مسالك التنفيذ

العنوان	الصفحة

الكلمات المفتاحية:

مسلك التنفيذ، مسلك على مستوى النواة، مسلك على مستوى المستخدم، النموذج كثير إلى واحد، النموذج كثير إلى كثير، النموذج واحد إلى واحد، إيقاف مسلك التنفيذ، المسلك المستهدف، مجمَّع المسالك، المكتبة Pthreads.

ملخص:

يركز هذا الفصل على مفهوم مسالك التنفيذ وفوائدها، وأنواع مسالك التنفيذ، مقارنة بينها وبين الإجراءات، بالإضافة إلى نماذج مسالك التنفيذ، ومجمَّع مسالك التنفيذ.

أهداف تعليمية:

يهدف هذا الفصل إلى:

- التعرف على مسالك التنفيذ.
- فوائد مسالك التنفيذ.
- مقارنة بين الإجراءات ومسالك التنفيذ.
- أنواع مسالك التنفيذ (مسالك على مستوى النواة، ومسالك على مستوى المستخدم).
- نماذج تعدد مسالك التنفيذ (كثير إلى واحد، واحد إلى واحد، كثير إلى كثير).
- مجمَّع مسالك التنفيذ.
- مثال عن استخدام المكتبة Pthreads في إنشاء مسالك التنفيذ.

المخطط:

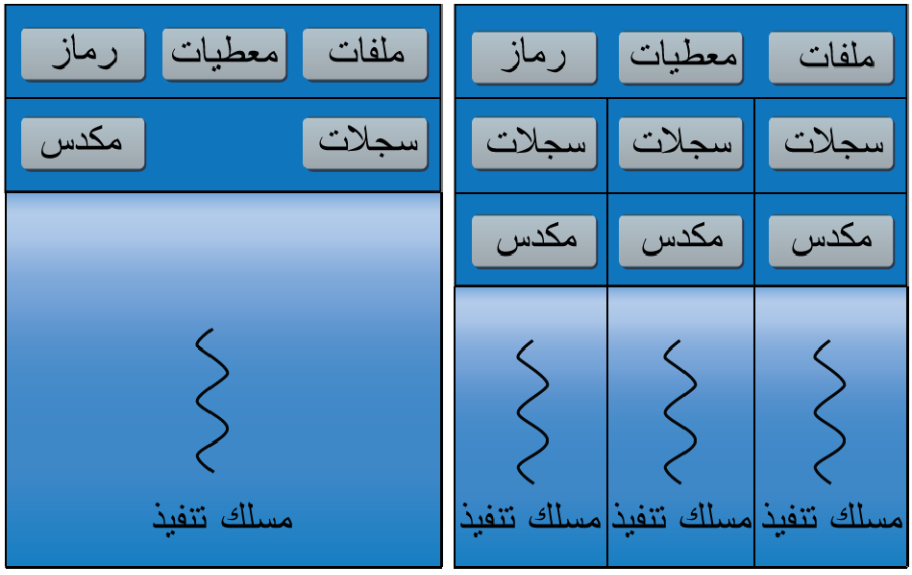
1. مسالك التنفيذ.
2. فوائد مسالك التنفيذ.
3. مقارنة بين الإجراءات ومسالك التنفيذ.
4. مسالك على مستوى النواة.
5. مسالك على مستوى المستخدم.
6. نماذج تعدد مسالك التنفيذ (النموذج كثير إلى واحد).
7. نماذج تعدد مسالك التنفيذ (النموذج واحد إلى واحد).
8. نماذج تعدد مسالك التنفيذ (النموذج كثير إلى كثير).
9. إيقاف مسالك التنفيذ.
10. مجمع مسالك التنفيذ.
11. مكتبة التعامل مع مسالك التنفيذ Pthreads.
12. التمارين.

1. مسالك التنفيذ

يُعرَّف مسلك التنفيذ بأنه الوحدة الأساسية في استخدام وحدة المعالجة، ويتألف من محدد هوية (ID)، وعدّاد برنامج، بالإضافة إلى مجموعة سجلات ومكدس.

تتشارك مسالك التنفيذ المنتمية إلى نفس الإجراء، في مقطع الرماز ومقطع المعطيات الخاصين بها، بالإضافة إلى موارد نظام التشغيل كالملفات.

يملك الإجراء التقليدي مسلك تنفيذ واحد، ولكن عندما يحوي الإجراء على عدة مسالك تنفيذ، فإنه يستطيع القيام بأكثر من مهمة في وقت واحد.



إجراء مع عدة مسالك تنفيذ إجراء مع مسلك تنفيذ وحيد

2. فوائد مسالك التنفيذ

يمكن تقسيم فوائد البرمجة باستخدام مسالك التنفيذ إلى أربع فوائد أساسية:

1. سرعة الإستجابة: يستمر التطبيق متعدد المسالك في التنفيذ، حتى لو كان جزءاً منه ينتظر أو يؤدي عملية طويلة، وهذا يرفع سرعة الاستجابة.
2. التشارك في الموارد: تتشارك مسالك التنفيذ تلقائياً في الذاكرة، وفي الموارد الخاصة بالإجراء الذي تنتمي إليه، وهذا ما يسمح للتطبيق بإمتلاك عدة مسالك تقع كلها ضمن فضاء عنونة واحد.
3. الاقتصاد: إن حجز الذاكرة والموارد من أجل إنشاء الإجراءات، هو عملية مكلفة، وبما أن مسالك التنفيذ تتشارك في موارد الإجراء الذي تنتمي إليه، يُعتبر إنشاء المسالك وتبديل سياقها، بدلاً من إنشاء الإجراءات، أمراً اقتصادياً أكثر.

4. استخدام بنية متعددة المعالجات: تتضاعف فوائد تعدد مسالك التنفيذ في بنية متعددة المعالجات، حيث يمكن تنفيذ كل مسلك على التوازي على معالج مختلف. بينما لا يمكن تنفيذ إجراء أحادي المسلك إلا على وحدة معالجة واحدة.

3. مقارنة بين الإجراءات ومسالك التنفيذ

الإجراءات	مسالك التنفيذ
يحتاج توليد إجراء جديد إلى موارد كثيرة.	لا يحتاج توليد مسلك تنفيذ جديد إلى موارد كثيرة.
تبديل السياق بين الإجراءات مكلف.	تبديل السياق بين المسالك أقل تكلفة منه في حالة الإجراءات.
الاتصال بين الإجراءات صعب.	الاتصال بين مسالك التنفيذ سهل.
يوجد حماية للإجراءات وعزل فيما بينها.	لا يوجد حماية للمسالك أو عزل فيما بينها.

4. مسالك على مستوى النواة

هنالك نوعان لمسالك التنفيذ: مسالك على مستوى النواة، ومسالك على مستوى المستخدم.

المسالك على مستوى النواة:

تكون بنية معطيات المسالك معرفة ضمن منطقة الذاكرة الخاصة بنواة نظام التشغيل، وتكون عملية إنشاء وإدارة المسالك من مهام نظام التشغيل (استدعاءات نظام). يحصل كل مسلك على فترة تنفيذ خاصة به. وإذا توقف أحد المسالك بسبب استدعاء نظام ما، فإن ذلك لا يؤثر على بقية المسالك.

5. مسالك على مستوى المستخدم

هنالك نوعان لمسالك التنفيذ: مسالك على مستوى النواة، ومسالك على مستوى المستخدم.

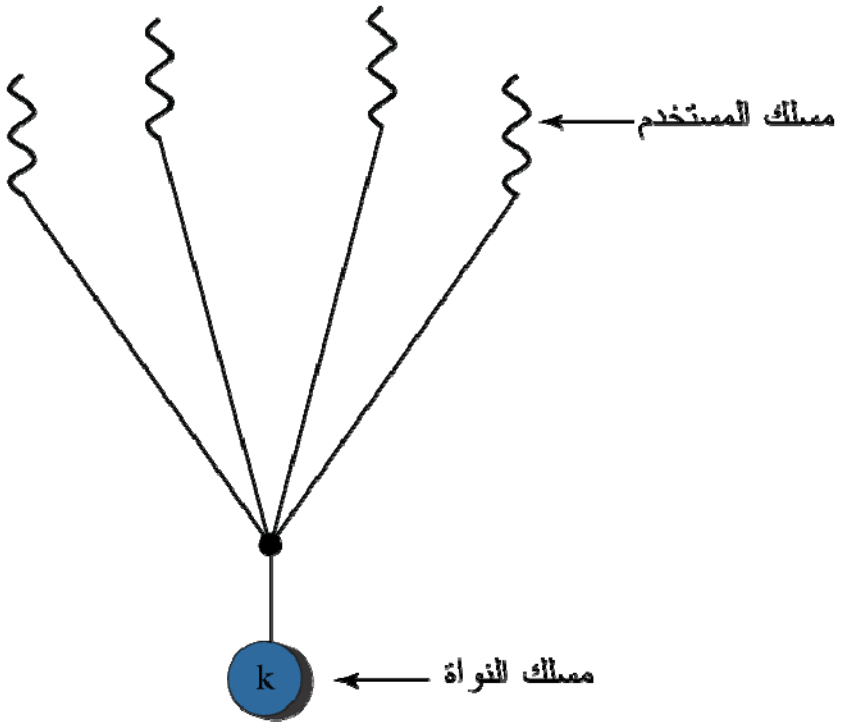
المسالك على مستوى المستخدم:

- تُعرَّف بنية المعطيات الخاصة بالمسالك ضمن منطقة الذاكرة الخاصة بإجراء المستخدم.
- تكون عملية إنشاء وإدارة المسالك من مهام الإجراء نفسه (باستخدام مكتبة من الإجراءات العادية) بدون استخدام استدعاءات نظام، وبالتالي يتم ذلك بسرعة أكبر.
- تتقاسم المسالك فيما بينها الفترة الزمنية المخصصة لتنفيذ الإجراء.
- إذا توقف أحد المسالك بسبب استدعاء نظام، فإن الإجراء بالكامل يتوقف، وتتوقف بالتالي جميع المسالك.

6. نماذج تعدد مسالك التنفيذ (النموذج كثير إلى واحد)

النموذج الأول من نماذج تعدد المسالك، هو النموذج كثير إلى واحد:

تجري في هذا النموذج مقابلة أكثر من مسلك على مستوى المستخدم، بمسلك واحد على مستوى النواة، حيث تجري إدارة المسالك في فضاء المستخدم ولذلك فهي فعالة، إلا أن الإجراء سيتوقف كله إذا قام مسلك واحد بطلب أحد استدعاءات النظام التي تسبب الانتظار. كما يمكن لمسلك واحد فقط النفاذ إلى النواة في وقت واحد، وبذلك لا تستطيع عدة مسالك أن تنفذ على التوازي حتى ولو تعددت المعالجات.

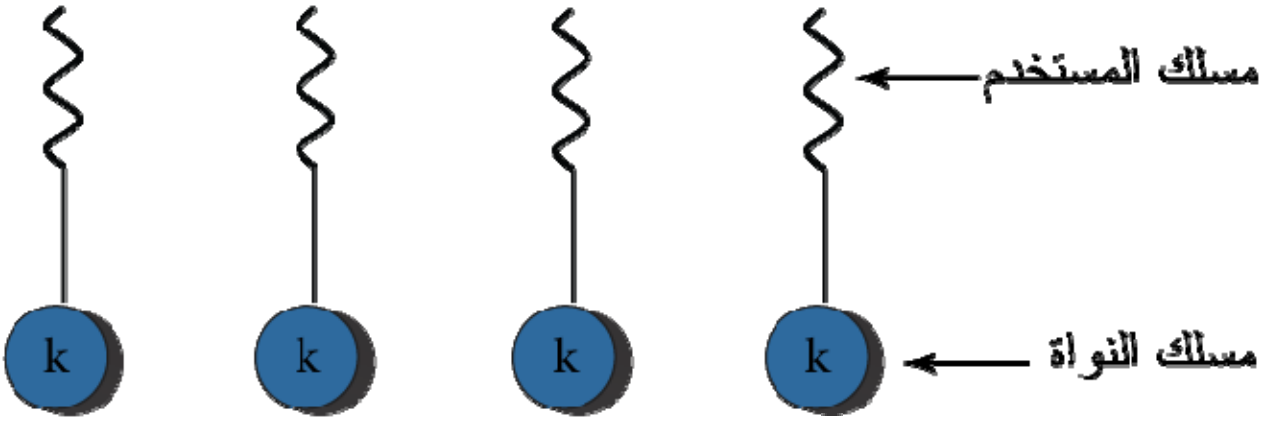


7. نماذج تعدد مسالك التنفيذ (النموذج واحد إلى واحد)

في النموذج واحد إلى واحد، تجري مقابلة كل مسلك مستخدم بمسلك نواة.

يوفر هذا النموذج قدرة أكبر على التوازي من النموذج كثير إلى واحد، وذلك من خلال السماح بتنفيذ مسلك ثاني حين يستدعي مسلك ما أحد استدعاءات النظام التي تسبب الانتظار. كما يسمح بتنفيذ عدة مسالك على التوازي في حال وجود عدة معالجات.

تكمن سيئة هذا النموذج في أن إنشاء مسلك مستخدم جديد يتطلب إنشاء مسلك النواة الموافق له، مما يؤدي إلى عبء إضافي على التطبيق ويؤثر سلباً على أدائه.



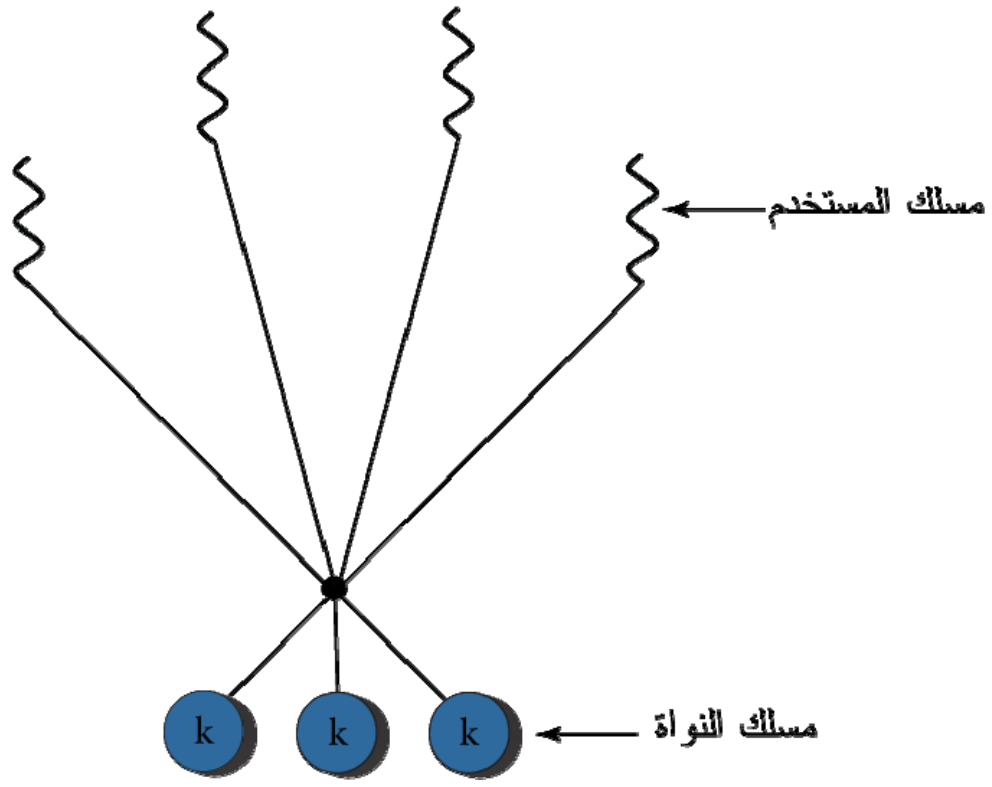
8. نماذج تعدد مسالك التنفيذ (النموذج كثير إلى كثير)

في النموذج كثير إلى كثير تجري مقابلة عدة مسالك على مستوى المستخدم مع عدة مسالك على مستوى النواة، بحيث يكون عدد مسلك النواة أقل أو يساوي عدد مسلك المستخدم.

يتيح النموذج "كثير إلى واحد للمطوِّر إنشاء العدد الذي يرغب به من مسالك المستخدم، لكنه لا يسمح له بتحقيق توازي حقيقي في التنفيذ، لأن النواة لا تجدول إلا مسلك واحد في وقت واحد.

ويتيح النموذج واحد إلى واحد توازي أكبر، لكن يتطلب من المطوِّر انتباهاً أكبر على عدد المسالك التي يجري إنشاؤها.

أما نموذج "كثير إلى كثير" فإنه لا يعاني من أي من هذه المساوئ: إذ يستطيع المطورون إنشاء العدد الضروري من المسالك، بحيث يمكن تنفيذ مسالك النواة الموافقة لها، على التوازي عند وجود عدة معالجات. كذلك فعند قيام مسلك ما باستدعاء أحد استدعاءات النظام التي تسبب الانتظار، فإن النواة تستطيع جدولة مسلك آخر للتنفيذ.



9. إيقاف مسالك التنفيذ

يمكن إيقاف مسلك التنفيذ، أي إنهاء مهمته قبل أن يُنهي تنفيذه، فعلى سبيل المثال، حين تبحث عدة مسالك على التوازي في قاعدة معطيات، ويعيد أحد هذه المسالك نتيجة ما، يمكن إيقاف المسالك الأخرى. كذلك يجري إيقاف مسلك التحميل عندما يضغط المستخدم على زر في متصفح الوِب من أجل إيقاف تحميل الصفحة (حيث يجري تحميل صفحة الوِب غالباً بواسطة مسلك تنفيذ مستقل).

يمكن إيقاف مسلك التنفيذ في حالتين:

إيقاف لا متزامن: يقوم أحد المسالك بإنهاء المسلك المستهدف مباشرةً.

إيقاف مؤجل: يستطيع المسلك المستهدف أن يتحقق دورياً، هل يجب عليه أن ينتهي أم لا؟ وبذلك يمكن أن يُنهي تنفيذه نهاية نظامية.

10. مجمَّع مسالك التنفيذ

تتلخص الفكرة العامة لمجمَّع مسالك التنفيذ في إنشاء عدة مسالك تنفيذ عند بدء الإجرائية، ومن ثم وضع هذه المسالك في مجمَّع، وإبقائها جاهزة للعمل.

عندما يستلم النظام طلب ما، يُسنِد مهمة الطلب إلى أحد المسالك المتاحة في المجمَّع، وحين ينتهي مسلك التنفيذ من مهمته، يعود إلى المجمَّع لانتظار مهمات جديدة لحين إسنادها إليه.

في حال عدم احتواء المجمَّع على مسالك متاحة، ينتظر النظام إنهاء أحد المسالك للمهمة الموكلة إليه.

أهم فوائد مجمَّع مسالك التنفيذ:

- خدمة أسرع للطلبات من حالة مسلك تنفيذ واحد، ومن حالة انتظار مسلك تنفيذ جديد.
- تحديد عدد المسالك الموجودة في وقت واحد، وهي ناحية مهمة جداً خصوصاً في النظم التي لا تدعم إلا عدد محدد من مسالك التنفيذ.

11. مكتبة التعامل مع مسالك التنفيذ Pthreads

- تعتبر Pthreads مكتبة برمجية تساعد على إنشاء مسالك التنفيذ ومزامنتها، وهي عبارة عن مكتبة على مستوى المستخدم، وجميع المسالك التي يتم تعريفها من خلال هذه المكتبة، هي عبارة عن مسالك على مستوى المستخدم.
- تتضمن جميع برامج Pthreads ملف الترويسة pthread.h، كما يمثل المتحول "pthread_t tid" محدد الهوية الخاصة بالمسلك المراد إنشاؤه، كما يمثل "pthread_attr_t attr" التصريح عن مجموعة واصفات المسلك (حجم المكدس، ومعلومات الجدولة).

```
#include <pthread.h>
#include <stdio.h>

int sum; /* this data is shared by the thread(s) */
void *runner(void *param); / the thread */

main (int argc, char *argv[])
{
    pthread_t tid; /* the thread identifier */
    pthread_attr_t attr; /* set of thread attributes */
    if (argc != 2) {
        fprintf(stderr, "usage: a.out <integer value>\n");
        exit() ;
    }
    if (atoi(argv[1]) < 0) {
        fprintf(stderr, "%d must be >=0\n",atoi(argv[1]));
        exit() ;
    }
    /* get the default attributes */
    pthread_attr_init(&attr);
    /* create the thread */
    pthread_create(&tid,&attr,runner,argv[1]);
    /* now wait for the thread to exit */
    pthread_join(tid,NULL);
    fprintf("sum = %d\n",sum);
}
```

مثال عن استخدام هذه المكتبة:

يمكن كتابة برنامج يقوم بإنشاء مسلك مستقل مهمته حساب ناتج جمع الأعداد من 1 وحتى عدد صحيح معين غير سالب.

يبدأ البرنامج بالتنفيذ من خلال مسلك التنفيذ الأساسي main، ومن ثم يُنشئ main مسلك ثاني يقوم بوظيفة الحساب وهو المسلك runner.

تتضمن جميع برامج Pthreads ملف الترويسة pthread.h، كما يمثل المتحول tid محدد الهوية للمسلك المراد إنشاؤه، كما يمثل attr التصريح عن مجموعة واصفات المسلك (حجم المكدس، ومعلومات الجدولة).

```
/* the thread will begin control in this function */
void *runner(void *param)
{
    int upper = atoi(param);
    int i;
    sum = 0;
    if (upper > 0) {
        for (i = 1; i <= upper; i++)
            sum += i;
    }
    pthread_exit(0);
}
```

12. التمارين

1. يُعرَّف مسلك التنفيذ بأنه الوحدة الأساسية في استخدام وحدة المعالجة، ويتألف من محدد هوية (ID)، وعدّاد برنامج، بالإضافة إلى مجموعة سجلات ومكدس:
   A. صح
   B. خطأ

2. من فوائد البرمجة باستخدام مسالك التنفيذ:
   A. التشارك في الموارد
   B. سرعة الاستجابة
   C. الاقتصاد
   D. جميع الإجابات صحيحة

3. يحتاج توليد إجراء جديد إلى موارد كثيرة على عكس توليد مسلك تنفيذ جديد:
   A. صح
   B. خطأ

4. يوجد حماية للمسالك أو عزل فيما بينها على عكس الإجراءات:
   A. صح
   B. خطأ

5. تتقاسم المسالك فيما بينها الفترة الزمنية المخصصة لتنفيذ الإجراء في المسالك على مستوى النواة:
   A. صح
   B. خطأ

6. تجري في هذا النموذج مقابلة أكثر من مسلك على مستوى المستخدم، بمسلك واحد على مستوى النواة:
   A. النموذج كثير إلى واحد
   B. النموذج واحد إلى واحد
   C. النموذج كثير إلى كثير
   D. جميع الإجابات خاطئة

7. يتيح النموذج "كثير إلى واحد" للمطوِّر إنشاء العدد الذي يرغب به من مسالك المستخدم، لكنه لا يسمح له بتحقيق توازي حقيقي في التنفيذ:

A. صح

B. خطأ

8. يقوم أحد المسالك بإنهاء المسلك المستهدف مباشرةً في الإيقاف المؤجل:

A. صح

B. خطأ

9. أهم فوائد مجمَّع مسالك التنفيذ: خدمة أسرع للطلبات من حالة مسلك تنفيذ واحد:

A. صح

B. خطأ

10. تعتبر Pthreads مكتبة برمجية تساعد على إنشاء مسالك التنفيذ ومزامنتها:

A. صح

B. خطأ

رقم التمرين	الإجابة الصحيحة
.1	(A)
.2	(D)
.3	(A)
.4	(B)
.5	(B)
.6	(A)
.7	(A)
.8	(B)
.9	(A)
.10	(A)