Όνο/επώνυμο:Ιωάννης Χατζής
Αριθμός Μητρώου:265

Θέμα Εργασίας

«Λειτουργικά Συστήματα Πραγματικού Χρόνου»

επιβλέπων: Μηνάς Δασυγένης
http://arch.icte.uowm.gr

Εισαγωγή

* **Το λειτουργικό σύστημα είναι ένα ολοκληρωμένο σύστημα προγραμμάτων το οποίο χρησιμοποιείται για την διαχείριση πόρων όπως η Κεντρική Μονάδα Επεξεργασίας, η μνήμη, οι δίσκοι, οι συσκευές εισόδου εξόδου.**

* **Το λειτουργικό σύστημα δημιουργεί ένα περιβάλλον επικοινωνίας του χρήστη με τον υπολογιστή.**

Ορισμός Λειτουργικού Συστήματος

* **Τα Λειτουργικά Συστήματα Πραγματικού Χρόνου ή Real Time Operating Systems (από εδώ και στο εξής RTOS), συνιστούν τυπικά μια υποκατηγορία των ενσωματωμένων λειτουργικών συστημάτων .**

* **Τα λειτουργικά συστήματα πραγματικού χρόνου διαφέρουν από τα υπόλοιπα διότι εξυπηρετούν αιτήματα από διάφορες διεργασίες σε πραγματικό χρόνο. Πρέπει να διαχειρίζονται τα δεδομένα όπως έρχονται χωρίς την χρήση ενδιάμεσης μνήμης ή με καθυστερήσεις, η εξυπηρέτηση πρέπει να είναι όσο το δυνατόν άμεση.**

Σκοπός Λειτουργικού Συστήματος

* **Βασικός σκοπός ενός λειτουργικού συστήματος είναι η διευκόλυνση του χρήστη στην επικοινωνία του με τον υπολογιστή. Η διευκόλυνση αυτή επιτυγχάνεται με την δημιουργία ενός περιβάλλοντος επικοινωνίας – φλοιού (shell). Το περιβάλλον αυτό μεσολαβεί ανάμεσα στον χρήστη και τον υπολογιστή και απαλλάσσει τον πρώτο από την ανάγκη να γνωρίζει το υλικό που διαθέτει ο δεύτερος.**

* **Ο δεύτερος σκοπός ενός λειτουργικού συστήματος είναι η αξιόπιστη και αποδοτική λειτουργία του συστήματος του υπολογιστή. Το λειτουργικό σύστημα αναλαμβάνει την διανομή των πόρων ώστε να γίνεται όσο το δυνατόν καλύτερη αξιοποίηση τους.**

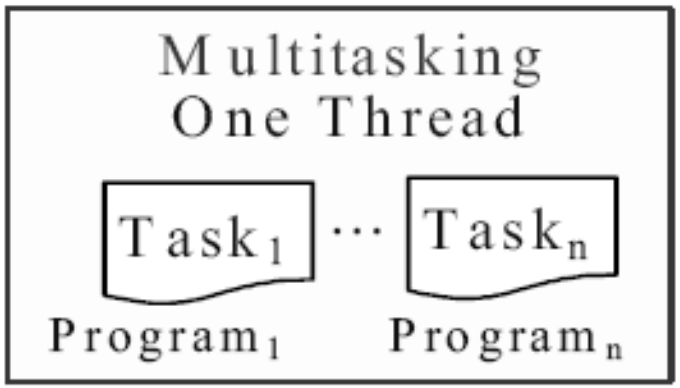
* **Τα λειτουργικά συστήματα είναι γραμμένα είτε σε χαμηλές γλώσσες προγραμματισμού (παλαιότερα) είτε σε σύγχρονες όπως πχ C, C++.**

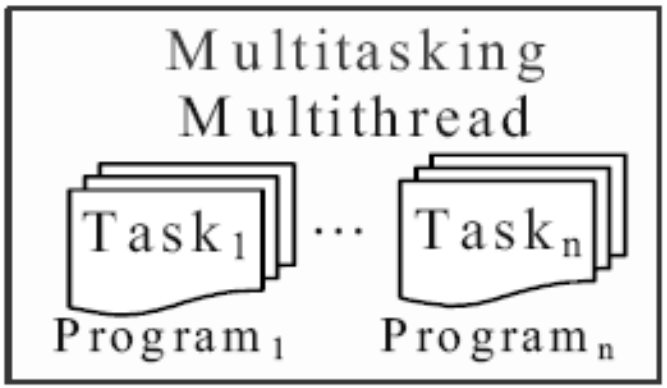
Εργασίες Λειτουργικού Συστήματος

* **Διαχείριση διεργασιών (χρονοπρογραμματιστής – scheduler)**
* **Διαχείριση μνήμης**
* **Διαχείριση Αρχείων**
* **Διαχείριση Συσκευών Εισόδου Εξόδου**

Multi-Tasking / Multi-Threading

* **Τα λειτουργικά συστήματα υποστηρίζουν δυνατότητες multi tasking δηλαδή επιτρέπουν την εκτέλεση πολλαπλών διεργασιών, αλλά και multi threading δηλαδή πολλαπλά νήματα για κάθε διεργασία που τρέχει. (Τεχνικές Πολυδιεργασίας)**





Δομή Λειτουργικού Συστήματος

* **Το λειτουργικό σύστημα αποτελείται από:**
  * **1. Πυρήνα**
    * Φορτώνεται πρώτο στην κύρια μνήμη και εκτελείται κατά την διάρκεια της λειτουργίας του υλικού. Τα προγράμματα επικοινωνούν με αυτόν μέσω των κλήσεων συστήματος.
  * **2. Σύστημα Αρχείων (File System):**
    * Διαχειρίζεται αρχεία, ονοματολογία – καταχώρηση ανάκτηση (NTFS, FAT32, EXT3)
  * **3. Διερμηνευτεί Εντολών (Command Interpreter) ή Φλοιό (Shell)**
    * Είναι το τμήμα που αναλαμβάνει να δέχεται και να δίνει στο σύστημα τις απαιτήσεις – εντολές χρήστη καθώς επίσης και να μεταφέρει μηνύματα στο χρήστη από το σύστημα.

Λειτουργικά Συστήματα Πραγματικού Χρόνου

* **Τα λειτουργικού συστήματα πραγματικού χρόνου είναι ειδικού σκοπού και χρησιμοποιούνται όταν υπάρχουν αυστηρές χρονικές απαιτήσεις για την ολοκλήρωση μιας λειτουργίας του υπολογιστή.**

* **Τα συστήματα αυτά χρησιμοποιούνται συχνά για τον έλεγχο άλλων μονάδων όπως βιομηχανικών μονάδων παραγωγής.**

* **Η επεξεργασία γίνεται μέσα από χρονικούς περιορισμούς, ένα σύστημα λειτουργεί σωστά μόνο όταν επιστρέφει σωστό αποτέλεσμα μέσα σε αυτούς αυστηρά του χρονικούς περιορισμούς που έχουν προκαθοριστεί.**

Διαφορές σε σχέση με τα παραδοσιακά Λειτουργικά Συστήματα

* **Τα παραδοσιακά λειτουργικά συστήματα παρέχουν παρόμοιες υπηρεσίες με τα συστήματα πραγματικού χρόνου με την διαφορά ότι τα δεύτερα είναι περισσότερο ντετερμινιστικά με την έννοια ότι οι υπηρεσίες τους καταναλώνουν μόνο προκαθορισμένο χρόνο και μετά το πέρας του χρόνου αυτού έχουμε σφάλμα.**

* **Τα λειτουργικά συστήματα μη πραγματικού χρόνου είναι κυρίως μη ντετερμινιστικά, δίνουν με βάση προτεραιότητες τυχαίο χρόνο στις διεργασίες με αποτέλεσμα να υπάρχουν και τυχαίες καθυστερήσεις ανάμεσα στις διεργασίες.**

Παραδείγματα RTOS

* **1. ATM**
* **2. Vending Machines**
* **3. Kiosks**
* **4. Embedded Systems**

Απαιτήσεις Συστημάτων Πραγματικού Χρόνου

* **Για να είναι ένα λειτουργικό σύστημα πραγματικού χρόνου πρέπει να ικανοποιεί τις παρακάτω απαιτήσεις:**

* **1. Διεργασίες με μεγαλύτερη προτεραιότητα πάντα εκτελούνται πρώτα σε σχέση με άλλες με χαμηλότερη προτεραιότητα.**

* **2. Το λειτουργικό σύστημα πρέπει να υποστηρίζει fixed – priority preemptive χρονοπρογραμματισμό τόσο για τα νήματα όσο και για τις διεργασίες.**

* **3. Το λειτουργικό σύστημα πρέπει να παρέχει κληρονομικότητα στην προτεραιότητα.**

Απαιτήσεις Συστημάτων Πραγματικού Χρόνου

* **Για να είναι ένα λειτουργικό σύστημα πραγματικού χρόνου πρέπει να ικανοποιεί τις παρακάτω απαιτήσεις:**

* **4. Ο πυρήνας πρέπει να είναι προεκτοπιστικός (preemptive). Αυτό έρχεται σε αντίθεση με τον πυρήνα των μη πραγματικού χρόνου που είναι non preemptive. Αυτό σημαίνει πως μία αίτηση κλήση από τον πυρήνα μπορεί να υπερβεί όλες τις άλλες διεργασίες – νήματα.**

* **5. Τα interrupts πρέπει να έχουν ένα προκαθορισμένο άνω όριο στην καθυστέρηση.**

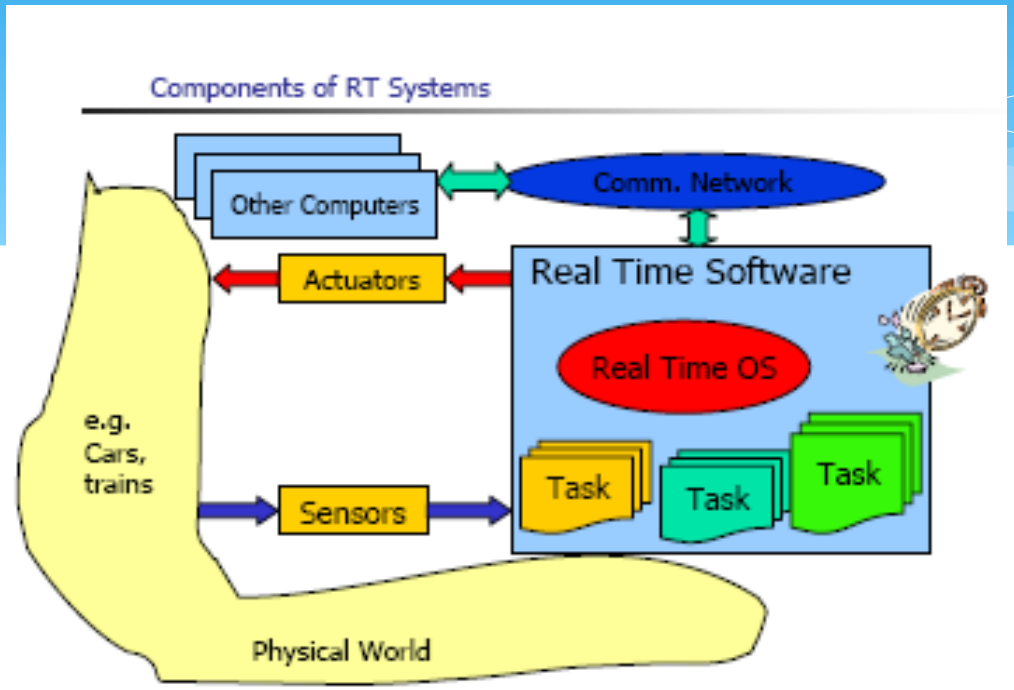
Χαρακτηριστικά Λειτουργικών Συστημάτων Πραγματικού Χρόνου

* **1. Προβλεψιμότητα (predictability)**
* **2. Καθοριστικότητα (determinism)**
* **3. Δυνατότητα χρονοπρογραμματισμού των διεργασιών με βάση προτεραιότητες (preemptive scheduling based on priorities)**
* **4. Γρήγορη μετάβαση σε διαδικασίες (content switching)**

Χαρακτηριστικά Λειτουργικών Συστημάτων Πραγματικού Χρόνου

* **5. Μικρό μέγεθος**
* **6. Επικοινωνία και συγχρονισμό ανάμεσα σε διεργασίες**
* **7. Ύπαρξη χρονιστών πραγματικού χρόνου.**
* **8. Δεν υπάρχουν καθυστερήσεις (no latencies) καθώς όλες οι διεργασίες – νήματα έχουν προκαθορισμένο χρόνο εκτέλεσης και εκτελούνται πάντα μέσα σε αυτό το χρόνο.**

Μέρη Συστήματος Πραγματικού Χρόνου



* **Παρατηρούμε το λειτουργικό σύστημα πραγματικού χρόνου ανάμεσα στο υλικό και το υπόλοιπο λογισμικό.**

Μέρη Λ.Σ Π.Χ.

* **Διακρίνουμε το υλικό μέρος που αποτελείται από τους πόρους δηλαδή την Κ.Μ.Ε, μνήμη, χρονιστές, περιφερειακές μονάδες I/O, ενεργοποιητές, κανάλια επικοινωνίας.**

* **Το λειτουργικό σύστημα πραγματικού χρόνου ή απλά έναν πυρήνα πραγματικού χρόνου (Real Time Kernel), με προβλέψιμη συμπεριφορά και καθορισμένη λειτουργικότητα.**

* **Ένα σύνολο από διεργασίες (task) πραγματικού χρόνου. Οι «διεργασίες» είναι ρουτίνες λογισμικού, γραμμένες σε κάποια γλώσσα πραγματικού χρόνου ή είναι μηχανισμοί που δημιουργεί το λειτουργικό σύστημα, για να διαχειρίζεται την εκτέλεση τμημάτων κώδικα σε ένα πρόγραμμα. Οι διεργασίες μοιράζονται τους πόρους του συστήματος, επικοινωνούν και συγχρονίζονται μεταξύ τους και με το περιβάλλον.**

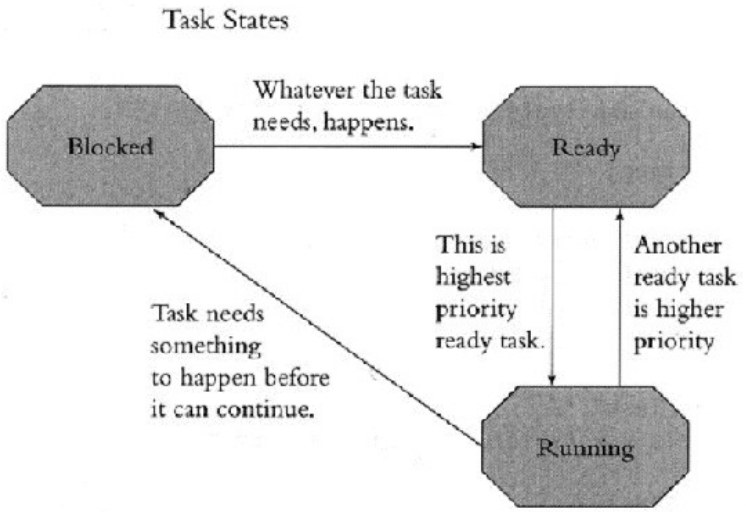
Διεργασίες Λειτουργικών Συστημάτων Πραγματικού Χρόνου

* **Ένα λειτουργικό σύστημα πραγματικού χρόνου διαχειρίζεται τα διάφορα προγράμματα χωρίζοντάς τα σε νήματα διεργασιών, τις οποίες και ελέγχει κατάλληλα. Μια διεργασία είναι μια βασική μονάδα προγραμματιστικού έργου, μια οντότητα λογισμικού, που δημιουργείται από το λειτουργικό σύστημα προκειμένου να διαχειρίζεται την εκτέλεση ενός τμήματος κώδικα. Κάθε διεργασία είναι ανεξάρτητη από τις άλλες διεργασίες και μπορεί να συγχρονιστεί και να επικοινωνήσει μαζί τους.**

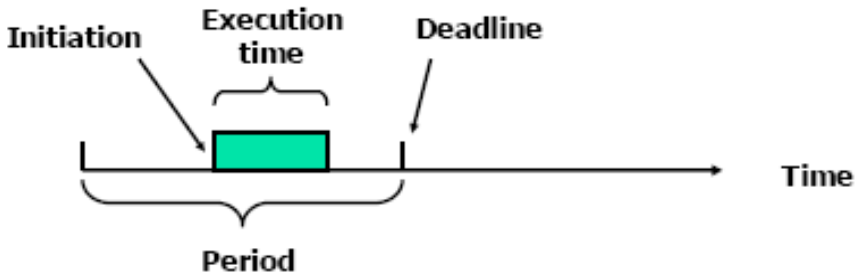
* **Αφού δημιουργηθεί μια διεργασία, έχει μια διάρκεια ζωής, κατά την οποία μπορεί να βρεθεί σε διάφορες «καταστάσεις», ανάλογα με το ποιες καταστάσεις διεργασιών υποστηρίζει το λειτουργικό σύστημα. Μπορεί, λοιπόν, να δημιουργείται, να βρίσκεται σε ανενεργή κατάσταση (sleeping) ή σε κατάσταση ετοιμότητας (ready), να εκτελείται και να τερματίζεται.**

Κατάσταση Διεργασιών Π.Χ.

* **Running: Η διεργασία εκτελείται.**
* **Blocked: Η διεργασία έχει μπλοκαριστεί.**
* **Ready: Η διεργασία είναι έτοιμη για εκτέλεση.**

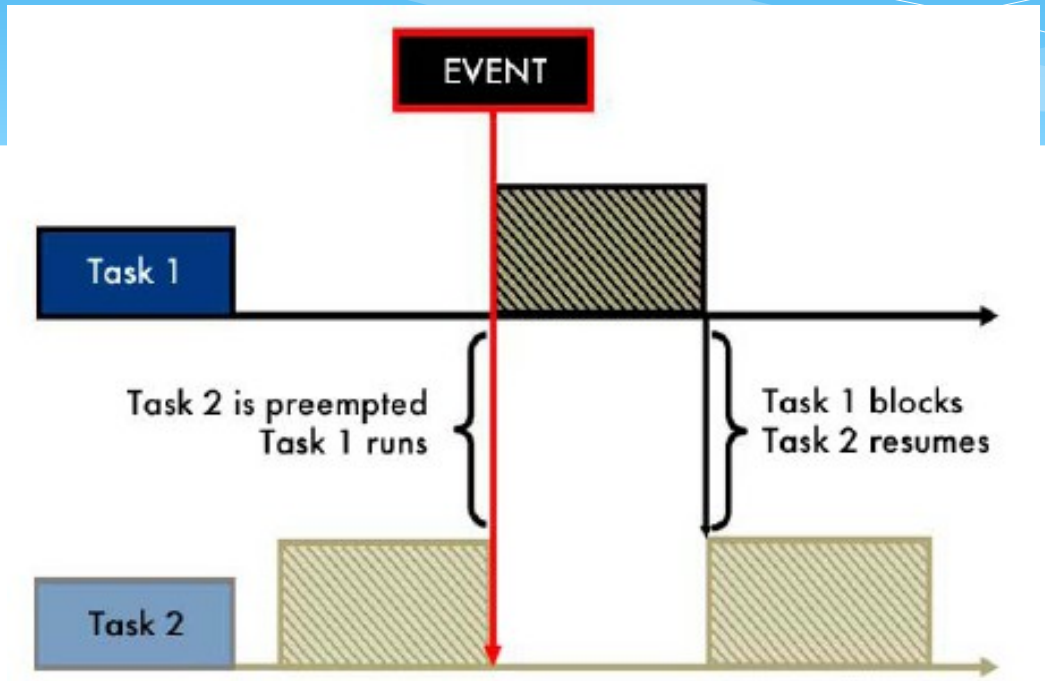


Χρονικές Παράμετροι Διεργασίας



Στην αρχή της περιόδου έχουμε τον χρόνο αποδέσμευσης, στη συνέχεια η διεργασία είναι έτοιμη (initiation) και εκτελείται. Το RTOS την προγραμματίζει να εκτελεστεί μέσα στα όρια της προθεσμίας της (deadline).

Διακοπή Διεργασιών



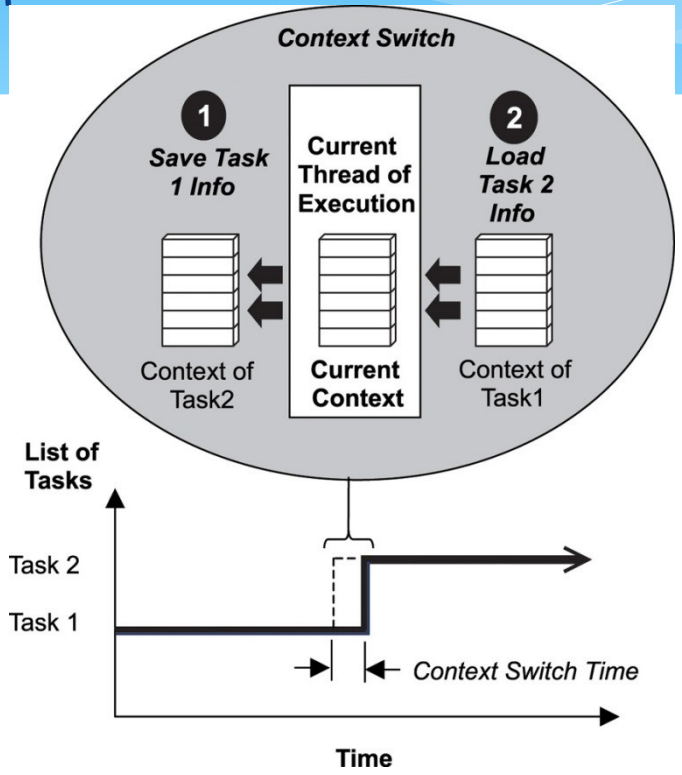
Απαιτήσεις Πολυδιεργασίας σε RTOS

* **1. Μετάβαση Περιεχομένου (content switching): Κάθε διεργασία απαιτεί ένα σύνολο από καταχωρητές που σχετίζονται με την εκτέλεση κώδικα.**
  * **Καταχωρητές εργασίας**
  * **Απαριθμητής προγράμματος**
  * **Καταχωρητής κατάστασης**
  * **Δείκτης στοίβας**
* **Οι τιμές αυτών των καταχωρητών είναι κρίσιμες όσο εκτελείται η διεργασία.**

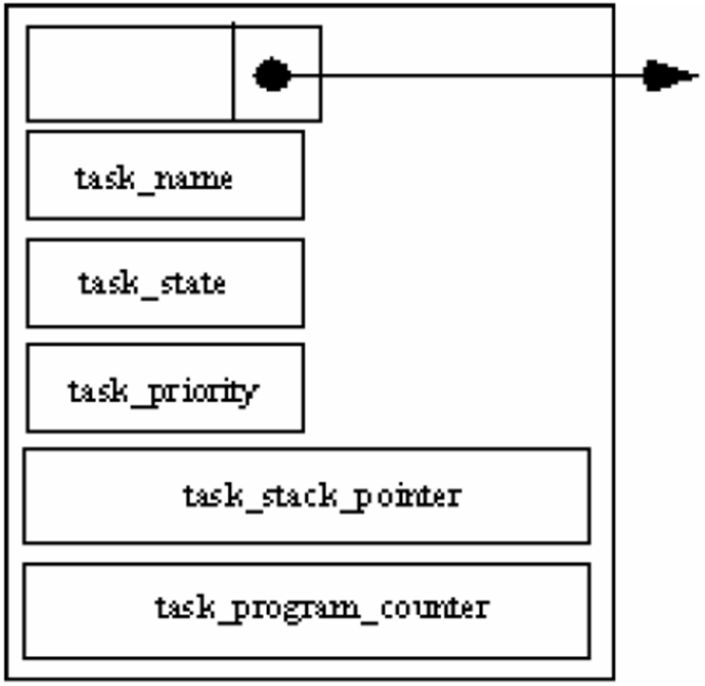
* **Όλες αυτές οι πληροφορίες αποτελούν το «περιεχόμενο» (context) της διεργασίας. Όταν σε ένα σύστημα πολυδιεργασίας η εκτέλεση του προγράμματος μεταβαίνει από τη μια διεργασία στην άλλη, το περιεχόμενο πρέπει να αποθηκεύεται κατάλληλα, ώστε να μπορεί να ξαναφορτώνεται στους κατάλληλους καταχωρητές, όταν επαναλαμβάνεται η εκτέλεση της διεργασίας.**

Απαιτήσεις Πολυδιεργασίας σε RTOS

* **Κάθε φορά που διακόπτεται μία διεργασία για να γίνει μία μετάβαση σε μία άλλη πρέπει όλες αυτές οι πληροφορίες να αποθηκεύονται σε μπλοκ μνήμης που έχει η κάθε διεργασία.**



Μπλοκ μνήμης Context Switch



- Task uses TCB to remember its *context*
- RTOS updates TCB when task is *switched*
- RTOS uses TCBs to control tasks

Μπλοκ μνήμης Context Switch

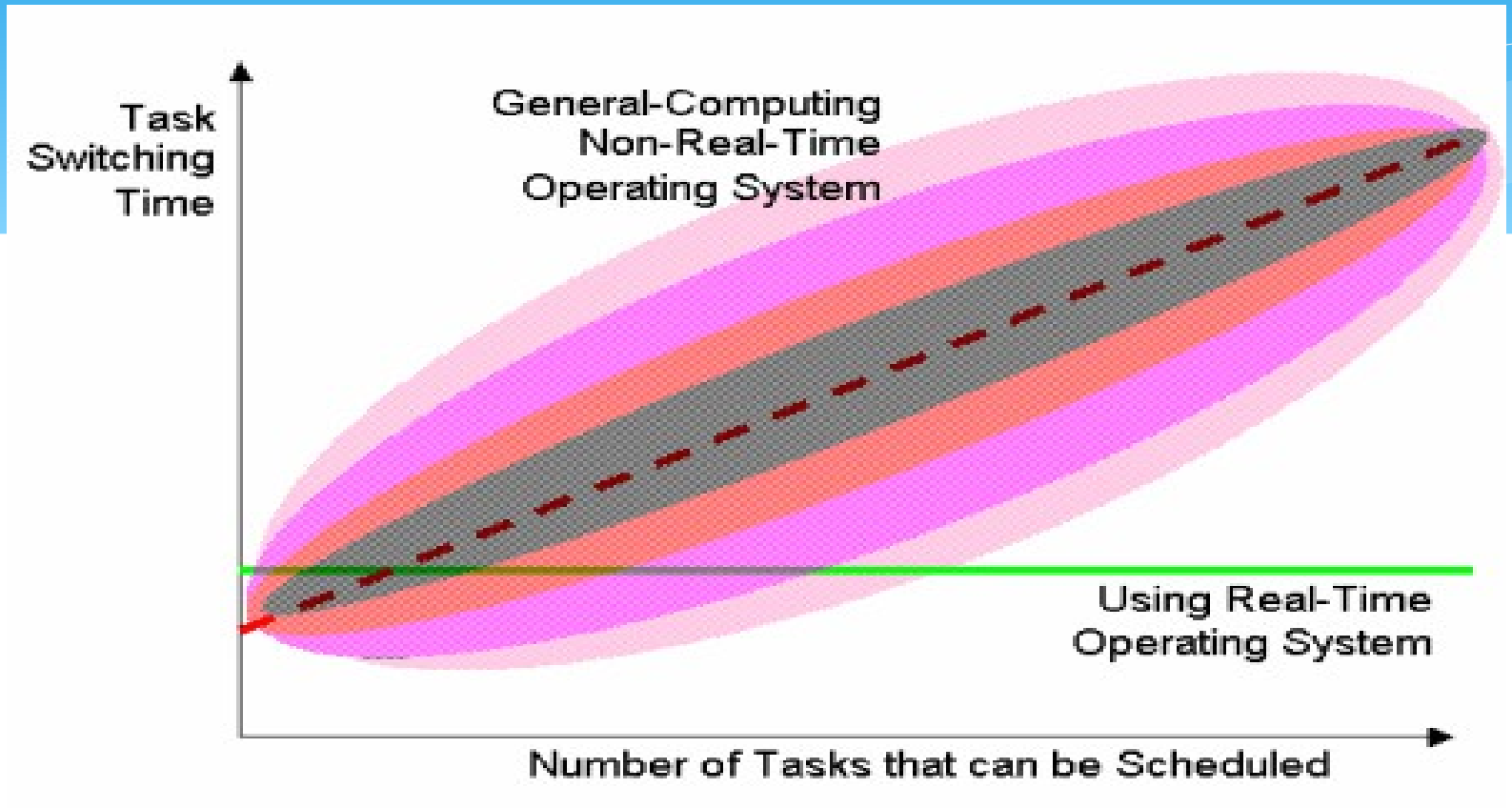
* **Στα λειτουργικά συστήματα γενικού σκοπού ο χρονοπρογραμματιστής ψάχνει στο TCB για να βρει την επόμενη εργασία. Όπως είναι εύλογο, όσο μεγαλύτερος είναι αυτός ο πίνακας, δηλαδή όσο περισσότερες εργασίες βρίσκονται σε αναμονή τόσο περισσότερο χρόνο θα κάνει ο χρονοπρογραμματιστής να βρει τη σωστή και να γίνει η εναλλαγή κάτι το οποίο είναι ενοχλητικό.**

* **Στα συστήματα πραγματικού χρόνου η εξάρτηση του χρόνου εναλλαγής από το πλήθος των εργασιών που βρίσκονται σε αναμονή, δε συνάδει καθόλου με το ντετερμινισμό που έχουν ως απαίτηση.**

* **Πρέπει να γνωρίζουμε εκ των προτέρων πόσο μεγάλη θα είναι η καθυστέρηση.**

* **Αυξητικά ενημερωμένοι πίνακες TCB που επιτρέπουν στον προγραμματιστή να αναγνωρίσει την επόμενη διεργασία ταχύτατα σε σταθερό χρόνο.**

Real Time OS VS General Purpose OS



Απαιτήσεις Πολυδιεργασίας σε RTOS

* **2. Επικοινωνία ανάμεσα σε διεργασίες (task communication): Σε μία διεργασία τα αποτελέσματα που παράγονται από ένα τμήμα κώδικα μπορεί να χρησιμοποιούνται από και να μεταβιβάζονται σε άλλο τμήμα κώδικα με μία φυσική ροή αποτελεσμάτων σε καταχωρητές ή μεταβλητές.**

* **Οι αλληλουχία των διεργασιών καθορίζεται από το λειτουργικό σύστημα και δεν είναι απαραίτητο να διαδέχεται η μία την άλλη ώστε η έξοδος της μία να γίνει είσοδο στην άλλη.**

* **Απαιτείται ένα σύστημα επικοινωνίας ανάμεσα στις διεργασίες που να καθορίζει την διαδικασία της επικοινωνίας.**

Απαιτήσεις Πολυδιεργασίας σε RTOS

* **3. Διαχείριση Προτεραιοτήτων:**
  * **Σε κάθε σύστημα πραγματικού χρόνου κεντρικό ρόλο παίζει η απόδοση προτεραιοτήτων στις διάφορες διεργασίες. Με βάση τις σχετικές προτεραιότητες των διεργασιών λαμβάνεται η απόφαση για το ποιά διεργασία είναι πιο σημαντικό να εκτελεστεί κάθε στιγμή.**
* **Η απόδοση προτεραιοτήτων γίνεται με τη βοήθεια ειδικών αλγορίθμων, ειδικά στα μεγάλα συστήματα.**
* **Ο προγραμματιστής θέτει τις προτεραιότητες με τέτοιον τρόπο ώστε όταν μία διεργασία χρειάζεται 100% της προτεραιότητας, καμία άλλη δεν μπορεί να πάρει σειρά παρά μόνο όταν τελειώσει η αρχική διεργασία.**

Απαιτήσεις Πολυδιεργασίας σε RTOS

* **4. Έλεγχος χρονισμού: Κάθε σύστημα πραγματικού χρόνου πρέπει να διαθέτει ένα σύστημα που να καθορίζει κάθε πότε θα εκτελείται η κάθε διεργασία.**

* **Ο χρονισμός των διεργασιών στηρίζεται σε χρονιστές, που πρέπει να προγραμματίζονται κατάλληλα.**

Τεχνικές Πολυδιεργασίας

* **Συνεργαζόμενη πολυδιεργασία**

  Με την συνεργαζόμενη πολυδιεργασία (cooperative multitasking) μια διεργασία παραδίδει τον έλεγχο σε άλλη μόνον «εθελοντικά». Για να χάσει η διεργασία τον έλεγχο πρέπει να καλέσει η ίδια το λειτουργικό σύστημα (RTOS). Με τον τρόπο αυτό οι διεργασίες μοιράζονται δίκαια τον επεξεργαστή και τους άλλους πόρους.

* **Ολοκληρωμένη εκτέλεση**

  Μια διεργασία παραδίδει τον έλεγχο μόνον όταν έχει ολοκληρώσει το έργο της (run to completion). Από πρακτική άποψη αυτή η τεχνική μπορεί να χρησιμοποιηθεί σε ένα RTOS μόνον όταν όλες οι διεργασίες είναι σχετικά σύντομες.

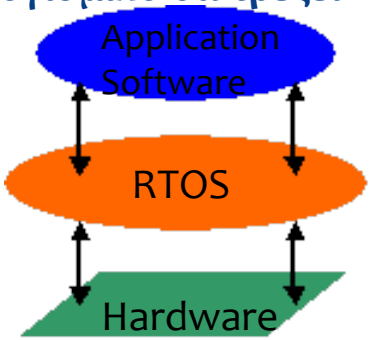
* **Διακοπή από τον προγραμματιστή διεργασιών (preemption)**

  Στην περίπτωση αυτή μια διεργασία παραδίδει τον έλεγχο όποτε υπαγορεύει ο χρονοπρογραμματιστής (scheduler). Η τεχνική αναφέρεται ως preemption. Γενικά, οι schedulers που στηρίζονται σ' αυτή την τεχνική μπορούν να ανταποκριθούν σε συγκεκριμένες χρονικές απαιτήσεις καλύτερα από άλλους.

Πυρήνας RTOS

* **Το κέντρο ενός λειτουργικού συστήματος ονομάζεται πυρήνας (kernel). Στα λειτουργικά συστήματα πραγματικού χρόνου ο πυρήνας αποτελείται από τον χρονοπρογραμματιστή διεργασιών (scheduler), τον διαχειριστή διακοπών (interrupt handler), και τον διαχειριστή πόρων (resource manager), που διαχειρίζεται τη μνήμη και τον διαμοιρασμό της στις διάφορες διεργασίες.**

* **Ο πυρήνας των λειτουργικών συστημάτων πραγματικού χρόνου παρέχει ένα αφηρημένο επίπεδο (abstraction layer) που διαχωρίζει – κρύβει το application software από τις τεχνικές λεπτομέρειες του υλικού για παράδειγμα του επεξεργαστή στον οποίο το λογισμικό θα τρέξει.**



Πυρήνας RTOS

* **Ο πυρήνας των λειτουργικών συστημάτων πραγματικού χρόνου παίρνει τον έλεγχο ώστε να:**

  * **1. Αποκριθεί σε μία κλήση συστήματος (system call)**
  * **2. Να κάνει χρονισμό και να εξυπηρετήσει τους χρονιστές.**
  * **3. Να χειριστεί εξωτερικές διακοπές.**
  * **4. Να ασχοληθεί με την ανάνηψη από εξαιρέσεις υλικού και λογισμικού.**

Interrupts RTOS

* **Ασύγχρονα: Είναι οι διακοπές που γίνονται από το υλικό από ένα συμβάν που μπορεί να συμβεί στο υλικό.**
* **Σύγχρονα: Είναι διακοπές που γίνονται από το λογισμικό, πχ διαίρεση με το μηδέν, segmentation fault κτλ.**
* **Προκλήσεις στα RTOS:**
  * **Interrupt Latency**
  * **Interrupt Enable / Disable**
  * **Interrupt Priority**

Χειρισμός Interrupts RTOS

* **Το πρώτο βήμα του χειρισμού διακοπών (άμεση εξυπηρέτηση διακοπής)εκτελείται ανάλογα με το επίπεδο προτεραιότητας διακοπής. Όλες οι προτεραιότητες διακοπών είναι μεγαλύτερες από αυτές των νημάτων.**

* **Τα RTOS υλοποιούν αποκλειστικά και μόνο διανυσματικές διακοπές.**

* **Δίνουν στις εφαρμογές την δυνατότητα να ελέγχουν τον χρόνο που θα επανεργοποιηθούν οι διακοπές ελέγχοντας με αυτό τον τρόπο τον ρυθμό με τον οποίο εξυπηρετούνται οι εξωτερικές διακοπές και το χρόνο που ξοδεύεται στο χειρισμό διακοπών.**

Χρονικοί Περιορισμοί - Constraints

* **Διαχωρίζονται σε τρία είδη περιορισμών ανάλογα με την απόδοση και την συμπεριφορά του συστήματος. Οι χρονικοί περιορισμοί που σχετίζονται με την συμπεριφορά διασφαλίζουν ότι το σύστημα συμπεριφέρεται σωστά ενώ αυτοί που σχετίζονται με την απόδοση διασφαλίζουν ότι το σύστημα λειτουργεί αποδοτικά.**

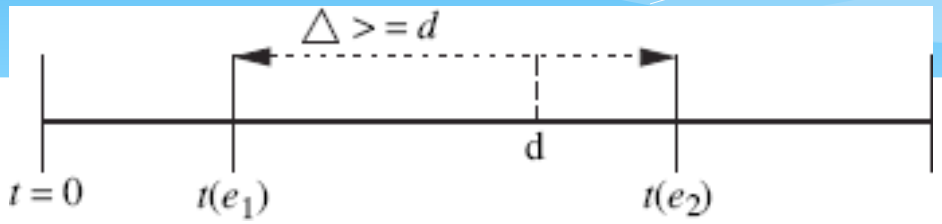
* **1. Delay Constraints (Περιορισμός Καθυστέρησης).**

* **2. Deadline Constraints (Περιορισμοί Αδιεξόδου).**

* **3. Duration Constraints (Περιορισμοί Διάρκειας).**

Χρονικοί Περιορισμοί - Constraints

* **Περιορισμός Καθυστέρησης ανάμεσα σε δύο συμβάντα e1, e2.**

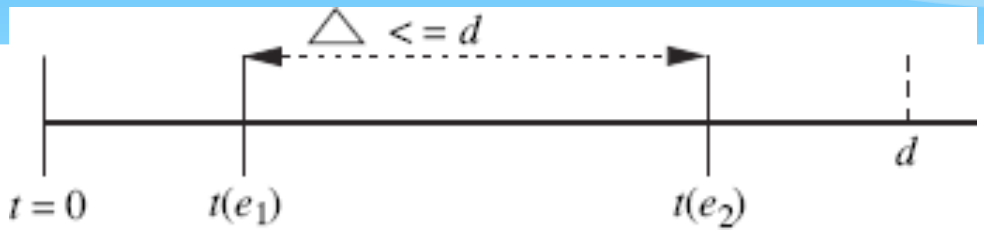


$$t(e_2) - t(e_1) \geq d$$

* **Ο περιορισμός καθυστέρησης πιάνει την ελάχιστη καθυστέρηση που μπορεί να συμβεί ανάμεσα σε δύο διαφορετικά συμβάντα e1, e2. Αφού συμβεί το e1, αν το e2 συμβεί νωρίτερα από την ελάχιστη αυτή καθυστέρηση τότε μία παράβαση καθυστέρησης έχει συμβεί.**

Χρονικοί Περιορισμοί - Constraints

* **Περιορισμός Αδιεξόδου συμβαίνει ανάμεσα σε δύο συμβάντα e1, e2.**



$$t(e_2) - t(e_1) \leq d$$

* **Σε περίπτωση που η διαφορά ανάμεσα σε δύο συμβάντα e1, e2 είναι μικρότερη από τον περιορισμό d τότε έχουμε αδιέξοδο.**

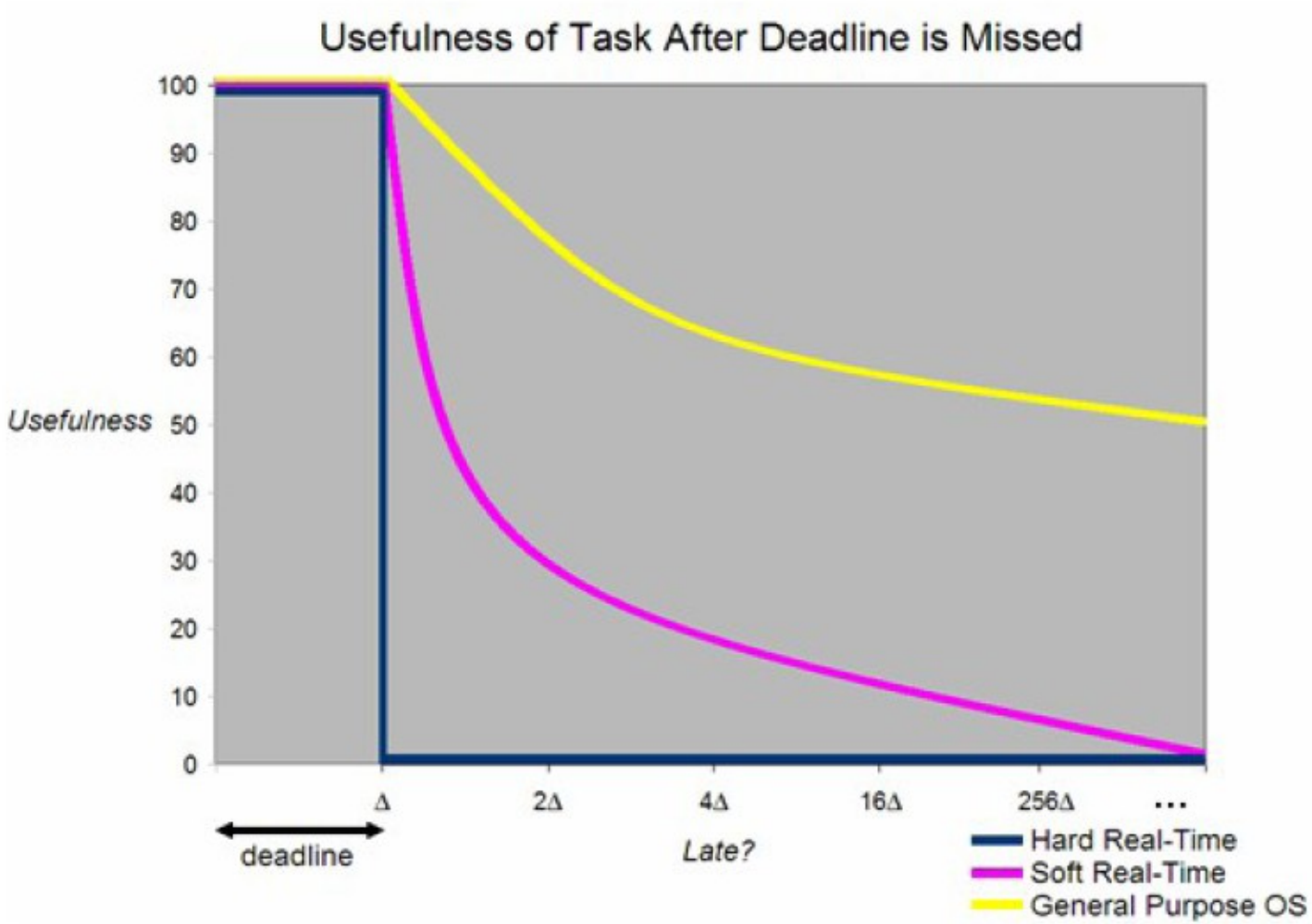
Αυστηρά και Χαλαρά Συστήματα Πραγματικού Χρόνου

* **Τα Λειτουργικά Συστήματα Πραγματικού χρόνου διακρίνονται σε soft real time operating systems και σε hard real time operating systems.**

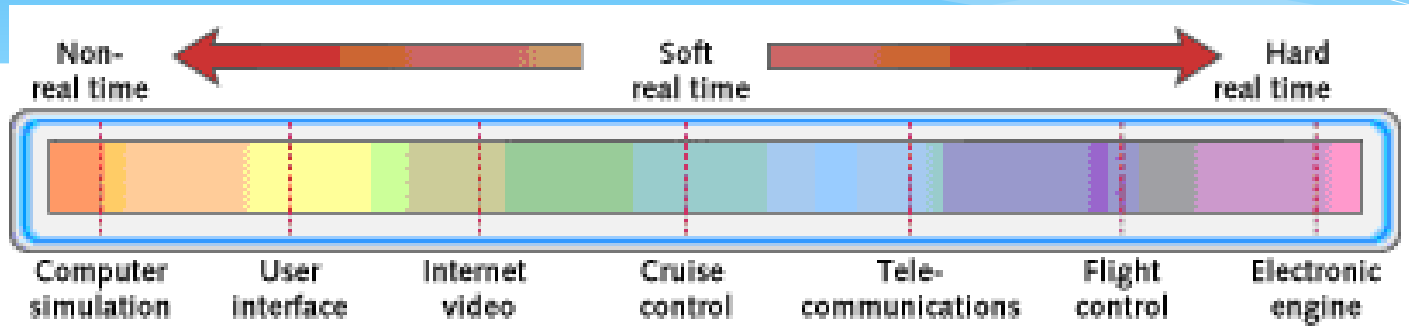
* **Τα αυστηρά συστήματα (hard) δεν επιτρέπουν έναν αργοπορημένο υπολογισμό καθώς αυτό θα αποτελέσει καταστροφικό σενάριο για το λειτουργικό σύστημα. Πολύ απλά είναι ένα σύστημα στο οποίο όλες οι δραστηριότητες πρέπει να ολοκληρώνονται εγκαίρως.**

* **Τα χαλαρά συστήματα (soft) κάνουν ότι κάνουν και τα αυστηρά με την διαφορά ότι παρέχουν δυνατότητες υπολογισμό και παρακολούθησης ακριβούς κόστους των διεργασιών του συστήματος. Πρέπει να αναφέρει ποιες διεργασίες δεν κατάφεραν να ολοκληρώσουν ή πότε υπερβαίνουν την χρονική χρήση του επεξεργαστή και να στέλνει τα κατάλληλα σήματα για τον σκοπό αυτό.**

Αυστηρά και Χαλαρά Συστήματα Πραγματικού Χρόνου



Φάσμα Εφαρμογών Πραγματικού Χρόνου



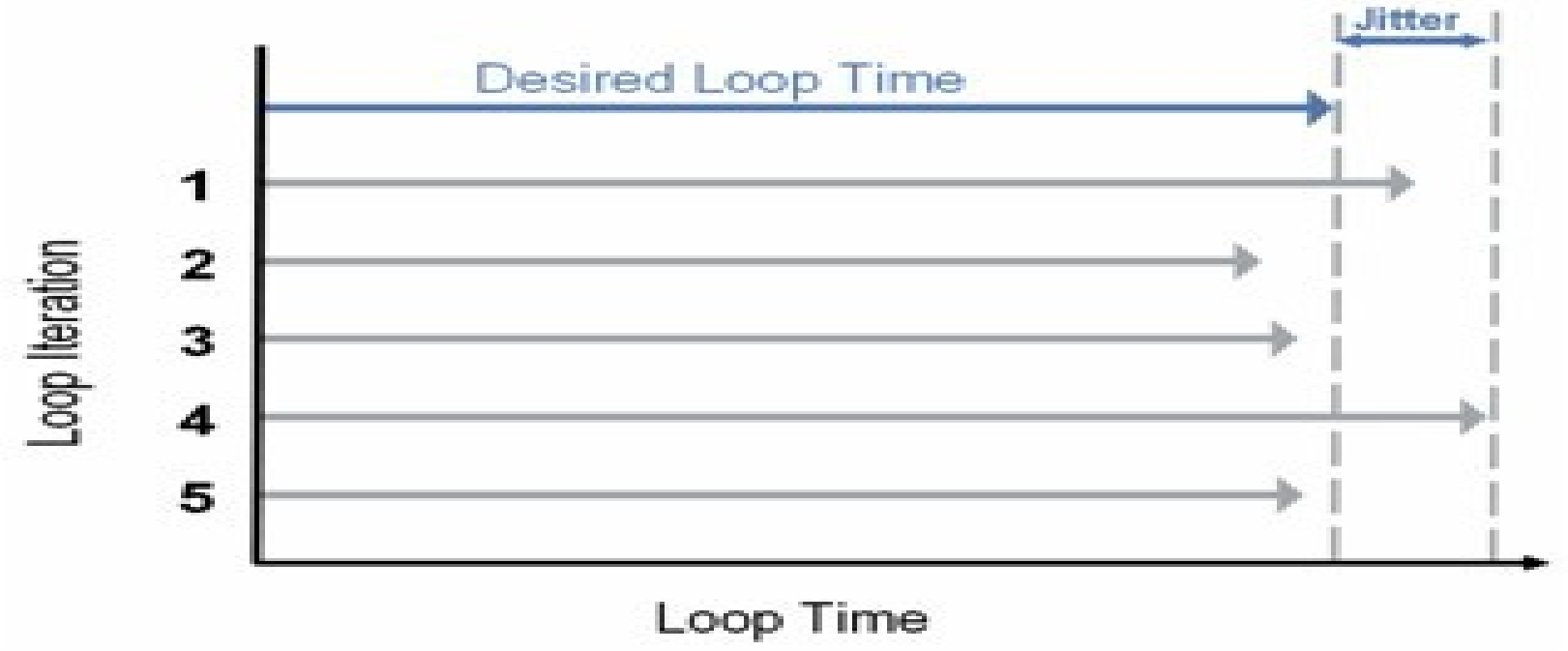
Χρονικές Διαταραχές - Jitter

* **Ο χρόνος εκτέλεσης μια διεργασίας μεταβάλλεται με βάση τον υπολογιστικό φόρτο κάποιου έργου.**

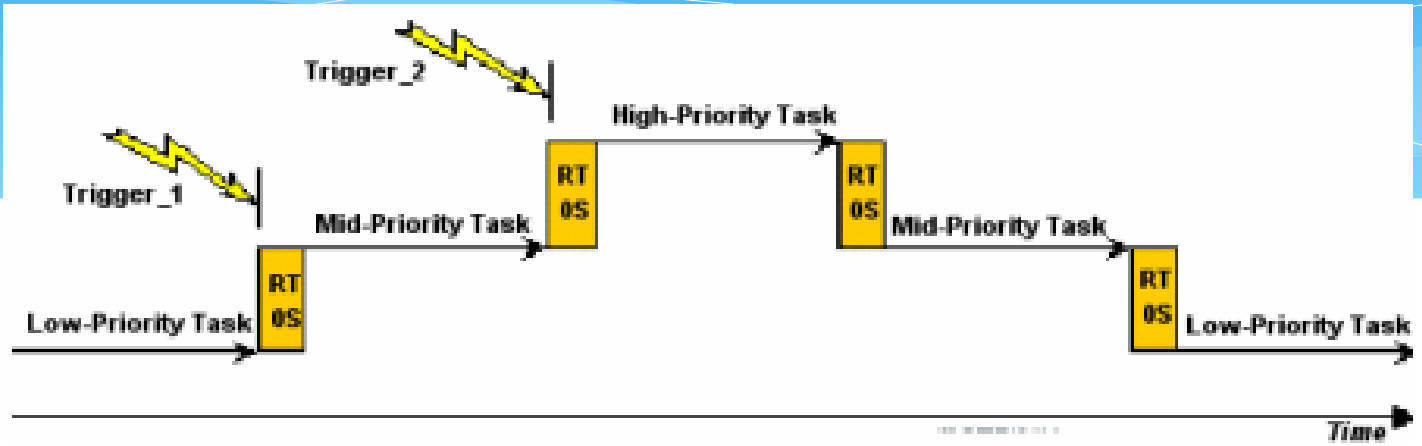
* **Υπάρχουν περιπτώσεις που μεταβάλλεται και ο χρόνος αποδέσμευσης οπότε έχουμε διαταραχή που ονομάζεται Jitter.**

* **Μη σταθερός χρόνος αποδέσμευσης: χρόνος ολοκλήρωσης ενός έργου να μην είναι πάντα σταθερός, αλλά να μεταβάλλεται μέσα σε κάποια όρια**

Χρονικές Διαταραχές - Jitter



Χρονοδρομολόγηση Πραγματικού Χρόνου



Η εργασία με την υψηλότερη προτεραιότητα που πρέπει να τρέχει, θα είναι και αυτή που θα τρέχει. Έτσι αν μια εργασία προτεραιότητας 1 τρέχει και πριν αυτή ολοκληρωθεί, ζητήσει τη χορήγηση της CPU μια εργασία προτεραιότητας 5, η πρώτη εργασία θα παρέχει ευγενικά τη σειρά της. Αν ενώ τρέχει και η δεύτερη εργασία, ζητήσει τη CPU μια τρίτη προτεραιότητας 10, η δεύτερη θα παραχωρήσει επίσης τη θέση της.

Χρονοδρομολόγηση Πραγματικού Χρόνου

* **Στατική ή δυναμική εκχώρηση προτεραιοτήτων.**
* **Λιμοκτονία μέχρι θανάτου – πρόβλημα starvation όπου διεργασίες με χαμηλή προτεραιότητα είναι αναγκασμένες να αργήσουν υπερβολικά αν παρουσιαστούν στο σύστημα διεργασίες με υψηλή προτεραιότητα.**
* **Αναίτια καταλαμβάνεται τμήμα της μνήμης από διεργασίες χαμηλής προτεραιότητας που δεν εκτελούνται ποτέ.**
* **Λύση: Αλγόριθμοι που ελέγχουν αν μία εργασία αναμένει την χρήση του επεξεργαστή για μεγάλο χρονικό διάστημα.**
* **Παράδειγμα: Να ανατίθεται σαν νέα προτεραιότητα μιας εργασίας ο αριθμός 1/f, όπου f το κλάσμα του χρόνου που χρησιμοποιήθηκε από τη εργασία πριν αυτή μπλοκαριστεί, προς το χρόνο που της αναλογούσε**

Χρονοδρομολόγηση Πραγματικού Χρόνου

- **Αλγόριθμοι από το WikiPedia.**

- **Cooperative scheduling**
- **Preemptive scheduling**
  - **Rate-monotonic scheduling**
  - **Round-robin scheduling**
  - **Fixed priority pre-emptive scheduling, an implementation of preemptive time slicing**
  - **Fixed-Priority Scheduling with Deferred Preemption**
  - **Fixed-Priority Non-preemptive Scheduling**
  - **Critical section preemptive scheduling**
  - **Static time scheduling**
- **Earliest Deadline First approach**
- **Stochastic digraphs with multi-threaded graph traversal**

Χρονοδρομολόγηση Πραγματικού Χρόνου

* **Βασική διάκριση ανάμεσα στα λειτουργικά συστήματα και σε αυτά του πραγματικού χρόνου βρίσκεται στον διαχειριστή διεργασιών (task manager) ο οποίος αποτελείται από τα εξής μέρη:**

* **1. Αποστολέας (dispatcher)**
* **2. Δρομολογητή (scheduler)**

Χρονοδρομολόγηση Πραγματικού Χρόνου - Dispatcher

* **Ο αποστολέας (dispatcher) ελέγχει τη ροή του προγράμματος η οποία αποφασίζεται από τον scheduler.**

* **Αναλαμβάνει το context switch δηλαδή την μετάβαση από την μία διεργασία στην άλλη ή τη μετάβαση σε ρουτίνες εξυπηρέτησης διακοπής**

* **Αναλαμβάνει την αποθήκευση της κατάστασης της διεργασίας η οποία λαμβάνει χώρο λίγο πριν το context switch καθώς και την φόρτωση της νέας διεργασίας που θα αντικαταστήσει την υπάρχουσα ώστε να εκτελεστεί δηλαδή να αλλάξει κατάσταση.**

Χρονοδρομολόγηση Πραγματικού Χρόνου - Scheduler

* **Ο χρονοδρομολογητής (scheduler) είναι το μέρος του λειτουργικού συστήματος που αποφασίζει κάθε φορά ποια θα είναι η επόμενη διεργασία που θα εκτελεστεί από τον επεξεργαστή, ώστε το σύστημα να είναι προβλέψιμο.**

* **Η επιλογή που γίνεται από τον χρονοδρομολογητή βασίζεται σε ειδικούς αλγόριθμους και εξειδικευμένων τεχνικών δρομολόγησης.**

* **Υπάρχουν περιπτώσεις που ο χρονοδρομολογητής εκτελείται περιοδικά ως διακοπή που προκαλεί κάποιος χρονιστής.**

Μορφές Χρονοδρομολόγησης Πραγματικού Χρόνου

* **Διακρίνουμε δύο μορφές χρονοδρομολόγησης:**

* **Χωρίς διακοπή διεργασιών (non- preemptive)**

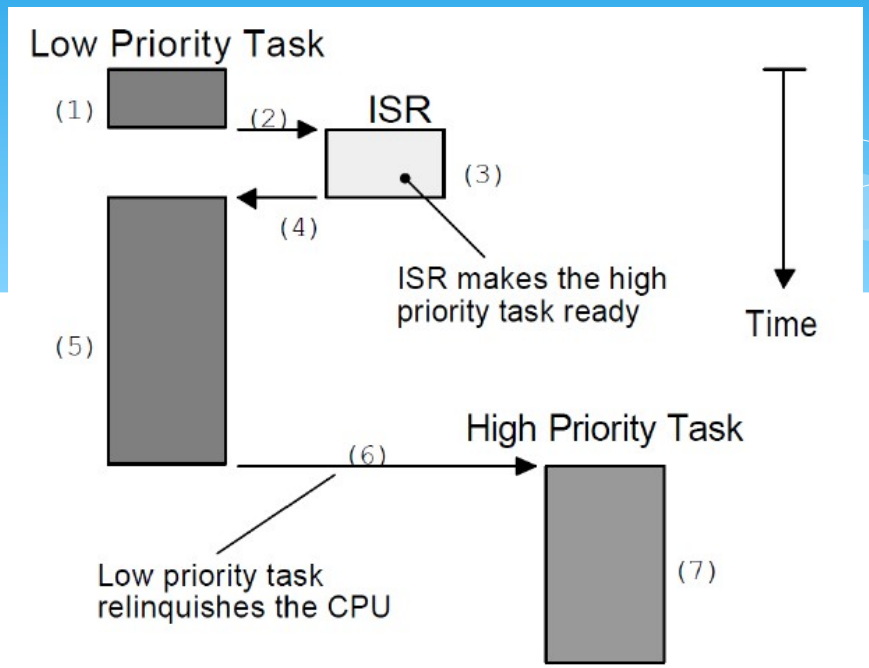
* **Με διακοπή διεργασιών (preemptive)**

Χωρίς διακοπή διεργασιών (non- preemptive)

* **Χρησιμοποιείται συνήθως σε εφαρμογές, όπου οι διεργασίες έχουν παρόμοιες απαιτήσεις για τη χρήση των πόρων του συστήματος και ο χρόνος εκτέλεσής τους είναι σχετικά μικρός.**

* **Επιτρέπει την ολοκλήρωση της τρέχουσας διεργασίας, πριν μεταβεί σε μια άλλη, υψηλότερης προτεραιότητας.**

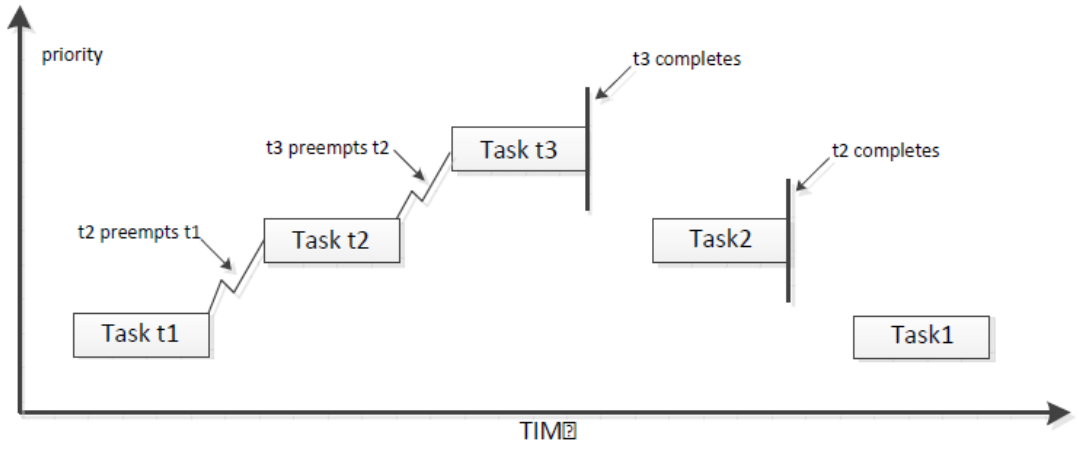
Χωρίς διακοπή διεργασιών (non- preemptive)



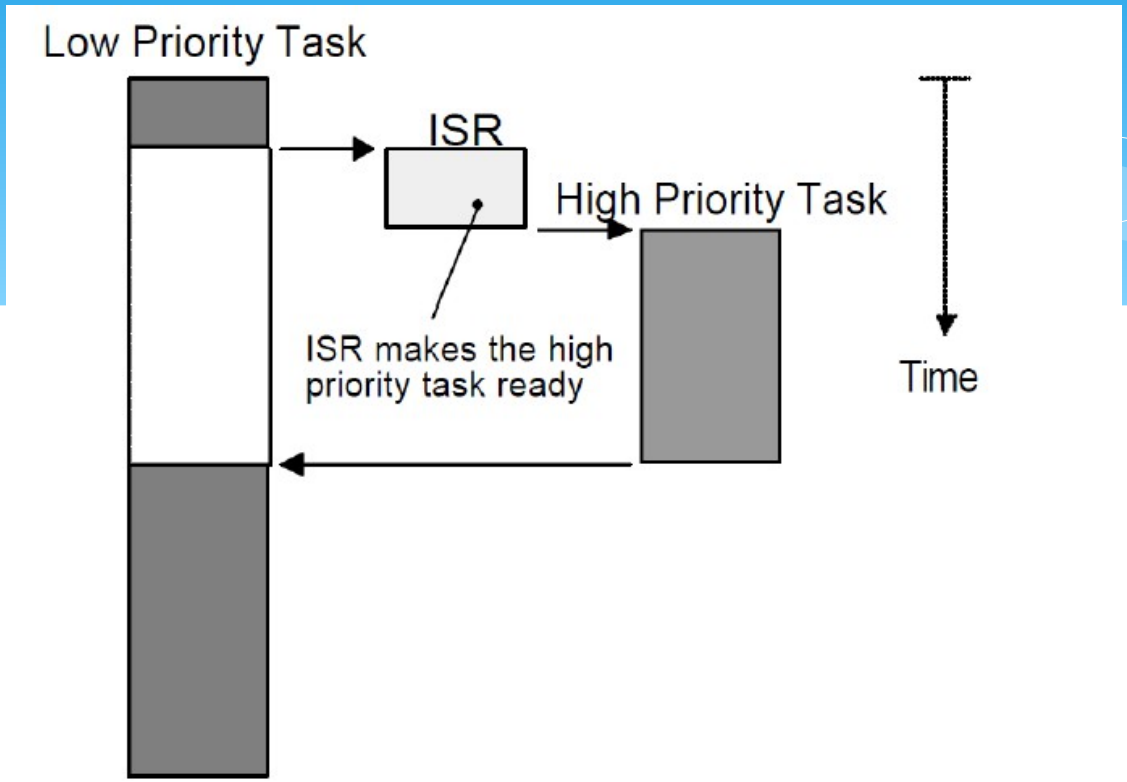
Καλείται ο scheduler και η υπορουτίνα ISR φέρνει σε κατάσταση ετοιμότητας την διεργασία υψηλότερης προτεραιότητας αλλά δεν την εκτελεί. Η διεργασία έρχεται σε κατάσταση εκτέλεσης μόνο με την ολοκλήρωση της τρέχουσας διεργασίας.

Με διακοπή διεργασιών (preemptive)

- **Διακοπή διεργασίας με χαμηλή κρισιμότητα – σπουδαιότητα και αντικατάσταση με άλλη υψηλότερης.**
- **Ο scheduler που υποστηρίζει διακοπή διεργασιών ονομάζεται preemptive scheduler.**



Με διακοπή διεργασιών (preemptive)

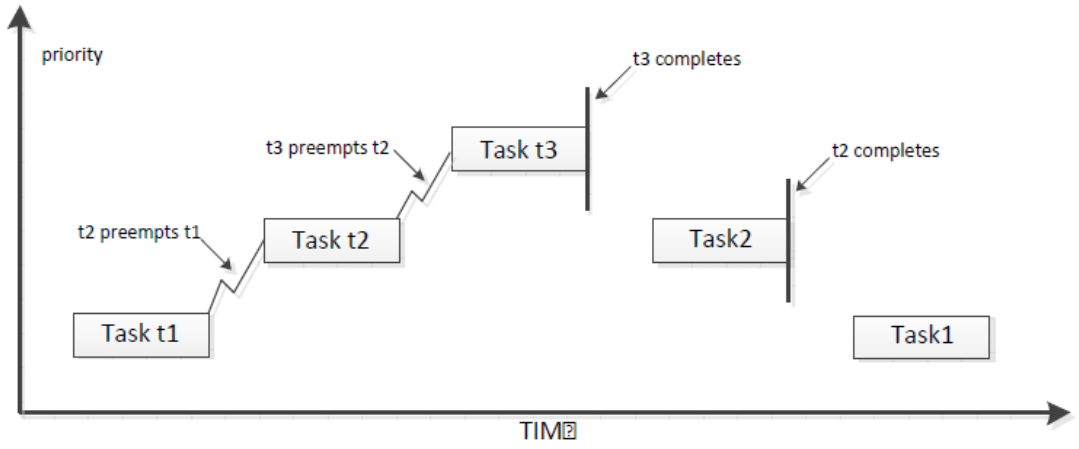


***Preemptive:* μια διεργασία μπορεί να διακοπεί και ο επεξεργαστής να δοθεί σε μια νέα διεργασία.**

Με διακοπή διεργασιών (preemptive)

* **Διαδοχική διακοπή διεργασιών κάθε φορά που καλείται ο scheduler ώστε να εκτελείται πάντα η διεργασία με την υψηλότερη προτεραιότητα.**

* **Η διεργασία t2 διακόπτει την t1, η t3 την t2 και βλέπουμε πως όταν η t3 ολοκληρώνει τότε ολοκληρώνεται η t2 και μετά η t1.**



Τεχνικές Χρονοδρομολόγησης

* **Διακρίνονται σε τρεις βασικές:**
  * **1. Οδήγηση από χρονιστή (clock driven)**
  * **2. Περιστρεφόμενη εκτέλεση(round robin)**
  * **3. Οδήγηση με προτεραιότητες (priorities)**

Οδήγηση από χρονιστή (clock driven)

* **Σε ένα σύστημα που χρησιμοποιεί χρονοδρομολόγηση με βάση την οδήγηση από χρονιστή όλες οι παράμετροι των διεργασιών πραγματικού χρόνου (περίοδος, χρόνος εκτέλεσης, προθεσμία, προτεραιότητα κλπ.) είναι γνωστές εκ των προτέρων.**

* **Το σύστημα υπολογίζει έναν πρόγραμμα εργασιών off-l ine, το αποθηκεύει και το χρησιμοποιεί κατά τον χρόνο εκτέλεσης.**

* **Ο χρονοδρομολογητής χρησιμοποιεί έναν χρονιστή, δηλαδή ένα προγραμματιζόμενο κύκλωμα παλμών, ώστε να παίρνει αποφάσεις δρομολόγησης διεργασιών σε καθορισμένα χρονικά διαστήματα.**

Οδήγηση από χρονιστή (clock driven)

* **Παράδειγμα:**

* **Τ1, Τ2 δύο διεργασίες, με περιόδους Π1 = Π2 = 4ms και χρόνο εκτέλεσης**
* **C = 2ms η κάθε μία.**

* **Υπολογίζεται από το πρόγραμμα εκ των πρότερων ένας πίνακας που περιλαμβάνεις τις 2 διεργασίες διαδοχικά και τους χρόνους λήψης απόφασης κατά την εκτέλεση του προγράμματος που προκύπτουν ανά 2 ms. Ο χρονιστής προγραμματίζεται να παράγει ένα παλμό λήψης απόφασης κάθε 2 ms όταν θα ολοκληρωθεί δηλαδή η εκτέλεση της προηγούμενης διεργασίας.**

* **Ένα πρόγραμμα που ελέγχεται από παλμούς ρολογιού κατά την εκτέλεσή του, παρουσιάζει ομαλή και αυστηρά προβλέψιμη συμπεριφορά, και άρα είναι κατάλληλο για αυστηρά (hard) συστήματα πραγματικού χρόνου.**

Περιστρεφόμενη εκτέλεση(round robin)

* **Εφαρμόζεται κυρίως στα συστήματα κατανομής χρόνου.**

* **Όταν μία διεργασία είναι έτοιμη για εκτέλεση μπαίνει σε μία ουρά καταχώρησης τύπου FIFO (First In First Out).**

* **Οι διεργασίες εκτελούνται με βάση μια θυρίδα χρόνου (time – slice), η οποία τυπικά έχει διάρκεια μερικές δεκάδες χιλιοστά του δευτερολέπτου.**

Περιστρεφόμενη εκτέλεση(round robin)

* **Όταν μία διεργασία δεν έχει ολοκληρωθεί στην θυρίδα του χρόνου τότε διακόπτεται από μία άλλη από τον scheduler (preempted) και τοποθετείται στο τέλος της ουράς για να περιμένει την επόμενη σειρά της.**

* **Αν υπάρχουν Ν διεργασίες στην ουρά τότε η εργασία Ν εκτελείται μετά από Ν χρονοθυρίδες.**

* **Παραλλαγή της παραπάνω τεχνικής έχει χρησιμοποιηθεί στον προγραμματισμό της κυκλοφορίας σε πραγματικό χρόνο.**

Οδήγηση με προτεραιότητες (priorities)

* **Οδήγηση με σταθερή ή δυναμική προτεραιότητα.**

* **Στην σταθερή η τιμή της προτεραιότητας κάθε διεργασίας υπολογίζεται μία φορά προσδίδεται κατόπιν στην διεργασία και παραμένει αναλλοίωτη για όλη τη διάρκεια ζωής της.**

* **Στην δυναμική η τιμή της προτεραιότητας υπολογίζεται κατά τη διάρκεια του χρόνου εκτέλεσης και συνδέεται με την προθεσμία που δίνεται στην εκάστοτε διεργασία. Η περίπτωση αυτή ονομάζεται οδηγούμενη από προθεσμία (deadline driven)**

Απόδοση Προτεραιοτήτων

* **Στατικοί και δυναμικοί αλγόριθμοι χρονοδρομολόγησης με βάσης τις προτεραιότητες.**

* **Για την εφαρμογή στατικών αλγορίθμων πρέπει να είναι γνωστές εκ των προτέρων όλες οι πληροφορίες για τον προγραμματισμό των διαδικασιών:**
  * **Αριθμός διεργασιών**
  * **Προθεσμίες**
  * **Προτεραιότητες**
  * **περίοδοι**

* **Το πρόγραμμα του προγραμματισμού στους στατικούς αλγόριθμους λύνεται πριν δημιουργηθεί, off line.**

Απόδοση Προτεραιοτήτων

* **Δυναμικοί Αλγόριθμοι:**
  * **Το πρόβλημα καθορίζεται κατά την διάρκεια της εκτέλεσης.**
  * **Αλλαγές στην διαμόρφωση του χρονικού προγραμματισμού μπορούν να γίνουν μόνο on line.**
* **Στατικοί αλγόριθμοι:**
  * **Καλύτερη προβλεψιμότητα**
  * **Άκαμπτοι και αδύναμοι να αντιμετωπίσουν νέες καταστάσεις που προκύπτουν κατά την διάρκεια της εκτέλεσης.**

Τεχνικές Αλγορίθμων στα RTOS

* **1. Μονοτονικού ρυθμού (Rate Monotonic – RM).**
* **2. Προήγηση συντομότερης προθεσμίας (Earliest Deadline First – EDF).**
* **3. Προήγηση ελάχιστου χρόνου χαλάρωσης (Least Slack-time First – LSTF).**

Μονοτονικού ρυθμού (Rate Monotonic – RM)

* **Η τεχνική αυτή αντιστοιχεί μια σταθερή προτεραιότητα σε κάθε διεργασία, σύμφωνα με την εξής αρχή: όσο πιο μικρή είναι η περίοδος της διεργασίας, τόσο μεγαλύτερη είναι η προτεραιότητα.**

* **Έχει αποδειχτεί ότι η τεχνική αυτή παρουσιάζει μια βέλτιστη συμπεριφορά ανάμεσα σε άλλες παρόμοιες τεχνικές που λειτουργούν με βάση σταθερές προτεραιότητες.**

Προήγηση συντομότερης προθεσμίας (Earliest Deadline First – EDF)

* **Όσο συντομότερη είναι η προθεσμία μιας διεργασίας, τόσο υψηλότερη είναι η προτεραιότητά της.**

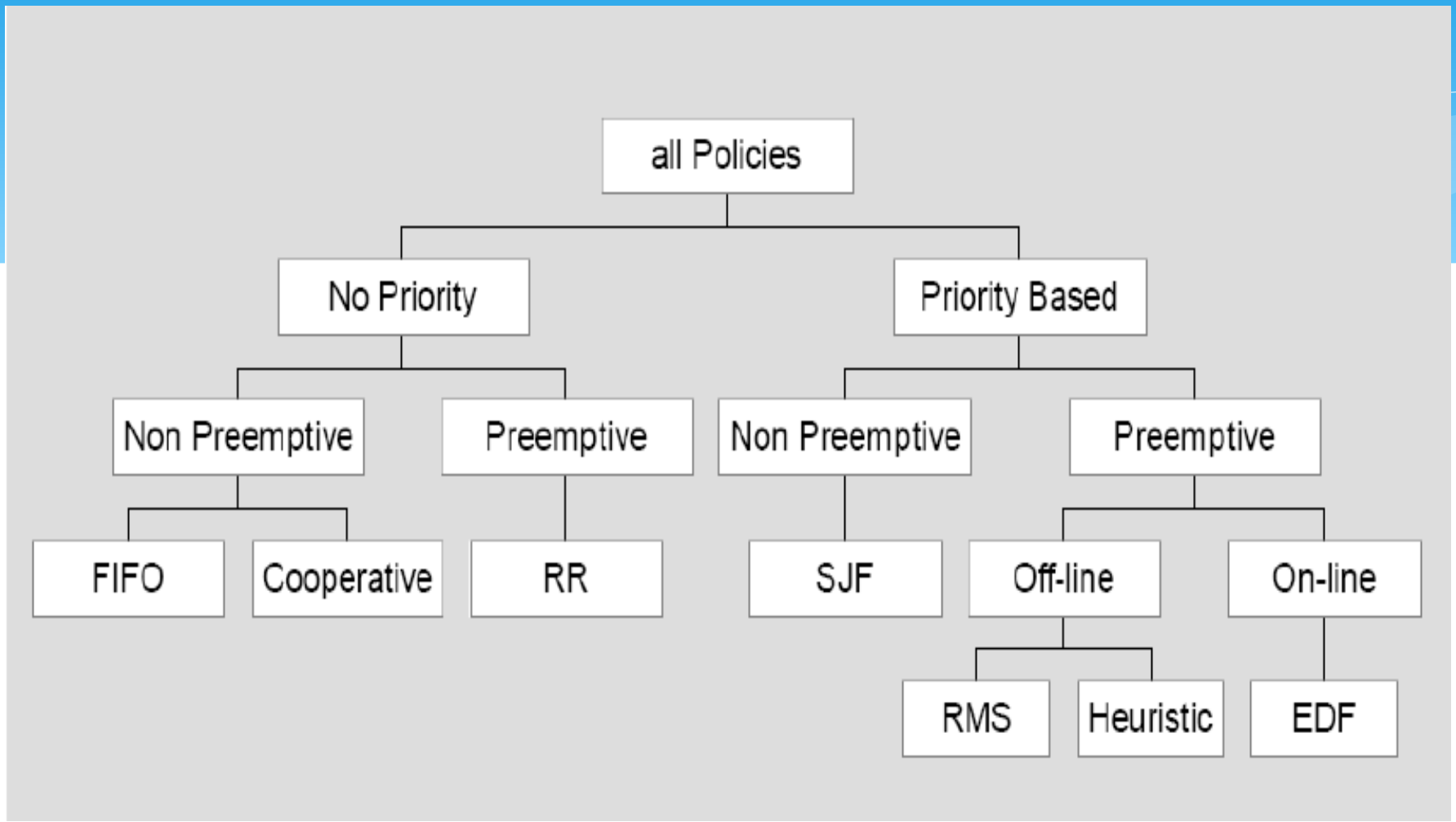
Προήγηση ελάχιστου χρόνου χαλάρωσης (Least Slack-time First – LSTF)

* **Ο χρόνος χαλάρωσης (slack-time) ορίζεται ως η διαφορά του διαστήματος από την τρέχουσα χρονική στιγμή μέχρι το πέρας της προθεσμίας μιας διεργασίας και του χρόνου εκτέλεσης της διεργασίας:**

* **(απόλυτη προθεσμία - τρέχουσα χρονική στιγμή) – χρόνος εκτέλεσης**

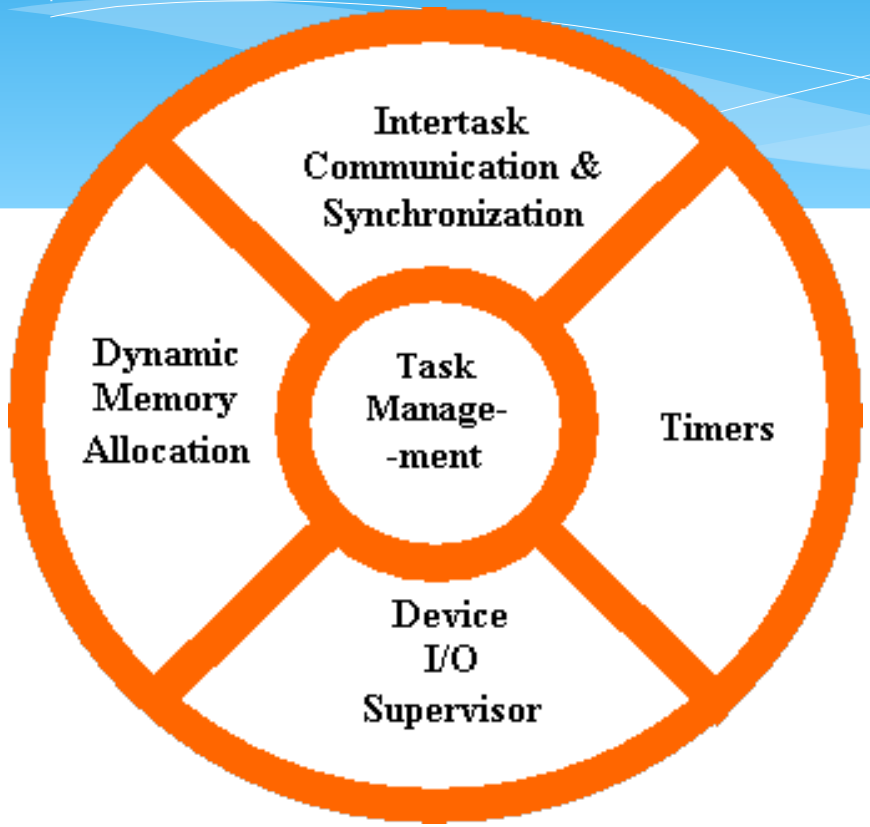
* **Σύμφωνα με την τεχνική αυτή η διεργασία που χαρακτηρίζεται την τρέχουσα στιγμή από τον ελάχιστο χρόνο χαλάρωσης έχει και την μεγαλύτερη προτεραιότητα.**

Κατηγορίες Τεχνικών Χρονοπρογραμματισμού στα RTOS



Πυρήνας RTOS - Υπηρεσίες

Στο παρακάτω σχήμα παρουσιάζονται οι υπηρισίες του πυρήνα RTOS:



Διεργασική Επικοινωνία και Συγχρονισμός

* **Κοινή πολιτική με τα κοινά λειτουργικά συστήματα.**
* **Σημαφόροι (mutexes) για συγχρονισμό αρνητικού τύπου**
* **Σημαίες, σήματα και μηνύματα για συγχρονισμό θετικού τύπου.**
* **Η επικοινωνία μεταξύ των διεργασιών γίνεται με μηνύματα:**

Διεργασική Επικοινωνία και Συγχρονισμός

* **Το μήνυμα είναι ουσιαστικά ένας πίνακας στον οποίο γράφει η εργασία - αποστολέας και από τον οποίο διαβάζει η εργασία - παραλήπτης, η διαδικασία ανάγνωσης του μηνύματος αποτελεί μια σειρά από δύο αντιγραφές.**

* **Όσο μεγαλύτερο είναι το μήνυμα τόσο μεγαλύτερος θα είναι ο χρόνος για τις αντιγραφές.**

* **Η λύση είναι να μεταδίδεται στην εφαρμογή παραλήπτη ένας δείκτης στον πίνακα αντί να αντιγράφεται ολόκληρος ο πίνακας.**

* **Για την αποφυγή μάλιστα συγκρούσεων στην πρόσβαση του πίνακα, αφαιρείται από την εφαρμογή - αποστολέα το αντίγραφο του δείκτη στον πίνακα.**

Δυναμική Διαχείριση Μνήμης

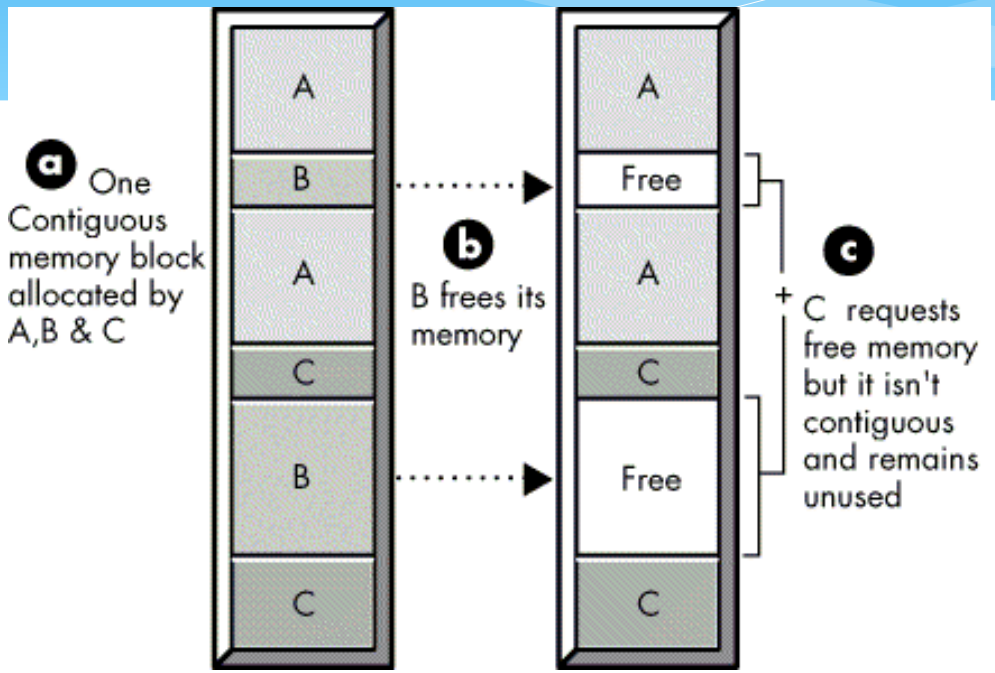
* **Διαχείριση μνήμης στα λειτουργικά συστήματα μη πραγματικού χρόνου παρουσιάζει προβλήματα καθώς δημιουργείται το πρόβλημα της κατάτμησης της εξωτερικής μνήμης.**

* **Οι σελίδες που δεσμεύονται από τις διεργασίες δεν είναι σειριακές με αποτέλεσμα να υπάρχουν κενές σελίδες που δεν μπορούν να χρησιμοποιηθούν από άλλες διεργασίες.**

* **Fragmentation=> μοιάζει με αυτό που υφίστανται οι δίσκοι και κάνουμε defrag για να το αντιμετωπίσουμε.**

* **Σε συστήματα όπου δεν υπάρχουν απαιτήσεις πραγματικού χρόνου, αλγόριθμοι όπως ο συλλέκτης σκουπιδιών (garbage collector) επιχειρούν να το αντιμετωπίσουν.**

Δυναμική Διαχείριση Μνήμης



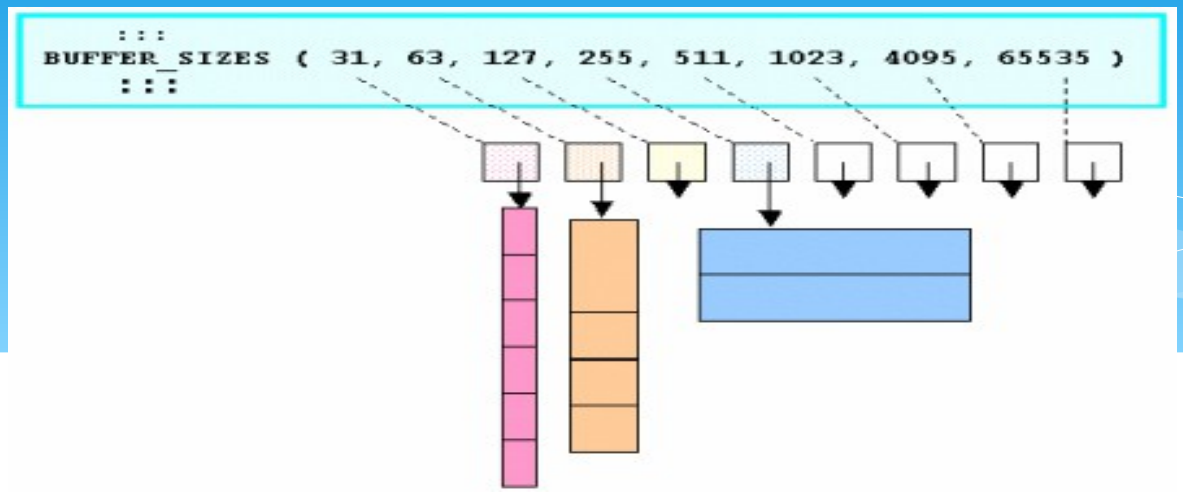
Δυναμική Διαχείριση Μνήμης

* **Στα λειτουργικά συστήματα πραγματικού χρόνου το φαινόμενο αντιμετωπίζεται την χρήση «πισίνων μνήμης» memory pools.**

* **Είναι περιοχές που μπορούν να σπάσουν σε επιμέρους τμήματα buffer ή blocks συγκεκριμένου πλήθους και συγκεκριμένου μεγέθους (δυνάμεις του 2).**

* **Όταν ένα τμήμα συγκεκριμένου μεγέθους εκχωρείται σε μια εργασία, αφού επιστραφεί κρατάει το μέγεθός του και δεν του επιτρέπεται να σπάσει σε μικρότερα. Έτσι σε επόμενη ζήτηση μνήμης ίδιου μεγέθους αυτό μπορεί να εκχωρηθεί γρήγορα, σε ντετερμινιστικό χρόνο.**

Δυναμική Διαχείριση Μνήμης



Χρήση «πισίνων» μνήμης (memory pools). Αυτές είναι περιοχές που μπορούν να σπάσουν σε επιμέρους τμήματα (buffers ή blocks) συγκεκριμένου πλήθους (από 4 μέχρι 8) και συγκεκριμένου μεγέθους (δυνάμεις του 2). Όταν ένα τμήμα συγκεκριμένου μεγέθους εκχωρείται σε μια εργασία, αφού επιστραφεί κρατάει το μέγεθός του και δεν του επιτρέπεται να σπάσει σε μικρότερα. Έτσι σε επόμενη ζήτηση μνήμης ίδιου μεγέθους αυτό μπορεί να εκχωρηθεί γρήγορα, σε ντετερμινιστικό χρόνο.

Βιβλιογραφικές Αναφορές

* 1. http://en.wikipedia.org/wiki/Real-time_operating_system
* 2. Αρχές Προγραμματισμού Πραγματικού Χρόνου - ΤΕΙ ΣΕΡΡΩΝ ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΕΠΙΚΟΙΝΩΝΙΩΝ
* 3. http://info.quadros.com/blog/bid/63579/RTOS-Explained-Preemptive-Scheduling
* 4. http://www.ni.com/white-paper/3938/en/
* 5. http://www.freertos.org/implementation/a00005.html
* 6.http://www.dauniv.ac.in/downloads/EmbsysRevEd_PPTs/Chap_8Lesson19EmsysNewTasksPreemptiveSch.pdf
* 7. http://user.it.uu.se/~yi/courses/rts/dvp-rts-08/notes/RTOS.pdf