



Systèmes
d'Exploitation

Didier Verna
EPITA

Architecture

Von Neumann

Interruptions

Entrées /
Sorties

Protection

Systèmes d'Exploitation

Architecture pour les systèmes

Didier Verna

didier@lrde.epita.fr
<http://www.lrde.epita.fr/~didier>



Table des matières

Systèmes
d'Exploitation

Didier Verna
EPITA

Architecture

Von Neumann

Interruptions

Entrées /
Sorties

Protection

- 1 Architecture Générale
- 2 Cycle de Von Neumann
- 3 Interruptions
- 4 Entrées / Sorties
- 5 Protection matérielle



Architecture

Composition d'un système informatique

Systèmes
d'Exploitation

Didier Verna
EPITA

Architecture

Von Neumann

Interruptions

Entrées /
Sorties

Protection

- Un (ou plusieurs) processeur(s)
- Mémoire
- Contrôleurs de périphériques
- Périphériques associés
- Bus de liaison





Bus

Systèmes
d'Exploitation

Didier Verna
EPITA

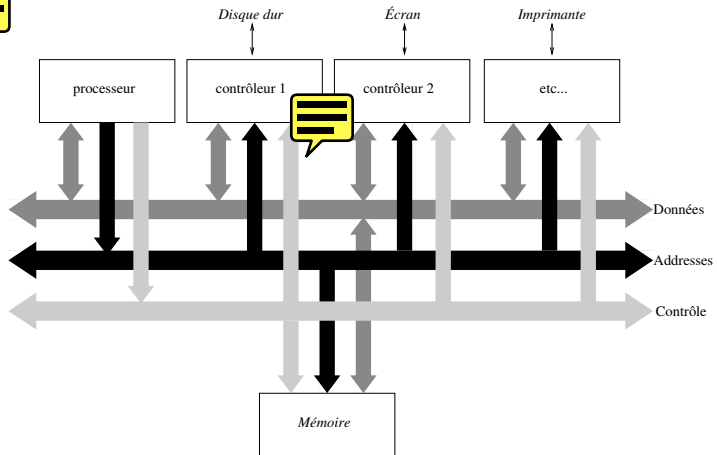
Architecture

Von Neumann

Interruptions

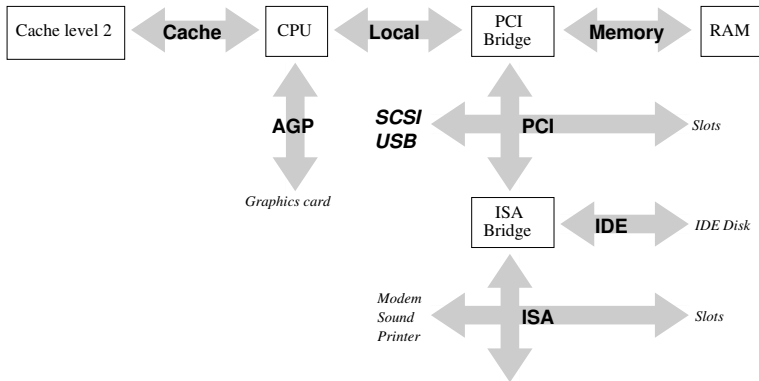
Entrées /
Sorties

Protection





Exemple : architecture Pentium



Et aussi IEEE 1394 (FireWire) etc.




Cycle de Von Neumann



■ Scenario

- ▶ Extraction d'une instruction
- ▶ Stockage dans le registre d'instruction
- ▶ Décodage
- ▶ Extraction de données éventuelles (opérandes)
- ▶ Exécution

■ Implémentation

- ▶ Jeu d'instructions spécifique à chaque CPU
- ▶ Utilisation de registres CPU. Registres spéciaux : PC (Program Counter), SP (Stack Pointer), PSW (Program Status Word)
- ▶ Architectures modernes : pipelines, CPU super-scalaires, RISC *etc.* 



Systèmes
d'Exploitation

Didier Verna
EPITA

Architecture

Von Neumann

Interruptions

Entrées /
Sorties

Protection



Initialisation

Systèmes
d'Exploitation

Didier Verna
EPITA

Architecture

Von Neumann

Interruptions

Entrées /
Sorties

Protection

■ BIOS (Basic Input Output System)

- ▶ initialise le matériel (registres processeur, mémoire *etc.*)
- ▶ scanne les bus (ISA et PCI d'abord) pour trouver un périphérique amorçable (bootable)
Premier secteur \implies partition active \implies deuxième
« boot loader » ou système
- ▶ charge le système d'exploitation en mémoire

■ Système d'Exploitation

- ▶ lance le premier processus (« init »)
- ▶ attend un événement

■ Les événements sont produits par des « interruptions »



Interruptions

Seul moyen d'interrompre le traitement en cours d'exécution

Systèmes
d'Exploitation

Didier Verna
EPITA

Architecture

Von Neumann

Interruptions

Entrées /
Sorties

Protection

■ Deux types d'interruption

▶ Interruptions matérielles

déclenchées par le processeur, les périphériques *etc.*
(ex.. fin d'une opération d'entrée / sortie)

▶ Interruptions logicielles

déclenchées par des « appels système »
(ex.. accès mémoire)



Vocabulaire : « Exception » = interruption causée par une erreur (division par 0, accès mémoire illégal *etc.*)

■ Remarque : Conflit d'IRQ, périphériques legacy *etc.*

⇒ « Plug and Play » (Plug and Pray).



Traitement des interruptions

Systèmes
d'Exploitation

Didier Verna
EPITA

Architecture

Von Neumann

Interruptions

Entrées /
Sorties

Protection

■ Scenario

- ▶ L'arrivée d'une interruption suspend le traitement en cours, exécute la routine associée puis reprend le traitement précédent l'interruption.

■ Implémentation

- ▶ Chaque interruption est associée à une routine fixe de traitement.
- ▶ **Vecteur d'interruptions** : tableau situé en mémoire basse et contenant les adresses de début de chaque routine.



L'adresse de l'instruction interrompue est stockée dans la pile du système. Les routines de traitement des interruptions doivent elle-même sauvegarder l'état du processeur si besoin est.



Interruption des interruptions

Systèmes
d'Exploitation

Didier Verna
EPITA

Architecture

Von Neumann

Interruptions

Entrées /
Sorties

Protection



Masquage simple

Aucune interruption ne peut intervenir pendant le traitement d'une interruption : leur traitement est différé jusqu'à la fin du traitement de l'interruption en cours.



Masquage sélectif

Chaque interruption est associée à un niveau de priorité. Une interruption peut interrompre une autre interruption de priorité inférieure. Les interruptions de même niveau sont gérées par masquage simple.



■ Contrôleur de périphériques

- ▶ Gérer un ou plusieurs périphériques de même type (IDE, SCSI *etc.*)
- ▶ Harmoniser leur comportement (vitesse d'accès, structure *etc.*)
- ▶ Assurer le transfert de données entre ces périphériques et la mémoire locale du contrôleur

■ Contrôle des périphériques par pilotes (drivers)



Fonctionnement des pilotes

Systèmes
d'Exploitation

Didier Verna
EPITA

Architecture

Von Neumann

Interruptions

Entrées /
Sorties

Protection

■ Scenario

- ▶ Requête d'E/S par positionnement des registres du contrôleur
- ▶ Signalement de la fin d'une E/S par une interruption



E/S configurées en mémoire

- ▶ Adresses mémoire correspondant aux registres des contrôleurs
- ▶ Pas besoin d'instructions d'E/S spécifiques
- ▶ Évite le contact direct entre le matériel et les programmes utilisateurs

■ Remarques :

- ▶ Pratique pour les périphériques à temps de réponse rapide (adaptateurs vidéos) ou utilisés fréquemment (modem)
- ▶ Il existe également des périphériques permettant l'acquisition de données avant que celles-ci soient requises.



Techniques d'E/S

Processus \Rightarrow Appel système \Rightarrow Pilote \Rightarrow Contrôleur

Systèmes
d'Exploitation

Didier Verna
EPITA

Architecture

Von Neumann

Interruptions

Entrées /
Sorties

Protection



E/S synchrones

Le pilote attend la fin de la requête : scrutation (sondage) du périphérique (polling), attente active. Une seule requête d'E/S à la fois.



E/S asynchrones

Demande d'interruption pour signaler la fin de la requête, retour immédiat du pilote. Possibilité d'E/S simultanées. Nécessité de maintenir une « table d'état des périphériques ».



Direct Memory Access (DMA)

Systèmes
d'Exploitation

Didier Verna
EPITA

Architecture

Von Neumann

Interruptions

Entrées /
Sorties

Protection

- **Problème** : les périphériques rapides (disques, réseau *etc.*) sont susceptibles de générer énormément d'interruptions (à une fréquence proche du temps de traitement des interruptions).
- **Solution** : transférer des blocs entiers de données entre le contrôleur et la RAM, sans intervention du processeur. \implies Une interruption par bloc et non plus par mot.



Mode double

Partage des ressources \implies problèmes de cohérence et de sécurité

Systèmes
d'Exploitation

Didier Verna
EPITA

Architecture

Von Neumann

Interruptions

Entrées /
Sorties

Protection

■ Mode utilisateur

Mode d'exécution des processus clients : le système commute en mode utilisateur dès qu'il donne la main à un processus.

■ Mode superviseur

Mode d'exécution du système d'exploitation : le matériel commute en mode superviseur dès qu'une interruption est reçue.

■ Instruction privilégiée

Instruction potentiellement nuisible ne pouvant s'effectuer qu'en mode superviseur. Au minimum : toutes les instructions d'E/S.



Protection de la mémoire

Systèmes
d'Exploitation

Didier Verna
EPITA

Architecture

Von Neumann

Interruptions

Entrées /
Sorties

Protection

■ Problèmes

- ▶ Protéger le système et les utilisateurs entre eux
- ▶ Empêcher la modification du vecteur d'interruptions et des routines de traitement de ces interruptions

■ Solution la plus simple

- ▶ À chaque processus, on associe un registre *base* et un registre *limite* définissant ses capacités d'accès à la mémoire.
- ▶ Des instructions privilégiées permettent au système d'exploitation de charger ces registres.
- ▶ Le matériel effectue le contrôle d'accès.



Protection du CPU

Systèmes
d'Exploitation

Didier Verna
EPITA

Architecture

Von Neumann

Interruptions

Entrées /
Sorties

Protection

■ But

- ▶ S'assurer que tout processus client finit par rendre la main au système

■ Solution : horloge



- ▶ **Temps fixe** : l'horloge génère une interruption à intervalle régulier.
- ▶ **Temps variable** : temps fixe + compteur. Le système d'exploitation fixe la valeur du compteur. Le compteur est décrémenté à temps fixe. Une interruption est générée quand le compteur atteint la valeur 0.