Université Sultan Moulay Slimane

Ecole Supérieure de Technologie de Béni Mellal

Département : ITG
Filière : ARI - *2ème année - S3*
Année universitaire: 2023-2024

Module: Sys. & Réseaux Informatiques Avancés (M11)
Elément: Systèmes d'exploitation avancés (E2)

Support de cours

Systèmes d'exploitation Avancés

Chapitre IV: Gestion de la mémoire secondaire : fichiers

Pr. Siham BAKKOURI

Email: s.bakkouri@usms.ma

Plan

- INTRODUCTION
- SYSTÈME DE GESTION DE FICHIERS
  - → Les fichiers logiques
  - → Les fichiers physiques
- LES RÉPERTOIRES
- LES OPÉRATIONS SUR LES FICHIERS
- PROTECTION DES FICHIERS

Introduction

Le S.E. doit pouvoir stocker à long terme de grandes quantités d'informations : en premier, ses propres modules, ensuite les programmes d'application, les bibliothèques de fonctions, les programmes et les données des utilisateurs.

→ **Les supports physiques** : **mémoire secondaire** :

- Grande capacité
- Faible coût

→ **Fonctions**

Organiser et gérer la mémoire secondaire en particulier faire la correspondance entre un fichier logique et un fichier physique.

Introduction

→ Définitions

- Un **Fichier** est une suite d'articles ou d'enregistrements spécifiques numérotés chacun avec des numéros différents et pouvant comporter chacun plusieurs champs.

- **Fichier physique :** fait référence à la représentation concrète des données sur un support de stockage. Il s'agit de l'implémentation matérielle de l'information, telle que les bits et les octets enregistrés sur un disque dur, une clé USB ou tout autre support physique.
- **Fichier logique :** est un concept abstrait qui représente l'ensemble des informations telles qu'elles sont perçues par l'utilisateur. Il s'agit d'un ***identificateur*** qui englobe à la fois le fichier physique et sa représentation logique. Il permet à l'utilisateur de manipuler les données de manière cohérente et compréhensible, indépendamment de leur localisation physique réelle.

Système de gestion des fichiers : Les fichiers logiques

Dans le contexte de la gestion des fichiers, différentes opérations de manipulation sont couramment utilisées. Parmi ces opérations:

→ **Les opérations de manipulation** : créer, ouvrir, fermer, détruire, éditer, copier et renommer.

→ **Les opérations d'accès** : lire, écrire, insérer, détruire et retrouver.

❑ les modes d'accès

Le jeu d'instructions permettant d'accéder aux informations d'un fichier définit la structure logique dont les utilisateurs disposent pour organiser le fichier. Cette structure appelée le mode d'accès aux fichiers varie d'un système à l'autre. Certains S.E. ne fournissent qu'une seule structure logique et donc qu'un seul **mode d'accès**.

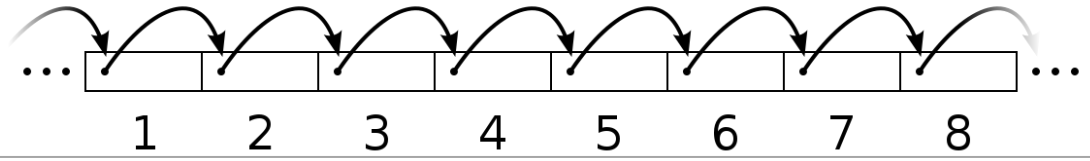
Système de gestion des fichiers : Les fichiers logiques

→ Mode accès: Séquentiel

- **L'accès séquentiel** est un mode d'accès dans lequel les numéros d'ordre des articles ne peuvent être utilisés que pour lire (ou écrire) l'article pointé, accéder à l'article suivant ou revenir au début du fichier.

→ ***Par exemple***, lors de l'opération de lecture, le contenu de l'article courant est lu et le pointeur est ensuite positionné sur l'article suivant.

- **Utilisation**: grand fichier et mises à jour peu fréquentes.



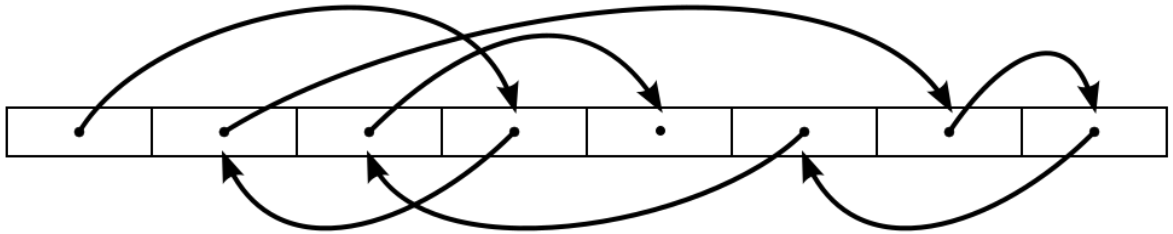
Il est **important** de **noter** que l'accès séquentiel est un mode hérité de l'époque où les fichiers étaient physiquement stockés sur des bandes magnétiques.
Dans ce mode, si la tête de lecture/écriture se trouve sur **l'article 5**, elle doit se déplacer successivement sur les articles 6, 7, 8 et 9 pour accéder à **l'article 10**.

Système de gestion des fichiers : Les fichiers logiques

→ Mode accès: Indexé

- **L'accès indexé** permet un accès direct à un article en utilisant des champs spécifiques appelés **clés**.
- Dans ce mode d'accès, on peut utiliser une **clé** pour accéder directement à l'article souhaité.
- Un cas particulier de l'accès indexé est **l'accès direct**, où la **clé** d'un article est **son numéro d'ordre**. En utilisant cette clé, il est possible d'accéder directement à l'article concerné.

→ ***Par exemple***, l'opération "*lire n*" permet d'accéder directement à l'article correspondant en utilisant la clé "*n*".



Système de gestion des fichiers : Les fichiers physiques

→ **Organisation:** implantation des articles

- **L'organisation des articles** dépend de la taille des informations lues ou écrites en une seule opération par le périphérique d'E/S, également appelé "bloc" et fixé par le matériel.

  → Dans le cas de UNIX/LINUX, les fichiers sont simplement des séquences d'octets, où chaque octet peut être adressé individuellement. Les enregistrements logiques ont une taille d'un octet. Ces octets sont regroupés par le système dans des blocs sur le disque, *par exemple*, 512 octets par bloc.

- En connaissant les tailles des articles et des blocs, il est possible de déterminer le nombre d'articles regroupés dans un bloc.

- L'organisation en zones de taille fixe entraîne naturellement le phénomène de **fragmentation** interne, où l'espace dans un bloc peut être partiellement utilisé, créant ainsi un gaspillage d'espace de stockage.

Système de gestion des fichiers : Les fichiers physiques

→ **Gestion:** Méthodes d'allocation de mémoire secondaire

Le S.E. doit trouver un espace libre, cela va dépendre de l'allocation contiguë ou non

1. Gestion des blocs libres

Il existe deux méthodes couramment utilisées pour gérer les blocs libres dans un système de fichiers :

a. **Vecteur de bits (bitmap)** : Un vecteur de bits est utilisé, contenant autant de bits que de blocs présents sur le disque. Chaque bit est associé à un bloc et est défini à **0** si le **bloc** est **libre**, et à **1** s'il est **occupé**.

b. **Liste chaînée** : Chaque bloc libre contient le numéro du bloc suivant dans la liste.

- **l'inconvénient** de cette méthode est que la recherche de ***n*** blocs libres nécessite le parcours de ***n*** blocs dans la liste.
- **Une amélioration** possible consiste à inclure dans le premier bloc de la zone **libre** à la fois le **nombre** de blocs libres disponibles et le numéro de la prochaine zone. Cela permet de réduire le nombre de blocs à parcourir pour trouver des blocs libres.

→ **Gestion:** Méthodes d'allocation de mémoire secondaire

2. Allocation contiguë

L'allocation contiguë est une méthode d'allocation de mémoire dans laquelle les fichiers ou les données sont stockés de manière contiguë sur le disque. Cette méthode utilise les méthodes de placement déjà présentées (*First fit, best fit et worst fit*), ainsi que des procédés de compactage.

❑ **Les avantages :**

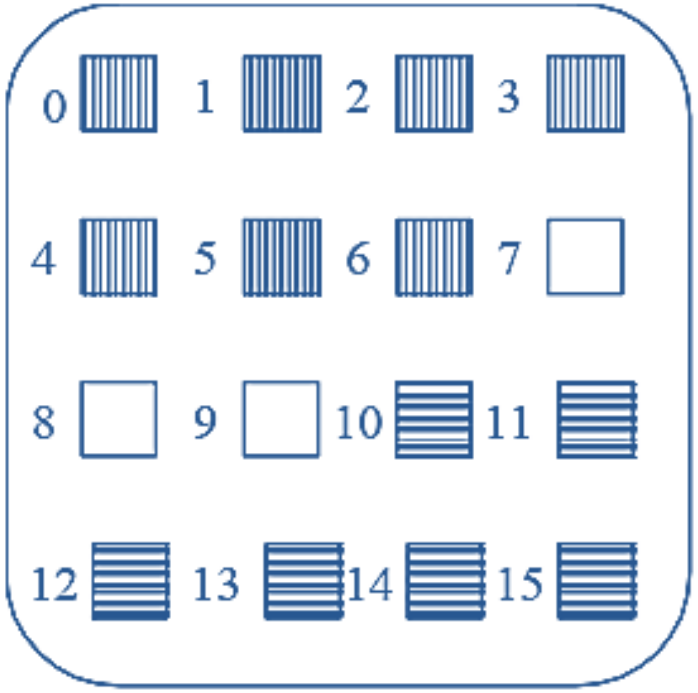
- Accélère les opérations de recherche.
- Implantation simple, car il suffit de maintenir un suivi de l'espace libre et occupé sur le disque.

❑ **Inconvénients :**

- Fragmentation interne et externe
- À la création d'un fichier, l'utilisateur doit spécifier sa taille pour trouver un emplacement approprié. Si aucun espace suffisamment grand n'est disponible, une réorganisation du disque peut être nécessaire pour libérer un bloc continu.
- Lorsqu'un fichier devient trop grand pour tenir dans son emplacement actuel, il doit être déplacé pour trouver un nouvel emplacement contigu.

→ **Gestion:** Méthodes d'allocation de mémoire secondaire

2. Allocation contiguë



Table

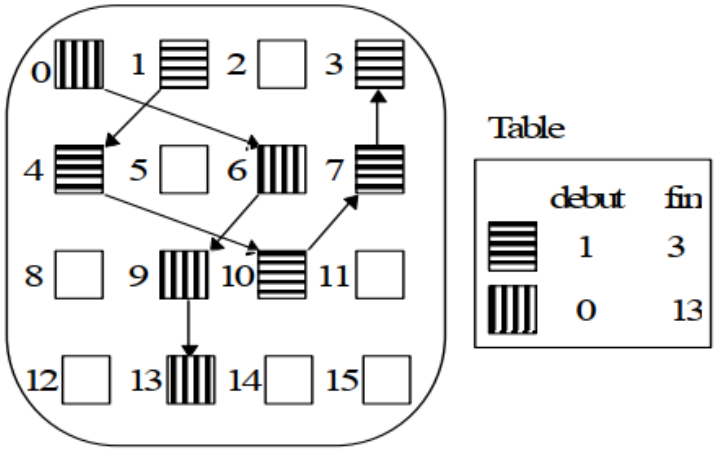
	debut	fin
	0	6
	10	15

→ **Gestion:** Méthodes d'allocation de mémoire secondaire

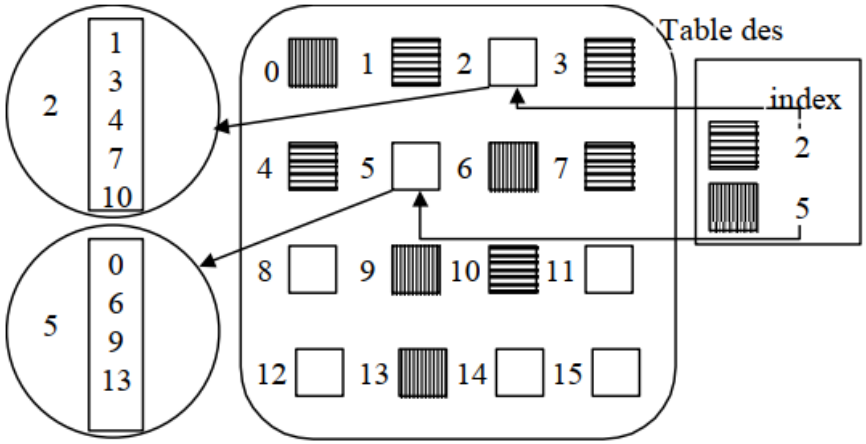
3. Allocation non contiguë

Dans le contexte de l'allocation non contiguë, il existe **deux** méthodes couramment utilisées : **l'allocation chaînée** et **l'allocation indexée**.

Allocation chaînée



Allocation indexée



→ **Gestion:** Méthodes d'allocation de mémoire secondaire

3. Allocation non contiguë

3.1. Allocation chaînée

les blocs constituant un fichier sont liés entre eux par des pointeurs. Chaque bloc contient le numéro du bloc suivant dans la chaîne.

❑ **Les avantages**

- Permet d'ajuster facilement la taille des fichiers.
- Pas besoin de déclarer la taille du fichier à sa création, car les blocs sont liés dynamiquement.
- Pas de besoin de procéder à un compactage pour libérer de l'espace fragmenté.

❑ **Les inconvénients**

- L'accès aux données se fait de manière séquentielle.
- La mise à jour des pointeurs lors de la modification d'un fichier peut être coûteuse en termes de temps et de ressources.

→ **Gestion:** Méthodes d'allocation de mémoire secondaire

3. Allocation non contiguë

3.2. Allocation indexée

Dans le contexte de l'allocation indexée, une table appelée "***bloc d'index***" est conservée dans un bloc spécial du disque. Ce bloc d'index contient la liste des numéros de blocs utilisés par chaque fichier.

❑ **Les avantages**

- Permet d'accéder directement à n'importe quel bloc du fichier en utilisant l'index correspondant dans la table.
- Permet une augmentation dynamique de la taille du fichier, car de nouveaux blocs peuvent être ajoutés à la table d'index au fur et à mesure de l'extension du fichier.

❑ **Les inconvénients**

- Peut entraîner de la fragmentation interne, car des espaces vides peuvent se former dans les blocs de la table d'index lors de l'ajout ou de la suppression de blocs.

Les répertoires

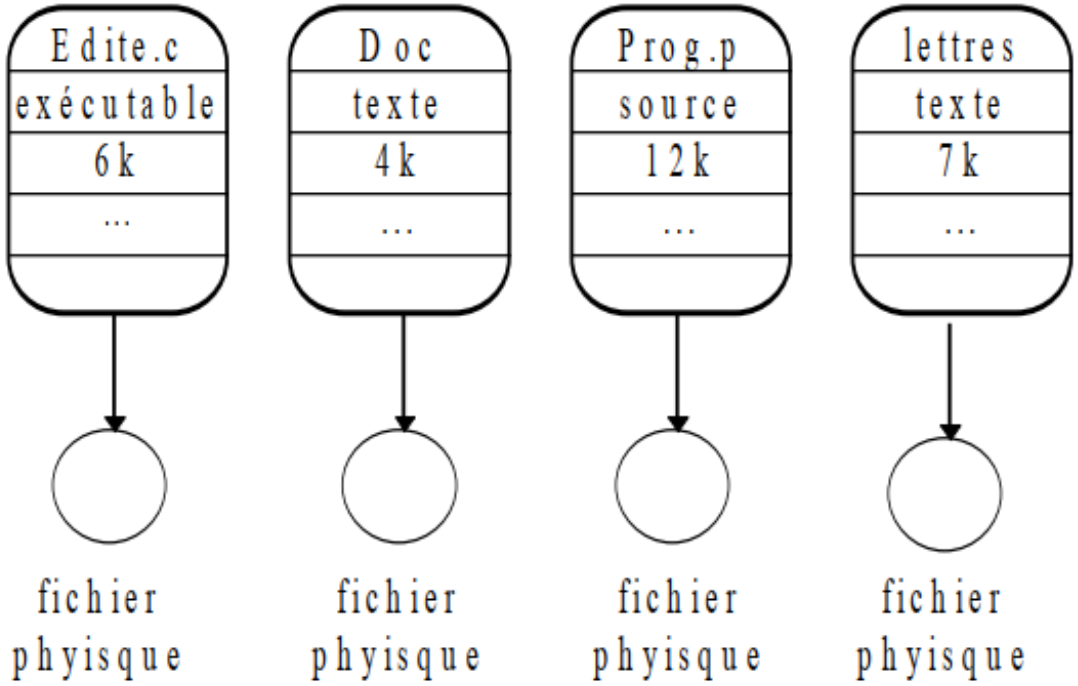
→ Définition

Le système d'exploitation est responsable d'établir la correspondance entre le nom logique d'un fichier et son emplacement physique sur le disque. À cette fin, il utilise une table appelée **répertoire**. Cette table contient plusieurs informations associées à chaque fichier, telles que :

- Nom logique du fichier
- Type de fichier
- Adresse physique du fichier
- Taille du fichier
- Caractère temporaire ou permanent du fichier
- Compteurs d'utilisation (nombre de lectures, d'écritures, de processus ayant ouvert le fichier, etc.)

Les répertoires

Exemple



Le répertoire permet ainsi au système d'exploitation de localiser et de gérer les fichiers de manière efficace en conservant les informations essentielles sur chaque fichier. Il constitue une composante clé du système de gestion des fichiers.

Les répertoires

→ Les types de répertoires

Il existe plusieurs types de répertoires qui permettent d'organiser et de gérer les fichiers de manière efficace.

a. **Le répertoire à un niveau**: est une table simple qui établit une correspondance directe entre le nom du fichier et ses attributs. Cependant, cette méthode est limitée en termes de capacité et de gestion des utilisateurs.

b. **Le répertoire à deux niveaux**: consiste à attribuer à chaque utilisateur son propre répertoire, avec un répertoire de niveau supérieur (répertoire principal) qui associe les noms des utilisateurs à leurs répertoires individuels.

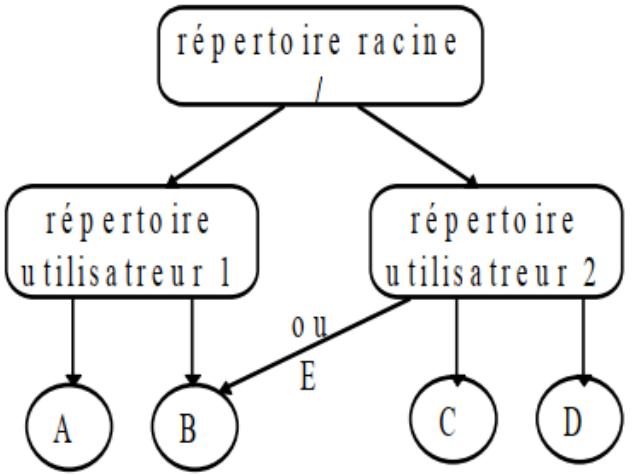
   - Cela permet de mieux organiser les fichiers des utilisateurs, mais peut poser des problèmes de coopération et de partage de fichiers entre utilisateurs.
   - Pour résoudre ces problèmes, les chemins (***path***) peuvent être utilisés pour retrouver n'importe quel fichier dans l'arborescence des répertoires.

Les répertoires

→ Les types de répertoires

c. **Le répertoire à structure d'arbre** : étend le concept du répertoire à deux niveaux en permettant une organisation hiérarchique des répertoires. Cela facilite la gestion des fichiers dans une structure d'arborescence plus complexe.

d. **Les répertoires à structure de graphe sans cycle** permettent d'avoir le même fichier dans plusieurs répertoires sans avoir à le dupliquer physiquement. Un lien (***link***) est établi entre le fichier original et les répertoires dans lesquels il doit apparaître. Cela permet d'économiser de l'espace de stockage, mais peut poser des problèmes lors de la suppression du fichier et peut entraîner la création de cycles dans la structure des répertoires. Pour l'implanter, on autorise qu'un répertoire puisse contenir comme entrée un pointeur vers un fichier (ou répertoire) qui n'est pas dans le sous-arbre dont il est la racine.



Opérations sur les fichiers

→ Création

- Vérifier si le nom du fichier n'existe pas déjà en explorant le répertoire.
- Ajouter une entrée si aucune n'est libre.
- Réserver suffisamment d'espace en mémoire secondaire.
- Enregistrer les informations correspondantes au fichier dans les champs appropriés du répertoire.

→ Destruction

- Rechercher si le nom du fichier existe dans le répertoire, sinon générer une erreur.
- Supprimer l'espace physique alloué au fichier.
- Le retirer du répertoire.

Opérations sur les fichiers

→ Ouverture

- Rechercher l'identificateur du fichier dans le répertoire.
- Vérifier les autorisations d'accès en lecture/écriture.
- Si le fichier peut être ouvert, le système d'exploitation doit :
  - Déterminer l'adresse physique en mémoire secondaire où le fichier est stocké.
  - Créer un bloc de contrôle de fichier (**BCF**) qui sera conservé en mémoire centrale et qui contient :
    - L'identificateur du fichier.
    - L'adresse physique du descripteur du périphérique d'E/S responsable des transferts entre la mémoire centrale et la mémoire secondaire.
    - L'adresse du bloc du fichier.
    - Dans le cas d'un accès séquentiel, l'adresse du bloc en cours.
- Lorsque plusieurs utilisateurs ouvrent simultanément un fichier, le système d'exploitation attribue à chacun un BCF et fournit à chacun un pointeur de connexion. Cela permet des accès indépendants et accélère les opérations de manipulation du fichier.

→ Fermeture

- Le contrôle de fichier (**BCF**) est simplement détruit.
- Accès aux articles :
  - Les opérations de lecture/écriture/mise à jour consistent à rechercher le bloc de la mémoire secondaire qui contient l'article, puis à effectuer l'opération qui entraîne le transfert entre la mémoire secondaire et la mémoire centrale.
  - L'article est identifié par son numéro d'ordre pour l'accès direct (adresse relative par rapport au début du fichier), ou il est le suivant de l'article courant pour l'accès séquentiel.

Protection des fichiers

→ Protection contre les accidents

- Sauvegarde périodique.
- Sauvegarde incrémentale.

→ Protection contre les attaques

- **Ver d'Internet**: est un type de programme informatique malveillant qui se propage rapidement à travers les réseaux informatiques Ils peuvent causer des dommages importants en détruisant des fichiers, en volant des informations sensibles …
- **Les virus**: Un virus est un morceau de programme attaché à un programme normal qui tente de s'introduire dans d'autres programmes.

→ Protection des accès

- Mots de passe :
  - Mot de passe simple.
  - Mot de passe généré en combinant un nombre aléatoire et crypté.
  - Mot de passe régulièrement changé par le S.E
  - Choix d'un algorithme de chiffrement par l'utilisateur.
  - Identification physique :
    - Carte magnétique avec mot de passe.
    - Empreintes digitales.
    - Intonations vocales.
    - Signature.
  - Identificateur de l'utilisateur.

Chapitre V: Gestion des entrées-sorties

Pr. Siham BAKKOURI Email: s.bakkouri@usms.ma

Plan

- INTRODUCTION
- LES PÉRIPHÉRIQUES D'E/S
- LES CONTRÔLEURS DE PÉRIPHÉRIQUES
- LES PILOTES DE PÉRIPHÉRIQUES
- LES PILOTES DE PÉRIPHÉRIQUES SOUS LINUX

Introduction

L'ordinateur est capable d'effectuer des **entrées-sorties**, c'est-à-dire d'échanger des données avec des périphériques externes tels que:

- **Des dispositifs électroniques** tels que les mémoires,
- **Des dispositifs magnétiques** comme les disques ou les disquettes
- **Des dispositifs mécaniques** comme le clavier et les imprimantes.

→ Les **périphériques** externes utilisés sont de natures très différentes, et l'un des objectifs du système de **gestion des entrées-sorties** est de maintenir une certaine homogénéité dans leur mode d'accès.

→ Les entrées-sorties représentent une part très importante, en termes de volume de code, des systèmes d'exploitation. Leur étude, qui présente plus d'aspects techniques que théoriques, est parfois négligée.

Les entrées sorties

→ Les périphériques d'E/S

On classe généralement les périphériques d'entrées-sorties en **deux catégories** principales : les ***périphériques blocs*** et les ***périphériques caractères***.

- Les données **des périphériques blocs** sont gérées à travers des blocs de taille fixe, souvent de 512 ou 1024 octets, parfois plus. L'accès à chacun de ces blocs est aléatoire.

Exemples: Les disques..

- Les **périphériques caractères** fournissent ou acceptent une séquence de caractères sans réelle structure.

Exemples: Les terminaux, les écrans, les claviers.…

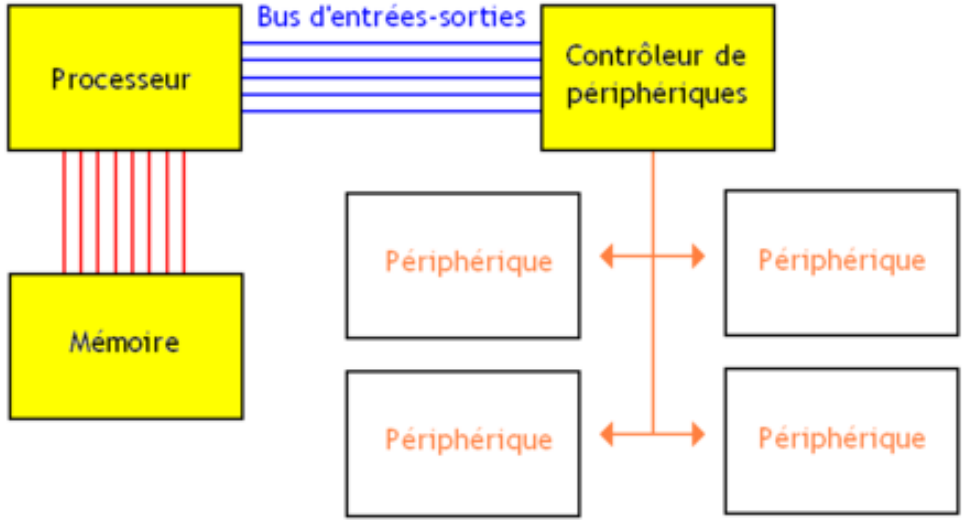
Les entrées sorties

→ Les contrôleurs

- La plupart du temps, le processeur ne commande pas directement les périphériques. Il utilise plutôt un circuit spécialisé appelé "**contrôleur**", qui possède des caractéristiques propres à chaque périphérique.
- Les commandes du processeur vers le périphérique seront transmises via ce contrôleur.
- Les contrôleurs sont connectés au bus de l'ordinateur et se voient attribuer un certain nombre d'adresses.
- À ces adresses différentes, des commandes de contrôle sont émises ou des entrées-sorties de données sont effectuées.
- Dans certains ordinateurs, les contrôleurs ne sont pas reliés au bus principal, mais à des voies d'entrées-sorties spécifiques.

→ Les contrôleurs

Les contrôleurs et le processeur fonctionnent en parallèle. Le contrôleur est équipé d'une mémoire tampon qui lui permet, par exemple, de lire des données provenant du périphérique d'entrée pendant que le processeur traite une autre tâche.



- Une fois son travail terminé, le contrôleur émet une interruption qui lui est spécifique. Lorsque le processeur reçoit cette interruption, il se dirige vers une adresse fixe.
- La valeur de cette adresse est contenue dans un vecteur spécifique à chaque interruption. À cette adresse d'interruption, une routine est généralement exécutée, réalisant ainsi le transfert des données vers la mémoire principale, puis rendant le contrôle au processus précédent.

Les entrées sorties

→ Les pilotes de périphériques

- Un pilote de périphérique est responsable du traitement d'un type spécifique de périphérique. Il utilise un ou plusieurs registres de commande du contrôleur pour envoyer des commandes et vérifier leur bon acheminement.
- Un pilote de disque doit effectuer les tâches suivantes :
  - Déterminer l'emplacement du bloc à rechercher.
  - Vérifier si le moteur du disque est en rotation.
  - Vérifier si le bras de lecture/écriture est positionné ; si ce n'est pas le cas, le déplacer.
  - Remplir les registres du contrôleur pour exécuter les requêtes.
  - Se bloquer en attendant la fin de l'opération.
  - Continuer après la réception d'une interruption signalant la fin de l'opération.
  - Vérifier s'il y a des erreurs.
  - En cas d'absence d'erreurs, transmettre le bloc lu à l'appareil demandeur.
  - Renvoyer un code d'erreur.
  - Exécuter la prochaine requête ou se bloquer en attente.

Les entrées sorties

→ Les pilotes de périphériques sous Linux

- L'écriture d'un pilote d'entrées-sorties est une tâche assez complexe. Nous n'examinerons que les grandes lignes de la structure des entrées-sorties sous Unix/Linux.

- Pour plus de détails, le lecteur pourra considérer les exemples se trouvant dans les répertoires : ***/usr/lib/drivers*** ou ***/systems/DRIVERS***.

- Le système Unix désigne chaque **périphérique** par un **fichier**. De manière courante, on rassemble les fichiers de tous les périphériques dans le répertoire : **/dev** *(device).* Ces fichiers sont appelés « ***spéciaux*** ».

Les entrées sorties

Exemple: On obtient les caractéristiques des fichiers de tous les périphériques avec la commande : **$ ls –l /dev**

```
total 11
crw--w----  1 pierre     0,   0 Mar 25 21:49 console
crw-rw-rw-  1 root      11,   0 Mar 11 16:13 des
crw-r-----  1 root       7,   0 Mar 11 16:11 drum
crw-rw----  1 root      41,   0 Mar 11 16:11 dump
crw-r-----  1 root       3,  11 Mar 11 16:11 eeprom
crw-rw-rw-  1 root      32,   3 Mar 11 16:18 gpone0d
crw-rw-rw-  1 root      29,   0 Mar 11 16:11 kbd
crw-------  1 root      37,  40 Mar 11 16:11 nit
crw-rw-rw-  3 root      18,   4 Mar 11 16:12 nrmt0
crw-rw-rw-  1 root      21,   2 Mar 25 14:13 ptyp2
crw-rw-rw-  1 root      21,  10 Mar 11 16:13 ptypa
crw-rw-rw-  1 root      21,  11 Mar 11 16:13 ptypb
```

La première lettre indique que c'est un fichier spécial de périphérique.

- « c » : un périphérique caractère
- « b » : un périphérique bloc