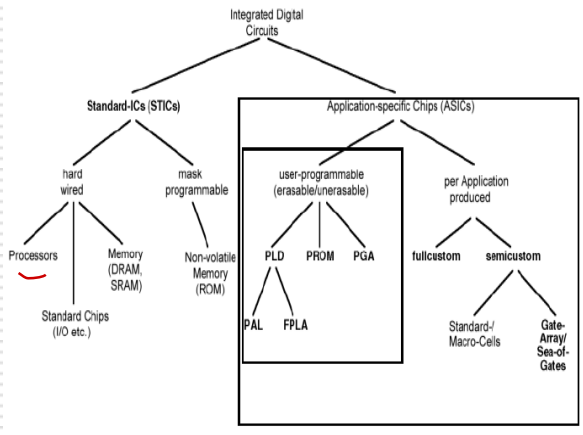
Introduction au microprocesseur

Du câblé au programmé

Solutions numériques

- 1- Définition du problème
- 2- Conception d'une solution
  - VHDL
  - Schéma ?
  - Autres :
- 3- Réalisation
- 4- Vérification
- 5- etc.
- 6- etc.



Logique câblée

- Toute réalisation d'un circuit spécifique à une fonction
  - Circuit intégré spécifique
  - Circuit programmable
  - Etc.
    - => logique câblée
    - « hardwired »
    - Câblée en 'dur'

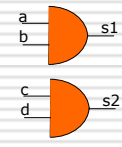
Limites de la logique câblée

- Modification fonction => recommencer conception-réalisation
  - Fonctions ne peut être modifiée « instantanément »
- Réutilisation de ressources
  - Peu, sinon conception trop compliquée
  - Extension en surface silicium
    - N'est pas infinie !

Réutiliser la logique

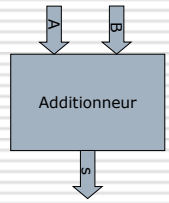
- Entity…
- end

- Architecture
  - Begin
    - S1 = a and b;
    - S2 = c and d;
  - End titi;
- 2 and en VHDL => 2 portes et réalisées
- Comment réutiliser les ressources ?



Réutilisation de Ressources

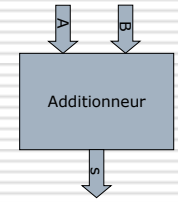
- Réaliser un additionneur n bits
- Bien l'optimiser



7

A + B

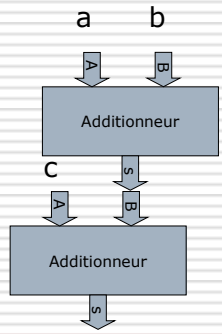
- Positionner A et B fil par fil !
- Positionner F !
- Recueillir Sortie S



8

a + b + c ?!

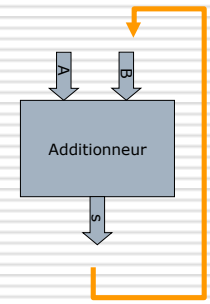
- Soit utiliser un deuxième additionneur



9

a + b + c ?!

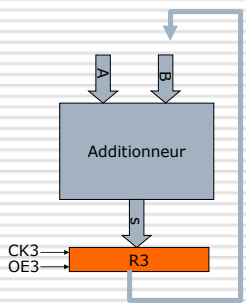
- Soit utiliser un seul additionneur, et appliquer successivement :
  - A et B, et récupérer S,
  - puis appliquer C et s, et récupérer S finale
- Mais il faut mémoriser S entre temps



10

a + b + c ?!

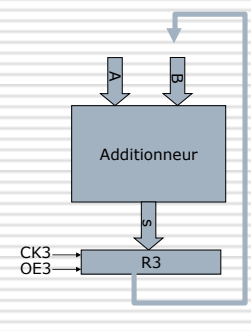
- Soit utiliser un seul additionneur, et appliquer successivement :
  - A et B, et récupérer S,
  - puis appliquer c et s, et récupérer s finale
- Mais il faut mémoriser s entre temps



11

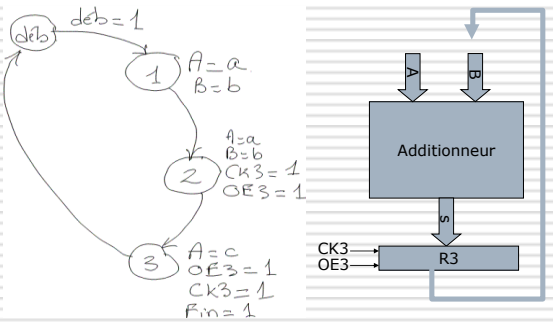
Séquence

- 1ere étape
  - A + B et S dans un registre
- 2e étape
  - S dans R3
- 3e étape
  - R3 sur R2
- 4e étape
  - S dans R3
- 5e étape
  - Fin calcul



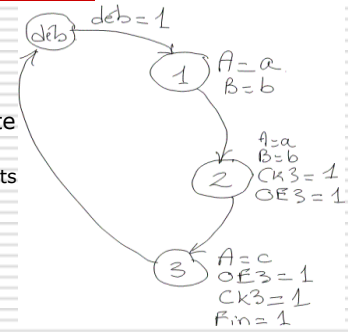
12

Diagramme d'états



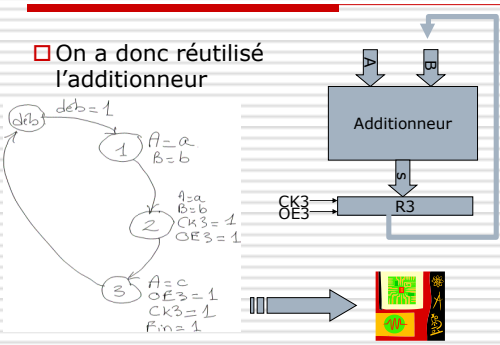
Machine à états

- Modélisation en VHDL
- Réalisation directe en silicium
  - Pas de composants programmables



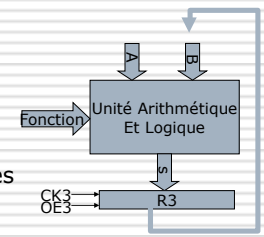
Réutilisation de ressources

- On a donc réutilisé l'additionneur



Bloc logique multifonction

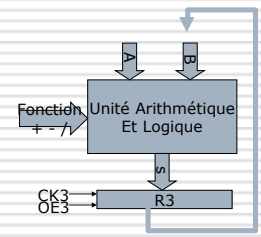
- Non seulement additionneur
  - Mais additionneur, soustracteur, décalage, etc.
  - Sélection de la fonction par des lignes de fonction



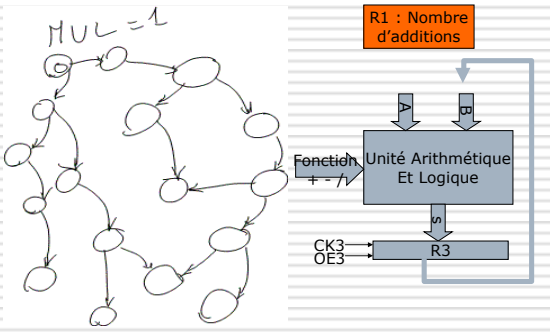
Multiplication*

- Difficile à concevoir
  - Surtout pour 16 bits * 16 bits = 32 bits !
    - Ou 32 * 32 = 64 bits !
- 2 solutions :
  - 1- Table de vérité géante
    - 16 bits * 16 bits
      - 2^16 * 2^16 lignes dans la table !!
  - 2- a * b = a + a, b fois
    - 6 * 4 = 6 + 6 + 6 + 6
    - *remarque TP multiplieur

Réalisation d'une multiplication

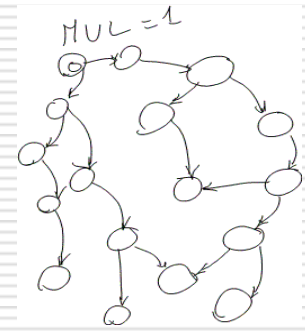


Réalisation d'une multiplication



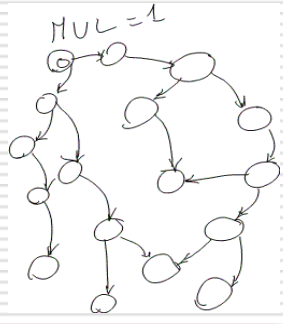
Réutilisation de ressources

- On a donc réutilisé l'additionneur
- Mais on doit fournir :
  - Mul = 1
  - Signal qui démarre la séquence
  - Les *opérandes* a et b
- On appellera ces éléments
  - une *instruction*



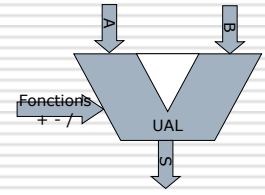
Terminologie

- La machine à états qui réalise l'instruction
  - Micro-séquence

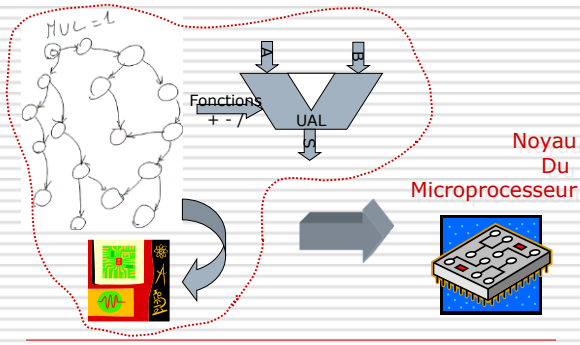


Bloc logique multifonction

- Symbole exacte d'une UAL



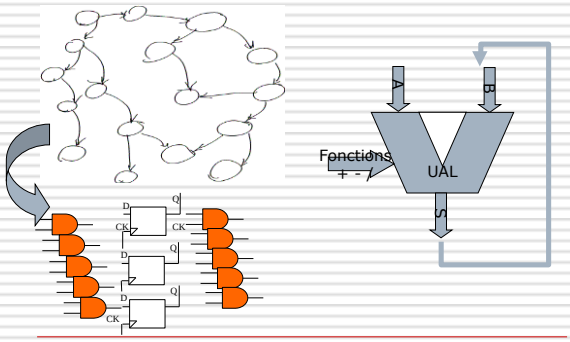
Microprocesseur



Résumons

- On a réalisé une multiplication a*b
  - En utilisant un seul additionneur
  - En multipliant
    - Le plus grand (a) par lui-même, b fois
    - $6 * 4 = \underbrace{6 + 6 + 6 + 6}_{4 \text{ fois}}$
    - $6 * 4 = \underbrace{4 + 4 + 4 + 4 + 4 + 4}_{6 \text{ fois}}$

Micro-séquence en Machine à états



Format instruction

- Code binaire
  - 11001001001001001001010 = $4E1F ?
    - Adresse Micro-séquence
    - Registres
  - Mnémonique
    - MUL R1,R2 *(R1 * R2 -> R2)*
      - *R1 : source, R2 : destination*
    - MUL #12,R1 *(12* R1 -> R1)*
      - *12 source = valeur directe, R1 destination*

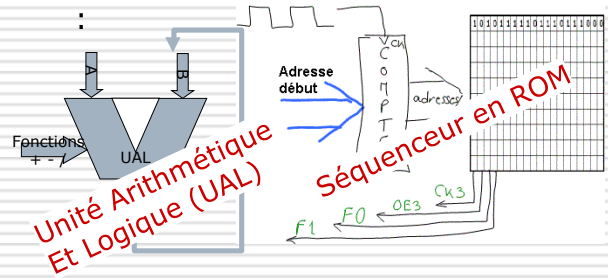
Séquence déclenchée par une instruction

- Micro-code
- Micro-séquence
- Micro-programme

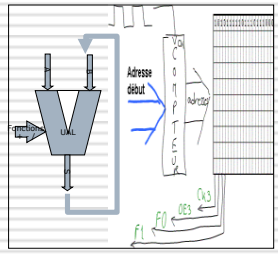
- La séquence est soit :
  - Micro-codée
    - tableau en ROM
  - Câblée
    - machine à états en logique (bascule D, logique transitions, logiques sorties)

Résumons

- Le noyau du microprocesseur est :

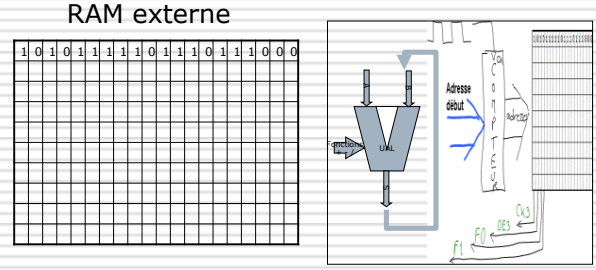


Comment présenter les instructions

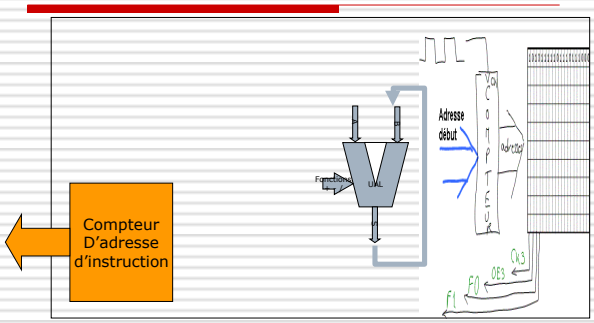


Comment présenter les instructions

- Elles seront stockées en mémoire RAM externe

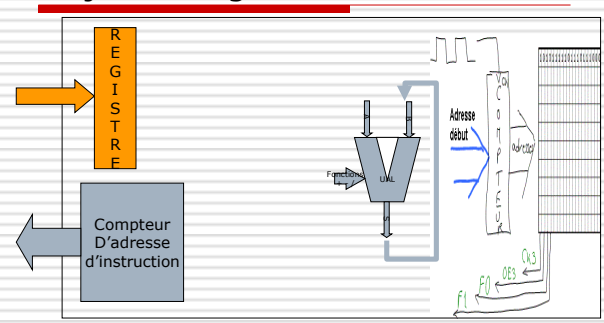


Rajouter compteur d'adresse d'instruction



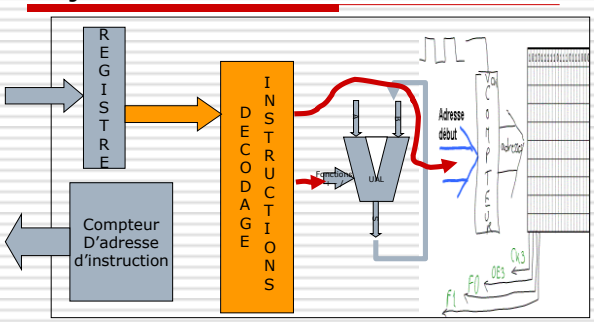
31

Rajouter registre instruction



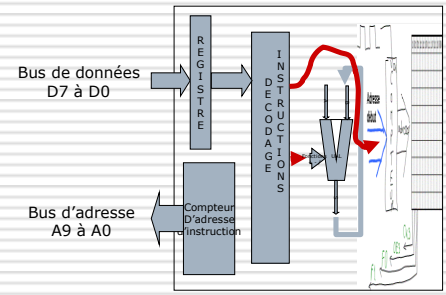
32

Rajouter décodeur d'instructions



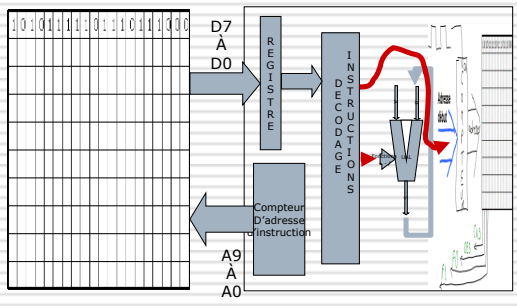
33

Interface externe



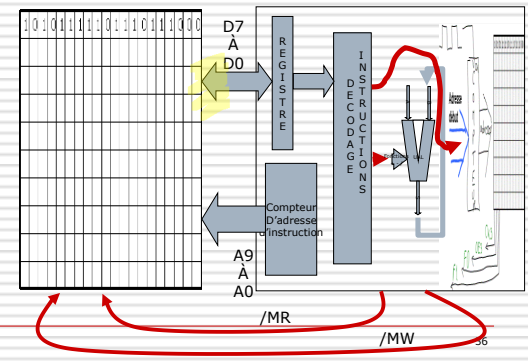
34

Interface mémoire processeur



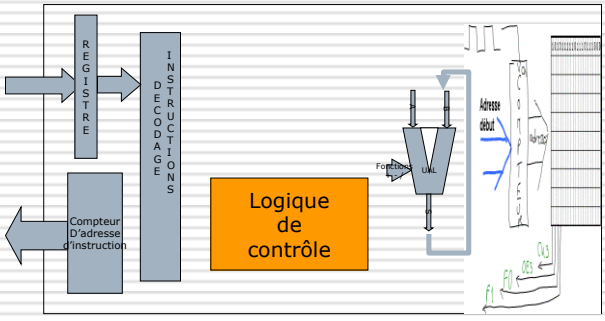
35

Interface mémoire processeur



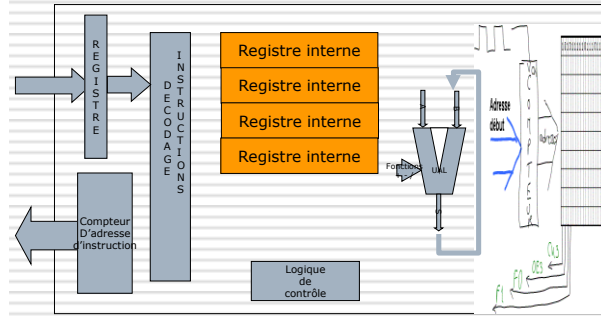
36

Rajouter logique de contrôle



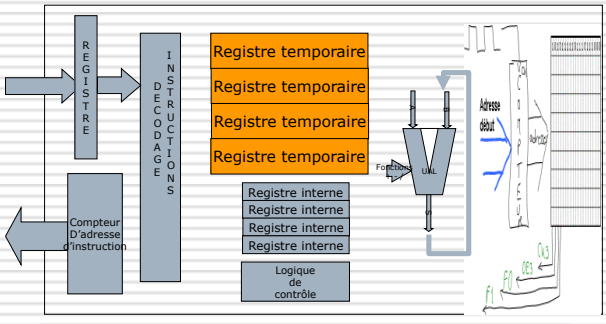
37

Rajouter registres internes



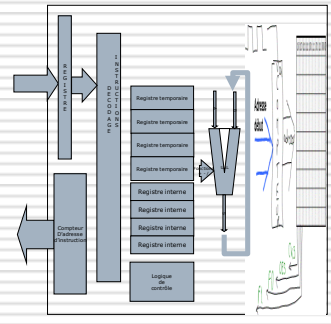
38

Rajouter registres temporaires



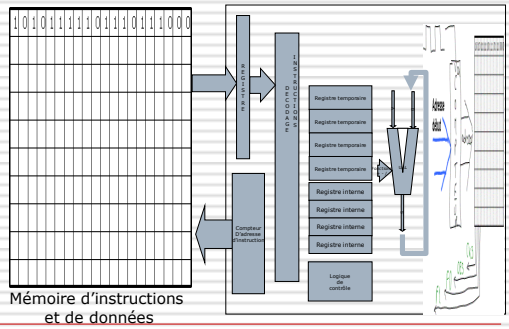
39

Un microprocesseur !



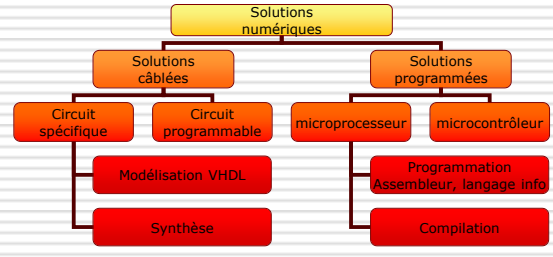
40

Modèle Von Neuman



41

Solutions numériques



42