DIE ONTWIKKELING VAN WASIGE BEHEERDERS MET BEHULP VAN ONTOEGEWYDE GROOTSKAALSE GEïNTEGREERDE BANE.

deur

MARTEN F. SCHEFFER

'n Verhandeling voorgelê aan die Fakulteit Ingenieurswese
ter gedeeltelike vervulling van die vereistes
vir die graad

MAGISTER INGENERIAE
IN ELEKTRIESE EN ELEKTRONIESE INGENIEURSWESE

aan die

RANDSE AFRIKAANSE UNIVERSITEIT.

STUDIELEIER: Prof. I.S. SHAW
MEDE–STUDIELEIER: Prof. P.L. SWART

DESEMBER 1993

OPSOMMING.

In die afgelope paar jaar het wasige logika en wasige beheer baie vinnig ontwikkel. Die gebruikers van wasige stelsels het al hoe verder gewaag in die toepassingsveld van **komplekse, nie–lineêre en swak–gedefinieerde** prosesse of stelsels. Sommige wasige algoritmes vereis baie berekeninge, en **berekeningstyd** word 'n groot oorweging in egtydse of hoëspoedtoepassings van hierdie algoritmes.

Die bestaande kommersiële beskikbare wasige inferensie prosesseerders is slegs in staat om die reeds bestaande en gevestigde wasige operasies, soos byvoorbeeld maks–min, uit te voer. Dit is 'n aansienlike probleem indien enige navorsings– of ontwikkelingswerk gedoen wil word aangaande **alternatiewe wasige strukture of algoritmes**. Veral die ontwikkeling van selflerende stelsels lewer groot uitdagings. Om die probleem van uitvoertyd te oorkom, het verskeie navorsers hulle al gewend tot **mikroprosesseerder** gebaseerde wasige beheerders. Hierdie stelsels se uitvoertye word egter beperk deur die inherente nie–wasige aard van hulle instruksiestelle, en het soms baie komplekse parallel–prosesseerderontwerpe tot gevolg. Die laaste oorblywende keuse om vinniger uitvoertye te verkry, is om **hardbedrade apparatuurweergawes** van die wasige algoritmes saam te stel.

In hierdie Verhandeling word die resultate van die navorsing uiteengesit met die doel om 'n spesifieke wasige selflerende algoritme [23] in apparatuur te realiseer. Hierdie wasige metodiek is gebaseer op 'n unieke moontlikheidsteoretiese benadering, wat veral geskik is vir gebruik in werksomgewings wat baie ruis bevat. Die oorspronklike programmatuurweergawe van die wasige algoritme moes omskryf en optimiseer word, spesifiek vanuit 'n apparatuur–realisasie oogpunt. Die apparatuur–ontwikkeling het unieke probleme opgelewer, wat grondliggende beperkings op die algehele projek geplaas het. Om die nodige **aanpasbaarheid** te verleen, was van hoogsgeïntegreerde programmeerbare

vlokkies (FPGA's) gebruik gemaak. Die onderliggende **wiskundige operators**, soos **vermenigvuldiging**, deling, ensovoorts, moes op stroombaanvlak ontwerp word, en was gekombineer om finaal as hardbedrade ekwiwalent van die genoemde wasige selflerende algoritme te dien. 'n **Apparatuurplatform**, in die vorm van 'n IBM–versoenbare rekenaarkaart, was geskep om die wasige mikrovlokkie–ontwerp te ondersteun, en meer gebruikersvriendelik te maak.

ooOOoo

SUMMARY:

In recent years fuzzy logic and fuzzy control developed rapidly. The users of fuzzy systems have ventured ever deeper into the realm of **complex, non–linear and ill–defined** processes and systems. Some fuzzy algorithms require **extensive computation** and consequently **computation time** becomes a **critical limitation** in the real–time implementation of these algorithms.

Commercially available fuzzy inference processors are limited to well **established** fuzzy algorithms like max–min. This is a considerable limitation especially for researchers who want to experiment with alternative fuzzy algorithms. To overcome this problem of execution time many researchers have turned to micro–processor based fuzzy systems. These systems are, however, limited by their essentially non–fuzzy instruction sets and often result in very complex parallel processor designs. The final alternative is to design **hard wired fuzzy processor** systems.

The purpose of this thesis is to design a specific[23] **self–learning fuzzy algorithm in hardware.** This algorithm is based in **probability theory** and is highly immune to noise. The original fuzzy algorithm had to be optimized specifically with hardware implementation in mind. The hardware development gave rise to unique problems that put constraints on the whole project.

Highly integrated semi–custom programmable logic chips **(FPGA's)** were used to provide the needed design flexibility. The **mathematical operators**, like multiplication and division had to be designed at circuit level and were then combined to configure the total fuzzy hardware system. An **IBM–compatible** computer **plug–in card** was designed as a hardware platform to make the use of the fuzzy hardware more user friendly.

ooOOoo

DANKBETUIGING.

Om dank te kan uitspreek is 'n voorreg en nie 'n plig nie.
Ek wil graag almal wat hierdie projek moontlik gemaak het innig bedank.

Aan prof. I.S. Shaw, wil ek my dank uitspreek vir die daarstel van hierdie projek en die hulp wat hy deurgaans aan my gegee het. Dit word hoog op prys gestel.

Ook wil ek my vriende bedank vir hulle ondersteuning en volgehoue geduld.

Ek wil my ouers bedank vir die "laat aande", "seer vingers", eindelose korreksies maar veral vir aanmoediging wat nou al 'n leeftyd lank sonder onderbreking volhou. Hulle het my die vryheid gegee om die beste van myself te maak, en hiervoor sal ek hulle ewig dankbaar wees.

ooOOoo

INHOUDSOPGAWE:

LYS VAN SIMBOLE.

Hoofstuk 2:

Afdeling 2.2.2.

z_k	reële toestandsveranderlike.
ρ	aantal referensiegebiede in gebied–van–belang.
$\underline{\bar{z}}(r_m)_k$	wasige moontlikheidsvektor.
$\underline{\ }$	onderstreep dui 'n vektor aan.
$\bar{\ }$	oorstreep dui 'n wasige waarde aan.
r_m	referensiegebied–indeks $1...\rho$.
k	observasie indeks $k = 1...N$.
N	aantal observasies.

Afdeling 2.3.

F	reële funksie.
$\mu(F)$	lidmaatskapsfunksie van F.
$R^{(i)}$	wasige implikasieveranderlike.
R	algehele wasige implikasieveranderlike.
$U(s)$	Laplace–transform van die stelsel–inset.
$Y(s)$	Laplace–transform van die stelsel–uitset.
$H(s)$	oordragsfunksie.
$\mu_Y(x)$	lidmaatskapsfunksie van die uitset.
$\mu_U(x)$	lidmaatskapsfunksie van die inset.
$R_{UY}(x)$	wasige relasie.
o	saamstellende operator.

Afdeling 2.4.

$u_n(k)$	reële stelselinsette.

$y(k)$	reële stelseluitset.
n	aantal insette.
$\underline{\bar{u}}(s_n)_k$	inset wasige moontlikheidsvektor.
$\underline{\bar{y}}(s)_k$	uitset wasige moontlikheidsvektor.
$s_1,\ldots,s_n$	referensiegebied–indekse van insette.
s	referensiegebied–indeks van uitset.
$\hat{\bar{R}}$	beraamde wasige relasie.
$\mathcal{U}_m$	gebied–van–belang van insette.
$\mathcal{X}_n$	gebied–van–belang van interne toestande.
$\mathcal{Y}$	gebied–van–belang van uitset.
τ_u	heelgetal dooie–tydsvertraging van inset.
τ_y	heelgetal dooie–tydsvertraging van interne toestande.
ϕ_ρ	verwasigingsoperator, wat 'n *1*–tot–ρ afbeelding van skerp–getalle na wasige getalle lewer.
M	aantal insette.
N	aantal interne toestande.
L	totale aantal monsters.
W	leervenster.

Afdeling 2.5.

n_a	onderste perk van gebied.
n_b	boonste perk van gebied.
u_i	observasie waarde.
$\hat{y}$	beraamde uitset.
n	indeks in gebied $[n_a;n_b]$
M	venstergrootte.

D	aantal waardes vooruit geskat.
$\hat{\underline{y}}$	wasige beraamde uitset.
N	totale aantal monsters.
K	eweredigheidskonstante.
$\hat{y}_{k-1}$	reële rekursiewe beraming.
$\hat{\underline{y}}_{k\ init}$	inisiële wasige beraming.
$\hat{\underline{y}}(r_n)_{k-\tau_y}$	wasige rekursiewe beraming.

Afdeling 2.6.

$\mu_z(j)$	lidmaatskapsfunksie van reële veranderlike.
z_o	ontwasigde ekwiwalent van lidmaatskapsfunksie.
ϕ^{-1}	ontwasigingsoperator.

HOOFSTUK 3.

Afdeling 3.2.

Inkrement	kwantum–inkrementwyser in die gebied–van–belang.
Gebied	bestek van gebied–van–belang.
N	aantal diskrete kwantumwysers in die gebied–van–belang.
G	nie–wasige getal in die gebied–van–belang
n	beginadres van wasige ρ–tal.
m	aantal referensiegebiede 1...ρ.
$int()$	heelgetal–operator.

Afdeling 3.4.3.

y_k	reële stelsel–uitset.
$\hat{y}_k$	reële beraamde uitset.

N	aantal beramings.
J	diskrete gemiddelde–kwadraatfout.

Afdeling 3.6.2.

u_2	inset tot 2–inset stelsel.
u_3	inset tot 3–inset stelsel.
α	ruisfaktor.

HOOFSTUK 4.

Afdeling 4.4.2.3.

n	aantal bisse in binêre getalle.

Afdeling 4.4.2.4.

k	binêre dividend.
d	binêre deler.

Afdeling 4.4.3.1.

a	geheue–posisie in opsoektabel.
$z(k,l)$	enige twee–dimensionele matriks.

Afdeling 4.4.3.3.

a,b,c	willekeurige binêre veranderlikes.

Afdeling 4.4.3.5.

$u(k,l,m)$	enige drie–dimensionele geheue–matriks.

HOOFSTUK 1: INLEIDING.

"So far as the laws of mathematics refer to reality, they are not certain. And so far as they are certain, they do not refer to reality."

Albert Einstein
Geometrie und Erfahrung.

"I do not know what I may appear to the world, but to myself I seem to have been only like a boy playing on the sea–shore, and diverting myself in now and then finding a smoother pebble or a prettier shell than ordinary, whilst the great ocean of truth lay all undiscovered before me."

Sir Isaac Newton,
L.T. More(1934).

HOOFSTUK 1: INLEIDING.

"Wasige logika" klink soos 'n teenstrydige begrip, omdat logika geassosieer word met hoë presisie, en wasigheid met die antitese daarvan. Logika word tradisioneel gedefinieer as akkurate manipulasies, met goed gedefinieerde getalle.

Die meeste menslike redenasies is egter onpresies of benaderd. Mense besit die besondere gawe, wat tot dusver nie goed verstaan word nie, om in 'n omgewing van onsekerheid en onakkuraatheid nog steeds rasionele besluite te neem. Mense is in staat om gebroke spraak te verstaan, slordige handskrif te ontsyfer, 'n motor in 'n klein spasie te parkeer, poësie te waardeer en om 'n komplekse en uitgerekte storie op te som. Om sulke dinge te kan doen, vertrou ons nie op berekeninge in die tradisionele sin van die woord nie, maar baseer mense denke, of redenasies, op wasige patrone met onakkurate gedefinieerde grense. Mense manipuleer wel inligting, soos 'n rekenaar ook inligting kan manipuleer, maar die voorwerpe van ons denke is selde gebaseer op kwantitatiewe syfers.

Gesien in hierdie lig is wasige logika 'n poging om 'n model van menslike redenasie, wat gebaseer is op die benaderde en kwalitatiewe aard van menslike redenasie, te verkry. In so 'n model word presiese berekeninge as 'n laer prioriteit, en ook as 'n onrealistiese beperking beskou.[1]

Die menslike brein is in staat om feitlik oombliklike besluite te neem, bv. in die geval van die beheer van 'n motor tydens die vermyding van 'n moontlike ongeluk. Dit kan egter etlike sekondes neem om 'n betreklike eenvoudige numeriese berekening uit te voer. Dit is dus duidelik dat 'n ander, nie–numeriese besluitnemingsmeganisme meer

natuurlik is vir 'n mens. Wasige logika kan dus beskou word as 'n poging om menslike denkwyse aan masjiene te leer, in stede van die konvensionele geval waar mense, wat met masjiene wil kommunikeer, die masjiene se "denkwyse" moet gebruik en dus moet aanleer. 'n Voorbeeld hiervan sou wees wanneer 'n mens 'n programmeringstaal moet aanleer, om met 'n rekenaar te kan werk.

Die oorspronklike konsep van wasige logika was in die middel–sestigerjare voorgestel deur 'n professor aan die Universiteit van Kalifornië, Berkeley. Sy naam was Lotfi Zadeh [2] en hy het voorgestel dat waardes in die werklike wêreld nie in netjiese en goed–gedefinieerde kategorieë verdeel kan word nie. In konvensionele Aristoteliaanse logika is die lidmaatskap tot 'n gebied, of versameling, nie 'n geval van graad nie, maar is 'n element of 100% lidmaat van 'n versameling of glad nie lid nie, dus 0% lidmaat. Hierdie aan– of af–, 1 of 0, benadering is geldig wanneer besluite geneem moet word oor goed–gedefinieerde konsepte soos binêre–getalle; maar is nie voldoende indien onsekere, maar tog alledaagse, konsepte hanteer moet word nie.

Om onsekere konsepte te kon hanteer, het Zadeh [2] voorgestel dat lidmaatskap nie beperk moes wees tot slegs 0 of 1 nie, maar dat ander waardes tussen 0 en 1 ook gebruik kon word om die mate van deelname aan 'n versameling te weerspiëel. In die versameling van lang afstande, kan byvoorbeeld, "baie ver" 'n lidmaatskap van 0.9, en "naby" 'n lidmaatskap van 0.1 hê.

In Tabel 1.1 word 'n moontlike opsomming van die historiese ontwikkeling van wasige beheer getoon, soos saamgestel uit [1][3][4].

Wasige logika het toepassing in die industrie begin vind in die jare 70 [5][6]. Veral in toepassings waar menslike operateurs besluite op basis van vorige ervaring moes neem, het wasige logika sy grootste ondersteuning gevind[7]. Sedert 1980 het wasige logika se populariteit baie toegeneem, veral in die Japanese industrie[4]. Die toepassing van wasige logika in algemene produkte, soos byvoorbeeld die outo–fokusstelsels van stil– en videokameras, het daartoe gelei dat wasige logika meer bekendheid verwerf het.[1] [4]

Tabel 1.1:

Historiese Verloop van die Ontwikkeling van Wasige Beheer:

1972	Zadeh	'n Rasionaal vir wasige beheer [7]
1973	Zadeh	Linguistiese benadering [8]
1974	Mamdani & Assilian	Beheer van stoom–aangedrewe enjin [6]
1976	Rutherford et al.	Ontleding van beheer–algoritmes [9]
1977	Ostergaard	Beheer van hittewisselaar en sement–oond [7]
1977	Willaeys et al.	Optimale wasige beheer [10]
1979	Komolov et al.	Meetbare automatisasie [11]
1980	Tong et al.	Afval–water behandelingsproses [13]
1980	Fukami et al.	Wasige voorwaardelike inferensie [14]
1983	Hirota en Pedrycz	Waarskynlikheid wasige versamelinge [15]
1983	Takagi en Sugeno	Afleiding van wasige beheer [16]
1983	Yasunabo et al.	Voorspellende wasige beheer [17]
1984	Sugeno en Murakami	Parkering–beheer van 'n model kar [18]
1985	Kiszka,Gupta et al.	Wasige stelsel–stabiliteit [19]
1985	Togai en Watanabe	Wasige mikro–vlokkie [20]
1986	Yamakawa	Wasige beheer apparatuurstelsel [21]
1988	Dubois en Prade	Benaderde redenering [22]
1990	Y.I. Kudinov	Stabiliteitsanalise [12]
1991	B. Kosko	Neuro–wasige Stelsels [1]

Maatskappye soos Canon, Nikon, Minolta, Nissan, Hitachi, Sony, Matsushita en

Fischer het wasige logika alreeds gebruik in hulle mees gevorderde produkte [4]. Veral die Japanese industrie, waarskynlik vanweë 'n verskil in kultuur en opvattings, het wasige logika egter aangegryp en veral die "menslike" natuur van hierdie tegnologie benut. Hedendaags word wasige logika internasionaal gebruik en veral Europese en Amerikaanse maatskappye het die meriete van hierdie teorie besef en probeer om die Westerse agterstand op hierdie gebied in te haal. En inderdaad was die eerste firma om 'n wasige logika–vlokkie te vervaardig 'n klein firma, Togai Infra Logic, in Kalifornië [20].

Bart Kosko [1], 'n opvolger van Zadeh en 'n professor in elektriese ingenieurwese aan die Universiteit van Kalifornië, het gesê, "the only barrier remaining , is the philosophical resistance of the West".

Waarskynlik die grootste voordeel van wasige beheerstelsels is die eenvoud van die opstel van die wasige beheerreëls. Alhoewel hierdie proses meer kompleks kan word indien 'n stelsel 'n groot aantal reëls benodig, was die meeste wasige beheerstelsels tot onlangs toe nog gekarakteriseer deur slegs 'n paar beheerreëls. Alhoewel die beginsels van wasige beheer al 'n geruime tyd verstaan word, het die toepassing van wasige beheer tot meer komplekse stelsels na die probleme van die fyner instemming van wasige beheerstelsels gewys. Die mees onlangse ontwikkeling in die wasige beheerveld is om neurale netwerke te gebruik om die fyner instemming van die wasige beheerder te behartig, terwyl die grondslag van die beheerder in die uitvoer van die wasige reëls lê. Deesdae is advertensies vir sogenaamde "wasige–neuro" produkte [4] algemeen in Japan en dit wil voorkom asof hierdie twee tegnologieë in tandem sal aanhou ontwikkel.

1.1. Uiteensetting van die inhoud van hierdie verhandeling:

–In **hoofstuk 2** van hierdie verhandeling sal daar gepoog word om die teoretiese beginsels van wasige beheerteorie wat direk van toepassing is op hierdie projek, meer breedvoerig uiteen te sit. Hierdie hoofstuk word aanbeveel vir lesers wat meer agtergrondkennis van die Shaw & Krüger–algoritme wil verkry.

–In **hoofstuk 3** sal die eksperimentele basis van hierdie projek vertoon en uiteengesit word in die vorm van programmatuursimulasies.

–**Hoofstuk 4** toon die prosedure wat gevolg was om 'n selflerende wasige–algoritme in apparatuur te ontwerp.

–Terwyl **hoofstuk 5** die ekperimentele resultate wat met die wasige–apparatuur verkry is, toon. Ook word die resultate van 'n vergelyking tussen die outeur se wasige–apparatuur, met ander bestaande, kommersiëel–beskikbare, wasige apparatuur getoon.

–**Hoofstuk 6** sluit dan die verhandeling af met 'n bondige opsomming van die mylpale wat in die projek behaal is en 'n uiteensetting van moontlike toekomstige navorsing.

ooOOoo

HOOFSTUK 2: WASIGE BEGINSELS EN VOORAFGAANDE NAVORSING.

" If I have seen further it is by standing on the shoulders of giants."

Sir Isaac Newton,(1675)

" This is not the end. It is not even the beginning of the end. But it is perhaps , the end of the beginning."

Winston Churchill,
Mansion House(1942).

HOOFSTUK 2: WASIGE BEGINSELS, EN VOORAFGAANDE NAVORSING.

2.1 Inleiding:

Die feit dat wiskunde, wat as 'n geheel sinoniem beskou word met akkuraatheid en presisie, relatief onsuksesvol toegepas kan word tot "regte wêreld" toepassings, het baie wetenskaplikes en filosowe ongelukkig gestem. Hierdie tekortkoming ontstaan omdat daar in beide logika en wetenskap 'n gaping is tussen teorie en praktiese resultate. Menige intellektueel het al bygedra tot die debat aangaande vaagheid en kwalitatiewe inligting. 'n Uitstaande bydrae om vaagheid, onsekerheid en kwalitatiewe denke te simboliseer word gevind in Zadeh[2] se wasige versamelingsteorie en wasige logika.

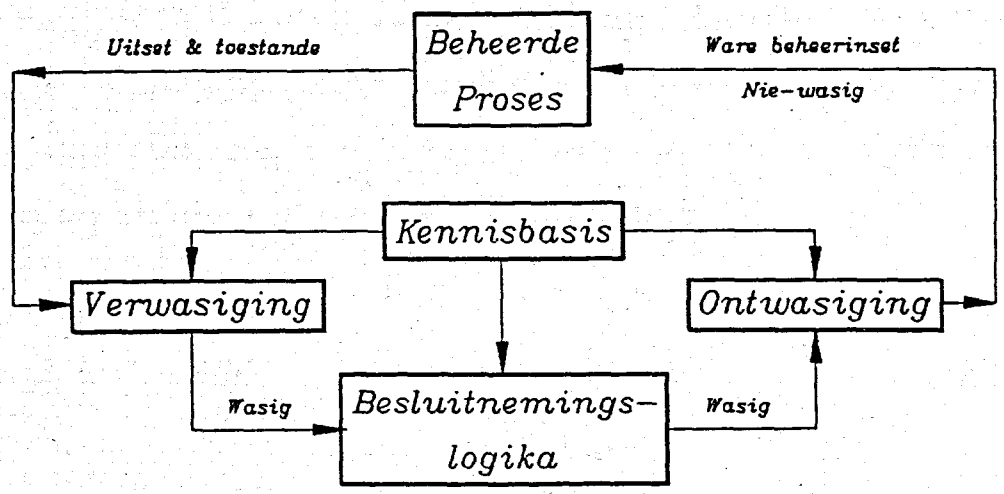
In hierdie hoofstuk word slegs die gedeeltes van wasige teorie wat pertinent van toepassing is op hierdie projek bespreek. Wasige versamelingsteorie, en wasige inferensie, word deeglik in ander literatuur [1] [2] [24] [25] [8] bespreek, en verduidelik, en word nie in hierdie verhandeling herhaal nie.

Wasige begrippe, soos verwasiging, lidmaatskap en moontlikheidsvektore, word wel gedefinieer. Verder word reëlgebaseerde en rasionele samestellings van die beheermodel bespreek. Hierop volg 'n uiteensetting van die relasionele–vergelykingsgebaseerde metode, wat die kern van hierdie projek vorm, naamlik die Ridley, Shaw en Krüger of RSK–metode[39]. Die hoofstuk word afgesluit met 'n bespreking van wasige beramings– en ontwasigingsmetodes.

2.2 Verwasiging en die moontlikheidsvektor: [24] [2] [3]

Enige wasige beheerproses, figuur 2.2.1, begin met die verwasiging van die stelselveranderlikes. Verwasiging behels die omskrywing van nie–wasige, of "skerp" getalle, na die wasige domein. Verwasiging kan gesien word as 'n koppelvlak tussen die

skerp– en wasige–domein.

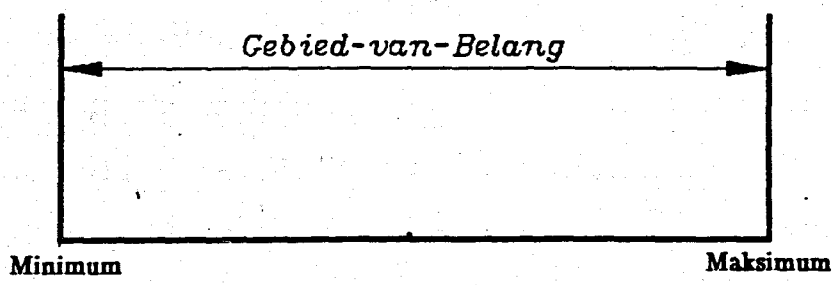


Figuur 2.2.1: Basiese konfigurasie van wasige beheerproses.

2.2.1. Die verwasigingskoppelvlak het die volgende funksies:

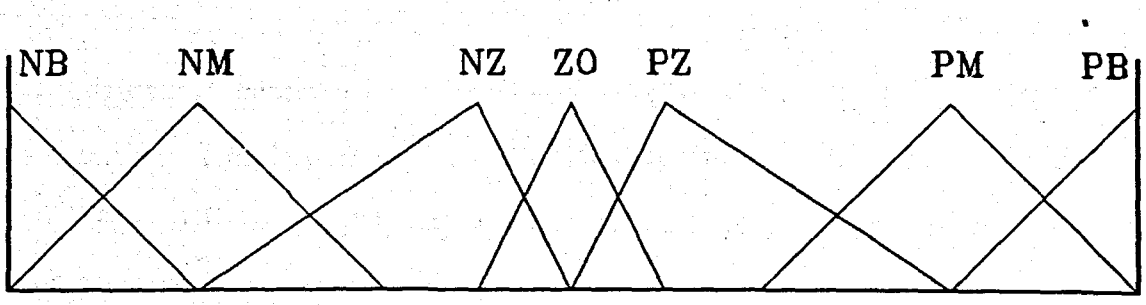
a) meting van die waardes van die stelselveranderlikes,

b) uitvoering van 'n verskalingsfunksie en oordrag van die uitsetwaardes na die ooreenstemmende gebied–van–belang. 'n Gebied–van–belang kan eenvoudig gesien word as die gebied tussen die maksimum en die minimum waardes in 'n versameling. In figuur 2.2.2 word 'n moontlike voorbeeld van so 'n gebied–van–belang getoon.



Figuur 2.2.2: Gebied–van–belang.

c) die verwasiging van die insetdata, wat hierdie data omskakel na gepaste linguistiese kwalitatiewe waardes. Die wasige ekwiwalent van 'n skerp–getal is 'n reeks lidmaatskapswaardes, wat die mate van deelname van die skerp–getal tot elke wasige versameling aandui. Hierdie wasige versamelinge, waarin 'n gebied–van–belang verdeel word, word referensiegebiede genoem. (Figuur 2.2.3.)



Figuur 2.2.3: Wasige referensiegebiede.

2.2.2. Die wasige moontlikheidsvektor.

'n Wasige moontlikheidsvektor is 'n reeks lidmaatskapswaardes wat toegeskryf kan word aan 'n sekere referensiegebied. Anders gestel, is dit 'n vektor van waardes waar elke vektorinskrywing 'n indikasie is van die mate van deelname van 'n skerp data element tot 'n enkele referensiegebied in 'n gebied–van–belang. Daar sal dus 'n moontlikheidsvektor bestaan vir elke referensiegebied, en daar sal 'n vektorinskrywing bestaan, wat ooreenkom met elke element, in 'n stelsel se datastel van toestandsveranderlikes. Die moontlikheidsvektor van 'n enkele reële toestandsveranderlike, z_k, met ρ aantal referensiegebiede, word gedefinieer in die simbole lys as:

$$\underline{\bar{z}}(r_m)_k \qquad \text{.... 2.2}$$

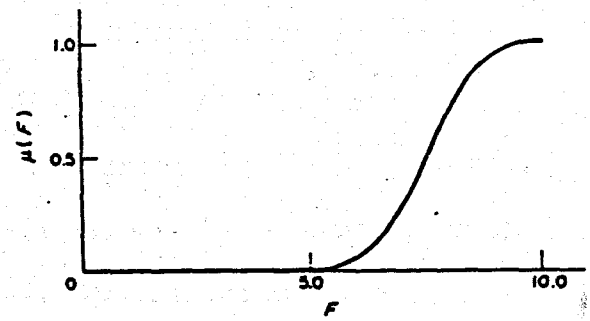
met: r_m *die indeks vir die referensiegebiede met waardes 1...ρ.*

2.3 Die wasige relasie:

'n Wasige relasie is 'n verteenwoordiging van die beheerreëls van 'n wasige stelsel. Die relasie verteenwoordig ook die mate van deelname van elke beheerreël tot die beheeruitset. Die mate van deelname van 'n reël is 'n uitspraak aangaande die mate van vertroue ten opsigte van 'n reël.

Elke linguistiese beheerreël kan verteenwoordig word met 'n sekere wasige gebied van lidmaatskap[24], soos byvoorbeeld die lidmaatskapsfunksie, $\mu(F)$, wat in figuur 2.3.1 getoon word. Tong definieer in [24] die kurwe in figuur 2.3.1 as 'n moontlike voorstelling van die linguistiese reël:

INDIEN brandstoftoevoer(F) hoog is DAN is die temperatuur(T) baie groter as 100° C.



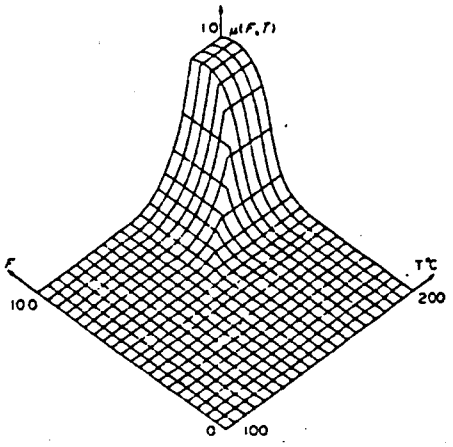
Figuur 2.3.1: Lidmaatskapsfunksie van veranderlike F.

Elke enkele beheerreël kan dus gesien word as 'n **nie–algemene** relasionele implikasie. In die algemeen word verskeie beheerreëls egter benodig om 'n proses te beheer, waar elke reël 'n wasige implikasieversameling $R^{(i)}$ sal lewer. Die individuele $R^{(i)}$'s moet gekombineer word om 'n algemene R te verkry, deur hulle vereniging te bereken.

Dus :

$$R = R^{(1)} + R^{(2)} + \ldots + R^{(N)} \qquad \ldots. \text{ 2.3.1.}$$

waar $R^{(i)}$ die wasige lidmaatskapversameling is, wat deur die i–de reël geproduseer word, en N die aantal reëls is. Indien $R^{(i)}$ met 'n lidmaatskapskurwe verteenwoordig kan word, is die logiese gevolg dat R as 'n lidmaatskapsvlak uitgebeeld kan word. In figuur 2.3.2 word 'n moontlike uitbeelding van die lidmaatskapsvlak, in die produkruimte van brandstoftoevoer, F , en temperatuur, T , getoon vir verskeie reëls. In hierdie voorbeeld val die temperatuur in die gebied [100° C;200° C] en die brandstoftoevoer in die gebied [0;10.0].



Figuur 2.3.2: Lidmaatskapsvlak.

Daar bestaan 'n analogie tussen die wasige relasie en die Laplace S–domein oordragsfunksie. Vir 'n enkele inset en 'n enkele uitset lineêre stelsel sal die Laplace oordragsfunksie $H(s)$ as volg voorgestel word:

$$H(s) = \frac{Y(s)}{U(s)} \text{ of } Y(s) = H(s).U(s) \qquad \text{.... 2.3.2}$$

waar $Y(s)$ die Laplace transform van die uitsetfunksie is, en $U(s)$ die Laplace transform van die insetfunksie. Indien die relasionele funksie, $\mu_Y(x) = R_{UY}(x) \circ \mu_U(x)$ [3], nou vergelyk word met vergelyking 2.3.2, en $\mu_Y(x)$ analoog aan $Y(s)$ en $\mu_U(x)$ analoog aan $U(s)$ beskou word, is die analogie tussen die wasige relasie, R, en die oordragsfunksie, $H(s)$, duidelik.

2.4 Bepaling van die wasige reëls, en die wasige relasionele model:

'n Wasige stelsel word gekarakteriseer deur 'n reeks stellings wat gebaseer is op kundige kennis. Die versameling van wasige stellings bepaal die reëlbasis van 'n wasige logika beheerder.

Met inag name van die besprekings in die vorige afdelings, waar verskeie verwysings na wasige beheerreëls gemaak is, is dit sinvol om nou die metodes te bespreek waarmee hierdie reëls bepaal word. Die metodes om 'n wasige model te bepaal, kan in twee hoofgroepe verdeel word, naamlik: reëlgebaseerde stelsels; en die benadering wat gebaseer is op die evaluasie van wasige relasionele vergelykings.

2.4.1 Reëlgebaseerde modelle:

Aangesien hierdie verhandeling hoofsaaklik op 'n wasige relasionele algoritme gebaseer is[26], word reëlgebaseerde modelle nie in diepte bespreek nie.

Die meeste wasige beheerstelsels word gebaseer op 'n stel linguistiese "Indien... Dan..." reëls. Hierdie linguistiese reëls word hoofsaaklik op drie verskillende metodes bepaal, naamlik:

i) **Reëls gebaseer op ervaring en beheerkundigheid** [6] [7] [13] [26] [28] [29] [30] [31]. Deur die introspektiewe verwoording van menslike kundigheid, of ondervinding van 'n ervare operateur, word die beheerreëls vasgelê.

ii) **Reëls gebaseer op 'n operateur se beheeraksies** [18] [32] [33]. Die beheeraksies van 'n operateur moet waargeneem word, en gekwantifiseer word, om die reëls af te lei.

iii) **Analitiese bepaling van die reëls** [34] [35]. Deur 'n alternatiewe analitiese

metode word die dinamika van die proses bepaal, en die beheerreëls word dan uit ervaring bepaal.

2.4.2 Wasige relasionele model:

Menige wasige logika–beheerders is al gebaseer op menslike besluitneming, maar baie min stelsels is al gebaseer op die menslike lering. Onder die begrippe van menslike lering verstaan ons die vermoë om beheerstelsels te skep, en ook aan te pas, gebaseer op ervaring. Procyk en Mamdani[36] het die eerste self–organiserende beheerder beskryf. Die self–organiserende beheerder was 'n hiërargiese stelsel met 'n algemene reëlbasis en 'n oorkoepelende reëlbasis om die algemene reëls aan te pas en te verskuif. Daar bestaan ook verskeie leer–algoritmes wat die wasige model in sy geheel bepaal, en nie–hiërargiese stelsels is nie.[37][38] Die wasige beheerreëls van 'n sekere stelsel kan verkry word deur die berekening van die wasige relasionele vergelykings, wat die verhouding tussen insette en uitsette verteenwoordig. Hierdie inset–uitset verhouding word die wasige relasie van die stelsel genoem. Gesien van uit 'n wiskundige oogpunt is daar geen verskil tussen die reëlbasis van 'n reëlgebaseerde stelsel en wasige relasie nie. Dieselfde wiskundige manipulasies kan met beide uitgevoer word.

In [26] word getoon hoe 'n wasige relasie, $\bar{R}$, bereken kan word. Gegewe 'n stelsel met insette $u_1(k),...,u_n(k)$ en 'n uitset $y(k)$, waar k die observasie indeks is, sal daar n wasige inset–moontlikheidsvektore $\underline{\bar{u}}(s_1)_k,...,\underline{\bar{u}}(s_n)_k$ en 'n uitset–moontlikheidsvektor $\underline{\bar{y}}(s)_k$ wees na verwasiging. Die indekse $s_1,...,s_n$ en s stem ooreen met die referensiegebiede en loop almal van $1...\rho$, waar ρ die aantal referensiegebiede in die gebied–van–belang is.

Dan word die wasige relasie gegee deur:

$$\bar{R}(s_1,\ldots,s_n,s)=\bigcup_{k=1+\tau}^{N}\{\bar{\underline{u}}(s_1)_k \cap \ldots \cap \bar{\underline{u}}(s_n)_k \cap \bar{\underline{y}}(s)_k\} \qquad \ldots.2.4.2$$

Die $\cup$ en $\cap$ verwys na die wasige vereniging en snyding [3].

Die beginsel van selflerende wasige stelsels is nie nuut nie, maar die aantal toepassings wat selflerende wasige stelsels benut is seldsaam. Dit wil egter voorkom om 'n natuurlike bron van beheerreëls te wees, aangesien juis swak gedefinieerde stelsels met wasige beheerders beheer kan word. Om 'n model van die stelsel uit die waargenome toestande te vorm is 'n baie bevredigende gedagte, en soos meer komplekse toepassings van wasige beheerders aangepak word sal hierdie tegnieke waarskynlik meer aandag geniet.

2.4.2.1 Die Ridley, Shaw en Krüger relasionele algoritme (RSK–algoritme):

LW: Vir 'n beskrywing van ander selflerende algoritmes, sien aanhangsel 2.4.P.

[39] het in 1988 'n nuwe metode van wasige relasionele identifikasie voorgestel. Hierdie metode is gebaseer in waarskynlikheidsteorie, en lewer 'n wasige relasie wat bereken word op basis van die geweegde gemiddeld van die wasige toestandsveranderlikes van 'n stelsel wat gemodelleer word. Hierdie algoritme beskou 'n wasige moontlikheidsvektor as 'n indikasie van die moontlikheid vir 'n veranderlike om in elk van die referensiegebiede te val. 'n Inskrywing in die wasige relasie ,$\bar{R}(s_1,\ldots,s_n,s)$, kan dus gesien word as die moontlikheid om 'n uitset, y, te verkry, wat in die s'de wasige referensiegebied sal val, as resultaat van insette $u_1,\ldots,u_n$ in gebiede $s_1,\ldots,s_n$ respektiewelik. Die geweegde gemiddeld van die vorige bepaalde moontlikheidsvektore lewer dan die relasie

$\hat{R}$. 'n Uitbreiding op hierdie metode is voorgestel in 'n opvolgende publikasie [23] in 1992, waar 'n bewegende–gemiddelde monsteringsvenster gebruik word. Hierdie metode aanvaar 'n multi–inset enkel–uitset stelsel. Verder word vooraf 'n aantal, ρ, vaste en identiese referensiegebiede gekies. Die berekende wasige relasie, $\hat{R}$, word volgens hierdie metode gegee as:

$$\hat{R}(\mathcal{U}_{m'}\mathcal{X}_{n'}\mathcal{Y}) = \hat{R}(r_{m'}r_{n'}r)$$

$$= \frac{\sum_{l=k-W-1}^{L}\left[\prod_{r_m,r_n,r=1}^{\rho}\left[\underline{\bar{u}}(r_m)_{m,l},\ \underline{\bar{x}}(r_n)_{n,\,l-1},\underline{\bar{y}}(r)_l\right]\right]}{\sum_{l=k-W-1}^{L}\left[\prod_{r_m,r=1}^{\rho}\left[\underline{\bar{u}}(r_m)_{m,\,l},\ \underline{\bar{x}}(r_n)_{n,\,l-1}\right]\right]} \qquad \text{.... 2.4.3}$$

waar: *$m=1,\ldots,M$ met M die aantal insette;*

$n=1,\ldots,N$ met N die aantal interne toestande of die orde van die model;

k die observasie–indeks; L die totale aantal monsters;

W die venstergrootte; ρ die aantal referensiegebiede in die gebied–van–belang;

$\underline{\bar{u}}(r_m)_{ml}=\phi_\rho(\underline{u}_{ml-\tau_u})$; $\underline{\bar{x}}(r_n)_{nl-1}=\phi_\rho(\underline{y}_{l-\tau_y})$; $\underline{\bar{y}}(r)_l=\phi_{\bar{\rho}}(\underline{y}_l)$;

ϕ_ρ is 'n verwasigingsoperator wat 'n 1–tot–ρ afbeelding van skerp getalle na wasige waardes lewer;

τ_u en τ_y is heelgetal–tydsvertragings toepaslik op u en x respektiewelik;

$\mathcal{U}_m$ is die gebied–van–belang van die insette u_m;

$\mathcal{X}_n$ is die gebied–van–belang van die interne toestande x_n;

$\mathcal{Y}$ is die gebied–van–belang van die enkel–uitset y.

Vir verdere verduideliking kan 'n enkel–inset enkel–uitset stelsel se wasige relasie $\hat{R}$ gegee word deur 'n vereenvoudigde vorm van die algemene geval, naamlik:

$$\hat{R}(\mathcal{U},\mathcal{X},\mathcal{Y}) = \frac{\sum_{l=k-W-1}^{N} [\underline{\bar{u}}_{1,l} \cdot \underline{\bar{x}}_{1,l-1} \cdot \bar{y}_l]}{\sum_{l=k-W-1}^{N} [\underline{\bar{u}}_{1,l} \cdot \underline{\bar{x}}_{1,l-1} \cdot]} \quad \text{.... 2.4.4}$$

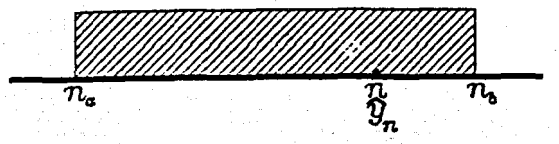
Die bewegende gemiddeld monsteringvenster en die geweegde gemiddeld operasie lei tot 'n wasige model wat hoogs immuun is teen ruis, soos deur [45] bewys.

2.5 Wasige beraming:

Voor wasige beraming in diepte bespreek word is dit sinvol om 'n grondslag te lê sover dit beramers in die algemeen aangaan.

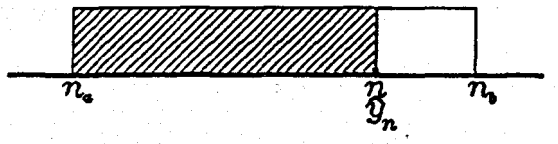
2.5.1 Bondelberamers: ("Batch estimators")

Bondelberamers [41][42] word in drie groepe verdeel, nl. i) gladstryking, ii) 'n filter, en iii) 'n voorspeller. Die verdeling geskied op grond van die spesifieke gedeelte van die interval $[n_a, n_b]$ waaruit die observasie waardes, u_i, geneem word om die beraamde uitset, $\hat{y}_i$, te bepaal.



Figuur 2.5.1.a: Gladstryking.

i) In **figuur 2.5.1.a**, word die gebied van die **gladstrykingsbewerking** getoon waar die totale aantal observasies $[n_a,n_b]$ gebruik word om 'n beraming te produseer. Aangesien sekere waarnemings na die **toekoms** van y_n verwys, is die proses nie kousaal nie, en die groepdata u_n moet in geheue gestoor word voordat die beraming begin kan word.



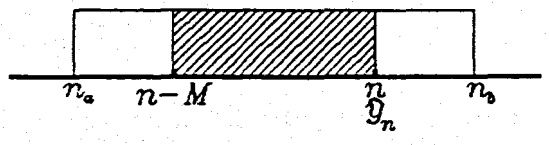
Figuur 2.5.1.b: Filter.

ii) **Figuur 2.5.1.b**, toon **filterwerking**, waar slegs die huidige monster en elke vorige monster gebruik word om die beraming $\hat{y}_n$ te bereking. Dus slegs die arseerde deel van die waarnemingsarea word gebruik om die beraming soos volg te bereking:

$$\hat{y}_n = \sum_{i=n_a}^{n} h(n,i)\cdot u_i \qquad \text{.... 2.5.1}$$

waar: *h(n,i) die diskrete oordragsfunksie van die stelsel is.*

Die beramer verkry dus data van die interval $[n_a,n]$ en lewer 'n beraming vir die gebied $[n+1,n_b]$.



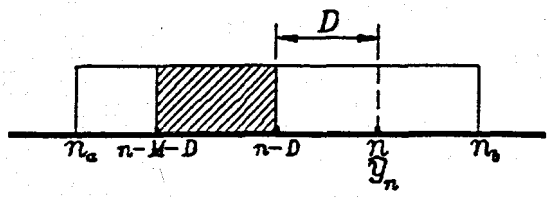
Figuur 2.5.1.c: Bewegende–gemiddeld.

Nog 'n weergawe van hierdie tipe beramer, is 'n **bewegende–gemiddeld**, wat slegs die huidige waarneming en die afgelope M waarnemings gebruik vir die berekening van die beraamde $\hat{y}_n$. Met ander woorde u_i, met $i=n,n-1,...,n-M$, word gebruik. Hierdie gebied word in **figuur 2.5.1.c**, getoon, en die beraming word as volg bereken:

$$\hat{y}_n = \sum_{i=n-M}^{n} h(n,i) \cdot u_i \qquad \text{.... 2.5.2}$$

iii) Daar is nog 'n verdere uitbreiding van die vorige beramer nl. die bewegende–gemiddeld–voorspeller, wat in **figuur 2.5.1.d** vertoon word. Die voorspelling van die D waardes word slegs op M waardes gebaseer. Die voorspeller verkry dus data in die gebied $[n-M-D,n-D]$ en verloor dus die oudste waarneming, soos elke nuwe waarneming in die berekening geneem word om beraming $\hat{y}_n$ te bereken. Dus slegs die arseerde deel van die waarnemingsarea word gebruik om die beraming soos volg te bereken:

$$\hat{y}_n = \sum_{i=n-M-D}^{n-D} h\,(n,i).u_i \qquad \text{.... 2.5.3}$$



Figuur 2.5.1.d: Voorspeller.

2.5.2. Wasige beraming.

Daar is al voorheen in afdeling 2.4.2 verwys na die werk van [26]. Hierdie navorsers het ook 'n wasige beramer voorgestel as die omgekeerde van die relasie berekening. Verder het hulle ook 'n moontlikheidsvlak, q, voorgestel. Indien 'n stelsel veranderlike se mate van deelname tot die uitset minder as die vlak, q is, word dit weggelaat. Volgens [26] word die beraamde uitset–moontlikheidsvektor verkry volgens:

$$\hat{\underline{y}}_k = \left(\bigcup_{u_{m,k}>q}\right) \cdots \left(\bigcup_{u_{n,k}>q}\right) \hat{R}(U_{m'}, X_{n'}, y) \qquad \text{.... 2.5.4}$$

waar: *m en n=1..ρ, met ρ die aantal referensiegebiede en*
k=1..N, met N die totale aantal monsters.

Omdat die wasige relasie, $\hat{R}$, 'n weergawe van die moontlikhede van alle inset/uitset kombinasies is, het [41] voorgestel dat die mees waarskynlike beraamde wasige **uitset–moontlikheidsvektor**, $\hat{\underline{y}}_k$, tydens die monstertyd, k, gegee sal word deur die matriksproduk van die inset–, $\underline{u}_k$, die interne toestand–moontlikheidsvektore, $\underline{x}_k$, en die **beraamde wasige relasie** $\hat{R}$. (Die interne toestandswaardes is vertraagde weergawes van die stelsel–uitset, y_k, en word vertraag met τ_y en die insette, $\underline{u}_{m,k}$, met τ_u, om dooie tyd in die stelsel te akkommodeer.):

$$\hat{\underline{y}}(\tau)_k = \sum_{r=1}^{\rho} \sum_{r_n=1}^{\rho} \sum_{r_m=1}^{\rho} [\underline{u}(r_m)_{m,k-\tau_u} \cdot \underline{x}(r_n)_{n,k-\tau_y} \cdot \hat{R}(r_{m'}, r_{n'}, r)] \qquad \text{.... 2.5.5}$$

Vergelyking 2.5.5 is die kumulatiewe som van die uitset–moontlikheidsvektore $\hat{\underline{y}}(1)_k$,....., $\hat{\underline{y}}(\rho)_k$. Elke moontlikheidsvektor word as volg bereken:

$$\hat{\underline{y}}(1)_k = \underline{\bar{u}}(1)_{m,k-\tau_u} \cdot \underline{\bar{x}}(1)_{n,k-\tau_y} \cdot \hat{R}(1,1,1) + \underline{\bar{u}}(2)_{m,k-\tau_u} \cdot \underline{\bar{x}}(1)_{n,k-\tau_y} \cdot \hat{R}(1,2,1) + ...$$

$$...+\underline{\bar{u}}(\rho)_{m,k-\tau_u} \cdot \underline{\bar{x}}(\rho)_{n,k-\tau_y} \cdot \hat{R}(\rho,\rho,\rho) \qquad .... \textbf{2.5.6}$$

vir: *k=τ....N monsters, met τ die maksimum van* τ_u, τ_y.

2.5.3. Rekursiewe beramer:

Die vorige afdeling het 'n nie–rekursiewe wasige beramer bespreek. In hierdie afdeling word die voordele van 'n rekursiewe beramer genoem.

2.5.3.1) Motivering vir die gebruik van 'n rekursiewe beramer:

'n Selflerende algoritme het minimale, of soms geen, voorkennis van stelselwaardes of –parameters nie. Tog moet sulke algoritmes in staat wees om alleenstaande te funksioneer. Dit is dus sinvol om 'n meganisme in 'n beramer of voorspeller te bou, wat die data van 'n selflerende algoritme gebruik, om te verseker dat die vooruitberamings so akkuraat as moontlik bly. Om 'n rekursiewe beramer te gebruik, wat sekere stelselveranderlikes terugvoer, is 'n moontlike oplossing van die bogenoemde probleem.

Die voordele van 'n rekursiewe beramer is as volg:

1) Vorige beramings genereer huidige beramings, dus kan die aanleg verwyder word en kan voordurend geleer word.

2) Lewer toestandsruimte uitbreiding van RSK–teorie[39] en is 'n toepassing van

Zadeh se reël van saamstellende afleiding [8].

3) Hoef slegs uitsetberaming 'n paar plekke terug in geheue te stoor.

4) Dit voorsien 'n addisionele inset tot die beramermodel, wat geskik is vir inherente foutbepaling.

'n Enkel–inset enkel–uitset rekursiewe beramer genereer 'n uitsetberaming op die basis van die huidige stelsel–inset $u_{k-\tau_u}$, die relasie $\hat{R}$, en 'n vertraagde vorige beraming van die uitset $\hat{y}_{k-\tau_y}$. 'n Algemene beramer wat gebaseer is op 'n enkel–inset enkel–uitset stelsel model sal gewoonlik slegs 'n enkel inset tot die beramer hê [26], volgens die vorm van die oordragsfunksie $Y(s)=U(s).H(s)$.

Die terugvoer van die vorige uitsetberaming vir 'n rekursiewe beramer lewer 'n ekstra inset tot die beramer, wat ook 'n ingeboude foutbepaling en korreksie meganisme insluit. Die vorige beraming voorsien dus 'n indikasie van die uitset van die stelsel, sonder dat die stelsel deel moet wees van die model. Verder voorsien die ekstra inset tot die beramer 'n fasiliteit wat die beraming op daardie oomblik meer uniek maak en gevolglik meer akkuraat [23]. 'n Groot voordeel van hierdie addisionele inset is dat dit intern gegenereer kan word en dat geen addisionele eksterne insette of inligting benodig word nie.

'n Nie–rekursiewe beramer is in essensie nie 'n alleenstaande stelsel nie [39], aangesien die uitset, y, van die stelsel wat gemodelleer word, $H(s)$, 'n inset tot die beramer moet wees, of dus volgens die vorm: [41]

$$\hat{\bar{y}}_k = \bar{u}_{k-\tau_u} \circ \bar{y}_{k-\tau_y} \circ \hat{\bar{R}} \qquad \text{.... 2.5.7}$$

met: "o" 'n saamstellende operator, soos gedefinieer

in [85] en verder verduidelik in [3] en [24].

Aangesien die ware stelsel–uitset, $y_{k-\tau_y}$, benodig word vir die beraming van $\hat{y}_k$, soos in figuur 2.5.3.b, is dit 'n logiese gevolg dat die stelsel, $H(s)$, ook benodig sal word om hierdie beraming te maak. Maar as die stelsel benodig word om 'n beraming te maak is dit sinneloos om 'n model van die stelsel te vorm. Die enigste vorm van 'n beramer wat op die vorm, $\hat{y}=u^{o}x^{o}\hat{R}$ gebaseer kan word, soos voorgestel deur [39], wat 'n werkbare selfstandige beramer sal lewer, is 'n rekursiewe beramer, soos voorgestel deur [23].

2.5.3.2) Beskrywing van rekursiewe SK–algoritme:

Die rekursiewe SK–beramer het die volgende eienskappe:

i) Tydens rekursiewe werking word ten meeste slegs 'n paar data items, instede van 'n blok data, gestoor.

ii) Daar moet 'n **inisiële beraming** gegenereer en gestoor word om die rekursiewe proses te begin.

iii) Die nuwe inkomende data–monster, wat beskikbaar word teen die volgende monstertyd, word saam met die inisiële beraming (wat nou $\hat{y}_{n-1}$ is) gebruik om die nuwe beraming te bereken.

Of anders gestel:

$$\hat{y}_n = \hat{y}_{n-1} + K(y_n - \hat{y}_{n-1}) \qquad \text{.... 2.5.8}$$

met: *K net 'n eweredigheidskonstante.*

Vir hierdie proses is daar dus baie minder geheue nodig, aangesien die beraming in 'n enkel geheue–sel gestoor kan word.

2.5.3.2.1) Inisiële beraming: (Begin van rekursiewe beramingsproses)

In hierdie algoritme moet die wasige identifiseerder, wat die wasige relasie bepaal, altyd van die filter tipe wees. In hierdie model is die filter in werklikheid die eerste stap van rekursiewe voorspelling, en dit genereer die waardes van die rekursiewe proses. Hierdie beginwaardes word bereken vanuit die gedwonge inset–moontlikheidsvektor, die gedwonge uitset–moontlikheidsvektor en wasige relasie, of as volg:

$$\hat{\underline{y}}_{k\ init} = \sum_{r_u=1}^{\rho} \sum_{r_y=1}^{\rho} \underline{\bar{u}}(r_u)_{m,1} \cdot \underline{\bar{y}}(r_y)_1 \cdot \hat{R} \qquad \text{.... 2.5.9}$$

waar: *die **inisiële toestandsvektor** beskikbaar is vir,*

$k = \tau+M+1$ met τ die grootste van τ_u en τ_y; $u_{m,k-\tau_u}$ die gedwonge insette; $\bar{y}_k$ die beraamde uitset; en $y_{k-\tau_y}$ die inisiële stelsel toestand; en M die grootte van die leervenster is.

Hierdie beraming genereer dus slegs een waarde wat dan oorgedra word na die rekursiewe proses.

2.5.3.2.2) Rekursiewe wasige voorspelling.

Die uitset van die inisiële wasige beraming is op daardie oomblik eintlik net 'n beraamde weergawe van die gedwonge wasige stelsel–uitset. Ons praat van voorspelling wanneer die tyd van belang na die laaste moontlike beskikbare waarneming is, sodat die

wasige beramer, van hier af aan as 'n voorspeller funksioneer.

Die rekursiewe aard van die voorspeller spruit uit die feit dat die beraamde uitset–moontlikheidsvektor terug gevoer word na die beramer, nadat dit met τ_y vertraag is, sodat dit in pas sal wees met die nuwe waarde van $\underline{\bar{u}}_k$ en die beraamde wasige relasie $\hat{R}$. Die waardes van die waargenome wasige uitset $\underline{\bar{y}}_k$ is dus nou nie meer nodig nie, wat 'n groot vermindering in geheue benodighede meebring. Die voorspelde uitset–moontlikheidsvektor
word dus as volg bereken:

$$\underline{\hat{\bar{y}}}_k = \sum_{r_n=1}^{\rho} \sum_{r_m=1}^{\rho} \underline{\hat{\bar{y}}}(r_n)_{k-\tau_y} \cdot \underline{\bar{u}}(r_m)_{k-\tau_u} \cdot \hat{R} \qquad \text{.... 2.5.10}$$

met: *k=M+2,...N, en ρ = die aantal referensie–intervalle.*

Die struktuur van hierdie wasige beramer is dus:

$$\hat{y}_k = \hat{y}_{k-\tau_y} \circ u_{k-\tau_u} \circ \hat{R} \qquad \text{.... 2.5.11}$$

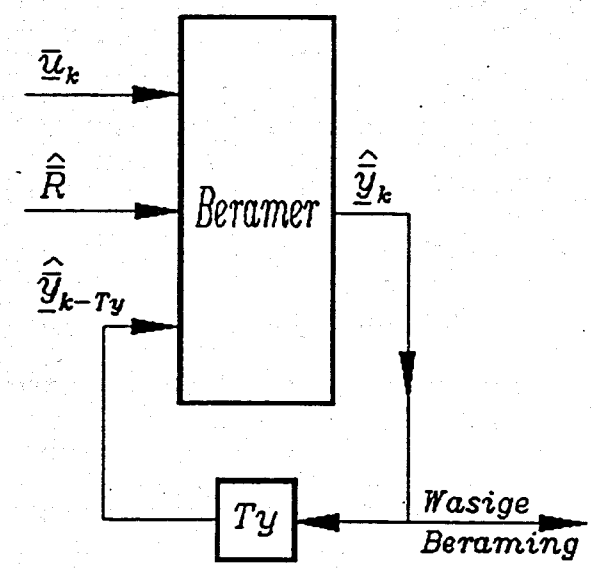
in plaas van:

$$\hat{y}_k = y_{k-\tau_y} \circ u_{k-\tau_u} \circ \hat{R} \qquad \text{.... 2.5.12}$$

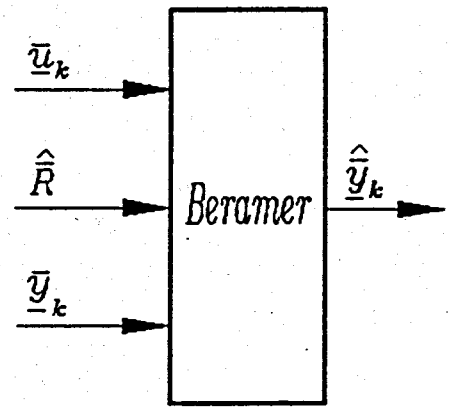
Nadat $\underline{\hat{\bar{y}}}_k$ verkry is moet dit nou ontwasig word, sodat $\hat{y}_k$ as uitset van die stelsel geskat word. Die wasige relasie is egter gebaseer op die eerste *M* waarnemings, en sal in gevalle wanneer baie ver vooruit beraam word, weer opgegradeer moet word.

Figuur 2.5.3.a toon die rekursiewe beramer, en vir 'n vergelyking, toon figuur 2.5.3.b die nie–rekursiewe beramer.

Figuur 2.5.3.a: Rekursiewe beramer.



Figuur 2.5.3.b: Nie–rekursiewe beramer.



2.6 Ontwasiging:

Ontwasiging is die afbeelding van 'n wasige beheerruimte, wat gedefinieer is oor 'n uitset gebied–van–belang, op 'n ruimte van skerp of nie–wasige beheeraksies.

'n Ontwasigingstrategie is daarop gemik om 'n nie–wasige beheeraksie te produseer, wat die beste verteenwoordiging van die moontlikheidsdistribusie van 'n geïnfereerde wasige beheeraksie is. Tot dusver is daar nog geen sistematiese metode om 'n ontwasigingstrategie te kies nie. Op die oomblik is die vier mees populêre strategiê die van: Maksimum–lidmaatskap, Mediaan–van–maksimum, Middelpunt–van–area[3] en Sentroïde ontwasiging[1]. In hierdie verhandeling word slegs sentroïde ontwasiging gebruik en gevolglik word dit in 'n opgesomde vorm volgende bespreek.

2.6.1. Sentroïde ontwasiging:[1]

Hierdie metode is 'n logiese gevolg van die middelpunt–van–area–metode[24]. Die verskil is slegs dat die getalle c_j vooraf gedefinieer word as die middelpunte, of sentroïdes, van elke referensiegebied. Die sentroïde van die totaal kan benader word met die additiewe kombinasie van die indiwiduele sentroïdes.[1]

Sodat:

$$z_o = \frac{\sum_{j=1}^{\rho} \mu_z(j) \cdot c_j}{\sum_{j=1}^{\rho} \mu_z(j)} \qquad \text{.... 2.6.1}$$

waar: c_j *die konstantes is wat ooreenstem met die sentroïdes van elke referensiegebied; en n die aantal*

referensiegebiede in 'n gebied–van–belang is en $z_0=\phi^{-1}(\mu_z(j))$
en ϕ^{-1} *'n ontwasigingsoperator.*

Hierdie konstantes kan dus vooraf bepaal, of gespesifiseer, word. Omdat die konstantes bekend is sal die ontwasiging dus minder berekeninge neem om uit te voer. Veral vir apparatuur realisasies is hierdie benadering van die middelpunt–van–area–metode van groot belang.

2.7 Wasige lekkasie. [23] [43].

Wasige lekkasie is deur [23] beskryf, en is in vorige navorsing opgemerk [43]. Wasige lekkasie is die verskynsel wanneer die lidmaatskapswaardes, in 'n wasige lidmaatskapsvektor, oordra van een gebied na 'n naburige referensiegebied. Gewoonlik sal een of twee lidmaatskapswaardes, in 'n wasige vektor, nie–nul waardes aanneem, en die res sal almal nul wees, soos byvoorbeeld {0,0.8,0.2,0,0}. Indien hierdie waardes "lek" na die naburige lidmaatskapswaardes, byvoorbeeld {0.1,0.7,0.1,0.1,0}, word die verskynsel "wasige lekkasie" genoem. Die leser sal opmerk dat die waardes al hoe meer versprei word, en minder gesentraliseerd is rondom die korrekte waardes in die vektor. Wasige lekkasie kan tot so 'n mate voorkom dat daar inligting verlore gaan, as gevolg van die uitsprei van die waardes.

Wasige lekkasie word hoofsaaklik veroorsaak deur ruis wat van buite die wasige stelsel ingevoer word; of deur interne foutsommasie, as gevolg van swak berekeningsakkuraatheid, of swak bis–resolusie[43].

2.8 Opsomming.

Wasige logika, soos in 1965 [2] voorgestel, bied 'n sistematiese tegniek om vaagheid, onsekerheid, en kwalitatiewe waardes voor te stel. Verwasiging omskryf skerp–waardes na die wasige domein. 'n Moontlikheidsvektor is 'n vektor van lidmaatskapswaardes van 'n reeks skerp–getalle tot 'n referensiegebied. 'n Wasige relasie is 'n sistematiese voorstelling van die beheerreëls van 'n stelsel. Daar bestaan twee groepe metodes om die relasie te verkry, naamlik: i) die reëlgebaseerde metode, en ii) die wasige relasionele model. Die RSK–algoritme [39] is 'n relasionele–identifikasiemetode wat gebaseer is op waarskynlikheidsteorie. Wasige beraming is die inverse bewerking van identifikasie, en die SK–algoritme [23] stel 'n rekursiewe wasige beramer voor. 'n Rekursiewe wasige algoritme kan outonoom funksioneer. Ontwasiging is die afbeelding van 'n wasige ruimte op 'n ruimte van skerp–, of nie–wasige, waardes.

LW.: Indien die leser belangstel in 'n metode vir die analise van die stabiliteit van wasige–beheerstelsels, kan verwys word na aanhangsel 1.

ooOOoo

HOOFSTUK 3: AFSPEELSTUDIES EN SUBSTELSELSTUDIES.

" May I ask whether these pleasing attentions proceed from the impulse of the moment, or are the result of previous study?"

Jane Austen,
Pride of Prejudice.

HOOFSTUK 3: AFSPEELSTUDIES EN SUBSTELSELSTUDIES:

3.1 Inleiding:

Die afdeling van hierdie projek, wat by verre die meeste tyd in beslag geneem het, was die navorsings– en ontwikkelingswerk wat gedoen is vir die saamstel van 'n mees geskikte selflerende wasige algoritme, met die oog op apparatuur implementasie. Soos voorheen in Hoofstuk 2 vermeld, word die wasige selflerende proses in vier hoofdele verdeel: (i) Verwasigingsmetodiek; (ii) Leer–algoritme; (iii) Beramingsalgoritme; en (iv) Ontwasigingsmetodiek. Elkeen van hierdie sub–dele was afsonderlik bestudeer, maar 'n oorkoepellende geheel moes steeds gehandhaaf word.

Deurgaans moes die apparatuur–beperkings en –oorwegings in gedagte gehou word. Daar moes egter ook gelet word dat die optimale keuse van selflerende apparatuur die doel is van hierdie projek, en nie slegs die lukrake opbou van enige selflerende wasige algoritme nie. Die apparatuur realisasie is dus nie die hoof bydrae van hierdie projek nie. Ook is die keuse en ontwikkeling van die sub–elemente van die selflerende algoritme nie die hoof bydrae nie, maar die vereniging van beide hierdie dele in 'n optimum realisasie van 'n selflerende wasige apparatuur stelsel, met inagname van die beperkings wat, byvoorbeeld, apparatuurstelsels en bestaande wasige algoritmes, ensovoorts, daar stel.

In hierdie afdeling sal gepoog word om die navorsing en besluitneming uiteen te sit, wat gelei het tot die saamstel van die finale selflerende algoritme. Die spesifieke apparatuur ontwikkeling moet in gedagte gehou word, maar sal later bespreek word in 'n afsonderlike hoofstuk. (Hoofstuk 4).

3.1.1: Doelstellings van die projek.

Die doelstellings van hierdie projek, soos in die Opsomming genoem, word hier vir die gerief van die leser weer puntsgewys genoem. Hierdie punte sal weer na verwys word in die laaste hoofstuk naamlik, hoofstuk 6: Gevolgtrekking en opsomming.

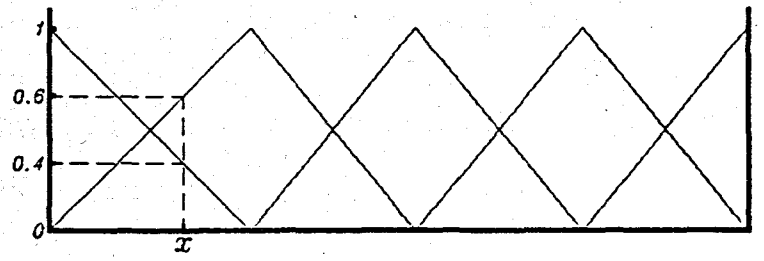
1) Om 'n spesifieke wasige selflerende algoritme [23] in apparatuur te realiseer.
2) Om die bestaande wasige programmatuur [41] te optimiseer met spesifieke verwysing na apparatuur–realisasie.
3) Die realisasie van die algoritme as 'n wasige mikro–vlokkie ontwerp.
3.1) Die realisasie van die wiskundige boublokke van die algoritme.
3.2) Die saamstelling van die wiskundige boublokke om die algoritme–bewerkings te realiseer.
4) Die ontwerp en opbou van 'n IBM–versoenbare rekenaarkaart om as ontwerpsplatvorm van die wasige mikro–vlokkies te dien.

Hierdie hoofstuk en die opvolgende hoofstukke is 'n weerspieëling van die resultate van die outeur se navorsing en ontwikkelingswerk wat spesifiek vir hierdie projek aangepak was.

ooOOoo

3.2: Tabel–verwasiging:

Verwasiging is 'n baie belangrike deel van wasige beheer, aangesien dit die deel van die proses is wat 'n omskakeling lewer van die skerp–domein na die wasige domein [24]. Die reële waardes in die gebied–van–belang word dus omskryf na lidmaatskapsvektore, waar elke element van die vektor die lidmaatskap van die reële getal tot elke referensiegebied verteenwoordig, soos in figuur 3.2.1 vertoon.



Lidmaatskapsvektor = {0.4;0.6;;0;0;0}

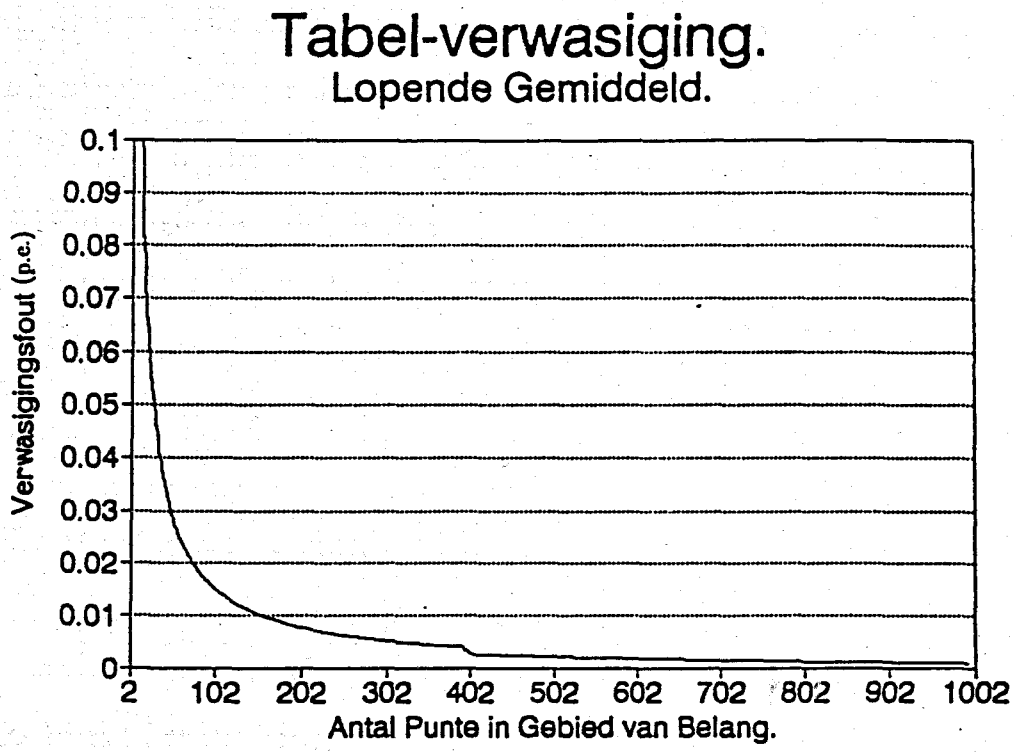
Figuur 3.2.1: Vektor lidmaatskap tot referensiegebiede.

Dit is alreeds algemene praktyk om die gebiede–van–belang van die stelselveranderlikes te normaliseer en die referensiegebiede, vir al die veranderlikes, identies te kies. Hierdie metode maak dit moontlik om slegs een enkele verwasigingsfunksie vir alle stelselveranderlikes te gebruik.

Die bestek van die genormaliseerde gebied–van–belang is dus [–1,1], en vooraf bekend. Netso, is die bestek van lidmaatskap [0,1], en dus vooraf bekend. Verder, kan die vorms en distribusie van die referensiegebied van elke veranderlike vooraf gespesifiseer word. Aangesien al hierdie eienskappe van die verwasiger alreeds vooraf bekend is, is dit 'n logiese gevolg om die verwasiging ook vooraf te doen, gebaseer op die bekende genormaliseerde waardes in die gebied–van–belang. Hierde verwasigde waardes kan dan later as 'n gekwantifiseerde opsoektabel gebruik word. Verwys asseblief na hoofstuk 4, vir

'n verdere uiteensetting van die apparatuurontwerp van so 'n opsoektabel.

Dit is sinneloos om 'n oneindige aantal punte in die gekwantifiseerde en genormaliseerde gebied–van–belang te verwasig. Dit is egter nodig om ekperimenteel die minimum aantal kwantumpunte in die gebied–van–belang, wat nog aanvaarbare resultate sal lewer, te bepaal. In figuur 3.2.2 word getoon hoe die verwasigingsfout afneem soos die aantal diskrete opsoekwaardes in die gebied–van–belang toeneem.



Figuur 3.2.2: Verloop van die verwasigingsfout soos die gebied van belang "growwer" gekwantifiseer word.

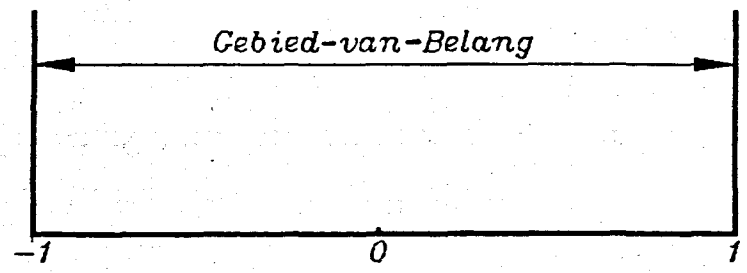
Figuur 3.2.2 is 'n eksperimentele verteenwoordiging van die verloop van die verwasigingsfout, volgens die opsoek–tabel metode. In figuur 3.2.2 kan gesien word dat die verwasigingsfout daal tot minder as 1% indien ongeveer 150 punte in die gebied–van–belang gebruik word. Dit is dus sinvol om $256 = 2^8$ diskrete waardes te gebruik vir algemene digitale stelsels. Hierdie aanname, sonder interpolasie tussen punte,

het 'n effense verswakking in die akkuraatheid van die wasige modellerings– en beramingsproses, meegebring, maar 'n ongeveer tweevoudige verbetering in die uitvoertyd van die algoritme veroorsaak. Sien asseblief afdeling 3.4.1 van hierdie hoofstuk, vir 'n indiepte bespreking van hierdie en ander simulasieresultate.

Hierdie metode van normalisering en kwantifisering, wat "Tabelverwasiging" genoem word deur die outeur, maak die verwasigingsproses aansienlik vinniger. Die wasige ekwiwalent van 'n skerp–getal kan dus intyds opgesoek word, en interpolasie is onnodig vir aanvaarbare uitvoertyd en akkuraatheid. Hierdie metode is veral geskik vir apparatuur realisasies. Die enigste vereiste om hierdie metode te gebruik is dat die stelselveranderlikes, gebiede–van–belange, en referensiegebiede vooraf bekend moet wees, of vasgestel moet word. Hierdie is egter, oor die algemeen, 'n vereiste in die meeste wasige beheerstelsels, en trouens in die meeste algemene beheermetodes is 'n mate van voorkennis nodig.

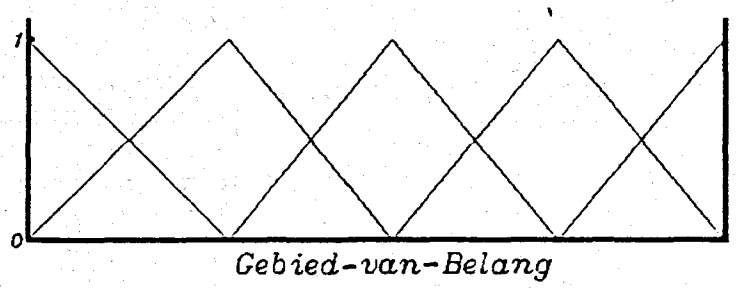
3.2.1 Afleiding van tabelverwasigingstegniek:

1) Hierdie tegniek vind sy oorsprong, soos in die vorige afdeling verduidelik, in 'n gebied–van–belang wat genormaliseer is om in die gebied $[-1,1]$ te val. Sien figuur 3.2.1.



Figuur 3.2.1: Genormaliseerde gebied–van–belang.

2) Die volgende stap is om 'n aantal, ρ, referensiegebiede te definieer in die gebied–van–belang. Sien figuur 3.2.2.



Figuur 3.2.2: Aantal referensiegebiede in die gebied–van–belang.

3) Verdeel nou die gebied–van–belang $[-1,1]$ in 'n aantal, N, waardes. Die inkrement tussen twee opeenvolgende waardes word dus gegee as:

$$Inkrement = \frac{Gebied}{N-1} = \frac{1-(-1)}{N-1} = \frac{2}{N-1} \qquad \text{.... 3.2.1}$$

Die waarde N bepaal dus die mate van kwantifisering van die gebied–van–belang.

4) Volgende word slegs hierdie N diskrete waardes verwasig, en in 'n geheue–tabel gestoor.

Die genormaliseerde gebied–van–belang is dus nou inkrementeel verdeel, en hierdie inkrementele waardes is vooraf verwasig en gestoor.

Die kern van die tabelverwasiger word gevind in die geheue–adres opsoek–metodiek. Inkomende reële waardes moet omskryf word om direk 'n geheue opsoekadres te

lewer wat ooreenstem met hierdie reële waarde se wasige ekwiwalent. Die nie–wasige getal, G, moet dus interpreteer word as 'n posisie in die gebied–van–belang. Hierdie posisie kan op sy beurt uitgedruk word as die aantal inkremente wat G verwyder is van 'n vaste verwysingspunt, soos byvoorbeeld, –1.

Uit bogemelde volg dat:

$$G = -1 + n \cdot (Inkrement)$$

waar: *n die aantal inkremente tussen –1 en die nie–wasige getal, G, is, en*

$$Inkrement = \frac{2}{N-1}.$$

Dus $G = -1 + n\left[\frac{2}{N-1}\right]$

en $$n = \frac{(G+1)(N-1)}{2} \qquad \text{.... 3.2.2.}$$

n kan dus ook geïnterpreteer word as die beginadres van die wasige ρ–tal, wat die wasige ekwiwalent van G is, sodat:

$$\phi(G) = \mu_G(n,m), \text{ waar:}$$

ϕ *die verwasigingsoperator is, soos voorheen gedefinieer,*

μ_G *die verwasigde waarde van G is,*

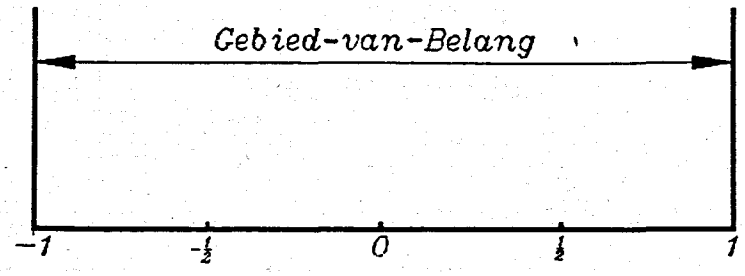
$m = 1.....\rho$ *referensiegebiede, en*

n die ρ–tal van wasige waardes is wat ooreenstem met reële getal G.

Voorbeeld:

As 'n gedagte–eksperiment, veronderstel dat die gebied–van–belang slegs vyf

waardes bevat, $N = 5$, soos in figuur 3.2.3 vertoon:



Figuur 3.2.3. Gebied–van–belang met vyf elemente in die opsoektabel.

Ons sal vervolgens vergelyking 3.2.2 kan toets:

Gestel die nie–wasige getal $G = -\frac{1}{2}$, dus volg dat:

$$n = \frac{(G+1)(N-1)}{2}$$
$$= \frac{(-\frac{1}{2}+1)(5-1)}{2}$$
$$= \frac{4}{4}$$
$$= \textit{1ste waarde.}$$

Dit bewys dat vergelyking 3.2.2 inderdaad na die korrekte geheue–adres in die tabel sou wys. Hou in gedagte dat $G = -1$, as die 0–ste getal in die geheue–tabel gedefinieer word.

Gestel nou $G = \frac{1}{4}$,

dus $$n = \frac{(G+1)(N-1)}{2}$$

$$= \frac{(\frac{1}{2}+1)(5-1)}{2}$$

$$= \frac{5}{2}$$

$$= 2\frac{1}{2} \text{ ste waarde.}$$

Daar bestaan egter nie 'n breukwaarde geheue–adres nie, dus n kan nie $2\frac{1}{2}$ wees nie. Vergelyking 3.2.2 is dus nog nie volledig nie. Die probleem van breukwaarde–adresse kan opgelos word deur die waardes van n ook te kwantifiseer deur n af te rond, en sodoende slegs heelgetal waardes te verkry, sodat vergelyking 3.2.2 uitgebrei word na:

$$n \approx int\left[\frac{(G+1)(N-1)}{2}\right] \qquad \text{....3.2.3.}$$

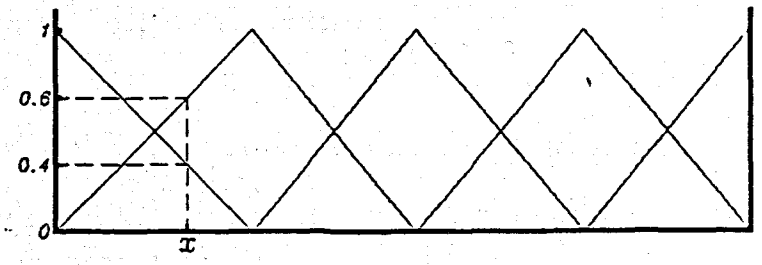
Vergelyking 3.2.3 is eenvoudig genoeg en akkuraat genoeg om bruikbare resultate te lewer, en vorm die teoretiese kern van die voorgestelde verwasigingsmetodiek. Verwys na hoofstuk 4, vir 'n verdere drastiese vereenvoudiging van hierdie metode met die klem op apparatuurimplementasie.

3.3. Nul–eliminasie:

'n Selflerende proses geskied sonder toesig, en dus moet alle moontlike inset–uitsetreëls ondersoek word om 'n beraming te verkry.

Enige stelsel wat nie 'n groot verspreiding van frekwensie–komponente het nie, en die meeste industriële stelsels val in hierdie kategorie, sal 'n groot hoeveelheid reëls hê, wat nie gebruik word nie, en gevolglik nul–gewigte sal dra [1]. So min as 4% van die relasie se reëls kan gebruik word om 'n industriële stelsel te beskryf [1]. Verder word die referensiegebiede gewoonlik so opgestel dat 'n reële getal ten meeste lidmaat kan wees van

twee referensiegebiede, soos in figuur 3.3.1 vertoon.



Figuur 3.3.1. Lidmaatskap tot referensiegebiede.

Indien 'n witruisstelsel gemodelleer word, kan verwag word dat alle reëls gewigte sal dra, aangesien alle frekwensie–komponente voorkom.[41]

Soos minder reëls gebruik word, so sal dit meer voordelig word om alle berekeninge wat die nul–gewig reël benodig, te elimineer. Die leer– en beramingsfases kan aansienlik vinniger uitgevoer word indien een of ander metode van nul–eliminasie gevolg word. Dit impliseer dat alle beramer– en leeroperasies slegs uitgevoer word, waar alle inset– en uitsetgewigte, en alle relasiegewigte, **nie–nul** inskrywings het. Die verwasigings– en ontwasigingsoperasies word nie deur hierdie metode beïnvloed nie.

3.4 Afspeelstudies ten opsigte van programmatuur–aanpassing:

Die opvolgende afdeling bespreek die optimisering ten opsigte van die uitvoertyd van die wasige leer– en beramingsalgoritme. Vir 'n afspeelstudie van 'n alternatiewe wasige beramingsalgoritme, sien aanhangsel 3.MM.

3.4.1. Optimisasie van wasige algoritmeprogrammatuur:

Die optimisering van programmatuur, om sodoende die uitvoertyd van die program te verbeter, is algemene praktyk in die gebied van simulasie. Dit is egter belangrik om in

gedagte te hou dat die program–optimisering, wat in hierdie afdeling voorgestel word, spesifiek gemik is op 'n apparatuur realisasie.

Die metodes van uitvoertyd–optimisasie wat ondersoek is, is Nul–eliminasie en Tabel–verwasiging. Hierdie metodes was al voorheen in afdelings 3.3 en 3.2 onderskeidelik bespreek. Die onderafdelings van Nul–eliminasie is die opstel van 'n nie–nul relasie–vektor, en die nul–eliminasie in die beramer. Tydens eksperimentering met die Tabel –verwasigingsmetode, is dit nodig gevind om 'n geheue–tabel van die wasige waardes op te stel vir die gebiede–van–belang van elke stelselveranderlike. Vir 'n enkel–inset–enkel–uitsetstelsel wat gebaseer is op die RSK–metodiek, soos in afdeling 2.4.2.1 bespreek, sal dit dus nodig wees om drie geheue–tabelle te stoor. Bo–en–behalwe die feit dat hierdie verskynsel meer geheue neem as om 'n enkele geheue–tabel te gebruik, is die opsoek en hernuwing van data in hierdie geheue–tabelle ook van so 'n aard dat dit stadiger word soos meer data geadresseer moet word. Om egter 'n enkele gebied–van–belang wat vasgestel is vir alle stelselveranderlikes te gebruik, het ook tekortkominge. Die enkel vaste gebied–van–belang moet alle moontlike waardes wat kan voorkom akkommodeer. 'n Stelselveranderlike wat slegs positiewe waardes het, sal dus net die helfte van die vaste gebied–van–belang in [–1,1] gebruik. Die aantal diskrete waardes wat in die wasige tabel gebruik sal word, moet egter vooraf gespesifiseer word, wat tot gevolg het dat dieselfde aantal waardes in die geheue–tabel ingesluit sal word in die gebied–van–belang [0,0.5] as in die gebied [–1,1].

Die volgende program–optimiseringstappe was ondersoek om te poog om die ideale programmatuur saam te stel:

3.4.1.1 Nul–eliminasie met nie–nul relasie vektor:

'n Vektor van alle relasie inskrywings se **indekse**, wat ooreenstem met die relasie inskrywings wat nie nul waardes het nie, is opgestel tydens die leerfase van die wasige proses. Gevolglik kan dit gebruik word om slegs hierdie nie–nul relasie waardes in die beramingsfase op te soek, en nie tyd te mors deur beramingsberekeninge uit te voer vir waardes wat in elk geval nul as resultaat sal oplewer nie.

Die nie–nul–vektor metode het 'n ander voordeel ook: dit elimineer alle matriks–indekstellers in die beramingsproses, omdat slegs die nie–nul–vektor geheue–indekse gebruik word om die ander indekse direk te spesifiseer. Vir 'n enkel–inset enkel–uitset RSK–stelsel sal elke nie–nul–vektor indeks drie ander indekse vir die inset $\underline{u}(s_1,l)$, die uitset $y(s_2,l)$, en relasie $\hat{R}(s_1,s_2,l)$, spesifiseer.

3.4.2. Optimisering ten opsigte van uitvoertyd van die RSK–algoritme, met spesifieke verwysing na nul–eliminasie ("zero–trapping"):

LW:

Alle programmatuursimulasies, in hierdie hoofstuk en die opvolgende hoofstukke, was uitgevoer met 'n IBM–versoenbare 386–SX–20 rekenaar met 'n wiskundige mede–verwerker, tensy spesifiek anders vermeld.

Soos voorheen genoem, kan die nul–eliminasie metodes wat in hierdie projek ondersoek was, in drie vlakke van optimisering verdeel word, naamlik:

Vlak 1: nul–eliminasie in die selfleringsfase,

Vlak 2: nul–eliminasie in die beramingsfase, en

Vlak 3: die opstel van 'n vektor van nie–nul relasie–inskrywings in die leerfase, vir latere gebruik in die beramingsfase. Vlak 3 kan dus gesien word as 'n metode om die voordele van vlak 1 te behou en te gebruik as voorkennis in die beramingsfase, om dus verdere onnodige berekeninge te elimineer.

In die onderstaande tabel, Tabel 3.4.2, word die verloop van die uitvoertyd vertoon soos die verskillende vlakke van nul–eliminasie by die oorspronklike programmatuur gevoeg word. Daar word begin deur die uitvoertye van elke nul–eliminasievlak afsonderlik te beskou, in gevalle O2 tot O4. In gevalle O5 tot O7 word die uitvoertye van die samevoeging van die afsonderlike nul–eliminasievlakke getoon.

Tabel 3.4.2: Tydoptimisering van die wasige proses:

Uitvoertyd van leer van 150 punte, en beraming van 296 punte van die Box en Jenkins gasoond–data, met mediaan–ontwasiging [3].

Vlak van Optimisasie:	Uitvoertyd (sekonde):
O1) Geen	9,1
O2) Vlak 1	5,4
O3) Vlak 2	5,0
O4) Vlak 3	5,8
O5) Vlak 1 & Vlak 2	3,7
O6) Vlak 1 & Vlak 2 & Vlak 3	3,4
O7) Vlakke 1&2&3 & Sentroïde	3,0

Geval O7 is dus 'n verteenwoordiging van die samevoeging van al die vlakke van

nul–eliminasie en ook die gebruik van sentroïde ontwasiging. Al hierdie simulasies is uitgevoer met die Box en Jenkins [44] gasoond datastel. [41] gee aanleiding tot die aaname dat die simulasieresultate empiries sal geld vir ander datastelle ook.

3.4.2.1. Uiteensetting van resultate van tabel 3.4.2:

Geval O1: Hierdie is die uitvoertyd van die oorspronklike RSK–algoritme [39]. Die uitvoertyd van 9.1 sekonde is dus die vergelykingsperk van die volgende gevalle.

Geval O2: **Nul–eliminasie in die selfleringsfase:** Die uitvoertyd van geval O2 is 1.69 keer vinniger as geval O1. Aangesien ongeveer 80% van die uitkomende verwasigde waardes nulle is, het hierdie stap 'n dramatiese verbetering gelewer.

Geval O3: **Nul–eliminasie in die beramingsfase** lewer selfs groter verbetering en die uitvoertyd van geval O3 is 1.82 keer vinniger as geval O1. Die invloed van die beramer is so groot aangesien sowat 60% van die berekeningstyd aan die beraming spandeer word. Die relasie–berekening is wel meer omvattend as die beramer, maar die beramingsfase verwerk ongeveer tweekeer soveel data–punte as die leerfase.

Geval O4: Die opstel van 'n **relasie–vektor** wat slegs die nie–nul inskrywings bevat is 'n samesmelting van die eerste twee vlakke van optimisasie. Dit lewer 'n kleiner verbetering in uitvoertyd. Die relatief klein verbetering in uitvoertyd kan toegeskryf word aan die feit dat die opstel van die relasievektor voordelig is in die beramingsfase, maar ekstra berekeningstyd afstaan in die leerfase.

Geval O5: Die kombinasie van **nul–eliminasie in beide die leer– en beramingsfase** het 'n

verdere verbetering te weeg gebring en die uitvoertyd van geval O5 is 2.45 keer vinniger as geval O1. Aangesien die beramings– en leerfase sekwensiëel geskied is daar geen probleme met die kombinasie van vlakke 1 en 2 nie.

Geval O6: Die byvoeging van die **nie–nul relasie–vektor by geval O5** het soos voorheen 'n verdere maar klein verbetering in uitvoertyd gelewer. Sien figuur 3.4.1.

Geval O7: Turbo–pascal program in aanhangsel 3.O1[41][23]: Die **drie vlakke van optimisasie tesame** lewer verre weg die grootste komponent van die verbetering in uitvoertyd, sodat die uitvoertyd van geval O6, **2.68** keer vinniger is as geval O1. 'n Verdere verbetering, na ongeveer **3.03** keer vinniger as geval O1, kan verkry word deur van die sentroïde ontwasigingsmetode gebruik te maak. Dit moet wel in gedagte gehou word dat hierdie metode van ontwasiging 'n benaderingsmetode is en gevolglik 'n verswakking in beramingsakkuraatheid sal veroorsaak. Die vorige gevalle van optimisasie het geen invloed op beramingsakkuraatheid nie.

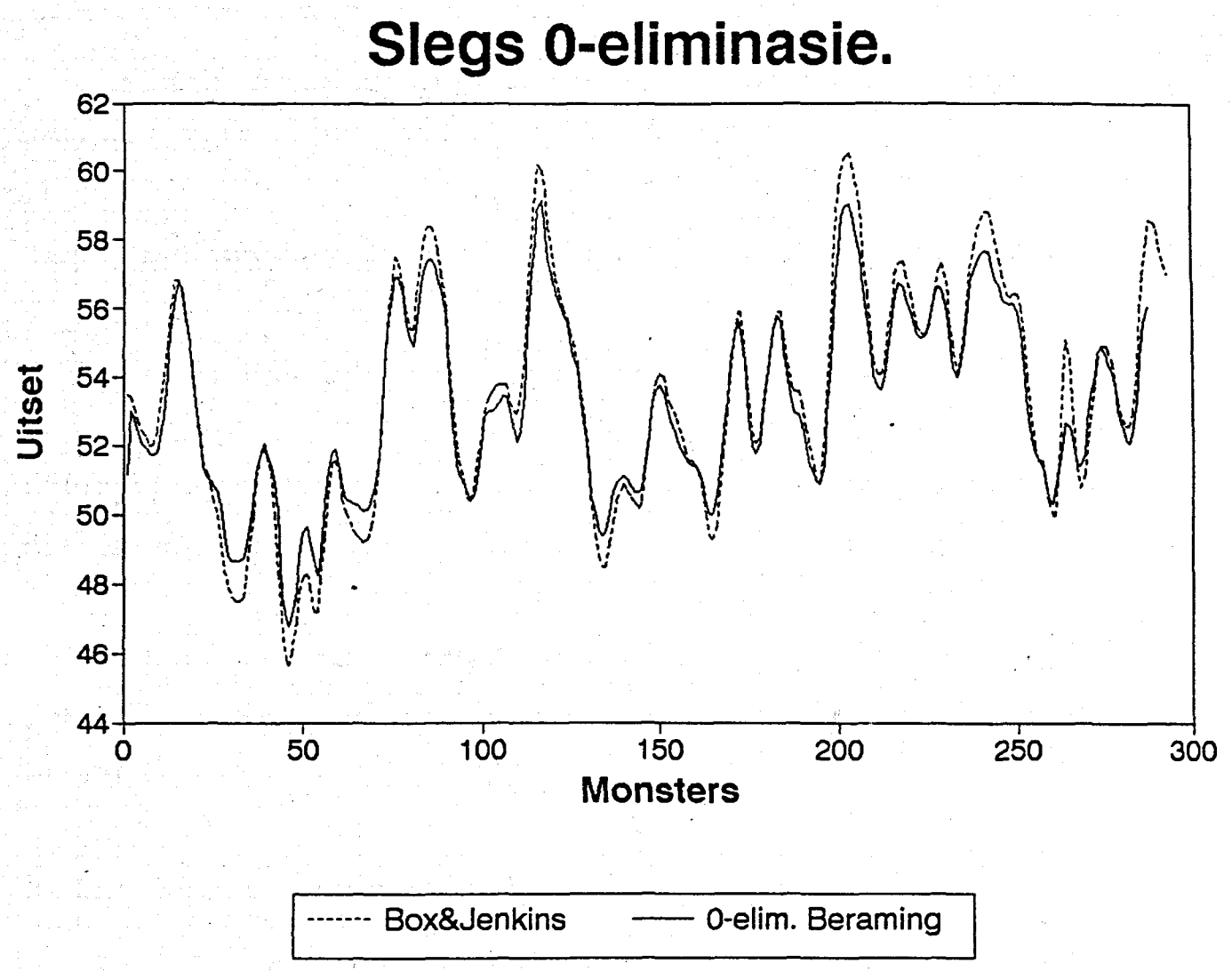
3.4.3 Afspeelstudie van tabel–verwasigingstegniek:

3.4.3.1 Tabel–verwasiging met 'n enkele vaste gebied–van–belang, en nul–eliminasie:

Slegs een gebied–van–belang in [–1,1] is in $256 = 2^8$ inkremente verdeel. Die verwasigde ekwiwalent van elke inkrementele waarde is gestoor as 'n wasige opsoektabel. Sentroïde ontwasiging, as ook alle vlakke van nul–eliminasie, is gebruik.

Die resultaat hiervan word vertoon in figuur 3.4.2. Die verswakking in beramingsfout, $J = \frac{1}{N}\Sigma_{k=1}^{N}(y_k - \hat{y}_k)^2$ na $J = 4.86$, is grootliks as gevolg van die

Figuur 3.4.1.


Slegs 0-eliminasie.
Uitset
62
60
58
56
54
52
50
48
46
44
0
50
100
150
200
250
300
Monsters
Box&Jenkins
0-elim. Beraming

verswakking in resolusie van die gebiede–van–belang, soos in hoofstuk 3.2 verduidelik, maar ook tot 'n minder mate as gevolg van die kwantifisering van die verwasigde waardes in die gebied–van–belang om die tabel–verwasiging te akkomodeer. Hierdie gevolgtrekking word beaam in die volgende afdeling, waar afsonderlike gebiede–van–belang vir elke veranderlike gebruik word.

Die uitvoertyd het aansienlik verbeter tot 1,53 s, as gevolg van die bogenoemde program–aanpassings.

3.4.3.2. Tabel–verwasiging met afsonderlike gebiede–van–belang, en nul–eliminasie:

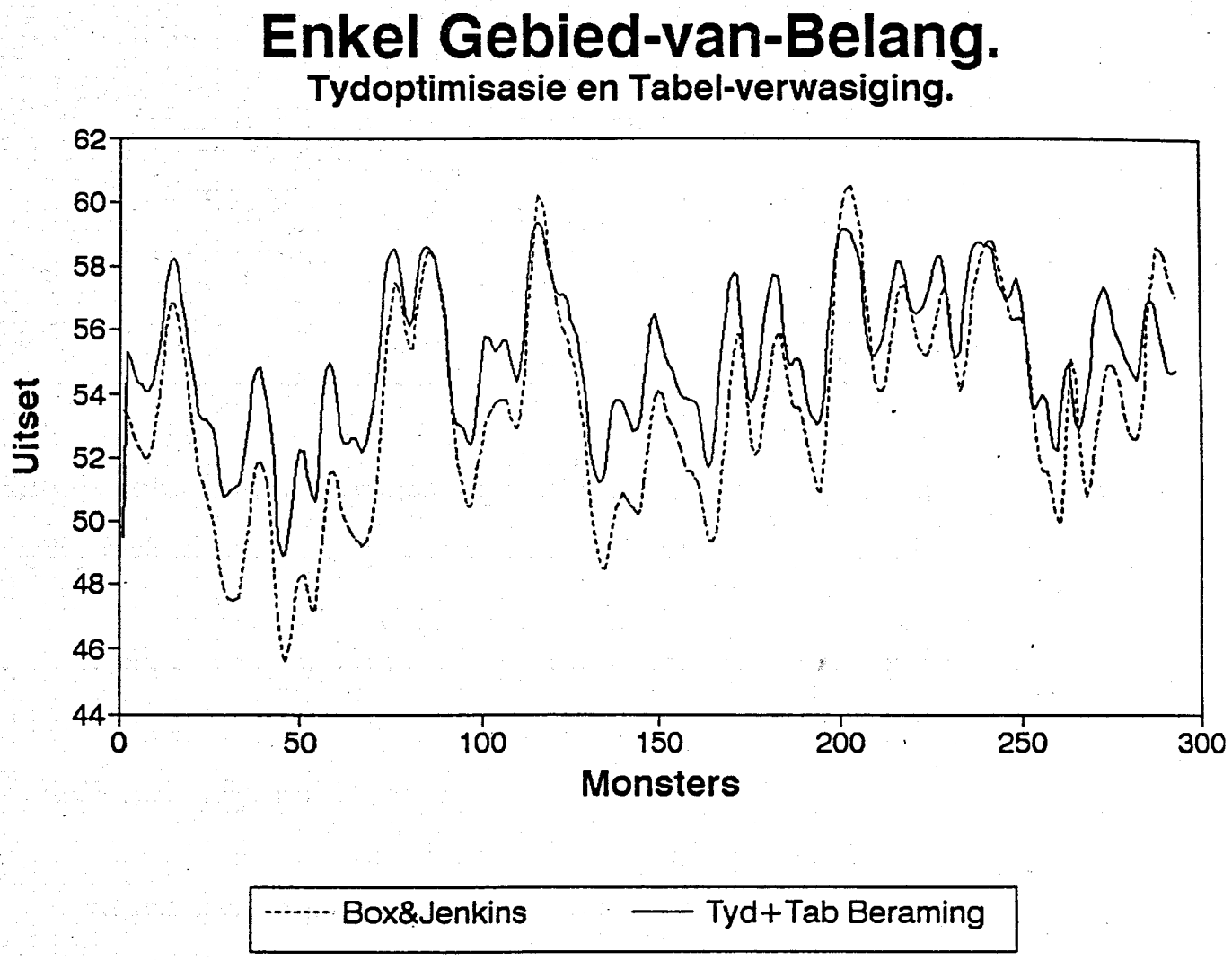
Die **afsonderlike gebiede–van–belang** is genormaliseer en spesifiek verdeel vir elke stelselveranderlike, met die oog op om die maksimum moontlike resolusie in elke wasige opsoektabel te verkry. Die res van die optimiseringsprosedure, soos voorheen verduidelik, is ook gebruik.

Die resultate van hierdie simulasie word in figuur 3.4.4 vertoon. Die uitvoertyd het verswak, in vergelyking met die vorige geval, na 1,7 s. Die beramingsfout het dramaties verbeter, in vergelyking met die vorige geval, na $J = 0.587$.

3.4.3.2.1. Resultate van tabel–verwasiging:

Soos voorheen in afdeling 3.2 verduidelik, vind tabel–verwasiging sy oorsprong in 'n genormaliseerde gebied–van–belang vir elke veranderlike. Verder het hierdie beginsel van normalisasie gelei tot die gebruik van slegs 'n enkele genormaliseerde gebied–van–belang. Die enkele gebied–van–belang het tot gevolg gehad dat sekere van die stelselveranderlikes 'n laer resolusie van besluitneming het, as wat die geval van afsonderlike

Figuur 3.4.2.


Enkel Gebied-van-Belang.
Tydoptimisasie en Tabel-verwasiging.
Uitset
62
60
58
56
54
52
50
48
46
44
0
50
100
150
200
250
300
Monsters
Box&Jenkins
Tyd+Tab Beraming

gebiede–van–belang sou gewees het. Hierdie laer resolusie veroorsaak 'n mate van verswakking in die akkuraatheid van die uitsetberaming.

Die resultate word opgesom in tabel 3.4.3. Alle resultate is verkry met die gebruik van sentroïde ontwasiging. Alle vlakke van tydoptimisasie is ook toegepas.

Tabel 3.4.3: Resultate van tabel–verwasiging.

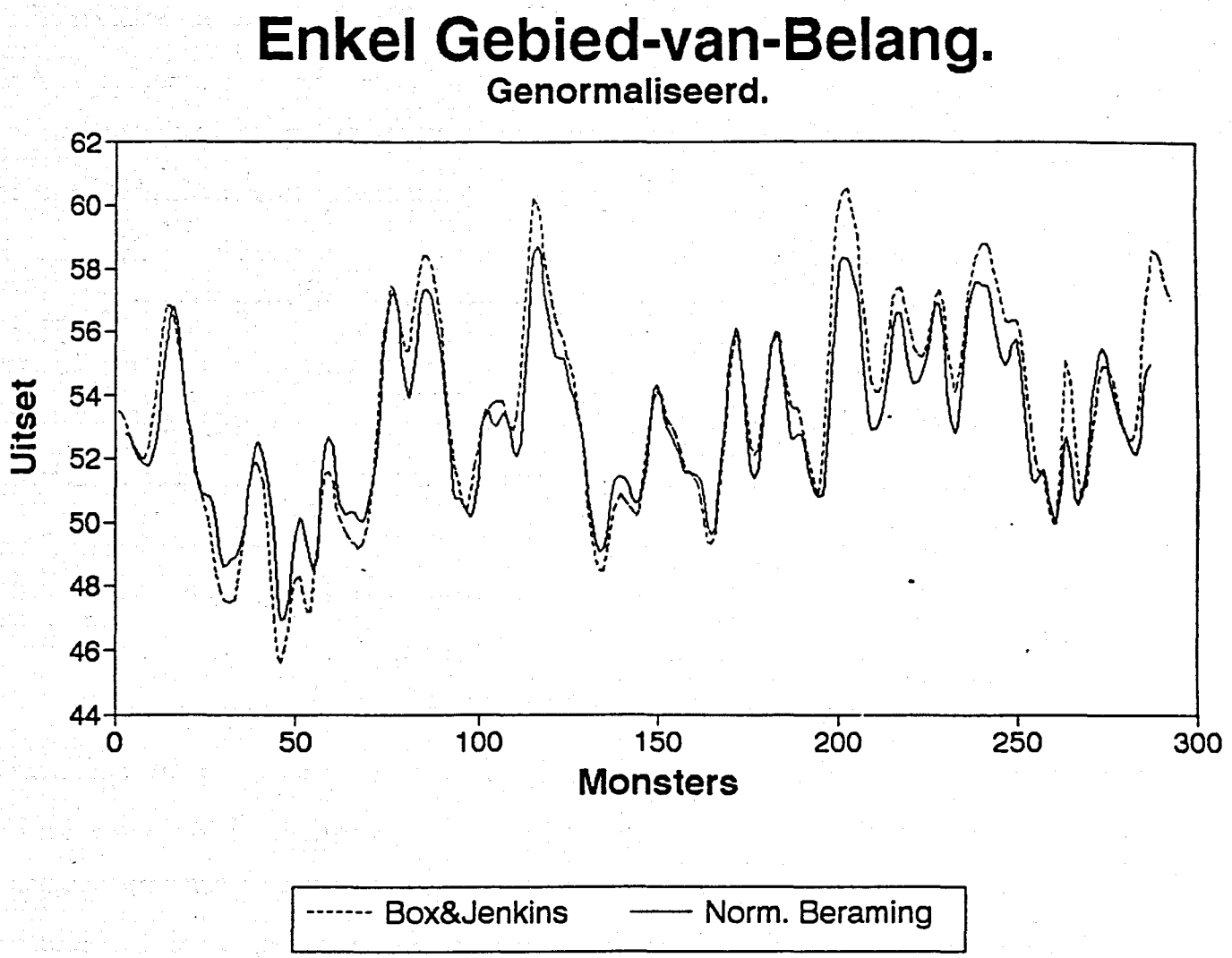
	Uitvoertyd(Sek.)	Fout J.
T1)	2.2	4.85
T2)	1.53	4.86
T3)	1.7	0.587

Geval T1: Program in aanhangsel 3.T1: Studie van die uitwerking van die gebruik van 'n enkele genormaliseerde gebied–van–belang, sonder die gebruik van tabel–verwasiging.

Hierdie stap het tot gevolg gehad dat die gemiddelde–kwadraat–fout, $J = \frac{1}{N}\Sigma_{k=1}^{N}(y_k - \hat{y}_k)^2$ verswak het van ongeveer $J = 0{,}4$ na $J = 4{,}85$. Die grafiese resultate word in figuur 3.4.3 vertoon.

Omdat daar minder gebiede–van–belang is, is daar gevolglik ook minder berekeninge tydens verwasiging nodig, en daarom het die uitvoertyd verder geval na 2,2 sekonde, in vergelyking met die vorige afdeling se resultate.

Figuur 3.4.3.


Enkel Gebied-van-Belang.
Genormaliseerd.
Uitset
62
60
58
56
54
52
50
48
46
44
0
50
100
150
200
250
300
Monsters
Box&Jenkins
Norm. Beraming

Geval T2: Program in aanhangsel 3.T2: Studie van die uitwerking van die gebruik van Tabel–verwasiging en 'n enkele vaste genormaliseerde gebied–van–belang.

Die gebruik van Tabel–verwasiging het feitlik geen verdere verswakking in J veroorsaak nie, $J = 4.86$, maar het egter 'n dramatiese verbetering in die uitvoertyd na 1,53 sekonde veroorsaak. **Sien figuur 3.4.2.**

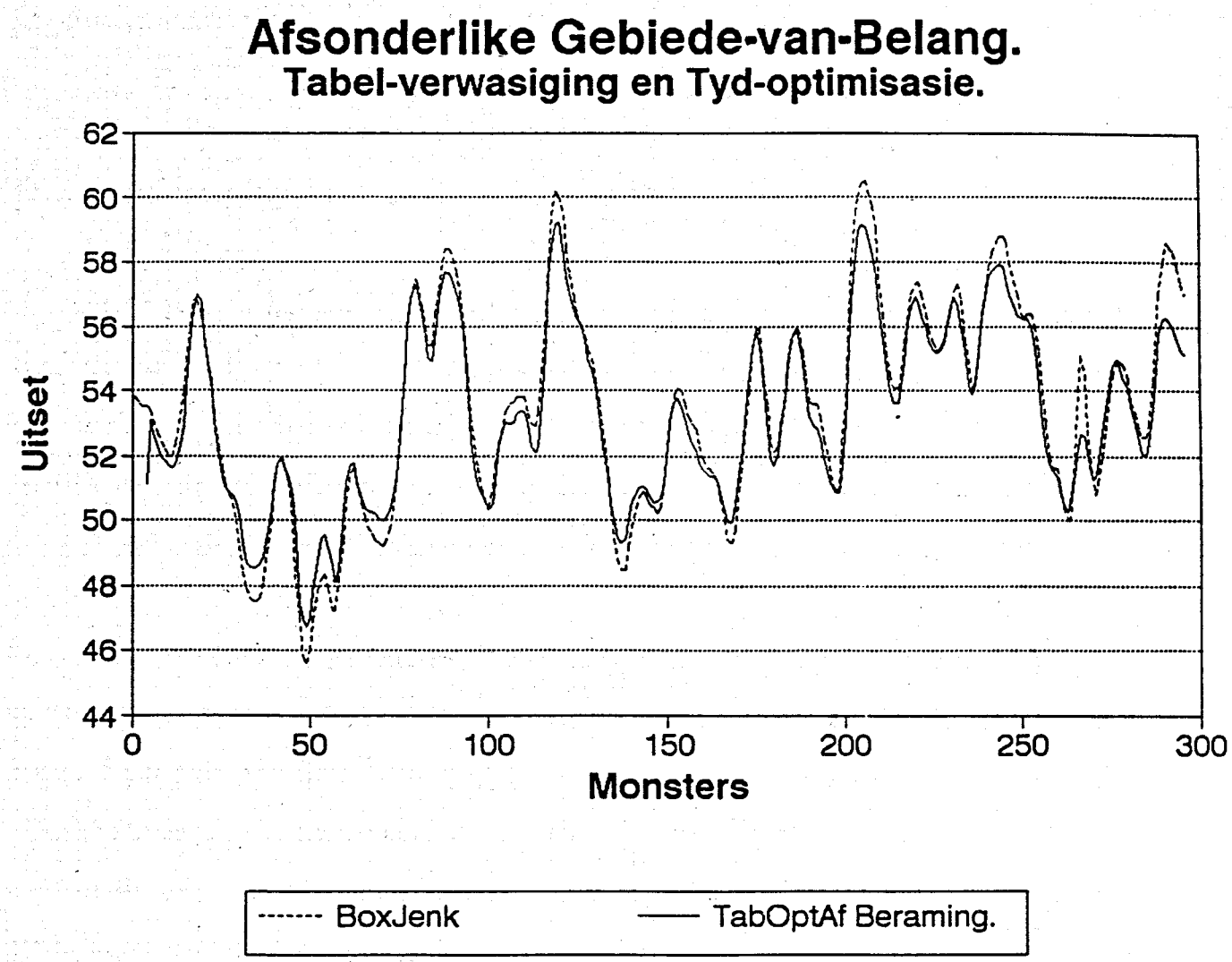
Geval T3: Program in aanhangsel 3.T3: Studie van die gebruik van Tabel–verwasiging met afsonderlike gebiede–van–belang. **Sien Figuur 3.4.4.**

Soos verwag is, het die uitvoertyd weer effens verswak, in vergelyking met geval T2, na 1,7 sekonde. Die akkuraatheid van die beraming het egter dramaties verbeter, en 'n fout $J = 0{,}587$ opgelewer. Hierdie J is amper net so goed soos die oorspronklike algoritme van Geval O1, waar $J = 0{,}42$ was. Die uitvoertyd is egter ongeveer 5.4 keer beter.

3.5. Uitvoertydoptimisasie van rekursiewe SK–beramer.

Om so min as moontlik eksterne faktore te hê wat die proses beïnvloed, is die ekperimentele simulasies van die vorige twee afdelings slegs met die nie–rekursiewe RSK–algoritme[39] uitgevoer. Dit moet egter in gedagte gehou word dat die apparatuurberamer wat uiteindelik geïmplementeer sal word, alleenstaande moet funksioneer. Die beste keuse, soos in afdeling 2.5.3 verduidelik van so 'n tipe beramer, is 'n rekursiewe beramer. Dit is dus sinvol om die resultate van die vorige afdeling se simulasies toe te pas op die rekursiewe SK–algoritme[23].

Figuur 3.4.4.


Afsonderlike Gebiede-van-Belang.
Tabel-verwasiging en Tyd-optimisasie.
Uitset
62
60
58
56
54
52
50
48
46
44
0
50
100
150
200
250
300
Monsters
BoxJenk
TabOptAf Beraming.

Tabel 3.5.1: Uitvoertydoptimisasie van die rekursiewe algoritme.

Leer van 150 data–punte en beraming van 296 waardes.

Optimisasie.	Uitvoertyd(s)	Fout(J).
R1) Geen	9.66	0.791
R2) Nul–eliminasie	6.1	0.791
R3) Nul–elim.&Tabel	1.97	1.465

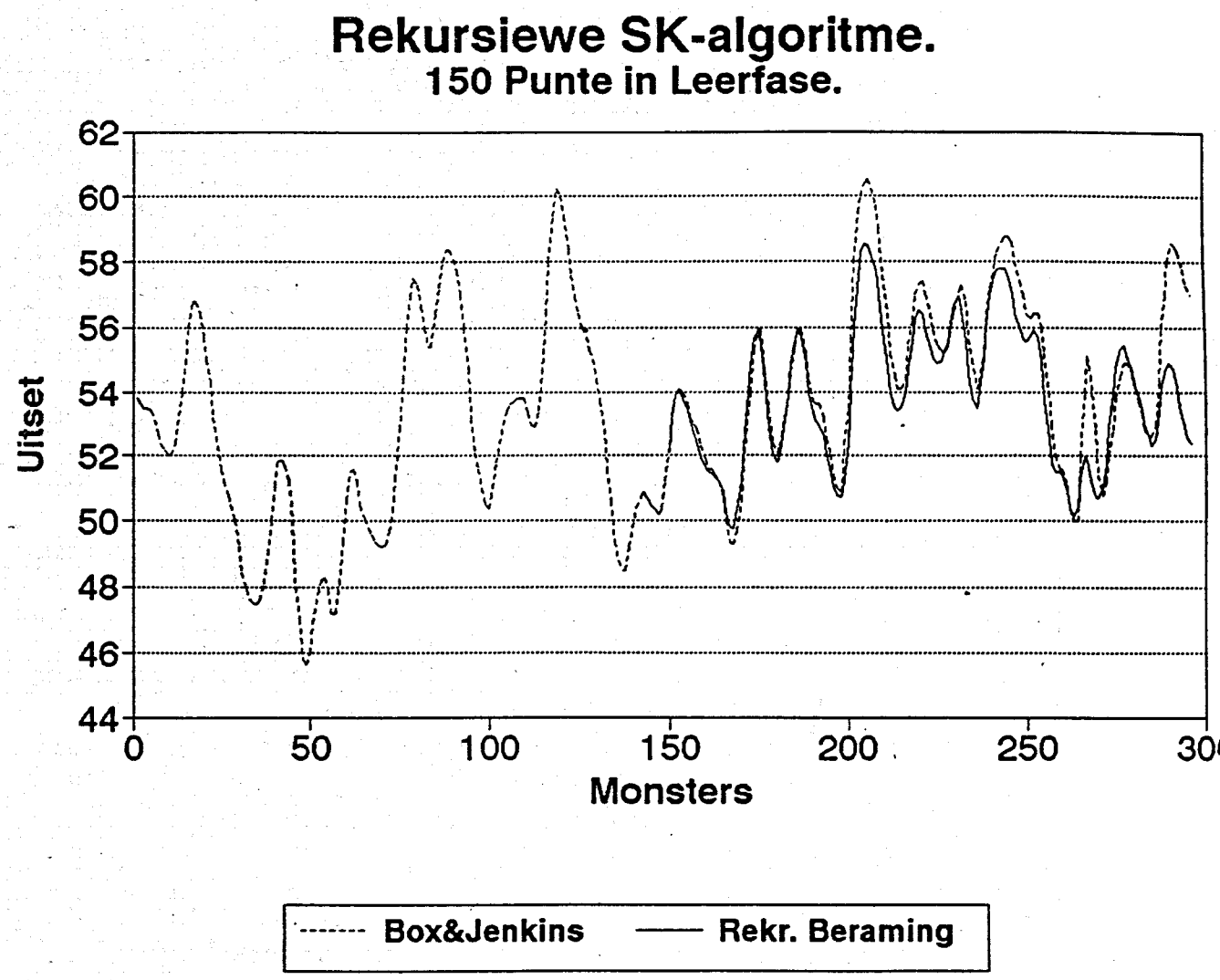
3.5.1. 'n Uiteensetting van die resultate van tabel 3.5.1.

Geval R1: Program in aanhangsel 3.R1: Hierdie resultaat word in **figuur 3.5.1** getoon en vorm die verwysingsraamwerk van die opvolgende resultate. Die leervenster was die eerste 150 waardes van die Box & Jenkins datastel [44]. Die beramingsfase herskat die eerste 150 waardes en skat dan 146 waardes vooruit. Die beramingsfase is aanvanklik baie akkuraat, maar na verwagting verswak die beraming, soos verder in die toekoms voorspel word.

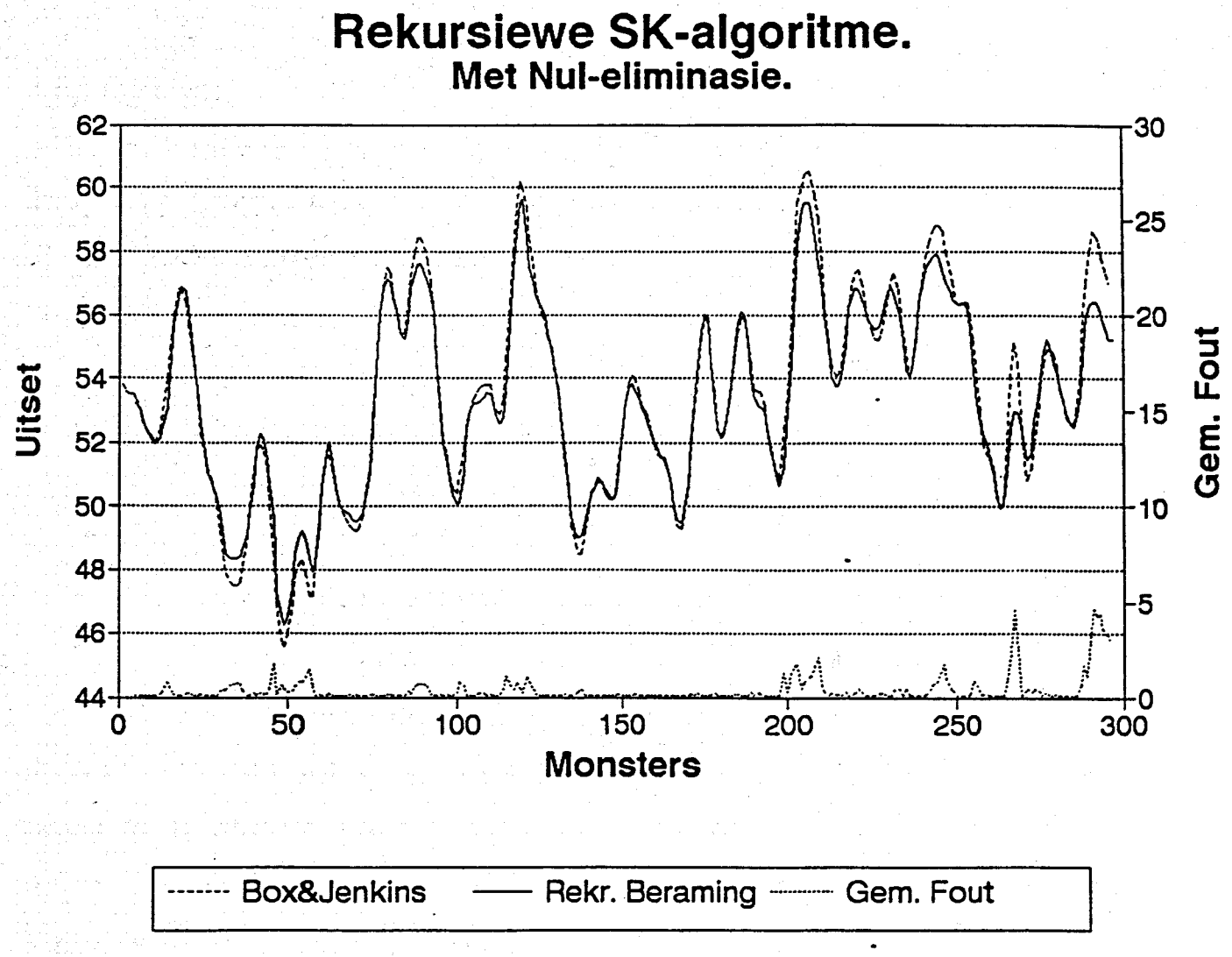
Geval R2: Program in aanhangsel 3.R1: Sentroïde ontwasiging en alle vlakke van nul–eliminasie, soos in afdeling 3.4.2 bespreek, word hier toegepas op die rekursiewe proses. **Figuur 3.5.2** toon dat die beraming baie akkuraat bly, maar wel swakker is as geval R1, waarskynlik as gevolg van die gebruik van sentroïde ontwasiging. Die verloop van die fout word onder aan dieselfde figuur getoon. Dit is duidelik sigbaar dat die fout sistematies groter word soos verder vooruit geskat word.

Geval R3: Program in aanhangsel 3.R1: Alle vlakke van tydoptimisasie en Tabel–verwasiging, soos in afdeling 3.4.3 uiteengesit, word toegepas. Die resultate van hierdie simulasie word in **figuur 3.5.3** getoon. Die beramingsakkuraatheid, alhoewel swakker as geval R2, is nog steeds aanvaarbaar vir 'n rekursiewe proses. Ook in hierdie

Figuur 3.5.1


Rekursiewe SK-algoritme.
150 Punte in Leerfase.
Uitset
62
60
58
56
54
52
50
48
46
44
0
50
100
150
200
250
Monsters
Box&Jenkins
Rekr. Beraming

Figuur 3.5.2


Rekursiewe SK-algoritme.
Met Nul-eliminasie.
Uitset
Gem. Fout
Monsters
62
60
58
56
54
52
50
48
46
44
30
25
20
15
10
5
0
0
50
100
150
200
250
300
Box&Jenkins
Rekr. Beraming
Gem. Fout

geval is die totale 296 punte beraam om 'n indrukwekkende uitvoertyd van 1.97 sekonde te lewer. Dit is om–en–by 'n vyfvoudige verbetering in uitvoertyd. Die resultate verkry in geval R3 is bevredigend en vorm die basis van die apparatuurontwerp wat in hoofstuk 4 uiteengesit sal word.

3.6. 'n Afspeelstudie van die RSK–algoritme aangaande veelvuldige insette:

3.6.1. Inleiding:

Die besluitnemingsbasis van 'n wasige beheerstelsel is opgesluit in die beheerreëls. Soos in [1] bespreek is, behoort alle reëls in parallel geaktiveer te word, en die resultaat van elke reël word dan gekombineer, deur gebruik te maak van een of ander vorm van wasige inferensie[25].

Meervoudige insette en meervoudige uitsette word geïnterpreteer deur die linguistiese "en" operator te gebruik, soos byvoorbeeld:

Indien Inset 1 = ZE en Inset 2 = NS en Inset 3 = PS en.....Inset n....

dan is Uitset 1 = NB en Uitset 2 = PS en .....Uitset k.

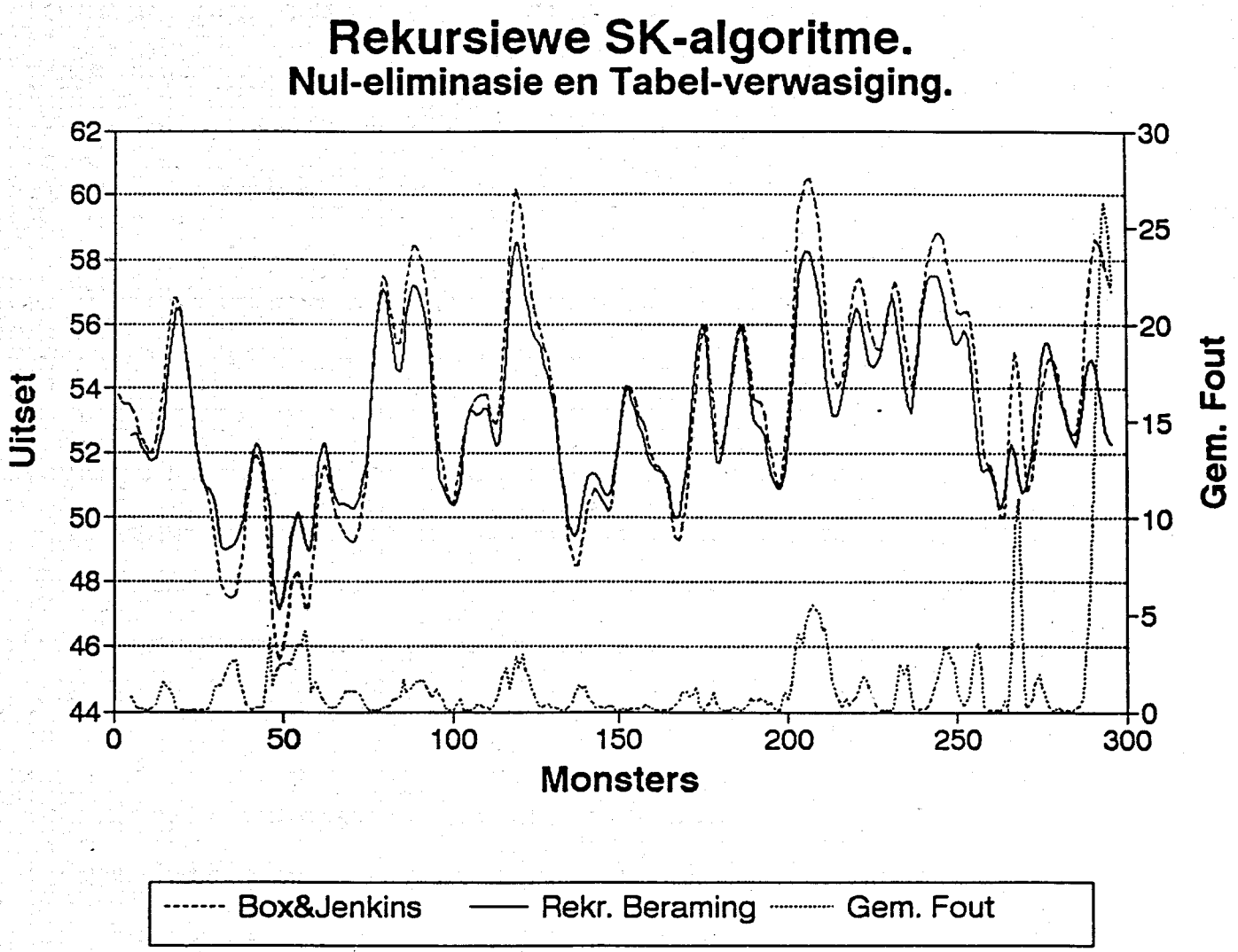
waar: $n = 1,2,...N$; $k = 1,2,...K$; N = *aantal insette*; K = *aantal uitsette.*

Die parallel uitvoer van reëls, en die feit dat elke linguistiese reël uitgebrei kan word, sou kon bewys dat wasige logika goed daar toe leen om multi–inset multi–uitset stelsels te akkomodeer. Die realiteit is egter aansienlik anders as die teorie. Veral selflerende algoritmes [26] [6] [32] [39], is gewoonlik toegespits op multi–insetstelsels, met slegs 'n enkel–uitset.

3.6.2. Die twee–inset enkel–uitset SK–model:

[26] het in hul publikasie 'n twee–inset enkel–uitset bilineêre model van 400

Figuur 3.5.3


Rekursiewe SK-algoritme.
Nul-eliminasie en Tabel-verwasiging.
Uitset
Gem. Fout
Monsters
62
60
58
56
54
52
50
48
46
44
30
25
20
15
10
5
0
0
50
100
150
200
250
300
Box&Jenkins
Rekr. Beraming
Gem. Fout

datapunte beskryf, van die volgende vorm:

$$y[k] = 0{,}8y[k-1]u_1[k] + 0{,}5u_1[k-1]y[k-2] + u_2[k-4] + \alpha\, e\,[k]$$

met: α *'n ruisfaktor, wat vir hierdie simulasies as 0 aanvaar word.*

Daar word egter geen melding gemaak van die presiese uitsetfunksies in [26] nie. Vir die opvolgende simulasies moes twee insette gegenereer word, en die volgende vorm is gekies: *Inset 1 = sin 2θ × cos θ* en *inset 2 = cos 2θ × sin θ.* Slegs die eerste siklus van θ is gebruik, om te verseker dat die RSK–algoritme se leerproses nie voordeel trek uit periodiese insette wat herhaal nie.

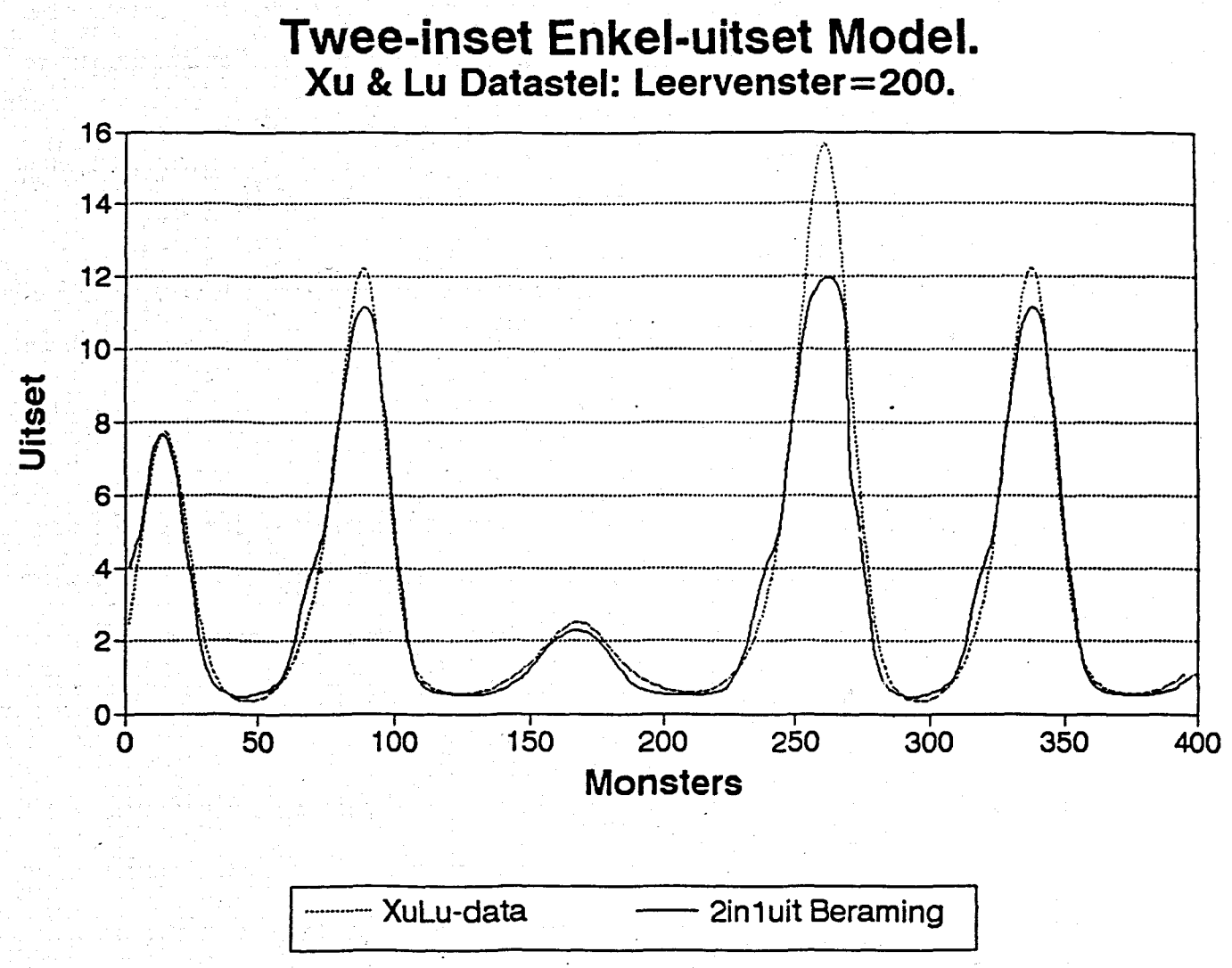
Die leervenster is, soos voorheen, beperk tot die helfte van die stelseldatastel. Die leerfase behels dus 200 punte en die beramingsfase herskat daardie punte en voorspel dan 200 verdere punte. Die rekursiewe beramer [23] is gebruik met sentroïde ontwasiging. Die uitvoertyd en fout, *J*, was eerstens sonder uitvoertydoptimisasie, soos voorheen gedefinieer, bepaal, en daarna met uitvoertydoptimisasie bepaal ter vergelyking. Die resultate van hierdie uitvoertydafspeelstudie word in **tabel 3.6.2** saamgevat en in **figuur 3.6.1** getoon. Die leser sal opmerk dat daar 'n tienvoudige verbetering in uitvoertyd te bespeur is in hierdie geval.

Tabel 3.6.2: Twee–inset en een–uitset Xu & Lu stelsel:

Vlak van Optimisering	Uitvoertyd (s)	Fout(J)
M21) Geen	64.81	0.441
M22) Alle Vlakke	6.48	0.540

Die resultate in tabel 3.6.2 bewys dat dit hoogswaarskynlik aanvaarbaar sou wees om hierdie twee–inset een–uitset stelsel in apparatuur te realiseer. Selfs al word 'n

Figuur 3.6.1


Twee-inset Enkel-uitset Model.
Xu & Lu Datastel: Leervenster=200.
Uitset
Monsters
0
2
4
6
8
10
12
14
16
0
50
100
150
200
250
300
350
400
XuLu-data
2in1uit Beraming

konserwatiewe tienvoudige verbetering in uitvoertyd van programmatuur na apparatuur voorspel, is die uitvoertye en beramingsfout, met uitvoertyd–optimisasie, nog steeds aanvaarbaar.

Alhoewel hierdie afdeling die twee–inset een–uitset model as die optimum keuse vir apparatuur realisasie bewys het, was die een–inset een–uitset model nog steeds vir die apparatuur gebruik, om redes wat in hoofstuk 4 duidelik sal blyk.

3.7 Die drie–inset enkel–uitset model:

Aangesien die twee–inset een–uitset model in die vorige afdeling so bevredigend presteer het, is dit sinvol om die model nog verder uit te brei na 'n drie–inset een–uitset model.

Daar was geen drie–inset een–uitset datastel aan die outeur bekend nie, en gevolglik geen data wat deur ander navorsers beproef was, soos in die twee–inset een–uitset geval nie. Die data moes dus gegenereer word, en om impiries te bewys dat die datastel van bruikbare formaat is, is vyf verskillende stelsels gebruik, naamlik:

1) $y[k] = 0{,}8\ y[k-1]u_1[k] + 0{,}5\ u_1[k-1]y[k-2] + u_2[k-4] + u_3[k-5]$
2) $y[k] = y[k-3] + u_1[k-1] + u_2[k-4] + u_3[k-5]$
3) $y[k] = 0{,}5\ y[k-3] - 0{,}5\ y[k-1] + u_1[k-1] - 0{,}7\ u_1[k-3] + u_2[k-4] + u_3[k-5]$
4) $y[k] = 7u_1[k-1] + 2u_2[k-4] - 20u_3[k-5] + y[k-2]$
5) $y[k] = 2u_1[k-3] + u_2[k-10] - 20u_3[k-2] + y[k-2]$

Vir insette is sin–cos–funksies en wit–ruis gebruik. Die leerfase het die eerste 240, van die 480 totale data–punte betrek. Die gemiddelde J en die gemiddelde uitvoertyd van

al vyf stelsels, met en sonder uitvoertyd–optimisasie, is bepaal, en word in tabel 3.7.1 opgesom.

Tabel 3.7.1:Drie–inset een–uitset model, gemiddelde waardes:

Vlak van Optimisering	Uitvoertyd (s)	Fout
M31) Geen	431,87	0,52
M32) Alle Vlakke	41,2	1,07

Behalwe dat daar 'n ongeveer viervoudige verlenging in die uitvoertyd, in vergelyking met die twee–inset een–uitset stelsel is, word die drie–inset een–uitset stelsel te groot om in apparaat te realiseer. Die relasie is byvoorbeeld 'n matriks van $5 \times 5 \times 5 \times 5 \times 5 = 3125$ reëls. Op gronde van die stadige uitvoertyd, en die kompleksiteit van die drie–inset een–uitset stelsel, word dit nie as ideaal vir apparatuur–realisasie beskou nie.

3.8. Opsommend:

Die wasige proses se uitvoertyd het aansienlik verbeter van **9,2 sekonde na 1,7 sekonde** vir die leer van 150 punte en die beraming van 296 punte van die Box en Jenkins industriële gasoond data [44]. Dit is dus meer as 5 keer vinniger as die oorspronklike algoritme. Die uitskakeling van onnodige berekeninge deur nul–eliminasie tegnieke het die grootste bydrae gehad tot hierdie verbetering. 'n **Verdere indrukwekkende verbetering** in die uitvoertyd is te weeg gebring deur van **tabel–verwasiging** gebruik te maak. In hierdie afdeling moes 'n afspeelstudie gedoen word tussen akkuraatheid en uitvoertyd. Die mees voordelige resultaat was bewys as die gebruik van **afsonderlike gebiede–van–belang**, wat 'n effense verswakking in uitvoertyd veroorsaak het, maar nog steeds goeie akkuraatheid

gehandhaaf het. Vinniger uitvoertye is dus moontlik, maar dan moet akkuraatheid ingeboet word. Die lesse wat geleer is met die nie–rekursiewe proses is toegepas op die rekursiewe beramer om sodoende die mees optimum alleenstaande apparatuur te kan ontwerp. Die opvolgende hoofstuk doen verslag van hierdie ontwerp. Die twee–inset en een–uitset stelsel het goed gepresteer, en is 'n goeie keuse vir 'n apparatuur realisasie, vanaf 'n uitvoertyd oogpunt. Die drie–instet een–uitset stelsel is te stadig en te kompleks vir apparatuur realisasie.

ooOOoo

HOOFSTUK 4: DIE WASIGE APPARATUUR.

" Knowledge comes, but wisdom lingers."

Alfred Lord Tennyson,
Locksley Hall.

HOOFSTUK 4: DIE WASIGE APPARATUUR.

4.1 Inleiding:

Prosessering, wat op wasige logika beginsels gebaseer is, is besonder berekeningsintensief. Die oorgrootte meerderheid van industriële toepassings van wasige beheer kan steeds uitgevoer word met mikroprosesseerders [6] [45] [46] [47] [48] [49] [50] [51] [52] [53] [54]. Sekere toepassings, soos outonome robotnavigasie [55][56]; beheer van industriële robotte [57]; vliegtuig outo–loodse; vlugbeheer [58]; en die prosessering van groot hoeveelhede data, soos in beeldprosessering [1], is aansporing vir die ontwerp van hoë prosesseringspoed wasige apparatuur.

4.2 Vorige navorsing in die gebied van wasige apparatuur.

Die ontwerp van die wasige apparatuur–realisasie wat in hierdie hoofstuk uiteen gesit word, is gemotiveer deur die werke van [59] [60] [61] [62] [63] [64]. [59] het in 1986, in 'n baanbrekende publikasie, 'n eenvoudige digitale wasige afleidingsenjin voorgestel. In 1988 het [60] 'n analoog hoë spoed wasige apparatuurstelsel voorgestel, wat teen ongeveer 1 000 000 wasige inferensies per sekonde kon funksioneer. In 1990 het [61] ook 'n eenvoudige 4–bis wasige prosesseerder voorgestel, wat gevolg het op die werk van [59]. Uit ervaring, met toepassings van die 4–bis prosesseerder, het dit duidelik geblyk dat die akkuraatheid van die prosesseerder uitgebrei moes word na 8–bis of 16–bis. In 1991 het [62] toe 'n 8–bis wasige prosesseerder, wat tot 256 reëls kon ondersteun, voorgestel. Hierdie werk het gelei tot 'n kommersiële beskikbare wasige prosesseerder stelsel [65], wat hedendaags groot ondersteuning geniet. In 1991 het navorsers van [63] LIFE (Laboratory for International Fuzzy Engineering Research) ook 'n wasige prosesseerder, met tot vyftig keer beter uitvoertye, as mikroprosesseerder gebaseerde stelsels, voorgestel. Die struktuur

van die wasige saamstel–stelsel, wat deur [64] voorgestel is, het ook 'n groot invloed gehad op die ontwerpstrategie van hierdie verhandeling.

4.3 Die XILINX ontwerpstrategie:

Inleiding:

Die XILINX programmeerbare logika stelsel [66] was gekies as ontwerpsplatform van die wasige apparatuurontwerp. Die aanpasbaarheid van hierdie apparatuur, met meegaande programmatuur stelsel, was as 'n groot voordeel beskou vir hierdie projek. Die interne ontwerp van die Xilinx–vlokkies is 'n hoogs geïntegreerde stelsels, wat tussen 2 400 en 13 000 logikahekke kan akkommodeer. Die vlokkie is verdeel in 'n aantal programmeerbare blokke. Afhangende van die spesifieke vlokkie is daar van 64 tot 900 van sulke programmeerbare blokke. Elke blok het vier insette en twee uitsette, met ingeboude geklokte wipbane [66].

4.3.1 Beperkinge van die blokontwerpsmetodiek:

Waarskynlik die grootste beperking op enige ontwerp, wat die Xilinx–vlokkies gebruik, is die aantal programmeerbare blokke wat op 'n vlokkie beskikbaar is, en die aantal insette en uitsette beskikbaar vir elke blok. Beskou enige van die ontwerpe in aanhangsel 4. Die gebruiker moet dus die grootte en die maksimum klokfrekwensie van 'n vlokkie vooraf oorweeg.

Die ontwerpstegnieke wat gebruik word om die interne stroombane te ontwerp word ook baie sterk beinvloed deur die blok–argitektuur. Hierdie feit belemmer die benutting van optimum ontwerpsmetodes, soos "bit–slicing" [67], en Wallace–strukture

[68]. Hierdie metodes is uiters geskik vir parallelle VLSI–implementasies, maar kan nie gebruik word, met die beperkte aantal programmeerbare blokke wat beskikbaar is in die Xilinx–vlokkies, nie. Die wasige ontwerp moes dus beperk word tot seriale en serie–parallelle metodes [67]. Hierdie metodes word in meer diepte verduidelik in afdeling 4.4.2.

4.3.2 Beperkinge aangaande inter–blokverbindings:

Om 'n logiese stroombaan saam te stel moet die insette en die uitsette van programmeerbare blokke aan mekaar gekoppel word. Sien enige ontwerspuitleg in aanhangsel 4. In teenstelling met meer konvensionele programmeerbare stelsels, soos GAL'S [69], PAL'S [70], ensovoorts, is die vertraging tussen insette en uitsette nie 'n vaste waarde nie, so ver dit die Xilinx–vlokkie aangaan. Die vertraging van die Xilinx–stroombaan is afhanklik van die aantal blokke en verbindings wat tussen insette en uitsette geplaas word. Die langste vertraging tussen inset en uitset moet deur simulasie bepaal word, en dit bepaal ook die maksimum klokfrekwensie waarteen die stroombaan kan funksioneer.

Die verbindings van die interne stroombaan word vermag deur 'n baie uitgebreide verbindingsnetwerk. Die verbindingsnetwerk, en die hoeveelheid moontlike verbindings, is egter vasgestel, en alhoewel dit 'n baie omvattende netwerk is, is daar wel 'n beperking op die aantal verbindings wat moontlik is. In hierdie projek was so 'n hoë persentasie van die totale aantal blokke van 'n vlokkie gebruik, dat die aantal verbindings 'n groot beperking op die ontwerp was. Hierdie beperking was in sekere ontwerpe, soos die relasieberekeninge in afdeling 4.4.3.2, 'n groter beperking as die blok–argitektuur. In hierdie ontwerp was daar nog 'n aantal blokke beskikbaar, maar die verbindingsnetwerk beperkings het daartoe gelei dat geen verdere stroombaan–uitbreidings gedoen kon word nie.

Die inter–blokverbindings lei ook tot 'n baie belangrike verdere beperking van die maksimum klokfrekwensie van 'n vlokkie. As deel van elke vlokkie is daar 'n netwerk van verbindingslyne wat spesifiek toegewy is aan die kloksein. Hierdie klok–netwerk word vir betroubaarheidsredes beperk tot slegs helfte van die maksimum klokfrekwensie van die vlokkie[66]. Dit is wel moontlik om die klok in te voer, teen die maksimum frekwensie, deur die gewone I/U–penne, maar dan kan die toegewyde klok–netwerk nie benut word nie en moet van die reeds oorbelaaide gewone verbindingsnetwerk gebruik gemaak word. Aangesien meeste van die ontwerpe in hierdie projek relatief ingewikkeld was, was daar ongelukkig geen alternatief as om die klok–netwerk te gebruik nie. Die vlokkies moes dus teen helfte die maksimum frekwensie bedryf word.

4.4 Beskrywing van die wasige apparatuur–ontwerp:

In die opvolgende afdeling word die navorsings– en ontwikkelingswerk getoon, wat gelei het tot die realisasie van wasige selflerende apparatuur. In die eerste gedeelte word die wiskundige funksies, wat die basis van die wasige algoritme vorm, bespreek. In die opvolgende deel word die sub–afdelings van die wasige selflerende algoritmes, soos in hoofstuk 3 beskryf, verduidelik. 'n Blokdiagram, die aantal logika–blokke, en die aantal verbindings van elke algoritme sub–element, word bespreek.

4.4.1 Apparatuur–realisasie van die SK–algoritme:

Soos in hoofstuk 3 verduidelik, was die SK– algoritme [23] aangepas om meer van pas te wees vir apparatuur realisasie. Die wasige proses bestaan uit vier sub–afdelings, naamlik: i) die verwasiger; ii) die relasie–identifikasie; iii) die wasige beramer; iv) die ontwasiger. Die vyfde sub–element van die wasige apparatuur realisasie is die geheue–adressering wat die eerste vier sub–afdelings oorkoepelend verbind en organiseer.

4.4.1.1 Basiese stelsel parameters:

i) Gebaseer op vorige ervaring met mikroprosesseerder–gebaseerde stelsels [43], en die werk van [61] [62], was besluit om die **interne bis–struktuur** van die wasige vlokkies tot **16–bis** te beperk. 'n Groot probleem wat in vorige navorsing [43] bespeur is, was dat wasige lekkasie, wat in hoofstuk 2.7 verduidelik is, meer algemeen plaas gevind het soos die interne bis–resolusie afneem. In [43] is empiries bewys dat die grootste verwagte verswakking in beramingsakkuraatheid sal plaas vind as gevolg van wasige lekkasie. Sien hoofstuk 5.2.2 vir 'n bespreking van eksperimentele resultate, wat hierdie verskynsel illustreer.

ii) 'n Geheue–segment van 32 k–greep is beskou as die grootste geheue–blok wat direk geadresseer kan word, met 'n redelike aantal, naamlik 15, adreslyne. Daar is dus besluit om 32 k statiese geheue, met 'n toegangstyd van 80 ns, beskikbaar te stel. Soos verduidelik word, in aanhangsel K, word die geheue in twee siklusse van 8–bis gelaai vanaf 'n IBM–versoenbare tipe rekenaar. Na hierdie proses voltooi is, word die geheue as 16–bis geheue gebruik deur die wasige vlokkies.

iii) Vanweë die 32 k–greep geheue beperking, moes die stelsels wat gemodelleer word, beperk word tot 3 000 16–bis data–punte. Dit transleer na 'n leervenster van 1 500 waardes.

iv) Soos voorheen genoem, is die grootste beperkende eienskap van die Xilinx–stelsel die aantal programmeerbare blokke wat per vlokkie beskikbaar is. Dit is dus sinvol om 'n vlokkie te gebruik met die grootste aantal programmeerbare blokke beskikbaar. Die koste van die Xilinx–programmatuur het dit slegs moontlik gemaak om die vlokkie met 9 000 logikahekke te gebruik. Die koste van elke vlokkie het daartoe gelei dat die 70 MHz vlokkies die enigste bekostigbare keuse was.

v) 'n Vooruitskatting van die aantal programmeerbare blokke wat benodig sal word, het getoon dat daar nie genoeg hulpmiddels vir die implementasie van die twee–inset en een–uitset stelsel sal wees nie. Die wasige selflerende apparatuur is daarom 'n enkel–inset– enkel–uitsetstelsel.

4.4.1.2 Samevatting van wasige apparatuur:

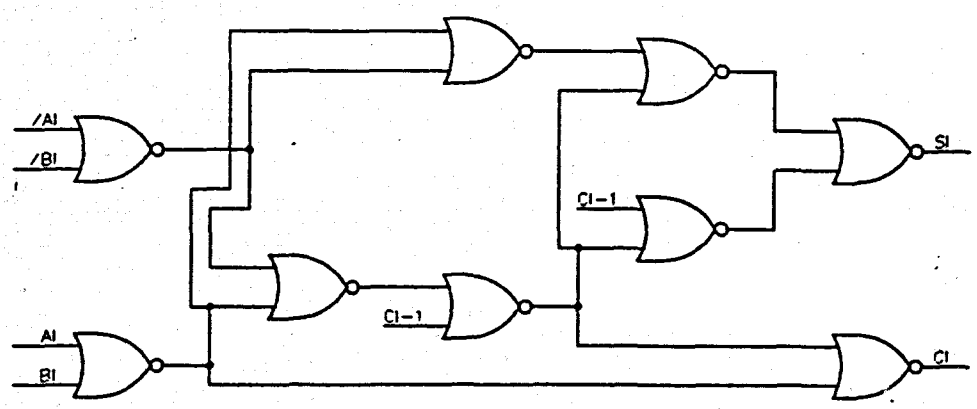
1) Enkel–inset enkel–uitset selflerende wasige stelsel, volgens SK–algoritme, met optimisasie en tabel–verwasiging. Sien hoofstuk 3.
2) Proses word beperk tot 3 000 ingesamelde waardes.
3) 16–bis interne struktuur.
4) Die Xilinx XC 3090–70, 9 000 logikahekke, teen 70 MHz word gebruik. Die ontwerp moes egter beperk word tot 35 MHz, soos verduidelik.
5) 32 k–greep SRAM, 80 ns geheue word voorsien vir die stoor van data.
6) 8–bis rekenaarkaart is as ontwerpsplatform gebruik. Die 8–bis data word omgeskakel na 16–bis vir gebruik deur die Xilinx–vlokkies.

4.4.2 Die wiskundige boublokke:

Om die SK–algoritme te kan implementeer in apparatuur, was dit nodig om die vier mees basiese wiskundige operatore, naamlik optel, aftrek, vermenigvuldig, en deel, op stroombaanvlak te ontwerp. Die stelsel beperkings, en veral die aantal programmeerbare blokke, was die grootste oorweging tydens hierdie proses. Na oorweging van die werk van [67] [68] [71] [72] [73] [74] [75] [76] [77] [78] [79] was besluit om die ontwerp te beperk tot serie–parallelle ontwerpstegnieke, aangesien parallelle–metodes te omslagtig is vir die beperkte hulpmiddels, en serie–metodes te stadig is vir hierdie toepassing.

4.4.2.1 Die optimale een–bis vol–opteller:

Die mees basiese element van enige wiskundige funksie, wat op serie–parallelle tegnieke gebaseer is, is die een–bis vol–opteller, en die meegaande afgeleide hiervan, die een–bis halfopteller. [79] het dertig verskillende optimale een–bis vol–optellers voorgestel. Hierdie voorgestelde vol–optellers is almal optimaal volgens een van dertig stelle beperkings. Die beste keuse van vol–optellers, vanuit hierdie voorstelle, was die stroombaan met die minimum vertraging en die minimum aantal stroombaan–elemente. Die optimum logikahekke vir die Xilinx stelsel was ook oorweeg. Die stroombaan word in figuur 4.4.1 getoon.



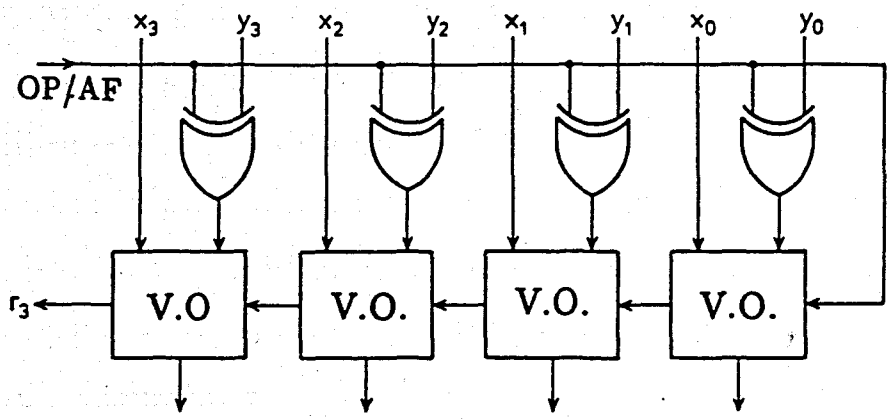
Figuur 4.4.1: Optimale 1–bis vol–opteller.

4.4.2.2 Die opteller– en aftrekkerontwerp:

In hoofstuk 3.2. was verduidelik dat die gebied–van–belang van die stelselveranderlikes genormaliseer word om in die gebied *[-1,1]* te val. Hierdie genormaliseerde gebied kan natuurlik ook verskuif word om in die gebied *[0,2]* te val. Verder kan die gebied *[0,2]* weer genormaliseer word om in die gebied *[0,1]* te val. Gevolglik hoef slegs positiewe getalle geakkommodeer te word vir die wiskundige

boublokke. Die waardes in die gebied *[0;0,5]* is dus 'n verteenwoordiging van die negatiewe waardes in die gebied–van–belang, en die gebied *[0,5;1]* verteenwoordig die positiewe waardes. Die lidmaatskapswaardes en die hele wasige selflerende proses kan uitgevoer word met getalle in die gebied *[0,1]*, aangesien die mate van deelname van 'n veranderlike slegs tussen 0% en 100% kan val. Die bogemelde aanname het die ontwerp van die binêre opteller en aftrekker aansienlik vereenvoudig.

Daar is verskeie opteller ontwerpe beskikbaar [71] [72] [67] [80], maar die mees uitbreibare en eenvoudige ontwerp is waarskynlik ook die mees bekende ontwerp. Die eenvoudige programmeerbare opteller– en aftrekkerstroombaan, wat op vol–optellers gebaseer is, word in figuur 4.4.2 getoon. Hierdie 4–bis ontwerp is eenvoudig om uit te brei na 'n 16–bis weergawe en lewer in een kloksiklus 'n uitset.



Figuur 4.4.2: Opteller–aftrekkerontwerp.

Hierdie stroombaan kon soveel vereenvoudig word dat dit slegs 16 programmeerbare blokke van die Xilinx–vlokkies opgeneem het. Die struktuur is dus dat n blokke gebruik sal word, waar n die aantal bisse van die ontwerp is.

4.4.2.3 Die serie–parallelle vermenigvuldiger:

Sedert die sestigerjare was daar al 'n magdom voorstelle vir digitale vermenigvuldigers [68] [78] [73] [74] [67] [80]. Vinnige vermenigvuldiging was inderdaad die grootste kopseer van die RISC–tipe prosesseerder se ontwerpers. Na die baanbrekerswerk van [68], in 1963, was daar 'n sterk beweging na groot parallelle matriksvermenigvuldigers. Parallelle tegnieke is egter nie geskik vir die Xilinx–vlokkies nie, en daarom was die seriale en serie–parallelle skuif–en–optel algoritmes [67] bestudeer. Die mees eenvoudige implementasie van die skuif–en–optel tipe van vermenigvuldigers word in figuur 4.4.3 getoon. Dit het gou duidelik geblyk dat hierdie seriale benadering tot vermenigvuldiging nie vinnig genoeg sal wees vir implementasie op die Xilinx–vlokkies nie.

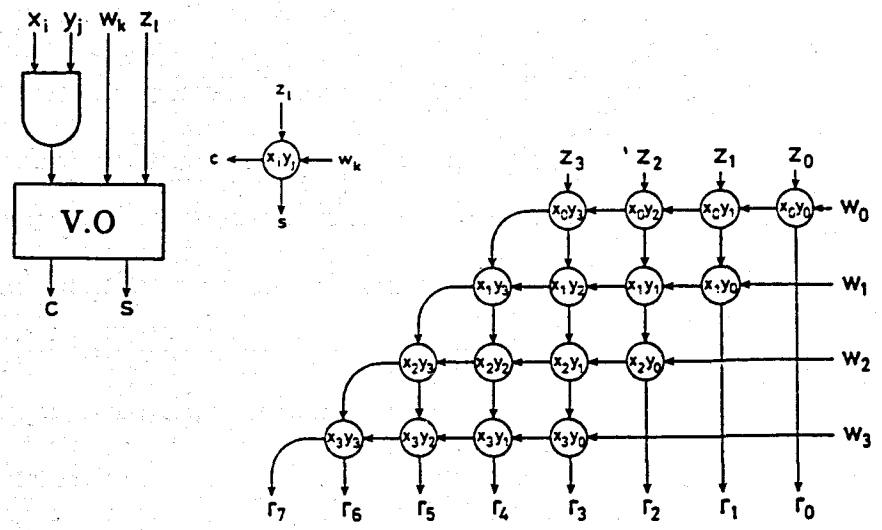
[73] het in 1978, en later het [67], 'n tegniek voorgestel wat 'n uitbreiding is op die parallelle oordragstegnieke [68]. In hierdie tegniek word die parallelle gegenereerde oordragswaardes nie direk oorgedra na die volgende vlak van vol–optellers van die vermenigvuldigingsmatriks nie, soos in figuur 4.4.4, getoon; maar word gestoor om een kloksiklus later deur dieselfde vlak van vol–opteller gebruik te word. Hierdie voorstel gebruik dus steeds die voordele van parallelle realisasie tegnieke, maar elimineer $(n-1)$ van die parallelle–matriksvlakke.

Die bogemelde tegniek van vermenigvuldiging lewer in n siklusse die produk van twee getalle, en gebruik slegs n Xilinx programmeerbare blokke, waar n ooreenstem met die aantal bisse van die ontwerp.

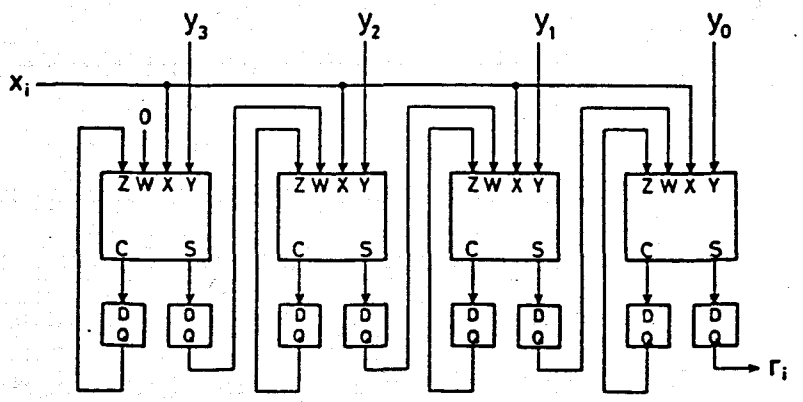
4.4.2.4 Die sekwensiële nie–herstellende deler:

Na die ekliptiese ontwikkeling van binêre vermenigvuldiging, het menige navorsers

Figuur 4.4.3. Die mees eenvoudige skuif–en–optel tipe vermenigvuldiger.



Figuur 4.4.4. Serie–paralelle vermenigvuldiger.



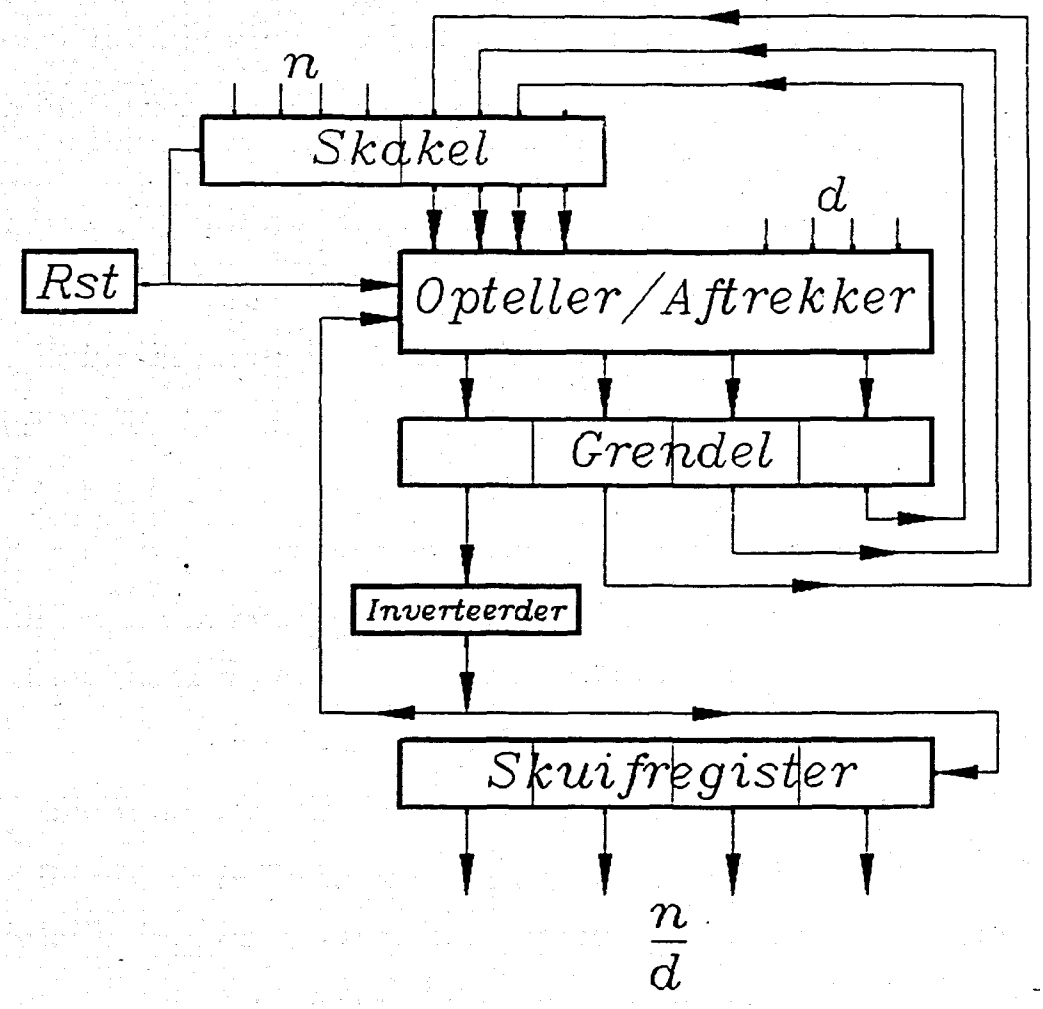
hulle aandag begin wend na die volgende leemte in die "wiskundige ketting", naamlik binêre deling. Alhoewel die probleem van deling alreeds in 1968 deur [75] aangespreek is, is daar nou relatief onlangs eers weer 'n opvlamming in die belangstelling in binêre deling [78] [76] [77] [67].

Die literatuur [67] [75] verwys na twee hoofgroepe van delers, naamlik: herstellende en nie–herstellende delers. Herstellende delers gebruik slegs herhaalde aftrekking om die kwosiënt te bepaal. Die dividend word na elke aftrekoperasie herstel sodat die deler elke keer daarvan afgetrek kan word. Die nie–herhalende deler gebruik egter beide optel– en aftrekoperatore. Die teken van die vorige resultaat, van optel of aftrek, bepaal die huidige operasie. Die nie–herstellende deler is dus vinniger, aangesien geen tyd gemors word met die herstellingsaksie nie, en is relatief eenvoudig om met 'n programmeerbare opteller/aftrekker, soos in afdeling 4.4.2.1 verduidelik, te realiseer. Die nie–herstellende sekwensiële deler–realisasie word in figuur 4.4.6 getoon, en lewer in *(n+1)* siklusse 'n kwosiënt. Daar is egter een vereiste dat die kwosiënt, k/d, slegs bestaan, waar $|k| \leq |d|$. Hierdie vereiste kan wel geakkommodeer word in die genormaliseerde getalstelsel van die wasige proses, soos in hoofstuk 3 verduidelik.

4.4.3 Apparatuur–realisasie van wasige selflerende algoritme:

Die SK–algoritme, soos beskryf in hoofstuk 2.5.3.2, bestaan uit vier afsonderlike sub–afdelings, naamlik: i) die verwasiger; ii) die relasie–identifikasie–berekeninge; iii) die rekursiewe wasige beramer; en iv) die ontwasiger. Al die wiskundige boublokke wat in die vorige afdeling beskryf is, moes gebruik word om die wasige sub–elemente te ontwerp in apparatuur.

Figuur 4.4.6: Binêre serie–paralelle deler .



4.4.3.1 Die tabel–verwasiger:

In hoofstuk 3.2 was die tabel–verwasigingsmetodiek voorgestel, en die voordele daarvan verduidelik. Die tabel–verwasigingsmetodiek kom neer op 'n enkele vergelyking, wat die geheue–posisie, a, in die tabel, van vooraf verwasigde waardes, spesifiseer. Vergelyking 3.2.3 baseer die geheue posisie op die nie–wasige getal self, en word hier vir gerief weer herhaal:

$$a = \frac{(Getal+1)(N-1)}{2}$$

Die waarde van $Getal \in [-1,1]$, en $N = 256$, die aantal elemente in die geheue opsoek–tabel.

Indien die gebied–van–belang van Getal nou verskuif, en gehernormaliseer word om in gebied $[0,1]$ te val, soos in hoofstuk 4.4.2.2 bespreek, sal vergelyking 3.2.3 ook vereenvoudig na:

$$a = int\ \tfrac{1}{2}(Getal \times (N-1)) \qquad \text{.... 4.4.1}$$

Vergelyking 4.4.1 kan verder vereenvoudig word indien die waarde van $N>>1$.
Vir hierdie projek word $N = 256$ gekies, dus kan vergelyking 4.4.1 as volg benader word:

$$a = int\ (\tfrac{1}{2}(Getal \times N)) \qquad \text{.... 4.4.2}$$

Die laaste stap van vereenvoudiging is om te onthou dat $256/2 = 2^7$, en dat heelgetal–waardes *(int)* slegs die "afkap" van die breukbisse van die binêre getal sal wees. Dit reduseer vergelyking 4.4.2 na die manipulasie wat in apparatuur gerealiseer is, naamlik:

$$a = Kap\ af\ (Skuif\ Getal\ 7\ na\ links) \qquad \text{.... 4.4.3}$$

Vergelyking 4.4.3 is eenvoudig genoeg om in 'n enkel kloksiklus uit te voer.

Die waarde n wys egter net na die begin–adres van die ρ–tal wasige ekwiwalent van *Getal*. 'n Twee–dimensionele matriks moet dan nog geadresseer word volgens vergelyking 4.4.4, wat volg uit vorige navorsing [43].

Adres van $z(k,l)$ word gegee as:

$$Adres = (k-1)\times\rho + l \qquad \text{.... 4.4.4}$$

met: k=*enige indeks;* $l = 0...(\rho-1)$,

en ρ *die aantal* referensiegebied*e.*

Hierdie apparatuurontwerp kan in aanhangsel 4.4.1.a gesien word en die lys van inter–blokverbindings kan in aanhangsel 4.4.1.b gesien word.

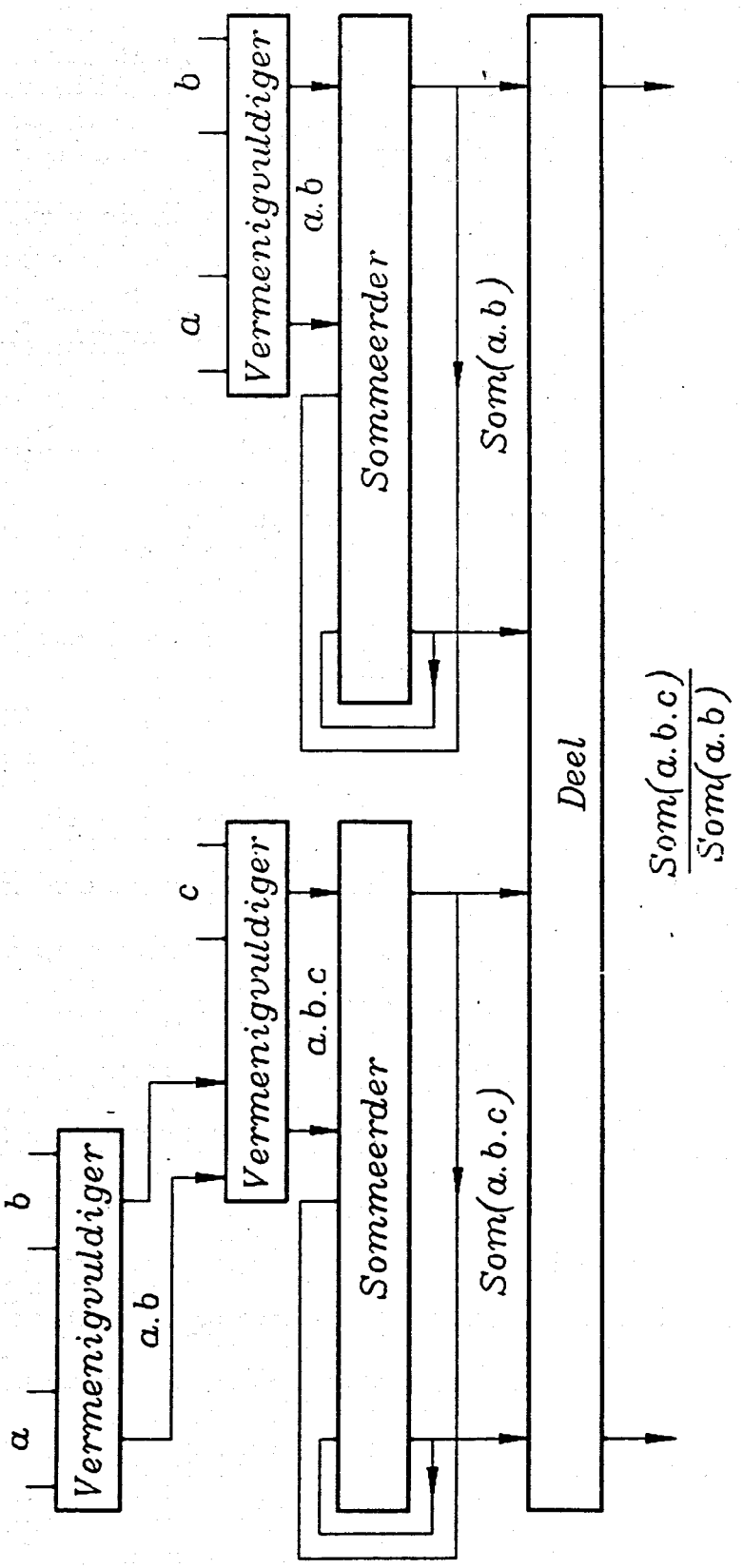
4.4.3.2 Die relasie–identifikasie–berekening:

Die relasie–inskrywings word gegee deur vergelyking 2.4.3, soos reeds voorheen verduidelik. Hierdie vergelyking kan gereduseer word tot: i) die herhaalde sommering van die produk van drie getalle, $\Sigma(a\times b\times c)$; ii) die herhaalde sommering van die produk van twee van eersgenoemde getalle, $\Sigma(a\times b)$; iii) ten laaste word die resultaat van i) gedeel deur die resultaat van ii), $\frac{\Sigma(a\times b\times c)}{\Sigma(a\times b)}$, om 'n relasie inskrywing te vorm.

Die binêre sommeerder, $\sum_{k=1}^{K}$, kan verwesenlik word met 'n eenvoudige opteller. Die uitset van die opteller word teruggevoer om in die opvolgende siklus by die nuwe inset getel te word. Sien die blokdiagram, figuur 4.4.7. Hierdie operasie word K keer herhaal, en daar moet dus 'n $1...K$ teller beskikbaar wees om die proses te koördineer.

Die vermenigvuldiging van die drie getalle kan saamgestel word uit serie–parallelle vermenigvuldigers. Sien die blokdiagram in figuur 4.4.7. Die seriale uitset van die eerste vermenigvuldiging word gebruik as die seriale inset tot die tweede vermenigvuldiging. Met hierdie opstelling word slegs een kloksiklus ekstra gebruik vir die oordrag na die tweede vermenigvuldiging. 'n Multi–inset vermenigvuldigerstroombaan is dus 'n relatiewe

Figuur 4.4.7: Relasie-berekening.



eenvoudige uitbreiding van die serie–parallelle vermenigvuldiging, en elke addisionele inset bring slegs 'n enkele kloksiklus vertraging van die finale uitset teweeg. Die algehele relasie berekening blokdiagram kan in figuur 4.4.7 gesien word.

Die totale relasie–berekeningstroombaan, met uitvoertydoptimisasie, kan in aanhangsel 4.4.2.a gesien word en die lys van meegaande inter–blokverbindings kan in aanhangsel 4.4.2.b gesien word.

Die ontwerp kan as volg opgesom word:

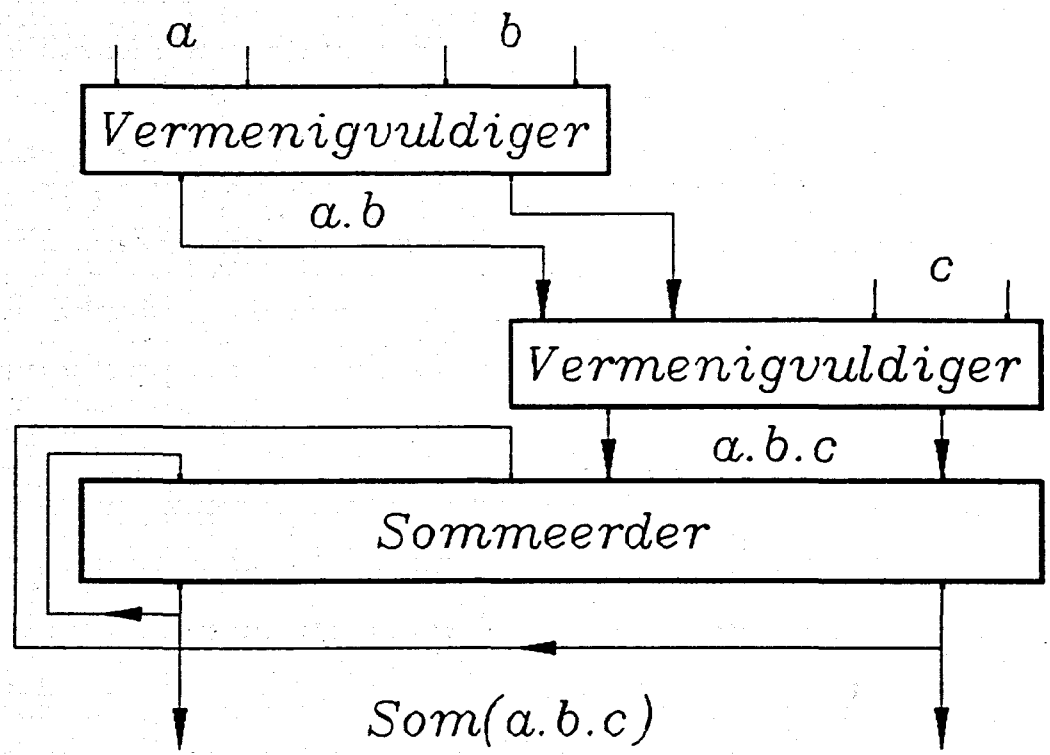
1) **Totale aantal programmeerbare blokke benodig:** 171
2) **Totale uitvoertyd:** 21 kloksiklusse per iterasie van algehele relasie berekening, plus 16 siklusse aan einde van elke relasie siklus vir delingbewerking.
3) **Aantal inter–blokverbindings:** 243.

4.4.3.3 Die rekursiewe wasige beramer:

Die wasige SK–beramer word in hoofstuk 2.5.3.2 bespreek, maar kan gereduseer word tot die herhaalde sommering van die produk van drie waardes, $\Sigma(a \times b \times c)$. Die leser sal opmerk dat hierdie basiese berekening deel vorm van die relasie–berekeninge van afdeling 4.4.3.2. Die beraming sub–afdeling van die wasige proses is kronologies voor die relasie sub–afdeling ontwerp, en die relasie–berekeningstroombaan is 'n uitbreiding van die wasige beramer.

'n Blokdiagram van die wasige beramerontwerp kan in figuur 4.4.8 gesien word, en as volledige stroombaandiagram, in aanhangsel 4.4.3.a en die meegaande inter–blokverbindings kan in aanhangsel 4.4.3.b gesien word.

Figuur 4.4.8: Die wasige beramer.



Die ontwerp kan as volg saamgevat word:

1) **Totale aantal programmeerbare blokke benodig:** 122

2) **Totale uitvoertyd:** 19 kloksiklusse per beramingsiterasie, sluit enkel sommasie siklus ook in.

3) **Aantal inter–blokverbindings:** 133.

4.4.3.4 Sentroïde ontwasiging:

Sentroïde ontwasiging [1] is alreeds in afdeling 2.6.1 bespreek. Vergelyking 2.6.1 bepaal die ontwasigde waarde van 'n wasige ρ–tal. Dit blyk duidelik uit vergelyking 2.6.1, dat die geweegde gemiddeld–berekening van die wasige relasie baie sterk ooreenstem met die geweegde gemiddeld–berekeninge, van die sentroïde ontwasigingsmetodiek.

Die ontwasigingsberekeninge reduseer na: i) die herhaalde som van die produk van twee veranderlikes, $\Sigma a{\times}b$; ii) die herhaalde sommering van die waardes van een van die veranderlikes, Σa; en iii) die kwosiënt van die resultaat van i) gedeel deur die resultaat van ii), $\frac{\Sigma a{\times}b}{\Sigma a}$.

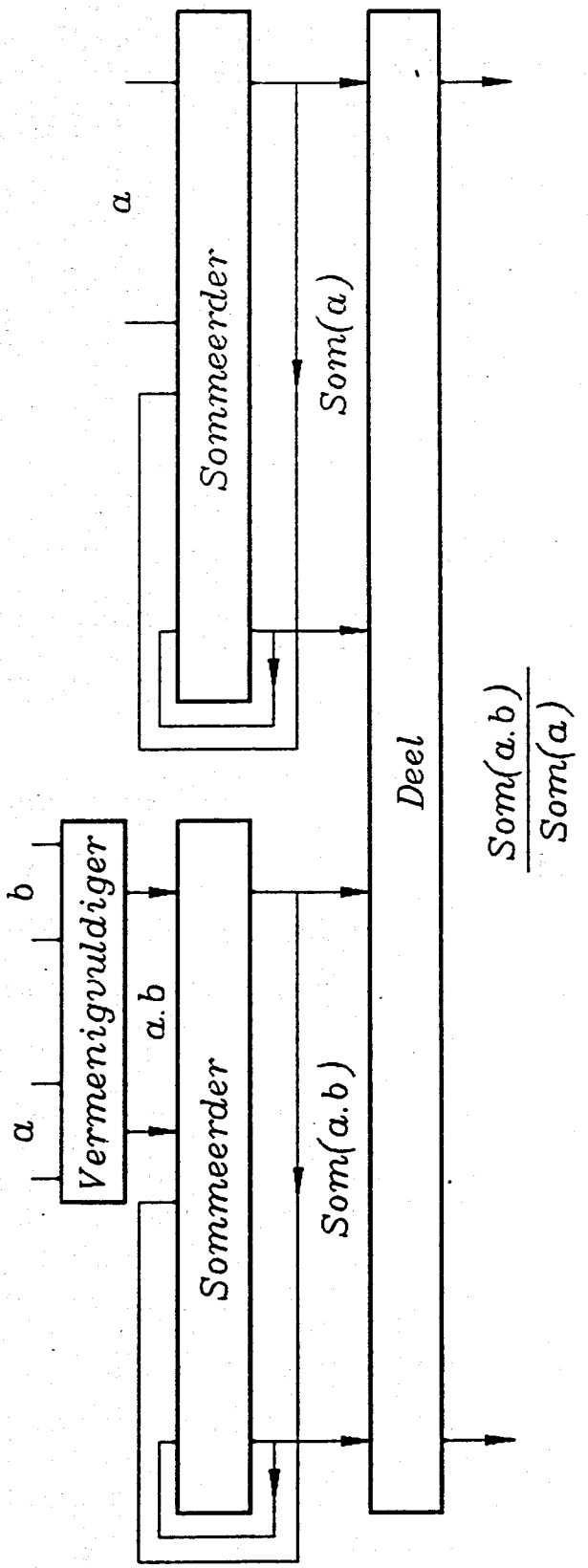
Die blokdiagram van die sentroïde ontwasiger–ontwerp word in figuur 4.4.9. getoon, en die volledige stroombaan uitleg word in aanhangsel 4.4.4.a getoon en die meegaande lys van inter–blokverbindings in aanhangsel 4.4.4.b.

Die ontwerp kan as volg opgesom word:

1) **Totale aantal programmeerbare blokke gebruik:** 156

2) **Totale uitvoertyd:** 20 kloksiklusse

Figuur 4.4.9: Die ontwasiger.



3) **Totale aantal inter–blokverbindings:** 233.

4.4.3.5 Geheue–adresberekeninge:

Bo en behalwe die bogenoemde wasige sub–elementberekeninge, is nog 'n tydsame element van die totale apparatuurproses, die bepaling van die geheue–adresse van die wasige stelsel–veranderlikes.

Die volgende data–matrikse moes direk geadresseer kon word:

i) Een–dimensionele matriks:

Die adressering van 'n data–matriks van een dimensie kon eenvoudig met 'n teller vermag word. Dit moet wel programmeerbaar wees, in die sin dat die teller moet begin tel by die begin–adres van die data–matriks.

ii) Twee–dimensionele matriks:

Die vergelyking in die bepaling van 'n spesifieke sel–adres van 'n geheue element in 'n twee–dimensionele matriks was al in vorige navorsing [43] afgelei en word gegee deur:

Vir $z(k,l)$ is die *adres* $= (k - 1) \times \rho + l$ **4.4.5**

waar: $l = 0.....(\rho - 1)$; ρ *die aantal referensiegebiede in die gebied–van–belang is; k enige dimensie kan hê.*

Die opsoek van 'n twee–dimensionele geheue–matriks was die mees algemene gebruikte geheue operasie. Die Xilinx–ontwerp word in aanhangsel 4.4.5.a getoon en die meegaande lys van inter–blokverbindings kan in aanhangsel 4.4.5.b gesien word.

iii) Drie–dimensionele matriks:

'n Drie–dimensionele matriks word gelewer deur die berekening van die relasie vir 'n enkel–inset en enkel–uitset stelsel.

Die bepaling van 'n geheue–element in 'n drie–dimensionele matriks is 'n uitbreiding van die algemene vergelyking van 'n twee–dimensionele matriks, en word gegee deur:

Vir $u(k,l,m)$ is $adres = (k-1)\times\rho^2 + (l-1)\times\rho + m$ **4.4.6**

waar: *k van enige dimensie kan wees;*

l en m = 0....(ρ −1), ρ die aantal referensiegebiede is.

4.5. Opsomming.

Vorige navorsing in die gebied van wasige apparatuur, tesame met die Xilinx ontwerpstrategie het daartoe gelei dat 'n 16–bis interne getalstruktuur en serie–parralelle stroombaantegnieke gevolg word vir die apparatuur ontwerp. Die beperkings aangaande die inter–blokverbindings het 'n klokfrekwensie van helfte die maksimum frekwensie tot gevolg gehad.

Die wasige apparatuur is beperk tot 'n enkel–inset en enkel–uitset stelsel wat 'n maksimum van 3 000 data–punte kan akkommodeer. 'n Rekenaarkaart, wat 8–bis data van 'n rekenaar ontvang en dit omskakel na 16–bis data, met 32k–greep SRAM geheue, is ontwerp. Bo en behalwe die geheue, bevat die rekenaarkaart ook nog twee 9 000 logikahek Xilinx XC3090, 70 MHz, vlokkies.

Vier basiese wiskundige boublokke was gebruik om die wasige onder–afdelings, naamlik: verwasiging, relasie–berekening, rekursiewe beraming, en ontwasiging, te realiseer. Laastens is die aantal kloksiklusse wat elke wasige sub–afdeling neem, om 'n enkele iterasie te voltooi, bepaal vir uitvoertydsbepaling in die opvolgende hoofstuk.

ooOOoo

HOOFSTUK 5: EKSPERIMENTELE RESULTATE.

"Where observation is concerned, chance favours only the prepared mind."

Louis Pasteur(1854).

HOOFSTUK 5: EKSPERIMENTELE RESULTATE.

5.1 Beskrywing van eksperimentele stelsel en optrede:

Die finale wasige selflerende apparatuur word bedryf as 'n bondel–bewerker. Met ander woorde, die insette tot elke sub–proses word beskikbaar gestel vanaf 'n rekenaar.

Die verwasiging was gedoen met die tabel–verwasigingstegniek. Daar is $2^8 = 256$ gekwantifiseerde 16–bis waardes, in die gebied–van–belang van elke stelselveranderlike, vooraf verwasig. Afsonderlike gebiede–van–belang vir elk van die drie wasige stelselveranderlikes was gebruik, soos in hoofstuk 3.4.3 verduidelik. Daar is van $\rho = 5$ aantal vaste gelykbenige, driehoekige, eweredig–verspreide referensiegebiede gebruik gemaak.

'n Rekursiewe skatter, wat gebaseer is op die SK–algoritme [23], is gebruik. Om die proses meer stroombelyn te maak, was nul–eliminasie gebruik om onnodige berekeninge uit te skakel.

Die eerste 150 punte van die Box en Jenkins [44] se industriële gasoond–data is gebruik in die leerfase, om die wasige relasie te bepaal. Tydens die beramingsfase word die res van die 296 punte van hierdie datastel beraam, om vergelyk te kan word met die gemonsterde data. Die beraamde wasige waardes word dan ontwasig, met behulp van sentroide ontwasiging.

5.2. Apparatuuruitvoertydsbepaling.

Die uitvoertyd van die wasige apparatuur was bepaal op basis van die aantal

kloksiklusse wat elke wasige sub–element neem om deur een iterasie te gaan. Die wasige sub–elemente was i)verwasiging, ii)relasieberekening, iii)beraming en iv) ontwasiging. Verder was van binêre tellers gebruik gemaak om die totale aantal iterasies van elke sub–element te bepaal. Die kloksiklusse van elke sub–element was dan almal bymekaar getel om die totale aantal kloksiklusse te bepaal wat benodig word om die Box en Jenkins datastel te prosesseer. Dit is dan eenvoudig om die uitvoertyd te bepaal vanuit die klokfrekwensie van die stelsel. Hierdie beginsel maak dit ook moontlik om akkurate vooruitskattings te maak aangaande die gebruik van vinniger klokfrekwensies. Die volgende resultate was verkry vir die uitvoer van elke sub–element:

i) Verwasiger: 36704 siklusse.
Uitvoertyd teen 35 MHz: **1.05 ms.**
Beraamde uitvoertyd teen 150 MHz: 0.245 ms.

ii) Relasieberekening: 663900 siklusse.
Uitvoertyd teen 35 MHz: **19 ms.**
Beraamde uitvoertyd teen 150 MHz: 4,4 ms.

iii) Rekursiewe beramer: 671328 siklusse.
Uitvoertyd teen 35 MHz: **19,2 ms.**
Beraamde uitvoertyd teen 150 MHz: 4,5 ms.

iv) Ontwasiging: 62456 siklusse.
Uitvoertyd teen 35 MHz: **1.784 ms.**
Beraamde uitvoertyd teen 150 MHz: 0.416 ms.

Die uitvoertyd en die beramingsfout, J, van die wasige apparatuur, teen 35 MHz, word in hierdie afdeling vergelyk met die oorspronklike programmatuur, hoë spoed mikro–verwerker implementasies, en ander kommersiële beskikbare apparatuur. Die uitvoertyd en fout van verskillende gevalle word in tabel 5.1 uiteengesit.

5.2.1 Uitvoertyd–analise:

LW. Tensy spesifiek anders vermeld, was alle programmatuursimulasies met 'n IBM–versoenbare 386–SX–20, met wiskundige mede–verwerker, uitgevoer.

Geval A1: **Figuur 5.2.1.** Hier word die uitvoertyd en fout, *J*, van die oorspronklike algoritme getoon, sonder uitvoertyd–optimisasie, of tabel–verwasiging.

Geval A2: **Figuur 5.2.2.** Die uitvoertyd en fout, *J*, van die rekursiewe SK–algoritme, met uitvoertyd–optimisasie, en tabel–verwasiging, word getoon.

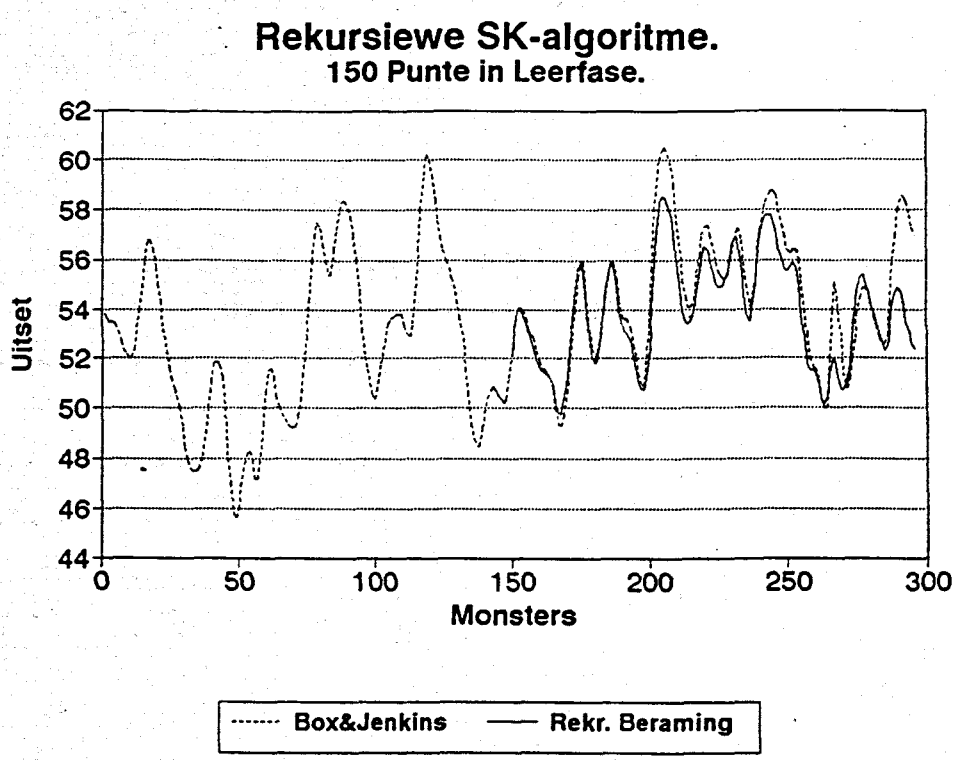
Geval A3: Die uitvoertyd en fout van geval A2, uitgevoer op 'n 486 DX–55, word getoon.

Geval A4: **Figuur 5.2.3.** Toon die resultate van die wasige apparatuur–realisasie.

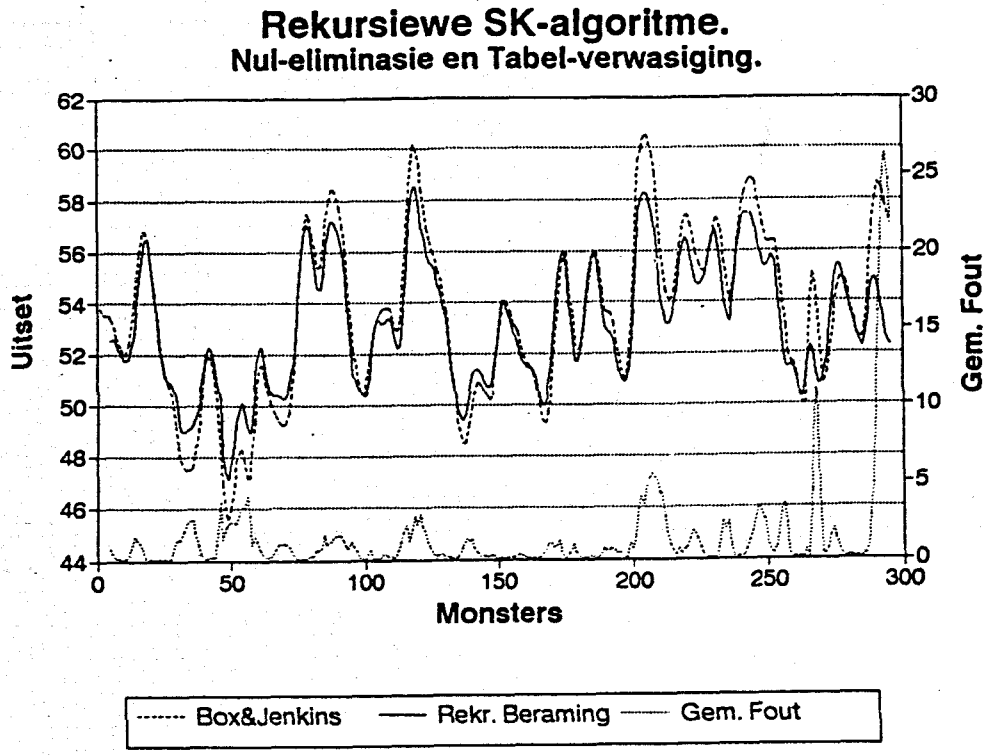
Tabel 5.1: Vergelyking van apparatuur–resultate:

Geval	Uitvoertyd(s)	Fout (J)
A1	9,66	0,791
A2	1,96	1,465
A3	0,21	1,465
A4	0,041	2,171

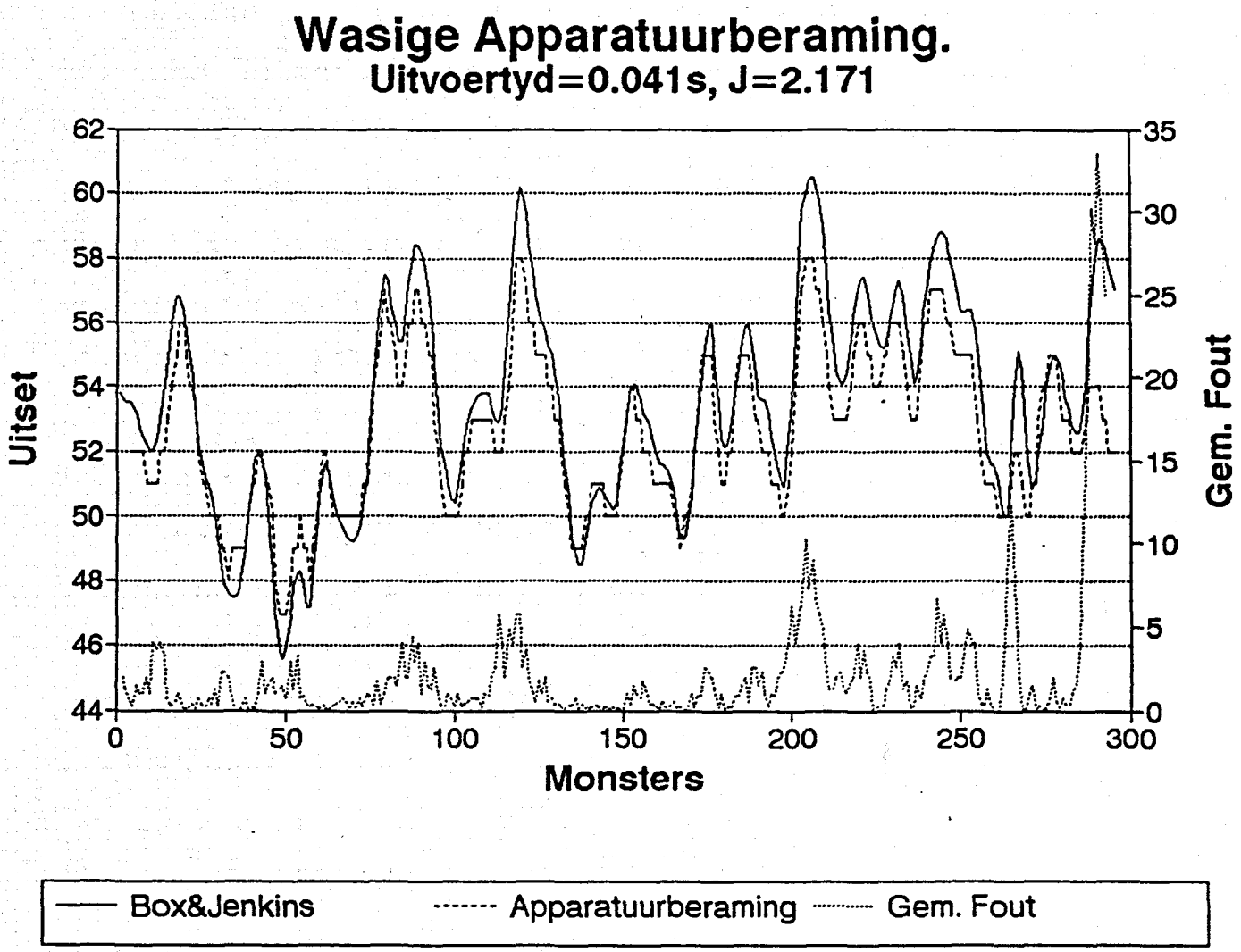
Figuur 5.2.1.



Figuur 5.2.2.



Figuur 5.2.3.


Wasige Apparatuurberaming.
Uitvoertyd=0.041s, J=2.171
Uitset
Gem. Fout
Monsters
62
60
58
56
54
52
50
48
46
44
35
30
25
20
15
10
5
0
0
50
100
150
200
250
300
Box&Jenkins
Apparatuurberaming
Gem. Fout

5.2.2 Opmerking:

1) Die leser sal merk dat daar 'n geleidelike verswakking in die beramingsakkuraatheid is. Die verswakking in akkuraatheid van gevalle A2 en A3 was alreeds in hoofstuk 3.4 bespreek. Die verdere verswakking in beramingsakkuraatheid vir die apparaat–realisasie word veroorsaak deur die vaste–punt 16–bis getalstruktuur van die apparatuur. Die afplattingsverskynsel, as gevolg hiervan, kan duidelik in figuur 5.2.3 gesien word, en was al in vorige navorsing [43] ook opgemerk. Die leser moet in gedagte hou dat die programmatuur–simulasies uitgevoer is met behulp van wiskundige mede–verwerkers, wat 10–greep of 80–bis, akkuraatheid handhaaf. Die sommering van die fout, as gevolg van 16–bis akkuraatheid, gee aanleiding tot hierdie verskynsel.

2) Tydens vorige navorsing [43], waar die oorspronklike SK–algoritme op 'n TMS 320 C10–25 geïmplimenteer was, was 'n uitvoertyd van 0.7 sekonde behaal, vir die leer van 100 punte, en die vooruitberaming van 62 punte, met 11–bis akkuraatheid. Die beramingsfout, J, oor hierdie 62 punte, was alreeds 0.9 gewees. Alhoewel hierdie resultate nie direk met die wasige apparatuur vergelyk kan word nie, wil dit wel voorkom dat die apparatuur aansienlik vinniger as selfs digitale seinverwerking–prosesseerders, soos die TMS 320 C10, sal wees.

5.2.3 Vergelyking met kommersiële beskikbare apparatuur:

Drie kommersiële beskikbare wasige prosesseerders was gekies om met die outeur se apparatuur te vergelyk:

a) Die FC 110 digitale wasige prosesseerder van Togai Infra Logic Inc. [63].

b) Die NLX 230 wasige mikro–beheerder van Neuralogix [81].

c) Die FP–3 000 wasige prosesseerder van Omron [82].

Op hierdie stadium het nog geen van die genoemde prosesseerders 'n selflerende apparatuur–opsie beskikbaar nie.

Die vergelyking word gedoen op die basis van die volgende punte:

1) Die aantal wasige reëls wat per sekonde geëvalueer kan word. (Sonder ontwasiging).
2) Die aantal nie–wasige evaluasies per sekonde. (Insluitend ontwasiging).
3) Die aantal reëls wat ondersteun word.
4) Die aantal insette en uitsette beskikbaar.
5) Die interne bis–resolusie.

Die resultate van hierdie vergelyking word in tabel 5.2 getoon.

Tabel 5.2: Vergelyking met kommersiële wasige prosesseerders.

Stelsel	(1)	(2)	(3)	(4)	(5)	(6)
FC110	200000	60000	800	n/b	n/b	8
NLX230	n/b	30 M	64	8	8	8
FP–3000	n/b	30769	128×3 groepe	8	4	12
Outeur (berekend)	99715	25669	125	1	1	16

Alhoewel tabel 5.2 toon dat die wasige apparatuur stadiger is as die ander kommersiële wasige prosesseerders, moet in gedagte gehou word dat hierdie prosesseerders beperk word tot eenvoudige wiskundige algoritmes, soos die maks–min algoritme [3] [1], en verder ook nie selflerende apparatuur is nie. Die kommersiële beskikbare stelsels kon ook van parallelle wiskundige strukture gebruik gemaak het, en was nie beperk tot serie–parallelle strukture nie.

Die leser word ook verwys na[91] waar die outers tot 250 miljoen reëls per sekonde kon evalueer, vir 'n vyf–inset en een–uitset stelsel, met 8–bis akkuraatheid en 10000 reëls wat ondersteun word. Hierdie outeurs het van oorspronklike inferensiemeganismes gebruik gemaak, wat nog steeds nie selflerend is nie, maar wat hierdie resultate meer indrukwekkend maak, is die feit dat dit met 'n 80486DX2–50 verkry en nie in apparatuur baseer is nie.

5.3. Opsomming.

Die wasige apparatuur was gebruik as 'n bondelprosesseerder. 'n Interne resolusie van 16–bis was gebruik om akkuraatheid te behou. Die afsonderlike genormaliseerde gebiede–van–belang was in 256 aantal kwantums verdeel vir tabel–verwasiging. Daar is van vyf eenvormige driehoekige referensiegebiede, in elke gebied–van–belang, gebruik gemaak. Tydens prosessering van Box en Jenkins se gasoonddata[44] was 'n uitvoertyd van 0.041 sekonde verkry. Dit was vir 'n leervenster van 150 waardes, en die beraming van 296 waardes. Hierdie resultaat is 'n 235–voudige verbetering in uitvoertyd in vergelyking met die oorspronklike SK–programmatuur [23]. Die wasige apparatuur vergelyk goed met kommersiële beskikbare wasige prosesseerders, indien in ag geneem word dat hierdie prosesseerders nie selflerend is nie.

ooOOoo

HOOFSTUK 6: GEVOLGTREKKING EN OPSOMMING.

"What we call the beginning is often the end
And to make an end is to make a beginning.
The end is where we start from."

T.S. Eliot,
Little Gidding.

HOOFSTUK 6: GEVOLGTREKKING EN OPSOMMING.

6.1. Programmatuursimulasies.

Die doel van hierdie projek was om 'n spesifieke wasige algoritme [23] in apparatuur te realiseer. Om dit te vermag was vorige navorsingswerk in hierdie gebied [39] [26] [23] [40] bestudeer, en tekortkominge identifiseer. Dit was nodig om die oorspronklike wasige algoritme te optimiseer, met die oog op apparatuur–realisasie. Die optimisasie was vermag deur twee groepe van tegnieke, wat "nul–eliminasie" en "tabel–verwasiging" genoem was deur die outeur. Beide tegnieke het hulself as uiters geskik vir apparatuur–implementasie bewys (sien hoofstukke 3.2 en 3.3). Na uitgebreide programmatuur simulasies, was die keuse gemaak om 'n tabel–verwasiger met 256 elemente te gebruik (sien hoofstukke 3.2 en 3.4.3); die gebied–van–belang van elke toestandsveranderlike afsonderlik te normaliseer (sien hoofstuk 3.4.3.2.1); en elke gebied–van–belang te verdeel in vyf identiese, eweredige gespasieerde driehoekige referensiegebiede [23][39]. Die leervenster was gestel op die helfte van die totale gemonsterde prosesdata, en 'n rekursiewe beramer was gebruik (sien hoofstukke 2.5.3 en 3.5). Laastens was sentroïde–ontwasiging geïmplementeer.

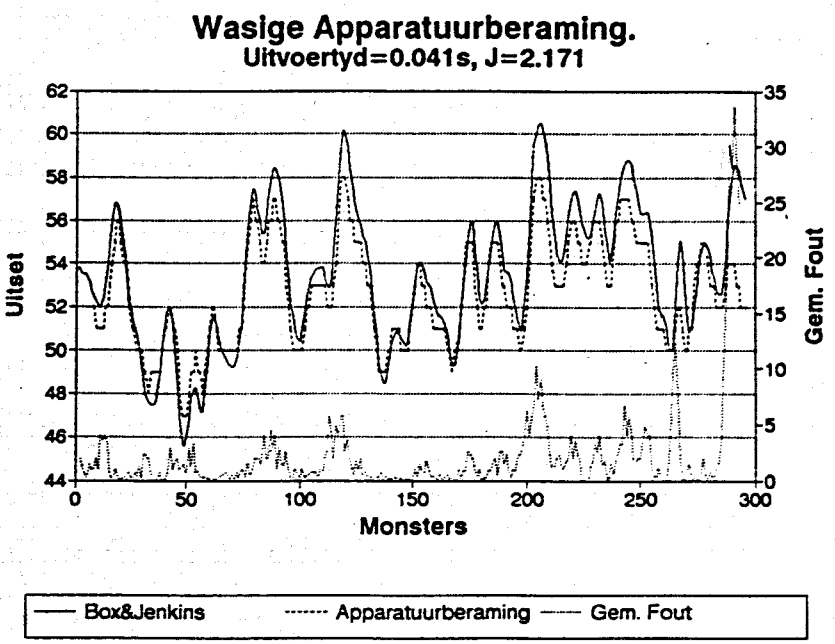
6.2. Die wasige apparatuur.

Die Xilinx programmeerbare–logikastelsel [66] was gekies as die apparatuurplatform vir die wasige apparatuur. Die Xilinx–vlokkies het egter groot beperkings op die ontwerp geplaas in die vorm van 'n beperkte aantal programmeerbare logikablokke, en beperkte aantal interne stroombaanverbindings (sien hoofstuk 4.3). Dit het die wasige apparatuur beperk tot 'n 35 MHz bondel–prosesseringstelsel wat op twee Xilinx–vlokkies ontwerp moes word.

Die wasige proses se vier sub–afdelings, naamlik die verwasiger, die relasie–identifiseerder, die rekursiewe skatter, en die ontwasiger, met meegaande geheue–adressering, was dus afsonderlik uitgevoer, en nie gekombineer nie, om die wasige proses op 'n enkele vlokkie daar te stel. Die uitvoertydsbepaling was egter nie deur hierdie stap geaffekteer nie, aangesien die wasige algoritme 'n bondel–prosesseringsproses is, en die vier sub–afdelings afsonderlik uitgevoer kan word.

6.3 Eksperimentele resultate.

Eksperimentele analise aangaande die uitvoertyd, en die beramingsakkuraatheid, het **'n uitvoertyd van 41 ms**, met 'n **gemiddelde–kwadraatfout van, J = 2.171**, opgelewer. Sien figuur 6.3.1. Dit transleer na 'n 235–voudige verbetering in uitvoertyd, in vergelyking met die **oorspronklike algoritme–programmatuur se 9.66 sekondes uitvoertyd** (verwys na hoofstuk 5). Hierdie resultate was verkry deur die wasige–apparatuur toe te pas op industriële gasoond–data [44].



Figuur 6.3.1: Apparatuur–resultate.

6.4. Tekortkominge en toekomstige navorsing.

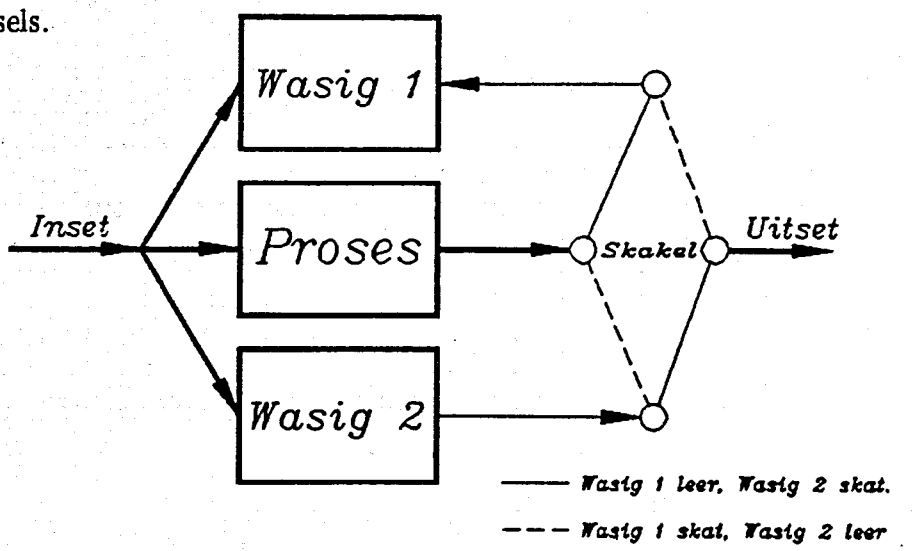
Die wasige apparatuur is egter nog steeds beperk tot 'n enkel–inset en enkel–uitset stelsel, as gevolg van die genoemde Xilinx apparatuurbeperkings, soos in afdeling 6.2 genoem. Daar was van serie–parallel ontwerpstegnieke gebruik gemaak (sien hoofstuk 4). Verder het die apparatuur–beperkings gelei tot 'n beperking op die maksimum klokfrekwensie waarteen die Xilinx–vlokkies bedryf kon word (sien hoofstuk 4.3.2). Die klokspoed was daarom beperk tot 35 MHz, terwyl daar vlokkies beskikbaar is wat tot teen 210 MHz geklok kan word [66]. Hierdie vlokkies was egter nie beskikbaar vir hierdie projek nie.

In toekomstige navorsing kan die wasige proses volledig in parallel opgebou word, indien die apparatuur se beperkinge oorkom kan word. Dit sal beteken dat die basiese wiskundige boublokke in parallel gerealiseer kan word, wat baie vinniger uitvoertye sal lewer. Ook kan die tabel–verwasigingsproses totaal in parallel uitgevoer word. Daar kan dus 'n opsoek–tabel bestaan vir elke referensiegebied van elke gebied–van–belang. Hierdie stappe behoort ten minste 'n verdere 16–voudige uitvoertydverbetering te lewer, en het die potensiaal om nog 'n heelwat groter verbetering te kan lewer. Verder word navorsing voorgestel in die gebied van multi–inset multi–uitset implementasie van die selflerende algoritme.

Een van die grootste voordele van die rekursiewe beramer [23], wat in hierdie projek gebruik was, is dat dit outonoom kan funksioneer. Aangesien die beraamde uitset terug gevoer word, beteken dit dat die proses wat gemodelleer word, verwyder kan word sodra die leerfase voltooi is, sien hoofstuk 2.5.3. Die wasige kennisbasis moet egter van tyd tot tyd opgedateer word. Om hierdie operasie intyds te vermag word 'n leer–en–beraam "tandem" stelsel voorgestel. Hierdie stelsel sou bestaan uit twee identiese wasige stelsels,

sodat die een stelsel kan leer, terwyl die ander stelsel die beramingsfunksie uitvoer. Die beramingsproses hoef dus nie gestaak te word nie, vir die opdatering van die wasige relasie. Die leerfase sal egter heelwat korter as die beramingsfase moet wees, anders sal die proses steeds voortdurend teenwoordig moet wees. Sien figuur 6.4.1 vir 'n moontlike grafiese uitbeelding van hierdie voorstel.

Laastens word voorgestel dat die wasige stabiliteitsanalise, wat deur [12] voorgestel is en in aanhangsel 1 bespreek word, ook deel moet vorm van toekomstige wasige apparatuurstelsels.



Figuur 6.4.1: Tandem, leer–en–beraam stelsel:
Een wasige stelsel leer, terwyl die ander beraam, en ruil dan om.

6.5. Vervulling van doelstellings.

Verwys na die doelstellings van hoofstuk 3.

Die doelstellings wat in hierdie projek bereik was, kan as volg saamgevat word:

1) Die wasige selflerende algoritme[23] was in apparatuur realiseer.

2) Die oorspronklike wasige programmatuur[41] was optimiseer, met verwysing na apparatuur–realisasie, deur van nul–eliminasie en tabel–verwasiging gebruik te maak.

3) Die wasige algoritme was egter nie as 'n enkel wasige vlokkie gerealiseer nie, maar moes vanweë apparatuurbeperkinge op twee programeerbare vlokkies ontwerp word. Die apparatuurbeperking het ook daar toe gelei dat die wasige vlokkies slegs teen 35 MHz opereer kon word. Verder kon al vier die wasige sub–elemente nie tesame opgebou word nie, maar slegs as bondelverwerkers. Hierdie stap het egter nie die bepaling van die uitvoertyd affekteer nie.

3.1) Laastens kan genoem word dat die wiskundige boublokke en die wasige sub–afdelings nie as optimale parallelle ontwerpe saamgestel kon word nie. Daar is wel van hoog doeltreffende serie–parallelle ontwerpstegnieke gebruik gemaak.

4) Die IBM–versoenbare rekenaarkaart was ontwerp en opgebou om die gebruik en ontfouting van die wasige vlokkies meer gebruikersvriendelik te maak.

ooOOoo

VERWYSINGS:

[1] KOSKO, B.(1992): *Neural networks and fuzzy systems.* Prentice–Hall International Editions.

[2] ZADEH, L.A.(1972): *A rationale for fuzzy control.* Trans. ASME. J. Dynam. Syst. Measur. Control., vol.94, pp 3–4.

[3] LEE, C.C.(1990): *Fuzzy logic in control systems: Fuzzy logic controller–Part 1.* IEEE Trans. Syst. Man Cybern.,Vol. 20, No. 2.

[4] SCHWARTZ, D.G.,KLIR, G.J.(1992): *Fuzzy logic flowers in Japan.*,IEEE Spectrum, pp. 32–35,July.

[5] MAMDANI, E.H.(1977): *Application of fuzzy logic to approximate reasoning using linguistic synthesis.*, IEEE Trans. Computers, Vol. C26, No. 12, pp. 1182–1191, December.

[6] MAMDANI, E.H., ASSILIAN, S.(1975): *An experiment in linguistic synthesis with a fuzzy logic controller.*,Int. J. Man Mach. Studies, Vol.7, No. 1, pp. 1–13.

[7] OSTERGARD, J.J.(1977): *Fuzzy logic control of a heat exchange process.*, van *Fuzzy automata and decision processes.*, M.M. Gupta, G.N. Saridis en B.R. Gaines, Eds. Amsterdam, North–Holland, pp. 285–320.

[8] ZADEH,L.A.(1973): *Outline of a new approach to the analysis of complex systems and decision processes.*, IEEE Trans. Syst. Man Cybern., Vol. SMC–3,pp. 28–44.

[9] BRAAE, M., RUTHERFORD, D.A.(1978): *Fuzzy relations in a control setting.*, Kybernetes, Vol.7, No. 3, pp. 185–188.

[10] WILLAEYS, D.(1980): *Optimal control of fuzzy systems.*, in Proc. Int. Congress on applied systems research and cybernetics.,Acapulco, December.

[11] KOMOLOV, S.V., MAKEEV, S.P., SHAKNOV, I.F.,(1979): *Optimal control of a finite automation with fuzzy constraints and a fuzzy target.*, Cybern., Vol. 16, No. 6, pp. 805–810.

[12] KUDINOV, Y.I.(1991): *Fuzzy control systems.*, Scripta Technica, Inc., ISSN0882–4002/91/0004–0103.

[13] TONG, R.M., BECK, M.B.,LATTEN,A.(1980): *Fuzzy control of the activated sludge wastewater treatment process.*, Automat.,vol. 16, no. 6, pp. 695–701.

[14] FUKAMI, S., MIZUMOTO, M., TANAKA, K.(1980): *Some considerations of fuzzy conditional inference.* Fuzzy Sets Syst., vol.4, pp. 243–273.

[15] HIROTA, K., PEDRYCZ, W.(1983): *Analysis and synthesis of fuzzy systems by the use of fuzzy sets.*, Fuzzy Sets Syst., vol. 10, no. 1, pp. 1–14.

[16] TAKGI, T., SUGENO, M.(1983): *Derivation of fuzzy control rules from human operator's control actions.*, van Proc. of the IFAC Symp. on fuzzy Information, Knowledge representation and decision analysis, Marseilles, France, July, pp.55–60.

[17] YASUNOBU, S., MIYAMOTO, S., IHARA, H.(1983): *Fuzzy control for automatic train opration system.*, van Proc 4th IFAC/IFIP/IFORS Int. Congress on control in transportation systems, Baden–Baden, April.

[18] SUGENO, M., MURAKAMI, K.(1984): *Fuzzy parking of a control model car.*, van 23 rd IEEE Conf. on decision and control, Las Vagas.

[19] KISZKA, I.B., GUPTA, M.M., NIKIFORUK, P.N.(1985): *Energetistic stability of fuzzy dynamic systems.*, IEEE Trans. Syst. Man Cybern., vol. SMC–15, no. 5, pp. 783–792.

[20] TOGAI, M., WATANABE, H.(1986): *Expert system on a chip: An engine for real–time approximate reasoning.*, IEEE Expert Syst. Mag., vol. 1, pp. 55–62.

[21] YAMAKAWA, T.(1986): *High–speed fuzzy controller hardware system.*, van Proc. 2nd fuzzy system symp., Japan, pp. 122–130.

[22] DUBOIS, D., PRADE, H.(1988): *An approach to computerized processing af uncertainty.*, New–York: Plenum Press.

[23] SHAW, I.S., KRüGER, J.J.(1992): *New fuzzy learning model with recursive estimation for dynamic systems.*, Fuzzy Sets Syst., vol.48, pp. 217–229.

[24] TONG, R.M.(1977): *A control engineering review of fuzzy systems.*, Automatica, Vol. 13, pp. 559–569.

[25] LEE, C.C.(1990): *Fuzzy logic in control systems: Fuzzy logic controller, Part 2.*, IEEE Trans. Syst. Man Cybrn., vol. 20, no. 2, March/April, pp. 419–435.

[26] XU, C., LU, Y.(1987): *Fuzzy model identification and self–learning for dynamic systems.*, IEEE. Trans. Syst. Man Cybrn., vol. SMC–17, no. 4, July/August, pp. 683–689.

[27] KICKERT, W.J.M., Van NAUTA LEMKE, H.R.(1976): *Application of a fuzzy controller in a warm water plant.*, Automat., vol. 12, no. 4, pp. 301–308.

[28] KING, P.J., MAMDANI, E.H.(1977): *The application of fuzzy control systems to industrial processes.*, Automat., vol. 13, no. 3, pp. 235–242.

[29] PAPPIS, C.P., MAMDANI, E.H.(1977): *A fuzzy logic controller for a traffic junction.*, IEEETrans. Syst. Man Cybern., vol. SMC–7, no. 10, pp. 707–717.

[30] UMBERS, I.G., KING, P.J.(1980): *An analysis of human–decision making in cement kiln control and the implication for automation.*, Int. J. Man Mach. Studies, vol. 12, no. 1, pp. 11–23.

[31] YAGISHITA, O., ITOH, O., SUGENO, M.(1985): *Application of fuzzy reasoning to the water purification process.*, van Industrial applications of fuzzy control., M. Sugeno, Ed. Amsterdam: North–Holland, pp.19–40.

[32] SUGENO, M., NISHIDA, M.(1985): *Fuzzy control of madel car.*,Fuzzy Sets Syst., vol. 16, pp. 103–113.

[33] SUGENO, M., MURAKAMI, K.(1985): *An experimental study on fuzzy parking control using a model car.*, van Industrial applications of fuzzy control, M. Sugeno, Ed. Amsterdam: North–Holland, pp. 125–138.

[34] BAAKLINI, N., MAMDANI, E.H.(1975): *Prescriptive methods for deriving control policy in a fuzzy–logic controller.*, Electron. Lett., vol 11, pp. 625–626.

[35] KING, P.J., MAMDANI, E.H.(1975): *The application of fuzzy control systems to industrial processes.*, in IFAC World congress, MIT, Boston.

[36] PROCYK, T.J., MAMDANI, E.H.(1979): *A linguistic self–organizing process controller.*, Automat., vol. 15, no. 1, pp. 15–30.

[37] SHAO, S.(1988): *Fuzzy self–organizing controller for dynamic processes.*, Fuzzy Sets Syst., vol. 26, pp. 151–164.

[38] TANSCHEIT, R., SCHARF, E.M.(1988): *Experiments with the use of a rule–based self–organising controller for robotic applications.*, Fuzzy Sets Syst., vol. 26, pp. 195–214.

[39] RIDLEY, J.N., SHAW, I.S., KRüGER, J.J.(1988): *Probabilistic fuzzy model for dynamic systems.*, Electrn. Letters, vol. 24, no. 14, pp. 890–892.

[40] POSTLETHWAITE, B.(1991): *Emperical comparison of methods of fuzzy relational identification.*, IEE Proceedings–D, vol. 138, no. 3, Mei.

[41] SHAW, I.S.(1990): *Expert fuzzy control based upon man–in–the–loop identification.*, Doktorale proefskrif in elektroniese ingenieurswese, R.A.U.

[42] STREMLER, F.G.(1982): *Introduction to communication systems.*, 2nd Ed., Addison–Wesley.

[43] SCHEFFER, M.F.(1991): *Die bestudering van 'n wasige beheermodel wat gebaseer is op 'n Texas Instruments mikro–verwerker vir toepassing op industriële gasoond–data.*, B.Ing. Elek. en Elekt. Skripsie, Rand Afrikaanse Universiteit, Desember.

[44] BOX, G.E.P., JENKINS, G.M.(1970): *Time series analysis, forecasting, and control.* Holden Day, San Francisco.

[45] POSTLETHWAITE, B.(1991): *Probabilistic fuzzy model in a noisy environment.*, Interne publikasie, Department of Chemical and Process engineering, University of Strathclyde, Glassgow, Skotland.

[46] MAMDANI, E.H.(1976): *Advances in the linguistic synthesis of fuzzy controllers.*, Int. J. Man Mach. Studies, vol. 8, no. 6, pp. 669–678.

[47] MAMDANI, E.H.(1977): *Application of fuzzy logic to approximate reasoning using linguistic synthesis.* IEEE Trans. Computer, vol. C–26, no. 12, pp. 1182–1191.

[48] KING, P.J., MAMDANI, E.H.(1977): *The application of fuzzy control systems to industrial processes.*, Automat., vol. 13, no. 3, pp. 235–242.

[49] YASUNOBU, S., HASEGAWA, T.(1986): *Evaluation of an automatic container crane operation system based on predictive fuzzy control.*, Control Theory Adv. Technol., vol. 2, no. 3, pp. 419–432.

[50] YASUNOBU, S., HASEGAWA, T.(1987): *Predictive fuzzy control and its application for automatic container crane operation system.*, uit Proc. 2nd IFSA Congress, Tokyo, Japan, July, pp. 349–352.

[51] YASUNOBU, S., SEKINO, S., HASEGAWA, T.(1987): *Automatic train operation and automatic crane operation systems based on predictive fuzzy control.*, van Proc. 2nd IFSA Congress, Tokyo, Japan, July, pp. 835–838.

[52] FUJITEC, F.(1988): *FLEX–8800 series elevator group control system.*, Fujitec Co., Ltd., Osaka, Japan.

[53] KASAI, Y., MORIMOTO, Y.(1988): *Electronically controlled continuously variable transmission.*, van Proc. Int. Congress on Transportation Electronics, Dearborn, MI.

[54] BERNARD, J.A.(1988): *Use of rule–based systems for process control.*,IEEE Contr. Syst. Mag., vol. 8, no. 5, pp. 3–13.

[55] MAEDA, Y.(1990): *Fuzzy obstacle avoidance method for a mobile robot based on the degree of danger.* Proc. of NAFIPS'90, pp. 169–172, Junie.

[56] TAKEUCHI, T.,NAGAI, Y., ENOMOTO, N.(1988): *Fuzzy control of a mobile robot for obstacle avoidance.*,Information Science, vol. 45, pp. 231–239.

[57] MURAKAMI, S., TAKEMOTO, F.,FUJIMURA, H., IDE, E.(1987): *Weld–line tracking control of arc welding robot using fuzzy logic controller.*, Proc. of 2nd Inter. Fuzzy Systems Association Congress, pp. 353–357, July.

[58] LARKIN, L.I.(1985): A fuzzy logic controller for aircraft flight control., van Industrial Applications of Fuzzy Control, M. Sugeno, Ed., pp.87–103.

[59] TOGAI, M., WATANABE, H.(1986): *Expert system on a chip: An engine for real–time approximate reasoning.*,IEEE Expert, vol. 1, no. 3. pp. 55–62.

[60] YAMAKAWA, T.(1988): *High–speed fuzzy controller hardware system: The mega–FIPS machine.*, Information Sciences, vol. 45, pp. 113–128.

[61] WATANABE, H., DETTLOFF, W.D., YOUNT, K.E.(1990): *A VLSI fuzzy logic controller with reconfigurable, cascadable architecture.*, IEEE Journ. Solid–state Circt., vol. 25, no. 2, pp. 376–382.

[62] WATANABE, H.(1991): *Some consideration on design of fuzzy information processors – form a computer architectural point of view.*, van Proc. IFES'91 Fuzzy engineering toward human friendly systems, Part IV, pp.387–398.

[63] KATSUMATA, A., TOKUNAGA, H., YASUNOBU, S.(1991): *Fuzzy set processor (FSP) for fuzzy information processing.*, uit Proc. IFES'91 Fuzzy engineering toward human friendly systems, Part IV, pp.399–406.

[64] BONISSONE, P.P.(1991): *A compiler for fuzzy logic controllers.*,uit Proc. IFES'91 Fuzzy engineering toward human friendly systems, Part IV, pp. 706–717.

[65] TOGAI(1991): *FC110 Fuzzy processor data sheets*, Togai InfraLogic, Inc., Irvine, CA., V.S.A.

[66] XILINX(1991): *Programmable gate array data book.*, XILINX, San Jose, California.

[67] DAVIO, M., DESCHAMPS, J.P., THAYSE, A.(1983): *Digital systems with algorithm implementation.*, Wiley–Interscience Publication, John Wiley & Sons.

[68] WALLACE, C.S.(1963): *A suggestion for a fast multiplier.*, IEEE Trans. EC–13, pp.14–17.

[69] LATTICE(1990): *GAL Data Book.*, Lattice Semiconductor Corporation.,Hillsboro, Oregon,V.S.A.

[70] TEXAS INSTRUMENTS(1992): *FPGA Application Handbook.*, Texas Instruments Incorporated.

[71] ATKINS, D.E., ONG, S.(1979): *Time–component complexity of two approaches to multioperand binary addition.*, IEEE Trans. Computers, vol. C–28, no. 12, pp. 918–926.

[72] LAI, H.C., MUROGA, S.(1979): *Minimum parallel binary adders with Nor(Nand) gates.*, IEEE Trans. Computers. vol. C–28, no. 9, pp. 648–659.

[73] CHEN, I., WILLONER, R.(1979): *An 0(n) parallel multiplier with bit–sequential input and output.*, IEEE Trans. Computers, vol. C–28, no. 10, pp. 721–727.

[74] MOU, Z., JUTAND, F.(1992): *"Overturned–stairs" adder trees and multiplier design.*,IEEE Trans. Computers, vol. 41, no. 8, pp. 940–948.

[75] STEFANELLI, R.(1972): *A suggestion for a high–speed parallel binary divider.* IEEE Trans. Computers, vol. C–21, no. 1, pp. 42–55.

[76] LU, M., CHIANG, J.(1992): *A novel division algorithm for the residue number system.*, IEEE Trans. Computers, vol. 41, no.8, pp. 1026–1032.

[77] WONG, D., FLYNN, M.(1992): *Fast division using accurate quotient approximations to reduce the number of iterations.*, IEEE Trans. Computers, vol. 41, no. 8, pp.981–995.

[78] HASAN, M.A., BHARGAVA, V.K.(1992): *Bit–serial systolic divider and multiplier for finite fields GF(2^m).* IEEE Trans. Computers, vol. 41, no. 8, pp. 972–980.

[79] LIU, T., HOHULIN, K.R., SHIAU, L. MUROGA, S.(1974): *Optimal one–bit full adders with different types of gates.*, IEEE Trans. Computers, vol. C–23, no. 1, pp. 63–69.

[80] FLETCHER, W.I.(1980): *Engineering approach to digital design.* Prentice–Hall, Inc. Englewood Cliffs, N.J.

[81] NEURALOGIX(1992): *NLX230 fuzzy micro controller data sheet.* American NeuraLogix, Inc. Sanford, FL., U.S.A.

[82] OMRON(1991): *FP–3000 fuzzy processor data sheet.* Omron Corporation, Fuzzy Technology Center, Kyoto, Japan.

[83] SANCHEZ, E.(1976): *Resolution of composite fuzzy relational equations.*, Inf. & Control, vol. 30, pp. 38–48.

[84] CZOGALA, E., PEDRYCZ, W.(1981): *On identification in fuzzy systems and its applications in control problems.*, Fuzzy Sets Syst., vol. 6, pp. 73–83.

[85] PEDRYCZ, W.(1984): *An identification algorithm in fuzzy relational systems.*, Fuzzy Sets Syst., vol. 13, pp. 153–167.

[86] SHINNERS, S.M.(1979): *Modern control system theory and applications.* Addison–Wesley, U.S.A.

[87] TONG, R.M.(1978): *Synthesis of fuzzy models for industrial processes.*, Int. Gen. Syst., vol. 4, pp. 143–162.

[88] CZOGALA, E., PEDRYCZ, W.(1981): *On identification in fuzzy systems and its applications in control problems.*, Fuzzy Sets Syst., vol. 6, no. 1, pp.73–83.

[89] CZOGALA, E., PEDRYCZ, W.(1982): *Fuzzy rule generation for fuzzy control.*, Cybern. Syst., vol. 13, no.3, pp. 275–293.

[90] TAKAGI, T.,SUGENO, M.(1985): *Fuzzy identification of systems and its applications to modeling and control.*, IEEE Trans. Syst. Man Cybern., vol. SMC–15, no. 1, pp.116–132.

[91] SUGENO, M., KANG, G.T.(1988): *Structure identification of fuzzy model.*,Fuzzy Sets Syst., vol. 28, no. 1, pp. 15–33.

[92] TITLI, T.A.W(1993): *Practical tools for simulation and optimization fuzzy systems with various operators and defuzzification methods.*, Proc. EUFIT'93, First European Congress on Fuzzy and Intelligent Technologies, Aachen, September.

ooOOoo

INDEKS VAN AANHANGSELS.

AANHANGSEL 1: ANALITIESE METODES VIR WASIGE BEHEER:

'n Algemene kritiek van wasige beheerstegnieke was dat daar geen analitiese metodes, soortgelyk aan wortellokus of Nyquistanalise, bestaan vir die bepaling van stabiliteit, verbyskiet, of ander ontwerpsparameters van 'n wasige beheerder, nie.

Hierdie kritiek, alhoewel te verstane, is gegrond op 'n konvensionele denkwyse. Die leser word daaraan herinner dat wasige metodes inherent kwalitatief en nie analities is nie. Dit is juis hierdie eienskap van wasige stelsels wat die gebruiker in staat stel om onpresiese en kwalitatiewe inligting voor te stel. Dit is dus verstaanbaar dat veral stabiliteitsanalise nou onlangs eers aandag geniet het.

A.1.1) Analise en sintese van die ontwerpsparameters:

King en Mamdani [48] het al reeds in 1975 'n metode voorgestel vir die verifikasie en die sintese van reëls deur gebruik te maak van 'n fasevlakbenadering[86]. Die reël verifikasie word gebaseer op 'n geslote stelsel trajek in die fasevlak. 'n Kennis van parameterverstelling gebaseer op fasevlak tegnieke, byvoorbeeld verbyskiet, stygtyd, ensovoorts, en 'n intuïtiewe voeling is egter nodig. Ander outeurs het ook 'n soortgelyke analise voorgestel[87][32]. Verdere voorstelle vir wasige relasionele vergelykings was gemaak in [88][89] en vir linguistiese beheerreëls in [16][90][91].

ii) Wasige Stabiliteitsanalise:

'n Baie interesante en oorspronklike bydrae is onlangs, 1991, gemaak in die gebied van wasige stabiliteitsanalise. YU. I. Kudinov[12] het voorgestel dat die stabiliteit van 'n wasige stelsel gebaseer kan word op die energie van die wasige reëlversameling of relasie.

Vir hierdie metode word die relasie, R, omskryf na die wasige versameling:

$$X_{t+1} = X_o \circ R^t \qquad \text{... A.1.1}$$

waar $R^t = \{R \circ R \circ \ldots \circ R\}$ t keer.

Indien die beginsel van energie van 'n wasige versameling X gegee word deur

$$P(X) = \frac{1}{n} \sum_{i=1}^{n} x_i \mu_X(x_i) \qquad \text{.... A.1.2}$$

met $x_i \epsilon X$

en wasige relasie R

$$P(R) = \frac{1}{n\ m} \sum_{i=1}^{} \sum_{j=1}^{} x_i y_i \mu_R(x_i, y_i) \qquad \text{.... A.1.3}$$

met $x_i, y_i \epsilon X$

met 'n begintoestand x_o bekend.

Die uitspraak aangaande stabiliteit kan dan gemaak word gebaseer op die volgende karakteristieke vergelykening

$$\Delta P = P(X_t) - P(X_{t-1}) = P(x_o \circ R^t) - P(x_o \circ R^{t-1}) \qquad \text{.... A.1.4}$$

Omdat $P(x_o)$ 'n konstante is kan die karakteristieke funksie bereken word as volg

$$\Delta P_c(R,t) = P(R^t) - P(R^{t-1}) \qquad \text{.... A.1.5}$$

Definisie: 'n Wasige stelsel is:

a) Stabiel indien :

$$\Delta P_c(R,t) \leq 0 \text{ as } t \to \infty$$

b) Onstabiel indien:

$$\Delta P_c(R,t) > 0 \text{ as } t \to \infty$$

c) Ossilatories indien:

$$|\Delta P_c(R,t)| = |\Delta P_c(R,t+\tau)| \text{ met } \tau \text{ 'n ossilasie periode.}$$

Die outeur[12] meld ook dat dit sinvol is om waardes van $t\epsilon$ [3;50] te kies, afhangende van die gebruiksmiddele tot beskikking. Dit is dus nie nodig om die karakteristiekefunksie analities op te los, vir $t \to \infty$ nie.

ooOOoo

AANHANGSEL 2.4.P:

AFSPEELSTUDIE VAN WASIGE SELFLERENDE ALGORITMES.

A.2.1 Inleiding:

'n Aantal metodes vir die identifikasie en selfleering van relasionele wasige modelle was al voorgestel [83] [84] [85] [1] [26]. Die wasige relasionele model bied 'n alternatief vir die reëlgebaseerde model. Die relasionele model het die voordeel dat dit direk van inset en uitset–data (i/u–data) verkry word en dat die iteratiewe opleiding en optimiseering van die modelreëls ingesluit is in die algoritme.

A.2.2. Studie van selflerende algoritmes:

i) Die metodes van Czogala , Pedrycz en Xu & Lu:

Alhoewel selfaanpassende en selflerende wasige stelsels nie 'n nuwe beginsel is nie [6], is dit baie seldsaam om 'n studie op te spoor wat verskillende algoritmes vergelyk. Dit is om hierdie rede dat dit baie bevredigend was om 'n publikasie aangaande 'n empiriese vergelyking tussen vier identifikasie– en selflerende algoritmes, te vind. Hierdie outeur, B. Postlethwaite[40][45], het ook 'n vergelyking, op die selfde basis, van die algoritme wat as die kern van hierdie projek gebruik word [39] gelewer, maar hierdie resultate is nog slegs as 'n interne publikasie van die Universiteit van Strathclyde in Skotland beskikbaar [45]. Dit is dus sinvol om 'n opsomming van hierdie publikasie en aanvullende artikel se resultate in hierdie aanhangsel te gee. Ook word daar 'n kritiese evaluasie van die Kosko–selflerende metodiek [1] gegee.

[40] vergelyk empiries twee identifiseringstegnieke, die Czogala–metode [84] en die

Pedrycz–metode[85], en 'n selflerende algoritme, die Xu & Lu– metode[26], met 'n vooraf ontwikkelde reël–gebaseerde model. 'n Selflerende algoritme is 'n algoritme wat beide die identifiserings– en beramingsmetodiek bevat.

Die mees algemene metode om waardes toe te ken aan 'n relasionele matriks is om dit te verkry vanaf versamelinge van die stelsel se i/u–data. Vir elke versameling van i/u–data bestaan daar 'n familie van relasionele matrikse wat die verbindtenis van uitset–data tot inset–data sal verteenwoordig. Sanches [83] het 'n metode voorgestel wat die teoretiese basis gevorm het vir die publikasies van Czogala en Pedrycz [84], en later Pedrycz alleen [85]. Hierdie twee publikasies stel 'n selflerende relasionele model voor wat as volg geskryf kan word vir 'n multi–inset enkel–uitset stelsel:

$$R_j=\cup_{j=0}^{J}(u_1\times u_2\times\ldots\times y)_j \quad \textit{met } R_0=0. \qquad \ldots. \text{A.2.1}$$

Gegewe 'n versameling van verwasige i/u–data vir die j–de monster, $\{u_1,u_2,\ldots,u_k,y\}_j$ *Waar die kruisproduk bereken kan word deur die minimum– of produk–saamstellende operator[3] te gebruik, en die wasige vereniging kan met die maksimum–operator[24] bereken word.*

Xu en Lu[26] het [85] se metode aangepas deur sy identifikasie metode te gebruik, maar gebruik 'n ander beramer tydens die beramingsproses. Vir die konvensionele Pedrycz relasionele metode, word die wasige Kartesiese produk van die individuele grade van lidmaatskap van die inset wasige versamelinge gebruik, naamlik:

$$Y'(\gamma)=maks\ maks.....maks\ min[R(\gamma,a,b,\ldots,z),u_1(a),u_2(b),\ldots,u_k(z)] \quad \ldots.\text{A.2.2}$$

vir *maks–min* operators soos gedefinieer in [3].

Die beramingsmetode wat deur Xu & Lu[26] voorgestel is gebruik egter nie die inset–lidmaatskapsgrade direk nie, maar gebruik die volgende uitdrukking:

$$Y'(\gamma) = \underset{a\epsilon M}{maks}\ \underset{b\epsilon M}{maks} \ldots \underset{z\epsilon M}{maks}\{R[\gamma,\lambda_1(a),\lambda_1(b),\ldots,\lambda_k(z)]\} \qquad \ldots\ \text{A.2.3}$$

met: $\lambda_1(a) \epsilon \lambda_1 = \{m | U_1(m) > q, m\epsilon(1,2,\ldots,M)\}$ en netso vir λ_2, λ_3, ensovoorts, en verder is q 'n vooraf gedefinieerde drumpel.

Hierdie is 'n baie interessante verskynsel aangesien die beramings dus suiwer van die relasie–inhoud afhanklik is, en nie deur die absolute inset–lidmaatskapswaardes beïnvloed word nie. Die vooraf–bepaalde drumpel, q, beperk die aantal relasiereëls wat geaktiveer word slegs tot daardie reëls waarvan die inset–lidmaatskap groter as die drumpel is. Die waarde van q gee dus 'n direkte beheer aangaande die mate van wasigheid van die beraming, met lae waardes van q wat hoë vlakke van wasigheid impliseer. Indien $q = 0$ gekies word sal die beramings dus nutteloos wees, aangesien elke relasie inskrywing geaktiveer sal word vir elke beraming. Netso, is die keuse van q baie belangrik waar van oorvleuelende referensiegebiede gebruik gemaak word.

[26] het dus identifikasie 'n stap verder geneem deur 'n algoritme voor te stel vir die selflering van die relasiemodel. Hierdie leerproses was egter nie intyds gedoen nie, maar wel deur gemonsterde toetsdata te gebruik en herhaaldelik hierdie waardes aan te pas.

Postlethwaite se gevolgtrekking aangaande hierdie metodes was as volg [40]:

Van die metodes van relasionele identifikasie en beraming wat getoets was het Pedrycz die beste algehele resultate gelewer. Die identifikasie metode van Czogala[84] het 'n ernstige tekortkoming gehad, tot so 'n mate dat die model in sekere gevalle glad nie instaat was om 'n beraming te lewer nie. Die beste resultate van die Xu & Lu–metode[26] was

gelykwaardig aan die van die Pedrycz–metode, maar was baie sensitief betreffende die waarde van q. Hierdie sensitieweteit sal nadelig wees vir industriële toepassings. Dit was egter interessant dat Postlethwaite gevind het dat geen van die selfidentifiseerende modelle die resultate van die reël–gebaseerende model kon oortref nie. Selflering wil voorkom om goed te werk vir ruis–vrye data, maar sodra ruis intree in die stelsel het die selflerende stelsels se resultate verswak.

Postlethwaite het in 'n opvolgende publikasie[45] die metode van [39] vergelyk met die metodes van Pedrycz[85] en Xu en Lu[26]. Hierdie outeur het gevind dat die algoritme van [39] beter beramingsresultate lewer as die metode van Pedrycz en ook beter instaat is om identifikasieruis te hanteer. Postlethwaite het ook die metode van [39] aangepas na 'n inkrimentele vorm om sodoende met sy aangepaste Xu & Lu–metode vergelyk te kan word. Die resultate het getoon dat die aangepaste vorm van die algoritme van [39], baie beter beramingsresultate gelewer het as die aangepaste Xu & Lu–metode. Weereens was die aangepaste vorm van [39] se algoritme minder geaffekteer deur stelselruis as die Xu&Lu–metode. Verder het hierdie navorser ook beweer dat die metode van [39] makliker was om te optimiseer vir 'n spesifieke aanleg as die aangepaste Xu & Lu–metode.

Postlethwaite kom tot die slotsom dat, van al die algoritmes wat getoets was, die algoritme van [39] die beste beramingsresultate, die grootste ruis–immuniteit en gemak van toepassing, getoon het

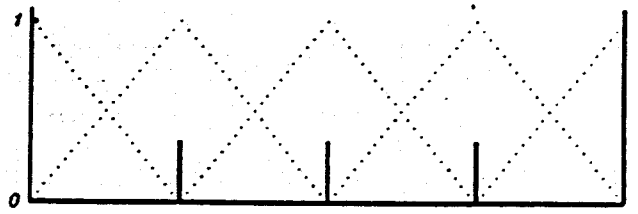
ii) Die Kosko–metodiek:

Metodes van selflering wat baie populêr in hedendaagse wasige logika kringe is, is twee metodes wat deur Bart Kosko [1] voorgestel is. Die metodes, genoem die **Lineêre mededingende leerproses**, en die **Differensiële mededingende leerproses**, vind beide hul

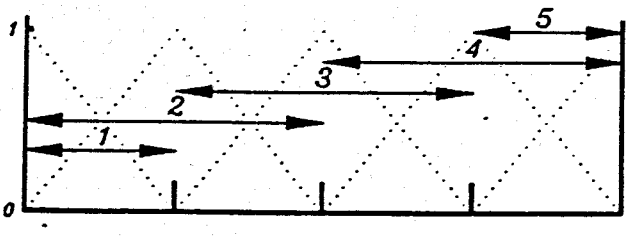
oorsprong in die gebied van neurale netwerke. Baie simplisties gestel, word 'n "mededingende" leerproses gesien as 'n "wedywering" tussen die neurone van 'n neurale netwerk[1]. Soortgelyk, kan gesê word dat die "wedywering" kan geskied tussen die reëls van 'n wasige relasie. Die reël wat oorwin, word die grootste mate van sekerheid toegeken, in die vorm van die gewig van so 'n reël.

Die verskil tussen die lineêre en die differensiële metodiek is dat die lineêre tegniek die reëls deurgaans aanpas, solank as wat die leerproses voortduur. In teenstelling hiermee word die reëls vir die differensiële tegniek slegs aangepas indien daar 'n verandering in die karakteristiek van die datastel bespeur word. Die gewig, of mate van sekerheid, toegeskryf aan 'n reël word dus in die lineêre geval, meer–en–meer aangepas soos die spesifieke reëls weer–en–weer voorkom, om dan net later weer genormaliseer te word. Hierdie onnodige leer–iterasies word vermy in die differesiële tegniek.

Die vraag wat egter nou nog bly bestaan is, hoe word die reëls afgely? Kosko, en ander navorsers[24], het groepering ("clustering") voorgestel. Soos in figure A.2.1 en A.2.2 vertoon, word die gebiede–van–belang van die stelsel veranderlikes, insette en uitsette, verdeel in 'n aantal skerp gedefinieerde gebiede.

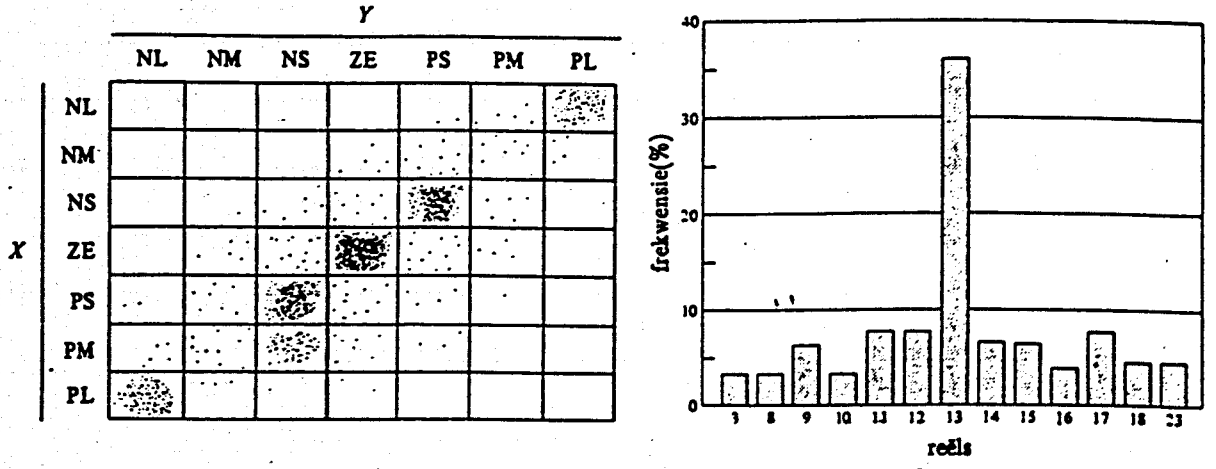


Figuur A.2.1: Skerp gedefinieerde gebiede.



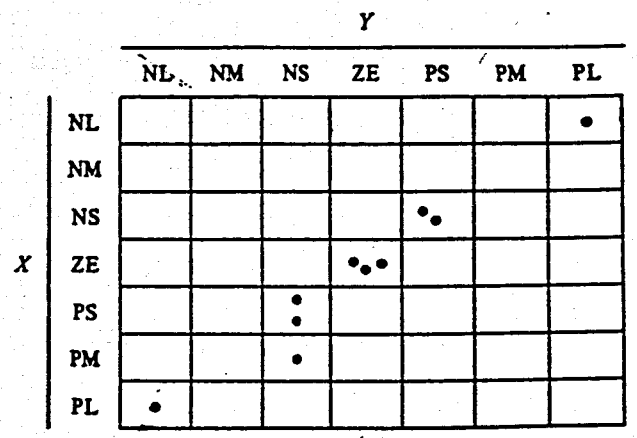
Figuur A.2.2: Basisse van referensiegebiede.

Hierdie skerp gebiede kan ekwiwalent gedefinieer word as die basisse van die referensiegebiede in 'n gebied–van–belang, soos figuur A.2.2 getoon. Die kombinasie van 'n sekere inset– en 'n sekere uitsetpaar, kan vir alle kombinasies van insette en uitsette gesien word as 'n matriks van skerp gedefinieerde gebiede. Die strategie om reëls te bepaal, word deur Kosko eenvoudig gedefinieer as: "Groep is gelyk aan reël". Aangesien elke gebied in die matriks 'n sekere aantal inset–uitset kombinasies verteenwoordig, wat in die respektiewelike inset– en uitsetgebiede val, is dit sinvol om te sê dat daardie matriksgebiede ekwivalent is aan relasie–reëls.



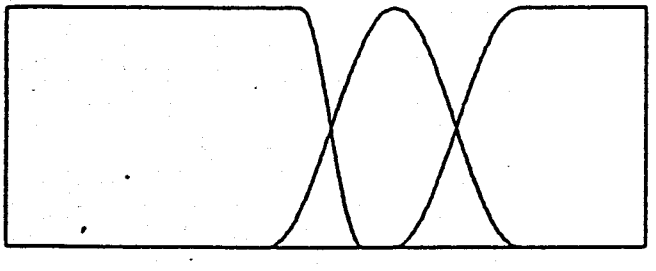
Figuur A.2.3: Frekwensie van voorkoms van reëls.

Die mate van sekerheid wat in 'n reël gestel kan word, of die gewig van 'n reël, word gegee as die frekwensie van voorkoms van die reël, soos in figuur A.2.3 vertoon. In figuur A.2.4 word gewys dat tien inset– en uitsetdatapunte 'n sekere verspreiding het in die reëlmatriks, en ook kan gesien word dat die reël: Indien X=ZE dan is Y=ZE, die hoogste frekwensie van voorkoms het en gevolglik die grootste gewig sal hê.



Figuur A.2.4: Verspreiding van tien data–punte.

Die grootste beswaar teen die groeperingsmetode, wat [1] voorgestel het, is dat dit in essensie nie wasig is nie. 'n Reël wat naby die buitenste dele van 'n reëlgebied voorkom word met net soveel gewig bedeel as 'n reël wat nader aan die middel van die reëlgebied val. Basiese wasige beginsels dikteer dat 'n referensiegebied, en gevolglik ook 'n reëlgebied, met 'n lidmaatskapsfunksie verteenwoordig kan word, soos in figuur A.2.5., met 'n bepaalde verspreiding van sekerheid. Hierdie verspreiding bereik gewoonlik, maar nie noodwendig nie, 'n maksimum in die middel van die gebied.



Figuur A.2.5: Voorbeeld van verspreiding van wasige referensiegebiede.

Hierdie gradering van lidmaatskap tot 'n gebied, is die eienskap wat wasige logika 'n voordeel gee bo ander tegnieke. Die Kosko–metodiek ruil dus eenvoud vir veralgemening.

ooOOoo

AANHANGSEL 3.MM.: BESKRYWING VAN REKURSIEWE MAKS–MIN EN MAKS–PRODUKBERAMER.

Verwys na figuur A.3.MM.

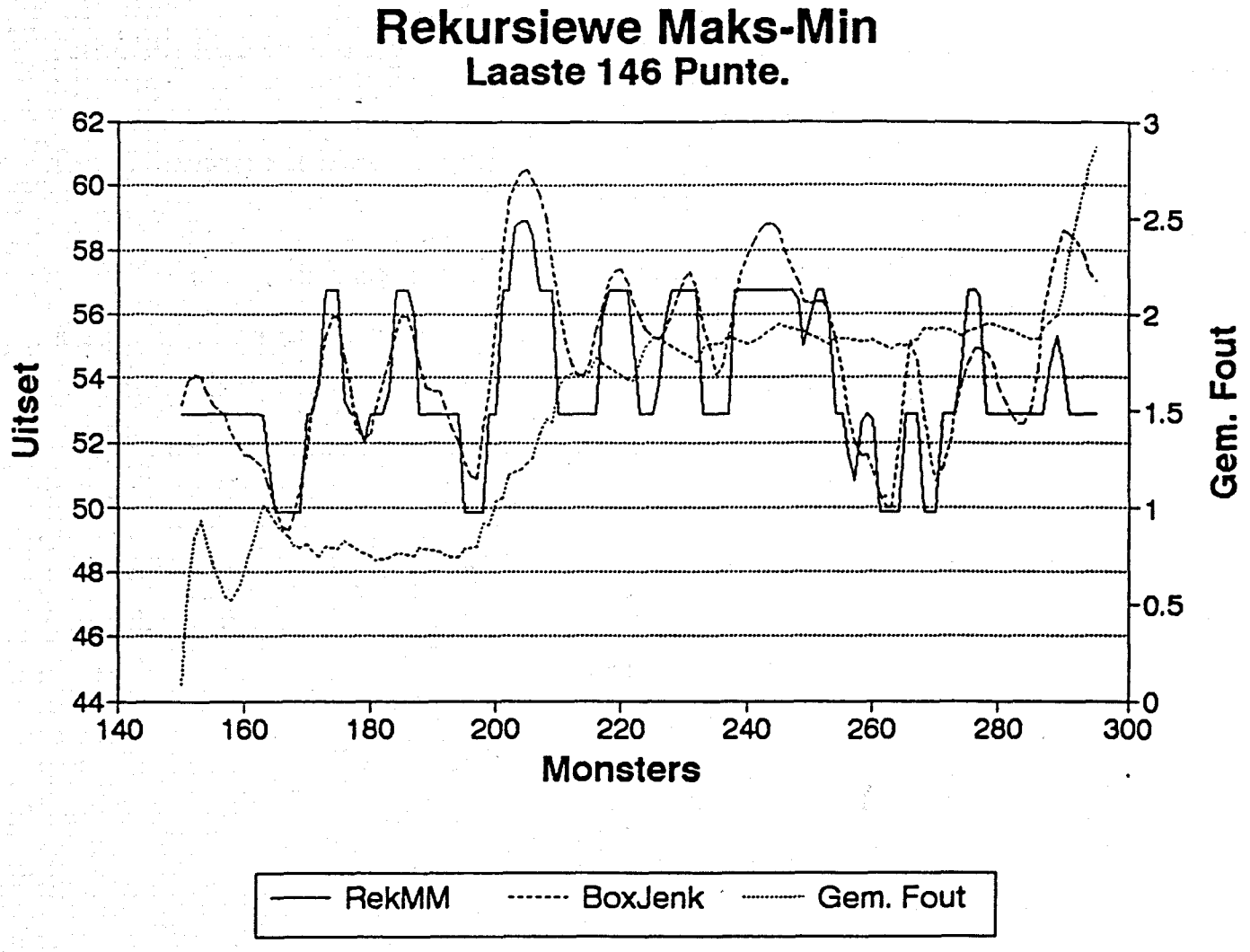
A.3.1. Die rekursiewe maks–min beramer.

Die maks–min tipe van wasige beramer is al in die sestigs deur Zadeh[2] voorgestel. Dit is waarskynlik die mees algemene gebruikte wasige beramermetodiek[24]. Die maks–min–metodiek is egter nog altyd as 'n nie–rekursiewe beramer gebruik. Dit is daarom 'n logiese gevolg dat hierdie metodiek se optrede as 'n rekursiewe beramer ondersoek moes word.

Die Rekursiewe Maks–Min–beramer het baie teleurstellende resultate gelewer, figuur A.3.MM. Die beraming van 149 punte in die toekoms lewer 'n swak $J = 2,866$. Die grootste bydraende gedeelte van hierdie gemiddeld kwadraat fout, J, word gevind in die laaste sowat 180 punte van die beraming. Vir die eerste sowat 111 waardes is J nog steeds onder die perk van $J = 1$, maar styg baie vinnig hierna om die swak $J = 2,866$ te lewer. Hierdie soort beramer kan dus slegs gebruik word vir die beraming van klein hoeveelhede datapunte, waar presiese volging nie so belangrik is nie.

Daar is 'n interessante verskynsel dat die beraming baie blokvormig en afgeplat is. Dit wil voorkom asof die tekort aan beramingsakkuraatheid die model baie minder sensitief maak vir stelselveranderinge. Daar moet relatiewe groot veranderinge plaasvind voor die model reageer.

Figuur A.3.MM: Rekursiewe Maks–min resultate.



Uitvoertyd:

Die rekursiewe maks–min is wel vinniger as die moontlikheidsteoretiese beramer[41].

Maks–min uitvoertyd = 5,05 s J = 2,866

Moontlikheidsteoretiese uitvoertyd = 9,66 s J = 1,791.

A.3.2 Die Rekursiewe Maks–Produk:

Hierdie algoritme het dieselfde versadigingsverskynsel getoon as die maks–min–algoritme. Dit lei tot die gevolgtrekking dat die maks–operator die versadigingseffek in 'n rekursiewe beramer tot gevolg het. Hierdie empiriese waarneming word ook deur ander navorsers beaam. Soos deur Kosko [1] gestel, tree een of ander vorm van die "limiet teorie" [42] in, en word die uitset sistematies na eenheid gedryf. Die rekursiewe maks–min–algoritme is dus ook nie aanvaarbaar vir toepassings in hierdie verhandeling nie.

ooOOoo

Aanhangsel 3.O.1 Turbo–Pascal program van wasige algoritme met slegs nul–eliminasie.

Die program wat in hierdie aanhangsel getoon word, is 'n geoptimiseerde weergawe van die oorpronklike, wat saamgestel was deur I.S. Shaw[41].

```
program fuzzy75(input,output);
{Boxjenk input, non-recursive prediction}
{1st order fuzzy model, with learning}
{$N+}
uses crt;

const
  N = 290;  {number of observations}
  M = 150;

type
  possvec = array[1..5] of extended;              {possibility vector}
  possvecptr = ^possvec;
  fuzzrel = array[1..5, 1..5, 1..5] of extended; {fuzzy relation}

var
  infil1,infil2, outfile,lst                : text;
  i, j, t, k, l, s1, s2, s, tau1, tau2,maxtau : word;
  Eind                                      : integer;
  ymaks,umaks,ymin, ymax, x1min, x1max, x2min, x2max,tyd   : extended;
  denom, temp, SumAy, SumyAy, SumTemp, maxtemp, YhatTemp : extended;
  deltax1, deltax2, deltay, x1factor, x2factor, yfactor : extended;
  p1, p2, p,pp                              : array[1..N] of possvecptr;
  y, u , yhat                               : array[1..N] of extended;
  NN                                        : array[1..3,1..125] of integer;
  ypeak, x1peak, x2peak                     : possvec;
  R                                         : fuzzrel; {fuzzy relation}
  CapYhat                                   : possvec;
  freq                                      : array[1..5,1..5] of extended;
  Initpossvec                               : possvecptr;
  CapJ                                      : extended;

function a1(j : word ; x : extended) : extended;{clustering function for y(k-t
begin
  a1 := 1.0 - abs((x - x1peak[j])/deltax1);    {triangle}
  if x < x1peak[j] - deltax1 then a1 := 0.0;
  if x > x1peak[j] + deltax1 then a1 := 0.0;
end; {a1}

function a2(j : word ; x : extended) : extended;{clustering function for u(k-t
begin
  a2 := 1.0 - abs((x - x2peak[j])/deltax2);     {triangle}
  if x < x2peak[j] - deltax2 then a2 := 0.0;
  if x > x2peak[j] + deltax2 then a2 := 0.0;
end; {a2}

function b(j : word ; x : extended) : extended;{clustering function for y(k)}
begin
  b := 1.0 - abs((x - ypeak[j])/deltay);         {triangle}
  if x < ypeak[j] - deltay then b := 0.0;
  if x > ypeak[j] + deltay then b := 0.0;
end; {b}

function max(x,y:extended):extended;
begin
if x<y then max:= y else max:=x;
end;

function min(x,y:extended):extended;
begin
if x<y then min:=x else min:=y;
end;

function DeFuzz2(A : possvec) : extended;   {defuzzification via fuzzy median}
```

```
begin
  DeFuzz2 := 0.0;
  SumAy := 0.0;
  for l := 1 to 5 do SumAy := SumAy + A[l];
  if SumAy > 0.0 then
  begin
    l := 0 ; temp := 0.0;
    while temp < 0.5 do
    begin
      l := l+1;
      temp := temp + A[l]/SumAy;
    end;
    DeFuzz2 := ypeak[l] + 0.5*deltay*((1.0-2.0*temp)*SumAy/A[l] +1.0);
  end;
end; {DeFuzz2}

function Defuzz1(A: possvec) : extended;{Defuzzification via Fuzzy centroid.}
begin
  Defuzz1:=0.0;
  temp:=0.0;
  SumTemp:=0.0;
  SumAy:=0.0;
    for l:=1 to 5 do
      begin
       temp:=A[l]*ypeak[l];
       SumTemp:=SumTemp+ temp;
       SumAy:=SumAy+A[l];
      end;
  Defuzz1:=SumTemp/SumAy;
end;

function predict(poss1 , poss2 : possvec ; Rel : fuzzrel)  : extended;
begin
for l:=1 to 5 do
  CapYhat[l]:=0.0;

 for t:= 1 to (Eind-1) do
   begin
     s1 := NN[1,t];
       if poss1[s1] > 0 then
        begin
         s2 := NN[2,t];
          l := NN[3,t];
         if (poss2[s2]>0) then
          begin
{          writeln(Eind,' ',t,' ',k,' ',s1,' ',s2,' ',l);}
            CapYhat[l] := CapYhat[l] + poss1[s1] *poss2[s2]*Rel[s1, s2, l];
{            pp[k]^[l] := pp[k]^[l] + poss1[s1]*poss2[s2]*Rel[s1,s2,l];}
          end;
        end;
{      writeln(pp[k]^[l],' ',poss1[s1]);}
    end;
  predict := DeFuzz1(CapYhat);    {predicted value}
end; {predict}

function maxmin(poss1,poss2:possvec; Rel: fuzzrel) : extended;
begin
 for l:=1 to 5 do
  begin
   maxtemp:=0.0;
    for s1:= 1 to 5 do
     begin
      if poss1[s1]>0.0 then
       begin
```

```
        for s2:= 1 to 5 do
         begin
          if (poss2[s2]>0.0) and (Rel[s1,s2,1]>0.0) then
           begin
             YhatTemp:= min(poss1[s1],min(poss2[s2],Rel[s1,s2,1]));
             maxtemp:= max(maxtemp,YhatTemp);
           end;
         end;{s2}
       end;
     end;{s1}
    CapYhat[1]:= maxtemp;
  end;
 Maxmin:= Defuzz1(CapYhat);
end;

procedure fuz;   {open files}
begin
  Assign(infil1, 'c:\wrk\boxjenk1.dat');  {driving function data file}
  Assign(infil2, 'c:\wrk\boxjenk2.dat');  {output function data file}
  Assign(outfile, 'c:\wrk\yhnnrel2.asc');
  Rewrite(outfile);
  Assign(lst,'LPT1');                {printer}
  Rewrite(lst);
end;

procedure AAA;
  begin
          {INPUT DATA AND COMPUTE FUNCTION PARAMETERS}


tau1:=1;tau2:=4;
    x1min := 10000; x1max := -10000; ymin := x1min; ymax := x1max;
    x2min := 10000; x2max := -10000;
    Reset(infil1);
    Reset(infil2);
    if tau1 > tau2 then maxtau:=tau1 else maxtau := tau2;
    ymaks:=0.0;
    umaks:=0.0;
    for k := 1 to N do
       begin
         Readln(infil1, u[k]);   {read variables}
         Readln(infil2, y[k]);
         ymaks:=max(ymaks,abs(y[k]));   {determine maximum values}
         umaks:=max(umaks,abs(u[k]));
       end;
    for k:=1 to N do
       begin
         y[k]:=y[k]/ymaks;
         u[k]:=u[k]/umaks;
          if k > maxtau then
           begin

{compute universes of discourse for all variables: y(k-tau1),u(k-tau2),y(k)}

            x1min := min(x1min, y[k-tau1]); x1max := max(x1max, y[k-tau1]);
            x2min := min(x2min, u[k-tau2]); x2max := max(x2max, u[k-tau2]);
            ymin := min(ymin, y[k]);        ymax := max(ymax, y[k]);
           end;
       end;

{quantizing of universes of discourse for all, variables}

    deltax1 := 0.25*(x1max - x1min);
    deltax2 := 0.25*(x2max - x2min);
```

```
    deltay := 0.25*(ymax - ymin);

{compute peaks of triangular clusters}

    x1peak[1] := 0.5*(x1min + x1max) - 2.0*deltax1;
    x2peak[1] := 0.5*(x2min + x2max) - 2.0*deltax2;
    ypeak[1] := 0.5*(ymin + ymax) - 2.0*deltay;
    for j := 2 to 5 do
    begin
      x1peak[j] := x1peak[j-1] + deltax1 ;
      x2peak[j] := x2peak[j-1] + deltax2 ;
      ypeak[j] := ypeak[j-1] + deltay;
    end;

                          {CREATE RELATION R}

for k:=1 to N do
begin                        {initialize pointers on heap}
    p1[k]:=Initpossvec;
    new(Initpossvec);
    p2[k]:=Initpossvec;
    new(Initpossvec);
    p[k]:=Initpossvec;
    new(Initpossvec);
    pp[k]:=Initpossvec;
    new(Initpossvec);
end;
writeln('Timing starts NOW!');
readln;
    for k:= maxtau to M do   {learning to M}
    begin
      for j := 1 to 5 do
      begin
        p1[k]^[j] := a1(j, y[k-tau1]); {do clustering for y(k-tau1)}
        p2[k]^[j] := a2(j, u[k-tau2]); {do clustering for u(k-tau2)}
        p[k]^[j] := b(j, y[k]);        {do clustering for y(k)}
{        writeln(lst,p1[k]^[j],' ');}
      end;

   end;

Eind:=0;
    for s1 := 1 to 5 do
    begin
      for s2 := 1 to 5 do
      begin
        for s := 1 to 5 do
        begin
          R[s1, s2, s] := 0.0;
          freq[s1, s2] := 0.0;
          for k:= maxtau to M do       {learning to M}
          begin
          if (p1[k]^[s1]>0.00) and (p2[k]^[s2]>0.00) and (p[k]^[s]>0.00) then
           begin
            temp := p1[k]^[s1]*p2[k]^[s2];
            freq[s1, s2] := freq[s1, s2] + temp; {frequency of occurrence}
                                                 {of input combinations}
            R[s1, s2, s] := R[s1, s2, s] + temp*p[k]^[s];{subrelation}
           end;
          end; {k}
          if freq[s1, s2] > 0.0 then R[s1, s2, s] := R[s1, s2, s]/freq[s1,s2];
                                                   {weighted average}
          if (R[s1,s2,s] >0.0) and (R[s1,s2,s]<2) then
             begin
```

```
                NN[1,Eind]:=s1;
                NN[2,Eind]:=s2;
                NN[3,Eind]:=s;
              Eind:=Eind + 1;
            end;
{         writeln(R[s1,s2,s]);}
        end; {s}
      end; {s2}
    end; {s1}
    writeln('R complete');

{  for j:=1 to (Eind-1) do
    begin
     write(j,' ',nn[1,j],' ',nn[2,j],' ',nn[3,j],' ',R[nn[1,j],nn[2,j],nn[3,j]
     writeln;
    end;
    readln;}

              {CALCULATE YHAT}
for k:=maxtau to N do
begin
 for j:=1 to 5 do
 begin
  pp[k]^[j] := 0.0;
  p2[k]^[j] := a2(j, u[k-tau2]);
  p1[k]^[j] := a1(j, y[k-tau1]);
 end;
end;

{for j:=1 to 5 do}        {initialize recursive pp vector,input vector}
{begin
  pp[M]^[j]:= a1(j,y[M-tau1]);
  p2[M]^[j]:= a2(j,u[M-tau2]);
end;}
      for k:=maxtau to N do
 yhat[k] :=ymaks* predict(p1[k]^, p2[k]^, R);

writeln('Timing ends here!');

     for k:=maxtau to N do
       begin
         y[k]:=y[k]*ymaks;
         writeln(yhat[k]:10:6,' ',y[k-maxtau+1]:10:6);
       end;
     for k:= maxtau to N do
       writeln(outfile,yhat[k]:10:6);

end; {AAA}


            {MAIN PROGRAM STARTS HERE}

BEGIN
  Fuz;
    AAA;
    close(infil1);
    close(infil2);       {close all files}
    close(outfile);
{    for k:=maxtau to N do
     begin
     tyd:=0;
      for l:=1 to 5 do
       begin
```

```
        writeln(lst,pp[k]^[l],' ',p[k]^[l]);
        tyd:=tyd+pp[k]^[l];
      end;
     writeln(lst,tyd);
    end;}
END.
```

Aanhangsel 3.T.1 Turbo–Pascal program van wasige algoritme met enkel gebied–van–belang sonder tabel–verwasiging.

Die program wat in hierdie aanhangsel getoon word, is 'n geoptimiseerde weergawe van die oorpronklike, wat saamgestel was deur I.S. Shaw[41].

```
program fuzzy75(input,output);
{Boxjenk input, non-recursive prediction}
{1st order fuzzy model, with learning}
{$N+}
uses crt;

const
  N = 290;  {number of observations}
  M = 150;

type
  possvec = array[1..5] of extended;            {possibility vector}
  possvecptr = ^possvec;
  fuzzrel = array[1..5, 1..5, 1..5] of extended; {fuzzy relation}

var
  infil1,infil2, outfile,lst                : text;
  i, j, t, k, l, s1, s2, s, tau1, tau2,maxtau : word;
  Eind                                      : integer;
  ymaks,umaks,ymin, ymax, x1min, x1max, x2min, x2max,tyd   : extended;
  denom, temp, SumAy, SumyAy, SumTemp, maxtemp, YhatTemp : extended;
  deltax1, deltax2, deltay, x1factor, x2factor, yfactor : extended;
  p1, p2, p,pp                              : array[1..N] of possvecptr;
  y, u , yhat                               : array[1..N] of extended;
  NN                                        : array[1..3,1..125] of integer;
  ypeak, x1peak, x2peak                     : possvec;
  R                                         : fuzzrel; {fuzzy relation}
  CapYhat                                   : possvec;
  freq                                      : array[1..5,1..5] of extended;
  Initpossvec                               : possvecptr;
  CapJ                                      : extended;

function a1(j : word ; x : extended) : extended;{clustering function for y(k-
begin
  a1 := 1.0 - abs((x - x1peak[j])/deltax1);    {triangle}
  if x < x1peak[j] - deltax1 then a1 := 0.0;
  if x > x1peak[j] + deltax1 then a1 := 0.0;
end; {a1}

function a2(j : word ; x : extended) : extended;{clustering function for u(k-t
begin
  a2 := 1.0 - abs((x - x2peak[j])/deltax2);     {triangle}
  if x < x2peak[j] - deltax2 then a2 := 0.0;
  if x > x2peak[j] + deltax2 then a2 := 0.0;
end; {a2}

function b(j : word ; x : extended) : extended;{clustering function for y(k)}
begin
  b := 1.0 - abs((x - ypeak[j])/deltay);         {triangle}
  if x < ypeak[j] - deltay then b := 0.0;
  if x > ypeak[j] + deltay then b := 0.0;
end; {b}

function max(x,y:extended):extended;
begin
if x<y then max:= y else max:=x;
end;

function min(x,y:extended):extended;
begin
if x<y then min:=x else min:=y;
end;

function DeFuzz2(A : possvec) : extended;   {defuzzification via fuzzy mediar
```

```
begin
  DeFuzz2 := 0.0;
  SumAy := 0.0;
  for l := 1 to 5 do SumAy := SumAy + A[l];
  if SumAy > 0.0 then
  begin
    l := 0 ; temp := 0.0;
    while temp < 0.5 do
    begin
      l := l+1;
      temp := temp + A[l]/SumAy;
    end;
    DeFuzz2 := ypeak[l] + 0.5*deltay*((1.0-2.0*temp)*SumAy/A[l] +1.0);
  end;
end; {DeFuzz2}

function Defuzz1(A: possvec) : extended;{Defuzzification via Fuzzy centroid.}
begin
  Defuzz1:=0.0;
  temp:=0.0;
  SumTemp:=0.0;
  SumAy:=0.0;
    for l:=1 to 5 do
      begin
       temp:=A[l]*ypeak[l];
       SumTemp:=SumTemp+ temp;
       SumAy:=SumAy+A[l];
      end;
  Defuzz1:=SumTemp/SumAy;
end;

function predict(poss1 , poss2 : possvec ; Rel : fuzzrel)  : extended;
begin
for l:=1 to 5 do
  CapYhat[l]:=0.0;

 for t:= 1 to (Eind-1) do
   begin
     s1 := NN[1,t];
       if poss1[s1] > 0 then
        begin
         s2 := NN[2,t];
          l := NN[3,t];
         if (poss2[s2]>0) then
          begin
{          writeln(Eind,' ',t,' ',k,' ',s1,' ',s2,' ',l);}
            CapYhat[l] := CapYhat[l] + poss1[s1] *poss2[s2]*Rel[s1, s2, l];
{            pp[k]^[l] := pp[k]^[l] + poss1[s1]*poss2[s2]*Rel[s1,s2,l];}
          end;
        end;
{      writeln(pp[k]^[l],' ',poss1[s1]);}
    end;
  predict := DeFuzz1(CapYhat);    {predicted value}
end; {predict}

function maxmin(poss1,poss2:possvec; Rel: fuzzrel) : extended;
begin
 for l:=1 to 5 do
  begin
   maxtemp:=0.0;
    for s1:= 1 to 5 do
     begin
      if poss1[s1]>0.0 then
       begin
```

```
        for s2:= 1 to 5 do
         begin
          if (poss2[s2]>0.0) and (Rel[s1,s2,1]>0.0) then
           begin
             YhatTemp:= min(poss1[s1],min(poss2[s2],Rel[s1,s2,1]));
             maxtemp:= max(maxtemp,YhatTemp);
           end;
         end;{s2}
       end;
     end;{s1}
    CapYhat[1]:= maxtemp;
  end;
 Maxmin:= Defuzz1(CapYhat);
end;

procedure fuz;   {open files}
begin
  Assign(infil1, 'c:\wrk\boxjenk1.dat');  {driving function data file}
  Assign(infil2, 'c:\wrk\boxjenk2.dat');  {output function data file}
  Assign(outfile, 'c:\wrk\yhnnrel2.asc');
  Rewrite(outfile);
  Assign(lst,'LPT1');                {printer}
  Rewrite(lst);
end;

procedure AAA;
  begin
          {INPUT DATA AND COMPUTE FUNCTION PARAMETERS}


tau1:=1;tau2:=4;
    x1min := 10000; x1max := -10000; ymin := x1min; ymax := x1max;
    x2min := 10000; x2max := -10000;
    Reset(infil1);
    Reset(infil2);
    if tau1 > tau2 then maxtau:=tau1 else maxtau := tau2;
    ymaks:=0.0;
    umaks:=0.0;
    for k := 1 to N do
       begin
         Readln(infil1, u[k]);   {read variables}
         Readln(infil2, y[k]);
         ymaks:=max(ymaks,abs(y[k]));   {determine maximum values}
         umaks:=max(umaks,abs(u[k]));
       end;
    for k:=1 to N do
       begin
         y[k]:=y[k]/ymaks;
         u[k]:=u[k]/umaks;
          if k > maxtau then
           begin

{compute universes of discourse for all variables: y(k-tau1),u(k-tau2),y(k)}

            x1min := min(x1min, y[k-tau1]); x1max := max(x1max, y[k-tau1]);
            x2min := min(x2min, u[k-tau2]); x2max := max(x2max, u[k-tau2]);
            ymin := min(ymin, y[k]);        ymax := max(ymax, y[k]);
           end;
       end;

{quantizing of universes of discourse for all, variables}

    deltax1 := 0.25*(x1max - x1min);
    deltax2 := 0.25*(x2max - x2min);
```

```
    deltay := 0.25*(ymax - ymin);

{compute peaks of triangular clusters}

    x1peak[1] := 0.5*(x1min + x1max) - 2.0*deltax1;
    x2peak[1] := 0.5*(x2min + x2max) - 2.0*deltax2;
    ypeak[1] := 0.5*(ymin + ymax) - 2.0*deltay;
    for j := 2 to 5 do
    begin
      x1peak[j] := x1peak[j-1] + deltax1 ;
      x2peak[j] := x2peak[j-1] + deltax2 ;
      ypeak[j] := ypeak[j-1] + deltay;
    end;

                          {CREATE RELATION R}

for k:=1 to N do
begin                          {initialize pointers on heap}
    p1[k]:=Initpossvec;
    new(Initpossvec);
    p2[k]:=Initpossvec;
    new(Initpossvec);
    p[k]:=Initpossvec;
    new(Initpossvec);
    pp[k]:=Initpossvec;
    new(Initpossvec);
end;
writeln('Timing starts NOW!');
readln;
    for k:= maxtau to M do   {learning to M}
    begin
      for j := 1 to 5 do
      begin
        p1[k]^[j] := a1(j, y[k-tau1]); {do clustering for y(k-tau1)}
        p2[k]^[j] := a2(j, u[k-tau2]); {do clustering for u(k-tau2)}
        p[k]^[j] := b(j, y[k]);        {do clustering for y(k)}
{        writeln(lst,p1[k]^[j],' ');}
      end;

   end;

Eind:=0;
    for s1 := 1 to 5 do
    begin
      for s2 := 1 to 5 do
      begin
        for s := 1 to 5 do
        begin
          R[s1, s2, s] := 0.0;
          freq[s1, s2] := 0.0;
          for k:= maxtau to M do       {learning to M}
          begin
          if (p1[k]^[s1]>0.00) and (p2[k]^[s2]>0.00) and (p[k]^[s]>0.00) then
           begin
            temp := p1[k]^[s1]*p2[k]^[s2];
            freq[s1, s2] := freq[s1, s2] + temp; {frequency of occurrence}
                                                 {of input combinations}
            R[s1, s2, s] := R[s1, s2, s] + temp*p[k]^[s];{subrelation}
           end;
          end; {k}
          if freq[s1, s2] > 0.0 then R[s1, s2, s] := R[s1, s2, s]/freq[s1,s2];
                                                    {weighted average}
          if (R[s1,s2,s] >0.0) and (R[s1,s2,s]<2) then
             begin
```

```
                NN[1,Eind]:=s1;
                NN[2,Eind]:=s2;
                NN[3,Eind]:=s;
              Eind:=Eind + 1;
             end;
{          writeln(R[s1,s2,s]);}
        end; {s}
      end; {s2}
    end; {s1}
    writeln('R complete');

{  for j:=1 to (Eind-1) do
    begin
     write(j,' ',nn[1,j],' ',nn[2,j],' ',nn[3,j],' ',R[nn[1,j],nn[2,j],nn[3,j]
     writeln;
    end;
    readln;}

              {CALCULATE YHAT}
for k:=maxtau to N do
begin
 for j:=1 to 5 do
 begin
  pp[k]^[j] := 0.0;
  p2[k]^[j] := a2(j, u[k-tau2]);
  p1[k]^[j] := a1(j, y[k-tau1]);
 end;
end;

{for j:=1 to 5 do}        {initialize recursive pp vector,input vector}
{begin
  pp[M]^[j]:= a1(j,y[M-tau1]);
  p2[M]^[j]:= a2(j,u[M-tau2]);
end;}
     for k:=maxtau to N do
 yhat[k] :=ymaks* predict(p1[k]^, p2[k]^, R);

writeln('Timing ends here!');

     for k:=maxtau to N do
       begin
         y[k]:=y[k]*ymaks;
         writeln(yhat[k]:10:6,' ',y[k-maxtau+1]:10:6);
       end;
     for k:= maxtau to N do
       writeln(outfile,yhat[k]:10:6);

end; {AAA}


            {MAIN PROGRAM STARTS HERE}

BEGIN
  Fuz;
    AAA;
    close(infil1);
    close(infil2);      {close all files}
    close(outfile);
{    for k:=maxtau to N do
     begin
     tyd:=0;
      for l:=1 to 5 do
       begin
```

```
        writeln(lst,pp[k]^[l],' ',p[k]^[l]);
        tyd:=tyd+pp[k]^[l];
       end;
      writeln(lst,tyd);
     end;}
END.
```

Aanhangsel 3.T.2 Turbo–Pascal program van wasige algoritme met nul–eliminasie en tabel–verwasiging met 'n enkele gebied–van–belang.

Die program wat in hierdie aanhangsel getoon word, is 'n geoptimiseerde weergawe van die oorpronklike, wat saamgestel was deur I.S. Shaw[41].

```
program fuzzy75(input,output);
{Boxjenk input, human operator data,non-recursive prediction}
{1st order fuzzy model, with learning}
{All 0-traps incorporated, vector of Non-zero relation values, 3 identical
 universes of discource, table-fuzzification and centroid defuzzification.}
{$N+}
uses crt;

const
  N = 296;  {number of observations}
  M = 150;

type
  possvec = array[1..5] of extended;            {possibility vector}
  possvecptr = ^possvec;
  fuzzrel = array[1..5, 1..5, 1..5] of extended; {fuzzy relation}

var
  infil1,infil2, outfile1,outfile2,lst     : text;
  i, j, k, l, s1, s2, s, tau1, tau2,maxtau : word;
  ymaks,umaks,ymin, ymax, x1min, x1max, x2min, x2max,tyd   : extended;
  denom, temp, SumAy, SumyAy, SumTemp      : extended;
  delta, x1factor, x2factor, yfactor       : extended;
  p1, p2, p, pp, posv, posv1, posv2        : array[1..N] of possvecptr;
  y, u , yhat                              : array[1..N] of extended;
  waarde                                   : array[0..256] of extended;
  peak                                     : possvec;
  R                                        : fuzzrel; {fuzzy relation}
  CapYhat                                  : possvec;
  freq                                     : array[1..5,1..5] of extended;
  Initpossvec                              : possvecptr;
  CapJ                                     : extended;
  tel, tel1, tel2                          : integer;

function verw(j : integer; x : extended) : extended;{clustering function for y
begin
  verw := 1.0 - abs((x - peak[j])/delta);    {triangle}
  if x < (peak[j] - delta) then verw := 0.0;
  if x > (peak[j] + delta) then verw := 0.0;
end;

function max(x,y:extended):extended;
begin
if x<y then max:= y else max:=x;
end;

function min(x,y:extended):extended;
begin
if x<y then min:=x else min:=y;
end;

function DeFuzz2(A : possvec) : extended;   {defuzzification via fuzzy median}
begin
  DeFuzz2 := 0.0;
  SumAy := 0.0;
  for l := 1 to 5 do SumAy := SumAy + A[l];
  if SumAy > 0.0 then
  begin
    l := 0 ; temp := 0.0;
    while temp < 0.5 do
    begin
      l := l+1;
      temp := temp + A[l]/SumAy;
    end;
```

```
    DeFuzz2 := peak[l] + 0.5*delta*((1.0-2.0*temp)*SumAy/A[l] +1.0);
  end;
end; {DeFuzz2}

function Defuzz1(A: possvec) : extended;{Defuzzification via Fuzzy centroid.}
begin
  Defuzz1:=0.0;
  temp:=0.0;
  SumTemp:=0.0;
  SumAy:=0.0;
    for l:=1 to 5 do
      begin
       temp:=A[l]*peak[l];
       SumTemp:=SumTemp+ temp;
       SumAy:=SumAy+A[l];
      end;
  Defuzz1:=SumTemp/SumAy;
end;

function predict(poss1 , poss2 : possvec ; Rel : fuzzrel) : extended;
begin
  for l := 1 to 5 do
  begin
     CapYhat[l] := 0.0;   {predicted membership function,ie poss.vector}
      for s1 := 1 to 5 do
      begin
       if poss1[s1] > 0 then
        begin
         for s2 := 1 to 5 do
          begin
           if (poss2[s2]>0) and (Rel[s1,s2,l]>0) then
            begin
{             writeln(poss1[s1],' ',poss2[s2],' ',Rel[s1,s2,s],' ',CapYhat[l])
             CapYhat[l] := CapYhat[l] + poss1[s1] *poss2[s2] *Rel[s1, s2, l];
             pp[k]^[l] := pp[k]^[l] + poss1[s1]*poss2[s2]*Rel[s1,s2,l];
            end;
         end; {s2}
{         writeln('End s2.',s1);}
        end;
      end; {s1}
{      writeln('End s1.',l);}
{      writeln(pp[k]^[l],' ',poss1[s1]);}
    end; {l}
{    writeln('End l');}
  predict := DeFuzz2(CapYhat);    {predicted value}
{  writeln('predict(',k,')');}
end; {predict}

procedure verwasig;
var inkr,W: extended;
       j,k:integer;
begin
 for k:=1 to N do
  begin
    p1[k]:=Initpossvec;
    new(Initpossvec);
    p2[k]:=Initpossvec;
    new(Initpossvec);
    p[k]:=Initpossvec;
    new(Initpossvec);
    pp[k]:=Initpossvec;
    new(Initpossvec);
    posv[k]:=Initpossvec;
    new(Initpossvec);
```

```
    posv1[k]:=Initpossvec;
    new(Initpossvec);
    posv2[k]:=Initpossvec;
    new(Initpossvec);
  end;
   inkr:=2/255;
    for k:=0 to 255 do
        waarde[k] := -1+(k*inkr); {Inkrementele opsoek indeks.}

   delta:=0.5;
   peak[1]:=-1;
      for k:=2 to 5 do
          peak[k]:=peak[k-1] + delta;

   for k:= 0 to 255 do
     begin
       for j:= 1 to 5 do
        begin
         posv1[k]^[j] := verw(j, waarde[k]);
         posv2[k]^[j] := verw(j, waarde[k]);
         posv[k]^[j] := verw(j, waarde[k]);
        end;
     end;
end;

procedure fuz;   {open files}
begin
  Assign(infil1, 'c:\wrk\boxjenk1.dat');  {driving function data file}
  Assign(infil2, 'c:\wrk\boxjenk2.dat');  {output function data file}
  Assign(outfile1, 'c:\wrk\yhtabopt.asc');
  Assign(outfile2, 'c:\wrk\rel1.asc');
  Rewrite(outfile1);
  Rewrite(outfile2);
  Assign(lst,'LPT1');                {printer}
  Rewrite(lst);
end;

procedure AAA;
  begin
          {INPUT DATA AND COMPUTE FUNCTION PARAMETERS}


tau1:=1;tau2:=4;
    x1min := 10000; x1max := -10000; ymin := x1min; ymax := x1max;
    x2min := 10000; x2max := -10000;
    Reset(infil1);
    Reset(infil2);
    if tau1 > tau2 then maxtau:=tau1 else maxtau := tau2;
    ymaks:=0.0;
    umaks:=0.0;
    for k := 1 to N do
       begin
         Readln(infil1, u[k]);   {read variables}
         Readln(infil2, y[k]);
         ymaks:=max(ymaks,abs(y[k]));   {determine maximum values}
         umaks:=max(umaks,abs(u[k]));
       end;
    for k:=1 to N do
       begin
         y[k]:=y[k]/ymaks;
         u[k]:=u[k]/umaks;
       end;

                        {CREATE RELATION R}
```

```
    for k:= maxtau to M do   {learning to M}
    begin
      for j := 1 to 5 do
      begin
      tel1 := trunc(((y[k-tau1]+1)*255)/2);
      tel2 := trunc(((u[k-tau2]+1)*255)/2);
      tel  := trunc(((y[k]+1)*255)/2);
        p1[k]^[j] := posv1[tel1]^[j]; {do table clustering for y(k-tau1)}
        p2[k]^[j] := posv2[tel2]^[j]; {do table clustering for u(k-tau2)}
        p[k]^[j] := posv[tel]^[j];    {do clustering for y(k)}
{        writeln(lst,p1[k]^[j],' ');}
      end;
   end;
writeln('Timing Starts NOW!');
writeln('Please Press Enter.');
readln;

    for s1 := 1 to 5 do
    begin
      for s2 := 1 to 5 do
      begin
        for s := 1 to 5 do
        begin
          R[s1, s2, s] := 0.0;
          freq[s1, s2] := 0.0;
          for k:= maxtau to M do       {learning to M}
           begin
            if (p1[k]^[s1]>0) and (p2[k]^[s2]>0) and (p[k]^[s]>0) then
             begin
              temp := p1[k]^[s1]*p2[k]^[s2];
              freq[s1, s2] := freq[s1, s2] + temp; {frequency of occurrence}
                                                   {of input combinations}
              R[s1, s2, s] := R[s1, s2, s] + temp*p[k]^[s];{subrelation}
             end;
           end; {k}
          if freq[s1, s2] > 0.0 then R[s1, s2, s] := R[s1, s2, s]/freq[s1,s2];
                                                     {weighted average}
        end; {s}
      end; {s2}
    end; {s1}
    writeln('R complete');
{    readln;}

{for s1:=1 to 5 do
 begin
  for s2:=1 to 5 do
   begin
    for s:=1 to 5 do
     begin
      writeln(outfile2,R[s1,s2,s]);
     end;
   end;
 end;}

               {CALCULATE YHAT}
for k:=maxtau to N do
begin
 tel2 := trunc(((u[k-tau2]+1)*255)/2);
 tel1 := trunc(((y[k-tau1]+1)*255)/2);
 for j:=1 to 5 do
  begin
   pp[k]^[j] := 0.0;
   p2[k]^[j] := posv2[tel2]^[j];
```

```
   p1[k]^[j] := posv1[tel1]^[j];
  end;
end;
writeln('Relasie');
      for k:=maxtau to N do
 yhat[k] :=ymaks* predict(p1[k]^, p2[k]^, R);

writeln('Timing ends here!');

     for k:=maxtau to N do
       begin
         y[k]:=y[k]*ymaks;
         writeln(yhat[k]:10:6,' ',y[k-maxtau+1]:10:6);
       end;
     for k:= maxtau to N do
       writeln(outfile1,yhat[k]:10:6);

end; {AAA}


            {MAIN PROGRAM STARTS HERE}

BEGIN
ClrScr;
verwasig;
  Fuz;
    AAA;
    close(infil1);
    close(infil2);       {close all files}
    close(outfile1);
    close(outfile2);
{    for k:=maxtau to N do
     begin
     tyd:=0;
      for l:=1 to 5 do
       begin
        writeln(lst,pp[k]^[l],' ',p[k]^[l]);
        tyd:=tyd+pp[k]^[l];
       end;
      writeln(lst,tyd);
     end;}
END.
```

Aanhangsel 3.T.3 Turbo–Pascal program van wasige algoritme met nul–eliminasie en tabel–verwasiging met afsonderlike gebiede–van–belang.

Die program wat in hierdie aanhangsel getoon word, is 'n geoptimiseerde weergawe van die oorpronklike, wat saamgestel was deur I.S. Shaw[41].

```
program fuzzy75(input,output);
{Boxjenk input, human operator data,non-recursive prediction}
{1st order fuzzy model, with learning}
{$N+}
uses crt,dos;

const
  N = 296;  {number of observations}
  M = 150;

type
  possvec = array[1..5] of extended;              {possibility vector}
  possvecptr = ^possvec;
  fuzzrel = array[1..5, 1..5, 1..5] of extended; {fuzzy relation}

var
  infil1,infil2, outfile1,outfile2,lst     : text;
  i, j, k, l, s1, s2, s, tau1, tau2,maxtau, H, Minute, Sec, Sec100, Time : wor
  ymaks,umaks,ymin, ymax, x1min, x1max, x2min, x2max,tyd : extended;
  denom, temp, SumAy, SumyAy, SumTemp      : extended;
  deltax1, deltax2, deltay, x1factor, x2factor, yfactor : extended;
  p1, p2, p, pp, posv, posv1, posv2        : array[1..N] of possvecptr;
  y, u , yhat                              : array[1..N] of extended;
  waardex1,waardex2,waardey               : array[0..256] of extended;
  NN                                       : array[1..3,1..125] of integer;
  x1peak,x2peak,ypeak                      : possvec;
  R                                        : fuzzrel; {fuzzy relation}
  CapYhat                                  : possvec;
  freq                                     : array[1..5,1..5] of extended;
  Initpossvec                              : possvecptr;
  CapJ                                     : extended;
  tel, tel1, tel2, t, Eind                 : integer;

function a1(j : integer; x : extended) : extended;{clustering function for y(k
begin
  a1 := 1.0 - abs((x - x1peak[j])/deltax1);    {triangle}
  if x < (x1peak[j] - deltax1) then a1 := 0.0;
  if x > (x1peak[j] + deltax1) then a1 := 0.0;
end;

function a2(j : integer; x : extended) : extended;{clustering function for y(k
begin
  a2 := 1.0 - abs((x - x2peak[j])/deltax2);    {triangle}
  if x < (x2peak[j] - deltax2) then a2 := 0.0;
  if x > (x2peak[j] + deltax2) then a2 := 0.0;
end;

function b(j : integer; x : extended) : extended;{clustering function for y(k-
begin
  b := 1.0 - abs((x - ypeak[j])/deltay);    {triangle}
  if x < (ypeak[j] - deltay) then b := 0.0;
  if x > (ypeak[j] + deltay) then b := 0.0;
end;

function max(x,y:extended):extended;
begin
if x<y then max:= y else max:=x;
end;

function min(x,y:extended):extended;
begin
if x<y then min:=x else min:=y;
end;
```

```
function DeFuzz2(A : possvec) : extended;   {defuzzification via fuzzy median}
begin
  DeFuzz2 := 0.0;
  SumAy := 0.0;
  for l := 1 to 5 do SumAy := SumAy + A[l];
  if SumAy > 0.0 then
  begin
    l := 0 ; temp := 0.0;
    while temp < 0.5 do
    begin
      l := l+1;
      temp := temp + A[l]/SumAy;
    end;
    DeFuzz2 := ypeak[l] + 0.5*deltay*((1.0-2.0*temp)*SumAy/A[l] +1.0);
  end;
end; {DeFuzz2}

function Defuzz1(A: possvec) : extended;{Defuzzification via Fuzzy centroid.}
begin
  Defuzz1:=0.0;
  temp:=0.0;
  SumTemp:=0.0;
  SumAy:=0.0;
    for l:=1 to 5 do
      begin
       temp:=A[l]*ypeak[l];
       SumTemp:=SumTemp+ temp;
       SumAy:=SumAy+A[l];
      end;
  Defuzz1:=SumTemp/SumAy;

end;

function predict(poss1 , poss2 : possvec ; Rel : fuzzrel) : extended;
begin
  for l := 1 to 5 do
     CapYhat[l] := 0.0;   {predicted membership function,ie poss.vector}
  for t:=1 to (Eind-1) do
    begin
      s1 := NN[1,t];
       if poss1[s1] > 0 then
        begin
         s2:= NN[2,t];
          l:= NN[3,t];
           if (poss2[s2]>0) then
            begin
{             writeln(poss1[s1],' ',poss2[s2],' ',Rel[s1,s2,s],' ',CapYhat[l])
             CapYhat[l] := CapYhat[l] + poss1[s1] *poss2[s2] *Rel[s1, s2, l];
             pp[k]^[l] := pp[k]^[l] + poss1[s1]*poss2[s2]*Rel[s1,s2,l];
            end;
        end;
{      writeln(pp[k]^[l],' ',poss1[s1]);}
    end;
  predict := DeFuzz1(CapYhat);    {predicted value}
end; {predict}

procedure verwasig;
var inkrx1,inkrx2,inkry,W: extended;
      j,k:integer;
begin

tau1:=1;tau2:=4;
    if tau1 > tau2 then maxtau:=tau1 else maxtau := tau2;
```

```
writeln('Begin Pre-processing.');
 for k:=1 to N do
  begin
    p1[k]:=Initpossvec;
    new(Initpossvec);
    p2[k]:=Initpossvec;
    new(Initpossvec);
    p[k]:=Initpossvec;
    new(Initpossvec);
    pp[k]:=Initpossvec;
    new(Initpossvec);
    posv[k]:=Initpossvec;
    new(Initpossvec);
    posv1[k]:=Initpossvec;
    new(Initpossvec);
    posv2[k]:=Initpossvec;
    new(Initpossvec);
  end;

    x1min := 10000; x1max := -10000; ymin := x1min; ymax := x1max;
    x2min := 10000; x2max := -10000;
    Reset(infil1);
    Reset(infil2);

    ymaks:=0.0;
    umaks:=0.0;
    for k := 1 to N do
       begin
         Readln(infil1, u[k]);    {read variables}
         Readln(infil2, y[k]);
         ymaks:=max(ymaks,abs(y[k]));    {determine maximum values}
         umaks:=max(umaks,abs(u[k]));
       end;

    for k:=1 to N do
       begin
         y[k]:=y[k]/ymaks;
         u[k]:=u[k]/umaks;
           if k > maxtau then
             begin
              x1min := min(x1min, y[k-tau1]); x1max := max(x1max, y[k-tau1]);
              x2min := min(x2min, u[k-tau2]); x2max := max(x2max, u[k-tau2]);
               ymin := min(ymin, y[k]);        ymax := max(ymax, y[k]);
             end;
       end;
       writeln('Universe Of Discourse');

   inkrx1:=(x1max-x1min)/255;
   inkrx2:=(x2max-x2min)/255;
   inkry:=(ymax-ymin)/255;
    for k:=0 to 255 do
      begin
       waardex1[k] := x1min + (k*inkrx1); {Inkrementele opsoek indeks.}
       waardex2[k] := x2min + (k*inkrx2);
       waardey[k]  := ymin  + (k*inkry);
      end;

   deltax1 := 0.25 * (x1max - x1min);
   deltax2 := 0.25 * (x2max - x2min);
   deltay  := 0.25 * (ymax  -  ymin);
   x1peak[1] := 0.5*(x1min + x1max) - 2.0*deltax1;
   x2peak[1] := 0.5*(x2min + x2max) - 2.0*deltax2;
    ypeak[1] := 0.5*(ymin + ymax) - 2.0*deltay;
```

```
      for k:=2 to 5 do
        begin
          x1peak[k] := x1peak[k-1] + deltax1;
          x2peak[k] := x2peak[k-1] + deltax2;
           ypeak[k] := ypeak[k-1] + deltay;
        end;

   for k:= 0 to 255 do
     begin
       for j:= 1 to 5 do
        begin
         posv1[k]^[j] := a1(j, waardex1[k]);
         posv2[k]^[j] := a2(j, waardex2[k]);
         posv[k]^[j] := b(j, waardey[k]);
{         writeln('posv1',posv1[k]^[j],' waarde',waardex1[k]);}
        end;
     end;
     writeln('End Pre-processing.');
end;

procedure fuz;   {open files}
begin
  Assign(infil1, 'c:\wrk\boxjenk1.dat');  {driving function data file}
  Assign(infil2, 'c:\wrk\boxjenk2.dat');  {output function data file}
  Assign(outfile1, 'c:\wrk\taboptaf.asc');
  Assign(outfile2, 'c:\wrk\rel1.asc');
  Rewrite(outfile1);
  Rewrite(outfile2);
  Assign(lst,'LPT1');                {printer}
  Rewrite(lst);
end;

procedure AAA;
  begin
          {INPUT DATA AND COMPUTE FUNCTION PARAMETERS}


                        {CREATE RELATION R}

    for k:= maxtau to M do   {learning to M}
    begin
      for j := 1 to 5 do
      begin
      tel1 := trunc(((y[k-tau1]-x1min)*255)/(x1max-x1min));
      tel2 := trunc(((u[k-tau2]-x2min)*255)/(x2max-x2min));
      tel  := trunc(((y[k]-ymin)*255)/(ymax-ymin));
        p1[k]^[j] := posv1[tel1]^[j]; {do table clustering for y(k-tau1)}
        p2[k]^[j] := posv2[tel2]^[j]; {do table clustering for u(k-tau2)}
        p[k]^[j]  := posv[tel]^[j];    {do clustering for y(k)}
{        write(p1[k]^[j],' ');}
      end;
   end;
writeln;
writeln('Timing Starts NOW!');
writeln('Please Press Enter.');
readln;
settime(0,0,0,0);
for s:=0 to 125 do
begin
  nn[1,s]:=0;
  nn[2,s]:=0;
  nn[3,s]:=0;
end;
```

```
Eind:=0;
    for s1 := 1 to 5 do
    begin
      for s2 := 1 to 5 do
      begin
        for s := 1 to 5 do
        begin
          R[s1, s2, s] := 0.0;
          freq[s1, s2] := 0.0;
          for k:= maxtau to M do       {learning to M}
           begin
            if (p1[k]^[s1]>0) and (p2[k]^[s2]>0) and (p[k]^[s]>0) then
             begin
              temp := p1[k]^[s1]*p2[k]^[s2];
              freq[s1, s2] := freq[s1, s2] + temp; {frequency of occurrence}
                                                   {of input combinations}
              R[s1, s2, s] := R[s1, s2, s] + temp*p[k]^[s];{subrelation}
             end;
           end; {k}
          if freq[s1, s2] > 0.0 then R[s1, s2, s] := R[s1, s2, s]/freq[s1,s2];
                                                     {weighted average}
          if R[s1,s2,s]> 0.0 then
            begin
              NN[1,Eind]:=s1;
              NN[2,Eind]:=s2;
              NN[3,Eind]:=s;
                    Eind:=Eind +1;
            end;
        end; {s}
      end; {s2}
    end; {s1}
    writeln('R complete');
    writeln;
{    readln;}

{for s1:=1 to 5 do
 begin
  for s2:=1 to 5 do
   begin
    for s:=1 to 5 do
     begin
      writeln(outfile2,R[s1,s2,s]);
     end;
   end;
 end;}

              {CALCULATE YHAT}
for k:=maxtau to N do
begin
 tel2 := trunc(((u[k-tau2]-x2min)*255)/(x2max-x2min));
 tel1 := trunc(((y[k-tau1]-x1min)*255)/(x1max-x1min));
 for j:=1 to 5 do
  begin
   pp[k]^[j] := 0.0;
   p2[k]^[j] := posv2[tel2]^[j];
   p1[k]^[j] := posv1[tel1]^[j];
  end;
end;
      for k:=maxtau to N do
 yhat[k] :=ymaks* predict(p1[k]^, p2[k]^, R);
gettime(H,Minute,Sec,Sec100);
time:=sec*100+sec100;
writeln('Timing ends here!');
writeln('Time= ',time/100:2:2,' sec.');
```

```
writeln('J=0.587');
writeln;
writeln('Please Press Enter.');
readln;

{     for k:=maxtau to N do
       begin
         y[k]:=y[k]*ymaks;
         writeln(yhat[k]:10:6,' ',y[k-maxtau+1]:10:6);
       end;
     for k:= maxtau to N do
       writeln(outfile1,yhat[k]:10:6);}

end; {AAA}


            {MAIN PROGRAM STARTS HERE}

BEGIN
ClrScr;
Fuz;
verwasig;
    AAA;
    close(infil1);
    close(infil2);       {close all files}
    close(outfile1);
    close(outfile2);
{    for k:=maxtau to N do
     begin
     tyd:=0;
      for l:=1 to 5 do
       begin
        writeln(lst,pp[k]^[l],' ',p[k]^[l]);
        tyd:=tyd+pp[k]^[l];
       end;
      writeln(lst,tyd);
     end;}
END.
```

Aanhangsel 3.R.1 Turbo–Pascal program van wasige algoritme met rekursiewe wasige beraming, nul–eliminasie en tabel–verwasiging.

Die program wat in hierdie aanhangsel getoon word, is 'n geoptimiseerde weergawe van die oorpronklike, wat saamgestel was deur I.S. Shaw[41].

```
program fuzzy75(input,output);
{boxjenk input, human operator data, recursive prediction}
{1st order fuzzy model, with learning}

uses crt,dos;

const
  N = 296;  {number of observations}
  M = 150;

type
  possvec = array[1..5] of extended;              {possibility vector}
  possvecptr = ^possvec;
  fuzzrel = array[1..5, 1..5, 1..5] of extended; {fuzzy relation}

var
  infil1,infil2, outfile,lst                  : text;
  i, j, k, l, s1, s2, s, tau1, tau2,maxtau,H,Minute,sec,sec100,time : word;
  te1, te11, te12, t, Eind                    : integer;
  ymaks,umaks,ymin, ymax, x1min, x1max, x2min, x2max   : extended;
  denom, temp, SumAy, SumyAy, SumTemp         : extended;
  deltax1, deltax2, deltay, x1factor, x2factor, yfactor : extended;
  p1, p2, p,pp,posv,posv1,posv2               : array[1..N] of possvecptr;
  y, u , yhat                                 : array[1..N] of extended;
  waardex1, waardex2, waardey                 : array[0..256] of extended;
  NN                                          : array[1..3,1..125] of integer;
  ypeak, x1peak, x2peak                       : possvec;
  R                                           : fuzzrel; {fuzzy relation}
  CapYhat                                     : possvec;
  freq                                        : array[1..5,1..5] of extended;
  Initpossvec                                 : possvecptr;
  CapJ                                        : extended;

function a1(j : word ; x : extended) : extended;{clustering function for y(k-t
begin
  a1 := 1.0 - abs((x - x1peak[j])/deltax1);    {triangle}
  if x < x1peak[j] - deltax1 then a1 := 0.0;
  if x > x1peak[j] + deltax1 then a1 := 0.0;
end; {a1}

function a2(j : word ; x : extended) : extended;{clustering function for u(k-t
begin
  a2 := 1.0 - abs((x - x2peak[j])/deltax2);     {triangle}
  if x < x2peak[j] - deltax2 then a2 := 0.0;
  if x > x2peak[j] + deltax2 then a2 := 0.0;
end; {a2}

function b(j : word ; x : extended) : extended;{clustering function for y(k)}
begin
  b := 1.0 - abs((x - ypeak[j])/deltay);         {triangle}
  if x < ypeak[j] - deltay then b := 0.0;
  if x > ypeak[j] + deltay then b := 0.0;
end; {b}

function max(x,y:extended):extended;
begin
if x<y then max:= y else max:=x;
end;

function min(x,y:extended):extended;
begin
if x<y then min:=x else min:=y;
end;
```

```
function DeFuzz2(A : possvec) : extended;   {defuzzification via fuzzy median}
begin
  DeFuzz2 := 0.0;
  SumAy := 0.0;
  for l := 1 to 5 do SumAy := SumAy + A[l];
  if SumAy > 0.0 then
  begin
    l := 0 ; temp := 0.0;
    while temp < 0.5 do
    begin
      l := l+1;
      temp := temp + A[l]/SumAy;
    end;
    DeFuzz2 := ypeak[l] + 0.5*deltay*((1.0-2.0*temp)*SumAy/A[l] +1.0);
  end;
end; {DeFuzz2}

function Defuzz1(A: possvec) : extended;{Defuzzification via Fuzzy centroid.}
begin
  Defuzz1 := 0.0;
  SumTemp := 0.0;
     temp := 0.0;
    SumAy := 0.0;
    for l:=1 to 5 do
      begin
       temp:=A[l]*ypeak[l];
       SumTemp:=SumTemp+ temp;
       SumAy:=SumAy+A[l];
      end;
  Defuzz1:=SumTemp/SumAy;
end;

function predict(poss1 , poss2 : possvec ; Rel : fuzzrel)  : extended;
begin
  for l := 1 to 5 do
    CapYhat[l] := 0.0;
  for t:=1 to (Eind-1) do
    begin
      s1 := NN[1,t];
      if poss1[s1] > 0 then
       begin
        s2 := NN[2,t];
         l := NN[3,t];
         if poss2[s2] > 0 then
           begin
             CapYhat[l] := CapYhat[l] + poss1[s1] *poss2[s2]*Rel[s1, s2, l];
{            pp[k]^[l] := pp[k]^[l] + pp[k-1]^[s1]*poss2[s2]*Rel[s1,s2,l];}
           end; {s2,l}
       end; {s1}
      pp[l]^[l]:=CapYhat[l];
     end;{t}
  predict := DeFuzz1(CapYhat);    {predicted value}
end; {predict}

procedure verwasig;
var inkrx1,inkrx2,inkry,W: extended;
      j,k:integer;
begin

tau1:=1;tau2:=4;
    if tau1 > tau2 then maxtau:=tau1 else maxtau := tau2;
    maxtau:=maxtau+1;
writeln('Begin');
 for k:=1 to N do
```

```
begin
  p1[k]:=Initpossvec;
  new(Initpossvec);
  p2[k]:=Initpossvec;
  new(Initpossvec);
  p[k]:=Initpossvec;
  new(Initpossvec);
  pp[k]:=Initpossvec;
  new(Initpossvec);
  posv[k]:=Initpossvec;
  new(Initpossvec);
  posv1[k]:=Initpossvec;
  new(Initpossvec);
  posv2[k]:=Initpossvec;
  new(Initpossvec);
end;

  x1min := 10000; x1max := -10000; ymin := x1min; ymax := x1max;
  x2min := 10000; x2max := -10000;
  Reset(infil1);
  Reset(infil2);

  ymaks:=0.0;
  umaks:=0.0;
  for k := 1 to N do
     begin
       Readln(infil1, u[k]);   {read variables}
       Readln(infil2, y[k]);
       ymaks:=max(ymaks,abs(y[k]));   {determine maximum values}
       umaks:=max(umaks,abs(u[k]));
     end;

  for k:=1 to N do
     begin
       y[k]:=y[k]/ymaks;
       u[k]:=u[k]/umaks;
         if k >= maxtau then
           begin
            x1min := min(x1min, y[k-tau1]); x1max := max(x1max, y[k-tau1]);
            x2min := min(x2min, u[k-tau2]); x2max := max(x2max, u[k-tau2]);
             ymin := min(ymin, y[k]);        ymax := max(ymax, y[k]);
           end;
     end;

 inkrx1:=(x1max-x1min)/255;
 inkrx2:=(x2max-x2min)/255;
 inkry:=(ymax-ymin)/255;
  for k:=0 to 255 do
    begin
      waardex1[k] := x1min + (k*inkrx1); {Inkrementele opsoek indeks.}
      waardex2[k] := x2min + (k*inkrx2);
      waardey[k]  := ymin  + (k*inkry);
    end;

 deltax1 := 0.25 * (x1max - x1min);
 deltax2 := 0.25 * (x2max - x2min);
 deltay  := 0.25 * (ymax  -  ymin);
 x1peak[1] := 0.5*(x1min + x1max) - 2.0*deltax1;
 x2peak[1] := 0.5*(x2min + x2max) - 2.0*deltax2;
  ypeak[1] := 0.5*(ymin + ymax) - 2.0*deltay;

    for k:=2 to 5 do
      begin
        x1peak[k] := x1peak[k-1] + deltax1;
```

```
          x2peak[k] := x2peak[k-1] + deltax2;
           ypeak[k] := ypeak[k-1] + deltay;
        end;

    for k:= 0 to 255 do
      begin
        for j:= 1 to 5 do
         begin
          posv1[k]^[j] := a1(j, waardex1[k]);
          posv2[k]^[j] := a2(j, waardex2[k]);
          posv[k]^[j] := b(j, waardey[k]);
{          writeln('posv1',posv1[k]^[j],' waarde',waardex1[k]);}
        end;
      end;
      writeln('End Verwasig.');
end;

procedure fuz;   {open files}
begin
  Assign(infil1, 'c:\wrk\boxjenk1.dat');  {driving function data file}
  Assign(infil2, 'c:\wrk\boxjenk2.dat');  {output function data file}
  Assign(outfile, 'c:\wrk\yhatreko.asc');
  Rewrite(outfile);
  Assign(lst,'LPT1');                {printer}
  Rewrite(lst);
end;

procedure AAA;
  begin
          {INPUT DATA AND COMPUTE FUNCTION PARAMETERS}

                        {CREATE RELATION R}

    for k:= maxtau to M do   {learning to M}
    begin
      for j := 1 to 5 do
      begin
       tel1 := trunc(((y[k-tau1]-x1min)*255)/(x1max-x1min));
       tel2 := trunc(((u[k-tau2]-x2min)*255)/(x2max-x2min));
       tel  := trunc(((y[k]-ymin)*255)/(ymax-ymin));
        p1[k]^[j] := posv1[tel1]^[j]; {do table clustering for y(k-tau1)}
        p2[k]^[j] := posv2[tel2]^[j]; {do table clustering for u(k-tau2)}
        p[k]^[j]  := posv[tel]^[j];    {do clustering for y(k)}
      end;
    end;

for s:=0 to 125 do
begin
  nn[1,s]:=0;
  nn[2,s]:=0;
  nn[3,s]:=0;
end;

writeln('Timing Starts Now.');
writeln('Please Press Enter.');
readln;
settime(0,0,0,0);

Eind:=0;
    for s1 := 1 to 5 do
    begin
      for s2 := 1 to 5 do
      begin
        for s := 1 to 5 do
```

```
        begin
          R[s1, s2, s] := 0.0;
          freq[s1, s2] := 0.0;
          for k:= maxtau to M do       {learning to M}
          begin
           if (p1[k]^[s1]>0) and (p2[k]^[s2]>0) and (p[k]^[s]>0) then
            begin
              temp := p1[k]^[s1]*p2[k]^[s2];
              freq[s1, s2] := freq[s1, s2] + temp; {frequency of occurrence}
                                                   {of input combinations}
              R[s1, s2, s] := R[s1, s2, s] + temp*p[k]^[s];{subrelation}
            end;
           end; {k}
          if freq[s1, s2] > 0.0 then R[s1, s2, s] := R[s1, s2, s]/freq[s1,s2];
                                                     {weighted average}
          if R[s1,s2,s]> 0.0 then
            begin
              NN[1,Eind]:= s1;
              NN[2,Eind]:= s2;
              NN[3,Eind]:= s;
               Eind:= Eind + 1;
            end;
        end; {s}
      end; {s2}
    end; {s1}
  writeln('R complete');

              {CALCULATE YHAT}
for k:= maxtau to N do
begin
 tel2 := trunc(((u[k-tau2]-x2min)*255)/(x2max-x2min));
 tel1 := trunc(((y[k-tau1]-x1min)*255)/(x1max-x1min));
 for j:=1 to 5 do
  begin
   pp[k]^[j] := 0.0;
   p2[k]^[j] := posv2[tel2]^[j];
   p1[k]^[j] := posv1[tel1]^[j];
  end;
end;
for j:=1 to 5 do        {initialize recursive pp vector,input vector}
begin
 tel2 := trunc(((u[maxtau-tau2]-x2min)*255)/(x2max-x2min));
 tel1 := trunc(((y[maxtau-tau1]-x1min)*255)/(x1max-x1min));
  pp[1]^[j]:= posv1[tel2]^[j];
  p2[maxtau]^[j]:= posv2[tel2]^[j];
end;
      for k:=maxtau to N do
 yhat[k] :=ymaks* predict(pp[1]^, p2[k]^, R);

GetTime(H,minute,sec,sec100);
Time:=sec*100 + sec100;
writeln('Timing ends here!');
writeln('Time=',time/100:2:2,' sec.');
readln;
      for k:=maxtau to N do
       begin
        y[k]:=y[k]*ymaks;
        writeln(yhat[k]:10:6,' ',y[k]:10:6);
       end;

      for k:=maxtau to N do
         writeln(outfile,yhat[k]:10:6);

end; {AAA}
```

```
                {MAIN PROGRAM STARTS HERE: DRAW THE GRAPHS}

BEGIN
ClrScr;
fuz;
verwasig;
    AAA;
    close(infil1);
    close(infil2);       {close all files}
    close(outfile);
END.
```

AANHANGSEL 4.4.1.a: Die stroombaandiagram vir die verwasiger.

Print World: VERWFIN.LCA (3090PC84-100), XACT 4. 0,

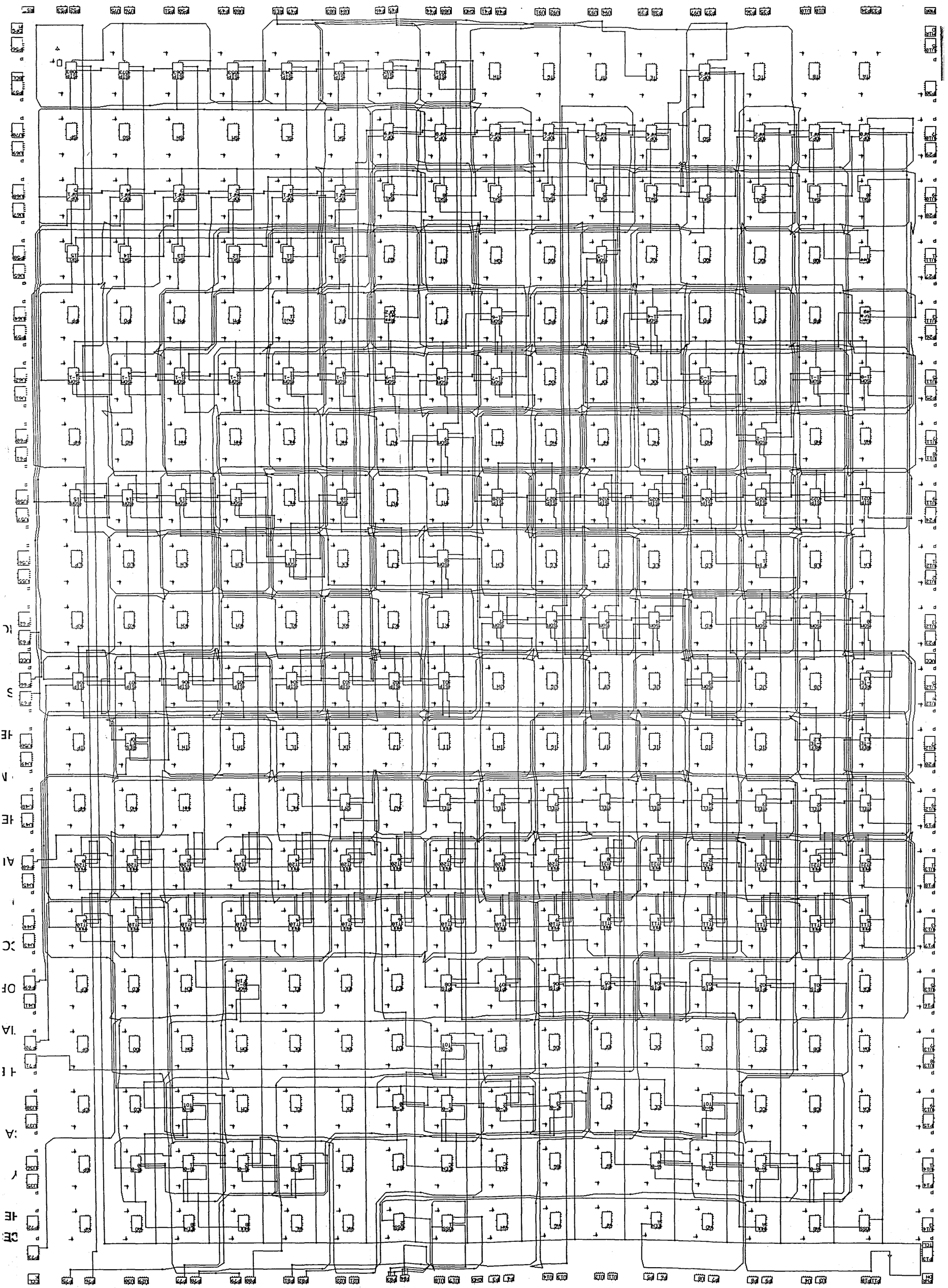
AANHANGSEL 4.4.1.b: Die verbindingslys vir die verwasiger.

```
---- net0 . . . . .    EC.X (MULT7) . .  1.4 ED.C (MULT6)
---- net1 . . . . .    ED.X (MULT6) . .  1.2 EE.C (MULT5)
---- net2 . . . . .    EE.X (MULT5) . .  1.4 EF.C (MULT4)
---- net3 . . . . .    EF.X (MULT4) . .  1.4 EG.C (MULT3)
---- net4 . . . . .    EG.X (MULT3) . .  1.2 EH.C (MULT2)
---- net5 . . . . .    EH.X (MULT2) . .  1.4 EI.C (MULT1)
---- net6 . . . . .    EI.X (MULT1) . .  1.4 EJ.C (MULT0)
---- net7 . . . . .    P59.I (A15). . . 38.4 EC.D (MULT7)
---- net8 . . . . .    P58.I (A14). . . 38.2 ED.D (MULT6)
---- net9 . . . . .    P56.I (A13). . . 42.6 EE.D (MULT5)
---- net10. . . . .    P52.I (A12). . . 38.8 EF.D (MULT4)
---- net11. . . . .    P51.I (A11). . . 39.1 EG.D (MULT3)
---- net12. . . . .    P50.I (A10). . . 35.9 EH.D (MULT2)
---- net13. . . . .    P49.I (A9) . . . 36.3 EI.D (MULT1)
---- net14. . . . .    P48.I (A8) . . . 43.2 EJ.D (MULT0)
---- net15. . . . .    DC.Y (PISO1) . .  1.4 DD.B (PISO2)
---- net16. . . . .    DD.X (PISO2) . .  5.5 EC.A (MULT7)
                                         5.1 ED.A (MULT6)
                                         1.9 EE.A (MULT5)
                                         2.8 EF.A (MULT4)
                                         3.2 EG.A (MULT3)
                                         6.1 EH.A (MULT2)
Press any key to continue.

                                         6.8 EI.A (MULT1)
                                         7.2 EJ.A (MULT0)
---- net17. . . . .    AJ.X . . . . . .  4.3 AJ.C
                                         3.6 AK.K
                                         3.6 P83.O
                                         3.6 P83.T
                                         3.8 P84.O
                                         3.8 P84.T
---- net18. . . . .    AA.X . . . . . . 38.5 P63.O (KLK)
---- net19. . . . .    P83.I. . . . . .  2.2 AJ.E
---- net20. . . . .    P84.I. . . . . .  1.2 AJ.B
---- net21. . . . .    AK.Y . . . . . . 18.9 GCLK.I
---- net22. . . . .    HM.Y (ADD0). . .  3.0 HL.C (ADD1)
---- net23. . . . .    GCLK.O . . . . .  3.2 AA.K
                                         3.3 DC.K (PISO1)
                                         3.3 DD.K (PISO2)
                                         3.3 EC.K (MULT7)
                                         3.3 ED.K (MULT6)
                                         3.2 EE.K (MULT5)
                                         3.2 EF.K (MULT4)
                                         3.2 EG.K (MULT3)
                                         3.2 EH.K (MULT2)
                                         3.2 EI.K (MULT1)
Press any key to continue.

                                         3.2 EJ.K (MULT0)
                                         3.2 EL.K (SIPO4)
                                         3.2 FG.K (SIPO1)
                                         3.2 FH.K (SIPO2)
                                         3.2 FJ.K (SIPO3)
                                         3.2 FM.K (SIPO5)
                                         3.1 FN.K (SIPO6)
```

```
                                                3.2 GA.K (TEL1)
                                                3.1 GB.K (TEL2)
                                                3.3 GC.K (TEL3)
                                                3.3 GD.K (TEL4)
                                                3.2 GE.K (TEL5)
                                                3.2 GF.K (TEL6)
                                                3.2 HI.K (ADD4)
                                                3.2 HJ.K (ADD3)
                                                3.1 HK.K (ADD2)
                                                3.2 HL.K (ADD1)
                                                3.2 HM.K (ADD0)
                                                3.3 IC.K (ADD10)
                                                3.2 IE.K (ADD8)
                                                3.2 IG.K (ADD6)
                                                3.2 IH.K (ADD5)
                                                3.3 JD.K (ADD9)
Press any key to continue.


                                                3.2 JF.K (ADD7)
---- net24. . . . .    EJ.X (MULT0) . .  8.1 FG.A (SIPO1)
                                               18.6 P66.O (SERMULT)
---- net25. . . . .    FJ.Y (SIPO3) . . 28.7 EL.A (SIPO4)
                                               10.8 IG.A (ADD6)
---- net26. . . . .    EL.Y (SIPO4) . .  5.8 FM.A (SIPO5)
                                               20.1 HI.A (ADD4)
---- net27. . . . .    FM.Y (SIPO5) . .  2.7 FN.A (SIPO6)
                                                7.5 HK.A (ADD2)
---- net28. . . . .    FN.Y (SIPO6) . . 10.8 HM.A (ADD0)
---- net29. . . . .    FN.X (SIPO6) . .  7.7 HL.A (ADD1)
---- net30. . . . .    HL.Y (ADD1). . .  3.7 HK.C (ADD2)
---- net31. . . . .    HK.Y (ADD2). . .  5.0 HJ.C (ADD3)
---- net32. . . . .    HJ.Y (ADD3). . .  3.0 HI.C (ADD4)
---- net33. . . . .    FM.X (SIPO5) . . 23.4 HJ.A (ADD3)
---- net34. . . . .    EL.X (SIPO4) . . 16.0 IH.A (ADD5)
---- net35. . . . .    FJ.X (SIPO3) . . 14.1 JF.A (ADD7)
---- net36. . . . .    FG.X (SIPO1) . .  3.4 FH.A (SIPO2)
                                               12.7 IC.A (ADD10)
---- net37. . . . .    FH.Y (SIPO2) . .  8.8 FJ.A (SIPO3)
                                               12.1 IE.A (ADD8)
---- net38. . . . .    FH.X (SIPO2) . . 14.6 JD.A (ADD9)
---- net39. . . . .    P60.I (RESET). .163.3 DC.D (PISO1)
Press any key to continue.


                                              160.1 DD.D (PISO2)
                                              172.3 EC.RD (MULT7)
                                              173.6 ED.RD (MULT6)
                                              158.7 EE.RD (MULT5)
                                              161.0 EF.RD (MULT4)
                                              154.9 EG.RD (MULT3)
                                              147.6 EH.RD (MULT2)
                                              132.6 EI.RD (MULT1)
                                              123.2 EJ.RD (MULT0)
                                              130.5 EL.RD (SIPO4)
                                              157.9 FG.RD (SIPO1)
                                              150.2 FH.RD (SIPO2)
                                              143.6 FJ.RD (SIPO3)
                                              ~61.4 FM.RD (SIPO5)
                                              ~52.6 FN.RD (SIPO6)
```

```
                                        152.9 GA.RD (TEL1)
                                        155.2 GB.RD (TEL2)
                                        157.0 GC.RD (TEL3)
                                        159.2 GD.RD (TEL4)
                                        160.0 GE.RD (TEL5)
                                        164.2 GF.D (TEL6)
                                          7.7 P67.O (RSTUIT)
---- net40. . . . .    HI.Y (ADD4). . .  5.1 IH.C (ADD5)
Press any key to continue.


---- net41. . . . .    IH.Y (ADD5). . .  5.0 IG.C (ADD6)
---- net42. . . . .    IG.Y (ADD6). . .  5.1 JF.C (ADD7)
---- net43. . . . .    JF.Y (ADD7). . .  6.5 IE.C (ADD8)
---- net44. . . . .    IE.Y (ADD8). . .  6.7 JD.C (ADD9)
---- net45. . . . .    JD.Y (ADD9). . .  2.7 IC.C (ADD10)
SL-- net46
---- net47. . . . .    GA.Y (TEL1). . .  1.2 GB.A (TEL2)
---- net48. . . . .    GB.Y (TEL2). . .  1.4 GC.A (TEL3)
---- net49. . . . .    GC.Y (TEL3). . .  1.4 GD.A (TEL4)
---- net50. . . . .    GD.Y (TEL4). . .  2.2 GE.A (TEL5)
---- net51. . . . .    GE.Y (TEL5). . .  1.4 GF.A (TEL6)
---- net52. . . . .    GF.Y (TEL6). . .  1.4 GG.E (TEL7)
---- net53. . . . .    P62.I (REF0) . . 14.5 HL.B (ADD1)
---- net54. . . . .    HM.X (ADD0). . . 16.9 P82.O (D0)
---- net55. . . . .    HL.X (ADD1). . . 14.4 P81.O (D1)
---- net56. . . . .    HK.X (ADD2). . . 15.3 P80.O (D2)
---- net57. . . . .    HJ.X (ADD3). . . 22.8 P79.O (D3)
---- net58. . . . .    HI.X (ADD4). . . 21.6 P78.O (D4)
SL-- net59
SL-- net60
---- net61. . . . .    IH.X (ADD5). . . 27.2 P77.O (D5)
---- net62. . . . .    IG.X (ADD6). . . 29.7 P76.O (D6)
---- net63. . . . .    P61.I (REF1) . . 14.0 HM.B (ADD0)
Press any key to continue.


---- net64. . . . .    JF.X (ADD7). . . 29.7 P75.O (D7)
---- net65. . . . .    GG.X (TEL7). . . 29.5 EL.EC (SIPO4)
                                        15.6 FG.EC (SIPO1)
                                        20.0 FH.EC (SIPO2)
                                        27.0 FJ.EC (SIPO3)
                                        38.9 FM.EC (SIPO5)
                                        40.2 FN.EC (SIPO6)
                                        34.4 HI.RD (ADD4)
                                        27.0 HJ.RD (ADD3)
                                        34.4 HK.RD (ADD2)
                                        34.4 HL.RD (ADD1)
                                        34.4 HM.RD (ADD0)
                                        47.9 IC.RD (ADD10)
                                       ~71.7 IE.RD (ADD8)
                                       ~74.1 IG.RD (ADD6)
                                        20.0 IH.RD (ADD5)
                                       ~64.3 JD.RD (ADD9)
                                       ~71.7 JF.RD (ADD7)
                                        41.6 P69.O (ECADD)
---- net66. . . . .    IE.X (ADD8). . . 33.2 P73.O (D8)
---- net67. . . . .    JD.X (ADD9). . . 25.1 P71.O (D9)
---- net68. . . . .    IC.X (ADD10) . . 30.8 P70.O (D10)
S--- net69. . . . . . . . . . . . . . .  *** P68.O
```

AANHANGSEL 4.4.2.a: Die stroombaandiagram vir die relasieberekening.



AANHANGSEL 4.4.2.b: Die verbindingslys vir die relasieberkening.

---- CLOCK. . . . . GCLK.O . . . . . 3.5 AA.K (OSS)
3.4 EB.K (PISO1)
3.4 EC.K (PISO2)
3.4 ED.K (PISO3)
3.4 EE.K (PISO4)
3.4 EF.K (PISO5)
3.4 EG.K (PISO6)
3.5 EH.K (PISO7)
3.5 EI.K (PISO8)
3.5 FA.K (MULT115)
3.4 FB.K (MULT114)
3.4 FC.K (MULT113)
3.4 FD.K (MULT112)
3.4 FE.K (MULT111)
3.4 FF.K (MULT110)
3.4 FG.K (MULT109)
3.5 FH.K (MULT108)
3.5 FI.K (MULT107)
3.3 FJ.K (MULT106)
3.4 FK.K (MULT105)
3.4 FL.K (MULT104)
3.4 FM.K (MULT103)

Press any key to continue.

3.4 FN.K (MULT102)
3.4 FO.K (MULT101)
3.4 FP.K (MULT100)
3.5 HA.K (TEL1)
3.4 HB.K (TEL2)
3.4 HC.K (TEL3)
3.4 HD.K (TEL4)
3.4 HE.K (TEL5)
3.4 HF.K (TEL6)
3.4 HG.K (TEL7)
3.5 HH.K (TEL8)
3.5 HI.K (TEL9)
3.4 HK.K (2FIN)
3.4 IO.K (RESET-OF)
3.4 JD.K (SOM3)
3.5 JI.K (SIPO1)
3.3 JJ.K (SIPO2)
3.4 JK.K (SIPO3)
3.4 JL.K (SIPO4)
3.4 JM.K (SIPO5)
3.4 JN.K (SIPO6)
3.4 JO.K (SIPO7)
3.4 JP.K (SIPO8)

Press any key to continue.

3.5 KA.K (SOM0)
3.4 KB.K (SOM1)
3.4 KC.K (SOM2)
3.4 KE.K (SOM4)
3.4 KF.K (SOM5)

3.4 KG.K (SOM6)
3.5 KH.K (SOM7)
3.5 KI.K (SOM8)
3.4 KL.K (SOM11)
3.5 MA.K (SIP021)
3.4 MB.K (SIP022)
3.4 MC.K (SIP023)
3.4 MD.K (SIP024)
3.4 ME.K (SIP025)
3.4 MF.K (SIP026)
3.4 MG.K (SIP027)
3.5 MH.K (SIP028)
3.4 MK.K (SOM10)
3.4 MM.K (SOM12)
3.4 MN.K (SOM13)
3.4 MO.K (SOM14)
3.4 MP.K (SOM15)
3.4 NC.K (SOM1-2)

Press any key to continue.

3.5 NI.K (SOM9)
3.5 OA.K (SOM1-0)
3.4 OB.K (SOM1-1)
3.4 OD.K (SOM1-3)
3.5 OH.K (SOM1-7)
3.5 OI.K (SOM1-8)
3.3 OJ.K (SOM1-9)
3.4 OK.K (SOM1-10)
3.4 OL.K (SOM1-11)
3.4 OM.K (SOM1-12)
3.4 ON.K (SOM1-13)
3.4 OO.K (SOM1-14)
3.4 OP.K (SOM1-15)
3.5 PA.K (Vertraag)
3.4 PE.K (SOM1-4)
3.5 PH.K (SOM1-6)
3.5 QA.K (Deel)
3.4 QF.K (SOM1-5)
3.4 RK.K (OP/AF10)
3.4 RL.K (OP/AF11)
3.4 RM.K (OP/AF12)
3.4 RN.K (OP/AF13)
3.4 RO.K (OP/AF14)

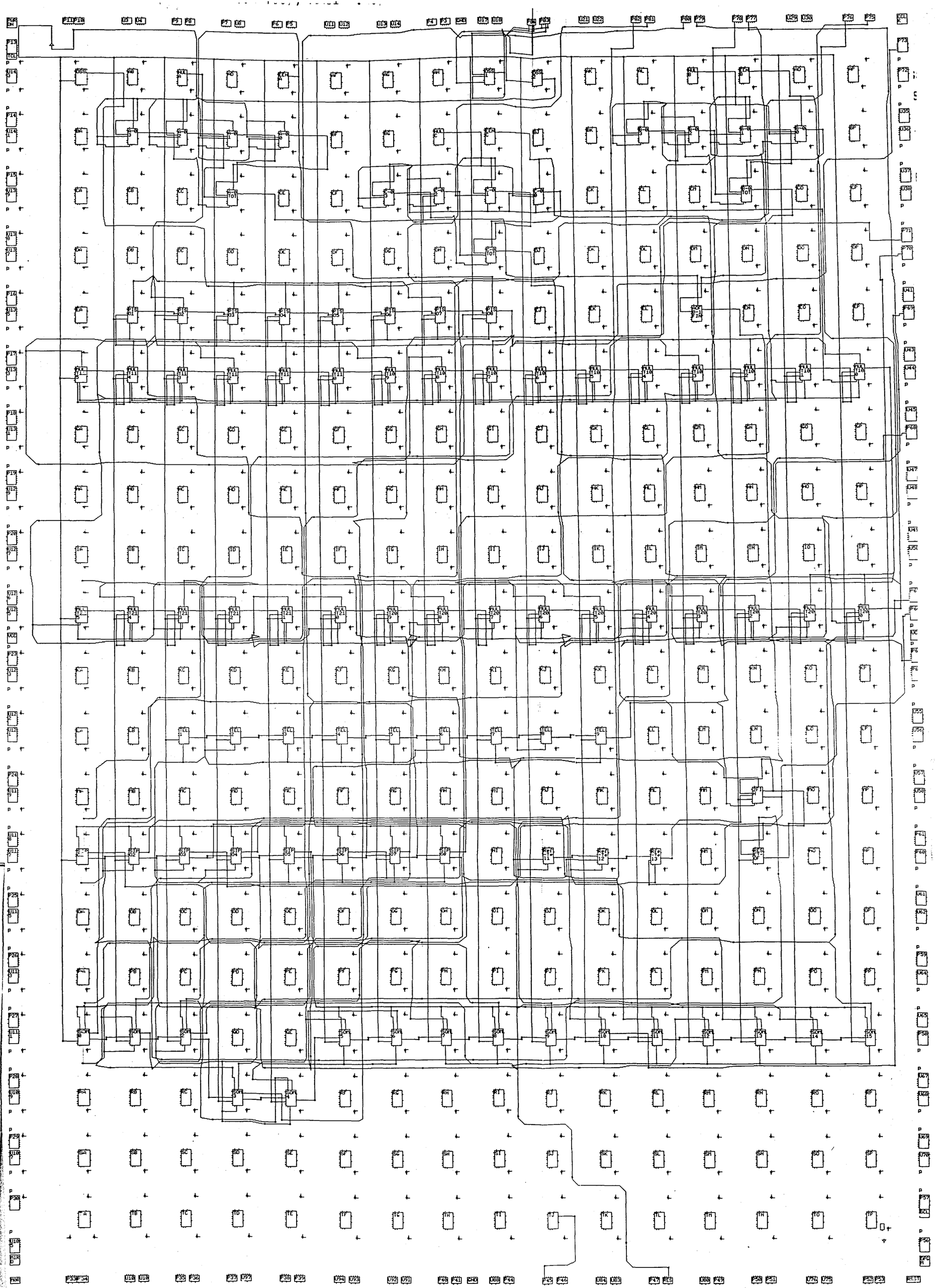
Press any key to continue.

3.4 RP.K (OP/AF15)
3.5 SA.K (OP/AF0)
3.4 SB.K (OP/AF1)
3.4 SC.K (OP/AF2)
3.4 SE.K (OP/AF4)
3.4 SF.K (OP/AF5)
3.4 SG.K (OP/AF6)
3.5 SH.K (OP/AF7)
3.5 SI.K (OP/AF8)
3.3 SJ.K (OP/AF9)
3.4 TD.K (OP/AF3)
3.5 TI.K (SIP031)
3.3 TJ.K (SIP032)

```
                                                3.4 TK.K (SIP033)
                                                3.4 TL.K (SIP034)
                                                3.4 TM.K (SIP035)
                                                3.4 TN.K (SIP036)
                                                3.4 TO.K (SIP037)
                                                3.4 TP.K (SIP038)
---- net0 . . . . .    BE.X (A-00). . .  2.9 CD.A (A-0TOT)
---- net1 . . . . .    BD.X (A-01). . .  2.2 CD.B (A-0TOT)
---- net2 . . . . .    BC.X (A-02). . .  2.0 CD.C (A-0TOT)
---- net3 . . . . .    BB.X (A-03). . .  3.1 CD.D (A-0TOT)
Press any key to continue.


---- net4 . . . . .    CG.X (C-03). . .  2.8 DI.D (C-0TOT)
---- net5 . . . . .    CH.X (C-02). . .  2.3 DI.C (C-0TOT)
---- net6 . . . . .    CI.X (C-01). . .  2.1 DI.B (C-0TOT)
---- net7 . . . . .    CJ.X (C-00). . .  3.5 DI.A (C-0TOT)
---- net8 . . . . .    BL.X (B-03). . .  2.8 CN.D (B-0TOT)
---- net9 . . . . .    BM.X (B-02). . .  2.3 CN.C (B-0TOT)
---- net10. . . . .    BN.X (B-01). . .  2.1 CN.B (B-0TOT)
---- net11. . . . .    BO.X (B-00). . .  3.5 CN.A (B-0TOT)
---- net12. . . . .    EB.X (PIS01) . .  0.0 EC.B (PIS02)
---- net13. . . . .    EC.X (PIS02) . .  0.0 ED.B (PIS03)
---- net14. . . . .    ED.X (PIS03) . .  0.0 EE.B (PIS04)
---- net15. . . . .    EE.X (PIS04) . .  0.0 EF.B (PIS05)
---- net16. . . . .    EF.X (PIS05) . .  0.0 EG.B (PIS06)
---- net17. . . . .    EG.X (PIS06) . .  0.0 EH.B (PIS07)
---- net18. . . . .    EH.X (PIS07) . .  0.0 EI.B (PIS08)
---- net19. . . . .    FA.X (MULT115) .  1.4 FB.C (MULT114)
---- net20. . . . .    FB.X (MULT114) .  1.2 FC.C (MULT113)
---- net30. . . . .    FL.X (MULT104) .  1.4 FM.C (MULT103)
---- net31. . . . .    FM.X (MULT103) .  1.4 FN.C (MULT102)
---- net32. . . . .    FN.X (MULT102) .  1.2 FO.C (MULT101)
---- net33. . . . .    FO.X (MULT101) .  1.4 FP.C (MULT100)
SL-- net34
SL-- net35
Press any key to continue.


SL-- net36
SL-- net37
SL-- net38
SL-- net39
SL-- net40
SL-- net41
SL-- net42
SL-- net43
SL-- net44
SL-- net45
SL-- net46
SL-- net47
SL-- net48
---- net49. . . . .    HA.Y (TEL1). . .  1.4 HB.C (TEL2)
---- net50. . . . .    HB.Y (TEL2). . .  1.4 HC.C (TEL3)
---- net51. . . . .    HC.Y (TEL3). . .  1.4 HD.C (TEL4)
---- net52. . . . .    HD.Y (TEL4). . .  1.4 HE.C (TEL5)
---- net53. . . . .    HE.Y (TEL5). . .  1.4 HF.C (TEL6)
---- net54. . . . .    HF.Y (TEL6). . .  1.4 HG.C (TEL7)
---- net55. . . . .    HG.Y (TEL7). . .  1.4 HH.C (TEL8)
---- net56. . . . .    HH.Y (TEL8). . .  1.4 HI.C (TEL9)
```

```
---- net57. . . . .    JI.Y (SIPO1) . .  1.9 JJ.C (SIPO2)
                                         14.0 MO.A (SOM14)
Press any key to continue.


---- net58. . . . .    JJ.Y (SIPO2) . .  4.5 JK.C (SIPO3)
                                         10.5 MM.A (SOM12)
---- net59. . . . .    JK.Y (SIPO3) . .  2.4 JL.C (SIPO4)
                                          7.7 MK.A (SOM10)
---- net60. . . . .    MA.Y (SIPO21). .  2.6 MB.C (SIPO22)
                                         29.5 OO.A (SOM1-14)
---- net61. . . . .    MB.Y (SIPO22). .  3.4 MC.C (SIPO23)
                                         28.5 OM.A (SOM1-12)
---- net62. . . . .    JN.Y (SIPO6) . .  2.3 JO.C (SIPO7)
                                         24.7 KE.A (SOM4)
---- net63. . . . .    JO.Y (SIPO7) . .  4.9 JP.C (SIPO8)
                                         35.8 KC.A (SOM2)
---- net64. . . . .    EI.X (PISO8) . . 28.7 FA.A (MULT115)
                                         31.6 FB.A (MULT114)
                                         32.4 FC.A (MULT113)
                                         32.8 FD.A (MULT112)
                                         36.7 FE.A (MULT111)
                                         37.4 FF.A (MULT110)
                                         37.7 FG.A (MULT109)
                                         20.0 FH.A (MULT108)
                                          4.7 FI.A (MULT107)
                                          9.4 FJ.A (MULT106)
                                         10.6 FK.A (MULT105)
Press any key to continue.


                                         11.4 FL.A (MULT104)
                                         11.7 FM.A (MULT103)
                                         15.7 FN.A (MULT102)
                                         16.3 FO.A (MULT101)
                                         16.6 FP.A (MULT100)
SL-- net65
SL-- net66
SL-- net67
SL-- net68
SL-- net69
SL-- net70
SL-- net71
SL-- net72
SL-- net73
SL-- net74
SL-- net75
SL-- net76
SL-- net77
SL-- net78
SL-- net79
---- net80. . . . .    MC.Y (SIPO23). .  3.5 MD.C (SIPO24)
                                         21.1 OK.A (SOM1-10)
---- net81. . . . .    FP.X (MULT100) . 31.6 MA.A (SIPO21)
Press any key to continue.


---- net82. . . . .    MD.Y (SIPO24). .  2.0 ME.C (SIPO25)
                                         15.4 OI.A (SOM1-8)
---- net83. . . . .    HK.X (2FIN). . .~95.1 IO.B (RESET-OF)
```

AANHANGSEL 4.4.3.a: Die stroombaandiagram vir die wasige–beramer.



AANHANGSEL 4.4.3.b: Die verbindingslys vir die wasige–beramer.

---- CLOCK. . . . . GCLK.O . . . . . 3.3 AA.K (OSS)
3.3 EB.K (PISO1)
3.4 EC.K (PISO2)
3.4 ED.K (PISO3)
3.4 EE.K (PISO4)
3.4 EF.K (PISO5)
3.4 EG.K (PISO6)
3.4 EH.K (PISO7)
3.3 EI.K (PISO8)
3.3 FA.K (MULT115)
3.3 FB.K (MULT114)
3.4 FC.K (MULT113)
3.4 FD.K (MULT112)
3.4 FE.K (MULT111)
3.4 FF.K (MULT110)
3.4 FG.K (MULT109)
3.4 FH.K (MULT108)
3.3 FI.K (MULT107)
3.3 FJ.K (MULT106)
3.3 FK.K (MULT105)
3.2 FL.K (MULT104)
3.2 FM.K (MULT103)

Press any key to continue.

3.3 FN.K (MULT102)
3.2 FO.K (MULT101)
3.2 FP.K (MULT100)
3.3 JA.K (MULT215)
3.3 JB.K (MULT214)
3.4 JC.K (MULT213)
3.4 JD.K (MULT212)
3.4 JE.K (MULT211)
3.4 JF.K (MULT210)
3.4 JG.K (MULT209)
3.4 JH.K (MULT208)
3.3 JI.K (MULT207)
3.3 JJ.K (MULT206)
3.3 JK.K (MULT205)
3.2 JL.K (MULT204)
3.2 JM.K (MULT203)
3.3 JN.K (MULT202)
3.2 JO.K (MULT201)
3.2 JP.K (MULT200)
3.4 LC.K (TEL1)
3.4 LD.K (TEL2)
3.4 LE.K (TEL3)
3.4 LF.K (TEL4)

Press any key to continue.

3.4 LG.K (TEL5)
3.4 LH.K (TEL6)
3.3 LI.K (TEL7)
3.3 LJ.K (TEL8)
3.3 LK.K (TEL9)

```
                                                3.3 MN.K (2FIN)
                                                3.3 NA.K (SIPO1)
                                                3.3 NB.K (SIPO2)
                                                3.4 NC.K (SIPO3)
                                                3.4 ND.K (SIPO4)
                                                3.4 NE.K (SIPO5)
                                                3.4 NF.K (SIPO6)
                                                3.4 NG.K (SIPO7)
                                                3.4 NH.K (SIPO8)
                                                3.3 NN.K (RESET-OF)
                                                3.3 QA.K (SOM0)
                                                3.3 QB.K (SOM1)
                                                3.4 QC.K (SOM2)
                                                3.4 QF.K (SOM5)
                                                3.4 QG.K (SOM6)
                                                3.4 QH.K (SOM7)
                                                3.3 QI.K (SOM8)
                                                3.3 QJ.K (SOM9)
Press any key to continue.


                                                3.3 QK.K (SOM10)
                                                3.2 QL.K (SOM11)
                                                3.2 QM.K (SOM12)
                                                3.3 QN.K (SOM13)
                                                3.2 QO.K (SOM14)
                                                3.2 QP.K (SOM15)
                                                3.4 RD.K (SOM3)
                                                3.4 RE.K (SOM4)
---- net0 . . . . .    BE.X (A-00). . .  2.9 CD.A (A-0TOT)
---- net1 . . . . .    BD.X (A-01). . .  2.2 CD.B (A-0TOT)
---- net2 . . . . .    BC.X (A-02). . .  2.0 CD.C (A-0TOT)
---- net3 . . . . .    BB.X (A-03). . .  3.1 CD.D (A-0TOT)
---- net4 . . . . .    CG.X (C-03). . .  2.8 DI.D (C-0TOT)
---- net5 . . . . .    CH.X (C-02). . .  2.3 DI.C (C-0TOT)
---- net6 . . . . .    CI.X (C-01). . .  2.1 DI.B (C-0TOT)
---- net7 . . . . .    CJ.X (C-00). . .  3.5 DI.A (C-0TOT)
---- net8 . . . . .    BL.X (B-03). . .  2.8 CN.D (B-0TOT)
---- net9 . . . . .    BM.X (B-02). . .  2.3 CN.C (B-0TOT)
---- net10. . . . .    BN.X (B-01). . .  2.1 CN.B (B-0TOT)
---- net100 . . . .    P84.I. . . . . .  2.2 AI.B (OSS1)
---- net101 . . . .    P83.I. . . . . .  4.7 AI.E (OSS1)
---- net11. . . . .    BO.X (B-00). . .  3.5 CN.A (B-0TOT)
---- net102 . . . .    AJ.Y (OSS2). . . 18.4 GCLK.I
Press any key to continue.


---- net12. . . . .    EB.X (PISO1) . .  0.0 EC.B (PISO2)
---- net103 . . . .    QB.X (SOM1). . . 51.5 P75.O
---- net13. . . . .    EC.X (PISO2) . .  0.0 ED.B (PISO3)
---- net104 . . . .    AE.X (EENA). . . 19.4 BB.A (A-03)
                                         19.4 BB.B (A-03)
                                         22.1 BB.C (A-03)
                                         22.1 BB.D (A-03)
                                         18.4 BC.A (A-02)
                                         18.4 BC.B (A-02)
                                         19.5 BC.C (A-02)
                                         19.5 BC.D (A-02)
                                         18.0 BD.A (A-01)
                                         18.0 BD.B (A-01)
```

```
                                          18.5 BD.C (A-01)
                                          18.5 BD.D (A-01)
                                          19.1 BE.A (A-00)
                                          19.1 BE.B (A-00)
                                          20.6 BE.C (A-00)
                                          20.6 BE.D (A-00)
                                          17.2 EI.A (PISO8)
---- net14. . . . .    ED.X (PISO3) . .   0.0 EE.B (PISO4)
---- net105 . . . .    BH.X (NULC). . .   6.6 CG.A (C-03)
                                           6.9 CG.B (C-03)
Press any key to continue.


                                           7.5 CG.C (C-03)
                                           7.5 CG.D (C-03)
                                           6.7 CH.A (C-02)
                                           6.7 CH.B (C-02)
                                           9.3 CH.C (C-02)
                                           6.7 CH.D (C-02)
                                           2.5 CI.A (C-01)
                                           2.5 CI.B (C-01)
                                           5.2 CI.C (C-01)
                                          13.4 CI.D (C-01)
                                          14.1 CJ.B (C-00)
                                           4.5 CJ.C (C-00)
                                          13.4 CJ.D (C-00)
                                         ~80.5 JA.D (MULT215)
                                         ~79.8 JB.D (MULT214)
                                         ~78.9 JC.D (MULT213)
                                         ~77.6 JD.D (MULT212)
                                          69.2 JE.D (MULT211)
                                          66.1 JF.D (MULT210)
                                          62.5 JG.D (MULT209)
                                          56.9 JH.D (MULT208)
                                          49.2 JI.D (MULT207)
                                          48.5 JJ.D (MULT206)
Press any key to continue.


                                          42.4 JK.D (MULT205)
                                          42.8 JL.D (MULT204)
                                          36.1 JM.D (MULT203)
                                          35.3 JN.D (MULT202)
                                          34.2 JO.D (MULT201)
---- net15. . . . .    EE.X (PISO4) . .   0.0 EF.B (PISO5)
---- net106 . . . .    BI.X (EENC). . .   2.2 CJ.A (C-00)
                                          21.2 JP.D (MULT200)
---- net16. . . . .    EF.X (PISO5) . .   0.0 EG.B (PISO6)
---- net107 . . . .    AN.X (EENB). . .   2.9 BO.A (B-00)
                                           8.9 FP.D (MULT100)
---- net17. . . . .    EG.X (PISO6) . .   0.0 EH.B (PISO7)
---- net108 . . . .    AM.X (NULB). . .   6.6 BL.A (B-03)
                                           6.9 BL.B (B-03)
                                           7.5 BL.C (B-03)
                                           7.5 BL.D (B-03)
                                           6.7 BM.A (B-02)
                                           6.7 BM.B (B-02)
                                           9.3 BM.C (B-02)
                                           6.7 BM.D (B-02)
                                           2.5 BN.A (B-01)
```

```
                                                2.5 BN.B (B-01)
                                                8.0 BN.C (B-01)
Press any key to continue.


                                                8.0 BN.D (B-01)
                                                4.5 BO.B (B-00)
                                                4.5 BO.C (B-00)
                                               11.2 BO.D (B-00)
                                              ~62.0 FA.D (MULT115)
                                              ~61.3 FB.D (MULT114)
                                              ~60.5 FC.D (MULT113)
                                              ~59.3 FD.D (MULT112)
                                              ~57.8 FE.D (MULT111)
                                              ~56.0 FF.D (MULT110)
                                              ~53.9 FG.D (MULT109)
                                              ~51.4 FH.D (MULT108)
                                              ~48.5 FI.D (MULT107)
                                              ~45.3 FJ.D (MULT106)
                                              ~41.7 FK.D (MULT105)
                                               37.6 FL.D (MULT104)
                                               32.9 FM.D (MULT103)
                                               29.6 FN.D (MULT102)
                                               23.5 FO.D (MULT101)
---- net18. . . . .    EH.X (PIS07) . .        0.0 EI.B (PIS08)
---- net20. . . . .    FB.X (MULT114) .        1.2 FC.C (MULT113)
---- net21. . . . .    FC.X (MULT113) .        1.4 FD.C (MULT112)
---- net22. . . . .    FD.X (MULT112) .        1.4 FE.C (MULT111)
Press any key to continue.


---- net23. . . . .    FE.X (MULT111) .        1.2 FF.C (MULT110)
---- net24. . . . .    FF.X (MULT110) .        1.4 FG.C (MULT109)
---- net25. . . . .    FG.X (MULT109) .        1.4 FH.C (MULT108)
---- net26. . . . .    FH.X (MULT108) .        1.2 FI.C (MULT107)
---- net27. . . . .    FI.X (MULT107) .        1.4 FJ.C (MULT106)
---- net28. . . . .    FJ.X (MULT106) .        1.4 FK.C (MULT105)
---- net29. . . . .    FK.X (MULT105) .        1.2 FL.C (MULT104)
---- net30. . . . .    FL.X (MULT104) .        1.4 FM.C (MULT103)
---- net31. . . . .    FM.X (MULT103) .        1.4 FN.C (MULT102)
---- net32. . . . .    FN.X (MULT102) .        1.2 FO.C (MULT101)
---- net33. . . . .    FO.X (MULT101) .        1.4 FP.C (MULT100)
---- net34. . . . .    JA.X (MULT215) .        1.4 JB.C (MULT214)
---- net35. . . . .    JB.X (MULT214) .        1.4 JC.C (MULT213)
---- net36. . . . .    JC.X (MULT213) .        1.2 JD.C (MULT212)
---- net37. . . . .    JD.X (MULT212) .        1.4 JE.C (MULT211)
---- net38. . . . .    JE.X (MULT211) .        1.4 JF.C (MULT210)
---- net39. . . . .    JF.X (MULT210) .        1.2 JG.C (MULT209)
---- net40. . . . .    JG.X (MULT209) .        1.4 JH.C (MULT208)
---- net41. . . . .    JH.X (MULT208) .        1.4 JI.C (MULT207)
---- net42. . . . .    JI.X (MULT207) .        1.2 JJ.C (MULT206)
---- net43. . . . .    JJ.X (MULT206) .        1.4 JK.C (MULT205)
---- net44. . . . .    JK.X (MULT205) .        1.4 JL.C (MULT204)
---- net45. . . . .    JL.X (MULT204) .        1.2 JM.C (MULT203)
Press any key to continue.


---- net46. . . . .    JM.X (MULT203) .        1.4 JN.C (MULT202)
---- net47. . . . .    JN.X (MULT202) .        1.4 JO.C (MULT201)
---- net48. . . . .    JO.X (MULT201) .        1.2 JP.C (MULT200)
```

```
---- net49. . . . .    LC.Y (TEL1). . .  1.4 LD.C (TEL2)
---- net50. . . . .    LD.Y (TEL2). . .  1.4 LE.C (TEL3)
---- net51. . . . .    LE.Y (TEL3). . .  1.4 LF.C (TEL4)
---- net52. . . . .    LF.Y (TEL4). . .  1.4 LG.C (TEL5)
---- net53. . . . .    LG.Y (TEL5). . .  1.4 LH.C (TEL6)
---- net54. . . . .    LH.Y (TEL6). . .  1.4 LI.C (TEL7)
---- net55. . . . .    LI.Y (TEL7). . .  1.4 LJ.C (TEL8)
---- net56. . . . .    LJ.Y (TEL8). . .  1.4 LK.C (TEL9)
---- net57. . . . .    NA.Y (SIPO1) . .  3.1 NB.C (SIPO2)
                                         29.4 QO.A (SOM14)
---- net58. . . . .    NB.Y (SIPO2) . .  2.1 NC.C (SIPO3)
                                         19.2 QM.A (SOM12)
---- net59. . . . .    NC.Y (SIPO3) . .  2.2 ND.C (SIPO4)
                                         16.6 QK.A (SOM10)
---- net60. . . . .    ND.Y (SIPO4) . .  3.5 NE.C (SIPO5)
                                         16.6 QI.A (SOM8)
---- net61. . . . .    NE.Y (SIPO5) . .  2.0 NF.C (SIPO6)
                                         13.3 QG.A (SOM6)
---- net62. . . . .    NF.Y (SIPO6) . .  2.9 NG.C (SIPO7)
                                         32.5 P78.0
Press any key to continue.


---- net63. . . . .    NG.Y (SIPO7) . .  3.5 NH.C (SIPO8)
                                         41.0 P80.0
                                         22.8 QC.A (SOM2)
---- net64. . . . .    EI.X (PISO8) . . 28.7 FA.A (MULT115)
                                         31.6 FB.A (MULT114)
                                         32.4 FC.A (MULT113)
                                         32.8 FD.A (MULT112)
                                         36.7 FE.A (MULT111)
                                         37.4 FF.A (MULT110)
                                         37.7 FG.A (MULT109)
                                         20.0 FH.A (MULT108)
                                          4.7 FI.A (MULT107)
                                          9.4 FJ.A (MULT106)
                                         10.6 FK.A (MULT105)
                                         11.4 FL.A (MULT104)
                                         11.7 FM.A (MULT103)
                                         15.7 FN.A (MULT102)
                                         16.3 FO.A (MULT101)
                                         16.6 FP.A (MULT100)
---- net65. . . . .    QA.Y (SOM0). . .  1.4 QB.C (SOM1)
---- net66. . . . .    QB.Y (SOM1). . .  1.4 QC.C (SOM2)
---- net67. . . . .    QC.Y (SOM2). . .  2.0 RD.C (SOM3)
---- net68. . . . .    RD.Y (SOM3). . .  1.4 RE.C (SOM4)
Press any key to continue.


---- net69. . . . .    RE.Y (SOM4). . .  2.0 QF.C (SOM5)
---- net70. . . . .    QF.Y (SOM5). . .  1.4 QG.C (SOM6)
---- net71. . . . .    QG.Y (SOM6). . .  1.4 QH.C (SOM7)
---- net72. . . . .    QH.Y (SOM7). . .  1.4 QI.C (SOM8)
---- net73. . . . .    QI.Y (SOM8). . .  1.4 QJ.C (SOM9)
---- net74. . . . .    QJ.Y (SOM9). . .  1.4 QK.C (SOM10)
---- net75. . . . .    QK.Y (SOM10) . .  1.4 QL.C (SOM11)
---- net76. . . . .    QL.Y (SOM11) . .  1.4 QM.C (SOM12)
---- net77. . . . .    QM.Y (SOM12) . .  1.4 QN.C (SOM13)
---- net78. . . . .    QN.Y (SOM13) . .  1.4 QO.C (SOM14)
---- net79. . . . .    QO.Y (SOM14) . .  1.4 QP.C (SOM15)
```

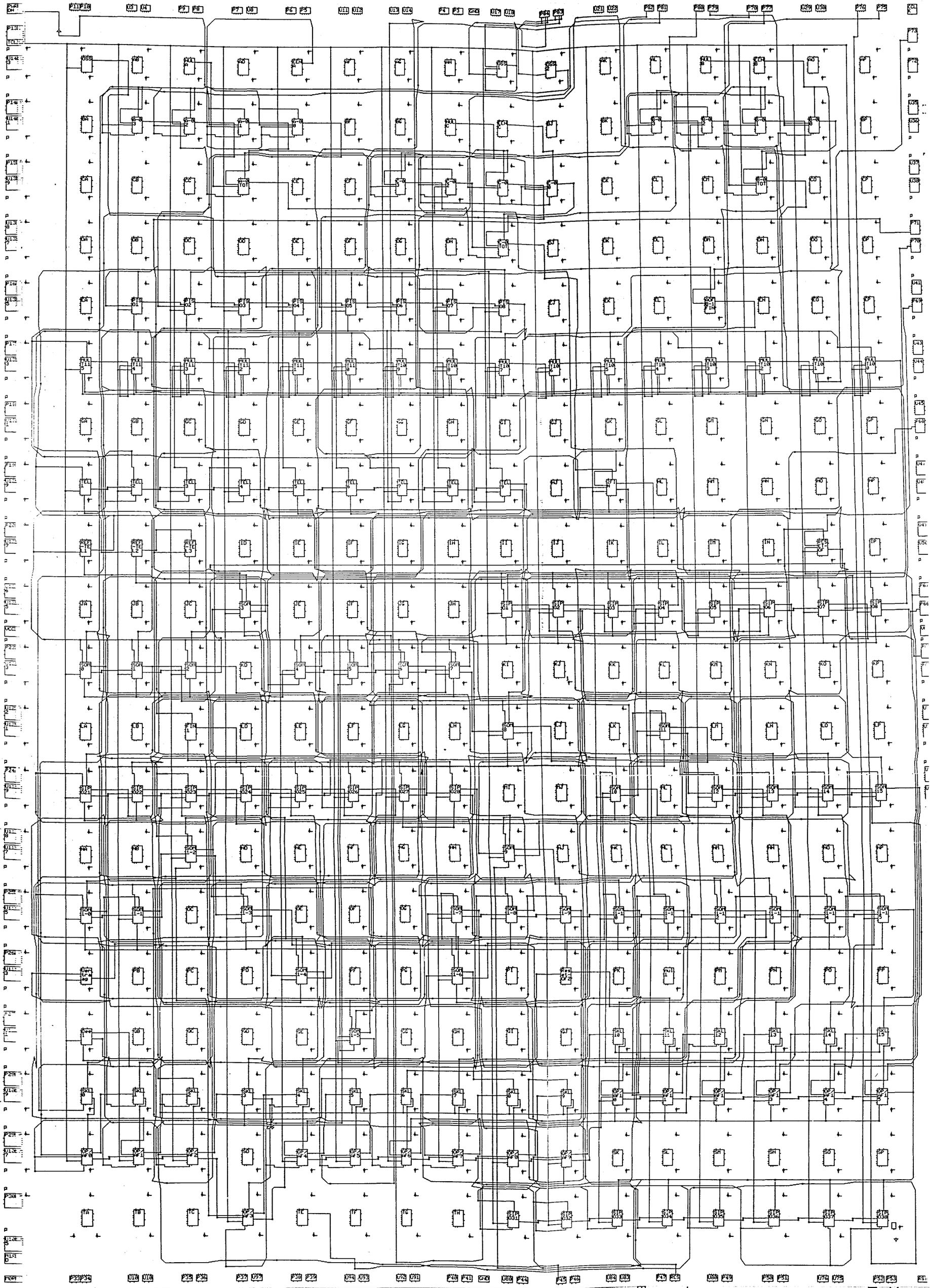
```
---- net80. . . . .    P48.I (RES). . . 25.3 NJ.RD (Ref-Tel1)
                                         29.8 NK.RD (Ref-Tel2)
                                         31.1 NL.RD (Rfe-Tel3)
                                         34.3 NN.A (RESET-OF)
                                         35.4 QA.RD (SOM0)
                                         34.8 QB.RD (SOM1)
                                         34.3 QC.RD (SOM2)
                                         39.6 QF.RD (SOM5)
                                         42.1 QG.RD (SOM6)
                                         17.8 QH.RD (SOM7)
                                         16.7 QI.RD (SOM8)
                                         20.4 QJ.RD (SOM9)
Press any key to continue.

                                         21.4 QK.RD (SOM10)
                                         22.0 QL.RD (SOM11)
                                         25.1 QM.RD (SOM12)
                                         26.1 QN.RD (SOM13)
                                         26.7 QO.RD (SOM14)
                                         29.1 QP.RD (SOM15)
                                         41.9 RD.RD (SOM3)
                                         45.3 RE.RD (SOM4)
---- net81. . . . .    FP.X (MULT100) . 39.7 JA.A (MULT215)
                                         39.5 JB.A (MULT214)
                                         37.4 JC.A (MULT213)
                                         38.7 JD.A (MULT212)
                                         44.2 JE.A (MULT211)
                                         44.6 JF.A (MULT210)
                                         46.5 JG.A (MULT209)
                                         47.0 JH.A (MULT208)
                                         25.5 JI.A (MULT207)
                                         26.2 JJ.A (MULT206)
                                         26.8 JK.A (MULT205)
                                         19.1 JL.A (MULT204)
                                         19.8 JM.A (MULT203)
                                         20.4 JN.A (MULT202)
                                          9.9 JO.A (MULT201)
Press any key to continue.

                                          7.3 JP.A (MULT200)
---- net82. . . . .    MN.Y (2FIN). . .  9.4 P66.O
                                         33.2 QA.EC (SOM0)
                                         33.5 QB.EC (SOM1)
                                         29.0 QC.EC (SOM2)
                                         20.3 QF.EC (SOM5)
                                         23.0 QG.EC (SOM6)
                                         23.8 QH.EC (SOM7)
                                         24.3 QI.EC (SOM8)
                                         27.0 QJ.EC (SOM9)
                                         27.8 QK.EC (SOM10)
                                         28.2 QL.EC (SOM11)
                                         31.0 QM.EC (SOM12)
                                         31.8 QN.EC (SOM13)
                                         32.2 QO.EC (SOM14)
                                         34.4 QP.EC (SOM15)
                                         38.1 RD.EC (SOM3)
                                         36.0 RE.EC (SOM4)
---- net83. . . . .    MN.X (2FIN). . . 26.1 NA.EC (SIPO1)
```

```
                                          25.9 NB.EC (SIPO2)
                                          25.3 NC.EC (SIPO3)
                                          21.9 ND.EC (SIPO4)
                                          21.6 NE.EC (SIPO5)
Press any key to continue.


                                          21.5 NF.EC (SIPO6)
                                          24.1 NG.EC (SIPO7)
                                          24.5 NH.EC (SIPO8)
                                          14.9 NJ.K (Ref-Tel1)
                                          16.2 NK.K (Ref-Tel2)
                                           6.2 NL.K (Rfe-Tel3)
                                           4.3 NN.B (RESET-OF)
---- net84. . . . .   CD.X (A-0TOT). . 14.6 EM.A (AOFB-1FIN)
---- net85. . . . .   CN.X (B-0TOT). .  5.7 EM.B (AOFB-1FIN)
---- net86. . . . .   DI.X (C-0TOT). . 18.3 MN.A (2FIN)
---- net87. . . . .   EM.X (AOFB-1FIN) 15.1 MN.B (2FIN)
---- net89. . . . .   AC.X (NULA). . . 16.0 EB.A (PISO1)
                                          17.2 EB.C (PISO1)
                                          15.9 EC.A (PISO2)
                                          16.5 EC.C (PISO2)
                                          19.7 ED.A (PISO3)
                                          18.1 ED.C (PISO3)
                                          20.3 EE.A (PISO4)
                                          21.1 EE.C (PISO4)
                                          23.3 EF.A (PISO5)
                                          21.8 EF.C (PISO5)
                                          24.1 EG.A (PISO6)
                                          22.3 EG.C (PISO6)
Press any key to continue.


                                          24.5 EH.A (PISO7)
                                          24.8 EH.C (PISO7)
                                          25.1 EI.C (PISO8)
SL-- net90
SL-- net91
SL-- net92
---- net93. . . . .   NE.X (SIPO5) . . 10.3 QH.A (SOM7)
---- net94. . . . .   ND.X (SIPO4) . . 20.4 QJ.A (SOM9)
---- net95. . . . .   NC.X (SIPO3) . . 15.3 QL.A (SOM11)
---- net96. . . . .   NB.X (SIPO2) . . 22.3 QN.A (SOM13)
---- net97. . . . .   NA.X (SIPO1) . . 38.6 P67.O
                                          29.4 QP.A (SOM15)
---- net98. . . . .   JP.X (MULT200) . 35.4 NA.A (SIPO1)
                                          11.3 P69.O
---- net99. . . . .   AI.X (OSS1). . .  5.3 AI.C (OSS1)
                                           4.2 AJ.K (OSS2)
                                           3.4 P83.O
                                           3.4 P83.T
                                           3.4 P84.O
                                           3.4 P84.T
---- net109 . . . .   NJ.Y (Ref-Tel1).  1.4 NK.C (Ref-Tel2)
---- net19. . . . .   FA.X (MULT115) .  1.4 FB.C (MULT114)
---- net110 . . . .   NK.Y (Ref-Tel2).  1.4 NL.C (Rfe-Tel3)
Press any key to continue.


---- net111 . . . .   NL.Y (Rfe-Tel3).  3.5 MN.C (2FIN)
```

```
---- net115 . . . .    QA.X (SOM0). . . 46.0 P76.0
---- net116 . . . .    QC.X (SOM2). . . 46.1 P73.0
---- net117 . . . .    RD.X (SOM3). . . 40.7 P71.0
---- net118 . . . .    QK.X (SOM10) . . 13.1 P70.0
SL-- net120
---- net123 . . . .    AA.X (OSS) . . . 39.7 P63.0
---- net124 . . . .    TJ.X . . . . . .  2.2 P45.0
---- net126 . . . .    NH.Y (SIPO8) . . 28.1 P82.0
                                        19.0 QA.A (SOM0)
---- net127 . . . .    NH.X (SIPO8) . . 31.7 P81.0
                                        18.0 QB.A (SOM1)
---- net128 . . . .    NG.X (SIPO7) . . 35.6 P79.0
                                        13.7 RD.A (SOM3)
---- net129 . . . .    NF.X (SIPO6) . . 43.8 P77.0
                                        11.4 QF.A (SOM5)
                                        12.9 RE.A (SOM4)
---- net131 . . . .    LK.Y (TEL9). . .  7.4 MN.D (2FIN)
                                        16.7 P68.0
---- RST. . . . . .    NN.Y (RESET-OF).114.0 EB.D (PISO1)
                                       113.9 EC.D (PISO2)
                                       116.0 ED.D (PISO3)
                                       117.8 EE.D (PISO4)
Press any key to continue.

                                       119.9 EF.D (PISO5)
                                       121.5 EG.D (PISO6)
                                       122.7 EH.D (PISO7)
                                       123.6 EI.D (PISO8)
                                       120.8 FA.RD (MULT115)
                                       122.3 FB.RD (MULT114)
                                       120.8 FC.RD (MULT113)
                                       122.4 FD.RD (MULT112)
                                       126.2 FE.RD (MULT111)
                                       127.2 FF.RD (MULT110)
                                       127.8 FG.RD (MULT109)
                                       130.8 FH.RD (MULT108)
                                       131.9 FI.RD (MULT107)
                                       132.5 FJ.RD (MULT106)
                                       135.5 FK.RD (MULT105)
                                       136.6 FL.RD (MULT104)
                                       137.2 FM.RD (MULT103)
                                       140.2 FN.RD (MULT102)
                                       141.1 FO.RD (MULT101)
                                       141.5 FP.RD (MULT100)
                                       140.2 JA.RD (MULT215)
                                       149.7 JB.RD (MULT214)
                                       157.5 JC.RD (MULT213)
Press any key to continue.

                                       170.9 JD.RD (MULT212)
                                       176.0 JE.RD (MULT211)
                                       177.7 JF.RD (MULT210)
                                       ~91.5 JG.RD (MULT209)
                                       ~85.3 JH.RD (MULT208)
                                       ~82.7 JI.RD (MULT207)
                                       ~76.3 JJ.RD (MULT206)
                                       ~67.2 JK.RD (MULT205)
                                       ~62.0 JL.RD (MULT204)
```

```
~49.9 JM.RD (MULT203)
~35.2 JN.RD (MULT202)
~39.4 JO.RD (MULT201)
~33.9 JP.RD (MULT200)
184.0 LC.RD (TEL1)
180.4 LD.RD (TEL2)
186.6 LE.RD (TEL3)
187.9 LF.RD (TEL4)
188.8 LG.RD (TEL5)
189.3 LH.RD (TEL6)
~99.6 LI.RD (TEL7)
~79.8 LJ.RD (TEL8)
101.6 LK.RD (TEL9)
191.3 NA.RD (SIP01)
Press any key to continue.

~62.0 JL.RD (MULT204)
~49.9 JM.RD (MULT203)
~35.2 JN.RD (MULT202)
~39.4 JO.RD (MULT201)
~33.9 JP.RD (MULT200)
184.0 LC.RD (TEL1)
180.4 LD.RD (TEL2)
186.6 LE.RD (TEL3)
187.9 LF.RD (TEL4)
188.8 LG.RD (TEL5)
189.3 LH.RD (TEL6)
~99.6 LI.RD (TEL7)
~79.8 LJ.RD (TEL8)
101.6 LK.RD (TEL9)
191.3 NA.RD (SIP01)
Press any key to continue.

190.3 NB.RD (SIP02)
189.1 NC.RD (SIP03)
191.7 ND.RD (SIP04)
195.5 NE.RD (SIP05)
196.1 NF.RD (SIP06)
198.8 NG.RD (SIP07)
199.2 NH.RD (SIP08)
Press any key to continue.
```

AANHANGSEL 4.4.4.a: Die stroombaandiagram vir die ontwasiger.



AANHANGSEL 4.4.4.b: Die verbindingslys vir die ontwasiger.

```
---- CLOCK. . . . .    GCLK.O . . . . . 3.5 AA.K (OSS)
                                         3.5 EB.K (PISO1)
                                         3.5 EC.K (PISO2)
                                         3.5 ED.K (PISO3)
                                         3.5 EE.K (PISO4)
                                         3.5 EF.K (PISO5)
                                         3.4 EG.K (PISO6)
                                         3.5 EH.K (PISO7)
                                         3.5 EI.K (PISO8)
                                         3.5 FA.K (MULT115)
                                         3.5 FB.K (MULT114)
                                         3.5 FC.K (MULT113)
                                         3.5 FD.K (MULT112)
                                         3.5 FE.K (MULT111)
                                         3.5 FF.K (MULT110)
                                         3.4 FG.K (MULT109)
                                         3.5 FH.K (MULT108)
                                         3.5 FI.K (MULT107)
                                         3.4 FJ.K (MULT106)
                                         3.5 FK.K (MULT105)
                                         3.4 FL.K (MULT104)
                                         3.4 FM.K (MULT103)
Press any key to continue.

                                         3.4 FN.K (MULT102)
                                         3.5 FO.K (MULT101)
                                         3.4 FP.K (MULT100)
                                         3.5 GA.K (MULT215)
                                         3.5 GB.K (MULT214)
                                         3.5 GC.K (MULT213)
                                         3.5 GD.K (MULT212)
                                         3.5 GE.K (MULT211)
                                         3.5 GF.K (MULT210)
                                         3.4 GG.K (MULT209)
                                         3.5 GH.K (MULT208)
                                         3.5 GI.K (MULT207)
                                         3.4 GJ.K (MULT206)
                                         3.5 GK.K (MULT205)
                                         3.4 GL.K (MULT204)
                                         3.4 GM.K (MULT203)
                                         3.4 GN.K (MULT202)
                                         3.5 GO.K (MULT201)
                                         3.4 GP.K (MULT200)
                                         3.5 HA.K (TEL1)
                                         3.5 HB.K (TEL2)
                                         3.5 HC.K (TEL3)
                                         3.5 HD.K (TEL4)
Press any key to continue.

                                         3.5 HE.K (TEL5)
                                         3.5 HF.K (TEL6)
                                         3.4 HG.K (TEL7)
                                         3.5 HH.K (TEL8)
                                         3.5 HI.K (TEL9)
                                         3.5 HK.K (2FIN)
                                         3.5 IO.K (RESET-OF)
```

3.5 JD.K (SOM3)
3.5 JI.K (SIPO1)
3.4 JJ.K (SIPO2)
3.5 JK.K (SIPO3)
3.4 JL.K (SIPO4)
3.4 JM.K (SIPO5)
3.4 JN.K (SIPO6)
3.5 JO.K (SIPO7)
3.4 JP.K (SIPO8)
3.5 KA.K (SOM0)
3.5 KB.K (SOM1)
3.5 KC.K (SOM2)
3.5 KE.K (SOM4)
3.5 KF.K (SOM5)
3.4 KG.K (SOM6)
3.5 KH.K (SOM7)

Press any key to continue.

3.5 LI.K (SOM8)
3.4 LL.K (SOM11)
3.5 MA.K (SIPO21)
3.5 MB.K (SIPO22)
3.5 MC.K (SIPO23)
3.5 MD.K (SIPO24)
3.5 ME.K (SIPO25)
3.5 MF.K (SIPO26)
3.4 MG.K (SIPO27)
3.5 MH.K (SIPO28)
3.5 MK.K (SOM10)
3.4 MM.K (SOM12)
3.4 MN.K (SOM13)
3.5 MO.K (SOM14)
3.4 MP.K (SOM15)
3.5 NC.K (SOM1-2)
3.5 NI.K (SOM9)
3.5 OA.K (SOM1-0)
3.5 OB.K (SOM1-1)
3.5 OD.K (SOM1-3)
3.5 OH.K (SOM1-7)
3.5 OI.K (SOM1-8)
3.4 OJ.K (SOM1-9)

Press any key to continue.

3.5 OK.K (SOM1-10)
3.4 OL.K (SOM1-11)
3.4 OM.K (SOM1-12)
3.4 ON.K (SOM1-13)
3.5 OO.K (SOM1-14)
3.4 OP.K (SOM1-15)
3.5 PA.K (Vertraag)
3.5 PE.K (SOM1-4)
3.5 PH.K (SOM1-6)
3.5 QA.K (Deel)
3.5 QF.K (SOM1-5)
3.5 RK.K (OP/AF10)
3.4 RL.K (OP/AF11)
3.4 RM.K (OP/AF12)
3.4 RN.K (OP/AF13)

```
                                                    3.5 RO.K (OP/AF14)
                                                    3.4 RP.K (OP/AF15)
                                                    3.5 SA.K (OP/AF0)
                                                    3.5 SB.K (OP/AF1)
                                                    3.5 SC.K (OP/AF2)
                                                    3.5 SE.K (OP/AF4)
                                                    3.5 SF.K (OP/AF5)
                                                    3.4 SG.K (OP/AF6)
Press any key to continue.


                                                    3.5 SH.K (OP/AF7)
                                                    3.5 SI.K (OP/AF8)
                                                    3.4 SJ.K (OP/AF9)
                                                    3.5 TD.K (OP/AF3)
                                                    3.5 TI.K (SIPO31)
                                                    3.4 TJ.K (SIPO32)
                                                    3.5 TK.K (SIPO33)
                                                    3.4 TL.K (SIPO43)
                                                    3.4 TM.K (SIPO53)
                                                    3.4 TN.K (SIPO63)
                                                    3.5 TO.K (SIPO73)
                                                    3.4 TP.K (SIPO83)
---- net0 . . . . .    BE.X (A-00). . .  2.9 CD.A (A-0TOT)
---- net1 . . . . .    BD.X (A-01). . .  2.2 CD.B (A-0TOT)
---- net2 . . . . .    BC.X (A-02). . .  2.0 CD.C (A-0TOT)
---- net3 . . . . .    BB.X (A-03). . .  3.1 CD.D (A-0TOT)
---- net4 . . . . .    CG.X (C-03). . .  2.8 DI.D (C-0TOT)
---- net5 . . . . .    CH.X (C-02). . .  2.3 DI.C (C-0TOT)
---- net6 . . . . .    CI.X (C-01). . .  2.1 DI.B (C-0TOT)
---- net7 . . . . .    CJ.X (C-00). . .  3.5 DI.A (C-0TOT)
---- net8 . . . . .    BL.X (B-03). . .  2.8 CN.D (B-0TOT)
---- net9 . . . . .    BM.X (B-02). . .  2.3 CN.C (B-0TOT)
---- net10. . . . .    BN.X (B-01). . .  2.1 CN.B (B-0TOT)
Press any key to continue.


---- net11. . . . .    BO.X (B-00). . .  3.5 CN.A (B-0TOT)
---- net12. . . . .    EB.X (PISO1) . .  0.0 EC.B (PISO2)
---- net13. . . . .    EC.X (PISO2) . .  0.0 ED.B (PISO3)
---- net14. . . . .    ED.X (PISO3) . .  0.0 EE.B (PISO4)
---- net15. . . . .    EE.X (PISO4) . .  0.0 EF.B (PISO5)
---- net16. . . . .    EF.X (PISO5) . .  0.0 EG.B (PISO6)
---- net17. . . . .    EG.X (PISO6) . .  0.0 EH.B (PISO7)
---- net18. . . . .    EH.X (PISO7) . .  0.0 EI.B (PISO8)
---- net19. . . . .    FA.X (MULT115) .  1.4 FB.C (MULT114)
---- net20. . . . .    FB.X (MULT114) .  1.2 FC.C (MULT113)
---- net21. . . . .    FC.X (MULT113) .  1.4 FD.C (MULT112)
---- net22. . . . .    FD.X (MULT112) .  1.4 FE.C (MULT111)
---- net23. . . . .    FE.X (MULT111) .  1.2 FF.C (MULT110)
---- net24. . . . .    FF.X (MULT110) .  1.4 FG.C (MULT109)
---- net25. . . . .    FG.X (MULT109) .  1.4 FH.C (MULT108)
---- net26. . . . .    FH.X (MULT108) .  1.2 FI.C (MULT107)
---- net27. . . . .    FI.X (MULT107) .  1.4 FJ.C (MULT106)
---- net28. . . . .    FJ.X (MULT106) .  1.4 FK.C (MULT105)
---- net29. . . . .    FK.X (MULT105) .  1.2 FL.C (MULT104)
---- net30. . . . .    FL.X (MULT104) .  1.4 FM.C (MULT103)
---- net31. . . . .    FM.X (MULT103) .  1.4 FN.C (MULT102)
---- net32. . . . .    FN.X (MULT102) .  1.2 FO.C (MULT101)
---- net33. . . . .    FO.X (MULT101) .  1.4 FP.C (MULT100)
```

Press any key to continue.

```
---- net34. . . . .    GA.X (MULT215) .  1.4 GB.C (MULT214)
---- net35. . . . .    GB.X (MULT214) .  1.4 GC.C (MULT213)
---- net36. . . . .    GC.X (MULT213) .  1.5 GD.C (MULT212)
---- net37. . . . .    GD.X (MULT212) .  1.3 GE.C (MULT211)
---- net38. . . . .    GE.X (MULT211) .  1.4 GF.C (MULT210)
---- net39. . . . .    GF.X (MULT210) .  1.5 GG.C (MULT209)
---- net40. . . . .    GG.X (MULT209) .  1.5 GH.C (MULT208)
---- net41. . . . .    GH.X (MULT208) .  1.4 GI.C (MULT207)
---- net42. . . . .    GI.X (MULT207) .  1.2 GJ.C (MULT206)
---- net43. . . . .    GJ.X (MULT206) .  1.4 GK.C (MULT205)
---- net44. . . . .    GK.X (MULT205) .  1.4 GL.C (MULT204)
---- net45. . . . .    GL.X (MULT204) .  1.2 GM.C (MULT203)
---- net46. . . . .    GM.X (MULT203) .  1.4 GN.C (MULT202)
---- net47. . . . .    GN.X (MULT202) .  1.4 GO.C (MULT201)
---- net48. . . . .    GO.X (MULT201) .  1.4 GP.C (MULT200)
---- net49. . . . .    HA.Y (TEL1). . .  1.4 HB.C (TEL2)
---- net50. . . . .    HB.Y (TEL2). . .  1.4 HC.C (TEL3)
---- net51. . . . .    HC.Y (TEL3). . .  1.4 HD.C (TEL4)
---- net52. . . . .    HD.Y (TEL4). . .  1.4 HE.C (TEL5)
---- net53. . . . .    HE.Y (TEL5). . .  1.4 HF.C (TEL6)
---- net54. . . . .    HF.Y (TEL6). . .  1.4 HG.C (TEL7)
---- net55. . . . .    HG.Y (TEL7). . .  1.4 HH.C (TEL8)
---- net56. . . . .    HH.Y (TEL8). . .  1.4 HI.C (TEL9)
```

Press any key to continue.

```
---- net57. . . . .    JI.Y (SIPO1) . .  1.9 JJ.C (SIPO2)
                                         14.0 MO.A (SOM14)
---- net58. . . . .    JJ.Y (SIPO2) . .  4.5 JK.C (SIPO3)
                                         10.5 MM.A (SOM12)
---- net59. . . . .    JK.Y (SIPO3) . .  2.4 JL.C (SIPO4)
                                          7.7 MK.A (SOM10)
---- net60. . . . .    MA.Y (SIPO21). .  2.6 MB.C (SIPO22)
                                         29.5 OO.A (SOM1-14)
---- net61. . . . .    MB.Y (SIPO22). .  3.4 MC.C (SIPO23)
                                         28.5 OM.A (SOM1-12)
---- net62. . . . .    JN.Y (SIPO6) . .  2.3 JO.C (SIPO7)
                                         24.7 KE.A (SOM4)
---- net63. . . . .    JO.Y (SIPO7) . .  4.9 JP.C (SIPO8)
                                         35.8 KC.A (SOM2)
---- net64. . . . .    EI.X (PISO8) . . 28.7 FA.A (MULT115)
                                         31.6 FB.A (MULT114)
                                         32.4 FC.A (MULT113)
                                         32.8 FD.A (MULT112)
                                         36.7 FE.A (MULT111)
                                         37.4 FF.A (MULT110)
                                         37.7 FG.A (MULT109)
                                         20.0 FH.A (MULT108)
                                          4.7 FI.A (MULT107)
```

Press any key to continue.

```
                                          9.4 FJ.A (MULT106)
                                         10.6 FK.A (MULT105)
                                         11.4 FL.A (MULT104)
                                         11.7 FM.A (MULT103)
                                         15.7 FN.A (MULT102)
```

```
                                          16.3 FO.A (MULT101)
                                          16.6 FP.A (MULT100)
SL-- net65
SL-- net66
SL-- net67
SL-- net68
SL-- net69
SL-- net70
SL-- net71
SL-- net72
SL-- net73
SL-- net74
SL-- net75
SL-- net76
SL-- net77
SL-- net78
SL-- net79
---- net80. . . . .    MC.Y (SIPO23). .  3.5 MD.C (SIPO24)
Press any key to continue.


                                          21.1 OK.A (SOM1-10)
---- net81. . . . .    FP.X (MULT100) . 34.2 GA.A (MULT215)
                                          39.5 GB.A (MULT214)
                                          46.9 GC.A (MULT213)
                                          52.9 GD.A (MULT212)
                                          54.8 GE.A (MULT211)
                                          57.1 GF.A (MULT210)
                                          60.2 GG.A (MULT209)
                                          61.8 GH.A (MULT208)
                                          69.1 GI.A (MULT207)
                                          69.1 GJ.A (MULT206)
                                          69.1 GK.A (MULT205)
                                          69.1 GL.A (MULT204)
                                          69.7 GM.A (MULT203)
                                          69.1 GN.A (MULT202)
                                          69.1 GO.A (MULT201)
                                          69.1 GP.A (MULT200)
                                          32.5 MA.A (SIPO21)
---- net82. . . . .    MD.Y (SIPO24). .  2.0 ME.C (SIPO25)
                                          15.4 OI.A (SOM1-8)
---- net83. . . . .    HK.X (2FIN). . .~95.1 IO.B (RESET-OF)
                                         ~71.2 JI.EC (SIPO1)
                                         ~75.4 JJ.EC (SIPO2)
Press any key to continue.


                                         ~79.2 JK.EC (SIPO3)
                                         ~82.6 JL.EC (SIPO4)
                                         ~85.6 JM.EC (SIPO5)
                                           6.0 JN.EC (SIPO6)
                                         ~90.3 JO.EC (SIPO7)
                                         ~91.4 JP.EC (SIPO8)
---- net84. . . . .    CD.X (A-OTOT). . 14.6 EM.A (AOFB-1FIN)
---- net85. . . . .    CN.X (B-OTOT). .  5.7 EM.B (AOFB-1FIN)
---- net86. . . . .    DI.X (C-OTOT). .  6.5 HK.A (2FIN)
---- net87. . . . .    EM.X (AOFB-1FIN) 12.9 HK.B (2FIN)
                                          39.2 LC.A (FIN1)
---- net88. . . . .    KA.Y (SOM0). . .  1.5 KB.C (SOM1)
---- net89. . . . .    KB.Y (SOM1). . .  1.4 KC.C (SOM2)
```

```
SL-- net90
SL-- net91
SL-- net92
SL-- net93
SL-- net94
SL-- net95
---- net96. . . . .    ME.Y (SIPO25). .  2.7 MF.C (SIPO26)
                                        16.5 PH.A (SOM1-6)
---- net97. . . . .    MF.Y (SIPO26). .  4.2 MG.C (SIPO27)
                                        10.3 PE.A (SOM1-4)
Press any key to continue.


                                        32.2 P70.O
---- net98. . . . .    GP.X (MULT200) . 21.7 JI.A (SIPO1)
                                         7.0 P69.O
---- net99. . . . .    AI.X (OSS1). . .  5.3 AI.C (OSS1)
                                         4.2 AJ.K (OSS2)
                                         3.4 P83.O
                                         3.4 P83.T
                                         3.4 P84.O
                                         3.4 P84.T
---- net100 . . . .    P84.I. . . . . .  2.2 AI.B (OSS1)
---- net101 . . . .    P83.I. . . . . .  4.7 AI.E (OSS1)
---- net102 . . . .    AJ.Y (OSS2). . . 18.4 GCLK.I
---- net103 . . . .    MG.Y (SIPO27). .  3.8 MH.C (SIPO28)
                                        16.3 NC.A (SOM1-2)
---- net104 . . . .    AE.X (EENA). . . 21.2 BB.A (A-03)
                                        21.2 BB.B (A-03)
                                        24.0 BB.C (A-03)
                                        24.0 BB.D (A-03)
                                        20.1 BC.A (A-02)
                                        20.1 BC.B (A-02)
                                        21.3 BC.C (A-02)
                                        21.3 BC.D (A-02)
                                        19.7 BD.A (A-01)
Press any key to continue.


                                        19.7 BD.B (A-01)
                                        20.8 BD.C (A-01)
                                        20.8 BD.D (A-01)
                                        21.0 BE.A (A-00)
                                        21.0 BE.B (A-00)
                                        24.0 BE.C (A-00)
                                        24.0 BE.D (A-00)
                                        33.1 EI.A (PISO8)
---- net105 . . . .    BH.X (NULC). . .  6.6 CG.A (C-03)
                                         6.9 CG.B (C-03)
                                         7.5 CG.C (C-03)
                                         7.5 CG.D (C-03)
                                         6.7 CH.A (C-02)
                                         6.7 CH.B (C-02)
                                         9.3 CH.C (C-02)
                                         6.7 CH.D (C-02)
                                         2.5 CI.A (C-01)
                                         2.5 CI.B (C-01)
                                         5.2 CI.C (C-01)
                                        13.4 CI.D (C-01)
                                        14.1 CJ.B (C-00)
```

```
                                          4.5 CJ.C (C-00)
                                         13.4 CJ.D (C-00)
Press any key to continue.


                                        ~93.3 GA.D (MULT215)
                                        ~96.0 GB.D (MULT214)
                                        ~96.4 GC.D (MULT213)
                                        ~97.2 GD.D (MULT212)
                                        ~97.7 GE.D (MULT211)
                                         67.1 GF.D (MULT210)
                                         68.0 GG.D (MULT209)
                                         66.2 GH.D (MULT208)
                                         69.2 GI.D (MULT207)
                                         71.3 GJ.D (MULT206)
                                         73.7 GK.D (MULT205)
                                         73.9 GL.D (MULT204)
                                         74.5 GM.D (MULT203)
---- net106 . . . .    BI.X (EENC). . .  2.3 CJ.A (C-00)
                                         21.6 GN.D (MULT202)
                                         21.2 GO.D (MULT201)
                                         19.3 GP.D (MULT200)
---- net107 . . . .    AN.X (EENB). . .  2.1 BO.A (B-00)
                                         11.4 FP.D (MULT100)
---- net108 . . . .    AM.X (NULB). . .  6.6 BL.A (B-03)
                                          6.9 BL.B (B-03)
                                          7.5 BL.C (B-03)
                                          7.5 BL.D (B-03)
Press any key to continue.


                                          6.7 BM.A (B-02)
                                          6.7 BM.B (B-02)
                                          9.3 BM.C (B-02)
                                          6.7 BM.D (B-02)
                                          2.5 BN.A (B-01)
                                          2.5 BN.B (B-01)
                                          8.0 BN.C (B-01)
                                          8.0 BN.D (B-01)
                                          4.5 BO.B (B-00)
                                          4.5 BO.C (B-00)
                                         11.2 BO.D (B-00)
                                        ~62.0 FA.D (MULT115)
                                        ~61.3 FB.D (MULT114)
                                        ~60.5 FC.D (MULT113)
                                        ~59.3 FD.D (MULT112)
                                        ~57.8 FE.D (MULT111)
                                        ~56.0 FF.D (MULT110)
                                        ~53.9 FG.D (MULT109)
                                        ~51.4 FH.D (MULT108)
                                        ~48.5 FI.D (MULT107)
                                        ~45.3 FJ.D (MULT106)
                                        ~41.7 FK.D (MULT105)
                                         37.6 FL.D (MULT104)
Press any key to continue.


                                         32.9 FM.D (MULT103)
                                         29.6 FN.D (MULT102)
                                         29.6 FO.D (MULT101)
```

```
SL-- net109
SL-- net110
SL-- net111
SL-- net112
SL-- net113
SL-- net114
---- net115 . . . .    LC.X (FIN1). . . 13.7 MA.EC (SIPO21)
                                        14.5 MB.EC (SIPO22)
                                        12.7 MC.EC (SIPO23)
                                         3.8 MD.EC (SIPO24)
                                         5.1 ME.EC (SIPO25)
                                         8.2 MF.EC (SIPO26)
                                         8.9 MG.EC (SIPO27)
                                         9.2 MH.EC (SIPO28)
---- net116 . . . .    MH.Y (SIPO28). . 13.7 OA.A (SOM1-0)
---- net117 . . . .    MH.X (SIPO28). . 18.6 OB.A (SOM1-1)
---- net118 . . . .    MG.X (SIPO27). . 13.1 OD.A (SOM1-3)
                                        31.8 P71.O
---- net119 . . . .    KC.Y (SOM2). . .  2.2 JD.C (SOM3)
SL-- net120
Press any key to continue.


---- net121 . . . .    JD.Y (SOM3). . . 19.9 KE.C (SOM4)
---- net122 . . . .    KE.Y (SOM4). . .  1.4 KF.C (SOM5)
---- net123 . . . .    AA.X (OSS) . . . 39.7 P63.O
---- net124 . . . .    TE.X . . . . . .  6.4 P45.O
---- net125 . . . .    KF.Y (SOM5). . .  1.4 KG.C (SOM6)
---- net126 . . . .    JP.Y (SIPO8) . . 45.0 KA.A (SOM0)
---- net127 . . . .    JP.X (SIPO8) . . 58.4 KB.A (SOM1)
---- net128 . . . .    JO.X (SIPO7) . . 14.7 JD.A (SOM3)
---- net129 . . . .    JN.X (SIPO6) . . 28.9 KF.A (SOM5)
---- net130 . . . .    KG.Y (SOM6). . .  1.4 KH.C (SOM7)
---- net131 . . . .    HI.Y (TEL9). . .  4.9 HK.D (2FIN)
                                        14.8 P68.O
---- net132 . . . .    KH.Y (SOM7). . .  2.3 LI.C (SOM8)
---- net133 . . . .    LI.Y (SOM8). . .  5.2 NI.C (SOM9)
---- net134 . . . .    TI.Y (SIPO31). .  1.4 TJ.C (SIPO32)
---- net135 . . . .    MK.Y (SOM10) . .  2.0 LL.C (SOM11)
---- net136 . . . .    LL.Y (SOM11) . .  2.0 MM.C (SOM12)
---- net137 . . . .    MM.Y (SOM12) . .  1.5 MN.C (SOM13)
---- net138 . . . .    MN.Y (SOM13) . .  1.4 MO.C (SOM14)
---- net139 . . . .    MO.Y (SOM14) . .  1.4 MP.C (SOM15)
---- net140 . . . .    HI.X (TEL9). . . 24.1 LC.D (FIN1)
SL-- net141
SL-- net142
Press any key to continue.


SL-- net143
---- net144 . . . .    OA.Y (SOM1-0). .  1.5 OB.C (SOM1-1)
SL-- net145
SL-- net146
SL-- net147
SL-- net148
---- net149 . . . .    P48.I (RES). . .201.3 IA.RD (REF-TEL1)
                                       134.0 IB.RD (REF-TEL2)
                                       130.7 IO.A (RESET-OF)
                                       204.5 JA.RD (REF-TEL3)
                                       194.0 JD.RD (SOM3)
```

```
                                    195.3 KA.RD (SOM0)
                                    194.9 KB.RD (SOM1)
                                    194.2 KC.RD (SOM2)
                                    191.6 KE.RD (SOM4)
                                    189.6 KF.RD (SOM5)
                                    187.0 KG.RD (SOM6)
                                    180.6 KH.RD (SOM7)
                                    178.1 LI.RD (SOM8)
                                    178.1 LL.RD (SOM11)
                                    160.0 MK.RD (SOM10)
                                    142.2 MM.RD (SOM12)
                                    110.7 MN.RD (SOM13)
Press any key to continue.

                                    129.3 MO.RD (SOM14)
                                    132.4 MP.RD (SOM15)
                                    188.9 NC.RD (SOM1-2)
                                      8.8 NI.RD (SOM9)
                                    190.6 OA.RD (SOM1-0)
                                    190.6 OB.RD (SOM1-1)
                                    190.6 OD.RD (SOM1-3)
                                    190.6 OH.RD (SOM1-7)
                                    190.6 OI.RD (SOM1-8)
                                    190.6 OJ.RD (SOM1-9)
                                    190.6 OK.RD (SOM1-10)
                                    190.6 OL.RD (SOM1-11)
                                    190.6 OM.RD (SOM1-12)
                                    190.6 ON.RD (SOM1-13)
                                    190.6 OO.RD (SOM1-14)
                                    190.6 OP.RD (SOM1-15)
                                    191.4 PE.RD (SOM1-4)
                                    ~45.5 PH.RD (SOM1-6)
                                    189.6 QF.RD (SOM1-5)
---- net150 . . . .    HK.Y (2FIN). . . 32.6 IA.K (REF-TEL1)
                                     36.3 IB.K (REF-TEL2)
                                     23.5 JA.K (REF-TEL3)
                                     39.9 JD.EC (SOM3)
Press any key to continue.

                                     23.4 KA.EC (SOM0)
                                     23.1 KB.EC (SOM1)
                                     22.2 KC.EC (SOM2)
                                     18.0 KE.EC (SOM4)
                                     17.5 KF.EC (SOM5)
                                     14.1 KG.EC (SOM6)
                                     13.7 KH.EC (SOM7)
                                     12.3 LI.EC (SOM8)
                                     21.6 LL.EC (SOM11)
                                    ~28.7 MK.EC (SOM10)
                                     14.8 MM.EC (SOM12)
                                     15.5 MN.EC (SOM13)
                                     19.5 MO.EC (SOM14)
                                     23.4 MP.EC (SOM15)
                                    ~75.9 NC.EC (SOM1-2)
                                    ~46.0 NI.EC (SOM9)
                                    ~81.9 OA.EC (SOM1-0)
                                    ~85.2 OB.EC (SOM1-1)
                                    ~86.2 OD.EC (SOM1-3)
```

```
                                          ~44.3 OH.EC (SOM1-7)
                                          ~44.6 OI.EC (SOM1-8)
                                          ~45.3 OJ.EC (SOM1-9)
                                          ~28.7 OK.EC (SOM1-10)
Press any key to continue.

                                           21.6 OL.EC (SOM1-11)
                                           21.4 OM.EC (SOM1-12)
                                           22.2 ON.EC (SOM1-13)
                                           25.0 OO.EC (SOM1-14)
                                           24.3 OP.EC (SOM1-15)
                                          ~68.7 PE.EC (SOM1-4)
                                          ~40.8 PH.EC (SOM1-6)
                                          ~63.1 QF.EC (SOM1-5)
---- net151 . . . .    JM.Y (SIPO5) . .  3.5 JN.C (SIPO6)
                                           30.4 KG.A (SOM6)
---- net152 . . . .    OB.Y (SOM1-1). .  2.0 NC.C (SOM1-2)
---- net153 . . . .    JM.X (SIPO5) . . 13.6 KH.A (SOM7)
---- net154 . . . .    JL.Y (SIPO4) . .  4.4 JM.C (SIPO5)
                                           10.4 LI.A (SOM8)
---- net155 . . . .    JL.X (SIPO4) . . 21.8 NI.A (SOM9)
---- net156 . . . .    JI.X (SIPO1) . . 13.9 MP.A (SOM15)
---- net157 . . . .    JJ.X (SIPO2) . .  7.7 MN.A (SOM13)
---- net158 . . . .    JK.X (SIPO3) . .  6.3 LL.A (SOM11)
---- net159 . . . .    AC.X (NULA). . . 23.8 EB.A (PISO1)
                                           25.0 EB.C (PISO1)
                                           22.6 EC.A (PISO2)
                                           23.6 EC.C (PISO2)
                                           23.9 ED.A (PISO3)
Press any key to continue.

                                           22.7 ED.C (PISO3)
                                           26.0 EE.A (PISO4)
                                          ~26.8 EE.C (PISO4)
                                          ~31.0 EF.A (PISO5)
                                          ~28.4 EF.C (PISO5)
                                          ~32.6 EG.A (PISO6)
                                          ~32.8 EG.C (PISO6)
                                          ~34.0 EH.A (PISO7)
                                          ~34.8 EH.C (PISO7)
                                          ~35.4 EI.C (PISO8)
---- net160 . . . .    NC.Y (SOM1-2). .  2.0 OD.C (SOM1-3)
---- net161 . . . .    OD.Y (SOM1-3). .  2.0 PE.C (SOM1-4)
---- net162 . . . .    PE.Y (SOM1-4). .  5.0 QF.C (SOM1-5)
---- net163 . . . .    QF.Y (SOM1-5). .  5.9 PH.C (SOM1-6)
---- net164 . . . .    PH.Y (SOM1-6). . 11.5 OH.C (SOM1-7)
---- net165 . . . .    OH.Y (SOM1-7). .  1.4 OI.C (SOM1-8)
---- net166 . . . .    OI.Y (SOM1-8). .  1.4 OJ.C (SOM1-9)
---- net167 . . . .    OJ.Y (SOM1-9). .  1.4 OK.C (SOM1-10)
---- net168 . . . .    OK.Y (SOM1-10) .  1.4 OL.C (SOM1-11)
---- net169 . . . .    OL.Y (SOM1-11) .  1.4 OM.C (SOM1-12)
---- net170 . . . .    OM.Y (SOM1-12) .  1.4 ON.C (SOM1-13)
---- net171 . . . .    ON.Y (SOM1-13) .  1.4 OO.C (SOM1-14)
---- net172 . . . .    OO.Y (SOM1-14) .  1.4 OP.C (SOM1-15)
Press any key to continue.

---- net173 . . . .    MF.X (SIPO26). . 14.7 QF.A (SOM1-5)
```

```
---- net174 . . . .    ME.X (SIPO25). .  8.5 OH.A (SOM1-7)
---- net175 . . . .    MD.X (SIPO24). . 10.9 OJ.A (SOM1-9)
---- net176 . . . .    MC.X (SIPO23). . 16.5 OL.A (SOM1-11)
---- net177 . . . .    MB.X (SIPO22). . 24.5 ON.A (SOM1-13)
---- net178 . . . .    MA.X (SIPO21). . 30.5 OP.A (SOM1-15)
---- net179 . . . .    OA.X (SOM1-0). .~47.0 P82.O
                                        9.0 RA.A (SKL0)
---- net180 . . . .    OB.X (SOM1-1). . 41.5 P81.O
                                        7.1 RB.A (SKL1)
---- net181 . . . .    NC.X (SOM1-2). .~46.7 P80.O
                                       12.7 RC.A (SKL2)
---- net182 . . . .    OD.X (SOM1-3). . 59.5 P79.O
                                       15.2 RD.A (SKL3)
---- net183 . . . .    RA.Y (SKL0). . .  0.0 SA.A (OP/AF0)
---- net184 . . . .    RB.Y (SKL1). . .  0.0 SB.A (OP/AF1)
---- net185 . . . .    RC.Y (SKL2). . .  0.0 SC.A (OP/AF2)
---- net186 . . . .    RD.Y (SKL3). . .  3.4 TD.A (OP/AF3)
---- net187 . . . .    RE.Y (SKL4). . .  0.0 SE.A (OP/AF4)
---- net188 . . . .    RF.Y (SKL5). . .  0.0 SF.A (OP/AF5)
---- net189 . . . .    RG.Y (SKL6). . .  0.0 SG.A (OP/AF6)
---- net190 . . . .    RH.Y (SKL7). . .  0.0 SH.A (OP/AF7)
---- net191 . . . .    RI.Y (SKL8). . .  0.0 SI.A (OP/AF8)
Press any key to continue.


---- net192 . . . .    RJ.Y (SKL9). . .  0.0 SJ.A (OP/AF9)
---- net193 . . . .    QK.Y (SKL10) . .  0.0 RK.A (OP/AF10)
---- net194 . . . .    QL.Y (SKL11) . .  0.0 RL.A (OP/AF11)
---- net195 . . . .    QM.Y (SKL12) . .  0.0 RM.A (OP/AF12)
---- net196 . . . .    QN.Y (SKL13) . .  0.0 RN.A (OP/AF13)
---- net197 . . . .    QO.Y (SKL14) . .  0.0 RO.A (OP/AF14)
---- net198 . . . .    QP.Y (SKL15) . .  0.0 RP.A (OP/AF15)
---- net199 . . . .    QA.Y (Deel). . . 50.1 HK.C (2FIN)
                                       ~95.1 IO.C (RESET-OF)
                                       ~50.0 PJ.A (Reset-OF2)
                                       ~93.7 P67.O
---- net200 . . . .    SA.X (OP/AF0). .  2.1 RB.B (SKL1)
---- net201 . . . .    SB.X (OP/AF1). .  2.1 RC.B (SKL2)
---- net202 . . . .    SC.X (OP/AF2). .  2.1 RD.B (SKL3)
---- net203 . . . .    TD.X (OP/AF3). .  2.9 RE.B (SKL4)
---- net204 . . . .    SE.X (OP/AF4). .  2.1 RF.B (SKL5)
---- net205 . . . .    SF.X (OP/AF5). .  1.9 RG.B (SKL6)
---- net206 . . . .    SG.X (OP/AF6). .  2.1 RH.B (SKL7)
---- net207 . . . .    SH.X (OP/AF7). .  2.2 RI.B (SKL8)
---- net208 . . . .    SI.X (OP/AF8). .  2.1 RJ.B (SKL9)
---- net209 . . . .    SJ.X (OP/AF9). .  3.2 QK.B (SKL10)
---- net210 . . . .    RK.X (OP/AF10) .  2.1 QL.B (SKL11)
---- net211 . . . .    RL.X (OP/AF11) .  2.1 QM.B (SKL12)
Press any key to continue.


---- net212 . . . .    RM.X (OP/AF12) .  2.1 QN.B (SKL13)
---- net213 . . . .    RN.X (OP/AF13) .  1.9 QO.B (SKL14)
---- net214 . . . .    RO.X (OP/AF14) .  2.1 QP.B (SKL15)
---- net215 . . . .    SA.Y (OP/AF0). .  1.4 SB.C (OP/AF1)
---- net216 . . . .    SB.Y (OP/AF1). .  1.4 SC.C (OP/AF2)
---- net217 . . . .    SC.Y (OP/AF2). .  2.0 TD.C (OP/AF3)
---- net218 . . . .    TD.Y (OP/AF3). .  2.0 SE.C (OP/AF4)
---- net219 . . . .    SE.Y (OP/AF4). .  1.4 SF.C (OP/AF5)
---- net220 . . . .    SF.Y (OP/AF5). .  1.4 SG.C (OP/AF6)
```

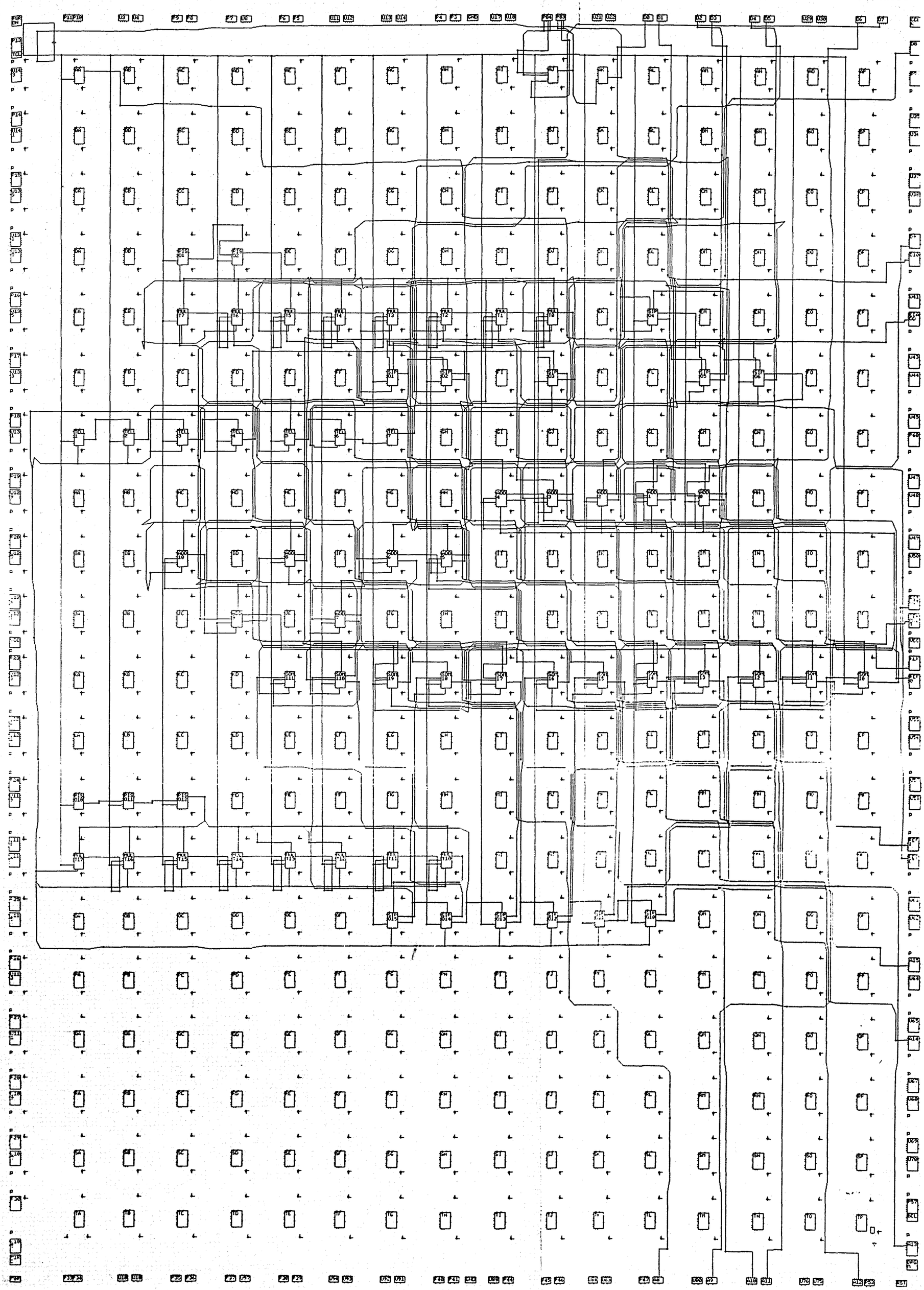
```
---- net221 . . . .    SG.Y (OP/AF6). .  1.4 SH.C (OP/AF7)
---- net222 . . . .    SH.Y (OP/AF7). .  1.5 SI.C (OP/AF8)
---- net223 . . . .    SI.Y (OP/AF8). .  1.5 SJ.C (OP/AF9)
---- net224 . . . .    SJ.Y (OP/AF9). .  2.0 RK.C (OP/AF10)
---- net225 . . . .    RK.Y (OP/AF10) .  1.4 RL.C (OP/AF11)
---- net226 . . . .    RL.Y (OP/AF11) .  1.4 RM.C (OP/AF12)
---- net227 . . . .    RM.Y (OP/AF12) .  1.4 RN.C (OP/AF13)
---- net228 . . . .    RN.Y (OP/AF13) .  1.4 RO.C (OP/AF14)
---- net229 . . . .    RO.Y (OP/AF14) .  1.4 RP.C (OP/AF15)
---- net230 . . . .    RP.X (OP/AF15) . 16.6 PJ.B (Reset-OF2)
                                       16.9 P66.O
                                       24.9 TI.A (SIPO31)
                                       24.1 TJ.A (SIPO32)
                                       17.4 TK.A (SIPO33)
Press any key to continue.

                                       16.5 TL.A (SIPO43)
                                       14.2 TM.A (SIPO53)
                                        8.9 TN.A (SIPO63)
                                        8.3 TO.A (SIPO73)
                                        7.4 TP.A (SIPO83)
---- net231 . . . .    PJ.X (Reset-OF2)~38.2 RK.D (OP/AF10)
                                      ~42.3 RL.D (OP/AF11)
                                      ~45.5 RM.D (OP/AF12)
                                      ~48.1 RN.D (OP/AF13)
                                      ~50.0 RO.D (OP/AF14)
                                      ~51.6 RP.D (OP/AF15)
                                      ~81.9 SA.C (OP/AF0)
                                      ~81.9 SA.D (OP/AF0)
                                      ~81.1 SB.D (OP/AF1)
                                      ~79.5 SC.D (OP/AF2)
                                      ~73.1 SE.D (OP/AF4)
                                      ~70.7 SF.D (OP/AF5)
                                      ~71.9 SG.D (OP/AF6)
                                      ~60.0 SH.D (OP/AF7)
                                      ~63.7 SI.D (OP/AF8)
                                      ~64.9 SJ.D (OP/AF9)
                                      ~74.7 TD.D (OP/AF3)
                                      ~64.7 TI.RD (SIPO31)
Press any key to continue.

                                      ~74.2 TJ.RD (SIPO32)
                                      ~74.2 TK.RD (SIPO33)
                                      ~77.6 TL.RD (SIPO43)
                                      ~80.3 TM.RD (SIPO53)
                                      ~83.0 TN.RD (SIPO63)
                                      ~85.3 TO.RD (SIPO73)
                                      ~86.0 TP.RD (SIPO83)
---- net232 . . . .    PE.X (SOM1-4). .  5.9 RE.A (SKL4)
---- net233 . . . .    JA.Y (REF-TEL3).~21.0 HA.EC (TEL1)
                                      ~18.6 HB.EC (TEL2)
                                      ~47.8 HC.EC (TEL3)
                                      ~20.8 HD.EC (TEL4)
                                      ~54.9 HE.EC (TEL5)
                                      ~56.8 HF.EC (TEL6)
                                      ~58.8 HG.EC (TEL7)
                                      ~59.5 HH.EC (TEL8)
                                      ~60.3 HI.EC (TEL9)
```

```
                                        ~45.7 PA.B (Vertraag)
---- net234 . . . .    JD.X (SOM3). . . 38.5 P75.0
                                        ~43.6 TD.B (OP/AF3)
---- net235 . . . .    TJ.Y (SIPO32). .  1.4 TK.C (SIPO33)
---- net236 . . . .    IB.Y (REF-TEL2).  4.3 JA.A (REF-TEL3)
---- net237 . . . .    OH.X (SOM1-7). .  8.0 RH.A (SKL7)
Press any key to continue.


SL-- net238
---- net239 . . . .    OI.X (SOM1-8). . 12.1 RI.A (SKL8)
---- net240 . . . .    OJ.X (SOM1-9). .  8.3 RJ.A (SKL9)
---- net241 . . . .    OK.X (SOM1-10) .  3.5 QK.A (SKL10)
---- net242 . . . .    OL.X (SOM1-11) .  4.5 QL.A (SKL11)
---- net243 . . . .    OM.X (SOM1-12) .  3.2 QM.A (SKL12)
---- net244 . . . .    ON.X (SOM1-13) .  3.4 QN.A (SKL13)
---- net245 . . . .    OO.X (SOM1-14) .  6.0 QO.A (SKL14)
---- net246 . . . .    OP.X (SOM1-15) .  3.1 QP.A (SKL15)
---- net247 . . . .    PH.X (SOM1-6). . 12.1 RG.A (SKL6)
---- net248 . . . .    QF.X (SOM1-5). . 22.6 RF.A (SKL5)
---- net249 . . . .    KA.X (SOM0). . . 43.6 P78.0
                                        15.6 SA.B (OP/AF0)
---- net250 . . . .    KB.X (SOM1). . . 36.3 P77.0
                                        23.1 SB.B (OP/AF1)
---- net251 . . . .    KC.X (SOM2). . . 50.0 P76.0
                                        22.3 SC.B (OP/AF2)
---- net252 . . . .    PA.Y (Vertraag). 19.5 QA.D (Deel)
---- net253 . . . .    KE.X (SOM4). . . 34.8 P73.0
                                         9.9 SE.B (OP/AF4)
---- net254 . . . .    KF.X (SOM5). . . 11.3 SF.B (OP/AF5)
---- net255 . . . .    KG.X (SOM6). . . 30.2 SG.B (OP/AF6)
---- net256 . . . .    KH.X (SOM7). . . 36.5 SH.B (OP/AF7)
Press any key to continue.


---- net257 . . . .    LI.X (SOM8). . . 20.4 SI.B (OP/AF8)
---- net258 . . . .    NI.X (SOM9). . . 15.7 SJ.B (OP/AF9)
---- net259 . . . .    MK.X (SOM10) . . 19.3 RK.B (OP/AF10)
---- net260 . . . .    LL.X (SOM11) . . 16.3 RL.B (OP/AF11)
---- net261 . . . .    MM.X (SOM12) . .  9.8 RM.B (OP/AF12)
---- net262 . . . .    MN.X (SOM13) . .  9.9 RN.B (OP/AF13)
---- net263 . . . .    MO.X (SOM14) . . 10.5 RO.B (OP/AF14)
---- net264 . . . .    MP.X (SOM15) . .  8.4 RP.B (OP/AF15)
---- net265 . . . .    TK.Y (SIPO33). .  1.4 TL.C (SIPO43)
SL-- net266
---- net267 . . . .    IA.Y (REF-TEL1).  5.8 IB.A (REF-TEL2)
---- net268 . . . .    TL.Y (SIPO43). .  1.5 TM.C (SIPO53)
---- net269 . . . .    TM.Y (SIPO53). .  1.4 TN.C (SIPO63)
---- net270 . . . .    TN.Y (SIPO63). .  1.4 TO.C (SIPO73)
---- net271 . . . .    NI.Y (SOM9). . . 19.8 MK.C (SOM10)
---- net272 . . . .    TO.Y (SIPO73). .  1.4 TP.C (SIPO83)
SL-- net276
---- net279 . . . .    QA.X (Deel). . . 21.8 QK.D (SKL10)
                                        21.8 QL.D (SKL11)
                                        15.3 QM.D (SKL12)
                                        21.8 QN.D (SKL13)
                                        21.8 QO.D (SKL14)
                                        21.8 QP.D (SKL15)
Press any key to continue.
```

```
                                          7.8 RA.D (SKL0)
                                          7.8 RB.D (SKL1)
                                          6.8 RC.D (SKL2)
                                         10.1 RD.D (SKL3)
                                         12.4 RE.D (SKL4)
                                         14.0 RF.D (SKL5)
                                         14.0 RG.D (SKL6)
                                         14.0 RH.D (SKL7)
                                         14.0 RI.D (SKL8)
                                         14.0 RJ.D (SKL9)
---- RST. . . . . .    IO.Y (RESET-OF).~74.1 EB.D (PISO1)
                                        ~74.0 EC.D (PISO2)
                                        ~76.1 ED.D (PISO3)
                                        ~77.9 EE.D (PISO4)
                                        ~80.0 EF.D (PISO5)
                                        ~81.6 EG.D (PISO6)
                                        ~82.8 EH.D (PISO7)
                                        ~83.7 EI.D (PISO8)
                                        ~80.8 FA.RD (MULT115)
                                        ~82.4 FB.RD (MULT114)
                                        ~80.8 FC.RD (MULT113)
                                        ~82.5 FD.RD (MULT112)
                                        ~86.2 FE.RD (MULT111)
Press any key to continue.


                                        ~87.3 FF.RD (MULT110)
                                        ~87.9 FG.RD (MULT109)
                                        ~93.5 FH.RD (MULT108)
                                        ~95.6 FI.RD (MULT107)
                                        ~97.3 FJ.RD (MULT106)
                                        102.7 FK.RD (MULT105)
                                        104.8 FL.RD (MULT104)
                                        106.1 FM.RD (MULT103)
                                        110.7 FN.RD (MULT102)
                                        112.5 FO.RD (MULT101)
                                        113.3 FP.RD (MULT100)
                                        ~81.7 GA.RD (MULT215)
                                        ~77.6 GB.RD (MULT214)
                                        ~64.0 GC.RD (MULT213)
                                        ~79.7 GD.RD (MULT212)
                                        ~64.3 GE.RD (MULT211)
                                        ~64.9 GF.RD (MULT210)
                                        ~85.2 GG.RD (MULT209)
                                        ~98.7 GH.RD (MULT208)
                                        ~97.5 GI.RD (MULT207)
                                        ~98.1 GJ.RD (MULT206)
                                        101.7 GK.RD (MULT205)
                                        104.8 GL.RD (MULT204)
Press any key to continue.


                                        107.9 GM.RD (MULT203)
                                        109.3 GN.RD (MULT202)
                                        112.6 GO.RD (MULT201)
                                        114.0 GP.RD (MULT200)
                                        ~82.3 HA.RD (TEL1)
                                        ~81.9 HB.RD (TEL2)
                                        ~66.7 HC.RD (TEL3)
```

```
~79.7 HD.RD (TEL4)
~55.1 HE.RD (TEL5)
~55.9 HF.RD (TEL6)
~85.2 HG.RD (TEL7)
~99.5 HH.RD (TEL8)
~98.5 HI.RD (TEL9)
~51.7 JI.RD (SIP01)
~51.7 JJ.RD (SIP02)
 26.7 JK.RD (SIP03)
 22.2 JL.RD (SIP04)
 21.4 JM.RD (SIP05)
 24.8 JN.RD (SIP06)
~51.7 JO.RD (SIP07)
~51.7 JP.RD (SIP08)
~74.6 MA.RD (SIP021)
~74.8 MB.RD (SIP022)
Press any key to continue.

~79.7 HD.RD (TEL4)
~55.1 HE.RD (TEL5)
~55.9 HF.RD (TEL6)
~85.2 HG.RD (TEL7)
~99.5 HH.RD (TEL8)
~98.5 HI.RD (TEL9)
~51.7 JI.RD (SIP01)
~51.7 JJ.RD (SIP02)
 26.7 JK.RD (SIP03)
 22.2 JL.RD (SIP04)
 21.4 JM.RD (SIP05)
 24.8 JN.RD (SIP06)
~51.7 JO.RD (SIP07)
~51.7 JP.RD (SIP08)
~74.6 MA.RD (SIP021)
~74.8 MB.RD (SIP022)
Press any key to continue.
~76.1 MC.RD (SIP023)
~78.8 MD.RD (SIP024)
~80.1 ME.RD (SIP025)
~81.7 MF.RD (SIP026)
~55.4 MG.RD (SIP027)
~55.4 MH.RD (SIP028)
Press any key to continue.
```

AANHANGSEL 4.4.5.a: Die stroombaandiagram vir die geheue–adressering.



AANHANGSEL 4.4.5.b: Die verbindingslys vir die geheue–adressering.

```
---- net0 . . . . .    EC.X (MULT7) . .  2.5 ED.C (MULT6)
---- net1 . . . . .    ED.X (MULT6) . .  2.3 EE.C (MULT5)
---- net2 . . . . .    EE.X (MULT5) . .  2.5 EF.C (MULT4)
---- net3 . . . . .    EF.X (MULT4) . .  2.5 EG.C (MULT3)
---- net4 . . . . .    EG.X (MULT3) . .  2.3 EH.C (MULT2)
---- net5 . . . . .    EH.X (MULT2) . .  2.5 EI.C (MULT1)
---- net6 . . . . .    EI.X (MULT1) . .  2.5 EJ.C (MULT0)
---- net7 . . . . .    P59.I (A15). . . 47.3 EC.D (MULT7)
---- net8 . . . . .    P58.I (A14). . . 47.1 ED.D (MULT6)
---- net9 . . . . .    P56.I (A13). . . 52.4 EE.D (MULT5)
---- net10. . . . .    P52.I (A12). . . 47.9 EF.D (MULT4)
---- net11. . . . .    P51.I (A11). . . 49.3 EG.D (MULT3)
---- net12. . . . .    P50.I (A10). . . 44.7 EH.D (MULT2)
---- net13. . . . .    P49.I (A9) . . . 44.6 EI.D (MULT1)
---- net14. . . . .    P48.I (A8) . . . 52.9 EJ.D (MULT0)
---- net15. . . . .    DC.Y (PISO1) . .  2.5 DD.B (PISO2)
---- net16. . . . .    DD.X (PISO2) . .  7.5 EC.A (MULT7)
                                          7.0 ED.A (MULT6)
                                          3.2 EE.A (MULT5)
                                          4.3 EF.A (MULT4)
                                          4.9 EG.A (MULT3)
                                          8.3 EH.A (MULT2)
Press any key to continue.


                                          9.2 EI.A (MULT1)
                                          9.7 EJ.A (MULT0)
---- net17. . . . .    AJ.X . . . . . .  6.3 AJ.C
                                          5.3 AK.K
                                          5.4 P83.O
                                          5.4 P83.T
                                          5.7 P84.O
                                          5.7 P84.T
---- net18. . . . .    AA.X . . . . . . 47.1 P63.O (KLK)
---- net19. . . . .    P83.I. . . . . .  3.5 AJ.E
---- net20. . . . .    P84.I. . . . . .  2.3 AJ.B
---- net21. . . . .    AK.Y . . . . . . 24.4 GCLK.I
---- net22. . . . .    HM.Y (ADD0). . .  4.6 HL.C (ADD1)
---- net23. . . . .    GCLK.O . . . . .  3.7 AA.K
                                          3.8 DC.K (PISO1)
                                          3.8 DD.K (PISO2)
                                          3.8 EC.K (MULT7)
                                          3.8 ED.K (MULT6)
                                          3.8 EE.K (MULT5)
                                          3.8 EF.K (MULT4)
                                          3.8 EG.K (MULT3)
                                          3.8 EH.K (MULT2)
                                          3.7 EI.K (MULT1)
Press any key to continue.


                                          3.8 EJ.K (MULT0)
                                          3.7 EL.K (SIPO4)
                                          3.8 FG.K (SIPO1)
                                          3.8 FH.K (SIPO2)
                                          3.8 FJ.K (SIPO3)
                                          3.6 FM.K (SIPO5)
                                          3.5 FN.K (SIPO6)
```

```
                                                3.7 GA.K (TEL1)
                                                3.6 GB.K (TEL2)
                                                3.8 GC.K (TEL3)
                                                3.8 GD.K (TEL4)
                                                3.8 GE.K (TEL5)
                                                3.8 GF.K (TEL6)
                                                3.7 HI.K (ADD4)
                                                3.8 HJ.K (ADD3)
                                                3.6 HK.K (ADD2)
                                                3.7 HL.K (ADD1)
                                                3.6 HM.K (ADD0)
                                                3.8 IC.K (ADD10)
                                                3.8 IE.K (ADD8)
                                                3.8 IG.K (ADD6)
                                                3.8 IH.K (ADD5)
                                                3.8 JD.K (ADD9)
Press any key to continue.

                                                3.8 JF.K (ADD7)
                                                3.8 KE.K (SOM111)
                                                3.8 KF.K (SOM110)
                                                3.8 KG.K (SOM19)
                                                3.8 KH.K (SOM18)
                                                3.7 KI.K (SOM17)
                                                3.8 KJ.K (SOM16)
                                                3.6 KK.K (SOM15)
                                                3.7 KL.K (SOM14)
                                                3.6 KM.K (SOM13)
                                                3.5 KN.K (SOM12)
                                                3.4 KO.K (SOM11)
                                                3.4 KP.K (SOM10)
                                                3.7 MA.K (PISO10)
                                                3.6 MB.K (PISO11)
                                                3.8 MC.K (PISO12)
                                                3.7 NA.K (MULT17)
                                                3.6 NB.K (MULT16)
                                                3.8 NC.K (MULT15)
                                                3.8 ND.K (MULT14)
                                                3.8 NE.K (MULT13)
                                                3.8 NF.K (MULT12)
                                                3.8 NG.K (MULT11)
Press any key to continue.

                                                3.8 NH.K (MULT10)
                                                3.8 OG.K (SIPO15)
                                                3.8 OH.K (SIPO14)
                                                3.7 OI.K (SIPO13)
                                                3.8 OJ.K (SIPO12)
                                                3.6 OK.K (SIPO11)
                                                3.7 OL.K (SIPO10)
---- net24. . . . .    EJ.X (MULT0) . . 10.9 FG.A (SIPO1)
                                           23.6 P66.0 (SERMULT)
---- net25. . . . .    FJ.Y (SIPO3) . . 35.2 EL.A (SIPO4)
                                           14.4 IG.A (ADD6)
---- net26. . . . .    EL.Y (SIPO4) . .  8.2 FM.A (SIPO5)
                                           27.0 HI.A (ADD4)
---- net27. . . . .    FM.Y (SIPO5) . .  4.2 FN.A (SIPO6)
                                           10.5 HK.A (ADD2)
```

```
---- net28. . . . .    FN.Y (SIPO6) . . 14.8 HH.A (ADD0)
---- net29. . . . .    FN.X (SIPO6) . . 10.8 HL.A (ADD1)
---- net30. . . . .    HL.Y (ADD1). . .  5.5 HK.C (ADD2)
---- net31. . . . .    HK.Y (ADD2). . .  7.3 HJ.C (ADD3)
---- net32. . . . .    HJ.Y (ADD3). . .  4.6 HI.C (ADD4)
---- net33. . . . .    FM.X (SIPO5) . . 29.9 HJ.A (ADD3)
---- net34. . . . .    EL.X (SIPO4) . . 20.9 IH.A (ADD5)
---- net35. . . . .    FJ.X (SIPO3) . . 18.8 JF.A (ADD7)
Press any key to continue.


---- net36. . . . .    FG.X (SIPO1) . .  5.1 FH.A (SIPO2)
                                         ~17.4 IC.A (ADD10)
---- net37. . . . .    FH.Y (SIPO2) . . 10.5 FJ.A (SIPO3)
                                          15.6 IE.A (ADD8)
---- net38. . . . .    FH.X (SIPO2) . . 19.1 JD.A (ADD9)
---- net39. . . . .    P60.I (RESET). .213.2 DC.D (PISO1)
                                         208.9 DD.D (PISO2)
                                         224.9 EC.RD (MULT7)
                                         226.7 ED.RD (MULT6)
                                         207.1 EE.RD (MULT5)
                                         210.1 EF.RD (MULT4)
                                         202.1 EG.RD (MULT3)
                                         192.7 EH.RD (MULT2)
                                         171.6 EI.RD (MULT1)
                                         160.6 EJ.RD (MULT0)
                                         170.3 EL.RD (SIPO4)
                                         206.1 FG.RD (SIPO1)
                                         196.0 FH.RD (SIPO2)
                                         186.0 FJ.RD (SIPO3)
                                         ~78.6 FM.RD (SIPO5)
                                         ~68.0 FN.RD (SIPO6)
                                         200.1 GA.RD (TEL1)
                                         203.1 GB.RD (TEL2)
Press any key to continue.


                                         205.6 GC.RD (TEL3)
                                         208.5 GD.RD (TEL4)
                                         209.6 GE.RD (TEL5)
                                         214.4 GF.D (TEL6)
                                         208.2 NA.RD (MULT17)
                                         213.6 NB.RD (MULT16)
                                         214.4 NC.RD (MULT15)
                                         218.0 ND.RD (MULT14)
                                         219.4 NE.RD (MULT13)
                                         220.2 NF.RD (MULT12)
                                         223.4 NG.RD (MULT11)
                                         223.9 NH.RD (MULT10)
                                         221.6 OG.RD (SIPO15)
                                         225.3 OH.RD (SIPO14)
                                         226.7 OI.RD (SIPO13)
                                         227.4 OJ.RD (SIPO12)
                                         230.6 OK.RD (SIPO11)
                                         231.2 OL.RD (SIPO10)
                                          10.8 P67.O (RSTUIT)
---- net40. . . . .    HI.Y (ADD4). . .  7.0 IH.C (ADD5)
---- net41. . . . .    IH.Y (ADD5). . .  7.3 IG.C (ADD6)
---- net42. . . . .    IG.Y (ADD6). . .  7.0 JF.C (ADD7)
---- net43. . . . .    JF.Y (ADD7). . .  9.2 IE.C (ADD8)
```

Press any key to continue.

```
---- net44. . . . .    IE.Y (ADD8). . .  9.5 JD.C (ADD9)
---- net45. . . . .    JD.Y (ADD9). . .  4.2 IC.C (ADD10)
SL-- net46
---- net47. . . . .    GA.Y (TEL1). . .  2.3 GB.A (TEL2)
---- net48. . . . .    GB.Y (TEL2). . .  2.5 GC.A (TEL3)
---- net49. . . . .    GC.Y (TEL3). . .  2.5 GD.A (TEL4)
---- net50. . . . .    GD.Y (TEL4). . .  3.5 GE.A (TEL5)
---- net51. . . . .    GE.Y (TEL5). . .  2.5 GF.A (TEL6)
---- net52. . . . .    GF.Y (TEL6). . .  2.5 GG.E (TEL7)
---- net53. . . . .    P62.I (REF0) . . 18.9 HL.B (ADD1)
---- net54. . . . .    HM.X (ADD0). . . 14.1 KP.A (SOM10)
                                        25.4 P82.O (D0)
---- net55. . . . .    HL.X (ADD1). . .  9.8 KO.A (SOM11)
                                       ~20.3 P81.O (D1)
---- net56. . . . .    HK.X (ADD2). . . 10.8 KN.A (SOM12)
                                        20.4 P80.O (D2)
---- net57. . . . .    HJ.X (ADD3). . . 16.2 KM.A (SOM13)
                                       ~33.3 P79.O (D3)
---- net58. . . . .    HI.X (ADD4). . . 15.0 KL.A (SOM14)
                                       ~31.8 P78.O (D4)
SL-- net59
SL-- net60
---- net61. . . . .    IH.X (ADD5). . . 12.7 KK.A (SOM15)
Press any key to continue.


                                       ~37.5 P77.O (D5)
---- net62. . . . .    IG.X (ADD6). . . 13.1 KJ.A (SOM16)
                                       ~40.3 P76.O (D6)
---- net63. . . . .    P61.I (REF1) . . 18.0 HM.B (ADD0)
---- net64. . . . .    JF.X (ADD7). . .  9.3 KI.A (SOM17)
                                        38.9 P75.O (D7)
---- net65. . . . .    GG.X (TEL7). . . 38.1 EL.EC (SIPO4)
                                        21.1 FG.EC (SIPO1)
                                        25.6 FH.EC (SIPO2)
                                        34.0 FJ.EC (SIPO3)
                                        50.2 FM.EC (SIPO5)
                                        51.9 FN.EC (SIPO6)
                                        44.1 HI.RD (ADD4)
                                        34.0 HJ.RD (ADD3)
                                        44.1 HK.RD (ADD2)
                                        44.1 HL.RD (ADD1)
                                        44.1 HM.RD (ADD0)
                                       ~67.0 IC.RD (ADD10)
                                       107.9 IE.RD (ADD8)
                                       111.7 IG.RD (ADD6)
                                        25.6 IH.RD (ADD5)
                                       ~93.9 JD.RD (ADD9)
                                       109.0 JF.RD (ADD7)
Press any key to continue.


                                       112.2 KE.RD (SOM111)
                                       110.5 KF.RD (SOM110)
                                       110.5 KG.RD (SOM19)
                                       110.6 KH.RD (SOM18)
                                       106.5 KI.RD (SOM17)
```

```
                                          ~93.9 KJ.RD (SOM16)
                                          ~94.0 KK.RD (SOM15)
                                          ~92.4 KL.RD (SOM14)
                                          ~91.5 KM.RD (SOM13)
                                          ~74.3 KN.D (SOM12)
                                           68.2 KO.RD (SOM11)
                                           63.2 KP.RD (SOM10)
                                           53.7 P69.O (ECADD)
---- net66. . . . .   IE.X (ADD8). . .  13.3 KG.A (SOM19)
                                           13.8 KH.A (SOM18)
                                           41.6 P73.O (D8)
---- net67. . . . .   JD.X (ADD9). . .  31.8 P71.O (D9)
---- net68. . . . .   KF.X (SOM110). .  47.0 P70.O (D10)
S--- net69. . . . . . . . . . . . . .   *** P68.O
---- net70. . . . .   KP.Y (SOM10) . .   6.7 KO.C (SOM11)
---- net71. . . . .   KO.Y (SOM11) . .   4.1 KN.C (SOM12)
---- net72. . . . .   KN.Y (SOM12) . .   7.3 KM.C (SOM13)
---- net73. . . . .   KM.Y (SOM13) . .   4.6 KL.C (SOM14)
Press any key to continue.


---- net76. . . . .   KL.Y (SOM14) . .   6.6 KK.C (SOM15)
---- net77. . . . .   KK.Y (SOM15) . .   6.0 KJ.C (SOM16)
---- net78. . . . .   KJ.Y (SOM16) . .   7.3 KI.C (SOM17)
---- net79. . . . .   KI.Y (SOM17) . .   5.0 KH.C (SOM18)
---- net80. . . . .   KH.Y (SOM18) . .   6.0 KG.C (SOM19)
---- net81. . . . .   KG.Y (SOM19) . .   7.3 KF.C (SOM110)
---- net82. . . . .   KF.Y (SOM110). .   5.2 KE.C (SOM111)
---- net83. . . . .   IC.X (ADD10) . .  11.0 KF.A (SOM110)
---- net84. . . . .   MA.Y (PISO10). .   2.5 MB.C (PISO11)
---- net85. . . . .   MB.Y (PISO11). .   2.5 MC.C (PISO12)
---- net86. . . . .   NA.X (MULT17). .   2.3 NB.C (MULT16)
---- net87. . . . .   NB.X (MULT16). .   2.5 NC.C (MULT15)
---- net88. . . . .   NC.X (MULT15). .   2.5 ND.C (MULT14)
---- net89. . . . .   ND.X (MULT14). .   2.3 NE.C (MULT13)
---- net90. . . . .   NE.X (MULT13). .   2.5 NF.C (MULT12)
---- net91. . . . .   NF.X (MULT12). .   2.5 NG.C (MULT11)
---- net92. . . . .   NG.X (MULT11). .   2.3 NH.C (MULT10)
---- net93. . . . .   MC.X (PISO12). .  10.4 NA.A (MULT17)
                                           9.8 NB.A (MULT16)
                                           5.5 NC.A (MULT15)
                                           6.6 ND.A (MULT14)
                                           7.5 NE.A (MULT13)
                                           8.3 NF.A (MULT12)
Press any key to continue.


                                          12.7 NG.A (MULT11)
                                          13.3 NH.A (MULT10)
SL-- net94
SL-- net95
SL-- net96
SL-- net97
SL-- net98
---- net100 . . . .   OG.Y (SIPO15). .  13.4 KF.B (SOM110)
                                           4.6 OH.A (SIPO14)
---- net101 . . . .   OH.Y (SIPO14). .  11.6 KH.B (SOM18)
                                           4.4 OI.A (SIPO13)
---- net102 . . . .   OI.Y (SIPO13). .   8.4 KJ.B (SOM16)
                                           3.7 OJ.A (SIPO12)
```

```
---- net103 . . . .    OJ.Y (SIPO12). . 15.0 KL.B (SOM14)
                                          4.8 OK.A (SIPO11)
---- net104 . . . .    OK.Y (SIPO11). . 13.8 KN.B (SOM12)
                                          3.7 OL.A (SIPO10)
---- net108 . . . .    NH.X (MULT10). .  4.7 OG.A (SIPO15)
---- net109 . . . .    OG.X (SIPO15). . 15.5 KE.B (SOM111)
---- net110 . . . .    OH.X (SIPO14). . 14.3 KG.B (SOM19)
---- net111 . . . .    OI.X (SIPO13). . 17.8 KI.B (SOM17)
---- net112 . . . .    OJ.X (SIPO12). .  8.3 KK.B (SOM15)
---- net113 . . . .    OK.X (SIPO11). . 11.4 KM.B (SOM13)
Press any key to continue.


SL-- net95
SL-- net96
SL-- net97
SL-- net98
---- net100 . . . .    OG.Y (SIPO15). . 13.4 KF.B (SOM110)
                                          4.6 OH.A (SIPO14)
---- net101 . . . .    OH.Y (SIPO14). . 11.6 KH.B (SOM18)
                                          4.4 OI.A (SIPO13)
---- net102 . . . .    OI.Y (SIPO13). .  8.4 KJ.B (SOM16)
                                          3.7 OJ.A (SIPO12)
---- net103 . . . .    OJ.Y (SIPO12). . 15.0 KL.B (SOM14)
                                          4.8 OK.A (SIPO11)
---- net104 . . . .    OK.Y (SIPO11). . 13.8 KN.B (SOM12)
                                          3.7 OL.A (SIPO10)
---- net108 . . . .    NH.X (MULT10). .  4.7 OG.A (SIPO15)
---- net109 . . . .    OG.X (SIPO15). . 15.5 KE.B (SOM111)
---- net110 . . . .    OH.X (SIPO14). . 14.3 KG.B (SOM19)
---- net111 . . . .    OI.X (SIPO13). . 17.8 KI.B (SOM17)
---- net112 . . . .    OJ.X (SIPO12). .  8.3 KK.B (SOM15)
---- net113 . . . .    OK.X (SIPO11). . 11.4 KM.B (SOM13)
Press any key to continue.
---- net114 . . . .    OL.X (SIPO10). . 16.3 KP.B (SOM10)
---- net115 . . . .    OL.Y (SIPO10). . 12.5 KO.B (SOM11)
Press any key to continue.
```

```
                                          ~71.2 JI.EC (SIP01)
                                          ~75.4 JJ.EC (SIP02)
                                          ~79.2 JK.EC (SIP03)
                                          ~82.6 JL.EC (SIP04)
                                          ~85.6 JM.EC (SIP05)
                                            6.0 JN.EC (SIP06)
                                          ~90.3 JO.EC (SIP07)
                                          ~91.4 JP.EC (SIP08)
---- net84. . . . .    CD.X (A-0TOT). .  14.6 EM.A (AOFB-1FIN)
---- net85. . . . .    CN.X (B-0TOT). .   5.7 EM.B (AOFB-1FIN)
---- net86. . . . .    DI.X (C-0TOT). .   6.5 HK.A (2FIN)
---- net87. . . . .    EM.X (AOFB-1FIN) 12.9 HK.B (2FIN)
                                          39.2 LC.A (FIN1)
---- net88. . . . .    KA.Y (SOM0). . .   1.5 KB.C (SOM1)
---- net89. . . . .    KB.Y (SOM1). . .   1.4 KC.C (SOM2)
SL-- net90
SL-- net91
SL-- net92
SL-- net93
SL-- net94
Press any key to continue.


SL-- net95
---- net96. . . . .    ME.Y (SIP025). .   2.7 MF.C (SIP026)
                                          16.5 PH.A (SOM1-6)
---- net97. . . . .    MF.Y (SIP026). .   4.2 MG.C (SIP027)
                                          10.3 PE.A (SOM1-4)
                                          32.2 P70.O
S--- net98. . . . . . . . . . . . . . .  *** JI.A (SIP01)
                                          *** P69.O
---- net99. . . . .    AI.X (OSS1). . .   5.3 AI.C (OSS1)
                                           4.2 AJ.K (OSS2)
                                           3.4 P83.O
                                           3.4 P83.T
                                           3.4 P84.O
                                           3.4 P84.T
---- net100 . . . .    P84.I. . . . . .   2.2 AI.B (OSS1)
---- net101 . . . .    P83.I. . . . . .   4.7 AI.E (OSS1)
---- net102 . . . .    AJ.Y (OSS2). . .  18.4 GCLK.I
---- net103 . . . .    MG.Y (SIP027). .   3.8 MH.C (SIP028)
                                          16.3 NC.A (SOM1-2)
---- net104 . . . .    AE.X (EENA). . .  21.2 BB.A (A-03)
                                          21.2 BB.B (A-03)
                                          24.0 BB.C (A-03)
                                          24.0 BB.D (A-03)
Press any key to continue.


                                          20.1 BC.A (A-02)
                                          20.1 BC.B (A-02)
                                          21.3 BC.C (A-02)
                                          21.3 BC.D (A-02)
                                          19.7 BD.A (A-01)
                                          19.7 BD.B (A-01)
                                          20.8 BD.C (A-01)
                                          20.8 BD.D (A-01)
                                          21.0 BE.A (A-00)
                                          21.0 BE.B (A-00)
                                          24.0 BE.C (A-00)
```

```
                                          24.0 BE.D (A-00)
                                          33.1 EI.A (PIS08)
---- net105 . . . .    BH.X (NULC). . .    6.6 CG.A (C-03)
                                           6.9 CG.B (C-03)
                                           7.5 CG.C (C-03)
                                           7.5 CG.D (C-03)
                                           6.7 CH.A (C-02)
                                           6.7 CH.B (C-02)
                                           9.3 CH.C (C-02)
                                           6.7 CH.D (C-02)
                                           2.5 CI.A (C-01)
                                           2.5 CI.B (C-01)
Press any key to continue.


                                           5.2 CI.C (C-01)
                                          13.1 CI.D (C-01)
                                          13.9 CJ.B (C-00)
                                           4.5 CJ.C (C-00)
                                          13.1 CJ.D (C-00)
---- net106 . . . .    BI.X (EENC). . .    1.4 CJ.A (C-00)
---- net107 . . . .    AN.X (EENB). . .    2.1 BO.A (B-00)
                                          17.3 FO.D (MULT101)
---- net108 . . . .    AM.X (NULB). . .    6.6 BL.A (B-03)
                                           6.9 BL.B (B-03)
                                           7.5 BL.C (B-03)
                                           7.5 BL.D (B-03)
                                           6.7 BM.A (B-02)
                                           6.7 BM.B (B-02)
                                           9.3 BM.C (B-02)
                                           6.7 BM.D (B-02)
                                           2.5 BN.A (B-01)
                                           2.5 BN.B (B-01)
                                           8.0 BN.C (B-01)
                                           8.0 BN.D (B-01)
                                           4.5 BO.B (B-00)
                                           4.5 BO.C (B-00)
                                          11.2 BO.D (B-00)
Press any key to continue.


                                         ~63.1 FA.D (MULT115)
                                         ~62.4 FB.D (MULT114)
                                         ~61.5 FC.D (MULT113)
                                         ~60.4 FD.D (MULT112)
                                         ~58.9 FE.D (MULT111)
                                         ~57.1 FF.D (MULT110)
                                         ~55.0 FG.D (MULT109)
                                         ~52.5 FH.D (MULT108)
                                         ~49.7 FI.D (MULT107)
                                         ~46.5 FJ.D (MULT106)
                                         ~42.8 FK.D (MULT105)
                                          38.7 FL.D (MULT104)
                                          34.0 FM.D (MULT103)
                                          30.5 FN.D (MULT102)
                                          29.9 FP.D (MULT100)
SL-- net109
SL-- net110
SL-- net111
SL-- net112
```

```
SL-- net113
SL-- net114
---- net115 . . . .    LC.X (FIN1). . . 13.7 MA.EC (SIPO21)
                                      14.5 MB.EC (SIPO22)
Press any key to continue.


                                      12.7 MC.EC (SIPO23)
                                       3.8 MD.EC (SIPO24)
                                       5.1 ME.EC (SIPO25)
                                       8.2 MF.EC (SIPO26)
                                       8.9 MG.EC (SIPO27)
                                       9.2 MH.EC (SIPO28)
---- net116 . . . .    MH.Y (SIPO28). . 13.7 OA.A (SOM1-0)
---- net117 . . . .    MH.X (SIPO28). . 18.6 OB.A (SOM1-1)
---- net118 . . . .    MG.X (SIPO27). . 13.1 OD.A (SOM1-3)
                                      31.8 P71.O
---- net119 . . . .    KC.Y (SOM2). . .  2.2 JD.C (SOM3)
SL-- net120
---- net121 . . . .    JD.Y (SOM3). . . 19.9 KE.C (SOM4)
---- net122 . . . .    KE.Y (SOM4). . .  1.4 KF.C (SOM5)
---- net123 . . . .    AA.X (OSS) . . . 39.7 P63.O
---- net124 . . . .    TE.X . . . . . .  6.4 P45.O
---- net125 . . . .    KF.Y (SOM5). . .  1.4 KG.C (SOM6)
---- net126 . . . .    JP.Y (SIPO8) . . 45.0 KA.A (SOM0)
---- net127 . . . .    JP.X (SIPO8) . . 58.4 KB.A (SOM1)
---- net128 . . . .    JO.X (SIPO7) . . 14.7 JD.A (SOM3)
---- net129 . . . .    JN.X (SIPO6) . . 28.9 KF.A (SOM5)
---- net130 . . . .    KG.Y (SOM6). . .  1.4 KH.C (SOM7)
---- net131 . . . .    HI.Y (TEL9). . .  4.9 HK.D (2FIN)
Press any key to continue.


                                      14.8 P68.O
---- net132 . . . .    KH.Y (SOM7). . .  2.3 LI.C (SOM8)
---- net133 . . . .    LI.Y (SOM8). . .  5.2 NI.C (SOM9)
---- net135 . . . .    MK.Y (SOM10) . .  2.0 LL.C (SOM11)
---- net136 . . . .    LL.Y (SOM11) . .  2.0 MM.C (SOM12)
---- net137 . . . .    MM.Y (SOM12) . .  1.5 MN.C (SOM13)
---- net138 . . . .    MN.Y (SOM13) . .  1.4 MO.C (SOM14)
---- net139 . . . .    MO.Y (SOM14) . .  1.4 MP.C (SOM15)
---- net140 . . . .    HI.X (TEL9). . . 24.1 LC.D (FIN1)
SL-- net141
SL-- net142
SL-- net143
---- net144 . . . .    OA.Y (SOM1-0). .  1.5 OB.C (SOM1-1)
SL-- net145
SL-- net146
SL-- net147
SL-- net148
---- net149 . . . .    P48.I (RES). . .200.9 IA.RD (REF-TEL1)
                                     134.1 IB.RD (REF-TEL2)
                                     134.1 IC.RD (REF-TEL3)
                                     130.7 IO.A (RESET-OF)
                                     194.0 JD.RD (SOM3)
                                     195.3 KA.RD (SOM0)
Press any key to continue.


                                     194.9 KB.RD (SOM1)
```

```
                                        194.2 KC.RD (SOM2)
                                        191.6 KE.RD (SOM4)
                                        189.6 KF.RD (SOM5)
                                        187.0 KG.RD (SOM6)
                                        180.6 KH.RD (SOM7)
                                        178.1 LI.RD (SOM8)
                                        178.1 LL.RD (SOM11)
                                        160.0 MK.RD (SOM10)
                                        142.2 MM.RD (SOM12)
                                        110.7 MN.RD (SOM13)
                                        129.3 MO.RD (SOM14)
                                        132.4 MP.RD (SOM15)
                                        188.9 NC.RD (SOM1-2)
                                          8.8 NI.RD (SOM9)
                                        190.6 OA.RD (SOM1-0)
                                        190.6 OB.RD (SOM1-1)
                                        190.6 OD.RD (SOM1-3)
                                        190.6 OH.RD (SOM1-7)
                                        190.6 OI.RD (SOM1-8)
                                        190.6 OJ.RD (SOM1-9)
                                        190.6 OK.RD (SOM1-10)
                                        190.6 OL.RD (SOM1-11)
Press any key to continue.

                                        190.6 OM.RD (SOM1-12)
                                        190.6 ON.RD (SOM1-13)
                                        190.6 OO.RD (SOM1-14)
                                        190.6 OP.RD (SOM1-15)
                                        191.4 PE.RD (SOM1-4)
                                        ~45.5 PH.RD (SOM1-6)
                                        189.6 QF.RD (SOM1-5)
---- net150 . . . .    HK.Y (2FIN). . . 31.7 IA.K (REF-TEL1)
                                         35.3 IB.K (REF-TEL2)
                                         37.5 IC.K (REF-TEL3)
                                         39.0 JD.EC (SOM3)
                                         22.5 KA.EC (SOM0)
                                         22.2 KB.EC (SOM1)
                                         21.7 KC.EC (SOM2)
                                         18.0 KE.EC (SOM4)
                                         17.5 KF.EC (SOM5)
                                         14.1 KG.EC (SOM6)
                                         13.7 KH.EC (SOM7)
                                         12.3 LI.EC (SOM8)
                                         21.6 LL.EC (SOM11)
                                        ~28.7 MK.EC (SOM10)
                                         14.8 MM.EC (SOM12)
                                         15.5 MN.EC (SOM13)
Press any key to continue.

                                         19.5 MO.EC (SOM14)
                                         23.4 MP.EC (SOM15)
                                        ~75.9 NC.EC (SOM1-2)
                                        ~46.0 NI.EC (SOM9)
                                        ~81.9 OA.EC (SOM1-0)
                                        ~85.2 OB.EC (SOM1-1)
                                        ~86.2 OD.EC (SOM1-3)
                                        ~44.3 OH.EC (SOM1-7)
                                        ~44.6 OI.EC (SOM1-8)
```

```
                                          ~45.3 OJ.EC (SOM1-9)
                                          ~28.7 OK.EC (SOM1-10)
                                           21.6 OL.EC (SOM1-11)
                                           21.4 OM.EC (SOM1-12)
                                           22.2 ON.EC (SOM1-13)
                                           25.0 OO.EC (SOM1-14)
                                           24.3 OP.EC (SOM1-15)
                                          ~68.7 PE.EC (SOM1-4)
                                          ~40.8 PH.EC (SOM1-6)
                                          ~63.1 QF.EC (SOM1-5)
---- net151 . . . .    JM.Y (SIP05) . .    3.5 JN.C (SIP06)
                                           30.4 KG.A (SOM6)
---- net152 . . . .    OB.Y (SOM1-1). .    2.0 NC.C (SOM1-2)
---- net153 . . . .    JM.X (SIP05) . .   13.6 KH.A (SOM7)
Press any key to continue.


---- net154 . . . .    JL.Y (SIP04) . .    4.4 JM.C (SIP05)
                                           10.4 LI.A (SOM8)
---- net155 . . . .    JL.X (SIP04) . .   21.8 NI.A (SOM9)
---- net156 . . . .    JI.X (SIP01) . .   13.9 MP.A (SOM15)
---- net157 . . . .    JJ.X (SIP02) . .    7.7 MN.A (SOM13)
---- net158 . . . .    JK.X (SIP03) . .    6.3 LL.A (SOM11)
---- net159 . . . .    AC.X (NULA). . .   23.8 EB.A (PIS01)
                                           25.0 EB.C (PIS01)
                                           22.6 EC.A (PIS02)
                                           23.6 EC.C (PIS02)
                                           23.9 ED.A (PIS03)
                                           22.7 ED.C (PIS03)
                                           26.0 EE.A (PIS04)
                                          ~26.8 EE.C (PIS04)
                                          ~31.0 EF.A (PIS05)
                                          ~28.4 EF.C (PIS05)
                                          ~32.6 EG.A (PIS06)
                                          ~32.8 EG.C (PIS06)
                                          ~34.0 EH.A (PIS07)
                                          ~34.8 EH.C (PIS07)
                                          ~35.4 EI.C (PIS08)
---- net160 . . . .    NC.Y (SOM1-2). .    2.0 OD.C (SOM1-3)
---- net161 . . . .    OD.Y (SOM1-3). .    2.0 PE.C (SOM1-4)
Press any key to continue.


---- net162 . . . .    PE.Y (SOM1-4). .    5.0 QF.C (SOM1-5)
---- net163 . . . .    QF.Y (SOM1-5). .    5.9 PH.C (SOM1-6)
---- net164 . . . .    PH.Y (SOM1-6). .   11.5 OH.C (SOM1-7)
---- net165 . . . .    OH.Y (SOM1-7). .    1.4 OI.C (SOM1-8)
---- net166 . . . .    OI.Y (SOM1-8). .    1.4 OJ.C (SOM1-9)
---- net167 . . . .    OJ.Y (SOM1-9). .    1.4 OK.C (SOM1-10)
---- net168 . . . .    OK.Y (SOM1-10) .    1.4 OL.C (SOM1-11)
---- net169 . . . .    OL.Y (SOM1-11) .    1.4 OM.C (SOM1-12)
---- net170 . . . .    OM.Y (SOM1-12) .    1.4 ON.C (SOM1-13)
---- net171 . . . .    ON.Y (SOM1-13) .    1.4 OO.C (SOM1-14)
---- net172 . . . .    OO.Y (SOM1-14) .    1.4 OP.C (SOM1-15)
---- net173 . . . .    MF.X (SIP026). .   14.7 QF.A (SOM1-5)
---- net174 . . . .    ME.X (SIP025). .    8.5 OH.A (SOM1-7)
---- net175 . . . .    MD.X (SIP024). .   10.9 OJ.A (SOM1-9)
---- net176 . . . .    MC.X (SIP023). .   16.5 OL.A (SOM1-11)
---- net177 . . . .    MB.X (SIP022). .   24.5 ON.A (SOM1-13)
---- net178 . . . .    MA.X (SIP021). .   30.5 OP.A (SOM1-15)
```

```
---- net179 . . . .    OA.X (SOM1-0). .~47.0 P82.0
                                          9.0 RA.A (SKL0)
---- net180 . . . .    OB.X (SOM1-1). . 41.5 P81.0
                                          7.1 RB.A (SKL1)
---- net181 . . . .    NC.X (SOM1-2). .~46.7 P80.0
                                         12.7 RC.A (SKL2)
Press any key to continue.


---- net182 . . . .    OD.X (SOM1-3). . 59.5 P79.0
                                         15.2 RD.A (SKL3)
---- net183 . . . .    RA.Y (SKL0). . .  0.0 SA.A (OP/AF0)
---- net184 . . . .    RB.Y (SKL1). . .  0.0 SB.A (OP/AF1)
---- net185 . . . .    RC.Y (SKL2). . .  0.0 SC.A (OP/AF2)
---- net186 . . . .    RD.Y (SKL3). . .  3.4 TD.A (OP/AF3)
---- net187 . . . .    RE.Y (SKL4). . .  0.0 SE.A (OP/AF4)
---- net188 . . . .    RF.Y (SKL5). . .  0.0 SF.A (OP/AF5)
---- net189 . . . .    RG.Y (SKL6). . .  0.0 SG.A (OP/AF6)
---- net190 . . . .    RH.Y (SKL7). . .  0.0 SH.A (OP/AF7)
---- net191 . . . .    RI.Y (SKL8). . .  0.0 SI.A (OP/AF8)
---- net192 . . . .    RJ.Y (SKL9). . .  0.0 SJ.A (OP/AF9)
---- net193 . . . .    QK.Y (SKL10) . .  0.0 RK.A (OP/AF10)
---- net194 . . . .    QL.Y (SKL11) . .  0.0 RL.A (OP/AF11)
---- net195 . . . .    QM.Y (SKL12) . .  0.0 RM.A (OP/AF12)
---- net196 . . . .    QN.Y (SKL13) . .  0.0 RN.A (OP/AF13)
---- net197 . . . .    QO.Y (SKL14) . .  0.0 RO.A (OP/AF14)
---- net198 . . . .    QP.Y (SKL15) . .  0.0 RP.A (OP/AF15)
---- net199 . . . .    QA.Y (Deel). . . 50.1 HK.C (2PIN)
                                        ~95.1 IO.C (RESET-OF)
                                        ~50.0 PJ.A (Reset-OF2)
                                        ~93.7 P67.0
---- net200 . . . .    SA.X (OP/AF0). .  2.1 RB.B (SKL1)
Press any key to continue.


---- net201 . . . .    SB.X (OP/AF1). .  2.1 RC.B (SKL2)
---- net21. . . . .    FC.X (MULT113) .  1.4 FD.C (MULT112)
---- net202 . . . .    SC.X (OP/AF2). .  2.1 RD.B (SKL3)
---- net22. . . . .    FD.X (MULT112) .  1.4 FE.C (MULT111)
---- net203 . . . .    TD.X (OP/AF3). .  2.9 RE.B (SKL4)
---- net23. . . . .    FE.X (MULT111) .  1.2 FF.C (MULT110)
---- net204 . . . .    SE.X (OP/AF4). .  2.1 RF.B (SKL5)
---- net24. . . . .    FF.X (MULT110) .  1.4 FG.C (MULT109)
---- net205 . . . .    SF.X (OP/AF5). .  1.9 RG.B (SKL6)
---- net25. . . . .    FG.X (MULT109) .  1.4 FH.C (MULT108)
---- net206 . . . .    SG.X (OP/AF6). .  2.1 RH.B (SKL7)
---- net26. . . . .    FH.X (MULT108) .  1.2 FI.C (MULT107)
---- net207 . . . .    SH.X (OP/AF7). .  2.2 RI.B (SKL8)
---- net27. . . . .    FI.X (MULT107) .  1.4 FJ.C (MULT106)
---- net208 . . . .    SI.X (OP/AF8). .  2.1 RJ.B (SKL9)
---- net28. . . . .    FJ.X (MULT106) .  1.4 FK.C (MULT105)
---- net209 . . . .    SJ.X (OP/AF9). .  3.2 QK.B (SKL10)
---- net29. . . . .    FK.X (MULT105) .  1.2 FL.C (MULT104)
---- net210 . . . .    RK.X (OP/AF10) .  2.1 QL.B (SKL11)
---- net211 . . . .    RL.X (OP/AF11) .  2.1 QM.B (SKL12)
---- net212 . . . .    RM.X (OP/AF12) .  2.1 QN.B (SKL13)
---- net213 . . . .    RN.X (OP/AF13) .  1.9 QO.B (SKL14)
---- net214 . . . .    RO.X (OP/AF14) .  2.1 QP.B (SKL15)
Press any key to continue.
```

```
---- net215 . . . .    SA.Y (OP/AF0). .  1.4 SB.C (OP/AF1)
---- net216 . . . .    SB.Y (OP/AF1). .  1.4 SC.C (OP/AF2)
---- net217 . . . .    SC.Y (OP/AF2). .  2.0 TD.C (OP/AF3)
---- net218 . . . .    TD.Y (OP/AF3). .  2.0 SE.C (OP/AF4)
---- net219 . . . .    SE.Y (OP/AF4). .  1.4 SF.C (OP/AF5)
---- net220 . . . .    SF.Y (OP/AF5). .  1.4 SG.C (OP/AF6)
---- net221 . . . .    SG.Y (OP/AF6). .  1.4 SH.C (OP/AF7)
---- net222 . . . .    SH.Y (OP/AF7). .  1.5 SI.C (OP/AF8)
---- net223 . . . .    SI.Y (OP/AF8). .  1.5 SJ.C (OP/AF9)
---- net224 . . . .    SJ.Y (OP/AF9). .  2.0 RK.C (OP/AF10)
---- net225 . . . .    RK.Y (OP/AF10) .  1.4 RL.C (OP/AF11)
---- net226 . . . .    RL.Y (OP/AF11) .  1.4 RM.C (OP/AF12)
---- net227 . . . .    RM.Y (OP/AF12) .  1.4 RN.C (OP/AF13)
---- net228 . . . .    RN.Y (OP/AF13) .  1.4 RO.C (OP/AF14)
---- net229 . . . .    RO.Y (OP/AF14) .  1.4 RP.C (OP/AF15)
---- net230 . . . .    RP.X (OP/AF15) . 16.6 PJ.B (Reset-OF2)
                                       16.9 P66.O
                                       24.9 TI.A (SIPO31)
                                       24.1 TJ.A (SIPO32)
                                       17.4 TK.A (SIPO33)
                                       16.5 TL.A (SIPO34)
                                       14.2 TM.A (SIPO35)
                                        8.9 TN.A (SIPO36)
Press any key to continue.


                                        8.3 TO.A (SIPO37)
                                        7.4 TP.A (SIPO38)
---- net231 . . . .    PJ.X (Reset-OF2)~38.2 RK.D (OP/AF10)
                                       ~42.3 RL.D (OP/AF11)
                                       ~45.5 RM.D (OP/AF12)
                                       ~48.1 RN.D (OP/AF13)
                                       ~50.0 RO.D (OP/AF14)
                                       ~51.6 RP.D (OP/AF15)
                                       ~81.9 SA.C (OP/AF0)
                                       ~81.9 SA.D (OP/AF0)
                                       ~81.1 SB.D (OP/AF1)
                                       ~79.5 SC.D (OP/AF2)
                                       ~73.1 SE.D (OP/AF4)
                                       ~70.7 SF.D (OP/AF5)
                                       ~71.9 SG.D (OP/AF6)
                                       ~60.0 SH.D (OP/AF7)
                                       ~63.7 SI.D (OP/AF8)
                                       ~64.9 SJ.D (OP/AF9)
                                       ~74.7 TD.D (OP/AF3)
                                       ~64.7 TI.RD (SIPO31)
                                       ~74.2 TJ.RD (SIPO32)
                                       ~74.2 TK.RD (SIPO33)
                                       ~77.6 TL.RD (SIPO34)
Press any key to continue.


                                       ~80.3 TM.RD (SIPO35)
                                       ~83.0 TN.RD (SIPO36)
                                       ~85.3 TO.RD (SIPO37)
                                       ~86.0 TP.RD (SIPO38)
---- net232 . . . .    PE.X (SOM1-4). .  5.9 RE.A (SKL4)
---- net233 . . . .    IA.Y (REF-TEL1).~12.5 HA.EC (TEL1)
                                       ~16.6 HB.EC (TEL2)
```

```
                                          ~38.7 HC.EC (TEL3)
                                          ~12.1 HD.EC (TEL4)
                                          ~45.7 HE.EC (TEL5)
                                          ~47.7 HF.EC (TEL6)
                                          ~49.6 HG.EC (TEL7)
                                          ~50.4 HH.EC (TEL8)
                                          ~51.1 HI.EC (TEL9)
                                          ~43.0 PA.B (Vertraag)
---- net234 . . . .    JD.X (SOM3). . .  38.5 P75.O
                                          ~43.6 TD.B (OP/AF3)
---- net236 . . . .    IB.Y (REF-TEL2).   1.4 IC.A (REF-TEL3)
---- net237 . . . .    OH.X (SOM1-7). .   8.0 RH.A (SKL7)
SL-- net238
---- net239 . . . .    OI.X (SOM1-8). .  12.1 RI.A (SKL8)
---- net240 . . . .    OJ.X (SOM1-9). .   8.3 RJ.A (SKL9)
---- net241 . . . .    OK.X (SOM1-10) .   3.5 QK.A (SKL10)
Press any key to continue.


---- net242 . . . .    OL.X (SOM1-11) .   4.5 QL.A (SKL11)
---- net243 . . . .    OM.X (SOM1-12) .   3.2 QM.A (SKL12)
---- net244 . . . .    ON.X (SOM1-13) .   3.4 QN.A (SKL13)
---- net245 . . . .    OO.X (SOM1-14) .   6.0 QO.A (SKL14)
---- net246 . . . .    OP.X (SOM1-15) .   3.1 QP.A (SKL15)
---- net247 . . . .    PH.X (SOM1-6). .  12.1 RG.A (SKL6)
---- net248 . . . .    QF.X (SOM1-5). .  22.6 RF.A (SKL5)
---- net249 . . . .    KA.X (SOM0). . .  43.6 P78.O
                                          15.6 SA.B (OP/AF0)
---- net250 . . . .    KB.X (SOM1). . .  36.3 P77.O
                                          23.1 SB.B (OP/AF1)
---- net251 . . . .    KC.X (SOM2). . .  50.0 P76.O
                                          22.3 SC.B (OP/AF2)
---- net252 . . . .    PA.Y (Vertraag). 19.5 QA.D (Deel)
---- net253 . . . .    KE.X (SOM4). . .  34.8 P73.O
                                           9.9 SE.B (OP/AF4)
---- net254 . . . .    KF.X (SOM5). . .  11.3 SF.B (OP/AF5)
---- net255 . . . .    KG.X (SOM6). . .  30.2 SG.B (OP/AF6)
---- net256 . . . .    KH.X (SOM7). . .  36.5 SH.B (OP/AF7)
---- net257 . . . .    LI.X (SOM8). . .  20.4 SI.B (OP/AF8)
---- net258 . . . .    NI.X (SOM9). . .  15.7 SJ.B (OP/AF9)
---- net259 . . . .    MK.X (SOM10) . .  19.3 RK.B (OP/AF10)
---- net260 . . . .    LL.X (SOM11) . .  16.3 RL.B (OP/AF11)
Press any key to continue.


---- net261 . . . .    MM.X (SOM12) . .   9.8 RM.B (OP/AF12)
---- net262 . . . .    MN.X (SOM13) . .   9.9 RN.B (OP/AF13)
---- net263 . . . .    MO.X (SOM14) . .  10.5 RO.B (OP/AF14)
---- net264 . . . .    MP.X (SOM15) . .   8.4 RP.B (OP/AF15)
SL-- net266
---- net271 . . . .    NI.Y (SOM9). . .  19.8 MK.C (SOM10)
---- net273 . . . .    TI.Y (SIP031). .   1.4 TJ.C (SIP032)
---- net274 . . . .    TJ.Y (SIP032). .   1.4 TK.C (SIP033)
---- net275 . . . .    TK.Y (SIP033). .   1.4 TL.C (SIP034)
SL-- net276
---- net277 . . . .    TL.Y (SIP034). .   1.5 TM.C (SIP035)
---- net278 . . . .    TM.Y (SIP035). .   1.4 TN.C (SIP036)
---- net279 . . . .    QA.X (Deel). . .  21.8 QK.D (SKL10)
                                          21.8 QL.D (SKL11)
                                          15.3 QM.D (SKL12)
```

```
                                          21.8 QN.D (SKL13)
                                          21.8 QO.D (SKL14)
                                          21.8 QP.D (SKL15)
                                           7.8 RA.D (SKL0)
                                           7.8 RB.D (SKL1)
                                           6.8 RC.D (SKL2)
                                          10.1 RD.D (SKL3)
                                          12.4 RE.D (SKL4)
Press any key to continue.


                                          14.0 RF.D (SKL5)
                                          14.0 RG.D (SKL6)
                                          14.0 RH.D (SKL7)
                                          14.0 RI.D (SKL8)
                                          14.0 RJ.D (SKL9)
---- net280 . . . .    TN.Y (SIPO36). .    1.4 TO.C (SIPO37)
---- net281 . . . .    TO.Y (SIPO37). .    1.4 TP.C (SIPO38)
---- RST. . . . . .    IO.Y (RESET-OF).~72.7 EB.D (PISO1)
                                         ~72.6 EC.D (PISO2)
                                         ~74.7 ED.D (PISO3)
                                         ~76.5 EE.D (PISO4)
                                         ~78.6 EF.D (PISO5)
                                         ~80.2 EG.D (PISO6)
                                         ~81.4 EH.D (PISO7)
                                         ~82.3 EI.D (PISO8)
                                         ~79.5 FA.RD (MULT115)
                                         ~81.0 FB.RD (MULT114)
                                         ~79.5 FC.RD (MULT113)
                                         ~81.1 FD.RD (MULT112)
                                         ~84.9 FE.RD (MULT111)
                                         ~85.9 FF.RD (MULT110)
                                         ~86.5 FG.RD (MULT109)
                                         ~91.2 FH.RD (MULT108)
Press any key to continue.


                                         ~92.5 FI.RD (MULT107)
                                         ~93.4 FJ.RD (MULT106)
                                         ~96.8 FK.RD (MULT105)
                                         ~97.8 FL.RD (MULT104)
                                         ~98.4 FM.RD (MULT103)
                                         101.4 FN.RD (MULT102)
                                         102.3 FO.RD (MULT101)
                                         102.8 FP.RD (MULT100)
                                         ~80.4 HA.RD (TEL1)
                                         ~80.1 HB.RD (TEL2)
                                         ~63.8 HC.RD (TEL3)
                                         ~78.2 HD.RD (TEL4)
                                         ~55.1 HE.RD (TEL5)
                                         ~55.9 HF.RD (TEL6)
                                         ~83.7 HG.RD (TEL7)
                                         ~96.6 HH.RD (TEL8)
                                         ~95.7 HI.RD (TEL9)
                                         ~51.7 JI.RD (SIPO1)
                                         ~51.7 JJ.RD (SIPO2)
                                          26.7 JK.RD (SIPO3)
                                          22.2 JL.RD (SIPO4)
                                          21.4 JM.RD (SIPO5)
                                          24.8 JN.RD (SIPO6)
```

Press any key to continue.

```
~78.2 HD.RD (TEL4)
~55.1 HE.RD (TEL5)
~55.9 HF.RD (TEL6)
~83.7 HG.RD (TEL7)
~96.6 HH.RD (TEL8)
~95.7 HI.RD (TEL9)
~51.7 JI.RD (SIP01)
~51.7 JJ.RD (SIP02)
 26.7 JK.RD (SIP03)
 22.2 JL.RD (SIP04)
 21.4 JM.RD (SIP05)
 24.8 JN.RD (SIP06)
```

Press any key to continue.

```
~51.7 JO.RD (SIP07)
~51.7 JP.RD (SIP08)
~74.6 MA.RD (SIP021)
~74.8 MB.RD (SIP022)
~76.1 MC.RD (SIP023)
~78.8 MD.RD (SIP024)
~80.1 ME.RD (SIP025)
~81.7 MF.RD (SIP026)
~55.4 MG.RD (SIP027)
~55.4 MH.RD (SIP028)
```

Press any key to continue.

AANHANGSEL K: Die ontwerp van die apparatuurplatform:

'n IBM–versoenbare rekenaarkaart was gekies as apparatuurplatform vir ondersteuning van die wasige vlokkies. Omdat die wasige vlokkies potensiëel teen hoë klokfrekwensies kan funksioneer, was dit nodig om hoë spoed geheue te gebruik, wat die hoë prosesseeringsfrekwensie kan akkommodeer. Vir hierdie doel is 32 k greep SRAM, met toegangstyd van 80 ns, gekies.

K.1) Omskakeling van 8–bis na 16–bis:

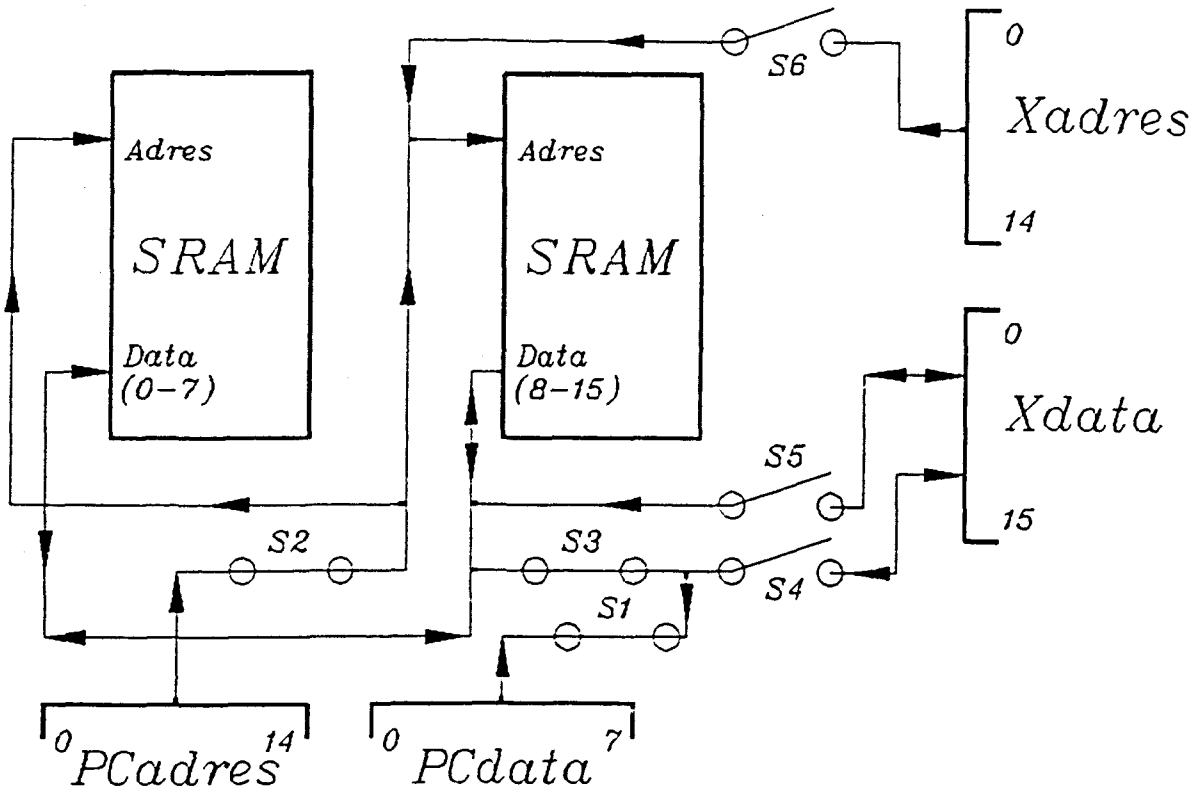
Aangesien die rekenaarkaart slegs oor 'n 8–bis datapoort fasiliteit beskik, moes die 16–bis data in twee kloksiklusse van die IBM–tipe rekenaar, na en van die geheue op die kaart geskryf en gelees word. 'n Blokdiagram van die geheue seleksie word in figuur K.1 getoon. Die geheue seleksie word beheer met ses digitale "skakelaars" of lyndrywers. In figuur K.1 word die toestand van die skakelaars getoon indien die rekenaar, in plaas van die Xilinx–stelsel, toegang het tot die geheue. Aangesien die Xilinx–stelsel gekonfigureer is om 'n afsonderlike inset–poort en uitset–poort te hê, word die skakelaar S3 gebruik om te verseker dat die uitset–poort nie direk na die inset–poort van dieselfde Xilinx–vlokkie skryf nie. Die ander skakelaars se gebruike is voordiehandliggend.

K.2) Geheue benodighede:

'n Geheue–greep van 32 k kan direk adresseer word met 15 adres–lyne, wat as 'n aanvaarbare hoeveelheid beskou word, vanuit 'n vlokkie–ontwerpsoogpunt. Die aantal gemonsterde data waardes wat deur hierdie hoeveelheid geheue geakkommodeer kan word, kan as volg bepaal word.

Vir 'n enkel–inset en enkel–uitset stelsel, met vyf referensie gebiede in die gebied

Figuur K.1: Blokdiagram van geheue–seleksiemetodiek.



van belang, en met t die aantal gemonsterde data waardes:

a) Vermenigvuldiging lewer $t \times 5 \times 3$ waardes.

b) Die relasie is $5 \times 5 \times 5 = 125$ waardes.

c) Wasige beraming lewer $5 \times t$ waardes.

d) Die ontwasigde ekwiwalent van die beraming lewer t waardes.

Sodat: $15t + 125 + 5t + t \leq 32\,768$

dus $t \leq 1\,554$

Die totale proses word beskryf deur $2t$ data–punte. Daarom is dit besluit om die proses, wat gemodelleer word, tot 3 000 data punte te beperk.

K.3) Die rekenaarkaart–adres.

Na vele eksperimentering is besluit om die eerste 32 k greep geheue, wat ooreenstem met die eerste 8–bisse van die 16–bis data, by adres D0000 te plaas, en die tweede stel van 8–bisse by D8000. Die adres–lyne A19, A18, A17, A16, en A15 selekteer die rekenaarkaart. Die geheue seleksie word bepaal deur slegs A15. (Die GAL–program, wat die geheue seleksie bewerkstellig, word in afdeling K.5 getoon.)

K.4) Die wasige vlokkie:

Die Xilinx–vlokkie is ontwerp met 'n interne bis–struktuur van 16–bisse. Die wasige vlokkies was ontwerp om 'n enkele 16–bis data inset en 'n enkele 16–bis uitset te kan hanteer. Ook word 'n 15–bis adresseringspoort benodig vir interaksie met die rekenaarkaart se geheue. Die ander vlokkie–konneksies word gebruik vir klok–insette en modus–programmeering, soos in afdeliing K.6 aangetoon word.

K.5). Die geheue–seleksie GAL–program:

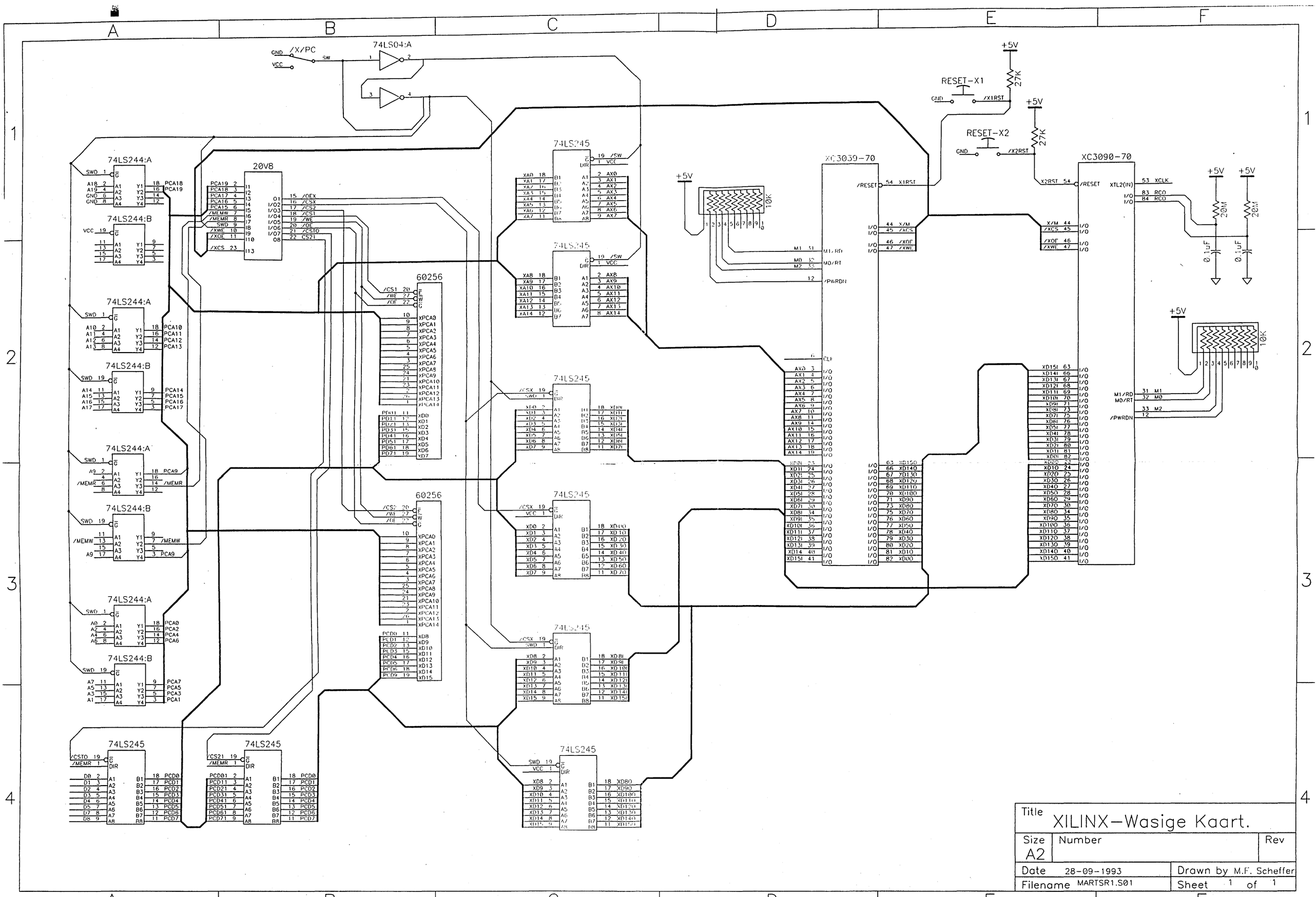
20V8

unused	01	24	VCC
A19	02	23	/XCS
A18	03	22	/CS21
A17	04	21	/CSTO
A16	05	20	/OE
A15	06	19	/WE
/MEMW	07	18	/CS1
/MEMR	08	17	/CS2
SWD	09	16	/CSX
/XWE	10	15	/WEX
/XOE	11	14	unused
GND	12	13	unused

Equations translated to Sum of Products form

```
/CSTO
  =     !A19
  |     !A18
  |      A17
  |     !A16
  |      SWD;

/CS1
  =      A15 & !SWD
  |     !SWD &  /CSTO
  |      SWD &  /XCS;

/CS2
  =     !A15 & !SWD
  |     !SWD &  /CSTO
  |      SWD &  /XCS;

/OE
  =      /MEMR & !SWD
  |      SWD &  /XOE;

/WE
  =      /MEMW & !SWD
  |      SWD &  /XWE;

/CSX
  =     !SWD
  |      /XCS;

/WEX
  =     !SWD
  |      /XWE;

/CS21
  =      SWD
  |      /CS2;
```

K.6) Die rekenaarkaart se stroombaandiagram:



Title	XILINX—Wasige Kaart.	
Size A2	Number	Rev
Date 28-09-1993		Drawn by M.F. Scheffer
Filename MARTSR1.S01		Sheet 1 of 1