

Deel I

Dit deel beslaat dezelfde stof als dat van de toets.

- 1a Leg het verschil uit tussen *user mode* en *kernel mode*. 5pt
- 1b MINIX is georganiseerd als een *client-server* besturingssysteem. Wat houdt dat in? 5pt

- 2a Als een CPU scheduler geïmplementeerd is als een apart proces, wanneer wordt het dan uitgevoerd? 5pt
- 2b Noem één belangrijk voordeel om een OS kernel *thread management* te laten doen. 5pt
- 2c Processen zijn doorgaans onderhevig aan *preemptive* scheduling, maar threads niet. Waarom is dit zo? 5pt

- 3a Leg uit hoe het volgende concurrente programma in een deadlock kan komen. Maak je eventuele assumpties expliciet. 5pt

<pre>(1) process producer(){ (2) while(true){ (3) produce_item(); (4) if (count==N) sleep(); (5) enter_item(); (6) count = count + 1; (7) if (count==1) wakeup(consumer); (8) } (9) }</pre>	<pre>(1) process consumer(){ (2) while(true){ (3) if (count==0) sleep(); (4) remove_item(); (5) count = count - 1; (6) if (count==N-1) wakeup(producer); (7) consume_item(); (8) } (9) }</pre>
---	--

- 3b Een binaire semafoor heeft twee *atomaire* operaties *lock()* en *unlock()* zoals hieronder gespecificeerd. Geef een deadlock-vrije oplossing voor het producer-consumer probleem met binaire semaforen. 5pt

<pre>atomic lock(s){ if (s.val == 1) s.val = s.val - 1; else{ enter_queue(s.q) sleep(); } }</pre>	<pre>atomic unlock(s){ if (length(s.q) > 0){ p = remove_head(s.q); wakeup(p); } else s.val = s.val + 1; }</pre>
--	--

- 4a Geef de algehele *flow of control* nadat een apparaat een elektrisch signaal naar de interrupt controller heeft gestuurd, tot het punt dat de interrupt afgehandeld is. 5pt
- 4b Kan een interrupt handler ook als een proces geïmplementeerd worden? Leg je antwoord uit. 5pt

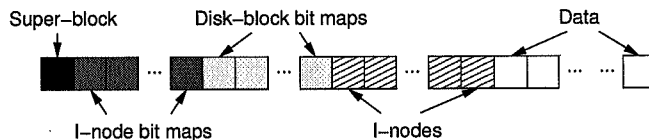
Deel II

- 4a Hoeveel rijen heeft een *single-level page table* in een *n-bit* adresruimte met een pagina grootte van 2^p bytes? 5pt
- 4b Hoe zou je de fysieke grootte berekenen (i.e., in bytes) van een *single-level page table*? 5pt
- 4c Beschouw een proces dat veel virtueel geheugen nodig heeft. Zo je voorkeur dan uitgaan naar een *single-level* of een *two-level page table*? Leg je antwoord uit. 5pt
- 4d Leg uit wat een *inverted page table* is, en hoe een geheugen locatie opgezocht wordt. 5pt

- 5a MINIX gebruikt een apart proces *system task* geheten. Wat doet dit proces en waarom is het nodig? 5pt

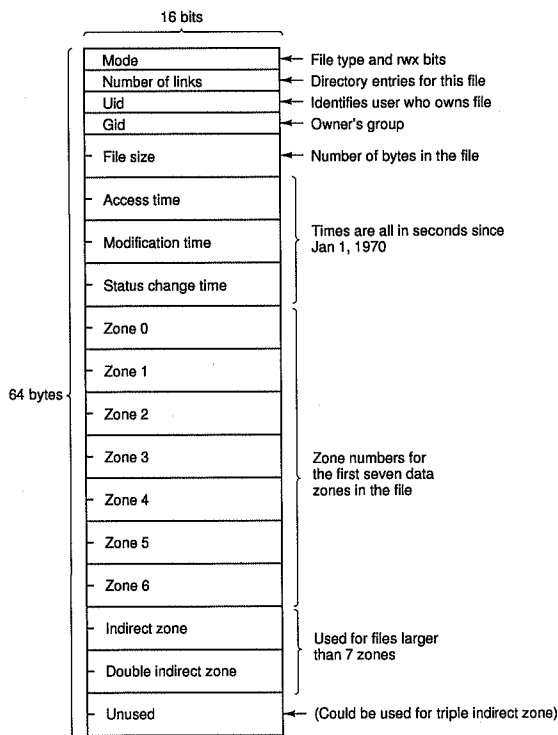
5b Geef minstens één overtuigend argument waarom de MINIX file manager van processen moet afweten. 5pt

6a Een UNIX-achtig filesystem heeft doorgaans een disk layout zoals hieronder weergegeven. Leg uit wat er typisch in het super-block gevonden kan worden. 5pt



6b Een MINIX i-node is 64 bytes. Met een disk-blok grootte van 1 Kbyte, hoeveel disk blocks dienen er gereserveerd te worden voor de opslag van i-nodes en i-node bit maps als er een totaal van N files ondersteund moeten kunnen worden? 5pt

6c Ga ervanuit dat zones en disk blokken elk een grootte hebben van 1Kbyte. Wat is dan de maximale grootte van een MINIX file, gegeven dat disk blokken geadresseerd worden met 32-bit pointers. 5pt



Cijferbepaling: (1) Tel per deel de behaalde punten bij elkaar op. (2) Laat T het totaal aantal behaalde punten voor de toets zijn ($0 \leq T \leq 45$); $D1$ het aantal behaalde punten voor deel 1; $D2$ het aantal behaalde punten voor deel 2. Het eindtotaal E is dan $\max\{T, D1\} + D2 + 10$.