Cours 2 : Gestion des interruptions (IT)

Introduction

- Interruption : un signal généré par le matériel pour notifier un événement au système d'exploitation

⇒ **Générée** par le **matériel**

⇒ **Traitée** par le **système** d'exploitation en **mode système**

Classification

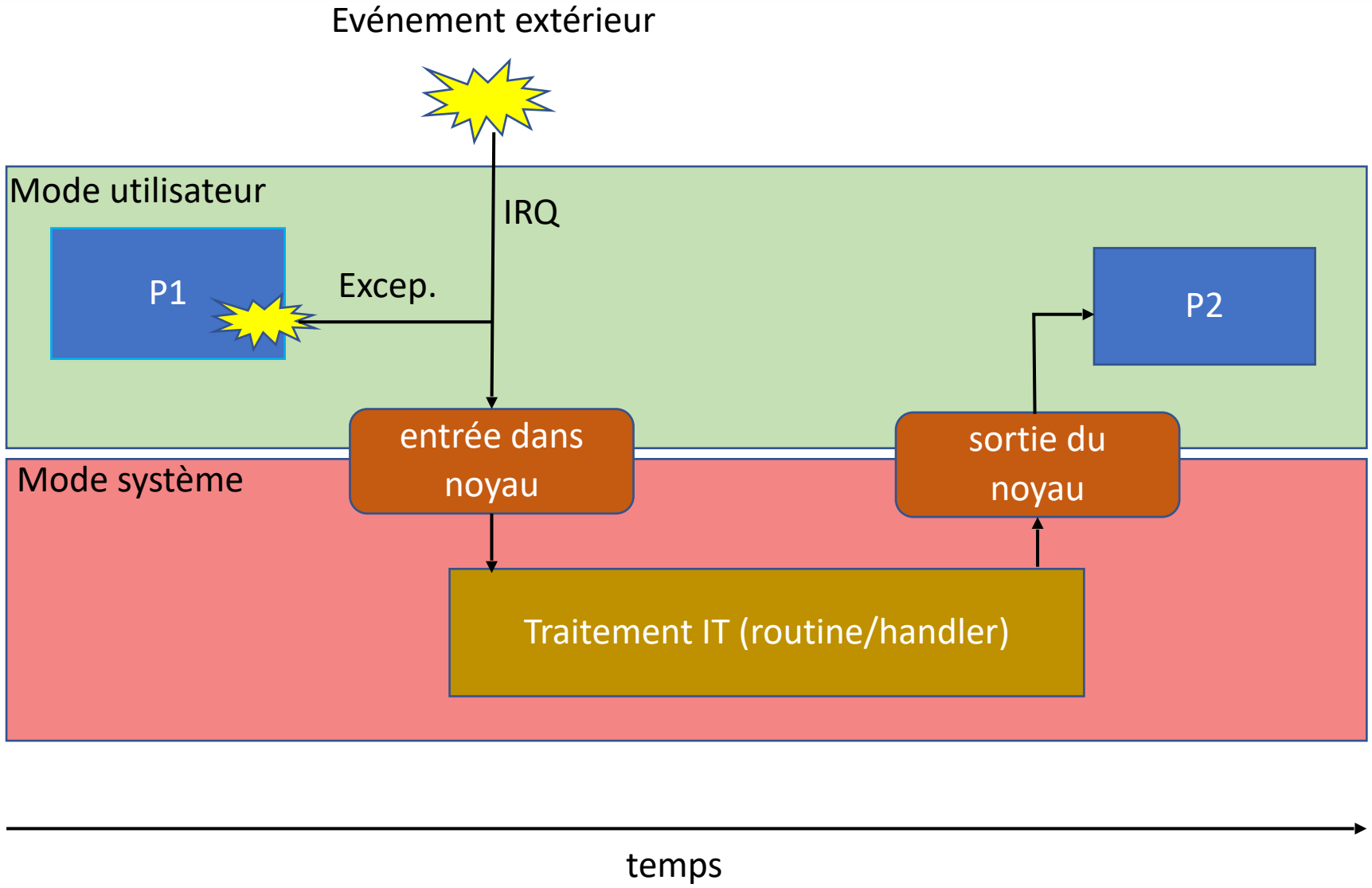
- Interruptions matérielles (Hardware Interrupt - IRQ)
  - Générées par les contrôleurs externes de façon **asynchrone** (indépendant de l'exécution en cours)
  - Exemples
    - Contrôleur disque (Fin de transfert),
    - clavier / souris
    - horloge
- Interruptions logicielles (exception)
  - Générer de façon **synchrone** par les **programmes** en cours d'exécution
  - Exemples :
    - erreurs : division par 0, erreur d'adresse, …
    - appel système (le processus déclenche une interruption)

Priorité des interruptions

- Déterminer un ordre de traitement en cas de concurrence (plusieurs interruptions déclenchées "simultanément")
- Priorités classiques :

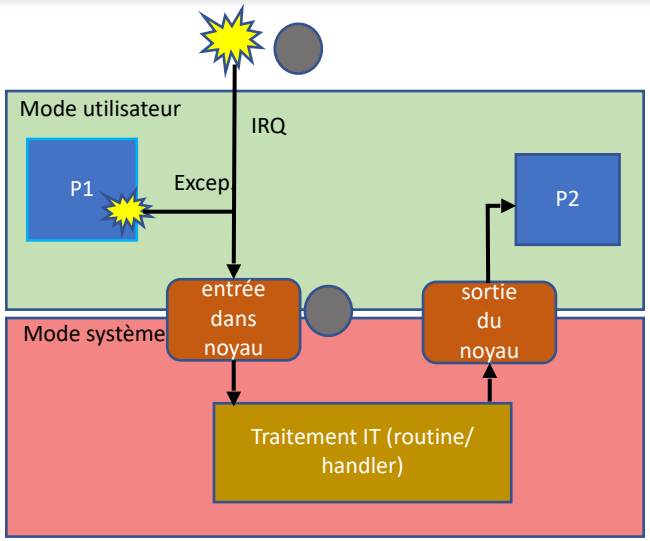
1. Horloge
2. Disque
3. Console (souris/clavier)
4. Autres périphérique
5. Appels système

Traitement des interruptions



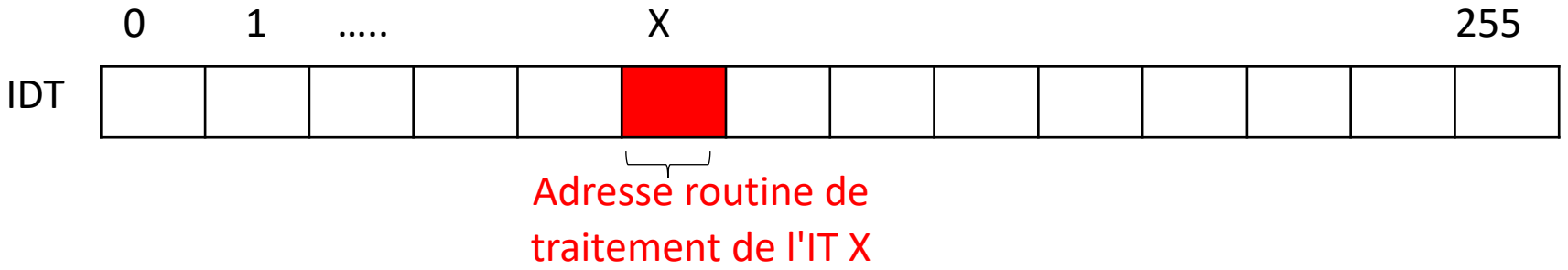
Traitement de l'interruption X

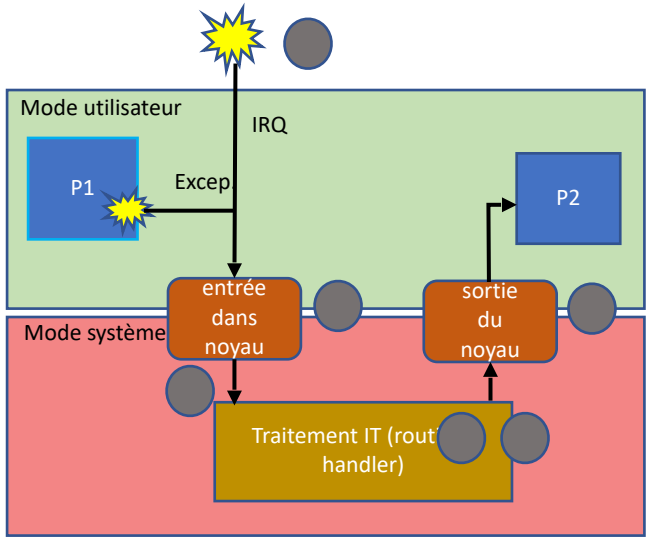
1. Déclenchement interruption X
2. Basculement en Mode Système
   - Mot d'état du processeur = Program Status Word (PSW)
   - PSW
     - Etat du processeur (actif/inactif) ;
     - Mode du processeur (utilisateur/système)
     - Masque d’interruption
     - PC/CO : program counter/compteur ordinal = la prochaine instruction à exécuter (RIP –Intel)
   - Sauvegarde du PSW courant / Changement du PSW
   - Sauvegarde des registres du processus courant



Traitement de l'interruption X

3. Exécution de la routine de traitement
   - 1 **vecteur d'interruption** avec 1 entrée par interruption (IRQ/exception)
   - Ex : Intel 255 entrées (IDT – Interrupt Descriptor Table)

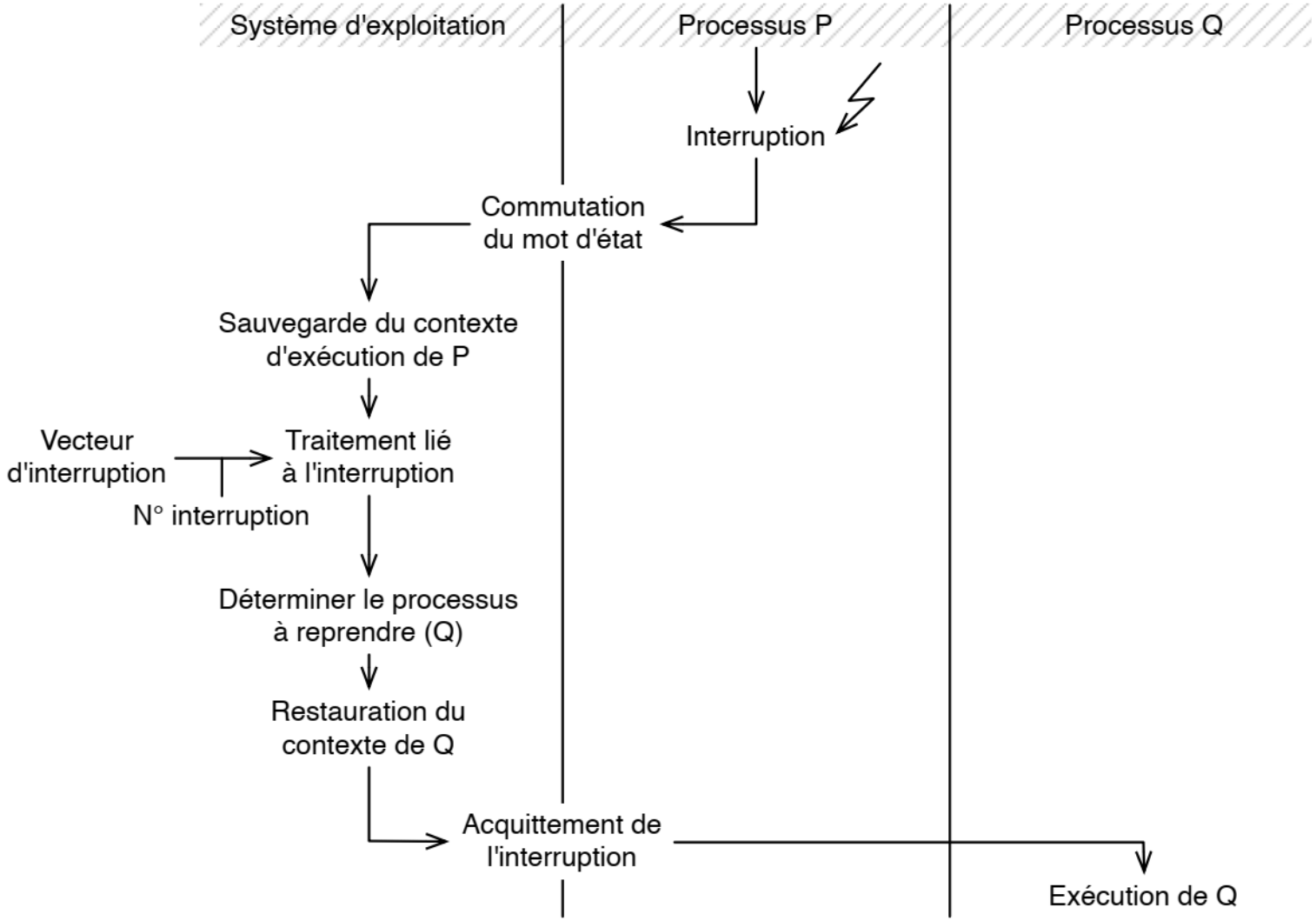




   - Saut pour exécuter la routine IDT[X]()
   - Au démarrage le système définit les entrées du vecteur d'interruption

4. Déterminer le processus à reprendre
5. Restaurer registres du processus
6. Retour d'interruption

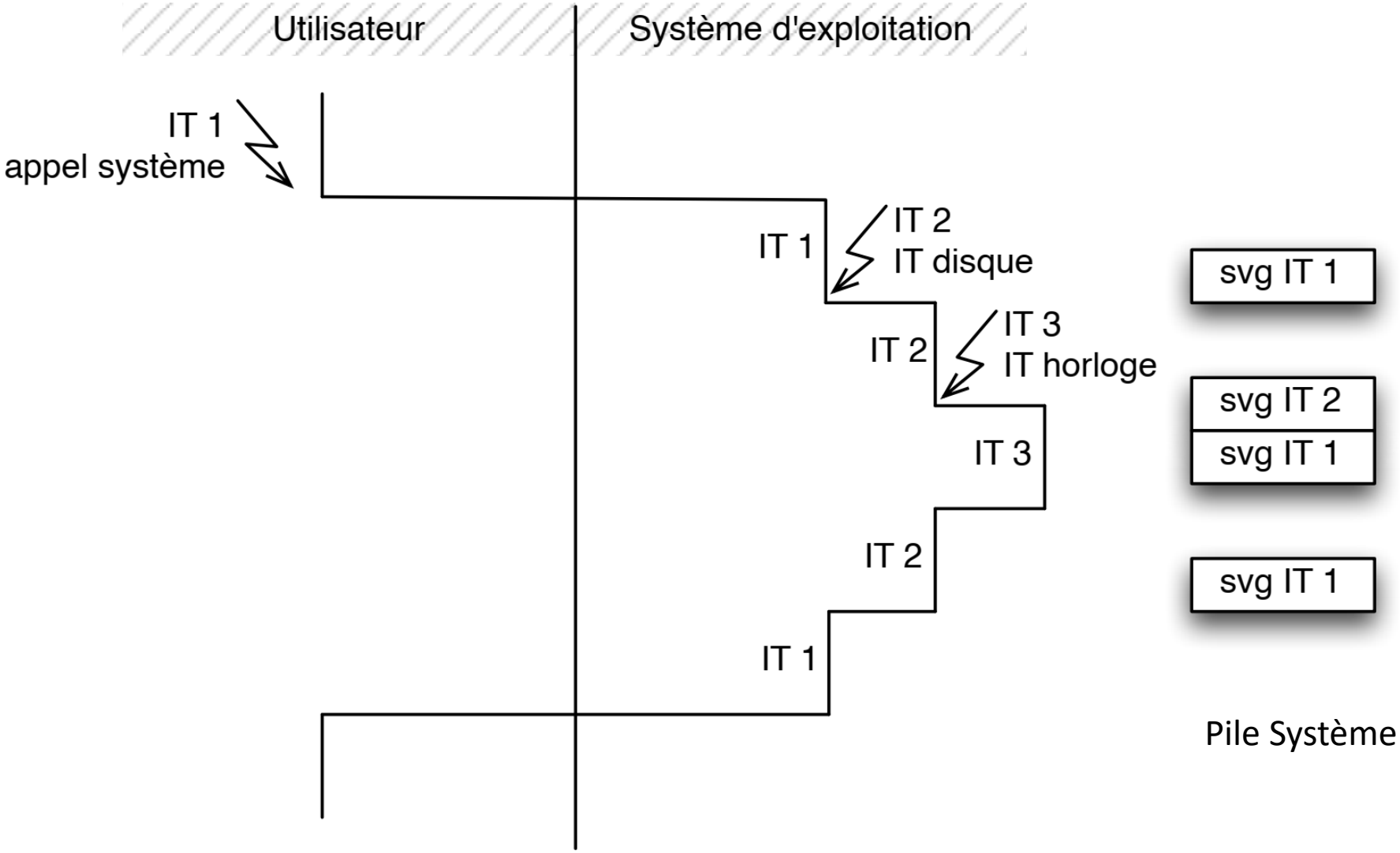
Traitement de l'interruption X



Pile d'interruption

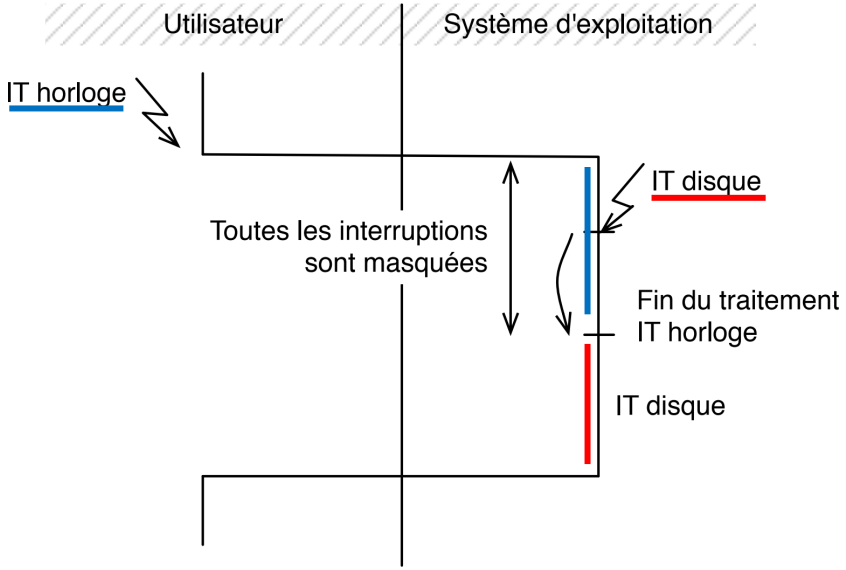
- Lorsqu'il y a plusieurs interruptions (IT), possibilité d'interrompre une routine en cours.
- Si priorité nouvelle IT > priorité IT en cours
  1. Sauvegarder le contexte IT en cours sur **la pile d'interruption**
  2. Traiter le nouvelle IT
  3. Restaurer le contexte de l'ancienne IT
- Le pile d'interruption (ou pile système) est utilisée uniquement en mode Système

Pile d'interruption



Masquage et désarmement

- Désarmement : **annulation** du traitement d'une interruption (eg. utile dans le cas d'un périphérique défaillant).
- Masquage : **retarder** le traitement d'une interruption.
- *Mask(X) :* masquer toutes les interruptions de priorité inférieure ou égale à X (Unix Mask() = spl())



Lors du traitement de l'IT X, les IT de priorité ≤ à X sont masquées

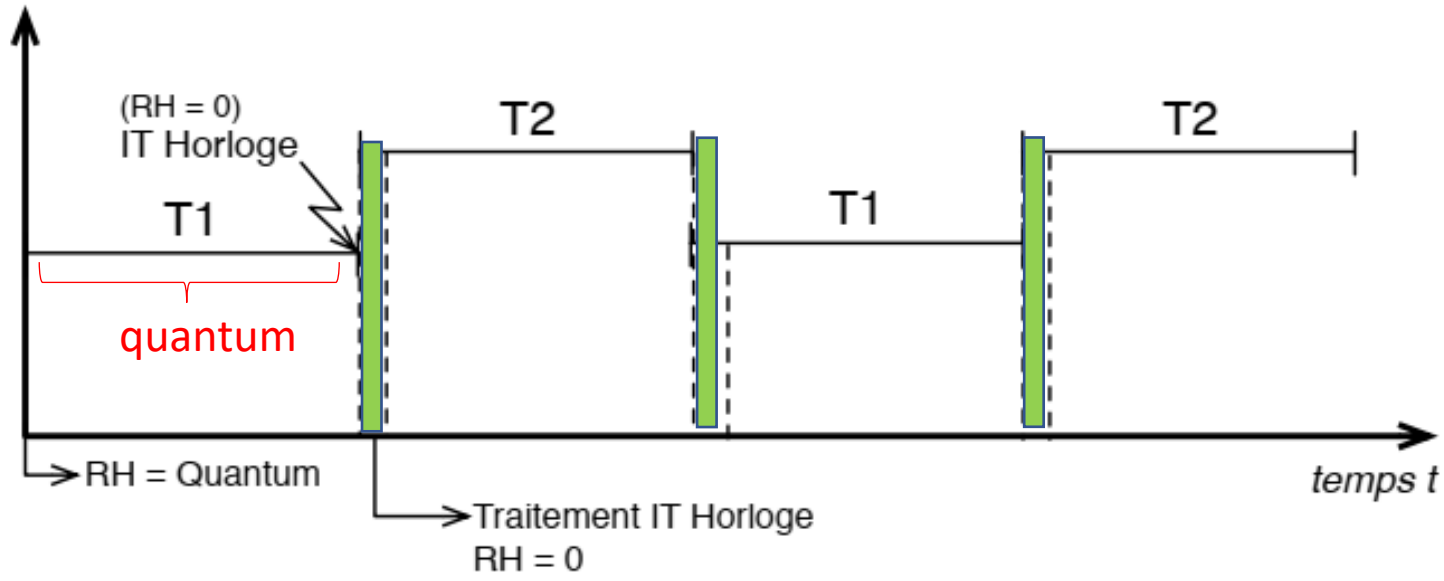
Temps partagé : Implémentation

- Implémentation du quantum grâce l'**interruption horloge** générée par le composant Horloge (*PIT : Programmable Interval Timer*)
- Horloge
  - 1 composant matériel externe au CPU (~contrôleur)
  - Fonctionne à une certaine fréquence (~1 MHz)
  - Gère le Registre Horloge (RH ou Rtempo) :

Toutes les μs (10^{-6}s)
    Décrémenter RH
    Si RH = 0
        Lever IT Horloge

Temps partagé : Implémentation

- Le RH est un registre privilégié : peut être réinitialisé uniquement en mode Système
- Le système met une valeur dans RH pour générer une interruption dans N µs.



Temps partagé : quantum et tick

- Dans la plupart des systèmes, quantum ≈100 ms (suffisant pour percevoir le pseudo parallélisme) $\Longrightarrow$ gestion des alarmes peu précise

$\Longrightarrow$ Utilisation de **Ticks =** la valeur (temps) entre deux interruptions horloge

Quantum = N x Tick