Execute

ARO

5 - Execute & Memory Access

Definition

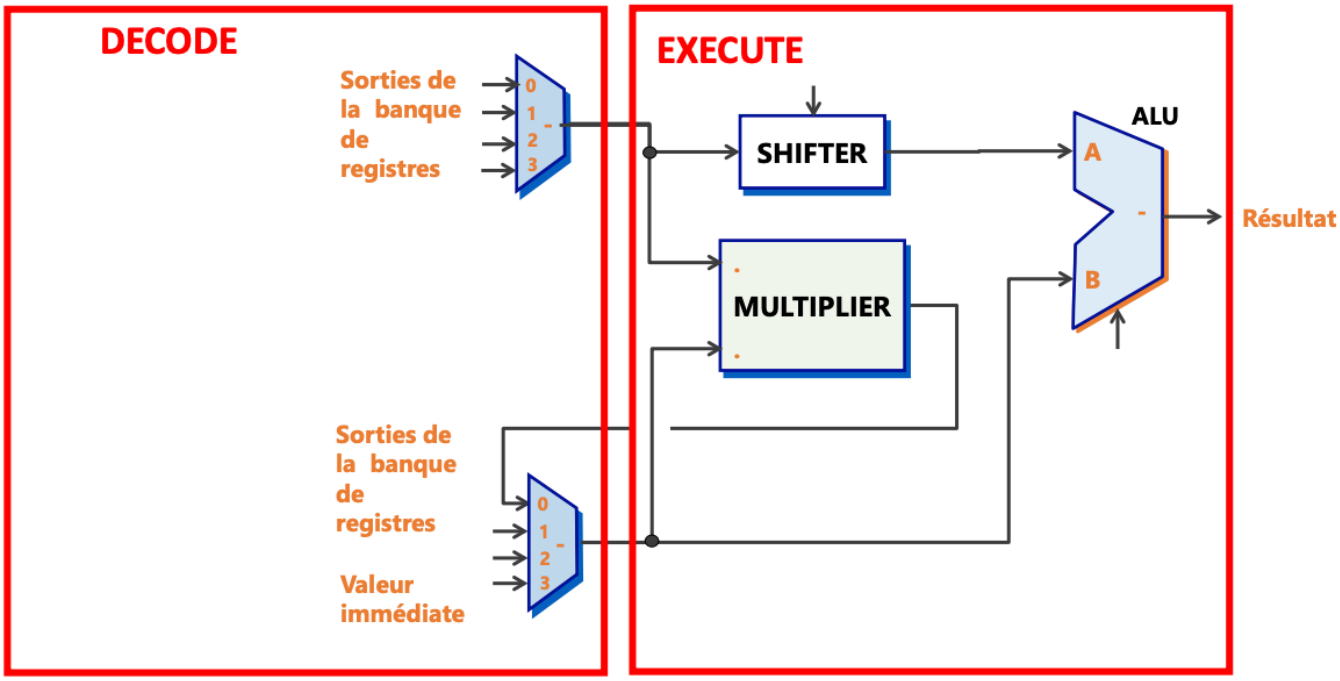
Le texte traite de l'étape d'exécution dans un processeur, où les instructions sont effectivement réalisées, incluant des opérations arithmétiques, logiques et de transfert de données. Il explique également les différents aspects de cette étape, tels que l'addition/soustraction avec gestion des dépassements, la multiplication avec précision des résultats selon qu'ils sont signés ou non, et les opérations de décalage (logical shift, arithmetic shift, rotation) implémentées efficacement à l'aide de circuits spécialisés.

Table des matières

1. Execute

L'opération d'exécution (execute) dans un processeur est l'étape où l'instruction décodée est effectivement réalisée. Cela implique d'effectuer les opérations arithmétiques, logiques ou de transfert de données spécifiées par l'instruction. Par exemple, si l'instruction est une addition, cette étape effectuera l'addition des valeurs des registres appropriés. De même, pour une instruction de branchement, cette étape modifiera le compteur de programme (PC) pour exécuter la prochaine instruction appropriée.

Le bloc EXECUTE contient le bloc **ALU**, le registre **CPSR**, le bloc **SHIFTER**. Les operations sont effectuées sur des entiers (signés/non-signés).



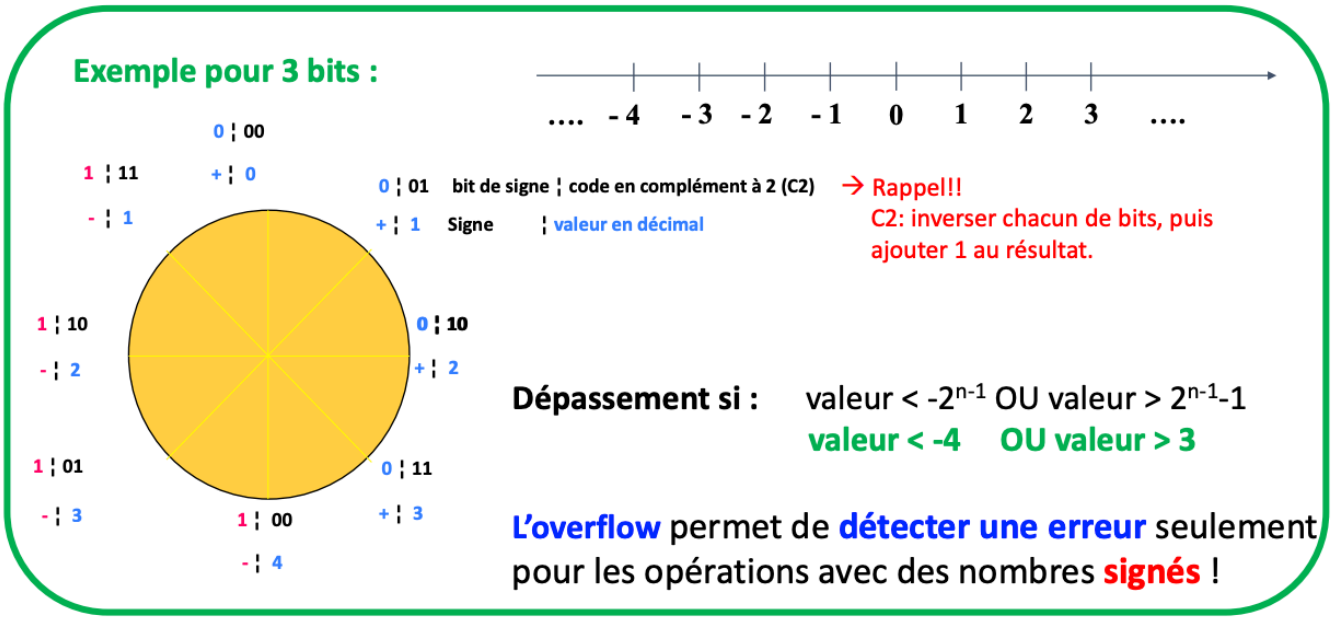
2. Addition/Soustraction

2.1. Non-signés

Pour n bits	**Addition**	**Soustraction**
Dépassement si le résultat est :	$> 2^n - 1$	< 0
Flag	Carry (report)	Borrow (emprunt)

- Génération d'un flag de **Carry** dans le registre d'état
  - Pour la soustraction, le Borrow est traduit par un flag de Carry **inversé** dans le registre d'état: **Carry** = **NOT(Borrow)**
- Le Carry permet de détecter une erreur pour les opérations avec des nombres non-signés!

2.2. Signés



2.3. Opérations et dépassement

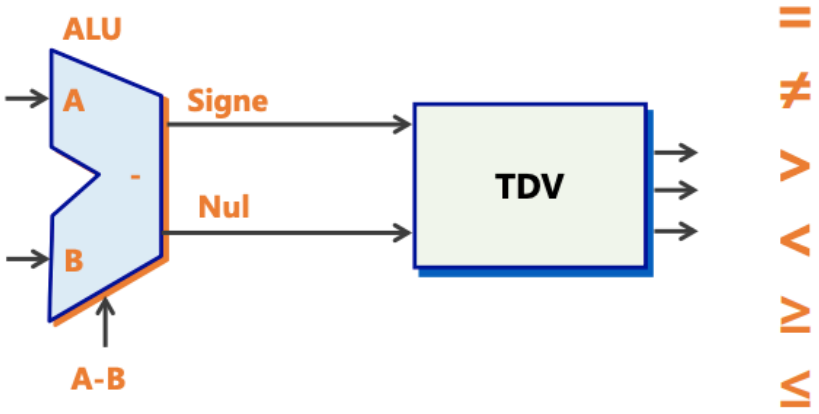
Détection des **cas de dépassement**:

- Représentation des nombres non signés:
  - Addition => dépassement indiqué par Carry « (C)
  - Soustraction => dépassement indiqué par **Borrow**
    - Borrow = Emprunt = not Carry
- Représentation des nombres "signés en C2":
  - Addition => dépassement indiqué par "Overflow « (V)
  - Soustraction => dépassement indiqué par "Overflow « (V)

Overflow = Cn xor Cn-1

2.4. Comparaison

- Comparaison de deux valeurs pour déterminer si elles sont égales, inférieures ou supérieures utilisent la soustraction.



3. Multiplication

Nombre de bits du résultat : Le nombre de bits du résultat d'une multiplication est la somme du nombre de bits des opérandes. Par exemple, si vous multipliez deux nombres de 8 bits, le résultat pourrait avoir jusqu'à 16 bits.

Stockage du résultat : Selon le processeur, le résultat peut être stocké dans deux registres (le bit de poids faible (LSB) et le bit de poids fort (MSB)) ou dans un seul registre (généralement le LSB uniquement).

Précision des multiplications : Les multiplications doivent être spécifiées comme signées ou non signées. Cela détermine si les opérandes et le résultat sont interprétés comme des nombres positifs ou des nombres signés (pouvant être positifs ou négatifs). C'est crucial pour interpréter correctement le résultat, surtout lorsqu'on manipule des nombres négatifs.

4. Shifter

4.1. Logical shift

Le logical shift revient à faire une multiplication ou une division par une puissance de 2. Cela se fait en ajoutant soit à gauche soit à droite un 0.

4.2. Arithmetic shift

L'arithmétique shift est similaire au logical shift, mais il conserve le bit de signe. Cela signifie que si le bit de signe est 1, il sera ajouté à gauche lors d'un décalage à droite. Cela permet de conserver le signe du nombre et ne s'applique qu'aux nombres signés.

4.3. Rotation

La rotation est une opération de décalage circulaire. Cela signifie que le bit qui sort d'un côté est réintroduit de l'autre côté. Cela permet de déplacer les bits d'un nombre sans perdre d'information.

4.4. Implémentation

- Le registre à décalage n'est pas utilisé (trop lent)
- Utilisation du barrel shifter : un circuit numérique qui permet instantanément les -hifts logiques et arithmétiques ou les rotations
- Un multiplexeur par bit avec en entrée tous les bits à shifter, 0 et 1
- Exemple pour 16 bits : 16 multiplexeurs