بسم الله الرحمن الرحيم

نظم التشغيل ـ الجزء الثانى

محاضرة رقم (8)

تزامن العمليات Process Synchronization

تعريف : هى آلية تستخدم للتأكد من أن تنفيذ العمليات المتعاونة والتى تتشارك فى المساحة التخزينية (Logical Add Space) يتم بصورة مرتبة (order) وبذلك تحافظ على إنسجام البيانات أو إستقرار البيانات (Data Consistency) .

- بمعنى أن الوصول للبيانات المشتركة Share data بين العمليات بصورة Concurrency قد يؤدى إلى data inconsistency وللمحافظة على الـ data consistency فإننا نحتاج إلى تزامن العمليات Process Synchronization .
- مثال لذلك : مشكلة المنتج والمستهلك product & consumer problem حيث أن العمليتان يتشاركان الـ bounded memory buffer وتظهر المشكلة إذا تم تنفيذ العمليتين بصورة Concurrency بمعنى أن يتم الآتى :

Counter ++ & Counter -- فى نفس الوقت إذا القيمة التى يأخذها العداد ستكون غير صحيحة وفى هذه الحالة يحدث ما بالـ interleaving .

* الـ Race condition : (حالة التسابق)

(1) هو الوضع الذى تصل فيه مجموعة من العمليات للبيانات المشتركة بينها وتقوم بالتعديل فيها بصورة متزامنة Concurrency .

(2) هو حالة التسابق بين العمليات للحصول على مورد معين (تسابق كل العمليات) .

* لمنع حدوث حالة التسابق لا بد من حدوث تزامن العمليات .

بمعنى مثلاً : فى مشكلة المنتج والمستهلك تنفيذ الجملتين : Counter ++ & Counter -- يجب أن يتم بصورة atomically

* الـ Atomic operation : تعنى أن يتم تنفيذ العملية كاملاً دون أى مقاطعة .

* كمثال لمشكلة الـ Race condition نأخذ برنامج الطباعة (عملية إدارة الطابعة لعدد من الملفات المراد طباعتها) .

Front (الرأس)	1	2	.....	4	5	6	7	.....	9	10	Rear (الذيل)

Out === Next file table print k n f b In ==== Next free slot

توضيح :

* الـ (printer spooler) هو عبارة عن صف توضع فيه أسماء الملفات أو العمليات المراد طباعتها ويقوم بدوره بوضعها فى slots .
* الـ (Printer daemon) هو عبارة عن Module (وحدة نمطية) تابعة لنظام التشغيل تقوم بقراءة الـ **Next file table print** ويقوم بطباعتها وبالتالى يغير الـ Out إلى العملية التالية أو الملف القادم .
* لنفترض أن لدينا عمليتان A & B المطلوب طباعتهما والملف A هو M والملف B هو S .

- يتم تبديل المهمات وفقاً لزمن محدد وتقوم الطابعة بطباعة الملف الموجود فى مقدمة الصف أو الرأس **Front** من خلال المؤشر **Front** بينما يتم إضافة ملف جديد لصف الطابعة من خلال المؤخرة أو الذيل **Rear** أى المؤشر **Rear .**

- مثلاً إذا أرادت المهمة (A) إرسال الملف M للطباعة فإنها تقوم بقراءة المؤشر **Rear** والذى يساوى مثلاً (10) وتقوم بتخزينه فى متغير محلى خاص بها مثلاً (X) أى أن X= 10 وفى هذه اللحظة إذا توقفت هذه العملية وإنتقل الـ cpu إلى العملية التالية (B) فإن (B) تقوم بقراءة المؤشر **Rear** وهو (10) وتقوم بتخزينه فى متغير محلى خاص مثلاً (Y) أى أن Y= 10 ويتم إرسال الملف إلى الصف .

- يستقبل برنامج الطباعة الملف (Y) ومعه المتغير الخاص بموقعه (10) وتزيد قيمة (Y) بمعدل واحد أى تصبح Y= 10 وقيمة المؤشر **Rear = 11**
- بعد فترة تعود المهمة (A) إلى إستئناف العمل وتواصل من النقطة التى توقفت فيها وتقوم بإرسال الملف ومعه قيمة المتغير (X) والتى تساوى (10) ويقوم برنامج الطباعة بوضع الملف فى الموقع (10) ويقوم بزيادة المتغير (X) بواحد أى يصبح الـ **Rear** = 11 وهنا نجد أن ملف (B) وهو (S) لن تتم طباعته أبداً وذلك لحدوث ما يسمى بـ overwriting له ولن يلاحظ برنامج الطباعة ذلك .

* ***المنطقة الحرجة*** (Critical Section) :

تعنى توفير بيانات مشتركة لعملية دون العمليات الأخرى فى نفس اللحظة بسبب حالة التسابق .

توضيح : إذا كانت هنالك عملية يتم تنفيذها وتحتاج إلى Share date (بيانات مشتركة)فهى تقوم بقراءتها من الـ Certain memory location وإذا لم تتوفر البيانات المشتركة لعملية أخرى (لم تطلبها)يتم توفيرها لهذه العملية على الرغم من وجود العملية الأخرى التى قد تحتاج لهذه البيانات وفى هذه الحالة يمكن القول أن الـ process موجودة فى المنطقة الحرجة (Critical Section) أو الـ (Critical region) أو منطقة التخزين المشترك .

- ***Each process has a code segment , called critical section in which share data is accessed and changed***
- ***The remainder of the process code is called the remainder section .***

* عند دخول العملية للمنطقة الحرجة (Critical Section) يجب أن لا تدخل أى عملية أخرى لنفس المنطقة .

س| كيف تم حل مشكلة حالة التسابق ؟ بتقنية المنع التبادلى Mutual Exclusion .

*** المنع التبادلى : Mutual Exclusion :**

هو يعنى أن العملية إذا دخلت المنطقة الحرجة أو كانت فى حالة تنفيذ فعندئذ لا يسمح لأى عملية أخرى بالتنفيذ فى نفس المنطقة .

تعريف آخر : هو تقنية لمنع حالة التسابق . أو : هو منع حدوث أى مقاطعة من نظام التشغيل لعملية دخلت المنطقة الحرجة .

طرق تحقيق المنع التبادلى :

(1) تعطيل المقاطعات Disabling Interrupts : بمعنى عندما تدخل العملية المنطقة الحرجة تعطى الصلاحية للعملية لتعطيل المقاطعات مثلاً صلاحية إيقاف الـ timer وبعد الإنتهاء من وظيفتها تقوم بعمل Enabling للـ timer .

(2) الإنتظار المشغول Mutual Exclusion With Busy Waiting : وبها ما يسمى Lock - Variable وهى متغير يأخذ إحدى قيمتين هما (0 & 1) وأى عملية تحتاج للدخول فى المنطقة الحرجة الخاصة بها تقوم بالتأكد (check) من الـ Lock -Variable فإذا كانت القيمة (0) تعنى عدم وجود عملية أخرى فى هذه المنطقة بمعنى عدم وجود عملية أخرى تملك البيانات المشتركة وفى هذه اللحظة تدخل إلى المنطقة الحرجة وتقوم بتغيير القيمة إلى (1) وبعد الإنتهاء من مهمتها تقوم بتغيير القيمة إلى (0) . وإذا كانت القيمة (1) منذ البداية فإنها تنتظر لذلك نطلق عليه مسمى المشغول بالإنتظار Busy Waiting .

*** الإستعصاء : Deadlock : (الجمود)**

عبارة عن مجموعة من العمليات كل عملية تمتلك مصدر معين (source) وفى نفس الوقت تحتاج لمصدر آخر مملوك لعملية أخرى (تكون فى حالة إنتظار)

*** يحدث الإستعصاء عند توفر أربعة شروط معاً وهى :**

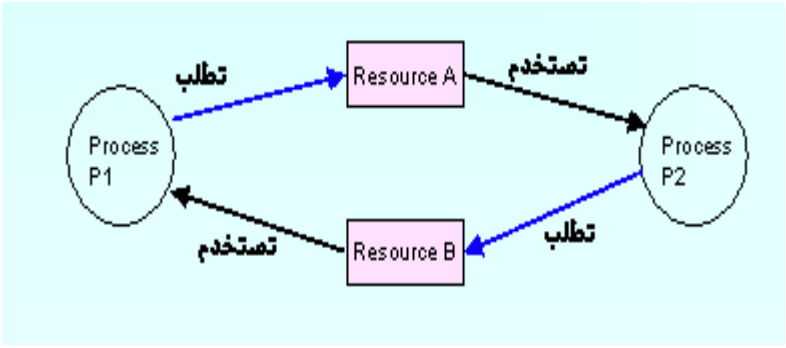
(1) المنع التبادلى Mutual Exclusion : بمعنى أن تستخدم المورد فى الوقت الحالى عملية واحدة فقط (كل مورد يرتبط حصرياً بعملية واحدة) .

(2) الاستخدام و الانتظار Hold and Wait : بمعنى وجود عملية تمتلك مورد معين ولكنها تحتاج لمورد آخر تمتلكه عملية أخرى حتى تنجز عملها .

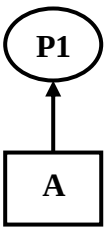
(3) عدم الإيقاف no preemption :(عدم المقاطعة) : بمعنى أن لا يتم إنتزاع المورد منها بصورة إجبارية وتركها تقوم بتحرير المورد بعد إكتمال التنفيذ

(4) الإنتظار الدائرى Circular wait : بمعنى أن تكون هنالك سلسلة دائرية من العمليات كل عملية تحتاج لمورد تمتلكه العملية التى تليها فى السلسلة .

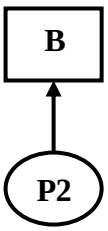
وفى هذه الحالة تقتل جميع العمليات . الشكل التالى يوضح الإنتظار الدائرى .



*** طريقة هولت للفراغات الموجهة :** (إتجاه السهم يحدد المرسل للطلب والمستقبل له)



العملية (p1) طلبت المورد (A) وحازت عليه .



العملية (p2) طلبت المورد (B) ولكنها لم تمتلكه بعد .

*** طرق التعامل مع الإستعصاء :**

(1) تجاهل الإستعصاء Ignore Deadlock :يعني تجاهله تماماً كما يحدث في نظام يونكس ويستخدم خوارزمية النعامة Ostrich Algorithm

(2) الكشف وعدم السماح Detection & Recovery : بمعنى أن نسمح له ونمنع حدوثه مستقبلاً بالقتل (killing) أى قتل العملية المسببة له وهو مكلف أو قتل عملية واحدة لكسر الـ cycle ونستخدم هنا خوارزمية تسمى graph .

توضيح : مثلاً إذا كان لدينا عمليتان تسببتا في المشكلة ولتتم المعالجة نقوم بقتلهما .

(3) تجنب الإستعصاء Deadlock Avoidance : منع النظام من الدخول فيه ويتم ذلك بشروط الإستعصاء وتستخدم هنا خوارزمية تسمى Banker's Algorithm . (وهى خوارزمية تمنع حدوث أى شرط من شروطه) .

محاضرة رقم(9)

إدارة الذاكرة : Memory Management

إحدى وظائف نظم التشغيل إدارة موارد الحاسوب ومن أهمها الذاكرة الرئيسية لأنها المكان الوحيد الذي منه تستدعى الـ CPU إيعازات البرامج والبيانات المراد تنفيذها والجزء من نظام التشغيل الذي يقوم بإدارة الذاكرة هو مدير الذاكرة Memory Manger ومن مهامه:

(1) مراقبة حالة جميع مواقع الذاكرة من حيث :

- ✓ المواقع الفارغة وذلك لتسكين العمليات المراد تنفيذها .
- ✓ المواقع الممتلئة من اجل تفريغ المواقع بعد إنتهاء العمليات من التنفيذ .

(2) تخزين الطريقة التي من خلالها يتم توزيع المواقع الفارغة للعمليات المراد تنفيذها مع تحديد الأولويات في التسكين ,

(3) نقل العمليات التي يتم تنفيذها من الذاكرة الرئيسية إلى الذاكرة الثانوية أو العكس .

ملحوظة : في حالة التنفيذ من الـHD إلى الـMain وفي حالة الإنتهاء (terminate) من الـMain إلى الـHD .

*(إدارة الذاكرة : المواقع الفارغة هل تخصص لعملية واحدة أم توزع على عدة عمليات و إذا كان الخيار الثاني فهل تقسم المواقع بالتساوي أم حسب حاجة العملية ؟؟)

** تقسم نظم التشغيل إلى نوعين هما :

(1) أحادية البرامج Mono Programming

(2) متعددة البرامج Multi Programming

أحادية بمعنى عملية واحدة في نفس الوقت (الوقت الواحد) إدارة الذاكرة هنا بسيطة وذلك لأن الأسلوب المتبع هو تخصيص جميع الموارد لعملية واحدة وباقي العمليات تكون في حالة خمود حتى تنتهي العملية الأولى .

العملية ***Process***
حيز غير مستخدم
Operating system

** عيوب نظم التشغيل الأحادية :

(1) عدم الإستغلال الجيد للذاكرة (دائماً توجد مساحات غير مشغولة)

(2) عدم الإستغلال الجيد للـCPU و لأنها تتوقف عن العمل وتراقب الـI /O Devices .

(3) طول مدة إنتظار العمليات الأخرى .

** تقسم الذاكرة إلى نوعين هما :

(1) الذاكرة المتطايرة Volatile Memory مثال لها Main Memory .

(2) الذاكرة غير المتطايرة Non Volatile Memory مثال لها Hard Disk .

وتقسم غير المتطايرة إلى :

- On line مثل الـ hard disk
- Off line مثل الأقراص المغناطيسية Magnetic tape والشرائط المغناطيسية magnetic disk .

** إدارة الذاكرة بالنسبة للبرامج المتعددة وفق منهجيتين هما :

الأول : التقسيم الثابت Memory Management with fixed partition

هو تقسيم الذاكرة إلى قطاعات ثابتة على أن يخصص لكل عملية القطاع المناسب لها ولكن قد تحدث بعض المشاكل هى :

(1) يمكن أن يكون هناك مساحات أو قطاعات غير مشغولة وذلك لعدم وجود عملية بنفس الحجم بالرغم من وجود عمليات في الـWaiting Queue

(2) يمكن أن يتم تسكين عملية في قطاع أكبر من حجمها وذلك لعدم وجود الحجم المناسب لها ولهذا يؤدي وجود مساحات غير مشغولة داخل القطاع وهذا ما يعرف بـInternal Fragmentation (التجزئة الداخلية) ولحل هذه المشكلة توضع جميع العمليات في صف واحد ويعامل الصف كـ FIFO ويقوم الـMemory Manger (مدير الذاكرة) بالتأكد من جميع القطاعات غير المشغولة ثم يختار أنسب قطاع لهذه العملية من حيث الحجم وعند وجود عملية صغيرة لا يوجد قطاع مناســـب لها فإنه يتخطاها إلى عملية أخرى ثم يعود لها مرة أخرى وهذه المشكلة قد تخلق Starvation للعملية ولحل هذا :

(أ) يجب أن يوضع عدد معين من التجاوزات (skips) لتخطى عملية إلى عملية أخرى أي لا يتم تجاوز عملية أكثر من (8) مرات مثلاً .

(ب) يكون هناك Partition صغيرة مخصصة فقط للعمليات الصغيرة .

* هنالك ما يسمى بالتجزئة الخارجية External Fragment وتكون بين Partition & Partition آخر ... أي مساحات لا يمكن استغلالها

س| إذا كان لدينا عملية حجمها 50 ك ب هل يمكن أن يتغير حجمها فى أى لحظة ؟ نعم إذا كانت مشاركة للبيانات وإذا قامت بتوليد عمليات أخرى

توضيح: من المعلوم أن العملية يمكن ان تنتج عملية أخرى و أيضاً يمكن أن تحتاج لبيانات أثناء التنفيذ و بالتالي حجمها غير ثابت فماذا يحدث إذا زاد حجم العملية عن حجم القطاع الذي توجد به ؟؟؟

* هناك خيارات :

(1) عند تسكين العملية يجب مراعاة أن حجم العملية غير ثابت بمعنى تسكينها في قطاع أكبر بقليل من حجمها . لأنها قد تزداد مستقبلاً .

(2) البحث عن قطاع آخر مناسب لها و إذا لم توجد يتم نقل العملية من Main Memory إلى الذاكرة الثانوية وإذا لم يوجد لها مكان في الـــ H.D تلجأ بعد ذلك إلى الـــ O. S إلى ما يسمى بـProcess Killing لتفادي حدوث Dead Lock للنظام ككل .

* هناك عدة إستراتيجيات يستخدمها الـــMemory Manger لتحديد القطاع المعين لتسكين العملية .

(1) First Fit (المكان الأول) أو الملائمة الأولى : أي تخزين العملية في أول قطاع غير مشغول يسع العملية بغض النظر عن حجم القطاع .

(2) Second OR next Fit (المكان التالي) أو الملائمة الثانية : أى التخزين فى الموقع الذى يلى آخر عملية تخزين بغض النظر عن الحجم .

(3) Best Fit (المكان الأفضل) أو الملائمة الأفضل : أى التخزين فى أنسب قطاع غير مشغول في الذاكرة من حيث الحجم .

(4) Worst Fit (المكان الأكبر) أو الملائمة الأسوأ : أي تخزين العملية في أسوأ مكان لها من حيث الحجم . (فى المكان الأكبر دون النظر لحجم الملف)

(5) Quick Fit (المكان الأسرع) أو الملائمة الأسرع : هي نفس الإستراتيجية الثالثة والفرق هنا أن مدير الذاكرة يقوم بعمل جدول فيه جميع عناوين القطاعات غير المشغولة وبذلك تكون عملية البحث أسهل بكثير .وبالتالى يتم النظر لحجم الملف أساساً .

الثانى : التقسيم الديناميكى Memory Management with Variable Partition

في هذه الطريقة تقسم الذاكرة إلى قطاعات حسب حوجة العمليات أي حسب حجمها و تعطي كل عملية حجم أكبر بقليل من الحجم المطلوب تجنباً لنمو العملية ، وفي هذه الطريقة توجد مشكلة External Fragment ولكن هذه الفراغات تكون موجودة لفترة بسيطة وذلك لأنه عند إنتهاء العملية من التنفيذ يقوم مدير الذاكرة بدمج هذا الفراغ مع الجزء الذي كانت تشغله العملية .

يقوم نظام التشغيل بتجميع المساحات الفارغة مع بعضها البعض حتى لا يحدث External Fragment وذلك بجعل جميع الـProcess في اسفل الذاكرة و المساحات الفارغة في الأعلى و تسمى هذه العملية بـــ Memory Compact .

س| كيف يستطيع نظام التشغيل معرفة الأجزاء الفارغة والمشغولة ؟؟؟

بما يسمى بحالة المراقبة الدائمة Keeping Track وتكون هذه المراقبة للقطاعات المشغولة والفارغة .

س| ما ذا يحدث إذا كانت التقسيمات :

1| أكبر من العمليات (مثلاً العمليات 50 والتقسيمات 489) ؟ 2| أصغر من العمليات (مثلاً العمليات 489 والتقسيمات 50) ؟

* فى الحالة الأولى يقوم النظام بتجاهل العمليات الصغيرة (تخطى) لعدد معين من المرات ويقوم بتخزين العمليات الكبيرة أولاً . وقد تحدث الـ Starvation

* فى الحالة الثانية يقوم النظام بتوفير التقسيم للعملية الأصغر وبعد الإنتهاء يقوم بتحرير الذاكرة للعملية التالية وهكذا وفى النهاية يقوم بتخزين العملية الأكبر التى حجمها 489 بالتخزين كاملة .

* حالة المراقبة الدائمة Keeping Track : : تتم المراقبة بطريقتين هما :

(1) Memory Management with Bit map : (الخارطة النقطية)

في هذه الطريقة يتم تقسيم الذاكرة إلى قطع صغيرة تسمى كل قطعة Allocation bit بحيث تكون هنالك خارطة تمثل هذه القطع الصغيرة .

- يقوم الـS.O بوضع الرمز (1) مقابل كل Bit به Process والرمز (0) مقابل كل Bit خالية .
- وعليه ينظر مدير الذاكرة إلى الـ Bit Map لتحديد المساحات الفارغة .إذا إنتهت الـProcess من التنفيذ فإنه يقوم بتحويل موقعها في الـMap من (1) إلى(0) ،، و غالباً ما تكون الأجزاء بين 0.5 kb & 1.0 kb .

1	1	0	1	1	1	1
1	0	1	1	0	0	1
0	0	0	0	0	0	0
1	1	1	0	1	1	1
1	1	0	1	0	0	0
0	0	0	0	0	0	0
1	1	1	1	1	1	1

(2) Memory Management with Linked List : (القوائم المتصلة)

في هذه الطريقة يتم تمثيل كل القطاعات إذا كانت مشغولة أو فارغة بواسطة القوائم المتصلة ،،، يتم تقسيم الـ List إلى أربعة أجزاء كالآتى :

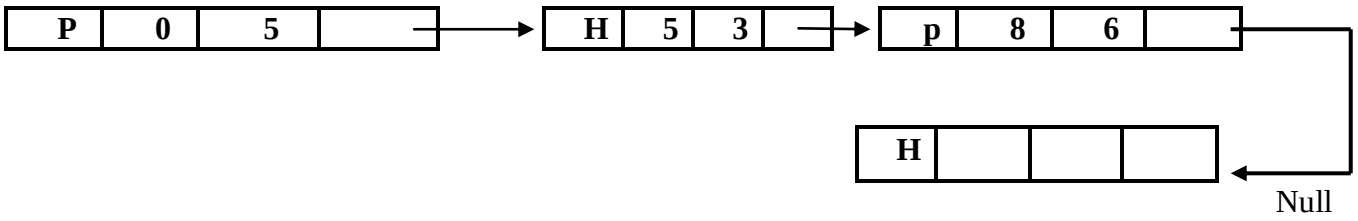
process (p /h)	address	size	pointer

من اليسار:

- الأول يشير إلى وجود عملية أو عدمه حيث تعطى (p) للدلالة على Process وتعطى (h) للدلالة على عدم وجودها (جرف) hole
- الثاني يشير إلى العنوان بمعنى بداية هذا الجانب من الـ Memory .
- الثالث يشير إلى الحجم .
- الرابع عبارة عن مؤشر يشير إلى الـList الذي يليه .

مثال : إذا كان لدينا التقسيم التالى بالذاكرة كيف نعبر عن هذه التقسيمات بالقائمة المتصلة ؟

الحل :



(3) Memory Management with Buddy system :

يتم تقسيم حجم الذاكرة فى كل مرة إلى قسمين مثلاً (1000) تصبح (500, 500) وهكذا وهذه الـ Buddy يجب أن تكون من مضاعفات العدد إثنين (2 , 4 , 16) . ويتم ذلك فى كل مرة حتى نصل إلى أقل حجم يمكننا من تخزين العملية

1024			
512		512	
128	128		
64	64		

محاضرة رقم (10) المحاضرة الأخيرة

إدارة أجهزة الإدخال والإخراج : I / O Devices Management

تعريف : أجهزة الإدخال والإخراج هي المكونات المادية للحاسوب ,

تنقسم أجهزة الإدخال و الإخراج إلى فئتين :

(1) فئة معنونة Block Devices

(2) فئة غير معنونة Character Devices

* الفئة الأولى هي عبارة عن أجهزة يمكن تخزين البيانات بها في كتلة ثابتة الحجم و كل كتلة لها عنوان خاص ، بالإضافة إلى إمكانية إجراء عمليات القراءة والكتابة و البحث و الإنتقال (كمثال القرص الصلب) .

* الفئة الثانية ترسل و تستقبل سلسلة من الإشارات لا يمكنها تخزين البيانات بها بالإضافة إلى عدم إمكانية إجراء أي من عمليات القراءة و الكتابة و البحث أو الإنتقال كمثال (الطابعة)

أي جهاز إدخال أو إخراج يتكون من جزئين :

(1) المكون المادي (الميكانيكي) Mechanical Component

(2) المكون المنطقي (البرمجي) Electronic Component

هي ما يسمى بالجزء الإكتروني أو المتحكمات Devices Controller وهو الجزء الذي يوفر التفاعل المنطقي مع الحاسب بحيث يتلقى الأوامر من الـCPU ثم بعد ذلك يقوم بترتيبها وتنفيذها .

Why Electronic Component ?

لأن الـI / O لا يتعامل إطلاقاً من الجزء الـ Mechanical مباشرة بمعنى إنه يتعامل مع هذا الجزء من خلال الـــController وهو الذي يقوم بالإتصال مع الجزء المادي .

* مثال عملية القراءة من الـ Block Devices كالـ Disk مثـــلاً حيث أن الـ I / O يعطي أمر للـ Controller لقراءة Block معين والـــController يعطي القراءة كـــ Bit characters و يقوم بتخزينها في Buffer موجود فيه حتى يكتمل الـــBlock وبعد ذلك يقوم بعمل Interrupt يوضح فيه بأن الـBlock جاهز للقراءة وقبل ذلك يتاكد من أن الـــBlock أصبح خالياً عن طريق Checksum ومن ثم يتم نقله للـــMemory ويقوم بهذا النقل للـــCPU من خلال أمر من الـــ I / O وهذا يقود إلى Low Lower task of CPU أو Overhead task (وقت كبير يهدر في نقل الـــBlock ولذلك جاءت تقنية أخرى تعرف بـ (Direct Memory Access (DMA حتى لا يحدث Overhead task وفي هذه التقنية يعطى الـــ Controller الصلاحية بنقل الـــBlock كاملاً .

- أهداف برمجيات أجهزة الإدخال و الإخراج :

(1) الإستقلالية عن الأجهزة Device Independence

نعني بذلك إمكانية كتابة برنامج يستطيع الوصول إلى أن جهاز إدخال غو إخراج دون تحديد نوع الجهاز بشكل مسبق .

■ مثلاً إذا كان لدينا برنامج مكتوب لقراءة ملف يجب أن يتمكن البرنامج من قراءة الملف من أي جهاز تخزين بمعنى يمكن أن يقرأ الملف من القرص الصلب او القرص المبرمج او القرص المرن أو غيرها من وسائط التخزين دون التعديل في البرنامج .

(2) معالجة الخطاء Error Handling

بمعنى معالجة الخطأ قريباً من الجزء المادي لأجهزة الإدخال و الإخراج إذ على الـ Devices Controller إكتشاف الأخطاء و محاولة إصلاحها بقدر الإمكان وإذا لم تستطيع فإنها تقوم بمناداة الـــ I / O لمعالجة الوضع ويتم ذلك بعرض المشكلة على الطبقة العلى من المتحكمات فإن لم تستطيع الطبقة العليا معالجة الخطاء يقوم الـــI / O بعرضها على الطبقة العلى و هكذا إلى أن تصل للمستخدم في شكل رسائل الخطأ .

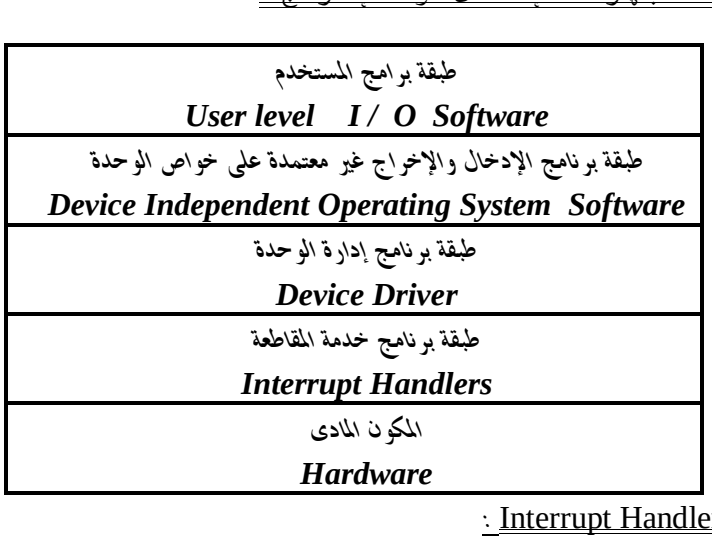
(3) التزامن Synchronization :

هو من المواضيع المهمة في تصميم برمجيات أجهزة الإدخال و الإخراج لأن معظم الـــ I / O غيرمتزامن مثلاً إذا كان المعالج يقوم بتنفيذ عملية و هذه العملية إحتاجت إلى I / O فعندئذ يقوم المعالج بتنفيذ عملية آخرى حتى يصل I / O للعملية السابقة و يتم غرجاع المعالج لها بصورة تلقائية حيث يجب على نظام التشغيل جعل العمليات التي تكون في الواقع مقادة بالمقاطعات (غير متزامنة) تبدو و كانها متزامنة للمستخدم .

(4) Sharable / Dedicated Devices :

أيضاً من المفاهيم الهامة في تصميم برمجيات اجهزة الإدخال والإخراج هي إمكانية التعامل مع الأجهزة التي يمكن المشاركة فيها مثل الأقراص حيث يمكن إستخدامها من قبل عدة مستخدمين في نفس الوقت وهنالك أجهزة مثل سواقات الأشرطة يجب أن تخصص لمستخدم واحد حتى ينتهي من إستخدامها وعندها يستطيع مستخدم أخر إستخدامها .

- مستويات برمجيات أجهزة الإدخال و الإخراج :



(1) طبقة برنامج حماية المقاطعة Interrupt Handlers :

يجب دائماً إخفاء المقاطعات بعيداً عن المستخدم و الطريقة الفضل لإخفائها هي جعل برنامج التشغيل الذي يبدأ عملية الإدخال و الإخراج يتوقف ريثما تنتهي عملية الدخل أو الخرج ، وقد تكون هذه العملية لأي وحدة من الوحدات و بالتالي تكون هناك أنواع متعددة للقطع و كل قطع له برنامج خدمة محدد موجود ضمن خدمات نظام التشغيل أو النظام الأساسي للإدخال و الإخراج (BIOS) و يتم الرجوع لهذا البرنامج بعد تحرير نوعية القطع وموضوع البرنامج الذي يقوم بهذه الخدمة .

(2) طبقة برنامج إدارة الوحدة Device Driver :

الوظيفة الأساسية لهذا البرنامج هي إستقبال الطلبات من البرامج في الطبقة البرمجية العليا ثم بعد ذلك تنفيذ هذه الطلبات بالتنسيق مع وحدة التحكم .

(3) طبقة برنامج الإدخال و الإخراج غير المعتمدة على خواص الوحدة I / O Device Independent O . S Software

تتلخص وظائف هذه الطبقة في الأتي :

i -تعتبر طبقة تداخل بين المستخدم و طبقة إدارة الوحدة .
ii -تسمية المكونات المادية Device Naming .
iii -حجز الذاكرة المؤقتة في الوحدة Buffering
iv -حجز سعة تخزينية في وحدات الإدخال و الإخراج .
v -تحديد الأخطاء .

(4) طبقة برامج المستخدم User Level I / O Software :

و هي الطبقة التي تقدم خدمات للمستخدم لإستخدام أجهزة الأدخال و الإخراج حيث يقوم المستخدم بطلب الخدمة من برامج هذه الطبقة والتي بدورها تحول الطلب إلى الطبقات البرمجية الأدنى ثم الأدنى حتى يتم تنفيذ الخدمة (Interface مع User) .

* طرق إنجاز عمليات الإدخال و الإخراج :

(1) الإدخال و الإخراج المبرمج Programmed I / O :

تتلخص الفكرة في جعل المعالج يقوم بالعمل كله . فمثلاً إذا كانت هنالك عملية نريد طباعة سلسلة من الحروف فإنها تقوم بتجميع السلسلة في مخزن مؤقت Buffer ثم بعد ذلك يقوم الـ O /S بنسخ المؤقت Buffer الذي يحول السلسلة في مصفوفة و بفحص ما إذا كانت الطابعة متاحة حالياً فإذا لم تكون متاحة فإنه يتم الإنتظار حتى تصبح متاحة و عندها يتم طباعة الحرف الأول يقوم بفحصها مرة أخرى فإذا كانت غير جاهزة فإنه ينتظر حتى تصبح متاحة و يستمر بعمل ذلك حتى تصبح متاحة و يستمر بعمل ذلك حتى تتم طباعة السلسلة بأكملها .(إهدار لوقت الـ CPU بين طباعة كل حرف و آخر)

(2)الإدخال و الإخراج المقاد بالمقاطعات :

هذه الطريقة تتيح للمعالجات تنفذ عمليات آخرى أثناء إنتظاره للطابعة كي تصبح جاهزة .

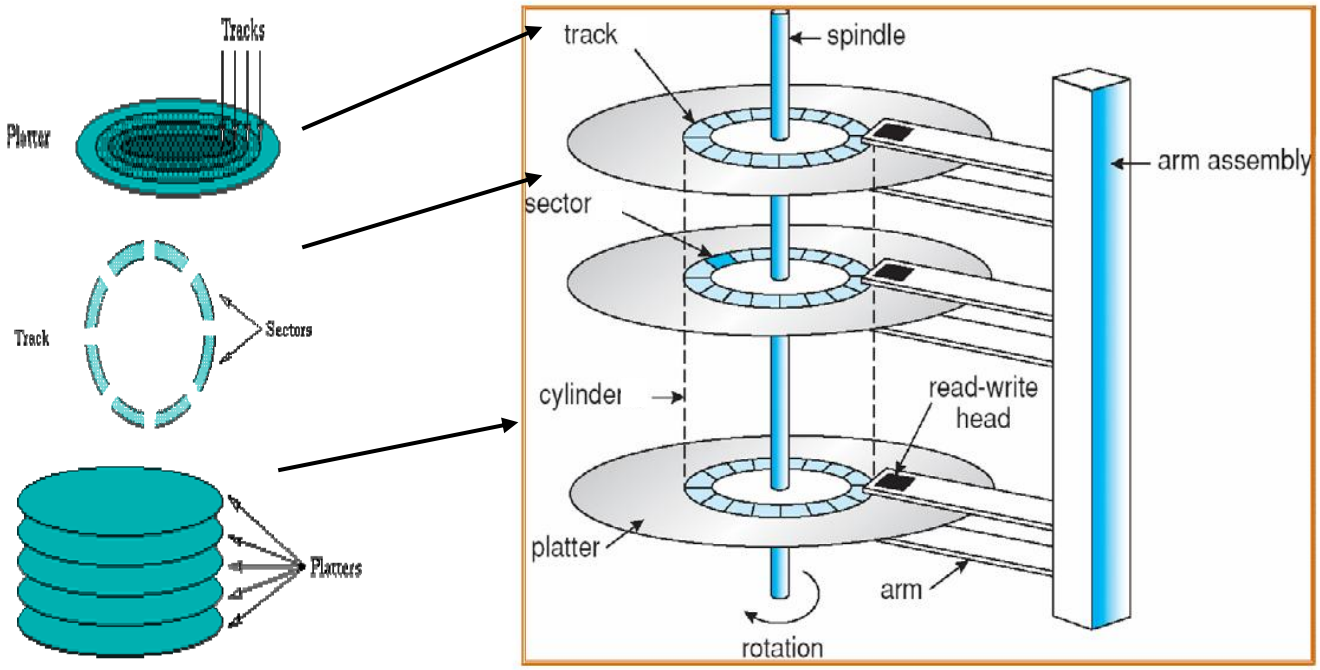
(3) الإدخال و الإخراج بإستخدام تقنية الـDMA :

تعتبر الطريقة السابقة مضيعة لوقت المعالج لأنه يقوم بعملية Content Switch متكررة لذلك الحل الفضل هو إستخدام تقنية DMA .

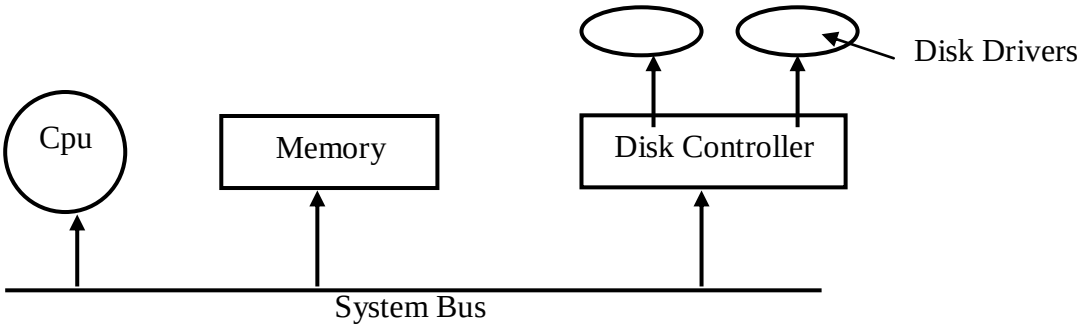
تتلخص الفكرة في جعل الـ Controller يقوم بنقل الحروف واحداً واحداً بالتتالى بعيداً عن المعالج .

الفائدة الأساسية عند إستخدام تقنية DMA هي عملية نقل الـController للبيانات مباشرة من الـBuffer الموجود بداخله إلى الذاكرة الرئيسية و بذلك يقل حدوث المقاطعات .

معمارية القرص الصلب : الشكل داخل المربع يوضح معمارية القرص الصلب والأشكال الجانبية لتكبير الأجزاء المشار إليها



يتكون القرص الصلب من مكونين :



(1) سواقة الأقراص Disk Drivers تعتبر المكون المادي التخزيني فهي التي تحتوي على القرص و الرؤوس و كل الأجزاء الميكانيكية الأخرى .

(2) Disk Controller وتمثل الوحدة الإلكترونية التي توفر التفاعل المنطقي مع الحاسب بحيث تقوم بتلقي الأوامر من وحدة المعالجة المركزية و بعد ذلك تقوم بتحويلها للـDriver .

فعلى سبيل المثال إذا أردنا إستخراج بيانات من القرص الصلب فإننا نحتاج إلى تحديد عدة أشياء هي رقم السواقة والسطح و المسار و الـ Block ثم بعد ذلك يقوم رأس القراءة بالتحرك نحو الـBlock ويسمى الزمن المستغرق في تحديد المقطع بزمن التحديد Seek Time أما الزمن المستغرق حتى دوران القرص التخزيني يسمى زمن الإنتظار Latency time

**إدارة الفراغات التخزينية Free Space Management :

(1) الخارطة النقطية Bit map : بنفس طريقة الخارطة النقطية في إدارة الذاكرة .

(2) القوائم المتصلة Linked List ربط المربعات الفارغة بواسطة مؤشرات فأول مربع فارغ يحتوي على عنوان المربع الفارغ التالي له مباشرة .

(3) التجميع Grouping في هذه الطريقة يتم تجميع عناوين مجموعة من المربعات الفارغة في أول مربع فارغ وهي طريقة سهلة في الحصول على مجموعة من المربعات الفارغة .

* * جدولة القرص الصلب Disk Scheduling

عندما تطلب عملية موقعاً تخزينياً تقوم العملية بإستدعاء نظام التشغيل بواسطة SYS Call حيث يتم تحديد عدة عوامل هي :

(1) نوع العملية إدخال / إخراج .

(2) عنوان الموقع التخزيني (سواقة – إسطوانة – سطح – قطاع)

(3) عنوان الذاكرة المراد تحميل البيانات منها و إليها .

(4) حجم البيانات المراد نقلها .

هنالك عدة خوارزميات يتبعها نظام التشغيل لجدولة و خدمة هذه الطلبات منها :

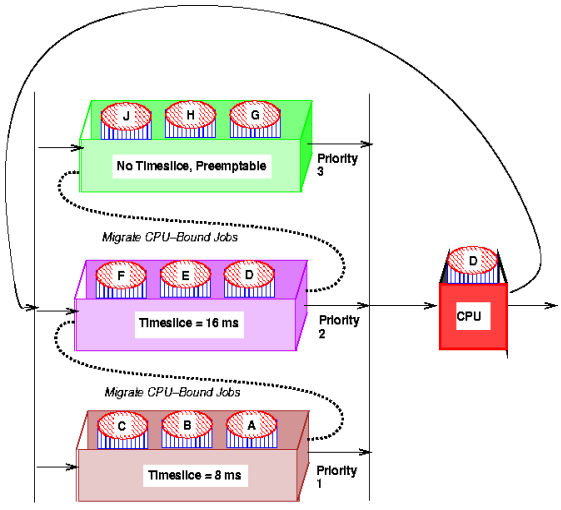
(1) خوارزمية First Come First Serve [FCFS]

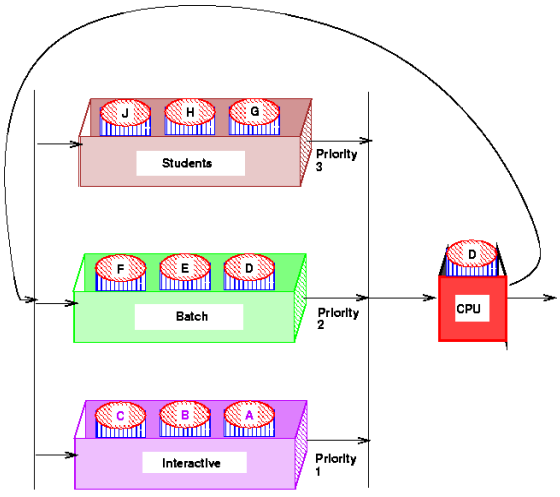
(2) خوارزمية تحديد أقل زمن Shortest Seek Time First [SSTF]

يتم خدمة العملية التي تستغرق زمن أقل فعلى سبيل المثال إذا كان رأس القراءة موجود في أسطوانة معينة وهناك عدة عمليات تطلب مواقع تخزينية مختلفة وفي أسطوانات مختلفة عندها يبحث نظام التشغيل عن العملية التي تطلب معلومة موجودة في نفس الأسطوانة الحالية أو الأقرب.

تم بحمد الله وتوفيقه

تنبيه للتوضيح فقط : الرسم التالي يوضح صفوف التغذية المرتدة Multilevel Feedback Queues وذات المستوياتMultilevel Queues (تابع المذكرة السابقة خوارزميات الجدولة) الرسم من اليمين ذات المستويات ومن اليسار التغذية المرتدة .





((اللهم صلى وسلم على سيدنا محمد الأمين بقدر ما خط القلم فى الورق وبقدر ما أشرق نور أو برق))

يارَبْ عَلمّني أنْ أحبّ النَاسْ كَما أحبّ نَفسْي
وَعَلّمني أنْ أحَاسِبْ نَفسْي كَما أحَاسِبْ النَاسْ
وَعَلّمْني أنْ التسَامح هَو أكْبَر مَراتب القوّة
وَأنّ حبّ الانتقام هَو أولْ مَظاهِر الضعْف.
يارَبْ لا تدعني أصَاب بِالغرور إذا نَجَحْت وَلا باليأس إذا فُشلت
بَل ذكّرني دائِماً أن الفَشَل هَو التجَارب الَتي تسْبق النّجَاح.
يـارَبْ لئن سألتني عن ذنبي يوم القيامة , لأسألنك عن رحمتك
و لئن سألتني يا رب عن تقصيري , لأسألنك عن عفوك

---------- صدى نوفمبر 2012م

(لاتنسونى من صالح دعاؤكم لى ولوالداى)

osmansada@gmail.com