk van het gegeven commando, een bepaalde subroutine aan
BLOCKS roept een of alle blokken aan
CONMOD verzorgt tijdens simulatiefase quasi-continue simulatie van MODEL door beurtelings te integreren en MODEL aan te roepen (CONMOD "integreert het bemonsterinterval vol"), in de andere fasen wordt MODEL slechts eenmaal aangeroepen.
OUTPUT voor definieeren en plegen van output
PARAM tonen of wijzigen van (array met) parameters
PARSHO tonen van (array met) parameters
CHANGE wijzigen van een parameter
PARN tonen of wijzigen van 2-dimensionale parameter array met "n kolommen" (vergelijk PARAM)
PARSHN tonen van 2-dimensionale parameter-array (vgl. PARSHO)
PREPAR bereidt plegen van output voor: test parameters, opent/sluit files, enz.
OUTTST test tabel-/grafiekparameters op juistheid
HDTXT verzorgt twee tekst regels boven uitvoer
SCALL levert "lege" grafiekregel met punten op schaalposities
GRHEAD verzorgt kop boven grafiek
GRAPH tekent grafiek
GRPOS berekent de positie van een signaal in de grafiek
TBHEAD verzorgt kop boven tabel
TABLE print tabel
EVHEAD leest kop van MEMORY-file
NAAMEV levert naam van signaal uit opgeslagen simulatie ("evaluatie-uitgang")
EVUIT waarde van "evaluatie-uitgang"
MMHEAD verzorgt kop van MEMORY-file
MEMO schrijft signalen weg in MEMORY-file
MMTAIL voltooit MEMORY-file
PARTST test juistheid tijdparameters van simulatie
RESET reset variabelen van interruptmechanisme
INTRUP bepaalt/test op interrupt door tijd, blok of toetsenbord
TTYATA koppelt invoer van terminal aan KBINTP
KBINTP signaleert keyboard-interrupt
TTYDET complement van TTYATA: geeft terminal vrij
TYDOP test of eindtijd van de simulatie bereikt is

1.3 VERBINDINGEN TUSSEN DE BLOKKEN.

De in- en uitgangen van de blokken zijn delen van een REAL array (dimensie 100). In fig. 1.3-1 is aangegeven hoe deze in- en uitgangen aangebracht zijn. De volgorde van links naar rechts komt overeen met de rekenvolgorde van de blokken in een simulatie-slag (bemonstering).

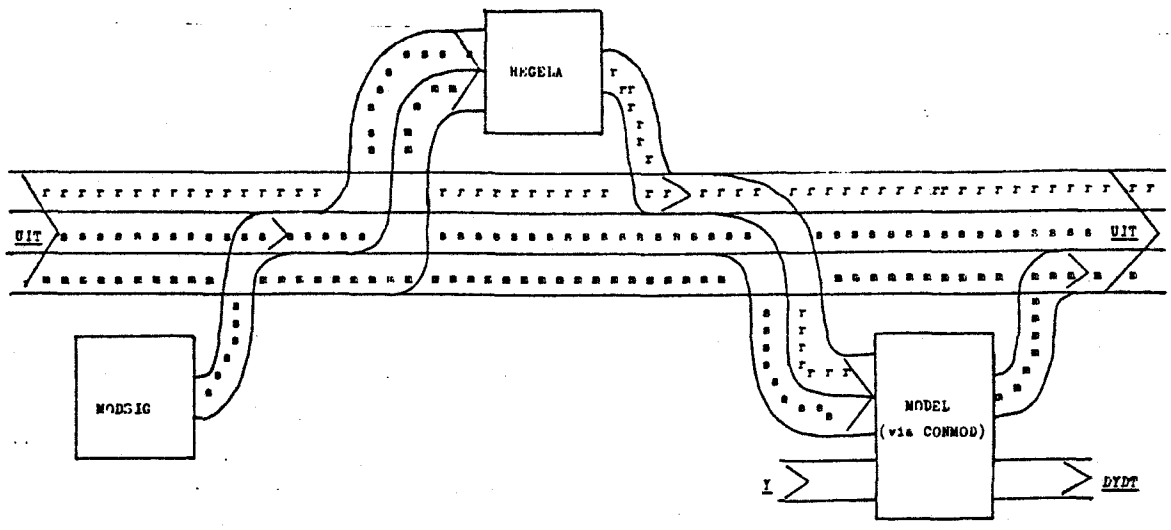


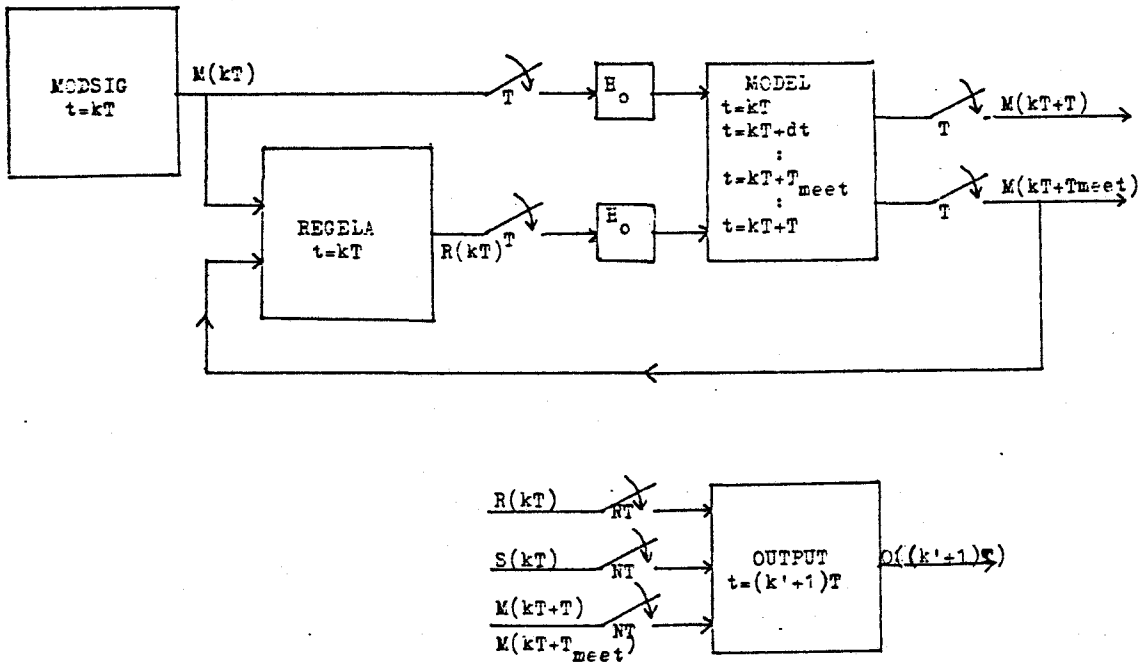
Fig. 1.3-1 Realisatie van verbindingen tussen de blokken.

1.4 Overige arrays

De parameter- en data-arrays van de blokken zijn eveneens onderdeel van grotere REAL arrays (dimensies 500 resp. 1000). Een gedeelte van deze arrays wordt gebruikt door PROGRAM KASSIM en SUBROUTINE OUTPUT. De arrays met namen van uitgangen en parameters zijn op dezelfde wijze onderdeel van grotere (type REAL*8).

2. REAL-TIME STRUCTUUR.

In fig. 2-1 is aangegeven op welke tijdstippen signalen van een blok doorgegeven worden aan de andere blokken. Bovendien is (het verloop van) de tijd binnen ieder blok aangegeven. De basistijd is dus de tijd op het voorgaande bemonstertijdstip. Voor het model wordt een quasi-continue tijd gegenereerd door subroutine CONMOD. Deze zorgt ervoor dat het bemonsterinterval "vol geintegreerd" wordt met de 2de-orde integratiemethode van RUNGA-KUTTA en integratieinterval dt.



R	:	uitgangen REGELA
S	:	uitgangen MODSIG
M	:	uitgangen MODEL
t	:	tijd
T	:	bemonsterinterval
dt	:	integratieinterval
Tmeet	:	moment binen bemonsterinterval waarop gemeten wordt (in ideale geval Tmeet=T, in praktijk Tmeet<T)
NT	:	=N*T, printinterval
k	:	-i,..,0,1,2,...,j
k'	:	<=k, veelvoud van N
H0	:	nulde-orde houdschakeling

Fig. 2-1 REAL-TIME structuur.

Opmerking: De simulatietijd is verschoven t.o.v. de tijd van de Siemens 330 computer. In de simulatie wordt het moment van regelen, bij de Siemens het meetmoment als referentie genomen, zie fig. 2-2.

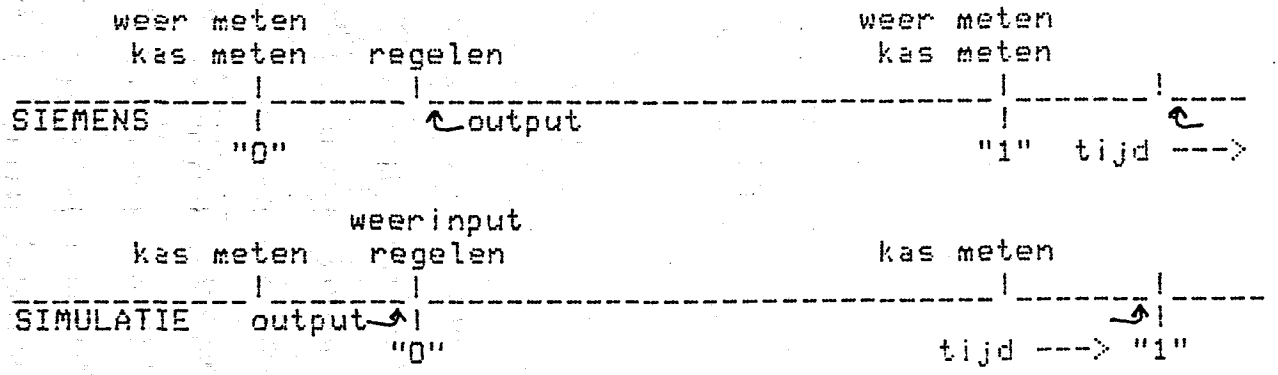


Fig. 2-2. Tijdreferentie bij Siemens 330 en in simulatie.

Consequenties: in de simulatie wordt de regelactie en zijn effekt tegelijkertijd weergegeven, in FERLIS verschijnt het effekt 1 minuut later.

2.1 Het quasi-continue deel van de simulatie.

Met behulp van de 2de-orde integratiemethode van RUNGA KUTTA wordt de continue integraal

$$y(t)=y(to) + \int_{to}^{t} f(\ x(t),y(t),t\ ).dt$$

benaderd met

$$y^{p}((k+1/2)*T)= y(kT) + 1/2*T*f(kT) \qquad *$$

$$y\ ((k+1)*T) = y(kT) + T*f^{p}((k+1/2)*T).$$

y en y^{p} worden door subroutine CONMOD berekend en f door subroutine MODEL (de berekening van f en f^{p} is identiek). T is het integratieinterval.

Is het bemonsterinterval geen veelvoud van het integratieinterval dan wordt de laatste integratieslag uitgevoerd met een aangepast integratieinterval, zodat het bemonsterinterval precies gevuld wordt.

---
* f(x(t),y(t),t) is afgekort tot f(t).

3. BLOKVOORSCHRIFTEN

3.1 Blokken als subroutines

De blokken MODSIG, MODEL, REGELA moeten als subroutines uitgevoerd zijn met de volgende argumentenlijst:

```
            REGELA
SUBROUTINE    of    (IN,UIT,TSIM,TBEM,MODUS,P,D,NAAMU,NAAMP)
            MODSIG

SUBROUTINE MODEL(IN,UIT,Y,DYDT,TCON,DT,MODUS,P,D,NAAMU,NAAMP)
```

Type en betekenis argumenten:

```
REAL IN(dim),UIT(dim*)  :in-/uitgangen zie ook par. 1.3
REAL Y(dim)             :uitgangen integratoren (max. 20),
                         worden berekend in CONMOD,
                         beginwaarden echter door MODEL
```

```
REAL DYDT(dim*)          :ingangen integratoren (max. 20),
                          worden berekend door MODEL
REAL TSIM,TBEM           :(discrete) simulatie- en
                          bemonstertijd
REAL TCON,DT             :quasi-continue simulatietijd, resp.
                          tijdinterval gebruikt door
                          integratieroutine, deze is gelijk
                          aan de helft van het opgegeven
                          integratieinterval! Beiden zijn van
                          belang voor bijv. de realisatie van
                          dode tijd m.b.v. SUBROUTINE DELAY
INTEGER MODUS            :MODUS/simulatiefase: 1: informatie
                          over dimensies en namen,
                          2: parameters wijzigen bijv. (zelf
                          te definieeeren), 3: initieeren,
                          4: simuleren, 5: afsluiten
REAL P(dim*),D(dim*)     :array met parameters resp. data
                          (=tijdelijk opgeslagen resultaten)
REAL*8 NAAMU(dim*),NAAMP(dim*) :namen (8 karakters) van
                                uitgangen resp. parameters
```

3.2 Simulatiefasen/dimensies

In de blokken moet rekening gehouden worden met de verschillende simulatiefasen (aangegeven door MODUS):

MODUS=1: INFORMATIE
In deze fase moeten de dimensies (dim*) van UIT, P, D, Y en DYDT in betreffend blok opgegeven worden via corresponderende variabelen in het COMMON gebied DIM:

```
      INTEGER NREG,NPREG,NDREG,NMOS,NPMOS,NDMOS,NMOD,NY,NPMOD,NDMOD
      COMMON/DIM/
      1 NREG,NPREG,NDREG,NMOS,NPMOS,NDMOS,NMOD,NY,NPMOD,NDMOD
```

NMOD: aantal uitgangen van MODEL
NY: aantal integratoren van MODEL
NPMOD: aantal parameters van MODEL
NDMOD: aantal geheugenplaatsen van MODEL
analoog de parameters voor de andere blokken.

Let op de juiste volgorde van de elementen.

De som van de dimensie's van de verschillende blokken is aan maxima gebonden, namelijk:

```
      UIT     :100
      P       :460 (500-(5+35))
      D       :850 (1000-150)
      Y,DYDT  :elk 20
```

Opmerking: Het hoofdprogramma en OUTPUT nemen een gedeelte van de arrays P en D in beslag.

MODUS=2: (PARAMETERS)
Deze modus/fase is zelf te definieeren, hierin kan men bijv. parameters wijzigen, tussenresultaten bekijken enz.

MODUS=3: INITIATIE
In deze fase, die voorafgaat aan de eigenlijke simulatie, moeten wellicht de elementen van UIT en D hun beginwaarde krijgen. Voorts kunnen acties verricht worden zoals het openen van files.

MODUS=4: SIMULATIE
In deze fase worden de simulatieformules doorgerekend.

MODUS=5: AFSLUITING
Deze fase sluit een simulatie af, hierin kunnen bijv. files gesloten worden.

3.3 Bewaren van data

Omdat het programma in overlay draait gaan resultaten van een berekening uit een vorige bemonsterperiode verloren als ze niet opgeslagen worden in array D. Uitgangssignalen echter, blijven hun waarde behouden en hoeven dus niet in D opgeslagen te worden.

3.4 Interruptie.

Een blok kan een interrupt plegen tijdens het simuleren door aan de variabele BLINTP de waarde .TRUE. toe te kennen, BLINTP staat in een COMMON:

```
      LOGICAL BLINTP          ! blokinterrupt
      COMMON/CINTRP/BLINTP
```

3.5 Service routines

De volgende subroutines staan ter beschikking voor gebruik in de blokken:

FNC. FASE(MODUS,'STRING') test en vraagt eventueel om MODUS
(FASE=MODUS, beide integers)

Manipuleren van arrays:
SUBR. COPARR(ARRIN,ARRUIT,N) voor het copieeren van REAL arrays (N elementen)
SUBR. COPAR8(ARRIN,ARRUIT,N) dito voor REAL*8 arrays
SUBR. NITARR(VAL,ARR,N) voor initieeren van REAL array

Parameters wijzigen/tonen:
SUBR. PARS(P,NAAMP,NPAR,IERR) voor tonen/wijzigen parameters d.m.v. commando's (gebruikt PARAM)
SUBR. PARAM(NR,VAL,NAAMP,P,NPAR)
als NR=-32768 worden de NPAR parameters uit P met de namen NAAMP op de terminal getoond in 3 kolommen (of minder indien aangegeven met VAL);

als 1<=NR<=NPAR en VAL=-32768 wordt voor P(NR) de waarde gevraagd (door dan uitsluitend <CR> te geven, blijft oude waarde behouden), vervolgens van volgende elementen tot "S" ingetypt wordt of NPAR bereikt is;
als 1<=NR<=NPAR en VAL#-32768 dan wordt VAL toegekend aan P(NR)

Voor commandogebruik:

SUBR. COMDIM(NCOM,NVAL) voor opgeven aantal te definieeren commando's, resp. max aantal in te lezen commando argumenten (getallen)

SUBR. COMDEF(NRCOM,'STRING') definieeert commando met nr. NRCOM, 'STRING' bestaat uit maximaal 18 tekens, beginnend met een "woord" (slechts letters), dat het eigenlijke commando vormt

SUBR. COMTST(NR,VAL) geeft het nr. van het gedecodeerde commando vrij of 0 bij ^Z, bovendien worden met het commando meegegeven getallen in REAL array VAL geplaatst. Is het aantal getallen kleiner dan NVAL dan staan in de overblijvende elementen -32768 (default)

Testen terminal invoer:

SUBR. REDANS(JA,NEE) test het antwoord op een vraag en retourneert:
JA=.T.,NEE=.F. op 'J','Y','1'
JA=.F.,NEE=.T. op 'N','0' of <CR>
JA=.F.,NEE=.F. op ^Z

SUBR. REDVAL(I,R) kent ingetypt getal toe aan resp. integer I en real R

Gebruik van files:

FNC. GETF(U) stelt vrij unitnummer beschikbaar via integers GETF en U

SUBR. CLOSEF(U) sluit unit U en geeft nummer vrij (beveiligd tegen sluiten van reeds gesloten unit)

SUBR. PUTF(U) maakt unitnummer U vrij, te gebruiken als bijv. een fout optreedt bij openen van een file

Rekenkundige routines:
In de bibliotheek USERLIB zijn diverse routines beschikbaar voor begrenzen, afronden e.d. van variabelen.

Specifieke routines:
In SIMLIB zijn speciale routines voor de regelaar en het model aanwezig:

Voor REGELA:
DAGTYD,DAPRO,LIMIT,MODPI,PROPR4,ROOS,SETPT,TEWIZO

```
Voor MODEL:
SUBR. DELAY(X,TDEL,DT,TCON,PYP,NPYP)
                  voor simuleren van quasi-continue dode tijd
Type en betekenis argumenten:
REAL X            :in-/uitgangsvariabele
REAL TDEL         :dode tijd
REAL DT           :stapgrootte (in de tijd)
REAL TCON         :quasi-continue tijd
REAL PYP          :array dat dient als geheugen
INTEGER NPYP      :dimensie van NPYP, moet minstens (TDEL/DT)+1
                   bedragen

SUBR. FIRSTO(U,Y,K,TAU,DYDT) berekent integratoringang DYDT
                          voor een 1ste-orde overdracht
SUBR. VRSTEL(MPULS,TSTUUR,TLOOP,STAND,ESTAND,MIN,MAX,DYDT)
                          zorgt in samenwerking met een
                          integrator (STAND=Y) voor
                          verstelling van een klep, raam e.d.
SUBR. NITMET(MPULS,TCON,DTMEET,TMEET)} zorgen voor het meten
SUBR. MEET(SIG,SIGM,RUIS,TCON,TMEET) } van signalen
SUBR. ENDMET(TCON,TMEET)             }
```

4. HET BOUWEN VAN EEN NIEUW PROGRAMMA

Na het inloggen onder UIC 200,101 (zie handleiding "Gebruik van KASSIM") wordt een programma (CMD-file) doorlopen, waarmee men onder andere een bestaand simulatieprogramma kan bouwen. De structuur van deze CMD-file is in fig. 4-1 aangegeven. Men hoeft slechts tussen acties te kiezen en filenamen op te geven, de commando's worden vervolgens door de CMD-file gegenereerd.

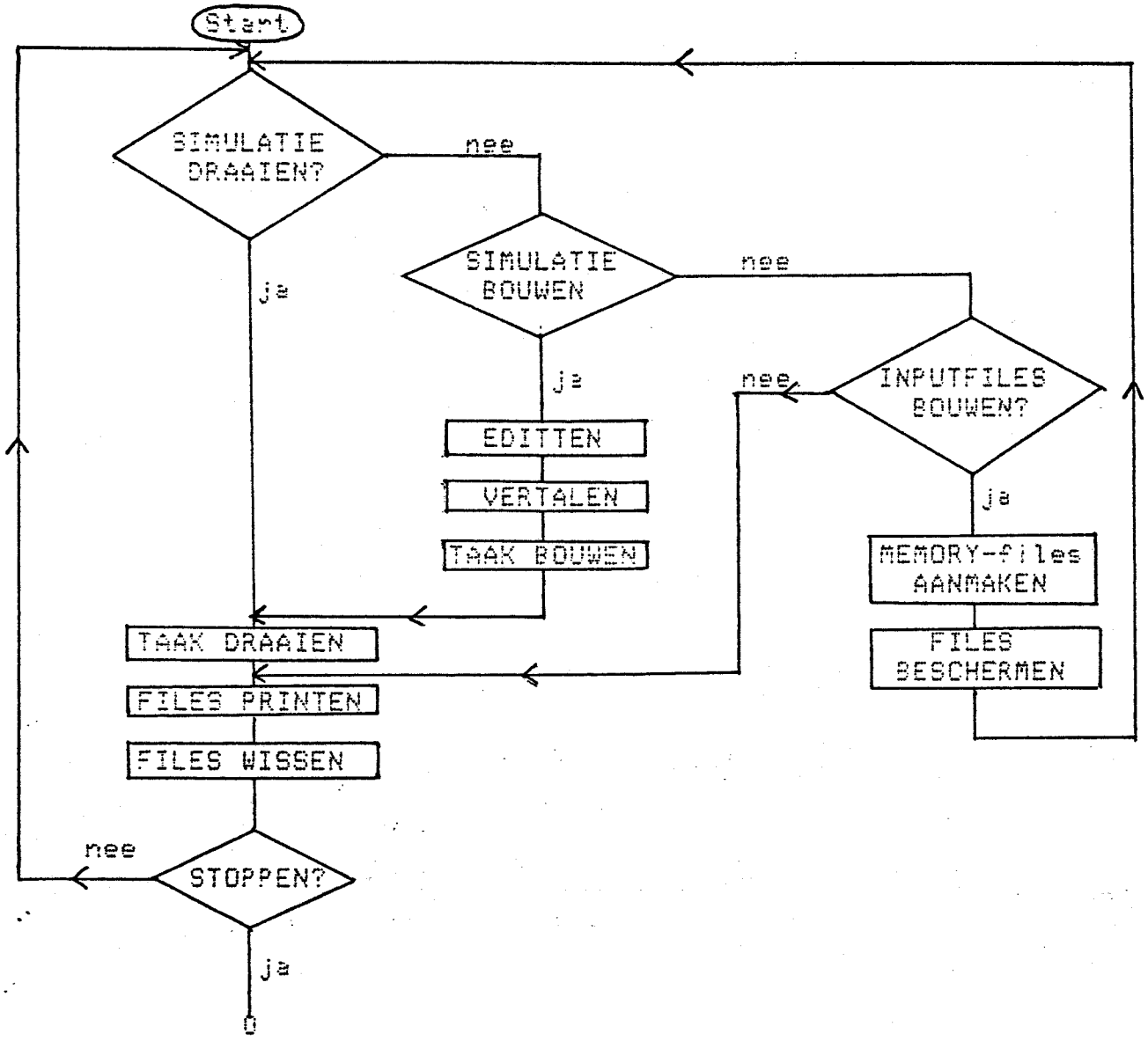


Fig. 4-1 Opbouw LOGIN.CMD file.

4.1 Het schrijven en vertalen van Fortran subroutines

Het schrijven van Fortran subroutines gebeurt m.b.v. een editor. Na het opgeven van een naam wordt deze automatisch door het programma aangeroepen. Het programma kan sumiere informatie over deze editor geven, meer informatie is in betreffende manuals te vinden. Voor het bouwen van de benodigde "blokken" kan men uitgaan van "voorbeeldblokken", namelijk MODSIG, REGELA en MODEL. Na het "editten" wordt betreffend programma automatisch vertaald. Vervolgens kunnen eventuele fouten verbeterd worden, enz.

4.2 Het bouwen en draaien van een simulatietaak

De vertaalde, losse subroutines moeten o.a. aan het

hoofdprogramma gekoppeld (linked) worden. Dit gebeurt m.b.v. een taskbuilder en resulteert in een "task". Het is voldoende om de (nieuwe) taaknaam en de namen van de verschillende onderdelen (blokken en eventueel BLOCKDATA) op te geven, het programma doet de rest. Wel moet men aangeven of men voor de blokken een overlay-structuur wenst. Als de blokken tesamen te groot zijn, kunnen ze namelijk niet een moduul vormen, maar moeten in twee delen "geknipt" worden. Deze delen worden tijdens het draaien om beurten in het werkgeheugen geladen. Het eerste deel bestaat dan uit MODSIG en REGELA, het tweede deel uit MODEL. Of deze overlay-structuur noodzakelijk is blijkt als tijdens het bouwen gemeld wordt dat er te weinig geheugenruimte beschikbaar is. Gebruik niet onnodig overlay, omdat "in- en uit-transfereren" van programmadelen in het geheugen ook tijd kost en het programma dus trager maakt.

4.3 Het bouwen van inputfiles

Als inputfiles voor het simulatieprogramma kunnen zogenaamde MEMORY-files gebruikt worden. Deze files hebben dezelfde structuur als die, die tijdens de simulatie met behulp van MEMORY aangemaakt worden. Deze files kunnen ook uit alphanumerieke sequentiele files (leesbare tekst) vervaardigd worden met behulp van programma MAKMEM. Dit programma kan maximaal vier alphanumerieke files (input) omzetten in twee MEMORY-files (output). Daarbij kan voor elk FER-nummer uit elke (input)file opgegeven worden of en in welke outputfile dat signaal weggeschreven moet worden en onder welke naam en welk uitgangsnummer. De vervaardigde files kunnen vervolgens beschermd worden tegen wissen.

4.4 Overige file-operaties

Na het draaien kunnen files geprint of gewist worden. Een aantal files zijn echter beschermd tegen wissen, zodat er een melding kan verschijnen als men dat toch probeert (dit geldt ook voor "purgen"). Zie voor meer informatie par. 4.11 van "Gebruik van KASSIM".

APPENDIX A

VOORBEELD BOUWEN INPUTFILE EN SIMULATIETAAK

De tekst die tijdens de hierna getoonde sessie ingetypt werd, is onderstreept. Ingesprongen tekst is commentaar.

```
>HELLO 200,1D1/KASSIM

RSX-11M BL32   [1,54] System     NAALDW
27-DEC-83 13:49 Logged on Terminal TT3:

Good Afternoon

MSTIME
Welcome to RSX-11M V4.0-B timesharing 11/44


##########################################################################


******** COMPUTER SYSTEEM PROEFSTATION NAALDWIJK ********

==========================================================================
       Vrijdags gaat de computer in verband met back-up maken om 16.00 uur uit.
       Tevens worden alle files op de laatste twee na uit het systeem
       verwijderd.
==========================================================================
Klachten of vragen  Leo van den Bos  tst 145/140/226
>@LOGIN.CMD
>;
>;
>;      SIMULATIEPAKKET   K A S S I M !
>;
>;
>;In dit programma zullen achtereenvolgens verschillende fasen
>;doorlopen worden. Een fase overslaan kan door RETURN te geven
>;zodra een filenaam gevraagd wordt.
>;
>* SIMULATIE DRAAIEN? [Y/N]: N
>* NIEUW PROGRAMMA BOUWEN? [Y/N]: N
>* INPUTFILES (MEMORY-files) AANMAKEN? [Y/N]: Y
>RUN MAKMEM
Inlezen van files met format 14X,66A1,
signalen vermenigvuldigen met 0.1,
wegschrijven in direct access MEMORY-files voor KASSIM.
AANTAL INPUTFILES (MAX.4): 2

GEEF FILENAAM: KLIMA2.DAT

GEEF FILENAAM: KL2LWB.DAT
OPNIEUW?
1 OF 2 OUTPUTFILES:1
```

```
GEEF FILENAAM: BUIS.MEM
OPNIEUW?
BESTEMMINGEN VAN MAXIMAAL 9 INPUTSIGNALEN PER OUTPUTFILE:
OUTPUTFILE BUIS.MEM
Geef ieder signaal een uitgangsnummer (0-999) en een naam (max. 8 karakters).
INPUTFILE KLIMA2.DAT
Ingangssignaal: Uitgangsnummer, naam:
    1.
    2.
    3.
    4.
    5.
    6.
   32.
   36.
   80.          27,TBUIS
   84.
INPUTFILE KL2LW8.DAT
Ingangssignaal: Uitgangsnummer, naam:
   80.          40,TBUIS0
   84.
OPNIEUW?
Gegevens voor MEMORY-files:
Bemonstertijd=1
Starttijd 1 hoger kiezen dan in input file
Printinterval=1

OUTPUTFILE BUIS.MEM
GEEF BEMONSTERTIJD: 1
STARTTIJD SIMULATIE: 1
PRINT-INTERVAL: 1

                PROGRAMMA CONVERTEERT FILES!


LAATSTE SIGNAALWAARDEN:

INPUTFILE: KLIMA2.DAT
 0.1100000E+02 0.3700000E+02 0.9080000E+03 0.6000000E+01 0.2715000E+04 0.0000000E+00 0
.2020000E+03 0.2010000E+03 0.5250000E+03 0.5020000E+03
INPUTFILE: KL2LW8.DAT
 0.5641200E+03 0.5758510E+03 0.0000000E+00 0.0000000E+00 0.0000000E+00 0.0000000E+00 0
.0000000E+00 0.0000000E+00 0.0000000E+00 0.0000000E+00
TIJD  --  STOP
        Zodra het eind van een inputfile bereikt is, stopt het programma
        met lezen en wegschrijven van data. Zijn de inputfiles ongelijk
        van lengte (zoals hier), dan staan dus de laatst gelezen en hier
        getoonde data doorgaans niet als laatste record in de MEMORY-file.
>:OUTPUTFILES BESCHERMEN TEGEN WISSEN?
>* GEEF FILENAAM (GEEN VERSIENR.) OF <CR> [S]: BUIS.MEM
>* GEEF FILENAAM (GEEN VERSIENR.) OF <CR> [S]:
>* SIMULATIE DRAAIEN? [Y/N]:
>* NIEUW PROGRAMMA BOUWEN? [Y/N]: Y
>* WILT U INFORMATIE OVER DIT PROGRAMMA EN DE EDITOR? [Y/N]:
>:EDITTEN...VERTALEN
>* FILENAAM (ZONDER EXT.) [S]: REGVOORB
```

```
>EDIT/EDT REGVOOR8.FTN
C       Programmanaam           :REGVOOR8.FTN
C       Doel                    :regelaar voor MODEL
C       Inputs:                 :IN,TSIM,TEEM,MODUS,P,D
C       Outputs:                :UIT,P,D,NAAMU,NAAMP
C-------------------------------------------------------------------------
        SUBROUTINE REGELA(IN,UIT,TSIM,TEEM,MODUS,P,D,NAAMU,NAAMP)
C
        REAL    IN(2),UIT(1),TSIM,TEEM,P(3),D(1),E
        INTEGER MODUS,IERR
        REAL*8  NAAMU(1),NAAMP(3)
C
C               aantal uitgangen,(integratoren),parameters,data van blokken
        INTEGER NREG,NPREG,NDREG,NMOS,NPMOS,NDMOS,NMOD,NY,NPMOD,NDMOD
        COMMON/DIM/
        1 NREG,NPREG,NDREG,NMOS,NPMOS,NDMOS,NMOD,NY,NPMOD,NDMOD
C
        IF(MODUS.NE.1)GOTO 200
        NREG=1
        NPREG=3
        NDREG=1
        NAAMU(1)='STURING'

*EX
DM2:[200,101]REGVOOR8.FTN;3 45 lines
        Na het openen van de file wordt deze door EDT getoond. Door het
        commando EX wordt de editor verlaten.
>FOR REGVOOR8/LIST
REGELA
>* LISTING(2X) EN EDITTEN? [Y/N]:
>* FILENAAM (ZONDER EXT.) [S]:
>* DEEF TAAKNAAM [S]: SIMVOOR8
>;Bij de volgende filenamen hoeft de default-extensie .OBJ
>;niet aangegeven te worden.
>* NAAM VAN BLOCKDATA [S]:
>* NAAM VAN SUBROUTINE MODSIG [S]: SIGVOOR8
>* NAAM VAN SUBROUTINE REGELA [S]: REGVOOR8
>* NAAM VAN SUBROUTINE MODEL  [S]: MODVOOR8
>* BLOKKEN IN OVERLAY? [Y/N]: N
>;
>;EVEN GEDULD A.U.B. SIMULATIETAAK WORDT GEBOUWD
>;
>;
>TKB @KASSIM
>DEL KASSIM.ODL;0,KASSIM.CMD;0
        Een BLOCKDATA subprogramma mag opgegeven worden, maar hoeft niet.
>;AANWEZIGE SIMULATIETAKEN (*.TSK), PARAMETER- EN MEMORYFILES (*.PAR,*.MEM):


Directory DM2:[200,101]
27-DEC-83 12:05

KASSIM.TSK;1         209.    C  31-OCT-83 16:59
KASSIM2.TSK;13       211.    C  08-SEP-83 09:02
UNISIM.TSK;1         233.    C  09-DEC-83 11:36
MAKMEM.TSK;1         101.    C  02-DEC-83 10:07
VENTVOUD.TSK;2       212.    C  21-SEP-83 09:52
SIMVOOR8.TSK;1       164.    C  27-DEC-83 12:03
```

Na het linken wordt de nieuwe taak automatisch gestart, zie verder handleiding "GEBRUIK VAN KASSIM".

```
### SIMULATIEPROGRAMMA KASSIM ###

PARAMETERS (ALS AFWEZIG <CR>):
GEEF FILENAAM: P2802.PAR
+-------------------------------------------------------------------------------+
#REGELAAR#          #SIGNALEN#          #MODEL#              #OUTPUT            #
#INLEZEN PAR.FILE   #PRINTEN PARAMETERS #OPSLAAN PARAMETERS #WISSEN PAR.FILES  #
#TSIM#=#            #RUN                #STOP                #                  #
+-------------------------------------------------------------------------------+
COM>STOP

EINDE KASSIMULATIE!


Directory DM2:[200,101]
27-DEC-83 12:06

P10REG.LST;1          31.          13-OCT-83 13:49
P300382.LST;1         12.          17-NOV-83 14:48
RVENT.LST;1           37.          18-NOV-83 11:05
KVENT.LST;1           37.          18-NOV-83 12:10
P10MOD2.LST;2         41.          05-DEC-83 13:33
 P280283.LST;1        4.           05-DEC-83 14:36
 ----- 83.LST;4       7.           21-DEC-83 13:31

>* TE PRINTEN FILES [S]:
>;GEEF BEGINDATUM VOOR FILE-DIRECTORY (vorm: nn-MON-83, vb: 01-MAY-83)
>* DATUM [S]: 27-DEC-83


Directory DM2:[200,101]
27-DEC-83 12:09
*.CMD;* excluded
Dates after 27-DEC-83

GUIS.MEM;1            113.         27-DEC-83 11:52
SIMVOOR8.TSK;1        164.     D   27-DEC-83 12:03
REGVOOR8.OBJ;10       5.           27-DEC-83 12:00
REGVOOR8.FTN;3        3.           27-DEC-83 12:00
REGVOOR8.LST;10       7.           27-DEC-83 12:00

Total of 292./294. blocks in 5. files

>* GEEF TE WISSEN FILES [S]:
>* STOPPEN? [Y/N]: Y
>* PURGEN VAN ALLE FILES TOEGESTAAN? [Y/N]:
>* GEEF TE PURGEN FILES [S]: *.OBJ
12:11:01  Task "AT.T5 " terminated
          Aborted via directive or CLI

$
Have a Good Afternoon
27-DEC-83 12:11  TT5:  logged off NAALDW
>
```

APPENDIX B

VOORBEELDEN BLOKSTRUCTUUR

```
C-----------------------------------------------------------------------------
C       Programmanaam           :REGVOORB.FTN
C       Doel                    :regelaar voor MODEL
C       Inputs:                 :IN,TSIM,TBEM,MODUS,P,D
C       Outputs:                :UIT,P,D,NAAMU,NAAMP
C-----------------------------------------------------------------------------
        SUBROUTINE REGELA(IN,UIT,TSIM,TBEM,MODUS,P,D,NAAMU,NAAMP)
C
        REAL    IN(2),UIT(1),TSIM,TBEM,P(3),D(1),E
        INTEGER MODUS,IERR
        REAL*8  NAAMU(1),NAAMP(3)
C
C               aantal uitgangen,(integratoren),parameters,data van blokken
        INTEGER NREG,NPREG,NDREG,NMOS,NPMOS,NDMOS,NMOD,NY,NPMOD,NDMOD
        COMMON/DIM/
        1 NREG,NPREG,NDREG,NMOS,NPMOS,NDMOS,NMOD,NY,NPMOD,NDMOD
C
        IF(MODUS.NE.1)GOTO 200
        NREG=1
        NPREG=3
        NDREG=1
        NAAMU(1)='STURING'
        NAAMP(1)='SETPOINT'
        NAAMP(2)='KP'
        NAAMP(3)='KI'
        GOTO 9000
200     IF(MODUS.NE.2)GOTO 300
        DO 220 I=1,3
                CALL CHANGE(D,NAAMP(I),P(I),IERR)
220     CONTINUE
        GOTO 9000
300     IF(MODUS.NE.3)GOTO 400
        UIT(1)=0.
        D(1)=0.
        GOTO 9000
400     IF(MODUS.NE.4)GOTO 500
        E=P(1)-IN(2)            !IN(1)=RUIS,IN(2)=UITGANG 1ste-orde overdracht
        D(1)=D(1)+E             !FOUTSOM
        UIT(1)=P(2)*E+P(3)*D(1) !PI-regelactie
        GOTO 9000
500     IF(MODUS.NE.5)GOTO 600
        GOTO 9000
600     CALL TYPE('MODUS ONJUIST, KIES MODUS TUSSEN 1 EN 5!')
9000    RETURN
        END
```

```
C-------------------------------------------------------------------------------
C       Programmanaam             :SIGVOOR8.FTN
C       Doel                      :GENEREREN RUIS VOOR MODEL
C       Inputs:                   :TSIM,MODUS,P,D
C       Outputs:                  :UIT,P,D,NAAMU,NAAMP
C-------------------------------------------------------------------------------
        SUBROUTINE MODSIG(IN,UIT,TSIM,TBEM,MODUS,P,D,NAAMU,NAAMP)
C
        REAL    UIT(1),TSIM,TBEM,P(1),D(2)
        INTEGER MODUS,IERR
        REAL*8  NAAMU(1),NAAMP(3)
C
C               aantal uitgangen,(integratoren),parameters,data van blokken
        INTEGER NREG,NPREG,NDREG,NMOS,NPMOS,NDMOS,NMOD,NY,NPMOD,NDMOD
        COMMON/DIM/
        1 NREG,NPREG,NDREG,NMOS,NPMOS,NDMOS,NMOD,NY,NPMOD,NDMOD
C
        IF(MODUS.NE.1)GOTO 200
        NMOS=1
        NPMOS=1
        NDMOS=2
        NAAMU(1)='RUIS'
        NAAMP(1)='AMPLITUD'
        GOTO 9000
200     IF(MODUS.NE.2)GOTO 300
        DO 220 I=1,NPMOS
              CALL CHANGE(D,NAAMP(I),P(I),IERR)
220     CONTINUE
        GOTO 9000
300     IF(MODUS.NE.3)GOTO 400
        UIT(1)=0.
        D(1)=0.
        D(2)=0.
        GOTO 9000
400     IF(MODUS.NE.4)GOTO 500
        UIT(1)=P(1)*(1-+2*RAN(D(1),D(2) ) )
        GOTO 9000
500     IF(MODUS.NE.5)GOTO 600
        GOTO 9000
600     CALL TYPE('MODUS ONJUIST, KIES MODUS TUSSEN 1 EN 5!')
9000    RETURN
        END
```

```
C-------------------------------------------------------------------------
C       Programmanaam           :MODVOORB.FTN
C       Doel                    :voorbeeld van een model (1ste-orde systeem)
C       Inputs:                 :IN,TSIM,TSEM,MODUS,P,D
C       Outputs:                :UIT,P,D,NAAMU,NAAMP
C-------------------------------------------------------------------------
        SUBROUTINE MODEL(IN,UIT,Y,DYDT,TCON,DT,MODUS,P,D,NAAMU,NAAMP)
C
        REAL    IN(2),UIT(1),TCON,DT,P(2),D(1),TAU,K,Y(1),DYDT(1)
        INTEGER MODUS,IERR
        REAL*8  NAAMU(1),NAAMP(2)
C
C               aantal uitgangen,(integratoren),parameters,data van blokken
        INTEGER NREG,NPREG,NDREG,NMOS,NPMOS,NDMOS,NMOD,NY,NPMOD,NDMOD
        COMMON/DIM/
        1 NREG,NPREG,NDREG,NMOS,NPMOS,NDMOS,NMOD,NY,NPMOD,NDMOD
C
        IF(MODUS.NE.1)GOTO 200
        NMOD=1
        NPMOD=2
        NDMOD=0
        NY=1
        NAAMU(1)='UITGANG'
        NAAMP(1)='K'
        NAAMP(2)='TAU'
        GOTO 9000
200     IF(MODUS.NE.2)GOTO 300
        DO 220 I=1,NPMOD
                CALL CHANGE(D,NAAMP(I),P(I),IERR)
220     CONTINUE
        GOTO 9000
300     IF(MODUS.NE.3)GOTO 400
        UIT(1)=0.
        Y(1)=0.
        GOTO 9000
400     IF(MODUS.NE.4)GOTO 500
        K=P(1)
        TAU=P(2)
        CALL FIRSTO(IN(1),Y(1),K,TAU,DYDT(1))
        UIT(1)=Y(1)+IN(2)                       !TOEVOEGING RUIS
        GOTO 9000
500     IF(MODUS.NE.5)GOTO 600
        GOTO 9000
600     CALL TYPE('MODUS ONJUIST, KIES MODUS TUSSEN 1 EN 5!')
9000    RETURN
        END
```