# Table des vecteurs d’interruption

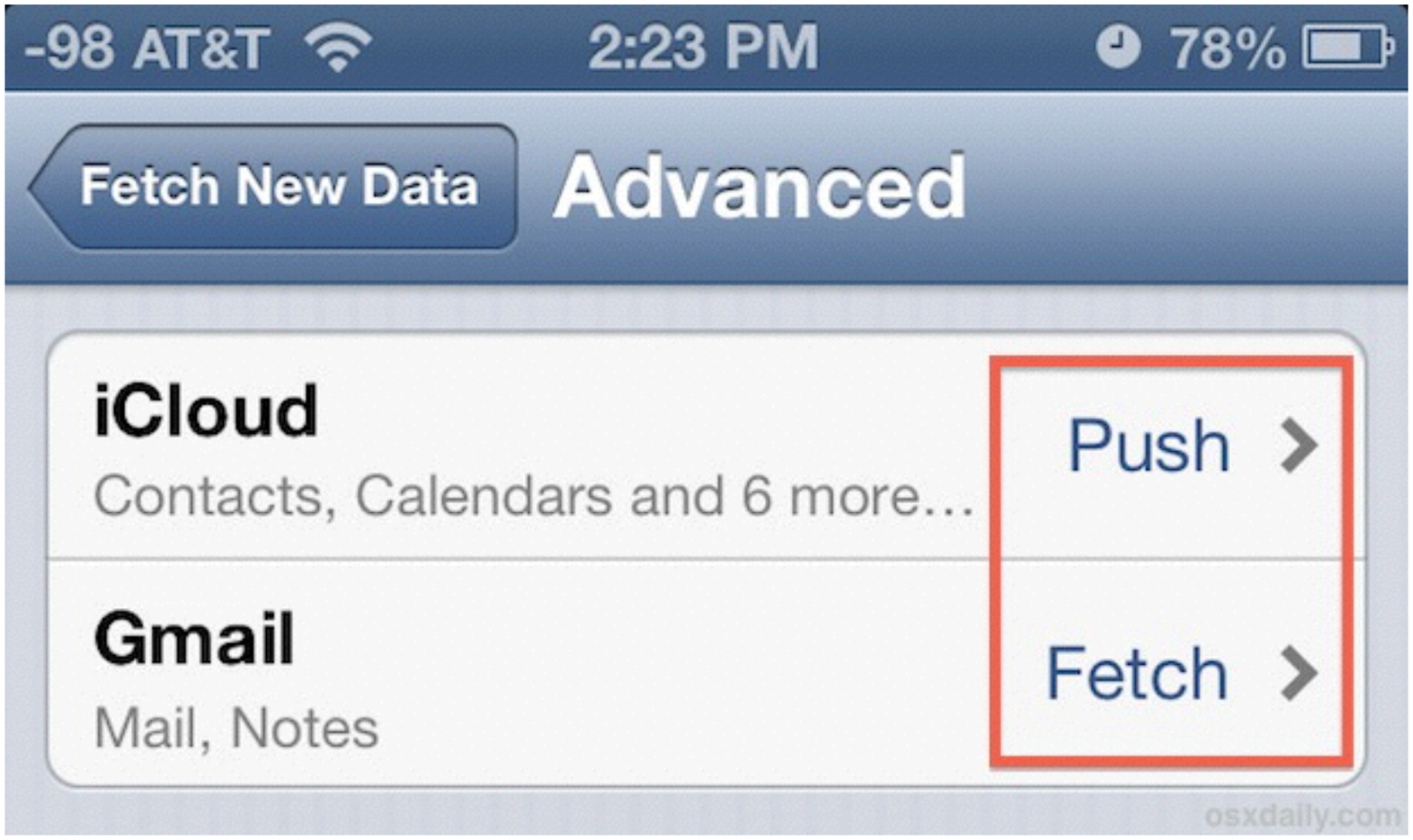

GIF-1001 Ordinateurs: Structure et Applications, Hiver 2017
Jean-François Lalonde

# Interruptions

- Pensons à un micro-processeur qui voudrait lire les valeurs écrites au clavier par un utilisateur.

- Comment faire?
  - Option #1: demander au clavier périodiquement s'il a reçu une nouvelle touche, sinon, attendre!
  - Option #2: c'est le clavier qui «dit» au micro-processeur qu'il a reçu une nouvelle touche!

# Interruptions



# Types d’interruptions

- **Système**: reset, faute matérielle générale, etc.
- **Exception**: le processeur peut générer des interruptions s’il n’est pas capable de lire ou d’exécuter une instruction (opcode invalide, division par 0, mémoire protégée, etc).
- **Matérielles**: générées par les périphériques
- **Logicielles**: le programmeur (nous!) peut générer une interruption sur demande

# Une interruption survient… que faire?

Source: soundbible.com


Source: soundbible.com


Source: soundbible.com

Source: soundbible.com

# Une interruption survient… que faire?

1. Terminer l’instruction en cours

2. Déterminer s’il faut traiter l’interruption.

3. Sauvegarder le contexte

4. Déterminer l’adresse de la routine de traitement de l’interruption

5. Exécuter cette routine

# Interruptions

- Une interruption interrompt l'exécution des instructions par le microprocesseur.

- Lors d'une interruption:

  1. l'exécution du programme principal est suspendue;
  2. une routine (fonction) traitant l'interruption est exécutée;
  3. puis le programme principal est continué.

- Quelle est la différence entre une interruption et un branchement?

  - les interruptions peuvent survenir n'importe quand pendant l'exécution.

# Une interruption survient… que faire?

1. Terminer l'instruction en cours

2. Déterminer s'il faut traiter l'interruption.

3. Sauvegarder le contexte

4. **Déterminer l'adresse** de la routine de traitement de l'interruption

5. Exécuter cette routine

# Routines de traitement d'interruptions

- Les routines de traitement d'interruptions sont des fonctions « spéciales » que l'on appelle que pour traiter les interruptions

- Où sont-elles situées?

  - en mémoire!

- Comment fait-on pour savoir:

  - quelle routine exécuter pour quelle interruption?

  - à quelle adresse est cette routine?

- Grâce à la table des vecteurs d'interruption, pardi!

# Table des vecteurs d'interruption

- Chaque entrée de la table « branche » vers la routine correspondante

| Adresse | Interruption | Signification |
|---|---|---|
| 0x00 | Reset | redémarrage |
| 0x04 | Instruction indéfinie | problème lors du décodage |
| 0x08 | Interruption logicielle | demandée par le programmeur: instruction `SVC` |
| 0x0C | « Prefetch abort » | « fetch » invalide |
| 0x10 | « Data abort » | accès mémoire invalide |
| 0x14 | Espace réservé | ne rien mettre ici |
| 0x18 | IRQ | « Interrupt ReQuest »: interruption matérielle générale |
| 0x1C | FIQ | « Fast Interrupt reQuest »: interruption matérielle rapide |

# Retour sur l'exemple de programme…

```
SECTION INTVEC

B main
```

**Table des vecteurs d'interruption**

Ici, nous ne supportons qu'une seule interruption, **laquelle?**

```
SECTION CODE

main

; Programme principal
LDR R0, b
LDR R1, c
ADD R1, R0, R1
LDR R3, =a
STR R1, [R3]

B main
```

Code principal

```
SECTION DATA

; Déclaration de la variable a
a DS32 1
```

Variables en mémoire
(non initialisées)

# Exemple de programme sur le simulateur

```
SECTION INTVEC

B main
B undefInterrupt ; Instruction indéfinie
B softInterrupt  ; Interruption logicielle

SECTION CODE

undefInterrupt
; routine de traitement de l'interruption
"instruction indéfinie"

softInterrupt
; routine de traitement de l'interruption
"interruption logicielle"

main
; routine de traitement de l'interruption
"reset", donc, notre code principal

SECTION DATA
```

Ici, nous supportons **3 interruptions**

Routine pour instruction indéfinie

Routine pour interruption logicielle

Code principal

Variables en mémoire

# Table des vecteurs d'interruption

- Contient une instruction (en ARM) qui branche vers la routine de traitement de l'interruption
- Commence à l'adresse 0x0 de la mémoire
  - peut être déplacée ailleurs
  - dans un système complet, la table est modifiée par le système d'exploitation.

# Démonstration

(Table des vecteurs d’interruption)

# Une interruption se termine… que faire?

1. Restaurer le contexte

2. Reprendre là ou le processeur était rendu

# Reprise du programme

- Dans plusieurs architectures, une instruction spéciale est utilisée pour indiquer la fin d'une interruption

- En ARM, c'est, comment dire, un peu bizarre…

  - il faut: utiliser une instruction avec **S** (change les drapeaux), qui stocke son résultat dans **PC**(!)

```
fiqInterrupt
  ; routine de traitement de l'interruption FIQ

  ...

  ; terminé!
  SUBS PC, LR, #4
```

# Résumé

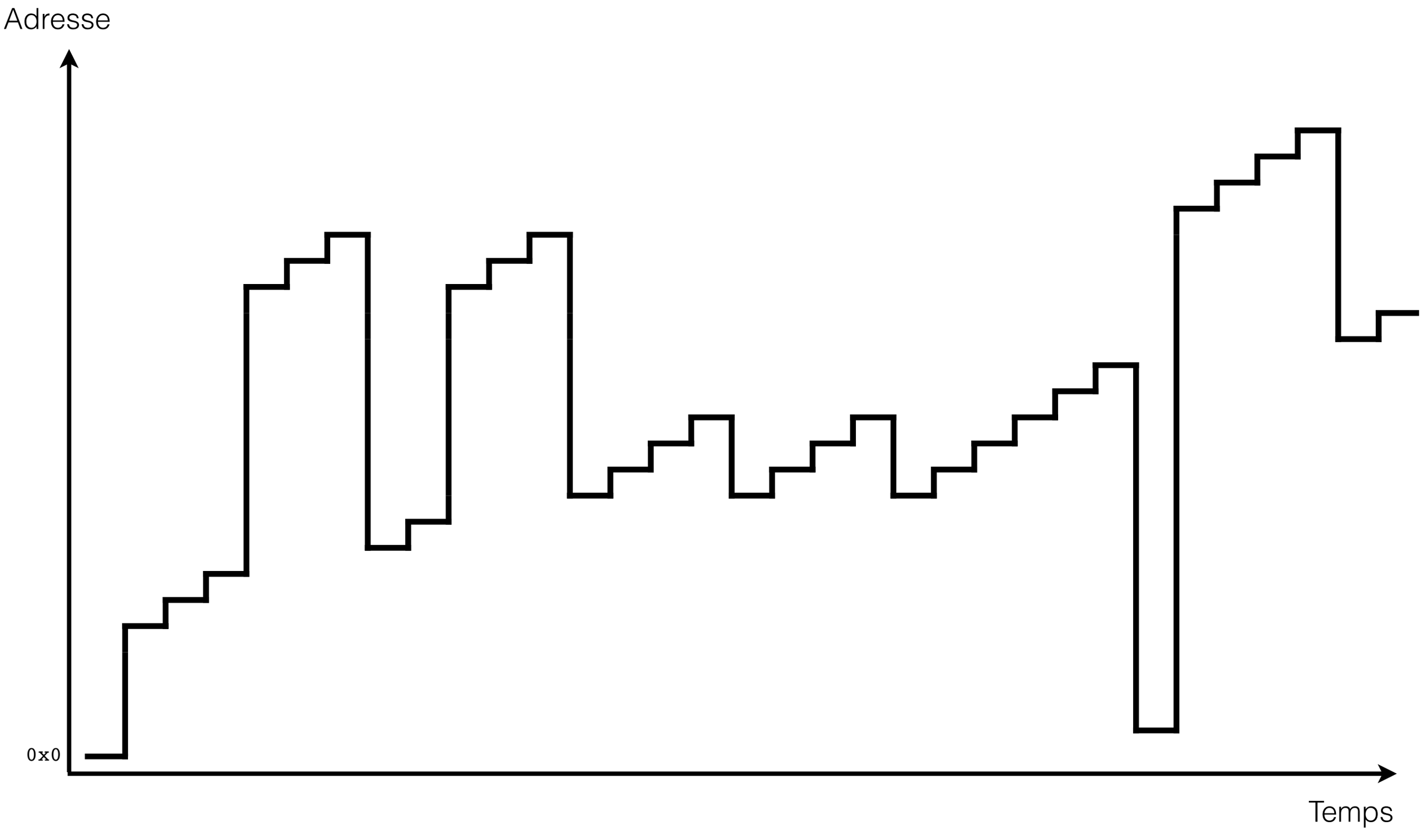
Interruption!

1. Terminer l'instruction en cours
2. Déterminer s'il faut traiter l'interruption.
3. Sauvegarder le contexte
4. Déterminer l'adresse de la routine de traitement de l'interruption
5. Exécuter cette routine

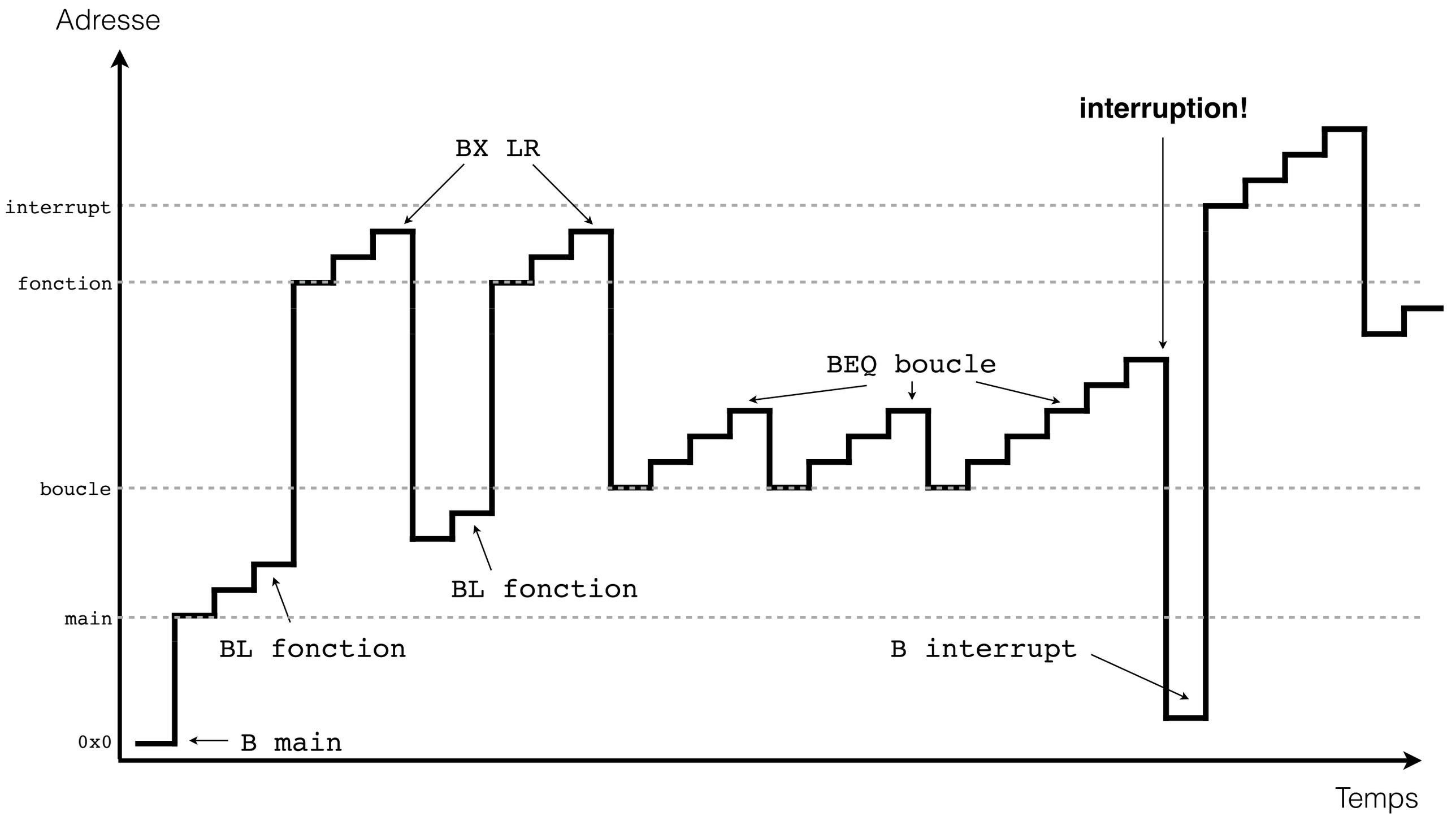
Traitement de l'interruption…

1. Restaurer le contexte
2. Reprendre là ou le processeur était rendu

# Évolution de PC…



# Évolution de PC…



# Évolution de PC…