Hardware debugging van het ISDN terminal board en software ontwikkeling t.b.v. de fysieke laag en protocol monitoring

Boormans, E.J.E.T.

Award date:
1993

Technische Universiteit Eindhoven
Faculteit der Elektrotechniek
Vakgroep Digitale Informatiesystemen

Hardware debugging van het ISDN terminal board en software ontwikkeling t.b.v de fysieke laag en protocol monitoring

E.J.E.T. Boormans
Afstudeerverslag

Eindhoven, Juli 1993
Hoogleraar: Prof. ir. M.P.J. Stevens
Begeleider: Ir. M.J.M. van Weert
Periode: Oktober 1992 - Juli 1993

Samenvatting

In de nabije toekomst zal het telefoonnetwerk vervangen worden door het Integrated Services Digital Network (ISDN). Dit is een volledig digitaal netwerk dat alle bestaande diensten en een aantal nieuwe diensten zal integreren.

Bij de vakgroep Digitale Informatie Systemen is een terminal board ontwikkeld dat op het ISDN aangesloten kan worden. Dit terminal board is eerst gebouwd op een quick-connect board zodat gemakkelijk wijzigingen aangebracht konden worden. Vervolgens is een print gemaakt. Dit verslag beschrijft onder andere het testen van deze terminal board print en de problemen die daarbij naar voren zijn gekomen.

Tijdens het functioneel maken van het terminal board is de ISDN software verder ontwikkeld op twee PC's, waarin twee Mitel ISDN expres kaarten zijn gestoken. Deze Mitel kaarten maken het mogelijk om een verbinding op te zetten en op deze manier de software te testen. Deze software is omgezet naar het terminal board. Dit verliep niet zonder problemen daar de software, ontwikkeld op de Mitel kaarten, gebruik maakt van het DOS operating systeem en het terminal board niet.

Teneinde de op de Mitel kaarten ontwikkelde ISDN software te testen, is het terminal board verbonden met een terminal. Deze terminal maakt het mogelijk dat de ISDN protocollen, geïmplementeerd op het terminal board, op een interactieve manier geobserveerd en gemanipuleerd kunnen worden. Dit verslag beschrijft ondermeer de manier waarop de communicatie tussen de terminal en het terminal board is gerealiseerd. Verder wordt ook het protocol monitoring programma besproken.

De ISDN software, die op het terminal board draait, communiceert via het uitwisselen van pakketten met de netwerkkant van het ISDN. Het verzenden en ontvangen van deze pakketten is hardware afhankelijk. De software die dit bestuurt kan niet van Mitel kaarten overgezet worden naar het terminal board. Hiervoor zijn nieuwe routines geschreven die ook in dit verslag beschreven worden.

Het ISDN terminal board project heeft een tijdje stil gelegen wat betreft de hardware ontwikkeling. Momenteel is het terminal board gereed om verbonden te worden met één van de PC's met Mitel kaart, zodat de ISDN software en de functionaliteit van het terminal board volledig getest kunnen worden.

Inhoudsopgave

Hoofdstuk 1 Inleiding

Al jarenlang is het telefoonsysteem de voornaamste internationale infrastructuur voor communicatie. Daar het telefoonnetwerk het meest verspreide netwerk is en over de wereld de meeste abonnees kent, wordt dit netwerk tegenwoordig ook gebruikt voor andere diensten, zoals facsimile, datacommunicatie en teletex. Het telefoonnetwerk is echter ontworpen voor het transporteren van analoge signalen en blijkt minder geschikt te zijn voor deze nieuwe vormen van communicatie.

In de nabije toekomst wordt de introductie van het Integrated Services Digital Network (ISDN) verwacht. Dit is een volledig digitaal netwerk waarin alle reeds bestaande communicatiediensten worden geïntegreerd. Verder zal het ISDN de introductie van een aantal nieuwe diensten mogelijk maken. Het ISDN zal een einde maken aan het inefficiënt gebruik van het telefoonnetwerk. De communicatiediensten zullen profiteren van een grotere bandbreedte.

Een 'basic rate' ISDN aansluiting bestaat uit twee B-kanalen van elk 64 kbps en één D-kanaal van 16 kbps. De B-kanalen zijn voor de overdracht van spraak of data. Het D-kanaal wordt gebruikt voor de overdracht van signalerings-informatie.

Bij de vakgroep Digitale Informatie Systemen is een terminal board ontworpen, dat met behulp van een 'basic rate' aansluiting aan het ISDN gekoppeld kan worden. Dit terminal board is eerst gebouwd op een quick-connect board, zodat gemakkelijk wijzigingen doorgevoerd konden worden. Inmiddels is van de layout van het terminal board een print gemaakt.

Dit verslag beschrijft de hardware debugging en het testen van alle componenten op de terminal board print. Verder is software ontwikkeld t.b.v. van de fysieke laag en protocol monitoring. Deze software ontwikkeling wordt in de volgende alinea's toegelicht.

Tijdens het functioneel maken van deze terminal board print is de software, die de ISDN protocollen implementeert, verder ontwikkeld op twee PC's waarin ISDN Mitel express kaarten zitten. De Mitel kaarten kunnen een ISDN netwerk emuleren. Met deze kaarten is het mogelijk om een verbinding op te zetten en op deze manier de software te testen. Deze ISDN software moet omgezet worden van de Mitel kaarten naar software voor

het terminal board, alvorens deze ISDN software uitgebreid kan worden.

Het is van groot belang dat de ISDN software, die op het terminal board draait, tijdens de ontwikkeling geobserveerd en gemanipuleerd kan worden teneinde de protocollen te testen. Daarvoor is een terminal aangesloten op het terminal board. Dit verslag beschrijft eveneens de software die de communicatie met deze terminal verzorgt. Tevens is een protocol monitoring programma geschreven dat het mogelijk maakt om op een interactieve manier de protocollen te observeren en te manipuleren.

Het gebruikte Link Access Protocol voor het D-kanaal (LAPD) zorgt voor de uitwisseling van signaleringsdata. Deze signaleringsdata worden in pakketten over het D-kanaal verzonden. Dit verslag behandelt tevens de software die zorgt voor het correct verzenden van deze pakketten over het D-kanaal.

Deze opdracht kenmerkt zich door zijn breedheid. Niet alleen moet inzicht verkregen worden in de layout en de complexe bouwstenen van het terminal board om de hardware problemen op te lossen, maar tevens moet inzicht verkregen worden in de protocollen van de ISDN software teneinde een goed en werkend protocol monitoring programma en routines t.b.v. de fysieke laag te schrijven.

Hoofdstuk 2 Integrated Services Digital Network (ISDN) en het terminal board

2.1 ISDN

2.1.1 Inleiding

De laatste 15 jaar zijn er steeds meer vormen van communicatie verschenen. Jarenlang is het telefoonsysteem de voornaamste internationale infrastructuur voor communicatie geweest. Het telefoonsysteem is ontworpen voor het transporteren van analoge spraaksignalen en blijkt minder geschikt te zijn voor nieuwe communicatiemogelijkheden, zoals datatransmissie, facsimile en video. De behoefte aan deze nieuwe soorten van communicatie heeft geleid tot een internationaal voornemen om het analoge telefoonsysteem te vervangen door een volledig digitaal netwerk. Dit nieuwe systeem heet ISDN (Integrated Services Digital Network) en heeft als doel het integreren van spraak- en andere diensten. De coördinatie van het ontwerp van dit nieuw communicatie netwerk is in handen van de CCITT en zijn vele studiegroepen. Zij bereiden in subcommissies aanbevelingen voor, die worden voorgelegd aan de Plenaire Zitting, die om de vier jaar wordt gehouden.

Bij de vakgroep Digitale Informatie Systemen is een terminal board ontworpen dat een gebruiker toegang kan verschaffen tot het ISDN netwerk.

In dit hoofdstuk wordt, na een inleiding over ISDN, een beschrijving gegeven van het gebruikte operating systeem en het ontwikkelde terminal board.

2.1.2 Diensten

De belangrijkste dienst blijft de overdracht van spraak, hoewel daar vele mogelijkheden aan worden toegevoegd. Eén van die mogelijkheden is het op een schermpje laten zien van het telefoonnummer, naam en adres van degene die opbelt, alvorens de telefoon wordt opgenomen. Een andere mogelijkheid is het verbinden van de telefoon met een computer zodat automatisch het databasebestand van degene die opbelt voor de beller zichtbaar wordt. Verder maakt ISDN het ook mogelijk dat degene die tevergeefs opbelt, omdat iemand bijvoorbeeld niet thuis is, een boodschap kan achterlaten.

Een aantal diensten die nu via het telefoonnet gerealiseerd zijn, zullen binnen ISDN

van een grotere bandbreedte gebruik kunnen maken dan momenteel het geval is.

Eén van de voordelen van ISDN is dat naast bijvoorbeeld een telefoonverbinding tegelijk ook een dataverbinding tot stand gebracht kan worden. Op deze manier kunnen bijvoorbeeld tijdens een telefoongesprek files overgezonden worden via een computerverbinding.

Eén van die diensten die nu in Frankrijk via het telefoonnet gerealiseerd is, is videotex. Dit systeem geeft de gebruiker interactieve toegang tot een database op afstand. Met dit systeem wordt ook reservering voor vliegtuigen, treinen, hotels, theaters en talrijke andere toepassingen mogelijk.

Een andere dienst, waarvan men verwacht dat het veel gebruikt wordt, is teletex. Dit is een vorm van elektronische post voor gebruik in huis of in een bedrijf.

Een dienst die ook nu al via het telefoonnet mogelijk is, is facsimile (fax).

Verder zijn er nog telemetrie- of waarschuwingsdiensten. Deze diensten hebben maar een kleine bandbreedte nodig. Tot deze diensten behoren bijvoorbeeld vuurdetectoren in huizen of bedrijven die automatisch de brandweer opbellen en die zichzelf kunnen identificeren.

2.1.3 ISO OSI referentie model

De normale gebruikersaansluiting op het ISDN netwerk bestaat uit twee B kanalen van 64 kbits per seconden en één D kanaal van 16 kbits per seconde. Dit heet een basic-access aansluiting. De B kanalen dienen voor gedigitaliseerde spraak- of datatransmissie. Het D kanaal zorgt voor signalering, d.w.z. het opzetten, afbreken en onderhouden van een verbinding, of voor een pakketdienst.

De apparatuur die op een ISDN netwerk wordt aangesloten via een dergelijk 'S'-referentie punt (basic access) moet voldoen aan bepaalde eisen. Het bij de vakgroep Digitale Informatie Systemen ontworpen terminal board voorziet in een dergelijke aansluiting. De CCITT heeft de eisen verdeeld in lagen volgens het zogenaamde ISO (International Standards Organization) OSI (Open System Interconnection) referentie model. Voor deze lagen zijn protocollen gedefinieerd die hoofdzakelijk betrekking hebben op het D kanaal. Zoals in figuur 1 te zien is zijn er zeven lagen gedefinieerd [19].

De fysieke laag houdt zich bezig met het overzenden van bits via een communicatie kanaal. Deze laag heeft alles te maken met het representeren van bits in voltages.

De datalink-laag heeft als taak het bieden van een foutvrije transmissielijn aan de

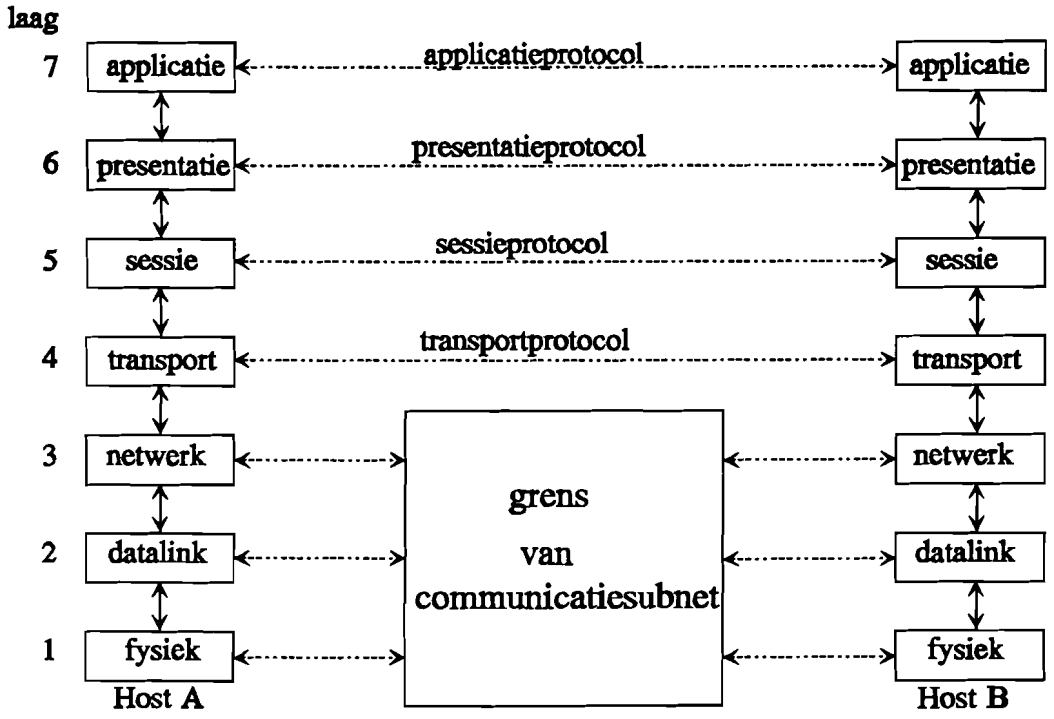

fig.1 Osi referentie model

netwerk laag. Eventuele fout ontvangen dataframes kunnen opnieuw verzonden worden door het toepassen van bevestigingsframes. Als in een bepaalde tijd de zender geen bevestigingsframe heeft ontvangen dan weet deze dat het zojuist verzonden frame fout is aangekomen. De zender reageert hierop door het frame nog een keer te zenden.

De netwerk laag houdt zich bezig met het regelen van de werking van het subnet. Een belangrijke taak van de netwerk laag is het bepalen langs welke route de pakketten van bron naar bestemming moeten gaan. Een andere taak van de netwerk laag is het oplossen van een te hoge belasting van het netwerk, genaamd congestie beheer.

De functie van de transportlaag is het accepteren van data afkomstig van de sessielaag en indien nodig het te splitsen in kleinere delen. Op zijn beurt geeft de transportlaag zijn data weer door aan de netwerklaag en zorgt ervoor dat alle datadelen correct aan de overkant aankomen.

De sessielaag zorgt ervoor dat gebruikers van verschillende machines in staat zijn een sessie tot stand te brengen. Gedurende een sessie kunnen data worden overgebracht, zoals in de transportlaag maar er zijn ook enkele verdergaande diensten die voor bepaalde toepassingen nuttig zijn. Een van de diensten die de sessielaag kan bieden is het regelen

van een dialoog.

De presentatielaag zorgt voor het converteren van data volgens de gebruikte standaarden. Andere aspecten van het representeren van informatie, zoals datacompressie of cryptografie, behoren ook tot de taken van de presentatielaag.

De applicatielaag biedt diensten aan aan de gebruiker, zoals het overbrengen van files, elektronische post en andere gespecialiseerde faciliteiten.

Op iedere laag zijn door de CCITT een aantal entiteiten gedefinieerd. Een entiteit is een soort toestand machine. Peer entiteiten kunnen met elkaar communiceren door middel van primitieven. Op ontvangst van een primitieve kan de betreffende entiteit van toestand veranderen. Door de heer Oudelaar is een operating systeem ontwikkeld, dat gebaseerd is op het uitwisselen van primitieven tussen entiteiten [16]. Dit operating systeem heeft de naam MBOS (Message Based Operating System) gekregen.

2.2 Operating systeem

2.2.1 MBOS (Message Based Operating System)

In MBOS worden de hierboven beschreven entiteiten voorgesteld door processen en primitieven door messages. Processen kunnen messages genereren. Centraal in figuur 2 staat de dispatcher. De dispatcher bepaalt aan de hand van het soort message en de toestand van het bestemmingsproces welke functie uitgevoerd wordt.

De hardware componenten op het terminal board zoals de DSC, IDPC en de ADMAC kunnen interrupts genereren. Deze interrupts worden door de interrupthandlers verwerkt. Op hun beurt kunnen de betreffende interrupthandlers indien nodig messages genereren. Een proces kan op de ontvangst van een message reageren door zelf een message te genereren.

In de programmeertaal, ansi-C, zijn processen en messages opgebouwd uit een aantal velden.

Een proces heeft een PublStat-veld dat aangeeft of het proces "Blocked" is of "Running". In de toestand "Blocked" kan het betreffende proces geen message ontvangen. In de toestand "Running" kan het betreffende proces wel een message ontvangen. Zie tabel 1.

Het Priorid-veld bepaalt de prioriteit van het proces. In een aantal protocollen zijn

tijdslimieten ingebouwd, hierdoor moeten bepaalde processen sneller reageren. Deze hebben dus een hogere prioriteit.

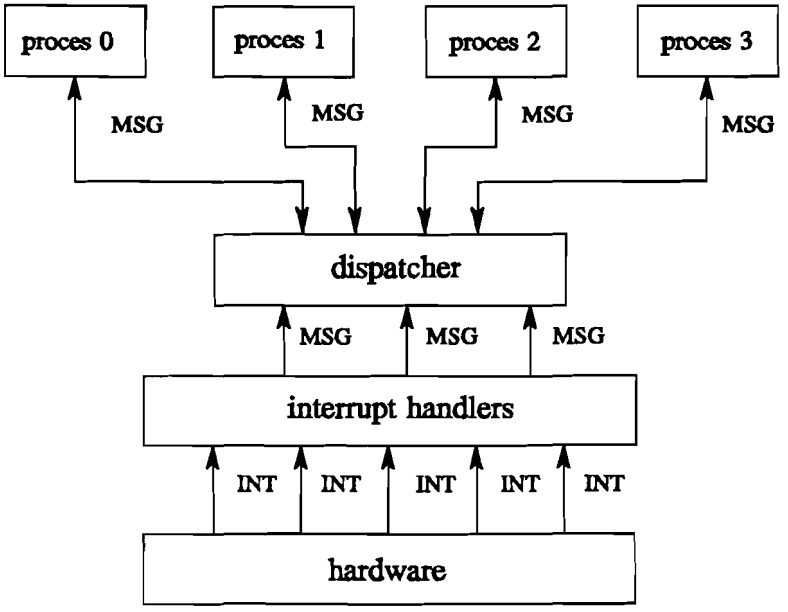

fig.2 MBOS (Message Based Operating System)

Tabel 1 Proces- en message velden

PROCES	MESSAGE
Publstat	Nucleo
Priority	Destination
State	Origin
Substate	Param_1
Param_1	Param_2
Param_2	...
...	Param_5
Param_7	

Het State-veld is een pointer die naar de state-tabel wijst die bij de huidige toestand van het proces behoort.

Het Substate-veld dient voor verdere onderscheiding van de toestand van het proces. Dit veld heeft alleen een betekenis voor bepaalde toestanden van laag twee processen.

De velden Data1..Data8 dienen voor het opslaan van procesparameters, die voor ieder proces een andere betekenis kunnen hebben.

Evenals een proces is in de programmeertaal een message opgebouwd uit een aantal velden. Zie tabel 1.

Het Nucleo-veld geeft aan welk bericht het is. Het Destination veld geeft aan voor welk proces het bericht bestemd is. Het Origin veld dient om aan te geven van welk proces het bericht afkomstig is. De velden Param_1 tot en met Param_5 worden gebruikt om extra parameters, zoals bijvoorbeeld het volgnummer van een frame in een sliding-window protocol [19], mee te geven.

Bij het afhandelen van een message wordt door de dispatcher de betreffende nucleo van de message in de state-tabel van het bestemmingsproces opgezocht. Vervolgens wordt dan de bijbehorende subroutine uitgevoerd. Hieronder is een voorbeeld van een state-tabel weergegeven.

```
State StateTabel[]=
{
        { Nucleo1, subroutine1 },
        { Nucleo2, subroutine2 },
        { Nucleo3, subroutine3 },
        { Nucleo4, subroutine3 }
};
```

2.3 Het terminal board

2.3.1 Historie van het terminal board

Het terminal board is ontworpen door P. Weterman (aug. '89) [20]. Hij heeft tevens het ontwerp gebouwd op een quick-connect board en gedeeltelijk getest.

H. Oudelaar (dec. '89) heeft zich in dezelfde periode bezig gehouden met de software kant van het terminal board [16]. Op laag één heeft hij routines geschreven die de

verschillende componenten op het terminal board, zoals de twee IDPC's, de DSC, de interrupt controller en de DMA coprocessor, aansturen. Verder heeft hij een Message Based Operating System (MBOS) ontwikkeld, dat de berichten, die door de protocollen gegenereerd worden, afhandelt. Op laag twee niveau heeft hij een aanzet gegeven tot het specificeren van het Link Access Protocol voor het D-kanaal (LAPD) protocol.

De heren C. Maastricht [14] en E. Koops (jan. '90) [11] hebben zich beiden ingezet om de hardware op het quick-connect board te testen en te verbeteren. C. Maastricht heeft een aantal testprogramma's in assembler taal geschreven, die de verschillende componenten testen. Deze testen zijn met behulp van de Intel 80186 emulator uitgevoerd. E. Koops heeft zich voornamelijk bezig gehouden met de ram controller.

J. Beijnsberger (feb. '90) [3] heeft ook meegeholpen aan het testen van de hardware. Op sofware gebied heeft hij een begin gemaakt met het schrijven van routines die de terminal aansturen via de Usart van één van de IDPC's. Via deze terminal is momenteel protocol monitoring mogelijk.

H. Jonkers (okt. '91) [10] heeft het 'S'-interface en het telefoon interface ontwikkeld.

Het functioneel maken van dit terminal board heeft nogal wat tijd gekost. Om met software toch verder te kunnen, zijn twee Mitel ISDN express insteekkaarten aangeschaft. Het is met deze kaarten mogelijk om een ISDN verbinding op te zetten en de protocollen uit te breiden en te testen.

De heren die met deze kaarten gewerkt hebben zijn S. van de Kuilen (jan. '90) [12], R. Lemmens (jan. '90) [13], D. van Egmond (dec. '91) [4] en L. Oorschot (aug. '92) [15]. S. van de Kuilen heeft gewerkt aan laag één, het aansturen van de componenten op deze kaarten. R. Lemmens heeft gezorgd voor de specificatie en implementatie van het laag drie signaleringsprotocol voor het ISDN D-kanaal. Hij heeft tevens een protocol monitoring programma aan het operating systeem toegevoegd. D. van Egmond heeft ook meegeholpen aan de software ontwikkeling op laag drie. L. Oorschot heeft deze software met behulp van het protocol monitoring programma onderworpen aan een grondige test en vervolgens de gevonden fouten verbeterd.

Momenteel is R. Claessens bezig met de implementatie van call control op laag vier en het aanpassen van laag drie zodat laag vier probleemloos kan functioneren.

Het testen van het terminal board gebouwd op een quick-connect board resulteerde in een definitieve layout. Van deze layout is een print gemaakt.

Mijn opdracht bestond uit het testen van deze print, het schrijven van interrupt routines, het omzetten van alle software die ontwikkeld was met behulp van de Mitel kaarten naar het terminal board en het schrijven van routines die nodig zijn voor protocol monitoring via de terminal.

2.3.2 Componenten van het terminal board

Zoals al eerder in dit verslag vermeld staat is bij de vakgroep Digitale Informatie Systemen een terminal board ontwikkeld dat aangesloten kan worden op een ISDN 'S'-referentie punt. Deze basic access aansluiting voorziet in twee B kanalen van ieder 64 kbits/s en in één D kanaal van 16 kbits/s. In figuur 3 is te zien uit welke componenten dit terminal board is opgebouwd.

De hardware bestaat uit de volgende componenten:

- Micro processor
- DMA controller
- Geheugen
- Interrupt Controller
- Digital Subscriber Controller (DSC)
- Integrated data protocol handler (IDPC)
- Parallelle poort
- Seriële poort

In de volgende paragraaf worden deze componenten afzonderlijk beschreven.

2.3.3 Hardware beschrijving

Microprocessor

De gebruikte microprocessor is een 80186 bestaande uit:

- een klok generator
- twee DMA kanalen
- een programmeerbare interrupt controller
- drie programmeerbare 16 bit timers.
- programmeerbare chip select logica
- programmeerbare wait state generator
- een lokale bus controller

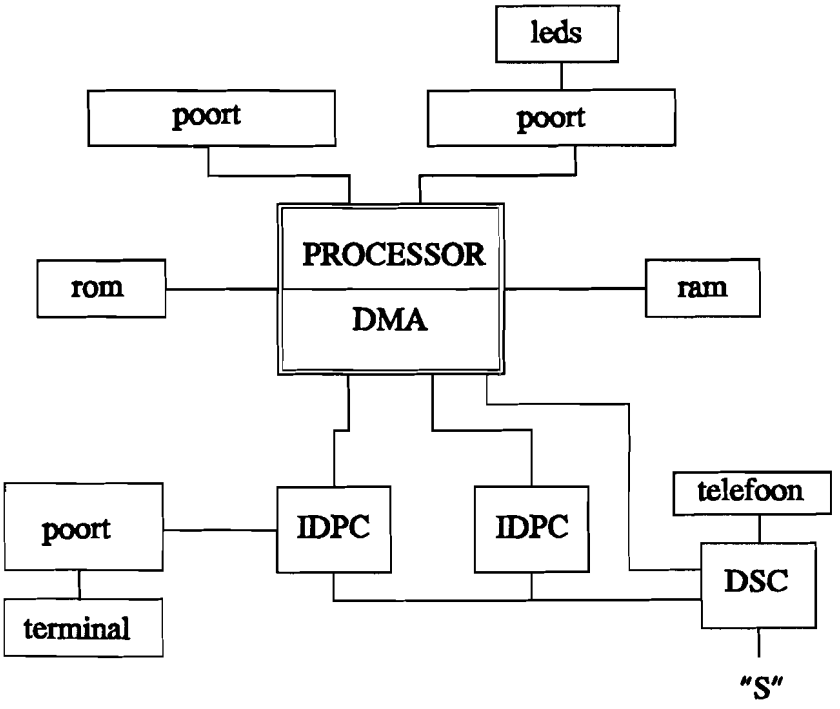

fig.3 Terminal board

De processor heeft een eigen interrupt controller die vier interrupt bronnen kan verwerken. Dit is echter niet voldoende. Een 8259A-2 Interrupt controller, die niet afgebeeld is in figuur 3, zorgt voor de benodigde extra capaciteit.

De microprocessor is voorlopig vervangen door een Intel 80186 emulator die in verbinding staat met een PC. Dit is gedaan om het debuggen van de software en hardware te vergemakkelijken.

Interrupt controller

De interrupt controller aan boord van de processor is in cascade mode geprogrammeerd. Op deze manier kunnen de INT1/INT3 lijnen van de microprocessor als interrupt request/acknowledge paar werken. Wanneer bij de interrupt controller (8259A-2) een interrupt binnen komt, wordt dit via een interrupt request kenbaar gemaakt aan de processor. Deze reageert met een interrupt acknowledge, waarna de interrupt controller (8259A-2) het interruptnummer op de databus zet.

In onderstaande tabel zijn de interrupt bronnen aangegeven die aangesloten zijn op de interrupt controller. De DLC en USART zijn delen van de IDPC die nog in deze paragraaf besproken worden. De laatste vier interruptbronnen zijn afkomstig van de DMA controller (ADMAC). Dit zijn de End Of Dma (EOD) interrupts voor de vier DMA kanalen.

Het programmeren van de interrupt controller (8259A-2) gaat door middel van Initialization Command Words (ICW's) en Operation Command Words (OCW's).

Tabel 2 De interrupt bronnen

PIN	BRON
IR0	DLC van IDPC 1
IR1	USART van IDPC 1
IR2	DLC van IDPC 2
IR3	DSC
IR4	EOD3
IR5	EOD2
IR6	EOD1
IR7	EOD0

Ram en rom

Er is 256K bytes rom en 512K bytes dynamisch ram aanwezig op het board. Een ramcontroller (8207) zorgt voor het aansturen van het in twee banken verdeelde ram. Er is gebruik gemaakt van 256k dynamische ram chips. Iedere bank bevat dus in werkelijkheid 256k woorden. Adreslijn nul wordt echter gebruikt om verschil te maken tussen byte of woord transfer en is bij iedere ram chip aangesloten op '1'. Dit betekent dat er maar 128k woorden toegankelijk zijn per bank. Verder is het geheugen voorzien van een pariteit generator en checker die op de Non Maskable Interrupt (NMI) van de 80186 microprocessor is aangesloten. In figuur 4 is een grafische weergave gegeven van de geheugen indeling en I/O indeling van de verschillende componenten op het terminal board.

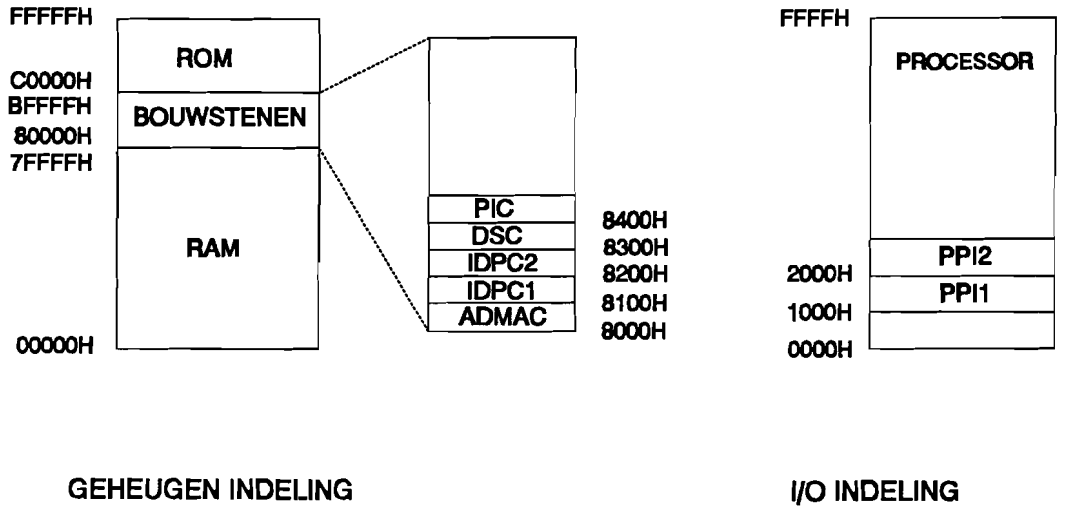

fig.4 Geheugen en I/O indeling

Digital Subscriber Controller (DSC)

De Digital Subscriber Controller (DSC) verzorgt de connectie van het terminal board met het 'S'-referentie punt. Verder verzorgt de DSC ook laag één en gedeeltelijke laag twee afhandeling van het gebruikte LAPD protocol. De DSC kan gebruikt worden als een telefoon, een digitale data terminal of een spraak en data terminal.

De DSC zal gedetailleerder besproken worden in hoofdstuk 5.

Integrated Data Protocol handler (IDPC)

De Integrated Data Protocol handler (IDPC) bestaat uit drie hoofdblokken. De Data Link Controller (DLC), de Universal Synchronous/Asynchronous Receiver/Transmitter (USART) en de Dual Port Memory Controller (DPMC). De DLC is het hart van de IDPC en kan gebruikt worden voor het afhandelen van bit georiënteerde protocollen zoals HDLC, SDLC, LAPB en LAPD. De USART kan gebruikt worden voor een terminal adaptor die bestaande terminals verbindt met netwerken die gebaseerd zijn op bit georiënteerde protocollen, zoals X.25, ISDN en SNA.

De Dual Port Memory Controller (DPMC) is nodig wanneer twee processoren gebruik maken van de systeembus. Deze DPMC zorgt er dan voor dat de twee processoren

toegang hebben tot een gemeenschappelijk stukje geheugen.

De USART van één van de twee IDPC's wordt gebruikt voor communicatie met een terminal.

In hoofdstuk 4 zal de IDPC nader besproken worden.

Advanced Direct Memory Access Coprocessor (ADMAC)

De 82258 DMA coprocessor kan zorgdragen voor het vullen en legen van de hardware buffers van beide IDPC's. Elke IDPC heeft twee First In First Out buffers (FIFO's): één voor de zend- en één voor de ontvangstkant. Er zijn dus vier DMA kanalen nodig.

De DMA coprocessor heeft vier DMA kanalen met elk een registerset. Verder zijn er nog een aantal registers die betrekking hebben op alle kanalen. Data overdracht wordt bestuurd door middel van channel command blocks. Deze bevatten het channel command word en andere parameters, die nodig zijn voor het uitvoeren van de data overdracht. De channel command blocks bevinden zich in het geheugen en ieder kanaal heeft een register waarin de pointer staat, die naar het begin van het betreffende channel command block wijst.

Een 'channel start opdracht' van de processor zorgt ervoor dat de DMA coprocessor het channel command block uitleest en opslaat in de interne registers. Na beëindiging van de opdracht wordt in het status register de reden van beëindiging gezet.

Programmable Peripheral Interface (PPI)

Op één van de twee parallelle poorten, PPI's, is een ledbalk aangesloten die gebruikt wordt om foutmeldingen in testprogramma's zichtbaar te maken. De andere parallelle poort kan in de toekomst gebruikt worden om een PC aan te sluiten op het terminal board. Deze PC kan dan de hogere lagen van het ISO-OSI referentie model afhandelen.

2.4 Samenvatting

In dit inleidende hoofdstuk zijn de diensten, die het ISDN te bieden heeft, en het ISO OSI referentie model beschreven. De eisen, waaraan de apparatuur, die op het ISDN wordt aangesloten, moet voldoen, zijn volgens dit ISO OSI referentie model opgedeeld in lagen. Deze lagen zijn uitgebeeld in figuur 1.

Bij de vakgroep Digitale Informatie Systemen is een terminal board ontwikkeld, dat aangesloten kan worden op het ISDN netwerk. Voor dit terminal board is een operating syteem ontwikkeld, MBOS, dat gebaseerd is op het uitwisselen van messages tussen processen. Na een overzicht van de historie van het terminal board, worden de hardware componenten, waaruit het terminal board is opgebouwd, in de laatste paragraaf beschreven.

Hoofdstuk 3 Hardware debugging van het ISDN terminal board

3.1 Inleiding

Ten behoeve van het testen van het terminal board zijn een aantal testprogramma's geschreven, die de componenten op het terminal board afzonderlijk testen. In dit hoofdstuk worden deze testprogramma's en de problemen, die bij het uitvoeren van deze testprogramma's naar voren zijn gekomen, beschreven.

3.2 Wijze van testen van de terminal board print

Er is begonnen met het op hardware niveau testen van alle componenten die op het terminal board aanwezig waren.

Met behulp van de Intel ICE™-186/188 In-Circuit Emulator is het mogelijk om produkten, die op Intel 80C186 of 80C188 gebaseerd zijn, hardwarematig en softwarematig te testen en te debuggen. Deze emulator bestaat uit een probe die op de plaats van de microprocessor in het board gestoken dient te worden. Om de emulator te programmeren moet deze via een RS232C kabel kunnen communiceren met een personal computer AT. De emulator controller unit zorgt voor deze communicatie en het aansturen van de probe. Testprogramma's kunnen in het geheugen van de emulator, dat zich op de controller unit bevindt, geladen worden. In het emulator programma is het mogelijk om het ingeladen programma te debuggen met behulp van breakpoints en de single step mogelijkheid. De emulator ondersteunt het inlezen van hexadecimale object files, files in SAVE formaat (SAVE is een emulator commando dat ingelezen files opslaat) en files in het Intel 8086 absoluut Object Module Formaat (OMF).

Om het terminal board in ontwikkelfase te testen, zijn een aantal testprogramma's in assembler (ASM 86) geschreven. De componenten die door deze programma's getest worden zijn:

Ram
Interrupt controller
Dma coprocessor
Dsc

Idpc

De testprogramma's gebruiken de ledbalken om aan te geven in welk gedeelte het programma zich bevindt en of er een fout is gedetecteerd.

3.3 Beschrijving van de testprogramma's

Ram testprogramma

Het ram testprogramma werkt volgens het marching 0/1's algoritme [14]. Het programma schrijft waarden naar het ram en leest ze vervolgens weer uit. Na het uitlezen worden de geschreven waarden en de uitgelezen waarden met elkaar vergeleken. Indien ze niet gelijk zijn, wordt een foutmelding naar één van de leds geschreven.

Interrupt controller testprogramma

Het interrupt controller testprogramma bestaat uit drie delen. Om aan te geven welk deel draait, wordt het nummer van het deel naar poort C geschreven en zichtbaar gemaakt op de ledbalken.

Als eerste deel van dit testprogramma wordt de eerder genoemde ram test uitgevoerd. Het tweede deel neemt de initialisatie van de processor, de interrupt controller en de interruptvectortabel voor zijn rekening. De processor moet zo ingesteld worden dat de lijnen INT1/INT3 als interrupt request/acknowledge paar werken. De interrupt controller wordt vervolgens middels ICW's en OCW's geïnitialiseerd. Vanaf adres 0H wordt daarna de interrupt vector tabel, bestaande uit 40 adressen van alle Interrupt Service Routines (ISR), opgebouwd.

Deel drie is het eigenlijke testgedeelte van de interrupt controller. Elke ISR schrijft zijn interruptnummer naar poort B. Op deze manier kan gecontroleerd worden of een bepaalde interrupt opgetreden is. Eerst wordt een 'divide by zero' interrupt gegenereerd. Daarna is een oneindige loop geprogrammeerd waarin met de hand interrupts gegenereerd kunnen worden. Dit kan gedaan worden door de interrupt pin met de positieve voedingsspanning te verbinden.

Advanced Direct Access Memory Controller (ADMAC)

Het Advanced Direct Access Memory Controller (ADMAC) 82258 testprogramma bestaat uit vier delen. Op poort A van de ledbalken wordt zichtbaar gemaakt dat testprogramma drie draait.

Het eerste deel is wederom het ram testprogramma.

In het tweede deel van dit testprogramma worden de interrupt controller en de processor geïnitialiseerd. Dit keer zijn er maar drie interrupt service routines aanwezig. Voor de NMI en de divide by zero interrupt zijn twee afzonderlijke interrupt service routines aanwezig. De derde routine is een lege routine, 'dummyret' genaamd, die aangeroepen wordt bij alle overige interrupts. Tevens verzorgt het tweede gedeelte de opbouw van de interrupt tabel vanaf adres 0H.

De initialisatie van de ADMAC wordt in het derde gedeelte gedaan. Het eerst wordt het geheugenbereik 30000H-40000H gevuld met 55H. Na de algemene initialisaties worden de 'channel command blocks' voor elk kanaal samengesteld. De 'channel command blocks' zijn zo samengesteld dat kanaal 0 32 bytes vanaf adres 30000H naar adres 40000H verplaats. Kanaal 1 verplaats 32 bytes vanaf adres 31000H naar adres 41000H. Kanaal 2 verplaats 32 bytes vanaf adres 32000H naar adres 42000H. En het laatste kanaal, kanaal 3, verplaats 32 bytes van adres 33000H naar adres 43000H. Deze verplaatsingen vinden allemaal binnen het ram geheugen plaats.

Het laatste programmadeel, deel vier, voert dan de eigenlijke DMA operaties op de vier kanalen uit. Dit gebeurt door een 'channel start commando' naar het General Command Register (GCR) te schrijven. De ADMAC copiëert vervolgens het 'channel command block' naar de interne registers en voert de DMA operatie uit. Met behulp van de emulator kunnen de dataverplaatsingen binnen het ram geheugen gecontroleerd worden. Deze controle wordt niet door het testprogramma uitgevoerd.

DSC testprogramma

Het vierde testprogramma is de DSC test. Dit programma bestaat ook weer uit drie delen. Deel één is weer de ramtest en deel twee de initialisatie van de interrupt controller en de processor. In dit programma heeft iedere interrupt weer zijn eigen interrupt routine en schrijft dus zijn eigen interrupt nummer naar de leds. Het derde deel is het DSC gedeelte. Het is niet mogelijk om een functionele test uit te voeren. Hiervoor zou namelijk implementatie van laag één en laag twee van het OSI referentie model nodig zijn. De DSC

test controleert alleen de bereikbaarheid van een aantal registers. Daartoe wordt in het INIT register de DSC in de 'idle' toestand gezet. In deze toestand kunnen de registers door de processor beschreven worden, zonder dat er door de DSC iets uitgevoerd wordt. Tenslotte worden de default waarden van sommige registers uitgelezen en gecopieerd naar het stukje geheugen dat start bij adres 10000H. In tabel 3 worden de default waarden van de registers die uitgelezen worden gegeven.

Tabel 3 Defaultwaarden van DSC registers

REGISTER	DEFAULTWAARDE
LMR1 en LMR2	00H
MF	00H
DMR1	80H
DMR2	00H
DMR3	03H
DMR4	0CH
MMR1 t/m MMR4	00H

IDPC testprogramma

Het vijfde programma test de IDPC. De ram test is hier weggelaten. De initialisatie van de interrupt controller en de processor is het eerste gedeelte. Verder bevat deze test de DSC test.

Voor de IDPC geldt hetzelfde als voor de DSC: om deze bouwsteen op zijn functie te kunnen testen is een implementatie van laag één en laag twee nodig van het OSI referentie model. Deze test beperkt zich dan ook tot het controleren van de bereikbaarheid van de registers. Deze controle wordt gedaan door middel van het marching 0/1's algoritme [14]. Op deze manier kunnen alle read/write registers getest worden.

3.4 Het testen van de componenten

Het uitvoeren van het ram testprogramma gaf problemen. Op de ledbalken werd een

foutmelding gegeven. Aangezien het ram testprogramma bestaat uit het beschrijven en lezen van geheugenplaatsen, is de fout gezocht in de uitvoering van een lees- of schrijfslag van de processor.

Voordat de fout, die met behulp van de Logic Analyser gevonden is, uitgelegd wordt, wordt eerst een leesslag van de processor toegelicht. Vervolgens wordt de uitvoering van de verschillende testprogramma's en de daarbij opgetreden problemen besproken.

Leesslag van de microprocessor

Een lees- of schrijfslag bij een 80186 bestaat uit vier of meer klokslagen. Na de derde klokslag kunnen door de processor waitstates ingevoegd worden. Waitstates maken het, voor langzamere componenten op het terminal board, mogelijk om te voldoen aan lees- of schrijfaanvragen van de processor. Een dergelijke langzame component is bijvoorbeeld de dynamische ramcontroller. Deze kan, op het moment dat de processor een aanvraag doet voor een leescyclus, nog bezig zijn met een refreshcyclus.

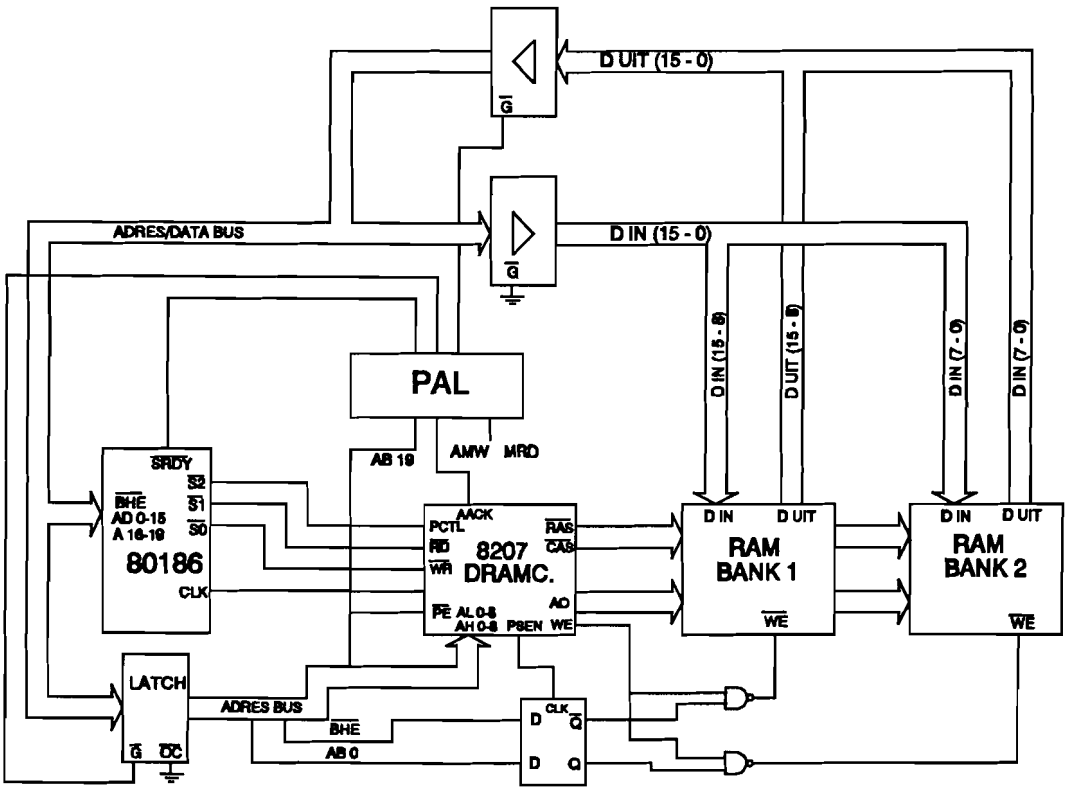

fig.5 Verbindingen tussen de processor en het geheugen

De processor maakt via de statuslijnen S0, S1 en S2 aan de ram- en de buscontroller kenbaar welke actie uitgevoerd dient te worden, zie figuur 5. De codes die voor het specificeren van deze acties voor S0, S1 en S2 gebruikt worden, worden gegeven in tabel 4. De statuslijnen zijn actief van halverwege de laatste klokslag van de vorige buscyclus tot halverwege de voorlaatste klokslag van de buscyclus, zie figuur 6. Bij het begin van de eerste klokslag is dus voor de ramcontroller en de buscontroller bekend welk operatie uitgevoerd dient te worden. De buscontroller genereert tijdens de eerste klokslag van een leescyclus een Address Latch Enable puls (ALE). Deze zorgt ervoor dat het geheugenadres van de locatie, die gelezen dient te worden, in de latches wordt geklokt.

Tabel 4 Processor status lijnen interpretatie

S2	S1	S0	Operatie
0	0	0	interrupt acknowledge
0	0	1	read I/O
0	1	0	write I/O
0	1	1	halt
1	0	0	instruction fetch
1	0	1	read memory
1	1	0	write memory
1	1	1	passive

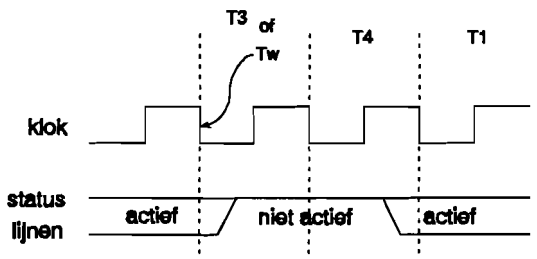

fig.6 Timing van de statuslijnen

Na de ALE puls wordt het adres, als de ramcontroller vrij is, in twee slagen aan het geheugen doorgegeven. De ramcontroller genereert eerst een Row Address Strobe (RAS) en

daarna een Column Address Strobe (CAS). Na de neergaande flank van de CAS puls is de betreffende geheugenlocatie geselecteerd en verschijnt de geheugeninhoud op de databus. Door de opgaande flank van de memory read lijn (mrd) wordt deze waarde in de processor geklokt. Deze flank valt samen met het begin van de vierde klokslag (T4).

De processor kijkt aan het begin van iedere derde klokslag (T3) en wait state (Tw) naar de synchronous ready lijn (SRDY). Als deze lijn actief is, d.w.z. '1', dan wordt de betreffende state onmiddellijk gevolgd door de laatste klokslag (T4). Als deze lijn niet actief is wordt de betreffende state echter gevolgd door een wait state (Tw). Het SRDY signaal wordt gegenereerd door een Programmable Array Logic (PAL). Deze PAL zorgt voor een delay. Als het SRDY signaal actief wordt, gebeurt dit altijd na de neergaande flank van de klok waarop de processor naar de SRDY lijn kijkt. Na de state, waarin het SRDY signaal actief wordt, moet dus nog een waitstate ingevoegd worden.

Het SRDY signaal bestaat uit een logische combinatie van een aantal andere signalen. De processor dient alleen een waitstate in te voeren als het ram geselecteerd wordt. Andere componenten op het board werken immers zonder waitstates. De signalen Advanced Memory Write (AMW), Memory Read (MRD) en AB19 bepalen of ram geselecteerd word. De eerste twee signalen worden door de buscontroller gegenereerd. Het derde signaal wordt gebruikt als chipselect voor de ramcontroller. Het vierde signaal is het AACK signaal dat door de ramcontroller gegenereerd wordt. Dit signaal geeft aan dat de ramcontroller klaar is met bijvoorbeeld zijn refreshcyclus en de processor verder kan.

Problemen tijdens uitvoeren ram testprogramma

Met behulp van de logic analyser is aangetoond dat de processor op een willekeurig tijdstip een waitstate te weinig invoert, ondanks het feit dat het SRDY signaal laag is. Dit is te zien in afbeelding 7. Een waitstate te weinig, betekent dat de statuslijnen te vroeg overgaan naar de passieve toestand. Dit betekent dat de buscontroller de memory read lijn (mrd) een klokslag te vroeg inactief maakt. De processor klokt op de opgaande flank van de memory read lijn de verkeerde data in. De ramcontroller heeft dan immers nog niet zijn column address strobe (CAS) operatie voltooid, waardoor de data van de geheugenlocatie, die gelezen dient te worden, nog niet op databus staat.

Een mogelijke oorzaak van de fout kan de aansluiting van de processor, in dit geval de emulator, zijn. Met name moet dan gedacht worden aan de ontkoppeling van de voeding en de dikte van de voedingslijnen.

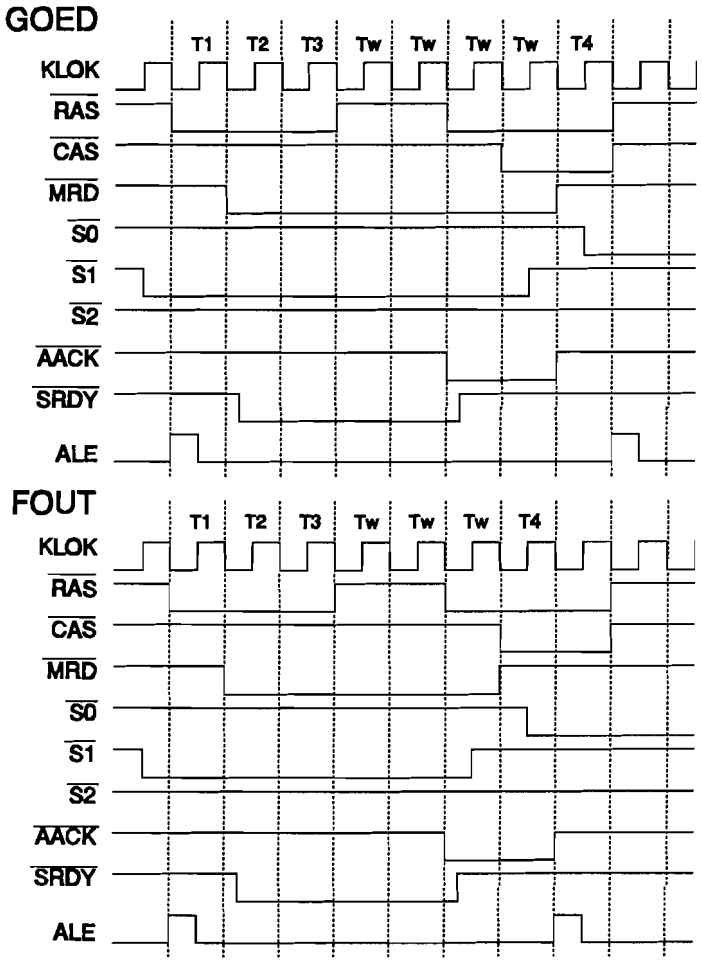

fig.7 Goede en verkeerde geheugen leesslag

Iedere chip heeft standaard op het ic-voetje een ontkoppelcondensator. Deze ontkoppelcondensatoren ontbreken echter op de ic-voetjes van de processor, de IDPC's, DMA controller en de ramcontroller omdat ze een afwijkend formaat hebben. Het toevoegen van deze ontkoppelcondensatoren gaf helaas geen verbetering.

De dikte van de voedingslijnen op het board is van belang, daar het terminal board met al zijn componenten ongeveer drie ampère aan stroom nodig heeft. De voeding die gebruikt wordt, moet zonder problemen vijf ampère kunnen leveren. Aangezien in het ram testprogramma niet alle componenten gebruikt worden, zijn de twee IDPC's, de DSC, de interruptcontroller, de DMA controller en een aantal chips, die met deze componenten te maken hebben, van het board gehaald. Op het terminal board zijn twee soorten voedingslijnen, te weten de hoofdvoedingslijnen en de component voedingslijnen. De hoofdvoedingslijnen liggen aan de rand rondom het board. De aarde aan de bovenkant en

de vijf volt voedingslijn aan de onderkant. Vanuit deze hoofdvoedingslijnen worden de afzonderlijke chips middels component voedingslijnen van stroom voorzien. Om beide voedingslijnen te omzeilen, zijn een aantal voor het ram testprogramma belangrijke componenten voorzien van directe voedingslijnen naar de plek, waar het terminal board gevoed wordt. Dit had een aanmerkelijke verbetering op het verloop van het ram testprogramma tot gevolg.

Het ram testprogramma verliep nu zonder problemen. Er deed zich echter een ander probleem voor. Op onverklaarbare wijze werd soms de Programmable Peripheral Interface (PPI), de parallelle poort waarop de ledbalken zijn aangesloten, gereset. Na de reset konden de ledbalken niet aangestuurd worden en bleven de acht leds van poort B en C branden. Pulsjes verschenen op de resetlijn die voor het resetten van de PPI zorgden. Deze pulsjes verschenen ook op de aarde lijnen van sommige chips. Het kon niet meer te wijten zijn aan de ontkoppeling van de voedingslijnen, dit was reeds bij iedere chip gebeurd. Naarmate er meer componenten van directe voedingslijnen voorzien werden, kwamen deze problemen steeds minder vaak voor. In een latere fase zijn de hoofdvoedingslijnen, die aan de rand rondom het board liggen aan de boven- en onderkant, overbrugd met extra draden. Hierdoor zijn deze problemen volledig verdwenen.

Uitvoering interrupt controller testprogramma

De uitvoering van het interrupt controller testprogramma heeft geen problemen gekend. Handmatig zijn alle interrupt lijnen met de positieve voedingsspanning verbonden. Hierna is het interrupt nummer op de ledbalk verschenen.

Bij aanvang van de test moet erop gelet worden dat alle interruptingangen van componenten, die niet aanwezig zijn op het board, met de aarde lijn verbonden worden. Indien dit niet gedaan wordt, kan de 'divide by zero' interrupt niet opgemerkt worden.

Uitvoering Advanced Direct Memory Controller (ADMAC) testprogramma

Het testprogramma dat de werking van de Advanced Direct Memory Controller (ADMAC) controleert, werkte niet. Met behulp van de logic analyser is vervolgens geconstateerd dat de adreslijnen verkeerd aangesloten zijn. Daar in de toekomst een nieuwe print vervaardigd wordt, is besloten deze ADMAC van het board te halen. De reeds geschreven software kan ook zonder ADMAC draaien.

Uitvoering DSC en IDPC testprogramma's

De DSC en IDPC testen zijn verder zonder problemen verlopen.

Voor een gedetailleerdere beschrijving van alle fouten op de terminal print verwijs ik graag naar bijlage 1 met als titel: Technische aanbevelingen t.b.v een nieuwe ISDN terminal board print.

3.5 Conclusie

Bij aanvang van mijn afstuderen was van de layout van het terminal board, gebouwd op het quick-connect board, een print gemaakt. De componenten op deze print zijn, met behulp van een aantal in assembler geschreven testprogramma's, grondig getest. Aangetoond is dat de processor (emulator) niet naar behoren functioneerde, waardoor tijdens het testen een foutmelding gegeven werd. De oorzaak van het niet naar behoren functioneren van de processor was het feit dat de voedingslijnen te dun waren. Door het dubbel uitvoeren van de voedingslijnen zijn deze problemen verdwenen. De processor emulator functioneert momenteel naar behoren.

Hoofdstuk 4 Software ontwikkeling t.b.v. de communicatie met een terminal

4.1 Inleiding

In de ontwikkelingsfase van het ISDN terminal board, zoals dat bij de vakgroep Digitale Informatie Systemen wordt ontwikkeld, is het wenselijk dat informatie over de toestand, waarin de software op het terminal board zich bevindt, voor de gebruiker toegankelijk is. Het protocol monitoring programma, dat in hoofdstuk zes beschreven wordt, zorgt voor deze interactieve informatie uitwisseling tussen gebruiker en software. De informatie wordt zichtbaar gemaakt op een terminal. De bouwsteen die de communicatie tussen het terminal board en de terminal verzorgt is de IDPC (Integrated Data Protocol Controller).

De Integrated Data Protocol Controller (IDPC) bestaat uit vier hoofdblokken. Deze blokken zijn de Data Link Controller (DLC), de Universal Synchronous/Asynchronous Receiver/Transmitter (USART), de Dual Port Memory Controller (DPMC) en de Microprocessor Interface (MPI). De DLC verzorgt de afhandeling van bit georiënteerde protocollen zoals HDLC, SDLC, LAPB en LAPD. De USART kan gebruikt worden als een terminal adaptor die bestaande terminals verbindt met netwerken die gebaseerd zijn op bit georiënteerde protocollen, zoals X.25, ISDN en SNA. De USART kan echter ook de communicatie met een normale terminal verzorgen. Het derde blok is de Dual Port Memory Controller (DPMC). Deze is nodig in een configuratie waarin twee processoren gebruik maken van de systeembus. De MPI verzorgt de communicatie tussen de IDPC en de processor.

Het terminal board is van het begin af aan ruim gedimensioneerd. Dit wil zeggen dat meer componenten op het board aanwezig zijn dan momenteel gebruikt worden. Het voordeel hiervan is dat later gemakkelijk meer functies aan het board toegewezen kunnen worden, door eenvoudig de software uit te breiden. De software die momenteel aanwezig is maakt eigenlijk alleen gebruik van de Digital Subscriber Controller (DSC) die zorgt voor de laag één en gedeeltelijk laag twee afhandeling van het D-kanaal LAPD protocol. De Advanced Direct Memory Access Controller (ADMAC) is niet functioneel en wordt ook niet gebruikt. Van de twee IDPC's die op het board aanwezig zijn, wordt er maar één

gebruikt. Deze verzorgt via de USART de communicatie met een ansi terminal, die als display voor het protocol monitoring programma werkt.

Door het uitbreiden van de software kan in een later stadium het board ook dienst doen als een 'R'-referentie punt. Hiervoor zijn de IDPC's nodig. Een 'R'-referentie punt maakt de aansluiting van een non ISDN terminal mogelijk op het 'S'-referentie punt. De DLC van de IDPC zorgt dan voor de omzetting van het gebruikte protocol op de non ISDN terminal naar LAPD voor het ISDN netwerk.

De USART bestaat uit een ontvanger, een zender en voor ieder een fifo buffer. De communicatie met de terminal geschiedt op interrupt basis. Door de heer Beijnsberger was reeds een functie geschreven die een string naar de terminal kon schrijven ('putmonitor'). Daarvoor was een interrupt routine geschreven die de interrupts van de zender van de USART afhandelde. Het was echter nog niet mogelijk om karakters in te lezen die van het keyboard van de terminal afkomstig waren.

In dit hoofdstuk worden de aanpassingen en de nieuwe routines beschreven. Tevens zijn er nog een paar functies geschreven die van pas zijn gekomen bij het schrijven van het protocol monitoring programma.

Eerst wordt een beschrijving gegeven van de IDPC.

4.2 Technische beschrijving van de IDPC

De IDPC bestaat uit vier hoofdblokken te weten de Data Link Controller (DLC), de Universal Synchronous/Asynchronous Receiver/Transmitter (USART), de Dual Port Memory Controller (DPMC) en de Microprocessor Interface (MPI), zie figuur 8. Deze onderdelen zullen achtereenvolgens in deze paragraaf toegelicht worden [2].

Data Link Controller (DLC)

Het hart van de IDPC is de DLC. Dit is een bit georiënteerde data link controller. Protocollen die de DSC aan kan, zijn SDLC, HDLC, LAPB (X.25) en LAPD. Deze protocollen gebruiken frames om data te versturen. De zojuist genoemde protocollen gebruiken allen hetzelfde frame formaat, zie figuur 11. Een frame begint met een begin vlag; deze is voor alle, zo net genoemde, protocollen hetzelfde. De begin vlag wordt gevolgd door het adres veld. Dit veld verschilt per protocol. De DLC kan zo geprogrammeerd worden dat het adres met vijf verschillende adressen vergeleken kan

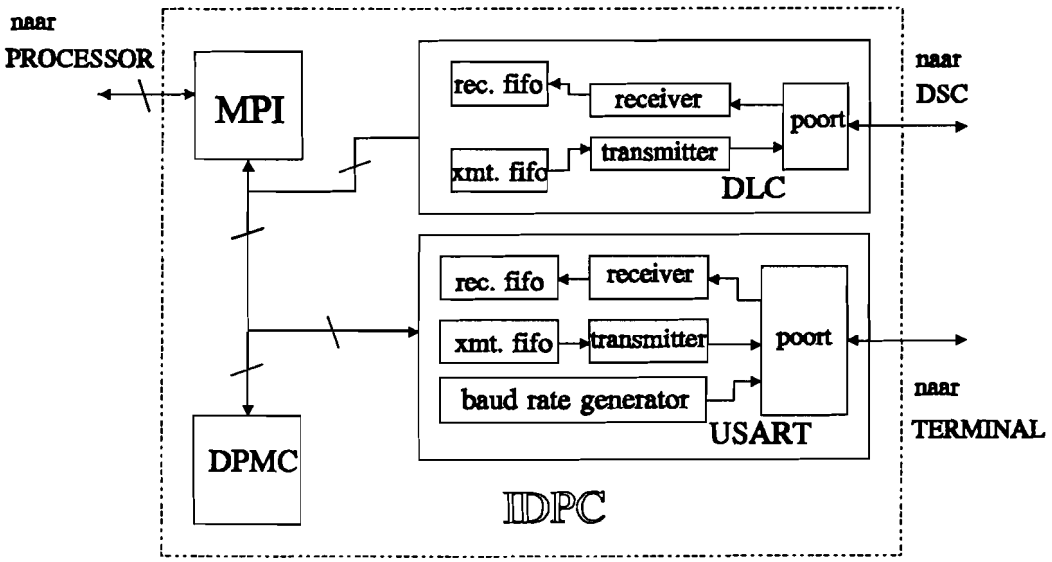

fig.8 Blokschema van de IDPC

worden. Indien het adres niet overeenkomt met één van de vijf, dan wordt het frame genegeerd. Bij uitgaande frames is de processor verantwoordelijk voor het toekennen van een adres. Het volgende veld is het controle veld. Dit veld wordt door de DLC als data behandeld en aan de processor doorgegeven. Het vierde veld is het informatie veld. Dit veld bevat de data die tussen twee gebruikers uitgewisseld wordt. De eind vlag wordt voorafgegaan door de controle som. Deze wordt berekend door de Cyclical Redundancy Check (CRC) generator.

De DLC verzorgt een full duplex verbinding tussen de Serial Bus Port (SBP). De SBP is verbonden met de multiplexer van de Digital Subscriber Controller (DSC), welke drie 64 kbits/s kanalen kan verbinden met de twee B-kanalen op het 'S'-referentie punt. De SBP kan in de non-multiplexed of in de multiplexed mode werken, waarin één of meerdere van de 31 time slots geselecteerd kunnen worden.

De zender van de DLC ontvangt van de processor het adres-, het controle- en het informatie veld. Deze drie datablokken kunnen oftewel onder DMA of onder geheugen of I/O controle in het 16-byte fifo buffer geladen worden. De zender voegt er dan het CRC, het begin veld en het eind veld aan toe en stuurt het naar de SBP. De SBP zorgt dan voor mark idle detectie, programmeerbare data inversie en eventuele multiplexing. De zender van de DLC verzorgt tevens vlag en abort detectie en zero bit insertion. Dezelfde handelingen worden in omgekeerde volgorde door de ontvanger van de DLC en de SBP verricht. De fifo buffer van de ontvanger echter heeft in vergelijking met die van de zender een capaciteit van 32 bytes.

De DLC heeft ook de mogelijkheid om de lengte van een pakket te controleren. Er kan een maximale en een minimale pakketgrootte opgegeven worden. Indien het pakket niet voldoet aan deze grenzen wordt een foutmelding gegenereerd.

Al deze mogelijkheden kunnen door het beschrijven van de DLC registers geprogrammeerd worden. Hieronder wordt een overzicht gegeven van deze registers.

Tabel 5 DLC registers

Offset (Hex)	Register Naam	Grootte (bytes)	Type
00	Command/Control register	1	Read/Write
01	Address Control register	1	Read/Write
02	Link Address Recognition register 0	2	Read/Write
04	Link Address Recognition register 1	2	Read/Write
06	Link Address Recognition register 2	2	Read/Write
08	Link Address Recognition register 3	2	Read/Write
0A	Serial Bus Port Control register	1	Read/Write
0B	Minimum Receive Packet Size register	1	Read/Write
0C	Maximum Receive Packet Size register	2	Read/Write
0E	Interrupt Source Interrupt Enable register	1	Read/Write
0F	Receive Frame Interrupt Enable register	1	Read/Write
10	Receive Link Interrupt Enable register	1	Read/Write
11	FIFO Status Interrupt Enable register	1	Read/Write
12	Transmit Byte Count register	2	Read/Write
14	FIFO Threshold register	1	Read/Write
15	Interrupt Source register	1	Read Only
16	Receive Byte Count register	2	Read Only
18	Receive Frame Status register	1	Read Only
19	Receive Link Status register	1	Read Only
1A	FIFO Status register	1	Read Only
1B	Receive FIFO Data register	1	Read Only
1C	Transmit FIFO Data register	1	Write Only
1D	Residual Bit Control Status register	1	Read/Write
1E-1F	Reserved	2	-

Universal Synchronous/Asynchronous Receiver/Transmitter (USART)

De USART is gelijk aan de industrie standaard 8250 UART. Deze verzorgt een asynchrone RS-232 seriële data communicatie met baud rate generatie. De USART voegt

daaraan nog drie mogelijkheden toe. Te weten:

- vier byte zend en ontvang fifo buffers
- karakter herkenning
- synchrone/transparante mode.

Tabel 6 USART register indeling

Offset (Hex)	Register naam	Grootte (Bytes)	Type
20	Receive FIFO Data Register (DLAB=0)*	1	Read Only
	Transmit FIFO Data Register (DLAB=0)*	1	Write Only
	Baud Rate Divisor LSB Register (DLAB=1)	1	Read/Write
21	Interrupt Enable Register (DLAB=0)	1	Read/Write
	Baud Rate Divisor MSB Register (DLAB=1)	1	Read/Write
22	Interrupt Identification Register	1	Read Only
23	Line Control Register	1	Read/Write
24	Modem Control Register	1	Read/Write
25	Line Status Register	1	Read Only
26	Modem Status Register	1	Read Only
27	Control Register	1	Read/Write
28	Status Register		
29	Special Character Bit-Map ADDR Pointer Register	1	Read/Write
2A	Special Character Bit	1	Read/Write
	Map Command Register	6	Read/Write
2B-3E	Reserved		-

De vier byte grote zend en ontvang fifo buffers verminderen het aantal interrupts van de USART.

In een seriële data stroom worden vaak besturings karakters ingevoegd. Deze

moeten normaal softwarematig uit de data stroom gedistilleerd worden. In de USART verzorgt de hardware deze dienst. Het is mogelijk om 128 karakters te definiëren. Op ontvangst van een dergelijk karakter wordt een maskeerbare interrupt gegenereerd.

In de transparante mode worden alle bits ontvangen en in porties van acht bits in de fifo buffers geladen. Er worden geen framing bits of idle bits afgehaald. Wanneer een aantal van deze acht bits porties zijn ontvangen, kunnen ze in een pakket bij elkaar gevoegd worden en over het netwerk via de DLC verzonden worden. De USART aan de ontvangst kant, kan dan de data blokken doorzenden zonder het toevoegen van framing of idle bits. Op deze manier kunnen alle mogelijke protocollen over het netwerk verstuurd worden.

De belangrijkste instellingen van de USART hebben te maken met de baudrate, pariteit, aantal data bits en stop bits. De klok van de USART wordt afgetapt van de DSC. De frequentie die in de DSC gegenereerd wordt is 12,288 MHz, maar wordt voordat deze de USART bereikt door twee gedeeld. De frequentie van het kloksignaal van de USART bedraagt dus 6,144 MHz.

In tabel 6 zijn de USART registers gegeven, waarin deze instellingen geprogrammeerd kunnen worden.

Dual-Port Memory Controller (DPMC)

In een configuratie waarin een ISDN kaart in een PC gestoken is, zijn twee processoren aanwezig. Eén op de kaart, welke meestal de onderste drie lagen van het ISO-OSI model afhandelt, en de ander in de PC. Het is wenselijk dat een speciale microprocessor de communicatie tussen deze twee microprocessoren regelt. De DPMC, aanwezig op de IDPC, kan hierin voorzien. Deze regelt de toegang van iedere processor tot het deel van het geheugen dat gezamenlijk is.

Aangezien het terminal board een stand-alone configuratie is waarop zich maar één hoofdprocessor bevindt, is deze DPMC niet van belang en wordt niet gebruikt.

Microprocessor Interface (MPI)

De MPI verzorgt de verbinding van de processor met de IDPC via de data- en adres bus, zodat registers gelezen en beschreven kunnen worden.

4.3 Adressering van interne registers

Zoals in figuur 4 te zien is, zijn de registers van de IDPC in de geheugen indeling opgenomen. Het segment waarin de registers zich bevinden zijn 8100H voor IDPC1 en 8200H voor IDPC2.

Voor het toegankelijk maken van registers zijn twee functies geschreven, die een byte kunnen lezen of schrijven uit/naar een geheugen locatie. Deze functies zijn:

```
void far wr_mem (byte value, word segment, word offset)
{
    point    mem;
    mem.pointer.offs = offset;
    mem.pointer.seg  = segment;
    *mem.p = value;
}

byte far rd_mem (word segment, word offset)
{
    point    mem;
    mem.pointer.offs = offset;
    mem.pointer.seg  = segment;
    return(*mem.p);
}
```

Het gebruik van deze functies vereist echter telkens het opzoeken van de offset en het segment van het betreffende register. Om de registers van de IDPC gemakkelijker toegankelijk te maken is daarom de volgende structuur opgezet:

```
struct idpc {   (idpcstruc.h file)
  word cmdcontrol;    /* Command/Control Register */
  word addrcontrol;   /* Address Control Register */
  word l2address[8];  /* Link Address Recognition Regs 0-3 */
  word sbpcontrol;    /* Serial Bus Port (SBP) Control Reg */
  word minrcvsize;    /* Minimum Receive Packet Size Reg */
  word maxrcvsize[2]; /* Maximum Receive Packet Size Reg */
  word intsrcIER;     /* Interrupt Source Interrupt Enable Reg */
  word rframeIER;     /* Receive Frame Interrupt Enable Reg */
  word rlinkIER;      /* Receive Link Interrupt Enable Reg */
  word fstatIER;      /* FIFO Status Interrupt Enable Reg */
  word xmitcntlo;     /* low-order Transmit Byte Count Reg */
  word xmitcnthi;     /* high-order Transmit Byte Count Reg */
  word fifothresh;    /* FIFO Threshold Register */
  word intsrc;        /* Interrupt Source Register */
  word rcvcntlo;      /* low-order Receive Byte Count Reg */
```

```
    word rcvcnthi;      /* high-order Receive Byte Count Reg */
    word rframestat;    /* Receive Frame Status Register */
    word rlinkstat;     /* Receive Link Status Register */
    word fifostat;      /* FIFO Status Register */
    word rcvfifo;       /* Receive FIFO Data Register */
    word xmitfifo;      /* Transmit FIFO Data Register */
    word rbconstat;     /* Residual Bit Control Status Reg */
    word reserved[2];
    word usartfifo;     /* fifo usart */
    word usartIER;      /* Interrupt enable reg */
    word usartIIR;      /* Interrupt Identificatiion Reg */
    word usartLCR;      /* Line control reg */
    word usartMCR;      /* Mode Control Reg */
    word usartLSR;      /* Line Status Reg */
    word usartMSR;      /* Modem Status Reg */
    word usartCR;       /* Control Reg */
    word usartSR;       /* Status Reg */

};
```

Deze structuur is opgebouwd volgens de in tabel 5 en 6 getoonde registerindeling van de IDPC. Het begin van deze structuur moet echter voor bijvoorbeeld IDPC1 vastgelegd worden op geheugenadres 81000H oftewel segment 8100H en offset 0H. Dit wordt gedaan middels de volgende programmaregels.

```
#define IDPC_BASE1 (struct idpc far *)0x80001000
extern struct idpc (far *IDPC);

IDPC=IDPC_BASE1;
```

De laatste regel zorgt ervoor dat de pointer IDPC naar de geheugenlocatie 81000H wijst. De registers zijn nu met hun naam toegankelijk. Een byte schrijven naar de usartfifo gaat als volgt:

```
IDPC->usartfifo=0x55
```

In de structuur is voor ieder register een word (16 bits) gereserveerd terwijl het register toch maar één byte breed is. Dit is zo gedaan omdat adreslijn 0 van de IDPC aangesloten is op adreslijn 1 van de adresbus. Adreslijn 1 van de IDPC is op zijn beurt aangesloten op adreslijn 2 van de adresbus enzovoort. Om het juiste register te adresseren moet de offset in tabel 5 en 6 vermenigvuldigd worden met twee. Dit is hetzelfde als voor ieder register een word reserveren.

4.4 Instellingen USART

De USART verzorgt de communicatie met een terminal. De terminal wordt op een PC geëmuleerd met het programma Telix. De seriële uitgang van de USART is via hardware buffers verbonden met de seriële COM poort van de PC. Voor het configureren van telix zijn de instellingen van de USART nodig. De software zorgt voor de initialisatie van de USART. De USART wordt op de volgende waarden ingesteld:

- 4800 baud
- no parity
- 8 databits
- 1 stopbit
- threshold interrupts voor ontvang en zend fifo
- zend fifo threshold: 2 bytes
- ontvang fifo threshold: 1 byte
- asynchronous mode
- interne ontvangst en zend klok generatie
- geen hardware of software flowcontrol

4.5 Terminal software

De terminal software maakt gebruik van twee roterende software buffers. De eerst volgende lege plek, waarin een byte in de buffer geschreven kan worden, wordt door een pointer aangegeven. De eerst volgende byte, die gelezen dient te worden, wordt ook aangegeven met een pointer. Voor iedere buffer wordt ook het aantal nog niet uitgelezen bytes bijgehouden. De parameter, waarin dit aantal wordt bijgehouden, wordt de lengte van de buffer genoemd. Aan de hand van de lengte van de buffer wordt bepaald of bytes uit de buffer gelezen dienen te worden.

De buffer voor de zendkant en de ontvangstkant heten respectievelijk 'Monitor' en 'Terminal'. De pointers, die het lezen uit en het schrijven naar de buffers coördineren, heten voor de zendkant 'MonitorOut' en 'MonitorIn' en voor de ontvangstkant 'Termout' en 'Termcount'. De lengte van de 'Monitor' buffer wordt bijgehouden in de parameter 'LengthMonitor' en de lengte van de 'Terminal' buffer in 'Lengthterm'.

In het vervolg van deze paragraaf wordt de software, die betrekking heeft op de communicatie met de terminal, opgedeeld in twee delen. Een gedeelte dat de zender van de

USART bestuurd en een gedeelte dat de ontvanger van de USART bestuurt.

Zender

Een drietal routines zijn geschreven die op de 'Monitor' buffer werken, te weten de cfprint, putmonitor en putvalonmonitor routine. Deze drie routines worden achtereenvolgens toegelicht.

cfprint:

De pointer van de cfprint routine, zie het onderstaande stukje programma tekst, wijst naar een variabele string die op de plaats van de parameterdeclaratie gezet wordt. Bij het aanroepen van deze routine wordt eerst de string, waarnaar de pointer wijst, in zijn geheel gecopieerd naar de 'Monitor' buffer. Vervolgens wordt de hardware buffer van de USART gevuld met bytes uit de software 'Monitor' buffer totdat deze hardware buffer vol is. Het usart status register geeft aan wanneer deze hardware fifo buffer vol is. Als de teller 'LengthMonitor' bij het aanroepen van de drie routines nul is, dan wordt de USART hardware buffer gevuld met de eerste bytes van de string. Is dit niet het geval, dan wordt de hardware buffer gevuld met bytes van strings, behorende bij vorige aanroepen van de drie routines. De twee pointers 'MonitorIn' en 'MonitorOut' en de teller 'LengthMonitor' coördineren deze overdracht van bytes van gedefinieerde string naar hardware buffer. De rest van de bytes in de string worden op interrupt basis van de software buffer 'Monitor' naar de hardware buffer van de USART gecopieerd. Als de bytes eenmaal in de hardware buffer staan, zorgt de USART automatisch voor het verzenden van deze bytes en het legen van de fifo hardware buffer telkens als er een byte verzonden is.

```
void far cfprint(char Msgstring[])
{
char *ptr;
    ptr=&Msgstring[0];
    mask(0x01);    /* entering critical region */

    while (*ptr !='\0') /* filling of Monitor buffer */
    {
        Monitor[(MonitorIn++) % MAXMONITOR]=*ptr;
        ptr++;
        LengthMonitor++;
    }
    if (LengthMonitor > MAXMONITOR)
```

```
    {
        outbyte(0x2002,0xFF); /* buffer overflow */
        halt();
    }
    while ((LengthMonitor != 0) && (IDPC->usartSR & BIT5))
    { /* filling of USART hardware buffer */
        IDPC->usartfifo = Monitor[(MonitorOut++) % MAXMONITOR];
        LengthMonitor--;
    }
    unmask(0x01); /* allow usart to interrupt */
}
```

putmonitor:

Als parameter wordt voor de cfprint en de putmonitor routines een pointer naar een string van ASCII karakters meegegeven. Het verschil tussen de eerste twee routines zit in deze pointer. De pointer van de putmonitor routine wijst naar een string die als constante is gedefinieerd in een tabel (file: tabl_msg.c). Verder werkt de putmonitor routine hetzelfde als de cfprint routine.

putvalonmonitor:

De cfprint, putmonitor en putvalonmonitor routines werken alle op dezelfde manier en schrijven tekst naar het terminal display. De putvalonmonitor routine zet een long integer om in ASCII tekst en roept vervolgens de putmonitor routine aan. Voor het omzetten van een long integer naar ASCII tekst wordt de C-funktie ltoa() gebruikt. Hiermee moet bij het linken van de object files rekening gehouden worden: de file clibl.lib moet meegelinkd worden.

De drie routines moeten als kritieke secties behandeld worden. De interrupt routines werken immers ook op de software buffers en de bijbehorende pointers en tellers. Om te voorkomen dat de USART interrupt routine midden in bijvoorbeeld een cfprint routine aangeroepen wordt, wordt de USART interrupt aan het begin van deze routine uitgezet met de functie mask(0x01) en aan het einde weer aangezet met de functie unmask(0x01).

Na het vullen van de Monitor buffer wordt in alle drie de routines gecontroleerd of de lengte van de Monitor buffer niet de grootte van de buffer overschrijdt. Is dit wel het geval dan wordt een foutmelding op de ledbalken zichtbaar gemaakt: alle leds op balk B gaan dan branden en de processor wordt gestopt.

Aan het einde van de cfprint routine is de hardware USART buffer gevuld. Dit wil

zeggen dat er vier bytes in de buffer staan. De USART start de transmissie. Wanneer er nog maar twee bytes in de buffer staan wordt een zend threshold interrupt gegenereerd. De aangeroepen USART interrupt routine vult de hardware buffer weer als de teller 'LengthMonitor' niet nul is.

Ontvanger

Als bron voor de ontvanger van de USART dient het keyboard van de PC waarop het terminal emulatie programma telix draait. Wanneer een toets wordt ingedrukt, wordt de ASCII code voor de betreffende toets naar de USART gezonden via de COM poort van de PC. Meestal is deze code een byte, maar ansi maakt ook gebruik van escape reeksen voor bijvoorbeeld de pijltjestoetsen. Dit zijn meerdere bytes. De USART plaatst deze byte automatisch in zijn hardware buffer. Daar de ontvanger threshold op één byte is geprogrammeerd tijdens de initialisatie, wordt een ontvanger threshold interrupt gegenereerd. Dit gebeurt hardwarematig als volgt: de USART genereert op de USART interrupt lijn een interrupt, de processor kan dan vervolgens in het usart interrupt identificatie register kijken wat voor soort interrupt het is. Aangezien er maar twee interrupts geprogrammeerd zijn kunnen ook alleen deze optreden. Dit zijn de ontvanger en zender threshold interrupts. Hieronder is de betreffende interrupt routine gegeven.

```
void far usart_int(void)
{
byte inter, term;

    inter = IDPC->usartIIR;
    if(inter & BIT2)      /* transmit threshold interrupt */
    {
       while((LengthMonitor != 0) && (IDPC->usartSR & BIT5))
           {
            /* fill usart transmit buffer */
              outbyte(0x2000,0x55);
              IDPC->usartfifo = Monitor[(MonitorOut++) % MAXMONITOR];
              LengthMonitor--;
           }
    }
    else if(inter & BIT3) /*#EB receive threshold interrupt */
        {
          while (IDPC->usartLSR & BIT1)
          {
            /* filling terminal receive buffer */
```

```
            outbyte(0x2000,0xff);
            Terminal[Termcount++ % MAXTERM] = IDPC->usartfifo;
            Lengthterm++;
        }
        if ((Lengthterm>MAXTERM) || (IDPC->usartLSR & (0x0e))
        {
            outbyte(0x2002,0xFF); /* buffer overflow  or error */
            halt();
        }
      }
}
```

De interrupt routine van de ontvanger copieert de bytes die in de hardware buffer staan naar de software buffer genaamd 'Terminal'. De pointer variabele 'Termcount' geeft daarbij aan op welke plaats deze bytes in de 'Terminal' buffer gezet moeten worden. Voor ieder byte die in de 'Terminal' buffer gezet wordt, wordt de teller 'Lengthterm' met één verhoogd. Aan het einde van de routine wordt gecontroleerd op fouten. Indien er iets mis is gegaan bij de overdracht van data gaan alle leds branden op poort B van de ledbalken. Tevens wordt in het begin van de routine de hexadecimale waarde 0xff naar poort A gestuurd. Wanneer dus een toets van het keyboard ingedrukt wordt, dan moet dit op de ledbalken zichtbaar worden. Als de hardware buffer van de zender gevuld wordt, wordt de hexadecimale waarde 0x55 naar poort A gestuurd, zodat het afwisselend ontvangen en zenden van bytes zichtbaar is op ledbalk A.

Aan de andere kant van de 'Terminal' buffer, de kant waar de bytes uit deze buffer gelezen worden, werkt de functie getch(). Dit is de enige functie die rechtstreeks op de 'Terminal' software buffer werkt, vanaf de processor kant. Deze functie maakt het voor de processor mogelijk om één byte uit de 'Terminal' buffer te lezen.

```
byte getch(void)
{
  while (Lengthterm==0) {};
  Lengthterm--;
  return Terminal[Termout++ % MAXTERM];
}
```

Voordat er een byte door de processor uit de 'Terminal' buffer gelezen kan worden, moet eerst een toets ingedrukt worden. Het indrukken van de toets activeert de ontvanger interrupt routine die de ASCII code voor de betreffende toets van de USART hardware buffer copieert naar de software 'Terminal' buffer. In deze interrupt routine wordt de teller 'Lengthterm' verhoogd. In de getch() functie wordt daarom eerst gewacht totdat de teller

'Lengthterm' groter is dan nul. Dit betekent dat er een byte uit de 'Terminal' buffer gelezen kan worden.

Ten behoeve van het terminal monitoring programma is nog een routine geschreven die een string van getallen in ASCII code omzet naar een byte. Deze routine staat bekent als scanf() in ANSI-C maar is echter niet aanwezig in de Intel-C bibliotheek. Op deze manier kan een decimaal getal, dat niet groter is dan 256, via de terminal ingelezen worden. Deze routine kan gemakkelijk omgezet worden naar een routine die een string van getallen in ASCII code omzet naar een word, zodat ook grotere getallen ingelezen kunnen worden. Het terminal monitoring programma neemt echter genoegen met de grootte van een byte. De routine copieert ieder ingelezen ASCII code, dat een getal voorstelt van '0' t/m '9', naar een string (intstring). Zodra er een ASCII code wordt ingelezen dat geen getal voorstelt, wordt de decimale waarde van deze getallen reeks berekent.

```
/* convert string of numbers to word (integer) */
byte readval(void)
{
  num;
  byte ch;

    num=0;
    ch=getch();
    while ((ch>='0')&&(ch<='9'))
    {
      num = num*10 + (ch-'0');
      ch = getch();
    }
    return (num);
}
```

4.6 Conclusie

Ten behoeve van protocol monitoring is een programma geschreven waarin de processen en messages binnen het ISDN protocol geobserveerd en gemanipuleerd kunnen worden. Dit protocol monitoring programma wordt in hoofdstuk zes beschreven. Het programma werkt via een terminal zodat de gebruiker, tijdens de testfase van het terminal board, interactief de protocollen kan besturen en bekijken.

De USART van één van de IDPC's zorgt voor de communicatie van het terminal board met de terminal. In dit hoofdstuk zijn de interrupthandlers en de routines, die nodig zijn voor het protocol monitoring programma, beschreven. Deze interrupthandlers en

Hoofdstuk 5 Software ontwikkeling rondom de Digital Subscriber Controller (DSC)

5.1 Inleiding

De laag twee en laag drie software die tot nu toe geschreven is heeft alleen betrekking op het D-kanaal. Laag twee heeft als taak het bieden van een foutvrije overdracht van data over het D-kanaal aan de netwerk laag, laag drie. Laag drie houdt zich bezig met de werking van het subnet. Een belangrijke taak van de netwerk laag is het bepalen langs welke route de pakketten van bron naar bestemming moeten gaan. Momenteel is Dhr. Claessens bezig met de implementatie van call control op laag vier op de ISDN Mitel kaarten. Deze call control zorgt voor het opzetten van een verbinding via het B-kanaal. Dit kan bijvoorbeeld een telefoonverbinding zijn. De berichten die nodig zijn om een verbinding te realiseren en af te breken worden ook over het D-kanaal verstuurd.

Het voorafgaande toont aan dat het D-kanaal een zeer belangrijke rol speelt binnen dit ISDN project. De Digital Subscriber Controller (DSC) verzorgt de verbinding van het terminal board met het ISDN 'S'-referentie punt, conform de internationale ISDN standaarden. De DSC scheidt de bitstroom in twee B-kanalen en een D-kanaal. Deze bouwsteen zorgt ervoor dat het D-kanaal toegankelijk wordt voor de processor. De B-kanalen kunnen via de multiplexer naar verschillende delen van de DSC gerouteerd worden. Eén van die delen waarnaar de B-kanalen gerouteerd kunnen worden is de Main Audio Processor (MAP). Dit deel van de DSC zorgt voor de aansluiting op een telefooncircuit. In de toekomst zal dit gedeelte van de DSC een grote rol spelen bij de implementatie van call control op het ISDN terminal board.

Eerst wordt in dit hoofdstuk een technische beschrijving van de DSC gegeven. Vervolgens wordt de software beschreven, die geschreven is ter aansturing van de DSC. Deze zogenaamde laag één software werkt interactief met laag twee. Het is dus van belang dat de nieuwste versie van laag twee en drie, zoals deze ontwikkeld is op de Mitel kaarten, ook draait op het terminal board. Aangezien de Mitel kaarten en het ISDN terminal board met verschillende systemen en compilers werken, moet deze software eerste aangepast worden.

5.2 Technische beschrijving van de DSC

De DSC zorgt voor de toegang van een Terminal Endpoint (TE), in dit geval het ISDN terminal board, tot het ISDN 'S'-referentie punt. De DSC verzorgt een full duplex verbinding tussen een TE en een Network Termination (NT) via een vieraderige lijn. De bitstroom van 192 kbps wordt door de DSC bij binnenkomst gesplitst in een D-kanaal en twee B-kanalen. De twee B-kanalen kunnen doorgegeven worden aan de verschillende onderdelen van de DSC. Het D-kanaal wordt voor een deel in de DSC bewerkt en is vervolgens voor de processor voor verdere verwerking toegankelijk.

Een 48 bits frame per 250 microseconde levert een transmissiesnelheid van 192 kbps. De framestructuur maakt frame synchronisatie en aansluitmogelijkheden voor meerdere terminals mogelijk. Op deze manier worden niet alleen point-to-point maar ook point-to-multipoint verbindingen ondersteund.

De gebruikte DSC, een AM79C30A, kan gebruikt worden als telefoon, digitale data terminal of beide. De DSC bestaat uit de volgende delen:

- LIU Line Interface Unit
- MUX Multiplexer
- MAP Main Audio Processor
- DLC Data Link Controller
- MPI Micro Processor Interface

Zie figuur 9 voor de grafische weergave van de samenhang van deze delen.

De Line Interface Unit (LIU)

De LIU verbindt de DSC via twee isolatie transformatoren met de vieraderige 'S'-interface. De LIU bestaat uit een ontvanger en een zender. De binnenkomende bitstroom wordt door de ontvanger gescheiden in een D en twee B-kanalen. De B-kanalen worden aan de MUX doorgegeven en het D-kanaal aan de DLC.

De ontvanger zorgt voor bit timing recovery, detectie van hoge en lage niveaus, frame recovery/synchronisatie, Alternate Mark Inversion (AMI) naar binair conversie en in geval van point-to-multipoint verbinding collision detectie.

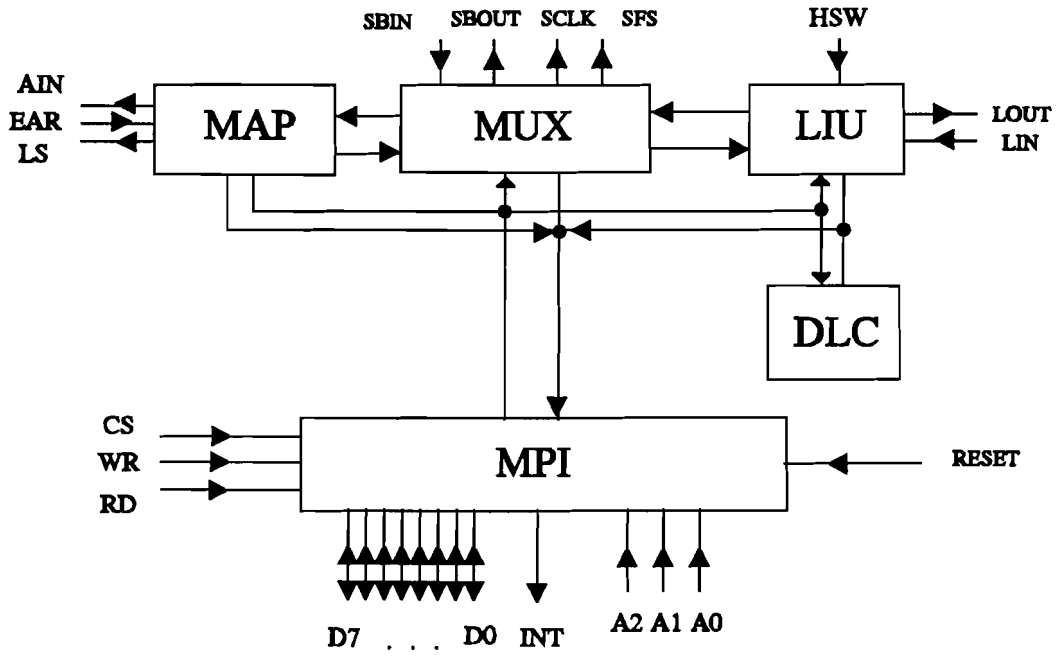

fig.9 Blokschema van de DSC

De zender multiplexed de B-kanalen en het D-kanaal in een bitstroom en verzorgt de synchronisatie bits. Voordat de bitstroom de isolatie transformatoren bereiken wordt deze omgezet naar AMI.

Middels het beschrijven van registers kan de LIU naar wens ingesteld worden. De belangrijkste registers zijn:

- Het LIU Status Register (LSR)
- De LIU Mode Registers (LMR1 en LMR2)

De eerste drie bits van het LIU Status Register zijn belangrijk daar deze de toestand van de LIU aangeven.

Het LIU Mode Register één (LMR1) geeft aan of de beide B-kanalen geactiveerd zijn en of de ontvanger en transmitter geactiveerd zijn. In het LIU Mode Register twee (LMR2) kunnen D-kanaal loopbacks geprogrammeerd worden. De tweede loopback, D-kanaal loopback bij de LIU, is gebruikt om de DLC interrupt routines te testen.

De Multiplexer (MUX)

De Multiplexer (MUX) is verbonden met de Main Audio Processor (MAP), de Microprocessor Interface (MPI) en de Serial Port (SP). De MUX routeert de 64 kbps B-kanalen van de LIU, MAP, SP of MPI naar de juiste bestemming. Deze bestemmingen kunnen in de MUX controle registers geprogrammeerd worden. Het is ook mogelijk om in de MUX controle registers een loopback te programmeren. Het D-kanaal wordt rechtstreeks van de LIU naar de DLC gerouteerd en passeert niet de MUX.

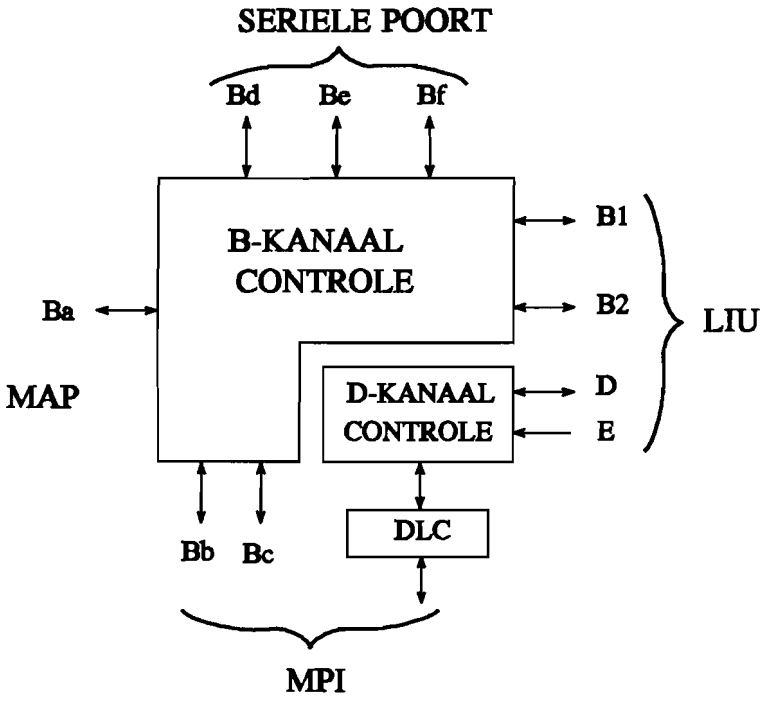

fig.10 MUX routerings schema

De SP heeft de mogelijkheid om drie full duplex 64 kbps B-kanalen te multiplexen in drie time slots.

De Main Audio Processor (MAP)

De MAP maakt het mogelijk om een telefoon, of een luidspreker met microfoon op de DSC aan te sluiten. Op het terminal board is daarvoor een telefooncircuit aangebracht zodat een telefoonhoorn aangesloten kan worden op de DSC. Een Digitaal Analoog

Converter (DAC) verzorgt de benodigde conversie in het pad van MUX naar telefoonhoorn. Een Analoog Digitaal Converter (ADC) verricht de conversie in het pad van telefoonhoorn naar MUX. Verder is het mogelijk om de signalen in beide paden digitaal te versterken of te filteren. Een Dual Tone Multi Frequency (DTMF) generator kan twee tonen, waarvan de frequentie en de amplitude programmeerbaar is, bij het signaal in het zend gedeelte optellen. In het ontvangstgedeelte zijn nog twee toon generatoren aanwezig die voor de bezettoon en de kiestoon gebruikt kunnen worden. In beide paden tussen de MUX en de MAP kan een A-law of een μ-law converter geprogrammeerd worden. Voor de programmering van de registers wordt verwezen naar de literatuurlijst [1].

De Data Link Controller (DLC)

Een 16 kbps D-kanaal wordt door de LIU met beide B-kanalen in de frame structuur van het 'S'-referentie punt gemultiplexed. De data die via dit kanaal verstuurd wordt, wordt volgens het Link Access Protocol D-kanaal (LAPD) gecodeerd. In figuur 11 is te zien uit welke velden een dergelijk LAPD pakket is opgebouwd. Het D-kanaal kan gebruikt worden voor signalering of een langzame pakketdienst.

VLAG	ADRES	CONTROL	INFORMATIE	FCS	VLAG
01111110	16-bits	8-bits	N-bits	16	01111110

fig.11 LAPD pakket

De DLC verzorgt laag één en gedeeltelijk laag twee protocol afhandeling. Tot de taken behoren:

- begin- en eindvlag detectie/generatie
- zero deletion en insertion
- Frame Check Sequence controle
- adres herkenning

De processor moet de overige laag twee taken en de taken van hogere lagen afhandelen. Hierbij moet gedacht worden aan:

- volgnummer controle
- verzenden van bevestigings pakketten
- hertransmissie van niet bevestigde pakketten

De status van de DLC wordt bijgehouden in het status register. Relevante interrupts kunnen gegenereerd worden indien de gebruiker deze programmeert. De DLC bevat ook een Random Nummer Generator (RNG) die gebruikt kan worden in de D-kanaal adres toekenningsprocedure. Verder heeft de zender en ontvanger van de DLC ieder een 8-byte data First In First Out buffer (FIFO).

Om ervoor te zorgen dat deze FIFO buffers ten alle tijde voldoende gevuld zijn of geleegd zijn, is het mogelijk om threshold interrupts te programmeren in het DLC Mode Register 1 (DMR1). Dit register regelt ook de adres herkenning en de 'end of receive packet' interrupt.

Voor een beschrijving van de overige registers wordt verwezen naar de literatuurlijst [1].

De Microprocessor Interface (MPI)

De MPI kan met iedere 8-bit microprocessor verbonden worden. De MPI is een op interrupt basis werkend interface, dat zorgt voor de toegang tot alle registers en buffers.

De registers zijn verdeeld in twee groepen. 12 Registers zijn direct adresseerbaar en staan vermeld in onderstaande tabel.

De meeste registers zijn write only of read only. Vandaar dat ieder adres twee keer voorkomt.

De tweede groep registers, de indirect adresseerbare registers, zijn alleen toegankelijk via het Command Register en het Data Register. Het Command Register is één byte groot en verdeeld in twee velden, het 'destination code field' (DCF) en het 'operation code field' (OCF). Het DCF veld geeft aan welk deel van de DSC geadresseerd dient te worden: het INIT register, de MUX, de MAP, de DLC of de LIU. Het OCF veld bepaalt het register in dat betreffende deel. Via het data register kan dan het register beschreven of gelezen worden met één byte.

Tabel 7 Direct adresseerbare DSC registers

ADRES (Hex)	REGISTER	AFKORTING
00H	Command Register	CR
00H	Interrupt Register	IR
01H	Data Register	DR
02H	D-channel Status Register 1	DSR1
03H	D-channel Error Register (2 byte FIFO	DER
04H	D-channel Transmit Buffer (8 byte FIFO)	DCTB
04H	D-channel Receive Buffer (8 byte FIFO)	DCRB
05H	Bb Transmit Buffer	BBTB
05H	Bb Receive Buffer	BBRB
06H	Bc Transmit Buffer	BCTB
06H	Bc Receive Buffer	BCRB
07H	D-channel Status Register 2	DSR2

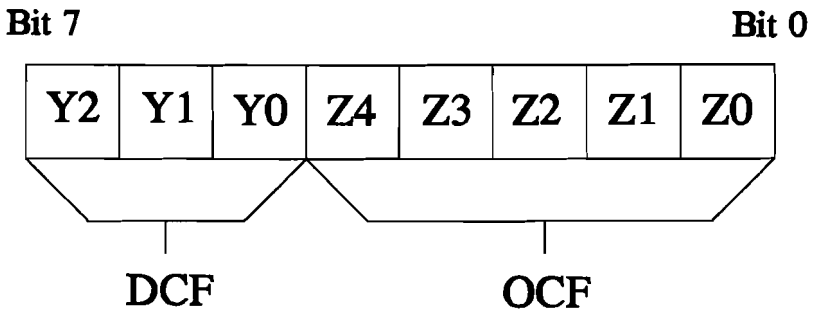

fig.12 Command register opdeling

5.3 Adressering van interne registers

De adressering van de direct adresseerbare registers binnen de DSC is op dezelfde manier opgelost als bij de IDPC, zie paragraaf 4.3. Wederom is een structuur voor de DSC gecreëerd. Rekening houdend met het feit dat adreslijn nul van de DSC aangesloten is op adreslijn één van de adresbus, en adreslijn één van de DSC op adreslijn twee van de adresbus enz., ziet deze structuur als volgt uit:

```
struct Dscreg
{
   byte     CR_IR;
```

```
    byte    INTER1;
    byte    DR;
    byte    INTER2;
    byte    DSR1;
    byte    INTER3;
    byte    DER;
    byte    INTER4;
    byte    DCTB_DCRB;
    byte    INTER5;
    byte    BBTB_BBRB;
    byte    INTER6;
    byte    BCTB_BCRB;
    byte    INTER7;
    byte    DSR2;
};
```

De velden met de naam INTER zorgen ervoor dat de adressen tussen de registers met een factor twee verspringen. De structuur is opgebouwd volgens de in tabel 7 getoonde registerindeling van de DSC.

Het begin van de structuur wordt vervolgens middels de volgende programmaregels vastgelegd op geheugenadres 83000H.

```
#define DSC_BASE (struct Dscreg far *) 0x80003000
extern struct Dscreg (far *DSC);

DSC = DSC_BASE;
```

De direct adresseerbare registers zijn nu met hun naam toegankelijk. Een byte schrijven naar bijvoorbeeld het D-kanaal Transmit Buffer (DCTB) gaat als volgt:

```
DSC -> DCTB_DCRB = 0x55
```

De indirect adresseerbare registers zijn ook met hun naam toegankelijk gemaakt. Met de 'C'-define operator is het DCF veld voor ieder deel van de DSC vastgelegd. Zie hieronder hoe dit gedaan is voor het DLC gedeelte van de DSC.

```
#define   DLC     (byte)0x80
```

De code die naar het Command Register (CR) geschreven moet worden, wordt bepaald door de DCF en de OCF. Deze code wordt met de define operator voor ieder register op de volgende manier vastgelegd:

```
#define   DMR1   (byte)(DLC / 6)
```

Een byte naar een indirect adresseerbaar register schrijven, in dit geval het DLC Mode Register 1, kan in twee programmaregels gedaan worden:

```
DSC->CR = DRM1
DSC->DR = 0x55
```

5.4 DSC interrupthandler

De DSC interrupthandlers moeten ervoor zorgen dat een LAPD pakket via de DLC en het D-kanaal verstuurd en ontvangen kan worden. In deze paragraaf worden de laag twee software en de DSC interrupt handlers, die de uitwisseling van LAPD pakketten besturen, besproken.

zender

De laag twee software kan middels een bericht, waarvan de nucleo PH_DATA_REQUEST heet, het TRANSMISSION proces verzoeken om een pakket te versturen. Dit pakket is opgeslagen in een software buffer. Het nummer van dit buffer wordt met het bericht in de vorm van een parameter meegegeven. Een dergelijk PH_DATA_REQUEST bericht kan afkomstig zijn van verscheidene processen. Het kan afkomstig zijn bijvoorbeeld van een laag twee signalingproces of van laag drie processen.

Op ontvangst van een PH_DATA_REQUEST voert het TRANSMISSION proces een functie uit. Welke functie uitgevoerd wordt hangt af van de toestand waarin het TRANSMISSION proces zich bevindt. Het TRANSMISSION proces kan zich in twee verschillende toestanden bevinden, te weten de toestand IDLE en BUSY. Is het proces in de BUSY toestand, wat betekent dat de zender bezig is met het versturen van een ander LAPD pakket, dan wordt de functie 'queue_packet' uitgevoerd. Deze functie zet het packet in een FIFO queue. Zodra de zender vrij is en het TRANSMISSION proces zich in de IDLE toestand bevindt, wordt het pakket uit de queue gehaald en verzonden.

Als het TRANSMISSION proces zich bij ontvangst van een PH_DATA_REQUEST bericht in de IDLE toestand bevindt, dan wordt de functie 'tx_packet' uitgevoerd. Deze functie gebruikt de 'xmtbuff' functie om een pakket te versturen. De functie 'xmtbuff' geeft de waarde XMT_INIT terug als de transmissie van het pakket geïnitieerd is en de waarde NO_ACTIVATION als de transmissie niet gestart kon worden. Indien de transmissie gestart

is, wordt in de 'tx_packet' functie de toestand van het TRANSMISSION proces middels de volgende programmaregel in de BUSY stand gezet. Is geen transmissie mogelijk dan wordt een bericht met een foutmelding naar het MANAGEMENT proces gestuurd.

De functie 'xmtbuff' vormt de scheiding tussen de software die niet hardware afhankelijk is en de software die wel hardware afhankelijk is. De software die tot nu toe beschreven is, is software onafhankelijk. Aangezien alle software van de ISDN Mitel kaarten overgezet is naar software voor het ISDN terminal board, moesten de routines die hardware afhankelijk waren opnieuw geschreven worden. Hieronder viel ook de functie 'xmtbuff'.

De toestand van de DLC zender wordt in een stukje geheugen, die de naam DscRam heeft gekregen, bijgehouden.

```
struct Dscstruc
{
   byte     emask[2];
   word     cmdstate;     /* Command Execution Status      */
   word     liustate;     /* Current LIU F-State           */
   word     hswstate;     /* Current HSW State             */
   address  xmtbuff;      /* Addr of Transmit Buffer       */
   address  xmtnext;      /* Addr of Next Transmit Char    */
   word     xmtrefno;     /* Transmit Buff Ref Num         */
   word     xmtcnt;       /* Transmit Buffer Byte Count    */
   word     xmtpktsz;     /* Transmit Packet Size          */
   word     xmtstat;      /* Transmit Status               */
   address  rcvbuff;      /* Addr of Receive Buffer        */
   address  rcvfree;      /* Addr of Next Free Location    */
   word     rcvrefno;     /* Receive Buffer Ref Num        */
   word     rcvstat;      /* Receive Status                */
   address  altbuff;      /* Addr of Alternate Rcv Buffer  */
   word     altrefno;     /* Alternate Rcv Buff Ref Num    */
   byte     frar[4];      /* Copy of FRAR Bytes            */
   byte     srar[4];      /* Copy of SRAR Bytes            */

};
```

De velden beginnend met de letters 'xmt-' hebben betrekking op de toestand van de zender of zijn pointers, die betrekking hebben op de software buffer of stellen het nummer van de buffer voor, dat momenteel verzonden wordt. Hetzelfde geldt voor de velden beginnend met de letters 'rcv-', maar dan voor de ontvanger.

De functie 'xmtbuff' zorgt voor het vernieuwen van het stukje DscRam, d.w.z. dat de pointers naar buffers, buffernummers en de toestand van de zender vervangen worden

door de huidige waarden, voordat het pakket verzonden wordt. Als de zender geactiveerd is, wordt de hardware buffer gevuld met de eerste bytes van de software buffer waarin het LAPD pakket staat. De transmissie wordt vervolgens geïnitieerd door de lengte van het pakket naar het D-kanaal Transmit Byte Count Register (DTCR) te schrijven. Dit is het einde van de functie 'xmtbuff'. De transmissie is dan gestart. De overige bytes in het pakket worden op interruptbasis verzonden. De interrupthandler is vanaf hier verantwoordelijk voor het afhandelen van de threshold interrupts.

In het DSC Interrupt Register kan de processor de identiteit van de interrupt achterhalen. In het geval dat deze een threshold interrupt van de zender is, wordt de hardware buffer bijgevuld met de volgende bytes die verstuurd moeten worden, totdat de hardware buffer vol is of de lengte van het pakket bereikt is. De DSC houdt zelf een teller bij die het aantal bytes bijhoudt die reeds verzonden zijn. Het LAPD pakket wordt automatisch met een eindvlag afgesloten als de teller de lengte van het pakket (DTCR) bereikt heeft. Het einde van een pakket wordt aan de processor bekend gemaakt met een 'End of valid transmit packet' interrupt (EOTP). Voordat de DSC deze EOTP interrupt genereert, wordt eerst de 'FCS sequence' berekend en vóór de eindvlag verstuurd. Het stukje programma, dat deze interrupt afhandelt, zet de toestand van de zender in de vrije stand. De software buffer, dat het momenteel verzonden LAPD pakket bevat, wordt vrijgegeven en de toestand van het TRANSMISSION proces wordt in de IDLE stand gezet. Deze handelingen worden ook verricht als een 'Underrun' fout optreedt. Deze fout treedt op als een leeg byte verzonden wordt, de DSC sluit dan onmiddellijk het pakket af en genereert een Underrun interrupt (UNRN_INT). Na de afhandeling van de EOTP interrupt zijn de DSC en het TRANSMISSION proces weer gereed om een ander LAPD pakket te versturen.

ontvanger

De processor wordt op ontvangst van het begin van een pakket geattendeerd, als een threshold interrupt van de ontvanger optreedt terwijl de toestand van de ontvanger niet 'busy' is. Voor het opslaan van het te ontvangen pakket wordt een buffer vrijgemaakt. De bij de ontvanger horende parameters in het DscRam worden ververst. Vervolgens wordt de toestand van de ontvanger in de busy stand gezet en worden de reeds ontvangen bytes van de hardware FIFO buffer naar de zojuist vrijgemaakte software buffer gecopieerd. Het onderstaande stukje programmatekst verzorgt deze handelingen.

```
if (DscRam->rcvstat != RCV_BUSY)
```

```
{
    newbuffer();
    (DscRam->rcvstat = RCV_BUSY);
}
while ((dsr2reg=DSC->DSR2) & RBA)
{
    *(DscRam->rcvfree++) = DSC->DCTB_DCRB;
}
```

De rest van de bytes in het pakket dat ontvangen wordt, wordt met behulp van threshold interrupts van hardware FIFO buffers naar de software buffer gecopieerd. De DSC houdt zelf een teller bij die het aantal ontvangen bytes telt. Dit hoeft niet softwarematig opgelost te worden en scheelt weer in snelheid.

Het einde van het pakket wordt aangegeven door de eindvlag. De DSC genereert een 'End of receive packet' (EORP) interrupt als deze eindvlag gedetecteerd wordt. Deze EORP interrupt wordt ook gegenereerd als een fout optreedt. Voordat het ontvangen pakket doorgegeven wordt aan de laag twee software moet eerst op fouten gecontroleerd worden. Dit wordt gedaan door het error register te lezen. De fouten, waarop gecontroleerd wordt bij een EORP interrupt, worden hieronder beschreven.

Frame Check Sequence error (FCS)

Deze error treedt op als het pakket niet goed ontvangen is. Dit wordt bij ontvangst van het laatste byte kenbaar gemaakt in het D-channel Error Register (DER).

Receive Packet LOST (RPLOST)

Deze error treedt op als twee pakketten ontvangen zijn en niet behandeld zijn door de processor. Een pakket is helemaal afgehandeld door de processor als de lengte van het ontvangen pakket uit het D-channel Receive byte Count Register (DCRB) is gelezen. Ieder pakket dat hierna ontvangen wordt, wordt afgebroken. Dit wordt in het DLC Status Register 2 (DSR2) kenbaar gemaakt.

Overflow (OVFL_INT)

Een overflow error treedt op als het aantal ontvangen bytes de limiet in het D-kanaal Receive Byte Limit Register (DRLR) overschrijdt. Deze limiet wordt tijdens de initialisatie van de DSC in het DRLR register geladen. Dit wordt voor de processor

eveneens zichtbaar gemaakt in het DER register.

Overrun (OVRN_INT)

Een overrun error treedt op als de hardware FIFO buffer vol is en nog een byte ontvangen wordt. Deze fout wordt ook zichtbaar gemaakt in het DER register.

Underflow (UNFL_INT)

Deze error treedt op als het ontvangen pakket met een twee bytes adres veld korter is dan vijf bytes. Eveneens zichtbaar in het DER register.

Non-Integral Number of Bytes (RESID_INT)

Deze error treedt op als het aantal bits dat tussen de beginvlag en de eindvlag ontvangen wordt geen veelvoud is van het getal acht. Ook deze error wordt kenbaar gemaakt in het DER register.

Voordat gecontroleerd wordt op fouten tijdens een EORP interrupt worden eerst de laatste bytes uit de hardware FIFO buffer gelezen en de lengte van het ontvangen pakket al uit het DRCR gelezen.

Indien geen fout tijdens de ontvangst is opgetreden, wordt het lengte veld van de software buffer, waarin het ontvangen pakket is opgeslagen, geladen met het aantal ontvangen bytes van het pakket. Vervolgens wordt de toestand van de ontvanger in de vrije stand gezet en een bericht naar laag twee gezonden. De nucleo van dit bericht heeft de naam PH_DATA_INDICATION en heeft als bestemmingsproces het TRANSMISSION proces. Het buffernummer wordt in parameter vijf van het bericht meegegeven. Het TRANSMISSION proces voert bij ontvangst van dit bericht de functie 'rcv_packet' uit. In deze functie wordt aan de hand van het adres het pakket naar het betreffende proces gestuurd.

De laag twee software draagt zorg voor de adresherkenning. De mogelijkheid om dit door de hardware van de DSC uit te laten voeren wordt momenteel niet gebruikt.

Deze interrupthandlers en bijbehorende routines zijn middels een D-kanaal loopback in de LIU, die in het LIU Mode Register 1 (LMR1) geprogrammeerd kan worden, getest. Deze loopback copieert de bytes in de hardware buffer van de zender zonder tussenkomst

van de LIU naar de hardware buffer van de ontvanger, met behoud van interrupt generatie. Het zenden van een pakket over het D-kanaal vormt een goede test van de software die de uitwisseling van LAPD pakketten bestuurt. Een software buffer, waarin het LAPD pakket tijdelijk wordt opgeslagen, kan vrijgemaakt worden met behulp van het protocol monitoring programma. Dit programma geeft het nummer van de buffer die vrijgegeven is. Deze buffer kan vervolgens, met behulp van het emulator programma, gevuld worden met waarden en met de lengte van de buffer. Het verzenden van dit pakket kan weer met het protocol monitoring programma gedaan worden. Het terminal display geeft vervolgens aan of het pakket in goede orde is ontvangen. Deze test verliep na een aantal aanpassingen zonder problemen.

Het was niet mogelijk om via de 'S'-bus een verbinding op te zetten met de ISDN Mitel kaarten, omdat er problemen waren met de 'S'-bus van deze kaarten. Indien dit in de toekomst wel gedaan kan worden moet de initialisatie van de LIU veranderd worden.

5.5 Initialisatie DLC

Tijdens de initialisatie procedure worden de volgende instellingen geprogrammeerd:

DLC

- De maximale pakket grootte wordt in het DLC Receive byte Limit Register (DRLR) geladen
- De toestand van de zender en de ontvanger in het DscRam wordt in de vrije stand gezet
- De beide threshold interrupts en de EORP interrupt worden in het DLC Mode Register 1 (DMR1) geactiveerd
- De EOTP interrupt dient geactiveerd te worden in het DMR3 register
- In het DMR4 register worden beide threshold interrupts op vier bytes vastgelegd
- Adresherkenning wordt uitgezet door in het DMR1 register het Transmit Address Register en in het DMR3 register de 'Valid Address/End of Address' interrupt uit te schakelen

5.6 Conclusie

De software die tot nu toe geschreven is t.b.v. het terminal board, heeft alleen betrekking op het D-kanaal. Deze software zorgt voor de signalering voor de beide B-kanalen door middel van het uitwisselen van pakketten tussen TE en NT.

De DSC bouwsteen op het terminal board verzorgt de toegang tot het D-kanaal op het 'S'-referentie punt. Deze bouwsteen moet het ontvangen en het verzenden van D-kanaal pakketten besturen. Dit gebeurt op interrupt basis. Tevens kan deze bouwsteen ervoor zorgen dat een telefoon op data terminal op het ISDN terminal board aangesloten kan worden.

In dit hoofdstuk zijn de interrupt handlers beschreven die het correct ontvangen en verzenden van D-kanaal pakketten verzorgen. Met behulp van een loopback in de DSC zijn de interrupt handlers getest. In de toekomst is het van belang dat deze software zo snel mogelijk getest wordt, door een verbinding te maken met één van de PC's met Mitel kaarten. Tot nog toe was dit, door problemen met de 'S'-bus van de Mitel kaarten, onmogelijk.

Hoofdstuk 6 Protocol monitoring programma

6.1 Inleiding

Het is van groot belang dat tijdens de ontwikkelfase van de ISDN software, bestemd voor het terminal board, informatie over de toestand van de protocollen en processen voor de gebruiker toegankelijk is. Alleen op deze manier is het mogelijk om nieuwe stukken software grondig te testen.

De software die draait op de Mitel express kaarten beschikt reeds over een protocol monitoring programma. Het is echter onmogelijk om dit programma over te nemen. De PC's, waarin de insteekkaarten zijn gestoken, gebruiken DOS als operating systeem. Het protocol monitoring programma van de Mitel express kaarten maakt veelvuldig gebruik van DOS routines om bijvoorbeeld het scherm aan te sturen.

Met behulp van de routines, die geschreven zijn ten behoeve van de communicatie met de terminal, zie paragraaf 4.5, is een protocol monitoring programma geschreven dat aangepast is aan het terminal board. De volgende paragraaf geeft een beschrijving van dit protocol monitoring programma.

6.2 Beschrijving protocol monitoring programma

Het protocol monitoring programma is ingebouwd in de dispatcher. Bij het opstarten van de terminal board software worden eerst de initialisatie routines uitgevoerd alvorens de dispatcher wordt bereikt. Om de activiteit van de dispatcher zichtbaar te maken worden oplopende getallen naar de ledbalk B geschreven.

Als de dispatcher actief is, worden de messages samen met de inkomende en uitgaande pakketten getoond op de terminal. De dispatcher kan gestopt worden door een willekeurige toets in te drukken. Het protocol monitoring programma wordt vervolgens actief. Het feit dat de dispatcher gestopt is, is te zien aan ledbalk B die geen activiteit vertoont.

In het hoofdmenu van het protocol monitoring programma kunnen drie mogelijkheden gekozen worden.

De eerste mogelijkheid is om informatie te verkrijgen over de toestand van bepaalde processen en/of de toestand van de DSC. Een willekeurig proces kan geselecteerd

worden, waarna informatie van dit proces, betreffende de toestand en de procesparameters, op de terminal zichtbaar wordt. Tevens kan met deze optie de inhoud van het Dscram en enkele LIU registers zichtbaar gemaakt worden. Het Dscram bevat alle parameters die betrekking hebben op de toestand van de zender en ontvanger van het D-kanaal, en op de pointers naar buffers van de zender en ontvanger. De inhoud van de LIU registers toont aan of een loopback geprogrammeerd is. Indien geen loopback geprogrammeerd is, dan kan door middel van deze optie de toestand van de LIU zichtbaar gemaakt worden, namelijk of deze wel of niet gesynchroniseerd is.

De tweede mogelijkheid is het manipuleren van de protocollen. Dit is een zeer krachtig medium waarmee de protocollen op hun correctheid getest kunnen worden. Het is mogelijk om met deze optie een willekeurig message samen te stellen en de toestand van een willekeurig proces te veranderen. Als de dispatcher dan weer actief is kan de uitwerking van de message op de protocollen op de terminal bestudeerd worden. Deze optie maakt het ook mogelijk om het terminal board te resetten.

De derde mogelijkheid biedt de gebruiker de mogelijkheid om operaties uit te laten voeren op buffers. Het is mogelijk om een vrije buffer te zoeken. Het buffernummer wordt dan zichtbaar gemaakt op de terminal. Verder kan een buffer vrij gegeven en verzonden worden.

Het vullen van een buffer met waarden of het wijzigen van de inhoud van een buffer, wordt op een andere manier bewerkstelligd. Dit kan gedaan worden met behulp van de emulator software. De symbolische namen van de buffers geven toegang tot de inhoud.

Het protocol monitoring programma biedt samen met emulator software de mogelijkheid om buffers te manipuleren. Na het vrijmaken van een buffer door het monitoring programma kan deze met behulp van de emulator software gevuld worden. Vervolgens kan deze buffer door het monitoring programma verzonden worden.

Aan het einde van het protocol monitoring programma kan gekozen worden om terug te gaan naar de dispatcher of om in het monitoring programma te blijven.

6.3 Conclusie

Het protocol monitoring programma is geschreven om op een gebruiksvriendelijke manier de processen en messages te kunnen observeren en manipuleren. De gebruiker kan

via een terminal interactief de protocollen observeren.

Het is tevens mogelijk om via de emulator software, zij het op een lager niveau, de terminal board software te observeren en te manipuleren. Via deze weg is het mogelijk om met behulp van symbolische namen de public gedefinieerde parameters toegankelijk te maken.

Het protocol monitoring programma tesamen met de emulator software vormen een zeer krachtig medium om de reeds geschreven protocollen op het terminal board te testen.

Hoofdstuk 7 Conclusie

In dit verslag wordt de hardware debugging van de terminal board print besproken. Tevens wordt de software toegelicht, die de communicatie van het terminal board met een terminal verzorgt. Ten behoeve van het observeren en het manipuleren van de laag twee en laag drie processen, zoals die geïmplementeerd zijn op het terminal board, is een protocol monitoring programma geschreven. Eveneens wordt in dit verslag de software besproken die het verzenden en ontvangen van ISDN D-kanaal pakketten verzorgt.

Met behulp van een aantal testprogramma's zijn de componenten op het terminal board getest. De eerste test, de ram test, gaf een foutmelding. Vervolgens kon met de Logic Analyser aangetoond worden dat de processor emulator niet naar behoren functioneerde. Het feit dat de voedingslijnen niet dik genoeg uitgevoerd waren, veroorzaakte het slecht functioneren van de processor emulator. Na het dubbel uitvoeren van de voedingslijnen en het aanbrengen van enkele ontkoppelcondensatoren zijn deze problemen verdwenen.

De terminal, die met het ISDN terminal board in verbinding staat, maakt protocol monitoring mogelijk. Eén van de twee Integrated Data Protocol Controllers (IDPC) op het terminal board zorgt voor de verbinding met de terminal via de geïntegreerde Universal Synchronous/Asynchronous Receiver Transmitter (USART). Op interruptbasis kunnen bytes van het terminal board naar het display van de terminal gezonden worden of bytes afkomstig van het keyboard van de terminal door het terminal board ontvangen worden. Het besturen van deze communicatie wordt door de interrupthandlers gedaan. Een drietal routines zijn geschreven die tekst naar het display van de terminal kunnen schrijven. Tevens is een routine geschreven die een byte van het keyboard van de terminal kan inlezen. Deze routines dienen als basis voor het protocol monitoring programma.

Het terminal board kan aangesloten worden op het ISDN. Het ISDN voorziet in twee B-kanalen en een D-kanaal. De B-kanalen zorgen voor de overdracht van spraak en data. Het D-kanaal zorgt voor de overdracht van signaleringsdata. Deze signaleringsdata worden in pakketten verzonden. De Digital Subscriber Controller (DSC) op het terminal board zorgt voor het verzenden en ontvangen van deze pakketten. Ten behoeve van het verzenden en ontvangen van pakketten zijn interrupthandlers geschreven. Deze interrupthandlers verzamelen de bytes behorende bij een pakket in roterende software buffers, zodat de pakketten toegankelijk zijn voor de processor. De werking van de

interrupthandlers is, met behulp van een loopback in de DSC, getest. Door problemen met de Mitel kaarten kon nog geen verbinding tussen het terminal board en één van de PC's met Mitel kaart opgezet worden.

Het protocol monitoring programma maakt het mogelijk om interactief de software, die draait op het terminal board, te observeren en te manipuleren. Samen met de emulator software, waarin de globale variabelen van de software bekeken en veranderd kunnen worden, is het protocol monitoring programma een krachtig programma om de software te testen. Het is mogelijk om de laag twee en drie processen te selecteren en de betreffende parameters te bekijken of te manipuleren. Het is ook mogelijk om een message te genereren. Verder zijn nog een aantal buffer operaties mogelijk, zoals het vrijgeven, selecteren en verzenden van een buffer over het D-kanaal.

Aan het begin van mijn afstudeerperiode lag het terminal board wat betreft ontwikkeling nog ver achter op de Mitel kaarten. Het was toen absoluut onmogelijk om een verbinding op te zetten tussen een PC met Mitel kaart en het terminal board. Het niet functioneren van de terminal board print en de achterstand op software gebied maakten dit onmogelijk. Momenteel is het zover dat deze verbinding wel gemaakt kan worden, wanneer de problemen met de Mitel kaarten zijn opgelost.

In de toekomst is het van belang dat, naast het opzetten van deze verbinding, ook de 'call control', ontwikkeld door R. Claessens, overgezet wordt naar het terminal board.

Literatuurlijst

[1] Advanced Micro Devices
Am79C30A/32A Digital Subscriber Controller (DSC), ISDN Data Controller (IDC)
Sunnyvale USA, juni 1989.

[2] Advanced Micro Devices
Am79C401 Integrated Data Protocol Controller, Technical Manual.
Sunnyvale USA, juni 1989.

[3] Beynsberger, J.,
Hardware en software debugging van het ISDN terminal board.
Eindhoven: Technische Universiteit, Elektrotechniek, vakgroep digitale informatie systemen. 1990.

[4] Egmond, D. van,
Implementation of the link access procedure on the ISDN D-channel.
Eindhoven: Technische Universiteit, Elektrotechniek, vakgroep digitale informatie systemen. 1991.

[5] Intel Corporation,
ASM86 macro assembler operating instructions for iRMX™ 286 systems
California: Santa Clara, 1985.

[6] Intel Corporation,
iAPX 86, 88 Family Utilities User's Guide for iRMX 286 Systems
California: Santa Clara, 1985.

[7] Intel Corporation,
ICE™-186/188 In-Circuit Emulator User's Guide
California: Santa Clara, 1989.

[8] Intel Corporation,
iC-86/286 Compiler user's guide for DOS systems
California: Santa Clara, 1989.

[9] Intel Corporation,
iAPX 86/88, 186/188 User's Manual, Hardware Reference.
California: Santa Clara, 1985.

[10] Jonkers, H.G.,
Realisatie van een S-interface en telefoon-interface voor het ISDN terminal board.
Eindhoven: Technische Universiteit, Elektrotechniek, vakgroep digitale informatie systemen. Oktober 1991.

[11] Koops, E.,
ISDN terminal board het debuggen van de hardware.
Eindhoven: Technische Universiteit, Elektrotechniek, vakgroep digitale informatie systemen. Januari 1990.

[12] Kuilen, S. van de,
Software ontwikkeling rond de Mitel ISDN express card.
Eindhoven: Technische Universiteit, Elektrotechniek, vakgroep digitale informatie systemen. Augustus 1990.

[13] Lemmens, R.J.,
Implementatie van het ISDN laag 3 signaleringsprotocol voor het D-kanaal.
Eindhoven: Technische Universiteit, Elektrotechniek, vakgroep digitale informatie systemen. 1991.

[14] Maastricht, C.,
Initialisatie en testen van het ISDN terminal board.
Eindhoven: Technische Universiteit, Elektrotechniek, vakgroep digitale informatie systemen. Februari 1990.

[15] Oorschot, L.,

Implementatie van protocollen voor het ISDN D-kanaal.
Eindhoven: Technische Universiteit, Elektrotechniek, vakgroep digitale informatie systemen. 1992.

[16] Oudelaar, H.,
Implementatie of a CCITT protocol on an ISDN terminal board.
Eindhoven: Technische Universiteit, Elektrotechniek, vakgroep digitale informatie systemen. 1989.

[17] PTT dr neher laboratorium,
Symposium: 'Geïntegreerde netten', lezingen.
Leidschendam, 12 december 1986.

[18] Stallings, W.,
ISDN: an introduction.
New York: MacMillan, 1989.

[19] Tanenbaum, A.S.,
Computernetwerken.
Englewood Cliffs USA, 1988.

[20] Weterman, P.J.G.,
Designing and building of an ISDN terminal board.
Eindhoven: Technische Universiteit, Elektrotechniek, vakgroep digitale informatie systemen. 1989.

Bijlage 1 Technische aanbevelingen t.b.v. een nieuw ISDN-print.

Inleiding

In navolging van het quick-connect ISDN board is een print ontworpen en gefabriceerd. Ondanks het feit dat het quick-connect board grondig getest is, zijn er toch fouten in deze print gevonden. Op de print zijn een aantal signalen ten onrechte doorverbonden met Gnd of Vcc. Deze fout kan toegerekend worden aan de fabrikant van deze print. Op zijn kosten kan een nieuwe print gemaakt worden. Buiten deze fout zijn er nog een aantal andere fouten gevonden die te wijten zijn aan een verkeerde layout.

Fouten te wijten aan de fabricage

Bij de fabricage van de print zijn een aantal signalen ten onrechte doorverbonden met Gnd of Vcc. Hieronder staat in een tabel aangegeven over welke signalen het hier gaat.

Tabel 1 Signalen die verkeerd doorverbonden zijn.

Verkeerd doorverbonden met Gnd of Vcc				Goede doorverbinding	
Chip nr.	Pin nr.	Signaal	Gnd/Vcc	Chip nr.	Pin nr.
U34	18	DRQ0	Gnd	U63	17
U34	53	S1	Vcc	U35	11
U34	52	S0	Vcc	U37	66
U35	14	S2	Vcc	U34	54
U37	14	PSEN	Vcc	U48	3 en 11
U37	35	AO0	Gnd	RPACK3	1

Deze signalen zijn in netlist aangegeven met NODE. De met NODENAME aangegeven signalen overschrijven de impliciet opgegeven signalen van de componenten.

Fouten te wijten aan de layout

Om de ISDN print te testen zijn een aantal programma's in assembler taal geschreven, die de verschillende componenten tot op een bepaald niveau testen. Deze programma's zijn:

test1.asm	ram test
test2_2.asm	PIC test
test4_2.asm	DSC test met ram test
test4_3.asm	DSC test zonder ram test
test5_4.asm	IDPC test U59
test5_5.asm	IDPC test U58

In dit overzicht zijn geen programma's opgenomen die de DMA controller testen. De DMA controller kon niet getest worden omdat de adreslijnen verkeerd doorverbonden waren. De testprogramma's die wel in dit overzicht zijn opgenomen zijn gecorrigeerd.

De eerste test is een ram test. Deze gaf meteen al problemen. Het bleek dat er op onverklaarbare wijze pulsjes op Gnd en Reset lijnen van sommige chips verschenen. De pulsjes op de Reset lijn zorgden voor het resetten van de PPI waarmee de ledbalken zijn verbonden. Verder voegde de Intel 80186 emulator op willekeurige wijze bij het lezen van het geheugen een waitstate te weinig in. Het onbehoorlijk functioneren van deze processor kan te wijten zijn aan de ontkoppeling van de voedingslijnen en/of een te lage stroomvoerende capaciteit van de voedingslijnen. Van beide was hiervan sprake. De volgende chips waren niet ontkoppeld:

U34	Intel 80186 emulator
U35	DMA controller
U37	RAM controller
U58	IDPC
U59	IDPC

Het zijn vooral de voedingsbanen Gnd en Vcc, die rondom het board aan de boven- en onderzijde lopen, die te dun zijn. Het met draden dubbel uitvoeren van deze banen en het ontkoppelen van de zojuist genoemde chips verhielpen de problemen.

De tweede test is de PIC (Programmable Interrupt Controller) test. Deze gaf geen problemen. De niet gebruikte interrupt inputs dienen wel gemaskeerd te worden of met Gnd

verbonden te worden.

De derde test gaf wederom problemen. Dit is de DMA test. Zoals in de inleiding vermeld staat kon deze test niet uitgevoerd worden daar de adreslijnen niet goed waren doorverbonden. De volgende pinnen van de DMA controller (U35) zijn ten onrechte doorverbonden met de gegeven adreslijnen:

pinnen 18 t/m 25 met respectievelijk ad0 t/m ad7

pinnen 27 t/m 34 met respectievelijk ad8 t/m ad15

De goede doorverbinding dient te zijn:

AD0	ad0	pin 34
AD1	ad1	pin 32
AD2	ad2	pin 30
AD3	ad3	pin 28
AD4	ad4	pin 25
AD5	ad5	pin 23
AD6	ad6	pin 21
AD7	ad7	pin 19
AD8	ad8	pin 33
AD9	ad9	pin 31
AD10	ad10	pin 29
AD11	ad11	pin 27
AD12	ad12	pin 24
AD13	ad13	pin 22
AD14	ad14	pin 20
AD15	ad15	pin 18

Toevallig zijn de pinnen 23 en 29 met de goede adreslijnen doorverbonden.
De DMA controller wordt dus niet gebruikt. Het signaal Hold is met Gnd verbonden zodat de processor niet in een oneindige waittoestand komt.

De volgende test is de DSC test. Deze test gaf geen problemen. Er zijn wel nog een aantal onvolkomendheden aangetroffen rondom de DSC.
DSC pin 13 is niet aangesloten aan de Vss (Gnd).
Met behulp van de DSC kan een digitale telefoon verbinding opgezet worden met bijvoorbeeld de Mitel Express kaarten via de S-bus. Ten behoeve van zo'n verbinding is er op het terminalboard een telefoon interface aangebracht.

De test voor een analoge loopback werkt. Je kunt jezelf in de telefoonhoorn horen.
De S-bus verbinding met de Mitel kaarten gaf een aantal problemen.
Ten eerste bleek dat de transmitter op de Mitel kaart verbonden was met de transmitter op het terminal board (receiver met receiver). Dit kan gemakkelijk opgelost worden door op de Mitel kaarten 2 jumpers te wisselen. Het kan nog eenvoudiger in de toekomst door op de volgende ISDN print ook jumpers te plaatsen.
Het was mogelijk om een verbinding op te zetten. Deze was helaas niet van hoge kwaliteit. De oorzaak hiervan is het feit dat het telefoon interface op het terminal board sterk afwijkt van die op de Mitel kaarten. Het telefoon interface op het terminal board maakt niet optimaal gebruik van het bereik van de AD converter in de DSC. Dit bereik is -2.0 V - 2.0 V top - top. Bij een normaal geluidsniveau is middels een simpele meting een gemiddelde top - top waarde gemeten van -0.06 V - 0.06 V. Dit signaal wordt in tegenstelling tot het telefoon interface op de Mitel kaarten niet meer versterkt. Op de Mitel kaarten wordt het signaal in geval van µ-law codering 7.3 db en in geval van A-law codering 16.7 dB versterkt. De vraag is nu welke codering wordt er op de Mitel kaarten toegepast µ-Law of A-Law? Om dit te weten te komen moet de Mitel kaart bekeken worden. In het geval dat een MT8994 geïntegreerd telefoon circuit aanwezig is wordt gebruik gemaakt van µ-Law codering. In het andere geval MT8995B A-Law codering. Bij µ-Law moeten R16 en R17 op de Mitel kaarten 100kΩ zijn. Bij A-Law zijn deze 18.7KΩ.
De designer's manual van de ISDN express card geeft echter aan dat er µ-Law codering gebruikt wordt, maar dit is natuurlijk geen bewijs ervan.
Aanbevolen wordt het telefoon interface op de ISDN print weg te laten en de signalen AREF, AINA, Gnd, Vcc, EAR1 en EAR2 middels een connector voor de buitenwereld toegankelijk te maken. Op deze manier kan een telefoon interface dat voldoet aan de eisen van de Mitel kaarten aangesloten worden.

De laatste testprogramma's voor de beide IDPC's verliepen zonder problemen.
De verbinding van de USART met een terminal gaf wel problemen.
Ten eerste was de verkeerde connector op het board geplaatst. Het transmitter data signaal Txd behoort verbonden te zijn met pin 3 van de connector. Dit was echter niet het geval. (null modem overbodig)
Ten tweede waren de drie receiver response control signalen op driver U57 met elkaar doorverbonden. Deze moeten echter ieder afzonderlijk via een condensator en weerstand met Gnd doorverbonden worden of niet verbonden worden. Indien ze met elkaar doorverbonden zijn kunnen de inputsignalen elkaar beïnvloeden.
Bij het ontkoppelen van deze drivers moeten de condensatoren aangepast worden aan de

spanning die gebruikt wordt. De transmitter driver gebruikt voedingsspanningen van -12V tot 12V.

Bij één van de IDPC's is een mankement geconstateerd. De datauitgang Txd is defect. De C uitgangen nummer 4 en 5 van een van de PPI's zijn defect.

Het is nu mogelijk om data naar de terminal toe te sturen en ook van de terminal te ontvangen.

Bijlage 2 Software beschrijving

Indeling files

De files van de laag twee en laag drie software, behorende bij het terminal board, zijn om overzicht te houden verdeeld in een aantal directories.

ASM:

In de directory ASM staat de assemblerfile start.asm. Deze file bevat de interrupttabel en de interruptroutines. Verder worden in deze file ook de processor en de timer van de processor ingesteld. Na het instellen van de processor en de timer van de processor wordt het programma 'main' aangeroepen. In dit programma wordt, na de initialisatie van het terminal board en de software, de dispatcher gestart.

NINCLUDE:

In deze directory staan alle files die declaraties bevatten welke betrekking hebben op een bepaald onderdeel van de software of component van het terminal board. Als bijvoorbeeld een programma geschreven dient te worden dat de IDPC aanstuurt, dan moet in deze directory gezocht worden naar de file waarin de declaraties staan die te maken hebben met de IDPC. Deze declaratie file moet vervolgens met de include operand in het nieuwe programma geopend worden.

NINIT:

De directory INIT bevat alle files die zorgen voor de initialisatie van de componenten op het terminal board en de processen.

NINTE:

Zoals de naam aangeeft staan in deze directory de interrupt routines. De DSC interrupt routine is de enige waarvan momenteel gebruik gemaakt wordt. De USART interrupt routine is samen met de USART initialisatie routine te vinden in de nog te bespreken directory NPROG. (filenaam: usart.c)

NPROG:

In deze directory staan alle programma's die de laag twee en laag drie protocollen en de hardware besturen.

NTABLE:

Deze directory bevat de statetabellen van de verschillen processen.

NTEST:

Alle object files worden in deze directory verzameld. Vanuit deze directory moeten ook de linker en de locater, die nog besproken worden, opgestart worden. Tevens staan in deze directory alle batch files die nodig zijn voor de compiler, de linker en locater.

In de root van de verzameling directories staan ook een aantal files. Deze files hebben betrekking op de laag twee en laag drie declaraties, externe declaraties en de definitie van globale variabelen. De namen van de laag drie files zijn als volgt bepaald '13*.*' en kunnen dus gemakkelijk van de laag twee files onderscheiden worden.

Het Intel pakket

ASM86:

De assembler files, waaronder de testfiles en de start.asm file vallen, worden geassembleerd met de Intel ASM86 assembler. Bij het aanroepen van deze assembler moeten een aantal parameters meegegeven worden. De batchfile 'make.bat' zorgt automatisch voor het meegeven van deze parameters bij het assembleren van de start.asm file. Voor een uitleg van de parameters verwijs ik naar de manual [5]. Het assembleren van een *.asm file resulteert in een object file en een list file.

IC86:

Het compileren van de C-files wordt gedaan met de Intel C compiler IC86. Bij het aanroepen van de IC86 compiler moeten eveneens een aantal parameters meegegeven worden. De batch file 'ic.bat' doet dit automatisch. Voor de uitleg van deze parameters wordt verwezen naar de manual [8]. Het aanroepen van bijvoorbeeld de file usart.c gaat nu als volgt:

```
ic usart
```

Deze aanroep moet gedaan worden in de directory waarin de usart file zich bevindt. (De directory waarin de batch files zich bevinden moet in het DOS PATH zijn opgenomen) Als

de compiler geen foutmelding geeft wordt automatisch de objectfile naar de directory NTEST gecopieerd. De IC86 compiler genereert ook een list file waarin de fouten bekeken kunnen worden.

LINK86:

De linker, LINK86, zorgt voor het aan elkaar koppelen van meerdere object files [6]. De object files worden opgesomd in de file link.con. In de laatste regel van deze link.con file wordt tevens een Intel bibliotheek meegelinkd die de funktie ltoa() bevat, waarvan de putvalonmonitor routine gebruik maakt. De file die na het linken gegenereerd wordt heet test.lnk. Eventuele fouten kunnen bekeken worden in de file test.mp1. Daar de interruptvectoren op adres 0H in het geheugen moeten staan, moet de file start.obj bovenaan in de file link.con staan. De batch file link.bat maakt het aanroepen van de linker een stuk gemakkelijker.

LOC86:

De locater kan ervoor zorgen dat de codesegmenten in ROM en de data- en stacksegmenten in ram terecht komen. Het aanroepen van dit programma gaat ook gepaard met het meegeven van enkele parameters. Wederom wordt voor een uitleg van deze parameters verwezen naar de manual [6]. De batch file 'loc.bat' zorgt automatisch voor het meegeven van deze parameters. De locater gaat uit van de file test.lnk en genereert de file test. In de file test.mp2 kunnen eventuele fouten bekeken worden.

De linker en locater genereren files in het Intel 8086 absoluut Object Module Formaat (OMF). Deze files kunnen ingelezen worden in het emulator programma [7]. Het is dus niet nodig om deze eerst te converteren naar een hexadecimale file. Het voordeel van het inladen van een OMF file in het emulator programma is het feit dat automatisch de symbolfile meegeladen wordt. Deze symbolfile maakt het mogelijk dat in het emulator programma gemakkelijk globale variabelen en registers met behulp van hun namen uitgelezen en beschreven kunnen worden.