KFKI-1977-47

JÁNOSY J.S.

A PROHYS PROGRAM
FELHASZNÁLÓI ISMERTETŐ

Hungarian Academy of Sciences

CENTRAL
RESEARCH
INSTITUTE FOR
PHYSICS

BUDAPEST

KFKI-1977-47

A PROHYS PROGRAM FELHASZNÁLÓI ISMERTETŐ

Jánosy János Sebestyén
Reaktor Elektronikai Főosztály
Központi Fizikai Kutató Intézet, 1525, Budapest, pf. 49.

HU ISSN 0368-5330
ISBN 963 371 270 X

TARTALOMJEGYZÉK

BEVEZETÉS

A PROHYS blokk-orientált, hibrid szemléletü szimulációs program ismertetése az [1,2] reportokban található. Jelen leirás csak a konkrét felhasználáshoz /adatszerkesztés, kezelés, futtatás, hibaüzenetek, stb./ nyujt segitséget. Feltételezzük, hogy a hibrid szimulációs feladat analóg része a PROHYS elemkészlete alapján felépitett konkrét kapcsolási vázlat, a digitális rész pedig FORTRAN nyelven irt algoritmusok formájában már készen van.

1. Általános tudnivalók

1.1. A PROHYS adaptálhatósága

A PROHYS szimulációs program FORTRAN-IV nyelven készült. A program könnyü adaptálhatósága érdekében kerültük az olyan utasitások használatát, amelyek egyes gépi reprezentációkban nem megengedettek. Számitva a kisszámitógépek több megszoritást tartalmazó forditóprogramjaira, az egyes utasitásokat a legegyszerübb formában alkalmaztuk /ciklushatárként csak egész szám és egész tipusu változó szerepel, kifejezés nem, a tömbök maximális dimenziószáma 3, stb./.

A program egy bemeneti /lyugszalag- vagy lyukkártyaolvasó/ és két kimeneti /sornyomtató és szalag- vagy kártyalyukasztó/ perifériát használ. Az egyes perifériák logikai perifériaszámai a READ illetve a WRITE utasitásokban:

lyukszalag- vagy lyukkártyaolvasó:	1
szalag- vagy kártyalyukasztó:	3
sornyomtató:	4

Valamennyi periféria alfanumerikus információt visz át.

Mivel a FORTRAN-IV nem tartalmaz utasitásokat a plotter kezelésére vonatkozóan, rajzokat készitení a PROHYS megváltoztatása nélkül csak ICL 1905 tipusu számitógéppel lehet. A rajzolást a szimuláció utolsó fázisaként külön FORTRAN szubrutin végzi, ez aktivizálja a megfelelő könyvtári rutinokat. A szubrutinba való belépéskor két vektor /PLX és PLY, dimenziójuk: 256/ azonos indexü elemei tartalmazzák az X-Y pontpárok értékeit, NEND változó pedig a pontpárok számát jelzi. Az adaptálás során ezt a rajzoló szubrutint kell az adott gépnél használatos könyvtári rutinok felhasználásával ujrairni.

A PROHYS programnak több változata van, melyek memóriaigény és szolgáltatások tekintetében erősen különböznek. Valamennyi változat számára az adatok szerkesztése azonos módon történik, a nagyobb változat mindig futtatható a kisebb számára szerkesztett adatokkal. Jelen leirás a legtöbb szolgáltatást nyujtó változat használatát ismerteti, amelynek helyfoglalása az ICL 1905 tipusu számitógép operativ memóriájában a következő volt:

közös adatmező	kb. 11 kszó
permanens programmező	kb. 6 kszó
felülirható programmező	kb. 10 kszó

1.2. A PROHYS üzemmódjai és felépitése

A felhasználónak a problémát a program számára feladatokra /job/ kell lebontania. A PROHYS egyetlen futás alkalmával tetszőleges mennyiségü feladatot végezhet el. A feladat lehet egy adott modell megadott ideig történő szimulációja, vagy egy adott modell vizsgálata a kb. 1 %-os pontossághoz szükséges lépésköz meghatározása érdekében, azaz lépésközbecslés.

A modell analóg részének megadása is kétféleképpen történhet: vagy a teljes modell szerepel az adathordozón, azaz a feladat elvégzése uj adatokkal történik, vagy a modell az előző feladatban szerepelttel azonos, attól csak koefficiensekben, kezdeti feltételekben, illetve időzitésekben különbözik; ilyenkor elegendő a módositó adatok megadása.

Ennek megfelelően a program négy üzemmódja lehetséges:

- uj adatok, szimuláció
- uj adatok, lépésközbecslés
- módositó adatok, szimuláció
- módositó adatok, lépésközbecslés

Az üzemmódokat - egyetlen megszoritással - tetszőlegesen lehet változtatni, azaz a feladatok sorrendje az adathordozón tetszőleges lehet. A megszoritás: lépésközbecslés után nem következhet ugyanannak a modellnek a szimulációja módositó adatokkal, mert a lépésközbecslés során helyszüke miatt a szimulációhoz szükséges adatok egy része felüliródik.

A modell digitális részét egy futás során nem lehet megváltoztatni, mert a felhasználói programrészeket a későbbiekben leirt módon szegmensekbe kell szervezni, és a PROHYS programmal konszolidálva kell a memóriába tölteni.

A PROHYS program a következő szegmensekből áll:
SIMULATION - ez a rövid szegmens a tulajdonképpen főprogram. A könyvtári rutinokon kivül csak ez tartózkodik állandóan az operativ memóriában. Csak vezérlést és hibadetektálást végez. Ez a szegmens állitja be az üzemmódokat, és aktivizálja a szükséges részfeladatokat.

PREPARATION - ez a szegmens olvassa be az adatokat, uj modell esetén feldolgozza az analóg kapcsolási vázlatot, beállitja az egyes szolgáltatások szükségességét jelző változókat, stb.

DECODE - az előbbi szegmenssel együtt van a memóriában, a felhasználó szempontjából kényelmes formában megadott kapcsolási vázlat értelmezésekor van szerepe.

RUNPREP - csak szimulációs üzemmódban müködik, Kezdeti feltételeket tölt be, időzitéseket állit, az analóg programban statikus ellenőrzést végez, stb.

EXECUTION - ez a szegmens végzi a tulajdonképpeni szimulációt, ellenőrzi a felhasználói digitiális modell végrehajtását, időzitését, stb.

FUNPLOT - a szimuláció befejezése után az eredmények kirajzolását végzi.

STEPCOUNTER - lépésközbecslés esetén ez a szegmens következik a PREPARATION után. A RUNPREP helyett jön be, és megállapitja a szükséges lépésközt.

EBERLEIN - az előző szegmenssel együtt müködik, max. 50 x 50 -es négyzetes mátrixok sajátértékeit állitja elő, Eberlein módszerével.

MULTISTEP - csak stiff üzemmódban müködik, a RUNPREP szegmenssel együtt. Az un. stiff lista alapján az EXECUTION szegmens integráló algoritmusának stiff müködéséhez táblázatokat készit.

SPECFUNCTIONS és TASKS - felhasználói programozásu szegmensek.

1.3. Adatformátum

A bemenő adatok megadása a FORTRAN szabályai szerint történik. Ennek megfelelően az információ legkisebb egysége a rekord /egy kártya vagy a lyukszalagon két terminátor közötti karaktersor/, és a rekordon belül egy vagy több mező lehet. Minden mezőhöz egy szám /karaktermező esetén egy karakterlánc/ tartozik. A mezők szélessége a programban deklarálva van, azon a felhasználó nem változtathat. A program háromféle tipusu mezőt használ:

a./ Karaktermező. Az egyes feladatok azonositására a program egy 48 karakter szélességü mezőt használ, amely egyben mindig egy teljes rekord is. /Jele a táblázatokban: A48/

b./ Egész szám tipusu mező. Hossza kétféle lehet: négy vagy tiz karakter hosszu. /Jele: I4 illetve I10/ A mezőn belül az egyes karakterpozicióknak decimális helyiértékük van. Balról az első értékes számjegykarakter előtti betüközöket a program nem veszi figyelembe, a tőle jobbra lévők pedig nullaként értelmezettek. A csak betüközből álló "üres" mező értéke nulla. A betüköz karaktert b-vel jelölve tehát I4-es mezők esetében:

b1b5 = 105	b-1b = -10
1bbb = 1000	-b10 = -10
bbb1 = 1	bbbb = 0

c./ Fix formátumu tizedes törtmező. /Jele: F16.6/ A programban mindig 16 karakter széles. A betüköz karakter értelmezése ugyanugy történik, mint az egész szám tipusu mezők esetében. A szám értelmezése kétféle lehet:

- Ha a beolvasott mezőben van tizedespont karakter, akkor a szám értelmezése ennek megfelelően történik;
- Ha a beolvasott mezőben nincs tizedespont karakter, /formailag egész szám/, akkor a program a mező utolsó hat karakterét tekinti tizedeseknek.

A -204.051 szám megadásának néhány lehetősége tehát:

bb-b2b4.05	1bbbbb
-204.051bb	bbbbbb
bbb-bbb2b4	051bbb
←——→	←——→
10	6

Az egyes rekordokban szereplő mezők tipusa az adathordozón nincs megjelölve, az értelmezés attól függ, hogy a program milyen tipusu mezőt vár. Ezért az adathordozón lévő adatok sorrendje szigoruan kötött, az egyes adatokat a program a sorrend alapján értelmezi. Mindhárom tipusu mezőre még a következők érvényesek:

- Ha a beolvasott rekord hosszabb, mint a rekordban várt mezők együttes hossza, akkor a fölösleges karaktereket a program nem veszi figyelembe /elvesznek/;
- Ha a beolvasott rekord rövidebb, mint a várt mezők együttes hossza, akkor a hiányzó karaktereket a program betüköz karakterekkel pótolja /ld. betüköz értelmezése!/.

Ennek megfelelően tilos felesleges üres rekordok beiktatása, ennek hatására a várt mezők értéke nulla lesz, és a következő rekordtól már a következő adatot várja a program, tehát az adathordozón eltéved.

Az adatszerkesztési segédletben /I. Függelék/ minden adatnál fel van tüntetve a rekordformátum. Például 2I4, I10, 2F16.6 azt jelenti, hogy az adott rekordban három egész és két tizedes tört számot kell megadni, és az öt mező sorrendben a következő: 2 db I4-es, 1 db. I10-es, 2 db F16.6-os, tehát a szabályos rekord hossza 50 karakter.

1.4. Az adatok szerkesztésének szabályai

Inditás után a program "uj adatok, szimuláció" üzemmódban várja az első feladatra vonatkozó információkat. Az adathordozón minden egyes feladatot az azonositó rekord vezet be, és az ellenőrző rekord zár le.

Az azonositó rekord egyetlen A48 karaktermezőből áll. A beolvasott karaktersorozatot a program valamennyi használt kimeneti periférián változtatás nélkül kiirja. A program számára jelentősége nincs, csak a feladat outputjának azonositására szolgál.

Az ellenőrző rekord egyetlen I4 mezőből áll. Az általa lezárt feladatra nincs hatása, azt adja meg a programnak, hogy a lezárt feladat végrehajtása után mi a teendő. Értékétől függően három változat lehetséges:

- Ha az I4 mező értéke nulla, akkor a program STOP utasitásra fut;
- ha a mező értéke 9999, akkor a program "uj feladat, szimuláció" üzemmódba vált, tehát uj feladatot vár;
- ha a mező értéke a fentiektől eltérő, akkor a program az előzővel azonos modellre és azonos tipusu feladatra /lépésközbecslés, szimuláció/ vonatkozó módositó adatokra vár.

Csak az ellenőrző rekord segitségével tehát valamennyi engedélyezett üzemmódváltás nem valósitható meg. Segitségével leállithatjuk /0/ vagy alaphelyzetbe hozhatjuk /9999/ a programot, esetleg /egyéb értékkel/ ugyanannak a feladatnak módositott adatokkal történő ujrafuttatását kezdeményezhetjük.

Az adott feladatra vonatkozó adatok a fenti két nevezetes rekord között, blokkos formában helyezkednek el az adathordozón. A blokkok száma és felépitése az üzemmódtól függ.

2. Feladatmegadás szimuláció esetén

2.1. Adatmegadás "uj adat, szimuláció" üzemmódban

Ebben az üzemmódban az azonositó és az ellenőrző rekord között öt adatblokk helyezkedik el:

- hibrid program deklaráció
- analóg program koefficiensei
- analóg program kezdeti feltételei
- időzitések
- output specifikáció.

Az egyes blokkok a következő adatokat tartalmazzák:

2.1.a. Hibrid program deklaráció

- analóg gépi üzemmód deklaráció /F16.6/; a megadott szám abszolut értéke a program által szimulált analóg számitógép számitási egysége lesz /ld. 1. táblázat, U/. Amennyiben a fenti szám negativ, az integrátorok, összegzők, szorzók és osztók előjelet forditanak /ld. ugyanott, SG/.
- Felhasználói TASK-ok száma /I4/; az elemkészletből a felhasznált elemek számát az un. kötéslistából állapitja meg a program. Mivel a TASK-ok az analóg számitási vázlattól függetlenek, számukat külön kell megadni /TASK fogalmát ld. a 4.2. pontban/.

- Kötéslista hossza /I4/; a megadott szám abszolut értéke jelzi, hogy a soron következő lista hány rekordból áll. Ha a szám pozitiv, akkor a program normál szimulációt /esetleg lépésközbecslést/ fog végezni, és a kötéslistával a hibrid deklarációs blokkot befejezettnek tekinti. Ha a szám negativ, akkor a kötéslista után a stiff müködéshez szükséges adatok beolvasása következik.

- Kötéslista /I4, I1O/; egy rekord segitségével egy adott elem kimenetét egy adott elem adott bemenetére köthetjük. A listának tehát annyi elemből kell állnia, ahány kötéssel az analóg számitási vázlat leirható. Egy rekord formailag két számból áll /kimenet megjelölése - bemenet megjelölése/:

AABBbbbbCCDDEE
I4 I1O

Egy számitási elem azonositása a programban tipusa, és tipusán belül sorszáma segitségével történik. Ezért a kötéslista készitése előtt az elemeket tipusonként 1-től kezdődően, folyamatosan sorszámmal kell ellátni. A tipust kétjegyü szám /un. kódszám/ jelöli /ld. 1. táblázat/. Egyes elemeknek több bemenetük van, ezért az egyes bemeneteket külön sorszámmal jelöljük. A bemenetek sorszámozása is 1-től kezdődően történik. Azon három elemnél /szabad potenciométer, relé és osztó/, amelyeknél a bemenetek nem felcserélhetők, az 1. táblázat "Funkció" oszlopában a bemeneti x változók indexei egyben a megfelelő sorszámot is megadják. A kötéslista egy rekordjában lévő kétjegyü számok jelentése:

AA - a kimenetként megjelölt elem sorszáma;
BB - -"- -"- -"- kódszáma /tipusa/;
CC - a bemenetként megjelölt elem sorszáma;
DD - -"- -"- -"- kódszáma;
EE - a bemenet sorszáma /csak több bemenettel rendelkező elemnél/;

Például, ha az 5. integrátor kimenetét a 3. összegző 2. bemenetére kivánjuk kötni, akkor a kötéslistában valahol a következő rekordnak kell szerepelnie: /integrátor kódja OO, összegzőé O1/:

O5OObbbbO3O1O2

Ha egy elemnek több bemenete van, de nem mindegyik szerepel a kötéslista jobboldali /bemeneti/ oszlopában, akkor a nem használt bemenetekre kerülő érték automatikusan azonosan egyenlő nullával.

Ha a kötéslista rekordjának első, kimenetet deklaráló I4 mezője nulla értékü /pl. négy blank/, akkor a program a második mezőben deklarált bemenetet szintén az előző rekordban deklarált kimenetre köti. Értelemszerüen ilyen hiányos rekord nem szerepelhet a kötéslista első soraként.

- Stiff lista; beolvasására csak akkor kerül sor, ha a kötéslista hosszának megadása negativ számmal történt. E lista segitségével történik az analóg vázlat stiff csoportokra bontása. Csak a lassitott csoportokat kell deklarálni, a nem deklarált elemek automatikusan a leggyorsabb csoportba kerülnek. Tilos a stiff listában deklarálni:

  - bemeneteket /kód: 07/
  - holtidős elemeket /14/
  - D/A konvertereket /16/ és TASK-okat /17/
  - kimeneteket /19/

  A holtidős elemek automatikusan a legkisebb lassitási tényezővel rendelkező, a lassitott csoportok közül a leggyakrabban végrehajtott csoporttal együttesen kerülnek végrehajtásra. A bemenetek és kimenetek időzitését az "időzitések", ill. "output specifikáció" blokkokban kell majd megadni. A stiff lista alakja /minden rekord egyetlen I4 mezőből áll/:

n	/a lassitott csoportok száma/
TSF_1	/az első csoport lassitási tényezője/
DB_1	/az első csoportba deklarált elemek száma/
AABB	/csoport első elemének sorszáma és kódja/
:	
AABB	/utolsó, DB_1-edik elem sorszáma és kódja/
.	
.	
.	
TSF_n	
DB_n	
AABB	/u.a., az n-edik csoportra/
.	
.	
.	
AABB	

  A PROHYS programban a lassitott csoportok száma /n/ maximum 10 lehet. A gyors csoporttal együtt tehát az analóg vázlat legfeljebb 11 stiff csoportra bontható.

2.1.b. Az analóg program koefficiensei

- az integrátorok koefficiensei /4F16.6/; A lista annyi rekordból áll, ahány integrátor szerepelt a kötéslistában. A rekordok megadásának sorrendje az integrátorok növekedő sorszáma szerint történjék. Hasonlóképpen a rekordon belül is az első mező az integrátor első bemenetének koefficiensét adja meg, stb. Ha az analóg számitási vázlat nem tartalmaz integrátort, akkor ez a lista elmarad.

  Teljesen hasonló szabályok szerint történik a többi elem bemeneti koefficienseinek megadása is. A holtidős blokk koefficiensén a kivánt időkésleltetést kell érteni. A különböző tipusu elemek koefficienseit

megadó listák növekvő kódszám szerint következnek, tehát amennyiben a megfelelő tipusu elemek a kötéslistában szerepeltek, a koefficienseiket tartalmazó listákat a következő sorrendben kell megadni:

- az összegzők koefficiensei /4F16.6/;
- a potenciométerek koefficiensei /F16.6/;
- a szabad potenciométerek koefficiensei /F16.6/
- a holtidős blokkok időkésleltetései /F16.6/
- véletlenszám generátorok paraméterei /F16.6/

A véletlenszám generátorok paramétereit is kódszám, és azon belül sorszám szerinti sorrendben kell megadni. Az egyes generátoroknál a paraméter a következőket jelenti:

UNI és NOR generátorok esetében ha paraméterük nulla, akkor a generátorok alaphelyzetből indulnak, s igy minden egyes ujrafuttatásnál ugyanazt a számsort generálják /ld. 1. táblázat/;

PSN és EXR generátorok esetében paraméterként a Poisson, ill. exponenciális eloszlást jellemző lambda értéket kell megadni.

2.1.c. Az analóg program kezdeti feltételei

Az adatmegadás szabályai megegyeznek az előbbi blokkéval. Az egyes listák megadása növekvő kódszám, a listán belül az adatok megadása növekvő sorszám szerint történik, ha egy elemtipust nem használunk, a neki megfelelő lista elmarad. A listák tehát:

- integrátorok kezdeti feltételei /F16.6/
- bemenetek kezdeti értékei /F16.6/
- a holtidős blokkok kezdeti /és Δt ideig fennmaradó/ értéke /F16.6/
- a D/A konverterek kezdeti értéke /F16.6/

2.1.d. Időzitések

Ez a blokk tartalmazza a hibrid program végrehajtásával kapcsolatos valamennyi időzitési adatot. A következő listákból áll:

- Lépésköz lista hossza /I4/; Ez az egész szám adja meg, hogy a soron következő lista hány rekordból áll. Ha értéke nulla, azaz nincs lépésköz lista, akkor ez a program számára a lépésközbecslés üzemmódba való áttérést jelenti /ld. 3.1. pont/.
- Lépésköz lista /2F16.6/; Ez a lista tartalmazza, hogy a szimuláció különböző szakaszaiban milyen diszkretizációs lépésközöket kell alkalmazni. A rekordok első mezőjében a lépésköz kivánt értékét kell megadni, a második mezőben pedig azt, hogy az adott lépésköz meddig érvényes. Például, ha a szimulált folyamat időtartama 30 mp, ebből az első 5 mp alatt a lépésköz értéke 0.1 mp, az ezt követő 10 mp alatt 0.5 mp, az utolsó 15 mp alatt pedig 0.01 mp kell hogy legyen, akkor ez a lista 3 rekordból áll:

0.1	5.0
0.5	15.0
0.01	30.0

Amennyiben a programban holtidős blokkokat is alkalmazunk, akkor a lépésközök értéke nem választható meg teljesen szabadon. A szimulálni kivánt holtidőkre /Δt_i/ és a választott lépésközökre /h_i/ együttesen a következő feltételeknek kell teljesülniük:

a./ Valamennyi szimulálni kivánt holtidőnek a szimuláció során alkalmazott valamennyi lépésköz egész számu osztója kell hogy legyen. Ez a feltétel biztositja a holtidő egész számu pont eltárolásával történő megvalósitását.

b./ A szimuláció során alkalmazott valamennyi lépésköz egész számu többszöröse kell hogy legyen valamennyi nála kisebbnek. A program csak ebben az esetben képes a holtidőt reprezentáló tárat a lépésközváltások alkalmával megfelelően kiterjeszteni, illetve lecsökkenteni.

A lépésköz lista maximálisan 20 rekordból állhat. Stiff müködés esetén a megadott lépésközök a leggyorsabb /nem lassitott/ csoportra érvényesek.

- Bemenet változtatási lista /I4/; Valamennyi, a kötéslistában szereplő bemenetre egész szám formájában meg kell adni, hogy az adott bemeneten megjelenő értéket hányszor kivánjuk a szimuláció során megváltoztatni. Az egyes rekordoknak a bemenetek növekvő sorszáma szerint kell követniük egymást. Ha a szám nulla, akkor a kezdeti feltételek blokkjában megadott érték a szimuláció befejezéséig változatlan marad. A bemeneteket a szimuláció során maximum 20 alkalommal lehet változtatni.
- Bemenet változtatási adatlista /2F16.6/; Az egyes változtatások adatait tartalmazza. Az első mező az időpontot határozza meg, a második mező pedig a megjelentetni kivánt értéket. Egy rekord egy változtatást határozhat meg. A változtatási adatlista rekordjainak a bemenetek növekvő sorszáma szerint, azon belül pedig időrendi sorrendben kell követniük egymást. Ha például az analóg programban három bemenet szerepel, amelyik közül az elsőt háromszor, a harmadikat egyszer kell megváltoztatni, azaz a bemenet változtatási lista a következő volt:

3
0
1

és az első bemenetet az 5., 10. és 20 másodpercben 0-ra, -1-re, illetve -2-re kell változtatni, és a harmadikat a 8. másodpercben +5-re, akkor a változtatási adatlista a következő lesz:

5.0	0.0	⎫
10.0	-1.0	⎬ /első bemenet/
20.0	-2.0	⎭
8.0	5.0	/harmadik bemenet/

- TASK időzitési lista /3F16.6/; Annyi rekordból áll, ahány TASK-ot a hibrid deklarációs blokkban megadtunk. A rekordokat a TASK-ok növekvő sorszám szerinti sorrendjében kell megadni. Az első mező a TASK első belépésének időpontját, a második a periodikusan belépő TASK két végrehajtása között eltelt időt, a harmadik az utolsó időpontját határozza meg. Megkötés, hogy a periódusként megadott szám nulla nem lehet. Ha tehát egy TASK-ot az egész szimuláció során csak egyszer kivánunk lefuttatni, akkor az első és utolsó belépés időpontjául ugyanazt a számot kell megadni, a periodicitás értéke ilyenkor nullától eltérő bármilyen szám lehet.

Megjegyzés: Stiff müködés esetén a bemenetek változtatása, a TASK-ok végrehajtása csak a leggyorsabb lassitott csoporttal együttesen történhet, a stiff csoport /nem lassitott/ végrehajtása során ugyanis ezen feladatok aktualitását a program nem vizsgálja. Ha tehát az általunk megjelölt időpontban csak a stiff csoport müködik, akkor a program a feladat végrehajtását elhalasztja a leggyorsabb lassitott csoport legközelebbi belépéséig. Ugyanez érvényes a sornyomtató és perforátor deklarációra.

2.1.e. Output specifikáció

Ez a blokk tartalmazza az eredmények megjelenitésére vonatkozó adatokat. A blokk a következő rekordokból áll:

- Sornyomtató deklaráció /I4/; Az itt megadott egész szám közli a programmal, hogy minden hanyadik számitási ciklus eredményeit kell kiiratni. Üres rekord esetén valamennyi eredmény kiiratódik.
- Perforátor deklaráció /I4/; Ha az itt szerepelő szám nullától eltérő pozitiv egész, akkor az eredmények lyukszalagra /kártyára/ is kiiratódnak. Ezen a periférián csak az eredmények jelennek meg, fejléc, üzenetek, stb. nélkül. A megadott szám a gyakoriságot jelöli, ugyanugy, mint a sornyomtató esetében. Ha a szám nulla /vagy a rekord üres/, akkor eredmény a perforátoron nem jelenik meg. A számitógépes feldolgozás megkönnyitése érdekében a lyukszalag első három rekordja /illetve az első három kártya/ a következő adatokat tartalmazza:
  - a job azonositó neve /A48/;
  - a rekordokban szerepelő adatok száma /I4/ → idő + kimenetek;
  - az utolsó rekord első mezőjében lévő időadat /F10.3/; ezen adat nem felel meg a valóságnak, ha a futást a tervezettnél korábban az un. ALARM szakitja meg.

  Az eredményeket tartalmazó rekordok formátuma azonos a sornyomtatóéval /F10.3, nF11.5/ - ahol n a kimenetek száma.
- Plotter deklaráció /I4/; Az itt megadott egész szám jelöli, hányadik kimenet értékeit kell az idő függvényében plotteren ábrázolni. Ha a szám értéke nulla, akkor a rajzolás, és a következő rekord megadása elmarad.

- Plotter skálázás /3F16.6/; A három szám a koordináta-tengelyek végértékeit adja meg, sorrendben: X_{max}, Y_{max}, Y_{min}. Az X tengely /idő/ kezdőértéke mindig nulla. Amennyiben $Y_{min} \neq 0$, a rajzolóprogram behuzza az Y=0 egyenest is. Ha a megadott értékek közül $X_{max}=0$, akkor a program az X tengely skálázását, $Y_{max}=Y_{min}=0$ esetén az Y tengely skálázását is automatikusan elvégzi. A rajz ilyenkor a teljes folyamatot ábrázolja.
- Alarm stop deklaráció /I4/; Ha az itt megadott érték nulla, akkor a szolgáltatás és a következő rekord megadása elmarad. Ha a megadott szám valamely kimenet sorszáma, akkor a program a szimulációt a lépésköz listában megadott időadatoktól függetlenül befejezi, ha a megjelölt kimenet értéke kilép a következő rekord segitségével megadott sávból. E szolgáltatás segitségével a futás előre programozottan befejezhető, ha a figyelt változó a szimuláció szempontjából érdektelen tartományba jut, vagy numerikus instabilitás lép fel, stb.
- Min-max alarm szint /2F16.6/; Az érvényességi sáv alsó és felső határértékét tartalmazza.
- TASK Monitor /I4/; A program lehetőséget nyujt az e rekordban megadott mennyiségü - akár valamennyi - TASK müködésének nyomon követésére. Ha egy ilyen figyelt TASK végrehajtódik, akkor erről a program a sornyomtatón üzenetet küld, és ezután valamennyi felhasznált D/A konverter értékét kiirja - ezeken keresztül kapcsolódhat ugyanis egy TASK az analóg számitási vázlathoz /ld. 4.2 pont/. Ha a TASK Monitor rekordban megadott szám értéke nulla, akkor a nyomkövetés, és a soron következő lista beolvasása elmarad. Magát a TASK Monitor rekordot sem kell az output specifikációban megadni, ha a hibrid deklarációs blokkban a felhasználó TASK-ok számaként nullát adtunk meg.
- TASK Monitor lista /I4/; Hosszát az előző rekord /TASK Monitor/ adta meg. Az egyes rekordokban a figyeltetni kivánt TASK-ok sorszámai szerepelnek. A rekordok sorrendje közömbös.

A program "uj adat, szimuláció" üzemmódjában tehát ez az öt adatblokk helyezkedik el a job-ot megnyitó azonositó név, és a lezáró ellenőrző rekord között.

2.2. Adatmegadás "módositó adat, szimuláció" üzemmódban

Ha egy szimulációs feladatot - néhány adat módositásával, variálásával - egymás után többször el akarunk végeztetni, akkor jelentős lyukasztási munkát takarithatunk meg a program "módositó adat, szimuláció" üzemmódjának felhasználásával. Elegendő ugyanis a program "uj adat, szimuláció" üzemmódjában egyszer megadni és módositásra utaló ellenőrző kóddal /kód ≠ 0 és ≠ 9999/ lezárni a teljes adatsort. Ebben az esetben a feladat egyszeri megoldása után csak a módositó adatokat várja a program az ujrafuttatás előtt.

A módositó adatokat szintén azonositó rekord vezeti be és ellenőrző rekord zárja le. Az ellenőrző kód - szükség esetén - ujabb módositást irhat elő, s igy segitségével egy szimulációs feladat tetszés szerinti számu ujrafuttatása végeztethető el, a PROHYS program egy futásával, ujrainditás nélkül.

A módositás során minden megváltoztatható a hibrid program deklaráción, a felhasználói programozásu TASK-okon és függvénygenerátorokon kivül. A módositásokra vonatkozó adatokat - az előbbiekben ismertetett üzemmódhoz hasonlóan - szintén blokkokba tömöritve kell a programnak megadni. Mivel a hibrid deklaráció nem módositható, az azonositó rekord és az ellenőrző rekord között itt csak négy blokk van:

- az analóg koefficiensek módositásai;
- a kezdeti feltételek módositásai;
- az időzitésekre vonatkozó módositások;
- az output specifikáció módositása.

A négy blokk felépitése azonos. Valamennyi a blokkfejből és a módositásokból áll. A blokkfej egy I4 formátumu rekord, amely a blokkban lévő módositások számát adja meg. Ha értéke nulla, akkor a blokk üres, azaz kizárólag a blokkfejből áll.

Minden egyes módositás az elemdeklarációból és a módositó adatból áll. Hatására a megjelölt elemnek a blokk tipusa által meghatározott adata /koefficiense, kezdeti feltétele stb./ megváltozik, és felveszi a módositó adat értékét.

Az elemdeklaráció egyetlen I4 tipusu, AABB formáju rekordból áll, ahol AA az elem sorszáma, BB a kódja. A harmadik blokkból a lépésköz lista, és a negyedik blokk összes adata azonban nem kapcsolható egyes elemekhez. Ezért a programban a lépésköz lista módositását a OOOO elemdeklarációval lehet kezdeményezni /a "nulladik integrátor" megnevezésével/. A negyedik blokkba tartozó adatokat azonban nem lehet egyenként módositani. A negyedik blokkra a következők érvényesek:

- Ha a blokkfej nulla, akkor a blokk üres és az ujrafuttatás során a régi output specifikáció érvényes;
- ha a blokkfej nullától eltérő, akkor utána a teljes output specifikációt meg kell ismételni, a 2.1.e pontban leirt szabályok szerint.

Az elemdeklarációt a módositó adat követi, ami lehet egy rekord, vagy egy összetett lista. A formátumok ugyanazok, mint "uj adat, szimuláció" üzemmódban. Tehát a módositó adat egyetlen rekord

- valamennyi koefficiens módositásakor /4F16.6 az integrátorok és összegzők esetére, F16.6 valamennyi egyéb elemre/;
- valamennyi kezdeti feltétel módositásakor /F16.6 valamennyi elemre/;
- a TASK időzitésének módositásakor /3F16.6/;

és összetett lista a következő adatok módositásakor:

- lépésköz lista;
- bemenet értékváltoztatási lista.

A lépésköz lista módositásánál a OOOO elemdeklarációt a következő módositó adatnak kell követni:

- az uj lépésköz lista hossza /egy I4 formátumu rekord/
- az uj lépésköz lista /az előző rekordban megadott számu, 2F16.6 formátumu rekord/.

A bemenet értékváltoztatási lista módositásakor a módositó adat felépitése a következő:

- az elemdeklarációval meghatározott bemenetet a szimuláció során hányszor kell változtatni /I4 rekord/;
- az egyes változtatások idő- és értékadatai idősorrendben /az előző rekord által meghatározott számu, 2F16.6 formátumu rekord/.

A módositás szabályainak használatát vizsgáljuk egy példán. Legyen a feladat egy szimuláció megismétlése a következő módositásokkal:

- a 3. és 4. potenciométer koefficiense legyen 0.82;
- a 2. integrátor kezdeti feltétele legyen 0;
- a szimuláció menete: 0.5 sec lépésközzel a 10. másodpercig, azután 0.1 sec lépésközzel a 30. másodpercig;
- az 1. bemenetet kétszer kell változtatni: a 2. mp-ben 0.1-re, és az 5. mp-ben 2.0-ra;
- az output legyen ugyanolyan, mint az előző feladatban;
- az ujrafuttatás után már egy másik rendszer szimulációja lesz a feladat.

Az 1. táblázat alapján a kódok: potenciométer: 03;
integrátor : 00;
bemenet : 07;

Az adathordozó megfelelő részlete a következő lesz:

```
   .
   .                    /az előző feladat adatai
   .
    3                   /az előző feladat ellenőrző kódja; ≠ 0 és ≠ 9999,
                                  tehát módositás;
ELSŐ MÓDOSITÁS          /az azonositó rekord az adott feladatra;
    2                   /első blokkfej: két koef. módositás;
0303                    /3. potm. elemdeklaráció   ]
0.82                    /  -"-    módositó adat    } 1. módositás
0403                    /4. potm. elemdeklaráció   ]
0.82                    /  -"-    módositó adat    } 2. módositás
    1                   /második blokkfej: egy kezd. felt. módositás;
0200                    /2. integrátor elemdeklaráció          ] 1. módosi-
0.0                     /  -"-  módositó adat: uj kezd. felt.  ] tás
```

```
   2                    /harmadik blokkfej: két időzités módositás;
0000                    /a lépésköz lista "elemdeklarációja";          ┐
   2                    /   -"-     -"- hossza┐                        │
0.5        10.0                               } módositó adat;        } 1. módositás
0.1        30.0         /a lista két eleme    ┘                        │
0107                    /a bemenet elemdeklarációja                    ┐
   2                    /hányszor kell változtatni┐                    │
2.0        0.1                                   } mód. adat;         } 2. módositás
5.0        2.0          /mikor és mire           ┘                     │
   0                    /negyedik blokkfej: nincs módositás az outputot
                                illetően
9999                    /a következő feladat uj adatokkal történő szimu-
                                láció lesz; ez a lezáró ellenőrző kód;
A KÖVETKEZŐ MODELL
                        /a köv. feladat azonositó rekordja
    .
    .                   /köv. feladat adatai
    .
```

E szimulációs feladat módositott adatokkal történő ujrafuttatásához tehát mindössze 20 rövid sort kell az adathordozóra lyukasztani az adott esetben.

3. Feladatmegadás lépésközbecslés esetén

3.1 Áttérések "lépésközbecslés" üzemmódba

Az eddigiek során áttekintettük a PROHYS szimulációs üzemmódjának használatát. Szimulációs feladatok esetén az azonositó név és az ellenőrző kód között az adatok uj feladat esetén öt, módositott adatokkal történő ujrafuttatás esetén négy blokkban helyezkednek el. Az ellenőrző kóddal a következő feladat adatainak tipusát /uj, módositó/ lehet meghatározni.

Lépésközbecslés esetén is a program az adatok beolvasását szimulációs üzemmódban kezdi. Uj feladat esetén az első három adatblokk beolvasása /hibrid program deklaráció, koefficiensek, kezdeti feltételek/, szimulációs feladat módositása esetén az első két blokk módositása /koefficiensek, kezdeti feltételek/ ugyanugy történik. Ezután következne az időzitések adatainak beolvasása, illetve módositása.

Ha az "időzitések" blokk első adatának, azaz a lépésköz lista hosszát deklaráló adatnak nulla értéket adunk /illetve erre az értékre módositjuk/, akkor a program azonnal áttér lépésközbecslés üzemmódba, és beolvassa az un. I. linearizációs blokk és a II. linearizációs blokk adatait. Ezek tartalmazzák a vizsgált rendszer állapotmátrixának összeállitásához szükséges adatokat.

Ha a program egy korábbi feladat során már áttért lépésközbecslés üzemmódba, akkor a további módositások során már nincs szükség a lépésköz lista értékének ujabb nullázására. Ebben az esetben az adatok módositása - a szimulációs üzemmódhoz teljesen hasonlóan - négy blokkban történik, amelyek a következők:

- analóg program koefficiensei
- analóg program kezdeti feltételei
- I. linearizálási blokk /időzitések helyett/
- II. linearizálási blokk /output specifikáció helyett/

A harmadik és negyedik blokk felépitése ugyanolyan, mint az első két blokké /blokkfej, elemdeklaráció, módosító adat, stb./; a linearizációs blokkok módosító adatai között nincs összetett lista jellegü, valamennyi egyszerü rekord /ld. linearizálás/.

Az adatbeviteli rendszer folyamatábrája az 1. ábrán látható.

3.2. Linearizálás

A numerikus integrálás lépésközének meghatározását a program a vizsgált rendszer állapotmátrixa alapján végzi [1]. Ezért, ha a rendszer nemlineáris, akkor egy adott munkapont környékén linearizálni kell.

Mivel a bemenetek, a holtidős blokkok, a D/A konverterek, a véletlenszám-generátorok egy integrálási ciklus alatt nem változtatják értékeiket, a Runge-Kutta egylépéses eljárás szempontjából a következő elemek tekinthetők nemlineárisnak:

- szorzók;
- osztók;
- relék;
- belső függvények /arctan, sin, cos, exp, log, abs/
- felhasználói programozásu függvénygenerátorok.

A fenti elemek linearizálása a következők szerint történik:

- a szorzók és osztók egyik /a felhasználó által kiválasztott/ bemenetére változó helyett konstans érték kerül, s igy ezen elemek mintegy potenciométerré alakulnak;
- a relék "érintkezői" a falhasználó által megadott helyzetben rögzitődnek;
- a felhasználói és belső függvénygenerátorokat állandó erősitésü, lineáris átvitelü elemekkel helyettesiti a program /szintén potenciométerekké alakulnak/.

Az itt felsorolt átalakitások elvégzéséhez szükséges adatokat a linearizációs blokkok tartalmazzák.

3.3. Adatmegadás lépésközbecslés üzemmódban

Lépésközbecslés esetén az időzitések és output specifikáció blokkokat az I. és II. linearizációs blokk helyettesiti. Ezek a következő adatokat tartalmazzák:

3.3.a. I. linearizációs blokk

- a szorzók linearizálási adatai /I2, 6X, F16.6/; Az első egész szám csak 1 vagy 2 lehet, és a szorzó kiválasztott bemenetét jelöli. Erre a bemenetre konstans értékként kerül a F16.6 mezőben megadott valós szám. Ha a megadott konstans értéke nulla, akkor a megjelölt bemeneten a t=0 időpillanatban fellelhető értéket rögziti a program, ebben az esetben a linearizálás a kezdeti feltételek által meghatározott munkapontban történik. Az I2 és az F16.6 mezők között lévő 6X azt jelenti, hogy e mezők között hat karaktert vár a program, amelyeket beolvasás után figyelmen kivül hagy. Eredetileg a nyolcas tabuláció miatt került a rekordba, de felhasználható megjegyzésre /változó neve, stb./. A listának növekvő sorszám szerinti sorrendben kell tartalmaznia valamennyi felhasznált szorzóra vonatkozó rekordokat.
- az osztók linearizálási adatai /I2, 6X, F16.6/; E lista adatait teljesen a szorzók linearizálási adataira vonatkozó szabályok szerint kell összeállitani. Ha az analóg kapcsolási vázlatban nincs szorzó, vagy osztó, akkor a megfelelő lista is elmarad.
- a relék linearizálási adatai /F16.6/; Ha a megadott valós szám nulla, akkor a relé helyzetét a kezdeti feltételek határozzák meg; pozitiv szám az $x_1 \geq x_2$, negativ szám az $x_1 < x_2$ esetnek felel meg. A listának itt is tartalmaznia kell valamennyi relé adatait, növekvő sorszám szerinti sorrendben.

3.3.b. II. linearizációs blokk

- a belső függvénygenerátorok linearizációs adatai /F16.6/;
Valamennyi belső függvénygenerátort a linearizálás során egy erősitési tényező /potenciométer/ helyettesiti. Ezt az erősitési tényezőt kell megadni. Ha nullát adunk meg, akkor az erősitési tényezőt a program a kezdeti feltételek alapján határozza meg, a következő képlet alapján:

$$A = \frac{f/x_b/}{x_b} ;$$

ahol f/x/ az adott belső függvény, x_b pedig a t=0 időpillanatban a bemenetére kerülő érték. A listának a különböző függvénygenerátorok munkaponti "erősitését" növekvő kódszám /arctan: 09, sin: 10, cos: 11, stb./ szerinti sorrendben, az egyes függvénytipusokon belül pedig növekvő sorszám szerinti sorrendben kell tartalmaznia. A fel nem használt tipusok a listából természetesen kimaradnak.

- Felhasználói programozásu függvénygenerátorok /F16.6/;
  Ugyanugy erősitési tényezőkkel kell őket helyettesiteni, mint a belső függvények esetében. Az egyetlen nemlineáris elemtipus, ahol implicit értékmegadásra /a kezdeti feltételek alapján/ nincs mód. Lépésközbecslés során ugyanis e programrészeket nem lehet végrehajtani, s igy a kimenet/bemenet arány megállapitása sem lehetséges. Ha nullát adunk meg, akkor ez esetben ez nulla átviteli tényezőnek felel meg. A rekordokat szintén növekvő sorszám szerinti sorrendben kell megadni.

Megjegyzés: A lépésközbecslés során - az állapotmátrix összeállitásakor - valamennyi véletlenszám generátor kimenetén egy konstans szám, az általuk szolgáltatott zaj várható értéke /elsőrendü momentuma/ jelenik meg.

4. A hibrid program digitális részének megadása

Az eddig leirtak tartalmazzák az adatszalag szerkesztési szabályait. Az adatszalag segitségével adható meg tehát:

- a hibrid program deklarációja;
- az analóg rész koefficienseivel és kezdeti feltételeivel;
- a numerikus integrálást vezérlő adatok /stiff, lépésköz/;
- a PROHYS vezérléséhez szükséges információ /output, időzitések/;

A hibrid program digitális része viszont a futás inditásakor már be kell hogy legyen épitve. A FORTRAN programnyelv tulajdonsága, hogy az egyes programszegmenseket külön-külön le lehet forditani, és egy szerkesztőprogram segitségével e leforditott szegmenseket futásra kész célprogrammá lehet konszolidálni. Ezért a felhasználói programozásu funkciók számára a PROHYS programban külön szegmensek vannak fenntartva. /SPECFUNCTIONS és TASK szegmensek/. Módositás, javitás, stb. alkalmával elég tehát a megváltoztatott felhasználói szegmenseket ujraforditani, s igy ezeket a PROHYS többi, már leforditott és könyvtárba felvett szegmensével összekonszolidálva nyerhető az uj célprogram.

4.1. A felhasználói függvénygenerátorok programozása

E függvénygenerátorokban a felhasználói programnak a bemenet, az idő és az éppen használt lépésköz ismeretében elő kell állitania a kimenet értékét. Az "üres" SPECFUNCTIONS szegmens a PROHYS programban a következő formában szerepel:

```
      SUBROUTINE SPECFUNCTIONS (TIME, STEP, NUM, VALIN, VALOUT)
      GO TO (1,2,3,4,5,6,7,8,9,10), NUM
1     VALOUT=VALIN
      RETURN
2     VALOUT=VALIN
      RETURN
      .
      .
```

```
10    VALOUT=VALIN
      RETURN
      END
```

ahol TIME a szimulált idő, STEP a lépésköz aktuális értéke. A szegmens aktivizálásakor NUM adja a végrehajtható függvény sorszámát, VALIN a bemenetén lévő értéket. A GO TO utasitás hatására a vezérlés megfelelő, sorszámot jelölő cimkére adódik, ahol VALOUT változónak /amely a kimenet értékét hordozza/ értéket kell adni. A VALOUT=VALIN értékadó utasitást kell tehát a felhasználónak saját programjára kicserélnie.

A SPECFUNCTIONS szegmensbe elhelyezett programok az analóg rész szempontjából "folyamatos függvényt" képviselnek, és kiszámitásukra minden egyes integrálási ciklus alkalmával négyszer kerül sor. Ezért ügyelni kell arra, hogy ezek a felhasználói programrészletek futási idő szempontjából optimalizálva legyenek.

4.2. A TASK fogalma, lehetőségei és beépitése

A hibrid program digitális részét a TASK-ok képezik. Ezek olyan egymástól független programrészletek, amelyek előre meghatározható időintervallumban, megadott időközönként aktivizálódnak. Elérhetik és megváltoztathatják az analóg számitási vázlat valamennyi pontján található értékeket. Adatokat olvashatnak be különböző perifériákról, és eredményeket, üzeneteket irhatnak ki.

A TASK-okat a felhasználónak FORTRAN nyelven kell megirnia, és a TASKS szegmensbe elhelyeznie. Összesen 20 db TASK beépitésére van lehetőség. A TASK-ok sorszáma egyben a prioritást is jelzi: ha egyidejüleg több TASK beléptetése esedékes, akkor ezek növekvő sorszám szerinti sorrendben kerülnek végrehajtásra.

A TASK-ok végrehajtása alatt az analóg számitási idő "áll". Tehát a szimulált analóg számitógép számára az egyes TASK-ok "futási ideje" nulla. Reális futási idő szimulációjáról a felhasználónak kell gondoskodnia /pl. a TASK holtidős blokkon keresztül avatkozik be, stb./, s igy a TASK futási ideje kézben tartható, független az alkalmazott számitógép tipusától.

Az analóg számitási vázlat különböző pontjaihoz a TASK a következő módon férhet hozzá:

Valamennyi számitási elem kimenetéhez hozzá van rendelve a programban a VAL vektor valamelyik eleme. Ez egyben azt is jelenti, hogy valamennyi bemenet is szerepel valahol a VAL vektorban, hiszen minden használt elem bemenete egy másik számitási elem kimenetére van kötve. Azt, hogy egy számitási elem bemenetén vagy kimenetén lévő érték hol található a VAL vektorban, az un. mátrix-pointerek határozzák meg. /Ha egy bemenetet nem használunk, akkor a

mátrix-pointer a VAL vektor első elemére mutat, amelyben azonosan nulla van./

A keresett mátrix-pointer nevét az M betü és az elemtipus mnemonikus nevének összeállitásával kapjuk. /Mnemonikus név: integrátor - INT, összegző - - SUM, stb., lásd 1. táblázat./ Igy az integrátorok helyzetére a MINT, az összegzőkére a MSUM, stb. mátrix pointerek mutatnak. A mátrix-pointer neve tehát a tipust határozza meg. A mátrix sorindexe viszont a poziciószámot, oszlopindexe pedig a sorszámot határozza meg. Poziciószáma csak azoknak az elemeknek van, amelyek kimenettel és bemenettel is rendelkeznek. A többi elem /bemenet, kimenet, D/A konv., stb/ esetében a sorindex elmarad, tehát mátrix--pointer helyett vektor-pointerük van.

A poziciószám:
1; → ha egybementü elem bemenetén lévő értéket keressük;
a bemenet sorszáma; → több bemenetü elemnél;
a bemenetek száma +1; → ha a kimenetet keressük.

Tehát például, ha az 5. integrátor bemenetein és kimenetén lévő értékeket keressük, akkor:

MINT (1,5) - az első bemenet helyére mutat a VAL vektorban;
MINT (2,5) - a második -"- -"- -"- ;
MINT (3,5) - a harmadik -"- -"- -"- ;
MINT (4,5) - a negyedik -"- -"- -"- ;
MINT (5,5) - az integrátor kimenetére mutat a VAL vektorban.

Ha például valamelyik TASK-ban RO2 változó értékének fel kell vennie a 6. szabad potméter /mnemonikus kódja: FPT/ kimenetének értékét, ZBA változó pedig a 4. analóg bemeneten /mnemokikus kódja: INP/ lévő értéket, akkor ezt a következő FORTRAN utasitássorozattal érhetjük el:

```
ITEMP = MFPT(3,6)
RO2 = VAL(ITEMP)
ITEMP = MINP(4)
ZBA = VAL(ITEMP)
```

ugyanis egy szabad potenciométernek két bemenete van, tehát kimenetének poziciószáma 3; egy bemenet pedig az analóg számitási vázlat szempontjából csak kimenettel rendelkező elem /amelyre különböző más elemek bemenetei lehet kötni/, s ezért poziciószáma nincs. ITEMP pedig egy egész tipusu munkarekesz. Ha a használt FORTRAN gépi reprezentáció megengedi a többszörös indexelést, két utasitással is célt érhetünk:

```
RO2 = VAL(MFPT(3,6))
ZBA = VAL(MINP(4))
```

Az analóg kapcsolási vázlatban forditott irányu értékadó utasitásokkal avatkozhatunk be. Mivel a számitási elemek kimeneteire és bemeneteire adott értékek a következő integrálási ciklus során felüliródnak, érdemes a beavatkozá-

sokat a D/A konverter blokkokon keresztül végrehajtani, ezek az analóg vázlat szempontjából bemeneteknek számitanak, s igy értékük megmarad. Ezen kivül mód nyilik a TASK müködésének nyomon követésére /lásd 2.1.e pont, TASK Monitor/. Ha például az 5. D/A konverternek egy előzetesen kiszámitott BC változó értékét kivánjuk adni, ez a következő utasitással történhet: /mnemonikus kód: DAC/;

VAL(MDAC(5)) = BC

A fenti rendszer alól kivételt képeznek a véletlenszám generátorok, amelyek csak kimenettel rendelkező elemek, s igy vektor-pointerrel kellene rendelkezniük. Ennek ellenére mátrix-pointerük van, amelynek kiosztása a következő:

MNOI (1,n) - UNI tipusu /fehér zaj/ generátorok kimeneteire mutat;
MNOI (2,n) - NOR tipusu /Gauss-el./ -"- -"-
MNOI (3,n) - PSN tipusu /Poisson/ -"- -"-
MNOI (4,n) - EXR tipusu /Exp. elo./ -"- -"-

ahol n a generátor sorszáma. Itt tehát a sorindex nem a pozicióra, hanem a tipusra utal.

Szükség lehet arra is, hogy az egyes TASK-ok müködését ne csak időpillanatokhoz, hanem egyéb feltételekhez is kössük. Hogy a felhasználónak szabad kezet nyujtson, a PROHYS nem tartalmaz ilyen irányu direktivákat. Ilyenkor a következőképpen lehet eljárni:

A legmagasabb prioritásu /1. sorszámu/ TASK-nak adjunk meg olyan periodicitást, hogy valamennyi többi TASK belépése esetén az 1. TASK is belépjen. Megvizsgálva a prioritást, a PROHYS először ennek adja a vezérlést, s igy ez a TASK megvizsgálhatja a soron következő müködési feltételeit. Esetleg készithető a TASKS szegmensben olyan "supervisor" programrészlet, amelynek valamennyi TASK beléptetése esetén átadja a vezérlést, s igy ott a müködés feltételei megviszgálhatók. Az a tény, hogy valamennyi TASK egy FORTRAN szegmensben szerepel, s igy az egyik értéket adhat a másik változóinak, a programozó számára gazdag lehetőségeket biztosit.

Az "üres" TASKS szegmens alakja a PROHYS programban:

```
      SUBROUTINE TASKS (TIME, STEP, NUM)
      DIMENSION VAL (409), . . . .
      COMMON /VALUE/VAL. . . . . .
      GO TO(1,2,3, . . . , 20), NUM
1     CONTINUE
      RETURN
2     CONTINUE
      RETURN
      .
      .
```

```
20    CONTINUE
      RETURN
      END
```

A TIME,STEP és NUM változók ugyanazok, mint a SPECFUNCTIONS szegmens esetében. A DIMENSION és COMMON utasitások biztositják a hozzáférést a VAL vektorhoz, és a mátrix-pointerekhez. Ha egy TASK aktivizálódik, akkor a sorszámának megfelelő cimkére adódik a vezérlés. A felhasználónak a TASK-ját a CONTINUE utasitás helyett kell beirnia.

Ha a felhasználó a TASK-kal adatokat beolvastatni, vagy eredményeket, üzeneteket kiiratni akar, akkor a PROHYS logikai perifériaszámait kell használnia /ld. 1.1. pont/.

5. A PROHYS futása során előforduló hibák és következményeik

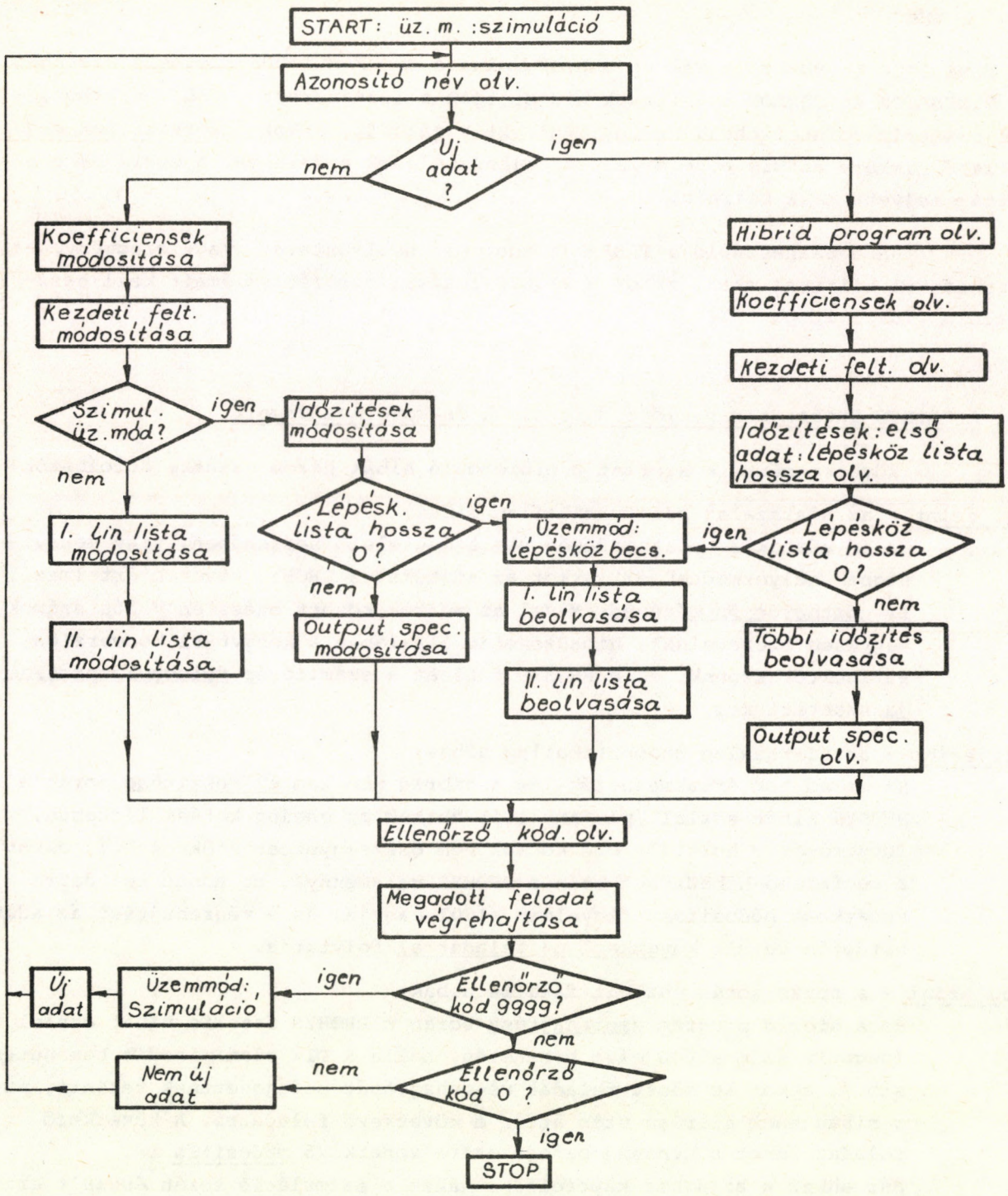
Következményeik szerint a előforduló hibák három szintbe sorolhatók:

1. szint - az adatszalag szintaktikailag hibás;
Ha az adatok az adathordozón nem a megfelelő sorrendben, vagy mennyiségben helyezkednek el, akkor az adatokat a PROHYS tévesen értelmezi, az adathordozón eltéved. Mivel az adatok között egész és valós számok vegyesen szerepelnek, hibaüzenetet általában a könyvtári konverziós rutinoktól kapunk, és a PROHYS futását a számitógép felügyelő programja szünteti meg.

2. szint - az adatszalag szemantikailag hibás;
Ha az adatok értelmezhetők, de a hibrid program előkészitése során a PROHYS hibát észlel /pl. implicit hurkok az analóg kötési listában, időzitések a holtidős blokkokkal nem összeegyeztethetők, stb./, akkor a megfelelő hibaüzenet után a PROHYS valamennyi, az adott feladatra vonatkozó módositást figyelmen kivül hagyja, és a végrehajtást az adathordozón talált következő uj feladattal folytatja.

3. szint - a futás során észlelt fatális hibák;
Ha a hibrid program végrehajtása során a PROHYS fatális hibát észlel /negativ szám a LOG elem bemenetén, nulla a DIV elem második bemenetén, stb./, akkor az adott feladat végrehajtását befejezettnek tekinti, és a hibaüzenet kiirása után áttér a következő feladatra. A következő feladat lehet a hibásan befejezettre vonatkozó módositás is.
Bár ehhez a szinthez kapcsolódhatnának a szimuláció során észlelt aritmetikai tulcsorgások, stb. is, ezeket azonban először rendszerint a könyvtári rutinok észlelik, amelyek a számitógép felügyelő programjának adják át a vezérlést. Ezért az aritmetikai hibák az első szinthez kapcsolódnak.

A 2. és 3. szintü hibák előfordulása esetén a PROHYS hibaüzenetet generál; ezek felsorolása a II. függelékben található.



1. ábra Az input rendszer folyamatábrája

ELEMKÉSZLET

ELEM	KÓD	FUNKCIÓ	MNEM.	DB	BE	KOEF	IC	KI	SG
Integrátor	00	$y=SG[y_o+\int\sum_i a_i x_i dt]$	INT	50	4	4	P	1	P
Összegző	01	$y=SG\sum_i a_i x_i$	SUM	50	4	4	-	1	P
Inverter	02	$y=-x$	INV	20	1	-	-	1	-
Potenciométer	03	$y=ax$	POT	100	1	1	-	1	-
Szabad potméter	04	$y=ax_1+(1-a)x_2$	FPT	10	2	1	-	1	-
Szorzó	05	$y=SG\cdot x_1\cdot x_2/U$	MPY	20	2	-	-	1	P
Relé	06	$y=x_3$; $x_1\geq x_2$; $y=x_4$; $x_1<x_2$;	REL	10	4	-	-	1	-
Bemenet	07	y=const /prog./	INP	10	-	"20"	P	1	-
Osztó	08	$y=SG\cdot U\cdot x_1/x_2$	DIV	20	2	-	-	1	P
Arc tan	09	$y=U\cdot\arctan(x/U)$	ARC	10	1	-	-	1	-
Sinus	10	$y=U\cdot\sin(x/U)$	SIN	10	1	-	-	1	-
Cosinus	11	$y=U\cdot\cos(x/U)$	COS	10	1	-	-	1	-
Exp.	12	$y=U\cdot\exp(x/U)$	EXP	10	1	-	-	1	-
Term.logaritmus	13	$y=U\cdot\ln(x/U)$	LOG	10	1	-	-	1	-
Holtidő	14	$y(t)=x(t-t_1)$	DTB	20	1	1	P	1	-
Abszolut ért.	15	y=abs(x)	ABS	10	1	-	-	1	-
Digitál-analóg konv.	16	y=const(TASK)	DAC	20	-	-	-	1	-
TASK	17	Programoható	TSK	20	-	"1"	-	-	-
Függvénygenerátor	18	y=F(x); /Prog./	FUN	10	1	-	-	1	-
Kimenet /lista/	19	Output=x	OUT	10	1	-	-	-	-
Fehér zaj /-1;+1/	20	y=korl.fehér zaj	UNI	2	-	0:al.	N	1	-
Normál zaj	21	y=Gauss; várh: 0.0 szór: 1.0	NOR	2	-	0:al.	N	1	-
Poisson-el. zaj	22	y=Poi(lambda)	PSN	2	-	1	N	1	-
Exp.eloszlásu zaj	23	y=Exp(lambda)	EXR	2	-	1	N	1	-

1. táblázat

Jelmagyarázat:
P: programozható;
N: nem programozható;
BE: bemenet;
KI: kimenet;
IC: kezdeti feltétel;
SG: előjelforditás;

I. Függelék

ADATSZALAG SZERKESZTÉSI SEGÉDLET - uj adat, szimuláció üzemmódban:

Blokknév	Adat jelentése	Kell-e?	Olv.	Formátum
Azonositó név	Tetszőleges karaktersor	F	1	6A8 /A48/
Hibrid pr. dekl.	Analóg gépi egys. előjellel	F	1	F16.6
	TASK-ok száma	F	1	I4
	Kötéslista hossza /stiff:neg./	F	1	I4
	A kötések sorban	F	C	I4, I10
	Stiff: lass. csoportok száma	D	1	I4
	első csoport lass. tényezője	D	1	I4
	első csop. elemeinek száma	D	1	I4
	első csop. elemeinek kódjai	D	C	I4
	. . .			
	utolsó csop. lass. tényezője	D	1	I4
	utolsó csop. elemeinek száma	D	1	I4
	utolsó csop. elemeinek kódjai	D	C	I4
Koefficiensek	Integrátorok, majd összegzők	D	C	4F16.6
	POT, FPT, DTB, zajgen. sorban	D	C	F16.6
Kezdeti feltét.	INT, INP, DTB, DAC sorrendben	D	C	F16.6
Időzitések	Lépésköz lista hossza	F	1	I4
	/Ha O: folyt. az I.lin.listától/ Lépésköz - meddig; idősorrendben	A	C	2F16.6
	Bemenetek változtatva hányszor	D	C	I4
	Mikor és mire /amelyik nem vált, kimarad e listából/	A	C	2F16.6
	TASK: mettől-periódus-meddig	D	C	3F16.6
Output specif.	LP-re minden hanyadik ciklus /O ugyanaz mint 1/	F	1	I4
	TP-re minden hanyadik ciklus /O: nem kell/	F	1	I4
	Melyik OUT rajzolandó /O: nem/	F	1	I4
	Lépték: Xmax, Ymax, Ymin	A	1	3F16.6
	Melyik OUT-ra ALARM /O:nem kell/	F	1	I4
	Min - max ALARM szint	A	1	2F16.6
	Hány TASK kisérendő nyomon	D	1	I4
	Melyek ezek /sorszámaik/	A	C	I4
Ellenőrző kód	O:STOP; 9999:Uj adat, szim; egyéb: módositás;	F	1	I4

-.-

Blokknév	Adat jelentése	Kell-e?	Olv.	Formátum
	Lépésközbecslés üzemmódban /ha a "lépésköz lista hossza" 0 lett/:			
I. lineariz. lista	Szorzók, majd osztók kiválasztott poz. - adat; /0: IC-ből/	D	C	I2, 6X, F16.6
	Relé vezérlés; +: X1≥X2; -: X1<X2; 0: az IC-ből;	D	C	F16.6
II. lin. lista	Belső fv. erősités /0: IC-ből/	D	C	F16.6
	Felh. fv. gen. erősités	D	C	F16.6
	Folytatás: ellenörző kód;	F	1	I4

-.-

Jelentések: F - feltétlenül szükséges; D - deklarációtól /elemkészlettől/ függ; A - előzetes adattól /pl. egy szolgáltatás kérésétől/ függ.
1 - egyszeres adat; C - ciklikusan, sorszám szerinti sorrendben.

ADATSZALAG SZERKESZTÉSI SEGÉDLET - a módositásokhoz;

Szimulációs feladat módositása

Azonositó név;	F	1	A48
1. blokkfej: Koefficiens módositások száma;	F	1	I4
első koef. módositás - elemdeklaráció;	A	1	I4
. - módositó adat INT, SUM	A	1	4F16.6
. a többi	A	1	F16.6
utolsó koef. módositás			
2. blokkfej: kezdeti feltételek módos. száma;	F	1	I4
első kezd. felt. módositás - elemdeklaráció	A	1	I4
. - módositó adat	A	1	F16.6
.			
utolsó kezd. felt. módositás			
3. blokkfej: időzités módositások száma	F	1	I4
első időzités módositás - elemdekl./0000:lépésk/	A	1	I4
. - módositó adat			
ha lépésk. mód: lépésk. lista hossza	A	1	I4
. /ha 0: folyt: uj adat, I.lin.lista/ a lépésköz lista /lépésköz-meddig/	A	C	2F16.6
. ha bemenet vált. mód: hányszor vált.:	A	1	2F16.6
. mikor és mire, időrendben	A	C	2F16.6
. ha TASK időzités módositás: mettől - periódus - meddig:	A	1	3F16.6
utolsó időzités módositás;			

Blokknév	Adat jelentése	Kell-e?	Olv.	Formátum
4. blokkfej:	output specifikáció változás	F	1	I4
	Ha nulla, az ellenőrző kód olv. jön, az output spec. marad;			
	Ha nem nulla, folyt. az uj adat, output spec. résztől.			
Ellenőrző kód beolvasása		F	1	I4

Lépésközbecslés feladat módositása:

- a 3. blokkfejig ugyanaz, mint szimulációs üzemmódban; onnan:

3. blokkfej:	I. lin. lista módositásainak száma	F	1	I4
első módositás	- elemdeklaráció	A	1	I4
.	- módositó adat szorzó, osztó esetén			
.	pozició - adat:	A	1	I2, 6X, F16.6
.	relé esetén adat:	A	1	F16.6
utolsó I. lista módositás;				
4. blokkfej:	II. lin. lista módositásainak száma	F	1	I4
első módositás	- elemdeklaráció	A	1	I4
.	- módositó adat /erősités/	A	1	F16.6
.				
utolsó II. lista módositás				
Ellenőrző kód beolvasása		F	1	I4

II. Függelék

Hibaüzenet - jegyzék

1. szint

PROHYS hibaüzenet nincs.

2. szint

NUMBER OF UNDEFINED INPUT POSITIONS: N

INPUT DATA ERROR

Jelentése: a kötéslistában N olyan elem van, amelynek kimenetét "bekötöttük", de egyetlen bemenetét sem.

NON-EXECUTABLE ANALOGUE PROGRAM

INPUT DATA ERROR

A kötéslistából nem készithető számitási vázlat. Az analóg kapcsolási vázlat minden hurkában kell legalább egy kezdeti feltétellel rendelkező elemnek lennie, ahonnan a számitás kiindulhat. Tehát az integrátor, holtidős blokk közvetlenül visszacsatolható, de például az összegző nem. Iterációs eljárás számára tehát a felhasználói függvénygenerátor, vagy TASK alkalmas.

DEAD TIME BLOCK NO. I AND ANALOGUE PERIOD NO. J ARE NOT COMPATIBLE

INPUT DATA ERROR

Az I-edik holtidős blokk által megvalósitani kivánt holtidő nem osztható a lépésköz lista J-edik lépésközével, s igy egész számu pont tárolásával a kivánt holtidő nem realizálható.

OVERFLOW IN DEAD TIME BLOCK NO. N

INPUT DATA ERROR

Az első öt holtidős blokk 600, a többi 50 pont tárolására képes. Ha a holtidő értéke osztva az alkalmazott legkisebb lépésköz értékével ennél nagyobb számot ad, a fenti hibaüzenetet kapjuk /stiff müködés esetén nem a legkisebb lépésközzel, hanem annak TSF_1-szeresével kell osztani, ahol TSF_1 az 1. csoport lassitási tényezője/.

TIMING IS NOT CORRECT FOR DEAD TIME BLOCKS

INPUT DATA ERROR

A hibaüzenet holtidős blokkok alkalmazása esetén jelenhet meg, ha valamelyik lépésköz nem osztható valamennyi nála kisebbel.

TIMING IS NOT CORRECT FOR STIFF SYSTEM

INPUT DATA ERROR

Stiff müködés esetén valamennyi lépésközzel $k * TSF_1$ ciklust kell futtatni, tehát lépésközváltás csak akkor lehetséges, ha az 1. csoport integrálása éppen folyik. Ha a lépésköz lista ezt a feltételt nem teljesiti, akkor a fenti hibaüzenetet kapjuk.

/k - tetszőleges pozitiv egész szám/

OVERFLOW IN STEP LIST

INPUT DATA ERROR

Stiff müködésnél léphet fel, ha nagyon nagy a feladat, és nagyon sok csoportra van bontva. A teendő: csökkenteni a csoportok számát. /Ritka hibaüzenet./

OVERFLOW IN STEP LIST

INPUT DATA ERROR

Stiff müködésnél léphet fel, ha nagyon nagy a feladat, és nagyon sok csoportra van bontva. A teendő: csökkenteni a csoportok számát. /Ritka hibaüzenet./

TIME SCALING FACTORS ARE NOT COMPATIBLE

INPUT DATA ERROR

A stiff módszer lényegéből adódik, hogy egy csoport integrálásának valamennyi, nála gyorsabb csoporttal együttesen kell történnie. Ezért egy lassitási tényező egész számu többszöröse kell hogy legyen valamennyi nála kisebbnek. Ha e feltétel nem teljesül, a fenti hibaüzenetet kapjuk.

INPUT DATA ERROR

A PROHYS által nem értelmezhető hibák esetén /pl. negativ sorszámu, vagy nem létező tipusszámu elem deklarálása, stb./ jelenhet meg.

3. szint

NO RESULT, SORRY

Lépésközbecslés esetén kapjuk, ha a sajátértékszámitás abszolut hibája 30 iterációs ciklus után még mindig nagyobb, mint 10^{-4} /nagyon ritka hiba/.

EXECUTION ERROR AT INTEGRATOR NO. N

Az integrálás megkezdésekor a bemenetösszegek előállitása lehetetlen. A feladat végrehajtása a step-lista N-edik sorában akadt el. /A step--lista az outputon a "THE EXECUTION ORDER" fejléc után következik./

EXECUTION ERROR IN MINICYCLE

BLOCK TYPE: I NR: J

Kisérlet I kódszámu, J sorszámu elem végrehajtására a kezdeti állapot kiszámitása során. Fatális hiba.

EXECUTION ERROR AT LOG BLOCK NO. N

Kisérlet negativ szám vagy nulla logaritmusának előállitására, az N-edik sorszámu logaritmus blokknál.

EXECUTION ERROR AT DIV BLOCK NO. N

Nulla az N-edik sorszámu osztó blokk második /osztó/ bemenetén.

EXECUTION ERROR IN THE CYCLE

Nem végrehajtható elem végrehajtási kisérlete /fatális hiba/.

EXECUTION ERROR

Nem felderithető fatális hiba a végrehajtás során.

NO DIAGRAM

Felhasználói tengelylépték esetén már az első pont is kivül esik a grafikon hasznos területén.

III. Függelék

Példafeladatok

Az itt közölt feladatok egyrészt az ismertetett szabályok szemléltetésére, másrészt az adaptált PROHYS tesztelésére szolgálnak. A két feladat - az amugy is gépfüggő zajgenerátorokon kivül - a PROHYS valamennyi szolgáltatását felhasználja.

1. feladat - a PROHYS általános tesztprogramja

Valamennyi elemtipusból legalább egyet felhasznál. Az analóg kapcsolási vázlat a 2. ábrán látható. A bekarikázott számok a számitási elemek sorszámát, a többi a koefficienseket jelöli. A szimuláció - három különböző lépésközzel - tiz másodpercig tart. A 2. bemeneten egy lépcsőfüggvény jelenik meg, ugyanis kezdeti értéke 1.0, amelyet a következőképpen változtat meg /ld. az 1. feladat adatszalagja/;

a 2. mp.-ben 2.0-ra;
a 4. mp.-ben 3.0-ra;
a 6. mp.-ben 4.0-ra;
a 8. mp.-ben 5.0-ra;
és a 10. mp.-ben 6.0-ra.

A feladat lefutása után módositott adatokkal történő ujrafuttatás következik. Megváltozik a lépésköz lista, a 3. TASK időzitése, és az output specifikáció /kiiratási gyakoriság, plotter, TASK monitor/. A felhasználói függvénygenerátorok egyszerü átvitelt végeznek /VALOUT=VALIN utasitás/, a TASK-ok pedig semmilyen feladatot nem látnak el /CONTINUE utasitás/, jelenlétük csak a beléptetések és a TASK Monitor funkció ellenőrzése miatt szükséges.

Helyes müködés esetén az egyes kimeneteken a következő függvények láthatók /t jelöli a szimulációs időt, L(t) a fent emlitett lépcsőfüggvényt/:

1. kimenet: t/10; az 1. bemenet értékének /1.0/ integrálása 0.1 bemenő koefficienssel, kiiratás az 1. felh. fv. generátoron keresztül;

2. kimenet: 0.4 t - 2; azaz -2-től +2-ig lineárisan nő;

3. kimenet: arctan (0.4 t -2);

4. kimenet: $0.25 \cdot 1.0 + (1-0.25) \cdot t/10 = 0.075\ t + 0.25$;

5. kimenet: a 0.÷2. mp időintervallumban -1.0 /ez a 2. holtidős blokk kezdeti feltétele/; ugyanezen idő alatt az első holtidős blokk kezdeti feltétele áttöltődik a másodikba, tehát:

a 2.÷4 mp időintervallumban 0.0; a negyedik másodperctől pedig megjelenik már L(t-4); mivel pedig a lépcsőfüggvény tulajdonságai miatt L(t-4)=L(t)-2; és ennek a holtidős blokkok kezdeti feltételei is megfelelnek, az 5. kimeneten megjelenő érték:

L(t)- 2;

6. kimenet: t/10 + 0.5; mert a 2. integrátor kezdeti feltétele ez esetben már nem nulla;

7. kimenet: +1.0[abs(0.4t-2)]; ha [t/10+0.5] ≥ 1.0;
és -1.0[abs(0.4t-2)]; ha [t/10+0.5] < 1.0;

Az ötödik másodpercig a második, attól kezdve az első feltétel teljesül, de az ötödik másodpercben vált előjelet negativról pozitivra a 0.4t-2 függvény is. Igy a relé és a szorzó tulajdonképpen az abs hatását kompenzálja; helyes müködés esetén a 7. kimeneten lévő érték meg-

egyezik tehát a 2. kimenetével, azaz:

0.4t-2;

8. kimenet: Ha a 2. kimenet értékét a következőképpen jelöljük;

x=0.4t-2;

akkor a 8. kimeneten megjelenő érték:

$x - \arctan \frac{\sin(x)}{\cos(x)}$; ami azonosan egyenlő nullával.

Erre az ellenőrzésre azért van szükség, mert az aritmetikai belső függvénygenerátorok nem a nagyon pontos, és ezért esetenként elég lassu müködésü FORTRAN könyvtári függvényeket alkalmazzák, hanem négy-öt számjegyre, azaz a szimuláció szempontjából rendszerint már elegendően pontos gyors közelitő eljárást.

9. kimenet: Mivel L(t-4)=L(t)-2; - ld. 5. kimenet - és ugyanugy L(t-2)= =L(t)-1; a kilencedik érték kimenete tehát:

1.0[L(t)-2]-2.0[L(t)-1]+1.0[L(t)]≡0;

Mivel a szimuláció során lépésközváltások is történnek, e kimenet segitségével a holtidős blokkok helyes müködése ellenőrizhető.

10. kimenet: Ha a 6. kimenet értékét a következőképpen jelöljük:

y=t/10+0.5 />0/;

akkor a 10. kimenet értéke:

exp[ln(y)]-y≡0;

Az ellenőrzés célja ugyanaz, mint a 8. kimenet esetében volt.

Ha tehát a PROHYS valamennyi számitási eleme helyesen müködik, akkor a tesztprogram eredményének tiz oszlopa a következő függvényeket tartalmazza:

Az 1., 2., 4., 6., 7. kimenet lineárisan növekvő függvényeket;
a 3. kimenet a 2. arcus tangensét;
az 5. kimeneten egy lépcsőfüggvényt;
a 8., 9., és 10. kimenet oszlopában végig nullát.

Ettől eltérő müködés esetén a hibás elem gyorsan behatárolható.

Az első feladat adatszalagja

A kapcsolási vázlat és az egyéb adatok alapján összeállitott adatszalag a következő:

```
PROHYS TEST                          /Azonositó név
1.0                                  /Anal. üz. mód: SG=U=1.0;
   3                                 /a TASK-ok száma
  43                                 /kötéslista hossza
0100   010101                       /1. INT - 1.SUM 1. bemenetére
       0118                          /még az 1. fv. generátorra is
0200   0619                          /2. INT - 6. OUT-ra
       010601                        /még az 1.REL 1. bemenetére
       0113                          /még az 1.LOG-ra, stb.
       040102
0101   0219
       0109
       0115
       020102
0201   0819
0301   0919
0401   1019
0102   020101
0103   030103
0203   010604
0104   0419
0105   0719
0106   010501
0107   010001
       010401
0207   0114
       030102
0307   020001
       010603
       010602
       0203
0108   0102
0109   0319
       0110
       0111
0110   010801
0111   010802
0112   040101
0113   0112
0114   0214
```

```
      0103
0214  0519
      030101
0115  010502
0116  010102
0118  0119
      010402                          /az utolsó, 43. kötés
0.1                                   /1. INT. koefficiense
0.1                                   /2. INT.      -"-
4.0           -2.0                    /1. SUM       -"-
1.0            1.0                    /2. SUM       -"-
1.0            1.0      -1.0          /3. SUM       -"-
1.0           -1.0                    /4. SUM       -"-
2.0                                   /1. POT       -"-
-1.0                                  /2. POT       -"-
0.25                                  /1. FPT       -"-
2.0                                   /1. DTB koef.: a holtidő;
2.0                                   /2. DTB       -"-
0.0                                   /1. INT kezdeti feltétele
0.5                                   /2. INT       -"-
1.0                                   /1. INP       -"-
1.0                                   /2. INP       -"-
1.0                                   /3. INP       -"-
0.0                                   /1. DTB       -"-
-1.0                                  /2. DTB       -"-
1.0                                   /1. DAC konv.-"-
   3                                  /Lépésköz lista hossza
0.1            5.0                    /0.1 sec-mal az 5. sec-ig;
0.01           6.0                    /0.01  -"-    a 6.   -"-
0.25           10.0                   /0.25  -"-    a 10.  -"-
   0                                  /1. INP: nem kell változtatni
   5                                  /2. INP: ötször változtatni;
   0                                  /3. INP: nem kell változtatni;
2.0            2.0                    /a most következő öt sor:
4.0            3.0                    /a 2. bemenet változtatási
6.0            4.0                    /adatai "mikor - mire"
8.0            5.0                    /felépitésben;
10.0           6.0                    /utolsó változtatás;
1.0            2.0       9.0          /három sor: a három TASK idő
2.0            1.0       8.0          /zitései "első belépés -
4.5            0.5       7.5          /periódus - utolsó belépés";
   5                                  /LP: minden ötödik ciklust kiirni;
   0                                  /TP: nem kell lyukasztani;
   0                                  /Plotter nem kell;
```

```
   4                                   /4.OUT: ALARM figyeltetés;
-10.0      0.9                         /alsó-felső sávhatárok;
   0                                   /TASK Monitor nem kell;
   1                                   /ellenőrző kód: ezután módositás következik.
PROHYS TEST WITH PLOTTER               /Azonositó név
   0                                   /1. blokkfej: nincs koefficiens módositás;
   0                                   /2. blokkfej: nincs kezdeti feltétel mó-
                                           dositás;
   2                                   /3. blokkfej: két időzités megváltozik;
0000                                   /az első: a lépésköz lista;
   1                                   /az uj lépésköz lista hossza;
0.05       10.0                        /"lépésköz - meddig";
0317                                   /a második módositás: a 3. TASK elem-
                                           deklarációja;
4.5        1.0          4.5            /egy belépés a 4.5 mp-ben;
   6                                   /a negyedik blokkfej nem 0:
                                        output spec. megváltozik;
   1                                   /LP: minden ciklus kiirni
  20                                   /TP: minden 20. ciklust;
   3                                   /3. kimenetet rajzolni kell;
0.0        0.0          0.0            /automatikus skálázással;
   0                                   /ALARM nem kell;
   2                                   /TASK Monitor: két TASK-ot nyomon kell
                                           követni;
   3                                   /mégpedig: a harmadikat
   2                                   /és a másodikat.
   0                                   /ellenőrző kód: több feladat nincs.
```

A feladat eredményeit a nagy terjedelem miatt jelen leirás nem tartalmazza.

2. feladat - példa a PROHYS stiff müködésére

Példaként vizsgáljuk egy atomreaktor un. egycsoportos pontkinetikai modelljét, egyszerüsitett hőmérséklet-visszahatásokkal. A modell hét differenciálegyenletből, és két algebrai egyenletből áll: [2]

$$\frac{dn}{dt} = \frac{\rho-\beta}{\ell^*}\, n + \sum_{i=1}^{6} \lambda_i c_i \; ; \qquad /1/$$

$$\frac{dc_i}{dt} = \frac{\beta_i}{\ell^*}\, n - \lambda_i c_i \; ; \quad i = 1,\ldots,6; \qquad /2.....7/$$

$$\beta = \sum_{i=1}^{6} \beta_i \; ; \qquad /8/$$

$$\rho = \rho_{rud} + \rho_{komp} + \Gamma_f T_f + \Gamma_R T_R \; ; \qquad /9/$$

ahol β, β_i, λ_i, ℓ^*, Γ_f, Γ_R fizikai tulajdonságokat leiró konstansok;

T_f, T_R üzemállapottól függő hőmérsékletek;

ρ_{komp} kompenzálja a hőmérsékletvisszahatásokat ugy, hogy $\rho=\rho_{rud}=0$ stacioner üzemállapotban;

c_1, c_2, ..., c_6 és n a rendszer változói;

ρ reaktivitás a bemenőjele, n neutronfluxus pedig a kimenőjel.

A fenti egyenletrendszernek megfelelő analóg kapcsolási vázlat a 3. ábrán látható.

A rendszer stiff jellegü; a |2|-ben foglaltak alapján integrálásának a következő módon kell történnie:

a 0. /leggyorsabb/ csoportba tartozik: a 7. integrátor, a 3. összegző és a szorzó; ezen elemek számára a lépésköz azonos a lépésköz listában megadottal;

az 1. /leggyorsabb lassitott/ csoportba tartozik a 6. integrátor, a 2. összegző; ezen elemek számára legyen a lépésköz a lépésköz listában megadottnak nyolcszorosa;

a 2. /lassitott/ csoportba tartozzék valamennyi elem; azaz az 1. - 5. integrátorok és az 1. összegző; ezen elemek számára a lépésköz legyen a lépésköz listában megadottnak negyvenszerese.

A rendszer nemlineáris; az első differenciálegyenletben két változó /ρ és n/ szorzata is szerepel. A szimuláció során a nyugalmi állapotban lévő reaktort /$\rho=0$/ az első másodpercben 14 cent értékü reaktivitászavar gerjeszti, /$\rho=0.14$/ és a feladat a válaszfüggvény felvétele. /Az adott skálázásban a reaktivitás relativ mértékegysége a "dollár", illetve századrésze, a "cent"./

A szimuláció elvégzése után a feladat lépésközbecslés végzése a reaktor különböző üzemállapotaira. Mivel a reaktor kinetikai tulajdonságait elsősorban a reaktivitás határozza meg, és nem a neutronfluxus, a linearizálás során legyen a szorzó 1. bemenete konstans, mégpedig:

- az első esetben legyen $\rho=0.14=const$; /üzemzavarnak megfelelő állapot;/
- a második esetben legyen $\rho=0=const$; /stacioner állapot;/
- a harmadik esetben pedig $\rho=-3.3=const$; /ekkora negativ reaktivitásugrást idéznek elő a bezuhanó biztonságvédelmi rudak./

A szorzó bemenetén az érték előállitható a kezdeti feltételből is - igy történik az első két lépésközbecslés alkalmával - ilyenkor az értéket az 1. bemenetre kell adni. Az érték megadható direkt módon is, mint a harmadik lépésközbecslés alkalmával történni fog.

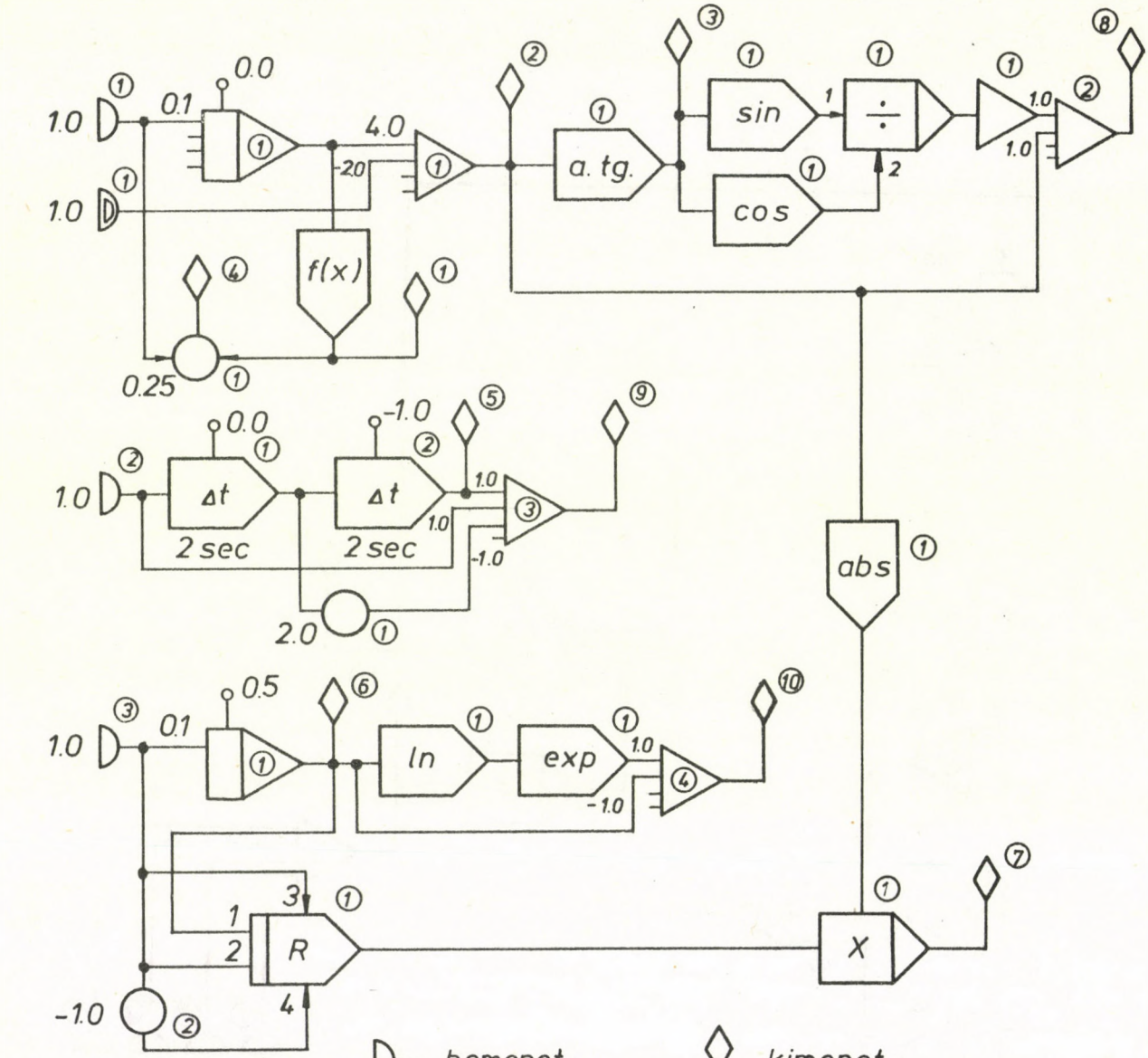
A szimulációt, valamint a három lépésközbecslést meghatározó adatszalag a következő lehet:

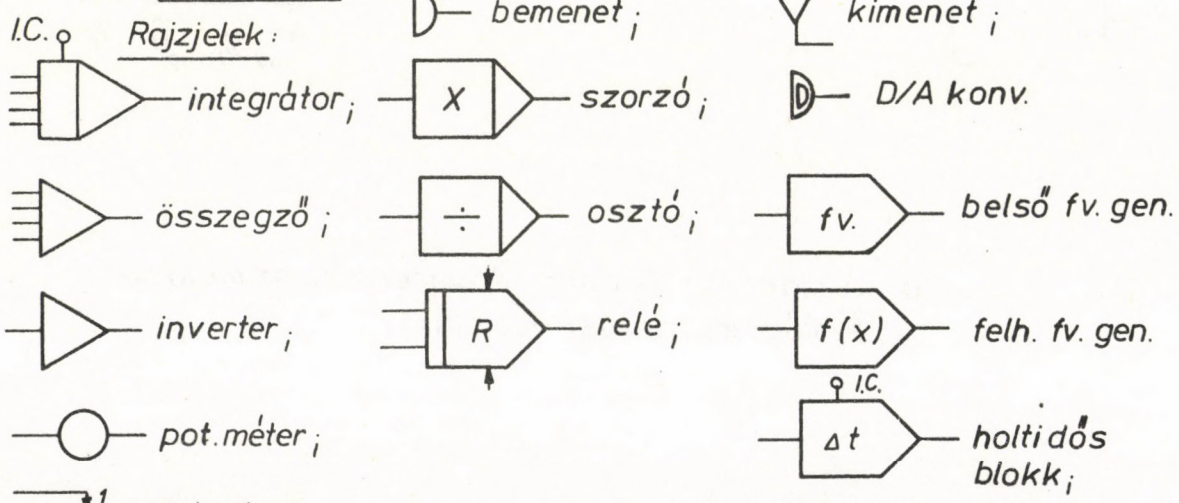
```
REAKTORZAVAR SZIMULÁCIÓ                 /Azonositó név;
1.0                                     /"Anal. gép" üzemmódja;
                                         SG=+1 és U=+1;
   0                                    /TASK nincs;
 -30                                    /30 kötés; negativ előjel:
                                         stiff müködés;
0100     010101                         /1. INT: 1. SUM 1. bemenetére;
         010002                         /még: a saját 2. bemenetére is
0200     010102                         /2.INT: 1. SUM 2. bemenetére;
         020002                         /stb.
0300     010103
         030002
0400     020101
         040002
0500     020102
         050002
0600     020103
         060002
0700     010001
         020001
         030001
         040001
         050001
         060001
         060004
         010502
         0219
0101     070001
0201     070002
0301     010501
0105     070003
         0119
0107     030104
0207     030103
0307     030102
0407     030101                         /az utolsó, 30. kötés
   2                                    /a lassitott stiff csoportok száma;
   8                                    /az első lassitott csoport lassitási
                                         tényezője
   2                                    /az elemek száma az első csoportban;
0600                                    /azaz: a 6. integrátor
0201                                    /és a 2. összegző;
  40                                    /a második lassitott csoport lassitási
                                         tényezője;
   6                                    /az elemek száma a második csoportban
```

```
0100                                    /azaz: az első öt integrátor;
0200
0300
0400
0500
0101                                    /és az 1. összegző.
0.006775    -0.0125                     /az első int.koef.
0.044856    -0.0315                     /a második;
0.288288    -0.154                      /stb.
0.671168    -0.456
1.840178    -2.328
1.6897      -13.85
100.0       100.0       204.167  -204.167  /a 7. int.koef.
0.125       0.315       0.308             /az 1. összegző
0.456       0.291       0.554             /a 2.SUM koef.
-0.016      -0.002      0.96948  1.0      /a 3.SUM koef.
2.3848                                    /az 1.INT kezdeti
6.2656                                    /feltétele
8.2368
6.2832
3.4780
0.5368
4.4                                       /a 7.INT  -"-
0.0                                     /az első bemenet kezdeti feltétele - ρ;
1.0                                     /a második bemenet kezdeti feltétele;
81.38                                   /a 3. bemenet kezdeti feltétele - T_f;
50.42                                   /a 4. bemenet kezdeti feltétele - T_R;
   1                                    /lépésköz lista hossza
0.005       28.8                        /"lépésköz - meddig";
   1                                    /az első bemenet egyszer változtatva;
   0
   0
   0                                    /a 2., 3., 4. nem változik;
1.0         0.14                        /az 1. mp-ben 0.14-re; - reaktivitás;
   5                                    /az LP-n kiiratva: minden 5. első cso-
                                         port számitás, azaz minden 5x8=40. meg-
                                         adott lépésköz, azaz 0.2 mp bontásban;
                                            - stiff müködés!
   0                                    /TP: nem kell lyukasztani eredményt;
   0                                    /Plotter: nem kell;
   0                                    /Alarm: nem kell;
   1                                    /ellenőrző kód: ≠0; ≠9999; tehát módo-
                                         sitás következik.
```

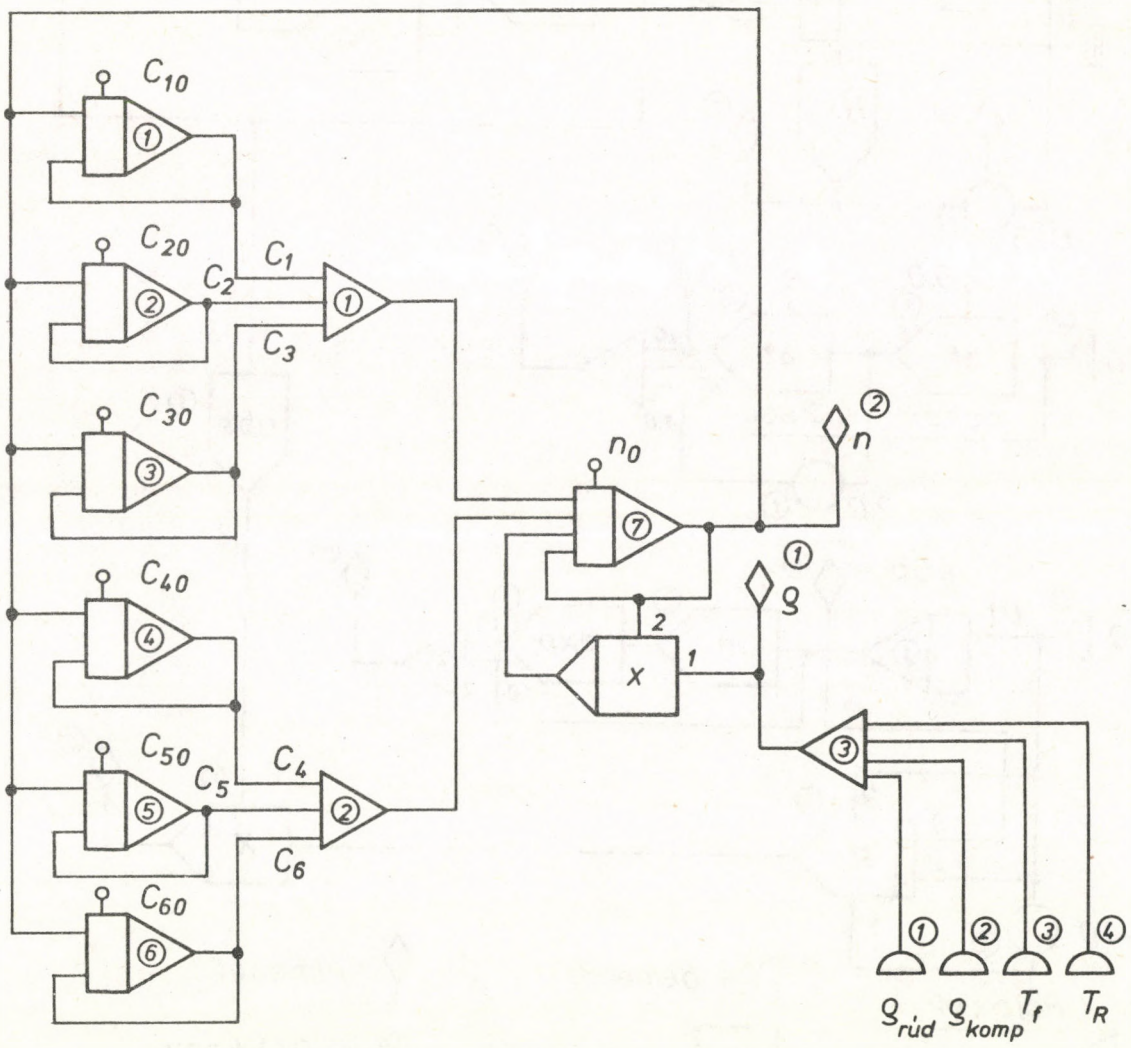
```
ELSŐ LÉPÉSKÖZBECSLÉS - RO = 14 CENT     /Azonosító név;
    0                                   /nincs koef. módositás
    1                                   /egy kezd. felt. módositás
0107                                    /az első bemenetre:
0.14                                    /ρ=0.14 - reaktivitás;
   1                                    /egy időzités módositás;
0000                                    /mégpedig a lépésköz listáé;
   0                                    /hossza: 0! - áttérés lépésközbecslésre;
01       0.0                            /szorzó 1. bemenet: kezd. felt. szerint;
                                         ennyi az I. linearizációs lista; a II.
                                         pedig teljesen elmarad, mert nincs
                                         hozzá tartozó elem - függvénygenerá-
                                         tor.
   2                                    /ellenőrző kód: ujra módositás;

MÁSODIK LÉPÉSKÖZBECSLÉS - RO = NULLA    /Azonosító név;
   0                                    /nincs koef. módositás
   1                                    /egy kezd. felt. módositás
0107                                    /az első bemenetre:
0.0                                     /nulla;
   0                                    /nincs I. lin. lista módositás
   0                                    /nincs II. lin. lista módositás
   3                                    /ellenőrző kód: ujra módositás;

HARMADIK LÉPÉSKÖZBECSLÉS - RO = -3.3 $  /Azonosító név;
   0                                    /nincs koef. módositás;
   0                                    /nincs kezd. felt. módositás;
   1                                    /egy módositás az I. lin. listában;
0105                                    /az 1. szorzónál:
01       -3.3                           /-3.3 az első bemenetre;
   0                                    /nincs II. lin. lista módositás;
   0                                    /ellenőrző kód: nincs több feladat.
```

Az adatszalag betáplálásával kapott eredmények közül az első és harmadik lépésközbecslés LP output-ját, valamint egy ugyanilyen, de plotterrel végzett szimuláció eredményeként kapott rajzot az [1] tartalmazza.





2. ábra Az 1. feladat (PROHYS TEST) analóg kapcsolási vázlata:

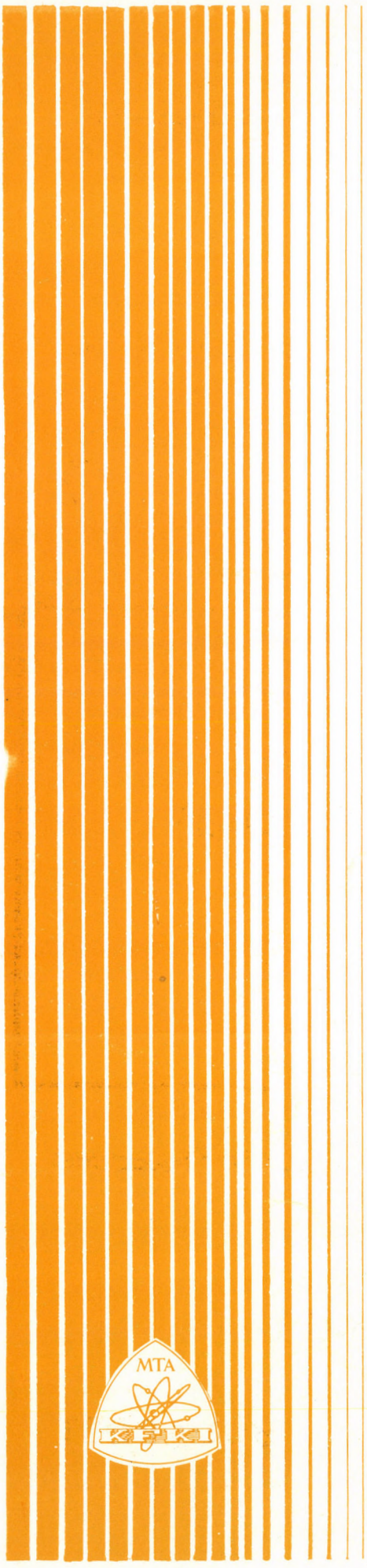


3. ábra A második feladat (Atomreaktor szimuláció)
analóg kapcsolási vázlata:

IRODALOM

[1] J.S. Jánosy - A. Szentgáli: PROHYS - a Program for Hybrid Simulation Using Digital Computers
Report KFKI-1976-37

[2] J.S. Jánosy: Extension of the PROHYS Program to Simulate Stiff Systems
Report KFKI, in preparation



Kiadja a Központi Fizikai Kutató Intézet
Felelős kiadó: Szabó Ferenc
Szakmai lektor: Szentgáli Ádám
Példányszám: 240 Törzsszám: 1977-721
Készült a KFKI sokszorosító üzemében
Budapest, 1977. augusztus hó