ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Λειτουργικά Συστήματα

Φροντιστηριακή ενότητα # 3 : Εισαγωγή στα νήματα με POSIX Threads (Μέρος 1ο)

Τμήμα: Πληροφορικής

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Σκοποί ενότητας

- Εισαγωγή στον πολυνηματικό προγραμματισμό
- Διάκριση πολυνηματικού & πολυδιεργασιακού προγραμματισμού
- Εισαγωγή στα POSIX Threads

Περιεχόμενα ενότητας

- Νήματα και Διεργασίες
- Νήματα POSIX
- Παραδείγματα
- Εισαγωγή στον συγχρονισμό

ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Νήματα και Διεργασίες

Μάθημα: Λειτουργικά Συστήματα, **Φροντιστηριακή Ενότητα # 3 :** Εισαγωγή στα νήματα με POSIX Threads (Μέρος 1ο)

Τμήμα: Πληροφορικής

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Βασικές έννοιες (1/3)

- Τι είναι ένα νήμα;
  - Είναι μια ανεξάρτητη ροή εντολών
  - Χρονοπρογραμματίζεται χωριστά από το λειτουργικό σύστημα
- Διαισθητικός ορισμός
  - Μια ελαφριά διεργασία (lightweight process)
  - Εκτελείται ξεχωριστά απο το βασικό νήμα του προγράμματος (main thread of execution)

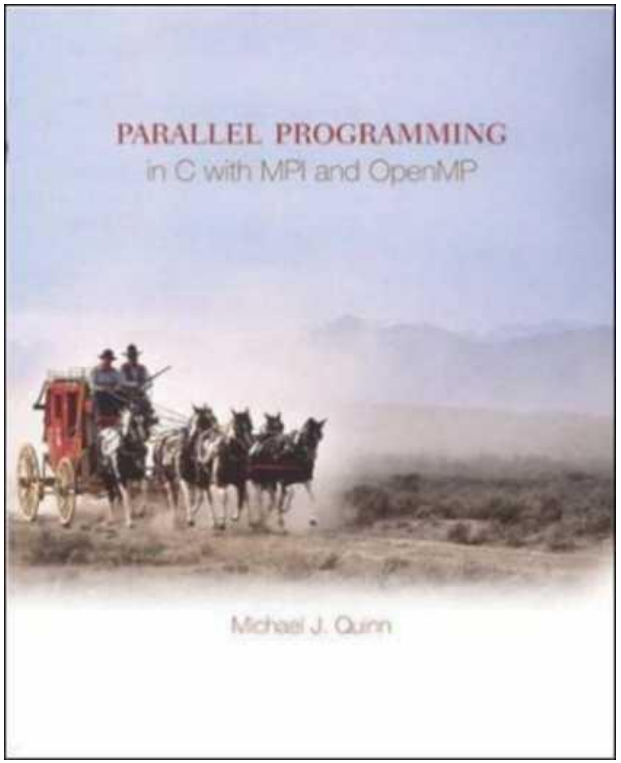
Βασικές έννοιες (2/3)

- Πολυνηματικός προγραμματισμός
- Παράλληλοι υπολογισμοί (ταχύτερη εκτέλεση)

$$\begin{pmatrix} a11 & a12 & a13 \\ a21 & a22 & a23 \\ a31 & a32 & a33 \end{pmatrix} \begin{pmatrix} b11 & b12 & b13 \\ b21 & b22 & b23 \\ b31 & b32 & b33 \end{pmatrix} =$$

$$\begin{pmatrix} a11.b11 + a12.b21 + a13.b31 & a11.b12 + a12.b22 + a13.b32 & a11.b13 + a12.b23 + a13.b33 \\ a21.b11 + a22.b21 + a23.b31 & a21.b12 + a22.b22 + a23.b32 & a21.b13 + a22.b23 + a23.b33 \\ a31.b11 + a32.b21 + a33.b31 & a31.b12 + a32.b22 + a33.b32 & a31.b13 + a32.b23 + a33.b33 \end{pmatrix}$$

- Αλλά και πιο απλοί λόγοι: αναβοσβήνει ο κέρσορας του κειμένου καθώς κατεβάζεις τη μπάρα κύλισης και αντίστροφα κ.α..



x4 πιο γρήγορα;
όχι.. Όμως πιο γρήγορα.

Βασικές έννοιες (3/3)

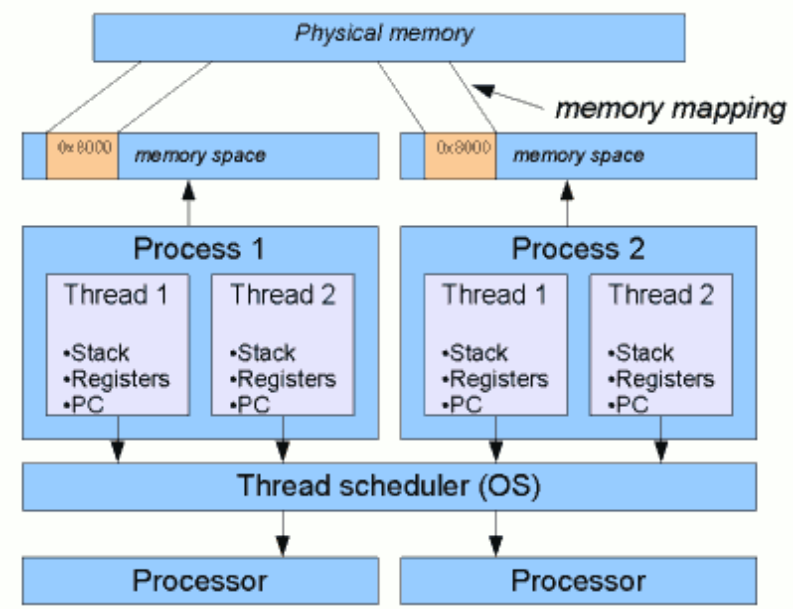
- POSIX Threads (pthreads)
- POSIX (Portable Operating System Interface) API standard για συγγραφή νημάτων
  - IEEE POSIX 1003.1c standard (1995)
  - http://en.wikipedia.org/wiki/POSIX
  - http://en.wikipedia.org/wiki/POSIX_Threads

Διεργασίες

- Πρόγραμμα σε εκτέλεση
- Κατάσταση (state) στο Unix
  - PID (Process ID), process group ID, user ID, group ID
  - Τρέχον ευρετήριο (working directory)
  - Εντολές προγράμματος σε εκτέλεση
  - Καταχωρητές (registers), «στοίβα» και «σωρός» (stack, heap)
  - Επικοινωνία μεταξύ διεργασιών (Inter-process communication)
    - Ουρές μηνυμάτων (message queues), σωληνώσεις (pipes), σηματοφορείς (semaphores), μοιραζόμενη μνήμη (shared memory).
  - FDs (file descriptors), Signals, Shared libraries …

Διεργασίες και Νήματα

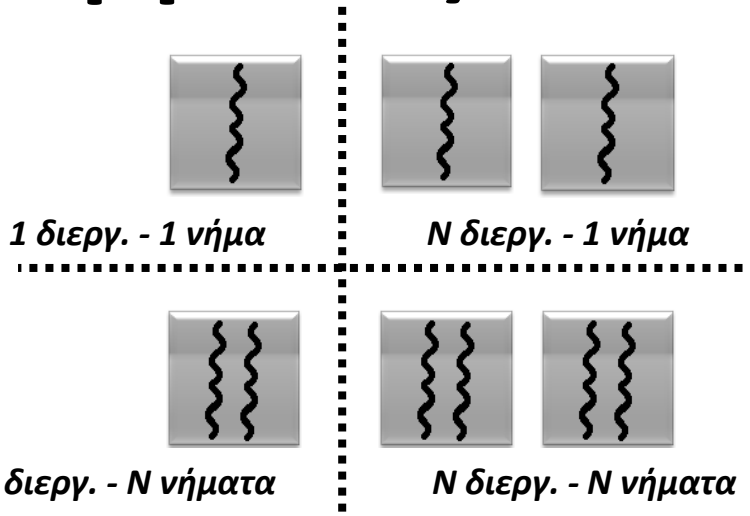
- Τα νήματα χρησιμοποιούν τους πόρους της διεργασίας που ανήκουν
- Εντούτοις, χρονοπρο/ζονται ως ανεξάρτητες ροές εκτέλεσης από το ΛΣ. Πώς;
  - Δικό τους stack pointer, program counter, registers, scheduling policies
  - Σύνολο από εκκρεμούντα ή σταματημένα σήματα (pending / blocked signals)
  - Δεδομένα ειδικά για threads

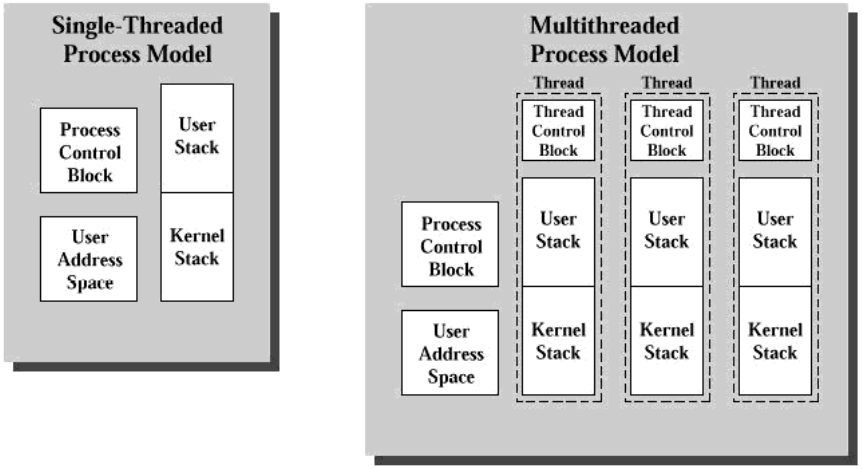


http://www.javamex.com/tutorials/threads/how_threads_work.shtml

Νήματα Vs. Διεργασίες

- Λιγότερος χρόνος δημιουργίας / τερματισμού / εναλλαγής μεταξύ νημάτων
  - Ίδια μνήμη
- Λιγότερο κόστος επικοινωνίας μεταξύ νημάτων
  - Πολύ εύκολα… μοιράζονται ήδη σχεδόν τα πάντα.





http://www.cs.cf.ac.uk/Dave/C/node29.html

Χρόνος δημιουργίας

Platform	fork()			pthread_create()		
	real	user	sys	real	user	sys
Intel 2.6 GHz Xeon E5-2670 (16 cores/node)	8.1	0.1	2.9	0.9	0.2	0.3
Intel 2.8 GHz Xeon 5660 (12 cores/node)	4.4	0.4	4.3	0.7	0.2	0.5
AMD 2.3 GHz Opteron (16 cores/node)	12.5	1.0	12.5	1.2	0.2	1.3
AMD 2.4 GHz Opteron (8 cores/node)	17.6	2.2	15.7	1.4	0.3	1.3
IBM 4.0 GHz POWER6 (8 cpus/node)	9.5	0.6	8.8	1.6	0.1	0.4
IBM 1.9 GHz p5-575 (8 cpus/node)	64.2	30.7	27.6	1.7	0.6	1.1
IBM 1.5 GHz POWER4 (8 cpus/node)	104.5	48.6	47.2	2.1	1.0	1.5
INTEL 2.4 GHz Xeon (2 cpus/node)	54.9	1.5	20.8	1.6	0.7	0.9
INTEL 1.4 GHz Itanium2 (4 cpus/node)	54.5	1.1	22.2	2.0	1.2	0.6

https://hpc-tutorials.llnl.gov/posix/why_pthreads/

Χρόνος: real, user, sys

- Real: πραγματικός χρόνος σύμφωνα με ρολόι τοίχου – χρόνος από εκκίνηση ως τέλος της κλήσης. Περιλαμβάνει όλο το χρόνο που πέρασε, καθώς και το χρόνο που χρησιμοποιήθηκε από άλλες διεργασίες και η διεργασία ήταν εμποδισμένη (για παράδειγμα, όταν περιμένει να τελειώσει η Ε/Ε)
- User: χρόνος ΚΜΕ που δαπανάται σε επίπεδο χρήστη (όχι μέσα στον πυρήνα) εντός της διεργασίας. Περιλαμβάνει μόνο τον πραγματικό χρόνο της ΚΜΕ για την εκτέλεση της διεργασίας. Οι άλλες διεργασίες και ο χρόνος που η διεργασία είναι εμποδισμένη δεν περιλαμβάνονται.
- Sys: χρόνος ΚΜΕ που δαπανάται σε επίπεδο πυρήνα εντός της διεργασίας. Περιλαμβάνει το χρόνο εκτέλεσης της ΚΜΕ που δαπανάται σε κλήσεις συστήματος μέσα στον πυρήνα, αλλά όχι τον χρόνο που δαπανάται σε βιβλιοθήκες στο επίπεδο του χρήστη. Όπως ο 'user', περιλαμβάνει μόνο χρόνο ΚΜΕ για τη διεργασία.

Επίπεδα νημάτων

User-Level Threads -- Thread Libraries

- Ο πυρήνας του ΛΣ δε γνωρίζει
  - Εύκολο thread switching εκτός kernel
  - Application specific scheduling.
  - Τρέχει σε κάθε ΛΣ
- Μπλοκάρει όλο τη διεργασία (άρα όλα τα νήματα) για κάποια system calls (άνοιγμα αρχείου)
- Ο πυρήνας αναθέτει μια διεργασία σε κάθε επεξεργαστή και όχι ένα νήμα ανά επεξεργαστή.

Kernel-level Threads -- System Calls

- Όλη η διαχείριση με κλήσεις του πυρήνα.
  - Ο πυρήνας χρονο/τίζει ταυτόχρονη εκτέλεση πολλαπλών νημάτων της ίδιας διεργασίας σε πολλαπλούς επεξεργαστές
  - Μπλοκάρει μόνο ένα thread
  - Οι κλήσεις του πυρήνα δυνητικά είναι multithreaded
- Κλήσεις προς τον πυρήνα είναι αργές…

Σύνοψη

- Μοιράζονται πόρους πατρικής διεργασίας.
  - Επηρεάζουν τα άλλα νήματα, πχ αλλαγή τιμής μεταβλητής
    - Δύο δείκτες (C pointers) με ίδια τιμή δείχνουν στην ίδια περιοχή μνήμης.
  - Ανάγκη για συγχρονισμό, αμοιβαίο αποκλεισμό και ατομική εκτέλεση πάνω στους πόρους (explicit synchronization) ρητά από το προγραμματιστή.
- Ανεξάρτητη ροή εκτέλεσης (independent flow of control).
- Υπάρχουν όσο υπάρχει η πατρική διεργασία / «Πεθαίνουν» όταν πεθάνει η πατρική διεργασία τους.
- Είναι ελαφρά (“lightweight”) σε σχέση με τις διεργασίες.

Πλεονεκτήματα

- Η σωστή υλοποίηση πολυνηματικών προγραμμάτων σε συνδυασμό με την εφαρμογή τους στα κατάλληλα προβλήματα παρουσιάζει τα εξής γενικά πλεονεκτήματα:
  - Βελτιώνει την απόκριση των εφαρμογών
  - Βελτιώνει την αξιοποίηση των πολυπύρηνων συστημάτων
  - Βελτιώνει τη δομή του κώδικα
  - Απαιτεί/δεσμεύει λιγότερους υπολογιστικούς πόρους

Νήματα POSIX

Μάθημα: Λειτουργικά Συστήματα, **Φροντιστηριακή Ενότητα # 3 :** Εισαγωγή στα νήματα με POSIX Threads (Μέρος 1ο)

Τμήμα: Πληροφορικής

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Κατηγορίες Pthreads API

- Διαχείριση: create, join, set/query attributes (joinable, scheduling) …
- Mutexes: “mutex -> mutual exclusion” για συγχρονισμό
  - Creating, destroying, locking and unlocking mutexes με χρήση των mutex attribute functions που αλλάζουν τα αντίστοιχα thread attributes.
- Condition variables: create, destroy, wait and signal
  - Για την επικοινωνία μεταξύ νημάτων που μοιράζονται το/α ίδιο/α mutexes.
  - Ο προγραμματιστής ορίζει τις συνθήκες (conditions).
- Synchronization: Ρουτίνες για read/write locks…

Pthreads API

Routine Prefix	Functional Group
pthread_	Threads & miscellaneous subroutines
pthread_attr_	Thread attributes objects
pthread_mutex_	Mutexes
pthread_mutexattr_	Mutex attributes objects
pthread_cond_	Condition variables
pthread_condattr_	Condition attributes objects
pthread_key_	Thread-specific data keys
pthread_rwlock_	Read/write locks
pthread_barrier_	Synchronization barriers

Δημιουργία Νήματος

- pthread_create args
  - thread: μοναδικό αναγνωριστικό για το νέο νήμα
    - Περνάμε έναν δείκτη σε ακέραιο
  - attr: αντικείμενο thread_attr_t
    - Ιδιότητες νήματος, NULL για τις default τιμές
  - start_routine: η C ρουτίνα που θα εκτελέσει το νήμα
    - Εδώ είναι που χρειάζονται οι function pointers
  - arg: παράμετρος που θα περάσει στην start_routine
    - Είναι void pointer (άρα, ό,τι θέλουμε)

Τερματισμός Νήματος

- Κλήση της pthread_exit
  - Εναλλακτικά:
    - Τερματίζει κανονικά η ρουτίνα start_routine
    - Κλήση της pthread_cancel από άλλο νήμα – δεν συνιστάται (διαρροές μνήμης)
    - Τερματίζει το βασικό νήμα, δηλαδή, φτάνει στο τέλος της η main() – σκοτώνει και όλα τα νήματα
    - Κλήση της exec() ή exit() για τερματισμό όλης της διεργασίας – σκοτώνει και όλα τα νήματα

Attributes του νήματος

- Κατά τη δημιουργία του νήματος δίνεται η δυνατότητα να οριστούν ιδιότητες οι οποίες καθορίζουν τόσο την κατάσταση όσο και την λειτουργία του νήματος.
  - Scheduling inheritance, sched policy, sched parameters, sched contention scope
  - Stack size, Stack address
- Ορισμός ιδιότητας: pthread_attr_init
- Κατάργηση ιδιότητας: pthread_attr_destroy

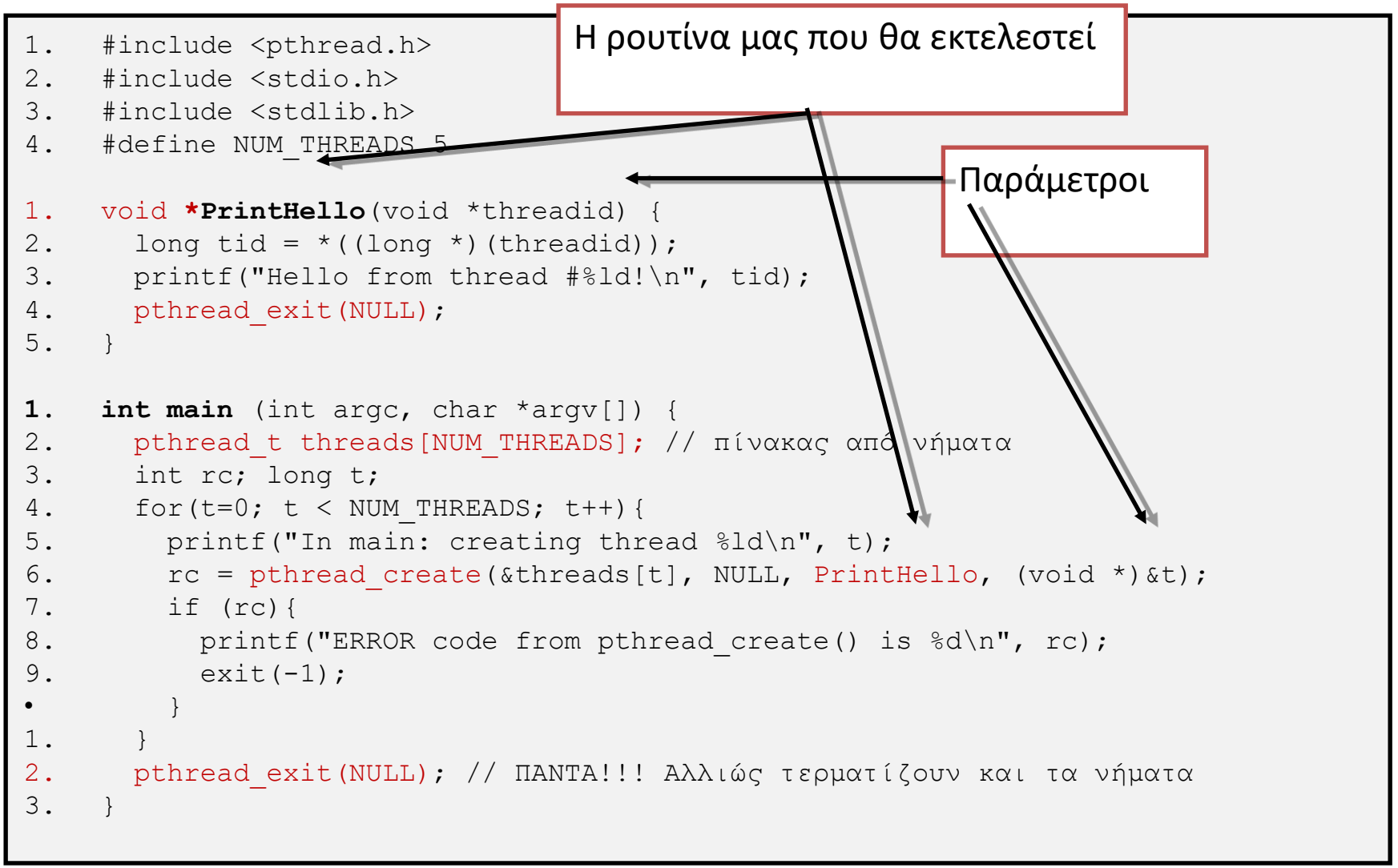
Παραδείγματα

Μάθημα: Λειτουργικά Συστήματα, **Φροντιστηριακή Ενότητα # 3 :** Εισαγωγή στα νήματα με POSIX Threads (Μέρος 1ο)

Τμήμα: Πληροφορικής

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Απλό παράδειγμα




```
1.   #include <pthread.h>
2.   #include <stdio.h>
3.   #include <stdlib.h>
4.   #define NUM_THREADS 5

1.   void *PrintHello(void *threadid) {
2.     long tid = *((long *)(threadid));
3.     printf("Hello from thread #%ld!\n", tid);
4.     pthread_exit(NULL);
5.   }

1.   int main (int argc, char *argv[]) {
2.     pthread_t threads[NUM_THREADS]; // πίνακας από νήματα
3.     int rc; long t;
4.     for(t=0; t < NUM_THREADS; t++){
5.       printf("In main: creating thread %ld\n", t);
6.       rc = pthread_create(&threads[t], NULL, PrintHello, (void *)&t);
7.       if (rc){
8.         printf("ERROR code from pthread_create() is %d\n", rc);
9.         exit(-1);
•         }
1.     }
2.     pthread_exit(NULL); // ΠΑΝΤΑ!!! Αλλιώς τερματίζουν και τα νήματα
3.   }
```

Παραπλήσιο εδώ: https://hpc-tutorials.llnl.gov/posix/creating_and_terminating/ (διορθωμένο)

Προσοχή στις παραμέτρους!

- Πρώτη παράμετρος: thread ID
  - Έχουμε πολλά νήματα οπότε φτιάχνουμε πίνακα
  - Περνάμε δείκτη σε ένα στοιχείο του πίνακα
- Δεύτερη παράμετρος: ιδιότητες (NULL=default)
- Τρίτη παράμετρος: starting function
  - Όνομα συνάρτησης που δέχεται και επιστρέφει void*
- Τέταρτη παράμετρος: παράμετρος συνάρτησης
  - Πρέπει να μετατραπεί σε void *
  - Η συνάρτηση θα την μετατρέψει στο σωστό τύπο

Εισαγωγή στον συγχρονισμό

Μάθημα: Λειτουργικά Συστήματα, **Φροντιστηριακή Ενότητα # 3 :** Εισαγωγή στα νήματα με POSIX Threads (Μέρος 1ο)

Τμήμα: Πληροφορικής

Απλό πρόβλημα αμοιβαίου αποκλεισμού

- Έστω δύο νήματα Α και Β και η μεταβλητή i με αρχική τιμή 5
  - Το Α εκτελεί: i ++;
  - Το Β εκτελεί: -- i;
- Ποιο είναι το (σωστό) αποτέλεσμα μετά την εκτέλεση των εντολών;

✓i = 6, i = 5
✓i = 4, i = 5

✘i = 6, i = 4
✘i = 4, i = 6

Πρόβλημα παραγωγού / καταναλωτή

- Ένα η περισσότερα νήματα παράγουν δεδομένα και τα τοποθετούν σε έναν ενταμιευτή (buffer)
- Ένα η περισσότερα νήματα αφαιρούν και καταναλώνουν τα παραχθέντα δεδομένα από τον ενταμιευτή
  - Απαγορεύεται η ταυτόχρονη πρόσβαση στον ενταμιευτή. Ένα νήμα μόνο προσθέτει ή αφαιρεί.
- Παραλλαγές κατά αύξουσα δυσκολία:
  - 1 παραγωγός| 1 καταναλωτής
  - Ν παραγωγοί | 1 καταναλωτής
  - Ν παραγωγοί | Κ καταναλωτές

Ενταμιευτής απείρου μεγέθους

- **Producer:**

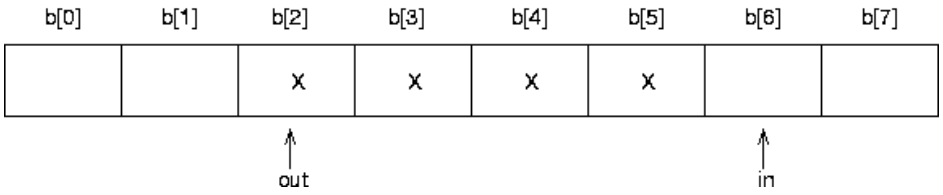
```
1.  while (true) {
2.    b [in] = v;
3.    in++;
4.  }
```

- **Consumer:**

```
1.  while (true) {
2.   while (in <= out) /* do
    nothing */;
3.   w = b [out];
4.   out++;
5.  }
```

Πρόβλημα 1: μη ρεαλιστικό

- *Οι ενταμιευτές δε μπορεί να είναι απείρου μεγέθους.*



Πρόβλημα 2: "Spin lock":

- *συγχρονισμός χωρίς condition variables*
- *Σπαταλάει χρόνο από τη CPU στα while όσο η διεργασία έχει στη κατοχή της τη CPU*

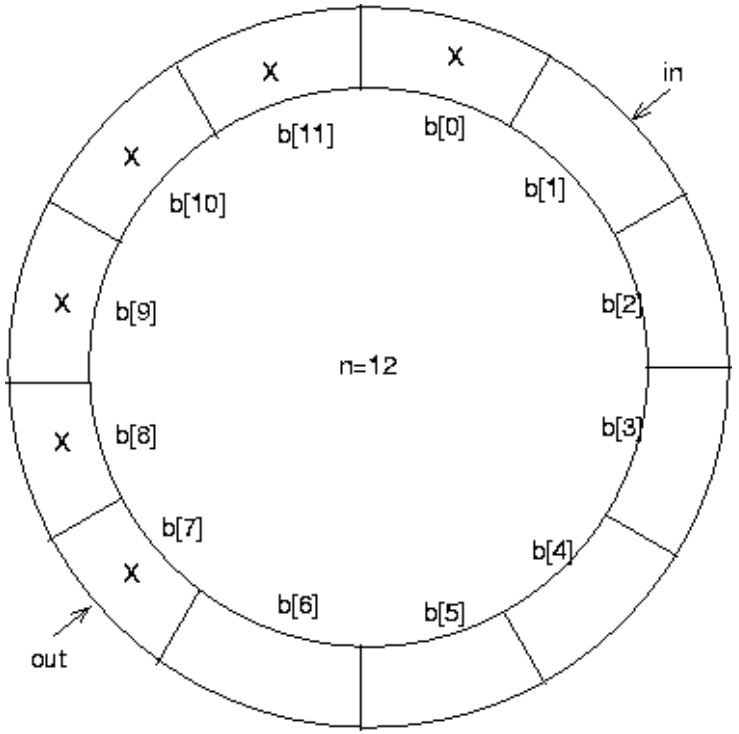
Λύση 1: Κυκλικός Ενταμιευτής

- **Producer:**

```
1.  while (true) {
2.    /*produce item v */
      while ((in + 1) % n ==
            out);
3.    b [in] = v;
4.    in = (in + 1) % n;
5.  }
```

- **Consumer:**

```
1.  while (true) {
2.    while (in == out);
3.    w = b [out];
4.    out = (out + 1) % n;
5.  }
```

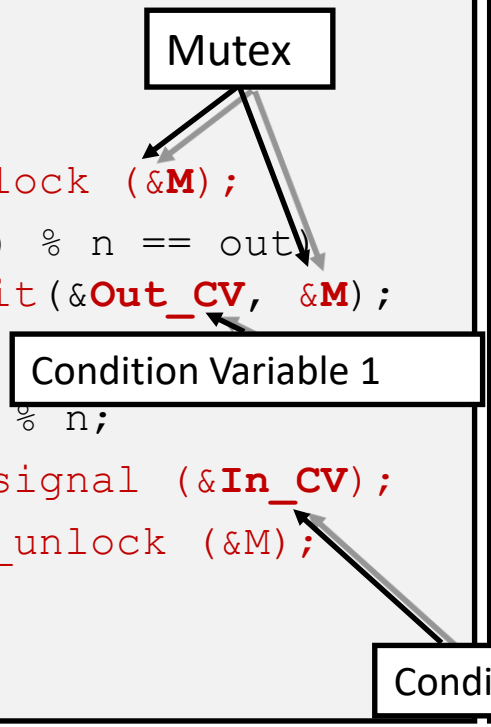


Το "Spin lock" όμως παραμένει..
(και σπαταλάμε και μία θέση!)

Συγχρονισμός με condition variables

❖ **Producer:**

```
1.while (true) {
2.   pthread_mutex_lock (&M);
3.   while ((in + 1) % n == out)
     pthread_cond_wait(&Out_CV, &M);
4.     b [in] = v;
5.     in = (in + 1) % n;
6.     pthread_cond_signal (&In_CV);
7.     pthread_mutex_unlock (&M);
8.}
```



❖ **Consumer:**

```
1.  while (true) {
2.    pthread_mutex_lock (&M);
3.    while (in == out)
      pthread_cond_wait
                    (&In_CV, &M);
4.      w = b [out];
5.      out = (out + 1) % n;
6.    pthread_cond_signal
                          (&Out_CV);
7.    pthread_mutex_unlock(&M);
8.
```

Κοινά Σφάλματα (1/2)

- Οι μεταβλητές συνθήκες σχετίζονται με mutex
  - Πρέπει να αναφέρεται το mutex στις πράξεις
  - Πρέπει να έχουμε κλειδώσει το mutex
    - Άρα wait/signal πρέπει να είναι μέσα σε lock/unlock
- Το ότι ξυπνήσαμε δεν είναι σίγουρα σωστό
  - Μπορεί η συνθήκη να μην ικανοποιείται
    - Μπορεί να ξύπνησαν πολλά νήματα
  - Πρέπει τα wait να είναι μέσα σε while
    - Ελέγχονται κάθε φορά που ξυπνάμε

Κοινά Σφάλματα (2/2)

- Προσοχή στη μνήμη!
  - Δεν επιστρέφουμε τοπική μνήμη
    - Οι τοπικές μεταβλητές χάνονται με τη συνάρτηση
    - Άρα εξαφανίζονται στο τέλος του νήματος
    - Είτε καθολικές μεταβλητές, είτε δυναμική μνήμη
  - Προσέχουμε τη δυναμική μνήμη
    - Να μην δημιουργείται μόνο στο νήμα
    - Πρέπει να απελευθερώνεται και κάπου

ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Τέλος Φροντιστηριακής Ενότητας # 3

Μάθημα: Λειτουργικά Συστήματα, **Φροντιστηριακή Ενότητα # 3 :** Εισαγωγή στα νήματα με POSIX Threads (Μέρος 1ο)

Τμήμα: Πληροφορικής

Ευρωπαϊκή Ένωση
Ευρωπαϊκό Κοινωνικό Ταμείο

ΕΠΙΧΕΙΡΗΣΙΑΚΟ ΠΡΟΓΡΑΜΜΑ
ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ
επένδυση στην κοινωνία της γνώσης
ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ & ΘΡΗΣΚΕΥΜΑΤΩΝ, ΠΟΛΙΤΙΣΜΟΥ & ΑΘΛΗΤΙΣΜΟΥ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ

ΕΣΠΑ 2007-2013
πρόγραμμα για την ανάπτυξη
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης