Des circuits aux systèmes sur puces

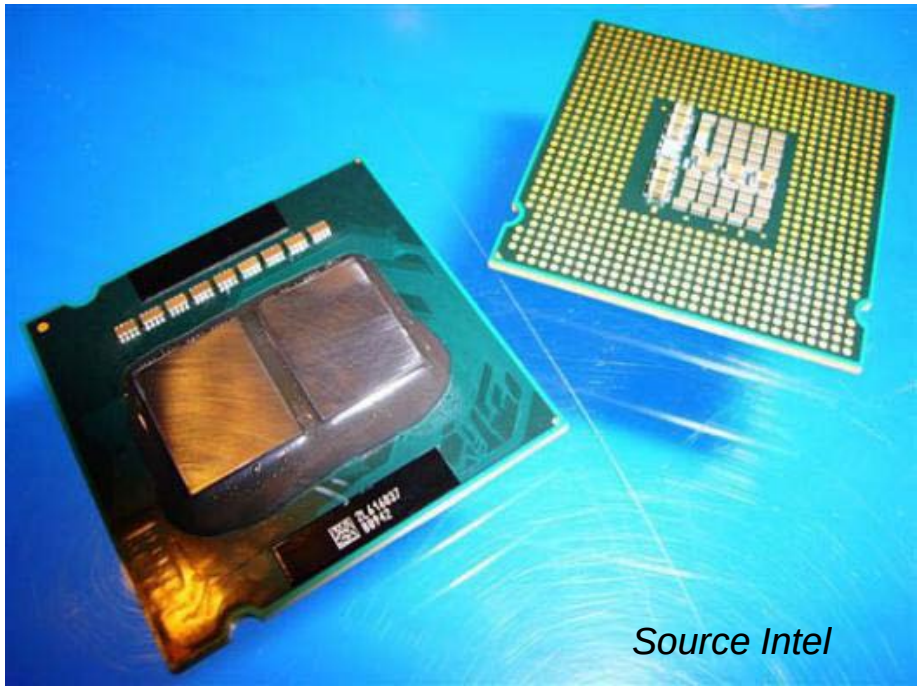
Gérard Berry (@college-de-france.fr)
Laurent Thénié (lthenie@cadence.com)

Chaire d'innovation technologique Liliane Bettencourt

Collège de France

1e février 2008

Les circuits

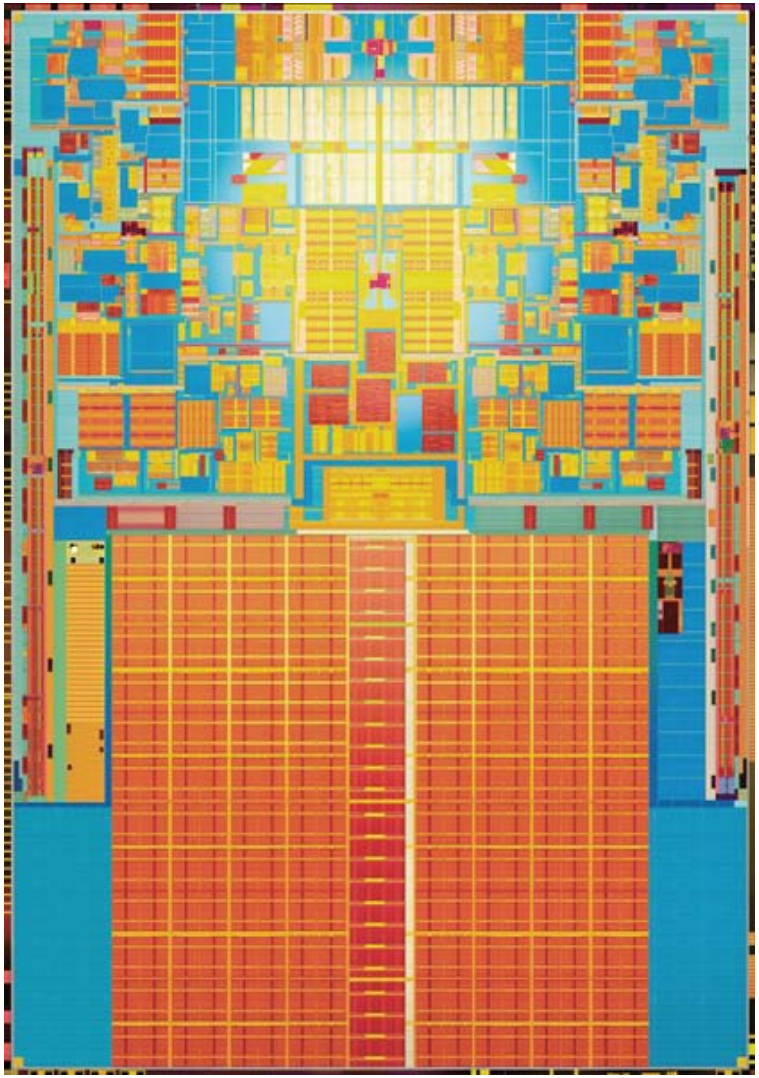


Source Intel

SoC = System on Chip

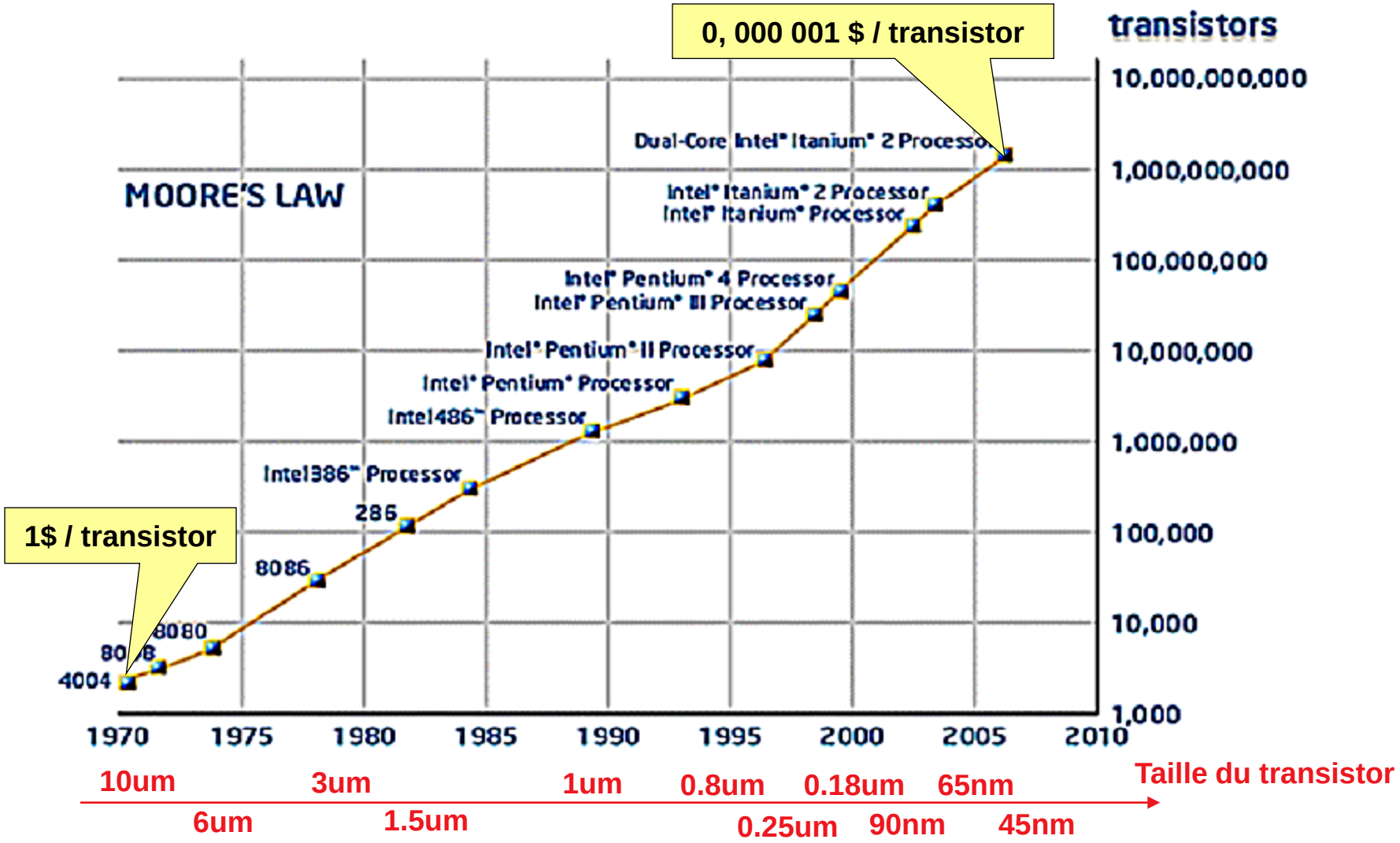
microprocesseurs de PC

téléphones, DVD, TV, GPS



Source Intel

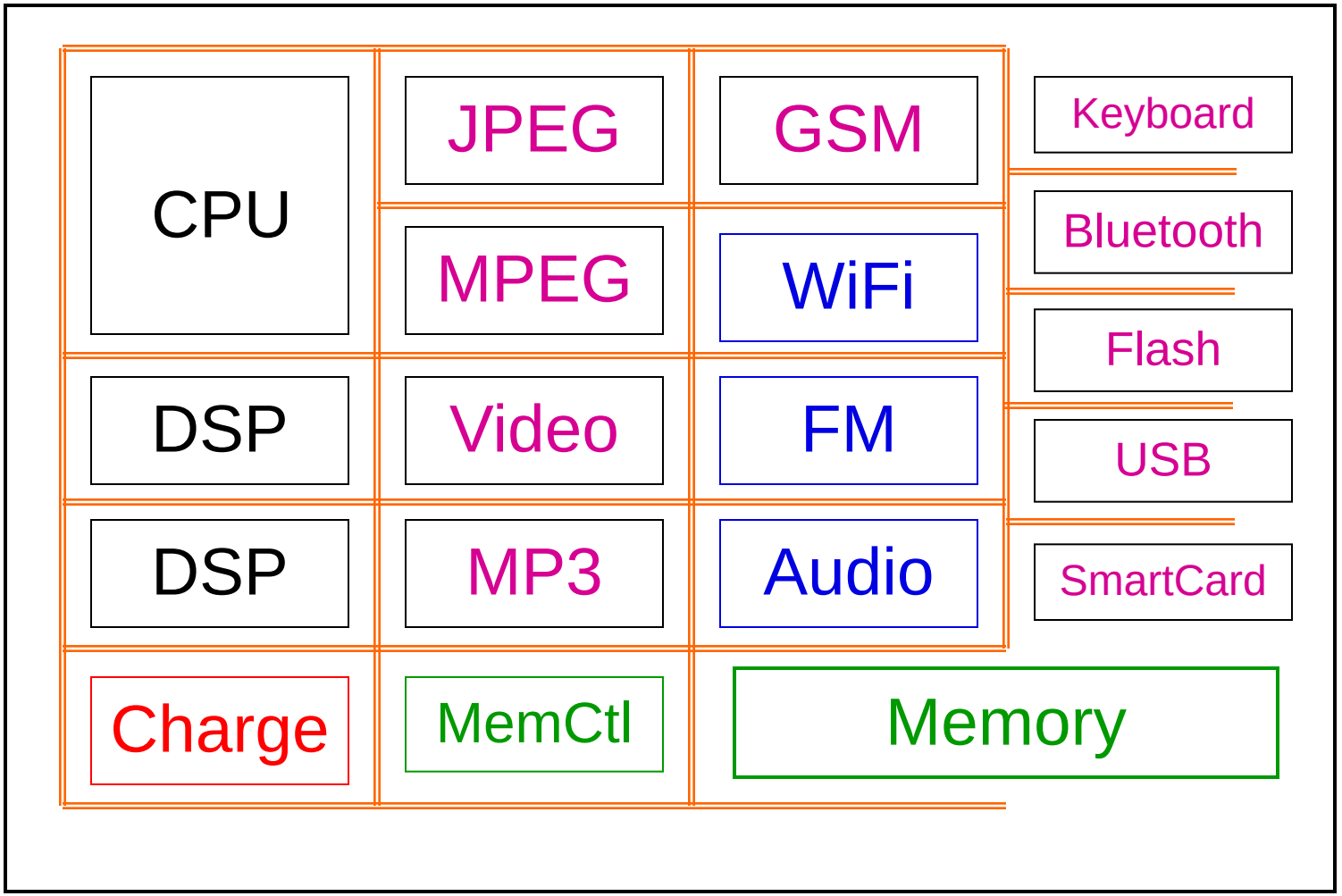
La loi de Moore



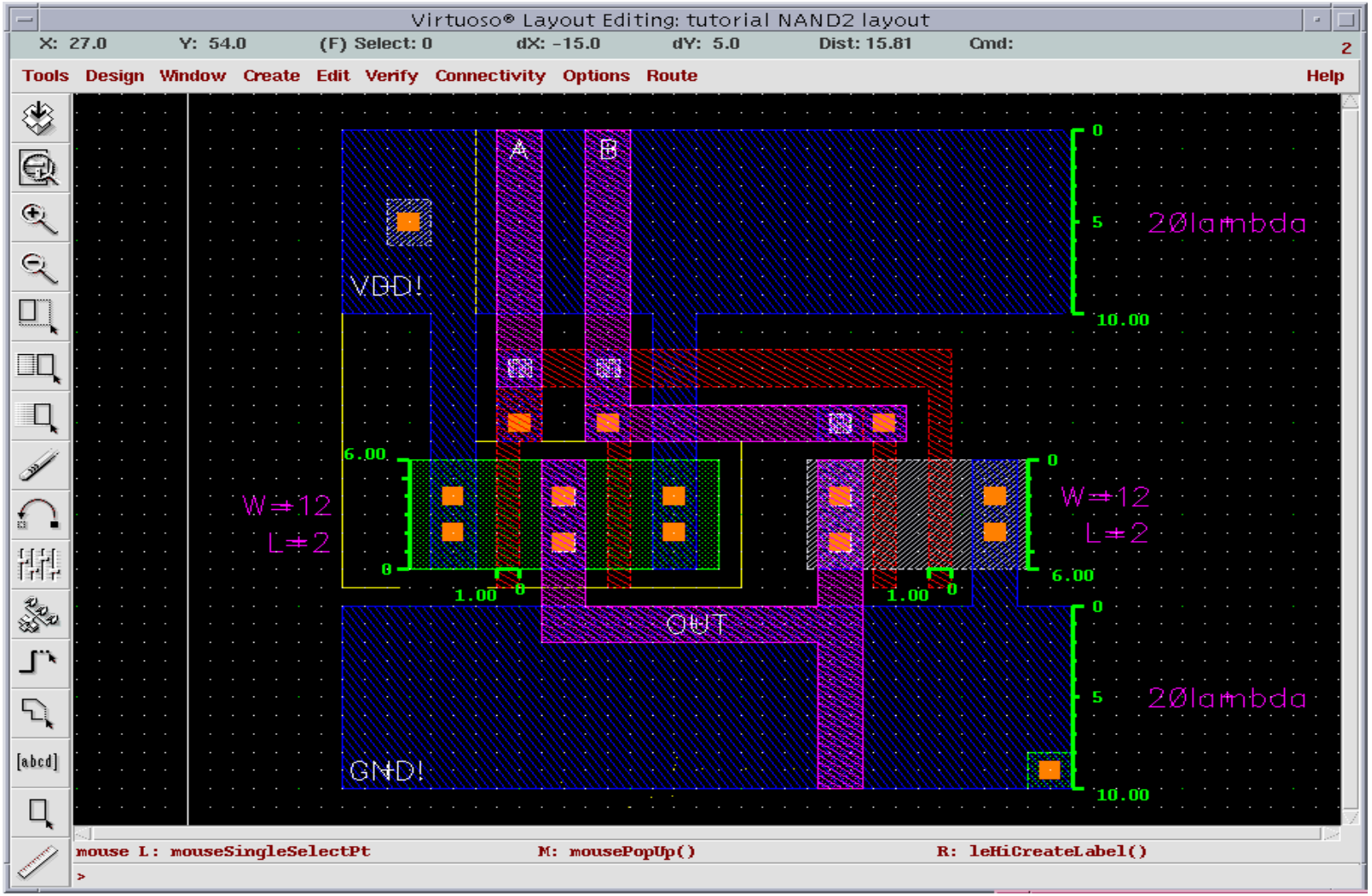
Les types de (sous) circuits

- Microprocesseurs : calculs généraux
  tous types de données, tous types d'ordres
- Digital Signal Processor (DSP) : calculs numériques généraux
  spécialité : addition / multiplication, matrices
- Accélérateurs graphiques : calculs parallèles sur images
- Accélérateurs divers : compression image / son, cryptographie, ...
- Contrôleurs de périphériques : mémoires, disques, réseau, ...
- Gestionnaires internes : horloges, puissance, ...

System on Chip : téléphone, DVD, TV, ...

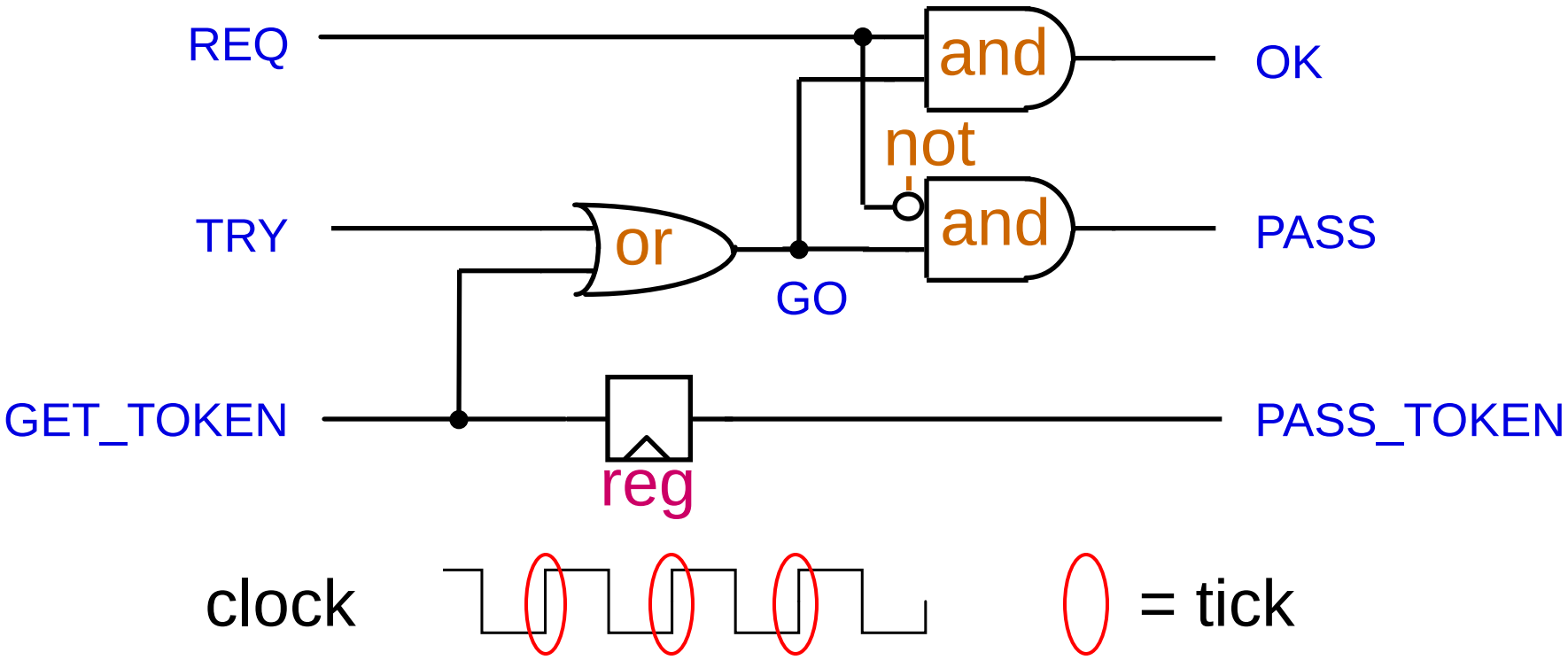


Une porte de base (NAND)



Source Cadence Design Systems

RTL = Register Transfer Level

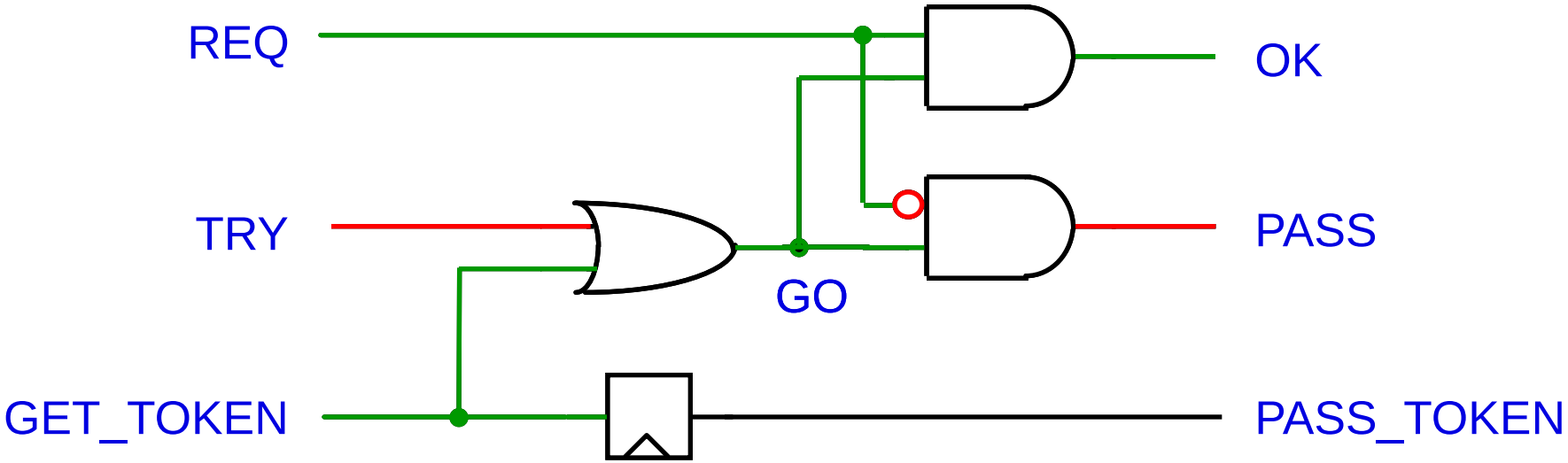


Calcul cyclique

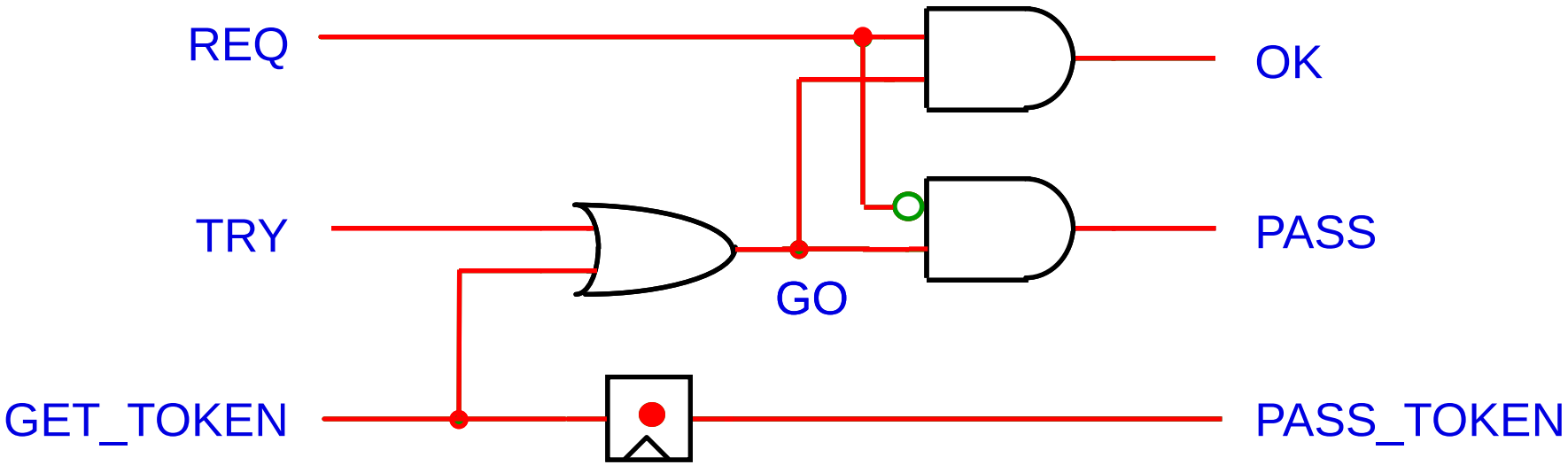
and, or, not : dans le cycle

reg : échantillonnage au tick

Propagation combinatoire

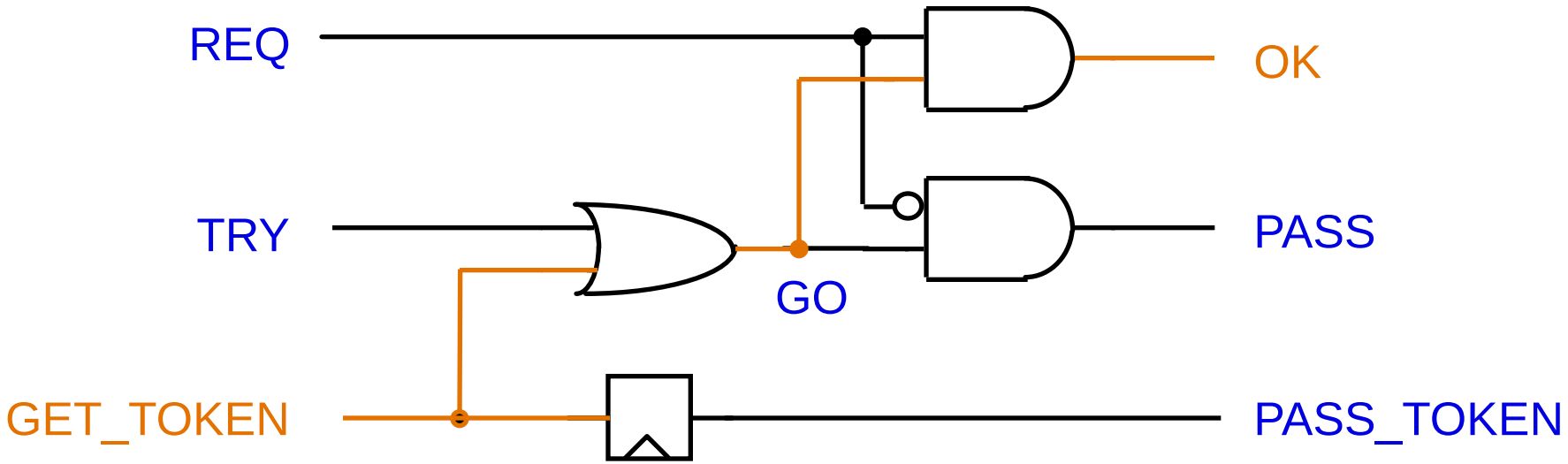


Echantillonnage



tick !

Chemin critique

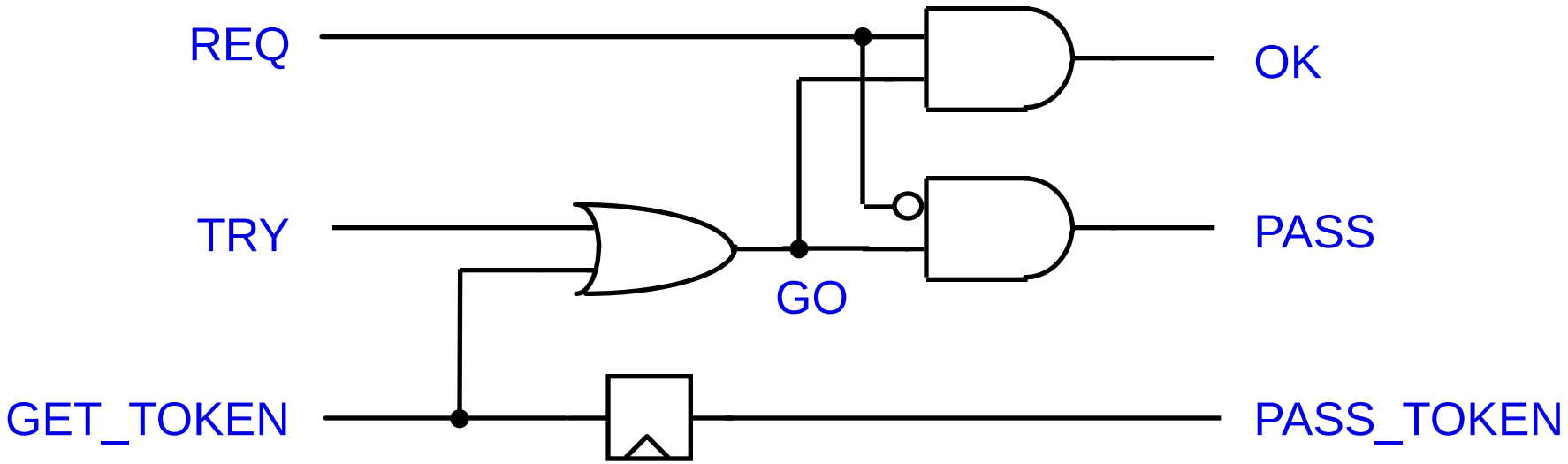


Chemin critique = le plus long à stabiliser

détermine la fréquence du circuit

1 ns $\rightarrow$ 1 GHz

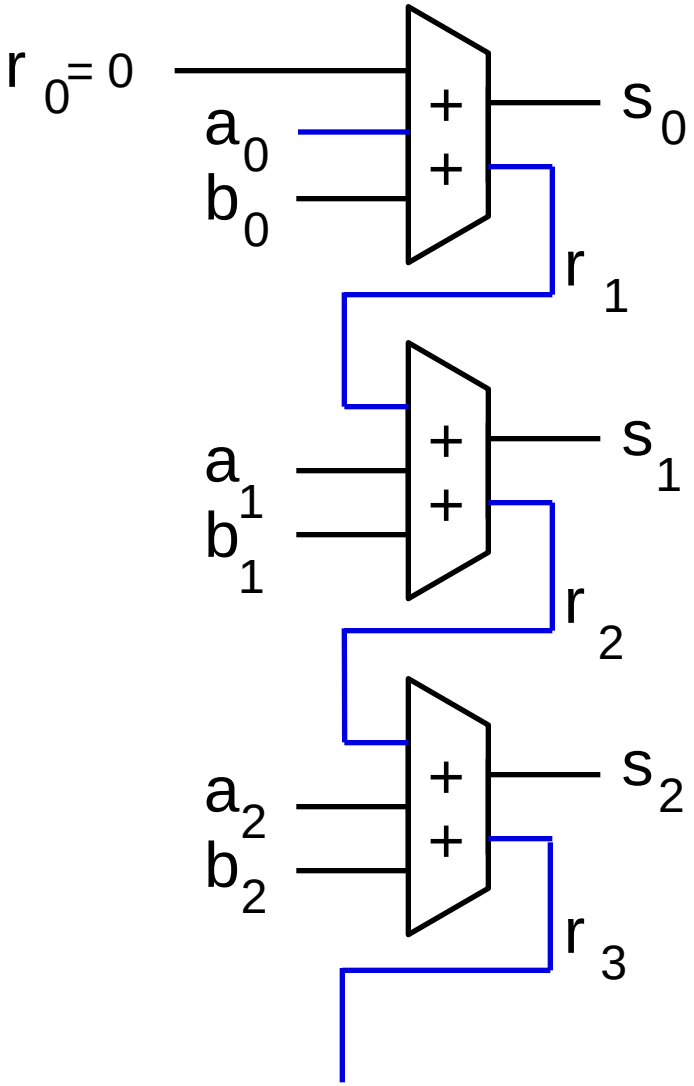
Vision logique zéro-délai



OK = REQ and GO
PASS = not REQ and GO
GO = TRY or GET_TOKEN
PASS_TOKEN = reg(GET_TOKEN)

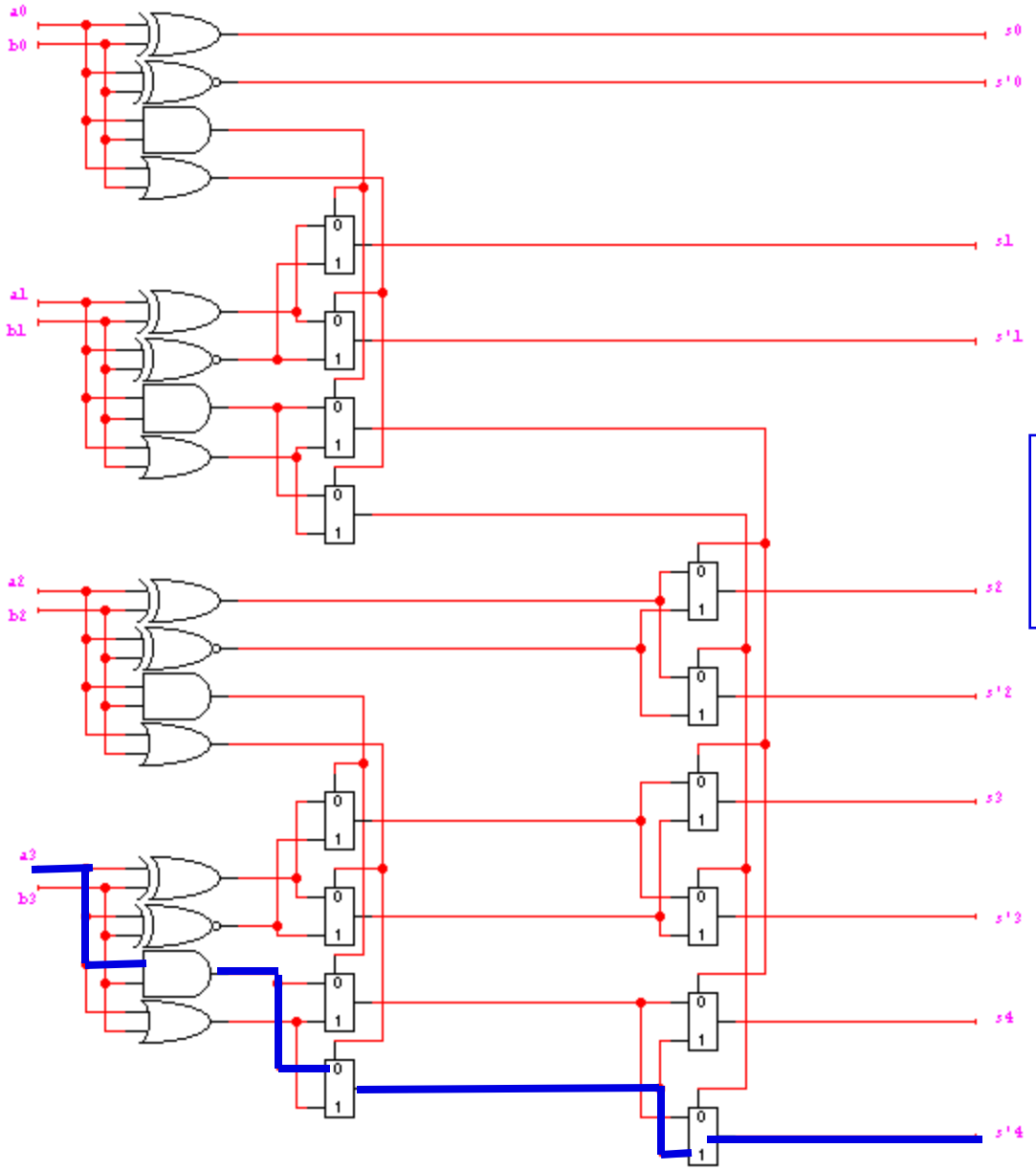
attendre le temps critique = résoudre les équations!

Circuit : propagation de retenue



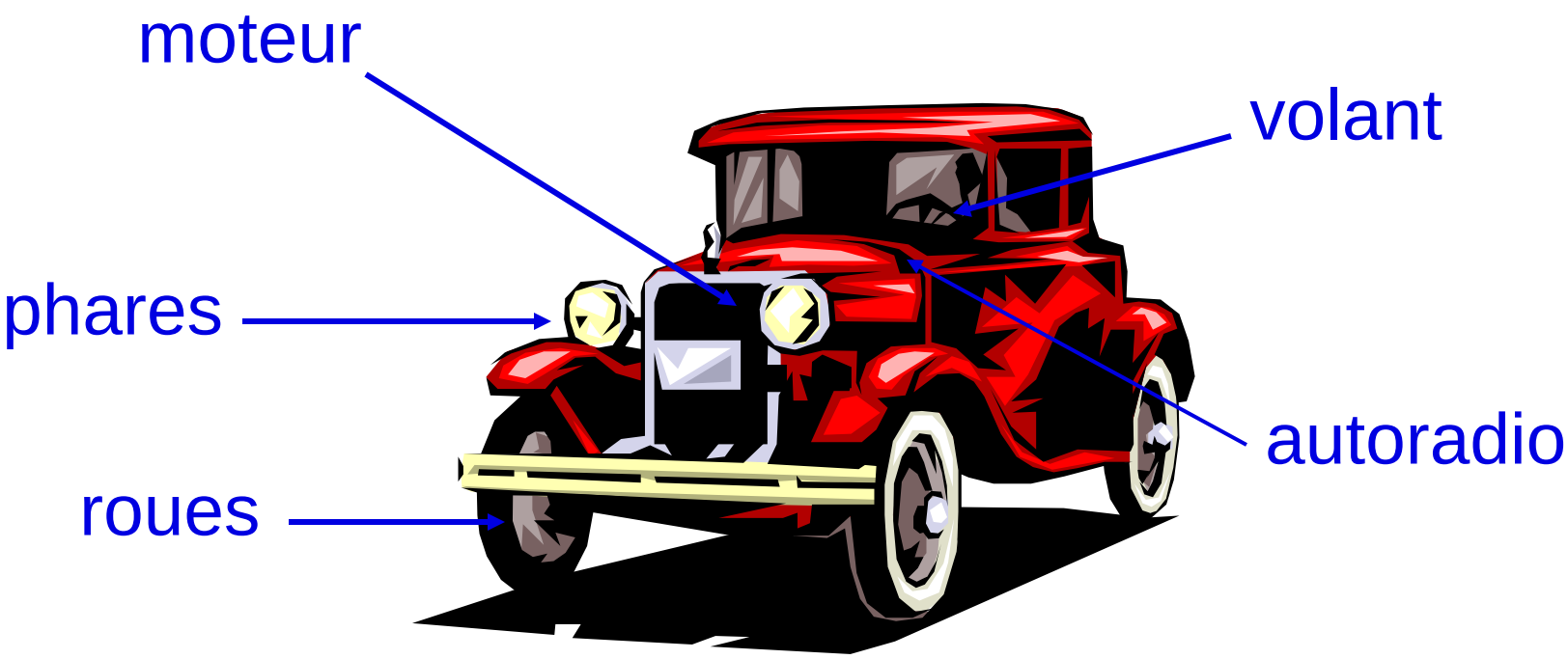
Pour n bits
temps n

Additionneur von Neumann



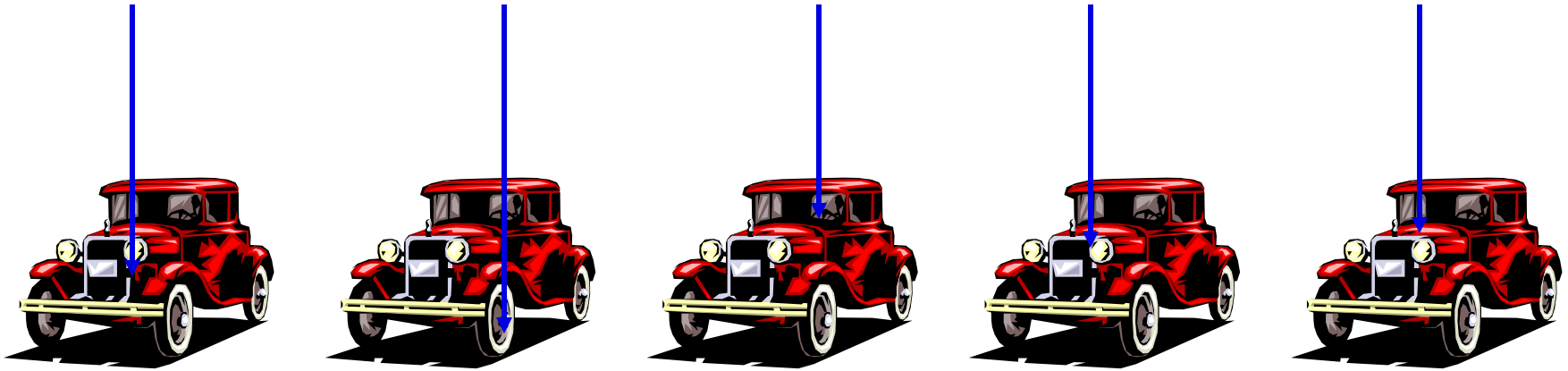
Pour n bits
temps log(n)

Pétrole et voiture : approche naïve

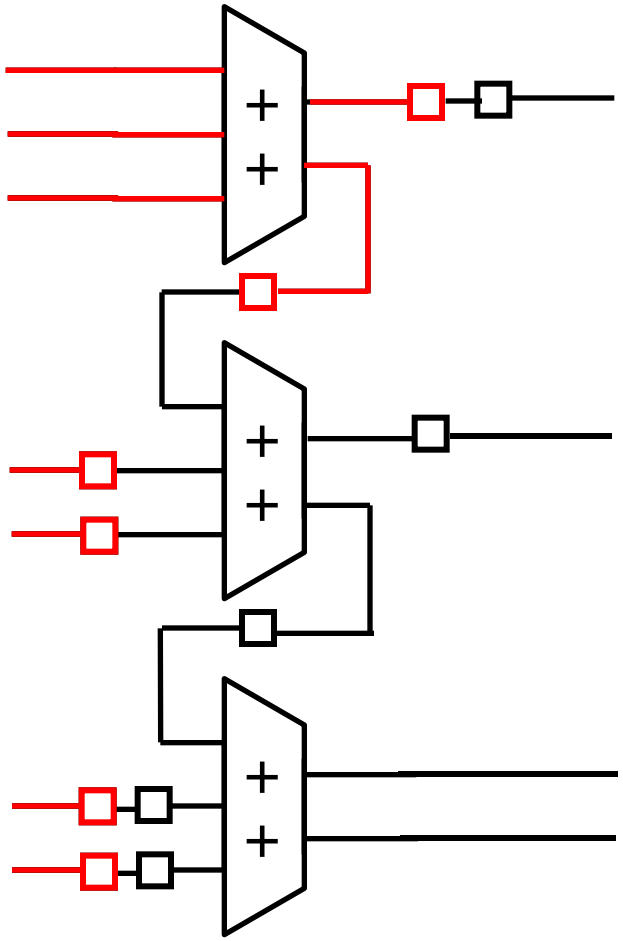


Pipeline et chaîne de montage

moteur roues volant phares autoradio

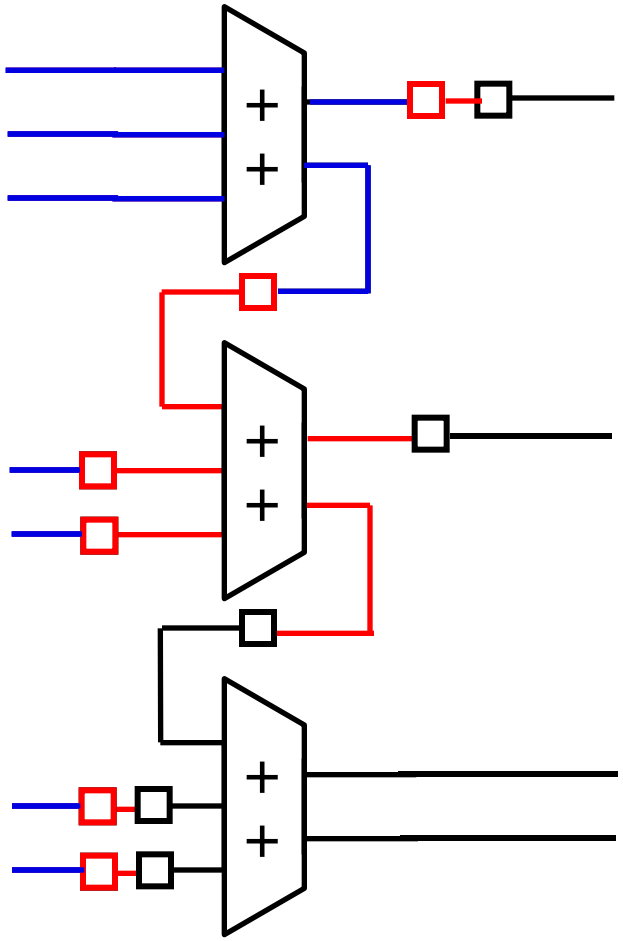


Additionneur pipeline

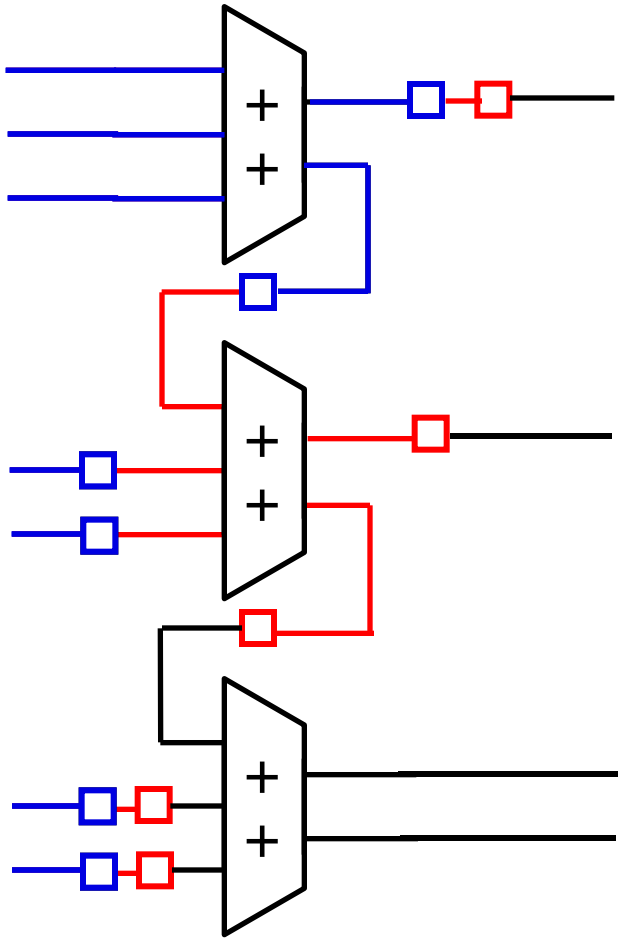


tick !

Additionneur pipeline

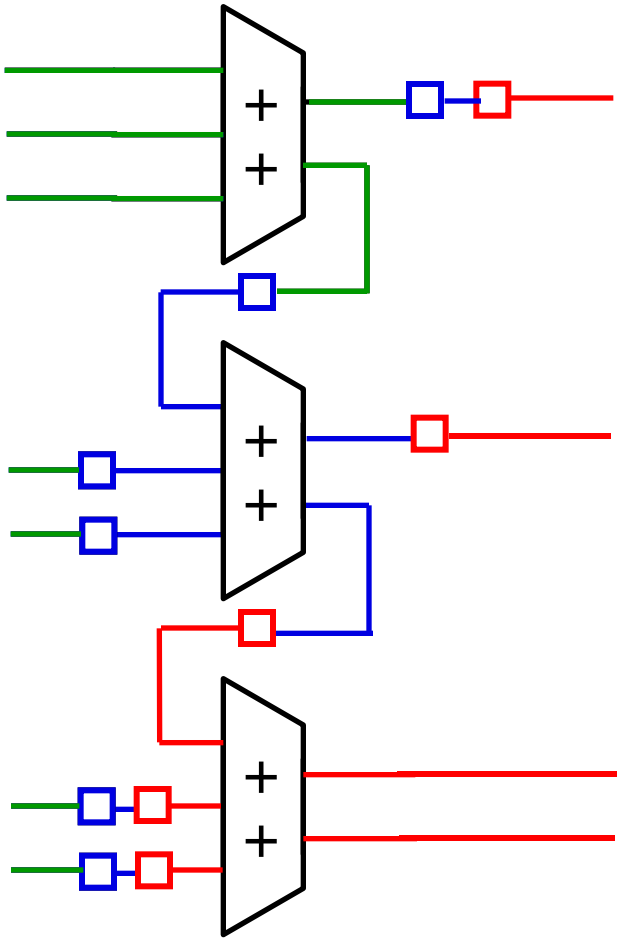


Additionneur pipeline

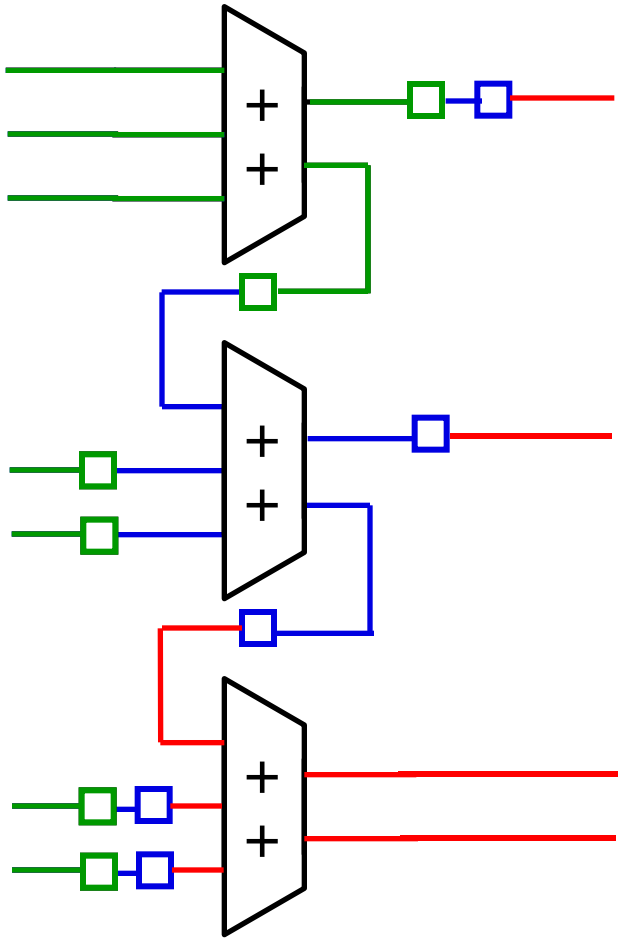


tick !

Additionneur pipeline

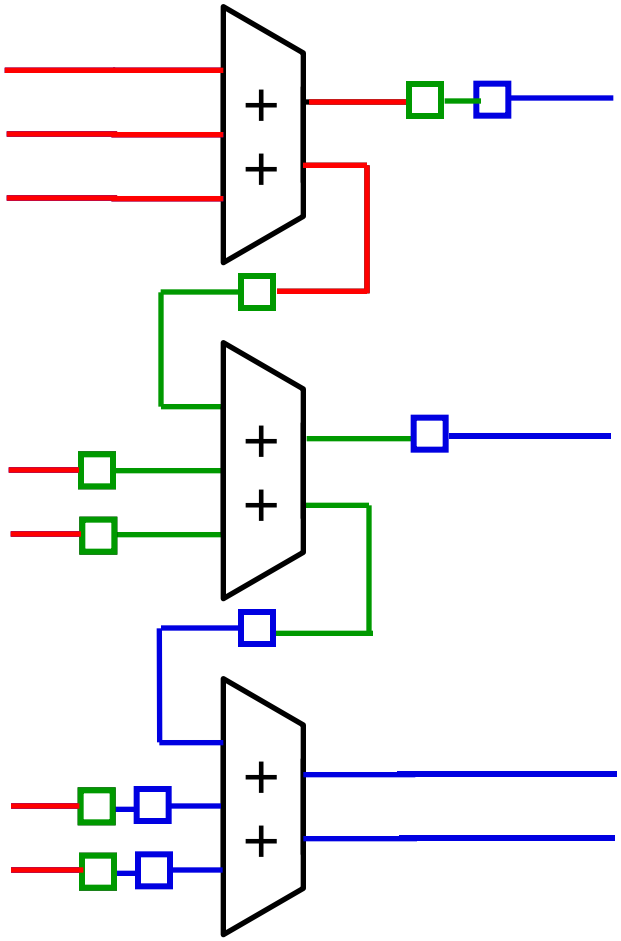


Additionneur pipeline

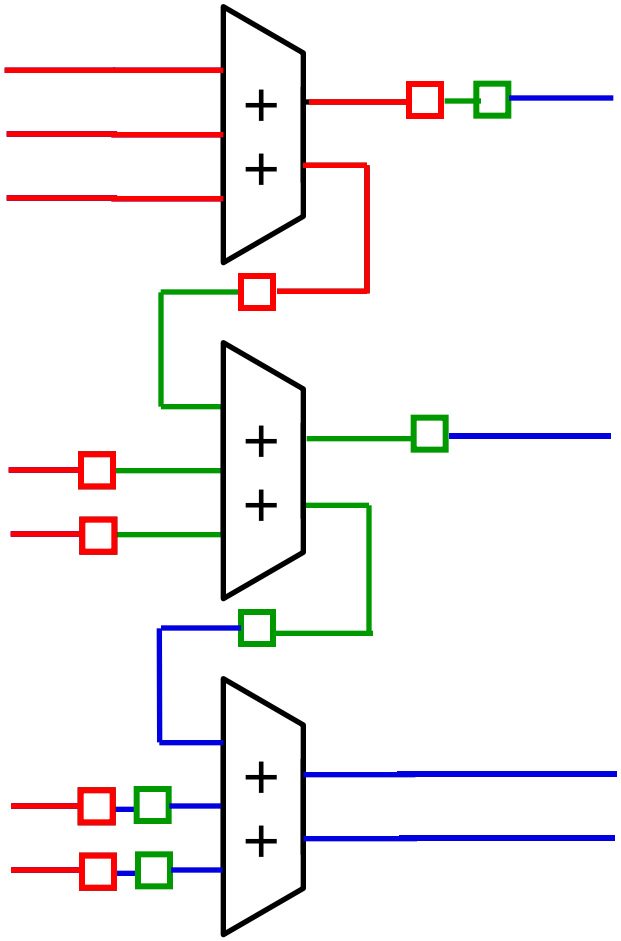


tick !

Additionneur pipeline

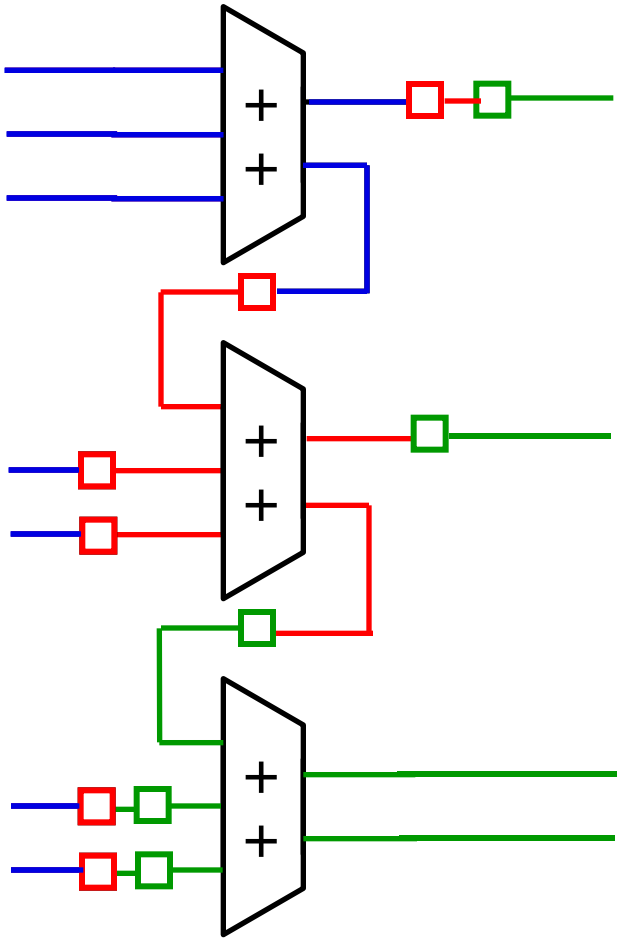


Additionneur pipeline



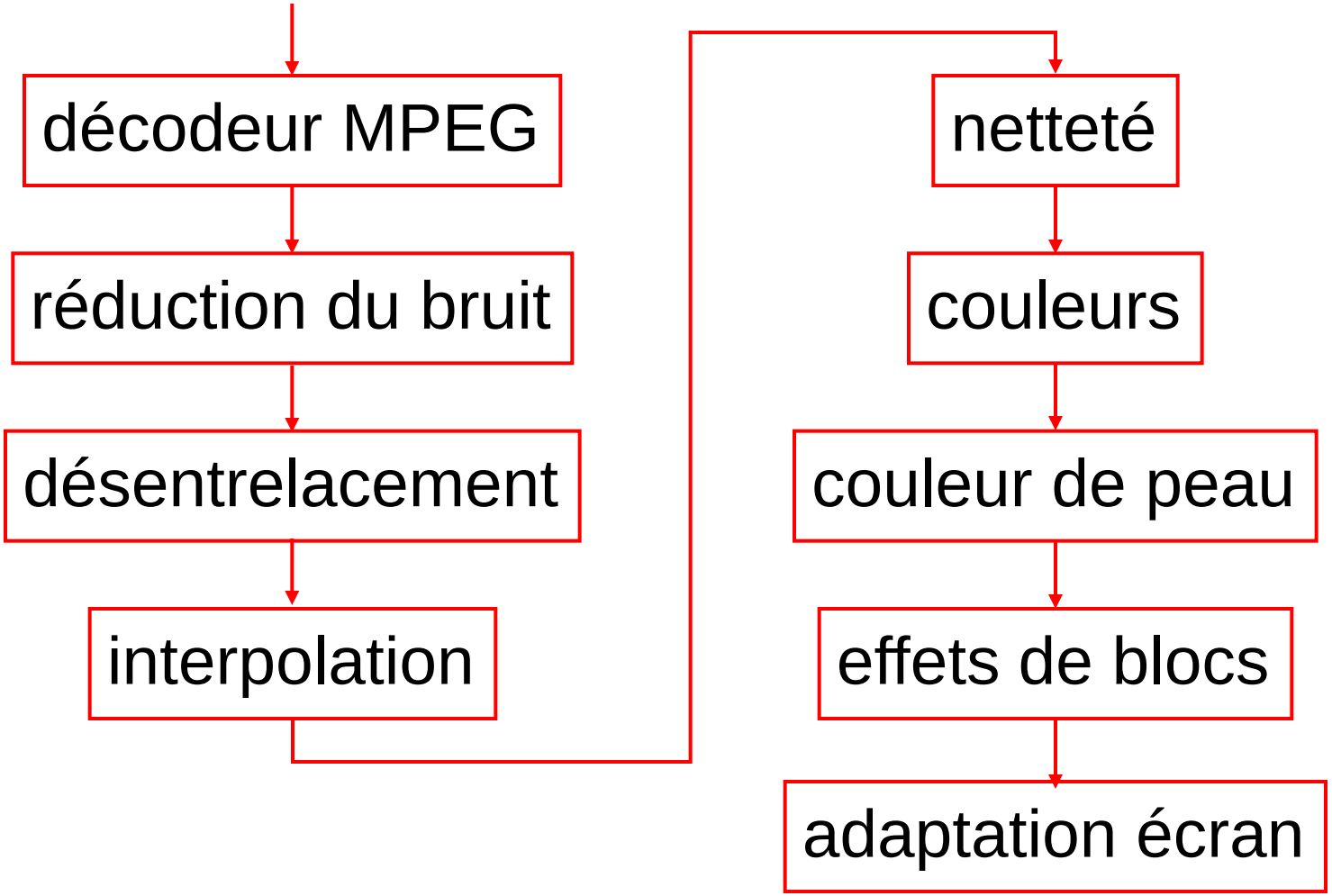
tick !

Additionneur pipeline



Echange vitesse contre latence

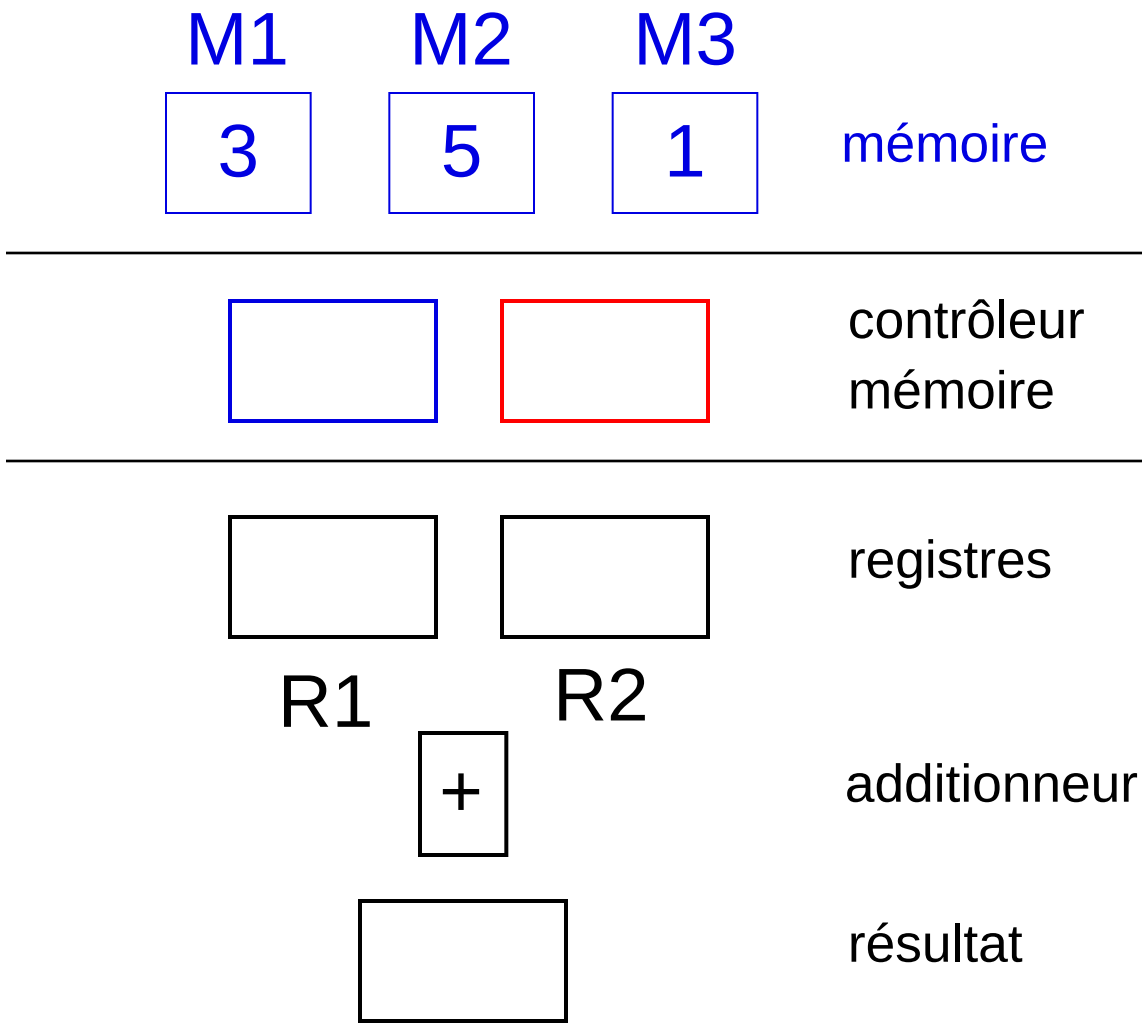
TVHD : pour faire une belle image...



Plusieurs téra-opérations par seconde!

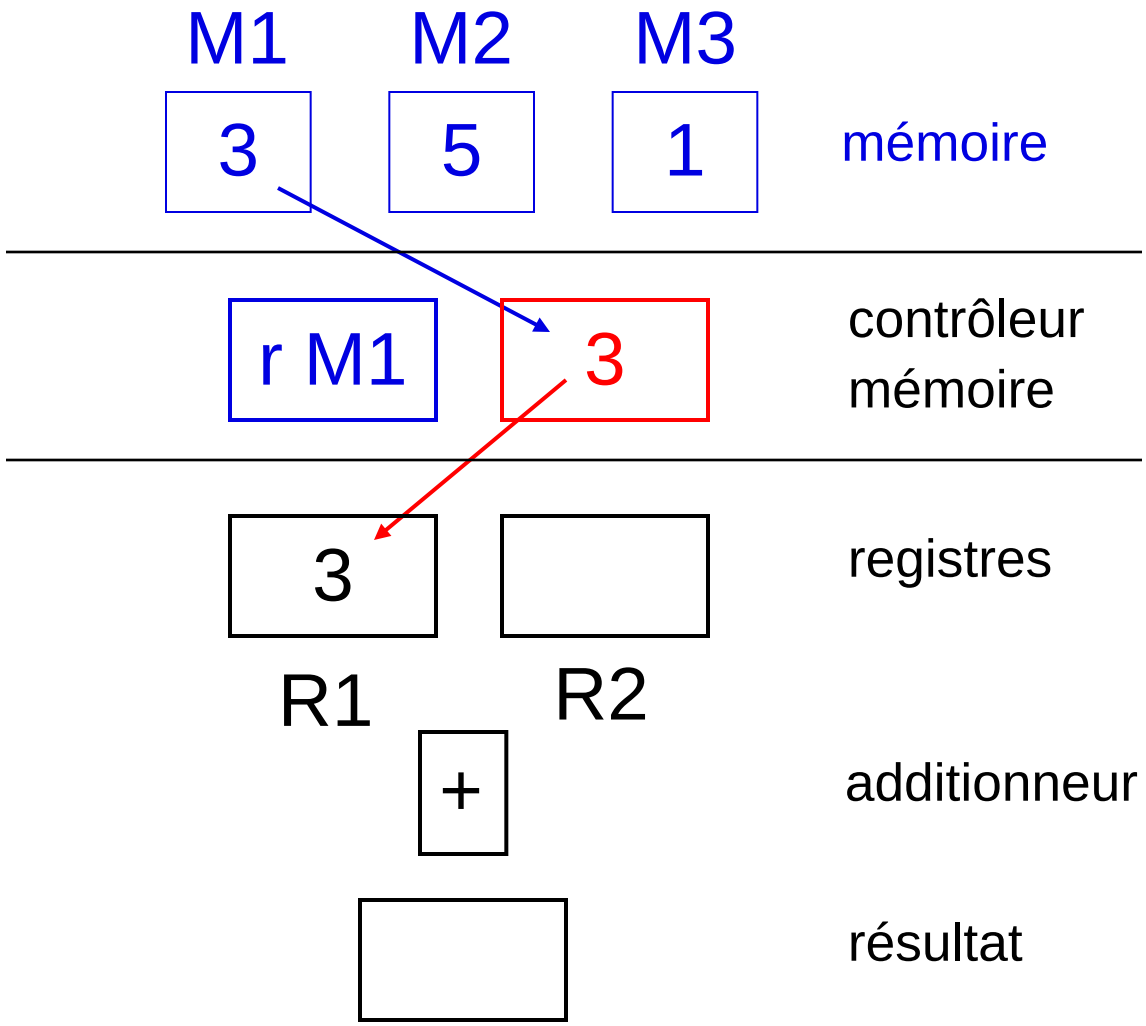
Microprocesseur : exécution simple

LD R1, M1
LD R2, M2
ADD R1, R2
ST R1, M3



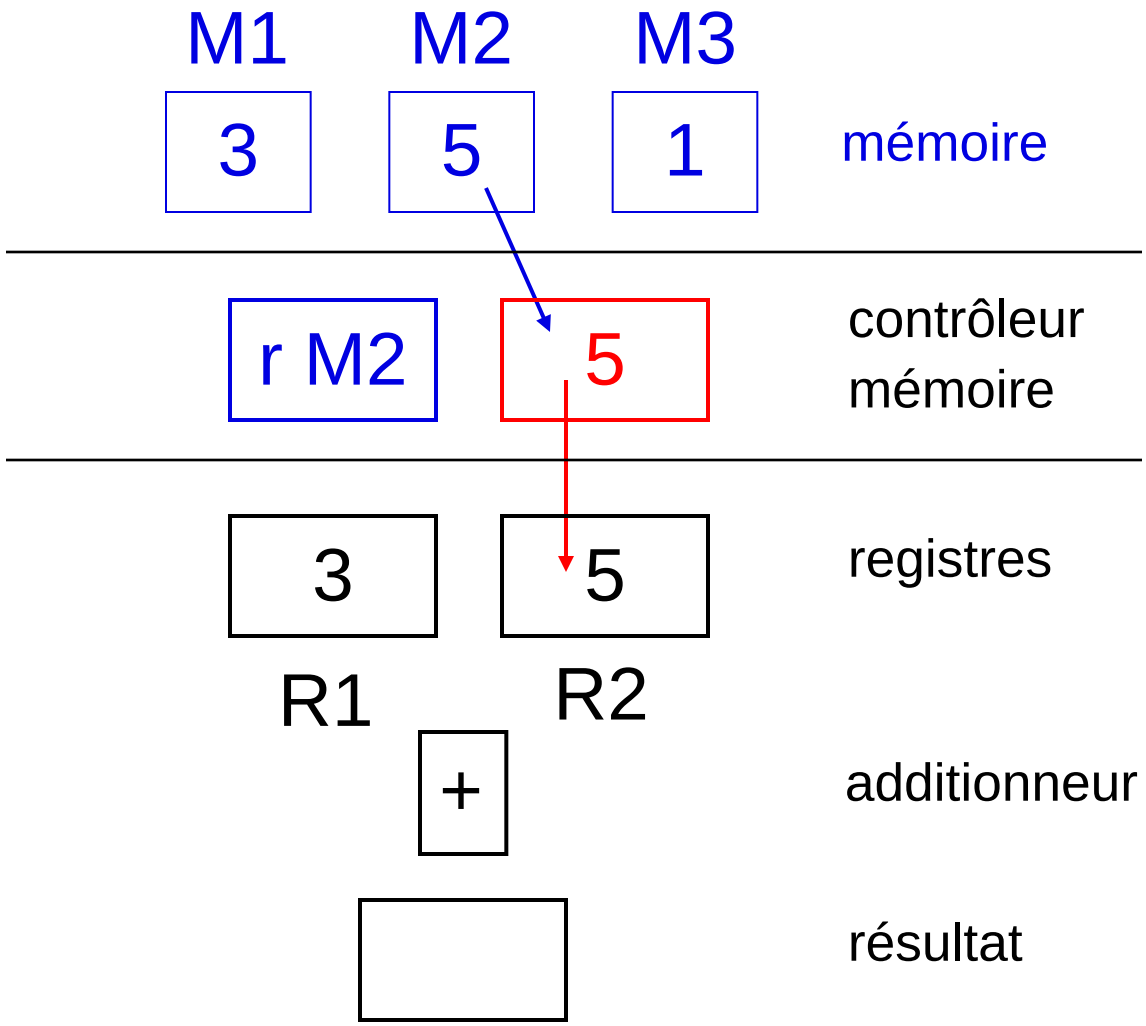
Exécution simple

LD R1, M1
LD R2, M2
ADD R1, R2
ST R1, M3



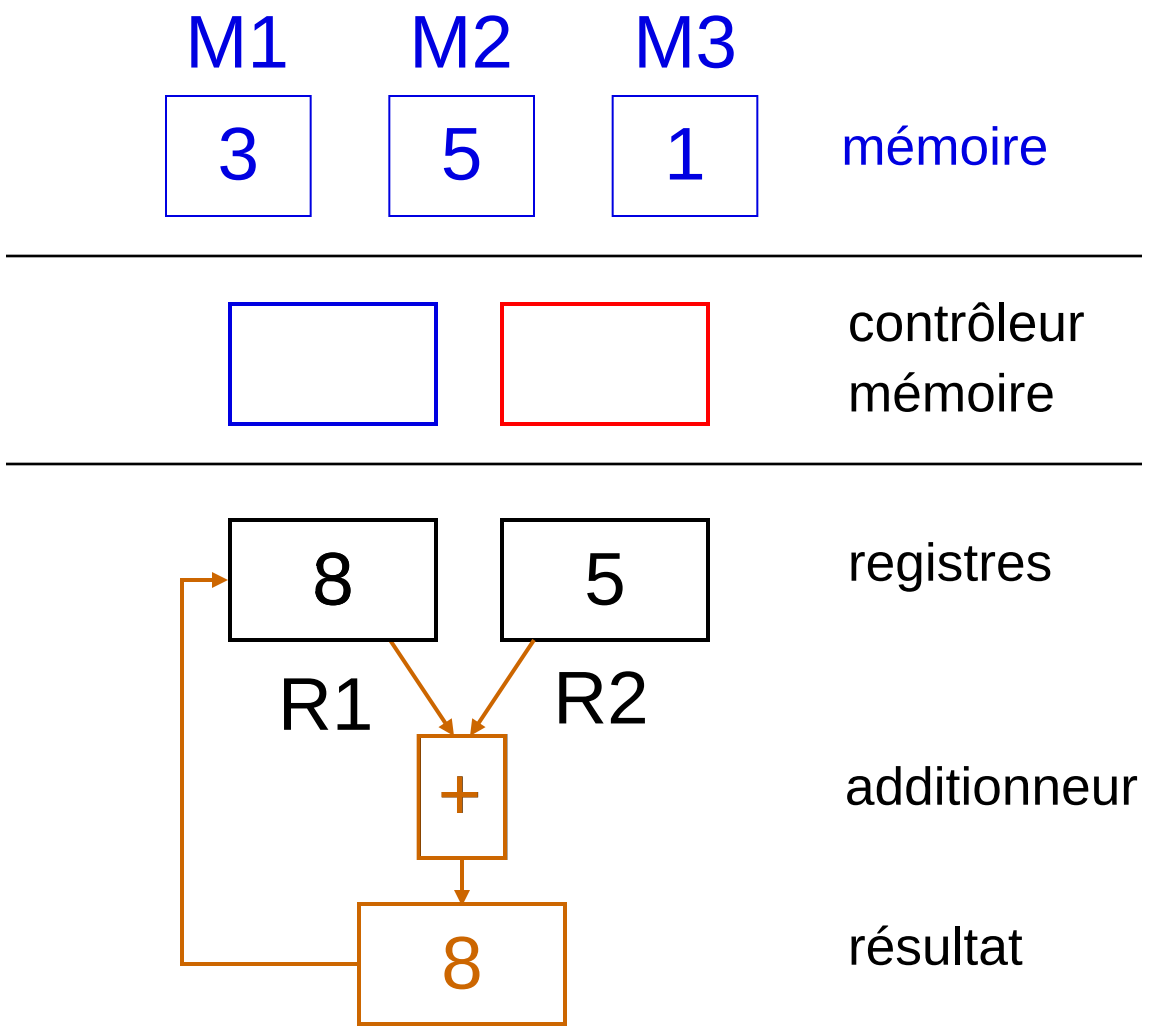
Exécution simple

LD R1, M1
LD R2, M2
ADD R1, R2
ST R1, M3



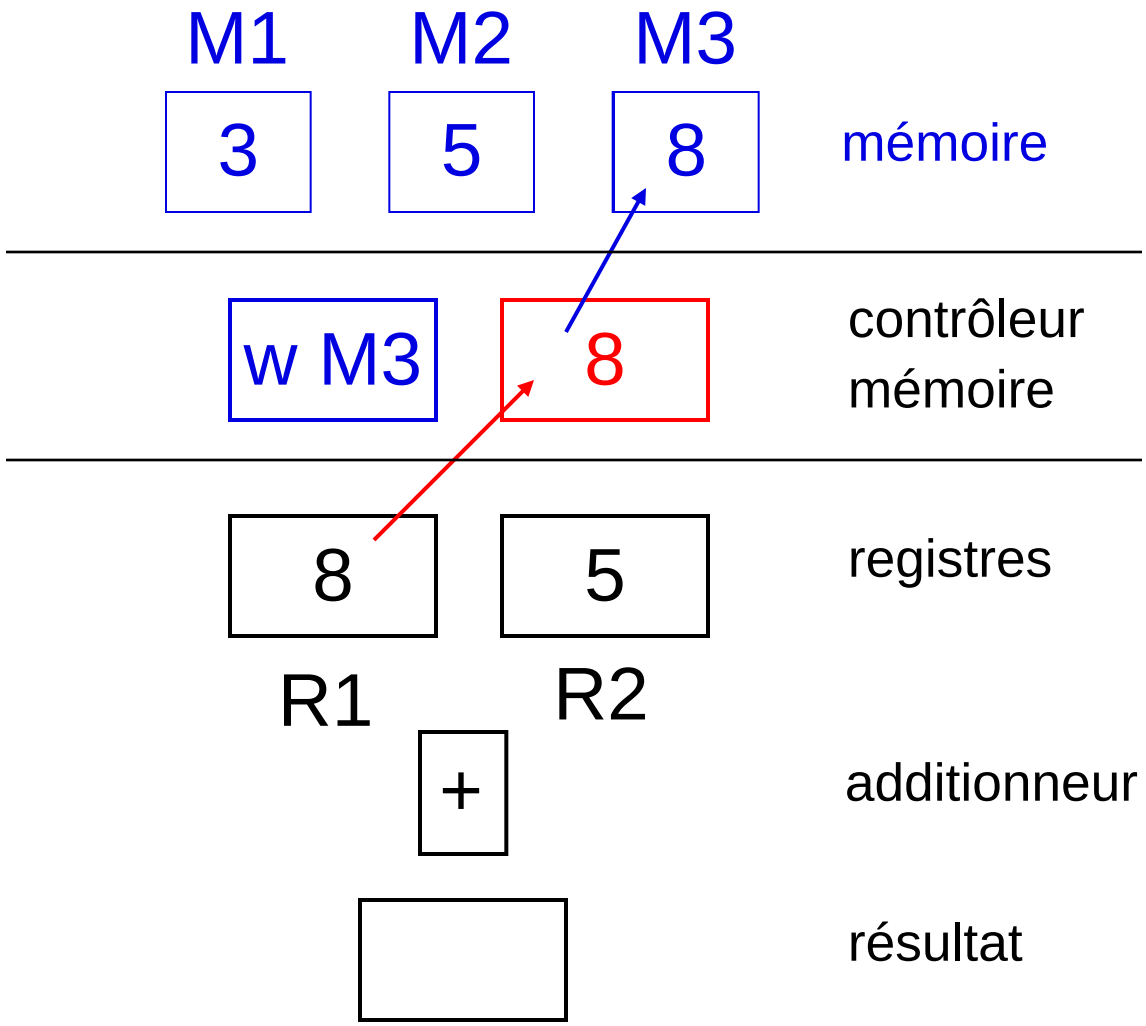
Exécution simple

LD R1, M1
LD R2, M2
ADD R1, R2
ST R1, M3



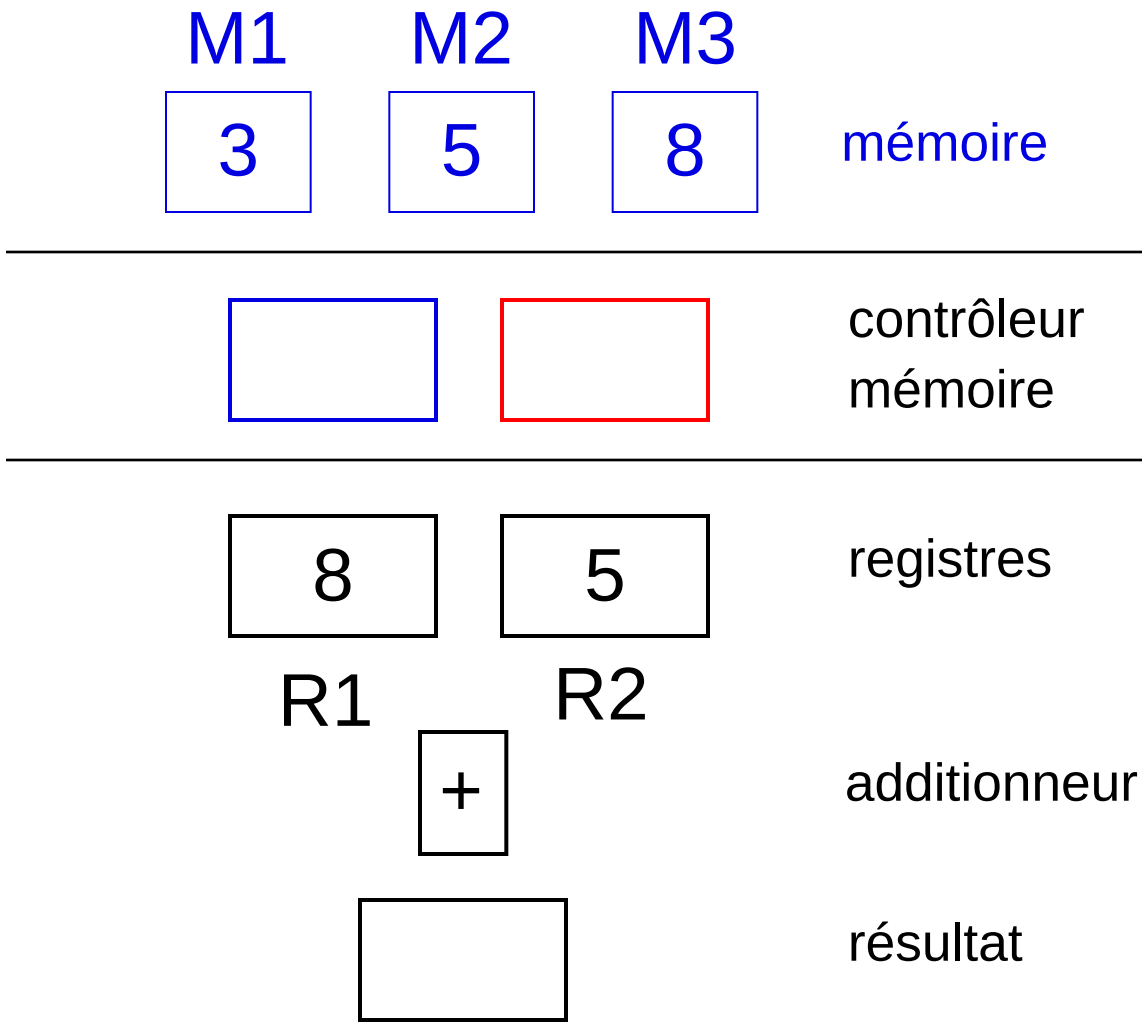
Exécution simple

LD R1, M1
LD R2, M2
ADD R1, R2
ST R1, M3



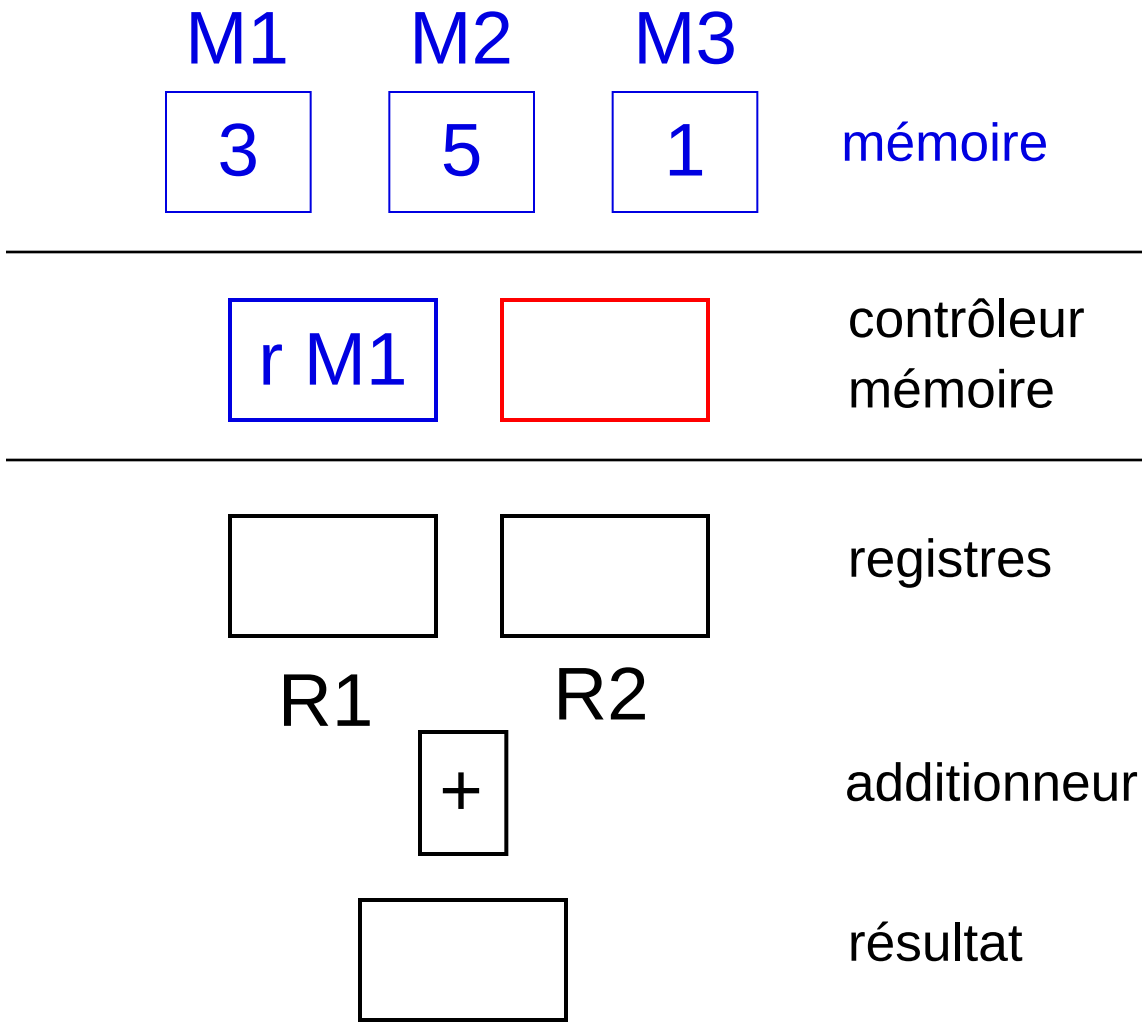
C'est fini!

LD R1, M1
LD R2, M2
ADD R1, R2
ST R1, M3



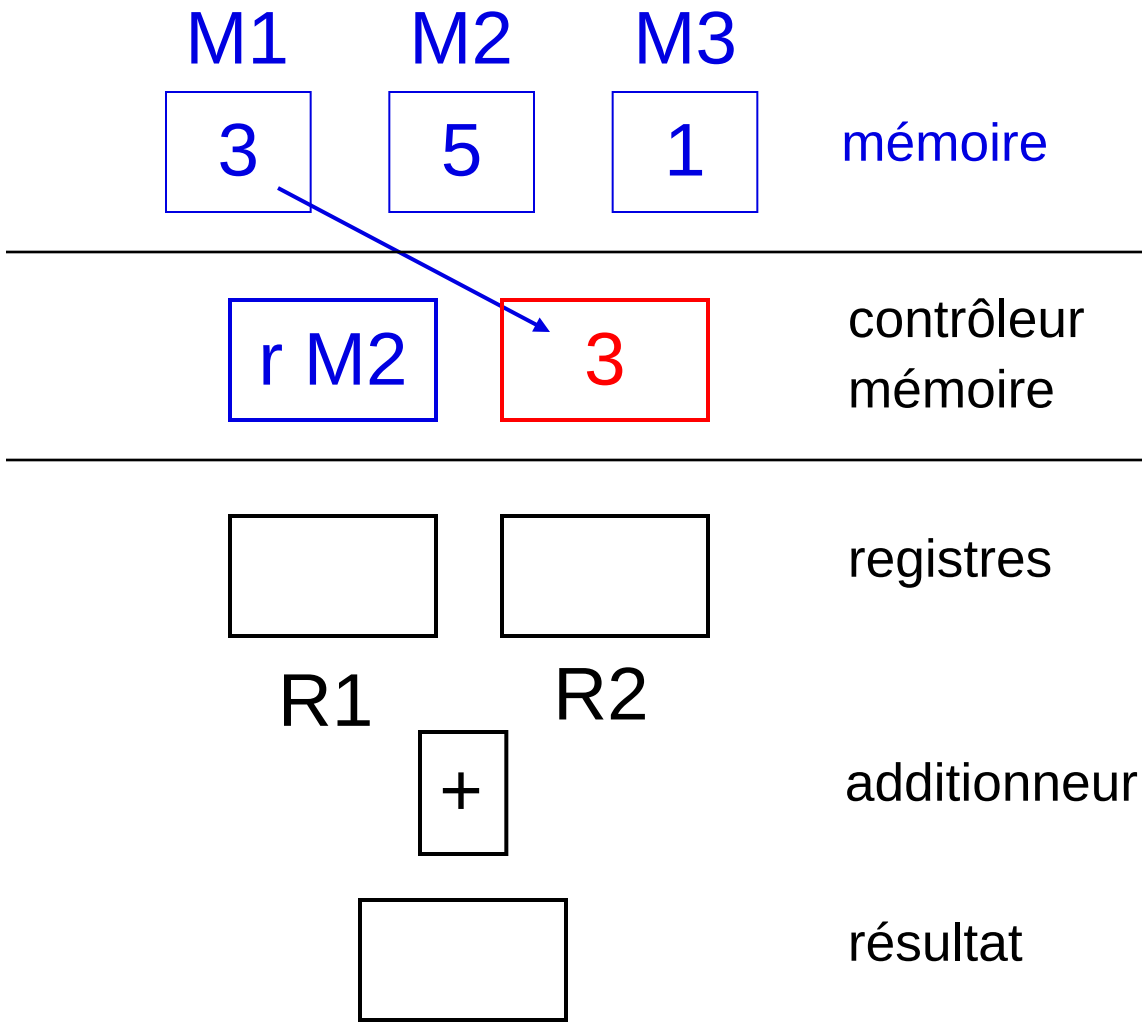
Exécution en pipeline

LD R1, M1
LD R2, M2
ADD R1, R2
ST R1, M3



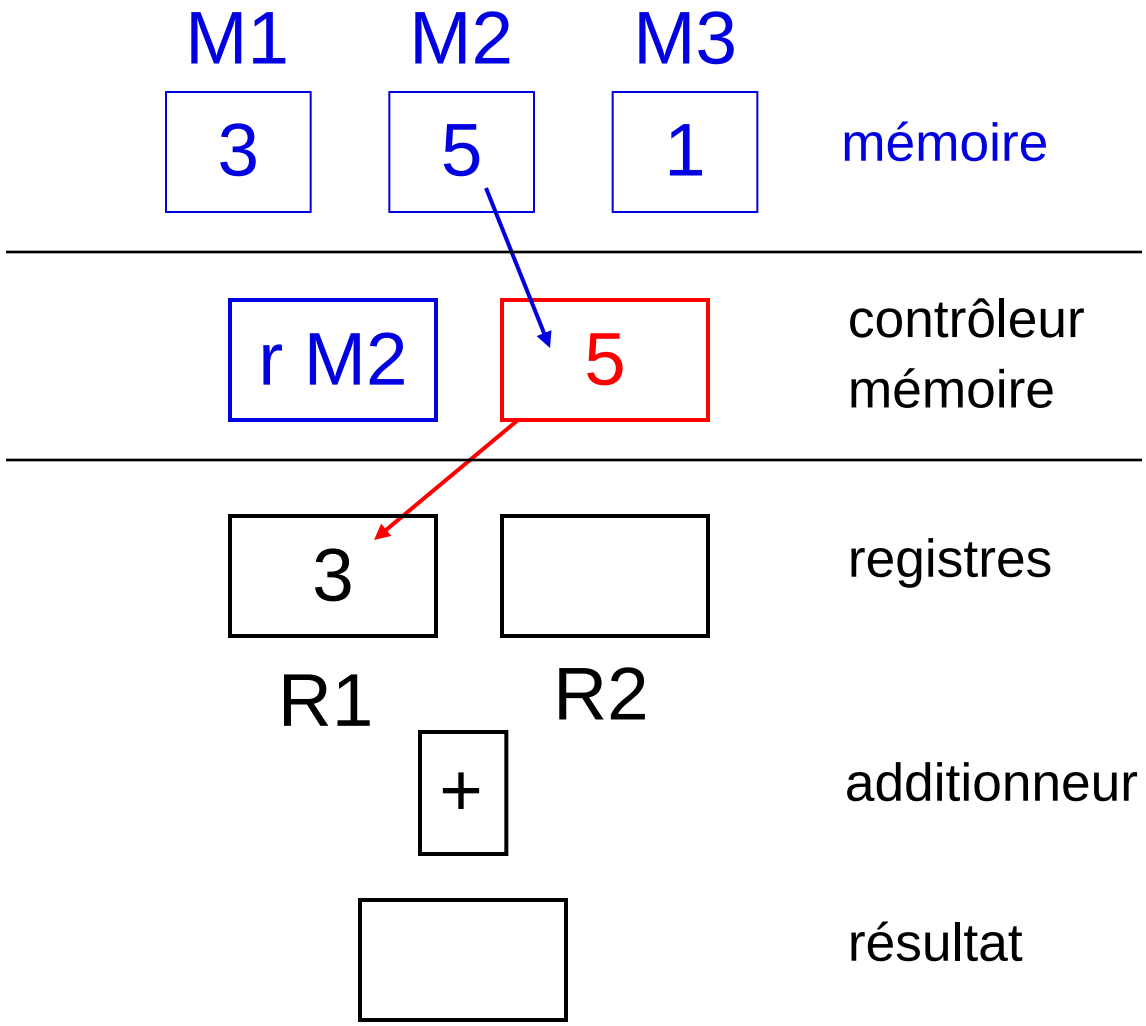
Exécution en pipeline

LD R1, M1
LD R2, M2
ADD R1, R2
ST R1, M3



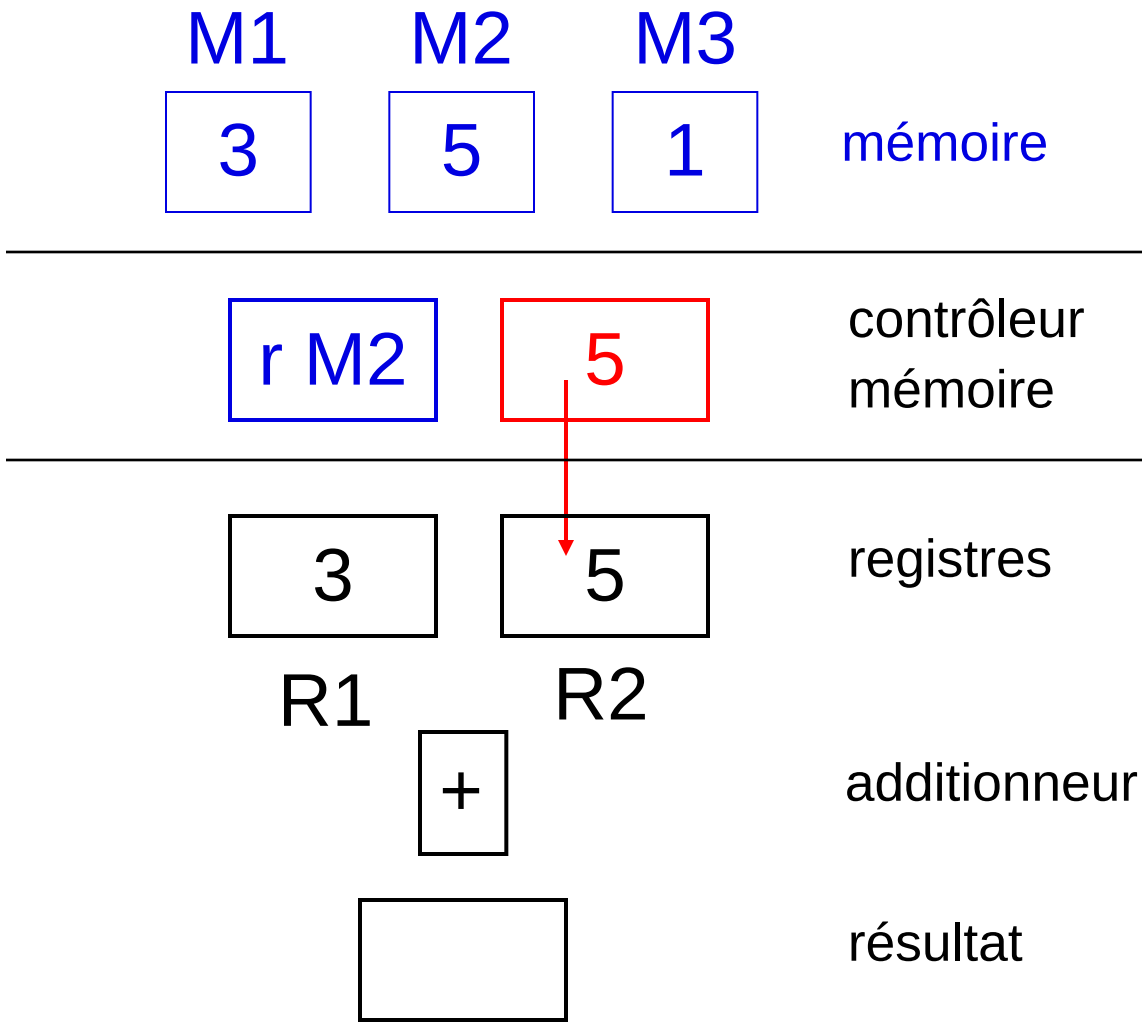
Exécution en pipeline

LD R1, M1
LD R2, M2
ADD R1, R2
ST R1, M3



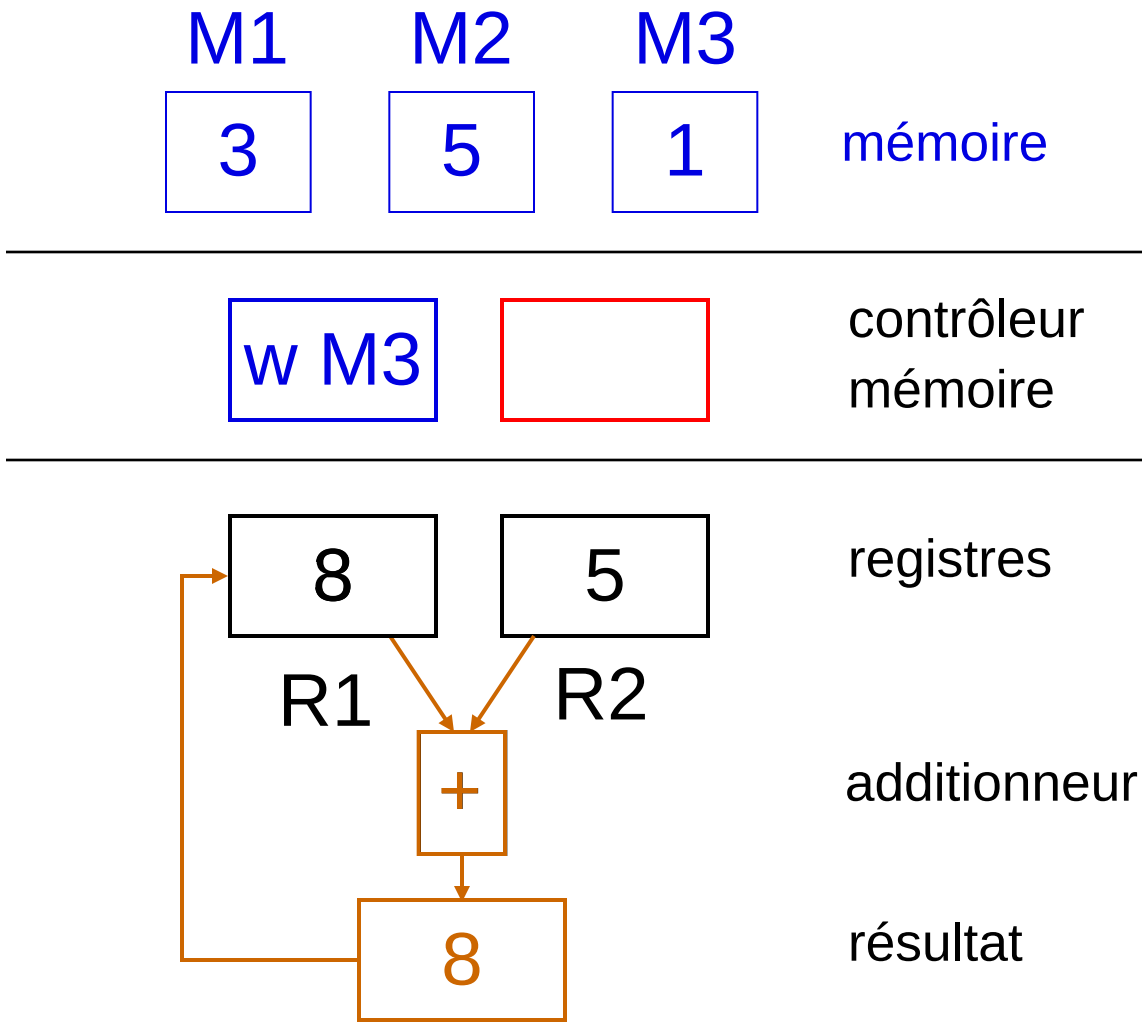
Exécution en pipeline

LD R1, M1
LD R2, M2
ADD R1, R2
ST R1, M3



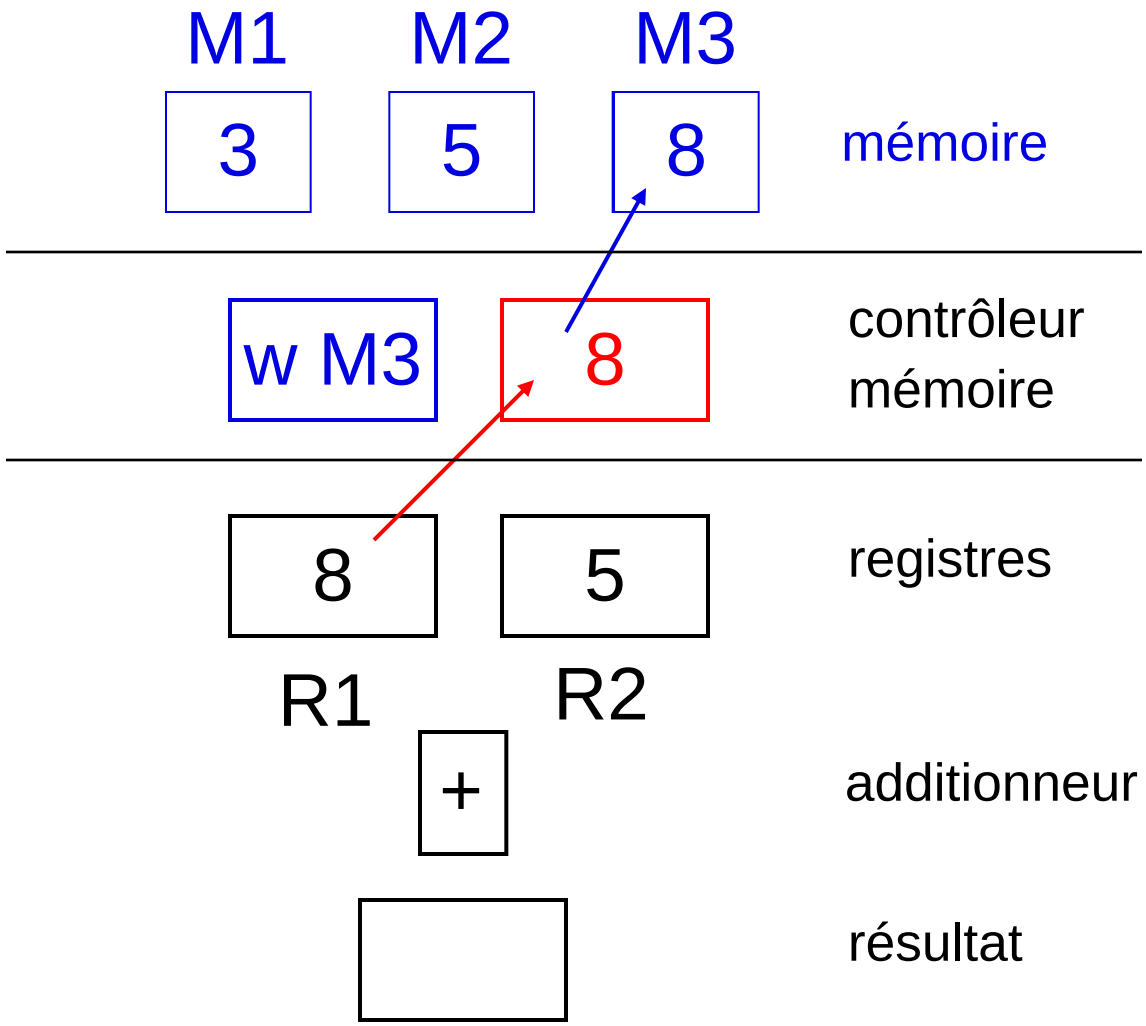
Exécution en pipeline

LD R1, M1
LD R2, M2
ADD R1, R2
ST R1, M3



Exécution en pipeline

LD R1, M1
LD R2, M2
ADD R1, R2
ST R1, M3



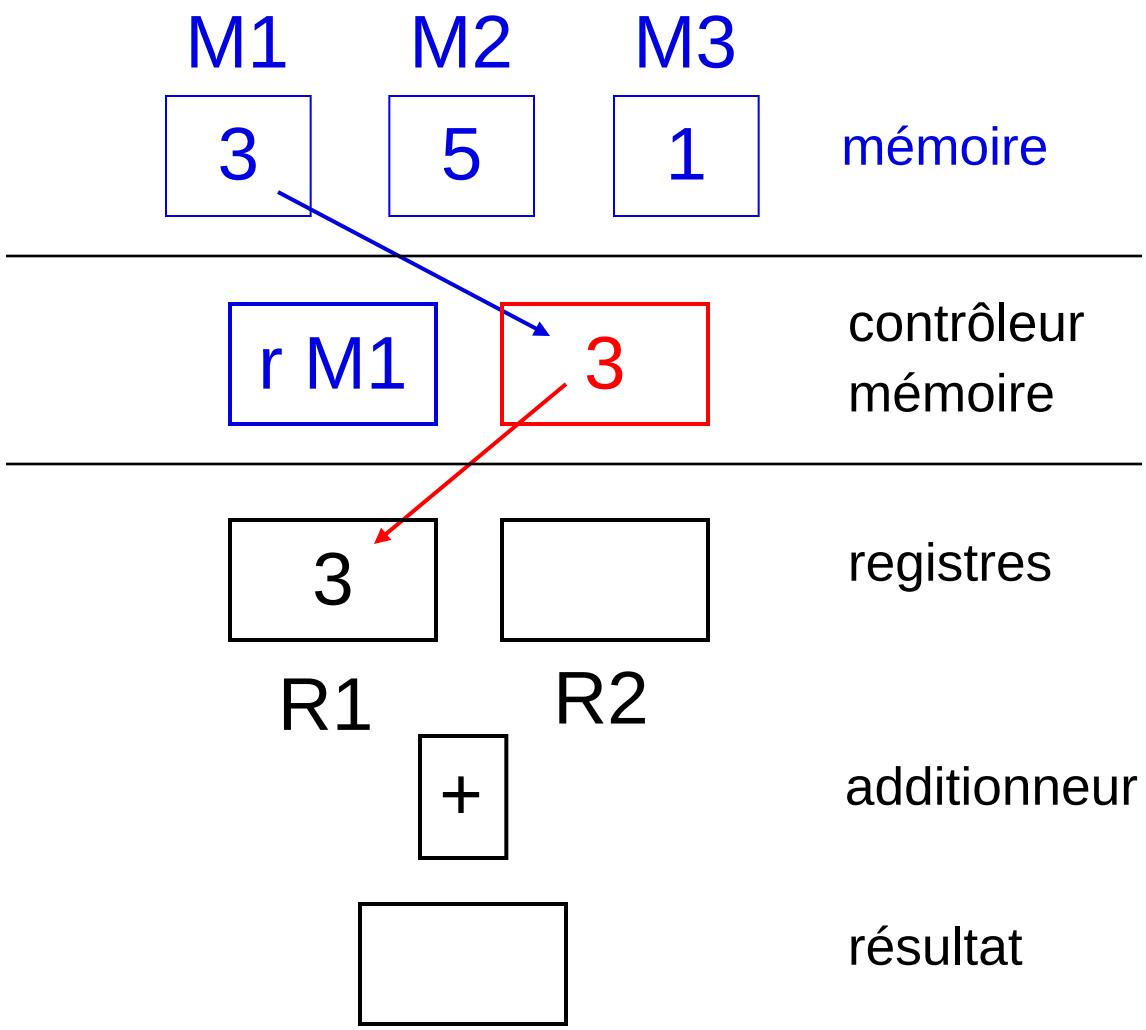
Attente mémoire....

LD R1, M1

LD R2, M1

ADD R1, R2

ST R1, M3



http://www.clipart.co.uk

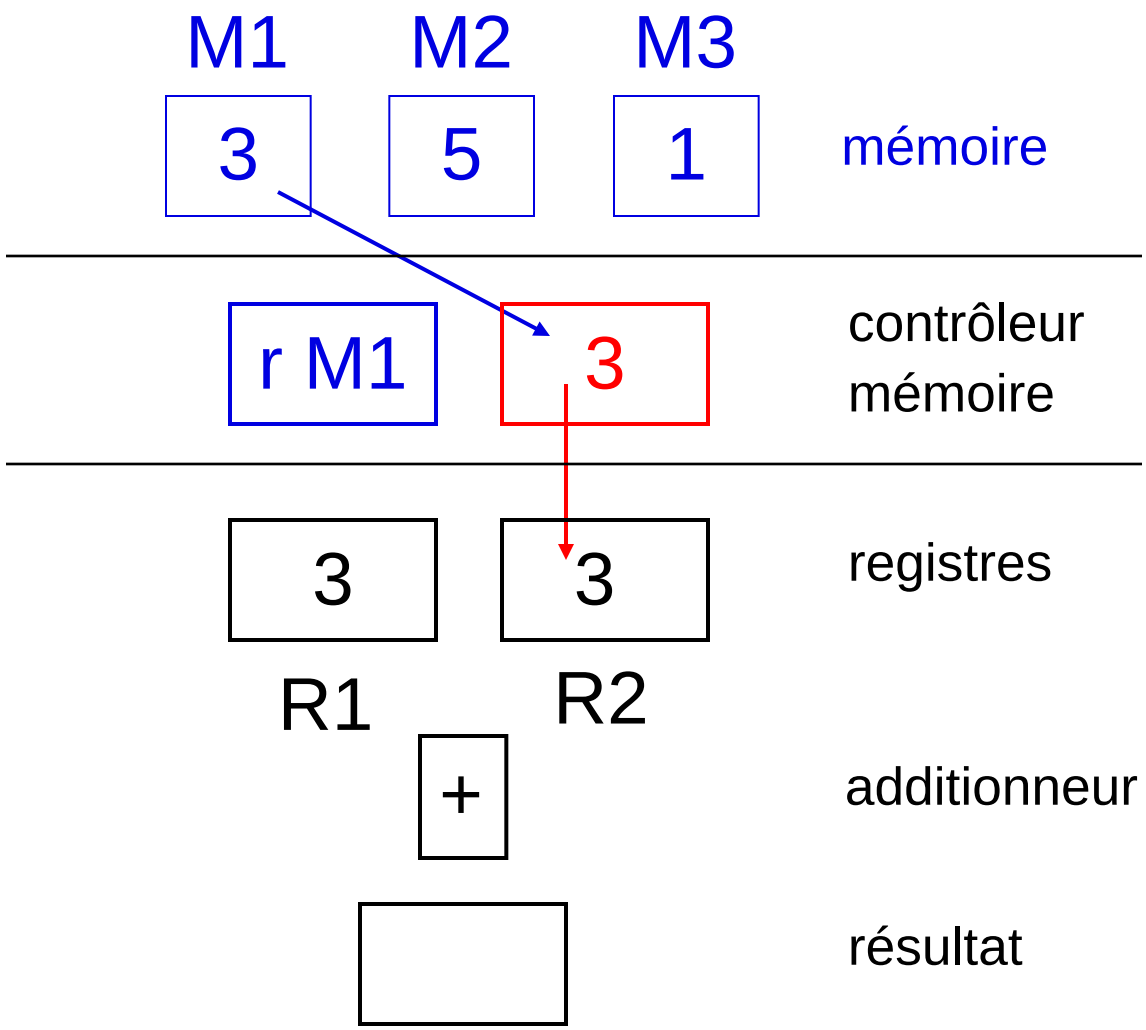
Attente mémoire....

LD R1, M1

LD R2, M1

ADD R1, R2

ST R1, M3



http://www.clipart.co.uk

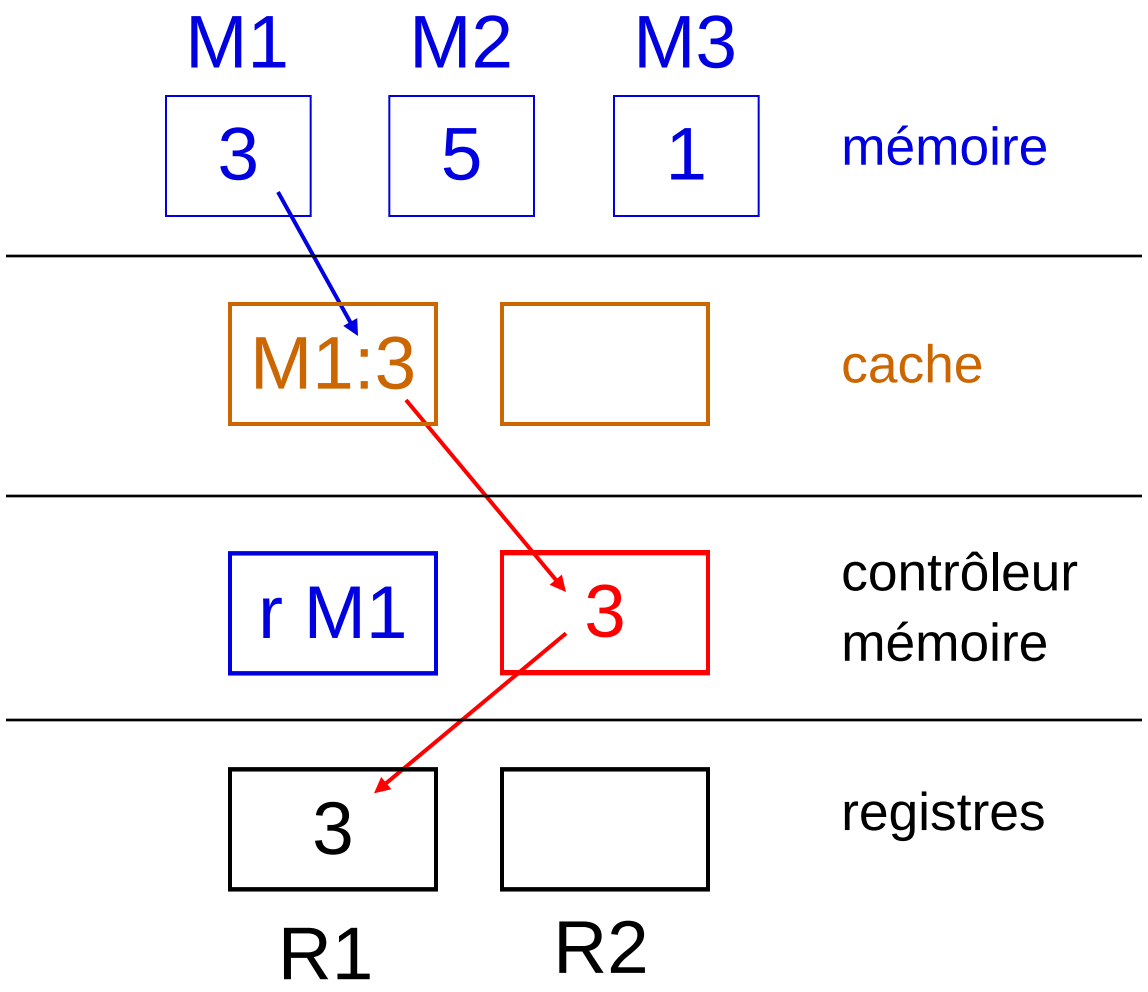
Mémoire cache

LD R1, M1

LD R2, M1

ADD R1, R2

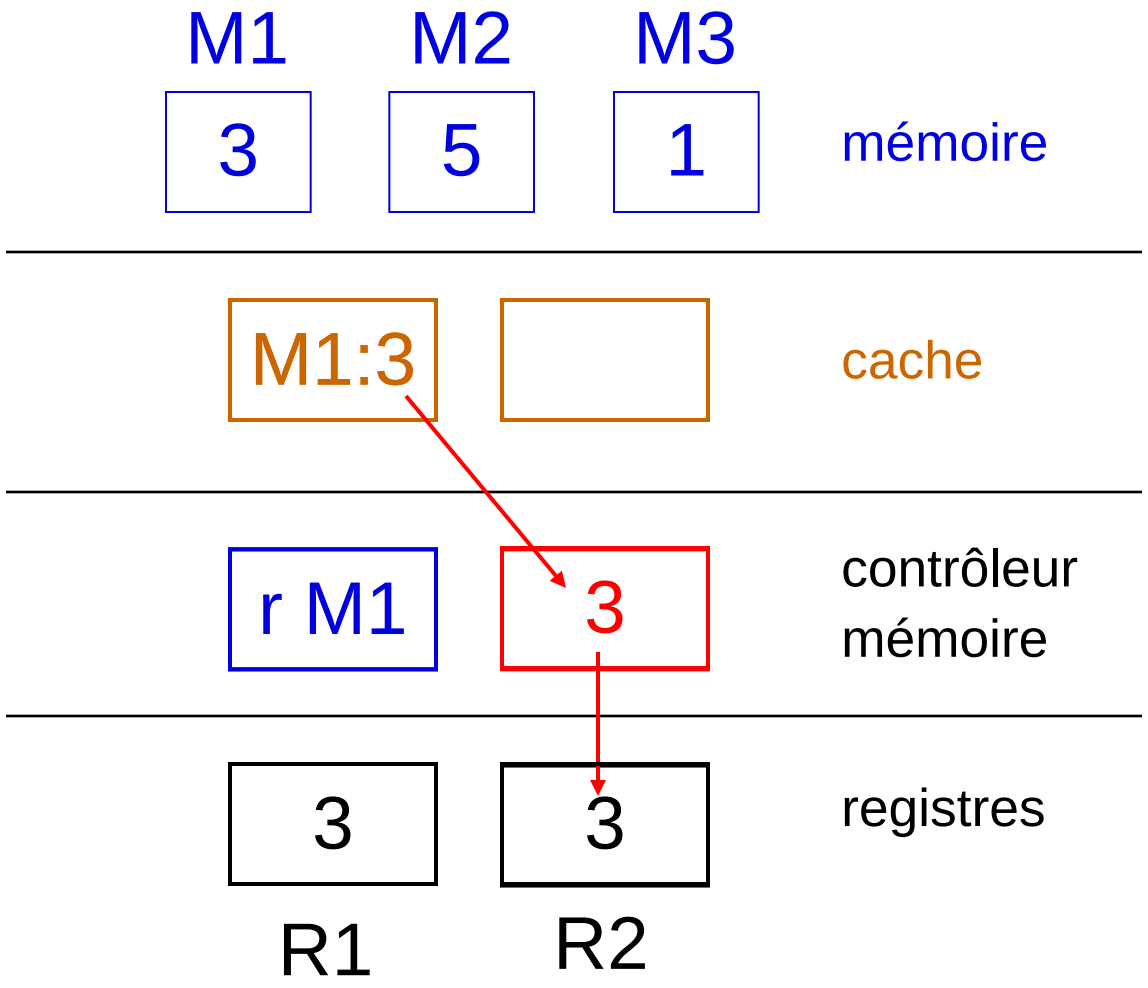
ST R1, M3



http://www.clipart.co.uk

Mémoire cache

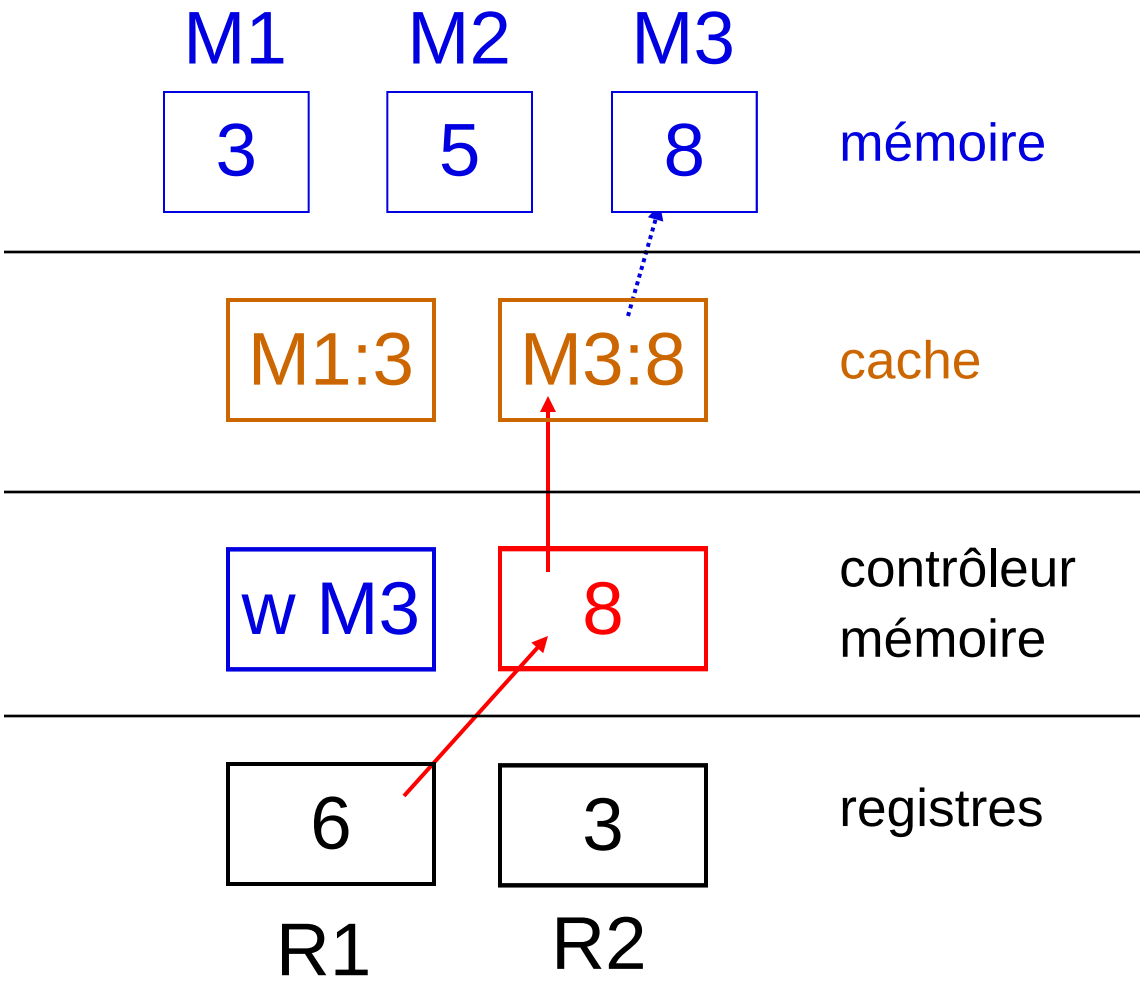
LD R1, M1
LD R2, M1
ADD R1, R2
ST R1, M3



http://www.clipart.co.uk

Mémoire cache

LD R1, M1
LD R2, M1
ADD R1, R2
ST R1, M3



http://www.clipart.co.uk

Cache = difficile !

- Cher, donc petit !
- Comment anticiper ?
- Qui garder : plus récent ? lu souvent ?
- Mémoire et cache toujours cohérents ?
- Cohérence garantie si multiprocesseur ?

Belle source de bugs
Particulièrement difficile à prouver correct

Anticipation / spéculation

si M1 > 0 alors R3 ← R1 * R2 sinon R3 ← R1 / R2

lire et tester M1 : 10 cycles
puis calculer R1*R2 : 5 cycles
ou calculer R1 / R2 : 7 cycles
stocker dans R3 : 1 cycle

pire cas : 18 cycles

lire et tester M1 : 10 cycles
calculer R1*R2 : 5 cycles
calculer R1 / R2 : 7 cycles
stocker le bon dans R3 : 1 c.
(et jeter l'autre)

pire cas : 11 cycles

gain de temps, perte de surface et d'énergie

Les compromis

- Profondeur de pipeline ?

  grande = profondeur logique faible => fréquence haute

  mais plein d'ennuis..

- Type et taille de cache ?
- Type et profondeur de spéculation ?

  grande = efficace en moyenne

  mais devient horriblement complexe et imprévisible

- Horloges multiples et gestion de puissance ?

Avec tout ça (et le reste), les microprocesseurs deviennent infernaux à mettre au point

Où vont les méga-microprocesseurs ?

- Comment profiter de la surabondance de transistors ?
  on ne sait plus augmenter la fréquence
  => cœurs multiples et grands caches!
- Mais comment programmer les multicœurs ?
  2-4, facile. 4-100 ?????
- Et comment évacuer / diminuer la chaleur ?
- Ou bien, comment changer de technologie ?
  le silicium marche si bien que les autrent peinent !

Un défi technologique majeur

Architecture

fonctionnalité OK?
performance OK?
marketing OK?

Expérience
Revues
prototypes Excel

Micro-Architecture

découpage OK?
performance OK?

Modélisation logicielle

Design logique

fonctionnalité OK?
vitesse / surface OK?
consommation OK?

test aléatoire dirigé
vérification formelle

circuits

équivalent
au source?

vérification formelle

DFT (test)

couverture de test
~100% ?

ATPG
SAT

Place&Route

connexions?
électrité OK?
timing OK?

Design Rules
Checking (DRC)

$ 1,000,000 Masks

Puces

marche vraiment?

Packaging
Test physique

Beaucoup d'algorithmes difficiles !

- Compilation des spécifications en portes logiques
- Optimisation logique
- Portes physiques optimisées
- Placement / routage des fils
- Construction de jeux de tests
- Constructions des masques (téra-octets)
- ...

Beaucoup sont NP-Complets => heuristiques !
Une grande industrie: EDA*, ~5 milliards de $

* EDA = Electronic Design Automation

L' avenir des circuits

- La réduction de taille peut continuer, en théorie
  - le problème actuel est la puissance dissipée
- Mais c'est l'économie qui pourrait flancher !
  - progression concomitante du coût des usines (> 2Md$)
  - et du coût de conception des puces (X*1000 hommes-an)
  - conçues à partir de composants hétérogènes

    mais faites en une fois et pas réparables
  - et rentables seulement en très grand volume

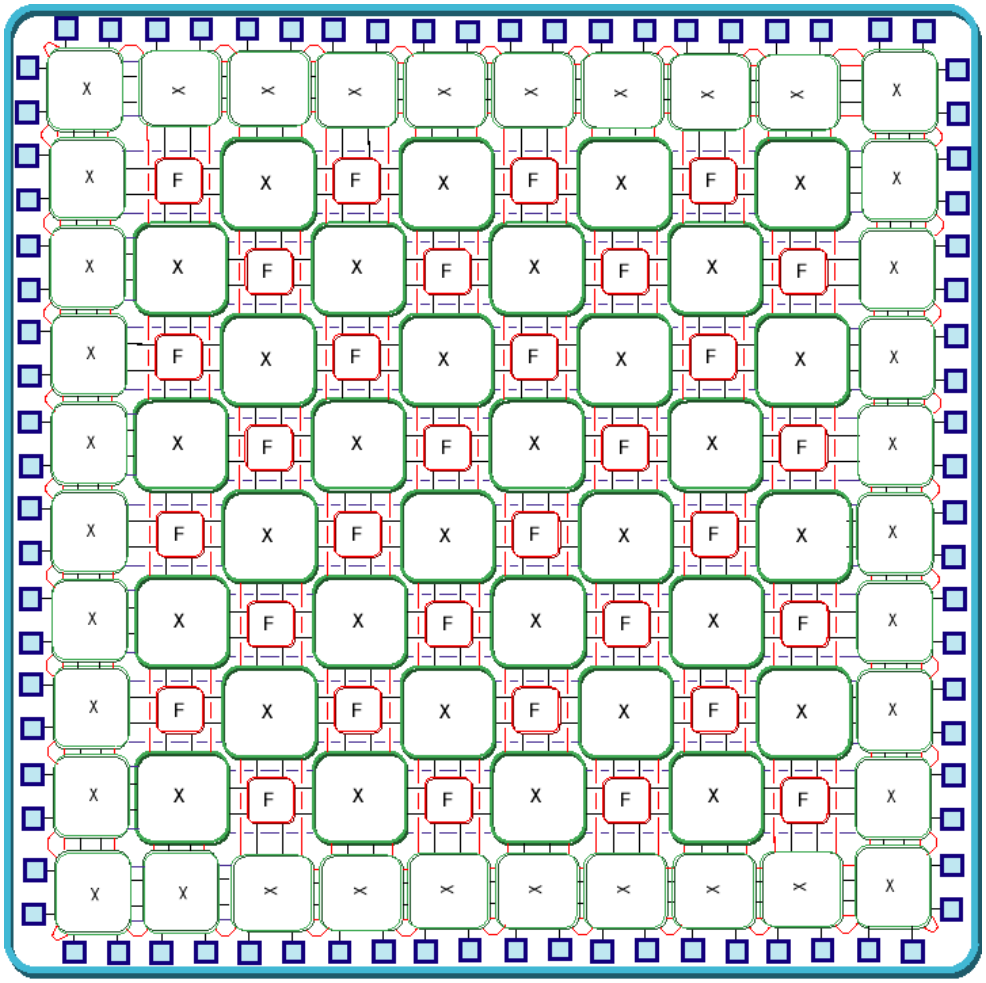
Aussi complexe qu'un avion

Evolution des clients

- Moins de nouveaux designs, mais plus complexes
- Demande de plate-formes flexibles
- Avec réutilisation maximale d'éléments
- Passage des application en logiciel

Deux évolutions: FPGA et ESL

FPGA = circuit programmable par logiciel



portes logiques configurables
routage configurable
=> téléchargement du circuit!

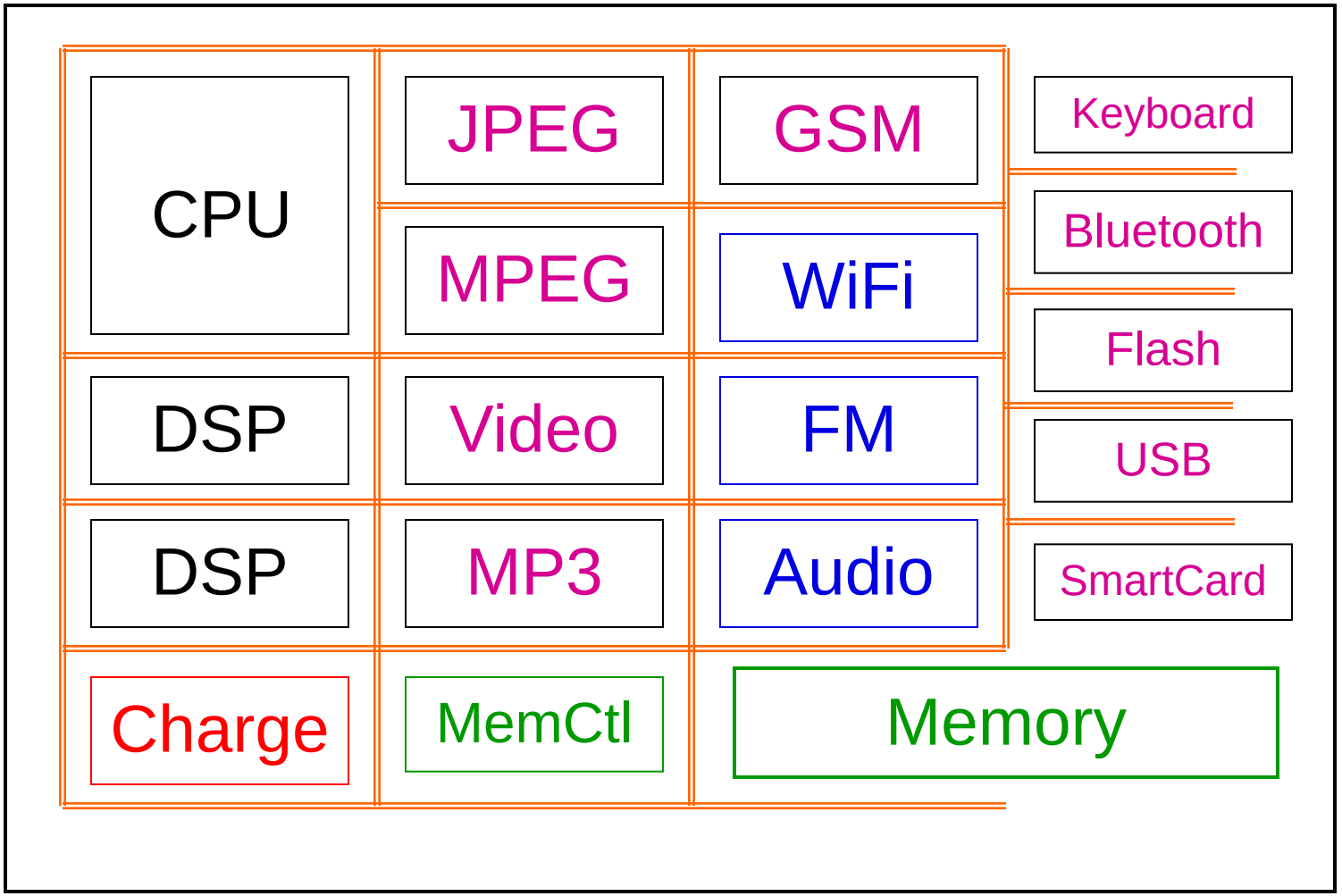
routeurs
émulation
petit volume
prototypage
recherche

ESL = Electronic System Level Design

- Plus vite et plus sûr
  - spécifications formelles de plus haut niveau
  - synthèse des niveaux inférieurs
  - vérification formelle très tôt dans la boucle
  - simulation 1000 fois plus rapide
  - prototypage rapide

- Beaucoup plus tôt !
  - mettre au point le logiciel avant le circuit
  - => modèle logiciel très rapide du circuit
  - => simulation 1 000 000 fois plus rapide !
  - mais plus ou moins conforme...

Une grande complication : deux parallélismes!



Esterel v7 et Esterel Studio