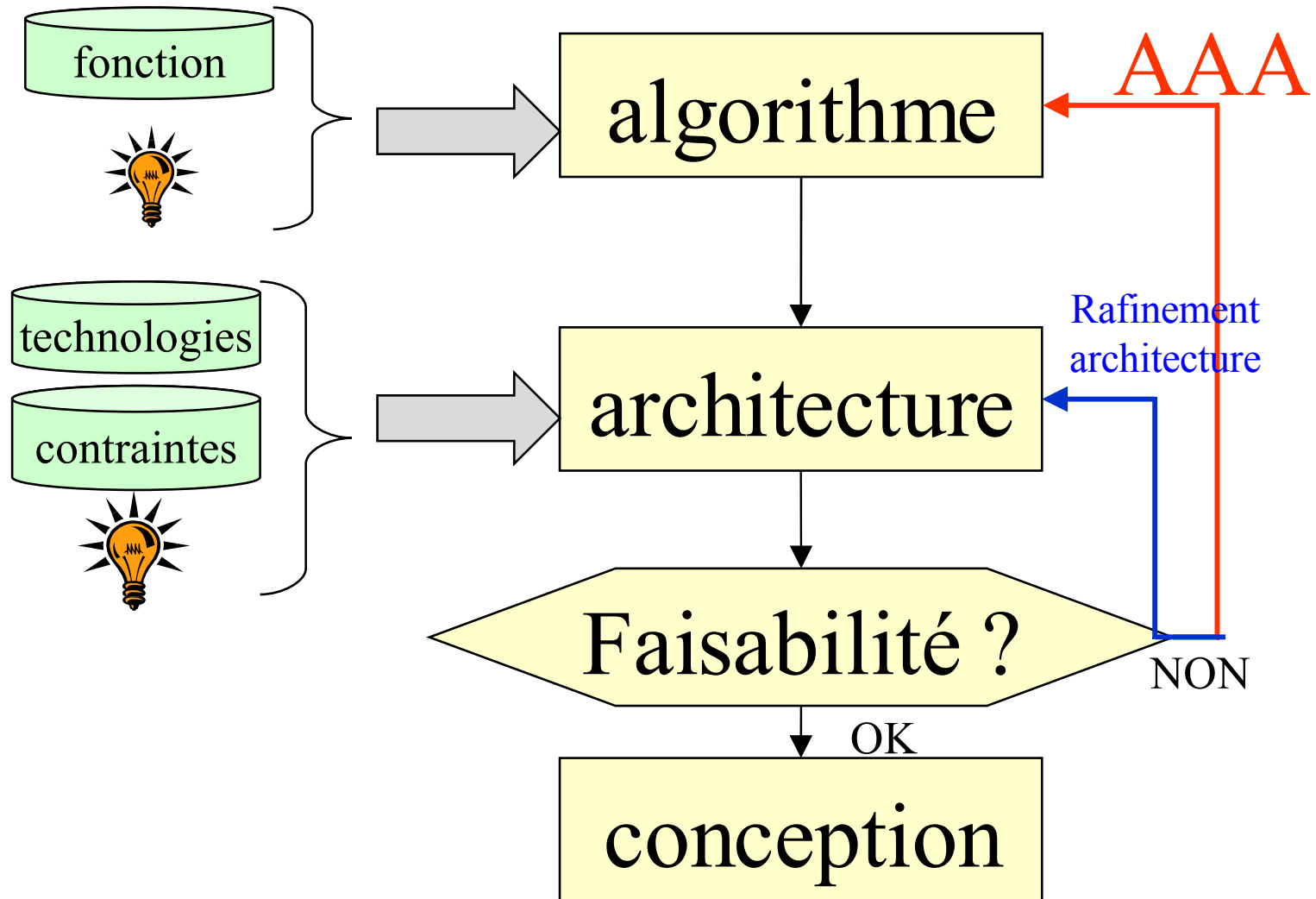


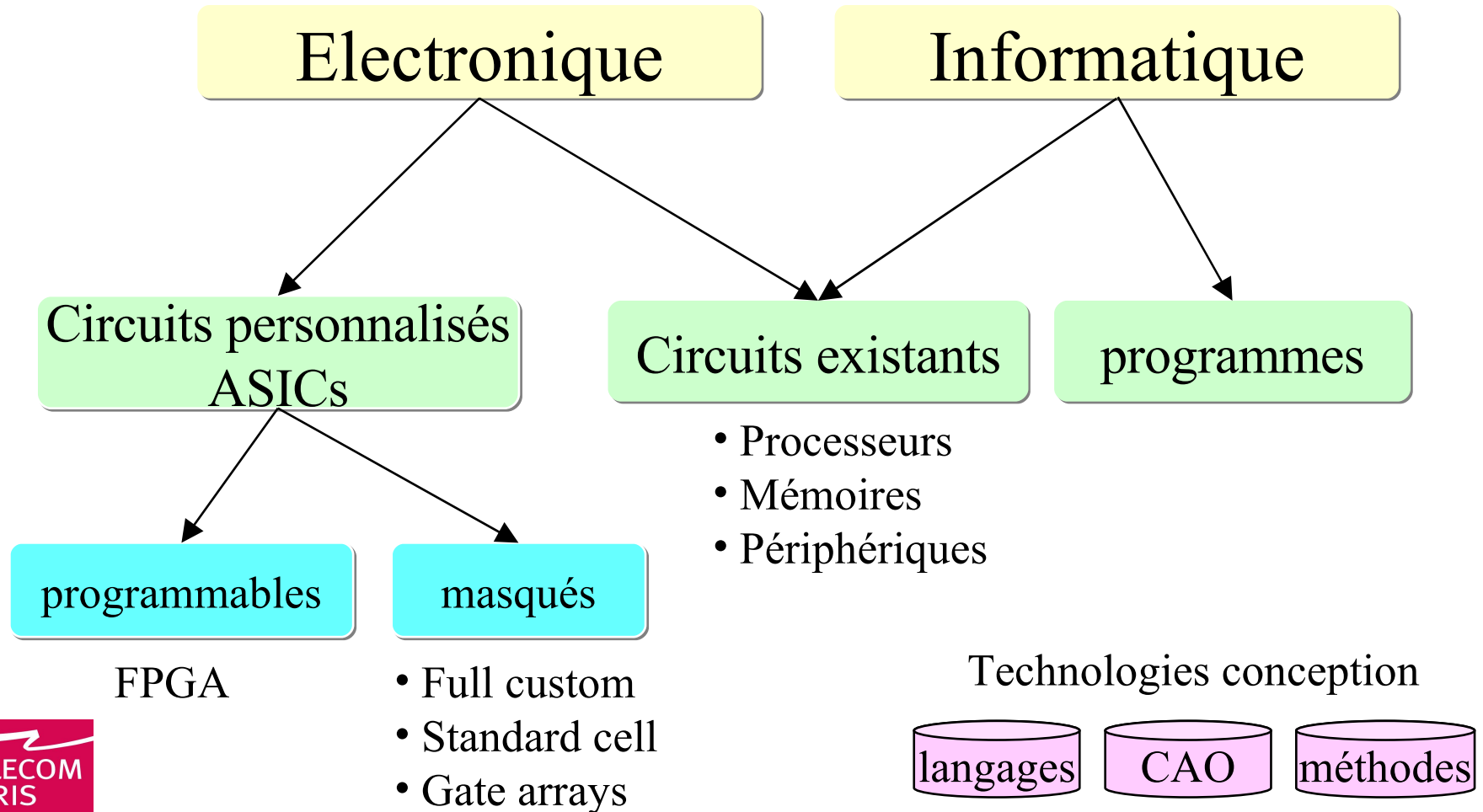
**TELECOM
PARIS**
école nationale
supérieure des
télécommunications



Etude de faisabilité d'un algorithme



Rappel : Technologies



Principales Contraintes

Contraintes physiques



Vitesse



Consommation

Contraintes de sécurité



Résistance aux piratage et intrusions

Contraintes économiques



Coût à la pièce

- surface du silicium si volume élevé

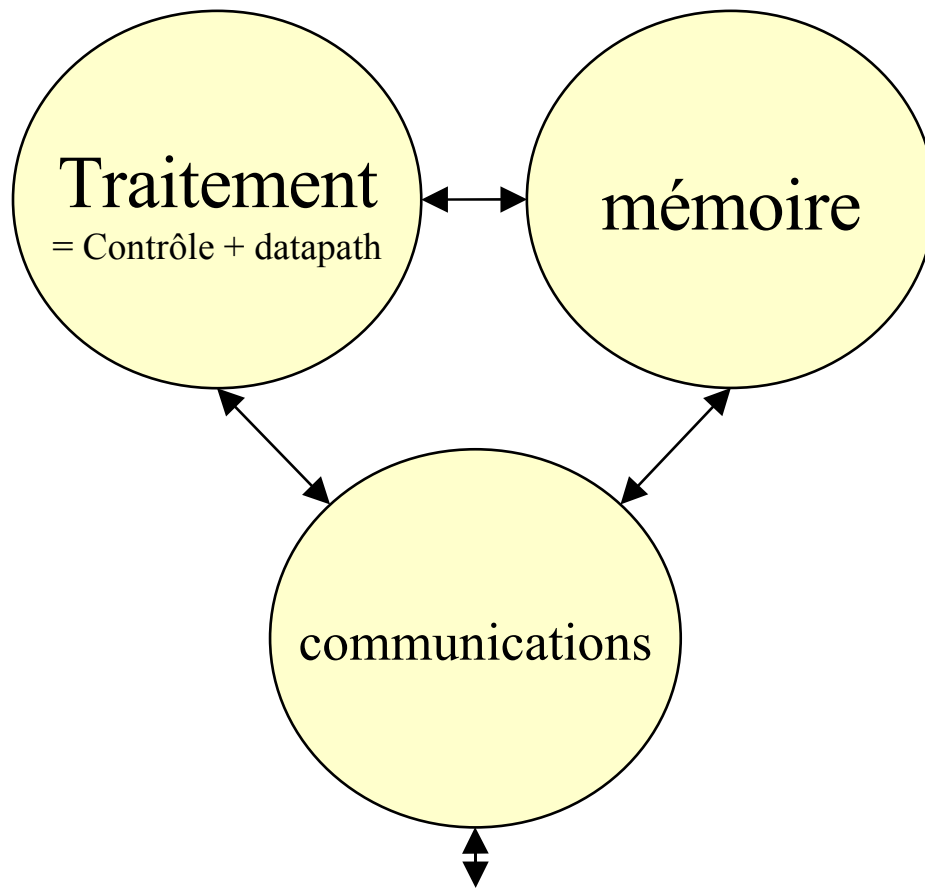
Coût et temps de développement

- importance du "time to market"

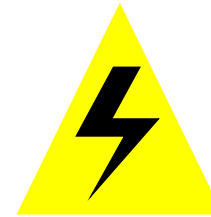
Flexibilité

- facilité d'adaptation à un marché ou standard

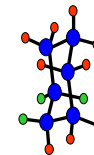
Architecture générique



test



Energie



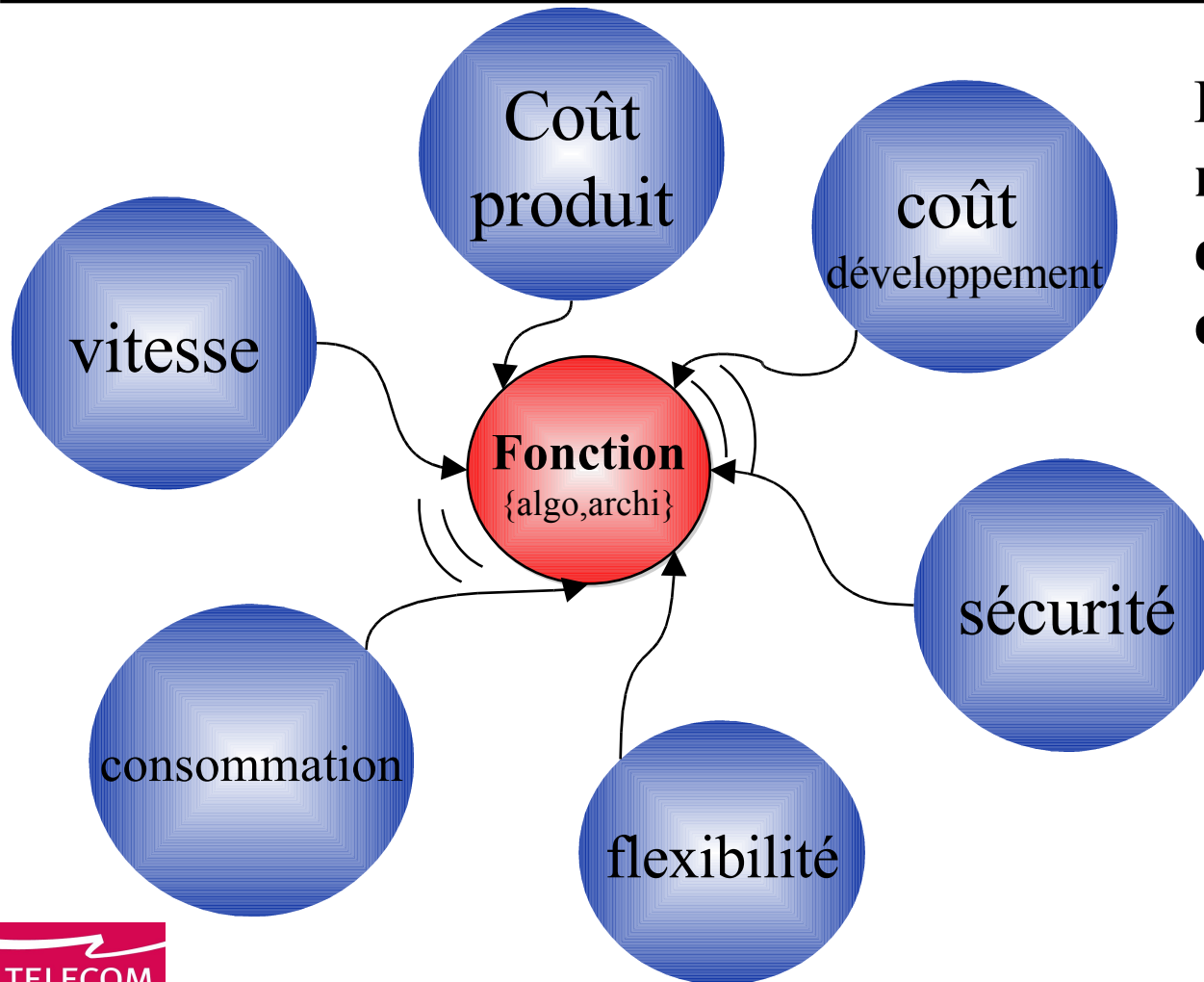
integration



securité

Hommes et Machines

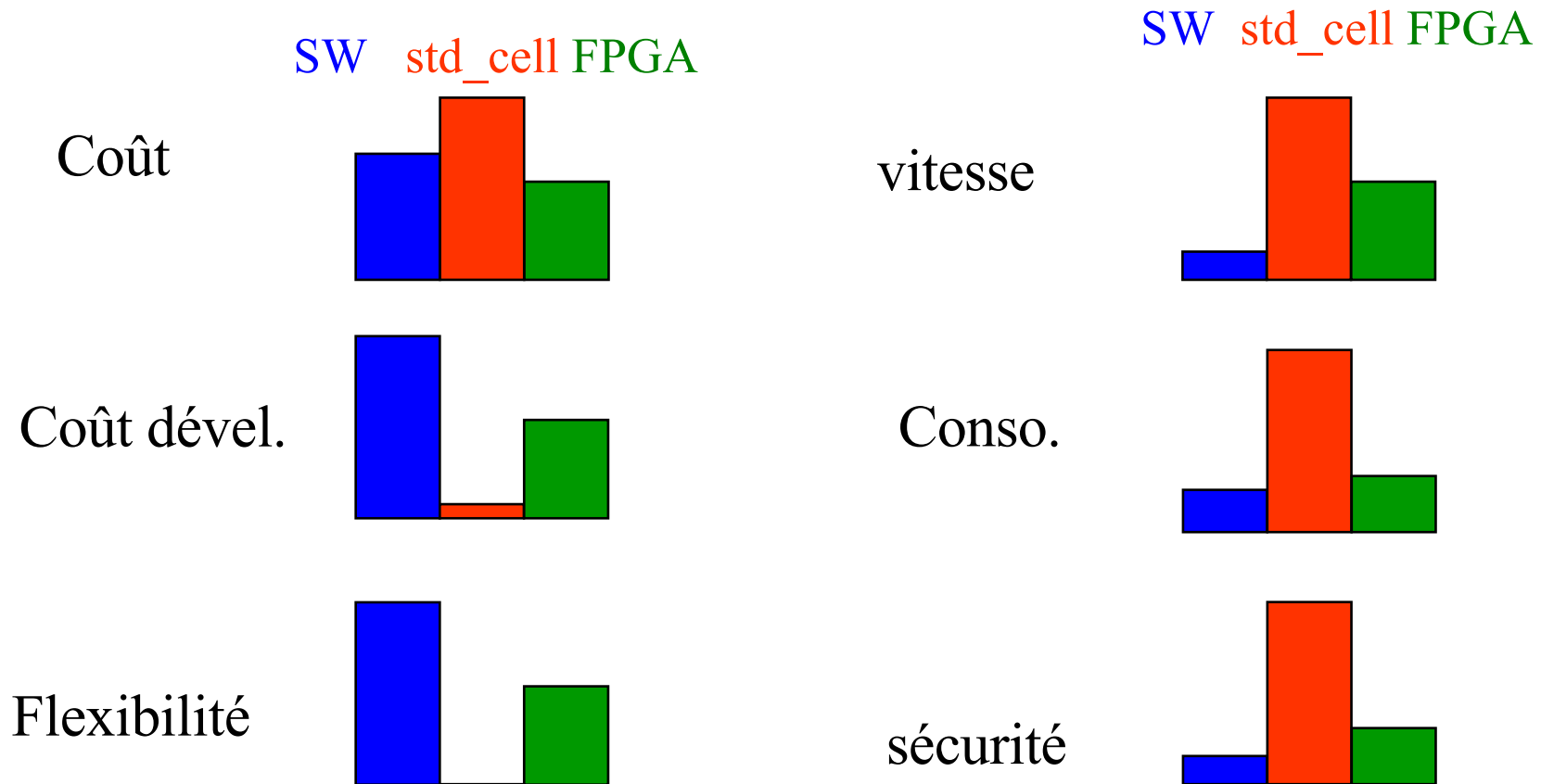
Faisabilité



**Recherche du
meilleur compromis
dans l'espace des
contraintes**

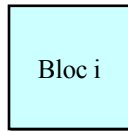
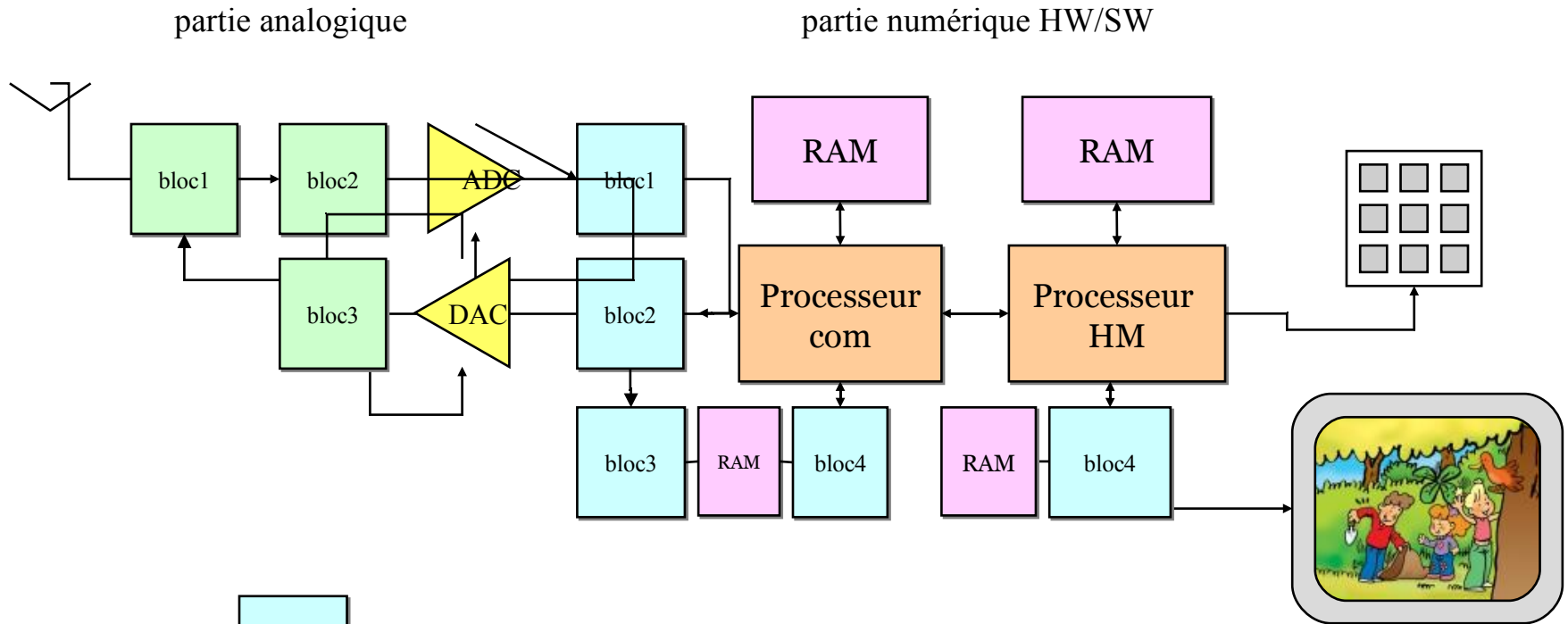
**Priorités ?
Métriques ?**

Facilité des technologies à respecter les contraintes

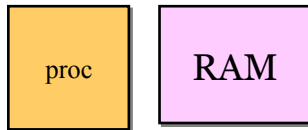


=> Complémentarité HW et SW

Exemple Architecture mixte HW/SW



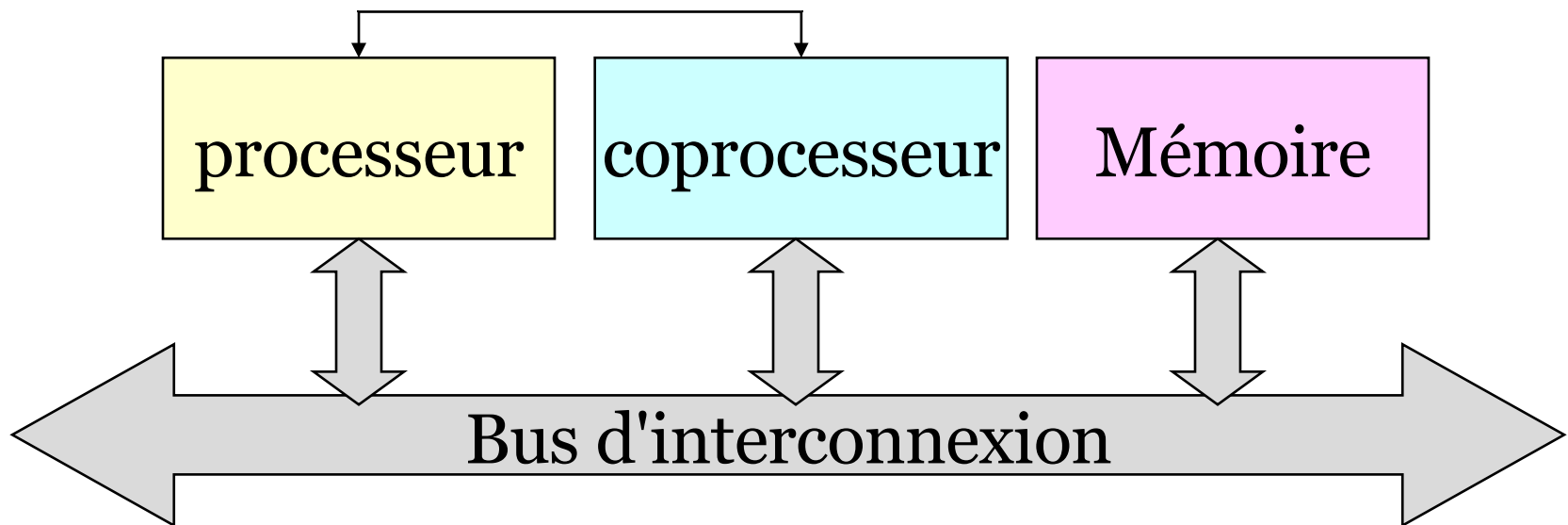
: HW spécifique : tâches répétitives, calcul intensif



: HW générique + SW : tâches des couches hautes et interface homme-machine

Architecture HW/SW classique

✓ **processeur+ accélérateur ASIC+ mémoire + bus**



Architecture ASIP

✓ "Application Specific Integrated Processor"

- processeur avec unité d'exécution spécifique => nouveau compilateur

	ASIC	ASIP	proc+ASIC	processeur
Performances	+++	++	+	
Consommation	+++	+	+	
Flexibilité	-	++	++	++
Temps de conception		--	-	+++

Méthodologie pour AAA

✓ Synthèse Algorithme $\Rightarrow$ Architecture

□ *Très difficile, problème NP complet*

- Heuristiques basée sur l'expérience propre
- Outils de synthèse d'architectures pour certaines classes d'algorithmes

✓ Validation Algorithme $\Rightarrow$ Architecture

□ *raffinement voire changement de modèle pour considérer :*

- Informations temporelles
- Parallélisme
- Précision finie

AAA pour la vitesse

✓ Critères :

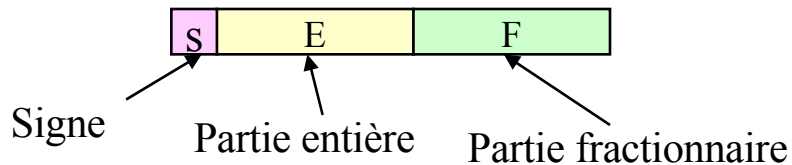
- ❑ *Latence de calcul*
- ❑ *Débit de calcul en opérations/seconde*
- ❑ *Débit E/S*
- ❑ *Débit mémoires*

✓ Optimisations

- ❑ *Représentation nombre flottant => virgule fixe*
- ❑ *Calculs parallèles*
- ❑ *Quantification non linéaire*
 - augmentation précision dans les zones de valeurs critiques (Log, Mu-Law,...)
 - Opérateurs plus petits => Gain en vitesse et complexité
- ❑ *Partage des ressources : mettre en facteur les opérateurs*
 - Gain en vitesse et complexité
- ❑ *Organiser la mémoire externe*
 - pour ne pas avoir à lire les mêmes données
 - Accéder en rafale
- ❑ *Travailler sur des petites mémoires locales (cache)*

Représentation virgule fixe/flottant

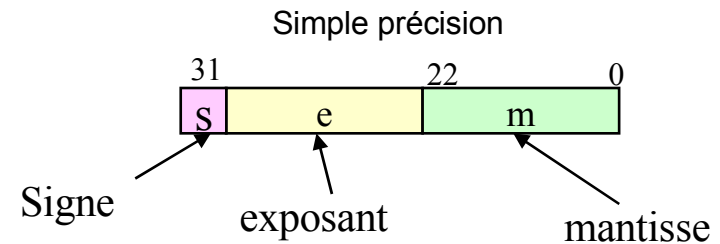
✓ Représentation en virgule fixe



Représentation en Complément à 2
 Nombre sur N bits, F sur b bits

$$x = \frac{1}{2^b} \left[-2^{N-1} x_{N-1} + \sum_{i=0}^{N-2} 2^i x_i \right]$$

✓ Représentation en flottant IEEE 754



e	m	valeur
$0 < e < 255$	$\forall$	$(-1)^s 2^{e-127} (1, m)$
0	$\neq 0$	$(-1)^s 2^{e-126} (0, m)$
0	0	0
255	$\neq 0$	NaN
255	0	$(-1)^s \infty$

Complexité de l'addition en flottant

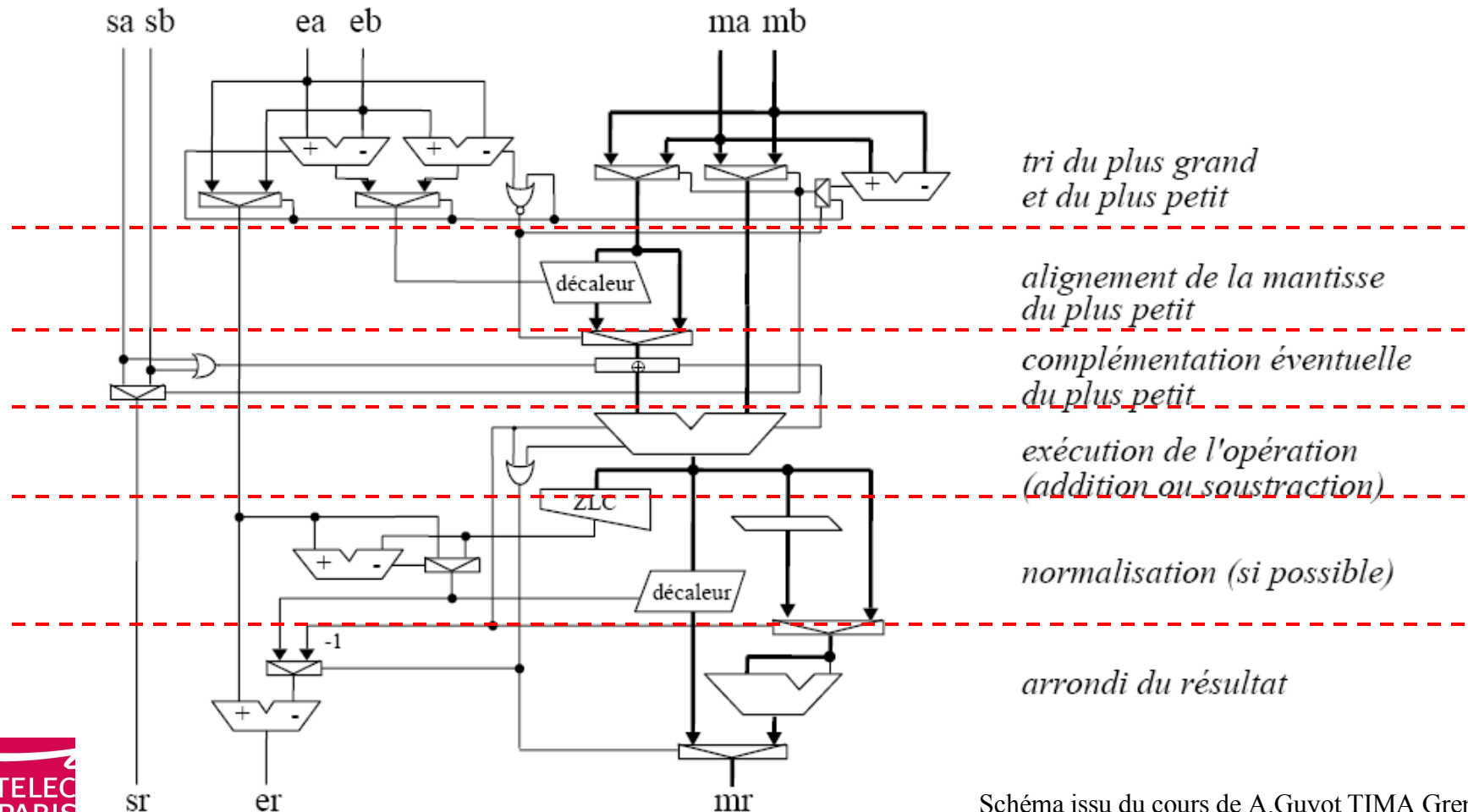
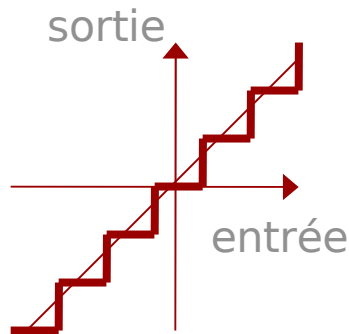


Schéma issu du cours de A.Guyot TIMA Grenoble

Problèmes engendrés par la précision finie

- ✓ Bruit de quantification à considérer en traitement du signal

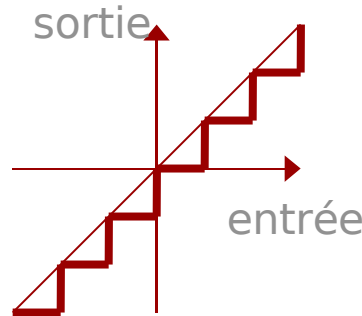
ARRONDI



Erreur moyenne $\mu_e = 0$

variance $\sigma_e^2 = \frac{2^{-2b}}{12}$

TRONCATURE



$$\mu_e = -\frac{2^{-b}}{2}$$
$$\sigma_e^2 = \frac{2^{-2b}}{12}$$

addition

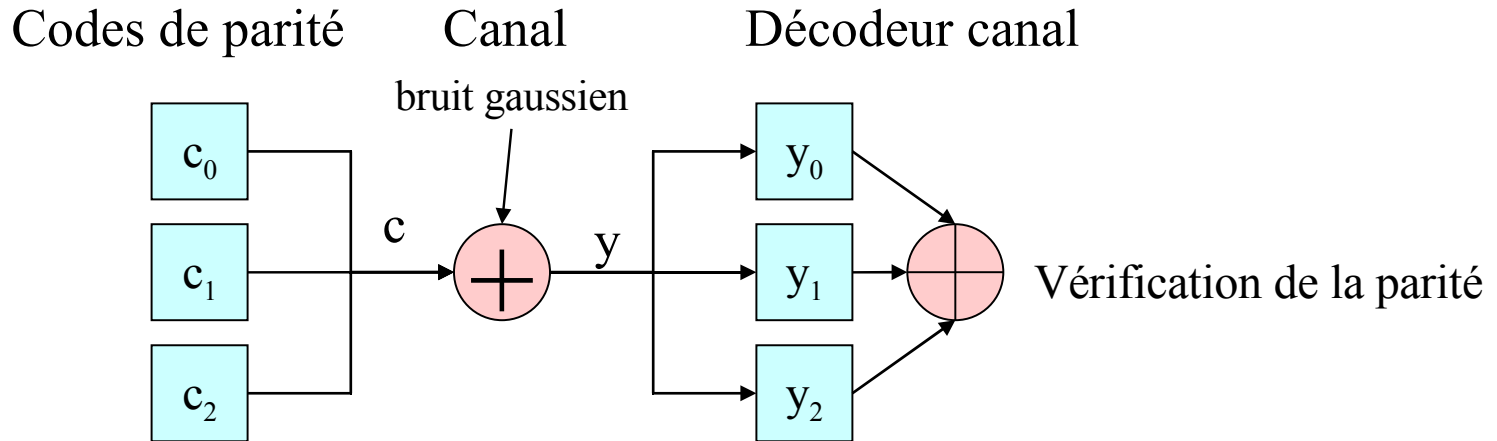
$$\mu_{x+y} = \mu_x + \mu_y$$
$$\sigma_{x+y}^2 = \sigma_x^2 + \sigma_y^2$$

multiplication

$$\mu_{x \times y} = \mu_x \times \mu_y$$
$$\sigma_{x \times y}^2 = \sigma_x^2 \times \sigma_y^2 + \mu_x^2 \times \sigma_y^2 + \mu_y^2 \times \sigma_x^2 + \mu_x^2 \times \mu_y^2$$

=> Validation algorithmique à refaire en précision finie

Intérêt du changement de domaine : Log



$$\begin{array}{c}
 \text{Information intrinsèque} \quad \text{Information extrinsèque} \\
 \underbrace{P(y_n/c_n)} \quad \underbrace{P(c_n/y_{n' \neq n})} \\
 P(c_n/y) = \frac{P(y_n/c_n)P(c_n/y_{n' \neq n})}{P(y_n/y_{n' \neq n})} \quad \Rightarrow \quad \text{LLR} \\
 \underbrace{\hspace{10em}}_{\text{normalisation}} \quad \text{Log-Likelihood Ratio} \\
 \log \frac{P(c_n=0/y)}{P(c_n=1/y)} = \log \frac{P(y_n/c_n=0)}{P(y_n/c_n=1)} + \log \frac{P(c_n=0/y_{n' \neq n})}{P(c_n=1/y_{n' \neq n})} \\
 \text{Information intrinsèque} \quad \text{Information extrinsèque}
 \end{array}$$

- Plus besoin de normalisation
- Pas de multiplication et divisions

Partage des ressources

✓ Convolution rapide

Terme identique

$$\begin{array}{l} S(t+1) = X_{t+1} H_0 + X_t H_1 \\ S(t) = X_t H_0 + X_{t-1} H_1 \end{array} \quad \Rightarrow \quad \begin{array}{l} S(t+1) = (X_{t+1} - X_t) H_0 + X_t (H_0 + H_1) \\ S(t) = X_t (H_0 + H_1) + (X_{t-1} - X_t) H_1 \end{array}$$

$$\begin{array}{l} \text{Avec } X_t = [x_t \ x_{t-2} \ \dots \ x_{t+2-Q}] \\ H_0 = [h_0 \ h_2 \ \dots \ h_{Q-2}] \\ H_1 = [h_1 \ h_3 \ \dots \ h_{Q-1}] \end{array}$$

- gain de 3/4 des multiplications lié au terme $X_t(H_0 + H_1)$ commun et division par ~ 2 de la bande passante

Se réitère avec 4 échantillons, gain en multiplication de 9/16! Et division par ~ 4 de la bande passante

Autre intérêt : pas besoin de stocker des résultats intermédiaires

AAA pour le coût (complexité)

✓ Critères

- ❑ *Nbre de portes ou surface silicium*
- ❑ *Taille et type des mémoires (ROM, RAM, FIFO Dual port,...)*
- ❑ *Nbre d'E/S (boitier)*

✓ Optimisations

- ❑ *Représentation nombre flottant => virgule fixe*
- ❑ *Sérialisation des algorithmes (boucles de calculs, algorithmes itératifs)*
- ❑ *Représentation des données dans un domaine différent :*
 - Temps $\Leftrightarrow$ fréquence
- ❑ *Quantification non linéaire*
 - augmentation précision dans les zones de valeurs critiques (Log,Mu-Law,...)
 - Opérateurs plus petits => Gain en vitesse et complexité
- ❑ *Partage des ressources*
- ❑ *Traitement de blocs données petits => réduction taille mémoires*
- ❑ *E/S multiplexée*

Complexité portes et mémoire

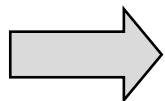
✓ Ordre de grandeur de complexité (en technologie STM 130nm)

□ *logique* :

- Multiplieur 32*32 signé : $40000\mu\text{m}^2$ @ 250MHz
- 1000 bascules D : $34000\mu\text{m}^2$

□ *mémoire*

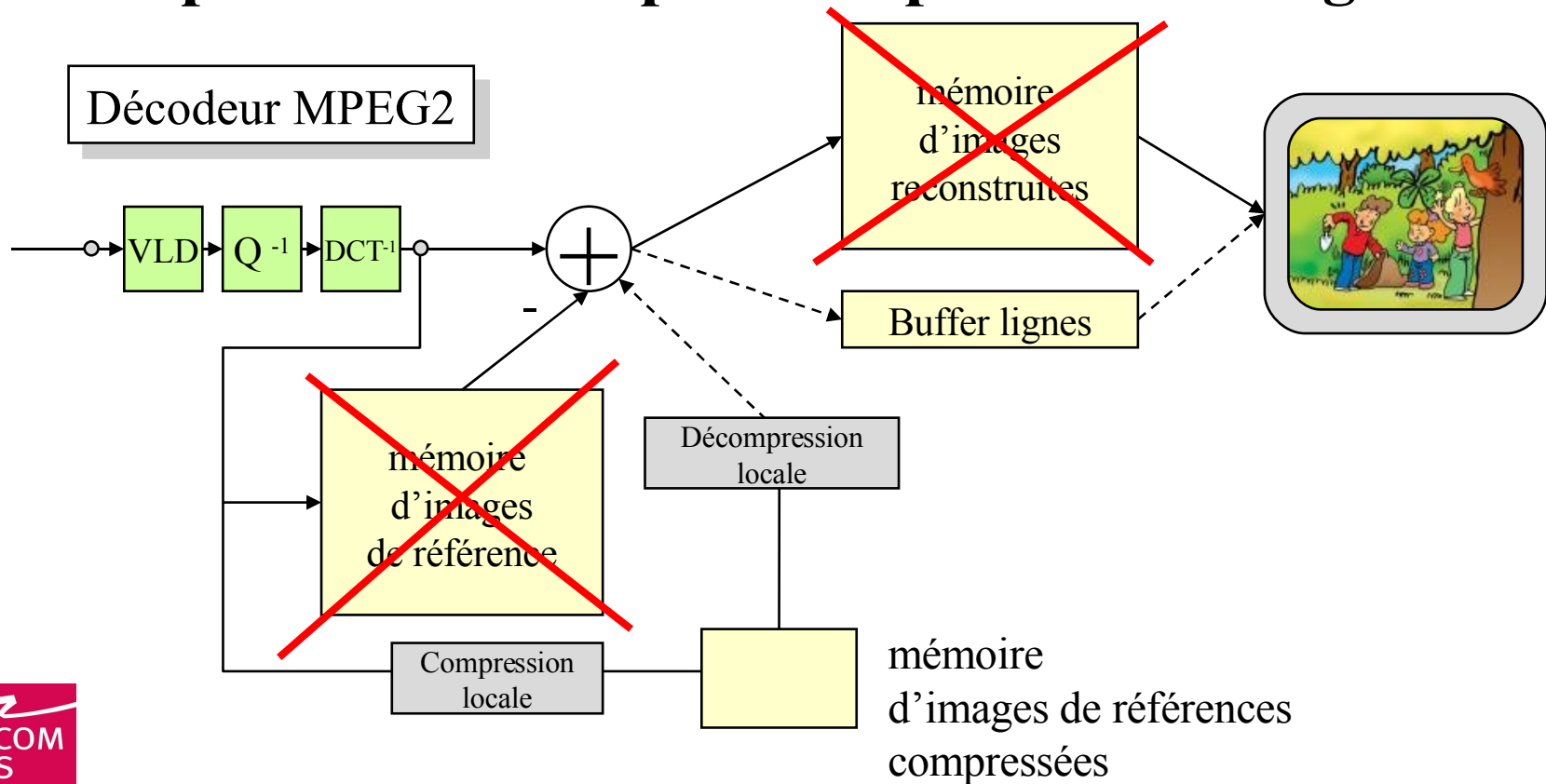
- 1K bit : $9000\mu\text{m}^2$
- 10K bits : $45000\mu\text{m}^2$
- 100K bits : $340000\mu\text{m}^2$
- 1M bits : 3mm^2
- Environ 2 fois plus si DUAL PORT ou FIFO



La mémoire est souvent prépondérante pour la complexité, et donc le coût !

Exemple de réduction mémoire

- ✓ Utiliser des algorithmes de compression/décompression pour le stockage



Intérêt du changement de domaine : FFT

✓ Convolution circulaire

$$y_t = \sum_{i=0}^{L-1} h_i x_{t-i}$$

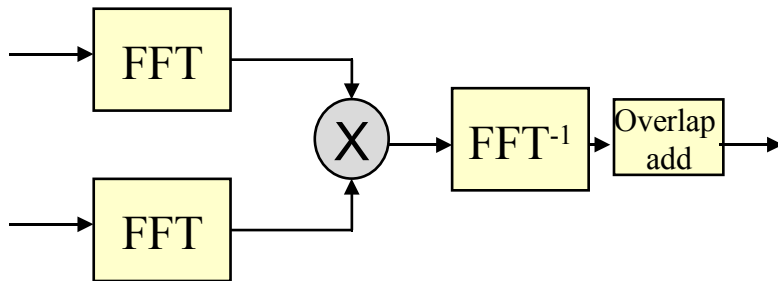
2L-1 points L points L points

DFT

$$Y(e^{j2\pi f}) = H(e^{j2\pi f}) X(e^{j2\pi f})$$

2L-1 points 2L-1 points dont L à 0

$O(L^2)$ $O(L \log(L))$ Si FFT



L	16	128	1024	8192
Nbre multiplications FFT/conv	81%	17%	3%	0,5%

Problème : latence de calcul $> 2L$

Exemples de sérialisation des algorithmes

✓ Filtres numériques FIR $\Rightarrow$ IIR

- *Attention aux problèmes de précision et stabilité*

✓ CORDIC

- *Calcul trigonométrique par approximation successive*

✓ Décodage canal itératif

- *Algorithme sous optimaux mais faisables par rapport au maximum de vraisemblance*
 - Turbodécodeurs
 - LDPC décodeurs

AAA pour la consommation

✓ Critères

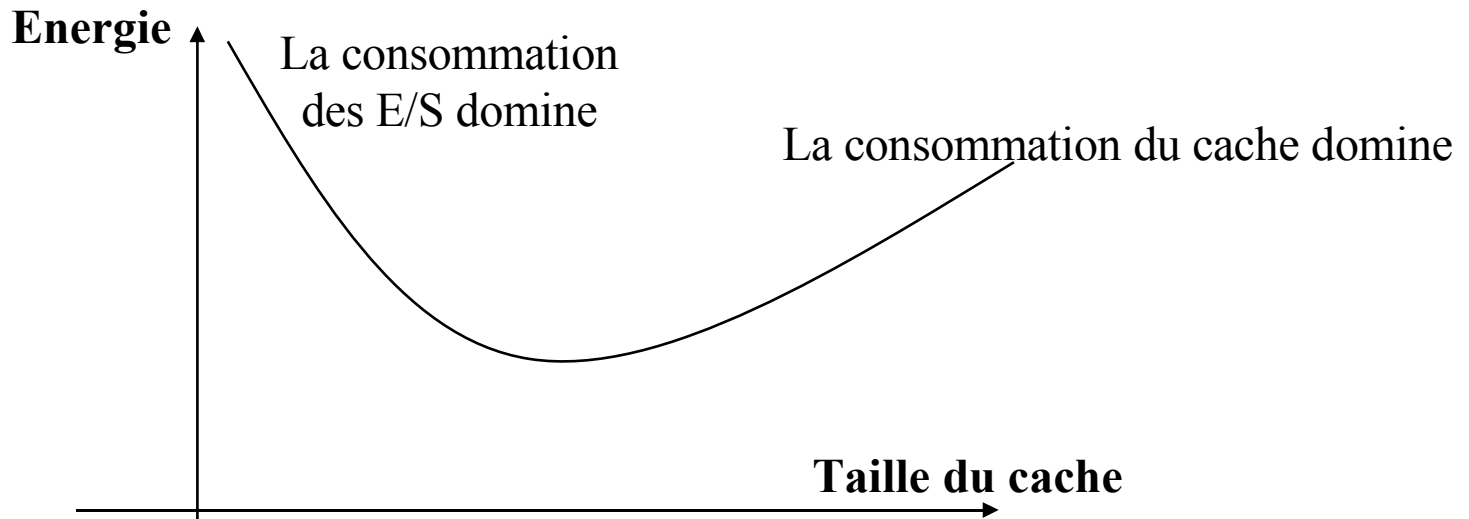
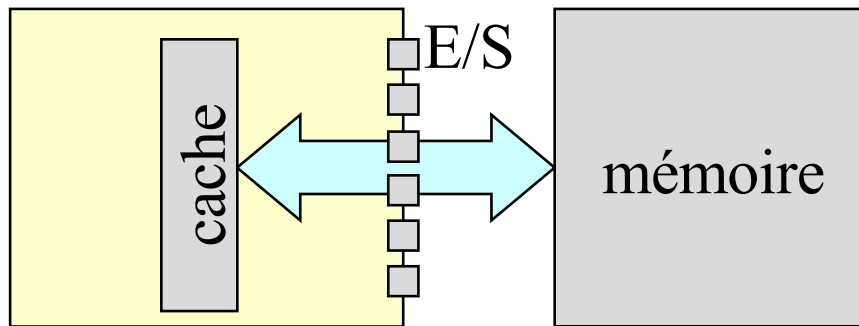
- ❑ *faible activité des signaux internes*
- ❑ *faible activité des signaux E/S*

✓ Solutions

- ❑ *Partage des ressources*
- ❑ *Diminution des débits E/S*

Exemple de réduction de la consommation

✓ dimensionnement d'une mémoire cache



AAA pour la flexibilité

✓ Critères

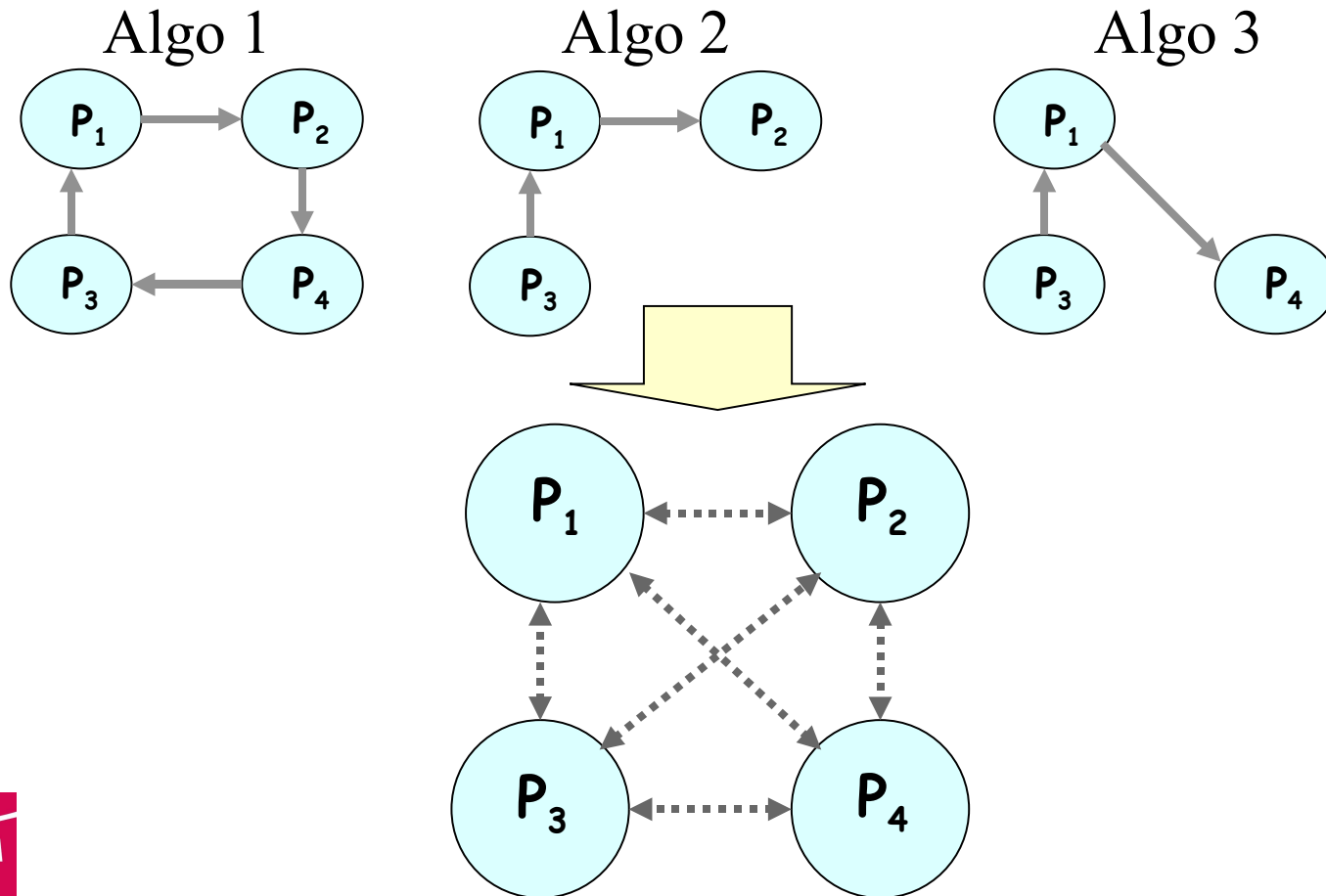
- ❑ *Changement rapide de fonction*
- ❑ *Nombre important de fonctions*

✓ Solutions

- ❑ *Partage de ressources*
- ❑ *Grain de calculs fin*
- ❑ *Calculs itératifs*

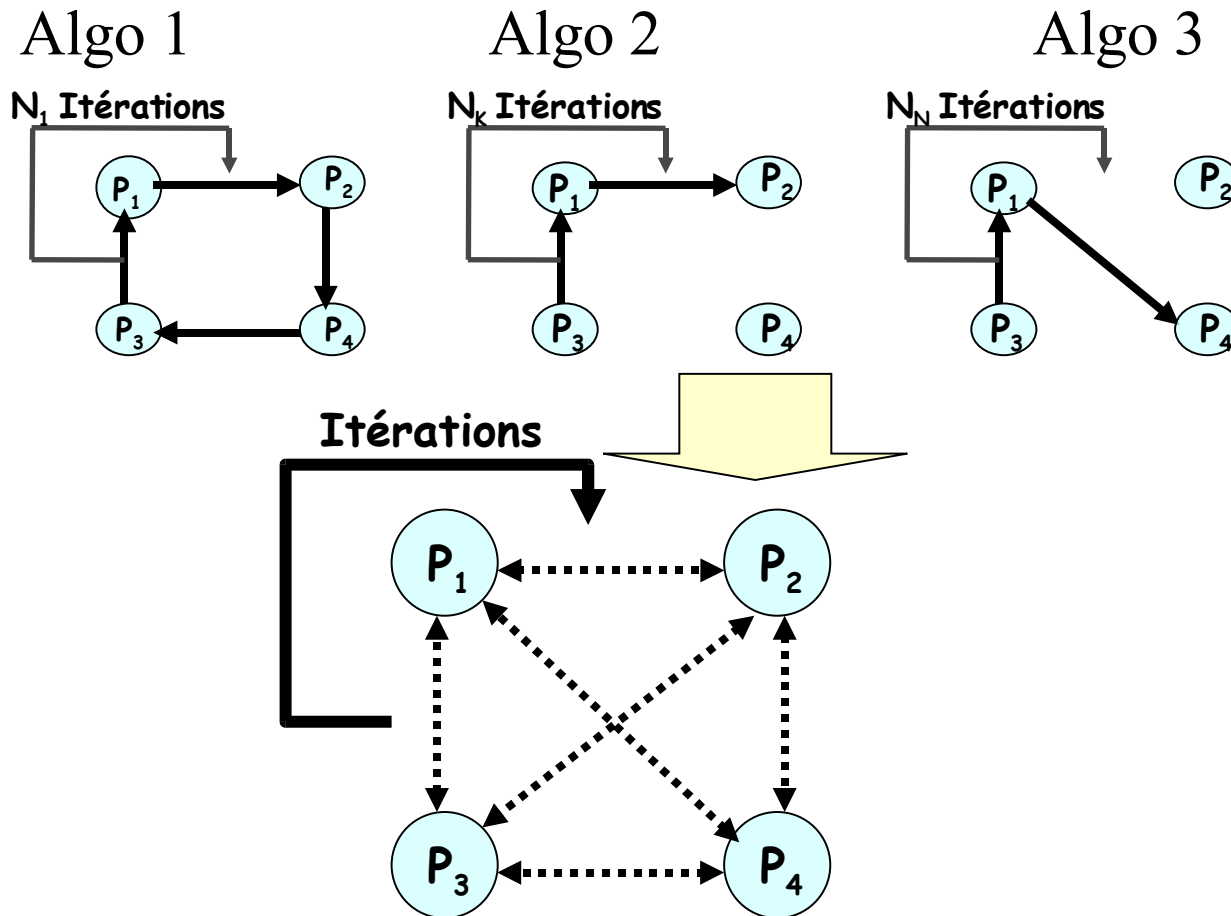
Approche "factorisation"

✓ Utiliser les grains de calculs communs



Approche itération

✓ Grain de calcul fin et calcul itératifs



AAA pour le coût de développement

✓ Critères

- ❑ *Ressources humaines*
- ❑ *Ressources outils*
- ❑ *Temps de développement*

✓ Solutions

- ❑ *Réutilisation des algorithmes existants*
- ❑ *Modèle unique algorithme et architecture*

AAA pour la sécurité

✓ Critères

- ❑ *Respect des standards cryptographiques*
- ❑ *Résistance aux "attaques" par cryptanalyse*
- ❑ *Résistance aux "attaques" par injection de fautes*

✓ Solutions

- ❑ *Structures sécurisées*
- ❑ *Masquage(HW) ou obfuscation(SW)*

Conclusions

- ✓ Nécessité **d'adapter l'algorithme** pour rendre son architecture faisable, c-à-d respecter les contraintes fonctionnelles et physico-économiques.
- ✓ Les solutions architecturales mixtes **HW/SW** font les meilleurs compromis pour le respect du maximum de contraintes.
- ✓ Les **contraintes économiques** sont souvent **prioritaires**, d'abord essayer une architecture SW
- ✓ Pas de **méthodologie** efficace pour passer du modèle algorithmique à l'architecture
- ✓ Nécessité d'avoir un **modèle architectural de référence** pour la validation et le passage à la conception d'une architecture => **brique DESSIN**

Domaines étudiés dans AAA (1)

✓ Communications Numériques

□ *Comment garder de bonnes performances fonctionnelles (en BER) après dégradations nécessaires pour la faisabilité*

- Décodeurs LDPC
- Décodeurs Turbocodes
- Détecteur à détection d'énergie multibande

✓ Radiologique

□ *Comment construire des architectures matérielles flexibles pour répondre aux besoins des communications multi-standard*

Domaines étudiés dans AAA (2)

✓ Multimédia

□ *Algorithmes standardisés et nécessitant d'énormes puissance de calcul. Quelles sont les "bonnes" architectures ?*

- Compression vidéo
- Synthèse d'images

✓ Synthèse d'architecture

□ *Peut-on obtenir automatiquement une architecture à partir d'un modèle algorithmique écrit en C ?*