

T E C H N I S C H E U N I V E R S I T E I T D E L F T

FACULTEIT DER ELEKTROTECHNIEK

Aard : Afstudeerverslag
Omvang : 59 pagina's + 30 pagina's APPENDIX
Datum : 24 November 1977

Lab/Afd. : Automatische Verkeerssystemen
Opdrachtnum. : Go / 1986 / 3
Auteur : G. TSIOURAKIS

Titel : Vervanging TCP18 processor in PRX/A
centrale door een microprocessor.

Afstudeerdocent : Prof. ir. H.J. Goebertus

Mentoren : ir. R.A. Beukers (TU Delft)
ing. M. Konings (APT)

Korte inhoud : In dit verslag wordt de mogelijkheid besproken van het emuleren van de TCP/18 processor op een microprocessor(bv. van het type MC68000) zonder wijzigingen in de huidige software aan te brengen. De " real-time " prestaties van een dergelijke emulatie worden in dit verslag ook besproken.

INHOUDSOPGAVE

VOORWOORD	1
SUMMARY	2
SAMENVATTING	3
1. HOOFDSTUK I	5
DE PRX/A TELEFOONCENTRALE	5
1.1 Inleiding	5
1.2 Opbouw van de PRX/A-centrale	5
2. HOOFDSTUK II	8
CENTRALE BESTURING VAN DE PRX/A-CENTRALE	8
2.1 Inleiding	8
2.2 Kwetsbaarheid en Betrouwbaarheid	9
2.3 Belangrijke functies van de CCU	9
2.4 Processor terminal gedeelte	10
2.5 Processor gedeelte	11
2.5.1 Aan de processor gekoppelde eenheden	11
2.5.2 Processor registers	12
2.5.3 Eenheden behorende tot de processor zelf	12
3. HOOFDSTUK III	16
BESCHRIJVING VAN HET EMULATIE PROGRAMMA	16
3.1 Inleiding	16
3.2 Inventarisatie en opbouw van de TCP18 instructies	17
3.2.1 Opbouw van memory reference instructies	18
3.2.2 Opbouw van memory constant niet gemodif. instructies	19
3.2.3 Opbouw van memory constant gemodif. instructies	19
3.2.4 Opbouw van halfword gemodificeerde instructies	19
3.2.5 Opbouw van data kanaal instructies	20
3.2.6 Opbouw van de LRR(Load Relocation Register) instructie	20
3.2.7 Opbouw van instructies met OP CODE 16-bits lang	20
3.3 Bepaling van de adresseringsmodes	21
3.4 Schematische weergave van het bepalen van het effectieve adres bij (MR) instructies	22
3.5 Algemene programma-opzet	27
3.5.1 Gebruikte variabelen in het emulatie programma	28
3.5.2 Tests benodigd voor het uitvoeren van een instructie	29
3.5.3 Gedefinieerde functies voor het emulatie programma	30
3.5.4 Programmaverloop bij executie van een instructie	31
3.5.5 Niet of gedeeltelijk geemuleerde instructies	33
4. HOOFDSTUK IV	36
NIEUW MICROPROCESSOR SYSTEEM EN ZIJN PRESTATIES	36
4.1 Inleiding	36
4.2 Gekozen nieuwe microprocessor	36
4.3 Bepaling van de prestaties van een processorbestuurd systeem	37
4.4 Gestelde eisen aan het nieuwe microprocessor systeem	38
CONCLUSIES EN AANBEVELINGEN	54
EPILOOG	57
LITERATUUROPGAVE	58
APPENDIX A	
APPENDIX B	

VOORWOORD

In dit voorwoord wil ik mijn dank betuigen aan alle personen die mij ondersteund hebben bij het verrichten van mijn afstudeerwerk.

In het bijzonder dank ik Prof.ir. H.J.Goebertus, ir. R.A.Beukers, ing. M.Konings en de PRX/A softwaregroep voor hun nimmer aflatende belangstelling en hun assistentie.

Verder gaat mijn dank uit naar mijn vrouw die mij voortdurend tot grote steun was.

SUMMARY

The subject of my thesis is to investigate to what extent it would be possible, in behalf of saving space, to replace the present voluminous PRX/A central control unit with the TCP18 processors by a new central control unit with microprocessors.

This replacement should be realized without any alterations in the existing software and in such way that the processing in real time of the present software will be maintained. In this initial phase an attempt is made to emulate the TCP18 processor on a new microprocessor, by means of software.

For this purpose, a new microprocessor, under control of this software package, should imitate the TCP18 processor. Because of limitations concerning compilers, out of the MC680X0 family the MC68000 microprocessor is chosen to be the new microprocessor.

Series of measurements were performed in order to judge whether the new microprocessor, under control of the designed software, would be able to actually imitate the TCP18 processor. These measurements were meant to give insight into the frequency with which TCP18 instructions are occurring in the present software.

Subsequently was determined how much time the MC68000 microprocessor would require to execute a TCP18 instruction. The data which emerged from these measurements together with the already known execution times of TCP18 instructions, when executed by the TCP18 processor itself, and with the corresponding times for the MC68000 microprocessor which meanwhile have been computed, have been of great help in determining whether or not the MC68000 would be able to process the programs which were originally written for the TCP18, in real time.

From this investigation it was found that if the MC68000 microprocessor would replace the TCP18 processor, this would, compared to the TCP18 processor result in a delay in program execution of approximately a factor 12.

The structure of the designed emulation program, the factors which could have played a part in the realization of the considerable delay, as well as possibilities to reduce this factor, are being discussed in this report.

SAMENVATTING

Het onderwerp van mijn afstudeerwerk is te onderzoeken in hoeverre het mogelijk zou zijn, ten behoeve van ruimtebesparing, de reeds volumineuze PRX/A besturing met de TCP18 processoren, zonder enige wijziging in de bestaande software aan te brengen, door een nieuwe besturing met microprocessoren te vervangen, zodanig dat real time verwerking van de huidige software gehandhaafd blijft.

In deze eerste fase is getracht om door middel van software de TCP18 processor op een nieuwe microprocessor te emuleren.

Hiervoor is het de bedoeling dat een nieuwe microprocessor, onder besturing van dit software pakket, de TCP18 processor gaat nabootsen. Als nieuwe microprocessor is uit de MC680X0 serie, door beperkingen betreffende compilers, de MC68000 microprocessor gekozen.

Om een uitspraak te kunnen doen of deze microprocessor, onder besturing van de ontworpen software, al dan niet in staat zou zijn de TCP18 processor na te bootsen, werd een reeks metingen verricht. Deze metingen hadden tot doel inzicht te geven in de frequentie waarmee de TCP18 instructies in de huidige software voorkomen.

Vervolgens is de tijd bepaald die de MC68000 microprocessor nodig zou hebben om een TCP18 instructie uit te voeren. De uit de metingen naar voren gekomen gegevens, tezamen met de al bekende executie tijden van de TCP18 instructies door de TCP18 processor zelf en met de inmiddels berekende overeenkomstige tijden voor de MC68000 microprocessor, zijn een goede hulp geweest bij het bepalen of de MC68000 in staat zou zijn, de voor de TCP18 geschreven programma's, in real time te verwerken.

Uit dit onderzoek is gebleken dat wanneer de MC68000 microprocessor de plaats van de TCP18 processor zou innemen, dit qua programmaverwerking zou resulteren in een vertraging ter grootte van circa een factor 12 t.o.v. de TCP18 processor.

De opbouw van het ontworpen emulatie programma, de factoren die een rol kunnen hebben gespeeld bij het tot stand komen van de aanzienlijke vertragingfactor, alsmede mogelijkheden om deze factor eventueel te verkleinen, worden in dit verslag besproken.

TECHNISCHE UNIVERSITEIT DELFT

FACULTEIT DER ELEKTROTECHNIEK

Vakgroep Telecommunicatie- en Verkeersbegeleidingssystemen

Laboratorium voor AUTOMATISCHE VERKEERSSYSTEMEN

Mekelweg 4 - 2628 CD Delft

AFSTUDEEROPDRACHT

Te verrichten door: G. TSIOURAKIS

De opdracht zal worden uitgevoerd bij APT Hilversum.

Mentoren: ing. M. Konings (APT)
ir. R.A. Beukers (TU Delft)

Onderwerp: Vervanging TCP18 in PRX/A door een microprocessor.

Omschrijving van de opdracht:

Ten behoeve van kleine telefooncentrales van het type PRX/A (bijvoorbeeld de toepassing als "container centrales") zou het wenselijk zijn over de mogelijkheid te beschikken de besturing met de huidige TCP18 processoren te vervangen door een besturing met micro-processoren, dit om ruimte te kunnen besparen.

De opdracht omvat te onderzoeken of het mogelijk is om zonder enige wijziging in de huidige software, deze software te kunnen draaien op een nader te bepalen microprocessor systeem.

Er kan van worden uitgegaan dat dit een processor uit de 68000 serie is. Dit houdt onder meer in dat er in een UNIX ontwikkelomgeving een simulatie programma van de TCP18 gemaakt moet worden, dat er onderzocht moet worden of het mogelijk is deze simulatie voldoende te optimaliseren om een kleine PRX-centrale in "real time" te besturen en eventueel aanbevelingen te doen voor de in een latere fase van dit project toe te passen hardware.

Bij deze opdracht mag worden uitgegaan van een minimale processor configuratie. Dit houdt in een processor werkend in single mode zonder extended memory.

Aanvang opdracht:
November 1986

Opdrachtnummer: Go /1986/ 3
Delft, 30-1-1987
De hoogleraar,

Prof.ir. H.J. Goebertus

1. HOOFDSTUK I

DE PRX/A TELEFOONCENTRALE

1.1 Inleiding

De telefonie groeide gestaag naar volledige elektronica, waardoor steeds meer elektromechanische delen door elektronica werden verdrongen.

Bovendien deed, kort voor 1970, de software van de computerwereld haar intrede in de telecommunicatie en kwamen de eerste programmabestuurde centrales (SPC, Stored Program Controlled) op de markt.

Een daarvan is de PRX/A semi elektronische telefooncentrale die Philips sinds 1973 met veel succes verkoopt. Inmiddels zijn er al meer dan 4 miljoen PRX/A lijnen in Nederland en buitenland geïnstalleerd.

PRX/A staat voor: **Processor Control Exchange/Analoog**, d.w.z. een processor bestuurd centrale van het analoge type waarbij voor elk gesprek een eigen weg wordt opgebouwd en voor de duur van dat gesprek in stand wordt gehouden (ruimte verdeelprincipe).

Aan de telefoonadministratie biedt de PRX/A-centrale alles wat ook door de oudere automatische telefooncentrales geboden wordt. Het enige verschil voor de abonnee is wellicht de snellere verbindingsofbouw.

De PRX/A-centrale biedt echter nog meer :

In de processor bijvoorbeeld, kan per abonnee een aantal kenmerken worden opgenomen.

Daarnaast is oproeper-identificatie een essentiële eigenschap van het systeem.

Opvallend is ook, dat een onbepaald aantal willekeurige nummers tot een groepnummer verenigd kunnen worden, terwijl elk nummer uit de groep, met uitzondering van het groepnummer, desgewenst individueel bereikbaar blijft. Hierdoor zijn, bij uitbreiding van het groepnummer, nummerwijzigingen niet meer nodig.

De PRX/A-centrale strooit nooit zo kwistig met 50 Hz-telimpulsen als de bestaande centrales, doch stuurt ze vanuit de AJ alleen naar abonnees, die een kostenteller hebben.

Als het abonneekenmerk ingesteld staat op gebruik van toondruktoestel, kan toch eventueel een toestel met kiesschijf worden aangesloten, etc.

Daar dit rapport handelt over een poging om de besturing van bovengenoemde centrale te vervangen, wordt in dit hoofdstuk en hoofdstuk II getracht enig inzicht in de opbouw van de PRX/A-centrale als geheel te geven.

1.2 Opbouw van de PRX/A-centrale

De PRX/A-centrale is te verdelen in vier blokken :

1. Het schakelnetwerk.
2. Het aanpassingsnetwerk.
3. De systeemtafel.
4. De centrale besturingseenheid.

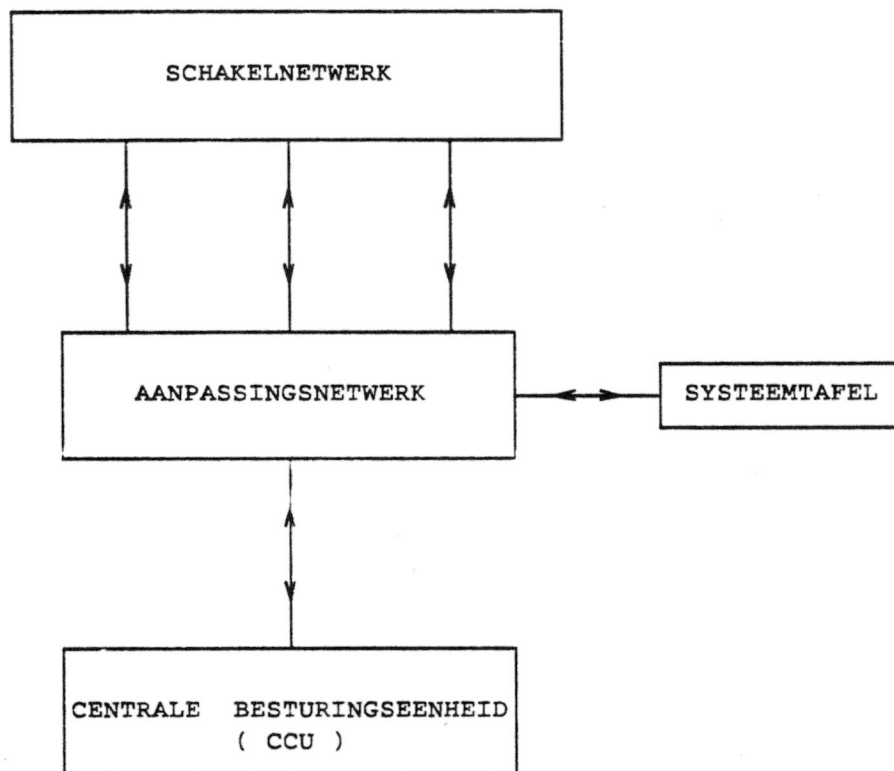


Figure 1. Blokschema van PRX/A-centrale.

1. Het schakelnetwerk

Het schakelnetwerk, opgebouwd uit reed-relais en elektronische componenten, maakt het mogelijk een verbinding tot stand te brengen tussen :

- de abonnees aangesloten op dezelfde centrales.
- verkeer van een andere centrale en abonnees van de eigen centrale.
- verkeer van de eigen centrale en abonnees van een andere centrale.
- verkeer van een andere centrale naar een andere centrale.

Het opbouwen van de verbinding gebeurt altijd met behulp van de centrale besturingseenheid CCU.

2. Het aanpassingsnetwerk

Veranderingen die plaatsvinden in het schakelnetwerk worden doorgegeven naar de centrale besturingseenheid CCU.

De TCP18 processor, een van de belangrijkste onderdelen van de CCU, is een snelle processor met een schakeltijd die in microseconden wordt uitgedrukt.

De schakeltijd van een Reed-relais in het schakelnetwerk daarentegen wordt uitgedrukt in milliseconden.

Om te voorkomen dat de processor zijn schakeltijden zelf moet bewaken is t.b.v. de snelheidsaanpassing van de processor aan het schakelnetwerk daartussen aanpassingsapparatuur aangebracht : de interface.

Dit is een buffer tussen de snelle processor en de relatief langzame reed-relais.

3. De systeemtafel

Op de systeemtafel bevindt zich de apparatuur benodigd om via het aanpassingsnetwerk allerlei informatie aan de processor te kunnen toevoeren en uit de processor informatie te kunnen aanvragen.

4. De centrale besturingseenheid

De centrale besturingseenheid bevat alle circuits en verbindingswegen die nodig zijn om de instructies uit te voeren.

Ter verduidelijking van het gedane onderzoek dat in dit rapport aan de orde komt, wordt in hoofdstuk II de centrale besturingseenheid afzonderlijk behandeld.

2. HOOFDSTUK II

CENTRALE BESTURING VAN DE PRX/A-CENTRALE

2.1 Inleiding

De centrale besturingseenheid, het "brein" van de PRX/A-centrale, bestaat uit twee delen, die aan elkaar gekoppeld zijn. Elk deel bevat een aantal eenheden die het telefonieproces besturen en controleren.

A. Processor terminal (0) en Processor terminal (1)

B. Processor (0) en Processor (1)

In dit hoofdstuk wordt ten eerste een globale indruk gegeven van het feit waarom de besturing van de PRX/A-centrale uit twee processor terminals en twee processoren bestaat.

Ten tweede, wordt een opsomming gegeven van de belangrijkste functies die de centrale besturing uitvoert.

Ten derde, volgt een korte beschrijving van alle eenheden waaruit de CCU is opgebouwd.

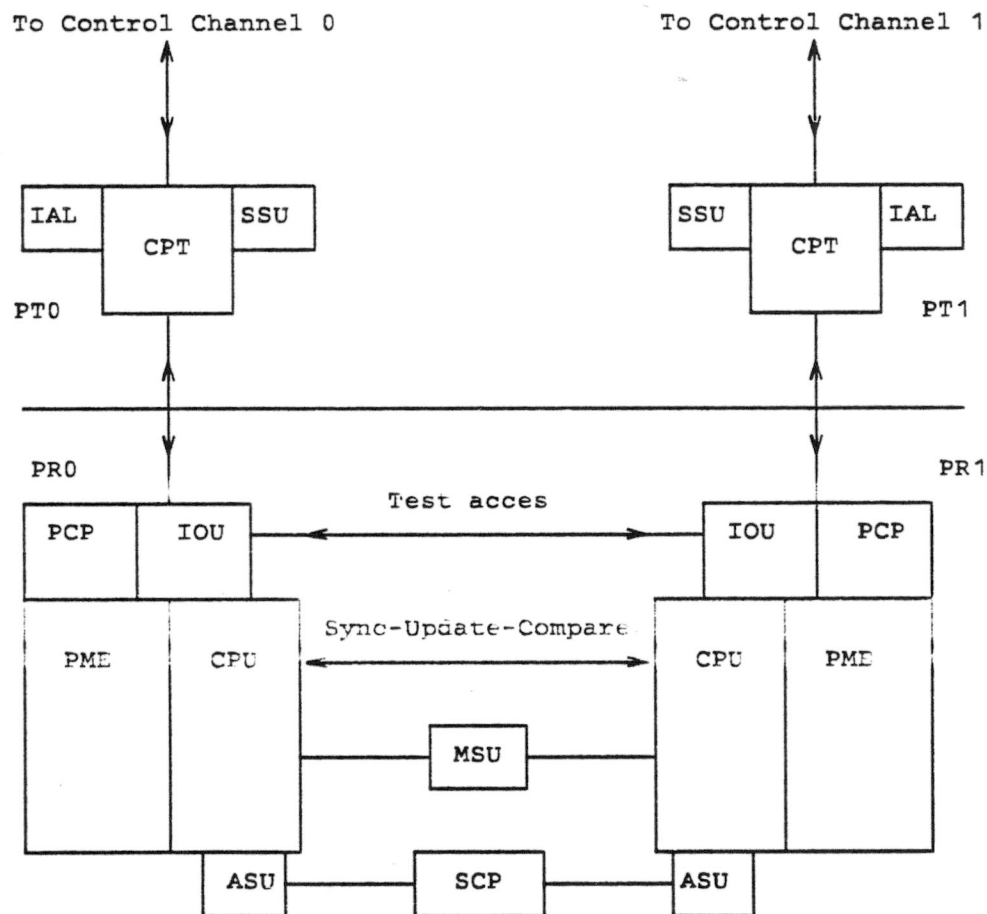


Figure 2. Centrale besturingseenheid.

2.2 Kwetsbaarheid en Betrouwbaarheid

Een probleem bij een computerbestuurde telefooncentrale is de kwetsbaarheid.

Als de centrale besturing uitvalt, is over de gehele centrale geen telefonie meer mogelijk. Om dit te vermijden is de centrale besturing opgebouwd uit twee processoren die, onafhankelijk van elkaar, in staat zijn het gehele telefonieproces te besturen.

In normaal bedrijf zijn beide processoren actief. Ze werken parallel en verwerken dezelfde informatie; deze informatie vergelijken ze continu met elkaar.

Wanneer een opdracht moet worden uitgevoerd, wordt eerst het resultaat van beide processoren vergeleken. Is dit gelijk, dan wordt de opdracht uitgevoerd.

Is dit niet het geval, dan wordt er een testprocedure gestart.

Een defecte schakelaar zal slechts een klein gedeelte van het telefoonverkeer verstoren. Daarom is het schakelnetwerk enkelvoudig uitgevoerd.

Om de betrouwbaarheid van het systeem te kunnen garanderen is het besturingskanaal, evenals de processor, gedupliceerd.

De overgang tussen de verdubbeling van besturings-apparatuur en het schakelnetwerk ligt in de interface.

2.3 Belangrijke functies van de CCU

De centrale besturingseenheid geeft o.a. opdracht tot:

- a. Het aftasten van de abonneelijnen en junctoren voor eventuele nieuwe signalen.
- b. Het verwerken van deze signalen(informatie) en het beslissen welke actie moet worden ondernomen.
- c. Opdracht geven aan relevante signalen die over de abonnee- en trunklijnen verzonden moeten worden.
- d. Het koppelen en ontkoppelen van verbindingswegen in het schakelnetwerk.

2.4 Processor terminal gedeelte

Elke processor terminal bevat de volgende eenheden :

1. CPT-Control channel processor terminal.
2. IAL-Interrupt alloter.
3. SSU-Subsysteem switching unit.

1. Control channel processor terminal

De CPT is de eenheid in de processor terminal, die verbonden is met de hoofd data bussen naar en van de CST's (Control subsystem terminals).

Er zijn maximaal 3 bussen, waarvan elk tot 10 CST's kan bedienen. De CPT is altijd met een van de drie verbonden. Verbinding met de overige bussen vindt alleen plaats wanneer het aantal CST's dit noodzakelijk maakt.

2. Interrupt alloter

De IAL is hardware apparatuur die, wanneer aan de processor interrupts worden aangeboden, de prioriteitsvolgorde bestuurt.

Slechts een paar interrupts opereren buiten de IAL, die zijn nl. direct verbonden aan de "Main sequencer". De IAL heeft 8 prioriteits niveau's (0 t/m 7 in volgorde van afnemende prioriteit) waarvan elk niveau tot 16 verzoeklijnen kan hebben, waardoor in totaal 128 mogelijke interrupts kunnen ontstaan.

3. Subsystem switching unit

De SSU unit is de eenheid die het "system assurance program" in staat stelt het systeem te beveiligen tegen interferenties, die het gevolg kunnen zijn van een defecte CPT, CST of een defect subsysteem.

De SSU wordt bestuurd door geprogrammeerde instructies, geadresseerd aan de IOU, die commando's zendt naar de SSU via de CPT.

Een belangrijke functie van de SSU is het aansturen van relais in andere apparatuur, zoals bijvoorbeeld de CST's.

2.5 Processor gedeelte

De TCP 18 processor, het zenuwcentrum van de PRX/A-centrale, is een binair enkel adres machine met een woord en instructie lengte van 16 bits.

In het processor gedeelte bevinden zich een aantal eenheden. Enkele daarvan zijn aan de processor gekoppeld en de rest behoort tot de TCP18 processor zelf.

2.5.1 Aan de processor gekoppelde eenheden

De volgende eenheden zijn aan de processor gekoppeld :

1. PCP-Processor control panel.
2. MSU-Master switch unit.
3. SCP-System control panel.

1. Processor control panel

Het PCP is hardware apparatuur die gebruikt wordt om in het geheugen van de processor te kunnen schrijven en lezen en om de processor zelf te kunnen starten of stoppen.

Dit panel bevat o.a. een aantal schakelaars die het adres en de in te schrijven data aangeven (respectievelijk adres selection, data selection), plus een aantal lampjes die aangeven welke hardware fout opgetreden is.

In het algemeen is dit panel buiten dienst, het wordt nl. alleen bij storingen in de processor gebruikt.

2. Master switch unit

De bekendste hardware apparatuur in de PRX/A is de Master switch unit. Deze unit wordt actief wanneer beide processoren hun continuïteit verliezen en probeert in dit geval een van beide processoren op te starten.

Lukt dit niet, d.w.z. de opgestarte processor verliest na enige tijd weer zijn continuïteit, dan wordt de andere opgestart. Dit herhaalt zich totdat een van de processoren in continuïteit blijft.

3. System control panel

Het SCP is de hardware apparatuur waarmee de toestand van de gehele centrale afgelezen kan worden.

De beheerder van het systeem kan via het SCP op elk gewenst moment informatie verkrijgen over de toestand van de centrale besturingseenheid, over de toestand van de control kanalen, en bovendien over alles wat er in het schakelnetwerk plaats vindt.

2.5.2 Processor registers

De TCP 18 processor bevat een aantal registers die worden ingedeeld in :

- A. Algemene registers
- B. Interne registers
- C. Speciale registers

A. Algemene registers

Algemene registers zijn de registers die alleen door het programma toegankelijk zijn.

B. Interne registers

Interne registers zijn registers voor de processor zelf en niet toegankelijk door het programma.

C. Speciale registers

Speciale registers zijn programma toegankelijke registers en worden alleen gebruikt voor interrupt afhandeling of input/output programma's.

2.5.3 Eenheden behorende tot de processor zelf

De volgende eenheden behoren tot de procesor :

- a. ASU Alarm and switchover unit.
- b. TAC Test acces circuit.
- c. SBA Standard boundary adapter.
- d. IOU Input / Output unit.
- e. PME Processor memory.
- f. CPU Central processing unit (PCU + AU).

a. Alarm and switchover unit

ASU is een veelzijdig functionele eenheid. Een van de functies van de ASU is de schakelfunctie. Met behulp van commando's, die via de IOU naar de ASU worden gestuurd, kan de CPU de configuratie schakelen. Hij schakelt de besturingskanalen in of uit en geeft opdracht voor synchronisatie met de andere processor.

Welke kanalen beschikbaar zijn, welke processor in "continuïteit" is, opereert het systeem in dual mode, alsmede fouten in de processor, worden aan de ASU gemeld en kunnen daaraan worden opgevraagd.

b. Test acces circuit

TAC is een circuit dat het mogelijk maakt om, indien nodig, de ene processor opdrachten aan de andere te laten geven en deze te besturen. Via de IOU wordt de ene processor met de TAC van de andere processor verbonden. In deze processor kan dan worden geschreven, gelezen en kan worden opgestart.

Toegang krijgen tot de TAC van de andere processor is alleen mogelijk wanneer er geen telefonieprogramma in deze processor aanwezig is. Toegang kan wel worden verkregen wanneer er een testprogramma aanwezig is, dat door de processor bestuurd wordt.

c. Standard boundary adapter

Daar in de loop van de jaren steeds nieuwe faciliteiten voor het telefonie proces ontwikkeld werden, moest noodzakelijkerwijs het geheugen worden uitgebreid.

Voor dit geval is de SBA eenheid onworpen die, het mogelijk maakt enerzijds een extern geheugen (768K) aan de oorspronkelijke 256K toe te voegen en anderzijds een aanpassing vormt om met elk type geheugen te kunnen communiceren.

d. Input - output unit

IOU is de unit die het data-transport verzorgt tussen de processor en de buitenwereld. Als de CPU een aantal taken, zoals scannen van testpunten, interrupt afhandeling etc. zelf zou moeten verrichten, zou er weinig tijd overblijven voor informatie verwerking.

Daarom is de IOU zodanig opgebouwd dat, indien nodig, deze een aantal CPU taken kan overnemen en uitvoeren. Bovendien worden enkele opdrachten autonoom door de IOU uitgevoerd en de resultaten daarvan worden rechtstreeks in het geheugen bijgewerkt.

e. Processor memory

PME is het processor geheugen, dat onderverdeeld is in BANKS.
De oorspronkelijke 4 BANKS(elke BANK 64K) zijn, door het toevoegen van extern geheugen, uitgebreid tot 16 BANKS(elke BANK 64K).
In deze BANKS wordt het gehele telefonieprogramma geladen plus informatie betreffende het schakelnetwerk.

f. Central processing unit

De Central processing unit vormt het hart van de processor.
Dit is nl. de eenheid die uit een PCU (Processing control unit) en een AU (Arithmetic unit) bestaat en zorgt voor het ophalen en het decoderen van de instructies, die vervolgens onder besturing van microinstructies worden uitgevoerd.

Uit de beschrijving van al deze besturingseenheden kan worden afgeleid dat met een aantal hiervan, te weten de PT, IOU, CPU en PME, in principe al telefonie mogelijk is.

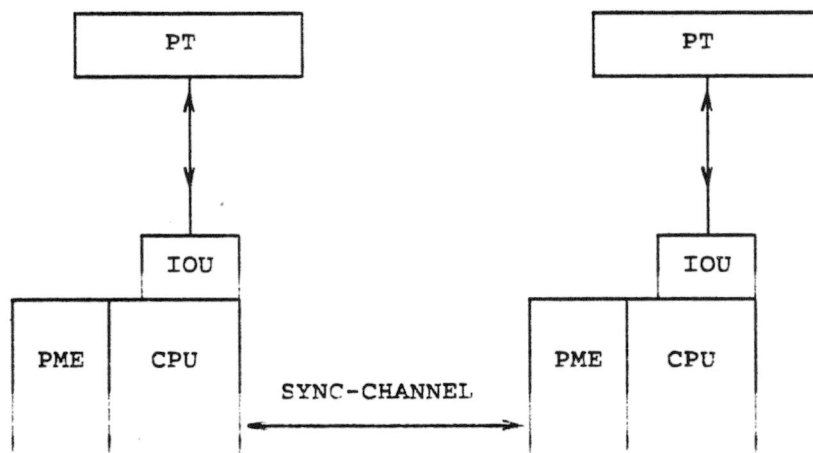


Figure 3. Schematische weergave voor mogelijke telefonie met de in 2.5.3.f genoemde eenheden.

Lijst van gebruikte afkortingen

ASU : Alarm and switchover unit.
CCU : Central control unit.
CPT : Control processor terminal.
CPU : Central processing unit.
CST : Control channel subsystem terminal.
IAL : Interrupt alloter.
IOU : Input/Output unit.
MSU : Master switch unit.
PCP : Processor control panel.
PME : Processor memory
SBA : Standard boundary adapter.
SCP : System control panel.
SSU : Subsystem switch unit.
TAC : Test acces circuit.

Hoogstwaarschijnlijk zou een nieuwe microprocessor de hierboven beschreven besturing volledig kunnen vervangen.

Door de enorme gecompliceerdheid van deze besturing zou het echter niet mogelijk zijn, binnen de mij ter beschikking staande tijd, een volledig beeld te geven van de noodzakelijke hardware- en software- aanpassingen die ongetwijfeld gemoeid zouden zijn met een dergelijke vervanging. Derhalve heb ik geprobeerd om een aanzet te geven voor de bedoelde vervanging, die eventueel in een volgende fase verder kan worden uitgewerkt.

Gezien het feit dat in de centrale besturing van de PRX/A de CPU eenheid de "bottle nek" is, wordt in deze eerste fase getracht aan te tonen in hoeverre het mogelijk is alleen de huidige TCP18-CPU door een nieuwe microprocessor te vervangen , zodanig dat het real-time verwerken van de TCP18-software gehandhaafd blijft.

Hiervoor is een programma gemaakt dat bedoeld is om een nieuwe microprocessor te besturen, zodanig dat de TCP18-CPU door deze nieuwe microprocessor kan worden nagebootst.

3. HOOFDSTUK III

BESCHRIJVING VAN HET EMULATIE PROGRAMMA

3.1 Inleiding

In dit hoofdstuk wordt het ontworpen emulatie programma beschreven dat, zoals vermeld in voorgaande paragraaf, bedoeld is om een nieuwe CPU zodanig te kunnen besturen dat de huidige TCP18 software in real-time verwerkt kan worden. Dit houdt dus in dat wanneer de TCP18-CPU en het TCP18 geheugen door een nieuwe CPU en een nieuw geheugen zullen worden vervangen, de nieuwe CPU met de overblijvende hardware, de reeds bestaande software in real-time moet blijven verwerken.

Een en ander wordt hieronder schematisch weergegeven.

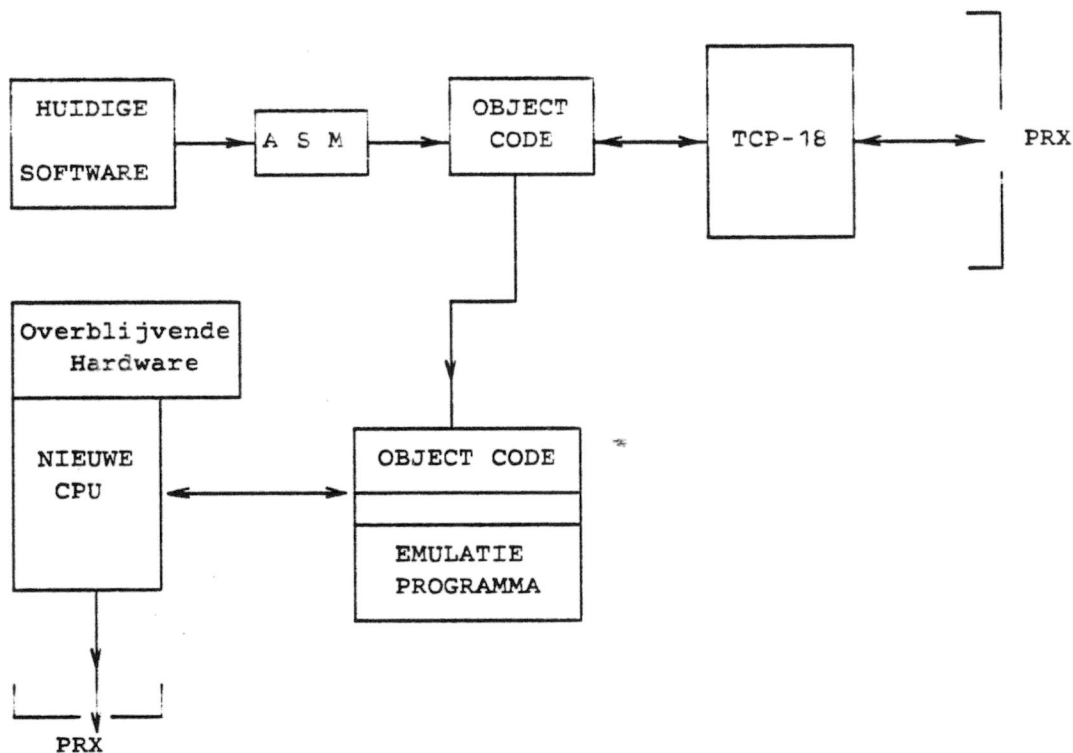


Figure 4. Schematische voorstelling van de bedoelde vervanging

Een goede inventarisatie van de TCP18 instructies alsmede van de acties die de TCP18-CPU onderneemt om elke instructie afzonderlijk uit te voeren was, voor de programma-opzet, eerste vereiste.

Daarom werd eerst de TCP18 instructieset grondig onderzocht, waarna van de, voor het opzetten van het emulatie programma, relevante gegevens een overzicht is gemaakt.

3.2 Inventarisatie en opbouw van de TCP18 instructies

De instructieset van de TCP18 Processor bestaat uit 80 instructies, waarvan 26 instructies "memory reference" instructies zijn en de rest(=54) voor control, overdracht en logische operaties is.

De tijd (acces time) nodig om gegevens uit het hoofdgeheugen op te halen is minder dan 600 nanoseconden en de tijd nodig voor een volledige lees-schrijf cyclus is 1.6 microseconden.

Alle instructies zijn 16-bits lang.

De OPCODES zijn verschillend van lengte, er zijn nl. instructies met een opcode van 5, 6, 7, 8, 12 en 16-bits.

Op grond van de OPCODE lengte worden de TCP18 instructies ingedeeld in :

- Memory reference instructies [OPC = 5 bits]
- Memory constant instructies niet gemodificeerd [OPC = 8 bits]
- Memory constant instructies gemodificeerd [OPC = 7 bits]
- Halfword instructies gemodificeerd [OPC = 7 bits]
- Data kanaal instructies [OPC = 12 bits]
- Instructies [OPC = 16 bits]

Tevens kent de TCP18 processor, voor wat de "memory reference" instructies betreft, de volgende adresseringsmodes :

- a. **Directe adressering.**
- b. **Indirecte adressering.**
- c. **Gemodificeerde adressering.**
- d. **Indirect gemodificeerde adressering.**

3.2.1 Opbouw van memory reference instructies

OPCODE (5)		n9 x2	i	b	n8 x1	n7 x4	n6 x3	n5n4n3n2n1	
16	12	11	10	9	8	7	6	5	1

b Indicator voor al of niet gemodificeerde adressering
i Indicator voor al of niet indirecte adressering
x4x3x2x1 Het memory bank nummer aangevende bits

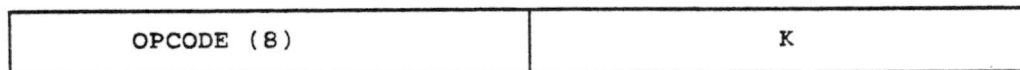
De wijze waarop het effectieve adres bij (MR) jump en niet jump instructies wordt bepaald, is bij alle typen adresseringen gelijk, behalve bij toepassing van de gemodificeerde adressering.

i	b	Effectief adres voor (MR) niet jump inst.	Bank nummer
0	0	$N = n + (R)$	R4R3R2R1
0	1	$N = n' + (B)$ ***	x4x3x2x1
1	0	$N = (n' + (R))$	x4x3x2x1
1	1	$N = (n' + (R)) + (B)$	x4x3x2x1

i	b	Effectief adres voor (MR) Jump inst.	Bank nummer
0	0	$N = n + (R)$	R4R3R2R1
0	1	$N = n + (R) + (B)$ ***	R4R3R2R1
1	0	$N = (n' + (R))$	x4x3x2x1
1	1	$N = (n' + (R)) + (B)$	x4x3x2x1

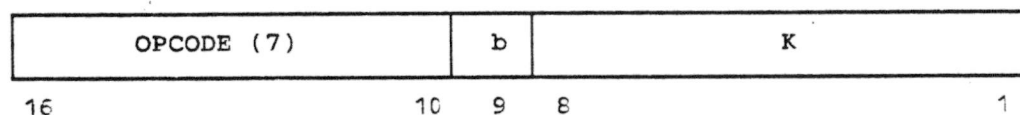
N Effectief operand adres
n n9 n8n1
n' n5...n1
R Relocatie Register R
B Index Register B
R4R3R2R1 Het memory bank nummer aangevende bits

3.2.2 Opbouw van memory constant niet gemodif. instructies



K Binaire Konstante 8 bits

3.2.3 Opbouw van memory constant gemodif. instructies

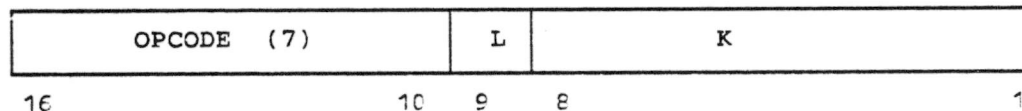


b Indicator voor al of niet gemodificeerde adressering

K Binaire Konstante 8 bits

b	Effectief adres is de konstante K
0	$K = K$
1	$K = K + (B)$

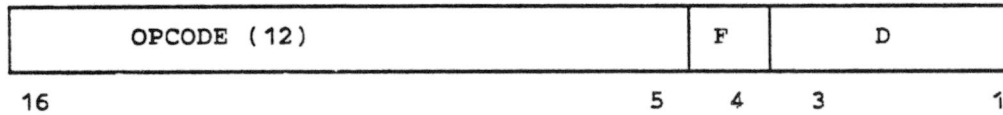
3.2.4 Opbouw van halfword gemodificeerde instructies



K Binaire konstante 8 bits

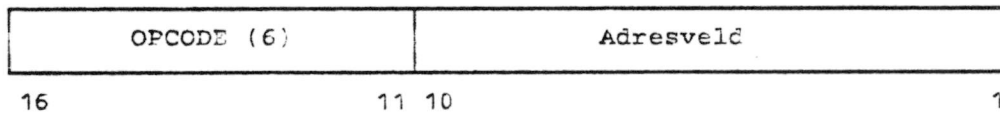
L	
0	Bits 1 t/m 8 refereren aan 1 t/m 8 bits van de Acc
1	Bits 1 t/m 8 refereren aan 9 t/m 16 bits van de Acc

3.2.5 Opbouw van data kanaal instructies



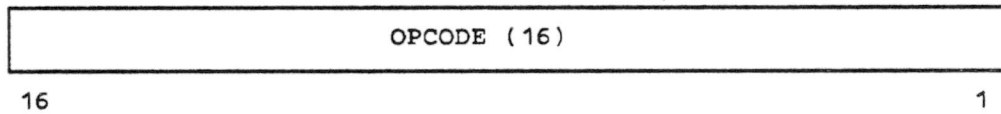
D Het subkanaal nummer aangevende bits
 F Functie bit

3.2.6 Opbouw van de LRR(Load Relocation Register) instructie



n10 n9.....n1 Absoluut operand adres
 Bij deze instructie wordt alleen directe adressering toegepast.

3.2.7 Opbouw van instructies met OPCODE 16-bits lang



De opcode is het gehele instructieveld.

3.3 Bepaling van de adresseringsmodes

Het onderstaande flow diagram illustreert het bepalen van het effectieve adres bij het toepassen van de 5 mogelijke typen adresseringen.

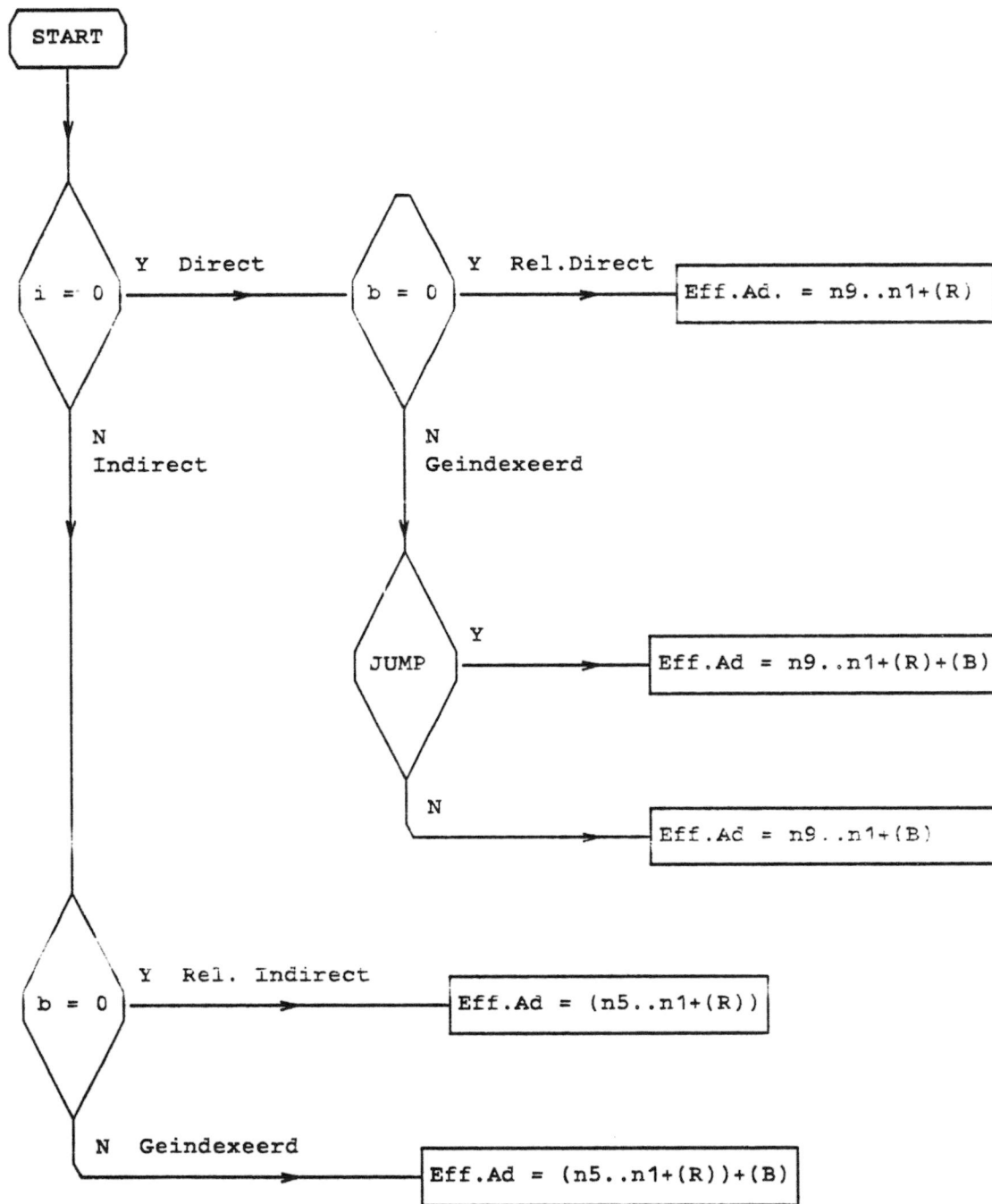
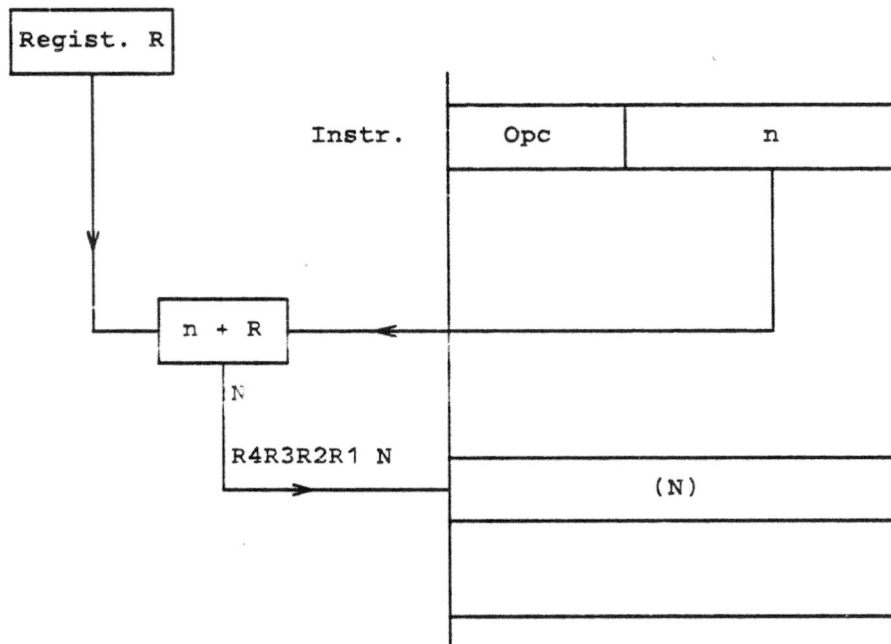


Figure 5. Flow diagram voor het bepalen van het effectieve adres bij alle adresseringsmodes

Het bepalen van het effectieve adres voor "memory reference" instructies wordt, voor elk type adressering, schematisch weergegeven in de volgende paragraaf.

3.4 Schematische weergave van het bepalen van het effectieve adres bij (MR) instructies

a. Directe adressering $I = 0$ en $b = 0$



$n = x2/n9 \ x1/n8 \ x4/n7 \ x3/n6 \ n5 \ n4 \ n3 \ n2 \ n1$

$N = 16\text{-bits effectief operand adres}$

Register $R = 16\text{-bits relocatie register}$

$R4R3R2R1 = \text{Memory bank nummer}$

Voordat de optelling $(n + R)$ plaats vindt, wordt eerst de waarde van de $R4R3R2R1$ bits van het relocatie register bepaald en pas daarna worden ze, voor de optelling, 0 (nul) gemaakt.

R16	R15	R14	R13	R12	R11	R10	R9	R8	R7	R6	R5	0	0	0	0					
												n9	n8	n7	n6	n5	n4	n3	n2	n1
												+								
												=								
Effectief Adres																				

N.B.

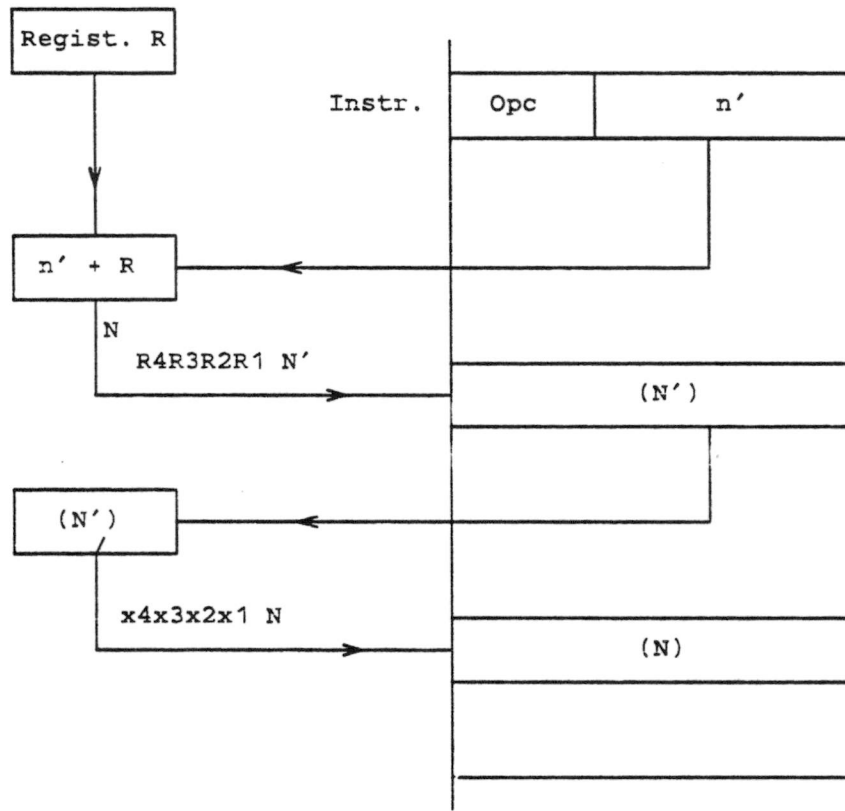
In de huidige situatie is dus het effectieve adres 16-bits lang. In het emulatie programma daarentegen wordt het effectieve adres 20-bits. Dit is nodig om alle locaties van het 1M woorden gereserveerde geheugen regelrecht te kunnen aanspreken.

Derhalve gebruik ik, voor het bepalen van dit 20-bits lange adres, de $R4R3R2R1$ bits respectievelijk als 20ste 19e 18e en 17e bit.

In het emulatie programma geldt dus voor N :

$N = R4R3R2R1 \ R16 \dots (R9+n9) \dots (R5+n5) n4 n3 n2 n1 = 20\text{-bits}$

b. Indirecte adressering $l = 1$ en $b = 0$



$n' = n5 \ n4 \ n3 \ n2 \ n1$

$N' = 16\text{-bits relatief operand adres}$

$N = (N')16\text{-bits finale effectief operand adres}$

Register R = 16-bits relocatie register

R4R3R2R1 = Memory bank nummer

$x4x3x2x1$ = Finale memory bank nummer

N.B.

Voor dit type adressering worden, voor het bepalen van het uiteindelijke effectieve adres, niet de R4R3R2R1 bits respectievelijk als 20ste 19e 18e en 17e bit gebruikt, maar de $x4x3x2x1 (=n7n6n9n8)$ bits van het instructie veld.

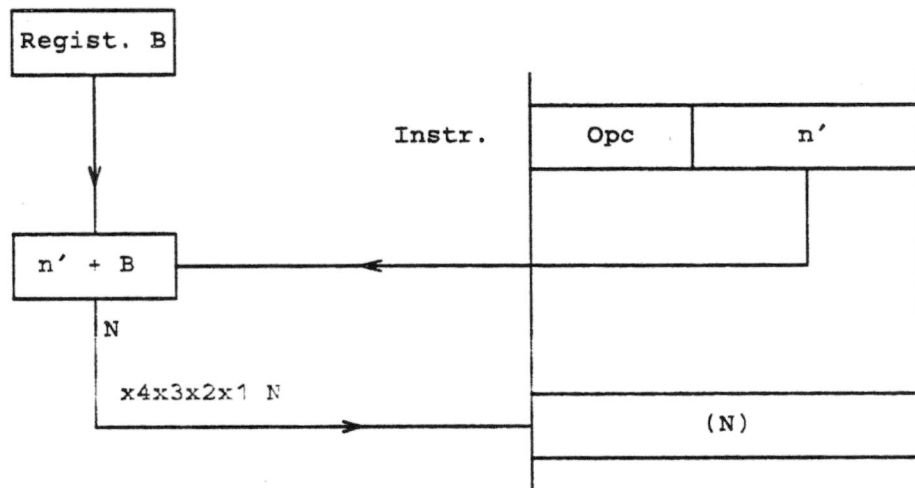
In het emulatie programma geldt dus voor N' en N :

$N' = R4R3R2R1 \ R16... (R5+n5)n4n3n2n1 = 20\text{-bits}$

$N = x4x3x2x1 (\quad (N') \quad) = 20\text{-bits}$

c. Gemodificeerde adressering voor (MR) niet jump instructies

$i = 0$ en $b = 1$



$n' = n5 \ n4 \ n3 \ n2 \ n1$

$N = 16\text{-bits effectief operand adres}$

Register B = 16-bits index register

$x4x3x2x1$ = Memory bank nummer

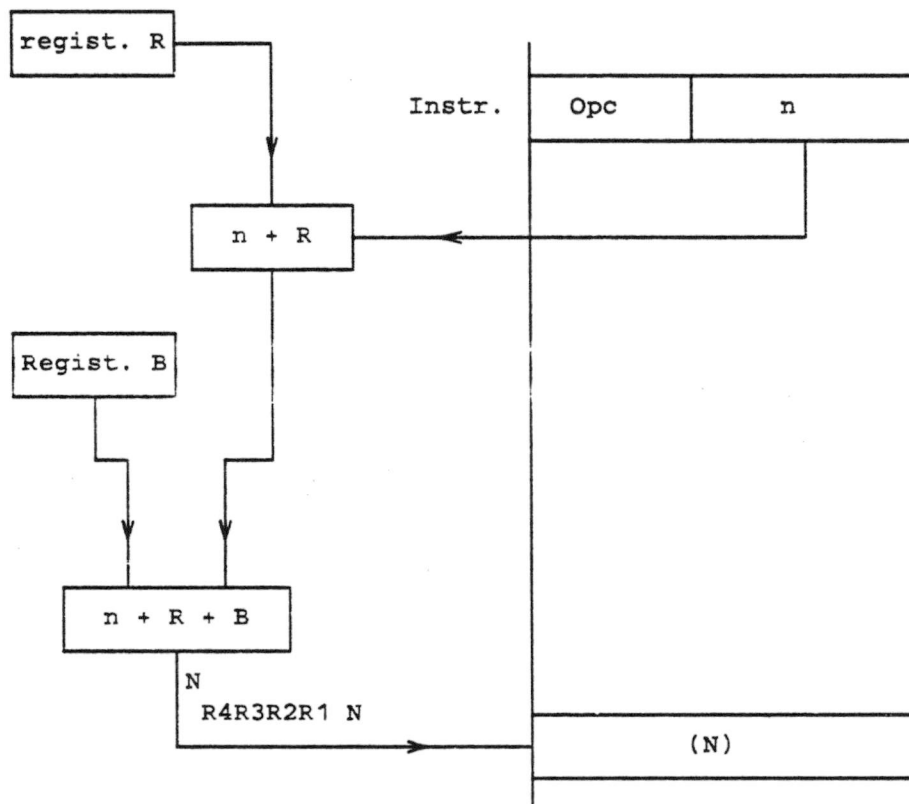
N.B.

In het emulatie programma geldt voor N :

$N = x4x3x2x1 \ b15...(B5+n5)...(b1+n1) = 20\text{-bits}$

d. Gemodificeerde adressering voor (MR) jump instructies

$i = 0$ en $b = 1$



$n = x2/n9 \ x1/n8 \ x4/n7 \ x3/n6 \ n5 \ n4 \ n3 \ n2 \ n1$

$N = 16\text{-bits effectief operandadres}$

Register R = 16-bits relocatie register

Register B = 16-bits index register

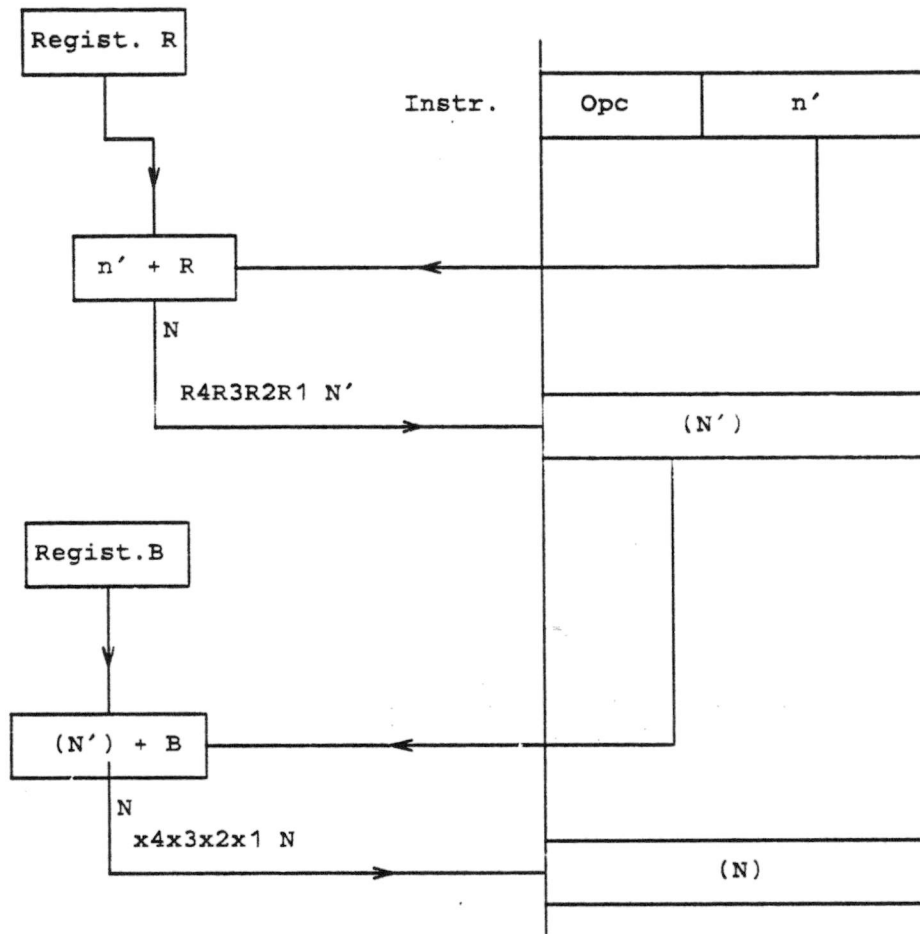
$R4R3R2R1$ = Memory bank nummer

N.B.

in het emulatie programma geldt voor N :

$N = R4R3R2R1 (R16+B16).....(B9+(R9+n9))..(B5+(R5+n5))..(B1+n1)$

e. indirect gemodificeerde adressering l = 1 en b = 0



$n' = n5 \ n4 \ n3 \ n2 \ n1$

$N' = 16\text{-bits relatief operand adres}$

$N = (N') \ 16\text{-bits finale effectief operand adres}$

$R4R3R2R1 = \text{Memory bank nummer}$

$x4x3x2x1 = \text{Finale memory bank nummer}$

N.B.

In het emulatie programma geldt voor N' en N :

$N' = R4R3R2R1 \ R16 \dots (R5+n5)n4n3n2n1 = 20\text{-bits}$

$N = x4x3x2x1 \ (\ ((N') + B) \) = 20\text{-bits}$

3.5 Algemene programma-opzet

Het ontworpen emulatie programma is gebaseerd op de onderstaande structuur.

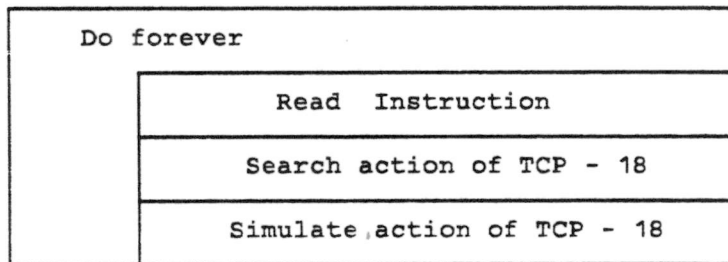


Figure 6. Programma structuur

Voor het bereiken van de uiteindelijke vorm is het emulatie programma een aantal maal gewijzigd. Deze wijzigingen waren nodig om een zo snel mogelijke uitvoering van het programma te realiseren.

Een functionele indeling van de door de CPU ondernomen acties bij het uitvoeren van een instructie, ziet er als volgt uit :

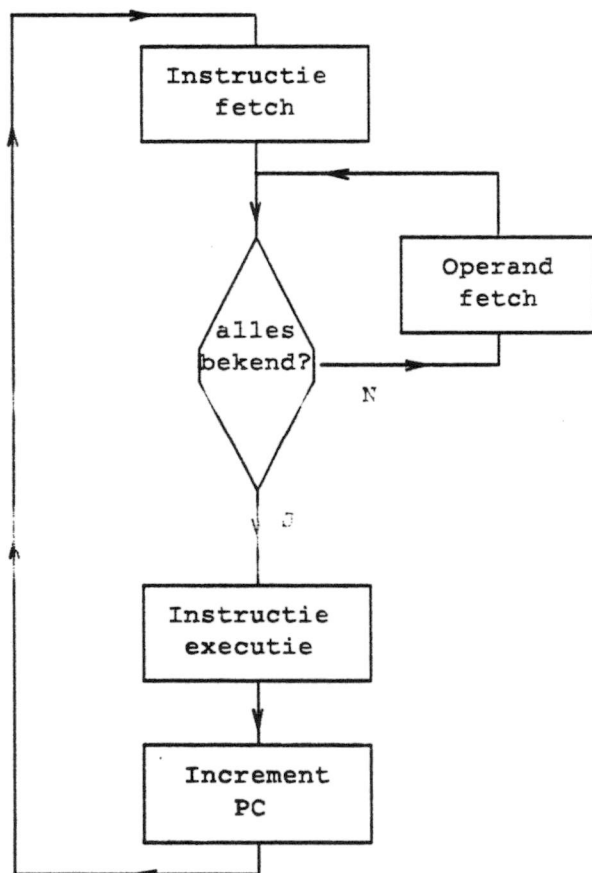


Figure 7. Flow diagram van een instructieverwerking

3.5.1 Gebruikte variabelen in het emulatie programma

In het emulatie programma worden de volgende variabelen gebruikt, die het volgende voorstellen :

Inst	: de uit te voeren instructie.
regR	: relocatie register R.
regB	: index register B.
regQ	: auxiliary register Q.
Acc	: accumulator.
pc	: program counter.
addr	: effectief operand adres.
regZ	: 8 bits interrupt mask register Z om het optreden van bepaalde soorten onderbrekingen te voorkomen.
OF	: 1 bit indicator die geset wordt wanneer een overflow optreedt.
BI	: 1 bit indicator. Wanneer dit bit geset is, zijn alle interrupts geblokkeerd. Is dit bit gereset, dan worden die interrupts toegelaten die overeenstemmen met de inhoud van register Z.
BY	: 1 bit indicator. Dit bit wordt geset wanneer het data kanaal, door een van de eenheden, bezet wordt gehouden.
MH18	: 1 bit indicator welke aangeeft of het protectie bit van de geheugen locatie geset of gereset is.
LLPR	: 1 bit indicator welke aangeeft of de 17e bit van een bepaalde geheugen locatie (protectie bit), de waarde van MH18 heeft aangenomen.
bank	: de waarde van de x4x3x2x1 memory bank bits.
R0	: de waarde van de R4R3R2R1 bits van het relocatie register R.
times	: het aantal keer dat, bij het uitvoeren van een shift instructie, geschoven moet worden.
sum, inhml, x, y, z, x2	: variabelen die bij verschillende logische operaties worden gebruikt.

m[1000000] : een array van een miljoen integers.

p[1000000] : een array van ook een miljoen integers, waarbij de inhoud van een bepaalde adreslocatie van deze geheugen ruimte aangeeft of de overeenkomstige adreslocatie van de geheugen ruimte, die door array m in beslag wordt genomen, al of niet protected is.

(*tabel[65535])(): een pointer naar een functie. Dit wil zeggen dat de waarde van de variabele, bv. tabel[7834 Hex], het begin adres van een functie is.

3.5.2 Tests benodigd voor het uitvoeren van een instructie

Voordat een instructie uitgevoerd kan worden moet eerst bekend zijn om welke instructie het gaat.

Blijkt bij decodering dat het om een memory reference instructie gaat, dan moet, voor het bepalen van het effectieve adres, nog getest worden op :

1. de i en b bits.
2. de x2(n9)x1(n8)x4(n7)x3(n6) memory bank bits.

Testen op de i en b bits is nodig omdat de waarde van deze bits aangeeft welk type adressering moet worden toegepast, en wel als volgt :

i = 0 b = 0	directe adressering
i = 0 b = 1	gemodificeerde adressering
i = 1 b = 0	indirecte adressering
i = 1 b = 1	indirect gemodificeerde adressering

Testen op de x2x1x4x3 bits is ook nodig omdat deze bits, in de huidige situatie, het memory bank nummer aangeven.

In het emulatie programma daarentegen worden zij, bij het bepalen van het effectieve adres, in de volgorde x4x3x2x1 respectievelijk als 20ste 19e 18e en 17e bits gebruikt.

Om dit soort testen te vermijden, zij zouden immers een aanzienlijke CPU tijd vergen, is gebruik gemaakt van een tabel van 64K.

De wijze waarop deze tabel wordt gebruikt en waarom daarbij deze testen niet behoeven te worden gedaan, komt verderop in dit hoofdstuk aan de orde.

3.5.3 Gedefinieerde functies voor het emulatie programma

Voor het emulatie programma zijn in totaal 1238 functies gedefinieerd. Hiervan betreffen 1232 functies de 80 TCP18 instructies zelf, 1 (een) functie betreft de tabelinitialisatie en de rest (mod0 (), mod1 (), mod11 (), mod2 (), mod3 ()) betreft de functies waarin het effectieve adres wordt bepaald, op grond van de 4 (vier) adresseringstypen, te weten :

in mod0 (); wordt het effectieve adres voor memory reference instructies bepaald door toepassing van directe adressering.

in mod1 (); wordt het effectieve adres voor memory "niet jump" instructies bepaald door toepassing van gemodificeerde adressering.

in mod11 (); wordt het effectieve adres voor memory "jump" instructies bepaald door toepassing van gemodificeerde adressering.

in mod2 (); wordt het effectieve adres voor memory reference instructies bepaald door toepassing van indirecte adressering.

in mod3 (); wordt het effectieve adres voor memory reference instructies bepaald door toepassing van indirect gemodificeerde adressering.

Voor elk van de 20 memory reference "niet jump" instructies zijn 49 functies gedefinieerd. Binnen elk van deze functies staat weliswaar dezelfde instructie geemuleerd, de gegevens echter, benodigd voor het uitvoeren van de instructie, verschillen van elkaar. Dat wil zeggen dat de waarde van de x4x3x2x1 bits, nodig voor het bepalen van het 20 bits effectieve adres, alsmede de waarde van de i en b bits, in elk van deze 49 functies verschillend zijn. Elke functie wordt gekenmerkt door de naam van de instructie die binnen deze functie geemuleerd staat, alsmede door de waarde van de i en b bits en door de waarde van de x4x3x2x1 bits. Een en ander wordt hieronder toegelicht aan de hand van de functie **cmp14 ();**.

cmp : geeft aan dat het om de instructie cmp gaat.

i : geeft aan dat de waarde van de i en b bits, in het adresveld van de instructie cmp, respectievelijk 0 en 1 zijn.

4 : geeft aan dat de waarde van de x4x3x2x1 bits, in het adresveld van de instructie cmp, respectievelijk 0 1 0 0 zijn.

Voor elk van de 5 memory reference "jump" instructies zijn op gelijke wijze 34 functies gedefinieerd. Dit zijn 15 functies minder dan het memory reference "niet jump" instructies gedefinieerde aantal en dit komt doordat de x4x3x2x1 bits, bij toepassing van gemodificeerde adressering, niet bekend hoeven te zijn.

Bij "jump" instructies en bij toepassing van gemodificeerde adressering, ziet het effectieve adres er als volgt uit :

$$N = R4R3R2R1 (R16+B16).....((R9+n9)+B9)..((R5+n5)+B5)...(B1+n1)$$

Zie paragraaf 3.4(d.).

Tot slot, voor elke "memory constant" en "halfword" instructie zijn 2 functies gedefinieerd. In de ene functie wordt rekening gehouden met de inhoud van index register B en in de andere niet.

Dit grote aantal gedefinieerde functies valt, wat geheugen ruimte betreft, tegen. Echter, dit nadeel wordt ten dele teniet gedaan door het feit dat, qua instructie decodering, aanzienlijke CPU tijd bespaard wordt.

3.5.4 Programmaverloop bij executie van een instructie

Het programmaverloop bij executie van een instructie wordt hieronder toegelicht aan de hand van een memory reference instructie, te weten de LDA instructie.

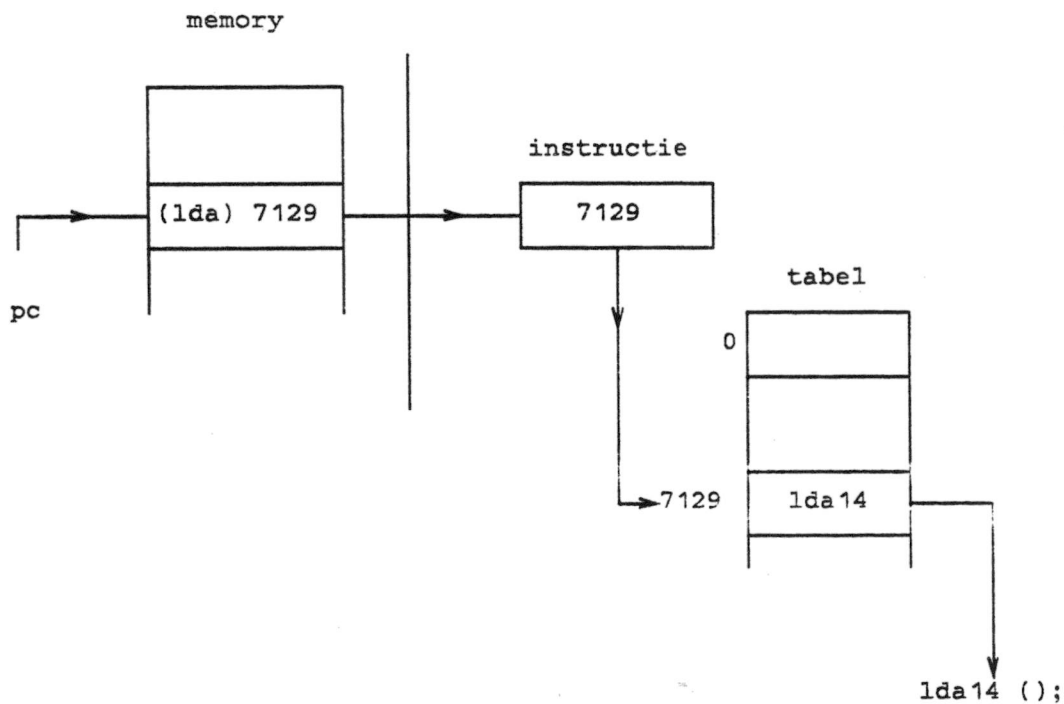
De OPCODE van deze instructie is 70 Hex., in binaire vorm 01110, 5 bits lang.

Verondersteld wordt dat deze instructie van in totaal 16 bits, de waarde 7129 Hex. heeft. Het instructieveld ziet er dan als volgt uit :

O P C O D E	x2/n9	i	b	x1/n8	x4/n7	x3/n6	n5	n4	n3	n2	n1
0 1 1 1 0	0	0	1	0	0	1	0	1	0	0	1

Daar de Hex. waarde van het adresveld (x2/n9 i bn1) van een memory reference instructie een van de waarden van 0 Hex. t/m 7FF Hex. kan hebben, werd door mij, bij elk van deze mogelijke waarden, de waarde van de i en b bits alsmede de waarde van de x-bits (in de volgorde x4x3x2x1) van te voren bepaald. Zou dus deze LDA instructie, met adresveld 129 Hex., moeten worden uitgevoerd, dan is voor deze instructie, in de functie waar ze geemuleerd staat, al bekend dat de i en b bits respectievelijk 0 en 1 zijn en dat de x4x3x2x1 bits respectievelijk 0100 zijn. Dit wil zeggen dat de gemodificeerde adressering moet worden toegepast en dat de 20ste 19e 18e en 17e bits van het finale operand adres, respectievelijk 0100 moeten zijn.

Wordt nu de LDA instructie (7129 Hex.) uit geheugen opgehaald, dan is deze gehele instructie het adres van een van de 64K tabellocatie. De inhoud van deze (7129 Hex.) tabellocatie is dan het begin adres van de functie **lda14 ()**, waarin de lda instructie (7129 Hex.) geemuleerd staat.



In het emulatie programma heb ik geen rekening gehouden met de pariteit. Wel heb ik rekening gehouden met al of niet protected instructies en geheugenlocaties.

Op elke instructie wordt nl. getest of zij al dan niet protected is. Is een instructie protected, dan wordt zij uitgevoerd. Is een instructie daarentegen niet protected, dan wordt zij niet uitgevoerd. Een poging om een niet protected instructie toch uit te voeren, zal leiden tot een "word protection error interrupt".

Elke geheugenlocatie wordt, alvorens daarin iets te schrijven, op protectie getest. Alleen wanneer de betreffende geheugenlocatie "niet protected" is kan erin geschreven worden. Een poging om in een protected geheugenlocatie te schrijven, zal leiden tot een "memory protection error interrupt".

3.5.5 Niet of gedeeltelijk geemuleerde instructies

Een aantal instructies is voor strikt laboratorium gebruik en deze instructies zijn derhalve niet geemuleerd. Het betreft de volgende instructies :

HLT (Halt)

TRC (Trace)

SPM (Simulate memory parity error)

TSA (Test program switch A)

TSB (Test program switch B)

TSC (Test program switch C)

ISC (Initiate Sub-Channel)

RCS (Request channel status)

Tevens zijn er instructies die gedeeltelijk geemuleerd zijn, nl. de I/O instructies.
De PRX/A kent m.b.t. de I/O 4 (vier) instructies :

OTA (Output data from Acc)

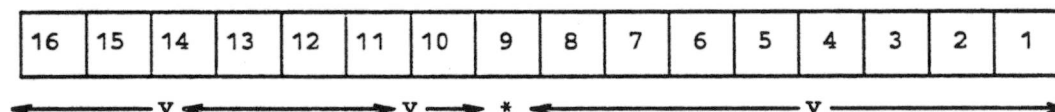
INA (Input data to Acc)

OTF (Output function code)

INF (Input response code)

Voor het verkeer met de interface zijn slechts de INA en OTF instructies van belang. Hieronder volgt een beknopte beschrijving van de betekenis alsmede van de afhandeling van een I/O instructie.

Een I/O instructie ziet er als volgt uit:



In elke I/O instructie bepalen twee bits, bit - 11 en bit - 10, de aard van een I/O instructie.

Bit 11 = 0 ==> informatie naar de processor toe.

Bit 11 = 1 ==> informatie van de processor af.

Bit 10 = 0 ==> operationele instructie.

Bit 10 = 1 ==> functionele instructie.

Bit - 9 is bepalend voor het finale adres van het subsysteem.

Bit 9 = 0 ==> adres wordt bepaald door bits 1 t/m 8.

Bit 9 = 1 ==> >> >> >> >> >> (1 t/m 8) + (regB).

Bits 12 t/m 16 hebben een vaste waarde die aangeeft dat het om een I/O instructie gaat.

Afhandeling van een I/O instructie

De processor leest in de instructie het adres van het subsysteem om te bepalen waar de opdracht naar toe moet en haalt vervolgens de opdracht uit de Acc. En dan ?. Dan kan de processor zonder de IOU eenheid niets doen. De processor stuurt informatie naar de IOU met het verzoek om het verder te regelen.

De IOU bevat onder andere een adres en een data register van respectievelijk 9 en 20 bits breed.

In het adres register wordt het adres van het subsysteem gezet en in het data register de inhoud van de Acc plus de bits 11 en 10 van de instructie, zodat het subsysteem na de decodering kan herkennen om wat voor soort instructie het gaat.

Nadat de IOU de pariteit heeft berekend, genereert ze een SYNC-out signaal om alle CST's (Control - Channel Subsysteem Terminals) te waarschuwen dat er informatie op het besturingskanaal aanwezig is en dat ze klaar moeten staan om aan het werk te gaan.

Bovendien houdt de IOU in de gaten of er een antwoord uit de INTERFACE terugkomt. Als er geen antwoord op tijd komt, wordt de processor hiervan op de hoogte gesteld.

Verondersteld wordt dat, op de een of andere manier, de informatie de gewenste CST bereikt.

Na ontvangst van het SYNC - out signaal, wordt, door de CST, het adres van het subsysteem op pariteit getest en vervolgens gedecodeerd.

Zowel de bits 11 en 10 van de instructie als data staan op de BUS. Het subsysteem weet nu op grond van de bits 11 en 10 wat voor instructie het is en na controle op pariteit in de data, voert het de opdracht uit.

Geven de bits 11 en 10 aan dat het om een INA of OTF gaat, dan stuurt het subsysteem een bevestigingssignaal naar de desbetreffende CST. De CST pakt de bevestiging op, controleert deze op pariteit en stuurt een SYNC - in signaal naar de processor toe, om hem te laten weten dat er data op het besturingskanaal aanwezig zijn.

Via het data register van de IOU komt de informatie tenslotte in de Acc en dan is de afhandeling voltooid.

Controle op pariteit van het ontvangen bericht vindt in eerste instantie plaats in de CPT (Control - Channel Processor Terminal) en daarna in de IOU. Dit, omdat in het geval dat de IOU van de ene processor verbonden is met de CPT van de andere processor, er op deze verbinding ook fouten kunnen optreden.

Daar uit het bovenstaande blijkt dat de aard van een I/O instructie eigenlijk buiten de processor wordt herkend, nl. door de betreffende CST. ben ik, bij het emuleren van deze 4 (vier) I/O instructies, niet verder gegaan dan alleen het bepalen van het adres van het subsysteem waarvoor informatie bestemd is.

Dit programma is geschreven in C-taal en er werd gebruik gemaakt van editors, compilers en disassemblers die allen werken onder een UNIX operating system.

Om de correctheid van dit emulatie programma te controleren, worden allerlei invoerwaarden, die de computer voor de uitvoering van het emulatie programma nodig heeft, tijdens de programma uitvoering (interactieve werking) aangeboden.

4. HOOFDSTUK IV

NIEUW MICROPROCESSOR SYSTEEM EN ZIJN PRESTATIES

4.1 Inleiding

Het vervangen van de huidige besturing van de PRX/A door een nieuwe microprocessor, zou alleen van nut zijn, indien deze nieuwe microprocessor in staat zou zijn de huidige telefonie programma's, zonder enige wijziging, in real-time te verwerken.

Aangezien de TCP18 processor een real-time processor is, moet de te kiezen microprocessor ook in staat zijn de bestaande telefonie programma's te verwerken op een dusdanig snelle wijze, dat de resultaten daarvan op tijd beschikbaar zijn om het lopende telefonieproces te beïnvloeden.

Daarom moet de voor deze vervanging te gebruiken microprocessor over tenminste een aantal eigenschappen beschikken die de keuze kunnen rechtvaardigen.

In dit hoofdstuk wordt een beknopte toelichting gegeven op de keuze van de nieuwe microprocessor, waarna uitgebreider wordt ingegaan op de manier waarop zijn prestaties zijn bepaald. Tot slot worden aan de hand hiervan enkele conclusies getrokken en aanbevelingen gedaan.

4.2 Gekozen nieuwe microprocessor

De mening bestaat dat de MC680X0-serie de meest geschikte kandidaat is voor de "Central Control Processor" van de PRX/A, vanwege een aantal eigenschappen zoals :

- De hoge snelheid (32-bits, maximaal 16 MHz klokfrequentie).
- De mogelijkheid een groot geheugen regelrecht aan te spreken.
- De sterke instructieset en de redelijke kosten.
- De VME-bus compatibiliteit van deze serie, etc.

Dit zijn uitgangspunten geweest om, in deze fase van dit onderzoek, een microprocessor uit deze serie te kiezen.

Omdat bij APT alleen compilers voor de MC68000 microprocessor beschikbaar zijn, is voor de vervanging van de TCP18 CPU gekozen voor de MC68000 microprocessor uit de MC680X0-serie. Zie figuur 8.

Dit is een 16-bits microprocessor met 23-bits voor woord-adressering en maximale kloktrequentie 12,5 MHz.

Daar bij APT drie versies van compilers voor de MC68000 beschikbaar zijn, is het emulatie programma op alle drie de versies gecompileerd.

Bij een versie (MC68cc, versie 3.1 van 06/01/1984) bevat de gegenereerde assemblercode minder instructies. Om deze reden is het uiteindelijke emulatie programma door deze versie gecompileerd.

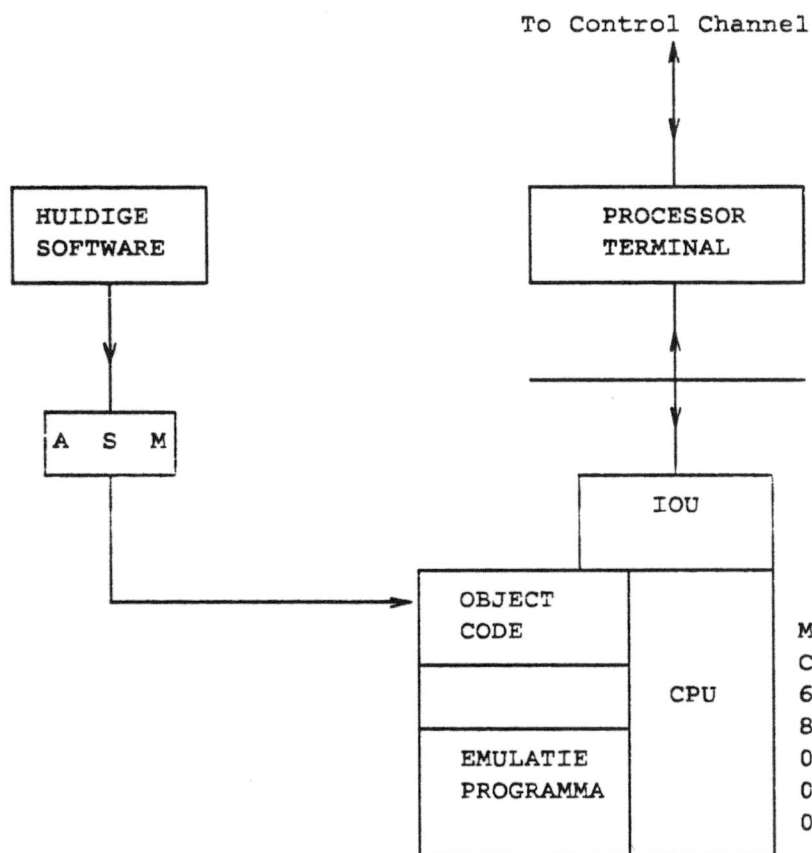


Figure 8. Systeem configuratie in single mode met MC68000 microprocessor

4.3 Bepaling van de prestaties van een processorbestuurd systeem

Voor het bepalen van de prestaties van een systeem, kunnen verschillende definities en formules worden gehanteerd.

Een eenvoudige manier om een indicatie te verkrijgen over de prestaties van een processor systeem, is door bv. het vermogen (capaciteit) van het systeem te meten. Dit kan door te bekijken hoe groot het aantal basisbewerkingen is dat de CPU eenheid van de processor in een seconde kan uitvoeren.

Het zal duidelijk zijn dat deze maatstaf weinig zegt over de feitelijke prestaties van het totale systeem, dat immers uit een aantal componenten is opgebouwd. Juist de wijze waarop deze componenten samen functioneren bepaalt de werkelijke prestatiemogelijkheden van het systeem.

De enig juiste manier, om de prestatieverhouding tussen twee computersystemen te bepalen, zou zijn door deze systemen in individuele situaties te vergelijken.

Ondanks het feit dat deze wijze de meest complexe is, worden hier, voor het bepalen van de prestaties van de EMULATOR t.o.v. de TCP18, deze beide werkwijzen gedeeltelijk gevolgd.

4.4 Gestelde eisen aan het nieuwe microprocessor systeem

Aan het te gebruiken nieuwe microprocessor systeem, dat de huidige besturing gaat vervangen, zijn de volgende eisen gesteld :

1. In staat zijn de bestaande software ZONDER wijziging in real-time te verwerken.
2. Minder ruimte vergend en te realiseren tegen redelijke kosten.

Echter, bij het beoordelen van de prestaties van een systeem, moet niet worden uitgegaan van de specificaties van de apparatuur maar van de mate waarin het systeem aan de meest essentiële eis voldoet.

Desondanks worden in het vervolg beide gestelde eisen atzonderlijk behandeld, zij het met meer nadruk op eis (1).

Het al dan niet voldoen van het gekozen microprocessor systeem (MC68000) aan de gestelde eisen, komt aan de orde bij de *CONCLUSIES EN AANBEVELINGEN*

ad. 1 Real-time verwerking

Hoewel het niet mogelijk is in deze fase programma's, geschreven voor de TCP18, op de MC68000 te laten uitvoeren, het nieuwe microprocessor systeem is immers niet geïmplementeerd, is het toch mogelijk een beeld te verkrijgen over het feit of de MC68000 al dan niet in staat is TCP18 programma's in real-time te verwerken.

Dit beeld kan nl. worden verkregen aan de hand van de verhoudingsfactor in executie tijd tussen de nieuwe en de oude CPU.

Hieronder volgt een definitie waar ik, voor het bepalen van de (VF) tussen twee systemen, van ben uitgegaan.

Definitie

Verhoudingsfactor is een factor die de verhouding in executie tijd tussen twee processor systemen aangeeft wanneer aan deze twee processor systemen dezelfde hoeveelheid "instructie mix" ter verwerking wordt aangeboden.

$$VF = \frac{\text{Totale executie tijd van de instructie mix door SYSTEEM 1}}{\text{Totale executie tijd van de instructie mix door SYSTEEM 2}}$$

De totale executie tijden van de TCP18 en van de EMULATOR (MC6800 + emulatie programma), benodigd om een gegeven hoeveelheid instructie mix uit te voeren, zijn dus de grootheden die bepaald moeten worden.

Om deze twee grootheden te kunnen bepalen is het in eerste instantie noodzakelijk over de volgende gegevens te beschikken :

- i. De door de EMULATOR benodigde executie tijd voor het uitvoeren van elke TCP18 instructie afzonderlijk.
- ii. De door de TCP18 processor benodigde executie tijd voor het uitvoeren van elk van zijn instructies afzonderlijk.

ad. i

Door gebruik te maken van de MC68clock disassembler is het aantal cycli per instructie af te leiden dat de EMULATOR nodig heeft om een TCP18 instructie uit te voeren.

Daarom is, naast de door de compiler gegenereerde assemblercode, tevens een MC68clock listing opgevraagd. Aan de hand van deze listing is het aantal cycli per instructie opgeteld en vervolgens vermenigvuldigd met de tijd van de MC68000 klok cyclus (= 80 nanoseconden).

Door deze berekening voor elke instructie afzonderlijk te doen, is een tabel opgesteld (zie APPENDIX A, par. 0.1), waarin de door de EMULATOR benodigde executie tijd per instructie te zien is. Bovendien werd voor deze berekening uitgegaan van de zgn. slechte toestand, d.w.z. de toestand waarbij het uitvoeren van een instructie de meeste tijd vergt.

Het volgende voorbeeld dient als toelichting:

De door de EMULATOR benodigde tijd om bv. de RESTORE (Direct) instructie uit te voeren wordt als volgt bepaald.

Instructie fetch : 242 cycli x 80 nanosec. = 19,36 microsec.

Operand fetch : 358 cycli x 80 nanosec. = 28,64 microsec.

Werkelijke executie : 640 cycli x 80 nanosec. = 51,20 microsec.

Executie tijd voor RESTORE instructie totaal : 99,20 microsec.

ad. II

De executie tijd van elke TCP18 instructie door de TCP18 processor zelf (zie APPENDIX A, par. 0.1) is bekend.

Omdat niet te bepalen is hoeveel keer, bij de uitvoering van een TCP18 SHIFT instructie geschoven wordt, heb ik, zowel voor de long als voor de short shift instructies, een gemiddelde executie tijd bepaald uitgaande van bestaande formules en een door mij gedane veronderstelling.

Door mij is nl. aangenomen dat, wanneer een shift instructie binnen een willekeurig gekozen tijdsinterval een aantal keer uitgevoerd wordt, elk mogelijk aantal maal shiften (0 maal shiften, 1 maal shiften,...,15 maal shiften) even vaak voorkomt.

Voorbeeld :

Stel dat de instructie SRQ (shift right register Q) binnen een willekeurig gekozen tijdsinterval 256 maal is uitgevoerd. Volgens mijn veronderstelling wil dat zeggen dat :

0 maal rechts schuiven, 16 maal is voorgekomen.

1 maal rechts schuiven, 16 maal is voorgekomen.

2 maal rechts schuiven, 16 maal is voorgekomen.

enz.

15 maal rechts schuiven, 16 maal is voorgekomen.

Bepaling van de gemiddelde executie tijd voor "SHIFT" instructies

Voor short shift instructies geldt:

$$\begin{aligned} 0 \leq n < 8 \quad t &= 0,88n + 2,97 \text{ microsec.} \\ 8 < n \leq 15 \quad t &= 0,88n - 3,97 \text{ microsec.} \end{aligned} \quad (1)$$

Voor long shift instructies geldt:

$$\begin{aligned} 0 \leq n < 15 \quad t &= 0,88n + 2,97 \text{ microsec.} \\ 15 \leq n \leq 31 \quad t &= 0,88n - 3,19 \text{ microsec.} \end{aligned} \quad (2)$$

n : Stelt het aantal keer voor dat er geschoven wordt.

Aan de hand van deze twee formules en de door mij gedane veronderstelling werden de volgende gemiddelde executie tijden bepaald :

- a. Een gemiddelde executie tijd voor de short shift instructies.
- b. Een gemiddelde executie tijd voor de long shift instructies.
- c. Een gemiddelde executie tijd uitsluitend voor de SRQ, SLO en RLO shift instructies.

ad. a

Uitgaande van (1) wordt hieronder een gemiddelde executie tijd bepaald voor de short shift instructies.

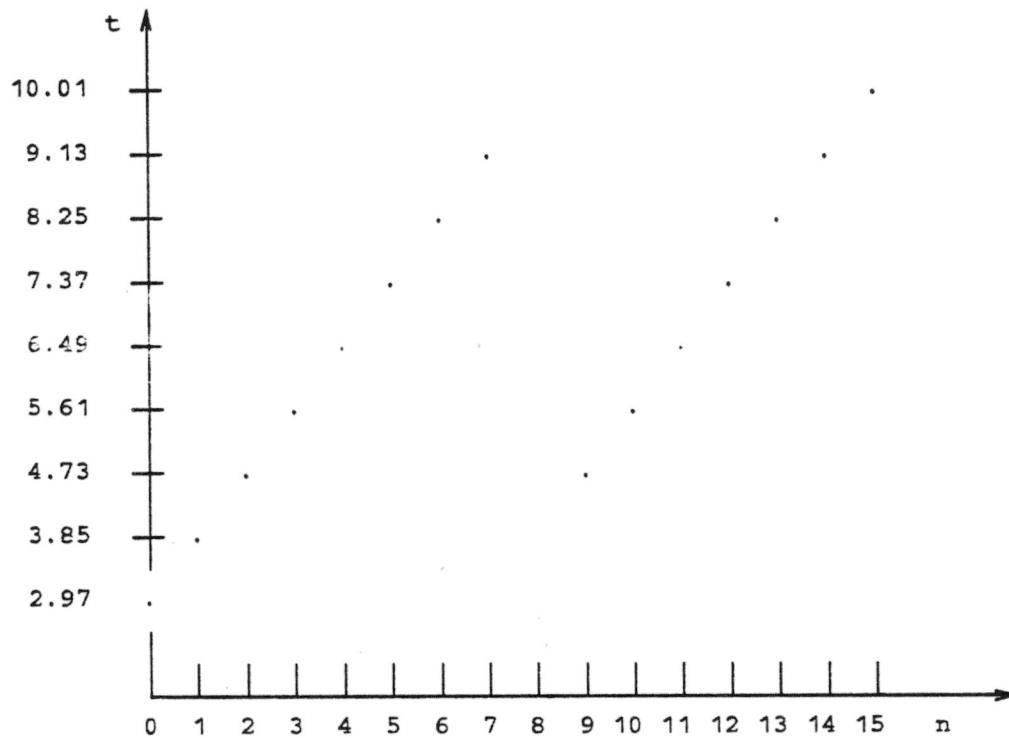


Figure 9. Grafische voorstelling van de tijden die nodig zijn wanneer bij een shift instructie van 0 t/m 15 maal geschoven wordt.

$n = 8$ wordt niet gedefinieerd

voor $n = 0$	$T = 1 \times 2.97 = 2.97$
$n = 1$	$T = 1 \times 3.85 = 3.85$
$n = 2$ en $n = 9$	$T = 2 \times 4.73 = 9.46$
$n = 3$ en $n = 10$	$T = 2 \times 5.61 = 11.22$
$n = 4$ en $n = 11$	$T = 2 \times 6.49 = 12.98$
$n = 5$ en $n = 12$	$T = 2 \times 7.37 = 14.74$
$n = 6$ en $n = 13$	$T = 2 \times 8.25 = 16.50$
$n = 7$ en $n = 14$	$T = 2 \times 9.13 = 18.26$
$n = 15$	$T = 1 \times 10.01 = 10.01$

Totale tijd = 99.99

$$T \text{ (gemiddeld)} = \frac{99.99}{15} = 6.66 \text{ microseconden}$$

ad. b

Uitgaande van (2) en dezelfde procedure volgend als bij de short shift instructies, werd voor de long shift instructies berekend dat :

$$T \text{ (gemiddeld)} = \frac{432.96}{32} = 13.53 \text{ microseconden}$$

ad. c

Wat de SRQ, SLQ en RLQ instructies betreft, zij maken een uitzondering en hun executie tijd wordt bepaald slechts door de volgende formule :

$$0 \leq n \leq 15 \quad t = 0,88n + 2,97 \text{ microsec.} \quad (3)$$

Volgens dezelfde procedure als bij (ad. a) en (ad. b), werd voor deze drie shift instructies berekend dat :

$$T \text{ (gemiddeld)} = \frac{153.12}{16} = 9.57 \text{ microseconden}$$

Verrichte metingen op de TCP18 instructies

Om nu de totale executie tijden te bepalen, die zowel de EMULATOR als de TCP18 nodig hebben om een hoeveelheid instructie mix uit te voeren, heb ik voor dit doel een reeks metingen op de TCP18 instructies verricht en wel als volgt :

Tijdens de uitvoering van de software door de TCP18 processor, heb ik, met behulp van een logic-analyzer en door middel van een trigger programma, telkens op willekeurige tijdstippen in de doorlopende software ingegrepen en, gedurende een door mij gekozen tijdsinterval van een minuut, gemeten hoe vaak, binnen de in deze tijdsinterval uitgevoerde instructie mix, een bepaalde TCP18 instructie uitgevoerd wordt.

Op deze wijze is voor elke instructie afzonderlijk een meting uitgevoerd en wel in twee toestanden van het systeem :

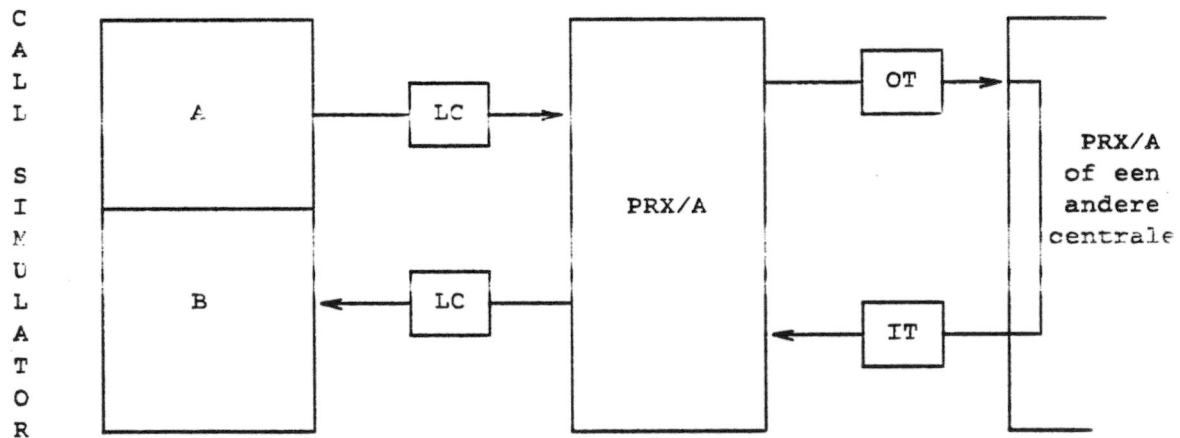
- In de toestand zonder telefonie.
- In de toestand met telefonie.

Zie APPENDIX A, par. 0.2 & 0.3

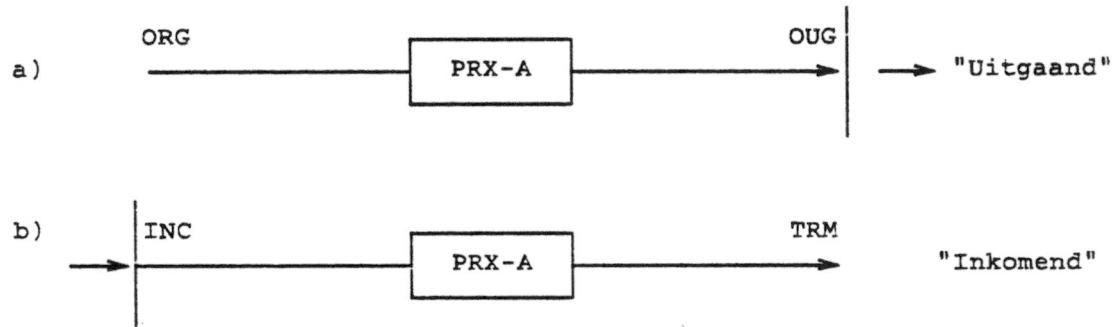
Bovendien werd, om een redelijk inzicht te krijgen in het werkelijk aantal keer dat een TCP18 instructie in een minuut tijd voorkomt, op elke instructie twee maal gemeten, waarna van beide uitkomsten het gemiddelde is genomen.

Voor de te meten instructies in de toestand met telefonie werd een systeem configuratie voor uitgaand verkeer opgesteld, waarin een groot aantal verschillende typen te gebruiken signaleringen werd opgenomen.

Configuratie



Uitgaande van deze configuratie kunnen twee soorten verbindingen tot stand worden gebracht :



Door verschillende typen signaleringen in een "junction group" op te nemen, worden verschillende signaleringen random door elkaar gebruikt, zodat in de tijd gezien de meting een totaal beeld geeft van het instructie gebruik in een geprojecteerde centrale welke meerdere signaleringstypen kent.

De signaleringen die voor deze configuratie opgenomen zijn in een "junction group" zijn de volgende :

1. Incoming F, 2/3 WIRE, HS,
2. Outgoing F, 2/3 WIRE, HS/MET
3. Incoming MFC, LOOP, HS,
4. Outgoing MFC, LOOP, HS/MET
5. Originating Subscriber, LOOP, RD
6. Originating Subscriber, LOOP, KD/RD
7. Terminating Subscriber, NO, IND
8. Incoming 7EN, HS
9. Incoming 7E, HS
10. Outgoing 7EN, HS/MET
11. Incoming 7D, HS
12. Outgoing 7D, HS/MET
13. Outgoing E/M, MFC, MET
14. Outgoing E/M, PULSE, MET
15. Incoming F, 2/3 WIRE, MET
16. Incoming E/M, MFC, MET
17. Incoming E/M, PULSE, HS
18. Incoming MFC, LOOP, MET
19. Outgoing E/M, MFC, HS
20. Outgoing E/M, PULSE, HS
21. Outgoing LOOP, HS, MFC, PCM
22. Incoming LOOP, HS, MFC, PCM
23. Incoming LOOP, MET, MFC, PCM
24. Outgoing LOOP, MET, MFC, PCM
25. Incoming F, HS, PULSE, PCM
26. Incoming F, MET, PULSE, PCM

- 27. Outgoing F, HS, PULSE, PCM
- 28. Outgoing F, MET, PULSE, PCM
- 29. Incoming E/M, HS, MFC, PCM
- 30. Outgoing E/M, HS, MFC, PCM
- 31. Incoming E/M, HS, PULSE, PCM
- 32. Outgoing E/M, HS, PULSE, PCM

Lijst van gebruikte afkortingen.

PCM : Pulse code modulation
MFC : Multi frequency code
MET : Metering
KT/RD: Keytone/Rotary dial
F : dc - signalering
E/M : Empfangen/Mittellen
HS : Hook signalering <Beantwoording - Neerleggen>

Door deze metingen te doen en uitgaande van de al berekende executie tijden per instructie door de EMULATOR en door de TCP18 processor, konden de grootheden worden bepaald die nodig zouden zijn om uitendelijk de verhoudingsfactor (VF) te kunnen berekenen. Tevens werd, door deze metingen te doen, een goed beeld verkregen over de meest gebruikte instructies in de TCP18 software.

De berekende grootheden waardoor de (VF) kan worden bepaald zijn de volgende :

- A. De executie tijd die de TCP18 nodig heeft om een aantal dezelfde instructies, die gedurende een minuut tijd in de doorlopende instructie mix zijn voorgekomen, uit te voeren.

De totale executie tijd die de TCP18 nodig heeft om het totale aantal verschillende instructies, die gedurende een minuut tijd in de doorlopende instructie mix zijn voorgekomen, uit te voeren.

- B. De executie tijd die de EMULATOR, in plaats van de TCP18, nodig heeft om hetzelfde aantal dezelfde instructies uit te voeren.

De totale executie tijd die de EMULATOR, in plaats van de TCP18, nodig heeft om hetzelfde totale aantal verschillende instructies uit te voeren.

Bepaling van de verhoudingsfactor

Uitgaande van de tot nu toe al berekende executie tijden, wordt hieronder een mathematische formule opgesteld, waaruit de verhoudingsfactor in executie tijd tussen de **EMULATOR** en de **TCP18** kan worden bepaald, wanneer aan de EMULATOR en aan de TCP18 een hoeveelheid instructie mix, voor een minuut tijd, ter uitvoering wordt aangeboden.

Hieronder worden, voor het bepalen van de formule, een aantal grootheden gedefinieerd :

(1) **I**

I stelt het aantal instructies van de TCP18 instructieset voor.

(2) **j**

j stelt een willekeurige instructie van de TCP18 instructieset voor.

(3) **Tj (EMULATOR)**

Tj (EMULATOR) stelt de tijd voor die de EMULATOR nodig heeft om een willekeurige TCP18 instructie uit te voeren.

(4) **Tj (TCP18)**

Tj (TCP18) stelt de tijd voor die de TCP18 zelf nodig heeft om een willekeurige instructie uit zijn instructies uit te voeren.

(5) **Aj**

Aj stelt het gemiddelde aantal keer voor dat een willekeurige TCP18 instructie, gemeten op twee willekeurige tijdstippen en telkens gedurende een minuut, wordt uitgevoerd.

Voorbeeld : Stel dat instructie j de LDB instructie is.

Deze A(LDB) instructie, gemeten een minuut lang, is gemiddeld 151856 maal uitgevoerd.

(6) **B**

B stelt het gemiddelde totale aantal verschillende TCP18 instructies voor die, in een tijd van een minuut, zijn uitgevoerd.

(7) $P_j = A_j/B \times 100\%$

P_j stelt het percentage voor van een aantal dezelfde instructies t.o.v het totale aantal verschillende instructies, die in een minuut tijd zijn uitgevoerd.

(8) $P_j \times T_j$ (EMULATOR)

$P_j \times T_j$ (EMULATOR) stelt de tijd voor die de EMULATOR nodig heeft om een aantal dezelfde TCP18 instructies, die in een minuut tijd voorkomen, uit te voeren.

(9)
$$\sum_{j=1}^I P_j \times T_j \text{ (EMULATOR)}$$

Deze som stelt de totale tijd voor die de EMULATOR nodig heeft om het totale aantal verschillende TCP18 instructies, die in een minuut voorkomen, uit te voeren.

(10) $P_j \times T_j$ (TCP18)

$P_j \times T_j$ (TCP18) stelt de tijd voor die de TCP18 zelf nodig heeft om een aantal van zijn instructies, die in een minuut voorkomen, uit te voeren.

(11)
$$\sum_{j=1}^I P_j \times T_j \text{ (TCP18)}$$

Deze som stelt de totale tijd voor die de TCP18 nodig heeft om het totale aantal verschillende instructies, die in een minuut voorkomen, uit te voeren.

Door (9) en (11) op elkaar te delen, krijgen wij een formule waaruit de gemiddelde verhoudingsfactor in totale executie tijd tussen EMULATOR en TCP18 kan worden berekend.

$$(VF) = \frac{\sum_{j=1}^I P_j \times T_j \text{ (EMULATOR)}}{\sum_{j=1}^I P_j \times T_j \text{ (TCP18)}}$$

Aanvankelijk was het de bedoeling om door middel van twee metingen, een meting in de toestand zonder telefonie en een meting in de toestand met telefonie, de performance van het nieuwe microprocessor systeem te bepalen in de toestand met alleen zuiver telefonie.

Als we nl. in staat zouden zijn de exacte bezettingsgraad van de TCP18, zowel in de toestand met als in de toestand zonder telefonie, te bepalen, zou het verschil daarvan een beeld kunnen geven over de mate van bezetting van de TCP18 processor, wanneer deze bezig is met het verwerken van alleen zuivere telefonie.

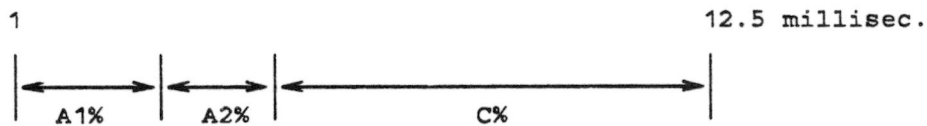
Daar niet exact te bepalen is hoe lang de processor, zowel in de toestand zonder als in de toestand met telefonie, gedurende het scanproces met verschillende werkzaamheden (aftasten van testpunten, interrupt afhandeling, call verwerking) bezig is, blijkt dat het bepalen van de bezettingsgraad van de TCP18 voor de toestand met alleen zuivere telefonie onmogelijk is. Dit werd aan de hand van hypothesen, gebaseerd op ervaringen, toch geprobeerd. Zie APPENDIX B.

Het scanproces is een autonoom proces dat elke 12.5 millisec. geïnitieerd wordt. In dit tijdsinterval leest de processor alle testpunten in de periferie uit en vergelijkt de nieuwe toestandswaarden van de testpunten met de laatste, in het geheugen opgeslagen, uitgelezen waarden.

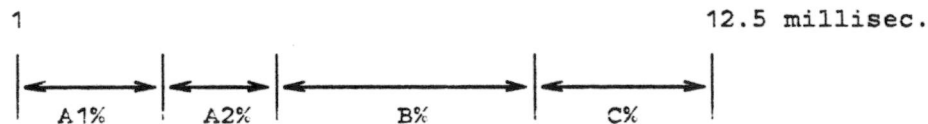
Indien er geen verschil wordt geconstateerd, betekent dit dat gedurende het laatste interval van 12.5 millisec. geen nieuwe events zijn opgetreden en hoeft de processor verder niets te doen.

Indien er wel verschil wordt geconstateerd, worden de nieuwe toestandswaarden in het geheugen weer opgeslagen en de processor gaat deze nieuwe toestanden uitwerken.

Scanproces voor de toestand zonder telefonie



Scanproces voor de toestand met telefonie



A1 :

A1 stelt de tijd voor waarin op een aantal testpunten getest wordt, uitgedrukt als percentage van de 12.5 milliseconden. Nadat op alle testpunten getest is, wordt een " end of scan " interrupt gegenereerd.

A2 :

A2 stelt de tijd voor waarin de processor bezig is met het afhandelen van het gegenereerde interrupt, uitgedrukt als percentage van de 12.5 milliseconden.

B :

B stelt de tijd voor die de processor nodig heeft om de nieuwe toestand(en) uit te werken, uitgedrukt als percentage van de 12.5 milliseconden.

C :

C stelt de tijd voor die de processor in de "idle loop" blijft, uitgedrukt als percentage van de 12.5 milliseconden.

Door gebruik te maken van de beschikbare call simulator, is gebleken, dat het aantal instructies die in de toestand met telefonie in een minuut zijn uitgevoerd, kleiner is dan het aantal instructies die in de toestand zonder telefonie in een minuut zijn uitgevoerd. Bovendien zijn deze tijdspercentages configuratie afhankelijk en kunnen daardoor moeilijk worden bepaald.

Derhalve zijn er twee afzonderlijke verhoudingsfactoren bepaald, een voor de toestand waarin het systeem zonder telefonie draait en een voor de toestand waarin het systeem wel met telefonie draait.

Bepaling van de verhoudingsfactor in de toestand zonder telefonie

Zie APPENDIX A, par. 0.4, pag. 20.

$$\sum_{j=1}^I P_j \times T_j \text{ (EMULATOR)} = 7924,8905 \text{ microseconden}$$

$$\sum_{j=1}^I P_j \times T_j \text{ (TCP18)} = 660,9053 \text{ microseconden}$$

$$(VF) = \frac{\sum_{j=1}^I P_j \times T_j \text{ (EMULATOR)}}{\sum_{j=1}^I P_j \times T_j \text{ (TCP18)}} = \frac{7924,8905}{660,9053} = 11,9909$$

Bepaling van de verhoudingsfactor in de toestand met telefonie.

Zie APPENDIX A, par. 0.5, pag. 25.

$$\sum_{j=1}^I P_j \times T_j \text{ (EMULATOR)} = 7619,9463 \text{ microseconden}$$

$$\sum_{j=1}^I p_j \times T_j \text{ (TCP18)} = 643,9897 \text{ microseconden}$$

$$(VF) = \frac{\sum_{j=1}^I P_j \times T_j \text{ (EMULATOR)}}{\sum_{j=1}^I p_j \times T_j \text{ (TCP18)}} = \frac{7619,9463}{643,9897} = 11,8324$$

ad. 2 Minder ruimte vergend en realisatie tegen redelijke kosten.

De ontwikkeling van nieuwe telefonie faciliteiten en de daarmee samenhangende uitbreiding van de PRX/A-software, maakte het noodzakelijk in het geheugen van de TCP18 extra informatie op te slaan.

Dit resulteerde in verdere uitbreiding van het geheugen, met als gevolg daarvan dat een volumineuze PRX/A-besturing ontstond.

De stormachtige ontwikkeling van de micro-elektronika die de laatste jaren plaats vindt, leidde ertoe dat ook voor de PRX/A-besturing een aantal aanpassingen konden worden gerealiseerd.

Daar de oorspronkelijke F- en Q-typen ringkernegeheugens qua prijs per bit, qua energieverbruik en qua omvang ongunstig waren, werd een project gestart om, gebruik makend van de nieuwe technologieën, een nieuw geheugen te ontwikkelen.

Dit resulteerde in het vervaardigen van een nieuw halfgeleider geheugen van het S-type, dat de conventionele F- en Q-typen heeft vervangen. Met dit nieuwe geheugen werd enerzijds qua prijs per bit en qua energieverbruik winst geboekt, anderzijds werd de omvang van de besturing enorm gereduceerd.

De toekomst weerspiegelt het steeds complexer worden van de geheugen- en de processor-chip. Dit zal resulteren in de ontwikkeling van nog krachtigere processor systemen met grotere opslagcapaciteit en nog veel meer mogelijkheden. Tevens zal het mogelijk worden dat het processor gedeelte op een chip aanwezig is en het geheel zal in een klein volume kunnen worden ondergebracht.

Vervanging dus van de huidige PRX/A-besturing door een nieuw microprocessor systeem zal de omvang van de besturing ongetwijfeld verkleinen en de kosten zullen, hoewel een groter geheugen en een relatief snelle en dure centrale verwerkingseenheid de kosten van een processor systeem nadelig beïnvloeden, meevallen, daar de nieuwe processoren en geheugens al goedkoper worden en in de nabije toekomst in prijs zullen blijven dalen.

Betrouwbaarheid van de gedane metingen

Om de betrouwbaarheid van de verrichte metingen te toetsen, ben ik ervan uitgegaan, dat de uitkomsten betreffende de verhoudings- factoren kleiner of ongeveer gelijk moeten zijn aan de verhoudingsfactor tussen de EMULATOR en de TCP18 van die instructie die het meest wordt uitgevoerd.

Dit, omdat uitvoering van deze instructie in feite de meeste CPU tijd vergt.

Uit de gedane metingen blijkt dat, zowel in de toestand zonder telefonie als in de toestand met telefonie, de meest voorkomende instructie de instructie LDQ (Load register Q from memory) is.

De verhoudingsfactor van de EMULATOR t.o.v. de TCP18 voor deze instructie is **12,15**.

Daar deze verhoudingsfactor inderdaad ongeveer gelijk is aan de hierboven berekende verhoudingsfactoren, kunnen de uitgevoerde metingen als betrouwbaar worden aangemerkt.

CONCLUSIES EN AANBEVELINGEN

Een telefonie systeem moet enerzijds betrouwbaar en anderzijds efficiënt in tijd zijn. Betrouwbaarheid is een algemene eis voor telefooncentrales, terwijl efficiëntie in tijd noodzakelijk is om een zo groot mogelijk aantal gesprekken per seconde door de centrale besturing te kunnen laten verwerken.

In het algemeen kan worden gesteld dat de economische ondergrens van een processor systeem wordt bepaald door de kosten van de centrale besturing en dat de mogelijke bovengrens wordt bepaald door de capaciteit (calls/sec.) van de centrale besturing.

Het kosten aspect zal in de nabije toekomst zeer zeker minder zwaar gaan wegen. Tevens ziet het er naar uit dat de omvang van de thans zo volumineuze PRX/A-besturing, gezien de technologische ontwikkelingen van de micro-elektronika, zal kunnen worden teruggebracht.

Aangaande het aspect capaciteit, wordt, gezien de resultaten in deze eerste fase van het onderzoek, een gewone software-oplossing, om de TCP18-CPU door een MC68000 microprocessor te vervangen, onder voorbehoud gesuggereerd.

Dit voorbehoud moet worden gemaakt omdat het softwarematig emuleren van de TCP18-CPU op een MC68000 microprocessor best mogelijk is, maar, aan de belangrijkste eis "real-time verwerking van de TCP18-software" wordt echter niet voldaan.

Uit het onderzoek blijkt nl. dat, wanneer de MC68000 microprocessor TCP18 programma's gaat uitvoeren, deze qua executie tijd, zowel in de toestand zonder als in de toestand met telefonie, ongeveer een factor 12 trager zal zijn dan de TCP18 processor zelf.

Deze verrassend hoge verhoudingsfactor c.q. vertragsingsfactor geeft aan dat de EMULATOR minder instructies (in de tijd gezien) dan de TCP18 kan verwerken, met als gevolg daarvan dat de capaciteit (calls/seconde) van het systeem sterk afneemt.

Derhalve kan worden geconcludeerd dat, althans volgens de resultaten van dit onderzoek, de EMULATOR niet in staat is TCP18 programma's in real-time te verwerken.

Deze conclusie zou een aanleiding kunnen zijn om verdere pogingen, om de huidige PRX/A-besturing te vervangen, snel te staken.

Echter, terugkijkend op de gevolgde werkwijze en de daarbij ter beschikking staande middelen, realiseer ik me, dat deze hoge vertragsingsfactor mede het gevolg kan zijn van :

- I. Het feit dat zeer veel hardware door middel van software geemuleerd werd.
- II. De gebruikte MC68000 compiler
- III. De beperking dat uit de MC680X0-serie alleen de MC68000 microprocessor kon worden gebruikt.

De vertragsingsfactor kan dus niet reeel maar slechts indicatief zijn. Onder optimale omstandigheden zou deze gevonden vertragsingsfactor zeer zeker lager zijn geweest en wellicht zelfs laag genoeg om verwerking van TCP18 programma's in real-time waar te kunnen maken.

ad. I

Uit de gedane metingen op de TCP18 instructies is gebleken, dat de memory reference instructies, in het bijzonder de LDQ, ADO en TNZ, de instructies zijn die in de TCP18-software het meest voorkomen.

Bovendien vergt het uitvoeren van elke memory reference instructie veel tijd. Dit doordat alleen al het bepalen van het effectieve adres een groot aantal assembler instructies vereist.

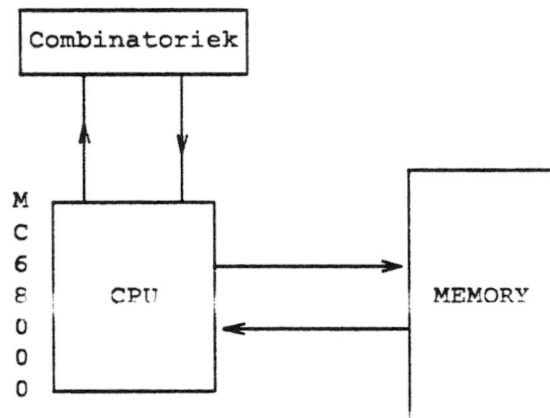
Namelijk **358** cycli zijn nodig voor directe adressering, **470** voor indirecte adressering, **184** voor gemodificeerde adressering, **398** voor gemodificeerde adressering voor jump instructies en **490** cycli voor indirecte gemodificeerde adressering.

De lange executie tijd van de EMULATOR benodigd voor het uitvoeren van de memory reference instructies, alsmede het heel vaak voorkomen van deze instructies, zijn factoren die mede bepalend zijn voor de gevonden hoge verhoudingsfactor tussen de EMULATOR en de TCP18 processor.

Het verdient dus de voorkeur om niet puur softwarematig te werk te gaan bij het uitvoeren van een instructie, maar om gebruik te maken van extra hardware.

Een zeer snelle combinatorische schakeling (die me niet zo moeilijk te realiseren lijkt) zou voorbereidend werk voor de processor kunnen doen en dit aan de processor kunnen aanbieden op het moment dat de processor dit nodig heeft.

Dit kan echter alleen zinvol zijn mits deze schakeling geen toegang tot het hoofdgeheugen heeft. Op deze wijze zou het uitvoeren van elke instructie versneld worden en dit zou leiden tot verlaging van de gevonden verhoudingsfactor (zie onderstaande schakeling).



Een verdere verlaging van de verhoudingsfactor zou eventueel bereikt kunnen worden, door te proberen het emulatie programma verder te optimaliseren.

Dit zou wellicht lukken door beter gebruik te maken van de trucs die de programmeertaal C biedt. Als deze poging tot verdere optimalizatie succesvol zou zijn, zou dit minder tijd voor het uitvoeren van elke instructie betekenen, wat uiteindelijk in een verdere daling van de verhoudingsfactor zou resulteren.

ad. II

De gevonden hoge verhoudingsfactor is o.a. mede afhankelijk van de gebruikte compiler. Deze conclusie is gebaseerd op het feit dat de drie verschillende versies van de MC68000 compiler alledrie een verschillende assembler code opleverden.

De berekeningen betreffende de executie tijden die de EMULATOR nodig heeft voor het uitvoeren van elke instructie afzonderlijk, zijn allen gebaseerd op de assembler code die de MC68000 compiler, versie 3.1 van 06/01/'84 gaf, omdat deze code, in vergelijking met de twee overige assembler codes, de kortste bleek te zijn.

Bij een kleine test, verricht op de TU DELFT (laboratorium A.V.S), waarbij een andere compiler gebruikt werd, bleek echter, dat de gegenereerde assembler code nog iets korter was dan de door mij gebruikte code.

Dit staft de conclusie dat de gevonden verhoudingsfactor hoe dan ook afhankelijk is van de gebruikte compiler.

Zou ik de mogelijkheid hebben gehad het emulatie programma op een nog "slimmere" compiler te laten compileren, dan zou ik zeker een kleinere verhoudingsfactor hebben gevonden.

ad. III

De tijd die de MC68000 microprocessor nodig zou hebben om een TCP18 instructie uit te voeren, werd bepaald op grond van zijn maximale frequentie van 12.5 MHz. De klokperiode (= 80 nanoseconden) stond dus bij alle berekeningen betreffende de verhoudingsfactor centraal. Zouden deze berekeningen echter gebaseerd zijn geweest op de klokperiode (= 62.5 nanoseconden) van de MC68020, dan zou de gevonden verhoudingsfactor niet 12 maar ongeveer 9.5 zijn geweest.

Hoewel de MC68020 objectcode MC68000 compatibel is, beschikt de MC68020 bovendien over extra mogelijkheden. Deze 32-bits microprocessor is namelijk uitgerust met een tussengeheugen (een snel, niet adresseerbaar buffer geheugen), dat zich tussen de verwerkingseenheid en het hoofdgeheugen bevindt om het snelheidsverschil tussen het relatief langzame geheugen en de snelle CPU op te vangen.

Het wordt voortdurend bijgewerkt en het bevat de allerlaatste gegevens uit het hoofdgeheugen. Kleine programma lussen kunnen dus zeer snel worden uitgevoerd doordat de daarvoor benodigde instructies niet voortdurend uit het geheugen behoeven te worden opgehaald. Dit met name resulteert in een toegangstijd verkorting.

Tevens is de MC68020 microprocessor uitgerust met de zogenaamde "pipelined architecture", waarbij drie instructies tegelijk uit het geheugen worden opgehaald en alle stappen voor het decoderen en uitvoeren van een instructie gelijktijdig plaatsvinden.

Is een instructie uitgevoerd, dan wordt de volgende instructie binnen gehaald. Dit proces herhaalt zich steeds om er voor te zorgen dat er altijd drie instructies tegelijk in de processor aanwezig zijn.

Het voordeel hiervan is dat de executie tijd van een instructie heel vaak binnen de executie tijd van de omringende instructies valt en dan dus nul kan zijn.

Uit het bovenstaande valt af te leiden dat, t.a.v de snelheid, het gebruik van een MC68020 bij de bedoelde vervanging voorkeur verdient boven het gebruik van een MC68000 microprocessor.

In de MC680X0-serie is inmiddels de MC68030 in opkomst die, volgens de summier informatie die bekend is, een factor 2.6 sneller is dan de MC68020.

Uit het feit dat de MC68020 ongeveer een factor 2 sneller is t.o.v. de MC68000, kan (althans theoretisch gezien) worden afgeleid dat de MC68030 ongeveer een factor 5 sneller is t.o.v. de MC68000.

Dit, altijd theoretisch gezien, betekent dat indien we een MC68030 i.p.v. een MC68000 zouden hebben gebruikt, de VF ongeveer 2.4 zou zijn geweest.

Door gebruik te maken van een slimmere compiler, door toepassing van extra hardware alsmede door te proberen het emulatie programma verder te optimaliseren, zou deze factor van 2.4 zeker tot 1 kunnen worden teruggebracht.

Dit alles in ogenschouw nemend maakt mij dermate optimistisch dat ik aanbeveel, indien dit in het belang is van APT, het reeds gestarte onderzoek voort te zetten.

EPILOOG

Daar dit onderzoek aanvankelijk erg abstract en ongelooflijk ingewikkeld was, zou het voor mij heel plezierig zijn geweest indien ik met iemand, b.v. een mede-afstudeerder, had kunnen samenwerken, zodat de software en hardware knelpunten beter onderzocht hadden kunnen worden.

Toch is mijn interesse tijdens dit onderzoek zodanig gewekt dat ik, indien APT zou besluiten dit onderzoek voort te zetten, daaraan graag zou willen meewerken.

LITERATUUROPGAVE

Overzicht van de cursus module-reeks voor het PRX/A-systeem, DIS-OS2 afdeling APT-systemen Hilversum.

SSA - TCP-18 SUPPORT SYSTEMS MANUAL.
Volume 2 : PROGRAMMERS REFERENCE MANUAL.
ST 2228-MO-H6037 APT-Hilversum.

THE C PROGRAMMING LANGUAGE, Brian W. Kernighan, Dennis M. Ritchie, PRENTICE-HALL SOFTWARE SERIES.

DE PROGRAMMEERTAAL C, Ir. L. Ammeral. ACADEMIC SERVICE

APPENDIX A

0.1 *Executie tijd van instructies zowel door TCP18 als door EMULATOR (worst case) in microseconden*

	Inst.j	Tj (TCP18)	Tj (EMULAT)	Verhoudingsfactor
RST				
DIRECT	08	16,94	99,20	5,85
GEMOD.	09	16,94	86,88	5,12
INDIR.	0A	19,03	109,76	5,76
IND.G.	0B	19,03	111,36	5,85
JRT	10	6,49	67,68	10,42
	11	7,37	70,88	9,61
	12	8,58	78,24	9,11
	13	9,46	79,84	8,43
JMP SKP	18	2,31	56,80	24,58
	19	3,19	60,00	18,80
	1A	4,40	71,84	16,32
	1B	5,28	73,44	13,90
CMP	20	5,06	72,00	14,22
	21	5,06	59,68	11,79
	22	7,15	82,56	11,54
	23	7,15	84,16	11,77
JBN	28	2,97	61,28	20,63
	29	3,85	64,48	16,74
	2A	4,40	76,32	17,34
	2B	5,28	77,92	14,75
SUB	30	4,40	69,12	15,70
	31	4,40	56,80	12,90
	32	6,49	79,68	12,27
	33	6,49	81,28	12,52
JAZ SKZ	38	2,31	58,88	25,48
	39	3,19	62,08	19,46
	3A	4,40	73,92	16,80
	3B	5,28	75,52	14,30
AND	40	4,40	62,88	14,29
	41	4,40	50,56	11,49
	42	6,49	73,44	11,31
	43	6,49	75,04	11,56
JSR	48	6,93	83,36	12,02
	49	7,81	86,56	11,08
	4A	9,02	98,40	10,90
	4B	9,02	100,00	11,08
LDB	50	4,40	62,24	14,14
	51	4,40	49,92	11,34
	52	6,49	72,80	11,21
	53	6,49	74,40	11,46

	Inst.j	Tj (TCP18)	Tj (EMULAT)	Verhoudingsfactor
STB	58	4,62	68,48	14,82
	59	4,62	56,12	12,14
	5A	6,71	79,04	11,77
	5B	6,71	80,64	12,01
LDQ	60	4,40	68,32	15,52
	61	4,40	56,00	12,72
	62	6,49	78,88	12,15
	63	6,49	80,48	12,40
STQ	68	4,62	68,48	14,82
	69	4,62	56,12	12,14
	6A	6,71	79,04	11,77
	6B	6,71	80,64	12,01
LDA	70	4,40	62,24	14,14
	71	4,40	49,92	11,34
	72	6,49	72,80	11,21
	73	6,49	74,40	11,46
STA	78	4,62	68,48	14,82
	79	4,62	56,12	12,14
	7A	6,71	79,04	11,77
	7B	6,71	80,64	12,01
LDR	80	4,40	65,44	14,87
	81	4,40	53,12	12,07
	82	6,49	76,00	11,71
	83	6,49	77,60	11,95
STR	88	5,50	75,52	13,73
	89	5,50	63,20	11,49
	8A	7,59	86,08	11,34
	8B	7,59	87,68	11,55
LDL	90	4,40	65,44	14,87
	91	4,40	53,12	12,07
	92	6,49	76,00	11,71
	93	6,49	77,60	11,95
STL	98	5,50	75,52	13,73
	99	5,50	63,20	11,49
	9A	7,59	86,08	11,34
	9B	7,59	87,68	11,55
XOR	A0	4,40	62,88	14,29
	A1	4,40	50,56	11,49
	A2	6,49	73,44	11,31
	A3	6,49	75,04	11,56
STX	A8	5,50	77,28	14,05
	A9	5,50	64,96	11,81
	AA	7,59	87,84	11,57
	AB	7,59	89,44	11,78

	Inst.j	Tj (TCP18)	Tj (EMULAT)	Verhoudingsfactor
ADD	B0	4,40	68,80	15,63
	B1	4,40	56,48	12,83
	B2	6,49	79,36	12,22
	B3	6,49	80,96	12,47
ADO	B8	5,50	81,12	14,74
	B9	5,50	68,80	12,50
	BA	7,59	91,68	12,07
	BB	7,59	93,28	12,28
ORA	C0	4,40	62,88	14,29
	C1	4,40	50,56	11,49
	C2	6,49	73,44	11,31
	C3	6,49	75,04	11,56
TNZ	C8	5,50	78,40	14,25
	C9	5,50	66,08	12,01
	CA	7,59	88,96	11,72
	CB	7,59	90,56	11,93
LDK	D2	2,31	29,12	12,60
GEM.	D3	2,31	36,32	15,72
MBA	D300	2,31	28,16	12,19
LKQ	D6	2,31	29,12	12,60
GEM.	D7	2,31	30,40	13,16
MBQ	D700	2,31	28,16	12,19
ADK	FC	2,31	29,44	12,74
GEM.	FD	3,19	36,64	11,48
ABA	FD00	3,19	28,80	9,02
LKB	D4	2,31	29,12	12,60
AKB	D5	2,31	29,44	12,74
SRQ	D0	9,57	32,56	3,40
	D1	9,57	34,48	3,60
SRA	E0	6,66	32,56	4,88
	E1	6,66	34,48	5,17
RRA	E2	6,66	45,28	6,79
	E3	6,66	47,20	7,08
SLA	E4	6,66	34,00	5,10
	E5	6,66	35,92	5,39
SLL	E6	13,53	358,88	26,52
	E7	13,53	360,80	26,66
SLQ	E8	9,57	34,00	3,55
	E9	9,57	35,92	3,75

	Inst.j	Tj (TCP18)	Tj (EMULAT)	Verhoudingsfactor
RLQ	EA	9,57	45,28	4,73
	EB	9,57	47,20	4,93
SRL	EC	13,53	314,24	23,22
	ED	13,53	316,16	23,36
RRL	EE	13,53	463,04	34,22
	EF	13,53	464,96	34,36
CPK	F4	2,97	34,24	11,52
	F5	2,97	35,36	11,90
TST	F6	2,31	34,24	14,82
	F7	2,31	35,36	15,30
SET	F8	2,31	37,12	16,06
	F9	2,31	40,16	17,38
CLR	FA	2,31	38,72	16,76
	FB	2,31	40,48	17,52
SKA	F0	2,31	35,04	15,16
SKB	F1	2,31	29,44	12,74
TBK	F2	2,97	31,04	10,45
LKZ	F3	2,31	32,32	13,99
NOP	FF	2,31	25,92	11,22
LRR	04	4,40	43,36	9,85
SRC	0140	6,66	85,76	12,87
ANQ	0180	2,97	28,80	9,69
XOQ	0201	2,31	28,80	12,46
MAQ	0202	2,31	28,16	12,19
AAQ	0203	2,31	28,80	12,46
MAB	0204	2,31	28,16	12,19
LPS	0208	2,31	42,08	18,21
ORQ	0210	2,31	28,80	12,46
IVQ	0221	2,31	28,80	12,36
LOQ	0222	2,31	27,52	11,91
SOQ	0223	2,31	27,52	11,91

	Inst.J	Tj (TCP18)	Tj (EMULAT)	Verhoudingsfactor
LOB	0224	2,31	27,52	11,91
LPC	0228	2,31	42,08	18,21
MQA	0240	2,31	28,16	12,19
MQB	0264	2,31	28,16	12,19
MZA	0280	2,31	45,76	19,80
TNB	0308	2,31	30,40	13,16
TNO	0310	2,31	30,40	13,16
ALI	0320	2,31	27,52	11,91
BKI	0340	2,31	27,52	11,91
TPB	0360	2,31	28,80	12,46

INA	D8	17,05	*	*
	D9	17,05	*	*
INF	DA	17,05	*	*
	DB	17,05	*	*
OTA	DC	17,05	*	*
	DD	17,05	*	*
OTF	DE	17,05	*	*
	DF	17,05	*	*
HLT	00	20,57	*	*
TRC	0120	-----	*	*
TSA	0301	2,31	25,92 *	11,22 *
TSB	0302	2,31	25,92 *	11,22 *
TSC	0304	2,31	25,92 *	11,22 *
SPM	0380	2,31	*	*

ISC	FEOX	-----	*	*
Voor laboratorium doeleinden !!				
RCS	FEDO	-----	*	*

0.2 *Gedane metingen zonder telefonie.*

	Inst.j	Meting(1)	Meting(2)	Aj	Pj (%)
RST					
DIRECT	08	0	0	0	0
GEMOD.	09	211546	210277	210912	1,6361
INDIR.	0A	0	0	0	0
IND.G.	0B	0	0	0	0
JRT	10	291128	269751	280440	2,1755
	11	0	0	0	0
	12	166231	160440	163336	1,2671
	13	903	1224	1064	0,0083
JMP SKP	18	1280	1223	1252	0,0097
	19	500	484	492	0,0038
	1A	11	1	6	4,6exp(-5)
	1B	0	2	1	7,7exp(-6)
CMP	20	10819	8678	9749	0,0756
	21	4854	4801	4828	0,0375
	22	5398	5470	5434	0,0422
	23	6	6	6	4,6exp(-5)
JBN	28	25	116	71	0,0006
	29	0	0	0	0
	2A	0	0	0	0
	2B	0	0	0	0
SUB	30	708	518	613	0,0048
	31	0	0	0	0
	32	11384	11614	11499	0,0892
	33	598	596	597	0,0046
JAZ SKZ	38	11065	10928	10997	0,0853
	39	0	0	0	0
	3A	0	0	0	0
	3B	0	0	0	0
AND	40	4794	14339	9567	0,0742
	41	14551	15240	14896	0,1156
	42	2	490	246	0,0019
	43	863	978	921	0,0071
JSR	48	12	98	55	0,0004
	49	0	0	0	0
	4A	8447	18042	13245	0,1027
	4B	0	0	0	0
LDB	50	150426	153291	151856	1,1780
	51	75029	75197	75113	0,5827
	52	127116	137371	132244	1,0259
	53	82097	81309	81703	0,6338

	Inst.j	Meting(1)	Meting(2)	Aj	Pj (%)
STB	58	81636	82051	81844	0,6349
	59	2408	2386	2397	0,0186
	5A	107762	109160	108461	0,8414
	5B	0	0	0	0
LDQ	60	4421	3663	4042	0,0314
	61	11975	11883	11929	0,0925
	62	5716613	5760075	5738344	44,5148
	63	1267	105	686	0,0053
STQ	68	8215	12437	10326	0,0801
	69	6046	6068	6057	0,0470
	6A	1920	2246	2083	0,0162
	6B	1309	2424	1867	0,0145
LDA	70	199141	196864	198003	1,5360
	71	289205	286435	287820	2,2327
	72	143069	183793	163431	1,2678
	73	88815	81998	85407	0,6625
STA	78	159150	190825	174988	1,3575
	79	98411	96426	97419	0,7557
	7A	35012	35111	35062	0,2720
	7B	18061	20156	19109	0,1482
LDR	80	0	0	0	0
	81	4816	4745	4781	0,0371
	82	0	0	0	0
	83	18	22	20	0,0002
STR	88	132	0	66	0,0005
	89	0	0	0	0
	8A	0	0	0	0
	8B	0	0	0	0
LDL	90	10	1	6	4,6exp(-5)
	91	0	0	0	0
	92	0	4	2	1,5exp(-5)
	93	18	22	20	0,0002
STL ***	98	0	0	0	0
	99	0	0	0	0
	9A	0	0	0	0
	9B	0	0	0	0
XOR	A0	5054	4800	4927	0,0382
	A1	37999	36711	37355	0,2898
	A2	2	438	220	0,0017
	A3	508	18	263	0,0020
STX ***	A8	0	0	0	0
	A9	0	0	0	0
	AA	0	0	0	0
	AB	0	0	0	0

	Inst.j	Meting(1)	Meting(2)	Aj	Pj (%)
ADD	B0	19607	14402	17005	0,1319
	B1	24997	24890	24944	0,1935
	B2	602	2090	1346	0,0104
	B3	7692	7644	7668	0,0595
ADO	B8	22097	22023	22060	0,1711
	B9	14171	14421	14296	0,1109
	BA	2854933	2829262	2842098	22,0474
	BB	56	86	71	0,0006
ORA	C0	14389	14332	14361	0,1114
	C1	12885	13072	12979	0,1007
	C2	3534	3055	3295	0,0256
	C3	36	95	66	0,0005
TNZ	C8	1486101	1498990	1492546	11,5783
	C9	37731	37995	37863	0,2937
	CA	39639	39942	39791	0,3087
	CB	43885	45396	44641	0,3463
LDK	D2	111	9	60	0,0005
GEM.	D3	0	5	3	2,3exp(-5)
MBA	D300	0	5	3	2,3exp(-5)
LKQ	D6	0	2	1	7,7exp(-6)
***GEM.	D7	0	0	0	0
MBQ	D700	0	0	0	0
ADK	FC	1185	1028	1106	0,0086
GEM.	FD	0	1	1	7,7exp(-6)
ABA	FD00	1	0	1	7,7exp(-6)
LKB	D4	0	3	2	1,5exp(-5)
AKB	D5	0	4	2	1,5exp(-5)
SRQ	D0	5	1	3	2,3exp(-5)
	D1	0	0	0	0
SRA	E0	9	2	6	4,6exp(-5)
	E1	1	1	1	7,7exp(-6)
RRA ***	E2	0	0	0	0
	E3	0	0	0	0
SLA	E4	4	0	2	1,5exp(-5)
	E5	0	0	0	0
SLL ***	E6	0	0	0	0
	E7	0	0	0	0
SLQ ***	E8	0	0	0	0
	E9	0	0	0	0

	Inst.j	Meting(1)	Meting2)	Aj	Pj (%)
RLQ	EA	0	0	0	0
***	EB	0	0	0	0
SRL	EC	0	0	0	0
***	ED	0	0	0	0
RRL	EE	0	0	0	0
***	EF	0	0	0	0
CPK	F4	1	4	3	2,3exp(-5)
	F5	0	0	0	0
TST	F6	0	0	0	0
***	F7	0	0	0	0
SET	F8	0	0	0	0
***	F9	0	0	0	0
CLR	FA	5	1	3	2,3exp(-5)
	FB	1	0	1	7,7exp(-6)
SKA	F0	1	0	1	7,7exp(-6)
SKB	F1	3	17	10	7,7exp(-5)
TBK	F2	0	4	2	1,5exp(-5)
LKZ	F3	0	0	0	0

NOP	FF	1199	238	719	0,0056
LRR	04	15732	21457	18595	0,1442
SRC	0140	12934	12884	12909	0,1001
ANQ	0180	0	0	0	0

XOQ	0201	0	0	0	0

MAQ	0202	6	0	3	2,3exp(-5)
AAQ	0203	0	0	0	0

MAB	0204	9480	9445	9463	0,0734
LPS	0208	0	0	0	0

ORQ	0210	0	2	1	7,7exp(-6)
IVQ	0221	0	0	0	0

LOQ	0222	0	0	0	0

	Inst.j	Meting(1)	Meting(2)	Aj	Pj (%)
SOQ ***	0223	0	0	0	0
LOB	0224	1201	1185	1193	0,0093
LPC ***	0228	0	0	0	0
MQA	0240	2	1	2	1,5exp(-5)
MQB	0264	0	1	1	7,7exp(-6)
MZA	0280	3	7	5	3,8exp(-5)
TNB ***	0308	0	0	0	0
TNO	0310	507	535	521	0,0040
ALI	0320	536	482	509	0,0039
BKI	0340	4738	4802	4770	0,0370
TPB	0360	9	1	5	3,8exp(-5)

INA	D8	0	0	0	0
	D9	7	0	4	3,1exp(-5)
INF ***	DA	0	0	0	0
	DB	0	0	0	0
OTA ***	DC	0	0	0	0
	DD	0	0	0	0
OTF	DE	8065	8180	8123	0,0630
	DF	0	0	0	0
HLT ***	00	-	-	-	-
TRC ***	0120	-	-	-	-
TSA ***	0301	-	-	-	-
TSB ***	0302	-	-	-	-
TSC ***	0304	-	-	-	-
SPM	0380	-	-	-	-

ISC	FE0X	-	-	-	-
Voor laboratorium doeleinden !!!					
RCS	FED0	-	-	-	-

Totaal uitgevoerde					
instructies per minuut :		12.890.892			

0.3 *Gedane metingen met telefonie*

	Inst.j	Meting(1)	Meting(2)	Aj	Pj (%)
RST					
DIRECT	08	0	0	0	0
GEMOD.	09	207027	215194	211111	1,6630
INDIR.	0A	0	0	0	0
IND.G.	0B	0	0	0	0
JRT	10	386091	311829	348960	2,7488
	11	0	0	0	0
	12	346338	244624	295481	2,3275
	13	147	1962	1055	0,0083
JMP SKP	18	3590	4160	3875	0,0305
	19	361	417	389	0,0030
	1B	246	267	257	0,0020
	1A	0	4	2	1,5exp(-5)
CMP	20	12273	33824	23049	0,1816
	21	5350	5290	5320	0,0419
	22	5499	5412	5456	0,0430
	23	1666	1642	1654	0,0130
JBN	28	154	99	127	0,0010
	29	0	0	0	0
	2A	0	0	0	0
	2B	0	0	0	0
SUB	30	1990	2738	2364	0,0186
	31	0	0	0	0
	32	17336	16456	16896	0,1331
	33	594	598	596	0,0047
JAZ SKZ	38	14465	14766	14616	0,1151
	39	0	0	0	0
	3A	0	0	0	0
	3B	0	0	0	0
AND	40	82931	61350	72141	0,5683
	41	13135	13245	13190	0,1039
	42	3116	2846	2981	0,0235
	43	12510	12279	12395	0,0976
JSR	48	4487	5100	4794	0,0378
	49	0	0	0	0
	4A	60150	122228	91189	0,7183
	4B	0	0	0	0
LDB	50	257347	209610	233479	1,8391
	51	137762	119936	128849	1,0150
	52	139946	169359	154653	1,2182
	53	87560	88297	87929	0,6926

	Inst.g	Meting(1)	Meting(2)	Aj	Pj (%)
STB	58	135869	110909	123389	0,9720
	59	3317	3326	3322	0,0262
	5A	109376	129310	119343	0,9401
	5B	0	0	0	0
LDQ	60	21988	21222	21605	0,1702
	61	22856	23866	23361	0,1840
	62	5246877	4642330	4944604	38,9493
	63	2073	2155	2114	0,0167
STQ	68	40747	24422	32585	0,2567
	69	23795	25208	24502	0,1930
	6A	3126	3313	3220	0,0254
	6B	22105	3483	12794	0,1008
LDA	70	269609	252754	261182	2,0574
	71	376852	421458	399155	3,1442
	72	174370	225818	200094	1,5762
	73	138872	153066	145969	1,1498
STA	78	277683	213248	245466	1,9336
	79	227522	171418	199470	1,5713
	7A	87502	43662	65582	0,5166
	7B	29271	36217	32744	0,2579
LDR	80	45	166	106	0,0008
	81	4791	4878	4835	0,0381
	82	4	18	11	8,6exp(-5)
	83	18	18	18	1,4exp(-4)
STR	88	0	0	0	0
	89	0	0	0	0
	8A	24	22	23	1,8exp(-4)
	8B	0	0	0	0
LDL	90	38	35	37	2,9exp(-4)
	91	0	0	0	0
	92	6	10	8	6,3exp(-5)
	93	18	18	18	1,4exp(-4)
STL ***	98	0	0	0	0
	99	0	0	0	0
	9A	0	0	0	0
	9B	0	0	0	0
XOR	A0	7051	6939	6995	0,0551
	A1	40626	40090	40358	0,3179
	A2	52	62	57	4,4exp(-4)
	A3	637	678	658	0,0052
STX	A8	0	0	0	0
	A9	0	0	0	0
	AA	24	22	23	1,8exp(-4)
	AB	0	0	0	0

	Inst.j	Meting(1)	Meting(2)	Aj	Pj (%)
ADD	B0	19254	19128	19191	0,1512
	B1	24508	25154	24831	0,1956
	B2	1175	1088	1132	0,0089
	B3	9447	9032	9240	0,0728
ADO	B8	57327	25795	41561	0,3274
	B9	16215	16262	16239	0,1279
	BA	2559668	2274656	2417162	19,0403
	BB	322	198	260	0,0020
ORA	C0	18895	18773	18834	0,1484
	C1	12973	12990	12982	0,1023
	C2	3371	3529	3450	0,0272
	C3	3369	1139	2254	0,0178
TNZ	C8	1340174	1199738	1269965	10,0037
	C9	39726	49360	44543	0,3509
	CA	44887	45018	44953	0,3541
	CB	44449	45330	44890	0,3536
LDK	D2	422	334	378	0,0029
GEM.	D3	34	8	22	1,7exp(-4)
MBA	D300	13	9	11	8,6exp(-5)
LKQ	D6	65	42	54	4,2exp(-4)
GEM.	D7	618	581	600	0,0047
MBQ	D700	599	603	601	0,0047
ADK	FC	766	723	745	0,0058
GEM.	FD	3	11	7	5,5exp(-5)
ABA	FD00	1	9	5	3,9exp(-5)
LKB	D4	4	6	5	3,9exp(-5)
AKB	D5	7	37	22	1,7exp(-4)
SRQ	D0	23	9	16	1,2exp(-4)
	D1	0	0	0	0
SRA	E0	360	115	238	0,0018
	E1	0	6	3	2,3exp(-5)
RRA	E2	0	0	0	0
	E3	1	1	1	7,8exp(-6)
SLA	E4	30	26	28	2,2exp(-4)
	E5	0	2	1	7,8exp(-6)
SLL	E6	7	4	6	4,7exp(-5)
	E7	1	5	3	2,3exp(-5)
SLQ	E8	32	4	18	1,4exp(-4)
	E9	0	0	0	0

	Inst.J	Meting(1)	Meting(2)	Aj	Pj (%)
RLQ	EA	1	0	1	7,8exp(-6)
	EB	0	0	0	0
SRL	EC	4	2	3	2,3exp(-5)
	ED	0	0	0	0
RRL	EE	5	5	5	3,9exp(-5)
	EF	0	0	0	0
CPK	F4	39	11	25	1,9exp(-4)
	F5	0	0	0	0
TST	F6	79	105	92	7,2exp(-4)
	F7	310	312	311	0,0024
SET	F8	3	9	6	4,7exp(-5)
	F9	274	259	267	0,0021
CLR	FA	35	3	19	1,4exp(-4)
	FB	17	9	13	1,0exp(-4)
SKA	F0	0	0	0	0

SKB	F1	13	1	7	5,5exp(-5)
TBK	F2	20	6	13	1,0exp(-4)
LKZ	F3	0	0	0	0

NOP	FF	619	405	512	0,0040
LRR	04	40527	29627	35077	0,2763
SRC	0140	11449	11525	11487	0,0404
ANQ	0180	11	19	15	1,1exp(-4)
XOQ	0201	3	2	3	2,3exp(-5)
MAQ	0202	21	103	62	4,8exp(-4)
AAQ	0203	1	15	8	6,3exp(-5)
MAB	0204	7905	7469	7687	0,0605
LPS	0208	0	0	0	0

ORQ	0210	4	10	7	5,5exp(-5)
IVQ	0221	0	0	0	0

LOQ	0222	0	0	0	0

SOQ	0223	3	3	3	2,3exp(-5)

	Inst.j	Meting(1)	Meting(2)	Aj	Pj (%)
LOB	0224	1138	1235	1187	0,0093
LPC	0228	0	0	0	0

MQA	0240	31	41	36	2,8exp(-4)
MQB	0264	962	920	941	0,0074
MZA	0280	1	3	2	1,5exp(-5)
TNB	0308	0	0	0	0

TNO	0310	474	562	518	0,0040
ALI	0320	520	697	609	0,0047
BKI	0340	5492	5426	5459	0,0436
TPB	0360	70	4	37	2,9exp(-4)

INA	D8	0	0	0	0
	D9	642	735	689	0,0054
INF	DA	0	0	0	0
***	DB	0	0	0	0
OTA	DC	1	0	1	7,8exp(-6)
	DD	0	0	0	0
OTF	DE	8126	8080	8103	0,0638
	DF	1075	1037	1056	0,0083
HLT	00	-	-	-	-

TRC	0120	-	-	-	-

TSA	0301	-	-	-	-

TSB	0302	-	-	-	-

TSC	0304	-	-	-	-

SPM	0380	-	-	-	-

ISC	FEOX	-	-	-	-
Voor laboratorium doeleinden !!					
RCS	FEDO	-	-	-	-

**Totaal uitgevoerde
instructies per minuut : 12.694.968**

In een tijd van 7,30 uur zijn 18131 calls gemaakt waarvan
er 5830 zijn mislukt.

Totaal geslaagde calls : 18131 - 5830 = 12301

Aantal gemaakte calls per minuut = 40

0.4 *Berekening van de $P_j \times T_j$ (TCP18) & $P_j \times T_j$ (EMULAT) zonder telefonie*

Inst.j	$P_j \times T_j$ (TCP18)	$P_j \times T_j$ (EMULAT)
RST		
DIRECT 08	0	0
GEMOD. 09	27,7117	142,1444
INDIR. 0A	0	0
IND.G. 0B	0	0
JRT 10	14,1189	147,2378
11	0	0
12	10,8717	99,1379
13	0,0785	0,6627
JMP SKP 18	0,0224	0,5510
19	0,0121	0,2280
1A	0,0002	0,0033
1B	0,0000	0,0006
CMP 20	0,3825	5,4432
21	0,1897	2,2380
22	0,3017	3,4840
23	0,0003	0,0039
JBN 28	0,0017	0,0368
29	0	0
2A	0	0
2B	0	0
SUB 30	0,0211	0,3318
31	0	0
32	0,5789	7,1075
33	0,0298	0,3739
JAZ SKZ 38	0,1970	5,0225
39	0	0
3A	0	0
3B	0	0
AND 40	0,3264	5,1287
41	0,5086	5,8447
42	0,0123	0,1395
43	0,0460	0,5328
JSR 48	0,0027	0,0333
49	0	0
4A	0,9263	10,1057
4B	0	0
LDB 50	5,1832	73,3187
51	2,5638	29,0884
52	6,6580	74,6855
53	4,1133	47,1547

	Inst.j	Pj x Tj (TCP18)	Pj x Tj (EMULAT)
STB	58	2,9332	43,4780
	59	0,0859	1,0438
	5A	5,6457	66,5043
	5B	0	0
LDQ	60	0,1381	2,1452
	61	0,4070	5,1800
	62	288,9010	3511,3274
	63	0,0343	0,4265
STQ	68	0,3700	5,4852
	69	0,2171	2,6376
	6A	0,1087	1,2804
	6B	0,0972	1,1693
LDA	70	6,7584	95,6006
	71	9,8238	111,4564
	72	8,2280	92,2958
	73	4,2996	49,2900
STA	78	6,2716	92,9616
	79	3,4907	42,4099
	7A	1,8251	21,4989
	7B	0,9944	11,9508
LDR	80	0	0
	81	0,1632	1,9708
	82	0	0
	83	0,0012	0,0155
STR	88	0,0027	0,0378
	89	0	0
	8A	0	0
	8B	0	0
LDL	90	0,0002	0,0030
	91	0	0
	92	0,0001	0,0011
	93	0,0012	0,0155
STL ***	98	0	0
	99	0	0
	9A	0	0
	9B	0	0
XOR	A0	0,1680	2,4020
	A1	1,2751	14,6523
	A2	0,0110	0,1248
	A3	0,0129	0,1501
STX ***	A8	0	0
	A9	0	0
	AA	0	0
	AB	0	0

Inst.j		Pj x Tj (TCP18)	Pj x Tj (EMULAT)
ADD	B0	0,5803	9,0747
	B1	0,8514	10,9289
	B2	0,0674	0,8253
	B3	0,3861	4,8171
ADO	B8	0,9410	13,8796
	B9	0,6099	7,6299
	BA	167,3397	2021,3056
	BB	0,0045	0,0560
ORA	C0	0,4901	7,0048
	C1	0,4430	5,0914
	C2	0,1661	1,8801
	C3	0,0032	0,0375
TNZ	C8	63,6806	907,7387
	C9	1,6153	19,4077
	CA	2,3430	27,4620
	CB	2,6284	31,3609
LDK	D2	0,0011	0,0146
GEM.	D3	0,0001	0,0008
MBA	D300	0,0001	0,0006
LKQ	D6	0,0000	0,0002
***GEM.	D7	0	0
MBQ	D700	0	0
ADK	FC	0,0198	0,2532
GEM.	FD	0,0000	0,0003
ABA	FD00	0,0000	0,0002
LKB	D4	0,0000	0,0004
AKB	D5	0,0000	0,0004
SRQ	D0	0,0002	0,0007
	D1	0	0
SRA	E0	0,0003	0,0015
	E1	0,0001	0,0003
RRA	E2	0	0
	E3	0	0
SLA	E4	0,0001	0,0005
	E5	0	0
SLL	E6	0	0
	E7	0	0
SLQ	E8	0	0
	E9	0	0

	Inst.j	Pj x Tj (TCP18)	Pj x Tj (EMULAT)
RLQ	EA	0	0
***	EB	0	0
SRL	EC	0	0
***	ED	0	0
RRL	EE	0	0
***	EF	0	0
CPK	F4	0,0001	0,0008
	F5	0	0
TST	F6	0	0
***	F7	0	0
SET	F8	0	0
***	F9	0	0
CLR	FA	0,0001	0,0009
	FB	0,0000	0,0003
SKA	F0	0,0000	0,0003
SKB	F1	0,0002	0,0023
TBK	F2	0,0000	0,0005
LKZ	F3	0	0

NOP	FF	0,0129	0,1452
LRR	04	0,6344	6,2525
SRC	0140	0,6667	8,5846
ANQ	0180	0	0

XOQ	0201	0	0

MAQ	0202	0,0001	0,0006
AAQ	0203	0	0

MAB	0204	0,1695	2,0669
LPS	0208	0	0

ORQ	0210	0,0000	0,0002
IVQ	0221	0	0

LOQ	0222	0	0

	Inst.j	Pj x Tj (TCP18)	Pj x Tj (EMULAT)
SOQ ***	0223	0	0
LOB	0224	0,0214	0,2559
LPC ***	0228	0	0
MQA	0240	0,0000	0.0004
MQE	0264	0,0000	0,0002
MZA	0280	0,0001	0,0017
TNE ***	0308	0	0
TNO	0310	0,0093	0,1228
ALI	0320	0,0090	0,1073
BKI	0340	0,0854	1,0182
TPB	0360	0,0001	0,0011

INA	D8	0	-
	D9	0,0005	-
INF ***	DA	0	-
	DB	0	-
OTA ***	DC	0	-
	DD	0	-
OTF	DE	1,0742	-
	DF	0	-
HLT ***	00	-	-
TRC ***	0120	-	-
TSA ***	0301	-	-
TSB ***	0302	-	-
TSC ***	0304	-	-
SPM	0380	-	-

ISC	FE0X	-	-
Voor laboratorium doeleinden !!			
RCS	FED0	-	-

Pj x Tj (TCP18) = 660,9053

Pj x Tj (EMULAT) = 7924,8905

0.5 *Berekening van de $P_j \times T_j$ (TCP18) & $P_j \times T_j$ (EMULET) met telefonie*

Inst.j	$P_j \times T_j$ (TCP18)	$P_j \times T_j$ (EMULAT)
RST		
DIRECT 08	0	0
GEMOD. 09	28,1712	144,4814
INDIR. 0A	0	0
IND.G. 0B	0	0
JRT 10	17,8397	186,0388
11	0	0
12	19,9699	182,1036
13	0,0785	0,6627
JMP SKP 18	0,0704	1,7324
19	0,0096	0,1800
1A	0,0088	0,1437
1B	0,0001	0,0011
CMP 20	0,9189	13,0752
21	0,2120	2,5006
22	0,3074	3,5501
23	0,0929	1,0941
JBN 28	0,0030	0,0613
29	0	0
2A	0	0
2B	0	0
SUB 30	0,0818	1,2856
31	0	0
32	0,8638	10,6054
33	0,0305	0,3820
JAZ SKZ 38	0,2659	6,7771
39	0	0
3A	0	0
3B	0	0
AND 40	2,5005	35,7347
41	0,4572	5,2532
42	0,1525	1,7258
43	0,6334	7,3239
JSR 48	0,2619	3,1510
49	0	0
4A	6,4791	70,6807
4B	0	0
LDB 50	8,0920	114,4656
51	4,4660	50,6688
52	7,9061	88,6850
53	4,4950	51,5294

	Inst.j	Pj x Tj (TCP18)	Pj x Tj (EMULAT)
STB	58	4,4906	66,5626
	59	0,1210	1,4703
	5A	6,3081	74,3055
	5B	0	0
LDQ	60	0,7488	11,6281
	61	0,8096	10,3040
	62	252,7809	3072,3208
	63	0,1120	1,3440
STQ	68	1,1859	17,5788
	69	0,8917	10,8312
	6A	0,1704	2,0076
	6B	0,6764	8,1285
LDA	70	9,0526	128,0526
	71	13,8345	156,9585
	72	10,2295	114,7474
	73	7,4622	85,5451
STA	78	8,9332	132,4129
	79	7,2594	88,1814
	7A	3,4664	40,8321
	7B	1,7305	20,7971
LDR	80	0,0035	0,0524
	81	0,1676	2,0239
	82	0,0006	0,0065
	83	0,0009	0,0109
STR	88	0	0
	89	0	0
	8A	0,0014	0,0155
	8B	0	0
LDL	90	0,0013	0,0190
	91	0	0
	92	0,0004	0,0048
	93	0,0009	0,0109
STL ***	98	0	0
	99	0	0
	9A	0	0
	9B	0	0
XOR	A0	0,2424	3,4647
	A1	1,3988	16,0730
	A2	0,0029	0,0323
	A3	0,0337	0,3902
STX	A8	0	0
	A9	0	0
	AA	0,0014	0,0158
	AB	0	0

Inst.j		Pj x Tj (TCP18)	Pj x Tj (EMULAT)
ADD	B0	0,6653	10,4026
	B1	0,8606	11,0475
	B2	0,0578	0,7063
	B3	0,4725	5,8939
ADO	B8	1,8007	26,5587
	B9	0,7034	8,7995
	BA	144,5159	1745,6147
	BB	0,0152	0,1866
ORA	C0	0,6530	9,3314
	C1	0,4501	5,1723
	C2	0,1765	1,9976
	C3	0,1155	1,3357
TNZ	C8	55,0203	784,2901
	C9	1,9299	23,1875
	CA	2,6876	31,5007
	CB	2,6838	32,0220
LDK	D2	0,0067	0,0844
GEM.	D3	0,0004	0,0062
MBA	D300	0,0002	0,0024
LKQ	D6	0,0010	0,0122
GEM.	D7	0,0108	0,1429
MBQ	D700	0,0108	0,1324
ADK	FC	0,0134	0,1708
GEM.	FD	0,0002	0,0020
ABA	FD00	0,0001	0,0011
LKB	D4	0,0001	0,0011
AKB	D5	0,0004	0,0050
SRQ	D0	0,0011	0,0039
	D1	0	0
SRA	E0	0,0120	0,0586
	E1	0,0002	0,0008
RRA	E2	0	0
	E3	0,0001	0,0004
SLA	E4	0,0015	0,0075
	E5	0,0001	0,0003
SLL	E6	0,0006	0,0169
	E7	0,0003	0,0083
SLQ	E8	0,0013	0,0048
	E9	0	0

	Inst.j	Pj x Tj (TCP18)	Pj x Tj (EMULAT)
RLQ	EA	0,0001	0,0004
	EB	0	0
SRL	EC	0,0003	0,0072
	ED	0	0
RRL	EE	0,0005	0,0181
	EF	0	0
CPK	F4	0,0006	0,0065
	F5	0	0
TST	F6	0,0017	0,0247
	F7	0,0055	0,0849
SET	F8	0,0001	0,0017
	F9	0,0048	0,0843
CLR	FA	0,0003	0,0054
	FB	0,0002	0,0040
SKA	F0	0	0

SKB	F1	0,0001	0,0016
TBK	F2	0,0003	0,0031
LKZ	F3	0	0

NOP	FF	0,0092	0,1037
LRR	04	1,2157	11,9804
SRC	0140	0,2691	3,4647
ANQ	0180	0,0003	0,0032
XOQ	0201	0,0001	0,0007
MAQ	0202	0,0011	0,0135
AAQ	0203	0,0001	0,0018
MAB	0204	0,1397	1,7037
LPS	0208	0	0

ORQ	0210	0,0001	0,0016
IVQ	0221	0	0

LOQ	0222	0	0

	Inst.j	Pj x Tj (TCP18)	Pj x Tj (EMULAT)
SOQ	0223	0,0001	0,0006
LOB	0224	0,0215	0,2559
LPC	0228	0	0

MQA	0240	0,0006	0,0079
MQB	0264	0,0171	0,2084
MZA	0280	0,0000	0,0007
TNB	0308	0	0

TNO	0310	0,0092	0,1216
ALI	0320	0,0108	0,1293
BKI	0340	0,1007	1,1999
TPB	0360	0,0007	0,0084

INA	D8	0	-
	D9	0,0921	-
INF	DA	0	-
***	DB	0	-
OTA	DC	0,0001	-
	DD	0	-
OTF	DE	1,0878	-
	DF	0,1415	-
HLT	00	-	-

TRC	0120	-	-

TSA	0301	-	-

TSB	0302	-	-

TSC	0304	-	-

SPM	0380	-	-

ISC	FEOX	-	-
Voor laboratorium doeleinden !!			
RCS	FEDO	-	-

Pj x Tj(TCP18) = 643,9897

Pj x Tj (EMULAT) = 7619,9463

APPENDIX B

VERSLAG EXAMENZITTING 24 September 1987

Verslag examenzitting 24 September 1987

Prof. Goebertus, voorzitter van de examencommissie, heeft de zitting geopend, waarna de volgende vragen gesteld werden :

Vraag :

Waarom worden de memory reference instructies niet voorberekt, b.v de X-bits op volgorde zetten ?.

Antwoord :

De memory reference instructies zijn wel voorberekt. Namelijk de waarde van de X-bits, en wel in de volgorde $x_4x_3x_2x_1$, is voor elke mogelijke waarde (2048 verschillende waarden) die het adresveld van een memory reference instructie kan hebben van te voren bepaald.

(Zie in het verslag pag. 31 paragraaf 3.5.4).

In de functie waarin een bepaalde mem. ref. instructie, met een bepaalde adresveldwaarde, geemuleerd staat, staat ook -zonder enige logische operatie te ondernemen- de waarde van de X-bits van het adresveld van deze instructie vermeld.

Vraag :

Waarin zit het verschil in cycli tussen directe en gemodificeerde adressering ?.

Antwoord :

Bij gemodificeerde adressering, voor het bepalen van het operand adres, worden de X-bits als 20ste, 19de, 18de en 17de bits gebruikt en de waarde van deze bits is bekend.

Bij directe adressering worden i.p.v. de X-bits de $R_4R_3R_2R_1$ bits van het relocatie register gebruikt. In dit geval, voor het bepalen van het uiteindelijke finale operand adres, moeten eerst op de inhoud van het relocatie register, ter bepaling van de $R_4R_3R_2R_1$, enkele logische operaties plaatsvinden, voordat de optelling van de inhoud van het relocatie register met de waarde van het instructie adresveld plaats vindt.

Vraag :

Maar R register behoeft niet zo precies geemuleerd te worden, want het wordt met een speciale instructie gevuld zodat maskeren e.d. niet nodig is. Dit zou wellicht een factor 2 of 3 opleveren.

Antwoord :

Bij directe adressering is, ter bepaling van het finale operand adres, bit manipulatie op het relocatie register onvermijdelijk. Vandaar dat het aantal cycli groter is dan het aantal cycli bij gemodificeerde adressering.

Vraag :

De LDQ, ADO en TNZ instructies worden zo veel gebruikt. Met telefonie echter iets minder, hoe komt dat?.

Antwoord :

In de "idle loop" wordt, zowel in de toestand met als in de toestand zonder telefonie, voornamelijk op het geheugen getest, waarbij deze instructies noodzakelijk zijn. Daar het systeem in de toestand met telefonie minder tijd in de "idle loop" verkeert, worden bovengenoemde instructies dus ook minder gebruikt.

Vraag :

De vertragingfactor is 12. Zou meer dan een micro dit aankunnen ?.

Antwoord :

Dat zou kunnen maar het is niet nodig om meerdere micro's te gebruiken. Met alleen de voorgestelde maatregelen kan de VF, althans in theorie, tot 1 worden teruggebracht.

Vraag :

Nergens in het verslag wordt gesproken over het doel, nl. de performance die gehaald moet worden. In de opdracht wordt gesproken over Container Centrales. Wat zou een acceptabele VF zijn voor een Container Centrale?.

Opmerking:

Een Container Centrale bevat 2000-4000 abonnees, gemiddeld 2500. Dit zou een VF ongeveer moeten opleveren.

Antwoord :

Als we aannemen dat ook voor een Container Centrale hetzelfde telefonie pakket als voor de PRX/A zal worden gebruikt, zal een VF 4 niet acceptabel zijn. Dit omdat dan de "basis load", die ongeveer 25% is, ook ongeveer een factor 4 trager zal zijn. Dit zou betekenen dat er geen tijd voor telefonie verwerking resteert. Een VF 2 zou net acceptabel kunnen zijn.

Vraag :

Moet interrupt afhandeling niet geemuleerd worden?.

Antwoord :

Normaal wel, maar aangezien interrupt zelf hardware is, is dit met alleen software niet mogelijk.

Vraag :

Op pagina 42 staat, voor short shift instructies, dat $n = 8$ niet gedefinieerd wordt, is dat zo?.

Antwoord :

Het geval $n = 8$ kan normaal gesproken voorkomen. In de literatuur betreffende het bepalen van de executie tijd van de short shift instructies blijkt dit echter niet.

Zou ik het geval $n = 8$ wel hebben meegerekend, dan zou de gevonden gemiddelde executie tijd voor deze instructies i.p.v. 6.66 microseconden ongeveer 6.87 zijn geweest. Deze executie tijd te hebben gebruikt, zou geen invloed hebben gehad op de gevonden gemiddelde VF.

Vraag :

Uit de systeem configuratie op pagina 44 en 45 blijkt dat er geen interne calls gesimuleerd zijn, is dat zo?.

Antwoord :

Dat klopt. Op de testvloer moest een keuze worden gemaakt. Namelijk voor intern verkeer of voor uitgaand verkeer. Er is gekozen voor uitgaand verkeer om zo veel mogelijk typen signaleringen te kunnen gebruiken.

Vraag :

Hoeveel trunks zijn er per signalering gebruikt?. Was het systeem single of multi?.

Antwoord :

a)

Het aantal trunks per signalering die op de testvloer worden gebruikt, varieert van minimaal 1 tot maximaal 8.

Van de in de "junction groep" opgenomen signaleringen is het echter moeilijk om per signalering precies het aantal trunks na te gaan.

b)

Ik heb op een "dual mono" systeem gemeten daar ik niet over de mogelijkheid beschikte om metingen op een multi systeem te verrichten.

Gezien het feit dat beide processoren exact hetzelfde werk doen en het feit dat ik alleen over een logic analyzer beschikte, heb ik uiteindelijk op een processor gemeten.

Als ik op een multi systeem metingen zou hebben verricht, dit kon dus niet, zou ik eventueel een iets ander beeld hebben gekregen, maar ook in dit geval zou ik slechts op de main processor hebben gemeten en daardoor geen uitspraak over de secundaire processor hebben kunnen doen.

Vraag :

Uit de metingen blijkt dat de LDQ, ADO en TNZ instructies de meest voorkomende instructies zijn. Echter, deze instructies worden hoofdzakelijk in de "idle loop" gebruikt. Wat zou de VF in de toestand met zuivere telefonie zijn geweest?.

Graag dit nader onderzoeken.

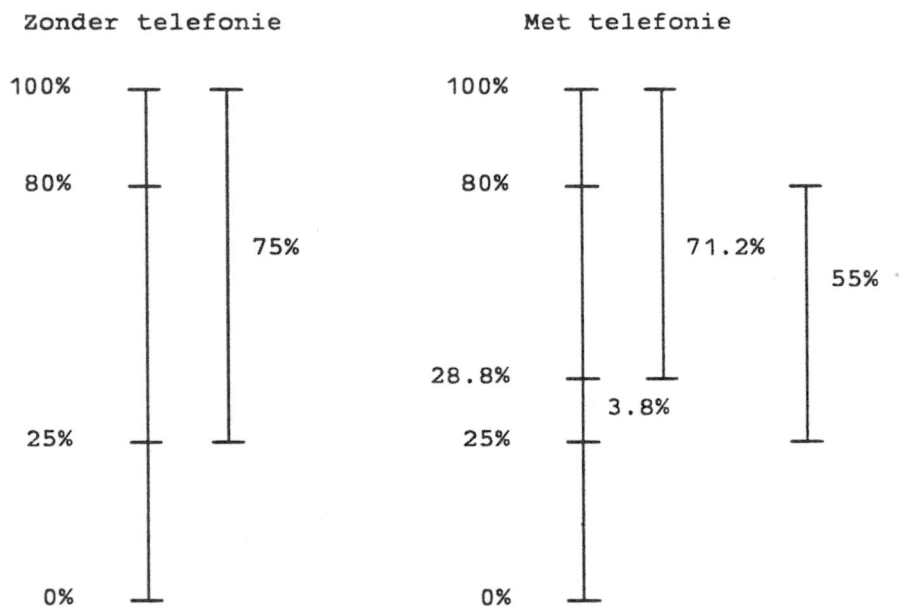
Antwoord :

Ik heb dit zo goed mogelijk onderzocht aan de hand van hypothesen die gebaseerd zijn op ervaringsfeiten.

Ik heb nl. aangenomen dat, zowel in de toestand zonder telefonie als in de toestand met telefonie, het systeem bij "basis load" 25% belast is.

Tevens heb ik aangenomen dat het systeem gemiddeld 80% belast is. Uitgaande van deze veronderstellingen, blijft voor telefonie en idel loop samen een belastingsmarge over van 55%.

Dit wordt in de onderstaande figuur schematisch weergegeven.



Het aantal calls per tijdseenheid dat tijdens de metingen is gemaakt, is 0.7 calls/seconde. We weten bovendien dat de PRX/A in dual mode in staat is maximaal 10 calls/seconde te verwerken. Hieruit is af te leiden in welke mate het systeem belast is door zuivere telefonie.

$$(0.7/10) \times 55\% = 3.8\% \text{ systeem belasting}$$

Dat wil onder meer zeggen dat het systeem in totaal (i.e. basis load + telefonie) voor 28.8% belast is.

Het totale aantal LDQ's, ADO's en TNZ's instructies die, zowel in de idle loop als in de toestand (telefonie + idle loop), in een minuut zijn uitgevoerd worden hieronder vermeldt.

<u>Idle loop</u>	<u>Telefonie + Idle loop</u>
LDQ : 5738344	LDQ : 4944604
ADO : 2842098	ADO : 2417162
TNZ : 1492546	TNZ : 1269965

Het aantal LDQ's , ADO's en TNZ's instructies dan, die voor zuivere telefonie zijn gebruikt, is:

$$\text{LDQ's} = ((71.2 : 75) \times 5738344) - 4944604 = 502299$$

$$\text{ADO's} = ((71.2 : 75) \times 2842098) - 2417162 = 280936$$

$$\text{TNZ's} = ((71.2 : 75) \times 1492546) - 1269965 = 146958$$

Aan de hand van de hieboven berekende aantallen LDQ's, ADO's en TNZ's instructies bij zuivere telefonie, zijn de in paragraaf 0.3 APPENDIX A genoemde percentages herberekend. Daarna, aan de hand van deze nieuwe percentages, zijn de in paragraaf 0.5 APPENDIX A genoemde executie-tijden $P_j \times T_j$ (TCP18) en $P_j \times T_j$ (EMULATOR) eveneens herberekend. Vervolgens zijn, zowel voor de EMULATOR als voor de TCP18, de nieuwe totale executie-tijden bepaald.

$$\sum_{j=1}^I P_j \times T_j \text{ (TCP18)} = 630,5616 \text{ microsec.}$$

$$\sum_{j=1}^I P_j \times T_j \text{ (EMULATOR)} = 7057,6860 \text{ microsec.}$$

De nieuwe vertragsingsfactor VF van de EMULATOR t.o.v. de TCP18 voor zuivere telefonie, is dan :

$$VF = 11.1927 \text{ bij zuivere telefonie}$$

Bij telefonie plus idle loop is de EMULATOR een factor 11.8324 trager dan de TCP18.

Het al of niet meerekenen van de "idle loop", bij het bepalen van de VF, heeft slechts een geringe invloed op de uitkomst. De VF blijft hoog.