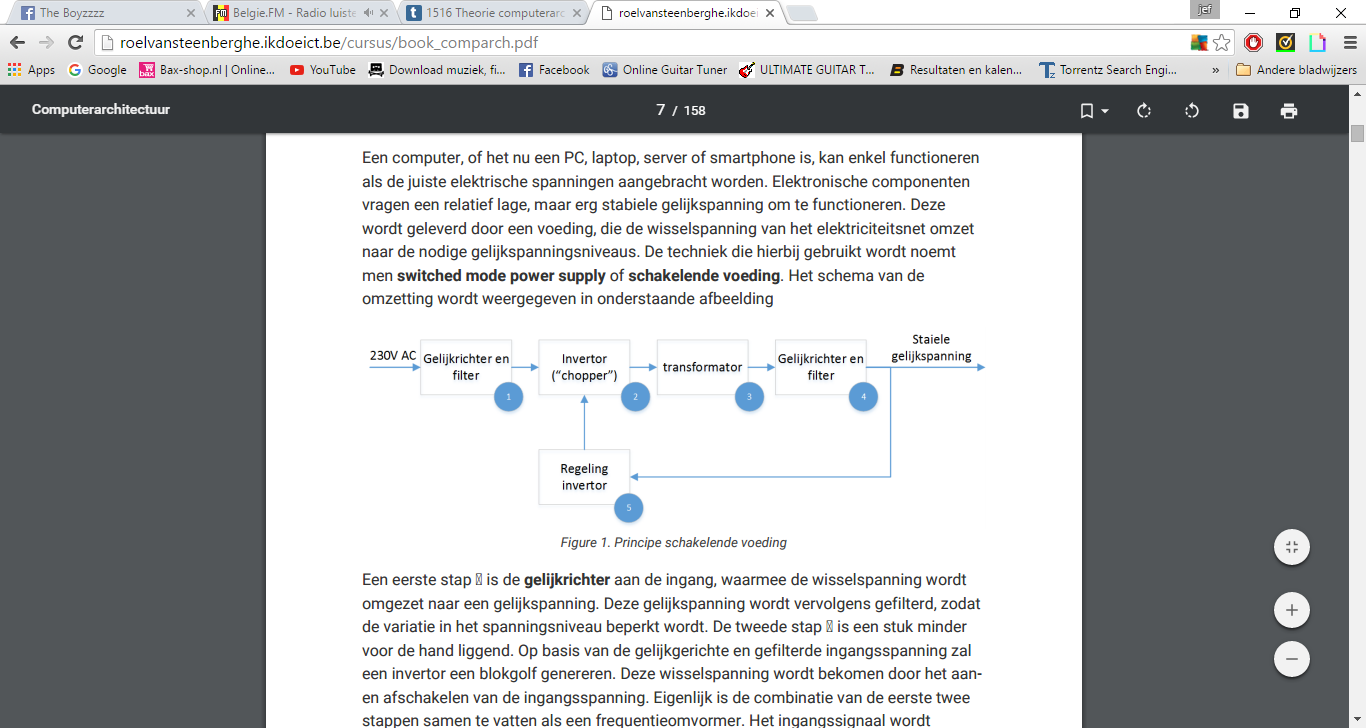
Computerarchitectuur

Chapter 4: computervoeding

## 4.1 principe en werking

Elektronische componenten hebben een lage spanning nodig. Deze wordt door de voeding geleverd die de netspanning omzet naar de gevraagde gelijkspanning. De techniek die hiervoor gebruikt wordt is **switched mode power supply** of **schakelende voeding**



De eerste stap is de gelijkrichter waar de wisselspanning wordt omgezet. Deze wordt dan gefilterd zodat de variatie in spanningsniveau beperkt wordt. De tweede stap genereert op basis van de gefilterde spanning een blokgolf die bekomen wordt door het af en aanschakelen van de ingangsspanning. De combo van die twee is een frequentie omvormer, van 50Hz naar 20kHz. Het voordeel hiervan is dat de volgende stap (transformator) kleiner kan gemaakt worden. De transo brengt de spanning naar het gewenste niveau, waarna het met de uitgangs-gelijkrichter en filter wordt omgezet naar een stabiele gelijkspanning.

## 4.2 Eigenschappen

4.2.1 Vormfactor en connectoren

Term vormfactor wordt gebruikt om de fysieke specificaties van het moederbord aan te geven (grootte, locatie van de gaten voor bevestiging…). De eerste PC beschikte over een at vormfactor en een daarbijhorende AT voeding. Ondertussen is al een opvolger, de ATX. Verschillen situeren zich op het vlak van spanning die de voeding kan leveren. De ATX heeft bijvoorbeeld een 3.3V spanning, 5V standby en power on signaal. Hierdoor kan de mechanische schakel van de AT vervangen worden door een elektronisch signaal. Het moederbord staat dus eigenlijk constant onder spanning zelfs al is je computer uitgeschakeld.

4.2.2 Vermogen

Belangrijke eigenschap is het vermogen. Het vergelijken ervan is complexer dan het kijken naar de waarden van de fabrikant. Belangrijker is de betekenis van dit getal.

4.2.3 Rendement

Groen aspect is belangrijk.

4.2.4 Geluid

Bron van lawaai: koeling en ventilator. Hoe groter de ventilator, hoe stiller die is. Sommige voedingen passief gekoeld, geen lawaai, maar duurder.

80-plus certificatieprogramma labelt voedingen adhv hun efficiëntie.

4.2.5 Problemen met voedingen

Oorzaken: te klein vermogen, onvoldoende koeling. Uit zich in het niet opstarten van het systeem, onverwachts afsluiten. Minder aangename gevolgen: bij beschadigde voeding: rook uit de computerkast. Slijtage aan de condensatoren (uitdrogen) zorgt ervoor dat de voeding rook ipv spanning produceert.

## 4.3 Accu’s

4.3.1 Eigenschappen

Capaciteit: uitgedrukt in Ah. 6Ah kan 1 ampere voor zes uur lang geven, 2 ampere voor drie uur… Sommige drukken de capaciteit uit in Wh. De stroom bereken je door de het vermogen te delen door de spanning.

Aantal cellen: accu is opgebouwd uit verschillende cellen. Zo kan een Li-ion accu cel 3V leveren.

Laadcurve: het is niet noodzakelijk zo dat een hogere stroom je accu sneller oplaadt. Gebruik van de juiste curve en adapter is belangrijk.

Memory-effect: capaciteit verliezen door halverwege het ontlaadt proces de accu opladen.

Li-ion accu’s: meest voorkomend type.  
Voordelen: Hoge energie dichtheid en geen memory-effect.   
Nadelen: degradatie, de capaciteit die doorheen de jaren sterk afneemt.

Toekomstige ontwikkelingen: druk om betere ontwikkelingen, de LiPo, 50% efficiënter

## 4.4 Uninterruptible PowerSupply (UPS)

UPS is een toestel dat het wegvallen van de netspanning kan opvangen. Hij bestaat uit een accu en een schakeling die de stroom in de accu omzet naar netspanning. De UPS neemt de netspanning tijdelijk over wanneer deze wegvalt (dit is afhankelijk van zijn capaciteit).

Types UPS:   
Online UPS: alle stroom die naar de IT apparaten gaat, gaat door de UPS. Hierdoor moet er niet worden overgeschakeld als de stroom wegvalt. Met een bypass kan je ook makkelijk defecten oplossen.

Offline UPS: bij het wegvallen word een bypass ingeschakeld. Dit proces duurt enkele milliseconden waar er geen uitgangsspanning is. De apparaten zitten ook rechtstreeks op de netspanning zodat als er storingen zijn op het net deze effect hebben op je IT apparatuur.

Line-interactive UPS: hybride oplossing. Het is een offline UPS waar de line-feed voorzien is van aanvullende filters. Hierdoor worden eventuele pieken en storingen weg gefilterd.

Chapter 5: CPU

## 5.1 Overzicht

## 5.2 Technologie en functionaliteit

Duidelijke evolutie: toename transistoren. Volgens de wet van Moore verloopt deze stijging zelfs exponentieel. Elke 24 maanden zou het aantal transistoren in een processor verdubbelen. Deze toename is enkel mogelijk als de transistordichtheid kan toenemen.   
Het toenemen van de transistoren zorgt voor het toenemen van de functionaliteit. Zo kent een x86 processor vanaf de 80286/80386 twee werkingsmodi:  
Real mode: hier heeft de processor dezelfde functionaliteit als de 8086, maar is deze sneller.  
Protected mode: extra functionaliteit. De naam protected komt door de toevoeging van bescherming op het vlak van geheugenbescherming. Deze modus ondersteund zo nog enkele onmisbare mogelijkheden zoals multitasking en virtueel geheugen. Een kanttekening hierbij is de processoren opstarten in real mode en dat het aan het besturingssysteem is om te schakelen naar protected mode.  
Andere voorbeelden: integratie van functies die voorheen door externe componenten werden vervuld, het toevoegen van cache geheugen.

## 5.3 Kloksnelheid

Tweede evolutie is die van de kloksnelheid. De processor wordt het snelste onderdeel van je computer. De trend naar een stijgende kloksnelheid is wel afgenomen. Het probleem daarbij is de warmte die moet afgevoerd worden en het verbruik.

## 5.4 Processorverbuik

5.4.1 Pipelining

Interne opbouw wordt verbeterd. Zo gaat de processor instructies niet één voor één, maar gedeeltelijk tegelijkertijd uitvoeren. Dit noemt men pipelining. Terwijl een instructie uit het geheugen wordt gehaald, kan een andere instructie al gedecodeerd worden en kan van nog een andere instructie het resultaat berekend worden.

5.4.3 Superscalaire processorden.

Als het pipelining wordt doorgedreven kunnen stappen die veel tijd in beslag nemen ontdubbeld worden. Verschillende eenheden kunnen verschillende berekeningen tegelijkertijd uitvoeren.

Multicore processoren zijn niet meer weg te denken. Ipv delen van de processor te ontdubbelen wordt een volledige rekeneenheid ontdubbeld. Moeilijkheid hierbij is om de caches op elkaar af te stellen. Deze methode is pas voordelig als meerdere cores tegelijk werk verrichten wanneer verschillende programma’s actief zijn of een programma gebruik maakt van verschillende threads.

5.4.3 Cache

Programmas worden steeds groter en de grootte van het geheugen is daarbij van belang. Zo is het beter om het geheugen te versnellen ipv de processor proberen nog rapper te maken.

## 5.5 APU, SoC

Men probeert functionaliteit die voorheen op verschillende plaatsen op het moederbord zaten, te combineren in 1 chip. Voordelen:  
- Het aantal verschillende chips kan worden teruggedrongen  
- De componenten zitten dichter bij elkaar dus zijn er geen trage bussen nodig tussen onderdelen  
-De oppervlakte nodig voor het bouwen verkleint, belangrijk voor mobile devices

Dit proces gaat zo ver dat je kan spreken van System on a Chip: alle belangrijke onderdelen op 1 chip.

## 5.6 Montage

Enkele zaken die in acht moeten genomen worden:  
- processor moet compatibel zijn met het moederbord.  
- Het plaatsen moet gebeuren zonder uitoefenen van kracht.

Tweede aandachtspunt: voldoende koeling. Dit betekent dat moet gezorgd worden voor een goede koeling en ventilator.

## 5.7 Processoren van de toekomst

Er is een kaper op de kust voor de x86 processoren: de ARM-architectuur gebaseerd op RISC. Hier vraagt elke instructie, ongeacht de complexiteit, evenveel tijd.

De kans dat dit op de desktopmarkt op korte termijn succesvol gaat worden is klein en ook omgekeerd dat de CISC op mobile devices komt.

CHAPTER 6: Het geheugen

## 6.1 Overzicht

Zowel uit te voeren instructies als opgeslagen data zitten in het geheugen. Dit legt twee belangrijke behoeften bloot: snelheid en grootte.  
Snelheid is belangrijk omdat het geheugen de processor moet voorzien van instructies. Snelle processor met traag geheugen=traag systeem.  
Grootte: zowel programmas als data worden opgeslaan in het geheugen. Ook voor multitasking

Geheugenpiramide: hoeveelheid geheugen neemt af naarmate je stijgt op de piramide, prijs per byte neemt toe naarmate je hoger gaat op de piramide, snelheid neemt toe naarmate je meer zakt in de piramide.

## 6.2 lokaliteitprincipe.

Onderscheid tussen plaatsgebonden en tijdsgebonden lokaliteit.   
tijdsgebonden: als een item gebruikt word, is de kans dat je het binnenkort terug nodig hebt.  
plaatsgebonden: als je een item gebruikt is de kans groot dat je binnenkort iets nodig hebt dat in de buurt komt.

## 6.3 Soorten geheugen

6.3.1 Behuizing

Meest tastbare onderscheid=behuizing. Origineel waren dit discrete chips, maar omdat de capaciteit steeg, werd dit te duur en ging men over op geheugenmodules.

Geheugenbanken op een moederbord bestaan uit één of meerdere sockets of geheugenvoeten. Het aantal hangt af van de uitvoeringsvorm van het gebruikte geheugen. Niet alle geheugenbanken moeten gevuld zijn, maar iedere geheugenbank waarin geheugen wordt geplaatst moet volledig gevuld zijn.

Langs beide zijden bevinden zich contactpunten aan de geheugenmodule. Indien ze inwendig verbonden zijn, spreekt men van SIMM (Single Inline Memory Module). Als ze afzonderlijk werken spreekt men van DIMM (Dual Inline Memory Module). DIMM bied op dezelfde afstand meer contactpunten en wordt toegepast in moderne modules.

## 6.4 Technologie

Een ander onderscheid is in de technologie die gebruikt wordt voor de geheugencellen op te bouwen: statisch of dynamisch. Statisch bestaat uit actieve cellen. Ze zijn complexer bij het IC ontwerp en dus duur, maar snel. Dynamisch bestaat uit een condensator. Als je over een C gelijkspanning zet en die wegneemt kan je meten hoeveel erover stond., dit is een soort van geheugen. Nadelen: leesoperatie zal de C ontladen en is dus destructief. Ten tweede bestaan er geen perfecte C’s en vertoont dit geheugen een lek. Mettertijd gaat dus geheugen verloren, tenzij dit ververst/gerefreshed wordt. Statisch is sneller en kan stroom doorsturen of opnemen, dynamisch kan dit niet, het is een C die opgeladen of ontladen is.

## 6.5 DRAM technologie

6.5.1 Gemultiplexte adresklemmen

Dynamische RAM’s hebben vanwege de grote densiteit een grote capaciteit op de chip. Dit betekent dat heel wat adressignalen nodig zijn om een cel te selecteren. Als elk signaal een pin heeft, wil dit zeggen dat een grootte behuizing nodig is. Om dit te voorkomen gebruikt men gemultiplexte adreslijnen: het adres wordt opgesplitst in een column adres en een row adres. Deze worden de één na de ander aangeboden aan de adresklemmen van het IC.

6.5.2 Destructieve leescyclus

Oplossing: als data gelezen is, wordt dezelfde data nadien terug weggeschreven, zodat de originele toestand hersteld wordt. Dit moet in de geheugenchip zelf geregeld worden. Het geheugen is als een matrix opgebouwd uit rijen en kolommen. Naast deze matrix van dynamische cellen is eer ook een rij statische. Op het ogenblik dat het rijadres wordt aangelegd, wordt de dynamische rij naar de statische gekopieerd. De dynamische verliest haar inhoud, de gewenste cel kan gelezen worden en de inhoud van de statische rij wordt terug gekopieerd naar de dynamische.

6.5.3 Refresh

Op geregelde tijdstippen wordt een RAS-only cyclus uitgevoerd. Hij past het zelfde principe toe van het herschrijven van de dynamische naar de statische rij. Op deze manier is de hele rij gerefreshed.

6.5.4 Bandbreedte bij DRAM

DRAM heeft de eig om te werken met cyclussen. Bereken van de bandbreedte gebeurt met volgende formule: aantal Bytes die getransformeerd worden bij een cyclus/tijdsduur van een cyclus.

## 6.6 Fast Page DRAM (FP-DRAM)

Voordeel gemultiplexte adresklemmen: minder adresklemmen, nadeel: schrijfcyclus is langer. FPDRAM gaat cycli combineren. FPDRAM geeft eenmaal een rij-adres en selecteert vervolgens een kolom om te lezen of te schrijven. Onmiddellijk hierna wordt een tweede kolom geselecteerd om te lezen of schrijven , daarna een derde…. Een aantal cycli worden vermeden. Id praktijk wordt steed gewerkt met een burst van 4 cycli.

## 6.7 EDO RAM

Extended Data Out-RAM= aanpassing FP concept. Bij EDO-RAM blijven de data op de uitgangen van het geheugen langer beschikbaar, waardoor men tijd wint: terwijl data gelezen wordt, kan het volgende kolom adres al gegeven worden

## 6.8 Synchronous DRAM

Bij SDRAM gaat men een stap verder op het lokaliteitsprincipe. Men gaat er van uit de opeenvolgende kolommen uitgelezen zullen worden. De cyclus bestaat nu uit het aanleggen van een rij-adres en vervolgens het uitlezen van een aantal opeenvolgende kolommen.

6.8.1 DDR SDRAM –DDR2 –DDR3

Principieel werkt DDR op dezelfde manier als SDRAM. Verschil zit in het tempo waarop dit gebeurt. Bij Double Data Rate wordt datta naar buiten gebracht op stijgende en dalende flank van de klok. Hiervoor worden gebruik gemaakt van een prefetch buffer. In elke cyclus worden twee bits getranfereerd naar het prefetch buffer. Voordeel is een hogere maximale bandbreedte. Nadeel is dat de tijd die nodig is voor het aanleggen van de adressen en het naar buiten brengen van de data groter is.

Tcl = CAS lateny time: tijd tussen CAS en beschikbaar worden van de data  
Trcd = DRAM RAS of CAS delay tijd: tijd tussen RAS en CAS (ook tussen active en read/write commando)  
Trp= DRAM Ras precharge = tijd tussen selecteren van twee rijen  
Tras = precharge delay = minimale tijd tussen actief worden en precharge volgende rij.

Aantal cellen dat in een burst gelezen wordt kan variëren. Geheugen kan bv in een burst 8 transfers van 8 bytes doen. Tras is in het verhaal nog niet naar voor gekomen. Tras bepaalt de tijd waarin volgende rij nog niet mag gelezen worden. Deze moet groot genoeg zijn om de buffer niet te overschrijden, want anders gaat er data verloren. D Tras moet dus minimaal Trcd + Tcl + 1 zijn. Opvolgers van DDR zijn DDR2 en 3. Behalve op het vlak van kloksnelheid en spanningen (DDR2 en 3 gebruiken lagere spanningen) is de prefetch buffer vergroot, 4 bits bij DDR2 en 8 bij DDR3. Bij DDR4 verdubbelt het niet maar is de nieuwe spec het 3D geheugen. Meerdere lagen geheugen worden op elkaar met als gevolg dat het geheugen data sneller naar buiten kan brengen.

## 6.9 Optimalisatietechnieken.

Evolutie DRAM tech = verbeteren van de maximale bandbreedte.

6.9.1 Interleaving

Gebruikt om dode tijd te vermijden, nl de tijd tussen 2 rijen. Dit kan vermeden worden indien de volgende operatie doorgaat op een andere geheugenmodule. Hiervoor worden naburige geheugenblokken verdeeld over afhankelijke aanspreekbare banken die onafhankelijk van elkaar aangesproken kunnen worden. Bel Opm: enkel snelheidswinst als er tussen banken gewisseld kan worden. Memory controller moet weten of er tss banken gewisseld kan worden zodat hij al dan niet rekening moet houden met de hersteltijd.

6.9.2 Dual channel

Techniek waarbij de snelheid van de databus wordt vergroot. Dit brengt enkele problemen met zich mee: 1) er moeten extra aansluitingen voorzien worden op de geheugenmodule en het moederbord. 2) bij toenemende kloksnelheid kunnen er tijdverschillen komen tss datalijnen.   
Dual channel is een techniek die de datasnelheid wil verhogen door de databus naar de memory controller te verdubbelen zonder de databus van de geheugenmodule te vergroten. Dit gebeurt door het gebruik maken van versch geheugenbanken. Elke bank heeft een databus van 64 bit, maar aangezien die niet samenvallen is er een 128bit databus naar de memory controller. De DIMM sockets op het moederbord zijn dus fysiek verbonden met een van de twee 64 bit kanalen en het is dus belangrijk om de RAM in de juiste sockets op het moederbord te steken (color coded meestal).

6.9.3 Buffered/registered RAM en foutdetectie

Veelvoorkomende termen: buffered en registered: zelfde betekenis = aanwezigheid van extra bufferchips die dienst doen als elektrische buffer/versterker tss geheugen IC’s en rest v/h systeem.

ECC en parity:  
Parity foutdetectie: per byte wordt een pariteitsbit berekend. Deze wordt mee verzonden. De ontvanger herkent die en vergelijkt zijn berekeningen met de pariteitsbit. Bij een verschil is er een fout geweest.  
ECC werkt op een gelijkaardige manier, maar maakt gebruik van de hash functie, meerder fouten per keer detecteren of enkelvoudige bitfouten corrigeren. Deze manier kost rekenkracht en dus tijd.

## 6.10 Cache geheugen

Instructies voor snelle processor zitten in het trage geheugen. Nadeel statisch geheugen is dat het duur is, dus de hoeveelheid in een PC is schaars en het komt erop aan deze zo efficiënt mogelijk te gebruiken.

6.10.1 Werking

Cache is een buffer tss geheugen en cpu, de nodige instructies zitten in het cache geheugen klaar. lokaliteitsprincipe wordt toegepast -> cache geheugen en hoofdgeheugen worden verdeeld in gelijke blokken blokken in het cache geheugen kunnen kopie bevatten van een blok uit het hoofdgeheugen.   
Leescyclus: 1) processor vraagt adres  
 2) adres is aanwezig in cache = cache-hit, snellere leescyclus  
 3) cache-miss = via trage leescyclus wordt het gevraagde woord uit het hoofdgeheugen gehaald en onmiddellijk ook de hele blok naar de cache gekopieerd.

Parameter voor snelheid van cache is dus de hitrate = percentage geheugentoegangen via het snelle cache geheugen (tussen 80 en 90 %)   
bij een schrijfcyclus worden gegevens aangepast: bij een cache hit wordt de inhoud van het cache geheugen aangepast en pas later dat van het hoofdgeheugen = write-back cache  
alternatief = write-through cache = gelijktijdig aanpassen met als nadeel dat het trage hoofdgeheugen gebruikt wordt.

6.10.2 Soorten caches

L1 & L2 l1 licht het dichts bij de processor, is het snelst  
snelheid processor neemt toe waardoor er een buffer nodig is, l2 cache = minder snel, maar groter.

6.10.3 Overschrijfstrategieën

bij een cache miss moet er in het geheugen plaats komen, een lijn moet worden verwijdert.

First in First out: (FiFo) degene die het langst in de cache. Hitrate blijft hetzelfde.  
Least Recently Used: (LRU) lijn die het langst niet meer gebruikt is. Hitrate neemt toe, nadeel: per lijn moet bijgehouden worden hoeveel die gebruikt wordt.  
Least Frequently Used: (LFU) frequentieel minst gebruikte, gelijkaardig aan LRU  
Adaptive replacement cache: (ARC) combineert LRU en LFU. Resultaten, maar ook de complexiteit nemen toe.  
Random: willekeurige lijn, soms gecombineerd met LRU.

## 6.11 Associativiteit

De associativiteit van het cache geheugen bepaalt welk geheugenblok op welke plaats in de cache komt en bepaald dus zo mee de hitrate

1/ Fully Associative Cache

Bij FAC kan een blok uit het geheugen in elke cacheline. Er wordt een tag gekoppeld aan de cacheline om te kunnen bepalne of het blok aanwezig is in het cache en zo de juiste cacheline te kunnen selecteren.

Voordeel: De cache kan optimaal benut worden aangezien er geen cachelines worden opengelaten. Nadeel: vrij duur + er zijn ook snelle comparatoren nodig om binnen de tijd van 1 cyclus te zien of er een cache hit is.

2/ Direct Mapped Cache

Bij DMC kan elk geheugenblok slechts in 1 cacheline. Overschrijfstrategieën zijn overbodig aangezien er maar 1 plaats is voor het geheugenblok. Geen keuze -> optimale keuze wordt niet gemaakt. Maar doordat een geheugenblok maar op 1 lijn terecht kan, worden tags kleiner en comparatoren ook. Dit kan gebruikt worden waar snelheid niet zo belangrijk is bv cache die verder van processor staan.

3/ Set-associative cache

Dit is de tussenvorm. Cachelines worden gegroepeerd in sets. Elk geheugenblok kan maar terecht in 1 set, maar binnen die set kan het in elke cacheline.

4/ Snelheid vd cache

1. Coldstart misses: zijn afkomstig van de eerste keer dat toegang tot een blok wordt gezocht.
2. Hangt samen met capaciteit, hoe groter, hoe trager.
3. Associativiteit. Hoe hoger, hoe minder missers.
4. Hit time: de tijd nodig om bij een hit gegevens op te halen
5. Miss time: De tijd nodig om bij een miss het nieuwe geheugenblok te laden en de gevraagde gegevens beschikbaar te maken
6. Miss penalty; de extra tijd nodig bij een cache-miss
7. Hit rate: Percentage toegang die rechtstreeks langs cache gaan.

T-totaal= HitRate L1 x tL1 + (1- HR L1) x HR L2 x tL2 + (1- HR L1) x ( 1 - HR L2 ) x t RAM

Ander voorbeeld: Grotere cachelines -> hogere hit rate omdat meer naburige gegevens ->

minder cachelines bij zelfde geheugen -> sneller lijnen verwijderd uit cacheline.

Groter cachegeheugen -> meer cachelines -> selectie van lijn wordt complexer -> miss time neemt toe .

6.12 Virtueel geheugen.

Misschien dat ik hier later nog iets deftig inzet maar tzag er te moeilijk uit dus dit effe overslagen zodat de rest gekend is.

7. Opslagmedia

7.1 Harde schijf

1/ Fysieke opbouw

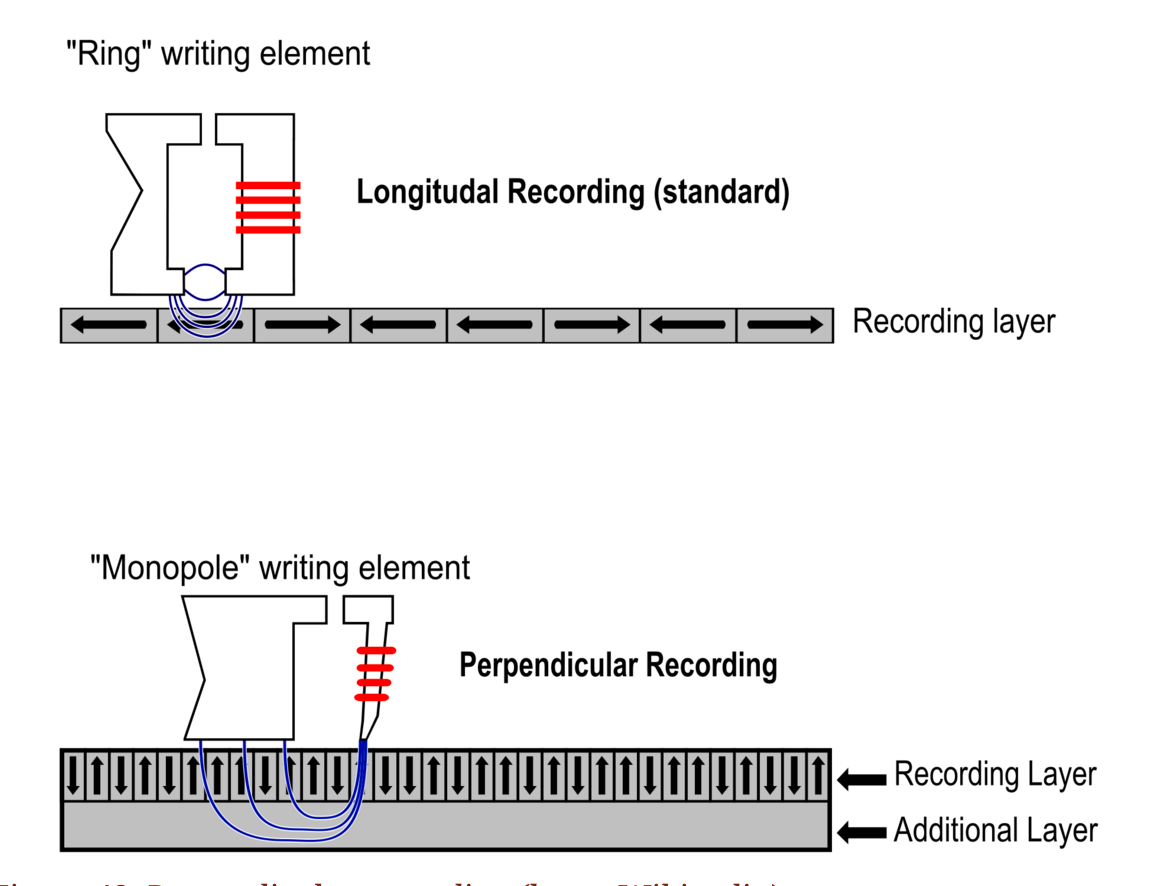
Platter van aluminium met magnetisch materiaal + hoofden om te schrijven/ lezen.

Kleinste hoeveelheid data dat kop in 1 keer kan verwerken = sector. Een track bestaat uit verschillende sectoren met tussenruimte tss sectoren. 1 sector kan gelezen worden zonder de kop te verplaatsen.

**SECTOREN/TRACKS**

Een sector begint met een preamble(patroon waardoor het begin van sector wordt aangegeven).

Daarna volgen de databits( meestal 4096 - Advanced Format ), en na de databits volgt de ECC(Error Correcting Code). De verdeling van de schijf in sectoren enz gebeurt tijdens het formatteren, hierbij gaat dus al een stuk van de nuttige ruimte weg aan tussenruimtes, preamble enz…

Omdat de dichtheden van een spoor stilaan niet meer kleiner kan worden, gaan ze id diepte gaan werken ipv in de tangentiële richting. (Perpendicular recording)

**TOEGANGSTIJD**

Snelheid hangt af van de seek( de schijfkop moet boven het juiste spoor gebracht worden)

-> Track to track ( aangrenzende sporen) en Average (van het een naar ander willekeurig spoor).

Eens de kop boven het juiste spoor is, is er nog de rotatiewachttijd. Average= halve rotatie.

Tenslotte is er de overdrachtssnelheid.

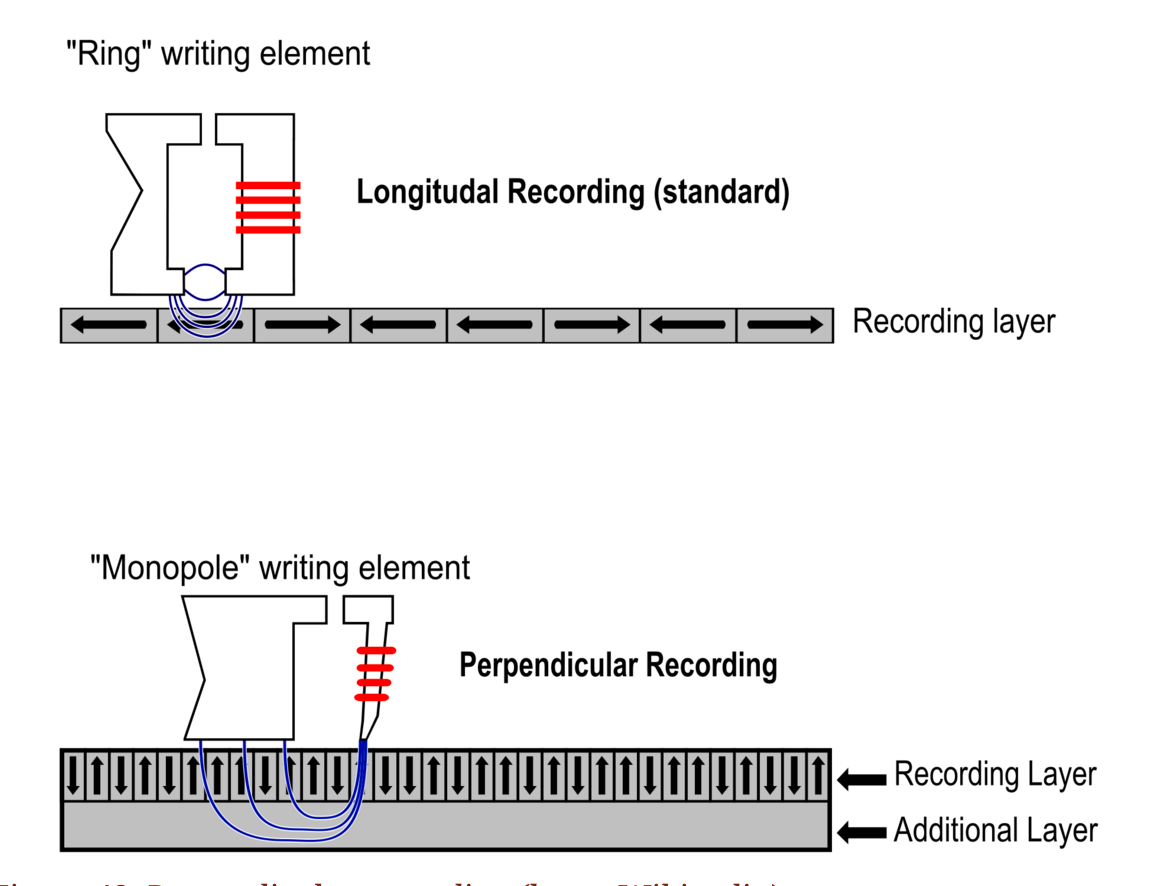
**INVLOED VAN FYSIEKE GEOMETRIE OP PERFORMANTIE**

Rotatiesnelheid: hoe sneller, hoe kleiner de average latency MAAR meer energie.

Meet platters: meer gegevens beschikbaar MAAR motoren moeten krachtiger zijn

Grotere bitdichtheid: snellere bits onder de kop dus sneller lezen en schrijven

2/ Adressering

Een sector wordt gekenmerkt door 1) kant van platter

2) een cilinder

3) een sector op het spoor

Origineel CHS-adressering: tien bits voor cilinder, vier voor kop en 6 voor sector.

Nadeel: slechts 20 bits beschikbaar voor adres ve sector.

2de nadeel: adressering direct gelinkt aan geometrie van harde schijf, kan dus niet volledig geadresseerd worden.

Ander soort adressering: Logical Block Addressing (LBA), hierbij heeft een sector gewoon een volgnummer, dan dat op de schijf zelf word omgezet naar fysiek CHS adres.

3/ Scheduling

Tijd nodig om sector te lezen: 1) seek time 2) latency time 3) transfer time.

Om totale tijd te beperken zal men sectoren niet 1 per 1 lezen maar in blokken of clusters.

Hoe groter de blokken, hoe efficiënter de gegevensoverdracht, maar ook hoe groter de interne fragmentatie.

Als schijf zwaar belast ordt en er versch schijfoperaties staant e wachten kunnen we volgorde veranderen om de prestaties vd schijf te verbeteren.

**FCFS: First Come First Serve**

Dit is het eenvoudigste, maar kan niet beste prestaties garanderen. Hierbij gaat veel tijd verloren naar het zoeken vd sporen.

**SSTF: Shortest seek time first**

Voorrang aan operaties die zich het dichtst bij de kop bevinden. Geen efficiënt algoritme

**SCAN: Scannen**

Overloopt schijf van eerste tot laatste spoor en voert hierbij alle tussenliggende operaties uit.

**C-SCAN: Circulair Scannen**

Zelfde als scannen maar op de terugweg gaat deze geen operaties bevinden, zodat gemiddeld oudere aanvragen sneller bediend zullen worden. Eerlijker, maar slechtere doorvoercapaciteit.

7.2: Solid State Drive

Ipv magnetische schijf wordt gebruik gemaakt van DRAM of FLASH.

DRAM verliest gegevens bij uitvallen stroom -> extra batterij nodig.

Flash niet, maar is wel duurder.

Voordelen: Schijf draait niet, willekeurige toegang sneller, minder energie en lawaai.

1/ Flash technologie

Flash geheugen bestaat uit cellen die in staat zijn een spanningsniveau te onthouden.

2 belangerijke types:

**SLC**

Bij SLC zal elke cel 1 bit bevatten. Meest eenvoudig+ snel maar duur; veel cellen nodig voor grote capaciteit.

**MLC**

Multi level cell: Meerdere bits per cell(vier verschillende spanningsniveaus).

MLC gevoeliger voor fouten: vandaar trager gelezen en beschreven. Voornamelijk gebruikt waar grote capaciteit belangrijker is dan snelheid of betrouwbaarheid.

2/ Schrijfcyclus van Flash Memory

Beschrijven flash heel wat moeilijker dan lezen.

Beschrijven: Huidige data wordt in de snelle cache gelezen; huidige en de te schrijven data worden samengevoegd en de waardeloze cellen leeggemaakt; cache wordt weggeschreven naar een (leeg!) block in flash.

3/ Optimalisatie van FLASH

SSD-technologie nog niet perfect; enkele problemen:

**Buffering**

Bijzonderheid van flash: voor schrijven moet je benodigde blokken wissen. Een cache zal dan ook schrijfacties bufferen. Als de cache te klein is, zal na een periode van continu schrijven de prestatie sterk degraderen.

**Schijfverlamming**

Schrijven werkt enkel snel als dat gebeurt naar volledig vrije blokken, anders moeten deze eerst ingelezen/leeggemaakt worden. Naarmate tijd vordert zijn deze blokken zeldzaam.

Oplossing: Garbage collection; Bij rustige momenten gaat schijf zelf blokken samenvoegen zodat er terug lege blokken bijkomen.

Andere oplossing: ATA-Trim commando. Het besturingssysteem zal hierbij zeggen welke data bv door NTFS als verwijderd staat aangeduid). Zo wordt vermeden dat sectoren verplaatst worden die allang verwijderd zouden moeten zijn.

7.3 Logische structuur van een opslagmedium

1/ Boot block

Opstarten systeem = ‘ bootstrappen ‘; Na het aanleggen vd spanning begint processor code uit te voeren op een bep vaste plaats in het geheugen. Door op die plaats een ROM-geheugen aan te brengen, zal een programma beginnen uitvoeren. Dit gaat na of de hardware zich gedraagt zoals het hoort, en gaat dan op zoek naar de systeemschijf en probeert 1 of meerdere sectoren te lezen

Deze bevatten de code die nodig os om andere delen vh besturingssysteem in te lezen. Deze zal op zijn beurt het volledige besturingssysteem inladen, de configuratiebestanden lezen en de controle overleveren aan het besturingssysteem.

2/ Master Boot Record ( MBR ) Layout

Het MBR is de eerste sector vd harde schijf bestaande uit de primarie partitietabel en de MBR-programmacode. Deze code zegt welke primaire partitie opgestart moet worden, om de bootsector van die partitie in het geheugen te laden on om te starten met het laden van de OS. Soms kan de MBR-code ook interactie met de gebruiker mogelijk maken ( denk aan de Linux Loader ).

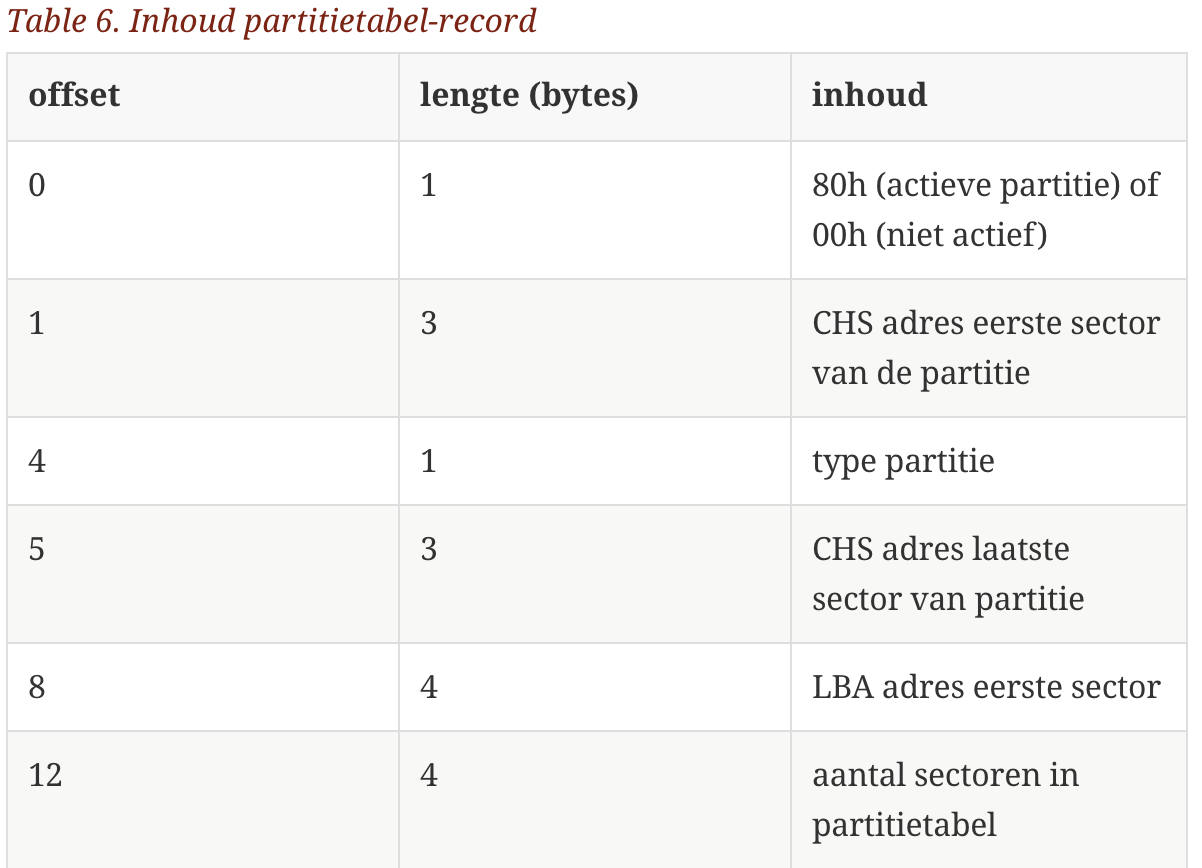
Partitionering is het onderverdelen van de harde schijf in verschillende blokken, met elk een bestandsysteem. Dit kan handig zijn om verschillende soorten gegevens te groeperen of om bv meerdere besturingssystemen mogelijk te maken. Eens de partities gekozen zijn kan je achteraf de groottes niet altijd meer aanpassen, want dit is gevaarlijk voor dataverlies.

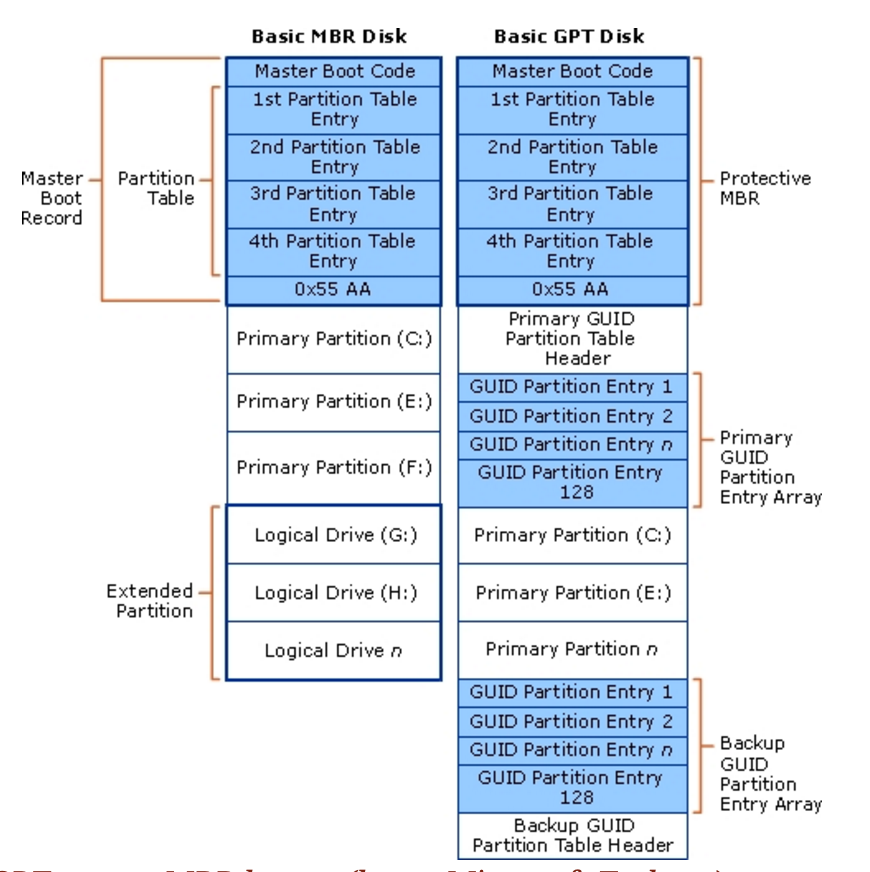
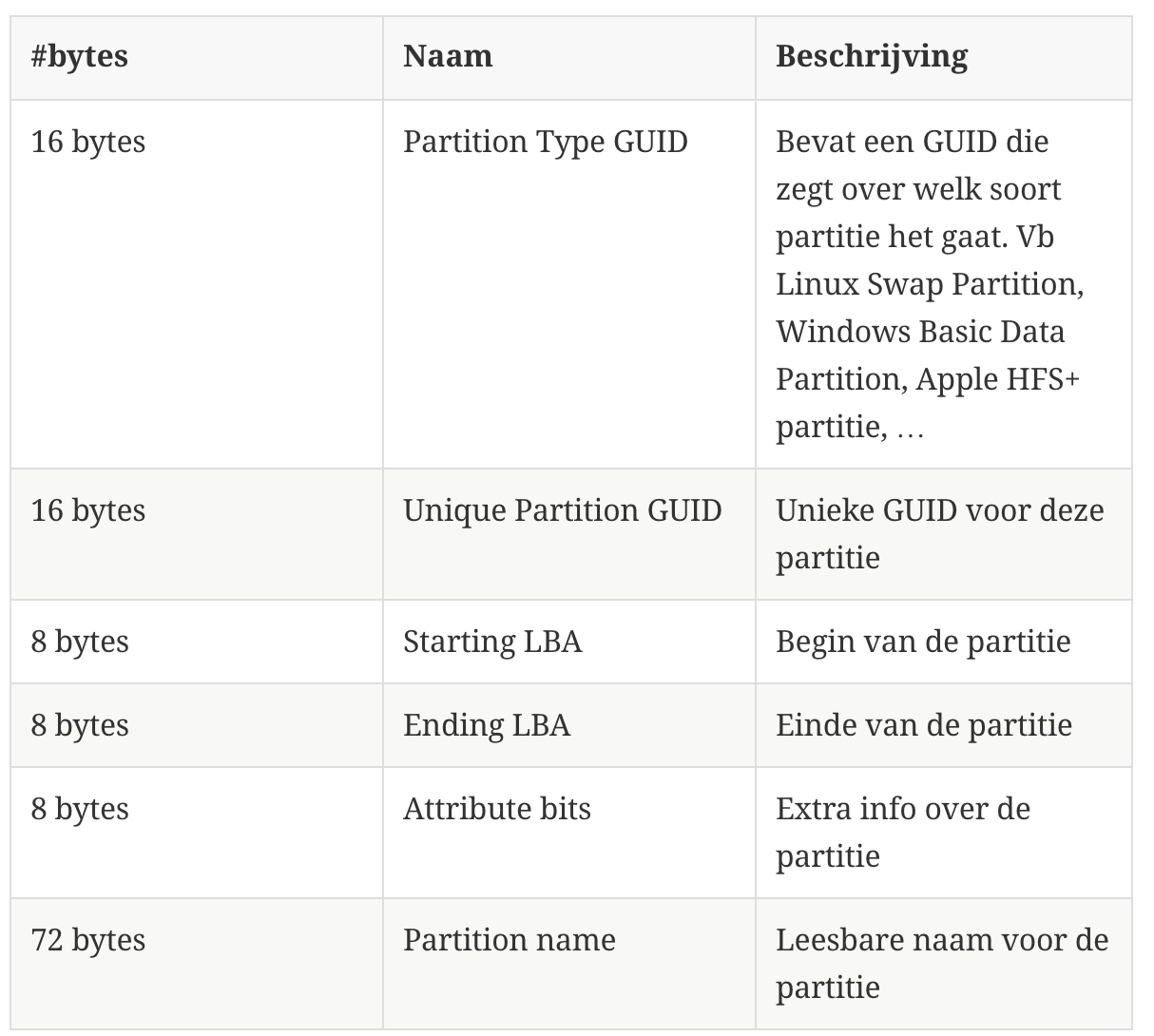
De primaire partitietabel bevat voor elke primaire partitie een record van zestien bytes.

Eerst is er een byte die zegt of de partitie actief is of niet. Daarnaast zijn er parameters die de locatie en grootte vd partitie zeggen en 1 byte dat het type vastlegt.

Er zijn maar 4 records in primaire partitietabel, dus er kunnen slechts vier primaire partities gedefinieerd worden. indien er meer nodig zijn moet er gebruik worden gemaakt van een extended partitie. Deze kan men vervolgens in logische partities definiËren. Je moet wel goed opletten dat je voldoende ruimte voorziet voor het definiëren van alle logische partities.

3/ GPT layour

Disk-layout met MBR wordt nog vaak gebruikt. Nadelen : partitiegrootte beperkt tot 2TB, en de partitietabel word op geen enkele manier beschermd. Als deze cluster defct is, dan is het moeilijk de logische layout van de schijf te achterhalen.



Oplossing: GPT layout.

De partitietabel staat op het einde vd schijf, zodat eend efect ih begin vd schijf niet hoeft te betekenen dat je de data op de partities kwijt bent.

De eerste sector van een GPT schijf bevat een valse MBR record om oude partitioneringstools te misleiden. Maximale partities: 128 stuks. Geen limiet voor grootte per partitie.

7.4 Bestandssystemen

1/ Algemeen

Een bestandssysteem bestaat uit verschillende lagen.

1. Het logische bestandsysteem. (Waarmee de gebruiker in contact komt)
2. De bestandsorganisatie. (vormt de verbinding tss het logische en het fysieke niveau)
3. Het fysieke bestandssysteem. (krijgt de logische schijfadressen binnen en vertaalt deze naar fysieke schijfadressen voor de desbetreffende schijven)
4. IO-controle. (de drivers voor de schijven, deze verbergen alle details vd schijfhardware voor de bovenliggende lagen)

Een bestandssysteem zal bestanden op een partitie organiseren en zorgen dat de OS over voldoende informatie beschikt om elk bestand terug te vinden. Er zijn veel verschillende soorten bestandssystemen, wij beperken ons tot FAT16 en NTFS.

Elke partitie begint met een PBS (Partition Boot Sector). Dit is de eerste fysieke sector die bij een partitie behoort. Deze bestaat op zijn beurt uit 2 delen. Helemaal vooraan staat de spronginstructie naar de eventuele aanwezige boot routine. Deze boot zal het OS laden.

Daarnaast is er het BIOS parameter block. Dit gedeelte bevat een aantal parameters die belangrijk zijn om toegang te krijgen tot het bestandsysteem.

2/ Bestanden

Hoe plaats je de clusters waaruit zo’n bestand bestaat op de schijf? Verschillende mogelijkheden.

**Contigue allocatie**

Eenvoudigst. Een bestand (n blokken) zal n opeenvolgende fysieke blokken op schijf innemen. Verplaatsingen lees- en schrijfkop minimaal -> snelste allocatiemethode.

Nadelen: 1) probleem externe fragmentatie: men heeft in werkelijkheid stukken geheugen van ongelijke lengte waarvoor het geheugenbeheer relatief moeilijk is.

2) Voor de creatie ve bestand moet men weten hoe groot het zal worden. Schat men dit te hoog, krijgt men een aanzienlijke interne fragmentatie. Schat men dit te laag, zal het programma dat het bestand aanmaakt afgebroken worden. Dit laatste kan vermeden worden door het bestand te kopiëren naar een grotere vrije ruimte en daar de uitvoering verder te zetten.

**Gelinkte allocatie**

Ipv alle blokken contigu op de schijt te zetten worden de blokken op de schijf met elkaar gelinkt, zodat (als er vrije blokken zijn) een bestand kan blijven groeien.

Nadelen: 1) Directe toegang onmogelijk omdat steeds de lijst vd blokken overlopen moet worden.

2) Sequentieel lezen ve bestand wordt zeer traag omdat er in principe per blok een verplaatsing van de kop kan nodig zijn.

3) Blokken zijn íets kleiner omdat de link naar het volgende blok ook opgenomen moet worden.

4) 1 corrupt blok kan de hele schijf inconsistent maken.

Men kan toch proberen om bij de allocatie vd blokken deze zoveel mogelijk sequentieel op te slaan, en men kan ook de blokgrootte vergroten zodat het percentage aan links kleiner wordt.

**Allocatietabel**

Overige problemen kunnen opgelost worden door alle wijzers (links) naar de blokken bij te houden in afzonderlijke blokken, de zogenaamde FAT ( File Allocation Table ).

Voordelen : 1) Om een bestand te zoeken moet men slechts een paar blokken van de schijf te lezen (de FAT).

2) Er moeten geen links in de gegevensblokken opgeslagen worden

3) Men kan de FAT meer dan eens opslaan om beperkte schijfdefecten op te vangen.

Vrije blokken worden teruggevonden door een speciaal teken op de plaats vd wijzer in de FAT.

Meestal bewaart men kopie vd FAT ih geheugen om snel een blok te kunnen terug te vinden. Indien de FAT niet in geheugen bijgehouden wordt, zal het telkens heen en terug swingen tussen de FAT blokken en de werkelijke gegevens en een aanzienlijke extra belasting voor de schijf vormen.

**Indextabel**

3/ Mappen (directories)

Er zijn relatieve (tov de huidige directory) en absolute(ten opzichte van de root-director) padnamen…

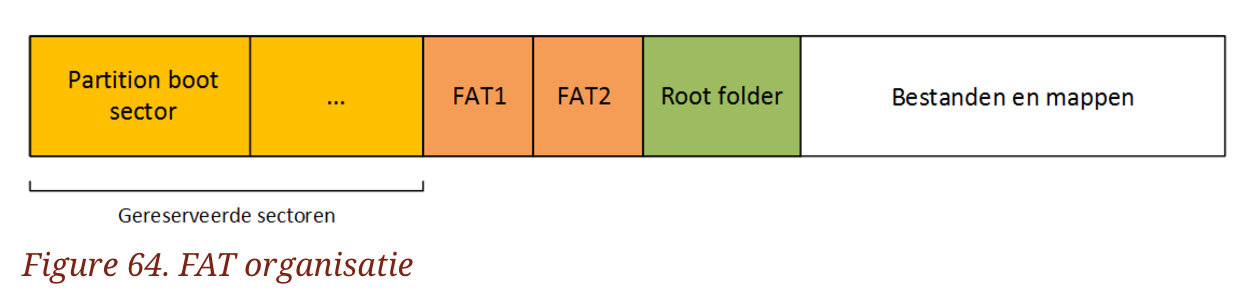
Fragmentatie van de schijf : Doordat er bestand in stukken worden opgedeeld en in vrije blokken worden geplaatst, kan het zijn dat de kop vd schijf zich verschillende keren zal moeten verplaatsen om 1 bestand te lezen, en de prestatie van het bestandssysteem zal sterk dalen .

Defragmentatie en compactering: De bestand op de schijf herschikken zodat alle bestanden contigu worden gemaakt en ook de vrije ruimte wordt samengebracht.

Voordeel: De fragmentatie van bestanden wordt weggewerkt -> nuttig voor prestatie.

Nadeel: Door de compactie wordt de volgorde van bestanden gewijzigd -> slecht voor prestatie.

**FAT FILE SYSTEM**

Een FAT-partitie bestaat steeds uit hetzelfde.

**NTFS**

Ntfs maakt geen gebruik van een FAT maar van een MFT(Master File Table). Dit is nog uitgebreider dan de root folder van een FAT systeem.

4/ Bestanden wissen

Aan de snelheid waaraan er gewist wordt zie je dat niet elke byte op de schijf gewist wordt. In de plaats daarvan zal de verwijzing in het bestandsysteem worden aangepast. In de FAT tabel zal bv de cluster als ongebruikt worden gemarkeerd door een speciaal karakter. In dit geval is het nog mogelijk om terug te vinden, tenzij het gefragmenteerd is. In dit geval is de (gewiste) FAT tabel nodig om de juiste clusters en hun volgorde terug te vinden.

Bij NTFS wordt enkel het MTF record aangepast, waarin een parameter zal aangeven dat het bestand gewist is.

8. Opstartroutine

8.1 Geheugens voor opstartroutine

Een processor spreekt bij het opstarten enkel met geheugen. Om de OS te laden is er dus een tussenstap nodig. Er zijn verschillende ROM geheugens die momenteel nog relevant zijn :

**MROM**

Masked-programmed ROM is écht read-only. Krijgt zijn inhoud tijdens fabricage en kan daarna niet veranderd meer worden door de gebruiker.

**PROM**

Programmable ROM is ook read-only. De inhoud wordt vastgelegd met behulp van een speciaal programmeertoestel en dit kan nadien ook niet meer veranderen.

**EEPROM** (Electrically Erasable PROM).

Met dit type geheugen kan de inhoud gewist worden en vervolgens aangepast. Nadeel: Kostprijs.

**FLASH**

Net zoals EEPROM te wissen en te beschrijven, maar er is geen speciaal programmeertoestel nodig. Nadeel: kan minder dikwijls herschreven worden dan EEPROM. De oplossing hiervoor is een high-endurance flash, hiermee kan je de inhoud een paar miljoen keer herschrijven. Hierbij kunnen wel geen individuele cellen worden aangepast, maar steeds blokken.

8.2 Opstartroutine met BIOS

Bij het aanzetten vd spanning wordt er een reeks stappen doorlopen.

1/ Systeemstart

Interne voeding start, eens spanning stabiel is geeft de voeding power good-signaal. Zodra reset-signaal wegvalt zal computer beginnen opstarten. Processor begint op FFFF0h, waar de uitvoering van de BIOS opstartroutine begint.

2/ Power On Self Test

Opstartroutine BIOS voert de POST uit. Indien er fatale fouten ontdekt worden, krijg je een foutmelding. Een fout met de grafische kaart geef enkele BEEP-signalen en stopt het startproces.

3/ Zoeken naar BIOS-uitbreidingen

Systeem-BIOS niet enige BIOS. Grafische kaarten, harde schijven… hebben allemaal hun eigen BIOS. Het Systeem-BIOS gaat dus onmiddelijk op zoek naar de grafische kaart en laat het ingebouwde video-BIOS programma uitvoeren.

Daarna voert de computer enkele tests uit op het systeem (vervolg POST): het geheugen wordt onderzocht en de geheugenplaatsen worden geteld.

Vervolgens is er de inventarisatie waarbij de meest klassieke randapparatuur ( printer, seriële poorten, controllers) gedetecteerd zullen worden.

8.3 Plug and play configuratie

Uit de uitleg in de cursus kan ik niet uitmaken wat een plug and play configuratie is. Sorry.

8.4 Extensible Firmware Interface

1/ BIOS op pensioen

Het systeem van BIOS 1 vd laatste onderdelen waar nog niks substantieel aan veranderd is sinds de introductie van PC. Progammeren ervan gebeurt in machinecode, is vnl patchwork. Weinig mensen hebben de ervaring om dit te programmeren. Vb hiervan is de max grootte vd opstartpartitie: 2TB.

Intel heeft een poging gedaan om BIOS overbodig te maken door een ander systeem. Deze technologieën zijn verzameld in de EFI (Extensible Firmware Interface).

Deze technologie verschilt superhard vd oude BIOS:

- Gecodeerd in breed gekende taal (c++)

- Volledig modulair

- Compatibel met oude BIOS om overgang mogelijk te maken

- Drivers kunnen opgenomen worden in deze EFI

- De 2TB-grens is hier niet bestaande

Door deze vernieuwing zijn er een pak nieuwe mogelijkheden. EFI is best vergelijkbaar met een micro besturingssysteem. Enkele voorbeelden van nieuwe toepassingen:

- Omdat drivers kunnen geïntegreerd worden, kan het veel makkelijker worden om nieuw besturingssysteem te installeren.

- EFI kan makkelijk uitgebreid worden met kleine toepassingen. Denk bv aan een mediaspeler dat je kan gebruiken voor je laptop zelfs ‘echt’ opgestart is.

- De mogelijkheden zijn eindeloos, maar de eerste EFI’s gaan niet veel meer functionaliteit hebben dan de vroegere BIOS’en.

2/ (U)EFI verspreiding

Sinds 2002 wordt aan EFI gewerkt door enkele grote firma’s. Bijna elke grote hardware bouwer zit bij deze organisatie. Toch gaat het nog even duren voor we BIOS weg gaan kunnen denken.

Apple: Standaard EFI op alle recente computers ( met BIOS suppor)

Microsoft: Standaar vanaf vista x64 sp1

Linux: Reeds lange tijd standaard geïmplementeerd.

Ook moederborden bieden tegenwoordig deze ondersteuning, al moeten ze manueel ingesteld worden om UEFI te selecteren, BIOS blijft voorlopig nog even default.

8.5 Coreboot

2de alternatief voor BIOS: Coreboot. Verschil Coreboot en EFI: Coreboot probeert zich zo snel mogelijk overbodig te maken, dus zo snel mogelijk na het opstarten controle geven aan het besturingssysteem. Coreboot gaat gewoon de belangrijkste hardware initialiseren(bv RAM-geheugen). Het besturinssysteem krijgt dan de verdere controle.

Coreboot heeft ook geen GUI. De code hiervan is daardoor erg compact en snel.

Besturingssystemen die nog niet kunnen booten via coreboot, kunnen gebruik maken van alternatieve ‘payload’ die compatibiliteit met BIOS garandeert.

-> Nog niet breed ondersteund, maar wel bv al bij chromebooks in gebruik.

9) Internal I/O

-> Gaat over de datauitwisseling op moederbord en met systeemapparaten.

9.1 I/O transfers