Besturingssystemen 1

- 2008 samenvatting David Machiels
- 2010 herhalingsvragen boek Stallings
- 2010 juni examen
- 2010 samenvatting van Onbekend
- 2011 juni examen
- 2012 augustus examen
- 2012 juni examen
- 2013 augustus examen
- 2013 Juni examen
- 2014 juni examen
- 2015 Juni examen
- 2016 juni examen
- 2017 augustus examen
- 2019 FAT 16 oplossingen
- 2019 juni examen
- 2020 augustus examen
- 2020 handige scripts - ISW
- 2020 januari examen
- 2020 juni examen
- 2020 PE
- FAT 16 berekenen

2008 samenvatting David Machiels

Inleiding

Het besturingssysteem is de softwarelaag tussen de toepassingssoftware en de uiteindelijke uitvoering van de machine-instructies.

Situering

Voor de gebruiker van een computersysteem is de toepassingslaag het belangrijkste onderdeel. Het computersysteem wordt slechts gebruikt omdat het deze software kan uitvoeren. Nochtans is de toepassingssoftware slechts een klein onderdeel van het volledige computersysteem.

Hardware

De hardware zorgt voor de uitvoering van bevelen, die door software d.m.v. machine-instructies worden gegeven. Deze instructies worden meestal niet rechtstreeks uitgevoerd door fysische onderdelen maar via de microprogrammatie of microcode: een verzameling van kleine programma’s in machinetaal die door de processoren begrepen worden en toelaten dat relatief ingewikkelde operaties via één machine-instructie kunnen worden uitgevoerd. Microcode is een voorbeeld van wat men firmware noemt: in ROM opgeslagen software. Microcode heef niet één specifieke taak, maar dient om het schrijven van programma’s in machinetaal te vereenvoudigen door vaak voorkomende taken aan te bieden in één oproep.

Evolutie

Zonder besturingssysteem

Programma’s werden in machinetaal geschreven, en de computer kon maar 1 programma inladen en dan uitvoeren. Pas als een programma afgelopen was kon het volgende geladen worden. Na een tijd stelde men vast dat de geschreven programma’s veel gelijkaardige routines bevatten, bijvoorbeeld voor in- en uitvoer. Hierdoor begonnen fabrikanten van computersystemen bibliotheken met vaak gebruikte routines bij hun producten te leveren. Er moest een planning

worden opgesteld om te bepalen wie de computer mocht gebruiken. Als een programma vroeger dan voorzien klaar was, stond het systeem stil tot de programmeur die het volgende tijdsblok gereserveerd had aankwam. Dat zo'n duur tijdens dergelijke "idle time" niet gebruikt werd wilde men natuurlijk zo veel mogelijk vermijden. Wanneer een programma langer moet rekenen dan voorzien ontstaat er ook een probleem. Bovendien bestaat de kans dat een programma door een programmeerfout vastloopt. Het computersysteem wordt dan ook niet gebruikt terwijl de programmeur de fout opspoort.

Een mens als besturingssysteem?

Om de planningsproblemen te ondervangen werd na verloop van tijd een menselijke operator ingeschakeld. Programmeurs brengen de uit te voeren programma's naar de operator. Het laden en starten van de verschillende programma's is nu de taak van de operator. De operator bezorgt de programmeur het resultaat van zijn programma, of eventueel de inhoud van het geheugen op het moment dat er een fout optrad. Dit noemt men de zogenaamde core dump.

Er kunnen prioriteiten toegekend worden. De operator kan hiermee rekening houden bij het bepalen van de volgorde waarin de programma's gestart worden. De operator kan gegevens bijhouden, bijvoorbeeld van de gebruikte computertijd per programma. Deze kan eventueel gefactureerd worden aan de gebruikers.

Computersystemen worden steeds sneller, maar de menselijke operatoren niet. Hierdoor worden zij meer en meer de vertragende factor.

Bovendien kunnen de door de fabrikant geleverde bibliotheken alsmaar meer en complexere taken vervullen. Zo wordt de vertraging van de menselijke operator gedeeltelijk weggewerkt. Als de bibliotheken ook nog proberen te voorkomen dat er misbruik wordt gemaakt van de beschikbare hulpmiddelen kunnen we stellen dat ze evolueren naar de eerste besturingssystemen.

De eerste besturingssystemen

We kunnen eigenlijk pas van een besturingssysteem spreken vanaf het moment dat de zogenaamde residente monitors gebruikt worden, omdat zo'n residente monitor instaat voor de controle van het systeem, en constant in het geheugen gehouden wordt.

De residente monitor zorgt ervoor dat een uit te voeren programma in het geheugen wordt geladen en gestart. De operator moet wel nog steeds zorgen voor het aanbieden van de uit te voeren programma's, maar de rest van het beheer is overgenomen door de residente monitor.

Uit de codebibliotheken die bij de computersystemen geleverd worden evolueren ook specifieke besturingsroutines voor randapparaten, device drivers genaamd. Door deze drivers wordt het geven van opdrachten aan randapparaten eenvoudig. Een programma moet gewoon de juiste functie van de driver aanroepen. Samen met de opkomst van de hogere programmeertalen zorgden de device drivers ervoor dat het voor veel meer geïnteresseerden mogelijk werd om computers te gebruiken.

De eerste besturingssystemen groeien uit tot eenvoudige batchbesturingssystemen. De term batch wijst op de gebruikswijze van dit soort systemen: men biedt een stel programma’s als een globaal pakket aan het besturingssysteem aan.

Multiprogrammatie en Time Sharing

Batchbesturingssystemen kennen een steeds groter wordend efficiëntieprobleem. Terwijl het programma dat ze aan het uitvoeren zijn wacht op het voltooien van in- of uitvoer, wordt de processor van het computersysteem niet gebruikt.

Multiprogrammatie zorgt voor de oplossing. Terwijl een programma moet wachten op een externe gebeurtenis laat het besturingssysteem de processor verder werken aan een ander programma. Een reeks programma’s is samen in het geheugen aanwezig en worden om de beurt uitgevoerd tot ze moeten wachten.

Wanneer een multiprogrammatiesysteem wordt uitgebreid met een spoolingsysteem (SPOOL: Simultaneous Peripheral Operation OnLine) dat een aantal jobs kan klaarhouden op een snel extern geheugen en de simultane uitvoer van verschillende jobs op een ordelijke manier kan afhandelen komt men tot een zeer krachtig besturingssysteem.

Er is echter nog een groot gebrek aan interactiviteit. Als het resultaat niet is wat men verwachtte, moet men het programma wijzigen en opnieuw laten uitvoeren. De schijnbaar simultane uitvoer van programma’s bij multiprogrammatie laat ook toe om verschillende gebruikers tegelijk op interactieve wijze met een computer te laten werken. Elke gebruiker beschikt dan over een in- en uitvoerapparaat, verbonden met de centrale verwerkingseenheid. Omdat de processor afwisselend gedurende korte tijd aan elke gebruiker wordt toegewezen krijgt elke gebruiker de indruk over de volledige capaciteiten van het systeem te beschikken. Deze vorm van multiprogrammatie wordt timesharing genoemd. Het belangrijkste verschil is dat er bij timesharing niet gewacht wordt tot een programma moet wachten om de processor aan een ander programma toe te kennen. Door dat de gebruiker op een time sharing systeem rechtstreeks met de computer kan communiceren, krijgt hij bijna onmiddellijk feedback van kleinere opdrachten. Door dat de gebruikers typisch niet tegelijkertijd pieken veroorzaken in het processorgebruik lijkt het systeem voor ieder van hen performant te werken. Voor het accepteren van invoer via een toetsenbord moet de processor zeer weinig werk verrichten, dus terwijl de ene gebruiker een opdracht typt, is er processorkracht over voor het inlezen van de toetsenborden van andere gebruikers, of het werken aan eerder gegeven opdrachten.

Besturingssystemen voor minicomputers

In de jaren ’60 kwamen de minicomputers op de markt.

Bij mainframes was het de gewoonte dat een fabrikant een besturingssysteem ontwikkelde telkens hij een nieuw model op de markt bracht. Dit was voor de fabrikant een zware inspanning, maar voor de gebruiker betekende het ook telkens grote conversieproblemen als er naar nieuwere en krachtigere computers overgeschakeld werd.

In 1964 lanceerde IBM voor het eerst een familie van computersystemen. De System/360 reeks van mainframes omvatte modellen met verschillende verwerkingscapaciteiten, maar ze waren allemaal gebaseerd op dezelfde architectuur. Alle modellen gebruikten dan ook hetzelfde besturingssysteem: OS/360. Omdat OS/360 voor allerlei soorten toepassingen moest gebruikt worden, werd het een zeer ingewikkeld besturingssysteem. Dit noemt men multimode-systemen.

De PDP's (Programmed Data Processor) van Digital Equipment waren een zeer succesvolle reeks van minicomputers. In de wetenschappelijke wereld raakt intussen stilaan UNIX bekend, een nieuw besturingssysteem dat niet door een computerconstructeur werd ontwikkeld en waar geen commerciële bedoelingen aan vastzaten. De belangrijkste nieuwigheid is dat het, mits enkele kleine aanpassingen, op elke hardware bruikbaar kon worden gemaakt en daarbij naar de gebruiker toe steeds dezelfde interface aanbood. Het werd niet ontwikkeld in een assembleertaal maar in een daarvoor ontwikkelde hogere programmeertaal genaamd C.

De PC en het Internet

Aan het eind van de jaren 70 dook de microcomputer op. Toen IBM zijn microcomputer lanceerde en hiervoor een nieuw besturingssysteem genaamd MS-DOS ontwikkelde, bleek dit zo'n succes te zijn dat de meeste andere constructeurs deze architectuur overnamen. Men kon nu toepassingssoftware ontwikkelen die op zowat 90% van de zich massaal verspreidende microcomputers kon worden gebruikt.

Er moest dringend werk worden gemaakt van de vereenvoudiging van de gebruikersinterface. Vandaar de ontwikkeling van GUI (Guided User Interface) waarmee op een symbolische manier bevelen konden worden gegeven aan de computer.

- In 1984 kwam de Apple Macintosh SE op de markt, de eerste computer met een GUI.
- In 1985 was er de eerste versie van Windows, met een grafische laag bovenop DOS.

De stijgende performantie van de microcomputer was ideaal voor multitasking: time sharing voor één gebruiker. De gebruiker kan verschillende opdrachten tegelijk laten uitvoeren en deze laten samenwerken. De processortijd werd door het systeem verdeeld over al de opdrachten.

Men introduceerde ook netwerken. Eerst LAN, dan WAN en uiteindelijk het Internet.

Voorbeelden van besturingssystemen

Aangezien besturingssystemen sinds OS/360 niet meer specifiek voor één bepaalde machine ontworpen worden, zijn ze nu meestal verbonden met één van de drie besproken types: mainframes, minicomputers of microcomputers.

IBM is nog steeds één van de grote spelers op de mainframe-markt, en er worden nog steeds mainframes ontwikkeld. Nu worden de IBM mainframes zSeries genoemd. De besturingssystemen zijn in de loop der jaren voor deze ontwikkeld met klinkende namen als MVS, VM en SE.

Ook minicomputers zijn nog steeds te verkrijgen en in gebruik. Men spreekt nu ook soms van midrange systems. IBM noemt ze iSeries.

Voor microcomputers waren er in de jaren ’80 drie families.

- Op IBM-compatibele PC’s had je de keuze tussen MS-DOS van Microsoft en OS/2 van IBM.
- Voor Macintosh was er System 7.

OS/2 is van de markt verdwenen en MS-DOS is geëvolueerd naar de verschillende versies van Windows. Windows was eerst slechts een grafische toevoeging aan MS-DOS, maar de recentere versies zijn volledige besturingssystemen. Voor Macintosh heb je tegenwoordig MacOS nodig.

Er zijn ook besturingssystemen die niet aan één van de drie categorieën gebonden zijn.

- Allereerst is er UNIX dat men op de meest diverse hardwaren aantreft, zij het in allerlei varianten.
- Op mainframes vind je AIX (IBM) als gast onder VM.
- Op VAX minicomputers werd Ultrix gebruikt
- Terwijl voor microcomputers SCO UNIX of HPUX beschikbaar is. Voor PC’s is er sinds 1991 ook het steeds populairdere GNU/Linux.

Functies van een besturingssysteem

Twee grote taken: het gebruiksgemak maximaliseren en de efficiëntie van het systeem maximaliseren.

- Toegang tot apparaten eenvoudig maken
- Informatie bewaren en overzichtelijk weergeven
- Operaties en gegevens beveiligen

Wat betreft de hardware:

- De onderlinge communicatie tussen de onderdelen organiseren
- De goede werking van de onderdelen bewaken
- Hulpmiddelen rechtvaardig verdelen

Merk op dat het besturingssysteem ook een programma is dat door de processor uitgevoerd moet worden. We kunnen stellen dat de tijd dat het besturingssysteem de processor gebruikt verloren gaat voor de gebruiker. Efficiëntie is dus een belangrijke vereiste voor het besturingssysteem, om de hoeveelheid overhead te beperken.

Een derde taak is het streven naar hardwareonafhankelijkheid. Als het besturingssysteem een uniforme interface aanbiedt aan de bovenliggende toepassingssoftware en de gebruiker kan de hardware aangepast worden zonder overgangsproblemen. De eerste stap naar hardwareonafhankelijkheid was de introductie van device drivers.

Onderdelen van een besturingssysteem

Apparaatbeheer

De toegang tot de randapparatuur moet door het besturingssysteem geregeld worden, om conflicten te vermijden. Als 2 programma’s tegelijkertijd naar de printer sturen zou men vreemde resultaten krijgen.

Geheugenbeheer

Programma’s mogen niet in mekaars geheugenindex schrijven of lezen. Het beschikbare werkgeheugen is bovendien beperkt en het besturingssysteem moet ook streven naar een efficiënt gebruik ervan.

Procesbeheer

Het verdelen van de beschikbare processortijd over de verschillende processen die uitgevoerd moeten worden. Hier wordt gezorgd voor multiprogrammatie en multitasking.

Communicatie

De communicatie tussen processen op het computersysteem. Maar ook voor eventuele communicatie met andere computersystemen via een netwerk.

Gebruikersinterface

Om het gebruik van de computer te vereenvoudigen moet het besturingssysteem ook zorgen voor een adequate gebruikersinterface. Dit kan een tekstinterface of een grafische omgeving zijn.

Basismechanismen van een besturingssysteem

Kernel- en gebruikerstoestand

Programma’s die deel uitmaken van het besturingssysteem worden uitgevoerd in kerneltoestand. Wanneer de processor zich in kerneltoestand bevindt, beschikt hij over al zijn mogelijkheden. Wanneer de processor niet in kerneltoestand is, maar in gebruikerstoestand, zijn bepaalde instructies niet toegelaten. Tussen de machine-instructies zijn er namelijk een aantal die men geprivilegieerde instructies noemt. Deze laatste kunnen enkel worden gebruikt in kerneltoestand en zijn daarvoor voorbehouden aan het besturingssysteem. Wanneer een programma probeert om in gebruikerstoestand een geprivilegieerde instructie uit te voeren zal een fatale fout optreden. Op die manier kunnen allerlei delicate operaties op een gegarandeerd veilige manier worden uitgeoefend. Wanneer het besturingssysteem een gewoon programma laat uitvoeren zal het de processor eerst naar gebruikerstoestand brengen alvorens het programma te starten. In de toestandsbeschrijving van de processor is één bit voorzien om de toestand aan te geven waarin deze zich bevindt: user- of kernel mode.

Interrupts

Het besturingssysteem komt slechts in actie na een signaal vanuit de hardware of na een vraag vanuit de hoger gelegen software. Men zegt dan ook dat een besturingssysteem gebeurtenisgestuurd of event driven is.

Interrupts

- Wanneer vanuit de hardware behoefte is aan een tussenkomst in het besturingssysteem treedt een interrupt of onderbreking op.
- Als de software de hulp van het besturingssysteem nodig heeft wordt er een software interrupt aangeroepen of system call.

Een interrupt is een elektrisch signaal naar de processor, of een gebeurtenis binnenin de processor zelf. Als de harde schijf bijvoorbeeld een leesopdracht voltooid heeft, zal de schijfbesturingseenheid dit via een interruptsignaal naar de processor aan het besturingssysteem melden.

1. Schijfbesturingseenheid ontvangt een leesopdracht
2. Wanneer de leesopdracht voltooid is, stuurt de schijfbesturingseenheid een interrupt naar de interrupt controller
3. De interrupt controller geeft de interrupt door aan de processor. Wanneer er een interrupt met hogere prioriteit wordt afgehandeld is het mogelijk dat de schijfinterrupt hierop moet wachten.
4. Interrupt controller stuurt het volgnummer van de bron van de interrupt naar de processor, zodat die weet welk apparaat de interrupt veroorzaakte.
5. De processor zoekt het adres op van de juiste Interrupt Handler in de tabel met interrupt-vectoren.
6. De inhoud van bevelenteller, alle registers en de processorstatus worden op de stapel gezet.

7. Adres van Interrupt Handler wordt in de bevelenteller geplaatst en de processor wisselt naar kerneltoestand.
8. Sprong naar het adres van de Interrupt Handler en de handler wordt uitgevoerd. Deze code in de interrupt handler handelt het verzoek af. Deze handler is onderdeel van het besturingssysteem.
9. Als het werk erop zit herstelt de interrupt handler de bevelenteller, registers en processortoestand en schakelt de processor terug in gebruikerstoestand. Het programma dat in uitvoering was voor de interrupt wordt verder gezet.

Het gevolg van een interrupt is dus dat een stuk code uitgevoerd wordt, dat specifiek bestemd is voor de opgetreden interrupt: de Interrupt Handler.

Stappen 6 en 7 worden een context switch genoemd. Een context switch wijzigt de toestand van de processor zodat die een ander programma kan uitvoeren, en bovendien zo dat de toestand weer hersteld kan worden om verder te gaan met de uitvoering van het programma dat ervoor in uitvoering was.

Merk op dat dit door de processor zelf afgehandeld wordt, en niet door een stukje machinecode uit te voeren.

Externe interrupts komen van hardware buiten de processor:

- Klokinterrupts: De interne klok van de computer signaleert hiermee aan de processor dat een bepaalde tijdsperiode is verstreken
- I/O-interrupts: Wanneer een randapparaat een I/O-operatie is uitgevoerd meldt de besturingseenheid die het randapparaat bestuurt, op deze manier het einde van de I/O-operatie
- Keyboard interrupts: Telkens wanneer een toets op het klavier wordt ingedrukt heeft dit een interrupt tot gevolg, die tot doel heeft het bestemmende karakter te laten inlezen in een buffer zodat het programma waarvoor het bestemd is eraan kan.

Interne interrupts worden veroorzaakt door de processor en bestaan in twee groepen:

- Uitzonderingen of exceptions zijn het gevolg van een fout tijdens de uitvoering van een instructie zoals het delen van een getal door nul, het adresseren van een ongeldig adres of een poging om in gebruikerstoestand een geprivilegieerde opdracht uit te voeren.
- Traps of software interrupts zijn het gevolg van de uitvoering van speciaal daarvoor voorziene instructies in een programma. Als een programma bijvoorbeeld gegevens wil inlezen van op de harde schijf zal het dit via een software interrupt aan het besturingssysteem vragen.

Interrupts identificeren

Hoe wordt nu bepaald welke interrupt handler dient te worden uitgevoerd wanneer een interrupt optreedt? Bij exceptions wordt de interrupt door de processor zelf veroorzaakt, en kan de processor de inhoud van de conditiecode in zijn toestandsbeschrijving nagaan.

Wanneer een programma in uitvoering een trapinstructie doet wordt de nodige informatie meegegeven om de juiste interrupt handler te kiezen.

In 8086 assembler programma's voor MS-DOS wordt vaak "int 21h" opgeroepen. Deze software interrupt zorgt ervoor dat MS-DOS een interrupt handler uitvoert.

Bij hardware interrupts ligt de oorzaak buiten de processor, en moeten we op zoek naar de bron van de onderbreking. Dat kan op 3 algemene manieren:

- Als iedere controller een eigen interruptlijn heeft, bepaalt de lijn waarop het interruptsignaal verschijnt meteen welke handler er moet worden uitgevoerd.
- Als de controllers via één lijn met de processor verbonden zijn kan de controller d.m.v. een bit in één van zijn registers aangeven of hij een interrupt heeft verstuurd. Na de context switch wordt een routine uitgevoerd die aan software polling doet: achtereenvolgens elke controller controleren tot de afzender van de interrupt wordt gevonden.
- Bij de derde manier zijn alle controllers via één interruptlijn en één zogenaamde acknowledge-lijn met de processor verbonden. Zodra een hardware-interrupt optreedt zendt de processor via deze acknowledge-lijn een signaal naar de eerste controller. Indien deze de interrupt veroorzaakte stuurt hij een identificatiecode naar de processor; in het andere geval geeft hij het signaal door naar de volgende controller, die hierop op dezelfde manier reageert. Deze werkwijze, daisy chaining genoemd, is erg snel omdat ze volledig via de hardware verloopt.

Prioriteiten voor interrupts

Wanneer twee of meer interrupts tegelijkertijd optreden, moet er gekozen worden welke interrupt eerst afgehandeld wordt. Vaak wordt gekeken naar de vereiste interrupt latency. Dit is de maximale tijd waarbinnen de interruptafhandeling begonnen moet zijn.

Hoe we de prioriteit van interrupts kunnen bepalen hangt af van de gebruikte identificatiemethode.

- Bij gebruik van meerdere interruptlijnen zal de processor zelf moeten beslissen aan welk signaal hij eerst aandacht zal besteden.
- Bij software polling zal de volgorde waarin de controllers worden nagekeken bepalend zijn voor de prioriteit.
- Bij daisy chaining is het de volgorde waarin de controllers zijn aangesloten op de acknowledgelijn die dit bepaald. Een duidelijk nadeel is dat een verandering in prioriteit een hardwarematige ingreep vereist.

Een tweede vraag die we ons moeten stellen is of we nieuwe interrupts willen toelaten tijdens het afhandelen van een interrupt. Vaak wordt een vlag voorzien waarmee alle interrupts tijdelijk geblokkeerd worden. Andere systemen accepteren alleen interrupts met een hogere prioriteit.

System Calls

System calls of systemaanroepen leggen een link tussen de overige software en het besturingssysteem. Bij alle activiteiten uitgevoerd in kerneltoestand is immers zeer frequent de tussenkomst nodig van het besturingssysteem om operaties te kunnen uitvoeren die alleen in kernel-toestand mogelijk zijn.

Het uitvoeren van een system call

Essentieel bij een system call is het overschakelen van de processor naar kerneltoestand. Dit gebeurt altijd via een trapinstructie, waardoor wordt vermeden dat na het instellen van de kerneltoestand het gebruikersprogramma de controle zou kunnen behouden. De trapinstructie zorgt er voordat de gepaste systeemroutine wordt gestart. Daartoe wordt als operand voor de trapinstructie een code meegegeven, die in een register wordt geplaatst.

Omdat de trap functioneert als een interrupt zal een interrupt handler worden gestart in kernel mode, die de meegegeven code zal vertalen naar het beginadres van een specifieke systeemroutine. Aan de hand van de code wordt tevens bepaald hoeveel bytes aan informatie als parameters aan de systeemroutine moeten worden doorgegeven. Deze bytes worden dan klaargezet, waarna de systeemroutine kan worden opgeroepen. Zodra de opdracht voltooid is wordt een eventueel resultaat op een afgesproken plaats gedeponeerd, waarna de processor wordt teruggeschakeld naar user mode en tenslotte terugkeert naar het opgeroepen programma.

Beschikbare system calls

- Opstarten en beëindigen van programma’s
- Laten wachten van programma’s op een gebeurtenis
- Toekennen en vrijgeven van geheugen
- Bestandsbeheer
- Besturing en beheer van apparaten
- Uitwisselen van informatie
- Communicatie

Voorbeeld

In Linux wordt de read-system call genaamd read (file.desc, buffer, nbytes) opgeroepen.

1. Oproepend programma zet de waarden van de parameters op de stapel.
2. Read functie wordt aangeroepen
3. Functie zet system call nummer in het juiste register en voert een trapinstructie uit
4. Besturingssysteem zoekt juiste routine op voor de system call
5. Besturingssysteem start systeemroutine
6. Systeemroutine zet resultaat op voorziene plaats, verlaat kernel mode en geeft controle terug aan read-functie
7. Read-functie ontvangt de gegevens, zet ze op hun plaats en geeft controle terug aan oproepend programma
8. Uitvoer programma gaat verder

Stappen 3 tem 6 worden uitgevoerd in kernel mode.

Bootproces

Basic Input/Output System (BIOS)

Het eerste programma dat de processor uitvoert is het Basic Input/Output System, vaak afgekort tot BIOS. Het zorgt dat er communicatie mogelijk is tussen enkele onderdelen van het computersysteem, in de eerste plaats tussen secundaire opslagapparaten en het werkgeheugen. Het Basic Input/Output System dankt zijn naam aan het feit dat het naast deze communicatie ook zorgt voor uitvoer op het scherm, en de mogelijkheid tot invoer via het toetsenbord.

Aangezien net de BIOS ervoor moet zorgen dat de opslagapparaten bruikbaar worden kan de BIOS-code niet op een dergelijk apparaat worden opgeslagen. Daarom wordt deze in een ROM-chip op het moederbord bewaard. Tegenwoordig wordt hiervoor EEPROM gebruikt. BIOS is een voorbeeld van firmware.

Omdat een processor altijd de opdracht uit het werkgeheugen inleest waarvan het adres in de bevelenteller staat, wordt de inhoud van de BIOS-chip beschikbaar gesteld via een deel van het adresbereik van het werkgeheugen. Hiervoor is een deel van het Upper Memory gereserveerd: van F0000h tot FFFFFh. In sommige systemen blijft dit deel van het werkgeheugen leeg en wordt van de ROM-chip gelezen. Andere systemen kopiëren de inhoud van de chip naar dit gedeelte van het werkgeheugen omdat RAM sneller is dan ROM. Dit noemt men ROM shadowing.

Wanneer een processor geactiveerd wordt zal hij altijd beginnen met de instructie op adres FFFF0h. Hier staat dan de jump instructie naar de eigenlijke opstartcode.

Power-On Self Test

De BIOS-opstartcode begint met het testen van de vitale systeemonderdelen. Het geheel van deze test wordt Power-On-Self-Test of POST genoemd. De aanwezigheid of correcte werking van bijvoorbeeld processor, werkgeheugen en toetsenbord worden getest. De feedback gebeurt d.m.v. geluidsignalen die we beeps noemen. Bij problemen kan men aan het aantal beeps en hun duur horen wat de oorzaak is.

Opstartvolgorde

Als de POST succesvol voltooid is zal de BIOS-code beginnen met het laden van het besturingssysteem. Wanneer een computersysteem meerdere opslagapparaten bevat moet in de BIOS aangegeven zijn vanop welk apparaat een besturingssysteem geladen moet worden. Het besturingssysteem kan op een harde schijf staan, maar ook op een diskette, CD-ROM of USB-opslagapparaat. In welke volgorde op alle aanwezige apparaten gezocht moet worden staat in de

zogenaamde opstartvolgorde of boot sequence. Omdat deze instellingen gewijzigd moeten kunnen worden kunnen ze niet samen met de code van de BIOS bewaard worden op de ROM-chip. Voor de veranderlijke instellingen is er een afzonderlijke CMOS-chip voorzien, die door een batterij onder spanning gehouden wordt wanneer het systeem uitgeschakeld is. Zo gaan de instellingen niet verloren.

Partities

Als in de opstartvolgorde wordt verwezen naar de harde schijf is de vraag weer hoe we het besturingssysteem terugvinden op zo'n schijf. De code in de BIOS zal een sprong uitvoeren naar de opstartcode opgeslagen in de eerste sector van de schijf, de Master Boot Record. Deze code wordt vaak Initial Program Loader (IPL) genoemd.

Een harde schijf kan logisch ingedeeld worden in partities. Hierdoor kunnen we meerdere besturingssystemen op één schijf bewaren, of kunnen we op één schijf verschillende bestandssystemen gebruiken.

In de partitietabel wordt het begin- en eindpunt en het type van iedere partitie opgeslagen. Op het PC-platform vind je de partitietabel in de MBR, achter de opstartcode. Er is dan nog plaats voor 4 maal een partitiebeschrijving en een magic number van 2 bytes. Dit magic number is een controlegetal en moet altijd als waarde 0xAA55 hebben, anders wordt de MBR als ongeldig beschouwd.

Elk van de 4 elementen van de partitietabel beschrijft een partitie, en bevat de volgende velden:

- Active geeft aan of deze partitie moet gebruikt worden om van te booten of niet (Windows)
- Start cylinder / kop / sector adres van de eerste sector van de partitie
- Type indicatie van het gebruikte bestandssysteem
- End cylinder / kop / sector adres van de laatste sector van de paritie
- Lengte lengte van de partitie

De partities die in de partitietabel gedefinieerd worden noemen we primitieve partities. Om meer partities toe te laten introduceerde men logische partities. Er is geen plaats om de partitietabel uit te breiden, dus wordt een primaire partitie onderverdeeld in logische partities. Een primaire partitie die verder onderverdeeld wordt noemen we extended partitie.

In de partitietabel geven de types 0x05 of 0x0f een extended partitie aan. Het start-veld bevat dan het adres van de eerste sector van de eerste logische partitie. Iedere logische partitie verwijst naar de volgende logische partitie. Als limiet voor het aantal logische partities wordt vaak 24 opgegeven. De gelinkte lijst kan langer zijn maar aangezien DOS letters vanaf C toekent kan de lijst niet langer dan 24 partities lang zijn.

Een logische partitie bevat een Extended MBR met een eigen partitietabel. Zo'n EMBR bevat een partitietabel die de structuur van de tabel in de MBR volgt, maar slechts 2 partities beschrijft: één logische en een nieuwe extended partitie. Deze extended partitie bevat ook weer zo'n tabel die

één logische en één extended partitie beschrijft, enz. De taak van de opstartcode in de MBR is om in de partitietabel de actieve partitie op te sporen. De eerste sector van deze partitie heeft Partition Boot Sector en bevat ook weer een stuk opstartcode. De code in de MBR zal de code in de PBS inladen en dan een sprong naar deze code uitvoeren. Het is deze code die uiteindelijk het besturingssysteem op de partitie zal inladen en starten.

Bestandssystemen

Computersystemen hebben er naast gegevensverwerking een taak bij gekregen: gegevensopslag. Het onderdeel van het besturingssysteem dat instaat voor permanente opslag is het bestandssysteem. De eisen die men hieraan stelt zijn:

- Persistente gegevensopslag
- Eenvoudige organisatie en logische ordening van gegevens
- Een zo snel mogelijke gegevenstoegang
- Zo goed mogelijk gebruik maken van de beschikbare capaciteit
- Het beveiligen van de gegevens (geen toegang tot gegevens v/e andere gebruiker)

Fysische en logische gegevensordening

Fysische ordening: Sectoren

Eén sector is de kleinst adresseerbare eenheid voor gegevensopslag op een harde schijf. Hoe de gegevens over de sectoren van de harde schijf verdeeld zijn noemen we de fysische ordening van gegevens.

Logische ordening: Bestanden

Een gebruiker wil een logische gegevensordening, die er voor zorgt dat gegevens gemakkelijk terug te vinden zijn. Een bestand is een persistent opgeslagen logische verzameling gerelateerde gegevens.

Het bestandssysteem vertaalt het logische beeld van de gebruiker en de bijbehorende operaties in de fysische schijftoegangen. Zo wordt een bestand altijd voorgesteld als een aaneengesloten reeks gegevens, zelfs al is dit in de fysische ordening niet zo is. Het opzetten van de nodige gegevensstructuren voor de fysische ordening van een bestandssysteem op een partitie noemt men formatteren.

Bestanden

Een bestand is een verzameling gerelateerde gegevens, die door een gebruiker of applicatie op een bepaalde manier geïnterpreteerd worden. Het bestandssysteem verzamelt alle gegevens over een bestand in de bestandsbeschrijving of file descriptor.

- De naam
- De adressen van de gebruikte sectoren

De lengte van de naam kan beperkt zijn door de beschikbare ruimte in de file descriptor. Welke karakters toegelaten zijn hangt af van de gebruikte codering (ASCII of Unicode) en het vermijden van tekens die een speciale betekenis hebben in het systeem. Ook de hoofdlettergevoeligheid kan van belang zijn. Sommige systemen leggen bovendien een vaste structuur op de naam, bijvoorbeeld de “8+3”-structuur van MS-DOS: elke bestandsnaam kan 8 tekens lang zijn en een extensie van 3 tekens hebben.

Naast de bestandsnaam worden in de file descriptor allerlei andere bestandskenmerken bijgehouden.

- De grootte van het bestand
- De tijdstippen waarop het bestand is aangemaakt of gewijzigd
- De eigenaar van het bestand
- Op welke manier toegang tot het bestand wordt geregeld

Om de plaats te kunnen terugvinden waar de gegevens op het extern geheugenmedium worden bijgehouden moet er ook een plaatsaanduiding bijgehouden worden. Het kan ook interessant zijn om te weten of het bestand in gebruik is, en of er een kopie bestaat.

Directories

Om een overzicht mogelijk te maken in de massale hoeveelheden informatie is het noodzakelijk dat bij elkaar horende bestanden gegroepeerd worden in directories. Directories zijn een toevoeging aan het logische beeld dat de gebruiker van de opgeslagen gegevens krijgt.

Er is ook een fysische gegevensstructuur op de schijf, die de bestandsbeschrijvingen van de bestanden bevat. Structuur van het bestandssysteem

Lineaire directory

De meest eenvoudige organisatievorm. De bestandsbeschrijvingen van alle in het bestandssysteem aanwezige bestanden worden in één enkele directory opgenomen. Nadelen:

- Indien het bestandssysteem een groot aantal bestanden bevat zullen de basisbewerkingen veel tijd vragen.
- Een dergelijke lange lijst van bestanden is uiterst onoverzichtelijk.
- Aan elk bestand wordt een unieke naam gegeven.

Voordeel:

- Bij schijfgeheugens kan de directory op de meest gunstige plaats van de schijf worden gezet, zodat snelle toegang kan worden verkregen.

Lineaire directories zijn slechts bruikbaar in kleine bestandssystemen in omgevingen met één gebruiker.

Hiërarchisch bestandssysteem

Hierbij wordt de volledige verzameling bestanden opgesplitst in een aantal kleine groepen, die elk hun eigen directory hebben. Een directory bevat dus bestanden of andere directories. De directories waarnaar een directory verwijst noemen we subdirectories.

Het is niet meer nodig dat elk bestand een unieke naam heeft. Bestanden in verschillende directories kunnen dezelfde naam hebben. Nadelen:

- Men moet aangeven waar het bestand zich bevindt in de hiërarchische structuur.
- Sommige bewerken worden complexer. Het opzoeken van een bestand waarvan alleen de naam bekend is vereist nu het recursief doorlopen van de volledige directorystructuur.
- Ook het schrappen stelt een probleem: wat gebeurt er met de bestanden en subdirectories waarnaar vanuit deze directory wordt verwezen? Indien deze mee moeten worden vernietigd moet dit eveneens op een recursieve manier gebeuren.

Pad: de opsomming van alle directories die moeten doorlopen worden vanaf de hoofddirectory om de directory waarin de file wordt beschreven te bereiken. Working directory: de directory waarin de gebruiker zich bevindt Relatief pad: Een pad dat vertrek van de huidige werkdirectory Absoluut pad: Een pad dat vertrekt vanuit de wortel van de boomstructuur

Boomstructuur

De meest eenvoudige hiërarchische structuur. Naar ieder bestand in het systeem leidt er juist één uniek pad. We gebruiken de term boomstructuur omdat deze structuur zich vertakt als een boom.

Algemene structuur

Wanneer we toch meer dan één pad naar een bestand toelaten krijgen we een algemene structuur. Dit betekent dat een ebstand een element van twee verschillende directories kan zijn, of dat een directory een subdirectory is van twee of meer directories.

De algemene structuur laat toe dat er lussen gecreëerd worden in het bestandssysteem. Iedere recursieve operatie op het bestandssysteem, zoals het zoeken naar een bestand, zal oneindig doorgaan in zo’n lus als het recursief algoritme niet detecteert dat het een bepaalde directory al heeft behandeld.

Een bijkomend probleem betreft het achterblijven van onbereikbare bestanden in het systeem na het verwijderen van de directory. Een directory moet steeds bereikbaar zijn vanuit de wortel. Als mogelijke oplossing voor dit probleem zouden we kunnen beslissen om altijd recursief alle subdirectories te verwijderen, zelfs al lijken ze nog bereikbaar. Maar dan stuiten we op een tweede probleem. Als we alle onderliggende directories recursief verwijderen, zullen we uiteindelijk een directory verwijderen die hoger in de hiërarchische structuur ligt dan de directory die we oorspronkelijk wilden verwijderen. Omwille van deze problemen worden dergelijke lussen in het bestandssysteem zelden toegelaten. Men krijgt dan een acyclische structuur.

Acyclische structuur

De acyclische structuur is nog steeds algemener dan de boomstructuur, maar er mogen geen lussen voorkoen. Lussen vermijden kan op twee manieren. We kunnen bij iedere operatie die een lus zou kunnen veroorzaken een controle uitvoeren. De operatie wordt enkel toegelaten als er geen lus ontstaat. Operaties in het bestandssysteem worden dan wel complexer en dus trager.

Een eenvoudigere oplossing is het verbieden van meervoudige verwijzingen naar directories. Wanneer enkel bestanden vanuit twee directories kunnen benaderd worden is het onmogelijk een lus te creëren.

Aaneengesloten bestanden

Vanaf een bepaald adres worden alle bytes van de file achtereenvolgens op de schijf gezet, in een reeks op elkaar volgende sectoren. We noemen dit een aaneengesloten bestand. Er ontstaat een belangrijk probleem: Er is voldoende grote vrije ruimte nodig om het bestand een plaats te kunnen geven. Naarmate het gebruik van de schijf voortduurt zal er door het verwijderen van bestanden een steeds groter aantal afzonderlijke vrije ruimten ontstaan. De omvang van de grootste vrije ruimte zal daarbij ook steeds kleiner worden. Men zegt dan dat er fragmentatie ontstaat.

Om een geschikte locatie te kiezen moeten we in de eerste plaats de grootte van het bestand kennen. Aangezien een bestand kan groeien naarmate de tijd vordert, is de grootte niet gekend op het moment dat het bestand aangemaakt wordt. De enige oplossing is om op de een of andere manier een schatting te maken van de uiteindelijke grootte.

Er zijn 3 algoritmes om een element van de tabel te kiezen wanneer een nieuw bestand bewaard moet worden:

- Het First Fit algoritme kiest de eerste vrije ruimte die voldoende groot is voor het bestand.
- Het Best Fit algoritme kiest de kleinste vrije ruimte die groot genoeg is voor het bestand.
- Het Worst Fit algoritme kiest de grootste beschikbare vrije ruimte.

Het belangrijkste voordeel van het werken met aaneengesloten bestanden is ongetwijfeld de hoge snelheid waarmee deze bestanden kunnen worden verwerkt. Door het gebruik van op elkaar volgende sectoren zal een bestand zich meestal binnen één cilinder van de schijf bevinden en moet de arm van de schijf zich niet bewegen.

Een ander voordeel is de mogelijkheid om niet alleen sequentiële, maar ook directe toegang tot een bepaald deel van het bestand te krijgen. Als we het startadres s van het bestand en de blokgrootte b kennen dan kan op eenvoudige wijze het adres a worden berekend van het blok dat de byte op positie B binnen het bestand bevat: A = s + B/b Ernstige nadelen:

- Een eerste moeilijkheid is de verplichting een schatting te geven van de grootte van elk nieuw bestand. Regelmatig zal bij het schrijven naar een ebstand blijken dat de voorziene ruimte niet volstaat.
- Dan kan het programma afgebroken worden om met een betere schatting te worden hernomen of kan het programma onderbroken worden om het bestaande deel van het bestand naar een grotere vrije ruimte te kopiëren.
- Fragmentatie zorgt ervoor dat het bijna onmogelijk is de capaciteit van de schijf volledig te gebruiken. In dat geval zal reorganisatie van de gegevens op de schijf noodzakelijk zijn. Dit defragmenteren kan eventueel ook zonder tweede harde schijf, maar is een tijdrovend proces.

Niet-aaneengesloten bestanden

De oplossing voor dit probleem kan gevonden worden in het verdelen van het bestand in kleinere stukken. Het gevolg is natuurlijk dat delen van eenzelfde bestand over de schijf zullen worden verspreid. De lees/schrijfkop moet zich vaak verplaatsen, wat erg tijdrovend is.

Blokken

Het bestandssysteem groepeert een aantal sectoren in een blok, ook soms cluster genoemd.

- Een verdeling in grotere eenheden zorgt ervoor dat het bestand in minder delen gesplitst wordt. Hierdoor wordt het bestand minder sterk verspreid dus moet de leeskop minder verplaatsingen maken.
- Bovendien zal de adresinformatie die we nodig hebben om de verschillende delen van het bestand terug te vinden, kleiner zijn omdat het aantal delen kleiner is.

Kleinere eenheden hebben dan weer als voordeel dat het plaatsverlies aan interne fragmentatie kleiner zal zijn. Bij het verdelen van een bestand over een aantal blokken zal het laatste blok zelden volledig gevuld zijn. Het laatste stuk van dit laatste blok is onbruikbaar om andere gegevens op te slaan, en gaat dus verloren.

Wanneer we bestanden opsplitsen moet het bestandssysteem natuurlijk ook bijhouden waarde verschillende delen van het bestand opgeslagen zijn.

Bestanden als gelinkte lijsten

De file descriptor bevat het adres van het eerste blok van het bestand, en elk blok bevat een verwijzing of pointer naar het volgende blok. Het laatste blok van het bestand bevat een speciale pointer die aangeeft dat de lijst stopt, soms nul-pointer genoemd (vaak 999). De vrije blokken

worden ook d.m.v. een gelinkte lijst bijgehouden. Ieder vrij blok bevat het adres van het volgende vrije blok. Voordeel:

- Adresinformatie neemt weinig plaats in beslag.

Nadeel:

- De vaststelling dat directe toegang hier niet mogelijk is. De enige manier om bij een bepaald deel van een bestand te komen is immers de ketting van verwijzingen te volgen tot bij de gezochte plaats. De toegangssnelheid is erg laag.
- Het verwijderen van een bestand verloopt erg traag omdat de ketting van alle blokken moet doorlopen worden om de nul-pointer te vervangen door een verwijzing naar het eerste blok van de lijst met vrije blokken.

Door in de bestandsbeschrijving ook een pointer naar het laatste blok bij te houden kunnen we het verwijderen van een bestand versnellen.

Wanneer er iets misloopt met een verwijzing gaat een deel van het bestand verloren, of kan het bestand onterecht verwijzen naar een deel van een ander bestand. Om dergelijke problemen te vermijden wordt soms gewerkt met een dubbel gelinkte lijst, waarin ieder blok ook verwijst naar het vorige blok in de lijst. Een foutieve verwijzing kan hersteld worden door de lijst van achter naar voor te doorlopen.

Bestanden met indexblokken

Dit probleem kan worden opgelost door de adressen van de verschillende onderdelen van een bestand te verzamelen in een speciaal daarvoor voorbehouden blok op de schijf: een indexblok. In de file descriptor staat dan de plaatsaanduiding van dit indexblok.

Iedere bestandsbeschrijving in de directory bevat het adres van een indexblok.

Elk van deze blokken bevat de adressen van de blokken met de eigenlijke gegevens van het bestand. Ook de vrije blokken worden bijgehouden in een indexblok. Indien één blok niet volstaat kan men een gelinkte lijst van indexblokken creëren of kan men een hiërarchische index-structuur opbouwen. Hierbij verwijst het eerste indexblok niet naar delen van het bestand, maar naar indexblokken die de adressen van de blokken met gegevens bevatten. Voordeel:

- De toegangssnelheid. Wanneer een bestand geopend wordt kan het indexblok volledig in het geheugen geladen worden. Zelfs bij seriële toegang is er sprake van een snelheidswinst omdat er niet moet gewacht worden tot het bestandssysteem het adres van het volgende blok uit het vorige blok gelezen heeft.

Nadeel:

- De indexblokken nemen schijfruimte in. Dit komt omdat er per bestand één blok extra nodig is: het indexblok. Dit plaatsverlies is vooral merkbaar wanneer er veel kleine bestanden zijn.

Daarom wordt in de bestandsbeschrijving vaak plaats voorzien voor een klein aantal adressen van blokken van het bestand. Als de hoeveelheid benodigde blokken onder dit aantal blijft, moet er geen indexblok gebruikt worden.

De toegangssnelheid kan verder geoptimaliseerd worden door extents te gebruiken. Een extent is een verzameling opeenvolgende blokken op de schijf. Hierdoor zullen grotere delen van een file op een zelfde plaats van de schijf terecht komen, zodat het lezen of schrijven minder onderbroken hoeft te worden voor een verplaatsing van de leeskop. Het adres van het eerste blok en de lengte van de extent zijn voldoende.

Bestanden beschrijven in een file map

We beschrijven de inhoud van de schijf in een aparte tabel. Deze tabel, de file map, wordt voor ieder blok van de schijf aangegeven of het vrij is, en indien niet tot welk bestand het behoort.

De eenvoudigste versie van een file map is een bitvector. Met ieder blok komt een bit overeen: 1 betekent dat het blok in gebruik is, 0 dat het blok vrij is. Vergeleken met indexblokken zorgt zo'n bitvector van vrije blokken voor minder plaatsverlies. De vaste lengte van de bitvector (1 bit blok) is wel nadeling als er weinig vrije blokken zijn. Dan ziet de bitvector er zo uit: 111...111101111..111.

Met elk blok komt een element van de tabel overeen. De elementen, behorend bij de blokken die samen een bestand vormen, bevatten pointers zodat ze een gelinkte lijst vormen. Het laatste element in de lijst bevat een code om het einde van het bestand aan te geven. De plaatsaanduiding in de file descriptor bestaat uit de index van het element dat overeenstemt met het eerste blok in de file.

Voordeel:

- De mogelijkheid om de volledige beschrijving van een schijf in het centrale geheugen te brengen. We werken eigenlijk weer met een gelinkte lijst, maar nu is de volledige lijst van adressen al beschikbaar in het werkgeheugen.

Men kan er ook op een eenvoudige wijze voor zorgen dat een bestand zo goed mogelijk gegroepeerd blijft: de combinatie van de beschrijving met de adresgegevens van het bestand in één tabel laat toe bijkomende ruimte voor het bestand te zoeken.

Directories (2)

Een directory is een verzameling bestandsbeschrijvingen. Er zijn verschillende fysische structuren mogelijk om deze bestandsbeschrijvingen op een schijf op te slaan: een gewone tabel, een gesorteerde tabel, een hashtabel of een gelinkte lijst.

Tabel

Wanneer we deze bestandsbeschrijvingen in een tabel bijhouden, zal vooral het opzoeken van een bestandsbeschrijving vrij veel tijd vragen. We moeten de tabel doorlopen tot we het gezochte element vinden. Het toevoegen van een file descriptor kan zeer snel gebeuren, aangezien die gewoon achteraan kan bijgeplaatst worden.

Gesorteerde tabel

Om het opzoeken in een tabel te versnellen kan de tabel gesorteerd worden. Het is dan mogelijk om het binair zoeken toe te passen: bekijk het middelste element, en dan weet je of het gezochte element ervoor of erachter ligt. Zo kan je onmiddellijk de helft van de tabel negeren, en op dezelfde manier verder zoeken in de overgebleven helft. Doordat we de tabel gesorteerd moeten houden zal het toevoegen van de file descriptors natuurlijk trager verlopen. Dit zoeken kan ook binair, maar dan moeten alle achterliggende elementen nog opgeschoven worden om plaats te maken voor de nieuwe bestandsbeschrijving. Bij het verwijderen moeten we de lege plaats opvullen door alle volgende elementen een plaats naar voor te schuiven.

Hashtabel

De positie in de tabel wordt berekend met behulp van de hashfunctie. In dit geval zal de gekozen hashfunctie de bestandsnaam transformeren naar een geheel getal. De bestandsbeschrijving zal dan op deze positie in de hashtabel bewaard worden. Wanneer we een bestandsbeschrijving zoeken, bereken we de hashwaarde van de bestandsnaam, en we kennen onmiddellijk de positie waarop de gezochte file descriptor staat. Een mogelijk probleem zijn collisions of botsingen. Wanneer de hashfunctie voor twee verschillende bestandsnamen dezelfde positie teruggeeft.

Hoe groter de hashtabel, hoe meer posities en hoe meer mogelijke uitkomsten van de hashfunctie, en hoe minder kans op een botsing. Het nadeel is natuurlijk dat er meer plaats verloren gaat voor de grotere tabel. In gunstige omstandigheden levert de hashfunctie dus onmiddellijk de plaats van een op te zoeken file descriptor. Ook het toevoegen of verwijderen van file descriptors kan zeer snel gebeuren.

Gelinkte lijst

In de tabel verwijst ieder element naar de positie van het volgende. We moeten dan alleen de startpositie kennen. Bijvoorbeeld: index 1 is de startpositie voor de directory, index 2 is de startpositie voor de vrije plaatsen in de tabel. Een verwijzing naar 0 geeft het einde van de lijst aan.

Wat betreft het opzoeken van een element is deze structuur te vergelijken met de klassieke tabel. Het moet sequentieel gebeuren.

Bij het toevoegen of verwijderen kunnen we gewoon de verwijzingen aanpassen, en alle elementen kunnen op hun plaats blijven staan. Het geringe plaatsverlies door de verwijzing naar de volgende bestandsbeschrijving levert dus snelheidswinst bij deze operaties.

Voorbeeld 1: FAT

Het FAT-bestandsysteem is het oorspronkelijke bestandssysteem van het MS-DOS besturingssysteem.

File Allocation Table

FAT staat voor File Allocation Table. Deze FAT is een file map met daarin de nodige informative over ieder blok van de schijf. Een FAT-partitie bevat een boot sector, 2 kopieën van de FAT, en een datagebied dat de directories en bestanden bevat.

Wanneer een blok onderdeel van een bestand is bevat het overeenkomstige FAT-element het adres van het volgende blok van het bestand. Het laatste blok van een bestand wordt aangeduid met de End Of File-code OxFFFF, en een vrij blok met 0x0000.

Het getal in de naam van deze systemen duidt aan hoeveel bits er gebruikt worden voor ieder element in de FAT. In een FAT12-systeem bevat ieder element van de FAT dus 12 bits.

Het aantal bits dat een FAT-element bevat bepaalt welke adressen er gebruikt worden voor de blokken. In een FAT-12 tabel zijn er voor ieder element 212 of 4096 mogelijkheden. We stellen dus vast dat de maximale adresseerbare schijfcapaciteit bepaald wordt door het beschikbaar aantal bits voor het adres, maar dat ook de blokgrootte een belangrijke rol speelt.

De totale grootte van de FAT-tabel speelt ook een belangrijke rol, aangezien we omwille van de performantie de volledige tabel in het werkgeheugen willen houden.

FAT12:

- De grootte van de tabel is dan 212 x 12 bits, dus 49152 bits of 6 KB.

Voor FAT32 wordt dit 232 x 32 bits, wat overeenkomt met 16GB. Door de blokgrootte te vergroten kunnen we dus oftewel zorgen voor een grotere adresseerbare schijfcapaciteit, of voor een kleinere FAT-tabel.

Partitiegrootte Blokgrootte 33 tot 64MB 512B 65 tot 128 MB 1KB 129 tot 256MB 2KB 257MB tot 8GB 4KB 8GB tot 16 GB 8KB 16GB tot 32GB 16KB Meer dan 33GB Niet ondersteund.

Voor meer dan 32GB is NTFS vereist.

Directories

De logische structuur van het FAT-besturingssysteem is een boomstructuur. Fysisch wordt een directory voorgesteld als een gewone tabel. Deze tabel wordt op dezelfde manier opgeslagen op de schijf als een gewoon bestand: blokken waarop de directory-tabel staan worden via een gelinkte lijst in de FAT bijgehouden.

In FAT12 en FAT16 staat de rootdirectory onmiddellijk na de 2 kopieën van de FAT. In FAT32 is dit niet meer verplicht. In de directory-tabel neemt iedere bestandsbeschrijving 32 bytes in.

- Naam
- Extensie
- Attributen
- Tijdstip creatie
- Hoogste bytes startadres
- Startadres
- Grootte

Het attributenveld verdient nog wat aandacht. Elk van de 8 bits komt overeen met een bepaald attribuut, en duidt aan of dit attribuut op het element van toepassing is. Eén van de attributen is het “directory”-attribuut. Als hier 1 staat beschrijft dit element een subdirectory, anders een bestand. Voor ieder bestand zijn er de read-only, hidden, system en archive attributen.

Het Volume Label wordt normaal slechts voor één speciaal element van de root-directory gebruikt. Hiervan is handig gebruik gemaakt om in Windows de mogelijkheid te voorzien om langere bestandsnamen te gebruiken in FAT-besturingssystemen.

De lange naam wordt verspreid over bepaalde delen van meerdere tabelelementen. Het volumeattribuut van deze elementen wordt op 1 gezet, zodat MS-DOS ze negeert.

Voorbeeld 2: NTFS

NTFS staat voor New Technology File System. NTFS bestandsnamen kunnen 255 tekens lang zijn, en de tekens worden m.b.v. unicode bewaard.

Master File Table

NTFS maakt gebruik van een Master File Table (MFT) In deze tabel wordt voor ieder bestand 1 bestandsbeschrijving bijgehouden. Deze beschrijving bevat naast de bestandsnaam, de grootte en de toegangsdata informatie over de toegangsrechten tot het bestand.

De MFT bevat één record per bestand, en niet één record per blok op de schijf. Hierdoor is de grootte van het MFT variabel. Standaard wordt 12,5% van de partitie voorzien voor de MFT, en 87,5% voor data. Wanneer het datagebied echter vol is, en de MFT neemt niet de voorziene 12,5% in, kan deze ruimte ook voor data worden gebruikt.

Header | Standard Information | File Name | Datagebied | … | Security Descriptor

Een record in de MFT bevat attributen. Welke overige attributen gebruikt worden staat niet vast in de specificatie van de NTFS. De gebruikte attributen op de partitie worden in het systeembestand $AttrDef bijgehouden.

De standaardinformatie omvat de grootte van het bestand, de toegangsdatums en andere eigenschappen. De toegangsrechten worden bewaard in de security descriptor. Omdat attributen geen vaste lengte hebben is er een hoofding nodig die de structuur van het record beschrijft.

Wanneer de inhoud van een attribuut te groot wordt om in het MFT-record te bewaren wordt het attribuut non-resident gemaakt. Dit betekent dat de inhoud van het attribuut in vrije blokken in het datagebied bewaard wordt, en dat het MFT-record een verwijzing naar deze blokken bevat.

Het data-attribuut in een MFT-record bevat verwijzingen naar blokken in het datagebied. Om de toegangssnelheid te verhogen en de hoeveelheid adresinformatie te beperken wordt gebruik gemaakt van extents, die in de context van NTFS vaak data runs of streams genoemd worden.

Voor kleine bestanden wordt de inhoud van het bestand zelf binnen het data-attribuut in het MFT-record opgeslagen. Zo gauw het bestand is opgezocht is directe toegang mogelijk zonder schijftoegangen. Dit gebeurt typisch voor bestanden kleiner dan 1,5KB.

Wanneer een bestand zo groot wordt dat de verschillende data runs niet meer binnen het data-attribuut kunnen beschreven worden is het mogelijk om het data-attribuut zelf buiten het MFT-record op te slaan. Door deze flexibele structuur is er bij NTFS geen sprake van een maximale bestandsgrootte.

Directories

De logische structuur van NTFS is een boomstructuur, net als bij FAT. Voor directories wordt ook een record in de MFT aangemaakt. Een directory-record heeft extra attributen die verwijzingen bevatten naar de bestanden en subdirectories. Net als bij gewone bestanden worden deze gegevens voor een kleinere directory in het record bijgehouden.

De fysische structuur van de inhoud van een directory wordt bijgehouden m.b.v. B-bomen. Ingewikkelde shit die niet in de cursus staat.

Second Extended Filesystem

Het Extended Filesystem wordt beschouwd als het standaard-bestandssysteem van Linux.

Inodes

Net zoals Linux zelf is het Extended Filesystem, en dus ook de opvolgers ervan, gebaseerd op Unix. De blokken van de schijf worden ingedeeld in 4 soorten:

- Bootblok
- Superblok
- Inodeblokken
- Datablokken

In het bootblok staat bootstrap code die het besturingssysteem moet laden en opstarten. Het heeft geen specifieke functie in het bestandssysteem. Het superblok bevat globale gegevens over het bestandssysteem en enkele instellingen.

Met ieder bestand in ext2 wordt een bestandsstructuur geassocieerd, die inode wordt genoemd. Hierin vinden we alle metadata en de verwijzingen naar de datablokken waarin de inhoud van het bestand opgeslagen is. Achter het superblok staat een lijst met inodeblokken, soms ook I-list genoemd. De overige blokken in het bestandssysteem worden gebruikt om de inhoud van bestanden in weg te schrijven.

Een inode bevat dus allerlei gegevens, waaronder de eigenaar van het bestand en de toegangsrechten. Merk op dat de bestandsnaam niet in de inode wordt bewaard. De bestandsnamen vinden we in de directories, samen met een verwijzing naar de inode.

Een ext2-bestandsnaam kan tot 255 tekens bevatten. Zoals gezegd is een directory een bestand met daarin een lijst va bestandsnamen gekoppeld aan een inode. Het inode-nummer wordt opgeslagen, gevolgd door 2 variabelen die respectievelijk de lengte van de entry en de naam aangeven. Als laatste wordt de eigenlijke bestandsnaam opgeslagen. Doordat de lengte van de bestandsnaam gekend is kan er naar het volgende element gesprongen worden.

Bestanden kunnen in ext2 niet-aaneengesloten gealloceerd worden. In de inode is ruimte voorzien om de adressen van 13 blokken te bewaren. De eerste 10 zijn directe blokadressen, die verwijzen naar datablokken. Het 11de adres verwijst naar een indexblok, waarin de adressen van datablokken opgeslagen kunnen worden. Het 12de en 13de adres in de inode zijn respectievelijk dubbel en driedubbel indirect. Het 11de adres verwijs dus naar een datablok met daarin adressen van indexblokken.

Het grote voordeel van deze structuur is dat de adresinformatie voor kleine bestanden volledig in het werkgeheugen beschikbaar is. Met bestanden die maximaal 10 datablokken in beslag nemen kan dus erg snel gewerkt worden. De maximale bestandsgrootte in bytes wordt gegeven door:

- $Smax = [12 + (b/4) + (b/4)^2 + (b/4)^3]b$

Vrije blokken en inodes

In het superblok worden tabellen bijgehouden met de adressen van een aantal vrije datablokken en een aantal vrije inodes. In deze tabellen is echter niet voldoende plaats om alle vrije elementen te bewaren. Wanneer bijvoorbeeld een bestand moet worden uitgebreid, wordt een adres van een vrij blok uit de tabel in het superblok genomen. Bij het verwijderen van een bestand worden de adressen van de vrijgekomen blokken in de tabel ingeschreven. Indien de tabel leeg is op het ogenblik dat een blok aan de file moet worden toegekend zal de tabel uit het eerste blok in de gelinkte lijst naar het superblok worden gekopieerd. Het hierdoor vrijgekomen blok wordt aan het bestand toegevoegd. Als de tabel helemaal gevuld is met adressen, en er wordt een blok vrijgegeven, dan wordt de tabel uit het superblok naar het vrijgekomen blok gekopieerd, waarna dit als eerste blok in de gelinkte lijst wordt opgenomen.

Voor vrije inodes wordt een gelijkaardige methode gebruikt. In het superblok staat een tabel voor 100 nummers van vrije inodes. Wanneer een nieuw bestand wordt aangemaakt vindt men hier het nummer van een beschikbare inode. Bij het verwijderen van een bestand wordt het nummer van de vrijgekomen inode toegevoegd aan de tabel.

Het veld voor het bestandstype staat op 0 als de inode niet in gebruik is. Wanneer de tabel in het superblok leeg is, kan de kernel de i-list doorlopen en de nummers van de 100 eerste vrije inodes in de tabel plaatsen.

Om het zoeken naar vrije inodes zo efficiënt mogelijk te laten verlopen, wordt de positie van de laatst gevonden vrije inode apart bijgehouden, om de volgende zoektocht vanop die positie te kunnen starten. Wanneer er een inode vrijkomt door het verwijderen van een bestand zijn er 2 mogelijkheden. Als de positie van deze inode voor de opgeslagen positie komt wordt de opgeslagen positie erdoor vervangen. Als de positie erna komt moet er niets gebeuren, want dan wordt de nieuwe vrije inode sowieso gevonden bij één van de volgende zoektochten. Harde en zachte links

Omdat de bestandsbeschrijving niet in de directory bewaard wordt, kan men op een eenvoudige en effectieve manier eenzelfde bestand via verschillende paden aanspreken. Men laat in dit geval 2 bestanden verwijzen naar eenzelfde inode, en dus ook naar hetzelfde bestand. Dit wordt een hard link genoemd.

Voor symbolische links wordt verwezen naar een ander pad, niet naar een inode.

Bij het verwijderen mogen de inode en datablokken pas verwijderd worden als er geen enkel link meer is. Daarom wordt in een inode een teller met het aantal links bijgehouden. Bij symbolic links is dit niet nodig. Wanneer het bestand waarnaar een symbolic link verwijst verwijderd wordt, blijft de link bestaan. Deze wordt dan wees of orphan genoemd.

Bijzonderheden

Ext 2 heeft enkele specifieke eigenschappen.

Het aantal inodes wordt bepaald bij het aanmaken van het bestandssysteem. Het kan gebeuren dat je geen nieuw bestand kan aanmaken, hoewel er toch datablokken beschikbaar zijn. Een ext2 bestandssysteem heeft dus een maximum aantal bestanden. Standaard wordt er één inode aangemaakt voor iedere 4096 bytes beschikbare schijfruimte. Als de gemiddelde grootte van je bestanden kleiner is dan 4 kB, kan je dus snel met een tekort aan inodes kampen.

Bij het creëren van het bestandssysteem wordt een deel (standaard 5%) toegekend aan de beheerder of root. Zo kan een gebruiker niet alles volschrijven. Een laatste bijzonder bestandstype is de zogenaamde device file. Deze bestanden bevatten geen data, maar stellen een systeemapparaat voor. Zo kan men communiceren met een apparaat door te lezen of schrijven van of naar de bijhorende device file.

2010 herhalingsvragen boek Stallings

Hoofdstuk 1

1.1 Benoem en beschrijf beknopt de vier belangrijkste elementen van een computer.

- - De processor beheert de werking van de computer en voert de functies voor gegevensverwerking uit
- - Het hoofdgeheugen bevat gegevens en programma’s? Dit geheugen is meestal vluchtig. Hoofdgeheugen wordt ook wel reëel of primair geheugen genoemd.
- - I/O-modules verplaatsen gegevens tussen de computer en zijn externe omgeving. De externe omgeving bestaat uit uiteenlopende apparaten zoals secundair geheugen, diskettes, communicatieapparatuur en werkstations.
- - De systeembus verzorgt de communicatie tussen de processors, het hoofdgeheugen en de I/O-modules.

1.2 Definieer de twee belangrijkste categorieën processorregisters.

- - Registers die zichtbaar zijn voor de gebruiker: bieden de programmeur om in machine- of assembleertaal verwijzingen naar het hoofdgeheugen te minimaliseren door registergebruik te optimaliseren.
- - Stuur en statusregisters: worden gebruikt door de processor voor het besturen van de werking van de processor en door geprivilegieerde routines van het besturingssysteem.

1.3 Wat zijn, in algemene termen, de vier te onderscheiden acties die een machine-instructie kan specificeren?

- - Processorgeheugen: gegevens kunnen worden overgebracht van de processor naar het geheugen of van het geheugen naar de processor
- - Processor-I/O: gegevens kunnen worden uitgewisseld met een randapparaat door ze over te sturen tussen de processor en een I/O-module.
- - Gegevensverwerking: de processor kan een rekenkunde of logische bewerking uitvoeren op gegevens
- - Besturing: een instructie kan de volgorde van uitvoering wijzigen. Bv in instructie 149 staat dat er naar 182 moet gesprongen worden.

1.4 Wat is een interrupt?

Een interrupt is feitelijk een onderbreking. Hiermee kan de processor andere instructies uitvoeren terwijl een tragere bewerking aan de gang is. Als deze bewerking gedaan is, stuurt het betreffende apparaat een interrupt, waardoor de processor weet dat hij verder kan gaan met bv de I/O.

1.5 Op welke manier worden de verschillende interrupts afgehandeld?

Indien bedoeld bij meervoudige interrupts:

- o Uitschakelen van interrupts terwijl er een interrupt wordt verwerkt, na afhandeling doet de processor deze
- o Nesten van interrupts: als er een interrupt bezig is, en er komt een nieuwe aan, met een hogere prioriteit, dan wordt deze genest, hierna wordt er verder gegaan met de eerste.

1.6 Welke kenmerken onderscheiden de verschillende elementen van een geheugenhiërarchie?

Als men van boven naar onder de hiërarchie doorloopt, gebeurd het volgende:

1. a. De kosten per bit dalen
2. b. De capaciteit neemt toe
3. c. De toegangstijd neemt toe
4. d. De frequentie van de processortoegang tot het geheugen neemt af

1.7 Wat is cachegeheugen?

Een klein, snel geheugen tussen de processor en het hoofdgeheugen. Het bevat een klein gedeelte van het hoofdgeheugen dat hierdoor veel sneller bereikbaar is. De cache werkt dus als een buffer tussen het hoofdgeheugen en de processor, waarmee een intern geheugen ontstaat met 2 niveaus,

Dit geeft betere prestaties dan met 1 niveau. (Dankzij lokaliteit).

1.8 Benoem en beschrijf beknopt drie technieken voor I/O-bewerkingen.

- Geprogrammeerde I/O: Processor doet oproep voor I/O-bewerking, I/O-module geeft echter geen waarschuwing wanneer deze klaar is, de processor zal zelf periodiek moeten kijken of de module al klaar is. Instructieset bevat daarom volgende categorieën:

1. o Besturen: extern apparaat activeren en vertellen wat het moet doen
2. o Status controleren
3. o Lezen en schrijven

- Tijdverslinden proces dat de processor onnodig bezighoudt
- - Interruptgestuurde I/O: Processor geeft opdracht voor I/O en gaat dan verder met iets anders, wanneer I/O-module klaar is, stuurt de module een interrupt, processor voert dan de gegevensoverdracht uit en hervat de vorige verwerking.
- Dit is efficiënter doordat de processor niet meer nodeloos moet wachten, maar het verbruikt nog steeds veel processortijd.
- - Directe geheugentoegang (DMA): De processor geeft een opdracht tot I/O, de DMA-module handelt de bewerking verder af zonder tussenkomst van de processor. Als dit klaar is stuurt de DMA-module een interrupt naar de processor. De processor is dus enkel betrokken bij het begin en het einde.

Hoofdstuk 2

2.1 Wat zijn de drie doelen van een ontwerp van een besturingssysteem?

- - Gemak: het maakt een computer makkelijker te gebruiken.
- - Efficiëntie: dankzij een besturingssysteem kunnen de systeembronnen efficiënt gebruikt worden
- - Flexibiliteit: Een besturingssysteem moet zo zijn gemaakt dat het ontwikkelen, testen en introduceren van nieuwe systeemfuncties zo goed mogelijk is.

2.2 Wat is de kernel van een besturingssysteem?

De nucleus, dit bevat de meest gebruikte functies in het besturingssysteem. Bevindt zich in het hoofdgeheugen.

2.3 Wat is multiprogrammering?

Meerdere programma’s in het geheugen laden, en wanneer er een programma wacht op I/O, overschakelen naar een ander.

2.4 Wat is een proces?

- - Een programma dat wordt uitgevoerd
- - Een geactiveerd programma dat op een computer draait
- - De entiteit die kan worden toegewezen aan en worden uitgevoerd door een processor
- - Een eenheid van activiteit die wordt gekenmerkt door een enkelvoudig e, sequentiële verwerking (thread), een actuele toestand en een aantal bijbehorende systeembronnen

2.5 Hoe wordt de uitvoeringscontext van een proces door het besturingssysteem gebruikt?

Het wordt apart gehouden van het proces zelf. Hierin zitten bv registerinhouden.

2.6 Benoem en beschrijf beknopt vijf opslagmanagementverantwoordelijkheden van een standaardbesturingssysteem.

- - Procesisolatie: Het besturingssysteem moet voorkomen dat onafhankelijke processen elkaars geheugen, gegevens of instructies, verstoren.
- - Automatische toewijzing en beheer. Programma’s moeten waar nodig dynamisch worden verdeeld over de geheugenhiërarchie
- - Ondersteuning voor modulair programmeren.
- - Bescherming en toegangsbeheer. Door et delen van geheugen ontstaat de mogelijkheid dat een programma de geheugenruimte van een ander programma adresseert. Soms is dit gewenst, soms brengt het de stabiliteit in gevaar. Het besturingssysteem moet toestaan dat gedeelten van het geheugen op verschillende manieren toegankelijk zijn.
- - Langetermijnopslag. Veel gebruikers en toepassingen vereisen middelen voor het opslaan van informatie gedurende langere perioden.

2.7 Bespreek het verschil tussen een reëel en virtueel adres?

Het virtuele adres wordt door het proces gebruikt om een woord aan te spreken, het bestaat uit een paginanummer en de offset. Het reëele adres is het eigenlijke adres binnen het geheugen.

2.8 Beschrijf de round-robin schedulingtechniek.

Alle processen in een wachtrij krijgen om beurten wat tijd van de scheduler.

2.9 Bespreek het verschil tussen een monolithische kernel en een microkernel.

Monolytische kernel is één kolossale kernel, deze verzorgt het overgrote deel van de functionaliteit van een besturingssysteem. Dit is meestal ook maar één groot proces met gedeelde adresruimte. Een microkernel is een kleine kernel die slechts enkele essentiële functies bevat, de andere diensten worden verzorgd door processen (of servers) die worden uitgevoerd in gebruikersmodus en worden behandeld zoals elk ander proces.

2.10 Wat is multithreading?

Een techniek waarbij een proces dat een toepassing uitvoert, wordt verdeeld in draden (threads) die tegelijkertijd kunnen uitgevoerd worden.
Een thread omvat een context (programmateller, registers,..) en een eigen gegevensruimte. Een thread wordt sequentieel uitgevoerd.
Een proces is een verzameling van een of meer threads.

Hoofdstuk 3

3.1 Wat is een instructiespoor?

De volgorde van instructies die voor een proces worden uitgevoerd

3.2 Welke algemene gebeurtenissen geven aanleiding tot het creëren van een proces?

- - Nieuwe batchtaak
- - Interactieve aanmelding (logon)
- - Gecreëerd door het besturingssysteem om een dienst te verzorgen

- - Verwekt door een bestaand proces

3.3 Geef een korte definitie van iedere toestand in het procesmodel van figuur 3-6.

- - Actief (running): proces wordt uitgevoerd
- - Gereed (ready): proces dat meteen uitgevoerd kan worden als het de gelegenheid krijgt
- - Geblokkeerd (blocked) een proces dat niet uigevoerd kan worden totdat een bepaalde gebeurtenis optreedt
- - Nieuw (new): een proces dat onlings is gecreeerd maar nog niet door het systeem is toegevoegd aan de verzameling met uitvoerbare processen
- - Einde (exit): een proces dat door het besturingssysteem wordt ontslagen uit de groep uitvoerbare processen

3.4 Wat wordt bedoeld met het preëmptief onderbreken van een proces?

Een proces keert terug van kernelmodus naar gebruikersmodus maar de kernel onderbreekt het preemptief* om een proces met hogere prioriteit dat gereed staat in te roosteren. Ten opzichte van het toedelen (dispatching) vormen de processen in de twee toestanden preempted en ready to run in Memory echter één wachtrij.

preemptief: zonder dat het volledig is uitgevoerd of moet wachten op een gebeurtenis

3.5 Wat betekent het begrip swappen en wat is het doel ervan?

Het verplaatsen van een deel of het geheel van een proces van het hoofdgeheugen naar schijf. Om plaats in het hoofdgeheugen vrij te maken door geblokkeerde processen naar de schijf te verplaatsen en deze dan de status opgeschort (suspended) mee te geven.

3.6 Waarom bevat figuur 3-9 twee geblokkeerde toestanden?

Er is een geblokkeerde toestand die nog in het hoofdgeheugen zit, de andere is al geswapped, en bevindt zich nu op de schijf.

3.7 Geef vier kenmerken van een opgeschort proces.

- - Het proces is niet onmiddellijk beschikbaar voor uitvoering
- - Het proces kan al of niet wachten op een gebeurtenis.
- - Het proces werd in de toestand Opgeschort geplaatst door het proces zelf of door een ander om de uitvoering van het proces te verhinderen
- - Het proces kan deze toestand niet verlaten tenzij een ander hiervoor expliciet een opdracht geeft

3.8 Voor welke soorten entiteiten houdt het besturingssysteem informatietabellen bij vanuit beheersoverwegingen?

- ○ - Geheugen
- - I/O
- - Bestand
- - Proces

3.9 Geef drie algemene informatiecategorieën in een procesbesturingsblok.

- Procesidentificatie
- Processortoestandinformatie

1. Registers
2. stack

- Procesbesturingsinformatie

1. o Toestand
2. o prioriteit

3.10 Waarom zijn twee verwerkingsmodi (gebruiker en kernel) nodig?

Het besturingssysteem en de belangrijkste tabellen van het besturingssysteem, zoals de procesbesturingsblokken moeten beschermd worden tegen verstoringen door gebruikersprogramma’s. In de kernelmodus heeft de software volledige controle over de processor en alle bijbehorende instructies, registers en geheugen. Dit is niet gewenst voor gebruikersprogramma’s.

3.11 Welke stappen voert het besturingssysteem uit bij het creëren van een nieuw proces?

1. Het besturingssysteem wijst een unieke procesidentificatie toe aan het nieuwe proces.
2. Het besturingssysteem wijst ruimte toe aan het proces
3. Het procesbesturingsblok moet worden geïnitialiseerd
4. De juiste koppelingen moeten worden ingesteld
5. Soms moeten andere gegevensstructuren worden gemaakt of uitgebreid

3.12 Wat is het verschil tussen een interrupt en een val (trap)?

Interrupt wordt veroorzaakt door een of andere gebeurtenis die zich buiten het actieve proces bevindt en daarvan onafhankelijk is, bv voltooiing I/O
Val hangt samen met een fout of uitzonderingsconditie die wordt gegenereerd binnen het actieve proces zelf, bv ongeldige poging tot bestandstoegang

3.13 Geef drie voorbeelden van een interrupt.

- - Klokinterrupt: Proces heeft maximaal toegestane uitvoeringstijd benut
- - I/O-interrupt: Een I/O actie is opgetreden. Indien dit een gebeurtenis is waar ene proces op wacht, wordt dit proces van geblokkeerd naar gereed veranderd
- - Geheugenfout: Processor krijgt een verwijzing naar een woordadres in het virtuele geheugen dat zich niet in het hoofdgeheugen bevindt. Processor geeft opdracht tot overzetten van secundair naar hoofdgeheugen en gaat dan verder met een ander proces. Als het overzetten klaar is, komt er een interrupt.

3.14 Wat is het verschil tussen een moduswisseling en een proceswisseling?

Moduswisseling is het wisselen van gebruikers naar kernel modus voor bv een interrupt, een proceswisseling is het veranderen van een status van een proces. Bv, van geblokkeerd naar gereed.

Hoofdstuk 12

12.1 Wat is het verschil tussen een veld en een record?

Een veld is een basiselement van gegevens, het bevat één waarde, bv achternaam, datum.
Een record is een verzameling gerelateerde velden die door bepaalde toepassingsprogramma's als een eenheid kunnen worden behandeld. Het bevat dus velden.

12.2 Wat is het verschil tussen een bestand en een database?

Een bestand (file) is een verzameling vergelijkbare records. Gebruikers en toepassingen behandelen het bestand als één entiteit en verwijzen ernaar aan de hand van een naam. Heeft een unieke bestandsnaam.
Een database is een verzameling gerelateerde gegevens. Er bestaan expliciete relaties tussen gegevenselementen. Het bestaat uit een of meerdere bestanden.

12.3 Wat is een systeem voor bestandsbeheer?

Een verzameling systeemsoftware die diensten verzorgt voor gebruikers en toepassingen bij het gebruiken van bestanden.

12.4 Welke criteria zijn van belang bij het kiezen van een bestandsorganisatie(=logische structuur)?

- - Korte toegangstijd
- - Gemak van bijwerken
- - Efficiënt gebruik van opslagruimte
- - Onderhoudsgemak
- - Betrouwbaarheid

12.5 Noem en beschrijf in het kort vijf vormen van bestandsorganisatie.

- - Stapelbestand: eenvoudigste soort bestandsorganisatie, het is een stapel (pile). Gegevens worden verzameld in de volgorde waarin ze binnenkomen. Geen structuur, dus altijd volledig doorzoeken.
- - Sequentieel bestand: Vaste indeling van records, alle records hebben dezelfde lengte een bestaan uit hetzelfde aantal velden met vaste lengte in specifieke volgorde. Bevat een sleutelveld.
- - Index-sequentieel bestand: Omzeilt nadelen van sequentieel. Behoudt de hoofdkenmerk: records zijn ingedeeld op volgorde op basis van het sleutelveld. Hieraan zijn 2 kenmerken

toegevoegd: een index naar het bestand voor het ondersteunen van willekeurige toegang en een overloopbestand (overflow file)

- - Geïndexeerd bestand: Dit gebruikt meerdere indexen, één voor elk veldtype dat zou kunnen worden gebruikt bij het zoeken.
- - Direct of hashed bestand: gebruikt mogelijkheid van schijven om directe toegang te geven tot elk blok met een bekend adres. Sleutelveld in elke record is vereist. Wordt vaak gebruikt als snelle toegang vereist is.

12.6 Waarom is de gemiddelde zoektijd voor het vinden van een record bij een index-sequentieel bestand minder dan bij een sequentieel bestand?

Het maakt gebruik van verwijzingen, waardoor men sneller op een bepaald veld kan uitkomen. Bv een sequentieel bestand van één miljoen records. Het zoeken naar een bepaalde sleutelwaarde vereist gemiddeld een half miljoen recordtoegangen. Als er nu echter een index met duizen ingangen wordt gemaakt. Dan kost het zoeken van de records gemiddeld 500 toegangen tot het indexbestand, en dan nog eens 500 tot het hoofdbestand. De gemiddelde zoeklengte wordt dus verminderd van een half miljoen tot duizend.

12.7 Wat zijn gebruikelijke bewerkingen die men kan uitvoeren bij een directory?

- - Zoeken
- - Creëren
- - Verwijderen
- - Weergeven
- - Bijwerken

12.8 Wat is het verband tussen een padnaam en een werkdirectory?

De volledige padnaam elke keer uitspellen is vervelend, hiervoor gebruikt men de werkdirectory van waaruit men relatief verwijst naar een bestand.

12.9 Wat zijn gangbare toegangsrechten die kunnen worden toegekend of geweigerd aan een individuele gebruiker van een bepaald bestand?

- - Geen
- - Kennis
- - Uitvoeren
- - Lezen

- - Toevoegen
- - Bijwerken
- - Beveiliging wijzigen
- - Verwijderen

12.10 Noem en beschrijf in het kort drie manieren van blokvorming.

- - Blokvorming met vaste lengte: Hierbij worden records met vaste lengte gebruikt en wordt een geheel aantal records opgeslagen in een blok. Er kan ongebruikte ruimte zijn aan het einde van elk blok; dit wordt aangeduid als interne fragmentatie
- - Gekoppelde blokvorming met variabele lengte: hierbij worden records met variabele lengte gebruikt en deze worden samengepakt in blokken, alle ruimte wordt gebruikt. Sommige records kunnen daarbij twee blokken omspannen, waarbij het vervolg wordt aangegeven door een verwijzing naar het volgende blok
- - Niet-gekoppelde blokvorming met variabele lengte: hierbij worden records met variabele lengte gebruikt, maar deze worden niet gekoppeld opgeslagen. De meeste blokken bevatten loze ruimte, omdat het onmogelijk is de rest van een blok te gebruiken als de volgende record groter is dan de resterende ruimte

12.11 Noem en beschrijf in het kort drie vormen van bestandstoewijzing

- - Aaneengesloten toewijzing: Een aaneengesloten verzameling blokken wordt toegewezen aan een bestand op het moment dat het bestand wordt gecreëerd. Tabel voor bestandstoewijzing bevat maar één ingang voor elke bestand te bevatte, die het beginblok en de lengte aangeeft.
- - Kettingtoewijzing: Toewijzing vindt plaats op basis van aparte blokken. Elk blok bevat een wijzer naar het volgend blok in de ketting. Tabel voor bestandstoewijzing bevat maar één ingang voor elke bestand te bevatte, die het beginblok en de lengte aangeeft.
- - Indextoewijzing: Hierbij bevat de tabel voor bestandstoewijzing een afzonderlijke index van één niveau voor elk bestand; de index heeft één ingang voor elke portie die aan het bestand is toegewezen

2010 juni examen

Strypsteen

- Leg het bootproces van een computer uit met als actieve partitie Windows XP. (11 stappen)
- Interruptverwerking bestaat uit 9 stappen.
  - Bij welke 4 stappen word de stack gebruikt ?
  - Geef ook bij elke stap welke hardware of software deze stap uitvoert
- Leg multiprogramming uit (schema gegeven, gewoon in te vullen)
- Processen
  - Waarom zij er opgeschorte processes ?
  - Welke 2 soorten opgeschorte proccessen zijn er ?
  - Leg de 2 processen uit.
  - Hoe komt een proces in deze statussen?
- Oefening op Fat 12

Geens

- Bespreek unix procestoestand diagram: entiteiten en relaties
- Bespreek en los op: unix inode(-12?)
- leg bootproces uit en leg connectie met linux of windows
- inzichtvraag: fat-labo: een vriend gaat op vakantie maar wist perongeluk wat foto's van zijn camera met flashkaartje. Welke tips heb je voor hem? hoe kan je de foto's recupereren met enkel een hex-editor. Leg stap voor stap uit.

2010 samenvatting van Onbekend

Hoofdstuk 1 Overzicht van computersystemen

Interrupts

Onderbreken de normale verwerking van een processor
De meeste I/O apparaten zijn trager dan de processor, de processor moet dus wachten op het apparaat. -> verspilling van processortijd

4 klassen van Interrupts:
- Programma: resultaat van uitvoering van een instructie. Bv delen door nul
- Timer: gegenereerd door een timer binnen de processor
- I/O: gegenereerd door een I/O-controller om de normale voltooiing van een bewerking of foutcondities te melden
- Hardwarefout: geheugenpariteitsfout, uitvallen van stroom, ..

Processor controleert of er een interrupt is opgetreden, indien dit zo is dan onderbreekt de processor het huidige programma en voert hij een routine uit voor interruptafhandeling

Interruptverwerking

1. Apparaatcontroller of andere systeemhardware geeft een interrupt
2. Processor voltooit de uitvoering van de huidige instructie
3. Processor bevestigt de ontvangst van de interrupt
4. Processor plaatst het programmastatuswoord (PSW) en de programmateller (PC) op de besturingsstack
5. Processor laadt nieuwe PC-waarde op basis van de interrupt
6. Sla het restant van de informatie over de processtatus op
7. Verwerk de interrupt
8. Herstel de informatie over de processtatus
9. Herstel het oude PSW en de oude PC
- 1 te.m. 5: Hardware
- 6 t.e.m. 9: Software

Multiprogramming

Processor heeft meer dan één programma om uit te voeren. Als de processor wacht op een I/O-module, gaat hij beginnen aan de uitvoering van een ander proces. Krijgt hij een interrupt dan gaat hij terug naar de eerste. Volgorde hangt af van hun relatieve prioriteit.
Komt er dus hierna een ander programma met hogere prioriteit, zal het tweede programma moeten wachten.

I/O technieken

1. Geprogrammeerde I/O
I/O-module doet de gevraagde actie, bv enkele bytes in buffer plaatsen. Hierna plaatst het de juiste bits in het I/O statusregister. Er komt geen interrupt. De processor moet regelmatig kijken of de I/O-instructie voltooid is.
=> tijdsverslindend proces dat de processor onnodig bezighoudt
2. Interruptgestuurde IO
I/O-module stuurt een interrupt wanneer het klaar is om gegevens uit te wisselen. Processor slaat context van het huidige programma op en begint aan de interruptafhandeling
=> Geen nodeloos wachten, maar gebruikt nogaltijd veel processortijd doorat de processor elk

geschreven of gelezen woord langs de processor moet.
3. Directe geheugentoegang (DMA)
Verstuurt een blok data meteen van of naar het geheugen. Een interrupt wordt gestuurd wanneer de overdracht compleet is. Dit is veel efficiënter.

Hoofdstuk 2 Overzicht van besturingssystemen

Besturingssysteem: Programma dat de uitvoering van toepassingsprogramma's regelt en de functie vervult van een interface tussen de gebruiker van de computer en de computerhardware.
Drie doelstellingen:
- Gemak: gemakkelijker te gebruiken
- Efficiëntie: Systeembronnen efficiënt gebruiken
- Flexibiliteit: ontwikkelen,testen en introduceren van nieuwe systeemfuncties mogelijk
Besturingssysteem verzorgt volgende diensten:
- Ontwikkelen van programma's
o Editors en debuggers
- Uitvoeren van programma's
- Toegang tot I/O-apparaten
- Beheerde toegang tot bestanden
- Toegang tot het systeem
- Opsporen en afhandelen van fouten
o Interne en externe hardwarefouten
o Softwarefouten
o Verzoek van toepassing waaraan het besturingssysteem niet aan kan voldoen
- Administratie
o Gebruiksstatistieken
o Prestatieparameters bewaken
o Anticiperen op behoefte
o Toerekenen van kosten
Het besturingssysteem is verantwoordelijk voor beheer van systeembronnen.
Het werkt hetzelfde als gewone computersoftware: het is een programma dat wordt uitgevoerd door de processor.
Het geeft de besturing veelvuldig uit de handen en is voor het terugkrijgen afhankelijk van de processor.

Kernel

Deel van het besturingssysteem dat zich in het hoofdgeheugen bevindt.
Het bevat de meest gebruikte functies. Wordt ook wel eens nucleus genoemd.

Evolutie van besturingssystemen

Een besturingssysteem zal zich om een aantal redenen in de loop van de tijd ontwikkelen:
- Hardwarevernieuwing en nieuwe typen hardware. Bv Paginerende hardware, grafische werkstations
- Nieuwe diensten: wensen van gebruikers
- Verbeteringen: verbeteren van fouten in BS

Seriële verwerking

Men werkte meteen met de hardware, er was geen besturingssysteem. Het werd bestuurd via een bedieningspaneel met lampjes, schakelaars, een invoerapparaat en een printer.
Twee problemen:
- Scheduling: reserveren van machinetijd. Reserveerde je een uur, en had je maar 45 min nodig was het verspilling, had je meer nodig, dan moest je stoppen.
- Insteltijden: één programma bestond uit meerdere stappen, zat hier een fout in, dan moest je teruggaan naar het begin.

Eenvoudige batchsystemen

Gebruik van een monitor. Dit is software die de volgorde controleerde. Gebruiker gaf meerdere opdrachten aan operator, deze maakte hier een batch van en gaf deze aan de monitor. Na uitvoering hiervan werd er teruggegaan naar de monitor die dan weer verder ging met het volgende programma.
Job Control Language: Speciaal type van programmeertaal (jobbesturingstaal) die wordt gebruikt voor het opgeven van instructies aan de monitor. Zoals welke compiler te gebruiken, welke data te gebruiken.
Hardwaremogelijkheden:
- Geheugenbeveiliging: Een gebruikersprogramma mag het geheugengebied met de monitor niet wijzigen
- Timer: Een taak krijgt maar een bepaalde tijd voor een taak, verloopt deze wordt er een interrupt gestuurd.
- Geprivilegieerde instructies: Sommige instructies mogen enkel door de monitor uitgevoerd worden
- Interrupts: interrupts verhogen de flexibiliteit bij het afstaan en verkrijgen van besturing aan en van gebruikersprogramma’s
Geheugenbeveiliging:
Gebruikersprogramma’s worden uitgevoerd in gebruikersmodus: bepaalde instructies mogen niet uitgevoerd worden.
Monitor wordt uitgevoerd in kernelmodus: geprivilegieerde instructies mogen uitgevoerd worden, beveiligde geheugengebieden mogen benaderd worden.

Timesharing systemen

Multiprogramming gebruiken om meerdere interactieve jobs af te handelen. Meerdere gebruikers gebruiken hetzelfde systeem dmv terminals. Elke gebruiker krijgt kleine periodes van processortijd wanneer dit nodig is.

Belangrijke prestaties

Vijf belangrijke intellectuele prestaties verricht op het gebied van:
- Processen
- Geheugenbeheer
- Bescherming en beveiliging van informatie
- Scheduling en beheer van bronnen
- Systeemstructuur

Processen

- Een programma dat wordt uitgevoerd
- Een geactiveerd programma dat op een computer draait
- De entiteit die kan worden toegewezen aan en worden uitgevoerd door de processor
- Een eenheid van activiteit die word gekenmerkt door een enkelvoudige sequentiële verwerking (thread), een actuele toestand en een aantal bijbehorende systeembronnen

Vier hoofdoorzaken van fouten:
- Onjuiste synchronisatie: proces wacht op I/O maar signaal komt niet of dubbel aan
- Mislukte wederzijdse uitsluiting: Twee processen mogen niet hetzelfde bestand tegelijk aanpassen
- Niet-vastomschreven programmawerking: als er gedeeld geheugen is tussen 2 programma’s, en een van de twee overschrijft dit op onvoorstelbare wijze, kan het andere onstabiel worden.
- Dodelijke omarming (deadlock): twee programma’s zitten vast omdat ze op elkaar wachten.

Proces bestaat uit drie delen:
- Een uitvoerbaar (executable) programma
- De bijbehorende gegevens die het programma nodig heeft (variabelen, werkruimte, buffers)
- Uitvoeringscontext van het programma (inhoud gegevensregisters, programmateller, prioriteit,..)

Geheugenbeheer

Vijf hoofdtaken:
- Procesisolatie: onafhankelijke processen mogen elkaars geheugen niet verstoren
- Automatische toewijzing en beheer: programma’s moeten juist verdeeld worden over geheugenhiërarchie
- Ondersteuning van modulair programmeren
- Bescherming en toegangsbeheer
- Langetermijnopslag

Virtueel geheugen

- Zorgt voor opslag op lange termijn, informatie word opgeslagen in bestanden
- Geheugen op logisch niveau adresseren

Paginering

- Proces opdelen in een aantal blokken van vaste lengte (pagina’s)
- Virtueel adres bestaat uit paginanummer en een relatieve locatie binnen de pagina (offset)
- Een pagina kan zich overal in het hoofdgeheugen bevinden
- Reëel adres is het geheugenadres

Bescherming en beveiliging van informatie

Vier categorieën:
- Beschikbaarheid: beveiligen van systeem tegen onderbrekingen
- Vertrouwelijkheid: Geen gegevens lezen waarvoor ze geen toestemming hebben
- Gegevensintegriteit: bescherming van gegevens tegen ongewenst gebruik
- Authenticiteit: correcte verificatie van identiteit en geldigheid van gegevens

Scheduling en beheer van bronnen

Drie overwegingen:
- Rechtvaardigheid: evenredige en rechtvaardige toegang tot een bron
- Gedifferentieerd reactievermogen: onderscheid maken tussen verschillende klassen taken die een andere afhandeling vereisen. Bv I/O voorlaten om het apparaat vrij te maken voor andere processen
- Efficiëntie: Doorvoer maximaliseren, antwoordtijd minimaliseren en zoveel mogelijk gebruikers tegelijk ondersteunen

Systeemstructuur

Het systeem is een serie van niveaus. Elk niveau heeft een eigen deelverzameling van functies. Elk niveau is afhankelijk van het volgende lagere niveau voor meer primitieve functies.
Dit splitst het probleem in beter hanteerbare deelproblemen.

Moderne besturingssystemen

Microkernelarchitectuur

Slechts enkele essentiële functies worden toegewezen aan de kernel. Waaronder adresruimten, communicatie tussen processen (IPC) en basisscheduling. Rest wordt afgehandeld door servers.

Multithreading

Proces wordt onderverdeeld in draden die tegelijkertijd kunnen uitgevoerd worden.
Een draad is een inzetbare eenheid werk, het wordt sequentieel uitgevoerd. Een proces is een verzameling van een of meer threads.

Symmetrische multiprocessing

- Meerdere processors
- Deze processors delen hetzelfde hoofdgeheugen en dezelfde I/O-voorzieningen
- Alle processors kunnen dezelfde functies uitvoeren

Het besturingssysteem van een SMP verdeelt processen of threads over alle processors.
Voordelen:
- Prestaties: meerdere processen tegelijkertijd actief
- Beschikbaarheid: het uitvallen van één processor stopt het systeem niet
- Stapsgewijze groei: de prestaties kunnen verbeteren door extra processoren toe te voegen
- Schaalbaarheid

Hoofdstuk 3 Beschrijving en besturing van processen

Vereisten waaraan een besturingssysteem moet voldoen in termen van processen:
- Het besturingssysteem moet de uitvoering van meerdere processen verweven om de processortijd te maximaliseren en tegelijk zorgen voor een aanvaardbare antwoordtijd.
- Het besturingssysteem moet bronnen toewijzen aan processen
- Communicatie tussen processen en creëren van processen door de gebruiker ondersteunen

Wat is een proces?

Concepten

1. Een computer bestaat uit een verzameling hardwarebronnen
2. Computerprogramma’s zijn ontwikkeld om een bepaalde taak uit te voeren
3. Het is inefficiënt om toepassing te schrijven voor een specifiek computersysteem
4. Het besturingssysteem werd ontwikkeld als veilige en consistente interface voor toepassingen
5. Het besturingssysteem is een uniforme, abstracte representatie van bronnen die door toepassingen benut kunnen worden.

Beheren van uitvoering van toepassingen:
- Bronnen beschikbaar maken voor verschillende toepassingen
- Fysieke processor schakelt tussen verschillende toepassingen, zodat deze allemaal verder worden uitgevoerd
- Processor en I/O-apparaten efficiënt kunnen worden gebruikt

Elementen van een proces:
- Identificatienummer
- Toestand
- Prioriteit
- Programmateller
- Geheugenwijzers
- Contextgegevens
- I/O-toestandinformatie
- Beheersinformatie

Dit wordt opgeslagen in een procesbesturingsblok. Dit wordt gecreëerd en beheerd door het besturingssysteem. Hierdoor is multiprogrammering mogelijk, de huidige waarden worden dan opgeslagen in het procesbesturingsblok.

Procestoestanden

Spoor van een proces: Volgorde van instructies die voor dat proces worden uitgevoerd
Dispatcher wijst de processor wisselend toe aan een ander proces.

Procesmodel met twee toestanden

Proces kan zich in twee toestanden bevinden: Actief en niet-actief

Creëren van processen

Redenen:
- Nieuwe batchtaak
- Interactieve aanmelding
- Gecreëerd door het besturingssysteem om een dienst te verzorgen
- Verwekt door een bestaand proces

Beëindigen van processen

Redenen:
- Normale voltooiing
- Tijdslimiet overschreden
- Onvoldoende geheugen beschikbaar
- Overtreding geheugengrens
- Beschermingsfout
- Rekenkundige fout
- Tijd verstreken
- I/O-fout
- Ongeldige instructie
- Geprivilegieerde instructie
- Onjuist gebruik van gegevens
- Ingreep van de gebruiker of het besturingssysteem
- Beëindiging van het ouderproces
- Verzoek van het ouderproces

Procesmodel met vijf toestanden

De dispatcher kan niet zomaar het proces selecteren dat het langst in de wachtrij zit want het kan op geblokkeerd staan.
- Actief
- Gereed
- Geblokkeerd
- Nieuw
- Einde

Opgeschorte processen

De processor is sneller dan I/O, dus alle processen zouden op I/O kunnen wachten. Deze programma’s worden verplaatst van het hoofdgeheugen naar een wachtrij Opgeschort op de schijf zodat er plaats is voor andere processen.
Twee nieuwe toestanden:

- Geblokkeerd – opgeschort: het proces bevindt zicht in het secundaire geheugen en wacht op een gebeurtenis
- Gereed – opgeschort: het proces bevindt zich in het secundaire geheugen maar is beschikbaar voor uitvoering zodra het in het hoofdgeheugen is geladen

Definitie:
1. Het proces is niet onmiddellijk beschikbaar voor uitvoering
2. Het proces kan al of niet wachten op een gebeurtenis. Wacht het, dan is deze toestand Geblokkeerd onafhankelijk van de toestand Opgeschort en zal het optreden van de blokkerende gebeurtenis het proces niet automatisch geschikt maken voor uitvoering
3. Het proces werd in de toestand Opgeschort geplaatst door het proces zelf of door een ander (ouderproces of besturingssysteem) om de uitvoering van het proces te verhinderen
4. Het proces kan deze toestand niet verlaten tenzij een ander hiervoor expliciet een opdracht geeft

Redenen voor het opschorten van processen:
- Swapping
- Andere reden van het besturingssysteem
- Verzoek van een interactieve gebruiker
- Timing
- Verzoek van het ouderproces

Beschrijvingen van processen

Beheersstructuren in het besturingssysteem

Om te beheren moet het besturingssysteem informatie bezitten over de huidige status van elk proces en elke bron. Hiervoor maakt en onderhoudt het tabellen met informatie over elke entiteit die het beheert.
Geheugentabellen bevatten volgende informatie:
- De toewijzing van hoofdgeheugen aan processen
- De toewijzing van secundair geheugen aan processen
- Eventuele beschermingsattributen van segmenten voor toegang tot gedeelde geheugengebieden
- Alle informatie die nodig is voor het beheren van het virtuele geheugen

I/O-tabellen worden gebruikt voor het beheren van I/O-apparaten en kanalen. Een apparaat kan op een bepaald moment beschikbaar of toegewezen zijn. Wordt er een I/O-bewerking uitgevoerd, dan moet het besturingssysteem kennis hebben van de status van de bewerking. Ook de locatie in het hoofdgeheugen die als bron of bestemming van de I/O-opdracht wordt gebruikt.
Bestandstabellen verschaffen informatie over het bestaan van bestanden, hun locatie, hun huidige status en andere attributen. Een groot deel hiervan wordt bijgehouden door het bestandsbeheersysteem.
Procestabellen worden bijgehouden om processen te beheren. Het bevat de locatie van het proces en de volgende attributen:
- Programma
- Gegevens
- Stack

Dit geheel noemen we het procesbeeld (process image).

Procesbesturingsblok

Procesidentificatie

Identificatiecodes
- Identificatiecode van proces
- Identificatiecode van het ouderproces

- Identificatiecode van gebruiker

Processortoestandsinformatie

Registers die zichtbaar zijn voor de gebruiker
Stuur en statusregisters
- Programmateller
- Conditiecodes
- Statusinformatie

Stackwijzers

Procesbesturingsinformatie

Scheduling en toestandsinformatie
- Procestoestand
- Prioriteit
- Scheduling gerelateerde informatie
- Gebeurtenis

Gegevensstructuur
Communicatie tussen processen
Procesprivileges
Geheugenbeheer
Eigendom en gebruik van bronnen

Procesbesturing

Creëren van processen

1. Het besturingssysteem wijst een unieke procesidentificatie toe aan het nieuwe proces
2. Besturingssysteem wijst ruimte toe aan het proces
3. Het procesbesturingsblok moet worden geïnitialiseerd
4. De juiste koppelingen moeten worden ingesteld
5. Soms moeten andere gegevensstructuren worden gemaakt of uitgebreid

Wisselen van processen

Wanneer wisselen?
- Klokinterrupt: proces heeft maximale tijd overschreden
- I/O-interrupt: I/O-actie is opgetreden
- Geheugenfout: verwijzing naar woord in virtuele geheugen dat zich niet in het hoofdgeheugen bevindt.
- Val: Een fout of uitzonderingsconditie wordt gegenereerd. Kan proces naar Einde-status brengen
- Supervisoraanroep: bv het openen van een bestand

Wisselen van status

- Opslaan van context van processor
- Aanpassen van procesbesturingsblok van het actieve proces
- Procesbesturingsblok verplaatsen naar juiste wachtrij (Gereed, geblokkeerd, ..)
- Ander proces selecteren
- Procesbesturingsblok van geselecteerd proces aanpassen
- Bijwerken van gegevensstructuren voor het geheugenbeheer
- Context van dit proces terugbrengen naar oorspronkelijke staat

Uitvoering van het besturingssysteem

Procesloze kernel: uitvoeren van kernel buiten alle processen. Het besturingssysteem wordt uitgevoerd als een afzonderlijke entiteit die werkt in een geprivilegieerde modus.
Uitvoering binnen gebruikersprocessen: uitvoeren in de context van een gebruikersproces. Er is een aparte stack voor beheren van aanroepen ren returns terwijl het zich in kernelmodus bevindt.

Op processen gebaseerd besturingssysteem

Het besturingssysteem is een verzameling van systeemprocessen. Nuttig in een omgeving met meerdere processors of computers.

Hoofdstuk 12
Bestandsbeheer

Bestandsbeheersystemen bestaan uit systeemhulpprogramma’s die worden uitgevoerd in kernelmodus.

Overzicht

Bestanden en bestandssystemen

Drie eigenschappen van bestanden:

- Langdurig bestaand
- Deelbaar tussen processen
- Structuur

Volgende functies zijn beschikbaar:

- Maken
- Wissen
- Openen
- Sluiten
- Lezen
- Schrijven

Vier begrippen:

• Veld

- Basiselement van gegevens
- Bevat één waarde
- Onderscheidt zich door een lengte en een gegevenstype

• Record

- Collectie van gerelateerde velden
- Wordt behandeld als een eenheid

• Bestand

- Een verzameling vergelijkbare records
- Één entiteit
- Bestandsnaam
- Toegangscontrole mogelijk

• Database

- Verzameling gerelateerde gegevens
- Expliciete relaties aanwezig tussen gegevenselementen

Standaardbewerkingen:

- Retrieve_All
- Retrieve_One
- Retrieve_Next
- Retrieve_Previous
- Insert_One
- Delete_One
- Update_One
- Retrieve_Few

Systemen voor bestandsbeheer

Manier waarop gebruiker of toepassing een bestand mag benaderen. Programmeur moet geen speciale software ontwikkelen voor elke toepassing.
Volgende doelstellingen:
- Voorzien in de behoeften en vereisten voor het gegevensbeheer van de gebruiker
- Zoveel mogelijk garanderen dat de gegevens in bestanden geldig zijn
- Prestaties optimaliseren
- Voorzien in I/O-ondersteuning voor diverse soorten opslagapparaten
- Minimaliseren of elimineren van de mogelijkheid dat gegevens verloren gaan of vernietigd worden.
- Voorzien in een gestandaardiseerde verzameling interfaceroutines voor I/O
- Ondersteuning van I/O bieden aan meerdere gebruikers

Minimale verzameling van eisen:

- Elke gebruiker moet bestanden kunnen creëren, verwijden en wijzigen
- Elke gebruik moet gecontroleerde toegang kunnen krijgen tot bestanden van andere gebruikers
- Elke gebruiker moet kunnen bepalen welke soorten toegang zijn toegestaan voor zijn bestanden
- Elke gebruiker moet de structuur van zijn bestanden kunnen aanpassen tot een vorm die aansluit op het op te lossen probleem
- Elke gebruiker moet gegevens kunnen verplaatsen tussen bestanden
- Elke gebruiker moet een reservekopie van zijn bestanden kunnen maken en de bestanden in geval van schade kunnen herstellen
- Elke gebruiker moet toegang kunne krijgen tot zijn bestanden via een symbolische naam

Architectuur van bestandssystemen

Apparaatstuurprogramma’s

- Communiceert op laagste niveau rechtstreeks met randapparaten
- Verantwoordelijk voor starten van I/O-bewerkingen en verwerken van voltooiing

Basisbestandsysteem is het volgende niveau

- Fysieke I/O
- Behandelt de gegevensblokken die worden uitgewisseld
- Verzorgt plaatsing van blokken en bufferen van die blokken

Supervisor voor basis-I/O

- Verantwoordelijk voor het starten en beëindigen van alle bestand-I/O
- Houdt besturingsstructuren bij
- Selecteert apparaat waarop I/O wordt uitgevoerd
- Verzorgt scheduling van toegangen om prestaties te optimaliseren
- Onderdeel van besturingssysteem

Logische I/O

- Verschaft gebruikers toegang to records
- Algemene voorziening voor record-I/O
- Houdt basisgegevens over bestanden bij

Toegangsmethode

- Verschillende toegangsmogelijkheden voor verschillende structuren en verschillende methoden om gegevens te verwerken en lezen

Functies van bestandsbeheer

- Geselecteerd bestand identificeren en vinden
- Gebruiken van directory die de locatie van alle bestanden beschrijft plus bijbehorende attributen
- Op een gedeeld systeem zorgen voor toegangscontrole van gebruikers
- Gebruik maken van blokken voor toegang tot bestanden
- Bestanden toewijzen aan vrije blokken
- Beheren van vrije opslagruimte zodat bekend is welke blokken beschikbaar zijn

Bestandsorganisatie en –toegang

Bij het kiezen van een bestandsorganisatie zijn diverse criteria belangrijk:

- Korte toegangstijd
- Gemak van bijwerken
- Efficiënt gebruik van opslagruimte
- Onderhoudsgemak
- betrouwbaarheid
- Korte toegangstijd

Stapelbestand

- Data wordt opgeslagen in volgorde waarin het binnenkomt
- Functie is slechts gegevensmassa te vergaren en op te slaan
- Records kunne verschillende velden hebben
- Geen structuur
- Om een record te zoeken moet het heel bestand doorzocht worden

Sequentieel bestand

- Vast formaat voor records
- Alle records hebben dezelfde lengte
- Alle velden zijn dezelfde(lengte en positie)
- Veldnaam en lengte zijn attributen
- Een bepaald veld wordt sleutelveld genoemd
- Unieke identificatie
- Records opgeslagen in sleutelvolgorde
- Nieuwe records worden in een logbestand geplaatst
- Batchverwerking wordt uitgevoerd om beide bestanden samen te voegen

Index-sequentieel bestand

- Index voorziet in een opzoekmogelijkheid waarmee snel de nabijheid van de gewenste record kan worden bereikt
- Bevat sleutelveld en verwijzing naar hoofdbestand
- Index wordt doorzocht naar hoogste sleutelwaarde die gelijk of kleiner is dan de gewenste sleutelwaarde
- Zoektocht gaat verder in hoofdbestand op positie aangeduid door verwijzing
- Vergelijking tussen sequentieel en index-sequentieel
- Bv een sequentieel bestand van één miljoen records. Het zoeken naar een bepaalde sleutelwaarde vereist gemiddeld een half miljoen recordtoegangen. Als er nu echter een index met duizen ingangen wordt gemaakt. Dan kost het zoeken van de records gemiddeld 500 toegangen tot het indexbestand, en dan nog eens 500 tot het hoofdbestand. De gemiddelde zoeklengte wordt dus verminderd van een half miljoen tot duizend.
- Nieuw record wordt toegevoegd aan een overloopbestand
- Record in hoofdbestand dat dit voorafgaat wordt aangepast met een verwijzing naar overloopbestand
- Bestanden worden samengevoegd bij een batchverwerking
- Meerdere indexen voor het zelfde sleutelveld is mogelijk

Geïndexeerd bestand

- Bevat meerdere indexen voor verschillende sleutelvelden
- Kan een index bevatten met een verwijzing naar elke record
- Kan een gedeeltelijke index bevatten

Direct of hashed bestand

- Directe toegang tot een blok op een gekend adres
- Sleutelveld nodig voor elke record

Bestandsdirectory’s

Bevat informatie over bestanden:

- Attributen
- Locatie
- Eigenaarschap

Directory is een bestand beheerd door het besturingssysteem.
Verzorgt vertaalslag tussen bestandsnamen en bestanden zelf
Informatie-elementen van een directory:

Basisinformatie

- Bestandsnaam

- Bestandstype
- Bestandsorganisatie

Adresinformatie

- Volume
- Beginadres
- Gebruikte grootte
- Toegewezen grootte
- Eigenaar
- Toegangsinformatie
- Toelaatbare acties

Gebruiksinformatie

- Datum gecreëerd
- Identiteit van maker
- Datum laatste leestoegang
- Identiteit van laatste lezer
- Datum laatste wijziging
- Identiteit voor laatste wijziging
- Datum laatste reservekopie
- Huidig gebruik

Simpele structuur

- Lijst van ingangen, één ingang voor elk bestand
- Sequentieel bestand met de naam van elk bestand dat dienst doet als sleutel
- Geeft geen hulp bij het organiseren van bestanden
- Verplicht gebruiker voorzichtig te zijn om niet twee keer dezelfde naam te gebruiken

Twee-niveau schema

- Een directory voor elke gebruiker en een hoofddirectory
- Hoofddirectory bevat ingang voor elke gebruiker
- Bevat adres en toegangscontrole-informatie
- Elke gebruikersdirectory is een eenvoudige lijst met bestanden van die gebruiker
- Voorziet geen hulp bij structureren

Hiërarchische of boomstructuur

- Hoofddirectory met gebruikerdirectories eronder
- Elke gebruikersdirectory kan meerdere subdirectories en bestanden bevatten
- Bestanden kunnen gevonden door een pad te volgen van de hoofddirectory langs subdirectories. Dit is de padnaam

- Meerdere bestanden kunnen dezelfde naam hebben zolang ze een unieke padnaam hebben
- De huidige directory is de werkdirectory
- Er wordt relatief ten opzichte van de werkdirectory verwezen naar bestanden

Gemeenschappelijk gebruik van bestanden

Toegangsrechten

- Geen
- Kennis
- Uitvoeren
- Lezen
- Toevoegen
- Bijwerken
- Beveiliging wijzigen
- Verwijderen

Eigenaar:

- Heeft alle voorgaande rechten
- Kan rechten uitdelen aan anderen volgens de volgende klassen van gebruikers:
- Specifieke gebruiker
- Gebruikersgroepen
- Alle

Gelijktijdige toegang

- Gebruiker kan een heel bestand vergrendelen wanneer het bijgewerkt wordt
- Individuele records vergrendelen tijdens bijwerken
- Wederzijdse uitsluiting en deadlocks zijn problemen voor gedeelde toegang

Blokvorming van records
Records = logische toegang
I/O: blokken op schijf
Grote blokken versus Kleine Blokken

Drie methoden:

- Blokvorming met vaste lengte: Hierbij worden records met vaste lengte gebruikt en wordt een geheel aantal records opgeslagen in een blok. Er kan ongebruikte ruimte zijn aan het einde van elk blok; dit wordt aangeduid als interne fragmentatie
- Gekoppelde blokvorming met variabele lengte: hierbij worden records met variabele lengte gebruikt en deze worden samengepakt in blokken, alle ruimte wordt gebruikt. Sommige records kunnen daarbij twee blokken omspannen, waarbij het vervolg wordt aangegeven door een verwijzing naar het volgende blok

- Niet-gekoppelde blokvorming met variabele lengte: hierbij worden records met variabele lengte gebruikt, maar deze worden niet gekoppeld opgeslagen. De meeste blokken bevatten loze ruimte, omdat het onmogelijk is de rest van een blok te gebruiken als de volgende record groter is dan de resterende ruimte

Beheer van secundaire opslag

- Ruimte moet toegewezen worden aan ebstanden
- Moet bijhouden welke ruimte vrij is voor toewijzing

Bestandstoewijzing

Maximumgrootte van een bestand bij de creatie?

Moeilijk om de maximale grootte te voorspellen
Vaak de neiging om de bestandsgrootte te overschatten

Methoden van bestandstoewijzing

- Aaneengesloten toewijzing (=Bij aaneengesloten toewijzing wordt één aaneengesloten verzameling blokken toegewezen aan een bestand op het moment dat het bestand wordt gecreëerd.
- Tabel voor bestandstoewijzing bevat maar één ingang voor elke bestand te bevatte, die het beginblok en de lengte aangeeft.
- Externe fragmentatie zal voorkomen
- □ Samenpakken is nodig
- Kettingtoewijzing; is het tegenovergestelde van aaneengesloten toewijzing. De toewijzing vindt meestal plaats op basis van afzonderlijke blokken. Elk blok bevat een wijzer naar het volgende blok in de ketting.
- Tabel voor bestandstoewijzing bevat maar één ingang voor elke bestand, die het beginblok en de lengte aangeeft.
- Geen externe fragmentatie
- Beste voor sequentiële bestanden
- Niet goed voor lokaliteit (cache)
- Indextoewijzing; Hierbij bevat de tabel voor bestandstoewijzing een afzonderlijke index van één niveau voor elk bestand. De index heeft één ingang voor elke portie die aan het bestand is toegewezen.
- Tabel voor bestandstoewijzing bevat bloknummer van index

Inodes

- Informatieknooppunt: (information node)
- Controlestructuur die sleutelinformatie bevat voor een bepaald bestand

Windows file system

Kernzaken van NTFS

- Herstelbaarheid
- Beveiliging
- Grote schijven en grote bestanden
- Meervoudige gegevensstromen
- Algemene indexvoorzieningen

NTFS volume en bestandsstructuur

- Sector: kleinste fysieke opslageenheid op de schijf
- Cluster: een of meer aaneengesloten sectoren
- Volume: een logische partitie op schijf

Hoofdstuk 6: Bootproces

Het besturingssysteem is een programma dat op extern geheugen staat en moet gekopieerd worden naar het werkgeheugen (= inladen). Het besturingssysteem inladen is een speciale procedure, want om iets in te laden moet er al programmatuur zijn.

Partities

Deel van de harde schijf: opeenvolgende cilinders.
- Maximum 4 primaire partities.
- In partities indelen gebeurt bij het begin van de installatie van het besturingssysteem.
- Eventueel kan men achteraf partities wijzigen, mits speciale tools.
In een partitie kan je een besturingssysteem installeren.
Er kan maximum 1 partitie actief zijn.
Bij opstarten zal het besturingssysteem starten in de actieve partitie.

MBR

= Master Boot Record (Record = sector).
MBR bevat:

- MBR-startprogramma (Initial Program Loader, IPL)
- Partitietabel
- Magisch getal

Bij het booten start het besturingssysteem in de actieve partitie

Partitietabel

De partitietabel kan 4 partitie-records bevatten die ieder een primaire partitie beschrijven. De partitietabel bevat per partitie:
- Bootable flag : actief / niet actief
- Cylinder/Head/Sector van 1e sector
- Partitie-type : geeft aan welk bestandssysteem op de partitie staat
- Cylinder/Head/Sector van laatste sector
- LBA-nummer van 1e sector
- LBA-nummer van laatste sector

Logische partities

4 primaire partities is soms te weinig. Een primaire partitie kan onderverdeeld worden in logische partities (= extended)

EMBR

= Extended Master Boot Record
EMBR voor de extended partities
EMBR voor elke logische partitie

Elke EMBR heeft 2 items:
- Info over de partitie zelf
- Info over de rest

Welke info?
- Type
- Chs 1e sector
- Chs laatste sector
- Grootte

PBR

Partitie wordt geformatteerd (bestandssysteem opgeven)
1e sector van de partitie is de Partition Boot Sector (sector = record; PBR = PBS)

PBR bevat:
- Sprongbevel (spring x aantal bytes verder, naar PBR-startprogramma)
- Info over de partitie
- PBR-startprogramma

Wat gebeurt als je computer start?

1. Interne voeding wordt geïnitialiseerd
- Testen van de geleverde spanningen
- Reset-signaal naar CPU sturen

2. Wanneer spanning O.K.: “Power Good” signaal naar moederbord (= stoppen met reset)
3. Processor begint bevelen uit te voeren
- Welke mode?
• Reële mode (Intel)
- Welk bevel?
• Op adres FFFF:0000
- Wat betekent dit?
FFFF0 segmentadres
+0000 verplaatsing
FFFF0 (in register CS staat FFFF)
FFFF0 is een adres in het werkgeheugen, toegewezen aan het BIOS (vroeger ROM, nu EEPROM).

BIOS EEPROM:
- Soms nog “CMOS” genoemd (Complementary Metal-Oxide Semiconductor)
- Eigenlijk is “de CMOS” 1 van de weinige chips in een computer die geen CMOS-technologie gebruikt
- Vroeger CMOS-geheugen dat door batterijen onder spanning werd gehouden
- Bevat:
• Opstartvolgorde
• Informatie over randapparaten
• BIOS-paswoord
• …

Processor begint met BIOS-startprogramma (op adres FFFF0)
• Wat doet dit startprogramma?

4. Power on self test uitvoeren, POST controleert vitale systeemonderdelen :
- CPU
- Toetsenbord (verbonden, controller O.K.?)
- RAM (lezen, schrijven)
- Video,…

5. Bios-startprogramma gaat na wat het boot-apparaat is
6. MBR kopiëren naar werkgeheugen (bevat IPL)
7. Sprong naar 1e bevel in IPL
8. IPL :
- Leest partitietabel (gaat na welke de actieve partitie is)
- Kopieert PBR (met PBR-startprogramma) van deze partitie naar werkgeheugen
- Sprong naar bevel vanaf 1e byte van PBR

9. 1e bevel in PBR : sprong naar PBR-startprogramma
10. PBR-startprogramma : ntldr inladen
- Indien niet gevonden : “ntldr missing” op het scherm zetten

11. Ntldr : processor naar protected mode en Windows starten

2011 juni examen

Geens

- Verklaar volgende begrippen en geef onderliggend verband
  - MBR
  - Partitietabel
  - EMBR
  - PBR
- Vraag 2: Interrupts
  - Geef de instructiecyclus met interrupt
  - Geef de stappen van een interruptverwerking (9 stappen)
  - Hoe werkt multiprogrammering met interrupts?
- Vraag 3: Geheugen
  - Dilemma van geheugen snelheid/capaciteit/kosten
  - Principe van lokaliteit
  - Leg uit: LRU
- Vraag 4: Geef het procestoestandschema van 2 opschortingen
- Vraag 5: bestandbeheer
  - 3 zinnen, zeggen of ze waar of fout waren en waarom
  - De 3 soorten geheugenblokken en hun voor en nadelen
- Vraag 6: FAT12
  - Je krijgt een descriptortabel en een klein stukje van de FAT tabel.
  - Geef de filesize
  - Geef de eerste cluster
  - Geef de nummers van de clusters van de dataclusters
  - Met 512 bytes clusters, hoeveel overschot heb je in de datatabel
  - Geef de Engelse term voor fragmentatie.

Strypsteen

- Idem examen Geens voor vraag 1, 2, 4 en 6
- leg uit:
  - Failed Mutual Exclusion
  - Deadlock
  - Hoe het splitsen van een proces in drie onderdelen het management van processen vereenvoudigd heeft.
- inode met 13 blokken, adres grootte van 4b, block grootte van 400b:
  - Hoe groot mag een bestand zijn als het geen indirecte toegang mag gebruiken (13 * 400b = 5,2 kb)

- Hoe groot mag een bestand zijn als het één niveau van indirecte toegang mag gebruiken (400b/4b) = 100 + 12 blocks, 112 blocks * 400b = 44.8kb
- Hoe groot mag een bestand zijn als het drie niveaus van indirecte toegang mag gebruiken (12 * 100 * 100 * 100 * 400b)= 480mb

edit: dees is fout volgens mij ze... dit is wat het zou moeten zijn:

- Hoe groot mag een bestand zijn als het geen indirecte toegang mag gebruiken: 10*400b = 4kb

(Moet het niet 12*400 zijn? Er zijn 12 blokken voor directe toegang.)

- Hoe groot mag een bestand zijn als het één niveau van indirecte toegang mag gebruiken: 4kb + 256*400b = 106,4kb

(Waar komt 256 vandaan? Uitleg had mooi geweest --> bekijk pagina 627 van uw boek. Het grijze kadertje)

- Hoe groot mag een bestand zijn als het drie niveaus van indirecte toegang mag gebruiken: 4kb + 106,4kb + 256*256*400b + 256*256*256*400b = 6,7 GB

2012 augustus examen

Strypsteen

- 1) Leg het bootproces van een computer uit met als actieve partitie Windows XP. (11 stappen)
- 2) Teken het procestoestandschema met 2 opschortingen, en benoem alle pijlen.
- 3) Leg uit:
  - Deadlock
  - Failed Mutual Exclusion
  - Symmetrische multiprocessing
  - Resident monitor
- 4) Welke 2 manieren zijn er om interrupts te behandelen? + Schets. En geef voor-en nadelen.
- 5) UNIX-inode met 13 blokken: 10 indirecte. Adres grootte van 1024 bytes (1kb), block grootte van 4b.
  - Hoe groot mag een bestand zijn als het alleen de niveaus van indirecte toegang mag gebruiken?
  - Hoe groot mag een bestand zijn als men alle niveaus mag gebruiken?

Toon berekening.

- 6) Vraag over LINUX.
- 7) Vraag over FAT 12.

2012 juni examen

Strypsteen

- 1) Geef de verschillende klassen (4) interrupts. Teken de instructiecyclus met interrupts.
- 2) Verklaar volgende begrippen + Schets
  - Interne fragmentatie
  - Externe fragmentatie
  - Aaneengesloten bestandstoewijzing
  - Kettingtoewijzing
  - Indextoewijzing
- 3) Batchmultiprogramming versus Timesharing: Wat is hun hoofddoel?
- 4) Leg het bootprocess van een PC uit, van POST tot en met het PBR-startcommando.

Leg de afkortingen die je gebruikt ook uit

- 5)a) Teken het procestoestandschema met 2 opschortingen.

b) Waarom is toestand opgeschort toegevoegd?

c) Geef onderlinge verbanden tussen de toestanden met opschortingen.

2013 augustus examen

Strypsteen

deel 1: opstarten

1) geef de 11 stappen bij het opstarten van een windows xp partitie.

deel 2: batchsystemen

2) wat zijn de 4 hardware features die essentieel zijn voor bij een monitor?

3) leg uit aan de hand van vorige vraag de kernel en user mode.

4) wat zijn de voordelen en nadelen van vaste grootte en variabele grootte geheugen?

deel 3: processen

5) een proces wordt preëmptief behandeld. wat is het en wanneer gebeurt het?
6) wat gebeurt er als een proces van toestand geblokkeerd opgeschort naar geblokkeerd word verplaatst?
7) wat gebeurt er als een proces wordt verplaatst van gereed naar gereed opgeschort?

deel 4: Linux

deel 5 : fat

8) Hoe groot is de file?
9) Wat zijn de fat-entry nummers?
10) Wat zijn de dataclusters die erbij horen?

deel 6: Active Directory

11) Er zijn verschillende group policies van GP01 - GP05. Wat als de "disable command prompt" in de group policy?

OU	GP	status
gt	GP01	disabled
	GP02	not configured
TI	GP03	enabled
	GP04	disabled
KHL	GP05	not configured

wat is de status van:

- gt: enabled / disabled
- TI: enabled / disabled
- KHL: enabled / disabled

12) Leg uit: permissies zijn cumulatief. Hoe werkt dit met 'Deny'?

2013 Juni examen

Voorbeeldexamenvragen

Geef de diensten die het besturingssysteem aanbiedt :

- Programma-ontwikkeling
- Programma-uitvoering
- Toegang tot I/O-apparaten
- Beheerde toegang tot bestanden
- Toegang tot het systeem
- Opsporen en afhandelen van errors
- Administratie

Welke zijn de oorzaken tot ontwikkeling van een besturingssysteem:

- Hardwarevernieuwing
- Nieuwe diensten
- Verbeteringen

Geef de hoofdtaken van het beheer van opslag :

- Procesisolatie
- Automatische toewijzing en beheer
- Ondersteuning modulair programmeren
- Bescherming en toegangsbeheer
- Langetermijnopslag

Geef de 4 categorieën van beveiliging die door het besturingssysteem gebruikt worden :

- Beschikbaarheid
- Gegevensintegriteit
- Authenticiteit
- Vertrouwelijkheid

Welke zijn de 3 doelen m.b.t. scheduling van processen:

- Rechtvaardigheid
- Efficiëntie
- Gedifferentieerd reactievermogen

Complexe besturingssystemen hebben standaard altijd meer problemen als simpele, geef deze problemen bij complexe besturingssystemen:

- Altijd te laat uitgebracht
- Altijd gevoelig voor virussen etc.
- Altijd geplaagd door bugs
- Prestaties vaak niet zoals verwacht

Maak de lijst van niveaus binnen een besturingssysteem:

1. Elektronische schakelingen
2. Instructieset
3. Procedures
4. Interrupts
5. Primitieve processen
6. Lokale secundaire opslag
7. Virtueel geheugen
8. Communicatie
9. Bestandsysteem
10. Apparaten
11. Directory’s
12. Gebruikersprocessen
13. Shell

Geef de elementen van een PCB (procesbesturingsblok):

- Identificatienummer
- Toestand
- Prioriteit
- Programmateller
- Geheugenwijzers
- Contextgegevens
- I/O-toestandsinformatie
- Beheersinformatie

Geef de 4 redenen voor het starten van een proces:

- Nieuwe batchtaak
- Interactieve aanmelding (logon)
- Gecreëerd door besturingssysteem omwille van een dienst
- Verwekt door bestaand proces

Geef alle redenen voor het beëindigen van een proces:

- Normale voltooiing
- Tijdslimiet overschreden

- Onvoldoende beschikbaar geheugen
- Overtreding geheugengrens
- Beschermingsfout
- Rekenkundige fout
- Tijd verstreken
- I/O-fout
- Ongeldige instructie
- Geprivilegieerde instructie
- Onjuist gebruik van gegevens
- Ingreep van gebruiker/besturingssysteem
- Beëindiging ouderproces
- Verzoek van ouderproces

Wat zijn de kenmerken van een opgeschort proces:

- Niet onmiddellijk beschikbaar voor uitvoering
- Kan wachten op gebeurtenis
- Werd in opgeschorte toestand geplaatst om uitvoering te verhinderen
- Kan opgeschorte toestand niet verlaten zonder expliciete opdracht door ander proces

Welke zijn de redenen voor het opschorten van een proces:

- Swapping
- Andere reden van besturingssysteem
- Verzoek interactieve gebruiker
- Timing
- Verzoek ouderproces

Geef de structuur van de creatie van een proces:

1. Procesidentificatie toewijzen aan nieuw proces
2. Ruimte toewijzen aan het proces
3. PCB initialiseren
4. Juiste koppelingen instellen
5. Eventueel andere gegevensstructuren maken / uitbreiden

Geef de volledige tabel van elementen van de standaard bestandsdirectory:

- Basisinformatie

- Bestandsnaam
- Bestandstype
- Bestandsorganisatie

- Adresinformatie

- Volume
- Beginadres
- Gebruikte grootte
- Toegewezen grootte

  - Toegangscontrole-informatie

- Eigenaar
- Toegangs-informatie
- Toelaatbare acties

  - Gebruiksinformatie

- Datum gecreëerd
- Identiteit maker
- Datum laatste leestoegang
- Identiteit laatste lezer
- Datum laatste wijziging
- Identiteit laatste wijziging
- Datum laatste reservekopie
- Huidig gebruik

Geef de structuur van het wisselen van processen:

1. Opslaan context huidig proces
2. Bijwerken PCB (toestand veranderen
3. PCB naar juiste wachtrij verplaatsen
4. Selecteren van nieuw proces
5. Bijwerken PCB geselecteerde proces (toestand op actief zetten)
6. Bijwerken gegevensstructuren voor geheugenbeheer
7. Terugbrengen context van dit proces dat voorheen was opgeslagen

Geef de kenmerken van NTFS:

- Herstelbaarheid
- Beveiliging
- Grote schijven en bestanden
- Meervoudige gegevensstromen
- Algemene indexeringsvoorziening

Piphi

- 1) (Opstarten):
  - FAT - 16 C// en D// schijf gegeven.
  - Zeggen wat wat is.

    - 1ste 2 bytes van iets kunnen geven.
- 2) (Leg uit):
    - ◦ Multiprogramming, time-sharing, real-time transactie systeem
- 3):
    - Geef beheersstructuren proces:
    - Geef van iedere categorie een voorbeeld:
- 4) (Processen):
    - Schema tekenen met 2 opgeschorte toestanden
    - Leg de pijlen uit die met de opgeschorte toestanden te maken hebben
- 5) (Bestanden):
    - Geef structuur
    - Veld (field): ....
    - Record: ....
    - Bestand (file): ....
    - Database: ...

Strypsteen

1),2),3),4) Hetzelfde als PiPhi

Ook nog bij 2) : De problemen dat destijds ontstonden door deze ontwikkelingen. (Onjuiste syncronisatie, mutual exclusions, niet-vastomschreven programmawerking en deadlocks; uitleg geven moest niet)

- 5) (bestanden)
    - UNIX-inode met 15 blokken (12 indirect)
    - Adres grootte van 8B block grootte van 8KB

> Antwoord (gemeenschappelijk deel): aantal adressen dat in een block op de schijf kunnen: block grootte / adres grootte = 8KB / 8B = 1024

- ◦ Hoe groot mag een bestand zijn als het alleen de niveaus van indirecte toegang mag gebruiken?

> 8KB / block * (12 * 1024 blokken) = 98304 KB = 96 MB

- ◦ Hoe groot mag een bestand zijn als men alle niveaus mag gebruiken?

> 8KB / block * ((3 + 12 * 1024) blokken) = 98328 KB = 96.02 MB
>
> Dit hangt er wel van af wat ze precies bedoelen, als "alle niveaus" direct + enkelvoudig indirect is, is het antwoord het bovenstaande... vraag wat er wordt bedoeld aan de lector

bron waarop ik me baseer om deze berekeningen te doen :

http://stackoverflow.com/questions/4383493/what-is-the-maximum-file-size-in-each-of-these-cases

6) Vraag in verband met de verschillende lagen

2014 juni examen

Strypsteen

1. Tekening van de harde schijf, er staan overal nummertjes en jij moet dan zeggen wat dit is of hoe groot iets is.

2. Geef de verschillende soorten interrupts

- Programma
- Timer
- I/o
- Hardwarefout

3. Geef de hardware features van een resident monitor

- geheugenbeveiliging
- timer
- geprivilegieerde instructies
- interrupts

4. Wat is deadlock + geef een vb

> 2 processen hebben dezelfde 2 bronnen nodig hebben om verder uitgevoerd te kunnen worden.
> Ze hebben elk al 1 bron maar wachten op elkaar tot als het ene proces de bron vrijgeefd.

5. Er is een foto gegeven van hoe er blokvorming ontstaat.

- Duid aan waar de blokvorming optreedt
- Benoem de delen die hier mee in contact komen

6. Waarom gaat een procestoestand van geblokkeerd/suspend => geblokkeerd

- Dit is zeldzaam maar gebeurt meestal wanneer een hoge-prioriteit proces op iets wacht dat bijna klaar is..

  > Dan kan BS deze al naar RAM brengen zodat die sneller uitgevoerd zal kunnen worden nadat event waarvoor die wacht plaats vindt.

- Wat zorgt ervoor dat een proces van geblokkeerd => gereed gaat

Als de gebeurtenis optreedt waar het proces op wacht

7. Invul oefening over bitmap, waar ge 0 (vrije plek) en 1 (plek is bezet) moet invullen

8. Bestandstoewjzing oefening:

Er zijn 100 blokken, hoeveel lees en schrijf bewerkingen worden er uitgevoerd voor de volgende methodes:

- Aaneengesloten toewijzing
- Kettingtoewijzing
- Indextoewijzing

9. Geef de drie soorten uitvoeringen van het besturingssyteem en teken het(+ wat uitleg)

- Procesloze kernel
- Uitvoering binnen gebruikersprocessen
- Op processen gebaseerd besturingssysteem

---

duur 90 min

Opdracht linux: Opdracht 1: In de directory /home/ldapusers staan home directories voor alle gebruikers. Toon al deze directories.(enkel de namen) Opdracht 2: Laat deze lijst gebruikers weggeschreven worden naar een bestand users.txt in je eigen home directory. Opdracht 3: Gebruik Gzip om deze lijst te comprimeren naar users.gz Opdracht 4: Vraag de bestandsgroottes op van user.txt en users.gz

Active directory: 1 maak nieuwe gebruikersgroep aan met naam BS1 2 Voer er stud1 en lect 1 toe als leden 3 maak nieuwe subdirectory in bestand op c schijf 4 verwijder de leesrechten (users) van dat bestand 5 zorg ervoor dat leden uit groep BS1 leesrechten krijgen 6 Zorg dat ook alle lectoren volledige controle krijgen over bs1

2015 Juni examen

Strypsteen

1. Overzicht van 2 partities (1 primaire en 1 extended). Benoem/beschrijf een aantal onderdelen (o.a. PBR, boot sector, EMBR) en geef de grootte van het lege deel na de MBR.

2. Teken een schema van een interrupt. Waarin verschilt een interrupt van een gewone subroutine?

3. Benoem alle onderdelen van fig. 2.11 in het handboek.

4. Geef 4 mogelijke voordelen van Symmetrical Multi Processing.

5. Hoeveel lees- en schrijfbewerkingen zijn er nodig om een blok in het midden toe te voegen bij 100 aaneengesloten, gelinkte en indexblokken + verantwoord.

6. Inodes van 12 blokken, een single, een double en een triple indirect pointer. Blokken van 8kB en adressen van 8B. Geef de grootte bij geen, 1 en alle indirecte pointers.

- Geen: 12 directe verwijzingen naar elk 8kB geeft 96kB.
- single indirect + direct: 8kB/8B adressen/blok = 1000 adressen/blok => 1000*8kB (S indirect) + 96 kB (direct) = 8.096.000B = 8MB
- S,D,T indirect en direct: [1000^3 (T indirect) + 1000^2 (D indirect) + 1000 (S indirect) + 12 (direct)] * 8kB = 1.001.001.012 * 8000B = 8.008.008.096.000B = 8TB

7. Van welke status naar welke status gaat een proces als het preëmptief behandeld wordt en waarom?

8. Uit welke 3 onderdelen bestaat een procesbesturingsblok + geef van elk onderdeel een voorbeeld.

2016 juni examen

Strypsteen

1. Wat is een trap?

2. Welke zijn de 3 minimum services van een microkernel?

3. Wat is het grootste nadeel van een microkernel?

4. Geef alle kenmerken van een procescontrollerblok.

5. Wat is een interrupthandler?

6. Wat is het verschil tussen acilic tree en tree bij directorys?

2017 augustus examen

Strypsteen

2u tijd, volledig schriftelijk, niks meenemen.

1. Geef 11 stappen van bootproces (windows)

2. Leg uit soft (symbolic) en hard links. Leg uit: dangling pointers.

3. Geef limitaties BIOS.

4. Welke info op PCB? Welke processen in ready queue? Hoe wordt er naar verwezen?

5. Hoe gebeurt hardware interrupt? Geef alle stappen in schema. Leg uit: trap.

6. Waarom gaat een procestoestand van geblokkeerd/suspend => geblokkeerd? Wat zorgt ervoor dat een proces van geblokkeerd => gereed gaat? Waarom toestand van running naar ready?

7. Inodes van 10 blokken, een single, een double en een triple indirect pointer. Blokken van 8kB en adressen van 4B. Geef de grootte bij geen, 1 en alle indirecte pointers.

8. 100 blokken bestandssysteem, hoeveel lees/schrijf I/O operaties voor directe, gelinkte en geïndexeerde allocatie.

9. AD vraag

10. FAT vraag

2019 FAT 16 oplossingen

GPT

find the disk GUID:

F2 5A 52 C8 C3 C6 A4 94 46 0D 3C E6 87 7D 61 D7

calculate the maximum partition size (use the first and last usable LBA's):

first: 0022 (34)d

last: 00 4F DE (20446)d

max bytes: 10 450 944

single partition entry:

128 bytes

PARTITION ENTRY

The size: 18 398 blocks or 9 419 776 bytes

first block: block #2048

offset: 0x100000

PARTITION LAYOUT

How many copies of the FAT are there?

2

How large is one FAT?

72 sectors, 36 864 bytes

How many file/directory entries can be stored in the root directory?

512 items

How large is the root directory? (32 bytes/entry)

512 * 32 = 16 384 bytes, of 32 sectoren

Compute the offsets of the following sections

Boot Sector = 0x1000000

FAT 1 = 0x101000

FAT 2 = 0x10A000

Root dir = 0x113000

Data section = 0x117000

DIRECTORY ENTRY

INTERPRET

filename = PROGRAMS Attributes = Directory last write time = 3:25:44 last write date = 26/10/2005 first logical cluster = 0140

LIST (name, attributes, size and first cluster)

BS LABO - volume- 0000 (first cluster)
hello.txt - archived - 13 bytes - 0002h
docs - Directory - 0 bytes - 0003h
system - directory - 0 bytes - 0004h
building.jpg - none - 79829 bytes - 25h
nala.jpg - Read-only - 69508 bytes - C1
bill.jpg - archived - 13024 bytes - 15E

FAT how many clusters will the file bill.jpg occupy?
#clusters = 26 clusters
first physical cluster of bill.jpg?
logical = 15E => offset = (15E - 2)*200 + 0x117000 = 0x142800

RECONTSTRUCT POEM 1) 151 2) 01D6 3) 0195 4) 018a 5) 15d 6) 1ae 7) 023a 8) 1fe (9) FFFF)

poem: i feel like i haven't been read right
for 10 years of drops from heights
waiting for scandisk
to repair me
your os says let's go
but my boot sector says no
if you want to read me, there's a proper way
I'm a floppy in a disk drive, you gotta format me the right way

LONG FILENAMES 1) DOCS subdirectory = ox117000 + ((0003 - 2)*200) = 0x117200 2) short = QUESTI~1 3) long = "questions exam operating systems (2017)"

RECOVERING DELETED FILES SYSTEM subdirectory = 0x117000 + ((0004 - 2) * 200) = 0x117400
directx.log
first cluster = 0020 (offset data 0x11AC00)

2019 juni examen

Arne Walschap

Leg volgende begrippen grondig uit:

- trashing
- daisy chaining
- timesharing
- master file table
- PU-gebonden processen

dan open vragen:

waarom kan je met een RAM van 8GB een programma/bestand van 10GB downloaden (iets met virtual host)

5 eigenschappen EUFI

5 doelstellingen scheduling

paginatie tabel tekenen en uitleggen (werkgeheugen onderverdeeld in 4 frames en geheugengrootte is 16 bytes)

processor kan niet in de toekomst kijken, voor welk scheduling strategie is dit een probleem + wat is de oplossing

verschil tussen boomstructuur en acyclische structuur

welk interrupt request is gemaskeerd en welke kunnen niet gemaskeerd zijn (interne synchroon en externe asynchroon)

xxx Kirito-kun <3

2020 augustus examen

De opgave van het examen in augustus met dank aan ISW:

2020 handige scripts - ISW

Een paar handige scripts met dank aan ISW:

2020 januari examen

Tiebe Van Nieuwenhove

1) Waarom is een besturingssysteem dat interrupts gebruikt sneller. Verklaar en maak een tekening.

2) Concreet begint de processor instructies uit te voeren vanaf adres 0xFFFFFFF0. Wanneer de processor het geheugenadres 0xFFFFFFF0 opvraagt aan de geheugen controller, zal die echter geen bytes vanuit het werkgeheugen halen. Vanwaar haalt hij ze en wat is de naam die hier aan gegeven wordt?

3) Wat is een aaneengesloten bestand? Vergelijk dit met de niet-aaneengesloten bestanden.

4) Leg uit hoe het Best Fit en Worst Fit werken. Geef een voorbeeld waar we dezen zouden kunnen gebruiken.

5) Leg uit wat 2-level paging is en maak een tekening.

6) Maak een tijdlijn van met de volgorde waarin de verschillende processen uitgevoerd worden voor Round Robin & Shortest Remaining Time

7) Leg volgende begrippen uit:

- Trap
- Starvation
- Interne fragmentatie

2020 juni examen

(Corona) - R. Swennen & F. Vogels

Het examen was een open-internet examen met 2 delen en een maximale tijd van 2h36. Door corona was het theorie en praktijk examen samengevoegd tot 1 examen. Deel 1 was meer praktisch en voor deel 2 moest je op een .docx maken met gegeven vragen (en hun antwoorden).

Deel 1

Active Directory

- Welk van de volgende services heeft AD nodig? kies één uit: DHCP, WINS, **DNS**, LDAP, geen van bovenstaande
- Welke van de onderstaande uitspraken is waar? kies één uit: Een domain maakt altijd deel uit van een site, **een domain maakt altijd deel uit van een forest**, een forest maakt altijd deel uit van een tree, een tree maakt altijd deel uit van een domain
- Waarop kunt u de eigenschap Block Inheritance instellen? kies één uit: Policy, **GPO**, OU, geen van bovenstaande

FAT

- Adres van eerste FAT tabel
- Adres van root directory
- Adres data region
- Totale grootte van FAT16 volume in bytes
- ...
- Stel dat bij unix de i-node 13 blok-adressen bevat: 10 directe, 1 indirect, 1 dubbel-indirect en 1 drievoudig-indirect. Stel dat de blokgrootte 4 Kbytes is en dat de blok-adressen 64 bits zijn. Wat is de maximale bestandsgrootte (in X KBytes) als er geen gebruik gemaakt wordt van indirecte adressen? Antwoord in Kbytes

Er was een FAT16 image meegegeven hier uit moest je wat info uit halen en op xToledo invullen. (Dus niet het HTML bestand) Bestand met dank aan ISW:

Scheduling strategieën

- Geef start/omlooptijd van proces Round Robin (RR)/Shorest Remaining Time (SRT)

Deel 2

- Leg aan de hand van een concreet voorbeeld uit wat het verschil is tussen interne én externe fragmentatie
- Leg in detail uit of een besturingssysteem, multiprogrammatie moet ondersteunen om efficiënt gebruik te kunnen maken van interrupts
- Leg het onderling verband uit tussen volgende termen: kernel-toestand, gebruikerstoestand en system calls

2020 PE

Tiebe Van Nieuwenhove

Active directory

- Maak in de OU Employees de OU education aan. Maak in education 2 OU's die op hetzelfde niveau zitten: students & teachers.
- Maak in teachers een user Teacher 1 aan en in students Student 1 aan.
- Maak een groep Locker Access waar Student 1 in zit en een groep Coffee Room waar Teacher 1 in zit. Voeg de gebruikers van de groep Coffee Room toe aan de groep Locker Access (DIT MAG JE NIET HANDMATIG VOOR ELKE GEBRUIKER DOEN!)
- Theorie: Wat is het verschil tussen een Organized Unit en een Security Group?

2.

- Zorg ervooor dat de computers niet opnieuw automatisch opstarten wanneer er nog gebruikers ingelogd zijn na updates.
- Maak een shortcut op de desktop van alle gebruikers in het domein naar de website: jynupper.com
- Installeer op alle computers van de Teachers het programma emacs.
- Enforce these Password policies

    - Wachtwoorden moeten minimaal 8 charachters lang zijn, moeten elk jaar verandert worden en de laatste 5 wachtwoorden moeten onthoud worden.
    - Wachtwoorden van teachers moeten minimaal 10 tekens lang zijn en moeten elk half jaar verandert worden.
    - Studenten mogen hun wachtwoord maar om de 7 dagen veranderen

- Stel dat de GPO om aan de de Control panel zo geconfigureerd is:

GPO	status	Kunnen ze aan de terminal?
Domein	Disabled	ja / nee
Employees	Not Configured	ja / nee
Teachers	Enabled	ja / nee
Students	Not Configured	ja / nee

FAT

We kregen een .bin bestand zoals tijdens de oefenzitting en er was gegeven dat een sector 512 bytes lang is en er is 1 sector per cluster. De FAT paritie start vanaf 0x11000, de data partitie start vanaf 0x25000.

1. Geef het unieke identificatienummer van de partitie.

2. Geef het Volume Serial Number.

3. Geef de short name, long name, last write date, last write time, attributen, eerste cluster van het volgende bestand:

```
4150 0075 006E 0074 0042 000F 006D 5300
3100 2E00 7800 6C00 7300 0000 7800 0000
5055 4E54 4253 7E31 584C 5324 001E EC7C
8A2A 332B 0000 A845 2B2B 4400 89EA 0000
```

4. Geef de opeenvolgende clusternummers van het bestand dat begint bij de logische cluster 0x07b6.

5. Geef de eerste 10 bytes van het bestand van de vorige oefening.

FAT 16 berekenen

Voor FAT16 te berekenen kan je kijken op https://github.com/ISW-Leuven/fat16 met dank aan ISW

Hieronder staat de repo gekopiëerd:

FAT16

This repo contains a few things to help you find / calculate FAT16 properties, ...

Disclaimer

The info in this repo may contain errors. Please do not rely on this and always check it for yourself!

GPT Header

gpt_header.png

GPT Partition Entry

gpt_partition.png

Boot Sector

boot_sector.png

Disk Layout

- Boot Sector Address [HEX] = `First LBA` (GPT Entry, DEC) * `512`
- Size of cluster [BYTES] = `Bytes per sector` * `Sectors per cluster`
- First FAT Address [HEX] = `Boot Sector Address` + `Size of cluster` [IN DEC]
- Root Directory Address = First FAT Address + (`Small number of sectors` * `Sectors per cluster`) [HEX]
- Root Directory Size [BYTES] = `Number of possible root entries` * `Directory Entry size` (often 32 bytes)
- Data Region Address = `Root Directory Address` + `Root Directory Size` [IN HEX]
- Total FAT16 Volume in Bytes = `Small number of sectors` * `Bytes per sector`

Files

files_directories.png

- To find the next cluster index = `First FAT Address` + `Previous Cluster` * 2
- File contents of the cluster = `Start of data region` + `cluster_size` * `(cluster_index - 2)`

Time

1. First 5 bits for hours
2. Next 6 bits for minutes
3. Last 5 bits for seconds (you have seconds multiply by 2 to get the value, eg.: 00011 => 3, 3 * 2 => 6 seconds)

Date

1. First 7 bytes years since 1980 (eg.: if value in dec is 28, 1980 + 28 = 2008)
2. Next 4 bits for the month
3. Last 5 bits for the day

Sources

- UCLL
- Pieter Philippaerts