Tim Mallezie 2de bach. III

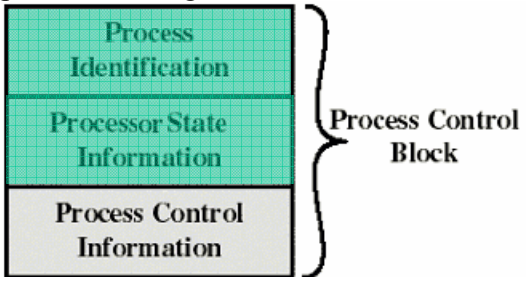
Architectuur van besturingssystemen: Vraag A4.

Procesbeheer: *creatie en wisselen van processen*.

a) Verduidelijk het begrip PCB.
b) Uit welke opeenvolgende stappen bestaat de creatie van een nieuw proces?
c) Hoe worden in UNIX en Linux nieuwe processen of threads gecreëerd?
d) Bij welke diverse gebeurtenissen krijgt een besturingssysteem de controle over het computersysteem? Leiden deze steeds tot een proceswisseling?
e) Bespreek stapsgewijs welke veranderingen aan de omgeving aangebracht worden bij een proceswisseling, en hoe dit gerealiseerd wordt.

PCB: Process Control Block.

Het besturingssysteem houdt de informatie in verband met het beheer van processen niet bij op een centrale plaats, maar in de geheugenruimte van elk proces afzonderlijk. De verzameling attributen wordt het PCB genmoed, het procesbesturingsblok. Het PCB moet steeds beschikbaar blijven in het hoofdgeheugen. Elke ingang in de primaire procestabel van het besturingssysteem bevat een verwijzing naar het PCB.
Het PCB mag enkel door instructies in kernelmodus gewijzigd worden. Het PCB bevat informatie in drie verschillende categorieën: procesidentificatie, processortoestandsinformatie en procesbesturingsinformatie.



Procesidentificatie: Vrijwel altijd wordt voor procesidentificatie gebruik gemaakt van een unieke numerieke code. De procesidentificatie kan ook codes bevatten die verwijzen naar andere processen, zoals bijvoorbeeld het ouderproces, of het volgende proces.

Processortoestandsinformatie: De processortoestandsinformatie bestaat uit de inhoud van alle processorregisters. Als het proces onderbroken wordt, wordt de registerinformatie hier opgeslaan. Bij systemen met meerdere processen bevindt zich hier ook een identificatie van de processor.

Procesbesturingsinformatie: De procesbesturingsinformatie bevat aanvullend informatie die het besturingssysteem nodig heeft voor het beheren van diverse processen.

- De scheduling- en toestandsinformatie, zoals procestoestand, de prioriteit, de gebeurtenissen waarop het proces wacht, …
- Structurele informatie waarmee PCB 's aan elkaar kunnen worden gekoppeld.
- Informatie over de precieze locaties van elk gedeelte van het procesbeeld.
- Verwijzingen naar andere geheugentabellen.
- Welke bronnen zijn aangevraagd, en welke effectief zijn toegewezen.
- Informatie over privileges van het proces.
- Eventuele limieten en quota.
- Informatie over diverse vlaggen, signalen en bericht die verband houden met de communicatie tussen onafhankelijke processen.

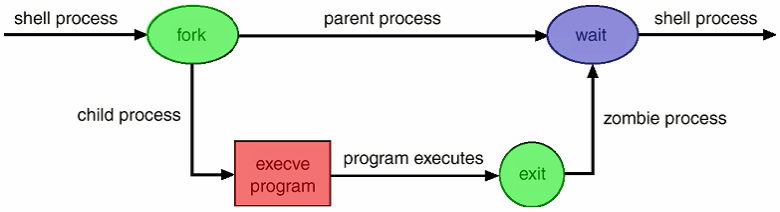
Architectuur van besturingssystemen: Vraag A4.

Creatie van een nieuw proces:

- Toewijzing van een unieke procesidentificatiecode en van een nieuwe ingang in de primaire procestabel.
- Toewijzing van ruimte voor alle elementen van het procesbeeld. Het besturingssysteem moet weten hoeveel ruimte noodzakelijk is, deze gegevens komen van het ouderproces of van een standaardinstelling.
- Initialisatie van het PCB, in het bijzonder de procesbesturingsinformatie.
- Instelling van de juiste koppelingen, bijvoorbeeld plaatsing in juiste wachtrij.
- Eventueel aanmaken of uitbreiden van andere gegevensstructuren.

Creatie van processen in UNIX:

UNIX ondersteunt geen multithreading.



In UNIX kan een nieuw proces enkel gecreëerd worden als kindproces van een al bestaand proces. Elk proces is altijd kindproces van het init() proces.

Een kindproces wordt gecreëerd via de aanroep fork(). Hierdoor wordt een element in de procestabel en een identificatie aan het proces toegekend. De kernel maakt dan een exacte kopie van de ouder, met uitzondering van gedeeld geheugen, en plaatst het kind in de toestand gereed. De identificatie van het kind wordt teruggegeven aan de ouder, en het kind krijgt de waarde 0. Dit vindt plaats in de kernelmodus van het ouderproces. Kind en ouder voeren op dit moment dezelfde code uit. In de meeste gevallen zal het kind nu andere code moeten uitvoeren. Hiertoe roept één van beide de systeemaanroep execve() aan. Hierdoor wordt de code in het procesbeeld vervangen door de inhoud van een bastand waarvan de naam als parameter door het ouderproces is opgegeven.
De duplicatie van de opdracht fork() is dus in dit geval overbodig.
De bewerkingen fork en execve worden in UNIX beschouwd als gescheiden opdrachten. Als een willekeurig proces execve() uitvoert, dan wordt het actieve proces onmiddellijk beëindigd, en start een nieuw programma in de context van het bestaande proces.

Creatie van processen en threads in Linux:

De rol van fork() in traditionele UNIX versies wordt in Linux vervuld door de systeemaanroep clone(). In tegenstelling tot fork() kunnen vlaggen als parameter worden meegegeven om aan te geven welke substructuren gedeeld worden door ouder en kindproces, en welke die gekopieerd dienen te worden. Als er geen vlaggen worden ingesteld doen clone() en fork() exact hetzelfde. Als er wel vlaggen worden ingesteld, worden de corresponderende substructuren niet gekopieerd maar gedeeld door ouder en kindproces. Het kindproces moet dus als kernel-thread beschouwd worden. Er is in de terminologie geen onderscheid tussen proces en thread, beide worden benoemd als task. Processen en threads worden intern identiek gerepresenteerd, een thread is een nieuwe task met dezelfde adresruimte als de oudertask.

Linux ondersteund de Pthreads API, op twee manieren; in de gebruikersmodus draaien alle threads in één proces (user-level implementatie) en in de modus met kernelondersteuning wordt gebruik gemaakt van de clone() aanroep (kernel-level implementatie).

Wanneer krijgt het besturingssysteem controle over het systeem?

Er zijn drie types van gebeurtenissen wanneer het besturingssysteem controle krijgt over het systeem.

1. **Interrupts:**

Interrupts worden veroorzaakt door een gebeurtenis die zich buiten het op dat moment actieve proces afspeelt. Bij het optreden van een interrupt wordt overgeschakeld naar de kernelmodus en wordt de besturing overgedragen aan een routine voor interruptafhandeling. De voert specifieke code uit afhankelijk van het type interrupt dat is opgetreden.
De belangrijkste interrupts zijn klokinterrupts, I/O-interrupts en paginafouten.
Klokinterrupts: Door periodieke (1-10 ms.) klokinterrupts kan het besturingssysteem vaststellen dat het actieve proces de maximaal toegelaten tijdsduur heeft bereikt en preëmptief moet onderbroken worden.
I/O-interrupts: I/O-interrupts verwittigen het besturingssysteem dat een I/O-gebeurtenis is uitgevoerd. Het besturingssysteem moet nu alle processen die op deze gebeurtenis wachtten in de toestand gereed of gereed-onderbroken brengen. Het kan beslissen het huidige proces te onderbreken ten gunste van een ander proces met een hogere prioriteit.
Paginafouten: Paginafouten worden veroorzaakt door verwijzingen naar geheugenelementen die ondertussen door het besturingssysteem tijdelijk uit het hoofdgeheugen zijn verwijderd. Dit komt omdat eigenlijk niet het volledige proces in het geheugen wordt geladen.
Paginafouten dienen synchroon afgehandeld te worden. het proces kan niet verder zolang het besturingssysteem geen gepaste actie heeft ondernomen.
I/O-interrupts en klokinterrupts zijn asynchrone interrupts, zij kunnen onmiddellijk uitgevoerd worden of uitgesteld worden. In Linux wordt dit uitstel geïmplementeerd via tasklets, in Windows NT worden dit Deferred Procedure Calls (DPC 's) genoemd.

2. **Traps:**

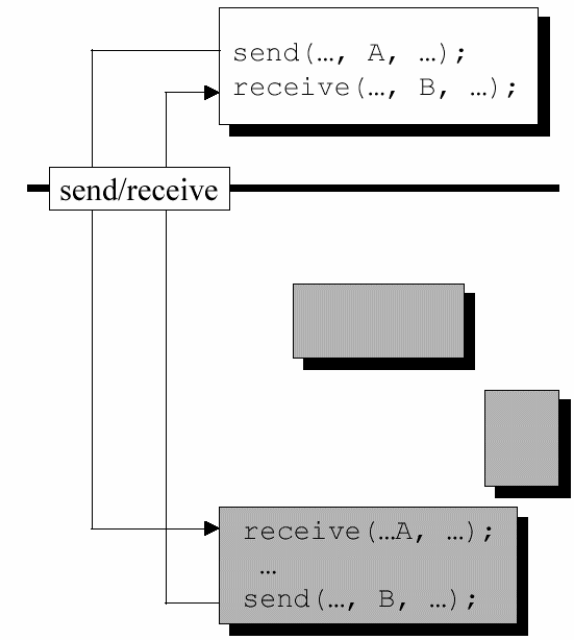
Een trap of exception, is een bijzondere vorm van een interrupt, die samenhangt met een fout of een uitzonderingsconditie die wordt gegenereerd binnen het op dat moment actieve proces. De reactie van het besturingssysteem is afhankelijk van het soort trap dat wordt gegenereerd. Traps worden synchroon met het veroorzakende proces afgehandeld.

3. **Systeemaanroepen:**

Systeemaanroepen of software-interrupts definiëren de interface tussen het besturingssysteem en de gebruikersprogramma's. De verzameling systeemaanroepen van een besturingssysteem wordt ook soms de software instructieset van het besturingssysteem genoemd. Gebruikersprogramma's kunnen niet rechtstreeks geprivilegieerde instructies uitvoeren, dus communiceren ze met behulp van systeemaanroepen met het besturingssysteem om diensten aan te vragen. Systeemaanroepen worden veroorzaakt op expliciet verzoek van het actieve proces. Ze veroorzaken een overdracht

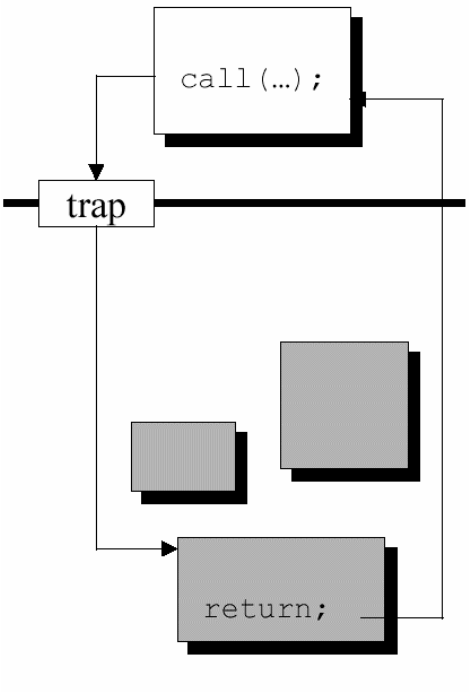
aan een routine die deel uitmaakt van de code van het besturingssysteem, die in kernelmodus wordt uitgevoerd. Systeemaanroepen kunnen op twee manieren gerealiseerd worden: berichtgestuurd of proceduregestuurd.

Berichtgestuurde systeemaanroepen:



In besturingssystemen met een microkernel of client/server architectuur is de interface tussen gebruikersprocessen en serverprocessen gebaseerd op het uitwisselen van berichten. Beide gebruiken een mechanisme dat beschouwd kan worden als I/O-operatie op een communicatiekanaal. Na het versturen van een aanvraagbericht laat het gebruikersproces zich door het besturingssysteem blokkeren tot het antwoord beschikbaar is. Analoog geven serverprocessen vrijwillig de controle aan de scheduler van zodra alle aan hun gerichte aanvragen beantwoord zijn, en blijven ze geblokkeerd zolang er geen nieuwe aanvragen zijn.

Proceduregestuurde systeemaanroepen:



In meer klassieke benaderingen komt met elke systeemaanroep een bibliotheekprocedure overeen die de parameters van de systeemaanroep op een bepaalde plaats plaatst en vervolgens een trap instructie genereert. Het doel van de bibliotheekprocedures is om ervoor te zorgen dat systeemaanroepen eruitzien als gewone procedureaanroepen. De trap instructie schakelt het systeem vanuit de gebruikersmodus naar de kernelmodus en geeft de controle over aan het besturingssysteem. Het besturingssysteem kijkt, aan e hand van de parameters, welke interne procedure moet uitgevoerd worden. De controle wordt hierna teruggegeven aan de bibliotheekprocedure die de statuscode als functiewaarde aflevert.

Software-interrupts krijgen meestal een relatief lage interruptprioriteit in vergelijking met I/O-interrupts. Zonder interrupts zou van simultane uitvoering van onafhankelijke activiteiten geen spraken kunnen zijn.

Wanneer kan het besturingssysteem een proceswisseling doorvoeren.

Het besturingssysteem kan een proceswisseling doorvoeren na het afhandelen van een klok- of I/O-interrupt.
Als een paginafout optreedt plaatst het besturingssysteem een I/O-verzoek om het stuk van het procesbeeld dat de toegangfout veroorzaakte binnen te halen. Op dat moment kan het besturingssysteem een ander proces activeren.
Het besturingssysteem kan een proceswisseling doorvoeren bij een trap, als die leidt tot een fatale fout die het huidige proces beëindigt.
Ook bij berichtgestuurde systeemaanroepen kan er een proceswisseling gebeuren.

Stappen bij het wisselen van processen.

Als er een interrupt wacht, dan wordt in een eerste fase volgende stappen doorlopen.

De context opgeslagen van het proces dat op dat moment wordt uigevoerd. Deze informatie is precies gelijk aan de processortoestandsinformatie van het PCB. Eén van de procesregisters moet dus steeds verwijzen naar het geheugenadres van de processortoestandsinformatie van dat proces. Het bewaren van de context kan dan eenvoudigweg in hardware worden uitgevoerd. Sommige processoren beschikken over meerdere registersets, zodat men niet bij elke contextwisseling moet kopiëren naar het geheugen, een wijziging van een pointer naar de huidige registerset volstaat.
De programmateller wordt hierna ingesteld op het beginadres van een programma voor de interruptafhandeling.
De processermodus wordt gewisseld naar kernelmodus, zodat de code voor interruptafhandeling geprivilegieerde instructies kan uitvoeren.
Hierna wordt verdergegaan met de instructiecyclus en wordt de eerste instructie van de routine voor interruptafhandeling uitgevoerd. Vervolgens wordt code uitgevoerd die tot de scheduler module van het besturingssysteem behoort. Deze code moet door de frequentie van interrupts zeer kort zijn. Tenslotte wordt de besturing terug overgedragen aan het proces dat net voor de interrupt werd uitgevoerd of aan een ander proces.
Het optrede van een interrupt leidt dus niet noodzakelijk tot een proceswisseling, wel zeker tot een context- en moduswisseling.
Indien beslist wordt de controle terug te geven aan het onderbroken proces, moet teruggeschakeld worden naar de gebruikersmodus, en moet de processortoestandsinformatie hersteld worden in de registers. Dit herstel wordt ook meestal in hardware uigevoerd.

Als de scheduler besluit om de besturing over te dragen aan een ander proces moeten softwarematig aanvullende veranderingen doorgevoerd worden.

- Naast het opslaan van de context van de processor moet ook de procesbesturingsinformatie van het procesbesturingsblok worden bijgewerkt.
- Het PCB moet naar de juist wachtrij verplaatst worden.

Architectuur van besturingssystemen: Vraag A4.

- Een ander proces moet als actief proces geselecteerd worden.
- Het PCB van dit proces moet worden bijgewerkt.
- De gegevenstructuren voor het geheugenbeheer moeten worden bijgewerkt.
- Het terugschakelen naar de gebruikersmodus en het laden van de processortoestandsinformatie van het PCB van het geselecteerde proces in de registers. Hiermee wordt de context teruggebracht naar de toestand zoals ze was op het moment dat het geselecteerde proces het laatst uit de toestand actief werd gewisseld.