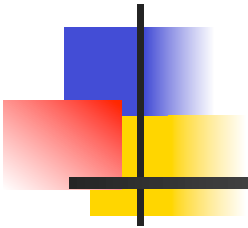


GLEE202

Architecture

Arnaud VIRAZEL
virazel@lirmm.fr





Plan du Cours

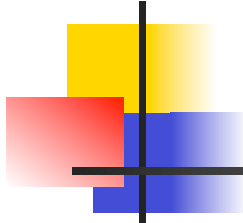
- Chapitre 1 : Introduction
- Chapitre 2 : Unité de Traitement
- Chapitre 3 : Décodage des Instructions
- Chapitre 4 : Interruptions
- Chapitre 5 : Entrées/Sorties
- Chapitre 6 : Circuits d'Interfaces



Architecture

Chapitre 1

Introduction



Plan

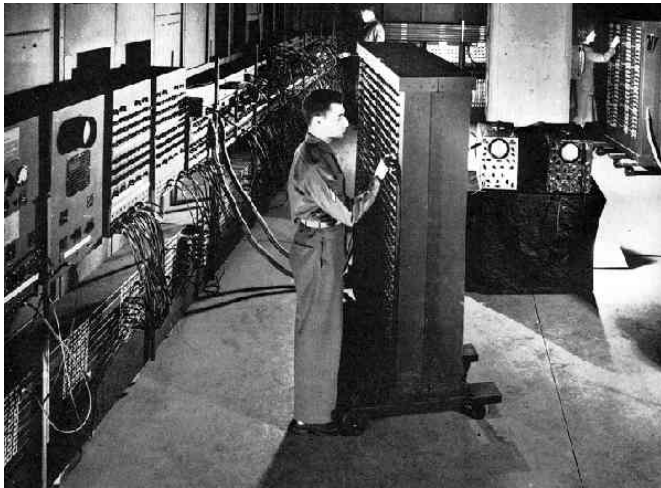
- Historique
- Principes de von Neumann
- Cycle de base du fonctionnement
- Format des instructions
- Exemples d'instruction



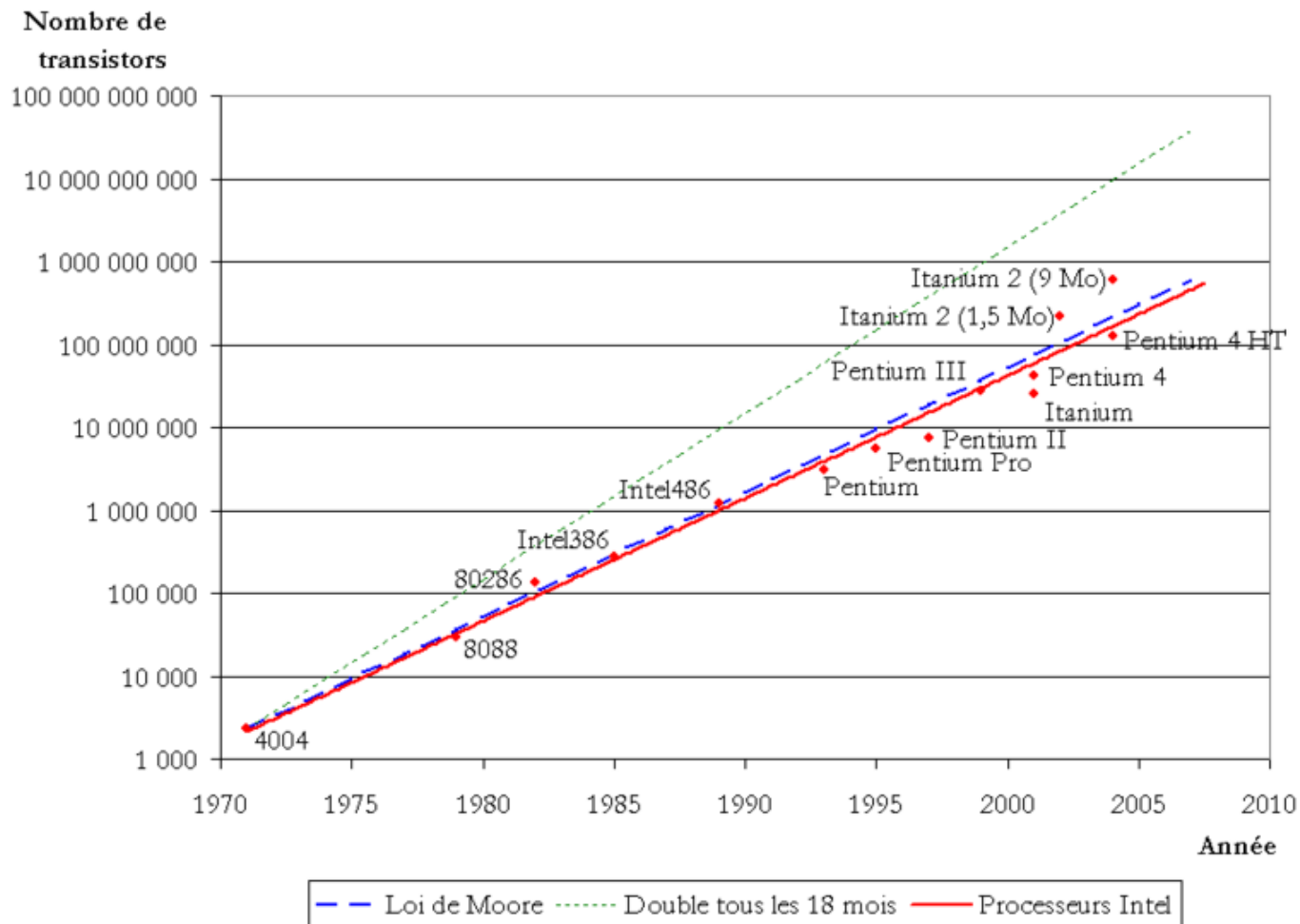
John von Neumann

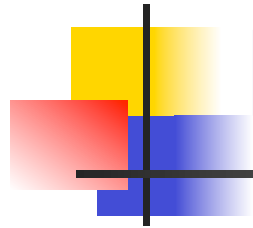
- Mathématicien et physicien américano-hongrois
- Il a établi les bases d'un ordinateur électronique et construit l'une des premières machines en 1945.

Evolution Technologique



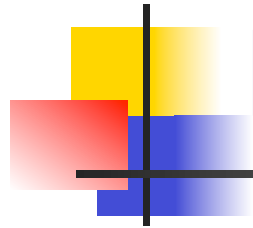
Loi de Moore





Principes de von Neumann

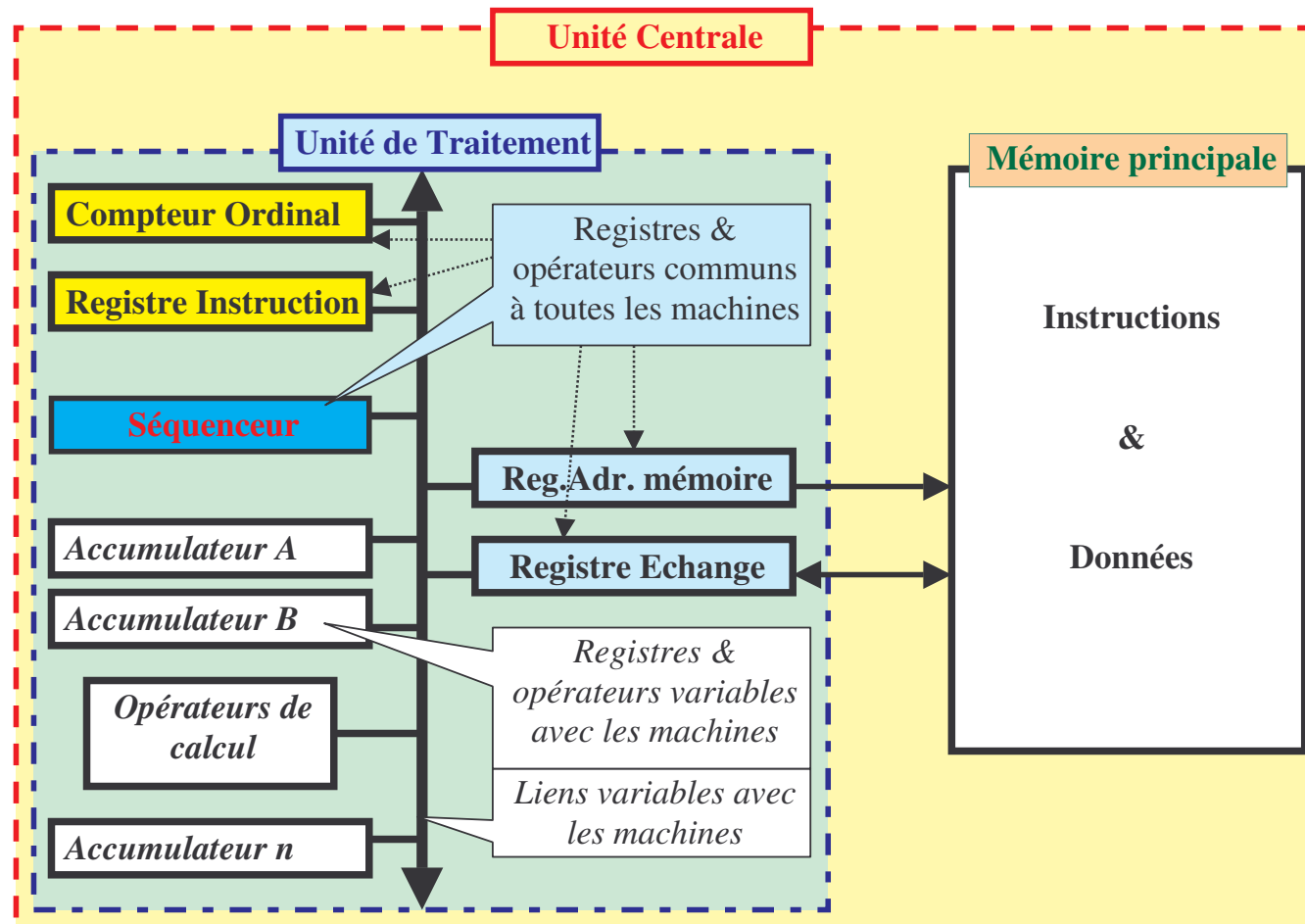
- L'architecture de von Neumann décompose l'ordinateur en 4 parties distinctes :
 - L'unité arithmétique et logique (UAL ou ALU en anglais) ou unité de traitement :
 - son rôle est d'effectuer les opérations
 - L'unité de contrôle :
 - chargée du séquençage des opérations



Principes de von Neumann

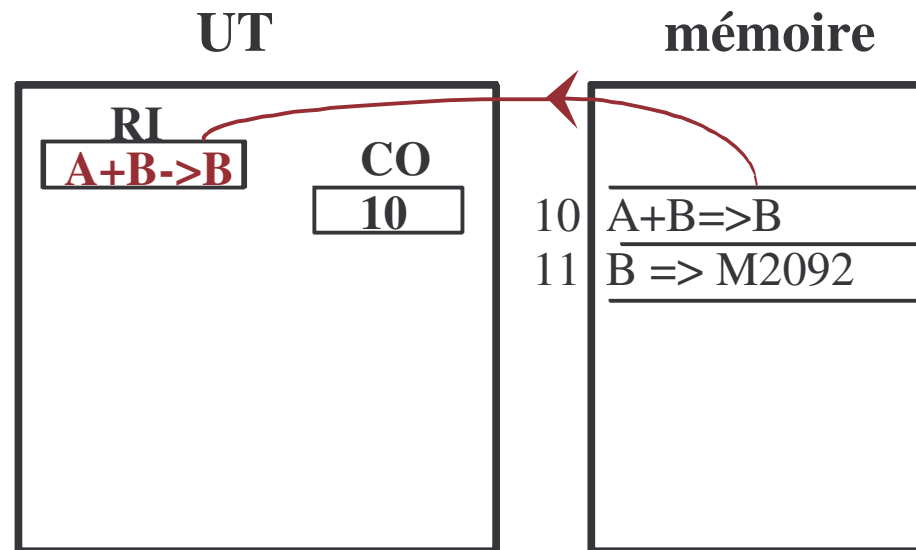
- La mémoire :
 - qui contient à la fois les données et le programme qui dira à l'unité de contrôle quels calculs faire sur ces données. La mémoire se divise entre mémoire volatile (programmes et données en cours de fonctionnement) et mémoire permanente (programmes et données de base de la machine).
- Les dispositifs d'entrée-sortie :
 - qui permettent de communiquer avec le monde extérieur.

Architecture de von Neumann



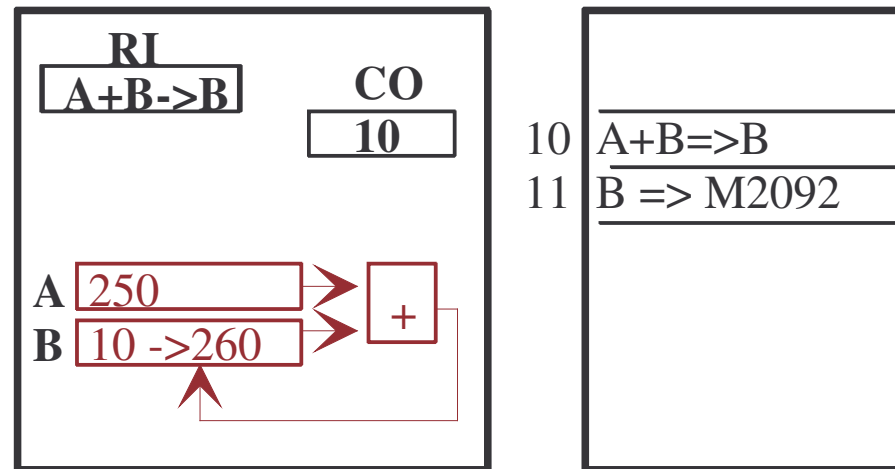
Cycle de base du Fonctionnement

- Recherche de l'instruction ou cycle Fetch



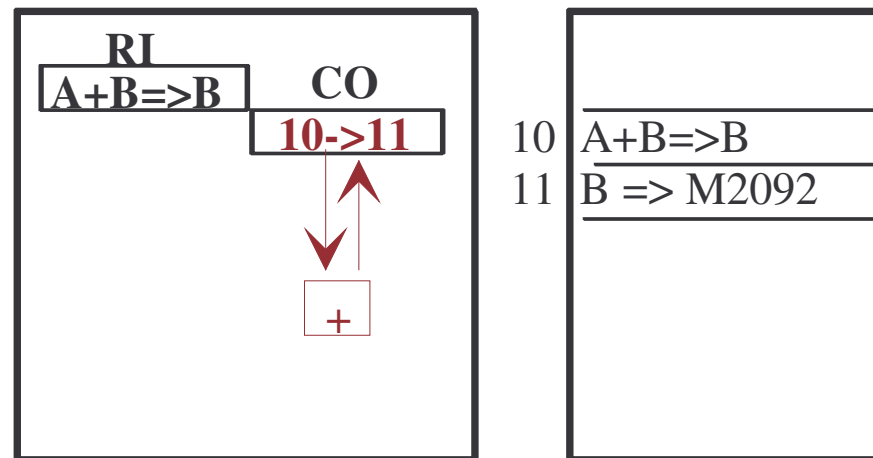
Cycle de base du Fonctionnement

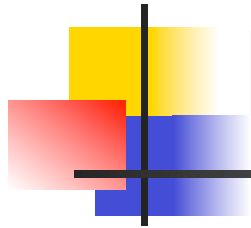
- Exécution de l'instruction



Cycle de base du Fonctionnement

- Progression du compteur ordinal pour pointer l'instruction suivante

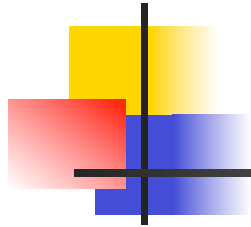




Format des Instructions

- Chaque instruction est représentée en mémoire par une combinaison de bits (un ou plusieurs mots selon sa complexité)
- La combinaison se compose d'une partie Code Opération et d'autant de (champs) de bits qu'il faut pour définir les autres paramètres de l'instruction.

COP	/	MA	/	RA
-----	---	----	---	----



Le Code Opération - COP

- Il définit le(s) objectif(s) de l'instruction
- Exemples :
 - LOAD A – charger une valeur dans le registre A
 - ADD B – Additionner une valeur au registre B et placer le résultats dans B.



Le Mode d'Adressage - MA

- Il définit le type d'accès à la mémoire
- Exemples :

- **Immédiat** – passage par valeur

COP / MA / RA

l'Opérande = RA

- **Direct** – passage par adresse
(pointeur en C)

COP / MA / RA

AdresseOpérande = RA



Le Mode d'Adressage - MA

- **Indirect** – passage par adresse d'adresse (pointeur de pointeur en C)

COP / MA / RA

AdresseOpérande = (RA)
() signifie *CONTENU* de la mémoire d'adresse RA

- **Relatif** – passage par adresse relatif à la position de l'instruction

α : **COP / MA / RA**

AdresseOpérande = α + RA

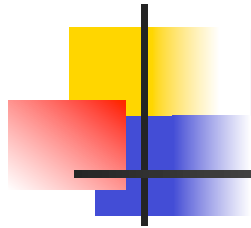


Le Mode d'Adressage - MA

- **Indexé** – passage par adresse
(adressage des tableaux en C)

COP / MA / RA

AdresseOpérande = RIndex+RA

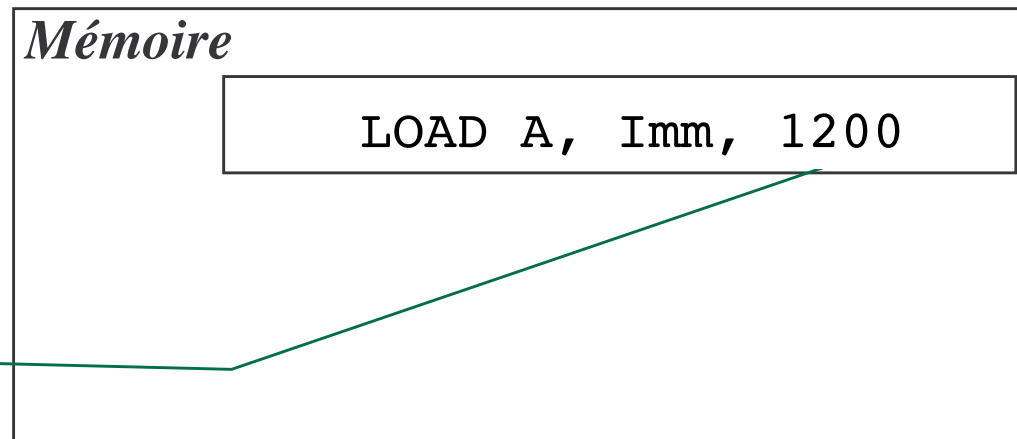
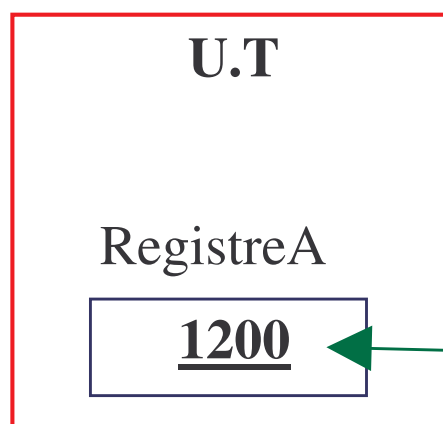


La Référence Adresse - RA

- Il définit la valeur ou adresse (en fonction du Mode d'Adressage à manipuler avec l'instruction
- Exemples :
 - LOAD A, Immédiat, 12 – 12 est une valeur qui doit être chargée dans le registre A
 - LOAD B, Direct, 153 – 153 est une adresse à laquelle se trouve la valeur à charger dans le registre B.

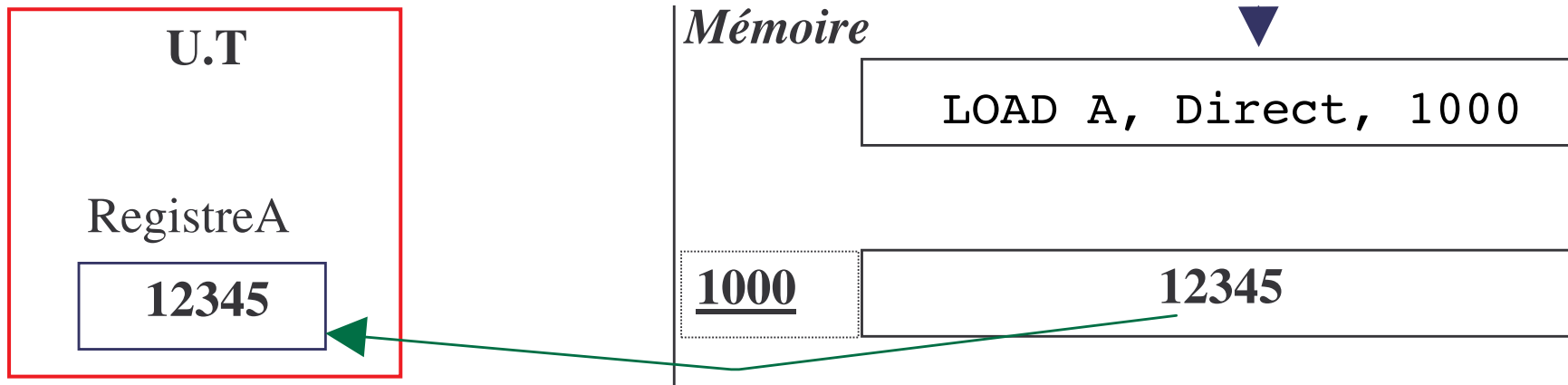
Exemples

■ Adressage **Immédiat**



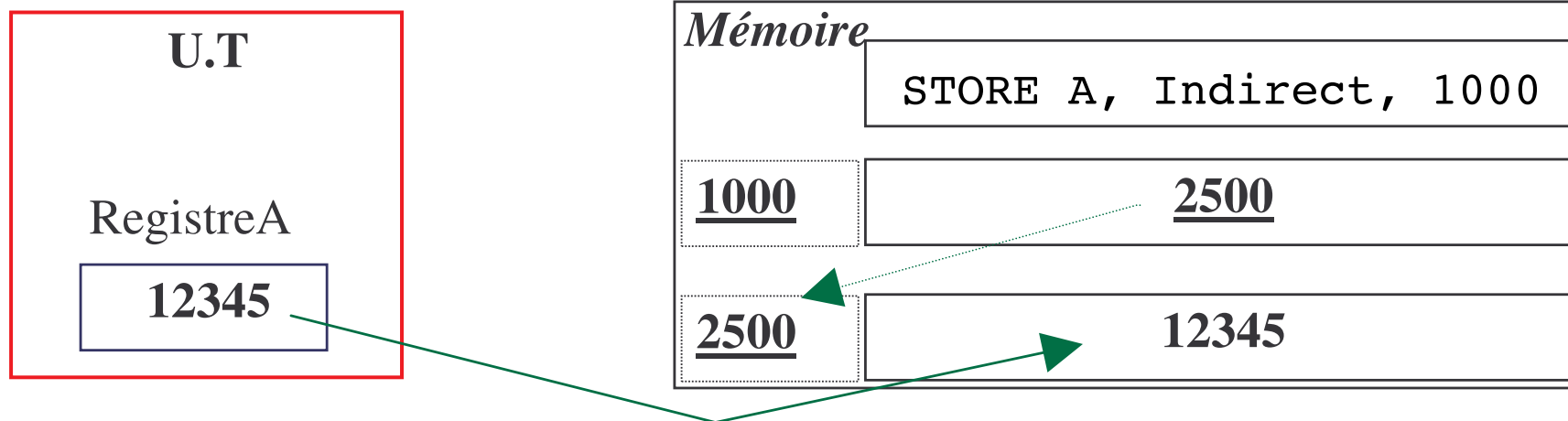
Exemples

■ Adressage **Direct**



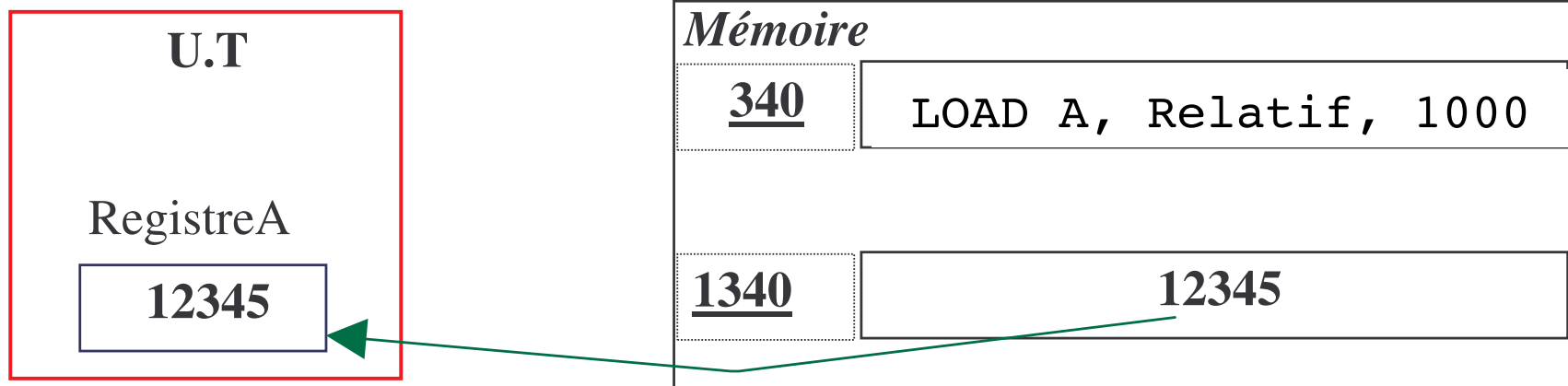
Exemples

■ Adressage **Indirect**



Exemples

■ Adressage **Relatif**



Exemples

■ Adressage **Indexé**

