ΔΙΑΧΥΤΑ ΚΑΙ ΕΝΣΩΜΑΤΩΜΕΝΑ ΣΥΣΤΗΜΑΤΑ

CPUs

Διδάσκων:
Παναγιώτης Καρκαζής

Περίγραμμα

- Συσκευές I/O

- Διακοπές, Εξαιρέσεις, Παγίδες

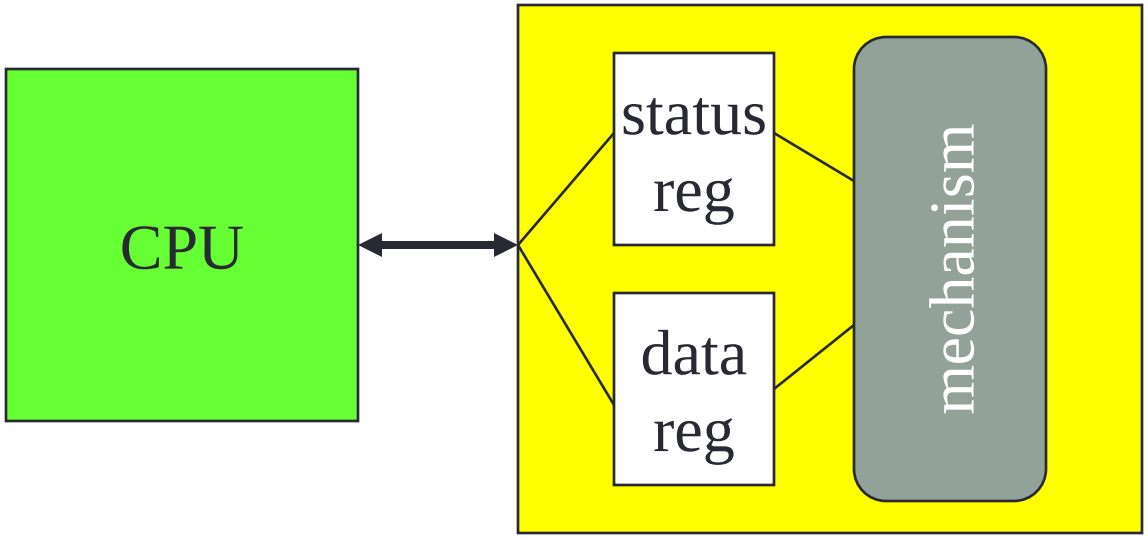
- Καταστάσεις λειτουργίας

- Συνεπαξεργαστές

Συσκευές I/O

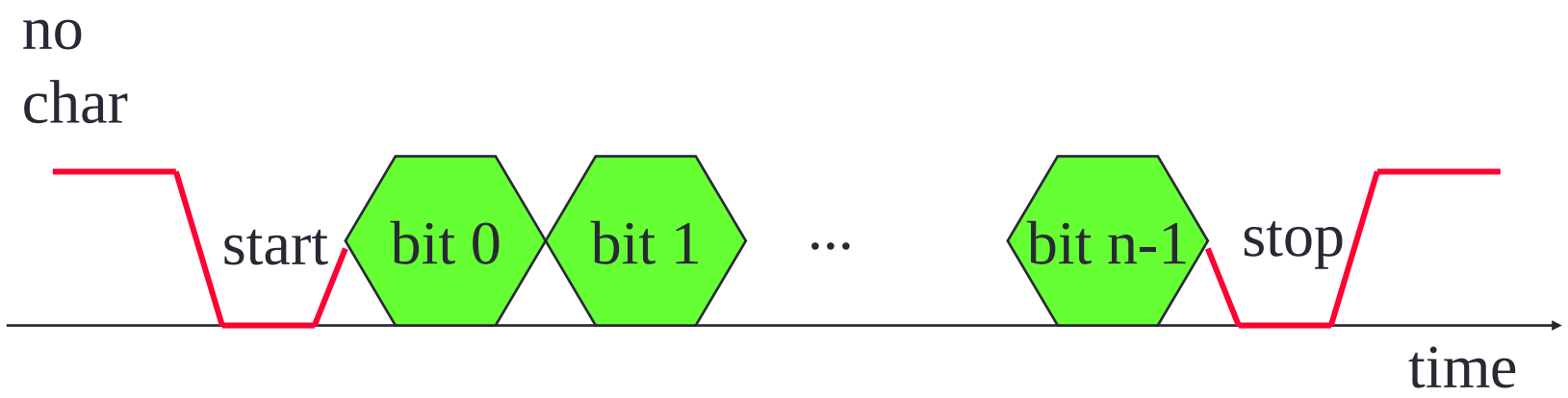
- Οι συσκευές I/O (input-output) διακρίνονται σε ψηφιακές και αναλογικές
- Η επικοινωνία μεταξύ της CPU και I/O γίνεται μέσω των καταχωρητών **δεδομένων** και **κατάστασης**

Τυπική διεπαφή I/O με την CPU:



Σειριακή επικοινωνία

- Τα δεδομένα μεταδίδονται σειριακά
- Η αρχή και το τέλος της μετάδοσης καθορίζεται από τα start/stop bits
- Η ταχύτητα μετάδοσης παραμένει σταθερή καθ' όλη την διάρκεια της μετάδοσης



Παράμετροι επικοινωνίας

Πριν την έναρξη της επικοινωνίας πρέπει να καθοριστούν οι παρακάτω παράμετροι και στα δύο μέρη:

- Ρυθμός μετάδοσης (baud rate)
- Αριθμός bit ανά χαρακτήρα
- ΄Υπαρξη ισοτιμίας (parity)
- Είδος ισοτιμίας (even/odd)
- Μήκος του stop bit (1,1.5,2 bits)

Τεχνικές διαχείρισης I/O

- Υπάρχουν διαφορετικές τεχνικές διαχείρισης των I/O μέσω εντολών:
  - ειδικού σκοπού (programmable I/O)
  - χαρτογράφησης μνήμης(memory mapped)
- Οι περισσότερες CPU χρησιμοποιούν memory mapped προσέγγιση
- Η μία τεχνική δεν αποκλείει την άλλη

Memory – mapped example

- Ορισμός θέσης μνήμης της συσκευής

```
DEV1 EQU 0x1000
```

- Κώδικας προσπέλασης:

```
LDR r1,#DEV1 ; set up device adrs
LDR r0,[r1] ; read DEV1
LDR r0,#8 ; set up value to write
STR r0,[r1] ; write value to device
```

Προσπέλαση με χρήση HLL

- Οι γλώσσες υψηλού επιπέδου (High Level Languages) παρέχουν συναρτήσεις για την προσπέλαση των I/O, οι οποίες δέχονται ορίσματα δείκτες μνήμης

- Ανάγνωση μέσω pointer:

```
int peek(char *location) {
  return *location; }
```

- Εγγραφή μέσω pointer:

```
void poke(char *location, char newval) {
  (*location) = newval; }
```

Πρόσβαση με αναμονή απασχόλησης (busy/wait)

- Βασίζεται στον συνεχή έλεγχο της κατάστασης της συσκευής, ώστε να αποφευχθεί η εγγραφή/ανάγνωση δεδομένων σε στιγμή που η συσκευή είναι απασχολημένη

- Είναι απλός αλλά αναποτελεσματικός τρόπος διαχείρισης

```
current_char = mystring;
while (*current_char != '\0') {
    poke(OUT_CHAR,*current_char);
    while (peek(OUT_STATUS) != 0);
        current_char++;
}
```

Busy Wait

Ταυτόχρονη πρόσβαση I/O

Παράδειγμα ταυτόχρονης πρόσβασης με αναμονή απασχόλησης:

```
while (TRUE) {
      /* read */
      while (peek(IN_STATUS) == 0);
            achar = (char)peek(IN_DATA);
            /* write */
            poke(OUT_DATA,achar);
            poke(OUT_STATUS,1);
      while (peek(OUT_STATUS) != 0);
 }
```

Διακοπές I/O

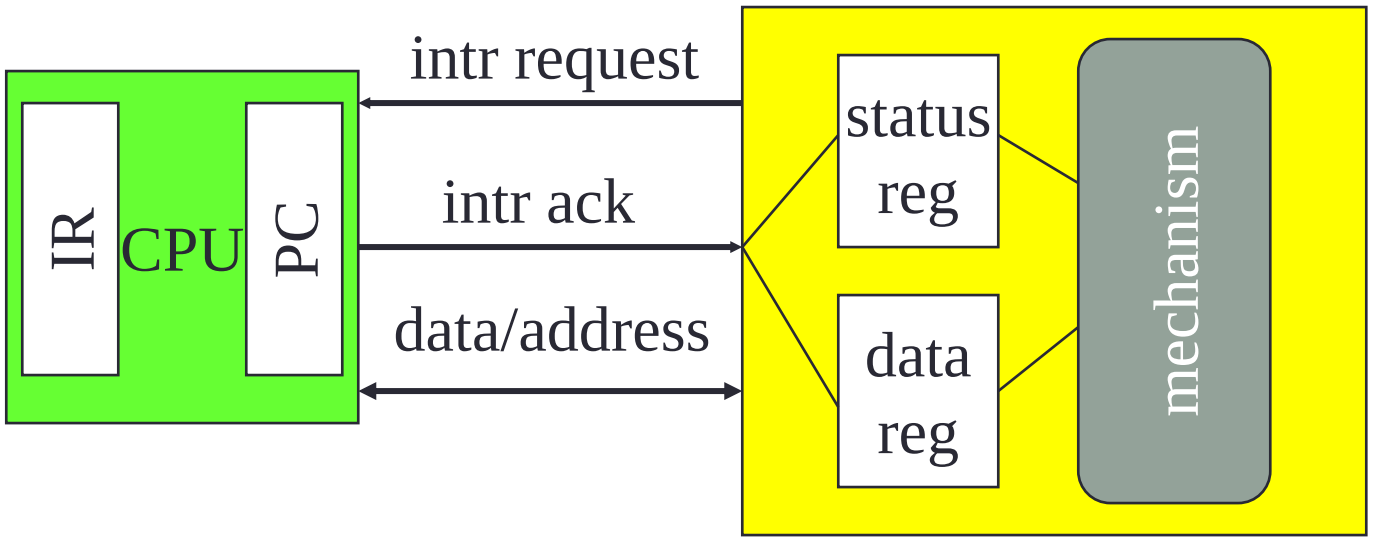
- Η πρόσβαση με αναμονή λόγω καθυστέρησης αναγκάζει την CPU να αναλώνεται στον συνεχή έλεγχο της κατάστασης των I/O με αποτέλεσμα να καθιστά δύσκολη την εκτέλεση παράλληλων εργασιών

- Οι διακοπές επιτρέπουν στην CPU να διακόψουν την εκτέλεση του τρέχοντος προγράμματος και να εξυπηρετήσουν την συσκευή I/O μέσω εκτέλεσης της κατάλληλης υπο-ρουτίνας χειρισμού

- Η εξυπηρέτηση των διακοπών απαιτεί την αποθήκευση του PC

Χειρισμός διακοπής

1. Η συσκευή σηματοδοτεί την διακοπή
2. Η CPU επιβεβαιώνει την λήψη και καλεί την υπο-ρουτίνα εξυπηρέτησης της διακοπής



Εξυπηρέτηση διακοπής

```
void input_handler() {
  achar = peek(IN_DATA);
  gotchar = TRUE;
  poke(IN_STATUS,0);
}

void output_handler() {}
```

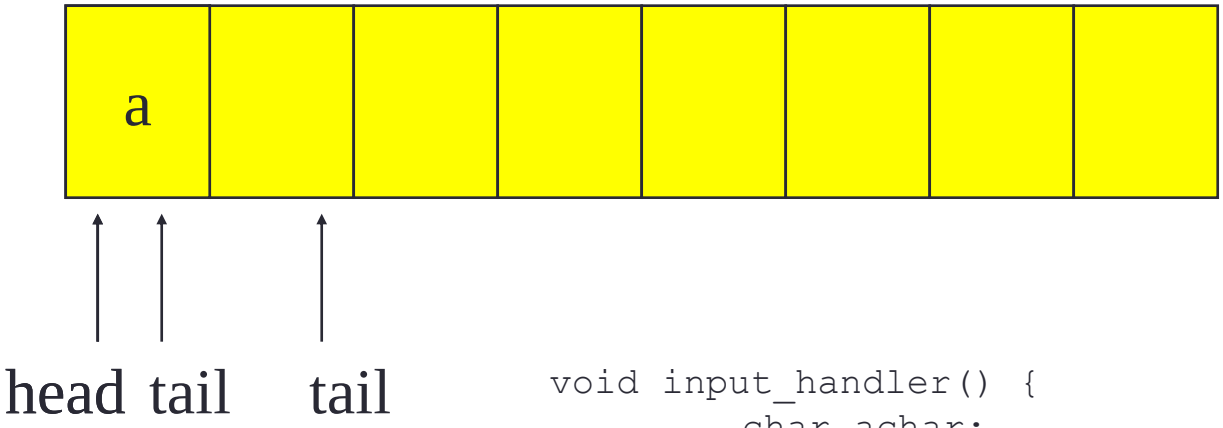
Υπο-ρουτίνες εξυπηρέτησης

```
main() {
  while (TRUE) {
        if (gotchar) {
              poke(OUT_DATA,achar);
              poke(OUT_STATUS,1);
              gotchar = FALSE;
              }
        }
  }
```

Κυρίως πρόγραμμα

Διακοπές I/O με χρήση buffer

- Η χρήση ουράς (buffer) επιτρέπει διαχείριση I/O με διαφορετικούς ρυθμούς ανάγνωσης εγγραφής




```
void input_handler() {
        char achar;
        if (full_buffer()) error = 1;
        else { achar = peek(IN_DATA);
               add_char(achar); }
        poke(IN_STATUS,0);
        if (nchars == 1){
            poke(OUT_DATA,remove_char();
             poke(OUT_STATUS,1); }
        }
```

Σφάλματα κατά την εξυπηρέτηση διακοπών

- Προκύπτουν κυρίως λόγω του ότι κάποιοι καταχωρητές που τροποποιούνται δεν επαναφέρονται στην αρχική τους τιμή

- Είναι πολύ δύσκολο να απομονωθεί το πρόβλημα κατά την διαδικασία αποσφαλμάτωσης, διότι οι επιπτώσεις του σφάλματος στο κύριο πρόγραμμα εξαρτώνται από την χρονική στιγμή που θα σηματοδοτηθεί η διακοπή

- Η υλοποίηση διακοπών απαιτεί την ύπαρξη «στοίβας»

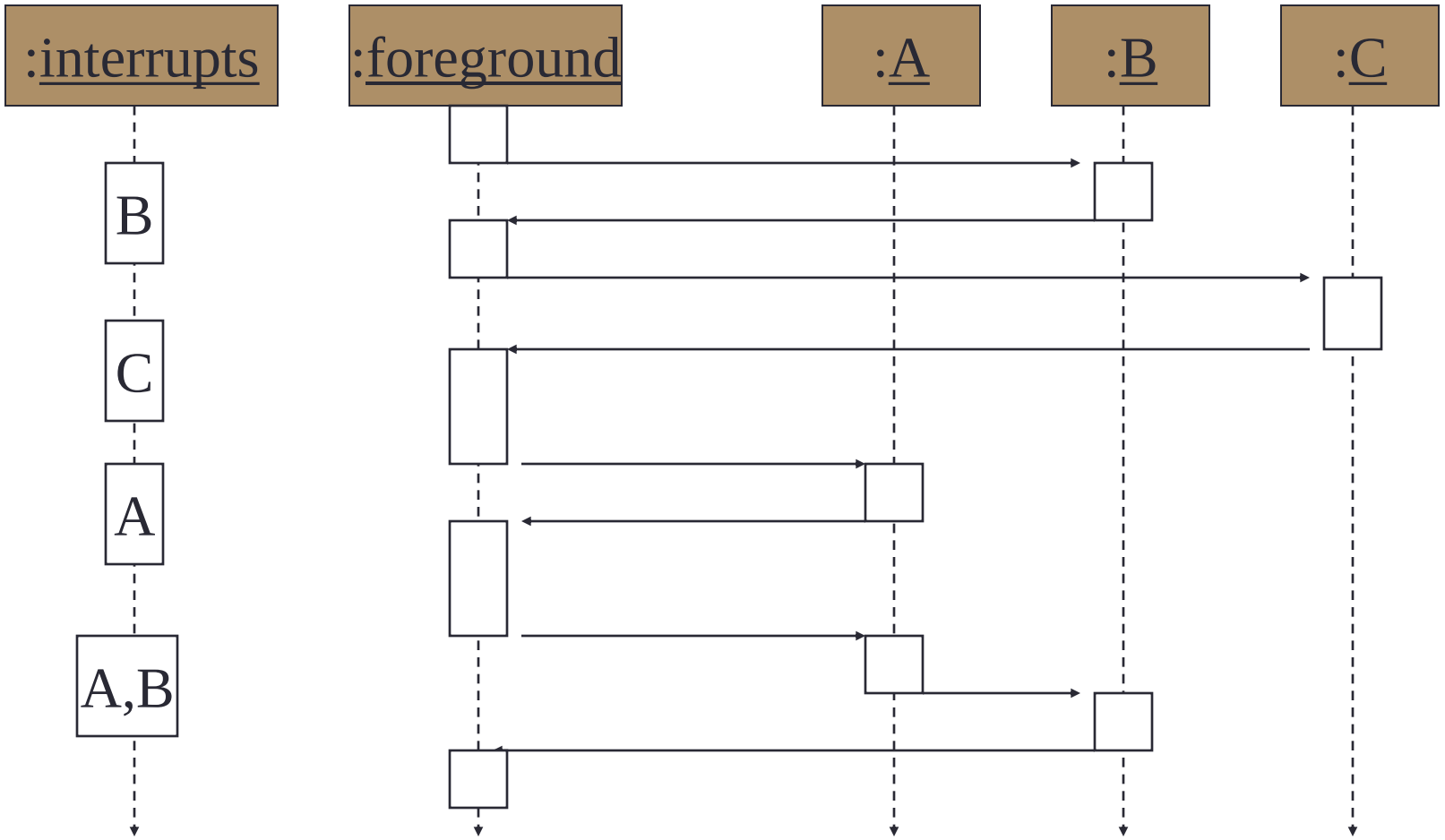
Προτεραιότητες

- **Masking:** Η εφαρμογή μάσκας καθορίζει ποια διακοπή θα εξυπηρετηθεί πρώτη.

- Διακοπή χαμηλότερης προτεραιότητας εξυπηρετείται μετά το πέρας της τρέχουσας

- **Non-maskable interrupt:** Στην διακοπή με την υψηλότερη προτεραιότητα δεν εφαρμόζεται ποτέ μάσκα ώστε να εξυπηρετείται πάντα άμεσα

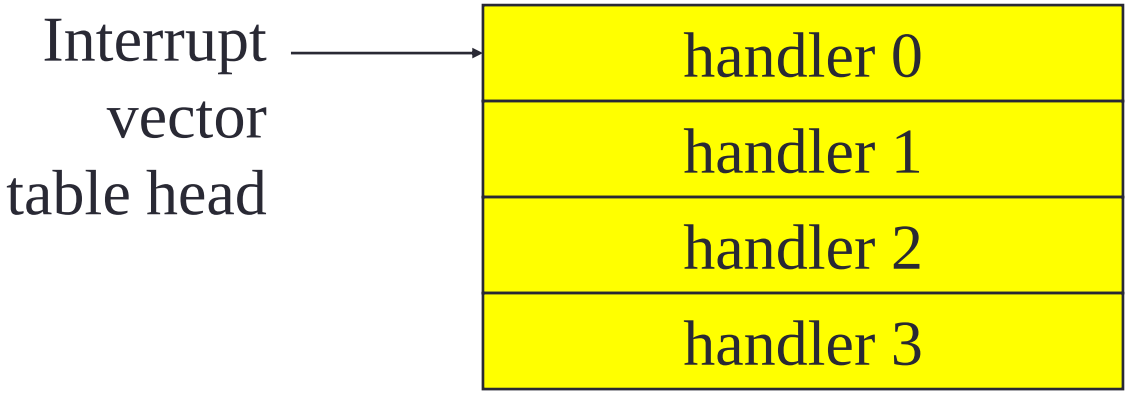
Παράδειγμα



Διανύσματα διακοπής

- Επιτρέπουν τον χειρισμό διαφορετικών διακοπών από διαφορετική υπο-ρουτίνα εξυπηρέτησης.

- Τα διανύσματα μπορεί να αποστέλλονται από τις συσκευές ή να είναι αποθηκευμένα στην μνήμη



Διαδικασία διακοπής

- Η CPU επιβεβαιώνει την διακοπή
- Η συσκευή αποστέλλει το διάνυσμα διακοπής
- Η CPU καλεί την υπο-ρουτίνα εξυπηρέτησης
- Εκτελείται η υπο-ρουτίνα
- Η CPU τον PC στην ροή εκτέλεσης του κανονικού προγράμματος

Κόστος εκτέλεσης διακοπών

- Χρόνος εκτέλεσης του κώδικα χειρισμού της διακοπής
- Καθυστέρηση στην CPU λόγω υλοποίησης του μηχανισμού της διακοπής
- Καθυστέρηση λόγω αποθήκευσης/αποκατάστασης των τιμών των καταχωρητών
- Καθυστέρηση κατά την εκτέλεση των πρώτων κύκλων του pipeline

Καταστάσεις λειτουργιάς

- Καταστάσεις λειτουργίας
  - User

    Unprivileged mode
  - FIQ

    High priority Interrupt is raised
  - IRQ

    Low priority Interrupt is raised
  - Supervisor

    Reset – privileged mode
  - Abort
    - Memory access violations

- Τα περισσότερα προγράμματα εκτελούνται σε κατάσταση χρήστη

Supervisor mode (κατάσταση επιβλέποντος)

- Όταν απαιτείται η λειτουργία πολλαπλών διεργασιών τότε πρέπει να διασφαλιστεί ότι οι κάθε διεργασία θα διαχειρίζεται την δική της μνήμη και θα έχει πρόσβαση σε συσκευές (πχ I/O) μόνο όταν αυτές είναι διαθέσιμές

- Αυτό το επίπεδο ασφάλειας παρέχεται από τα λειτουργικά συστήματα

- Όταν μια διεργασία εκτελείται σε κατάσταση supervisor mode της παρέχεται πρόσβαση (privileges) σε όλο το υλικό του συστήματος

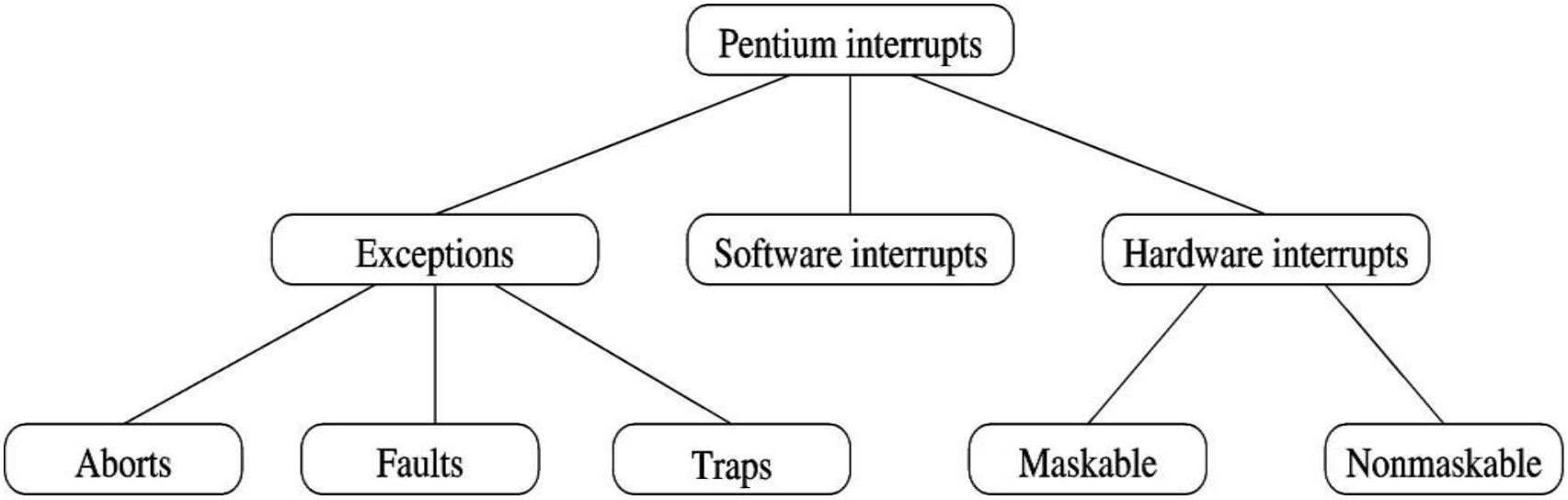
- Τα λειτουργικά συστήματα συνήθως λειτουργούν σε κατάσταση supervisor mode

Εξαιρέσεις

- Προκαλούνται από εσωτερικό σφάλμα (πχ. διαίρεση μηδέν)
- Δεν είναι προβλέψιμες
- Εκτρέπουν την ροή του προγράμματος, όπως και οι διακοπές
- Εξυπηρετούνται με τον μηχανισμό των διακοπών
- Υποστηρίζουν προτεραιότητες και διανύσματα

Παγίδες

- Είναι μια διακοπή πού παράγεται από εντολή
  Πχ: κατά την μετάβαση σε supervisor mode



Συνεπεξεργαστές

- Οι συνεπεξεργαστές συνδέονται με τη CPU που παρέχουν επιπρόσθετες λειτουργίες
  Πχ. εκτέλεση πράξεων κινητής υποδιαστολής
- Υπάρχουν συγκεκριμένες εντολές (op-codes) που εκτελούνται στον συνεπεξεργαστή
- Η CPU συνήθως λειτουργεί παράλληλα εκτελώντας την επόμενη εντολή

Βιβλιογραφία

- W. Wolf, - “Computers as Component”

Καλό διάβασμα