Een I/O controller voor een IBM 2314 disk gerealiseerd met een microprocessor

Harbers, B.Th

Award date:
1978

Een I/O controller voor een IBM 2314 disk gerealiseerd met een microprocessor

Afstudeerverslag van
B.Th.Harbers

De opdracht is uitgevoerd bij de vakgroep EB, Digitale Systemen,
o.l.v. ir.M.P.J.Stevens,
ir.J.P.Kemper,
hoogleraar prof.ir.A.Heetman.

Begindatum: 9 september 1977
Einddatum :31 augustus 1978

Trefwoordenlijst:

I/O controller
Microprogrammeren
Microprocessor
IBM 2314 disk
Hardware
Software

INHOUD.

Samenvatting.

In dit verslag wordt het ontwerp en de realisatie van een I/O controller beschreven waarin een microprocessor toegepast wordt.
Deze I/O controller moet in staat zijn een IBM 2314 disk te kunnen besturen die een datarate van 312 kbyte/sec heeft.
Na de bestudering van de signalen, die van en naar de disk drive gaan, en de daarbij horende timing is een mogelijk ontwerp gegeven waarvan al een gedeelte gerealiseerd is.
Met de controller, die tot nu toe verwezenlijkt is, zijn we in staat de juiste besturingssignalen aan de disk aan te bieden voor een seek, write en read commando.

INLEIDING.

In de vakgroep EB wordt gewerkt aan het realiseren van een computersysteem. Bij dit systeem is het bedoeling om de taken, die specifiek zijn voor een bepaald device uit te laten voeren door een intelligente device controller. Op deze manier wordt de CPU niet belast met het besturen van het device waarmee veel CPU-tijd verloren zou gaan. De zo ontstane zelfstandige devices worden aan het kanaal gekoppeld via een buffergeheugen en een channel controller. Deze devices kunnen door middel van een schakelmatrix met elkaar verbonden worden.Om de device controller voor verschillende devices geschikt te maken moet hij snel en universeel zijn. Het "universele" kan worden bereikt door de controller microprogrammeerbaar te maken. Door een ander microprogramma te gebruiken is de controller geschikt te maken voor een ander device.

Mijn afstudeeropdracht hield in het koppelen van een 2314 disc van IBM aan één van de 8080 systemen, die in de vakgroep EB gebruikt worden. In eerste instantie is bekeken of de device controller, ontworpen door voorgaande afstudeerders (literatuur 1,2 en 3), voor dit doel gebruikt kan worden. Vervolgens is bestudeerd of het mogelijk is een universele controller te realiseren m.b.v. een microprocessor, hetgeen reeds door A.M.G. Claessen is gesuggereerd.

In het eerste hoofdstuk zal ingegaan worden op verschillende I/O configuraties, zoals die zich in de loop der jaren ontwikkeld hebben. Daarna zal het ontwerp van een microprogrammeerbare controller besproken worden. Hierna zal in het kort beschreven worden waarom en hoe een microprocessor gebruikt zou kunnen worden. In hoofdstuk IV zal de IBM 2314 disc beschreven worden. Vervolgens zal de mogelijke realisatie van de besturing van de disc m.b.v. een microcomputer gegeven worden.

HOOFDSTUK I: Algemene geschiedenis van de I/O.

1.1. Peripheral devices.

Een computersysteem bestaat uit één of meerdere processoren en een aantal devices, die zorg dragen voor de communicatie van de centrale processor met de buitenwereld ook wel peripherals genoemd.
Heel algemeen kan men een computersysteem weergeven met de tekening in figuur 1.1.

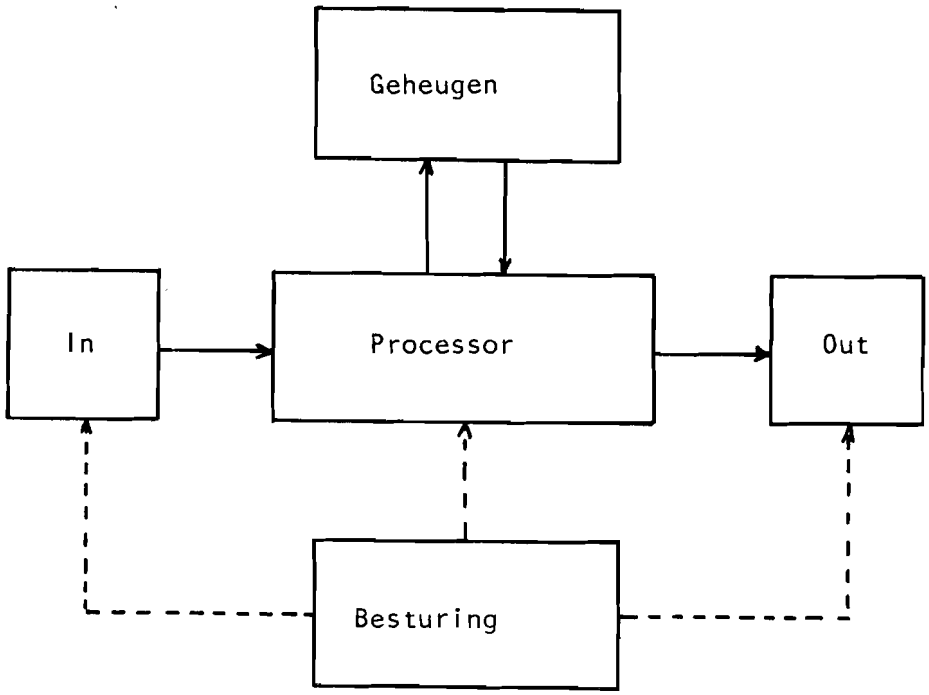


fig.1.1. De globale opbouw van een computersysteem

Deze peripherals hebben in het algemeen de eigenschap elektrische signalen om te zetten en op te slaan in een of ander medium en/of omgekeerd. Voorbeelden van deze devices zijn: kaartlezer, regeldrukker, ponsbandlezer, teletype en CRT-terminal, magnetic tape unit, disc drive en card strip.

De genoemde devices verschillen in het algemeen wat betreft de besturing, dataformaten en datacodes.

1.2. De ontwikkeling van de I/O controller.

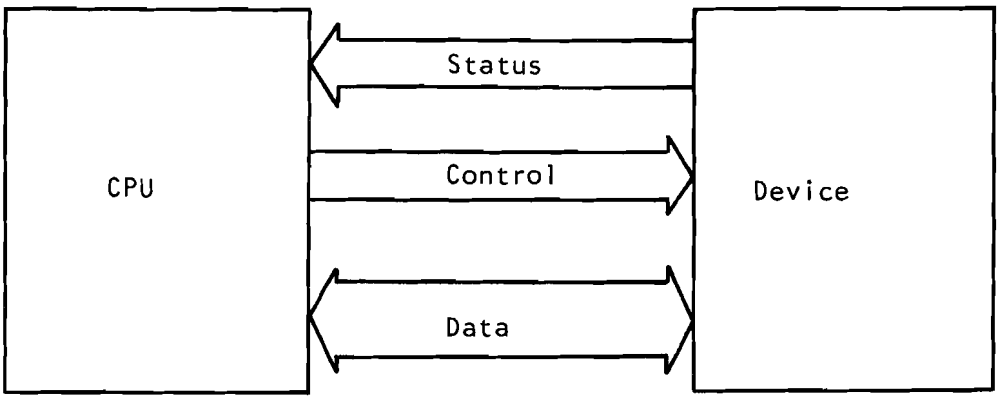


fig.1.2. Geprogrammeerde I/O

We spreken van geprogrammeerde I/O wanneer alle functies nodig voor de realisatie van datatransport van en naar een processor door deze processor zelf wordt verzorgd. Dat wil zeggen het testen van de status van een I/O device en het besturen en het verzorgen van het datatransport van en/of naar het device. In het operating system van de processor bevinden zich een aantal zgn. I/O-handlers voor elk van de aangesloten typen I/O devices.
Het nadeel van bovenstaand concept is dat de processor wanneer hij met I/O bezig is niet beschikbaar is voor specifieke CPU taken.
De volgende stap in de geprogrammeerde I/O is het toepassen naast de normale processor van een specifiek voor I/O ontworpen processor. Het principe hiervan is getekend in figuur 1.3.

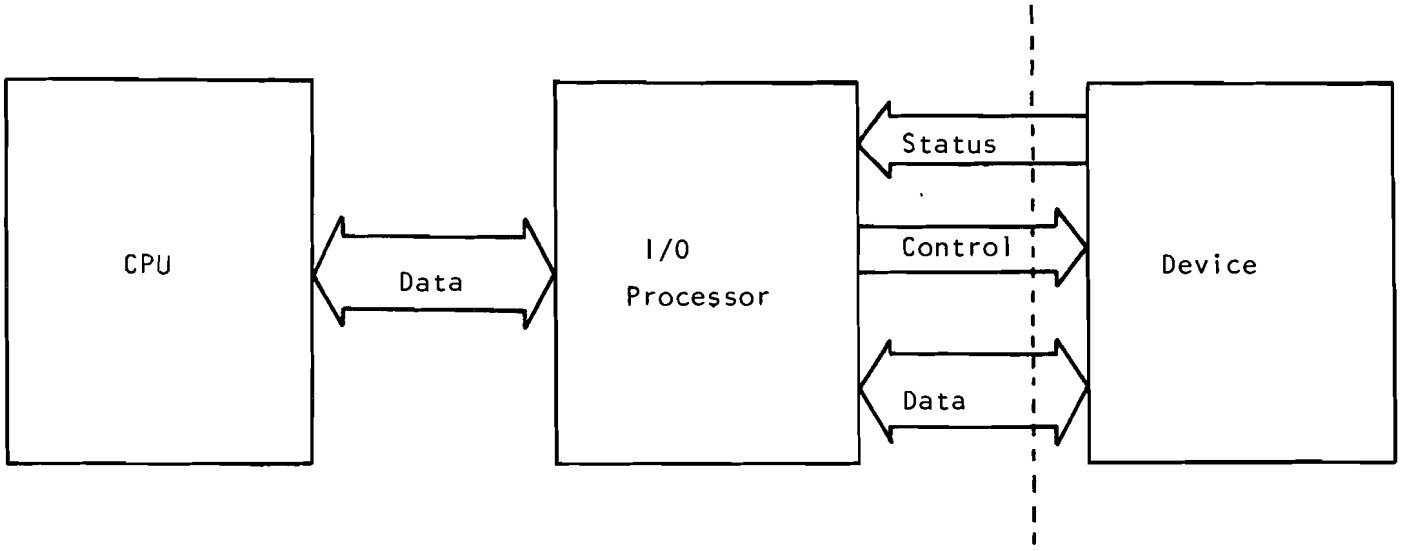


fig.1.3. I/O m.b.v. een I/O processor

Omdat het hardware ontwerp m.b.v. de moderne LSI circuits steeds eenvoudiger, goedkoper en completer wordt,wordt elk van de peripherals uitgerust met een stuk autonome hardware (I/O controller) waardoor we de situatie krijgen, die weergegeven is in figuur 1.4.

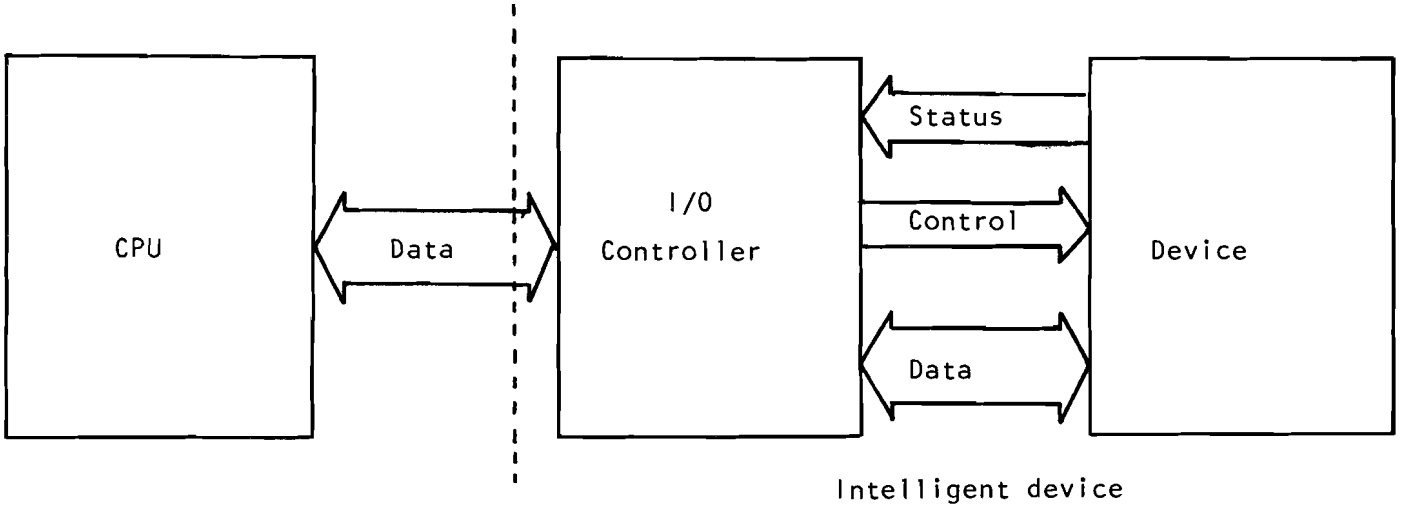


fig.1.4. I/O controller

De CPU kan nu volstaan met het geven van een bestemming of oorsprong van de data en hij kan deze data met grote snelheid van het device overnemen of naar het device zenden.
Het ontwerp,zoals dat in de vakgroep EB gemaakt is,verdeelt de I/O controller in twee delen nl. de channel controller en de device controller. De eerste deelcontroller zorgt voor de communicatie van en naar de CPU en de aanpassing aan de CPU zijde.
Aan de kant van de CPU zijn de devices door dit concept aan elkaar gelijk. De device controller zorgt voor de verdere aanpassing aan het device met name het datatransport naar of van het device zijnde: het omzetten van codes en het omzetten van dataformaten waaronder het toevoegen of strippen van foutencorrigerende codes, en de besturing zijnde: het testen op status en het geven van commando's. Verder wordt in dit deel van de I/O controller gezorgd voor de elektrische aanpassing aan het device, zoals het omzetten van signaalniveau's.
Samenvattend zijn de taken van de I/O controller:

- het omzetten van de commando's van de CPU naar commando's voor het device.
- het testen van de status van het device.
- het besturen van het datatransport
- het controleren van het datatransport.

Wanneer we de eerste I/O controller wat betreft opzet bekijken dan zien we dat deze geheel waren opgebouwd uit hardware. Zij waren specifiek ontworpen voor één device en een beperkt aantal CPU's. Het nadeel is dus dat zij niet universeel zijn en alleen gewijzigd kunnen worden door wijziging van de hardware.
Door het gebruik van een microprogrammeerbare controller of controller gebaseerd op het gebruik van een processor kan de I/O controller een veel universeler karakter krijgen.

HOOFDSTUK II: Microprogrammeerbare I/O controller.

In dit hoofdstuk wil ik in het kort ingaan op het principe van microprogrammeren. Ook de globale opbouw van een microprogrammeerbare machine zal gegeven worden.
Als voorbeeld wordt ingegaan op de realisatie van het concept van een microprogrammeerbare I/O controller van Claessen.

2.1.Microprogrammeren.

In een microprogrammeerbare machine worden verschillende commando's uitgevoerd door een opeenvolging van microinstructies.
Een microinstructie kan men zien als een binair woord waarvan de breedte bepaald wordt door het aantal controlesignalen, die nodig zijn om de machine te besturen. Deze woorden worden opgeslagen in een geheugen (het control store). Door nu de juiste woorden te adresseren verkrijgt men de gewenste controlesignalen. De adressen voor het control store worden geleverd door de sequencer.
Hieruit kan men dus afleiden dat de machine in eenvoudige vorm uit twee delen bestaat:

1) Een sequencer, die de adressen voor het control store genereert.
2) Een control store, die de microprogramma-woorden bevat.

De woorden in het geheugen worden ingedeeld in een aantal velden. Deze velden kunnen uit een verschillend aantal bits bestaan en de naam van het veld geeft aan voor welk deel van de schakeling de besturingssignalen dienen.
Wanneer we een control store van N woorden bij M bits bekijken dan zien we het volgende:

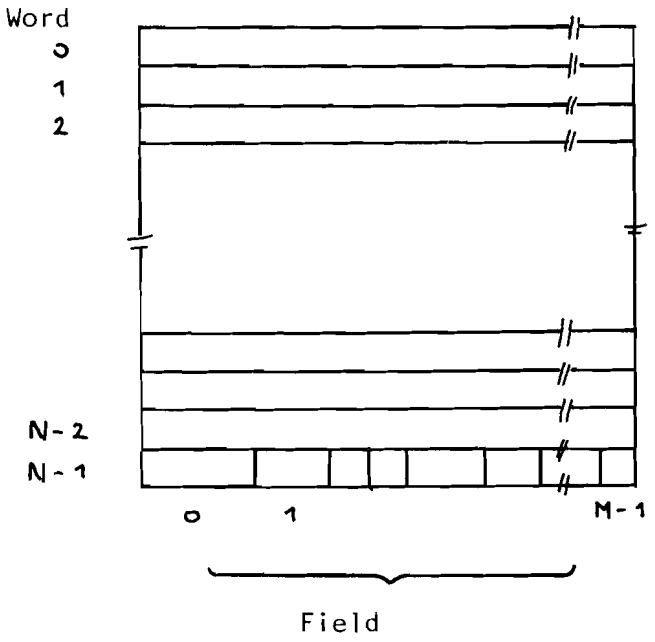


fig.2.1. De opbouw van het control store

2.2. De opbouw van een microprogrammeerbare machine.

Voor het besturen van de schakeling moeten een aantal micro-instructies na elkaar worden uitgevoerd. De sequencer zorgt voor de adressen van de microinstructies.
In de eenvoudigste vorm kan men als sequencer een adres

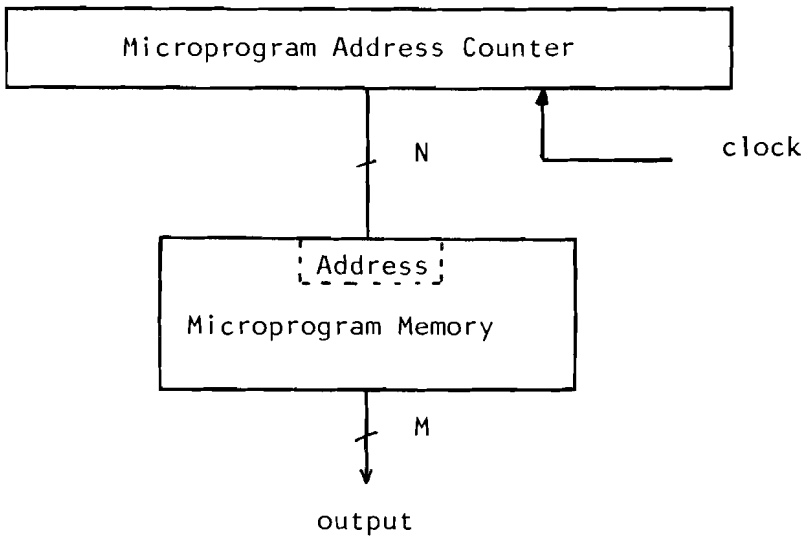


fig.2.2. Sequencer opgebouwd m.b.v. een counter

counter gebruiken. Bij elke klok wordt de counter met één opgehoogd en levert het adres voor de volgende instructie. Men krijgt dan de schakeling, die weergegeven is in figuur 2.2.
Het nadeel van deze schakeling is wel duidelijk. De machine is niet flexibel omdat hij alleen maar in staat is een vast patroon van microinstructies uit te voeren. Het intern met één ophogen van het vorige adres wordt aangeduid met Continue of Execute.
De volgende stap is het inbouwen van de mogelijkheid een adres te kiezen afhankelijk van het al of niet optreden van een conditie (Branch or Conditional Branch). Deze situatie is weergegeven in figuur 2.3.

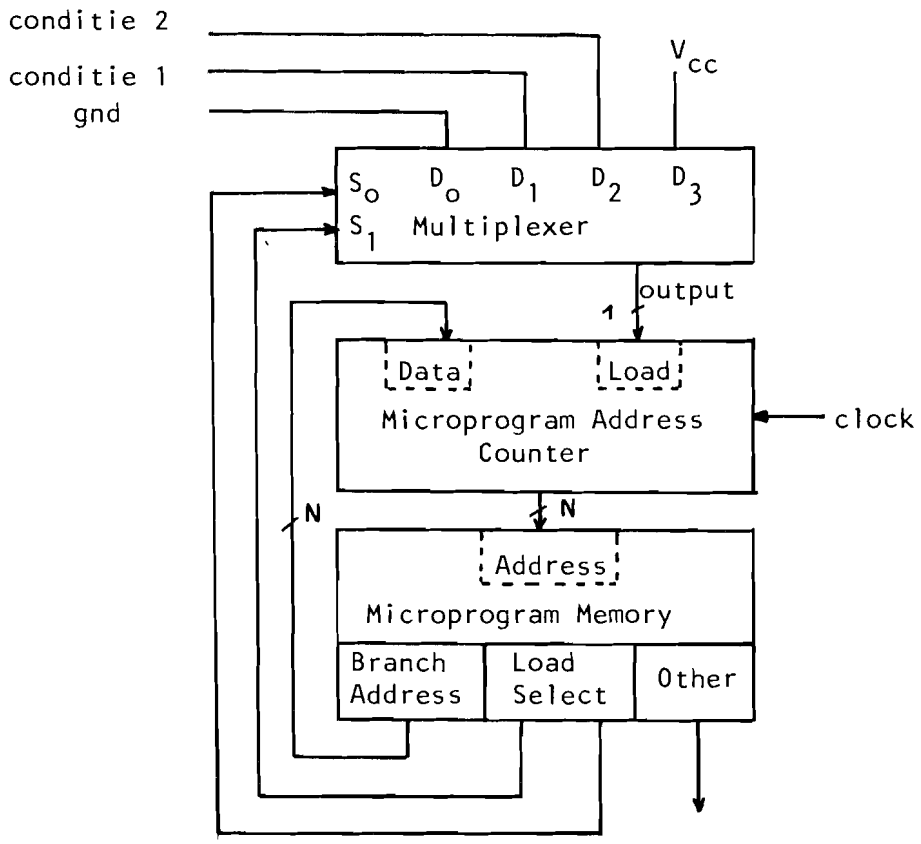


fig.2.3. Sequencer, die de instructies Continue, Branch of Conditional Branch kan uitvoeren.

Uit figuur 2.3 blijkt dat we nu een sprong kunnen uitvoeren afhankelijk van twee condities. We hebben nu vier mogelijkheden en welke van de vier wordt bepaald door de selectielijnen S_0 en S_1.
Wanneer S_0 en S_1 beide nul zijn wordt D_0 geselecteerd, die de load control inactief maakt, zodat opeenvolgende instructies uitgevoerd worden (continue). Bevat het load select field binair een "1" dan wordt of het vorige adres met één opgehoogd als de conditie false is of wordt het branch adres bij de volgende klok in de counter geladen als de conditie true is (conditional branch). Hetzelfde geldt als input D_2 geselecteerd wordt (S_0=0, S_1=1). Wordt D_3 geselecteerd (S_0=1, S_1=1) dan wordt de counter onafhankelijk van condities bij de volgende klok geladen met het branch adres (unconditional branch).

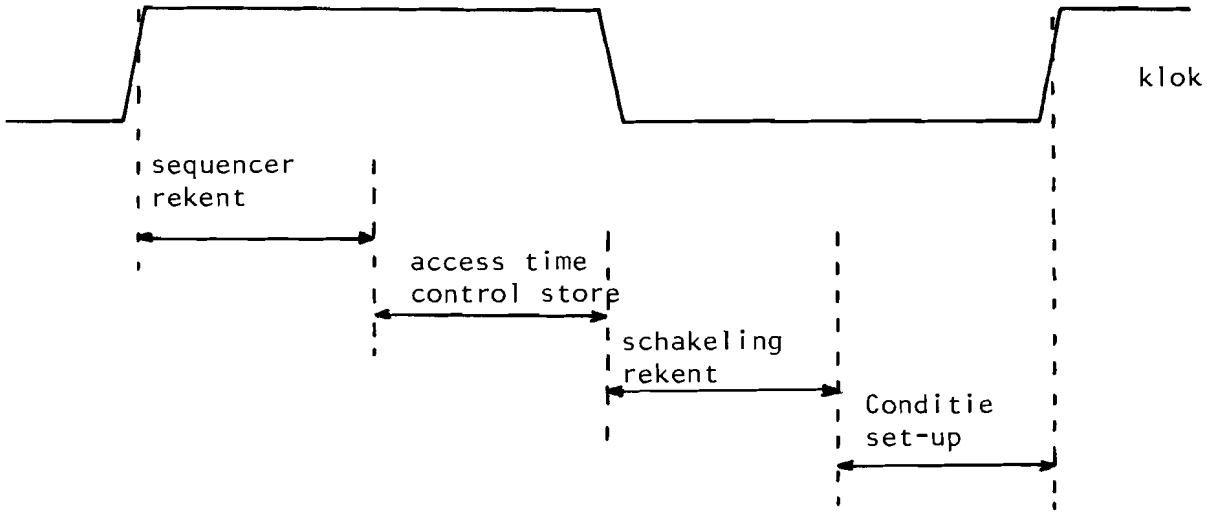


fig.2.4. Timing van de machine wanneer er geen gebruik van een pipeline register gemaakt wordt.

In figuur 2.4 ziet U de timing van deze machine. De snelheid wordt bepaald door de tijd, die de afzonderlijke delen van de schakeling nodig hebben. Om nu snelheid te winnen wordt

tijdens het uitvoeren van de instructie de volgende micro-instructie bepaald. Dit staat bekend als "pipelining". Het blokdiagram vindt U in figuur 2.5. Het pipeline register bevat de microinstructie, die door de machine uitgevoerd wordt. Parallel hiermee wordt het adres van de volgende instructie berekend en doorgegeven aan het control store. De inhoud van het geadresseerde geheugenwoord wordt gelatched en klaar gezet aan de ingang van het pipeline register.

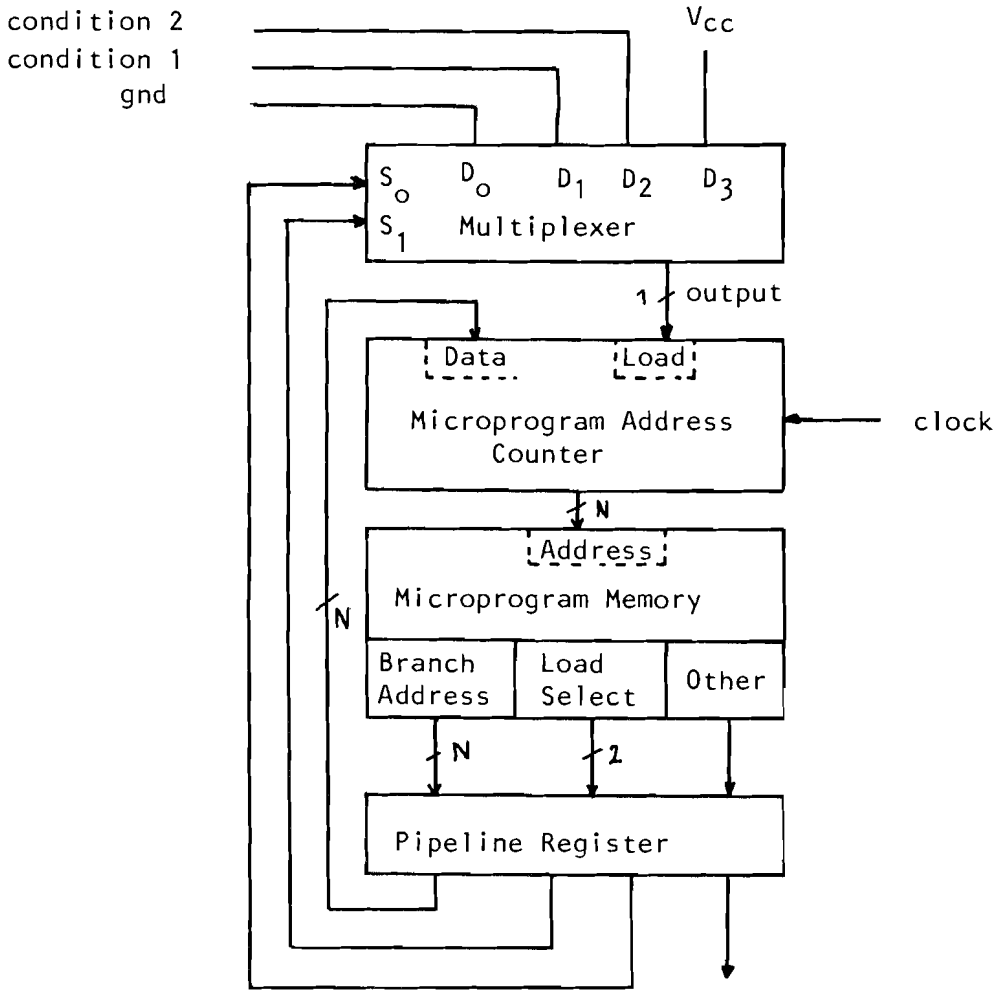


fig.2.5. Sequencer en control store met tebruik van een pipeline register.

Op deze manier winnen we de access time van het control store. Het nadeel is echter dat de conditie voor het uitvoeren van conditionele sprongen te laat bekend is. Bij het ontvangen van de conditie staat het nieuwe controle woord voor het pipeline register klaar om ingeklokt te worden. Er moet dan dus een dummy slag gedaan worden,waarbij de schakeling geen operatie uitvoert. Figuur 2.6 laat de timing van deze schakeling zien.

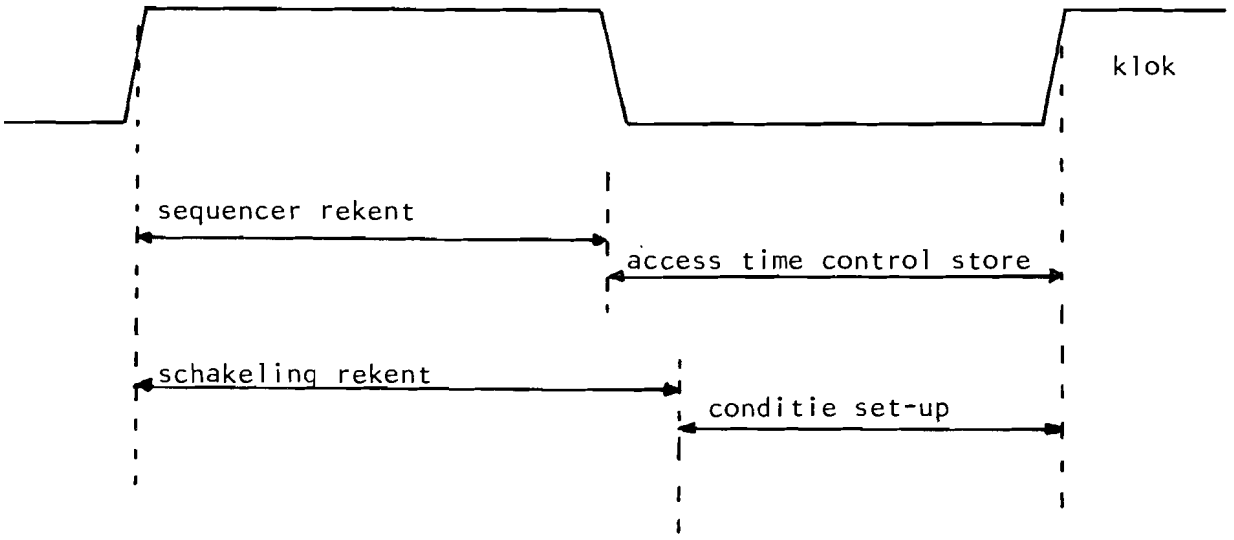


fig.2.6. Timing van de machine bij gebruik van een pipeline register.

Bij de programmering van mini- en microcomputers biedt het gebruik van subroutines veel voordelen. Het aantal controle woorden dus ook de omvang van het control store kan daardoor gereduceerd worden. Het is dan nodig het adres te onthouden, waarnaar na het uitvoeren van de subroutine teruggekeerd moet worden. Om dit te realiseren bevatten veel sequencers een ingebouwde stack en een daarbij horende stack pointer. De controlesignalen, die voor deze stack nodig zijn, zijn FE (File Enable), die de stack enabeld en een push/pop controlesignaal (PUP) om aan te geven of het een push of pop is.

De stack pointer geeft het adres van de laatste micro-instructie, die op de stack weggezet is. Via de multiplexer kan dit adres doorgegeven worden aan het control store.
Sommige sequencers bezitten ook conditie logica, waardoor een (un)conditional branch address bepaald kan worden.
Met deze bouwstenen kan een microprogrammeerbare machine opgebouwd worden. In figuur 2.3 en 2.5 is met een pijl met de aanduiding "other" een aantal bits aangegeven, die nodig zijn voor de te besturen schakeling. Wanneer men een computer met deze bouwstenen wil ontwerpen dan zal op dit veld "other" de ALU (Arithmetic Logic Unit) en verschillende I/O-poorten aangesloten worden.

2.3. Microprogrammeerbare universele I/O controller.

De I/O controller is een van de CPU losstaande processor geworden. De data waarmee de controller bestuurd wordt is opgeslagen in een microprogramma geheugen ofwel control store. Dit control store is vaak opgebouwd uit Read Only Memory (ROM) of bij meer recente controllers uit Random Access Memory (RAM). Dit laatste heeft als voordeel dat de inhoud gemakkelijk gewijzigd kan worden maar het nadeel is dat het voor het opstarten van de machine gevuld moet worden. Deze microprogrammeerbare I/O controller is meestal opgebouwd, zoals aangegeven in figuur 2.7.

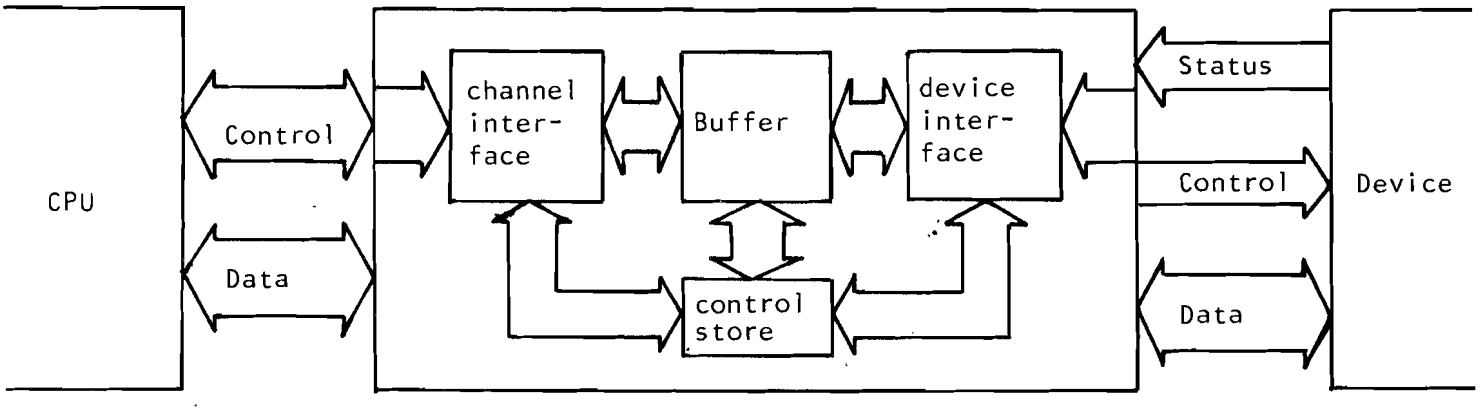


fig.2.7. Microprogrammeerbare I/O controller.

Uit figuur 2.7 blijkt dat men vier gedeelten kan onderscheiden. De beide interfaces zorgen voor een hardware aanpassing aan de channelzijde en aan de kant van het device. Het device interface bevat o.a. serie-parallel en parallel-serie omzetters en een CRC-generator of checker. Verder treft men het control store aan,die het microprogramma bevat en een klein buffer. Dit buffer dient om eventuele vertragingen tussen de beide interfaces op te vangen. De controller uit figuur 2.7 is universeel van opzet omdat door een ander microprogramma en, indien nodig, het vervangen van een van de interfaces de controller aangepast kan worden aan een andere CPU of device. Nu is het nog mogelijk om het control store in twee delen te splitsen, zoals figuur 2.8 toont. Het ene control store zorgt voor de communicatie tussen databuffer en device en het andere voor de communicatie tussen CPU en databuffer. Men kan dan twee deelcontrollers onderscheiden nl. de device controller en de channel controller.
Het splitsen van het control store heeft een aantal voordelen:

1) Bij het aanpassen aan een ander device of CPU hoeft er minder programmatuur gewijzigd te worden.
2) De beide controllers werken nu onafhankelijk van elkaar en zijn in staat om elkaar te testen bij het optreden van fouten bij één van hen.
3) De snelheid van het datatransport van de CPU naar de controller is onafhankelijk van de snelheid van het device dus de CPU wordt minimaal bezet gehouden.
4) Voor de CPU zijn alle peripheral devices, hoe complex deze ook zijn, precies hetzelfde.

Het buffer zal echter groter moeten worden omdat de controllers informatie van elkaar moeten overnemen en ieder voor zich een datatransport op moet zetten. Bij het buffergeheugen behoort hardware, die zorgt voor de toewijzing van de bus, de in dit concept genoemde Bus Allocation Unit.

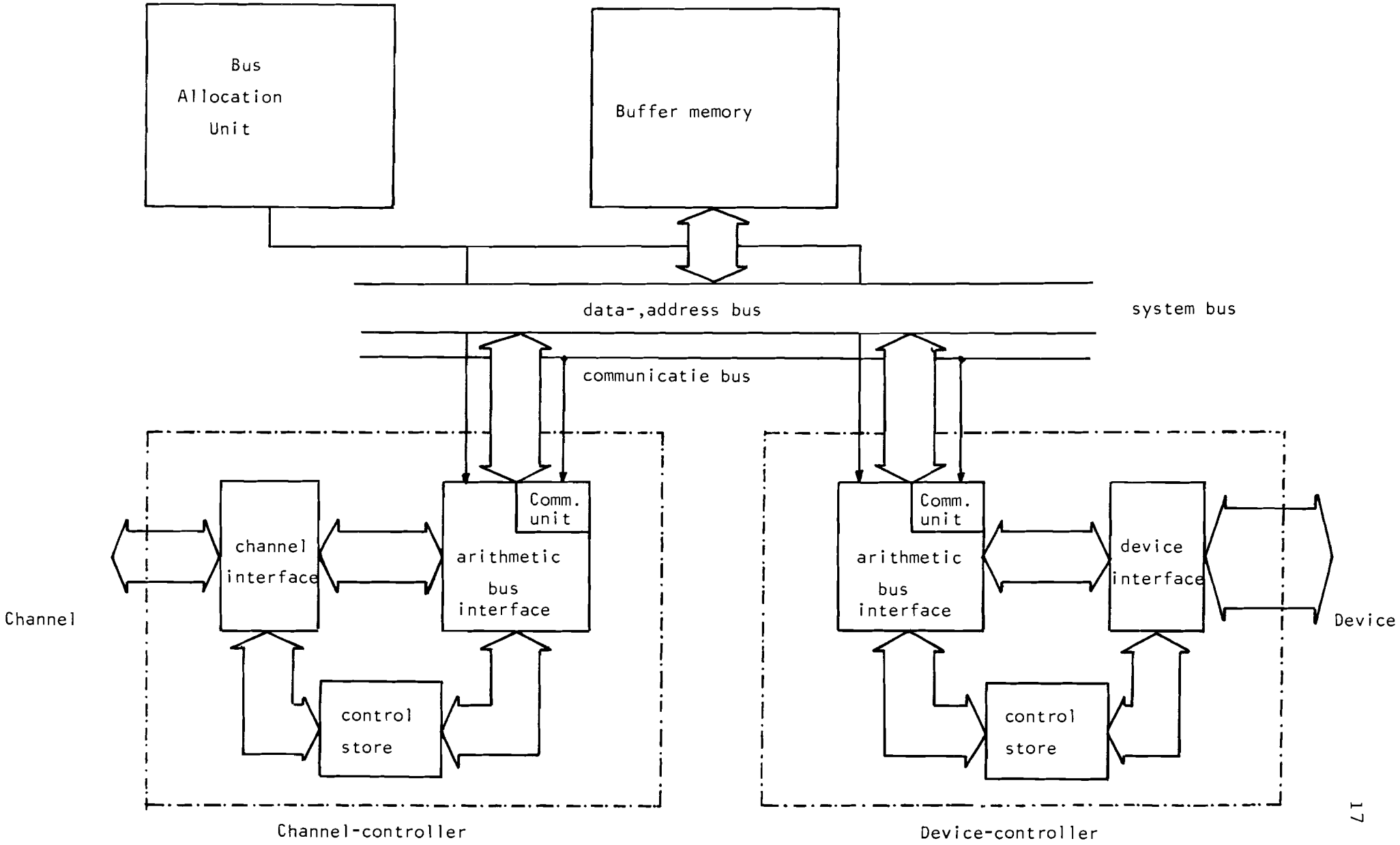


fig.2.8. Splitsing in twee deelcontrollers.

Deze unit is noodzakelijk omdat beide controllers via één systeembus met het geheugen verbonden zijn.
De toewijzing zou ook geregeld kunnen worden m.b.v. de communicatie bus. Dit verdient niet de voorkeur wanneer men meerdere devices op een computer aan wil sluiten. In deze situatie zijn immers meerdere device controllers op één channel controller aangesloten via de systeembus. Dit is weergegeven in onderstaande figuur.

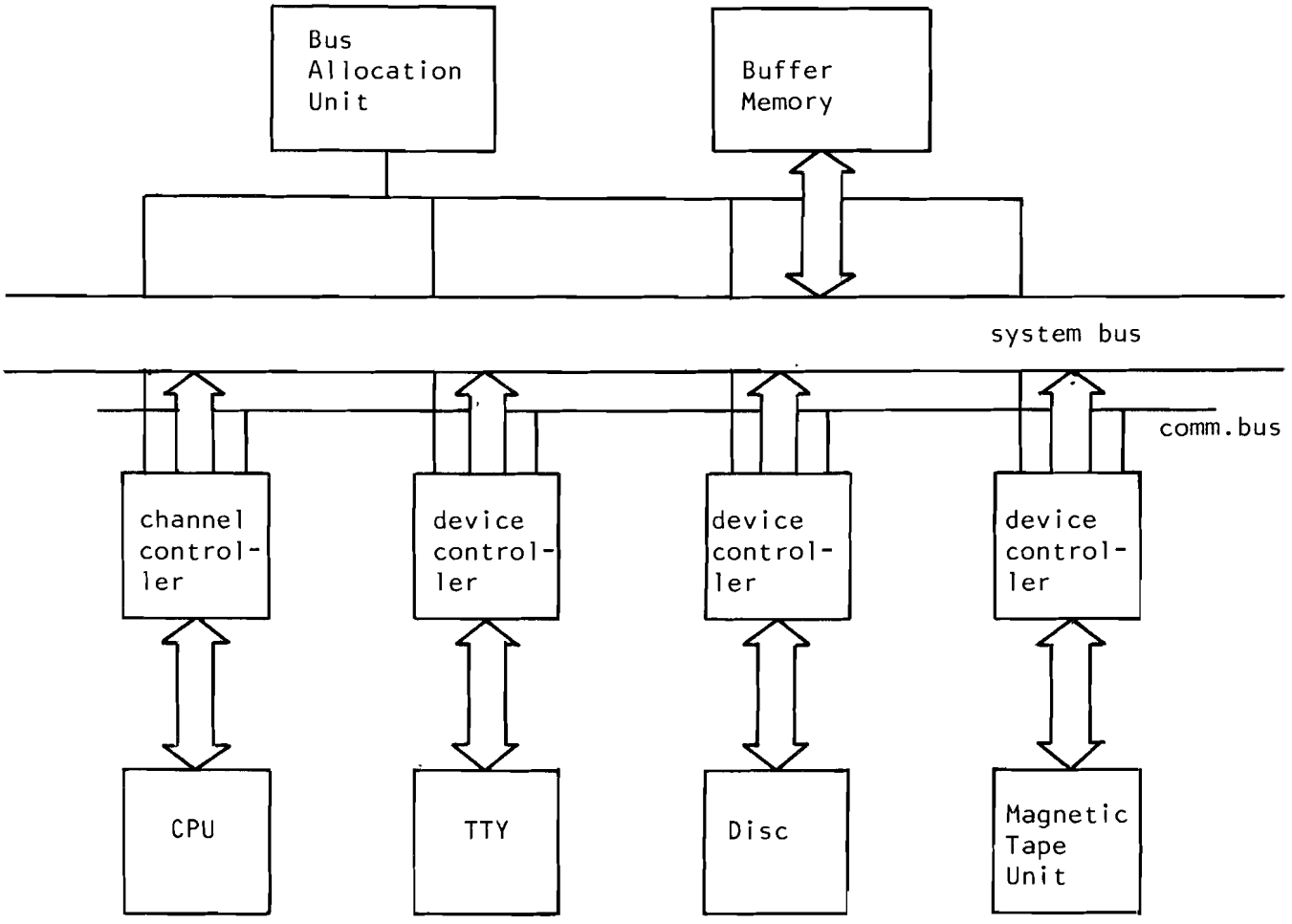


fig.2.9. Meerdere devices aansluiten op één CPU.

Het is mogelijk met deze bus allocation unit de boek-houding van het buffergeheugen bij te houden. Hij kan

dan informatie bevatten, zoals de adressen waarop de datablokken beginnen, de lengte van de blokken en of de data al door een andere controller is overgenomen.
De communicatie bus verzorgt dan de verdere communicatie tussen de beide controllers.

2.4. Het gebruik van microprogramma's.

Vrij recent is een publikatie van AMD (Advanced Micro Devices) verschenen, waarin een microprogrammeerbare device controller beschreven wordt (literatuur 5). In opzet verschilt deze controller op één ding na met het ontwerp van Claessen. Bij het ontwerp van AMD heeft men het zoeken op key buiten beschouwing gelaten waardoor er minder hardware nodig is.
Tot slot nog iets over het maken van microprogramma's. Het is vereist dat men dan precies weet wat de functie is van elk onderdeel en wanneer de controlesignalen aangeboden moeten worden. Helaas is er nog geen programma in de roulatie waarmee men m.b.v. een hogere programmeertaal een microprogramma kan maken.

HOOFDSTUK III: Microprogramming versus microprocessor.

Wanneer men een keuze moet maken hoe de I/O controller opgebouwd zal moeten worden dan zal men zich af moeten vragen waarom de controller opgebouwd zal worden met microprogrammeerbare bouwstenen en niet met een microprocessor, die in vele versies in grote mate voor handen zijn.
Omdat de controller universeel van opzet dient te zijn is de snelheid van de controller erg belangrijk. De eis, die opgelegd werd, was er voor te zorgen dat ook toekomstige devices hiermee bestuurd kunnen worden.
Om een indruk te geven van de grootte van de baudrate het volgende:

De baudrate van: - magnetic tape unit : 1,25 Mbyte/sec
- disc drive (IBM 2314): 2,5 Mbit/sec
(312 kbyte/sec)

In de toekomst zal het waarschijnlijk mogelijk zijn om in plaats van één track één cylinder parallel uit te lezen, waardoor de datarate van de disc 2,5 Mbyte/sec zal bedragen of misschien nog meer wordt.
De microprogrammeerbare I/O controller is zodanig ontworpen dat deze een baudrate van 6 Mbyte/sec heeft.
Een inventarisatie van de huidige microprocessoren leert ons dat er geen enkele microprocessor op de markt is die een klok van 6 MHz heeft, hetgeen niet wil zeggen dat hij bij een klok van 6 MHz een datarate van 6 Mbyte/sec kan halen. Een I/O controller waarbij op key getest moet worden is dus niet realiseerbaar.
Een benadering van deze snelheid kan verkregen worden met de interpreter 8X300 van Signetics, die een datarate van 4 Mbyte/sec heeft. De 8X300 is eigenlijk geen microprocessor maar is door zijn beperkte instructieset en mogelijkheden eerder ontworpen voor snelle controller doeleinden.
De instructie-cycle time bedraagt 250 nsec. In deze tijd

decodeert de interpreter de instructie, haalt de data binnen, bewerkt deze data en staat de resultaten van de bewerking weer af. Tevens is het met de 8X300 mogelijk om bitoperaties uit te voeren op data, wat goed gebruikt kan worden voor het testen van de statussignalen.
Een nadeel van deze interpreter is dat deze geen interrupt voorziening heeft. Dit kan wel gerealiseerd worden ten koste van veel hardware. Voor het controleren op keywoorden en headers moet een hardware comparator toegevoegd worden.
Mede uit practische overwegingen (lange levertijd van componenten voor de microprogrammeerbare controller) en het feit dat het in de verwachting ligt dat er snellere microprocessoren op de markt komen, is besloten de toepassing van een microprocessor in een I/O controller te bestuderen.
Omdat binnen de vakgroep veel gebruik gemaakt wordt van de 8080 microprocessor van Intel is deze processor toegepast bij de realisatie.
De 8080 processor zelf is erg traag. Het datatransport verloopt via de accumulator. Wanneer we gebruik maken van Direct Memory Access (DMA) gaat het datatransport tussen het device en het geheugen rechtstreeks via de databus. De toepassing van een DMA controller is weergegeven in figuur 3.1.
Omdat zowel de DMA controller als de CPU directe toegang tot het geheugen hebben moet om problemen te voorkomen of de CPU of de DMA controller busmaster zijn. Om busmaster te worden moet de DMA controller een busrequest naar de CPU zenden. Wanneer de DMA controller busmaster is vinden we voor de datatransfer een tijd van 1 µ sec, hetgeen neerkomt op een datarate van 1 Mbyte/sec.

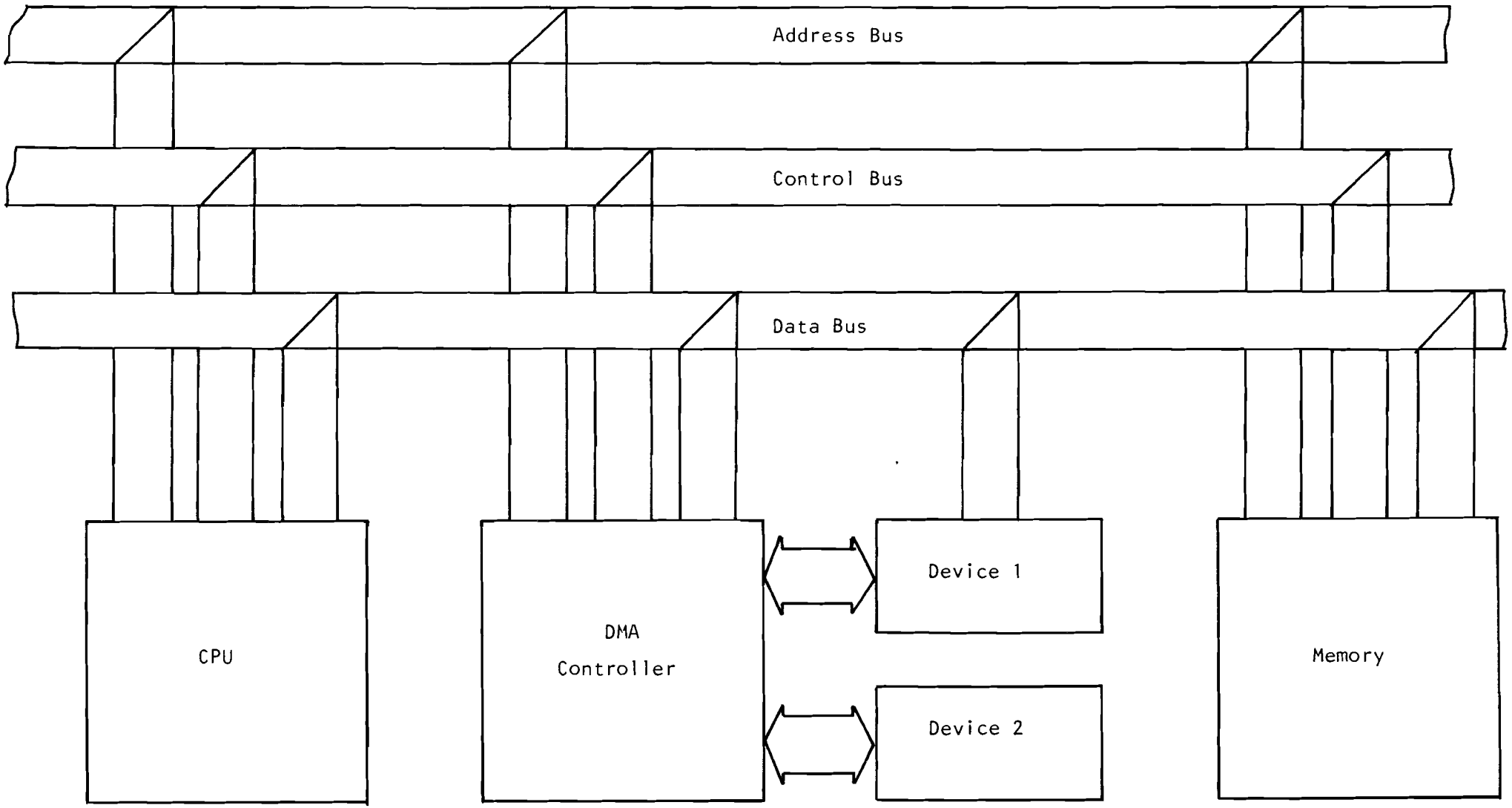


fig.3.1. De toepassing van een DMA controller.

HOOFDSTUK IV: De IBM 2314 disc.

4.1. De verdeling van data in file's, records en fields.

Voordat we overgaan tot de beschrijving van de IBM 2314 disc zal eerst in het kort aangegeven worden hoe data op verschillende wijzen in quanta verdeeld kan worden.
Om een hoeveelheid data aan te geven worden hoofdzakelijk de begrippen file, record en field gebruikt. We moeten hierbij een onderscheid maken of met deze hoeveelheid data programmatuur of een databestand wordt bedoeld.
Als men het over programmatuur heeft dan is de file de belangrijkste hoeveelheid informatie voor de gebruiker. Een file bestaat uit één of meer records, welke een fysische eenheid van informatie is. Deze records samen bevatten data, die voor de gebruiker bij elkaar hoort.
Bij een databestand is de record de belangrijkste hoeveelheid informatie. Een record is een verzameling informatie over één individu of item. Bij een boekenwinkel bevat een record bv. het aantal, dat nog voorradig is, de prijs, de schrijver enz.
Een record is onderverdeeld in fields, die de verschillende aspecten van het individu of item, waarop de record betrekking heeft, beschrijven. De fieldname, die aan het field gegeven wordt, is meestal hetzelfde als de naam van het individu of item waarop de field value, de waarde van het field, betrekking heeft. Door een key field wordt een record een-eenduidig bepaald, zoals bv. het identiteitsnummer bij een studentenadministratie. Een verzameling soortgelijke records wordt een file genoemd.
Deze indeling van data in file-record-field wordt gehandteerd door de gebruiker van het computersysteem. In de machine wordt de data ingedeeld in blocken, wat sterk afhangt van de bovenstaande indeling.
Een block is de grootste hoeveelheid informatie, die gedurende een I/O operatie naar of van een device wordt

getransporteerd.
De totale hoeveelheid informatie waarover de CPU via een device kan beschikken wordt een volume genoemd. Bij een disc drive is dat een disc pack.
De blocken zelf zijn weer onderverdeeld in subblocken, deze in areas en dan volgen de words, bytes en de bit, als kleinste eenheid. De grootte van een block hangt af van de indeling van de disc. Hierop komen we in de volgende paragraaf terug.

4.2. De algemene beschrijving van een disc.

Wanneer grote hoeveelheden informatie moeten worden opgeslagen, die toch relatief snel toegankelijk moeten zijn, wordt vaak een Direct Access Storage Device (DASD) gebruikt waartoe o.a. een disc behoort.
Bij een disc wordt de data op een laag magnetiseerbaar materiaal opgeslagen, die op een aluminium schijf is aangebracht. Een aantal van deze schijven, die aan één of beide zijden van magnetiseerbaar materiaal zijn voorzien, vormen een disc pack, zoals U kunt zien in figuur 4.1.

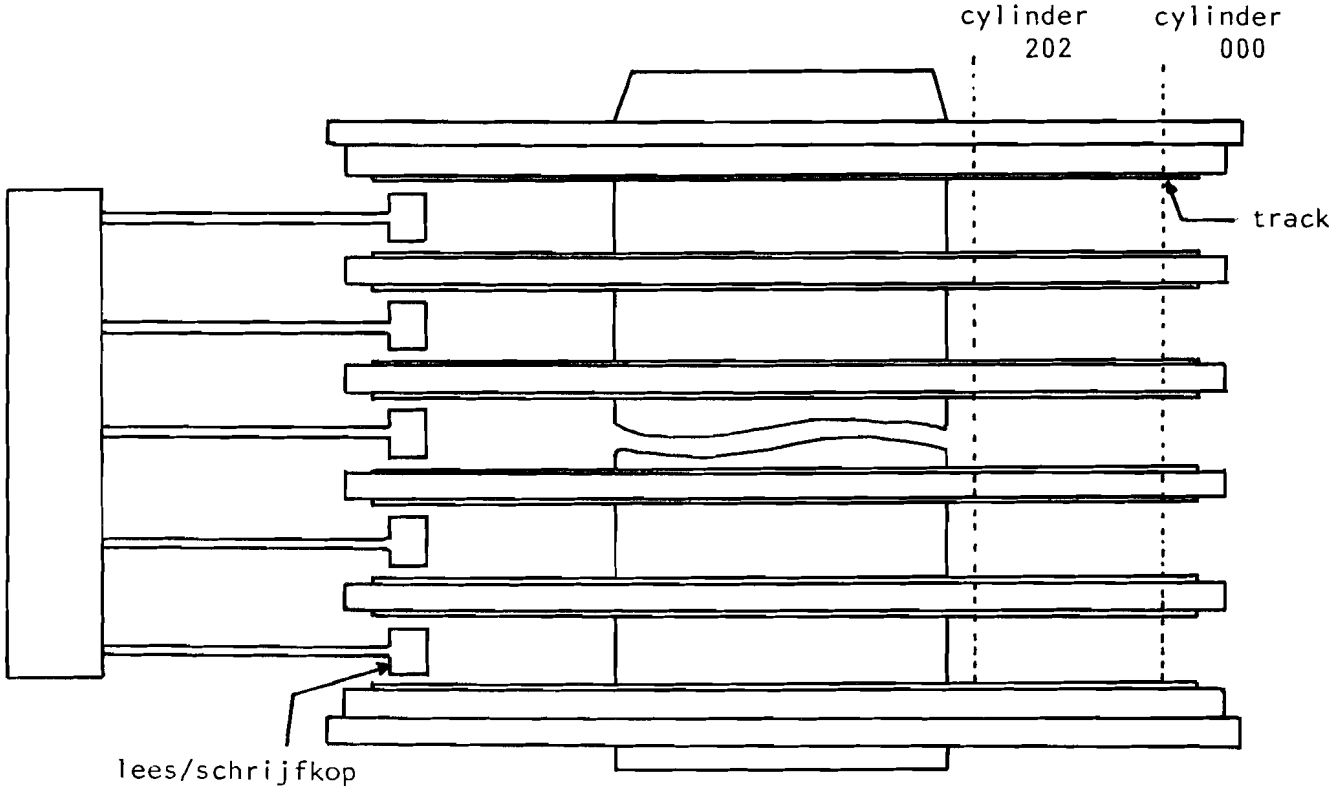


fig.4.1. Disc pack met toegangsmechanisme.

Om de disc te kunnen beschrijven hebben we voor elke schijf van de disc pack minstens een lees/schrijfkop nodig. Deze zijn bevestigd op een kamvormige constructie, die door een mechanisme in de pack geschoven kan worden.
Het spoor op een oppervlak, dat zonder beweging van de arm door een lees/schrijfkop bereikt kan worden, wordt een track genoemd. De IBM 2314 heeft per oppervlak 203 tracks waarvan 200 normale en 3 reserve.
De tracks in een pack, die zonder beweging van de arm bereikbaar zijn door een andere kop te adresseren, worden een cylinder genoemd. Bij de 2314 heeft elke cylinder 20 tracks.
De tijd, die nodig is voor de access en de datatransfer bestaat uit vier delen: access motion time, head selection time, rotational delay time en de datatransfer time.

Access motion time : Dit is de tijd die nodig is om het toegangsmechanisme te positioneren op een cylinder, die het gespecificeerde record bevat. Wanneer het mechanisme zich al op de juiste cylinder bevindt dan is het niet nodig het mechanisme te bewegen en dan is de access time dus nul.
Voor de 2314 is in figuur 4.2 de minimale access time gegeven wanneer het mechanisme bewogen moet worden.
Uit figuur 4.2 blijkt dat de access time hoofdzakelijk een functie is van het aantal cylinders, dat bewogen moet worden. Voor de beweging van een cylinder is de minimum tijd 25 msec, de maximum tijd 135 msec en het gemiddelde 75 msec.

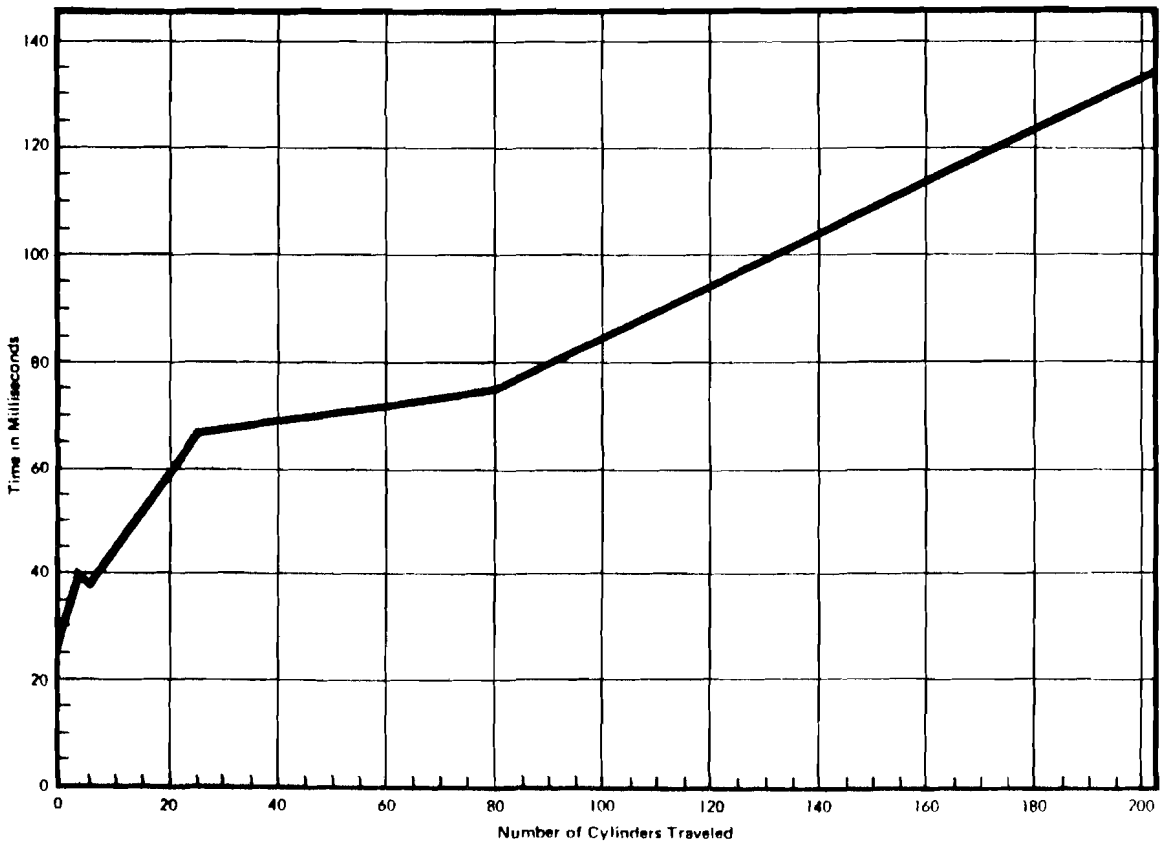


fig.4.2. De access time van de 2314.

Head selection time : Dit is de tijd die nodig is om de juiste lees/schrijfkop van het mechanisme te selecteren. Deze tijd is verwaarloosbaar klein.

Rotational delay time: Wanneer we een bepaald punt van de disc willen bereiken moet de disc meestal een stuk gedraaid worden voordat het gewenste punt zich onder de lees/schrijfkop bevindt. Deze tijd wordt de rotational delay time genoemd. De volle rotational delay time voor de 2314 is 25 msec en de gemiddelde tijd is 12,5 msec.

Datatransfer time : De tijd nodig voor de datatransfer tussen device en geheugen wordt bepaald door de

snelheid van de disc en de dichtheid van de data. Bij de 2314 is de data-rate 312 kbyte/sec, wat neerkomt op 3,2 µsec/byte.

De data wordt verdeeld in blocken. De grootte van een block wordt bepaald door de indeling van een track. Men maakt hierbij onderscheid tussen drie methoden:

1) Hard-sectorized: De disc is voorzien van inkeepingen aan de rand waardoor het oppervlak in een aantal sectoren van gelijke grootte verdeeld wordt. Dit noemt men een hard-sectorized-disc. Voor het lezen en schrijven maakt men dan gebruik van de sectorpulsen. De blockgrootte is dan bepaald door de grootte van de sector.
2) Soft-sectorized: De disc wordt ook hierbij ingedeeld in sectoren waarvan de grootte kan worden aangepast aan de te verwachten blockgrootte. De sectorscheiding wordt aangegeven m.b.v. magnetische merktekens. Dit gebeurt tijdens het zgn. formatten van de disc. De grootte van het block staat ook nu vast.
3) IBM-methode: De blockgrootte is bij deze methode variabel in tegenstelling tot de vorige twee methoden. Het is dan mogelijk om een veel efficienter gebruik van de disc te maken want bij de twee andere methoden wordt per sector een block geschreven. Het kan dan voorkomen dat de blockgrootte veel kleiner is dan de grootte van de sector.

4.3. Blockformaten.

De records zullen in blocken verdeeld moeten worden. Dit wordt gedaan door de CPU maar soms heeft ook de controller daarin een aandeel.
Bij IBM wordt het datablock in drie subblocken verdeeld, het count-, key- en datasubblock. Deze subblocken worden

door gaps van elkaar gescheiden.
Men kan vijf verschillende recordformaten onderscheiden (zie figuur 4.3):

Fixed Unblocked : Alle records hebben dezelfde lengte en elk datasubblock bevat één record.

Fixed Blocked : Ook bij dit formaat hebben alle records dezelfde lengte. Elk datasubblock bevat meer dan één record. Alle blocken hebben dezelfde lengte behalve het block op het einde van de file, die korter kan zijn. Het keysubblock bevat meestal de key van het laatste record. De records zelf bezitten ook een key waarmee ze geidentificeerd kunnen worden.

Variable Unblocked: De records hebben een variabele lengte en het datasubblock bevat één record."BL" (Blocklengte) zijn de eerste vier bytes van het block, die het aantal bytes in het block aangeven inclusief zichzelf. Daarna volgen vier bytes die de recordlengte aangeven inclusief zichzelf.

Variable Blocked : De records hebben een variabele lengte en elk datasubblock bevat een aantal records. Ook hier geven de bytes BL en RL de lengte van het block en de records aan.

Undefined : Hierbij wordt aan de gebruiker overgelaten om de records tot een datablock samen te voegen.

Bij de 2314 zijn de records variable unblocked.
Afhankelijk van de grootte van het block is het mogelijk dat een track niet vol is. Om er voor te zorgen dat de track vol komt moet een deel van een block aan het einde van een track geschreven worden en de rest op de volgende track. Door middel van de flagbyte moet aangegeven worden dat een record een overflow record is. Bij het lezen en schrijven gaat de controller

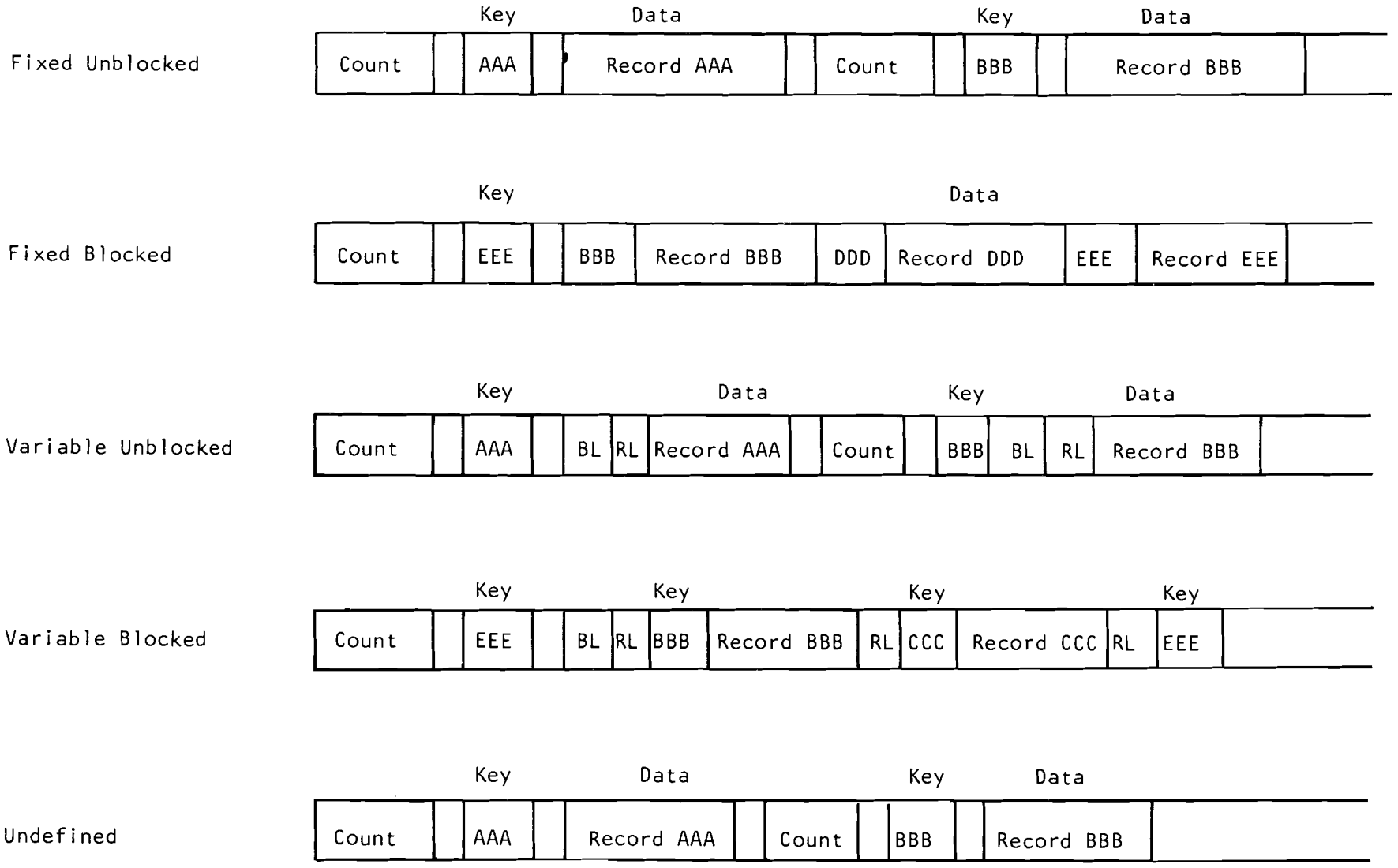


fig.4.3. Recordformaten.

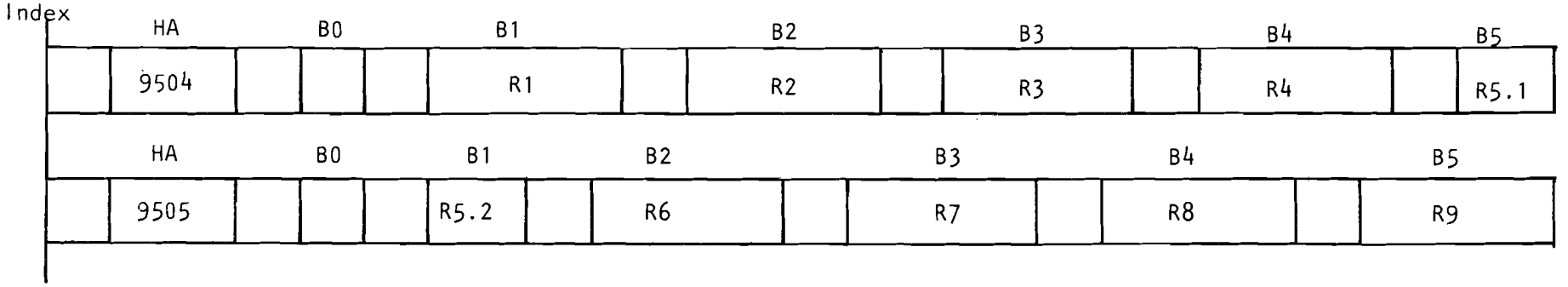


fig.4.4. Track overflow.

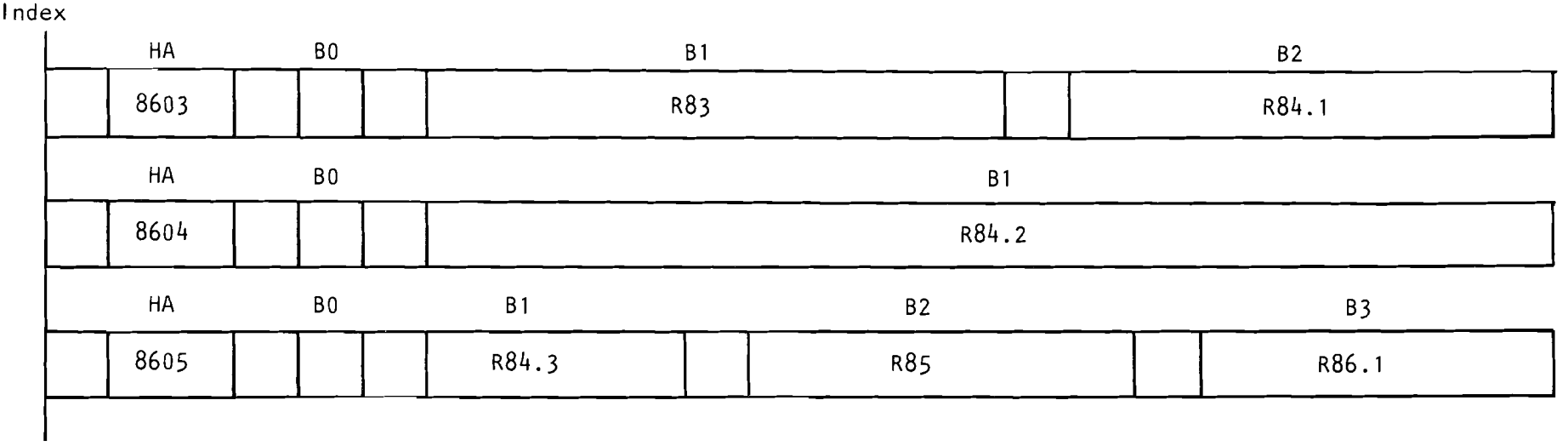


fig.4.5. Multitrack block.

na detectie van de flag door op de volgende track (figuur 4.4) Het is ook mogelijk dat een block zich over meerdere tracks uitstrekt. Dit noemt men multitrack block. Deze tracks moeten wel allen in dezelfde cylinder liggen. De record wordt nu in segmenten verdeeld. De overflow flag wordt geset in alle segmenten van de record behalve in het laatste segment. De segmenten, die na het eerste segement volgen, beslaan op het laatste segment na een hele track. Dit is weergegeven in figuur 4.5.

4.4. De track layout.

Elke track bevat zowel "nondata" informatie, zoals het track-adres, de recordlengte en de gaps tussen de datablocken en de subblocken, als data informatie.
In figuur 4.6 is de track layout weergegeven.
De track bestaat uit:

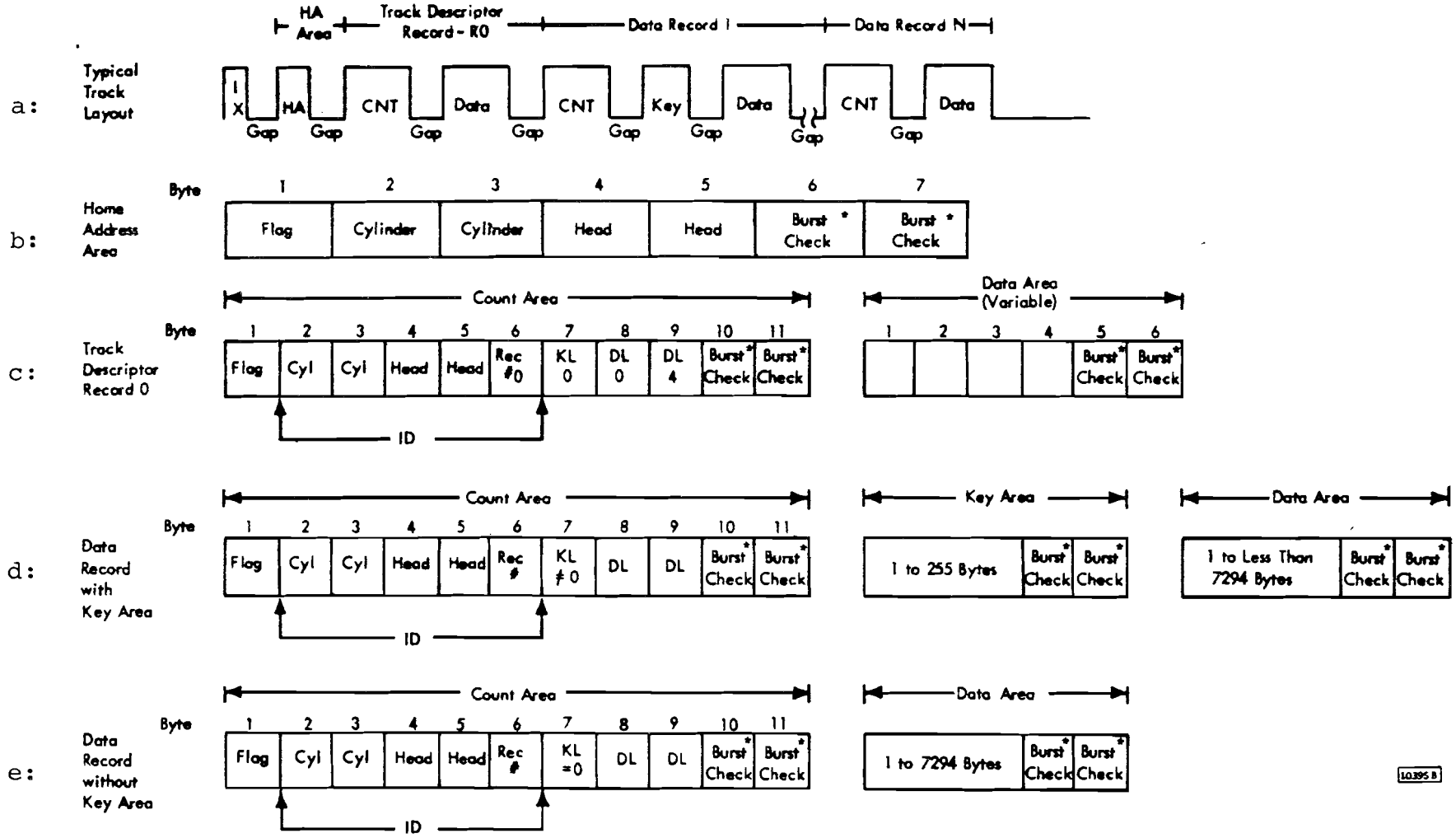
- Het Home Address (HA)
- Het Track Descriptor Record (R0)
- Een of meer datarecords, die een variabele lengte hebben.
- Gaps

De index marker (IX) aan het begin van de track dient om het fysische begin van de track aan te geven.

Home Address.

Op elke track bevindt zich het home address om de fysische locatie van de track te definieren en de conditie van de track aan te geven. Uit figuur 4.6b blijkt dat het een block van zeven bytes is, die de volgende functie hebben:

Byte	Naam	Bits	Functie
1	Flag	0-5	Geen doel, altijd nul
		6	Track conditie
			0 - goede track
			1 - defecte track



* Cyclic Code Check

fig.4.6. Track layout.

Byte	Naam	Bits	Functie
		7	Track use 0 - originele track 1 - vervangende track
2-3	Cylinder	0-7	Deze specificeren het cylinderadres
4-5	Head	0-7	Deze specificeren het headadres
6-7	Code check	0-7	Voor de foutendetectie

Track Descriptor (R0).

Na het home address volgt het tweede block: de track descriptor R0. Dit block bestaat uit een count- en data-subblock. Het countsubblock bestaat uit elf bytes (zie figuur 4.6c). De eerste byte komt overeen met de eerste byte van het home address. Byte 2-6 (ID=Identifier) bevat het cylinder-, head- en recordnummer, dat het volgnummer van de record op de track aangeeft. Bij R0 is deze dus nul. Byte 7 geeft aan uit hoeveel bytes de key bestaat (maximaal 255 bytes). Met bytes 8 en 9 wordt aangegeven uit hoeveel bytes het datasubblock bestaat waarbij de twee code check bytes niet meegerekend worden (maximaal 7294 bytes,hoewel twee bytes een grotere datalengte aan zouden kunnen geven).
In het geval dat de track defect raakt bevat het count-subblock van R0 het trackadres van de vervangende track in plaats van het trackadres van de originele track. In het countsubblock van de vervangende track staat het adres van de defecte track.

Datablock.

Het datablock kan uit drie subblocken bestaan (zie figuur 4.6d en 4.6e). Het keysubblock hoeft niet aanwezig

te zijn.
In het countsubblock van het datablock hebben alle bytes dezelfde functie als in het countsubblock van R0. Alleen in de flagbyte geven bit 0 en 1 aan of het een even record (R0, R2, R4,....) of oneven record (R1, R3, R5,.....) is en of het een overflow record is.
De lengte van het datasubblock mag varieren van 1 tot 7294 bytes.
Het is nu mogelijk om op twee manieren data te zoeken. De eerste manier om data te localiseren is het zoeken op een speciale identifier (CCHHR). De tweede manier is het zoeken naar een speciale key.

Gaps.

De gaps hebben de volgende functies:

a) ze scheiden de opeenvolgende blocken en subblocken.
b) ze geven de controller tijd om zijn taak uit te voeren.
c) ze zorgen voor de synchronisatie van de controller met de data op de disc.

We kunnen drie soorten gaps onderscheiden:

Gap 1: Dit is de gap tussen de index en het home address. Hij heeft een vaste lengte van 73 bytes (HA gap).

Gap 2: De gap tussen het einde van het home address en het R0 countsubblock (45 bytes) en ook de gap tussen de subblocken (43 bytes).

Gap 3: De gap tussen het einde van een datasubblock van het ene record en het begin van het countsubblock van de volgende record. Deze gap heeft een variabele lengte van 45 bytes plus 4,3 procent van de som van de key- en datalengte van de vorige record.

Wanneer we een meer gedetailleerde beschrijving van de inhoud van een gap bekijken (zie figuur 4.7) dan zien we dat deze uit vijf gedeelten bestaat.
De lead byte volgt direct op de check bytes en deze geeft aan

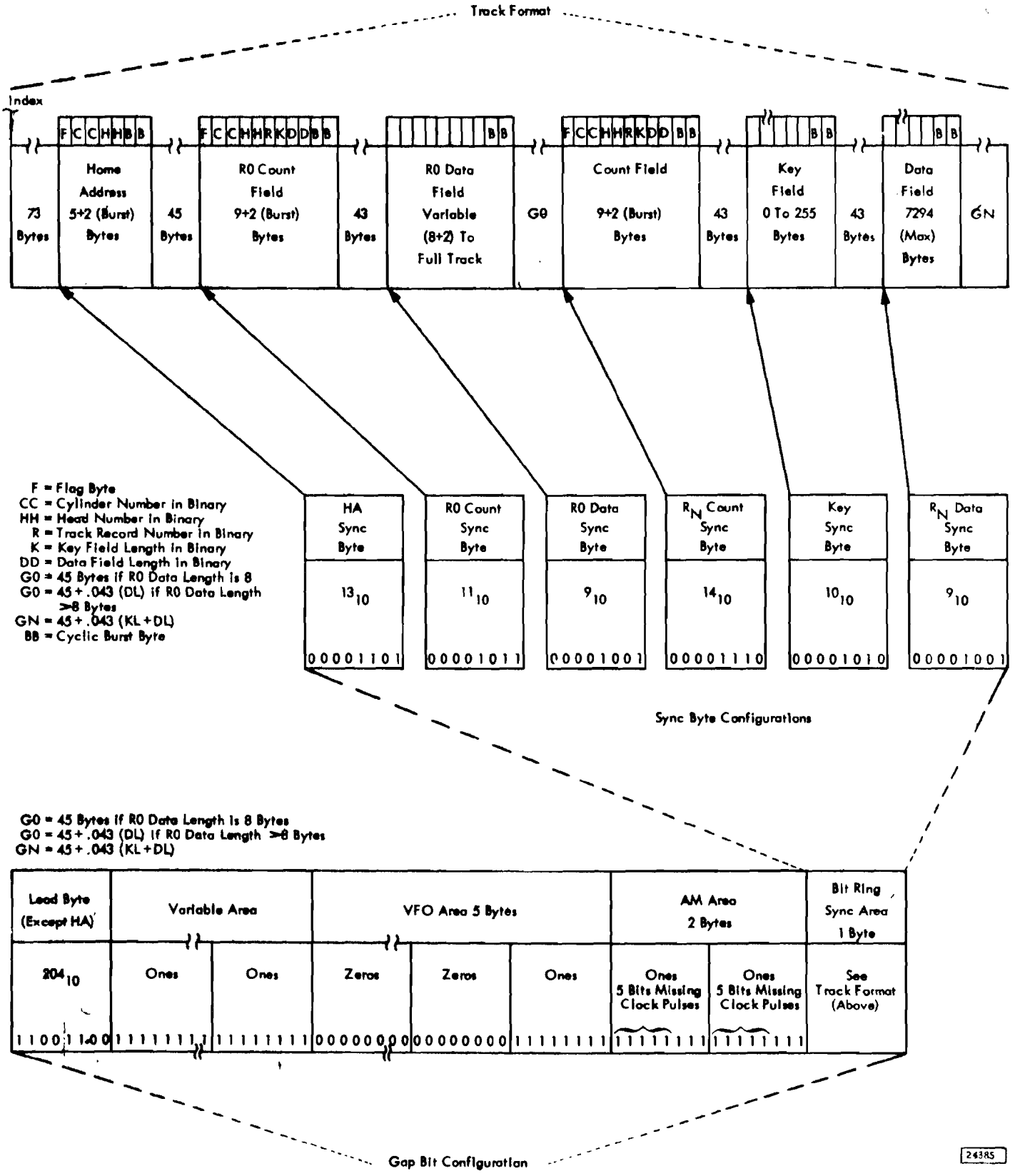
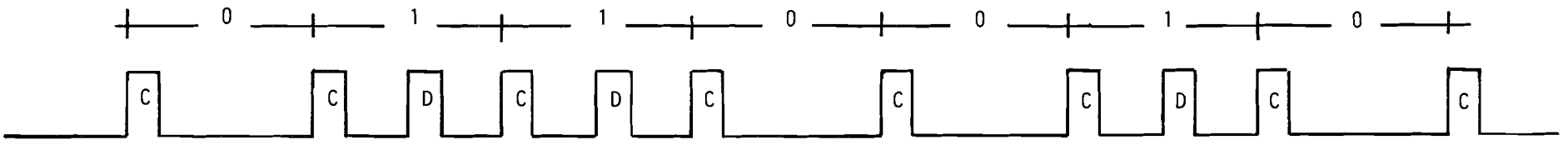


fig.4.7. Track layout en gap configuratie.

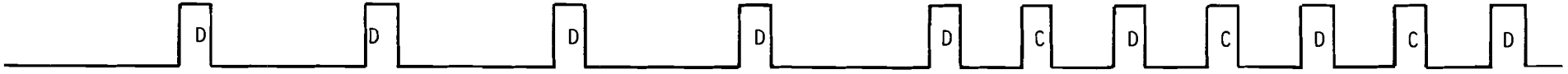
dat een gap begint. Het lead area geeft de controller tijd om zijn taak uit te voeren. Het VFO area zorgt voor de bit-synchronisatie. Vervolgens komen er twee address marker bytes. Deze worden bij de 2314 gebruikt om het begin van een sub-block te localiseren voor search, write en read operaties, zodat men in staat is ergens op de track met een operatie te starten zonder te hoeven wachten op de index.
Het datasignaal, wat op de disc geschreven wordt, wordt om-gezet van parallel naar serie en daarna wordt er een klok-signaal tussendoor gemengd (zie figuur 4.8).
Wanneer we de address marker vergelijken met een normaal datasignaal dan zien we dat bij de address marker gedurende vijf datapulsen de klokpulsen ontbreken.
De sync byte dient voor de woordsynchronisatie, zodat we het begin van een databyte aan kunnen geven.

Foutendetectie.

Om bij de serie-data, die wordt opgeslagen op de disc de betrouwbaarheid van het lezen of schrijven te controleren worden aan het einde van elk block twee bytes toegevoegd: de cyclic code check bytes.
In figuur 4.9 is het principe van de cyclic code check aangegeven. De control unit van de 2314 vult eerst twee één byte-registers met enen. Elk character, dat gelezen of geschreven wordt, wordt exclusive-ored met een van de twee registers. Alle even genummerde characters gaan naar het ene register en alle oneven genummerde naar het andere register.
Bij de schrijfoperatie worden deze twee cyclic code bytes toegevoegd aan het einde van de record en worden ze ge-schreven op de disc.
Bij een leesoperatie wordt het exclusive-or proces voort-gezet tot en met de cyclic code bytes. Als er geen fout is



Data



Address marker

fig.4.8. Vergelijking van de address marker met normale data.

Example: Write	
	Register 1
Set to 1's	1 1 1 1 1 1 1 1
1st Character	1 1 1 1 0 0 0 1
Result	0 0 0 0 1 1 1 0
3rd Character	1 1 1 1 0 0 1 1
Result	1 1 1 1 1 1 0 1
	Code Check byte one
	Register 2
Set to 1's	1 1 1 1 1 1 1 1
2nd Character	1 1 1 1 0 0 1 0
Result	0 0 0 0 1 1 0 1
4th Character	1 1 1 1 0 1 0 0
Result	1 1 1 1 1 0 0 1
	Code Check byte two
Example: Read	
	Register 1
Set to 1's	1 1 1 1 1 1 1 1
1st Character	1 1 1 1 0 0 0 1
Result	0 0 0 0 1 1 1 0
3rd Character	1 1 1 1 0 0 1 1
Result	1 1 1 1 1 1 0 1
Code Check 1	1 1 1 1 1 1 0 1
Result	0 0 0 0 0 0 0 0
	Register 2
Set to 1's	1 1 1 1 1 1 1 1
2nd Character	1 1 1 1 0 0 1 0
Result	0 0 0 0 1 1 0 1
4th Character	1 1 1 1 0 1 0 0
Result	1 1 1 1 1 0 0 1
Code Check 2	1 1 1 1 1 0 0 1
Result	0 0 0 0 0 0 0 0

24375A

fig.4.9. Code check byte generatie.

opgetreden is het resultaat in de check-registers nul.
De cyclic code check bytes worden burst bytes genoemd.
De cyclic code check, die in de 2314 gebruikt wordt, heeft de volgende eigenschappen:

1) hij detecteert alle fouten waarbij een oneven aantal bits fout is.
2) hij detecteert een burst van fouten, die zestien bits of kleiner is in lengte.(Een burst met lengte b is een willekeurig foutenpatroon waarbij het aantal bits van de eerste tot de laatste foute bit gelijk is aan b)

4.5. De IBM 2314.

De IBM 2314 bestaat uit negen disk drives en een control unit. De drives zijn onafhankelijk van elkaar en acht drives kunnen tegelijkertijd "on line" zijn. Een van de drives fungeert als reserve. De operator kan zelf m.b.v. de nummer-plug aan de voorkant van de disk het adres van de disk drive module bepalen en zo aangeven welke drive als reserve dienst doet.

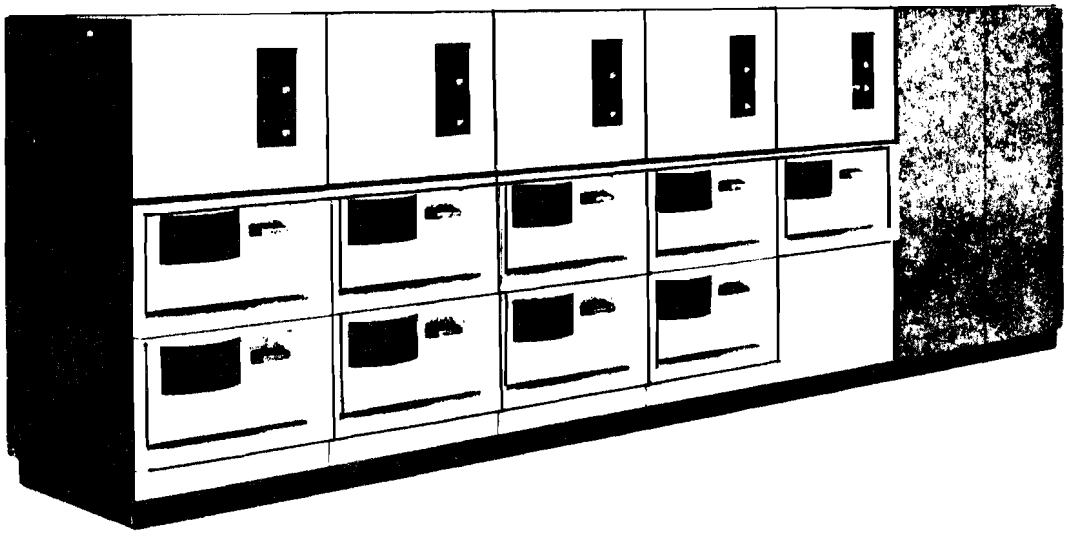
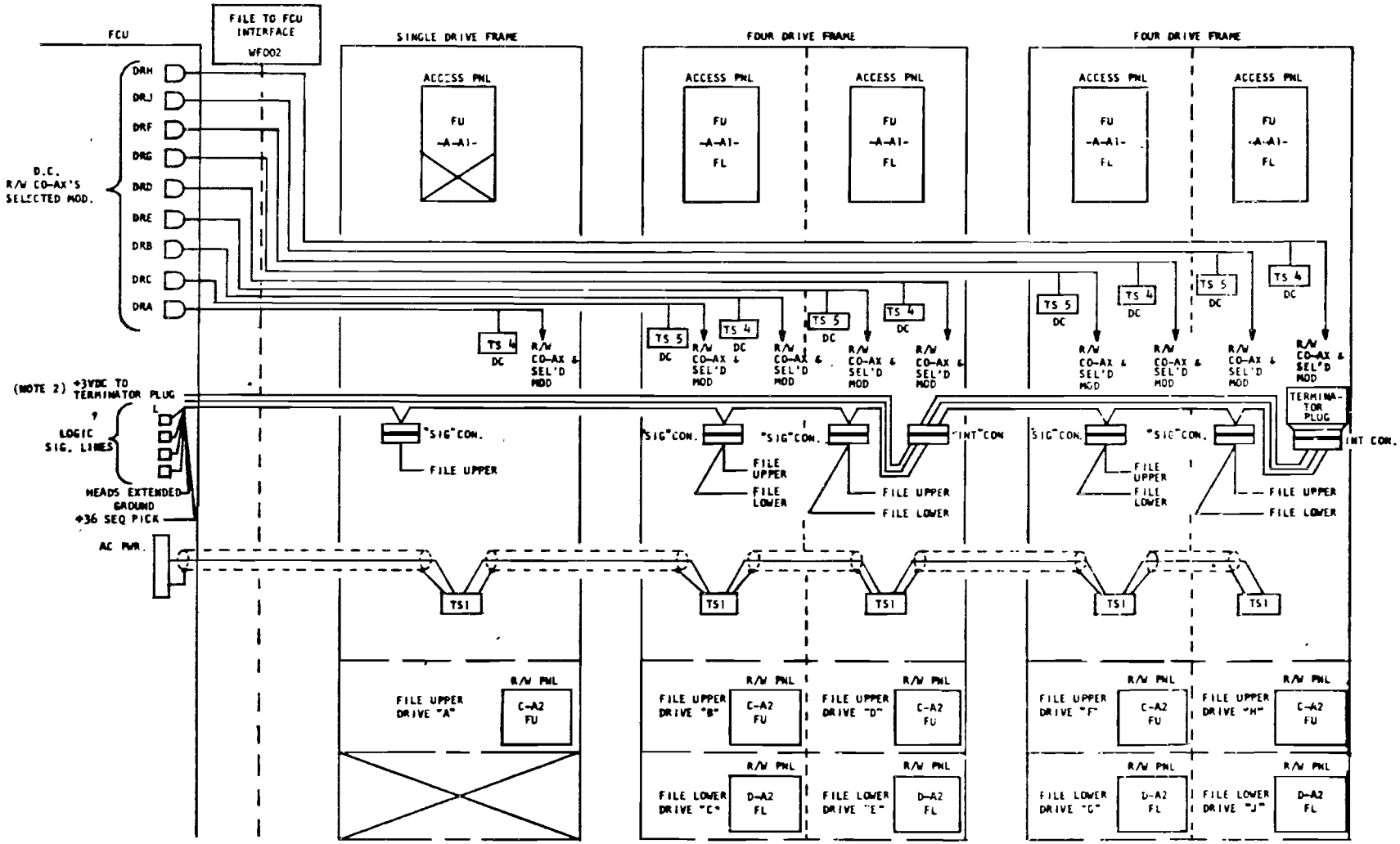


fig.4.10. De IBM 2314 Direct Access Storage Facility.

Omdat de records een variabele lengte hebben varieert de totale capaciteit van een disk. De maximum capaciteit is 29,2 miljoen bytes.
De schematische opbouw en de kabel layout van de 2314 vindt U in figuur 4.11. We kunnen,zoals uit het figuur blijkt, hierbij drie soorten verbindingen onderscheiden:

1) De power lijnen (AC)
2) De logische signaallijnen
3) De DC-lijnen en de read/write coax

In figuur 4.12 vindt U de aansluitgegevens van de logische signaallijnen op de disk drive module. Deze lijnen worden



NOTES:

1. ONE SET OF LOGICS ARE PROVIDED FOR ONE FILE UPPER AND ONE FILE LOWER BUT ARE APPLICABLE TO THE ENTIRE FILE.
2. +3VDC IS CARRIED IN THE SIGNAL CABLE THRU THE "INT" CONNECTORS TO THE TERMINATOR PLUG ONLY.

DATE	EC NUMBER	DATE	EC NUMBER	2314 FILE DRIVE & INTERFACE
APR66	416034			CABLE LAYOUT
				DATE P/N 2218331
				TYPE
				IBM WF001

fig.4.11. Kabel layout van de 2314.

FCU TO FILE TO FILE

FROM WF011 LINE TITLE	FCU LOGIC PAGE		SIG. CONNECTOR PIN	TWISTED WIRE	FILE LOGIC PAGES	FILE FD'S	FILE LINE TITLE
FILE BUS 0	WF011		SIG - A	SIG - B	FL/FU020	FD101	-2.5* +1.5* FILE BUS 0
FILE BUS 1	WF011		SIG - C	SIG - D	FL/FU020	FD101	-2.5* +1.5* FILE BUS 1
FILE BUS 2	WF011		SIG - E	SIG - F	FL/FU020	FD101	-2.5* +1.5* FILE BUS 2
FILE BUS 3	WF011		SIG - H	SIG - J	FL/FU020	FD101	-2.5* +1.5* FILE BUS 3
FILE BUS 4	WF011		SIG - K	SIG - L	FL/FU020	FD101	-2.5* +1.5* FILE BUS 4
FILE BUS 5	WF011		SIG - M	SIG - N	FL/FU021	FD101	-2.5* +1.5* FILE BUS 5
FILE BUS 6	WF011		SIG - P	SIG - R	FL/FU021	FD101	-2.5* +1.5* FILE BUS 6
FILE BUS 7	WF011		SIG - S	SIG - T	FL/FU021	FD101	-2.5* +1.5* FILE BUS 7
SET DIFFERENCE	WF011		SIG - U	SIG - V	FU/FL021	FD101	-2.5* +1.5* SET DIFFERENCE
SET CYLINDER	WF011		SIG - W	SIG - X	FU/FL021	FD101	-2.5* +1.5* SET CYLINDER
SET HEAD	WF011		SIG - Y	SIG - Z	FU/FL021	FD101	-2.5* +1.5* SET HD + DIRECTION
CONTROL	WF011		SIG - a	SIG - b	FU/FL021	FD101	-2.5* +1.5* CONTROL
MOD 0 SELECT	WF011		SIG - c	SIG - d	YB001		MOD 0 SELECT
MOD 1 SELECT	WF011		SIG - f	SIG - g	YB001		MOD 1 SELECT
MOD 2 SELECT	WF011		SIG - h	SIG - i	YB001		MOD 2 SELECT
MOD 3 SELECT	WF011		SIG - j	SIG - k	YB001		MOD 3 SELECT
MOD 4 SELECT	WF011		SIG - m	SIG - n	YB001		MOD 4 SELECT
MOD 5 SELECT	WF011		SIG - p	SIG - q	YB001		MOD 5 SELECT
MOD 6 SELECT	WF011		SIG - r	SIG - s	YB001		MOD 6 SELECT
MOD 7 SELECT	WF011		SIG - t	SIG - u	YB001		MOD 7 SELECT
SPARE MOD SELECT	WF011		SIG - v	SIG - w	YB001		SPARE MOD SELECT
CONTROLLED GND FROM FCU	WF011		SIG - CK	- -	YB001		CONTROLLED GND FROM FCU
SEQUENCE PICK IN	WF011	NOTE 3	SIG - CL	- -	YB001		SEQ. PICK (INCOMING) NOTE 3
+3 FILE TERMINATOR	WF011		INT - AT	- -	WF001		+3V DC } NOTE 2
			INT - AU	- -			+3V DC }
AC POWER	WF011		TS 1	- -	YA001		208V AC - 3 PHASE
+36	WF011	NOTE 4	TS 4/5	- -	YB001		+36
-36	WF011	NOTE 4	TS 4/5	- -	YB001		-36
+6	WF011	NOTE 4	TS 4/5	- -	YB001		+6
+3	WF011	NOTE 4	TS 4/5	- -	YB001		+3
-3	WF011	NOTE 4	TS 4/5	- -	YB001		-3
GND	WF011	NOTE 4	TS 4/5	- -	YB001		GND
-3 PWR ON RESET	WF011		SIG - BP		FU/FL090	FD103	-3 PWR ON RESET + C.E. RESET SELECT LOCK
READ/WRITE COAX	WF011	NOTE 4	R/W BOARD		YA001	FD102	READ/WRITE COAX
SELECTED MODULE	WF011	NOTE 4	EDGE CONN		YB001	FD101	SELECTED MODULE
FILE TO FILE TO FCU							
GATED ATTENTION 0	WF011		SIG - x	SIG - y	YB001		GATED ATTENTION 0
GATED ATTENTION 1	WF011		SIG - z	SIG - AA	YB001		GATED ATTENTION 1
GATED ATTENTION 2	WF011		SIG - AB	SIG - AC	YB001		GATED ATTENTION 2
GATED ATTENTION 3	WF011		SIG - AD	SIG - AE	YB001		GATED ATTENTION 3
GATED ATTENTION 4	WF011		SIG - AF	SIG - AH	YB001		GATED ATTENTION 4
GATED ATTENTION 5	WF011		SIG - AJ	SIG - AK	YB001		GATED ATTENTION 5
GATED ATTENTION 6	WF011		SIG - AL	SIG - AM	YB001		GATED ATTENTION 6
GATED ATTENTION 7	WF011		SIG - AN	SIG - AP	YB001		GATED ATTENTION 7
GATED ATTENTION SPARE MOD	WF011		SIG - AR	SIG - AS	YB001		GATED ATTENTION SPARE MOD
NOTE 6				SIG - AW			
CYLINDER ADDR REG 1	WF011		SIG - AR	SIG - AY	FU/FL026	FD101	-2.5* +1.5* CYL ADDR REG 1
CYLINDER ADDR REG 2	WF011		SIG - AZ	SIG - BA	FU/FL026	FD101	-2.5* +1.5* CYL ADDR REG 2
CYLINDER ADDR REG 4	WF011		SIG - BB	SIG - BC	FU/FL026	FD101	-2.5* +1.5* CYL ADDR REG 4
CYLINDER ADDR REG 8	WF011		SIG - BD	SIG - BE	FU/FL026	FD101	-2.5* +1.5* CYL ADDR REG 8
CYLINDER ADDR REG 16	WF011		SIG - BF	SIG - BH	FU/FL026	FD101	-2.5* +1.5* CYL ADDR REG 16
CYLINDER ADDR REG 32	WF011		SIG - BJ	SIG - BK	FU/FL026	FD101	-2.5* +1.5* CYL ADDR REG 32
CYLINDER ADDR REG 64	WF011		SIG - BL	SIG - BM	FU/FL026	FD101	-2.5* +1.5* CYL ADDR REG 64
CYLINDER ADDR REG 128	WF011		SIG - BN		FU/FL026	FD101	-2.5* +1.5* CYL ADDR REG 128
SELECTED FILE BUSY	WF011		SIG - BR	SIG - BS	FU/FL025	FD104	-2.5* +1.5* SEL FILE BUSY
SELECTED ON LINE	WF011		SIG - BT	SIG - BU	FU/FL025	FD104	-2.5* +1.5* SEL ON LINE
SELECTED INDEX	WF011		SIG - BV	SIG - BW	FU/FL025	FD104	-2.5* +1.5* SEL INDEX
FILE UNSAFE	WF011		SIG - BX	SIG - BY	FU/FL025	FD104	-2.5* +1.5* FILE UNSAFE
SELECTED SEEK INCOMP	WF011		SIG - BZ	SIG - CA	FU/FL025	FD104	-2.5* +1.5* SEL SK INCOMP
SELECTED END OF CYL	WF011		SIG - CB	SIG - CC	FU/FL025	FD104	-2.5* +1.5* SEL END OF CYL
SELECTED PACK CHANGE	WF011		SIG - CD	SIG - CE	FU/FL025	FD104	-2.5* +1.5* SEL PACK CHG
WRITE CURRENT SENSE	WF011		SIG - CF	SIG - CH	FU/FL025		-2.5* +1.5* WR CURRENT SNS
HEADS EXTENDED	WF011		SIG - CJ	- -	YB001		HEADS EXTENDED
			SIG - CM		YB001		SEQUENCE PICK (OUTGOING)
							(TO NEXT FILE FRAME)
NOTE 6			SIG - CS	- -			
* -3V	WF011				YB003		-3V
* +6V	WF011				YB003		+6V
* -3V	WF011	NOT USED			YB003		-3V
* -ENABLE A	WF011	WITH REMOTE			YB003		-ENABLE A
* -DISABLE A	WF011	ATTACHMENT			YB003		-DISABLE A
* -ENABLE B	WF011	SWITCH			YB003		-ENABLE B
* -DISABLE B	WF011	FEATURE			YB003		-DISABLE B
* +6 TO -3 MULTI-TAG SW	WF011				YB003		+6 TO -3 MULTI-TAG SW

DATE	EC NUMBER
APR 66	416034
JUN 66	416124
AUG 66	416128 REDRAW
DEC 66	420637
1APR68	422947

			DATE	P/N 2218332
				TYPE
			IBM	WF002 :
	EC NUMBER	DATE		
	420946	1AUG68		

NOTES:
1. FOR CABLE INTERFACE LAYOUT SEE LOGIC PAGE WF001
2. THE "INT" CONNECTOR CARRIES ALL THE SIGNALS THAT THE "SIG" CONNECTORS CARRY PLUS TWO +3V DC LINES FOR THE FILE TERMINATOR PLUG.
3. SEQUENCE PICK (INCOMING) MAY BE FROM THE FCU OR FROM "SIG-CM" - SEQUENCE PICK OUTGOING OF THE PRECEDING FILE FRAME.
4. THESE LINES OCCUR NINE (9) TIMES, ONCE TO EACH FILE DRIVE.
* 5. OPTIONAL FEATURE TWO CHANNEL SWITCH
6. JUMPERS WITH MALE PINS BETWEEN AW AND CS IN FEMALE CONNECTORS ARE NON-FUNCTIONAL - TO ENSURE CORRECT PLUGGING ONLY

fig.4.12. Aansluitgegevens van de logische signaallijnen.

in de drive aangesloten op de zogenaamde "sign"connector. Verder zijn in figuur 4.13a de aansluitingen van de DC lijnen en de read/write coax vermeld. Tevens vindt U in figuur 4.13b de aansluitgegevens van de PC 1 connector. Via deze connector worden een aantal terugmeldlijnen voor en van de CPU met de control unit verbonden. Om er voor te zorgen dat de juiste relais in de control unit bekrachtigd worden moet PC 1-1 verbonden worden met PC 1-2 en PC 1-6. Voor de relaisschema's verwijs ik naar de bladzijden YA 007 en YA 021 in IBM Field Engineering 2314 (73-10554) ALD (NS-ZZ vol.A4).
De 2314 kan aangesloten worden op een IBM 360 systeem. Het 360 systeem gebruikt een standaard I/O interface voor controle en communicatie met alle aangesloten I/O devices (zie figuur 4.14).

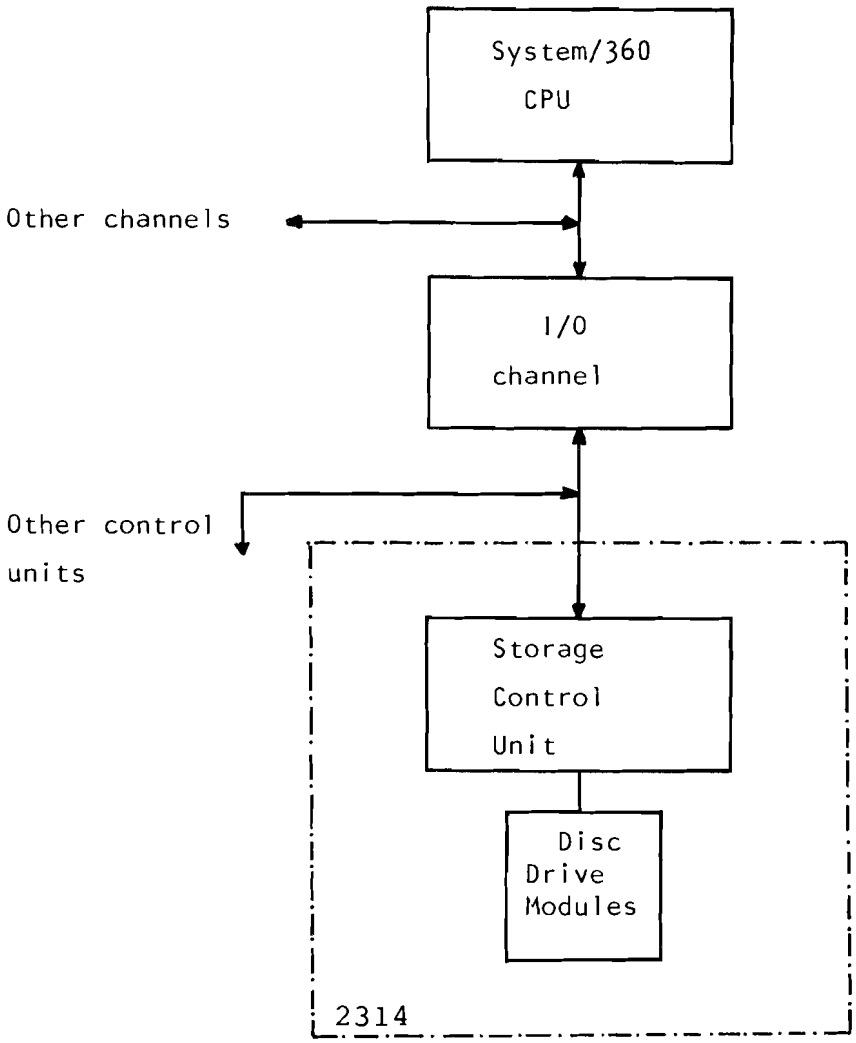
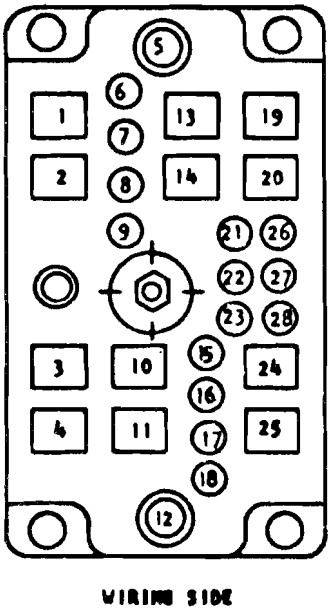


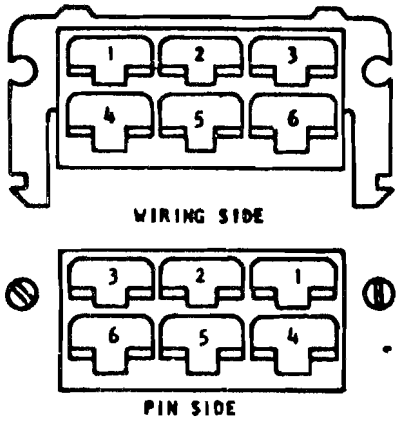
fig.4.14. IBM systeem met de 2314.



FILE DC POWER

	PIN	SIGNAL	PAGE	PIN	SIGNAL	PAGE	
	1	+6VDC	YA061	15	-36VDC	YA051	
	2	-3VDC	YA061	16			
	3	DC GND	YA061	17			
	4	DC GND	YA061	18			
	5			19	+36 VDC	YA061	
NOTE 1	6	SEQ PK - IN		20	+36 VDC	YA061	
NOTE 1	7	HEAD EXTD		21	PWR ON RESET	WF012	NOTE 1
NOTE 1	8	CNTLD GND		22	SELECTED MOD	MA041	NOTE 2
NOTE 1	9	SEQ PK - OUT		23			
	10	DC GND	YA051	24			
	11	DC GND	YA051	25	FRAME GND		
	12	READ/WRITE DATA	NS031 NS032	26			
	13	+3VDC	YA051	27			
	14	+3VDC	YA051	28			

fig.4.13a. Aansluitingen van de DC-connector.



PC1 CONNECTOR - CHANNEL

PIN	SIGNAL	PAGE	PIN	SIGNAL	PAGE
1	24 VAC CPU	YA007	4	PWR COMPLETE OUT	YA031
2	EPO CTRL CPU	YA021	5	PWR CONT. HOLD	YA021
3	PWR COMPLETE IN	YA031	6	PWR CONT. PICK	YA021

fig.4.13b. De PC-1 connector.

Om een I/O operatie te beginnen voert de CPU een I/O instructie uit. Vervolgens wordt een commando woord uit het geheugen gehaald en naar het kanaal gezonden. Terwijl de geselecteerde control unit bezig is met het uitvoeren van dit commando kan de CPU zelf verdergaan met hetgeen waarmee hij voor de I/O operatie bezig was.
De File Control Unit (FCU), ook wel Storage Control Unit (SCU) genoemd, interpreteert deze commando's en zorgt er voor dat één van de vijf opdrachten uitgevoerd wordt.
Deze opdrachten zijn:

1) Seek : Om de lees/schrijfkoppen op de juiste plaats te positioneren bepaalt de FCU de huidige positie van het toegangsmechanisme, de richting en de afstand waarover de arm verplaatst moet worden. Verder zorgt hij voor de selectie van de juiste lees/schrijfkop om de operatie te voltooien.
2) Search: Om directe toegang tot de informatie te krijgen wordt door de FCU die informatie gelezen, die nodig is om de plaats van een record of een gedeelte van een record te bepalen.
3) Write : De data wordt in serie op de disk geschreven. Voordat de data uit het geheugen op de disk gezet wordt moet eerst informatie over het formaat op de disk geschreven worden. Om na te kunnen gaan of de data goed weggeschreven is worden check bytes aan de data toegevoegd.
4) Read : De data, die gelezen wordt, wordt door de FCU van serie naar parallel omgezet. Daarbij wordt m.b.v. de check bytes gecontroleerd of bij het lezen geen fouten gemaakt zijn.
5) Sense : Bij deze opdracht wordt de status informatie van de module en de control unit doorgegeven aan de CPU.

COMMAND		COMMAND CODE: Multiple Track Off Decimal	Multiple Track Off Hexadecimal	Multiple Track Off Binary	Multiple Track On (If Applicable) Decimal	Multiple Track On (If Applicable) Hexadecimal	Multiple Track On (If Applicable) Binary	DATA ADDRESS	COUNT
Control	No Op	03	03	0000 0011				X	X
	Release*	23	17	0001 0111				X	X
	Reserve*	180	B4	1011 0100				X	X
	Recalibrate	19	13	0001 0011				X	X
	Seek	07	07	0000 0111				CPU storage location of seek address	6
	Seek Cylinder	11	0B	0000 1011					6
	Seek Head	27	1B	0001 1011					6
	Sense	04	04	0000 0100				CPU storage location to which 6 sense bytes are sent.	6
	Set File Mask	31	1F	0001 1111				CPU storage location of mask byte.	1
	Space Count	15	0F	0000 1111				X	X
	Transfer In Channel	X8	X8	XXXX 1000				CPU storage location of next CCW (Must be divisible by 8.)	X
Search	Home Address Equal	57	39	0011 1001	185	B9	1011 1001	CPU storage location of search argument	4 (usually)
	Identifier Equal	49	31	0011 0001	177	B1	1011 0001		5 (usually)
	Identifier High	81	51	0101 0001	209	D1	1101 0001		5 (usually)
	Identifier Equal or High	113	71	0111 0001	241	F1	1111 0001		5 (usually)
	Key Equal	41	29	0010 1001	169	A9	1010 1001		From 1 to 255
	Key High	73	49	0100 1001	201	C9	1100 1001		From 1 to 255
	Key Equal or High	105	69	0110 1001	233	E9	1110 1001		From 1 to 255
	Key and Data Equal	45	2D	0010 1101	173	AD	1010 1101		Number of bytes (Including mask bytes) in search argument
	Key and Data High	77	4D	0100 1101	205	CD	1100 1101		
	Key and Data Equal or High	109	6D	0110 1101	237	ED	1110 1101		
Read	Home Address	26	1A	0001 1010	154	9A	1001 1010	CPU storage location to which areas read will be transferred	5
	Count	18	12	0001 0010	146	92	1001 0010		8
	Record R0	22	16	0001 0110	150	96	1001 0110		Number of bytes to be transferred
	Data	06	06	0000 0110	134	86	1000 0110		Number of bytes to be transferred
	Key and Data	14	0E	0000 1110	142	8E	1000 1110		Number of bytes to be transferred
	Count, Key and Data	30	1E	0001 1110	158	9E	1001 1110		Number of bytes to be transferred
	IPL	02	02	0000 0010					Number of bytes to be transferred
Continue Scan	Search Equal	37	25	0010 0101	165	A5	1010 0101	CPU storage location of search argument	Number of bytes in search argument
	Search High	69	45	0100 0101	197	C5	1100 0101		
	Search High or Equal	101	65	0110 0101	229	E5	1110 0101		
	Set Status Modifier	53	35	0011 0101	181	B5	1011 0101		
	Do not Set Status Modifier	85	55	0101 0101	213	D5	1101 0101		
	Set Status Modifier	117	75	0111 0101	245	F5	1111 0101		
Write	Home Address	25	19	0001 1001				CPU storage location from which areas to be written will be transferred	5 (usually)
	Record R0	21	15	0001 0101					8+Key Length + Data Length of Record R0
	Erase	17	11	0001 0001					8 + Key Length + Data Length
	Count, Key and Data	29	1D	0001 1101					8+Key Length + Data Length
	Special Count Key and Data	01	01	0000 0001					8+Key Length + Data Length
	Data	05	05	0000 0101					Data Length
	Key and Data	13	0D	0000 1101					Key Length + Data Length

* Two channel switch special feature or 2314/2844 configuration only.
X Not significant.

11539

fig.4.15. Channel commando's.

In figuur 4.15 vindt U een overzicht van de commando's, die aan de control unit gegeven kunnen worden.

4.6. De registratiemethode.

De registratiemethode, die bij de IBM 2314 wordt toegepast is de Non Return to Zero (NRZ)-methode. Bij deze methode verandert de stroom alleen van richting als er een verandering in signaalinformatie is (zie figuur 4.16).

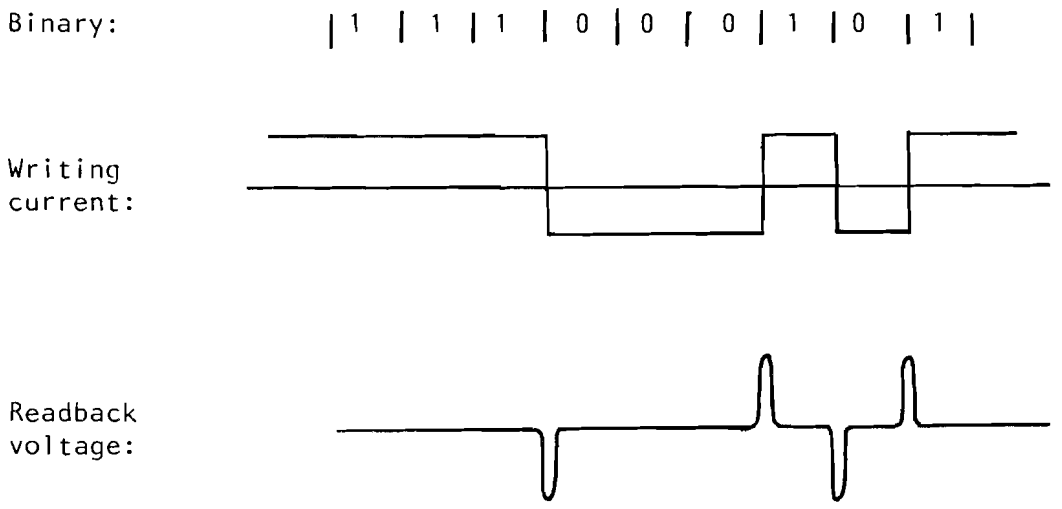


fig.4.16. De Non Return to Zero-methode.

Voor een serie nullen of enen wordt dus een lang gebiedje op de disk gemagnetiseerd. Bij het schrijven loopt de stroom in een vaste richting voor een "1" en in tegengestelde richting voor een "0". De NRZ-methode is niet self-clocking. Om toch een self-clocking systeem te krijgen, zodat we geen clock track nodig hebben, levert de control unit double frequency data. Dit houdt in dat aan het begin van elke bit-cell time een clock bit toegevoegd wordt. De frequentie van een logische "1" data is het dubbele van die

bij een logische "0" data. Bij het lezen wordt de data van de disk weer gescheiden in de clock en de originele data. Verder is het bij deze methode niet nodig dat de informatie eerst gewist wordt want bij het schrijven wordt de oude informatie overschreven. Het principe van double frequency is weergegeven in figuur 4.17.

4.7. De IBM 2314 control unit.

De control unit van de 2314 is een voorbeeld van een micro-programmeerbare I/O controller.
De IBM 2314 control unit is opgebouwd uit zes delen:

1) Het 360 systeem kanaal interface
2) De Arithmetic Logic Unit (ALU)
3) De serie/parallel en parallel/serie omzetter
4) Dertien registers die bij het besturen voor verschillende doeleinden gebruikt worden.
5) Read Only Memory (ROM) dat dient als control store.
6) De disk drive module interface

Het microprogramma, dat opgeslagen is in het ROM zorgt voor het uitvoeren van de commando's van de CPU.
De taken van de control unit kunnen in het kort samengevat worden: De control unit

a) decodeert het control unit selectie adres van het kanaal
b) decodeert het commando afkomstig van de CPU via het kanaal
c) stuurt status informatie naar de CPU
d) controleert de data flow, de mechanische operaties en zorgt voor het dataformaat.
e) checkt de data als die gelezen of geschreven is.

4.8. De disk drive module.

Elke module bevat een disk drive, een toegangsmechanisme en een read/write circuit.

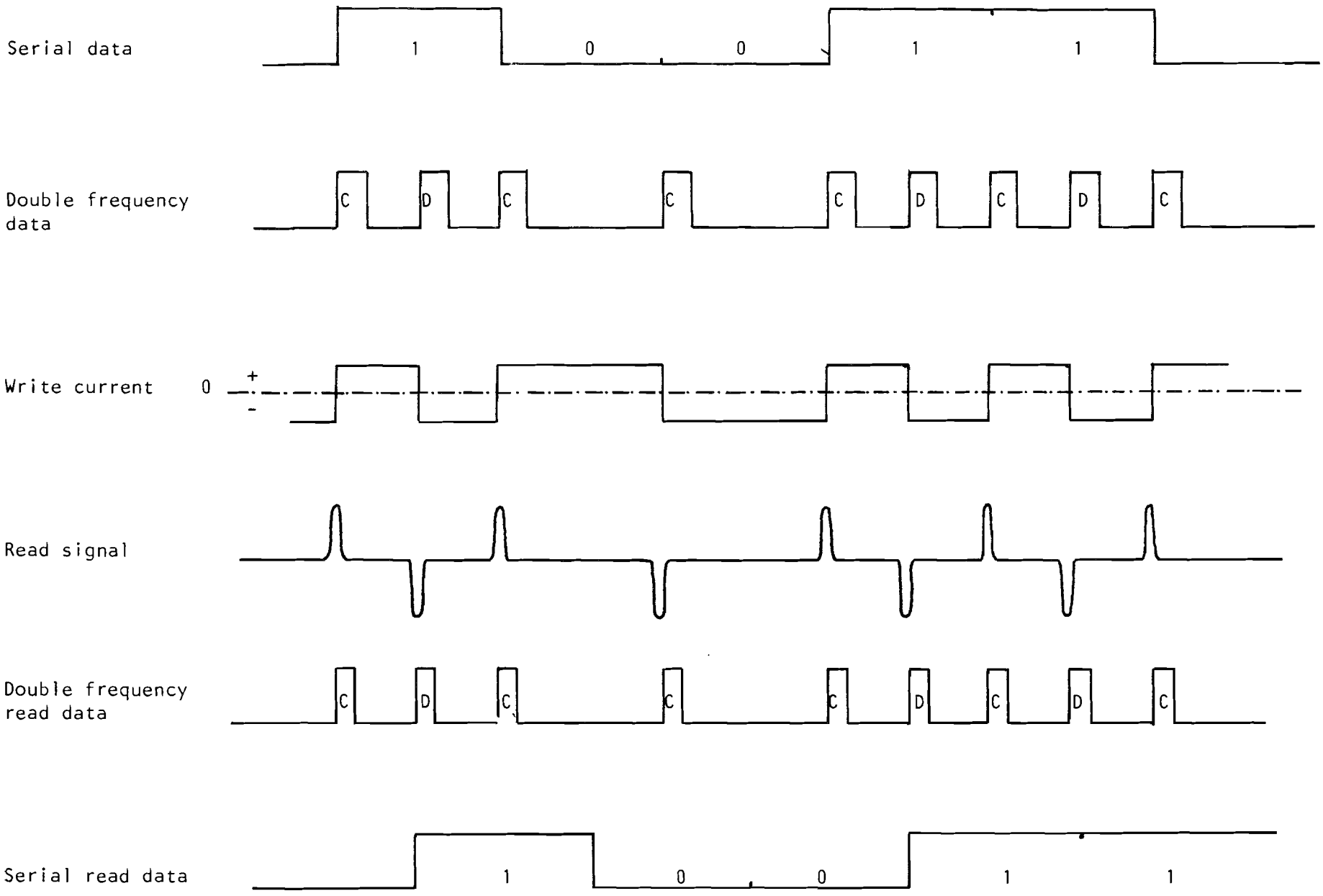


fig.4.17. Double Frequency Data.

De hoofdcomponenten van de module zijn:

1) De selectiecircuits: module, cylinder en headselectie
2) De drive motor en de hydraulische aandrijving
3) Het toegangsmechanisme en de controle daarvan
4) De read/write en erase heads en de daarbij behorende controle circuits
5) De module interface

Om de data te lezen of te schrijven wordt het toegangsmechanisme zodanig geplaatst dat de lees/schrijfkoppen zich op de juiste cylinder bevinden. De koppen worden dan dicht bij het diskoppervlak gebracht maar door de luchtstroom, die opgebouwd wordt door de draaiende disk zelf, wordt er voor gezorgd dat ze het oppervlak niet raken. Wanneer een van de koppen geselecteerd wordt wordt het read/write circuit geactiveerd om data op de gewenste track te lezen of te schrijven.
De disk drive module is met de control unit verbonden door input en output lijnen.
De input communicatie interface is samengesteld uit:

a) Acht time-shared adres bus lijnen
b) Vier tag lijnen
c) Een module select lijn
d) Read/write data lijn

Adres bus lijnen en tag lijnen zijn gemeenschappelijk voor alle 2314 disk drive modules. Een module select lijn, voor elke module één, "poort" de tag lijnen in de selectiecircuits van de corresponderende module. Dit is weergegeven in figuur 4.18.

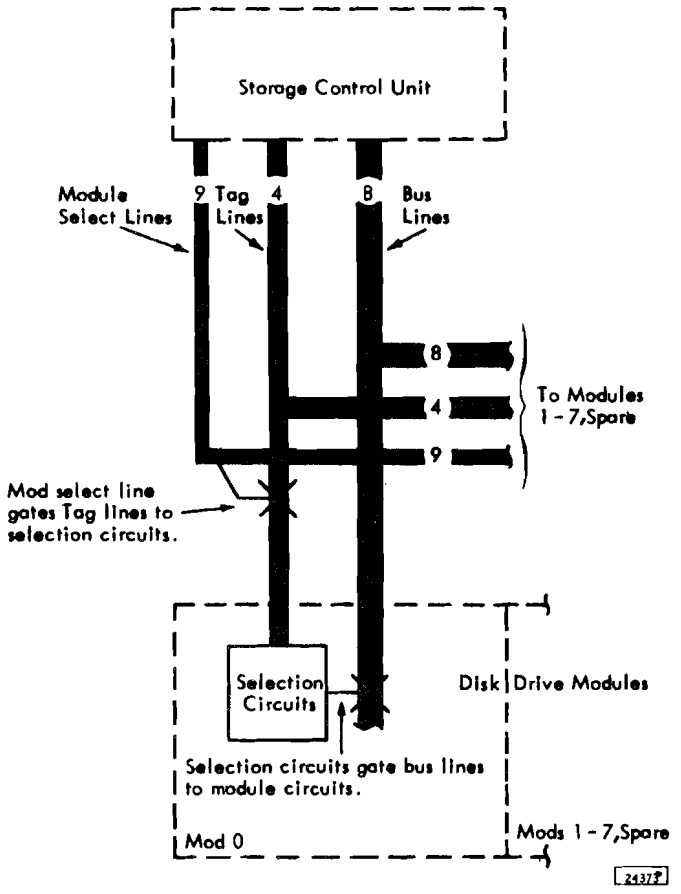


fig.4.18. De input interface lijnen.

De disk drive heeft achttien output lijnen waarvan acht lijnen het cylinderadres en tien lijnen de status van de drive aangeven.
De cylinderadres bus lijnen zijn afkomstig van het cylinder adres register. In dit register staat het adres van de cylinder waar de koppen zich bevinden (present address).
De tien status lijnen die van de module naar de FCU gaan zijn:

1) Gated Attention:

   Deze lijn geeft aan dat:
   1) pack change heeft plaatsgevonden
   2) seek voltooid is
   3) 600 msec zijn verstreken nadat het seek commando was ontvangen en geen "detent in" was gedetecteerd of te wel de juiste positie nog niet bereikt is

2) Module_Selected:

   Deze geeft aan dat de drive geselecteerd is.

3) Busy:

   Het toegangsmechanisme is in het seek proces.

4) Pack_Change:

   Pack change heeft plaatsgevonden.

5) On_Line:

   De koppen bevinden zich in de disk en zijn gereed om te lezen of te schrijven.

6) Index:

   Deze geeft het fysische begin van een track aan.

7) Unsafe:

   Bij unsafe kan geen read/write/erase of head select plaatsvinden.

8) Seek_Incomplete:

   Het toegangsmechanisme is er niet in geslaagd om binnen een bepaalde tijd de track te vinden.

9) End_of_Cylinder:

   Het head select register gaat van 19 naar 20.

10) Write_Current_Sense:

   De geselecteerde drive is aan het schrijven.

Deze in- en uitgaande lijnen zijn weergegeven in het disk drive functional block diagram (figuur 4.19).

4.9. De timing van de 2314.

Wanneer we de disk willen besturen zal de timing van de signalen van en naar de disk bekend moeten zijn.
Nadat een bepaalde module geselecteerd is krijgt deze informatie van de control unit aangeboden via de acht adres bus lijnen. De vier tag lijnen geven aan voor welk deel van de drive de informatie bedoeld is.

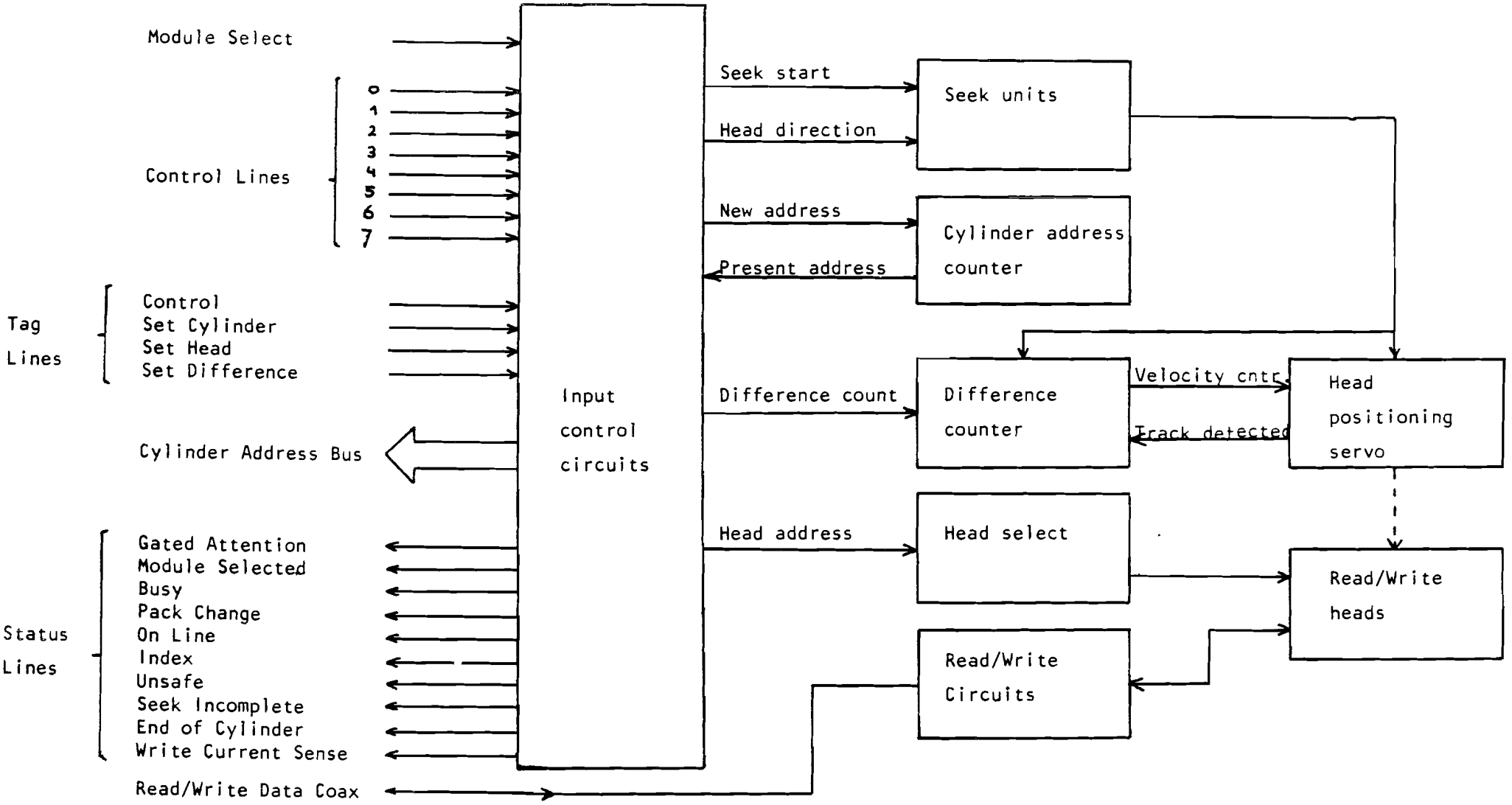


fig.4.19. Disc drive functional block diagram.

BUS	FUNCTIE			
	Control	Set Cylinder	Set Head	Set Difference
Bus 0	Write Gate	Cyl.128	Forward Latch	Not 128
Bus 1	Read Gate	Cyl. 64	Sel.Lock Reset	Not 64
Bus 2	Seek Start	Cyl. 32		Not 32
Bus 3	Reset Head Reg.	Cyl. 16	Head Adres 16	Not 16
Bus 4	Erase Gate	Cyl. 8	Head Adres 8	Not 8
Bus 5	Select Head	Cyl. 4	Head Adres 4	Not 4
Bus 6	Return to 000	Cyl. 2	Head Adres 2	Not 2
Bus 7	Head Advance	Cyl. 1	Head Adres 1	Not 1

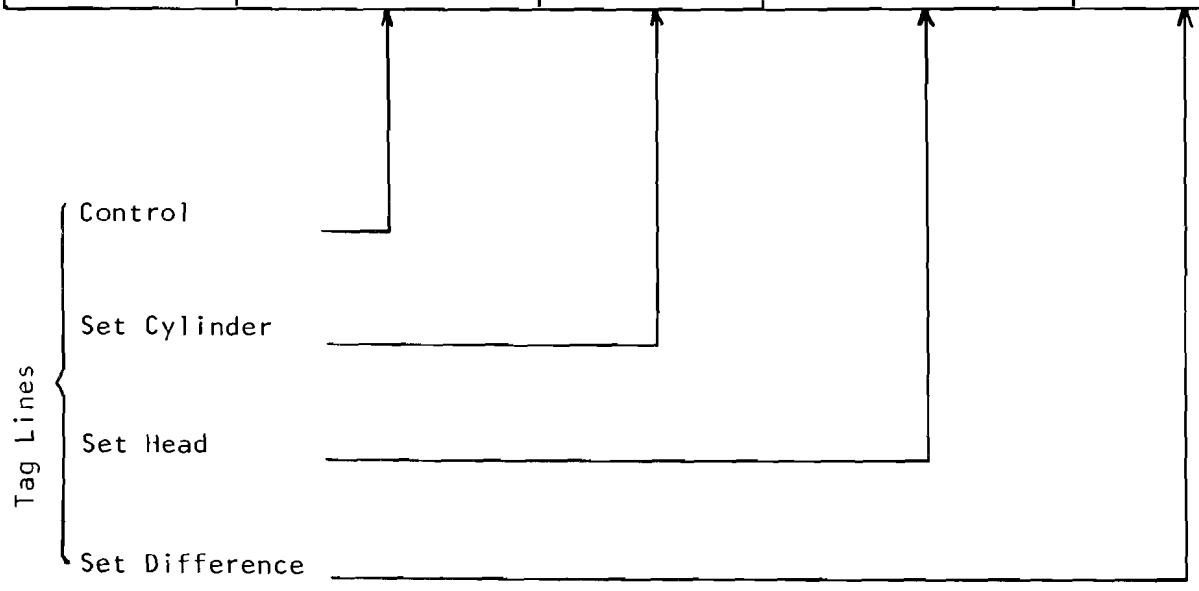


fig.4.20. Module input lijnen.

Slechts één van de vier tag lijnen is hoog om de informatie op de bus te beschrijven. Wanneer de control tag lijn hoog is worden de signalen op de bus herkend als control-cycle lijnen, zoals "write gate" en "read gate" enz. Bij de set-cylinder tag lijn worden de signalen op de bus geidentificeerd als het cylinderadres. Als de set-head tag lijn hoog is wordt m.b.v. de informatie op de bus de juiste head geselecteerd. Dit geldt ook voor de set-difference tag lijn, die de signalen op de bus identificeert als het berekende verschil tussen het oude en het nieuwe adres.
In figuur 4.20 is aangegeven hoe de informatie op de bus wordt geinterpreteerd afhankelijk van de tag lijn, die actief is.
De timing voor een seek, write of read opdracht vindt U in figuur 4.23,4.25,4.29 en 4.31. Hoe de timing bij deze opdrachten verloopt is verduidelijkt m.b.v. de flow charts, die U aantreft in figuur 4.26,4.27,4.30 en 4.32.
Om verwarring te voorkomen zijn de twee seek operaties toegelicht met figuur 4.22 en 4.24. Bij IBM wordt nl. met het tracknummer de head bedoeld, die binnen de cylinder geselecteerd moet worden.
Wanneer de controller een opdracht ontvangt moet,nadat de gewenste module geselecteerd is,de juiste informatie op de bus gezet worden vergezeld van de daarbij horende tag lijn.
De signalen op de file bus, de tag bus en de terugmeldlijnen zijn in tegenstelling tot wat in de figuren is aangegeven allemaal laag actief. Dit houdt verband met het IBM niveau, dat een gecombineerde stroom-spanningssturing is.
De breedte van de pulsen op de file bus bij de seek opdracht is 5 µsec. De tag-pulsen hebben een breedte van 1,5 µsec.
Wanneer we de koppen naar cylinder nul willen verplaatsen dan kan dat op twee manieren:

1) we bieden als nieuw cylinderadres adres 0 aan en doorlopen de timing zoals aangegeven in figuur 4.23 en 4.25.
2) we maken file bus 6 hoog en geven een control tag zodat de disk de opdracht return to zero krijgt. De breedte

van deze controle puls moet 15 msec bedragen. Dit is aangegeven in figuur 4.28.

Wanneer de seek operatie voltooid is wordt dit aangegeven door gated attention en seek complete (d.w.z. seek incomplete inactief). Willen wij opnieuw een seek operatie uitvoeren dan zullen we eerst gated attention moeten resetten. Dit gebeurt door middel van het aanbieden van read gate (bus 2 hoog en een control tag puls).
Bij het lezen en schrijven wordt de control tag gedurende de tijd, dat de kop geselecteerd is, hoog gehouden.
Bij een read operatie wordt eerst het home address en R0 gelezen. Na detectie van de gaps worden de records achtereenvolgens gelezen.
Voordat de data op de disk geschreven wordt, wordt eerst een gedeelte van de track gelezen, zoals het home address en R0. Vervolgens worden bus 0,1 en 5 hoog gemaakt (write gate, erase gate en head select). Nadat de data op de disk geschreven is wordt eerst write gate laag gemaakt. Een korte tijd daarna wordt erase gate laag gemaakt. Dit is noodzakelijk om de tunnel erase tussen de naburige tracks te voltooien. Nadat de informatie geschreven is passeert dit de erase heads, die de buitenste randen van de informatie verwijderen, hetgeen getoond wordt in figuur 4.33. Dit wordt tunnel erase genoemd.

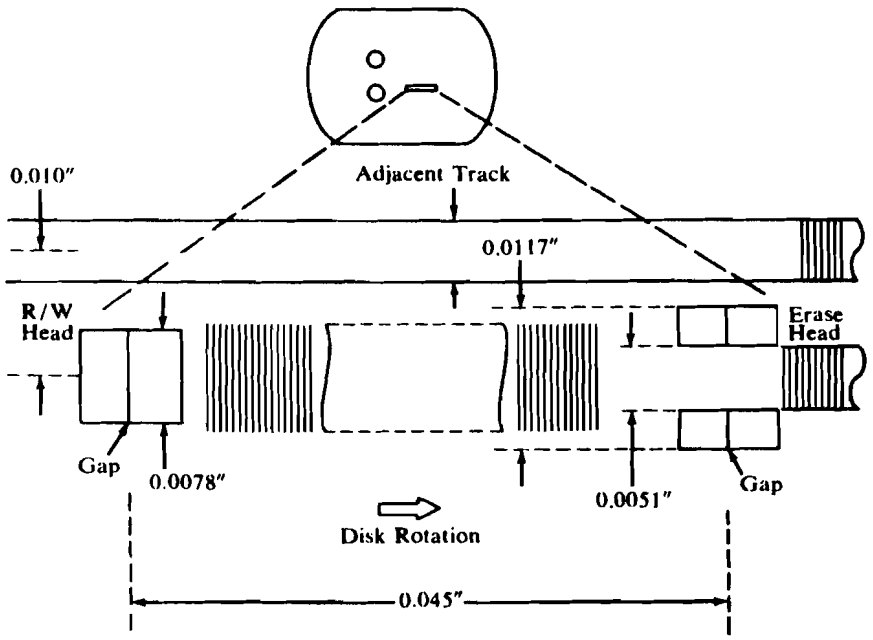


fig.4.33. Tunnel erase.

De breedte van de track wordt op deze manier gereduceerd. Wanneer dit principe niet toegepast zou worden zou de kop door een kleine afwijking bij het positioneren maar een gedeelte van de track zien bij het lezen. Dit houdt in dat de signaalsterkte minder wordt. Als de breedte van de track dus kleiner is hebben kleine afwijkingen bij het positioneren geen enkel effect omdat de kop de hele track ziet en dus een signaal van constante sterkte produceert.
In figuur 4.34 is een tabel gegeven, waarin de belangrijkste technische gegevens vermeld zijn.

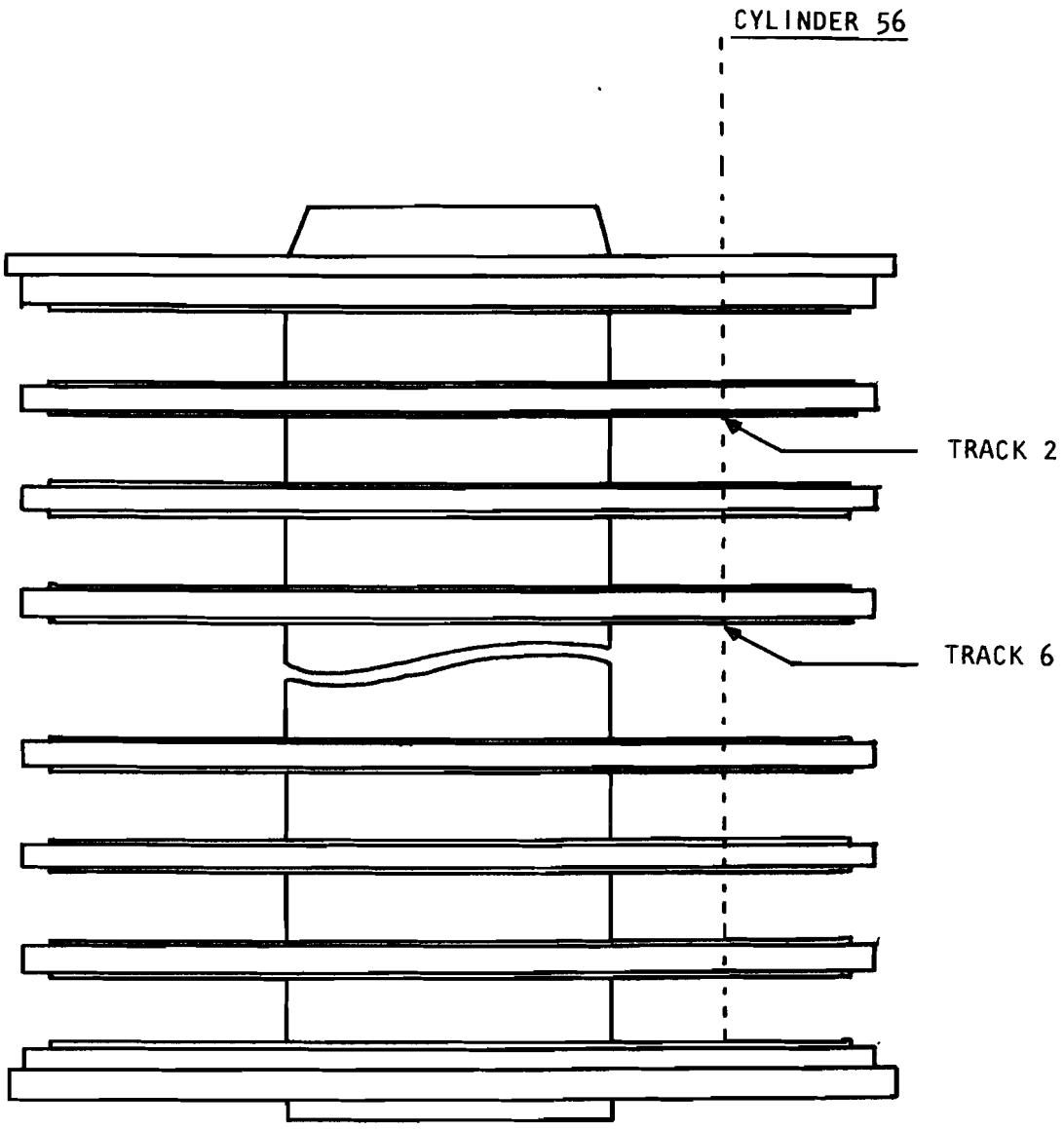
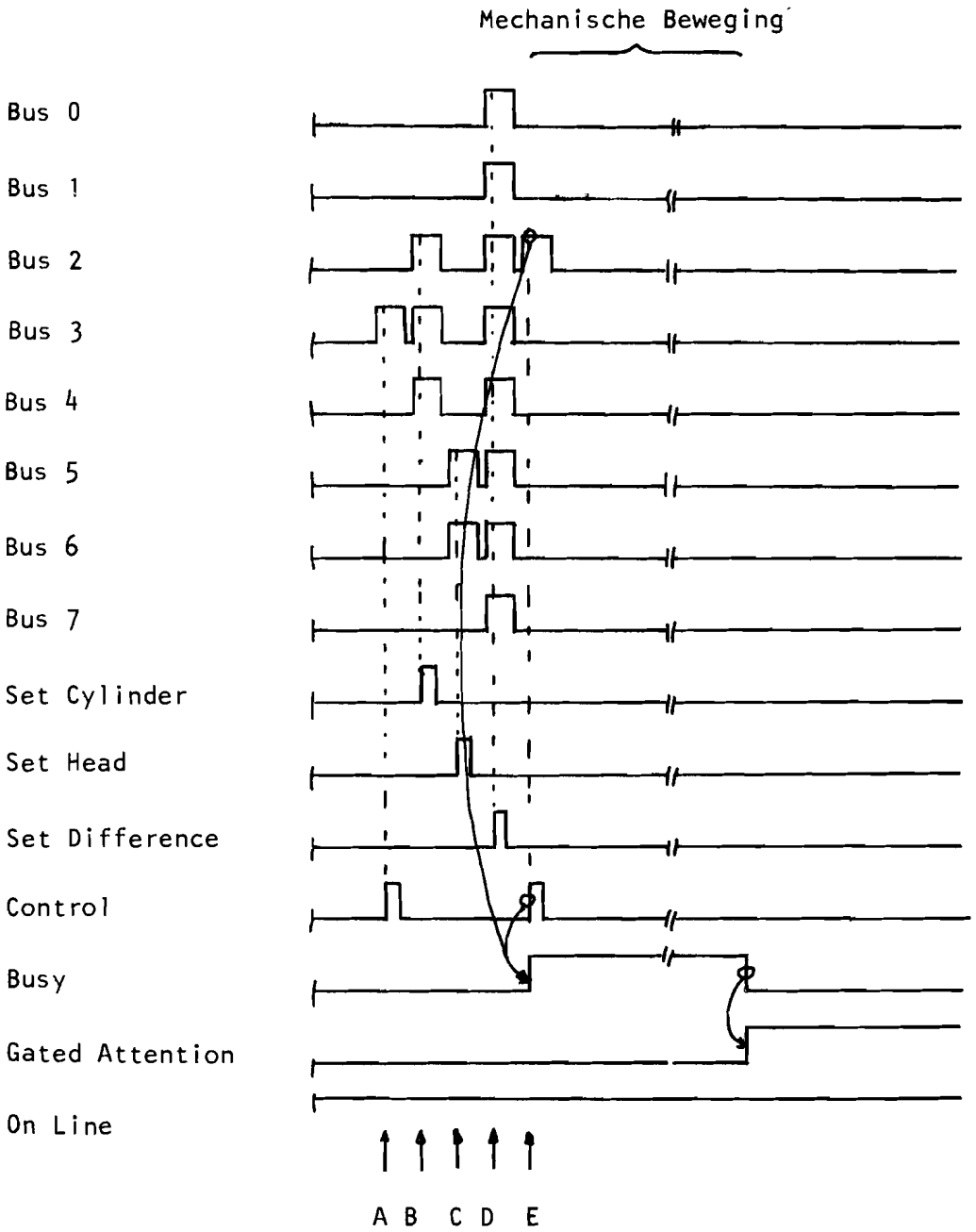


fig.4.22. Seek from cyl.56 track 2 to cyl.56 track 6.

SEEK FROM CYL.56 TRACK 2 TO CYL.56 TRACK 6.



A: Reset Head Register

B: Set Cylinder Adres

C: Set Head Adres + Direction

D: Set Difference

E: Seek Start

fig.4.23.

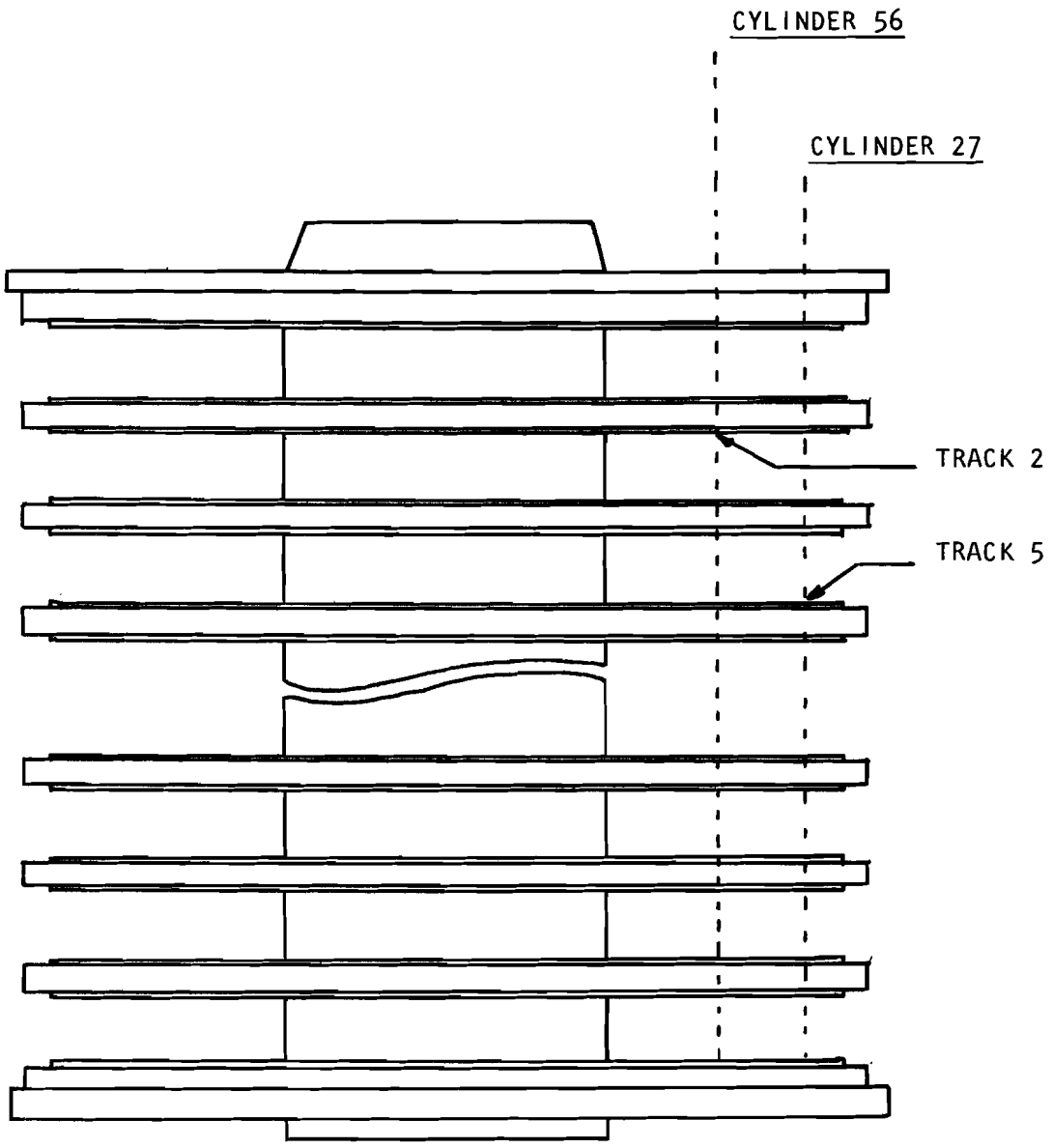
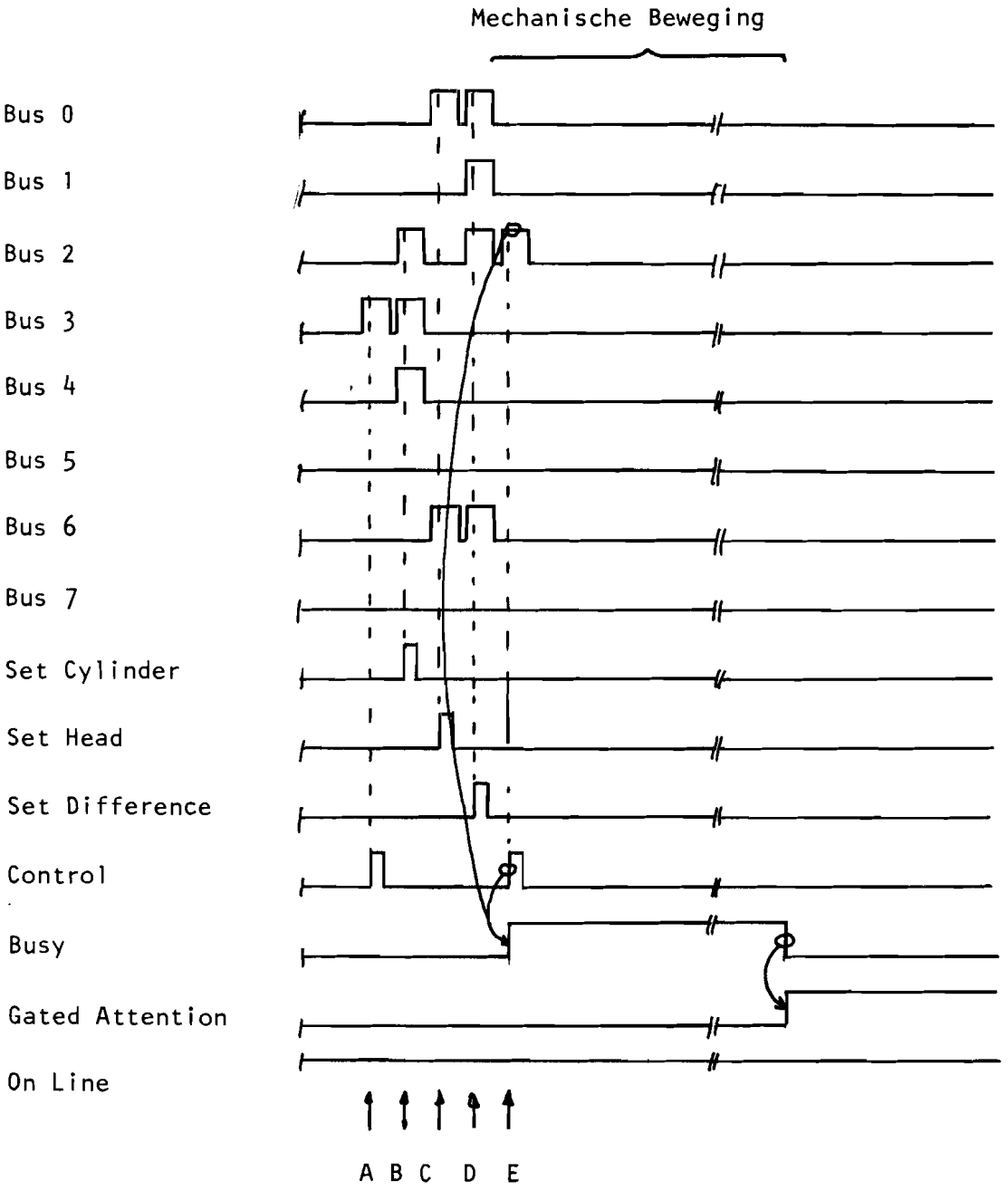


fig.4.24. Seek from cyl.27 track 5 to cyl.56 track 2.

SEEK FROM CYL.27 TRACK 5 TO CYL.56 TRACK 2.



A: Reset Head Register

B: Set Cylinder Adres

C: Set Head Adres + Direction

D: Set Difference

E: Seek Start

fig.4.25.

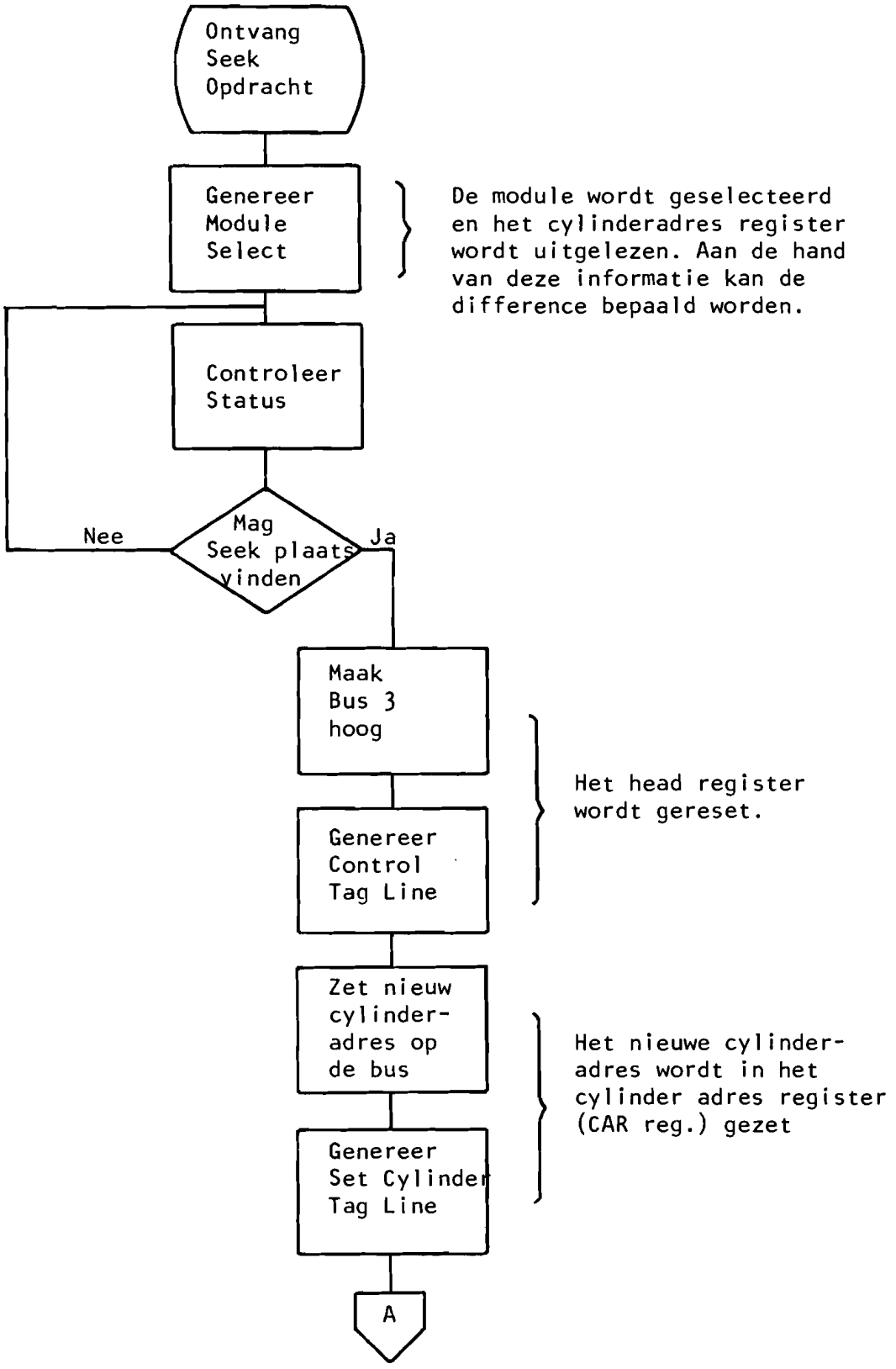


fig.4.26. Flow chart seek operatie.

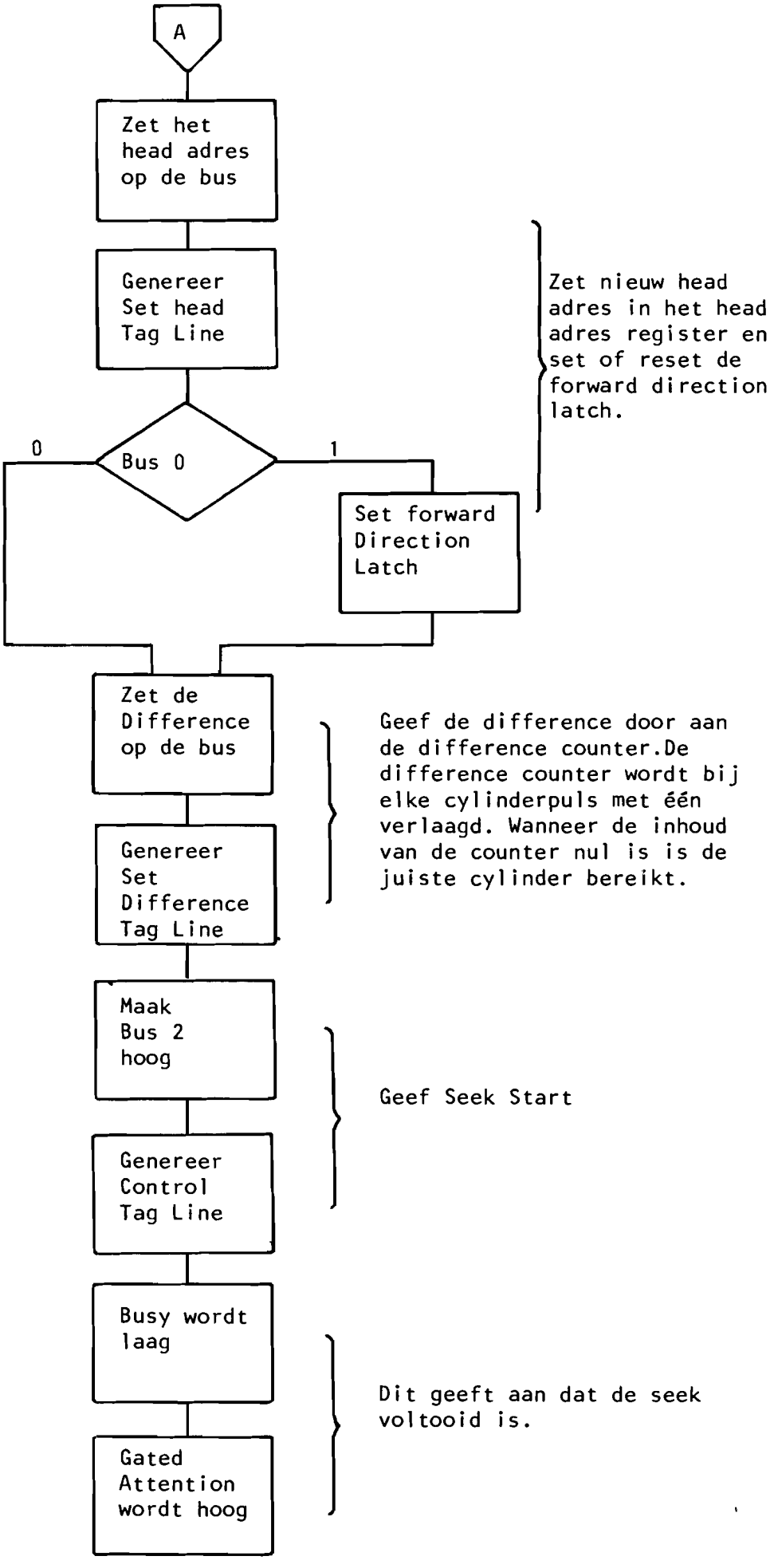
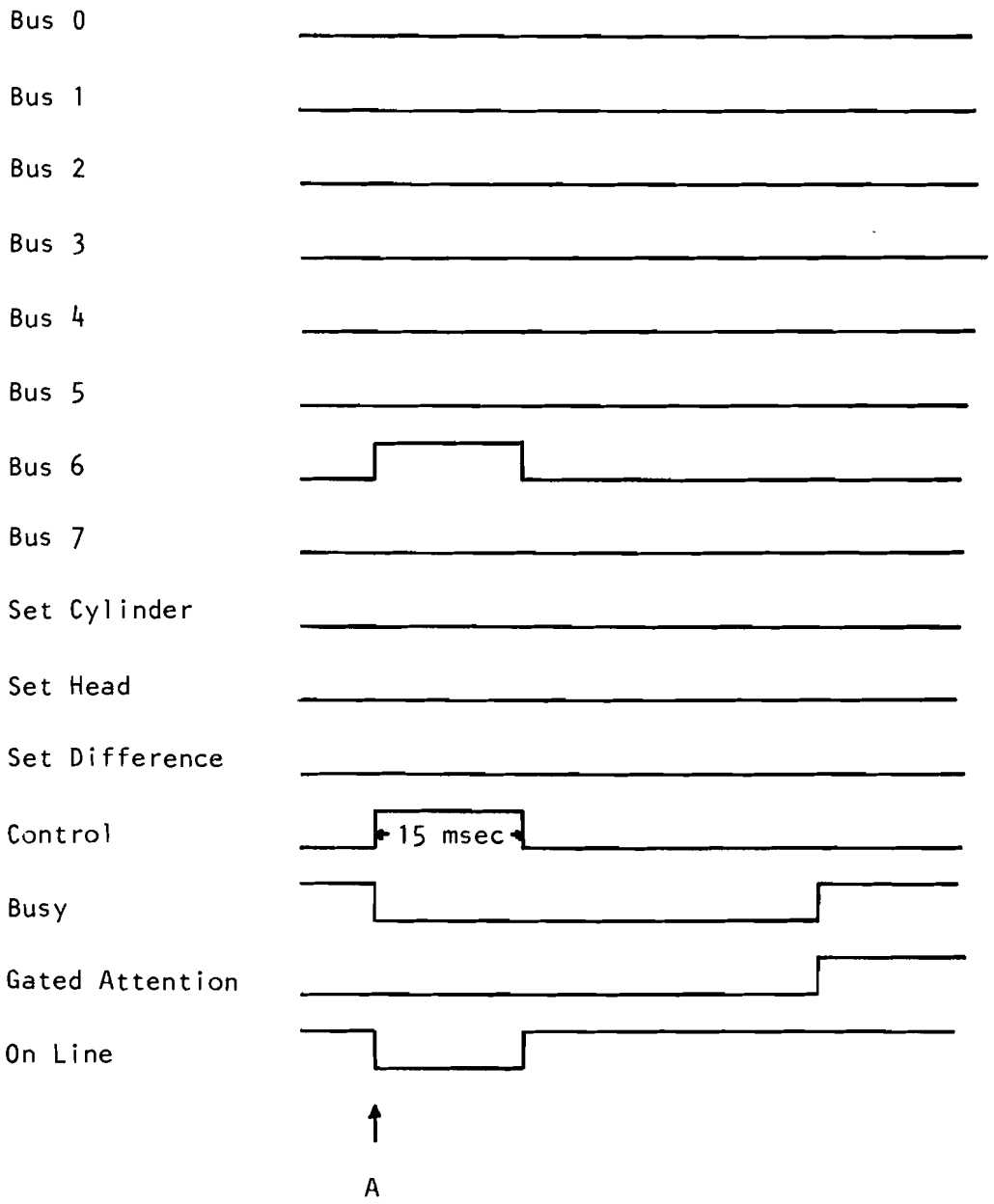

A
Zet het head adres op de bus
Genereer Set head Tag Line
Zet nieuw head adres in het head adres register en set of reset de forward direction latch.
0
Bus 0
1
Set forward Direction Latch
Zet de Difference op de bus
Geef de difference door aan de difference counter.De difference counter wordt bij elke cylinderpuls met één verlaagd. Wanneer de inhoud van de counter nul is is de juiste cylinder bereikt.
Genereer Set Difference Tag Line
Maak Bus 2 hoog
Geef Seek Start
Genereer Control Tag Line
Busy wordt laag
Dit geeft aan dat de seek voltooid is.
Gated Attention wordt hoog

fig.4.27.

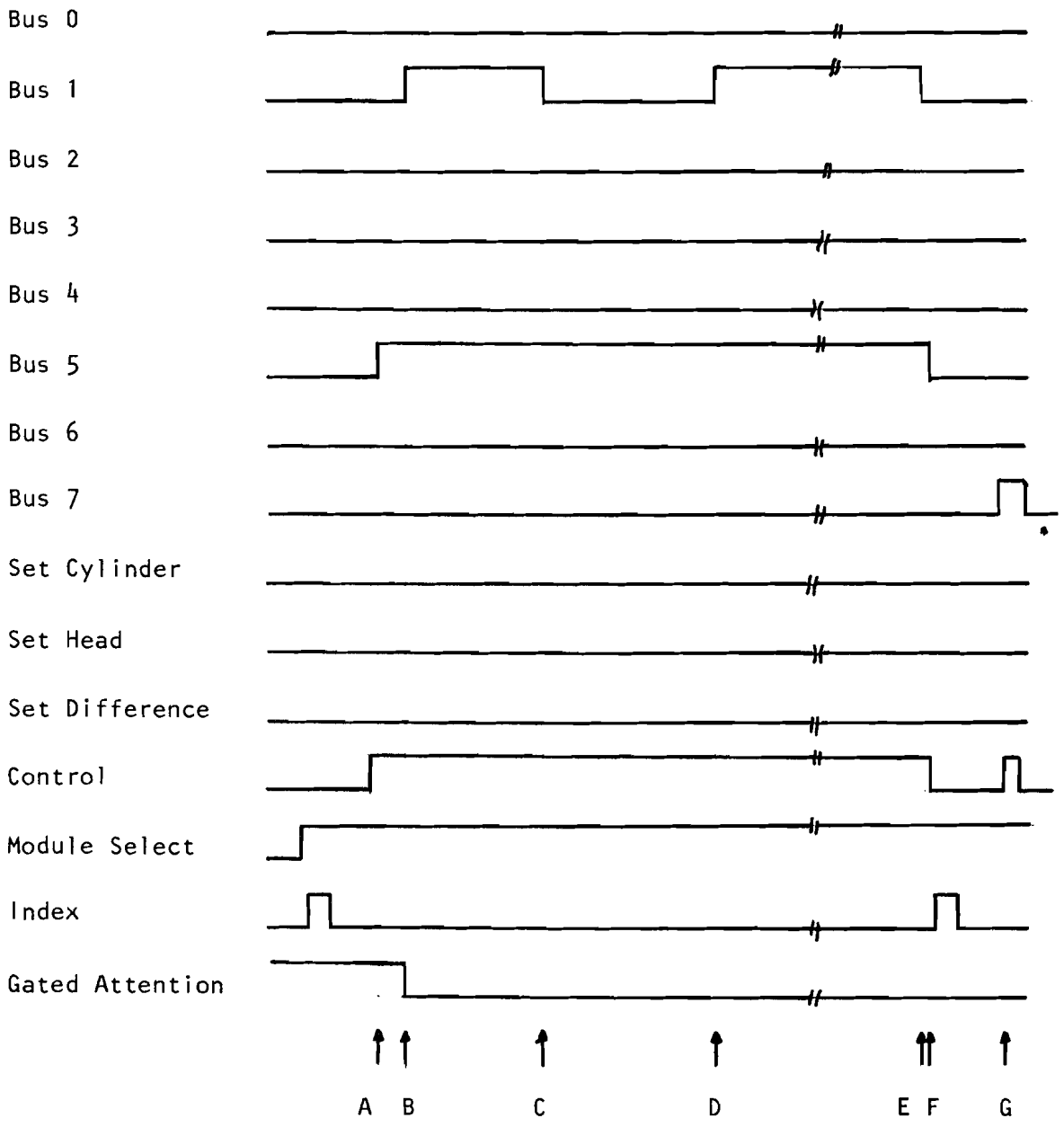
RETURN TO CYL 0



A: Return to zero

fig.4.28.

READ OPERATIE



A: Select Head

B: Set Read Gate

C: Reset Read Gate

D: Set Read Gate

E: Reset Read Gate

F: Reset Head Select + Control

G: Head Advance

fig.4.29.

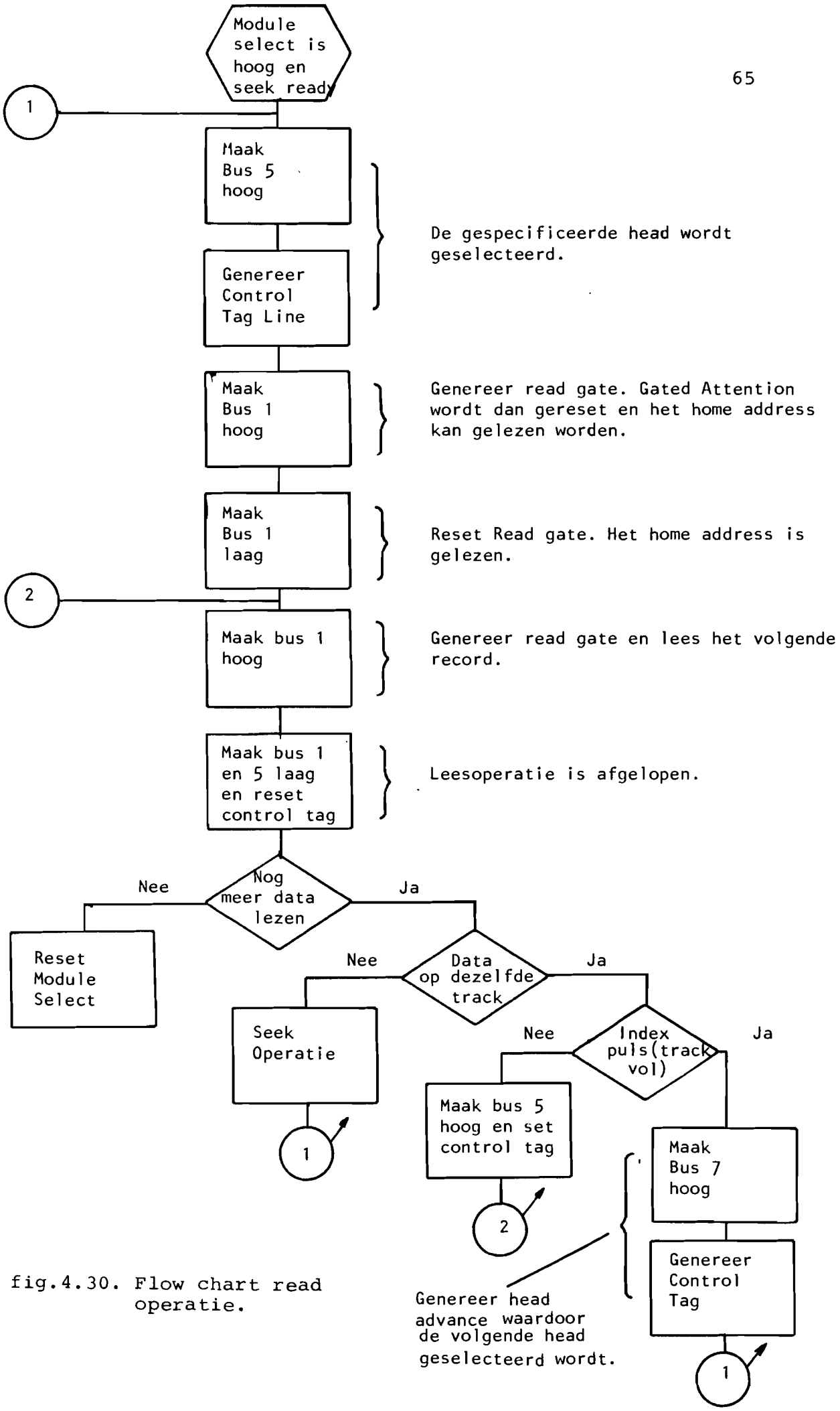
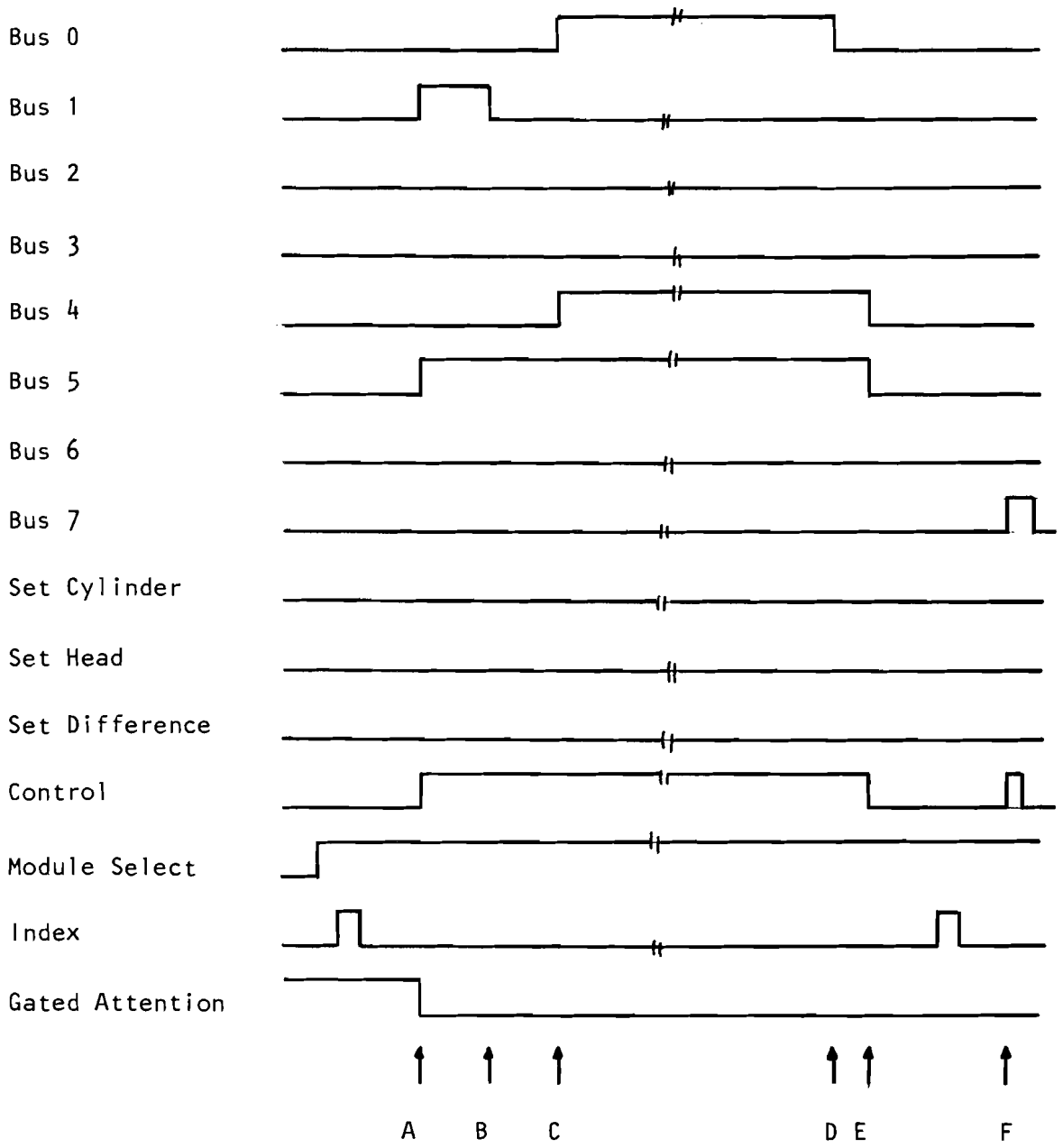


fig.4.30. Flow chart read operatie.

WRITE OPERATIE



A: Set Read Gate + Head Select

B: Reset Read Gate

C: Write Gate + Erase Gate

D: Reset Write Gate

E: Reset Erase Gate + Head Select + Control

F: Head Advance

fig.4.31.

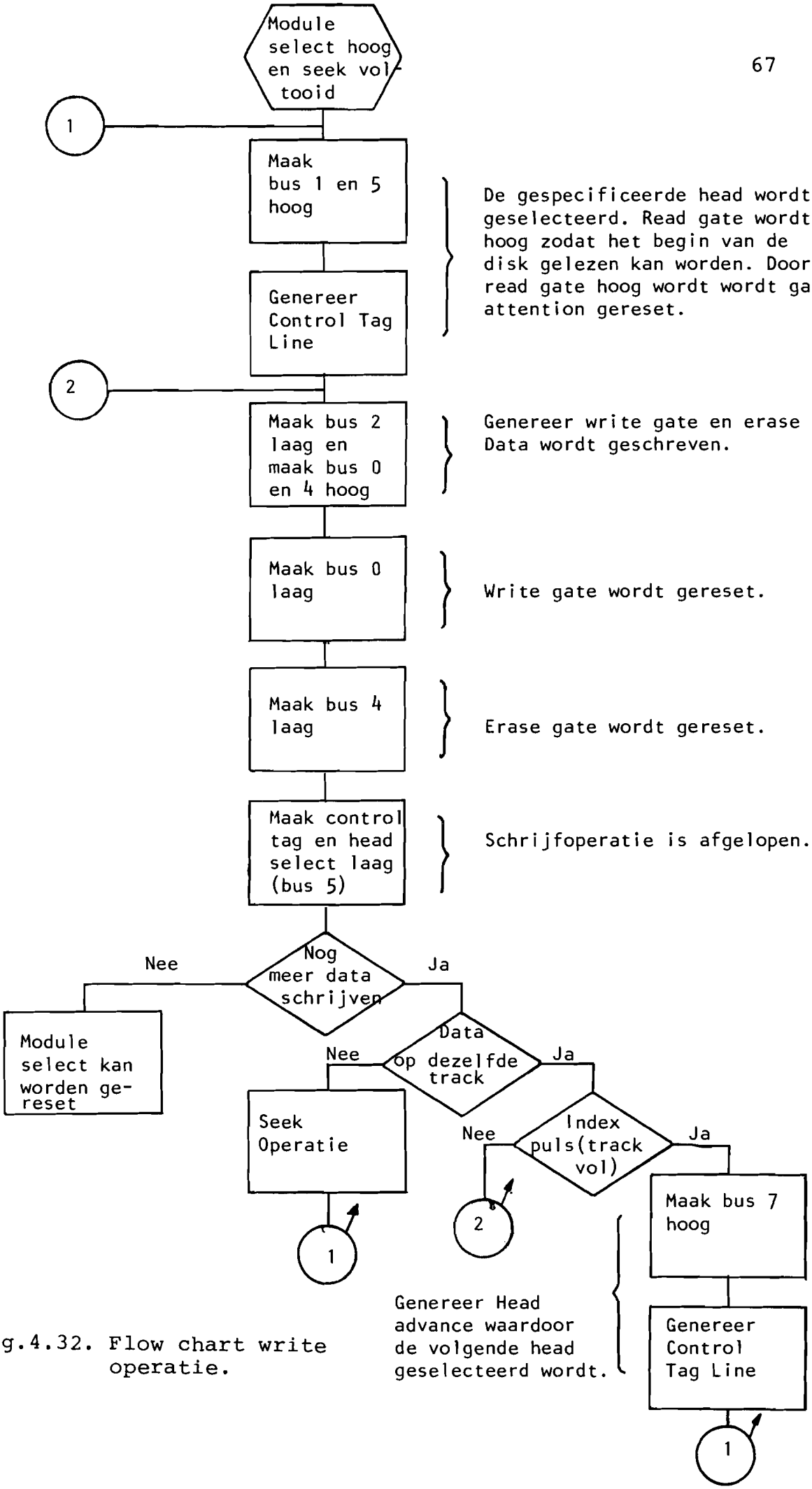


fig.4.32. Flow chart write operatie.

De belangrijkste technische gegevens van de IBM 2314.

Capaciteit:	203 cylinders (3 spares)
	20 tracks/cylinder
	7294 bytes/track

DC-voedingsspanningen:	+ 36 V.
	- 36 V.
	+ 6 V.
	+ 3 V.
	- 3 V.

Access time:

access motion time	: minimum 25 msec
	maximum 135 msec
	gemiddeld 75 msec
rotational delay time	: gemiddeld 12,5 msec

Recording technology:

recording method	: double frequency NRZ-method
datarate	: 2,5 Mbit/sec (312 kbyte/sec)
disk rotational speed	: 2400 rpm
bit-cell time	: 400 nsec
number of heads	: 20
read data pulse width	: minimaal 60 nsec
	nominaal 80 nsec
	maximaal 100 nsec

fig.4.34.

HOOFDSTUK V: De I/O controller voor de IBM 2314 disk.

5.1. De hardware.

De opbouw van de I/O controller waarbij we gebruik gemaakt hebben van de 8080 microprocessor is gegeven in figuur 5.1. Uit dit figuur blijkt dat de controller opgebouwd is uit vijf delen:

1) CPU kaart
2) DMA kaart
3) I/O kaart
4) Niveau omzetter voor de besturingssignalen
5) Een of meerdere kaarten,die zorgen voor de niveau conversie en de serie/parallel en parallel/serie omzetting van de data en die verder de CRC-checker en de VFO schakeling bevat.

Als CPU is een CPU kaart gebruikt, die gebaseerd is op het SYS 8000 systeem, zoals dat in de vakgroep EB gebruikt wordt. Door de stagiair H.J.J.Levels is reeds een DMA kaart ontworpen en gemaakt. Deze kaart is echter nog niet in staat om DMA via de bus te plegen. Wanneer de hiervoor benodigde hardware toegevoegd is kan deze kaart voor de controller gebruikt worden.
De I/O kaart, die toegepast is, vindt U in figuur 5.2.
Omdat het IBM niveau tussen - 1,7 V. en + 1,3 V. ligt moeten de signalen van en naar de disk omgezet worden van IBM naar TTL niveau en omgekeerd. In figuur 5.3 is het principeschema van de niveau omzetting weergegeven.
De pulsen, die aan de disk aangeboden moeten worden en van de disk afkomen,zien er als volgt uit:

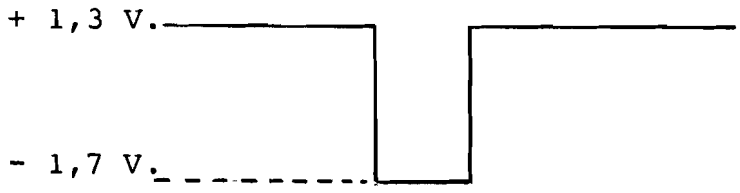


fig.5.4. Het IBM niveau.

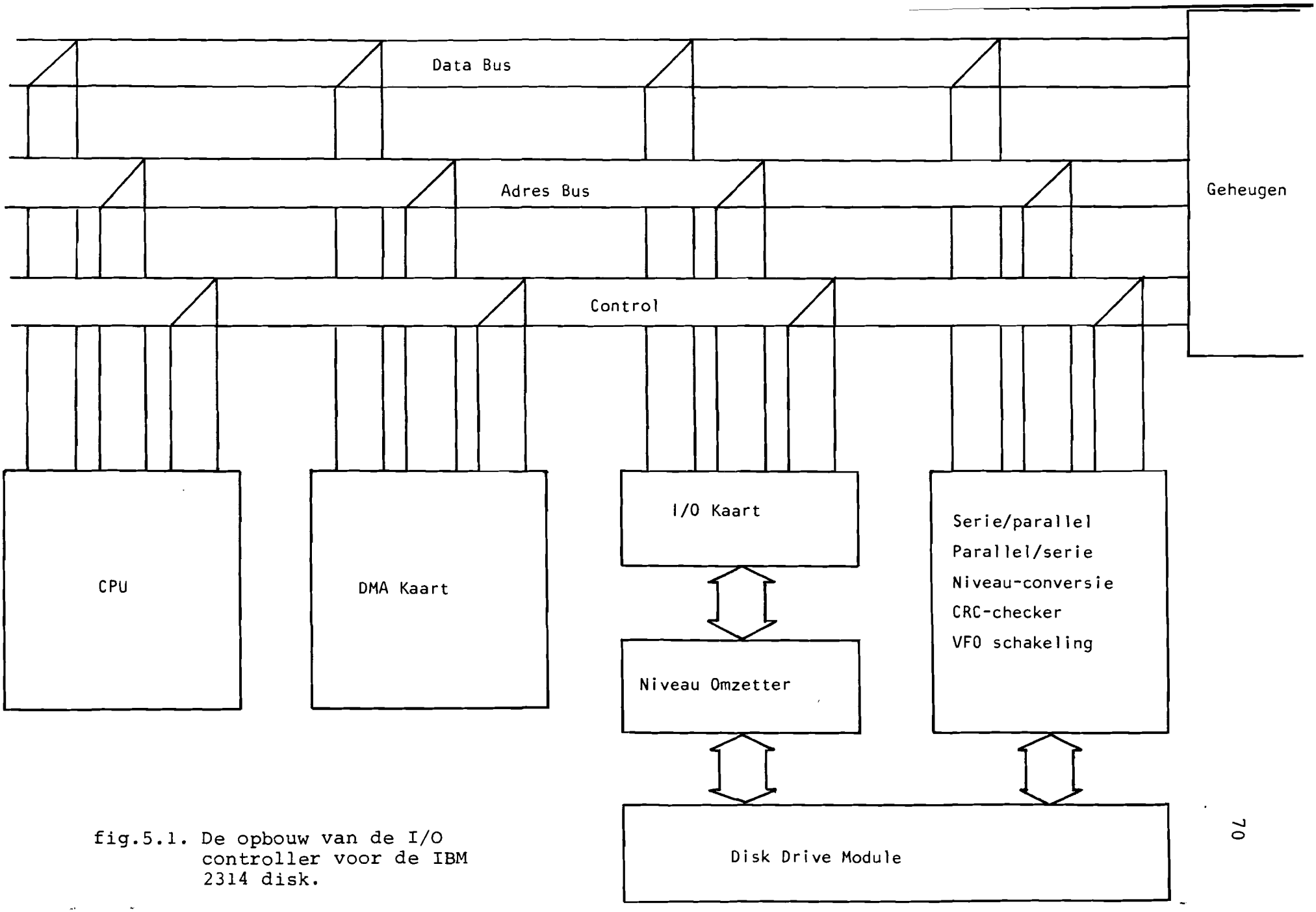


fig.5.1. De opbouw van de I/O controller voor de IBM 2314 disk.

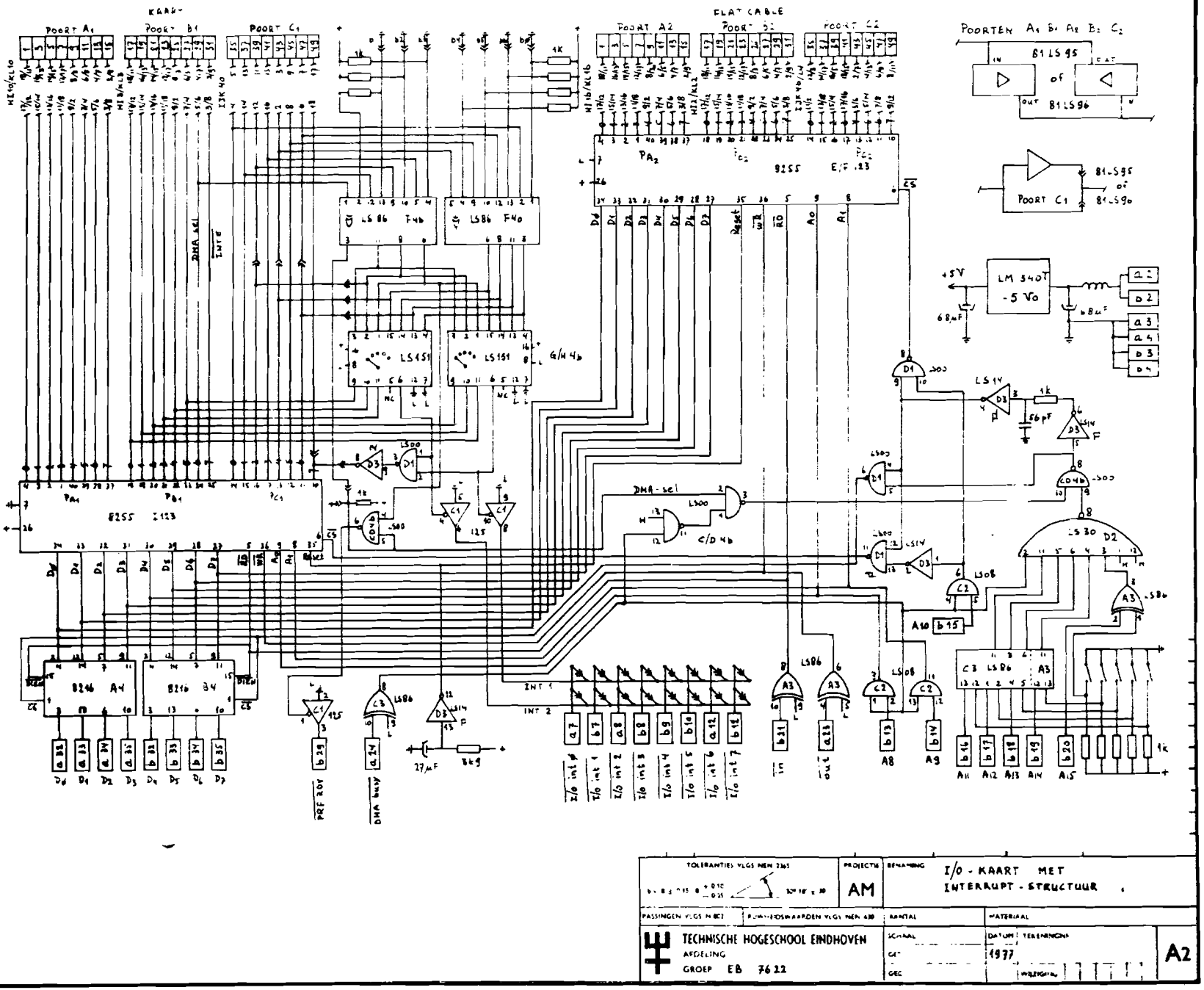
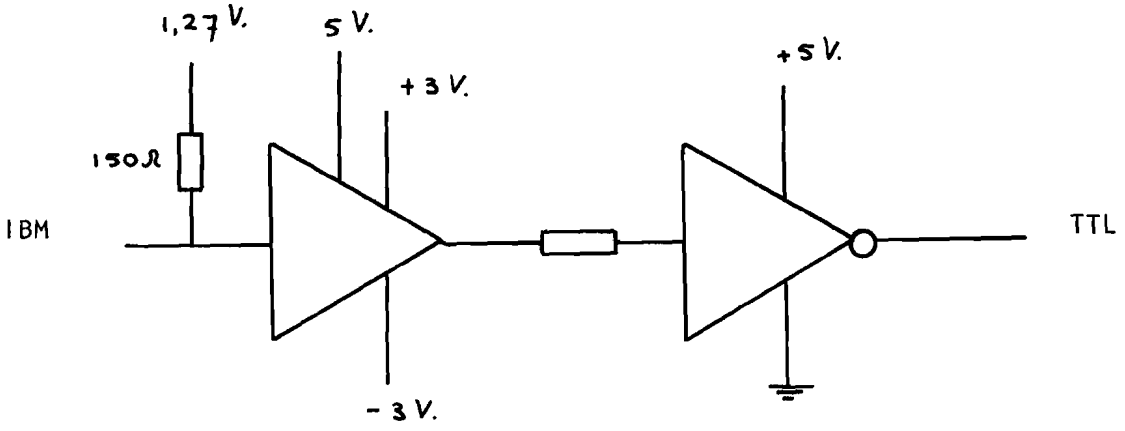
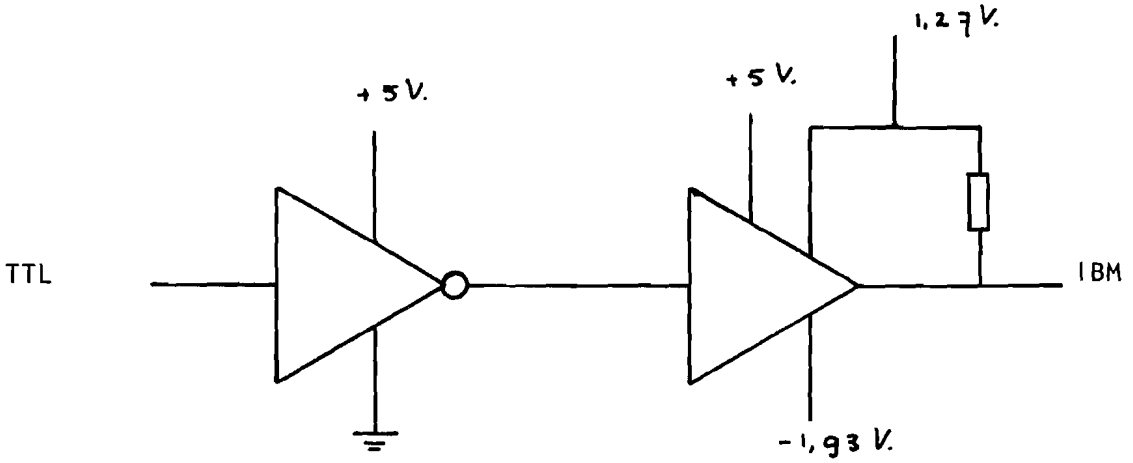


fig.5.2. De I/O kaart.



Van IBM naar TTL niveau.



Van TTL naar IBM niveau.

fig.5.3. De niveau omzetting.

Bij een puls wordt het niveau van + 1,3 V. naar - 1,7 V. getrokken m.b.v. een stroom van 30 mA over 100 Ohm.
Bij de 2314 wordt gebruik gemaakt van de double frequency NRZ-methode. De data, die van de disk afkomt ("ruwe data") wordt gescheiden in data en clock door de schakeling, die U aantreft in figuur 5.5. Om van de data nette pulsjes te maken is een Op-amp µA 733 toegepast. De schakeling wordt gesynchroniseerd op een nul. Het nadeel hiervan is dat de schakeling wanneer een clock puls mist uit de pas gaat lopen en bij de eerstvolgende nul weer gesynchroniseerd wordt. Voor de timing van deze schakeling wordt verwezen naar figuur 5.6.
Tot nu toe wordt alleen de CPU,de I/O kaart en de "VFO" schakeling gebruikt. Met deze delen van de controller kunnen we de juiste besturingssignalen aan de disk aanbieden, die nodig zijn voor een seek, write en read operatie.
De rest van de controller zal nog gerealiseerd moeten worden. Hierbij zullen eventueel al bestaande ontwerpen veranderd en aangepast moeten worden waarbij vooral gelet moet worden op de snelheid, zodat deze schakelingen de datarate van 312 kbyte/sec kunnen halen.
Verder is het aan te raden de "VFO" schakeling van figuur 5.5 te vervangen door een schakeling waarbij een phase lock loop gebruikt wordt. Hiermee wordt bereikt dat de schakeling minder storingsgevoelig is d.w.z. niet uit de pas gaat lopen bij het missen van een clock puls.
Bij het ontwerpen zal ook rekening gehouden moeten worden met het dataformaat dat men wil toepassen.

5.2. De software.

Over de software kunnen we vrij kort zijn. Er zijn een aantal testroutines gerealiseerd waarmee een aantal commando's uitgevoerd kunnen worden, zoals "return to zero"

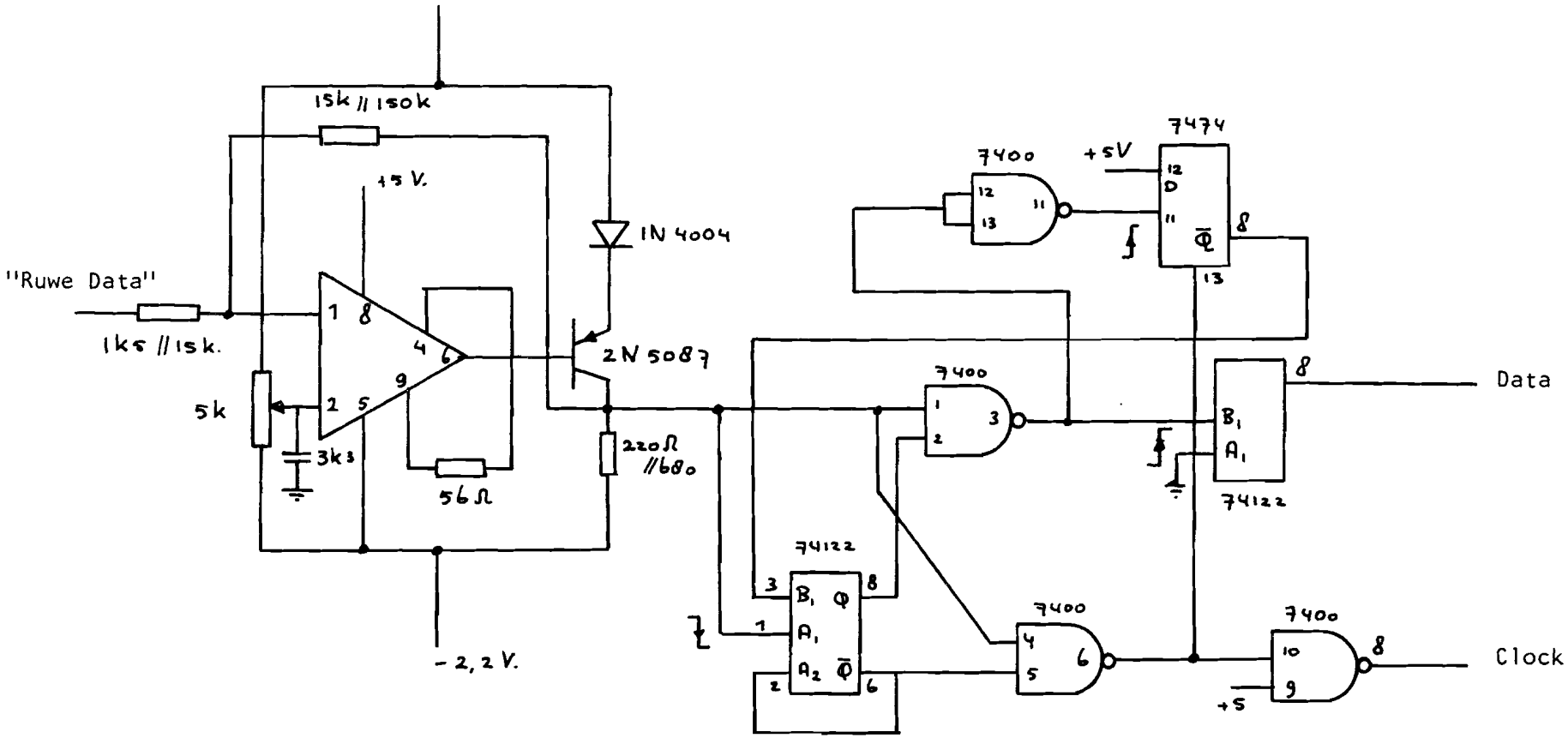


fig.5.5. De "VFO" schakeling.

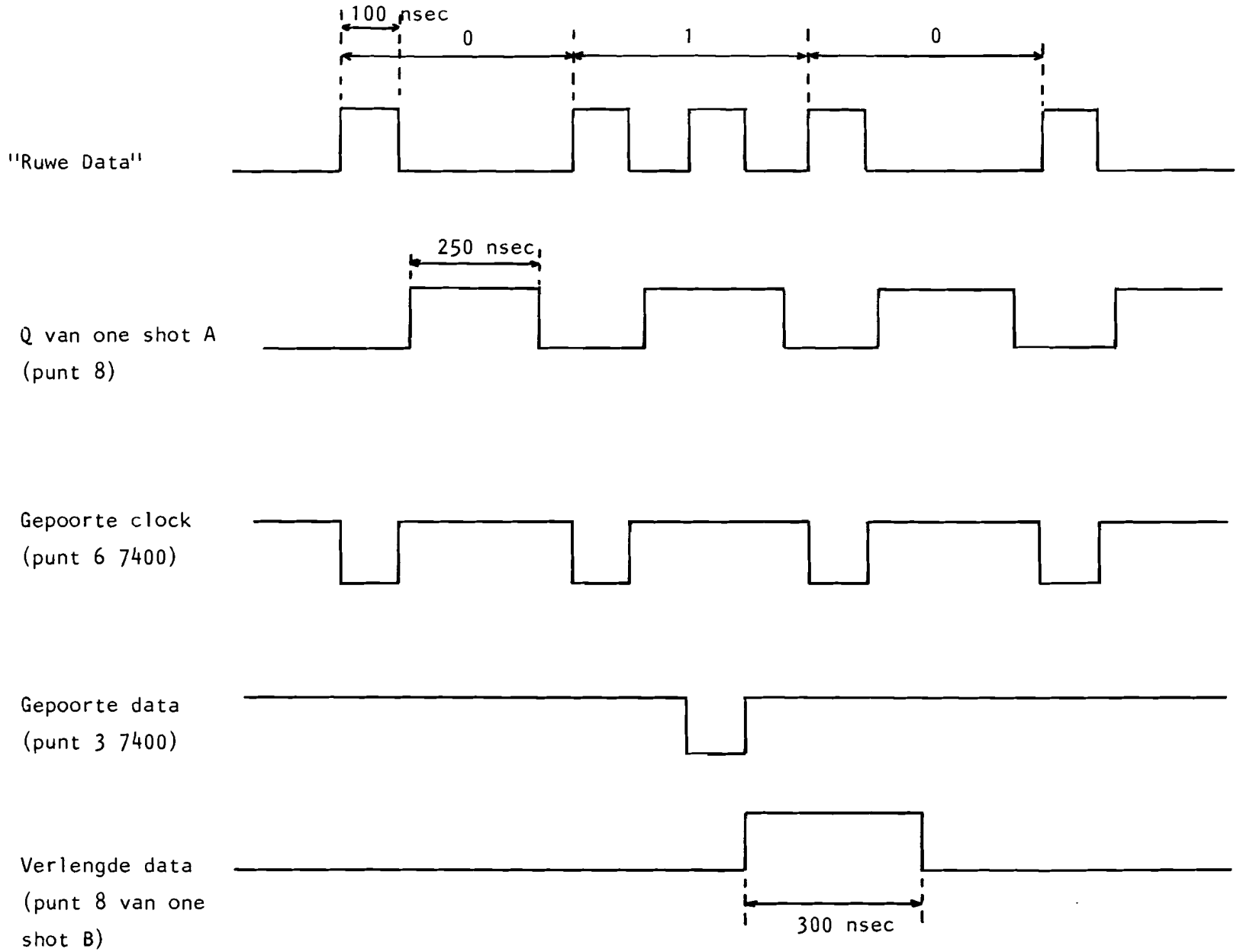


fig.5.6. De timing van de "VFO" schakeling.

en seek. In bijlage 1 t/m 9 vindt U de listings van deze programma's.
Bij het aanroepen van één van deze routines worden die signalen op de file bus en de tag bus gezet, die nodig zijn om de desbetreffende operatie uit te voeren. Over de volgorde van het aanbieden van deze signalen verwijs ik naar de timing diagrammen van de verschillende operaties met de daarbij horende flow charts.

Conclusie.

In dit verslag is aangetoond dat disks o.a. het type als de IBM 2314 bestuurd kunnen worden m.b.v. een microprocessor.
Bij de realisatie van de schakelingen, die gebruikt worden bij de dataverwerking, zal rekening gehouden moeten worden met de eis dat de componenten zo snel moeten zijn dat de datarate van 312 kbyte/sec gehaald kan worden.
Zoals uit de opbouw van de controller blijkt is nog wel de nodige hardware rond de microprocessor vereist. Het ligt in de verwachting, zoals al eerder opgemerkt is, dat er snellere microprocessoren op de markt komen. Het is dan misschien mogelijk dat door het toepassen van zo'n processor de hardware aanzienlijk eenvoudiger en flexibeler wordt.
Tot slot wil ik nogmaals van deze gelegenheid gebruik maken om iedereen te bedanken, die mij tijdens mijn afstudeerwerk hebben geholpen, en wel met name prof.ir.A.Heetman, mijn beide coaches ir.M.P.J.Stevens en ir.J.P.Kemper en de heer L.A.v.Bokhoven.

LITERATUURLIJST.

Snelle microprogrammeerbare I/O controller
A.M.G.Claessen,
Afstudeerverslag ECB 680.

Disk controller m.b.v. microprogrammeerbare bit-slices
A.J.F.M.Dhaeze,
Afstudeerverslag ECB 723.

Lineprinter controller m.b.v. bit-slices
R.W.Hoeke,
Afstudeerverslag ECB 722.

Microprogramming a bipolar microprocessor
J.R.Mick, J.Brick,
Microprogramming Handbook (Advanced Micro Devices)

A high performance disk controller
Advanced Micro Devices (AMD)

Universele I/O communicatie in computersystemen
P.J.C.M.Matthee,
Afstudeerverslag ECB 706.

DMA controller
H.J.J.Levels,
Stageverslag ECB 734.

Peripheral Devices
I.Flores,
Prentice Hall, 1973 (BP 73144).

IBM Field Engineering:

- Operation Manual, Instruction Manual
  73-10554 X2
- Maintenance Manual
  73-10554 X1
- Maintenance Diagrams:
  MDM 73-10554 vol.M1
  ALD 73-10554 vol.1
  ALD 73-10554 vol.A1
  ALD 73-10554 vol.A2
  ALD 73-10554 vol.A3
  ALD 73-10554 vol.A4
  CLD 73-10554 vol.C1
  CLD 73-10554 vol.C2
  CLD 73-10554 vol.L1
  MDP vol.T1, vol.S01
  MDP vol.T4

```
        ORG     300H
BFILE   EQU     0E8H    ;PLAATS FILE BUS
BMODS   EQU     0E9H    ;PLAATS MODULE SELECT
BTAG    EQU     0EAH    ;PLAATS TAGBUS
BCAR    EQU     0ECH    ;PLAATS CAR REGISTER
BCONT1  EQU     0EDH    ;PLAATS BUSY, ONLINE, UNSAFE, SEEK INC,
                        ;END CYL, PACK CHANGE, WRITE CURR SENSE
                        ;HEADS EXTENDED
BCONT2  EQU     0EEH    ;PLAATS GATED ATT., MODULE SELECTED,INDX
TDIFF   EQU     01H     ;TAG DIFFERENCE
TCYL    EQU     02H     ;TAG CYLINDER
THEAD   EQU     04H     ;TAG HEAD
TCONTR  EQU     08H     ;TAG CONTROL
BUSY    EQU     01H     ;SELECTED FILE BUSY
ONLINE  EQU     02H     ;SELECTED ON LINE
UNSAFE  EQU     04H     ;FILE UNSAFE
SEEKIN  EQU     08H     ;SELECTED SEEK INCOMPLETE
ENDCYL  EQU     10H     ;SELECTED END OF CYLINDER
PACKCH  EQU     20H     ;SELECTED PACK CHANGE
CURSEN  EQU     40H     ;WRITE CURRENT SENSE
HEADEX  EQU     80H     ;HEADS EXTENDED
MODSL1  EQU     01H     ;MODULE SELECT 1
MODSL2  EQU     02H     ;MODULE SELECT 2
GTATT1  EQU     01H     ;GATED ATTENTION 1
GTATT2  EQU     02H     ;GATED ATTENTION 2
MD1SLD  EQU     04H     ;MODULE 1 SELECTED
MD2SLD  EQU     08H     ;MODULE 2 SELECTED
INDEX   EQU     10H     ;SELECTED INDEX
WRITEG  EQU     80H     ;WRITE GATE
READG   EQU     40H     ;READ GATE
SEEKST  EQU     20H     ;SEEK START
RESHR   EQU     10H     ;RESET HEAD REG
ERASEG  EQU     08H     ;ERASE GATE
SELHEA  EQU     04H     ;SELECT HEAD
RTZERO  EQU     02H     ;RETURN TO ZERO
HEADAD  EQU     01H     ;HEAD ADVANCE
FORWLV  EQU     80H     ;FORWARD LATCH VOORUIT
FORWLA  EQU     00H     ;FORWARD LATCH ACHTERUIT
SLRES   EQU     40H     ;SELECT. LOCK RESET
ZERO    EQU     00H
CR      EQU     0DH     ;CARRIAGE RATURN
ETX     EQU     03H     ;END TEXT
LF      EQU     0AH     ;LINE FEED
HT      EQU     09H     ;HORIZENTAL TAB
SSTAT:  DB      'STATUS',LF,CR,ETX
SBUSY:  DB      'BUSY',CR,ETX
SNBUSY: DB      'NOT BUSY',CR,ETX
SONL:   DB      'ON LINE',CR,ETX
SNONL:  DB      'NOT ON LINE',CR,ETX
SUNSAF: DB      'UNSAFE',LF,CR,ETX
SSAFE:  DB      'SAFE',LF,CR,ETX
SPCH:   DB      'PACK CHANGE',LF,CR,ETX
SNPCH:  DB      'NO PACK CHANGE',LF,CR,ETX
SWCS:   DB      'DISK WRITING',CR,ETX
SNWCS:  DB      'DISK NOT WRITING',CR,ETX
SHDEX:  DB      'HEADS EXTENDED',CR,ETX
```

Bijlage 2.

```
SNHDEX: DB      'HEADS NOT EXTENDED',CR,ETX
SSKINC: DB      'SEEK INCOMPLETE',LF,CR,ETX
SSKCOM: DB      'SEEK COMPLETE',LF,CR,ETX
SEOC:   DB      'END OF CYLINDER',LF,CR,ETX
SNEOC:  DB      'NO END OF CYLINDER',LF,CR,ETX
SGTA1H: DB      'GATED ATTENTION 1 HIGH',CR,ETX
SGTA1L: DB      'GATED ATTENTION 1 LOW',CR,ETX
SGTA2H: DB      'GATED ATTENTION 2 HIGH',LF,CR,ETX
SGTA2L: DB      'GATED ATTENTION 2 LOW',LF,CR,ETX
STPWRO: DB      'TURN POWER ON',LF,CR,ETX
SFR:    DB      'FILE READY',LF,CR,ETX
SF:     DB      'FILE NOT READY',LF,CR,ETX
SREAD:  DB      'FILE IS READING',LF,CR,ETX
SSKF:   DB      'CYLINDERADRES IN ROUTINE SEEK '
        DB      'CYLINDER TO HIGH',LF,CR,ETX
SHDF:   DB      'HEADADRES IN ROUTINE SEEK CYLINDER'
        DB      ' TO HIGH',LF,CR,ETX
SRTZ:   DB      'RETURNED TO ZERO',LF,CR,ETX
SSLRES: DB      'SELECT. LOCK RESETTED',LF,CR,ETX
GTATT:  DB      00H         ;GATED ATTENTION
MODSL:  DB      00H         ;MODULE SELECT
CAR:    DB      00H         ;CAR
FORWL:  DB      00H         ;FORWARD LATCH
CYL:    DB      00H         ;CYLINDER
HEAD:   DB      00H         ;HEAD
NONDIF: DB      00H         ;NON DIFFERENCE
CONTRL: DB      00H         ;CONTROLEWOORD
HEADFL: DB      00H         ;HEAD + FORWARD LATCH
MDSLD:  DB      00H         ;MODULE SELECTED
RDISPL: MOV     A,M         ;ROUTINE "DISPLAY"
        CPI     ETX         ;DE ASCI CODES DIE OP DE ADRESSEN VANAF HET
                            ;ADRES IN REG.PAIR H ZITTEN WORDEN OP DE
                            ;MONITOR GEDISPLAYED TOT HET EERST-
                            ;VOLGENDE ETX (03) SYMBOOL.
        RZ
        MOV     C,A
        CALL    0F809H
        INX     H
        JMP     RDISPL      ;GEBRUIKTE REGISTERS: A,H,C
        RET
RLF:    MVI     C,LF        ;ROUTINE "X MAAL LINEFEED"
        CALL    0F809H      ;GETAL X MOET IN REG L ZITTEN
        DCR     L           ;DE MONITOR KRIJGT X MAAL EEN LINEFEED
        JNZ     RLF         ;GEBRUIKTE REGISTERS: A,C,L
        MVI     C,CR
        CALL    0F809H
        RET
R4TAB8: MVI     H,4         ;ROUTINE "4 MAAL TAB 8"
R4THO:  MVI     C,9H        ;DE MONITOR KRIJGT 4 MAAL EEN TAB
        CALL    0F809H      ;GEBRUIKTE REGISTERS: A,C,H
        DCR     HH
        MOV     A,H
        SUI     ZERO
        JNZ     R4THO
        RET
RSTAT:  LXI     H,SSTAT     ;ROUTINE "DISPLAY STATUS"
```

Bijlage 3.

```
        CALL    RDISPL  ;HET WOORD "STATUS" VERSCHIJNT OP HET SCHERM
        IN      BCONT1  ;HET WOORD OP CONTROLEBUS 1 KOMT IN REG.A
        MOV     E,A     ;EN WORDT WEGGEZET IN REG.E.DIT WOORD WORDT
        ANI     HEADEX  ;TELKENS GECHECKED OP EEN BEPAALD BIT
        LXI     H,SHDEX ;OVEREENKOMEND MET EEN BEPAALDE
        JNZ     RST0    ;TOESTAND EN DEZE TOESTAND WORDT WEERGE-
        LXI     H,SNHDEXX       ;GEVEN OP DE MONITOR.
RST0:   CALL    RDISPL  ;"HEADS (NOT) EXTENDED" VERSCHIJNT OP MONITOR
        CALL    R4TAB8  ;VIER MAAL TAB 8
        MOV     A,E
        ANI     PACKCH
        LXI     H,SPCH
        JNZ     RST1
        LXI     H,SNPCH
RST1:   CALL    RDISPL  ;"(NO) PACK CHANGE" VERSCHIJNT OP MONITOR
        MOV     A,E
        ANI     ONLINE
        LXI     H,SONL
        JNZ     RST2
        LXI     H,SNONL
RST2:   CALL    RDISPL  ;"(NOT) ON LINE" VERSCHIJNT OP MONITOR
        CALL    R4TAB8  ;VIER MAAL TAB 8
        MOV     A,E
        ANI     UNSAFE
        LXI     H,SUNSAF
        JNZ     RST3
        LXI     H,SSAFE
RST3:   CALL    RDISPL  ;"(UN)SAFE" VERSCHIJNT OP DE MONITOR
        MOV     A,E
        ANI     BUSY
        LXI     H,SBUSY
        JNZ     RST4
        LXI     H,SNBUSY
RST4:   CALL    RDISPL  ;"(NOT) BUSY" VERSCHIJNT OP MONITOR
        CALL    R4TAB8  ;VIER MAAL TAB 8
        MOV     A,E
        ANI     SEEKIN
        LXI     H,SSKINC
        JNZ     RST5
        LXI     H,SSKCOM
RST5:   CALL    RDISPL  ;"SEEK (IN)COMPLETE" VERSCHIJNT OP MONITOR
        MOV     A,E
        ANI     CURSEN
        LXI     H,SWCS
        JNZ     RST6
        LXI     H,SNWCS
RST6:   CALL    RDISPL  ;"DISK (NOT) WRITING" VERSCHIJNT OP MONITOR
        CALL    R4TAB8  ;VIER MAAL TAB 8
        MOV     A,E
        ANI     ENDCYL
        LXI     H,SEOC
        JNZ     RST7
        LXI     H,SNEOC
RST7:   CALL    RDISPL  ;"(NO) END OF CYLINDER " VERSCHIJNT OP MONITOR
        IN      BCONT2  ;WOORD OP CONTROLEBUS 2 KOMT IN REG.A EN WORDT
        MOV     E,A     ;WEGGEZET IN REG.E. OOK NU WORDT TELKENS WEER
```

Bijlage 4.

```
        ANI     GTATT1  ;GECHECKED OP EEN BEPAALDE BIT OVEREENKOMEND
        LXI     H,SGTA1H        ;MET EEN TOESTAND.DEZE TOESTAND WORDT WEE
        JNZ     RST8    ;GEDISPLAYED.
        LXI     H,SGTA1L
RST8:   CALL    RDISPL
        CALL    R4TAB8
        MOV     A,E
        ANI     GTATT2
        LXI     H,SGTA2H
        JNZ     RST9
        LXI     H,SGTA2L
RST9:   CALL    RDISPL
        IN      BCAR
        STA     CAR
        RET
RDLXMS: MOV     B,H     ;ROUTINE "DELAY X MSEC"
RDLH0:  MVI     A,92    ;GETAL X MOET IN REG. H ZITTEN
RDLH1:  SUI     1       ;X WORDT GEZET IN REG.B
        JNZ     RDLH1   ;EEN LOOP VAN 1 MSEC TREEDT OP EN
                        ;B WORDT 1 VERLAAGD. ALS B NIET NUL IS
                        ;DAN WORDT DE LOOP WEER DOORLOPEN TOTDAT
                        ;B GELIJK IS AAN NUL
        DCR     B
        JNZ     RDLH0
        RET
RDLXDS: MVI     B,100   ;ROUTINE "DELAY X/10 SEC"
        CALL    RDLH0   ;GETAL X MOET IN REG H ZITTEN
                        ;ER TREEDT X MAAL EEN VERTRAGING
                        ;VAN 100 MSEC OP. DEZE WORDT WEER
                        ;GEREALISEERD MET ROUTINE "DELAY
                        ;XMSEC"
        DCR     H
        JNZ     RDLXDS
        RET
RPWRON: MVI     A,80H   ;ROUTINE "POWER ON"
        OUT     0EBH    ;CONTROLEWOORD IN EERSTE 8255
        MVI     A,9BH
        OUT     0EFH    ;CONTROLEWOORD IN TWEEDE 8255
        MVI     A,ZERO
        OUT     BMODS   ;MODULE SELECTBUS WORDT 0
        OUT     BFILE   ;FILEBUS WORDT 0
        OUT     BTAG    ;TAGBUS WORDT 0
        LXI     H,STPWRO;DISPLAY "TURN POWER ON"
        CALL    RDISPL
        EI
RPWRH2: HLT
        EI
        MVI     H,150   ;DELAY 15 SEC.
        CALL    RDLXDS
        LDA     MODSL   ;MODULE WORDT GESELECTEERD
        OUT     BMODS
        IN      BCONT1  ;CONTROLEBUS 1 IN ACCUMULATOR
        ANI     ONLINE  ;TEST OP "ON LINE"
        JZ      RPWRH1  ;NOT ON LINE;DAN JUMP NAAR RPWRH1
        IN      BCONT1  ;CONTROLEBUS 1 IN ACCUMULATOR
        ANI     HEADEX  ;TEST OP HEADS EXTENDED
```

Bijlage 5.

```
        JZ      RPWRH1  ;HEADS NOT EXTENDED JUMP NAAR RPWRH1
        IN      BCONT1  ;CONTROLEBUS 1 IN ACCUMULATOR
        ANI     UNSAFE  ;TEST OP UNSAFE
        JNZ     RPWRH1  ;UNSAFE DAN JUMP NAAR RPWRH1
        IN      BCONT2  ;CONTROLEBUS 2 IN ACCUMULATRO
        MOV     B,A
        LDA     GTATT   ;GATED ATTENTION VAN DE GESELECTEERDE
        ANA     B       ;MODULE LAAG DAN JUMP NAAR RPWRH1
        JZ      RPWRH1
RPWRHO: MVI     A,READG ;RESET GATED ATTENTION D.M.V.
                        ;READ GATE OP DE BUS
        OUT     BFILE   ;PLUS CONTROL TAG
        MVI     A,TCONTR
        OUT     BTAG
        MVI     A,ZERO
        OUT     BTAG
        OUT     BFILE
        IN      BCONT2  ;CONTROLE BUS 2 IN ACCUMULATOR
        JNZ     RPWRHO  ;JA ? DISPLAY TOESTAND EN RETURN
        CALL    RSTAT   ;NEE ? JUMP NAAR RPWRHO
        MVI     A,ZERO
        OUT     BMODS
        MOV     B,A
        LDA     GTATT   ;GATED ATTENTION VAN DE GESELECTEERDE DRIVE LAAG
        ANA     B
        RET
RPWRH1: CALL    RSTAT   ;DISPLAY STATUS
        JMP     RPWRH2
RSKCYL: LDA     CYL     ;ROUTINE "SEEK CYLINDER"
        SUI     203     ;ALS HET CYLINDERADRES HOGER/GELIJK IS AAN 203
        JC      RSKCHO  ;DAN DISPLAY "CYLINDERADRES TO HIGH"
        LXI     H,SSKF
        CALL    RDISPL
        RET
RSKCHO: LDA     HEAD    ;ALS HET HEAD ADRES HOGER/GELIJK IS AAN 20
        SUI     20      ;DAN DISPLAY "HEAD ADRES TO HIGH"
        JC      RSKCH1
        LXI     H,SHDF
        CALL    RDISPL
        RET
RSKCH1: LDA     MODSL   ;SELECTEER MODULE
        OUT     BMODS
        MVI     A,RESHR;RESET HEAD REGISTER
        OUT     BFILE
        MVI     A,TCONTR
        OUT     BTAG
        MVI     A,ZERO
        OUT     BTAG
        LDA     CYL     ;SET CYLINDER ADRES
        OUT     BFILE
        MVI     A,TCYL
        OUT     BTAG
        MVI     A,ZERO
        OUT     BTAG
        LDA     HEADFL  ;SET HEAD + FORWARD LATCH
        OUT     BFILE
```

```
        MVI     A,THEAD
        OUT     BTAG
        MVI     A,ZERO
        OUT     BTAG
        LDA     NONDIF  ;SET NON DIFFERENCE
        OUT     BFILE
        MVI     A,TDIFF
        OUT     BTAG
        MVI     A,ZERO
        OUT     BTAG
        MVI     A,SEEKST;SEEK START
        OUT     BFILE
        MVI     A,TCONTR
        OUT     BTAG
        MVI     A,ZERO
        OUT     BTAG
        OUT     BFILE
        OUT     BMODS
        RET
RRTZ:   LDA     MODSL;ROUTINE "RETURN TO ZERO"
        OUT     BMODS   ;SELECTEER MODULE
        MVI     A,RTZERO
        OUT     BFILE   ;FILE BUS 6 EN TAG CONTROL WORDEN 24 MSEC
        MVI     A,TCONTR;HOOG GEHOUDEN D.W.Z. OPDRACHT RETURN TO ZERO
        OUT     BTAG
        MVI     H,24
        CALL    RDLXMS
        MVI     A,ZERO
        OUT     BTAG
        OUT     BFILE
        OUT     BMODS
        LXI     H,SRTZ  ;DISPLAY "RETURNED TO ZERO"
        CALL    RDISPL
        RET
RHDADV: LDA     MODSL;ROUTINE "HEAD ADVANCE"
        OUT     BMODS   ;SELECTEER MODULE
        MVI     A,HEADAD
        OUT     BFILE   ;FILE BUS 7 + TAG CONTROL WORDEN KORTE TIJD
        MVI     A,TCONTR        ;HOOG D.W.Z. OPDRACHT HEAD ADVANCE
        OUT     BTAG
        MVI     A,ZERO
        OUT     BTAG
        OUT     BFILE
        OUT     BMODS
        LDA     HEAD    ;HET HEAD ADRES IN HET RAM VAN DE 8080 WORDT
        ADI     1       ;MET 1 OPGEHOOGD.
        STA     HEAD
        RET
RSLRES: LDA     MODSL   ;ROUTINE "SELECT LOCK RESET"
        OUT     BMODS   ;SELECTEER MODULE
        MVI     A,SLRES
        OUT     BFILE   ;FILE BUS 1 + TAG HEAD WORDEN 2 SEC HOOG
        MVI     A,THEAD ;GEHOUDEN D.W.Z. OPDRACHT "RESET SELECTED LOCK"
        OUT     BTAG    ;(DE MODULE WORDT WEER BESTUURBAAR)
        MVI     H,20
        CALL    RDLXDS
```

Bijlage 7.

```
        MVI     A,ZERO
        OUT     BTAG
        OUT     BFILE
        OUT     BMODS
        RET
RBEREK: IN      BCAR    ;ROUTINE "BEREKEN HEAD + FORWARD LATCH,
        MOV     B,A     ;EN NONDIF"
        LDA     CYL
        SUB     B
        JNC     RBERH1
        SUI     1
        STA     NONDIF
        MVI     A,FORWLA
        STA     FORWL
        JMP     RBERH2
RBERH1: CMA
        STA     NONDIFF
        MVI     A,FORWLV
        STA     FORWL
RBERH2: LDA     FORWL
        MOV     B,A
        LDA     HEAD
        ORA     B
        STA     HEADFL
        RET
RSKRDY: LDA     MODSL   ;ROUTINE "SEEK READY"
        OUT     BMODS   ;SELECTEER MODULE
        IN      BCONT1
        ANI     BUSY    ;NIET BUSY ? ZO JA DAN JUMP TERUG NAAR RSKRDY
        JNZ     RSKRDY
        IN      BCONT2
        MOV     B,A
        LDA     GTATT   ;GATED ATTENTION HOOG ? NEE DAN JUMP TERUG
        ANA     B
        JZ      RSKRDY  ;NAAR RSKR
        IN      BCONT1
        ANI     SEEKIN  ;SEEK INCOMPLETE ? NEE GA DAN NAAR RSKH1
        JZ      RSKH1
RSKH0:  CALL    RSTAT
        CALL    RSLRES
        CALL    RRTZ
        MVI     A,ZERO
        OUT     BMODS
        RET
RSKH1:  IN      BCONT1
        ANI     UNSAFE
        JZ      RSKH0   ;UNSAFE ? JA DAN JUMP NAAR RSKH0
        CALL    RRESGA
        OUT     BMODS
        RET
RRESGA: LDA     MODSL   ;ROUTINE "RESET GATED ATTENTION"
        OUT     BMODS   ;SELECTEER MODULE
        MVI     A,READG
        OUT     BFILE   ;GATED ATTENTION WORDT GERESET D.M.V. READ
        MVI     A,TCONTR        ;GATE (BUS 1) + CONTROL TAG
        OUT     BTAG
```

Bijlage 8.

```
        MVI     A,ZERO
        OUT     BTAG
        OUT     BFILE
        OUT     BMODS
        RET
RREAD:  LDA     MODSL   ;ROUTINE "READ"
        OUT     BMODS   ;SELECTEER MODULE
        MVI     A,TCONTR
        OUT     BTAG
        MVI     A,SELHEA
        OUT     BFILE   ;SELECT HEAD OP FILE BUS
        MVI     A,SELHEAA OR READG
        OUT     BFILE   ;SELECT HEAD + READ GATE OP FILE BUS
        LXI     H,SREAD
        CALL    RDISPL  ;DISPLAY "FILE IS READING"
        EI
        HLT
        EI
        MVI     A,SELHEA        ;SELECT HEAD OP FILE BUS
        OUT     BFILE
        MVI     A,ZERO
        OUT     BFILE   ;FILE BUS WORDT NUL
        OUT     BTAG
        OUT     BMODS
        RET
RWRITE: LDA     MODSL   ;ROUTINE "WRITE"
        OUT     BMODS   ;SELECTEER MODULE
RWH2:   IN      BCONT2
        ANI     INDEX
        JZ      RWH2
RWH0:   IN      BCONT2
        ANI     INDEX
        JNZ     RWH0    ;WACHT OP INDEX
        MVI     H,1
        CALL    RDLXMS  ;1 MSEC DELAY
        MVI     A,TCONTR
        OUT     BTAG
        MVI     A,SELHEA OR WRITEG OR ERASEG    ;SELECT HEAD +
        OUT     BFILE   ;WRITE GATE + ERASE GATE OP FILE BUS
RWH1:   IN      BCONT2
        ANI     INDEX
        JZ      RWH1    ;WACHT OP INDEX
        MVI     A,SELHEA OR ERASEG
        OUT     BFILE
        NOP
        OUT     BFILE
        MVI     A,ZERO
        NOP
        OUT     BTAG
        OUT     BMODS
RBPMD1: MVI     A,MODSL1        ;ROUTINE "BEPAAL MODULE 1"
        STA     MODSL
        MVI     A,GTATT1
        STA     GTATT
        MVI     A,MD1SLD
        STA     MDSLD
```

Bijlage 9.

```
        RET
START:  CALL    RPWRON
        MVI     A,MODSL1
        OUT     BMODS
        RST     0
START2: CALL    RBEREK
        CALL    RSKCYL
        CALL    RSKRDY
        RST     0
START3: CALL    RRTZ
        RST     0
START4: CALL    RSLRES
        CALL    RRTZ
        RST     0
START5: CALL    RBEREK
        CALL    RSKCYL
        CALL    RSKRDY
        CALL    RRESGA
        CALL    RREAD
        RST     0
START6: CALL    RRTZ
        MVI     A,ZERO
        STA     CYL
ST6H0:  LDA     CYL
        INR     A
        MOV     B,A
        SUI     203
        MOV     A,B
        STA     CYL
        JNC     ST6H1
        CALL    RBEREK
        CALL    RSKCYL
        MVI     H,100
        CALL    RDLXMS
        CALL    RSKRDY
        JMP     ST6H0
ST6H1:  LDA     CYL
ST6H2:  DCR     A
        STA     CYL
        JZ      ST6H3
        CALL    RBEREK
        CALL    RSKCYL
        MVI     H,100
        CALL    RDLXMS
        CALL    RSKRDY
        JMP     ST6H1
ST6H3:  CALL    RRTZ
        RST     0
START7: CALL    RSTAT
        RST     0
START8: CALL    RWRITE
        RST     0
        END     START
```