ΛΕΙΤΟΥΡΓΙΚΑ ΣΥΣΤΗΜΑΤΑ

Επισκόπηση Λ.Σ.
Ερωτήσεις Επανάληψης

Υλικό από:
Modern Operating Systems, A.S. Tanenbaum

Σύνθεση
Κ.Γ. Μαργαρίτης, Τμήμα Εφαρμοσμένης Πληροφορικής, Πανεπιστήμιο Μακεδονίας

ΠΡΟΒΛΗΜΑΤΑ

1. Ποιες είναι οι δύο κύριες λειτουργίες ενός λειτουργικού συστήματος;
2. Τι σημαίνει πολυπρογραμματισμός;
3. Τι σημαίνει παροχέτευση; Πιστεύετε ότι οι εξελιγμένοι προσωπικοί υπολογιστές θα έχουν την παροχέτευση ως τυπικό (standard) χαρακτηριστικό στο μέλλον;
4. Στους πρώτους υπολογιστές, η CPU χειριζόταν κάθε byte δεδομένων που διαβαζόταν ή γράφονταν (δηλαδή δεν υπήρχε DMA). Ποιες είναι οι συνέπειες αυτής της οργάνωσης στον πολυπρογραμματισμό;
5. Γιατί ο χρονομερισμός δεν ήταν διαδεδομένος στους υπολογιστές δεύτερης γενιάς;
6. Η ιδέα της οικογένειας υπολογιστών παρουσιάστηκε τη δεκαετία του 1960 με τους μεγάλους υπολογιστές System/360 της IBM. Είναι σήμερα αυτή η ιδέα ζωντανή ή έχει φάει τα ψωμιά της;
7. Ένας από τους λόγους που τα GUI άργησαν να υιοθετηθούν ήταν το κόστος του υλικού που χρειαζόταν για την υποστήριξή τους. Πόση μνήμη (RAM) οθόνης χρειάζεται για να υποστηρίξει μια μονόχρωμη οθόνη κειμένου, με χωρητικότητα 25 γραμμές × 80 στήλες; Πόση χρειάζεται για μια έγχρωμη οθόνη ψηφιογραφικών (bitmap) με ανάλυση 1024 × 768 pixel στα 24 bit; Ποιο ήταν το κόστος της μνήμης αυτής σε τιμές του 1980 (κόστιζε 5 δολάρια το κάθε KB μνήμης); Ποιο είναι το κόστος σήμερα;
8. Ποια από τις ακόλουθες εντολές μπορεί να επιτραπεί μόνο σε κατάσταση λειτουργίας πυρήνα;
   (α) Απενεργοποίηση όλων των διακοπών.
   (β) Ανάγνωση του ρολογιού που δείχνει την κανονική ώρα (24ωρη).
   (γ) Απόδοση τιμής στο παραπάνω ρολόι.
   (δ) Αλλαγή του χάρτη μνήμης.
9. Δώστε μερικές διαφορές ανάμεσα στα λειτουργικά συστήματα προσωπικών υπολογιστών και τα λειτουργικά συστήματα μεγάλων υπολογιστών.

10. Ένας υπολογιστής διαθέτει μια διοχέτευση με τέσσερα στάδια. Κάθε στάδιο χρειάζεται τον ίδιο χρόνο για να ολοκληρώσει την εργασία του, για παράδειγμα 1 nsec. Πόσες εντολές εκτελεί αυτή η μηχανή ανά δευτερόλεπτο;

11. Ένας διορθωτής διαπιστώνει ένα ορθογραφικό λάθος που εμφανίζεται συνέχεια στο χειρόγραφο ενός εγχειριδίου λειτουργικού συστήματος που πρόκειται να σταλεί για εκτύπωση. Το εγχειρίδιο εκτείνεται σε 700 περίπου σελίδες, που η κάθε μία τους έχει 50 γραμμές των 80 χαρακτήρων. Πόσος χρόνος χρειάζεται για την ηλεκτρονική σάρωση του κειμένου, όταν το πρωτότυπο βρίσκεται στο καθένα από τα επίπεδα μνήμης της Εικόνας 1-7; Για τις μεθόδους εσωτερικής αποθήκευσης, υποθέστε ότι ο χρόνος πρόσβασης δίνεται ανά χαρακτήρα, για τις συσκευές δίσκων ο χρόνος δίνεται ανά μπλοκ των 1024 χαρακτήρων, και για τις ταινίες ο χρόνος δίνεται για την έναρξη της διαδικασίας ενώ στη συνέχεια ο χρόνος πρόσβασης είναι ταυτόσημος με αυτόν που δόθηκε για τους δίσκους.

12. Στην Εικόνα 1-9, η MMU συγκρίνει την εισερχόμενη (εικονική) διεύθυνση με τον καταχωρητή ορίου, προκαλώντας σφάλμα εκτέλεσης αν είναι μεγαλύτερη από αυτόν. Μια εναλλακτική σχεδίαση προτείνει την πρόσθεση της εικονικής διεύθυνσης στον καταχωρητή βάσης και στη συνέχεια τη σύγκριση του αποτελέσματος με τη φυσική διεύθυνση του καταχωρητή ορίου. Είναι οι δύο μέθοδοι λογικά ισοδύναμες; Είναι ισοδύναμες με βάση τις επιδόσεις;

13. Όταν ένας χρήστης προκαλεί μια κλήση συστήματος για να διαβάσει ή να γράψει ένα αρχείο στο δίσκο, την εφοδιάζει με παραμέτρους που καθορίζουν: (α) ποιο αρχείο τον ενδιαφέρει, (β) ένα δείκτη προς την περιοχή προσωρινής αποθήκευσης των δεδομένων και (γ) το μετρητή byte. Ο έλεγχος μεταφέρεται τότε στο λειτουργικό σύστημα, το οποίο καλεί τον κατάλληλο οδηγό. Ας υποθέσουμε ότι ο οδηγός ξεκινά τη λειτουργία του δίσκου και την τερματίζει όταν εκδηλώνεται μια διακοπή. Στην περίπτωση της ανάγνωσης από το δίσκο, προφανώς η διεργασία του χρήστη πρέπει να μπλοκαριστεί (επειδή δεν υπάρχουν ακόμη τα δεδομένα που ζήτησε). Τι συμβαίνει στην περίπτωση της εγγραφής στο δίσκο; Πρέπει να μπλοκαριστεί η διεργασία του χρήστη περιμένοντας την ολοκλήρωση της μεταφοράς των δεδομένων στο δίσκο;

14. Ποια είναι η κυριότερη διαφορά ανάμεσα στις παγίδες και τις διακοπές;

15. Ένας υπολογιστής χρησιμοποιεί τη διάταξη επανατοποθέτησης της Εικόνας 1-9(α). Ένα πρόγραμμα έχει μέγεθος 10.000 byte και έχει φορτωθεί στη διεύθυνση 40.000. Τι τιμές θα αποδοθούν στους καταχωρητές *βάσης* και *ορίου* σύμφωνα με τη διάταξη που περιγράφεται στο κείμενο;

16. Γιατί ο πίνακας διεργασιών είναι απαραίτητος στα συστήματα χρονομερισμού; Είναι επίσης απαραίτητος στα συστήματα προσωπικών υπολογιστών στα οποία υπάρχει μία μόνο διεργασία, η οποία καταλαμβάνει ολόκληρη τη μηχανή μέχρι να τελειώσει;

17. Υπάρχει κάποιος λόγος να ενσωματωθεί κάποιο σύστημα αρχείων σε έναν κατάλογο που δεν είναι άδειος; Αν ναι, ποιος είναι ο λόγος αυτός;

18. Για κάθε μία από τις επόμενες κλήσεις συστήματος, δώστε μια συνθήκη που τις αναγκάζει να τερματιστούν αναπάντεχα: fork, exec, και unlink.

19. Μπορεί η κλήση:

```
count = write(fd, buffer, nbytes);
```

να επιστρέψει διαφορετική τιμή από την *nbytes*; Αν ναι, για ποιο λόγο;

20. Ένα αρχείο που διαθέτει τον περιγραφέα αρχείου *fd* περιέχει την επόμενη ακολουθία byte: 3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5. Προκαλούνται οι επόμενες κλήσεις συστήματος:

```
lseek(fd, 3, SEEK_SET);
read (fd, &buffer, 4);
```

όπου η κλήση lseek αναζητά το τρίτο byte του αρχείου. Τι περιέχει η μεταβλητή *buffer* μετά το πέρας της ανάγνωσης;

21. Ποια είναι η ουσιαστική διαφορά μεταξύ ενός ειδικού αρχείου μπλοκ και ενός ειδικού αρχείου χαρακτήρων;

22. Στο παράδειγμα που δόθηκε στην Εικόνα 1-17, η διαδικασία βιβλιοθήκης ονομάζεται *read* και η αντίστοιχη κλήση συστήματος ονομάζεται επίσης read. Είναι απαραίτητο να έχουν το ίδιο όνομα; Αν όχι, ποια από τις δύο είναι πιο σημαντική;

23. Το μοντέλο πελάτη-διακομιστή είναι δημοφιλές στα κατανεμημένα συστήματα. Μπορεί να χρησιμοποιηθεί και στα συστήματα που απαρτίζονται από ένα μόνον υπολογιστή;

24. Από τη σκοπιά του προγραμματιστή, οι κλήσεις συστήματος μοιάζουν με τις υπόλοιπες κλήσεις σε διαδικασίες βιβλιοθήκης. Είναι σημαντικό να γνωρίζει ο προγραμματιστής ποιες διαδικασίες βιβλιοθήκης έχουν ως αποτέλεσμα κλήσεις συστήματος; Κάτω από ποιες περιστάσεις και για ποιο λόγο;

25. Στην Εικόνα 1-23 φαίνεται ξεκάθαρα ότι για μια σειρά κλήσεων συστήματος του UNIX δεν υπάρχουν αντίστοιχες κλήσεις στο Win32 API. Για κάθε μία από τις κλήσεις αυτές, μελετήστε τις συνέπειες που προκαλεί στην προσπάθεια του προγραμματιστή να μετατρέψει ένα πρόγραμμα του UNIX ώστε να μπορεί να εκτελεστεί στα Windows.

26. Ακολουθούν μερικές ερωτήσεις για να γίνει εξάσκηση στη μετατροπή μονάδων:

(α) Πόσα δευτερόλεπτα διαρκεί ένα μικρο-έτος (microyear);

(β) Τα μικρόμετρα (micrometers) λέγονται συχνά microns. Ποιο είναι το μήκος ενός gigamicron;

(γ) Πόσα byte περιέχονται σε μνήμη 1 TB;

(δ) Η μάζα της Γης είναι 6000 yottagram. Πόσα χιλιόγραμμα (kilograms) είναι η ποσότητα αυτή;

27. Γράψτε ένα κέλυφος παρόμοιο με αυτό της Εικόνας 1-19, αλλά που να περιέχει αρκετό πραγματικά εκτελέσιμο κώδικα ώστε να μπορέσετε να το ελέγξετε. Μπορείτε επίσης να προσθέσετε διάφορες δυνατότητες όπως ανακατεύθυνση εισόδου/εξόδου, διυχετεύσεις, και δυνατότητα εκτέλεσης εργασιών στο παρασκήνιο.

28. Αν έχετε διαθέσιμο κάποιο προσωπικό σύστημα τύπου UNIX (Linux, MINIX, Free BSD, κλπ.), στο οποίο μπορείτε άνετα να προκαλείτε κατάρρευση (crash) και στη συνέχεια να το επανεκκινείτε, γράψτε ένα σενάριο κελύφους (shell script) το οποίο επιχειρεί να δημιουργήσει άπειρες θυγατρικές διεργασίες και παρατηρήστε τι θα συμβεί.

Πριν κάνετε το πείραμα, δώστε τη διαταγή sync στο κέλυφος για να αδειάσετε τις περιοχές προσωρινής αποθήκευσης του συστήματος, αποφεύγοντας την καταστροφή του συστήματος αρχείων. **Σημείωση:** Μη δοκιμάσετε το πείραμα αυτό σε κοινόχρηστα συστήματα, χωρίς άδεια από το διαχειριστή του συστήματος. Οι συνέπειες θα είναι ορατές σε δευτερόλεπτα, με αποτέλεσμα να σας πιάσουν στα πράσα και να ακολουθήσουν οι ανάλογες κυρώσεις.

29. Εξετάστε και προσπαθήστε να ερμηνεύσετε τα περιεχόμενα ενός καταλόγου τύπου UNIX ή Windows με κάποιο εργαλείο όπως το πρόγραμμα *od* του UNIX ή το πρόγραμμα *DEBUG* του MS-DOS. *Υπόδειξη:* Ο τρόπος που θα επιλέξετε εξαρτάται από το τι επιτρέπει το λειτουργικό σύστημα. Ένα τέχνασμα που είναι πιθανό να δουλέψει είναι η δημιουργία ενός καταλόγου σε δισκέτα με βάση κάποιο λειτουργικό σύστημα, και στη συνέχεια η ανάγνωση των φυσικών δεδομένων (raw data) της δισκέτας με τη χρήση ενός διαφορετικού λειτουργικού συστήματος, που να επιτρέπει τέτοια πρόσβαση.

5. Κλήσεις συστήματος

Σε ένα περιβάλλον προγραμματισμού C/C++ που έχετε πρόσβαση προσπαθήστε να εντοπίσετε τις βιβλιοθήκες C που σας επιτρέπουν να εκτελέσετε τις κλήσεις συστήματος POSIX.

Στη συνέχεια προσπαθήστε να μεταγλωττίσετε και να εκτελέσετε τα απλά προγράμματα φλοιού που δίνονται στις σημειώσεις.

Βρείτε στο Διαδίκτυο αντίστοιχα απλά προγράμματα και δοκιμάστε να τα εκτελέσετε.