Conception de Systèmes de Communications Numériques

– CSCN –

Markus Muck, Xavier Miet
Markus.Muck@motorola.com

Motorola Labs Paris (CRM)

Motorola Labs – CRM Paris

Motorola consacre chaque année environ 10 % de son chiffre d'affaires à la recherche et au développement. Ce montant représentait 4,4 milliards de dollars en 2000. Le **Centre de Recherche de Motorola CRM Paris** fait partie des « **Motorola Labs** » qui regroupent plus de 1000 chercheurs aux États-Unis, en Europe et en Asie.

La mission du centre de recherche est de développer les technologies correspondant aux besoins de **Motorola**, en particulier pour servir les marchés européens. A ce titre, il participe à des programmes de recherche européens et s'implique largement dans les organismes de normalisation.

Le développement de **systèmes de communications sans fil** dans les bâtiments (**Wireless LAN**) s'est accéléré avec l'émergence de nouvelles normes telle que **l'ETSI BRAN HIPERLAN/2** ou **IEEE 802.11a/b/g**, visant à établir des communications cellulaires allant de 6 à 54 Mbps par bande de 20 MHz à la fréquence de 5 GHz. Cette technologie succède aux systèmes de communications personnels propriétaires tels que Bluetooth et Home RF limités à des débits du mégabit/s.

Loin de se limiter à ces systèmes, le centre de recherche s'intéresse aussi à plus long terme aux technologies bande de base et radio fréquence permettant de franchir la barrière des 100 Mbps par terminal en s'attaquant à la bande des 60 GHz.

Dans ce contexte, les challenges pour la recherche sont nombreux. Le « **Broadband Systems and Technologies Lab** » a pour but d'étudier ces systèmes large bande et de développer et implémenter sur une plateforme de recherche ces nouveaux **algorithmes avancés de communications numériques**.

Métier d'Ingénieur Motorola

Recherche:

Niveau d'études BAC+5, DEA ou Thèse
Compétences assez vastes
Mission plutôt à long terme et à définir soi-même
Participation aux projets européens
Contact avec des secteurs produits
Participation à la standardisation
Recherche moyen/long terme nécessitant une auto-motivation

Développement:

Niveau d'études BAC+5, éventuellement DEA/Thèse
Compétences assez pointues
Missions très précises plutôt à court/moyen terme
Participation aux projets de développement (téléphonie mobile, etc.)

Support Client / vente:

Niveau d'études BAC+5, DEA ou Thèse
Compétences assez pointues
Missions très précises plutôt à court/moyen terme
Responsabilité d'un produit et/ou d'un client
Contacts avec des clients

La normalisation IEEE

La normalisation IEEE:

Voir le site « 802wirelessworld.com » pour 802.11
La participation / droit de vote pour individus (pas l'entreprise // ETRI)
Réunions tous les 2 mois, par exemple 2005:
- janvier: Monterey, USA
- mars: Atlanta, USA
- mai: Cairns, Australia
- juillet: San Francisco, USA
- septembre: Anaheim, USA
- novembre: Vancouver, Canada
- janvier 2006: Big Island, Hawaii
- **+ F2F meetings**

Participants:

En générale des PhD en communication numériques / MAC
Souvent très expérimenté
A la fois des directeurs de recherche et des chercheurs
Majorité: Chercheur des entreprises privées
Quelques universitaires (plutôt américain)

Déroulement à l'IEEE:

1) Mise en place d'un « study group »
2) Definition d'un PAR (Project Authorization Request)
3) Definition du « Draft Standard »
4) Letter Ballot phase

Motivation / entreprises:

Discussion

Objectifs du cours

Conception de Systèmes de Communications Numériques

L'objectif de ce cours est de permettre aux élèves d'approfondir leurs connaissances et de se familiariser de manière pragmatique aux concepts de traitement de signal et de communications numériques.

Au delà du domaine algorithmique du traitement du signal, ce cours offre une perspective orientée vers les contraintes d'implémentation matérielle (ressources matérielles disponibles, représentation en virgule fixe…) notamment sur des processeurs dédiés, appelés Digital Signal Processor DSP.

On se focalisera sur les problèmes d'adéquation et d'éternels compromis algorithme-architecture à travers divers études de cas.

Un séminaire supplémentaire va illustrer le monde de la normalisation à l'exemple d'IEEE802.11(n) : ici, les contraintes de l'implémentation sont définis et doivent donc être mis en œuvre par l'ingénieur.

En d'autres termes, ce cours constitue un recueil des thèmes suivants :

- Culture générale et Introduction aux différentes technologies et méthodologies
- Tendances à moyen et long terme du marché ainsi que de la technologie et de la méthodologie employées
- Concentration de définitions et de termes utilisés dans la conception de systèmes numériques
- Capacité à mesurer les avantages et inconvénients d'une approche ASIC, FPGA et DSP

Mots clés ASIC, DSP, FPGA, SoC, SoPC, Virgule Fixe, Virgule Flottante, Pipeline, Parallélisme, Architecture et Conception Numérique, Notion de Temps Réel

Propositions de Stage

Site internet de propositions de stages – **Obligatoire**

http://www.motorola.fr/stages

	Date	Ecole Organisatrice	Lieu	Ville	Plus d'Infos
Forum Supelec	7 & 8 Dec 2005	Supelec Paris	Supelec Paris	Gif-Sur-Yvette	http://www.forum.supelec.fr
Forum Odyssée	1 Dec 2005	INP Grenoble & Grenoble Ecole de Management	Alpexpo	Grenoble	http://www.forum-odyssee.com
Forum Telecom	23 Fev 2006	ENST & INT	Paris Expo – Pte de Versailles	Paris	http://www.forum-telecom.com

3 principaux sites en France : Angers, Toulouse et Région Parisienne.

Les propositions de stage du Centre de Recherche de Motorola Paris sont référencés sur le site de Saclay.

Exemple de proposition de stage non pourvu

- Etude d'ameliorations couche MAC future generation de reseaux IEEE802.11

-Connaissances requises : C/C++ est absolument necessaire. Une connaissance de l'environnement de simulation OPNET est un plus indeniable. Connaissance des protocoles de MAC standards et des WLANs en general

- Niveau : 3ème année d'école d'ingénieur (Bac+5) et/ou DEA
- Date : selon disponibilité
- Durée : 6 mois
- Domaine : Réseaux, Electronique pour les communications numériques
- Nombre de stagiaires pour ce stage : 1

Plan

- **Chapitre I Introduction**
- **Chapitre II Architecture et Circuits DSP**
- **Chapitre III Architecture et Conception d'Opérateurs Numériques**
- **Chapitre IV Application : Conception d'un Filtre RII – ASIC et DSP**
- **Chapitre V Travaux Pratiques Sur DSP**
- **Chapitre VI Travaux Pratiques Sur FPGA**

4 Travaux Dirigés et 3 Travaux Pratiques Sur DSP

3 à 5 Travaux Pratiques VHDL/FPGA

Plan Détaillé

- **Chapitre I Introduction**
  - Introduction sur les Circuits SoC et SoPC
  - Problématiques et Remarques Générales
  - « Fast Prototyping » - Exemple de Plateformes de Recherche
- **Chapitre II Architecture et Circuits DSP**
  - Puissance de Calcul Nécessaire pour Différentes Applications
  - Opérations et Algorithmes DSP
  - Représentation Fixe vs Flottante
  - Architecture DSP
  - Caractéristiques du DSP TMS320VC5402

Plan Détaillé

- **Chapitre III Architecture et Conception d'Opérateurs Numériques**
  - Représentation en Virgule Fixe
  - Représentation en Virgule Flottante
  - Influence de la représentation en Virgule Fixe – Effets de Quantification Finie
  - Opérations d'addition et de multiplication en Virgule Fixe
  - Travaux Dirigés #1
  - Pipeline et Parallélisme
  - Travaux Dirigés #2
- **Chapitre IV Conception d'un filtre RII – ASIC et DSP**
  - Méthodologie de spécification de filtre
  - Filtre RII à correction de phase
  - Architecture et Conception d'un filtre RII – Approche Globale
  - Travaux Dirigés #3
  - Architecture et Conception d'un filtre RII – Approche Spécifique sur DSP
  - Travaux Dirigés #4

Plan Détaillé

- **Chapitre V Travaux Pratiques sur FPGA**
  - TP # 1 : Prise en main des Outils et du langage VHDL: Du Développement à la Synthèse
  - TP # 2 : Création de Librairies d'Opérateurs Numériques (Blocs Altera)
  - *TP # 3 : Exemple d'Etude Architecturale*
  - *TP # 4 : Exemple de Machine à Etats à base de Registre d'Etats*
  - *TP # 5 : Exemple de Machine à Etats à base de Machine de Moore*
  - *TP # 6 : Mini-Projet de Maximum Ratio Combining*
- **Chapitre VI Un projet: IST-BROADWAY**

Plan

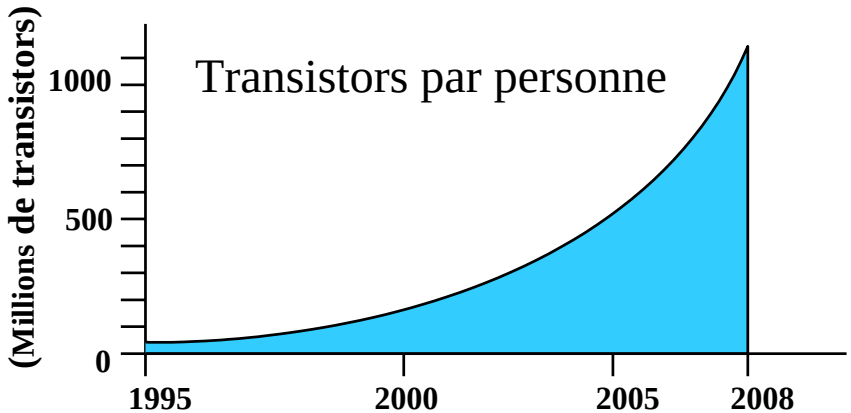
- **Chapitre I Introduction**
  - Introduction sur les Circuits SoC et SoPC
  - Problématiques et Remarques Générales
  - « Fast Prototyping » - Exemple de Plateformes de Recherche (obsolètes aujourd'hui)

Evolution du Marché des Systèmes Numériques

Applications & Besoins

Les systèmes électroniques sont de plus en plus présents dans notre quotidien :

- Ordinateurs, PDA, …
- Téléphone portable
- Audiovisuel numérique fixe ou portable
- Électronique embarquée dans l'automobile
- Baladeurs CD/MP3



Source : *Semiconductor Industry Association* (SIA)

On leur demande d'offrir :

- plus de fonctionnalités
- plus de puissance de calcul
- une consommation réduite (applications mobiles)
- une réduction des dimensions géométriques
- la reconfiguration (adaptation à tous les standards)
- un coût bas

Leur conception doit se faire :

- rapidement (mise sur le marché rapide – *Time To Market*)
- avec un souci d'évolutivité (pour suivre les normes)

Evolution du Marché des Systèmes Numériques

Problématiques : Interconnexions

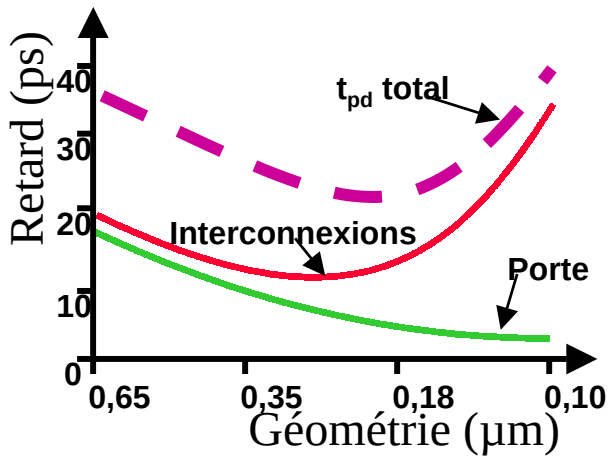
Les progrès technologiques sont tels que les pertes de performances sont maintenant imputables aux temps de propagation dans les interconnexions.

Limites technologiques :

- **Vitesse de la lumière (dans le vide) : 30 cm/ns**
- **Longueur d'onde λ = 1,5 mm à 2 GHz**

Contraintes technologiques :

- **Conditions de propagation d'onde (adaptation)**
- **Puissance : $P = \alpha CV^2f$**



Tant que l'on reste sur la puce, les distances à parcourir sont courtes et les capacités mises en jeu sont faibles. Si l'on doit échanger des données avec d'autres composants, les ordres de grandeurs changent (temps de propagation, puissance, …), ce qui limite les performances potentielles.

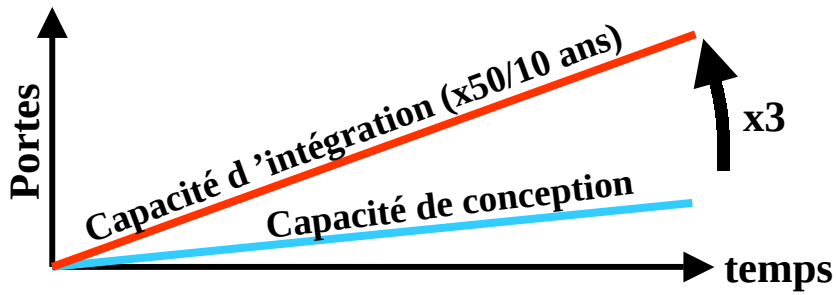
Exemple :

Les processeurs fonctionnent à 2 GHz (T = 500 ps) en interne mais ils communiquent avec l'extérieur à 400 MHz au mieux.

Evolution du Marché des Systèmes Numériques

Problématiques : Outils de Conception

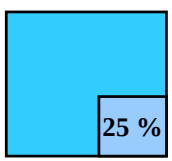
La technologie évolue vite mais les outils de conception évoluent plus lentement



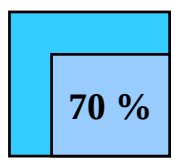
Le délai moyen de conception reste d'environ 1 an

Il est donc impératif de réutiliser des éléments existants
(notions de : bibliothèques, macrofonctions, IP)

Année 2001(CMOS 130 nm)
puce 100 MTransistors
concepteur : 1,2 MT/an
réutilisation/IP : 25 %

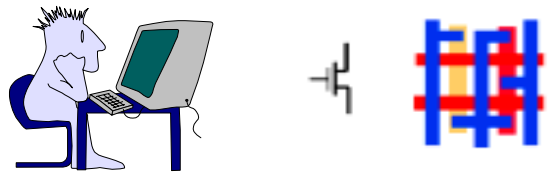


Année 2007 (CMOS 65 nm)
puce 600 MTransistors
concepteur : 5,9 MT/an
réutilisation/IP : 70 %

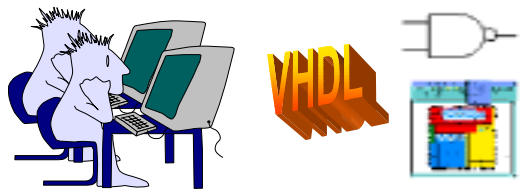


Evolution du Marché des Systèmes Numériques

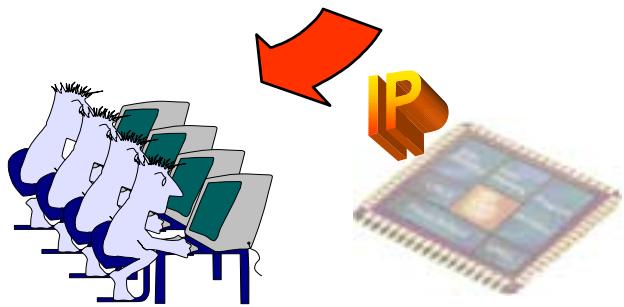
Années 70/80 : Conception *Full-Custom*
- schéma
-dessin des masques
- Exemple: Motorola Genève !
- simulation SPICE



Années 80/90 : Conception précaractérisé + FPGA
- réutilisation du matériel (briques élémentaires : *Standard Cells*)
- modélisation, simulation, synthèse



Années 00/xx :Conception SoC
- réutilisation du matériel et du logiciel (IP)
- *co-design*, vérification



Principes de Conception

La conception d'un système de traitement, c'est :

- la construction d'une architecture matérielle composée
  - de blocs logiques standards (processeurs, mémoires)
  - de blocs logiques spécifiques,
  - de bus de communications.

- le développement de ressources logicielles.

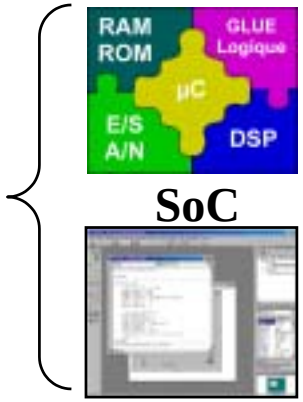
Optimiser un système, c'est adapter les ressources matérielles et logicielles aux besoins spécifiques.

L'approche SoC, c'est la cohabitation de ces ressources sur une même puce
Prise en compte globale du système dans ses réalisations matérielles et logicielles

Conception SoC

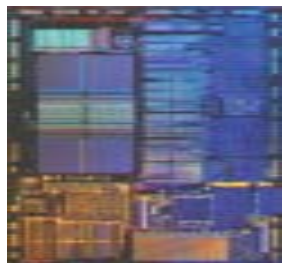
Le Concept SoC apporte une réponse à ce besoin de conception hard/soft

Réutilisation



SoC

Co-design



Bibliothèques :

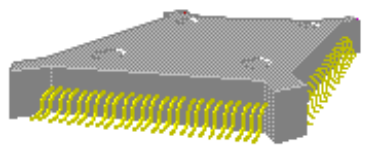
Matériel :

Blocs IP,
Composants virtuels,
Cœurs de processeurs.

Logiciel :

routines de base,
système d'exploitation,
modules spécifiques.

Développement rapide
Intégration poussée
Faible puissance



Renouvellement fréquent des gammes de produits

Les Blocs IP – Intellectual Property

Blocs fonctionnels complexes pouvant être réutilisés dans plusieurs conceptions

Hard IP : bloc déjà physiquement implanté
dépendant de la technologie
très optimisé

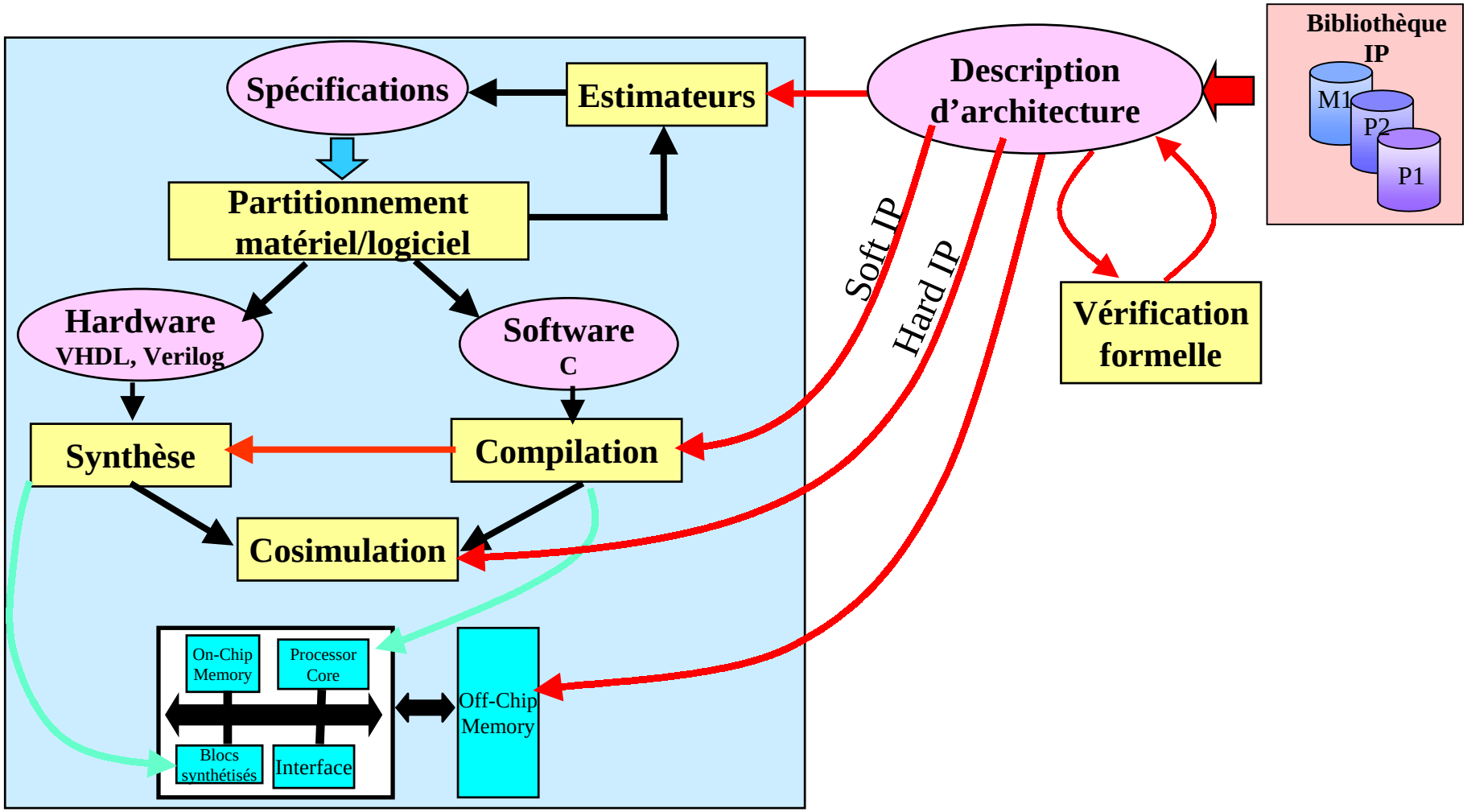
Soft IP : bloc décrit en langage de haut niveau (VHDL,verilog),
souvent paramétrable,
synthétisable

Besoin de normalisation des interfaces (AMBA, CoreConnect, …) (compatibilité fonctionnelle, temporelle, électrique)

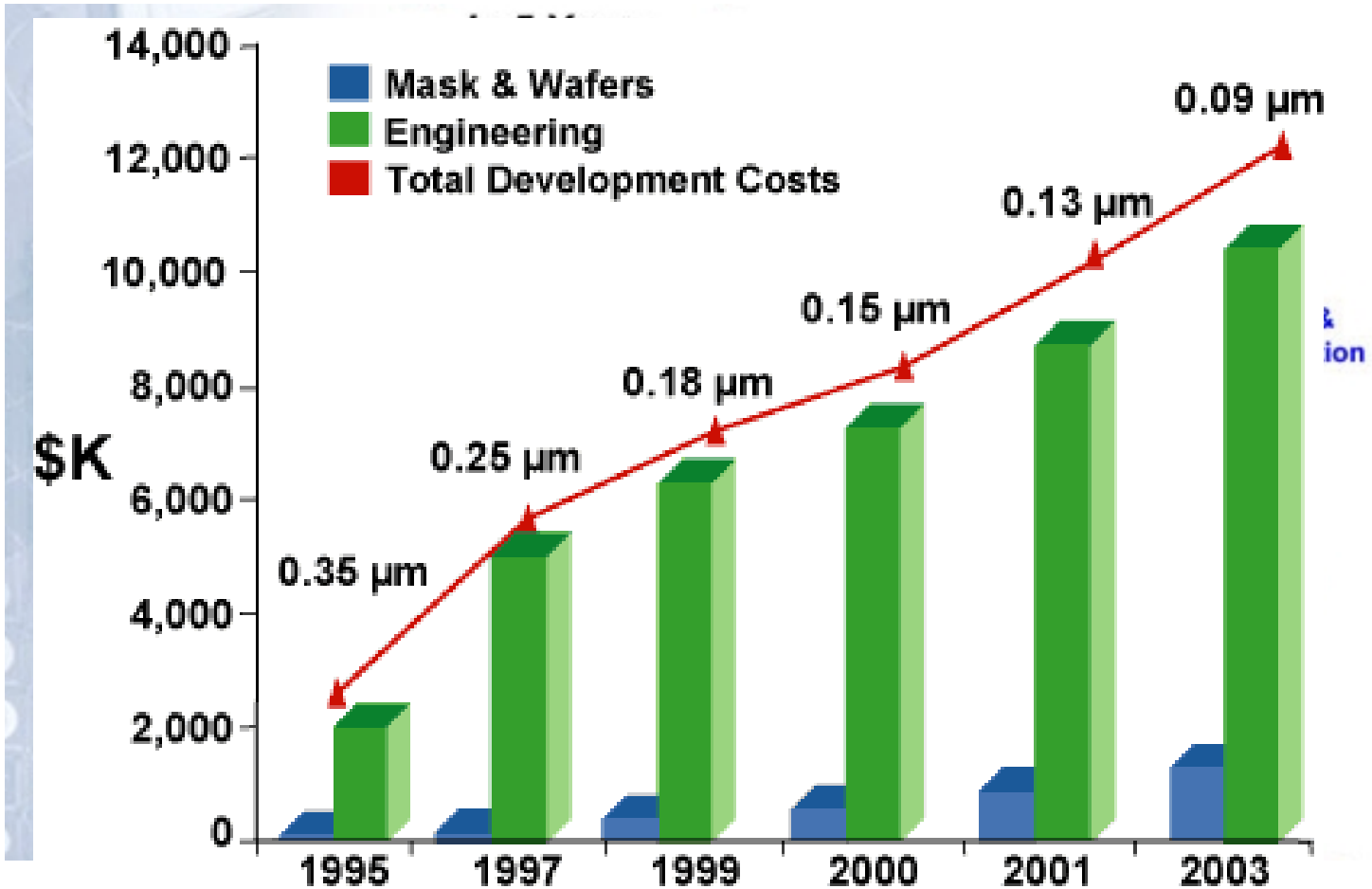
L'emploi de ces blocs dans un projet implique l'utilisation d'un environnement de développement adapté (co-design, co-vérification, …) : IBM Core Blue, Mentor-Altera, …

L'optimisation de ces blocs par rapport à la technologie n'est pas très poussée, ce qui limite leur emploi à des fonctions de performances moyennes.

Flot de Conception SoC



Estimation des Coûts de Développement ASIC



Approche SoPC

System on a Programmable Chip

L'approche SoC (technologie ASIC) répond aux besoins de performances et d'intégration mais :

- elle est peu adaptée à l'évolutivité des systèmes
- elle reste réservée aux grands volumes de production
- la fabrication et le test sont des étapes longues et coûteuses

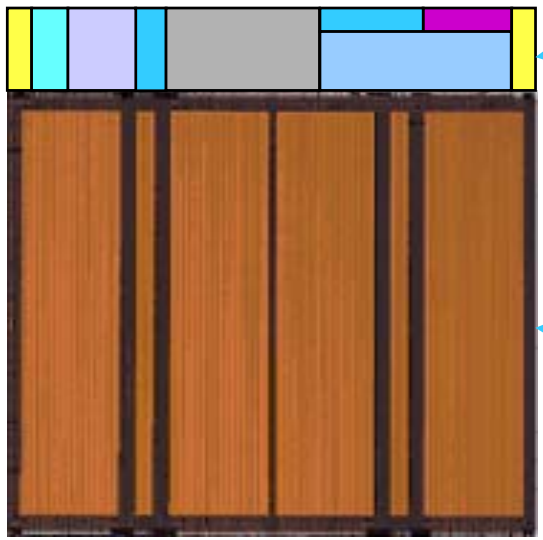
L'approche SoPC (technologie FPGA) résoud ces problèmes :

- développement et prototypage rapides
- composant reconfigurable en quelques ms et à volonté

mais

- la densité d'intégration est moindre (~10 Millions de portes)
- la consommation est plus grande
- les performances sont moindres

Exemple SoPC
EPXA10 (Altera, coeur ARM)



“Stripe” :

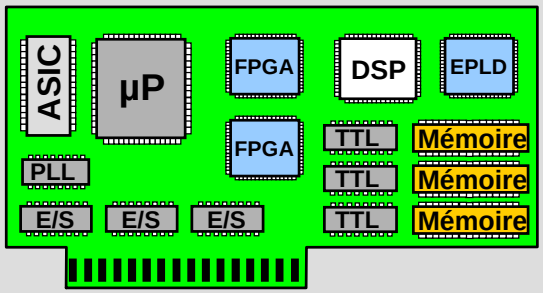
CPU ARM922T (200MIPS)
Caches instructions et données
SRAM simple et double port
périphériques (UART, WDT, INT, …)
interface bus AMBA

Zone logique programmable :

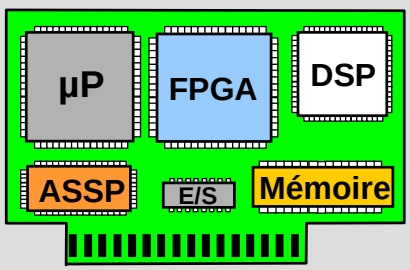
1 million de portes FPGA
38400 éléments logiques
SRAM, PLL
521 E/S

Evolution des Systèmes

Hier

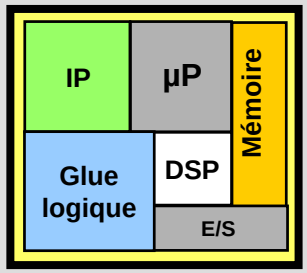


Aujourd ’hui



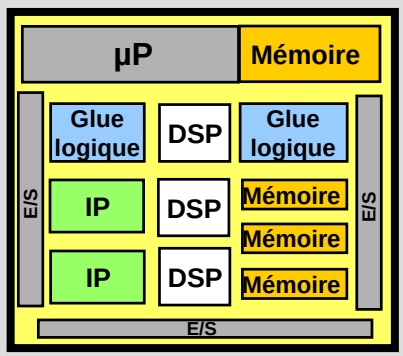
Demain

Grande diffusion : approche SoC



Reconfigurable (logiciel)
Réutilisation de composants virtuels, IP
Performances (vitesse et consommation)

Grande souplesse : approche SoPC



FPGA (jusqu’à 10 millions de portes)
Reconfigurable (logiciel ET matériel)
Réutilisation de composants virtuels, IP
Développement rapide

Influence de la Technologie

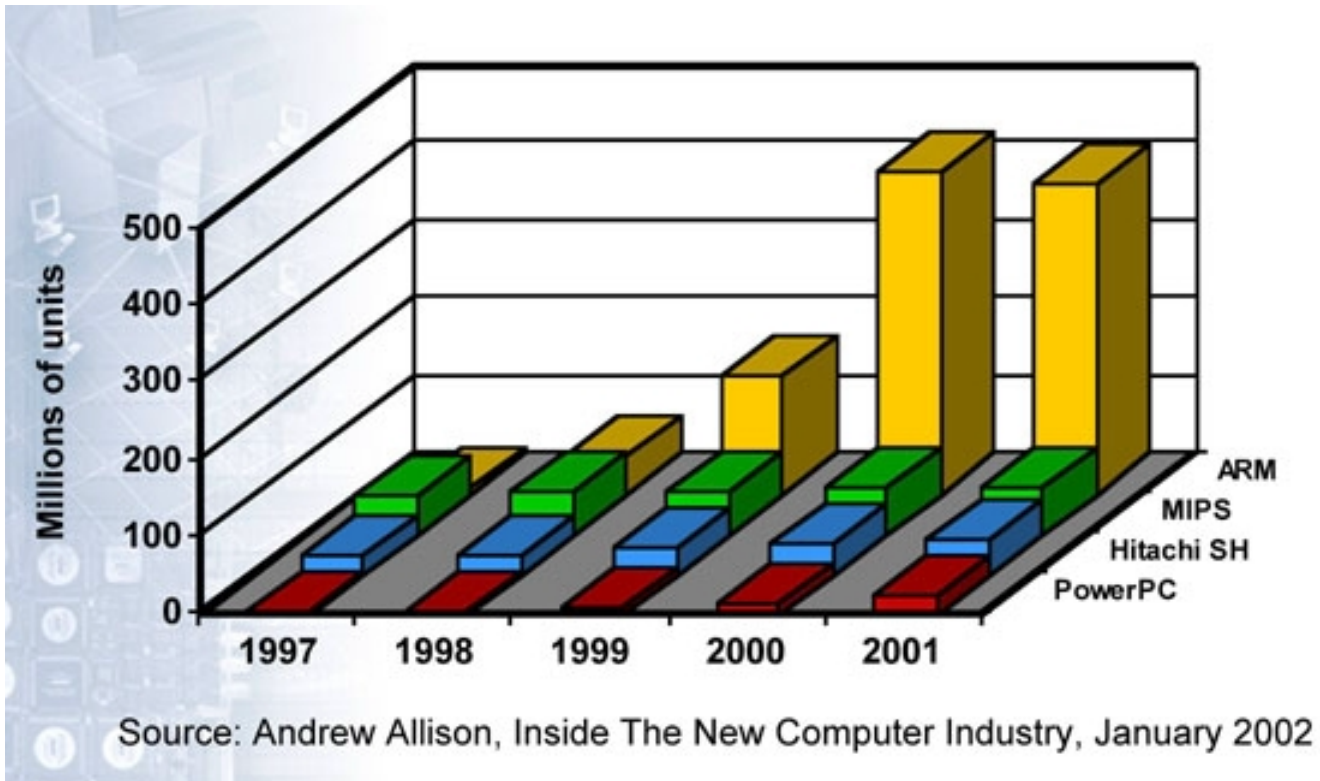
Même Architecture mais Technologies Différentes

- Processeur ARM 922T
- Technologies 0,18 µm et 0,13 µm
- Fondeur : TSMC

	ARM922T TSMC 0.18_m G	ARM922T TSMC 0.13_m G
Area	8.1mm²	3.2mm²
Frequency (typical)**	290MHz	400MHz
Frequency (worst case)*	200MHz	250MHz
Average Power (mW)	0.8mW/MHz	0.25mW/MHz
Mips/W	1375	4400

** Typical frequency: std silicon, 25°C, nominal voltage
* Worst case frequency: slow silicon, 125°C, Vcc -10%

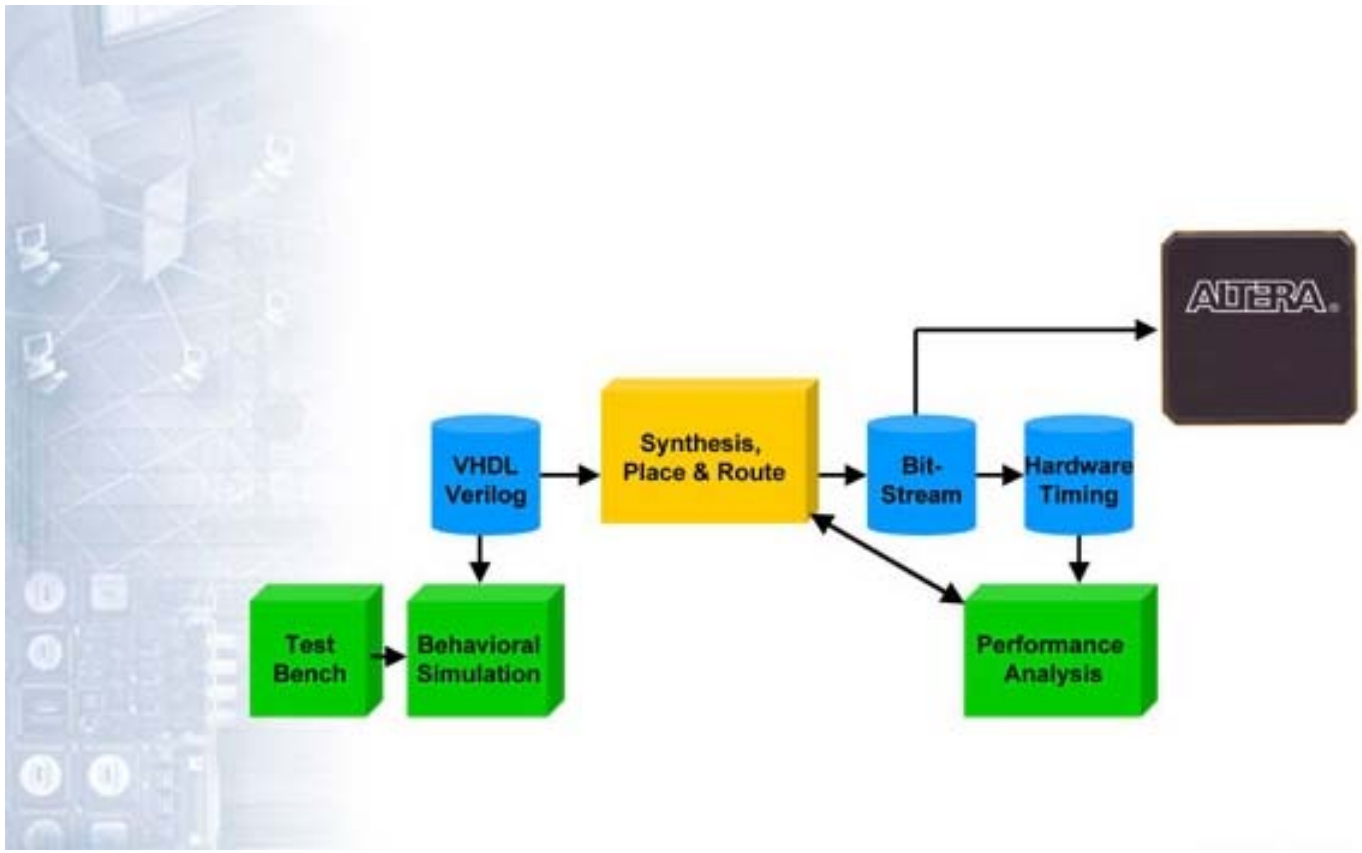
Parts de Marché des Processeurs Embarqués



Source: Andrew Allison, Inside The New Computer Industry, January 2002

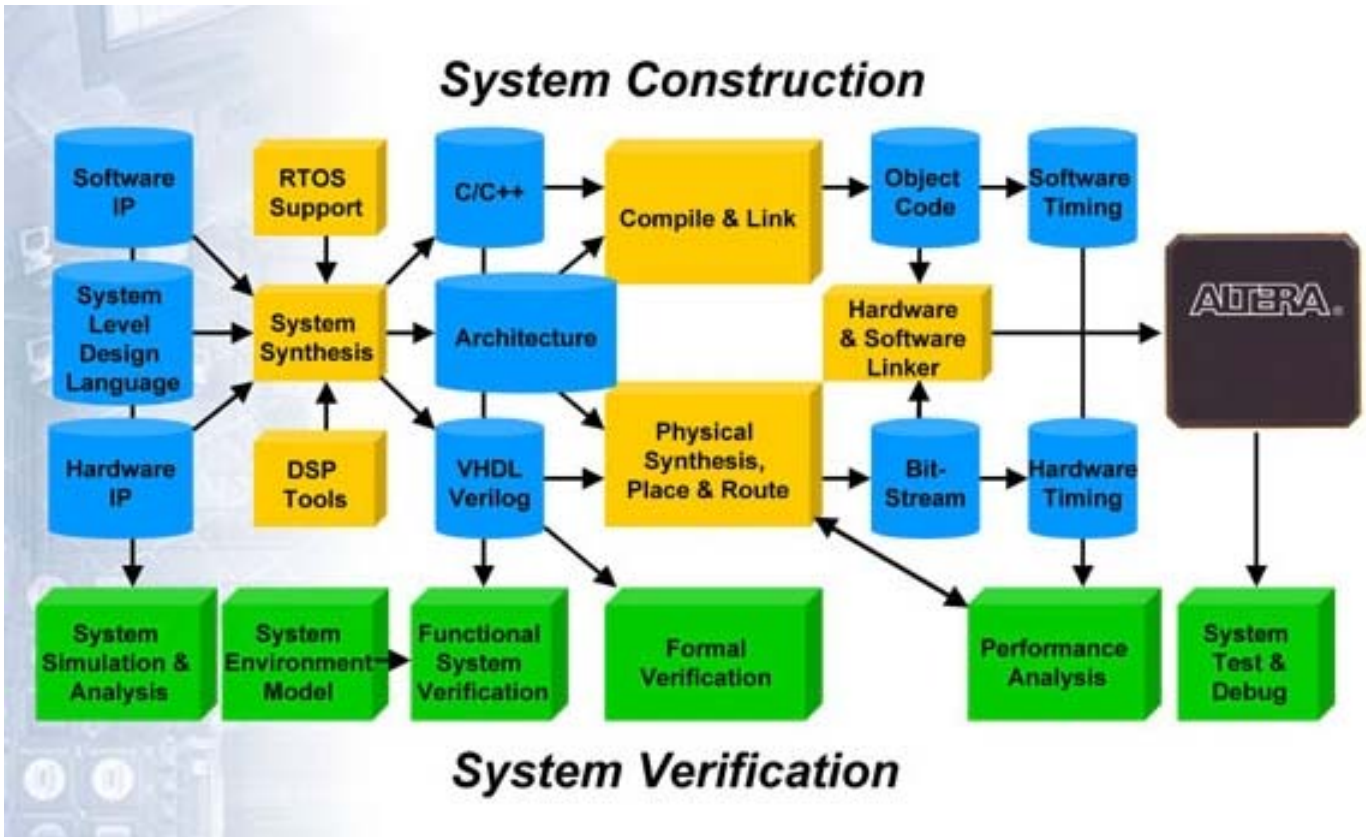
Flot de Design FPGA

Flot classique et simple

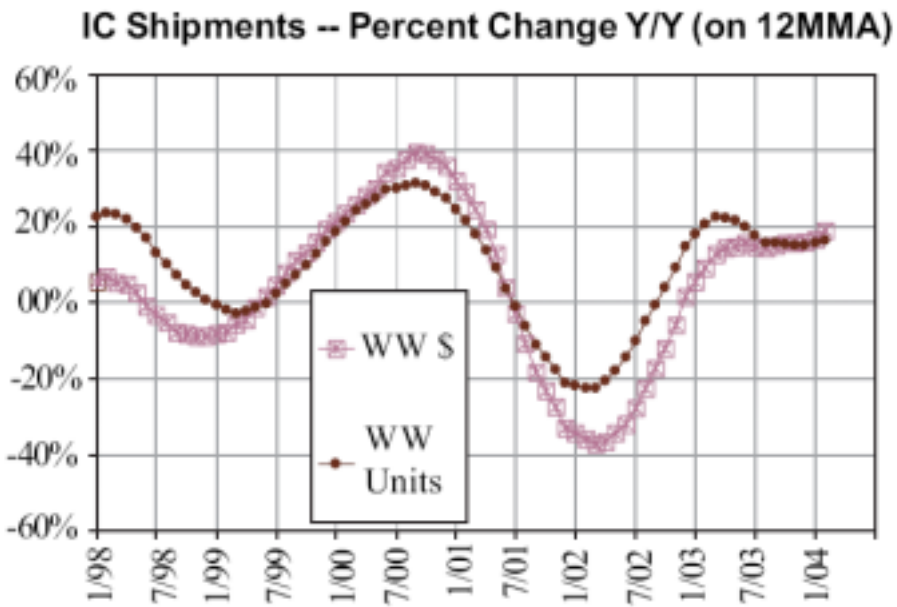


Flot de Design FPGA - SoPC

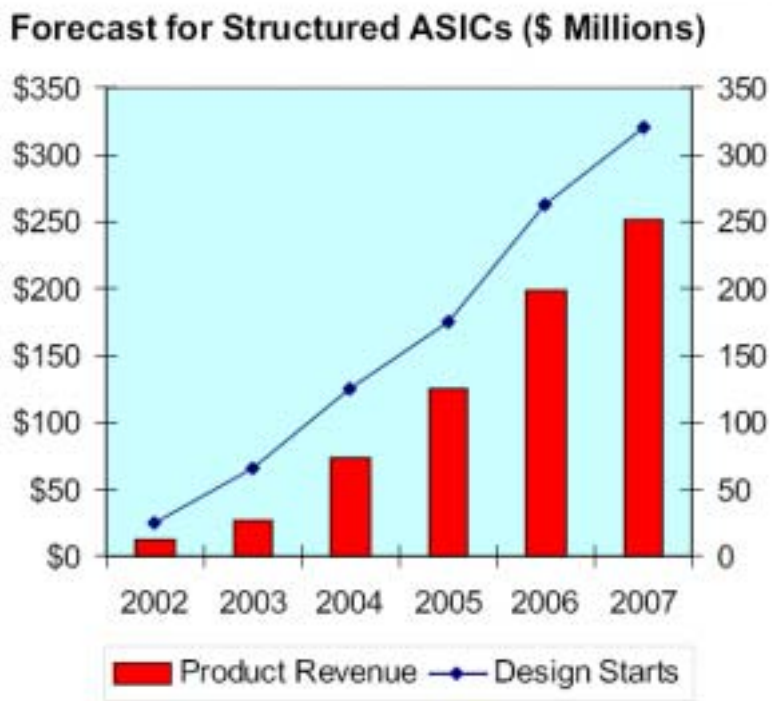
Flot complexe même si une certaine logique prédomine



Croissance Annuelle des Semiconducteurs



Source: Advanced Forecasting



Source: iSuppli

Prototypage Rapide

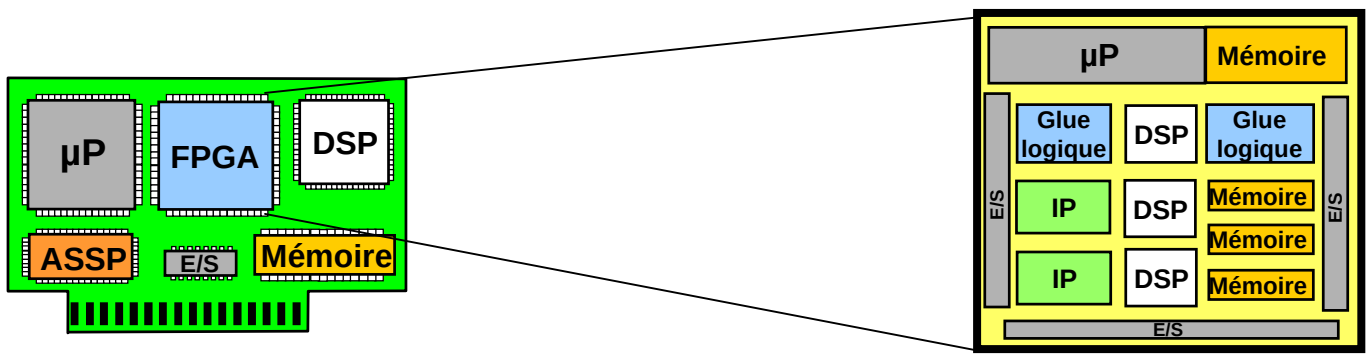
Fast Prototyping

Le prototypage rapide apparaît comme un bon moyen de valider la viabilité et la faisabilité d'un algorithme ou d'un système.

Plateforme de Recherche ou Plateforme de Prototypage

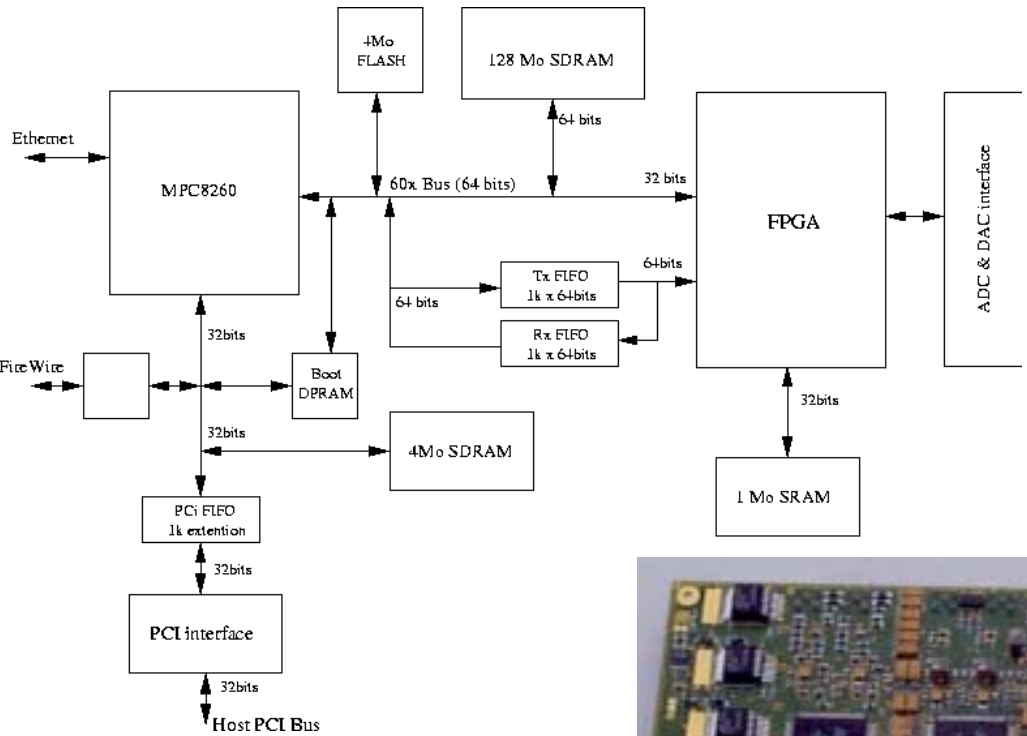
La plupart du temps composée de :

- FPGA
- Microprocesseur Embarqué (PowerPC ou ARM)
- DSP (Texas Intruments)
- Connectivités (PCI, Ethernet, FireWire, USB, Compact PCI, Rapid IO)
- Blocks Analogiques (Alimentation, ADC, DAC …)
- Modules RF



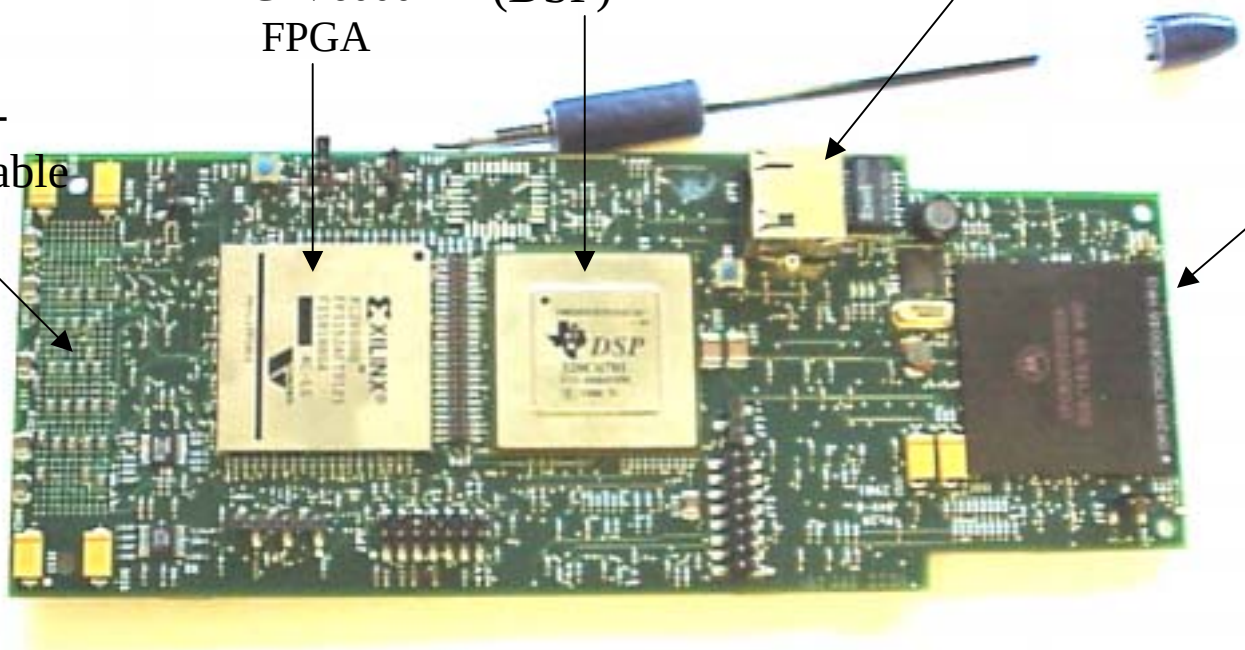
Plateforme de Recherche Motorola Labs - CRM

Advanced Research Technologies ART Platform



- **DLC software on PPC core @ 200MHz**
- **PHY on 1.5 M gate FPGA**
- **Generic MAC/DLC/PHY Interface (HL2/802.11)**
- **PCI, Ethernet, Firewire connectivities**
- **CMS based RF front-end**
- **Multipurpose prototyping (5/25/60 GHz)**

Plateforme de Recherche Lucent

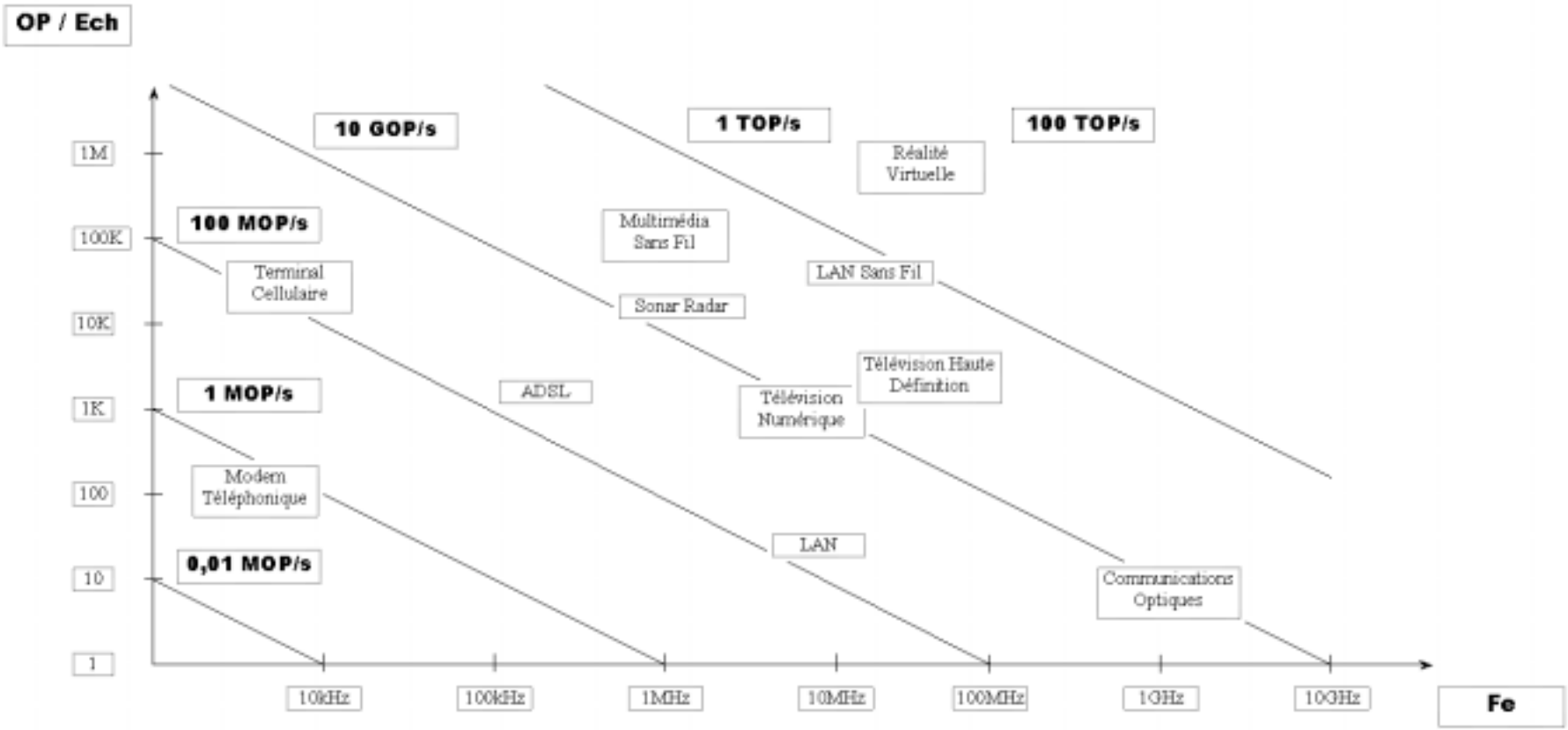


- 6 million gates of programmable logic
  - 2.5 Megabits DPRAM in FPGA
  - 144 dedicated multipliers
- 1 GFLOPS TMS320C6701 DSP
- 280 MIPS MPC8260 (10/100 Base T Ethernet interface / Embedded Linux OS)

Plan

- **Chapitre II Architecture et Circuits DSP**
  - Puissance de Calcul Nécessaire pour Différentes Applications
  - Opérations et Algorithmes DSP
  - Représentation Fixe vs Flottante
  - Architecture DSP
  - Caractéristiques du DSP TMS320VC5402

Applications



Opérations et Algorithmes DSP

Opérations Dédiées

- Principalement arithmétique (+,-,x, …)
- Codage virgule fixe ou flottant
- Opérations non linéaires
- Tris, sélections, …

Mémoires Temporaires

- Retards (z^{-1})
- Accumulation
- Réordonnancement des données
- FIFOs

Pour Quels Types d'Algorithmes ?

- Débit de calcul important
- Débits de communications importants
- Algorithmes
  - Très répétitifs
  - A faible grain (boucles de dizaines d'instruction)
  - Fonctions connues (MAC, FIR, FFT, …)
  - Fonctions spécifiques (temps réel)
- Applications professionnelles et grand public

Peut-on tirer avantage d'une des architectures spécifiques DSP ?

Représentation Fixe vs Flottante

Virgule Fixe

- Satisfaisant pour la plupart des applications (voix, modem, drives de disques, …)
- Systèmes embarqués
- Bas coût de production
- Le coût du logiciel peut être augmenté

Virgule Flottante

- Quand l'application l'exige (Graphique, Données mal conditionnées)
- Coût de production plus élevé
- Consommation plus élevée
- Souvent au détriment de fonctions de communication (coût système plus élevé)

Clés Architecturales

Au fur et à mesure du temps, les architectures de DSP ont évoluées. Les clés architecturales que l'on rencontre maintenant le plus souvent sont les suivantes :

- Chercher à augmenter la vitesse d'horloge (Technologie, Astuces architecturales)
- Notions de Parallélisme et Pipeline
- Unités arithmétiques spécifiques (MAC, barrel shifter, FPU,…)
- Communications dédiées (bloc de données 1D et 2D…)
- Multiplication de mémoires et registres embarqués
- Unités arithmétiques et contrôle câblés

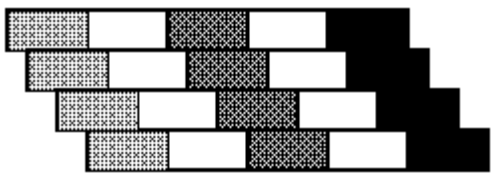
Architecture RISC et Superpipeline

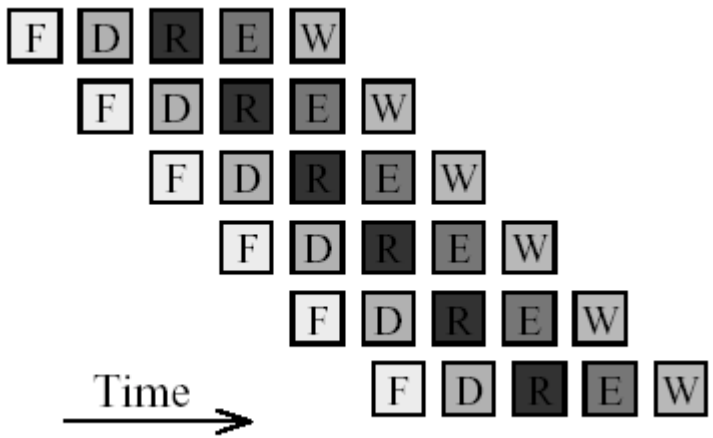
Reduced Instruction Set Computer

- Jeu d'instruction réduit, modes d'adressages simples
- Taille et encodage des instructions uniforme
- Architecture load/store
- Grand nombre de registres généraux
- Latence des instructions
- 4 ou 5 niveaux de Pipeline

Superpipeline

- Équivaut à une décomposition de l'opérateur en plus d'étapes
- Temps de cycle plus faible : très grandes fréquences d'horloge

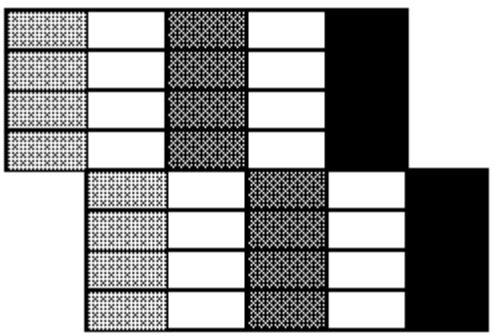


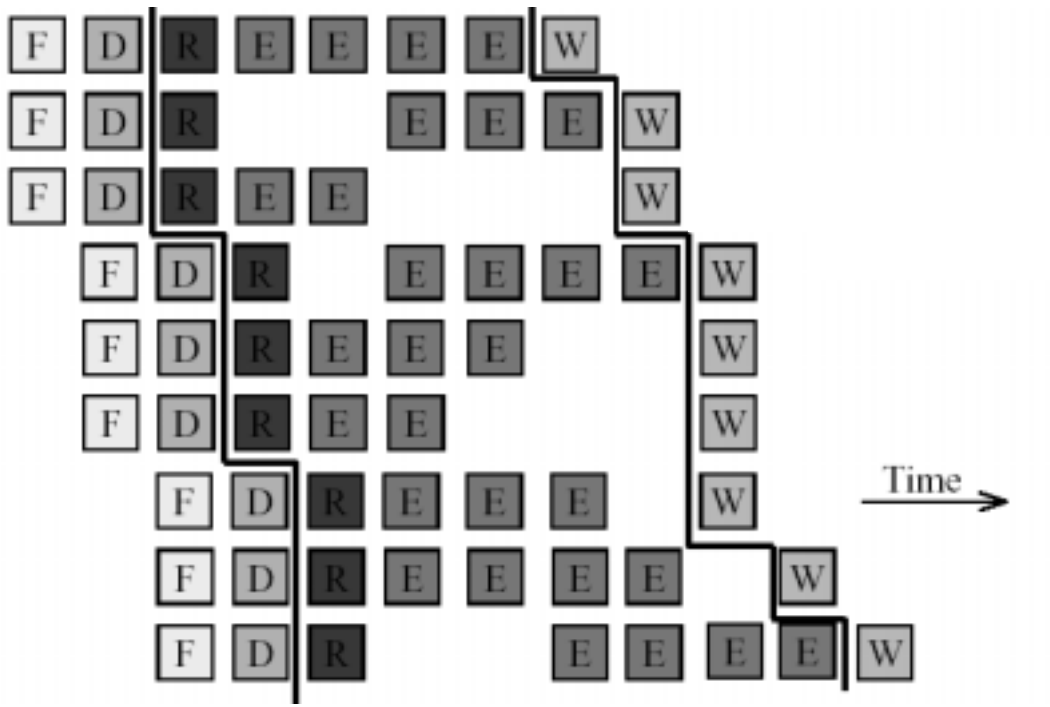


Architecture Superscalaire

Superscalaire

Le Superscalaire désigne un type de processeur permettent d'exécuter plusieurs instructions en parallèle en un seul cycle d'horloge. Aujourd'hui la plupart des machines superscalaires peuvent gérer 2 à 6 instructions en parallèle.





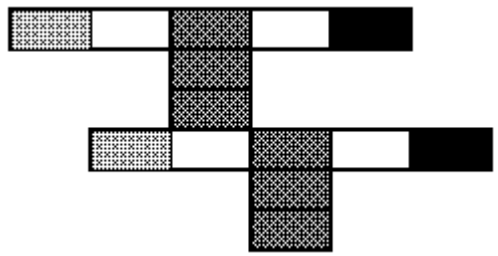
Architecture VLIW

Very Long Instruction Word

Le VLIW décrit une philosophie de jeux d'instructions dans le lequel le compilateur empaquète un certain nombre d'opérations simples et non-interdépendants dans le même mot d'instruction. Une fois ce mot d'instruction fournit au processeur, ces simples opérations sont départagés et expédié dans des unités indépendantes d'exécution.

Caractéristiques du VLIW

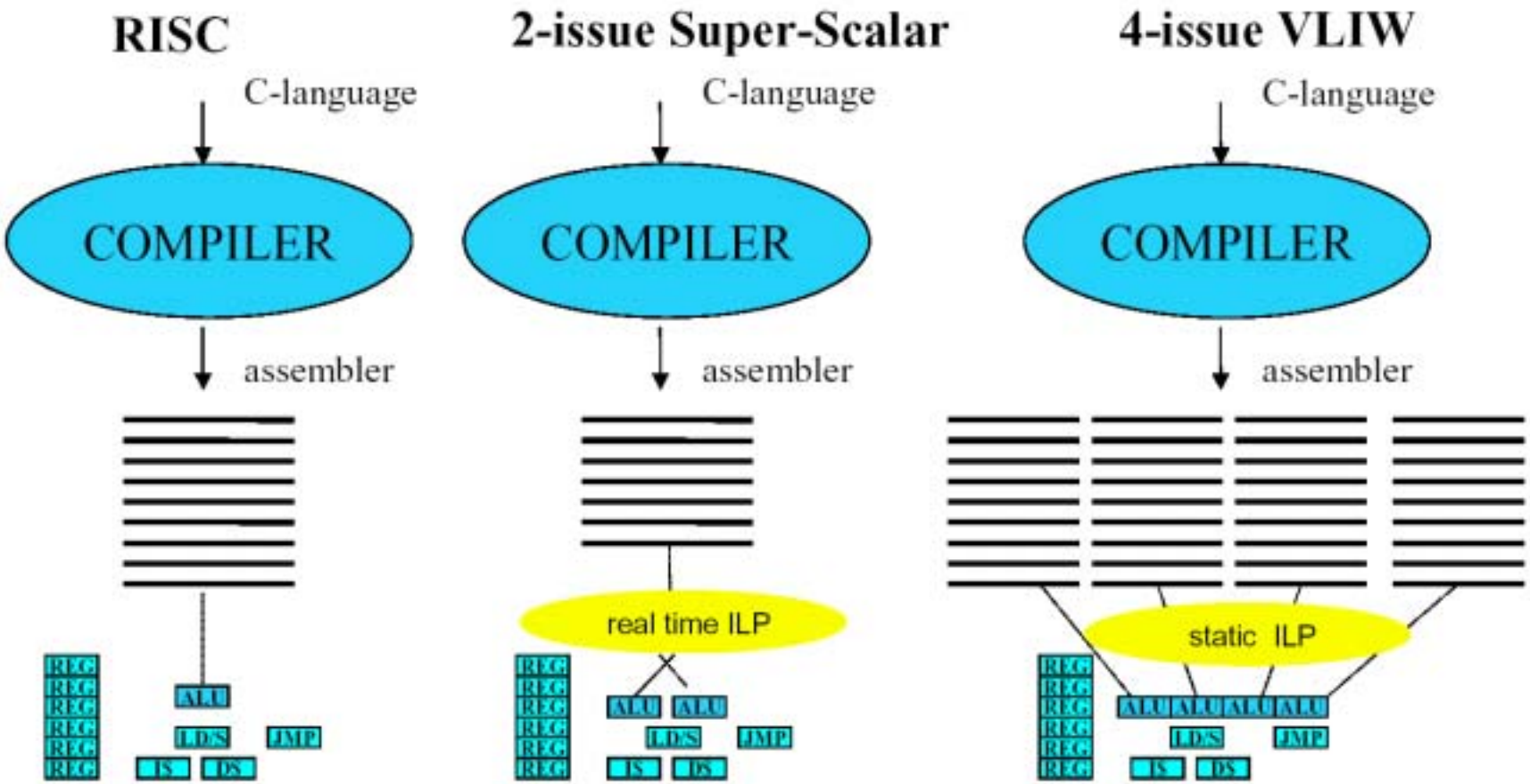
- 1 unité de contrôle, 1 instruction très longue par cycle
- Formats fixes de l'instruction (champs indépendants)
- Ordonnancement statique à la compilation
- Latence courte et déterministe pour chaque opération
- Opérations pipelinées
- Architecture load/store



Regain d'Intérêt pour le VLIW

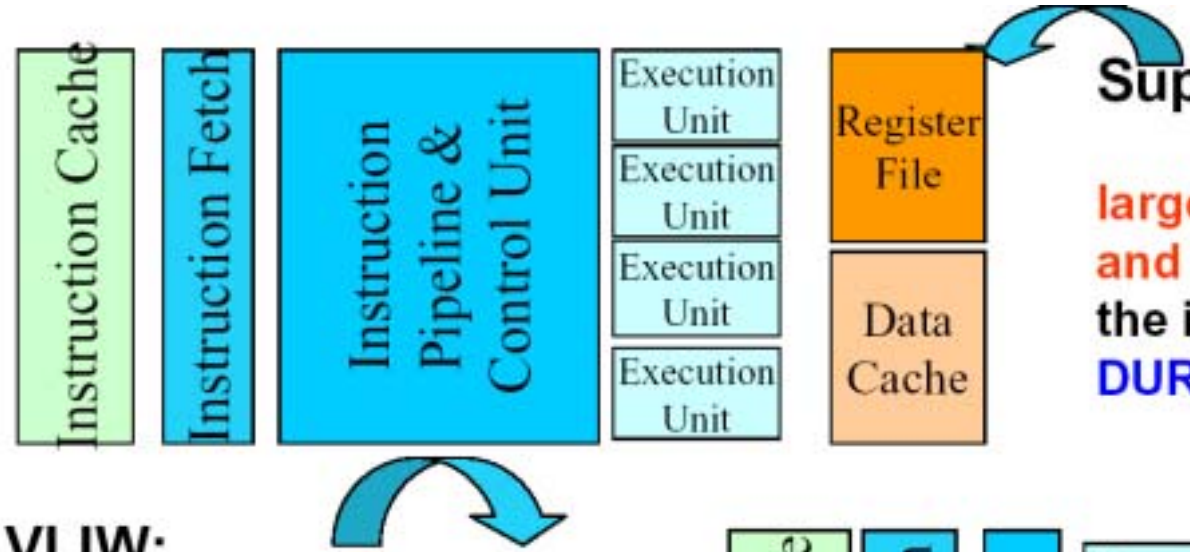
- Densité d'intégration accrue : surface de silicium difficile à exploiter avec le superpipeline/supersacalaire
- Complexité exponentielle du contrôle dynamique de grands pipeline et/ou d'un parallélisme instruction important
- Croissance des applications de type DSP

Architecture VLIW vs Superscalar



ILP : Instructions Level Parallelism

Architecture VLIW vs Superscalar

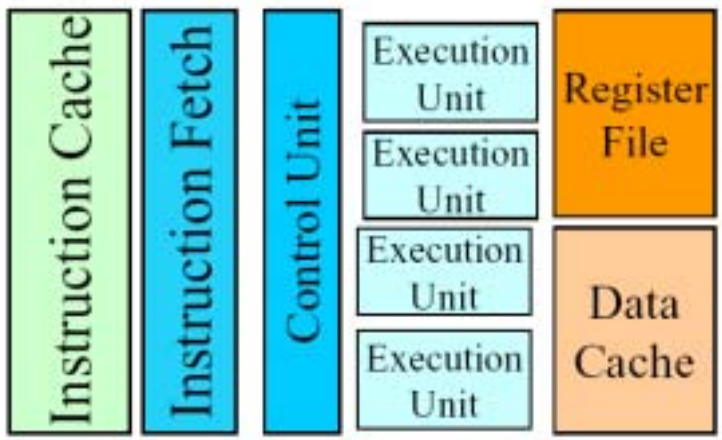


Super scalar RISC:

large instruction pipeline and control unit, to handle the instruction scheduling DURING program run time.

VLIW:

very small control unit, since the compiler schedules the instructions efficiently BEFORE program runs.



Architecture VLIW vs Superscalar

Technology	Performance/ Cost	Time to market	Time to change functionality
ASIC	Very High	Very Long	Impossible: Redesign
DSP	High	Long	Long
Custom VLIW	High	Short	Short
RISC	Low-Medium	Very Short	Very Short

Conclusion

DSP à base de VLIW considéré comme le nec plus ultra…

Exemples de Processeurs « Multimédia »

- CISC + arithmétique + VLIW ? : Intel MMX
- Superscalaire : SGS Thomson Chameleon
- VLIW : Philips TRIMEDIA
- RISC + VLIW : Apple PowerPC/TRIMEDIA ainsi que le DSP ST210 aux caractéristiques ci-dessous

- VLIW with 4 operations per clock cycle.
- 64 general purpose 32-bit registers.
- 1 load/store per cycle.
- 2 multiplies (16x16 -> 32) per cycle.
- 32k I-cache, 32k D-cache.
- worst case: 250 MHz in HCMOS-8 (0.18 micron).

- Total core size (cache included): 10. 8 mm^2
  - CPU: 3 mm^2, caches: 7 mm^2, peripherals: 0.8 mm^2

- ▲ Video
  - MPEG2 loop encoder: 200MHz @ 15Mbps 25 PAL frames/s
  - MPEG2 whole decoder: 250MHz @ 15Mbps 25 PAL frames/s
  - MPEG4 SP L3 decoder: 12MHz @ 64Kbps 30 QCIF frames/s
  - OpenDivX decoder: 15MHz @ 64Kbps 30 QCIF frames/s
- ▲ Audio
  - MPEG1 layer 2 encoder, 256Kbps stereo:
    - 28MHz @ 32KHz s.r. and 37MHz @ 44.1KHz s.r.
  - Mpeg1 layer 2 Audio decoder: 24MHz @ 256Kbps stereo at 44.1KHz
  - Dolby AC3 encoder stereo: < 40MHz
  - Dolby AC3 decoder 6 Ch: < 60MHz
  - MP3 decoder: < 55MHz

Plan

- **Chapitre III Architecture et Conception d'Opérateurs Numériques**
  - Représentation en Virgule Fixe
  - Représentation en Virgule Flottante
  - Influence de la représentation en Virgule Fixe – Effets de Quantification Finie
  - Opérations d'addition et de multiplication en Virgule Fixe
  - Travaux Dirigés #1
  - Pipeline et Parallélisme
  - Travaux Dirigés #2

Représentation des Nombres

Représentation Binaire

La représentation binaire est utilisée pour coder les nombres dans la plupart des systèmes numériques. Le nombre et alors représenté par un ensemble de 0 et de 1 appelés bits, avec une virgule binaire séparant partie entière et partie fractionnaire. En général un nombre binaire N constitué de B bits « entiers » et b bits « fractionnaires » se présente sous la forme suivante :

$$N = a_{B-1}\, a_{B-2} \ldots a_1\, a_0\ \Delta\ a_{-1}\, a_{-2} \ldots a_{-b}$$
$$= \sum_{i=-b}^{B-1} a_i \times 2^i$$

Où Δ désigne la virgule binaire et chaque ai un bit de N.

Représentation en Virgule Fixe

Entiers Positifs

$$N = \sum_{i=0}^{B-1} b_i \times 2^i$$

Entiers relatifs

Binaire Signé

$$N = (-1)^s \sum_{i=0}^{B-2} b_i \times 2^i$$

Complément à 2

$$N = -b_p 2^{B-1} + \sum_{i=0}^{B-2} b_i \times 2^i$$

Changement de Signe en Complément à 2

L'inverse –N du nombre codé en complément à 2 s'écrit comme suit :

$$-N = -\overline{b_{B-1}}\,2^{B-1} + \sum_{i=0}^{B-2} \overline{b_i}\,2^i + 1$$

Ce qui revient à écrire :

$$-N = \overline{N} + 1$$

Pour déterminer le complément à 2 du nombre N, il suffit de complémenter tous ses bits, puis d'ajouter 1 au nombre ainsi obtenu.

Extension de Signe – Codage en Complément à 2

Passage d'une représentation de P bits à une représentation P+1 bits

$$N = -b_{B-1}2^{B-1} + \sum_{i=0}^{B-2} b_i 2^i = -b_{B-1}2^B + \sum_{i=0}^{B-1} b_i 2^i \quad car \quad -b_p = -2b_p + b_p$$

On peut donc étendre à loisir le signe d'un nombre en complément à 2 sans changer sa valeur.

Il est souvent nécessaire en arithmétique d'étendre la représentation d'un entier relatif sur un nombre de bits supérieurs. Pour additionner 2 nombres par exemple, il faut au préalable les coder sur un même nombre de bits.

Quelques Examples

Addition (1):

```
    0 1 1 1
+     0 1 1
--------------
```

Addition (3):

```
    1 1 0 0
+   0 0 1 1
--------------
```

Addition(2):

```
    0 1 1 1
+   0 1 1 0
+   0 1 1 0 0
-----------------
```

Addition (5): Overflow intermédiaire

```
+ 0 0 1 1 1 0
+ 0 0 1 1 1 1
+ 0 0 1 0 1 1
+ 1 1 0 1 1 0
+ 1 1 0 1 1 1
-------------------
```

Addition(4):

```
    1 1 0 1
+   0 1 1 1
+   0 1 1 1
--------------
```

Extension de Signe – Codage en Complément à 2

Passage d'une représentation de P bits à une représentation P+1 bits

$$N = -b_{B-1}2^{B-1} + \sum_{i=0}^{B-2} b_i 2^i = -b_{B-1}2^B + \sum_{i=0}^{B-1} b_i 2^i \quad car \quad -b_p = -2b_p + b_p$$

On peut donc étendre à loisir le signe d'un nombre en complément à 2 sans changer sa valeur.

Il est souvent nécessaire en arithmétique d'étendre la représentation d'un entier relatif sur un nombre de bits supérieurs. Pour additionner 2 nombres par exemple, il faut au préalable les coder sur un même nombre de bits.

Représentation en Virgule Flottante

Dans la représentation normalisée de la virgule flottante, un nombre positif η est représenté en utilisant 2 paramètres la mantisse et l'exposant E sous la forme :

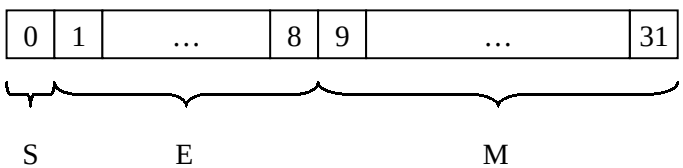
$$\eta = M \times 2^{E}$$

Où la mantisse M est une fraction binaire comprise dans l'intervalle :

$$\frac{1}{2} \leq M < 1$$

Les nombres en virgule flottante sont stockés dans un registre en assignant B_E bits au registre de l'exposant et les B_M bits restants à la mantisse.

Les formats les plus largement utilisés en virgule flottante pour les représentations 32 bits et 64 bits sont donnés par la norme ANSI/IEEE 754-1985. Par exemple un registre de 32 bits est divisé en champs de la sorte:



$$\eta = (-1)^{S} \times 2^{E-127} \times M \qquad avec \;\; 0 \leq M < 1$$

Virgule Fixe – Effets de Quantification Finie

La quantification est la base de la numérisation des signaux. Malheureusement, celle-ci engendre des erreurs non-linéaires qu'il est nécessaire de maîtriser.

Par exemple, la quantification des coefficients d'un filtre a pour principal effet de modifier la réponse d'un filtre, voire de le rendre instable.

A cela, il faut ajouter le problème dû à l'implémentation numérique d'opérations arithmétiques : la longueur finie des registres de mémoire et les résultats des opérations arithmétiques en sont les principales causes.

En effet, prenons un exemple en virgule fixe : le produit de 2 nombres de b bits donnent un nombre de 2b bits. Or afin qu'il puisse être contenu dans les registres de mémoire et utilisé par la suite, c nombre de 2b bits est lui-même tronqué et quantifié à b bits.

L'analyse des effets de la quantification sur les performances d'un filtre numérique dépend en pratique de plusieurs critères :

- Le format des nombres (virgule fixe ou virgule flottante);
- Le type de représentation des nombres négatifs;
- Le type de quantification (troncature ou arrondi);
- Enfin et surtout la taille en bits des nombres considérés.

Typiquement, les 2 premiers points mentionnés ci-dessus sont fixés. Pour les 2 derniers points, seule une étude et des simulations approfondies et systématiques peuvent donner des éléments de réponse.

Virgule Fixe – Dimension des Opérandes

Dans la représentation en virgule fixe, il est absolument primordial de dimensionner et donc d'optimiser au mieux la taille des nombres et opérandes.

Il est en effet d'autant plus judicieux de réduire la taille des opérandes de 12 à 9 bits que les performances sont pratiquement identiques. Les gains en taille et consommations sont loin d'être négligeables.

Conclusion

- On en déduit l'intérêt du **Fast Prototyping**: valider en temps réel les résultats du simulateur;
- Sur le prototype HiperLAN/2 de CRM Paris, la différence en terme de ressources matérielles utilisées et donc en terme de surface entre les formats 12 et 9 bits était de l'ordre de 30% à performance identique! Soit pratiquement **30% d'autonomie supplémentaire pour des systèmes embarqués** !
- Eternel compromis coût/ performance. Un représentation sur 8 bits ferait chuter la taille du circuit mais avec des performances fortement amoindrie : **circuit bas de gamme et bas-coût** mais avec une autonomie accrue.

Addition à 2 ou Plusieurs Opérandes

Dynamique de la somme de 2 opérandes de B bits

La somme de 2 opérandes de B bits donne un opérande de B+1 bits.

Dynamique de la somme de N opérandes de B bits

Un opérande codé sur B bits a une valeur comprise entre 0 et 2^B-1. Sa dynamique est 2^B. La valeur maximale atteinte par la somme de N opérandes B bits est donc égale à :

$$N(2^B - 1) = 2^{B+\log_2 N} - N$$

Le logarithme en base 2 de cette valeur donne le nombre de bits nécessaires au codage du résultat. En négligeant le terme N, on obtient le nombre de bits égal à :

$B + \log_2 N$

Multiplication

Multiplication à 2 opérandes de N bits

La multiplication de 2 opérandes de N bits donnent un opérande de 2N bits.

Exemple d'une multiplication de 2 mots signés A et B de 4 bits codés en complément à 2.

					a_3	a_2	a_1	a_0
				$\times$	b_3	b_2	b_1	b_0
$2^0.b_0.A$	a_3b_0	a_3b_0	a_3b_0	a_3b_0	a_3b_0	a_2b_0	a_1b_0	a_0b_0
$2^1.b_1.A$	a_3b_1	a_3b_1	a_3b_1	a_3b_1	a_2b_1	a_1b_1	a_0b_1	.
$2^2.b_2.A$	a_3b_2	a_3b_2	a_3b_2	a_2b_2	a_1b_2	a_0b_2	.	.
$-2^3.b_3.A$	$\overline{a_3b_3}$	$\overline{a_3b_3}$	$\overline{a_2b_3}$	$\overline{a_1b_3}$	$\overline{a_0b_3}$	.	.	.
$=2^3.b_3.(-A)$					$+\,b_3$			
$=2^3.b_3.(\overline{A}+1)$								

Résultats sur 8 bits

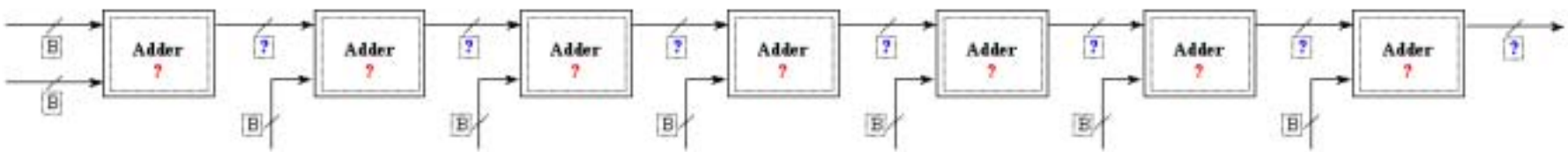
Multiplication d'un opérande par une constante

La multiplication d'un opérande par 2 équivaut à un décalage vers la gauche du mot dans le registre.
La division d'un opérande par 2 équivaut à un décalage vers la droite du mot dans le registre.
La multiplication d'un opérande par une constante équivaut à des décalages et des additions.

TD #1 – Exercice 1

Additionneur à N opérandes – Structure Linéaire

La première structure étudiée d'un additionneur à N opérandes est linéaire. Chaque opérateur est un additionneur à propagation de retenue (ripple carry adder). Le premier effectue la somme de deux opérandes B bits, le deuxième additionne le troisième opérande à la somme partielle obtenue, etc… N-1 étages d'additionneurs sont nécessaires pour sommer N opérandes.



Exercice 1 - Dynamique

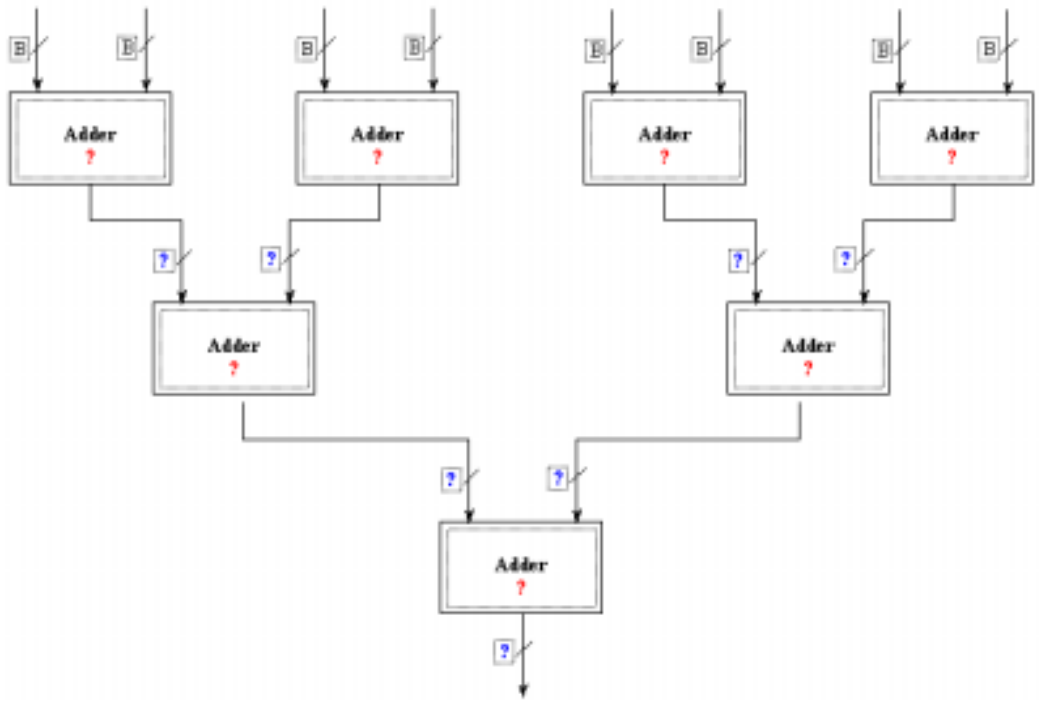
On considère le système d'additionneurs à structure linéaire ci-dessus à 8 opérandes.

1) Déterminer pour chaque additionneur son type, c'est-à-dire s'il s'agit d'un additionneur à B bits ou B+1, etc…
2) Déterminer en sortie de chaque additionneur le nombre de bits nécessaires au codage du résultat.
3) Déterminer la complexité de cette structure en fonction du nombre d'additionneurs, B, B+1, B+2 et B+3 bits.

TD #1 – Exercice 2

Additionneur à N opérandes – Structure Arborescente

La deuxième structure étudiée d'un additionneur à N opérandes est arborescente. Chaque opérateur est le même que précédemment à savoir un additionneur à propagation de retenue (ripple carry adder). Le premier étage somme les opérandes B bits deux à deux. Les sommes partielles sont ensuite sommées deux à deux jusqu'à réduction complète du résultat.



Exercice 2 - Dynamique

1) Déterminer le nombre d'étages nécessaires pour sommer N opérandes.

On considère le système d'additionneurs à structure linéaire ci-dessus à N=8 opérandes.

2) Déterminer pour chaque additionneur son type, c'est-à-dire s'il s'agit d'un additionneur à B ou B+1 bits, etc…

3) Déterminer en sortie de chaque additionneur le nombre de bits nécessaires au codage du résultat.

4) Déterminer la complexité de cette structure en fonction du nombre d'additionneurs, B, B+1, B+2 et B+3 bits.

TD #1 – Exercice 3

Exercice 3 – multiplication en complément à deux

$$\begin{array}{rrrrrr} & 0 & 1 & 0 & 1 & (5) \\ \times & 0 & 1 & 1 & 0 & (6) \\ \hline \end{array}$$

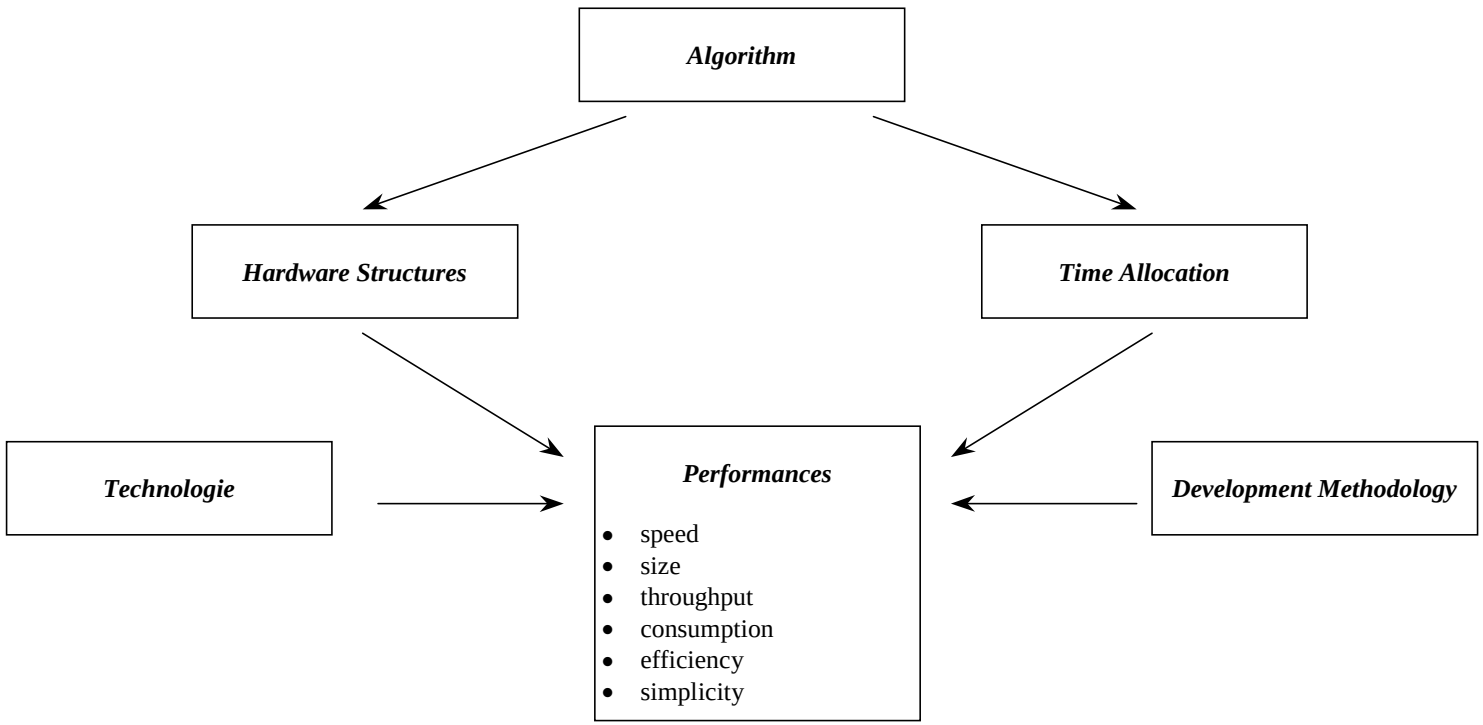
$$\begin{array}{rrrrrr} & 1 & 1 & 0 & 1 & (-3) \\ \times & 0 & 1 & 0 & 1 & (5) \\ \hline \end{array}$$

$$\begin{array}{rrrrrr} & 1 & 0 & 1 & 1 & (-5) \\ \times & 1 & 0 & 1 & 0 & (-6) \\ \hline \end{array}$$

Base de l'Architecture

Architecture de Système Numérique : l'architecte doit intégrer ***la dimension du temps***.

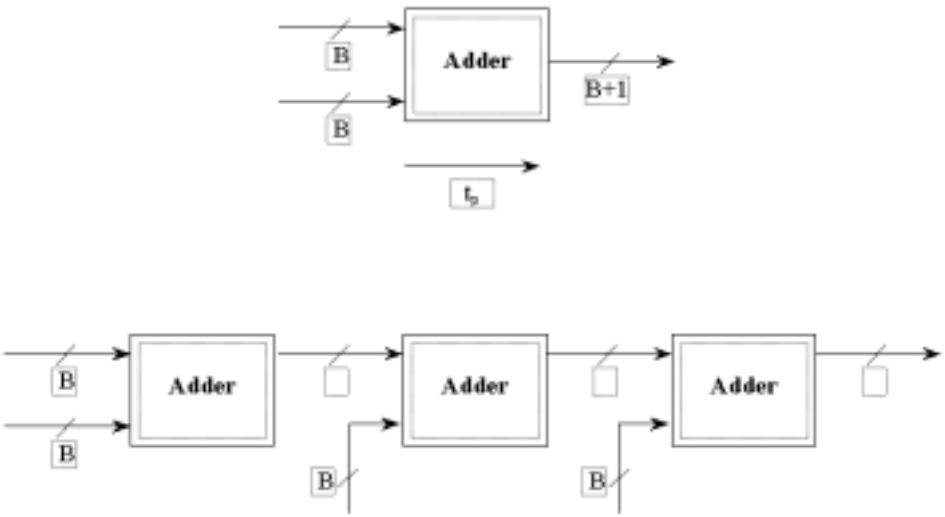
Chaque resource matérielle peut être utilisée plusieurs fois et l'allocation temporelle des ressources matérielles choisies peut s'effectuer de plusieurs manières (parallélisme, pipeline).



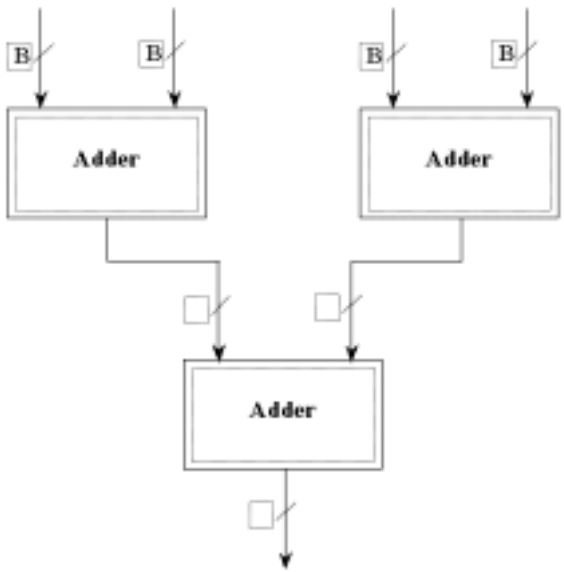
Architectures Série et Parallélisme

Reprenons les exemples d'additionneur à plusieurs opérandes des exercices 1 et 2 du TD #1

Si on considère que chaque adder a un temps de propagation t_p, alors on obtient un temps de propagation de 2 t_p, et 3 t_p, respectivement des architectures parallèle et série à surface pratiquement équivalente.



Calcul Série

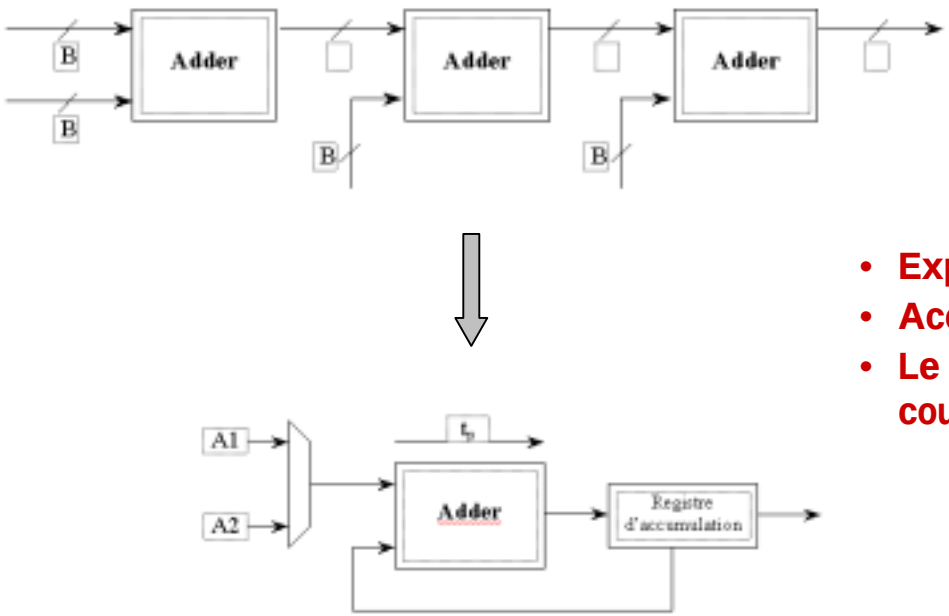


Parallélisme

Architecture Série/Parallèle

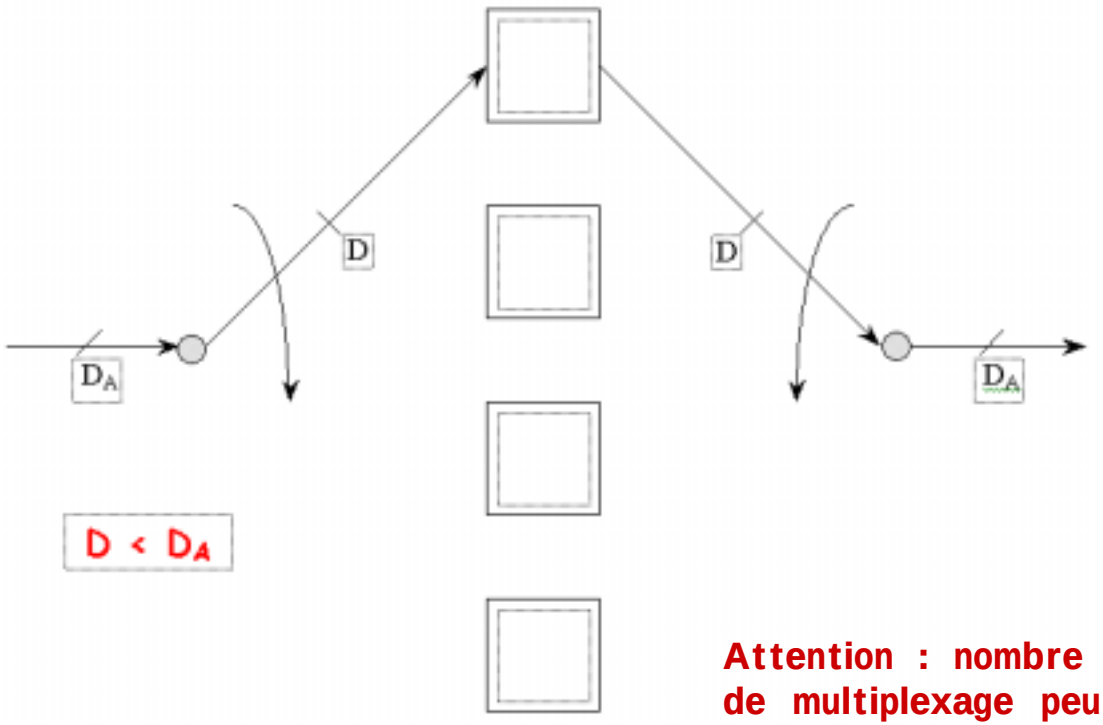
Diminution de la surface à performance fixée

Si on considère l'exemple de la structure série d'un additionneur à 3 opérandes, on peut réduire pratiquement sa surface de 3 fois moins par l'intermédiaire d'une architecture série-parallèle



- **Exploitation séquentielle d'un opérande (horloge)**
- **Accumulations successives de résultats partiels**
- **Le résultat est disponible en parallèle au dernier coup d'horloge**

Démultiplexage Temporel



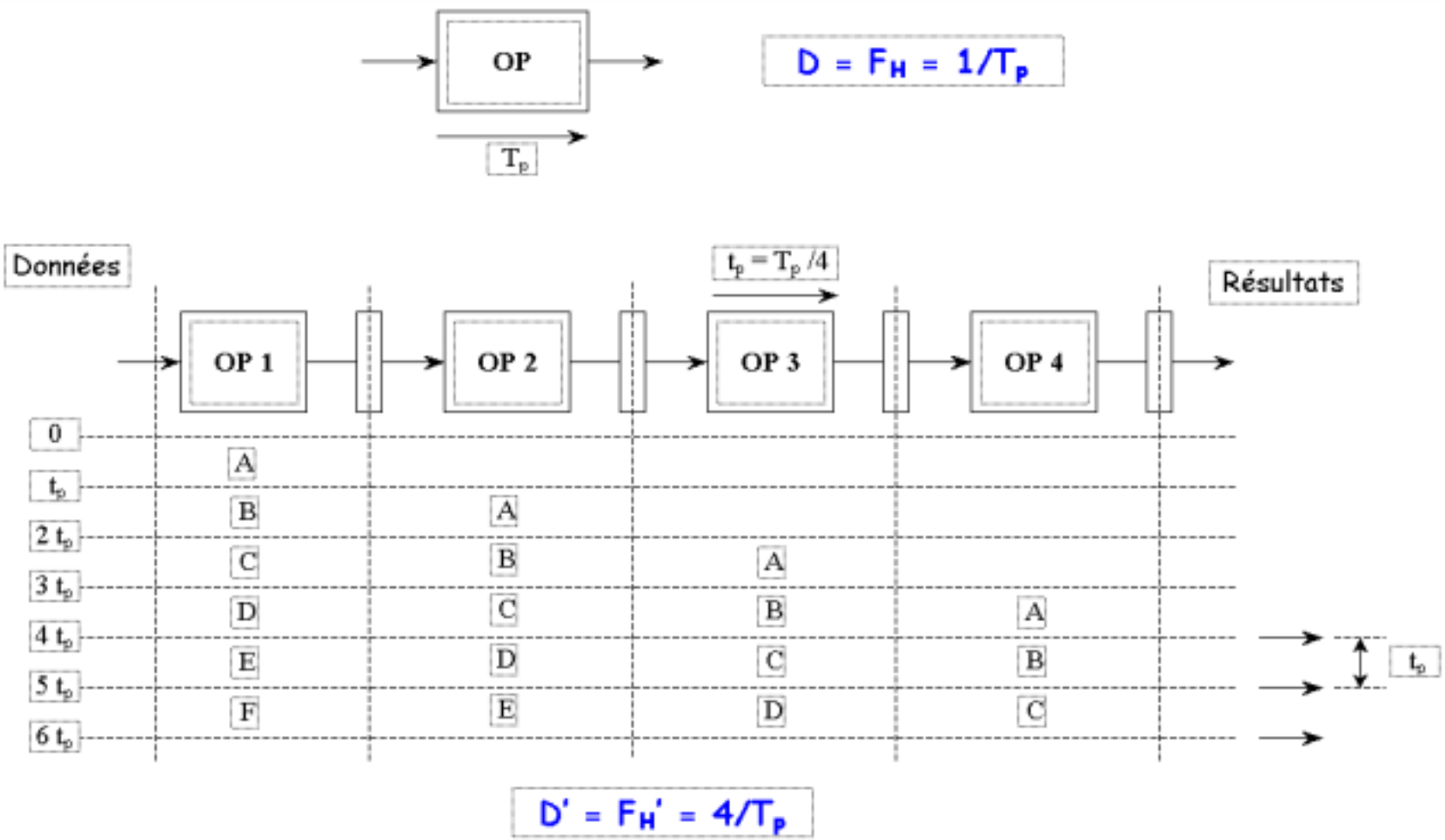
Applications D_A **Ressources D**

- On dispose de n ressources de débit D
- Débit total : $nD = D_A$

Attention : nombre de ressources n limités. La procédure de multiplexage peut s'avérer très lourd en termes de ressources matérielles.

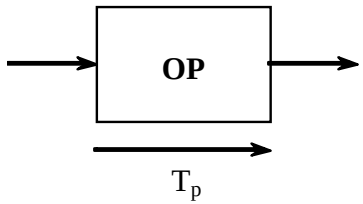
Pipeline : Augmenter Le Débit

Principe

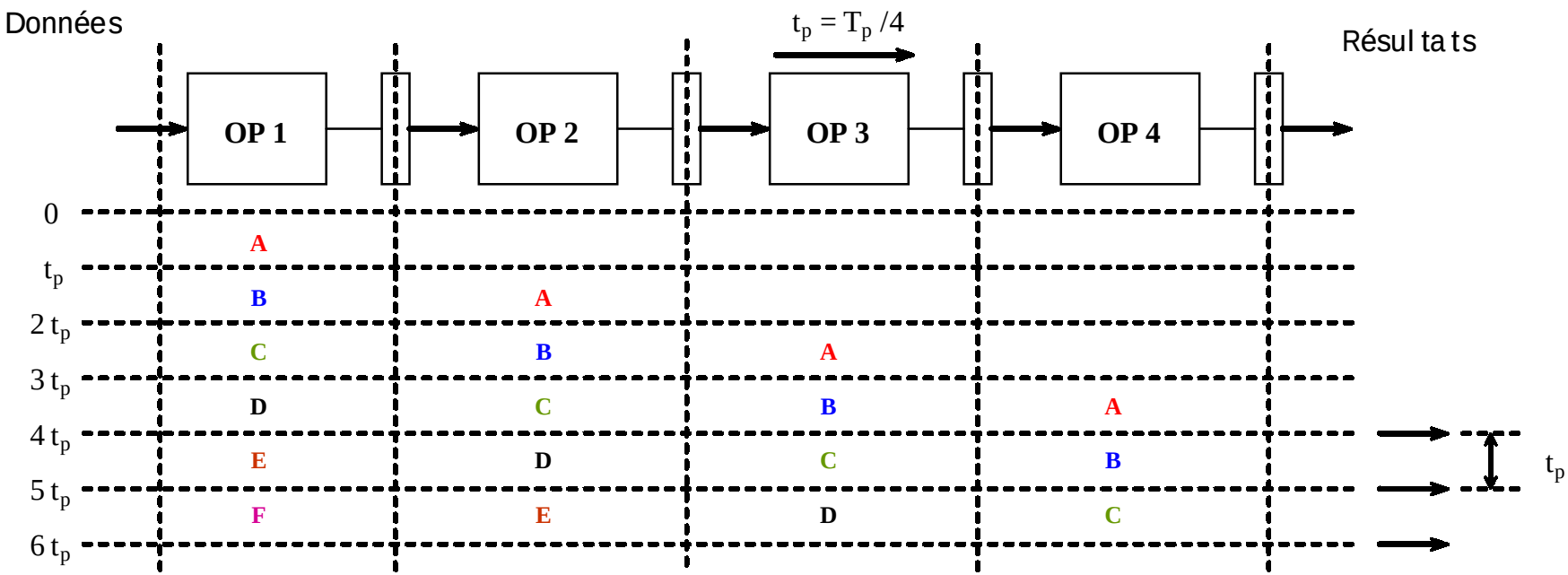


Pipeline : Augmenter Le Débit

Principe



$D = F_H = 1/T_p$



$D' = F_H' = 4/T_p$

TD #2 – Exercice 1

Tout au long de cet exercice, on négligera les temps de set-up et de propagation des registres

On dispose d'un opérateur à une entrée E et une sortie S, réalisant une opération notée OP, figure 1. Son temps de propagation est de 40 ns. La fréquence maximale de fonctionnement de cet opérateur est donc de :

Fs = 25 MHz.

On admet la possibilité d'insérer une barrière de pipeline dans l'opérateur OP, de façon à réduire le chemin critique à 20 ns. La fréquence maximale de fonctionnement de l'opérateur modifié, noté OP*, est donc de :

Fs* = 50 MHz.

L'application que l'on souhaite implanter est illustrée par la figure 2, où X est un signal échantillonné à la fréquence Fe (x_n est un échantillon de X). L'opération notée OP sur la figure 2 est celle réalisée par l'opérateur décrit précédemment.

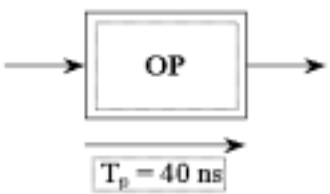


Figure 1

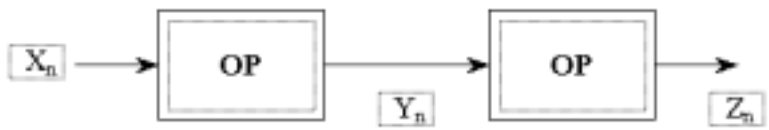


Figure 2

TD #2 – Exercice 1

Questions

1) Exprimer en Méga Opérations Par Seconde (MOPS) la puissance de calcul nécessaire à l'application dans le cas où la fréquence d'entrée Fe vaut :
   a) Fe = 12,5 Mhz
   b) Fe = 25 MHz
   c) Fe = 50 MHz.
2) Pour chacun des cas a, b et c, donner deux architectures utilisant les opérateurs OP ou OP*, et si besoin des registres et des multiplexeurs, permettant d'atteindre la puissance de calcul nécessaire à l'application. On représentera chaque possibilité à l'aide d'un schéma succinct.
3) On considère à présent le détail de l'opérateur OP. Sa structure est donnée dans la figure 3. On ignorera le contenu et la fonctionnalité des sous-opérateurs représentés en grisé. On considérera leur temps de propagation (entre n'importe quelle entrée et n'importe quelle sortie) égal à 20 ns.
   a) Tracer le chemin critique de l'opérateur et vérifier que sa fréquence maximale de fonctionnement est de 25 MHz.
   b) On souhaite réaliser l'opérateur modifié OP*, tel que défini en début d'énoncé, fonctionnant à fréquence double. On place pour cela des registres sur les sorties des sous-opérateurs. Quelles sont les précautions à prendre pour garder la fonctionnalité de l'opérateur OP intacte? Quel est le coût matériel de cette modification?
4) Connaissant à présent la fonctionnalité de l'opérateur, quelles sont parmi les architectures recensées dans la question 2, celles qui restent valides?

TD #2 – Exercice 1

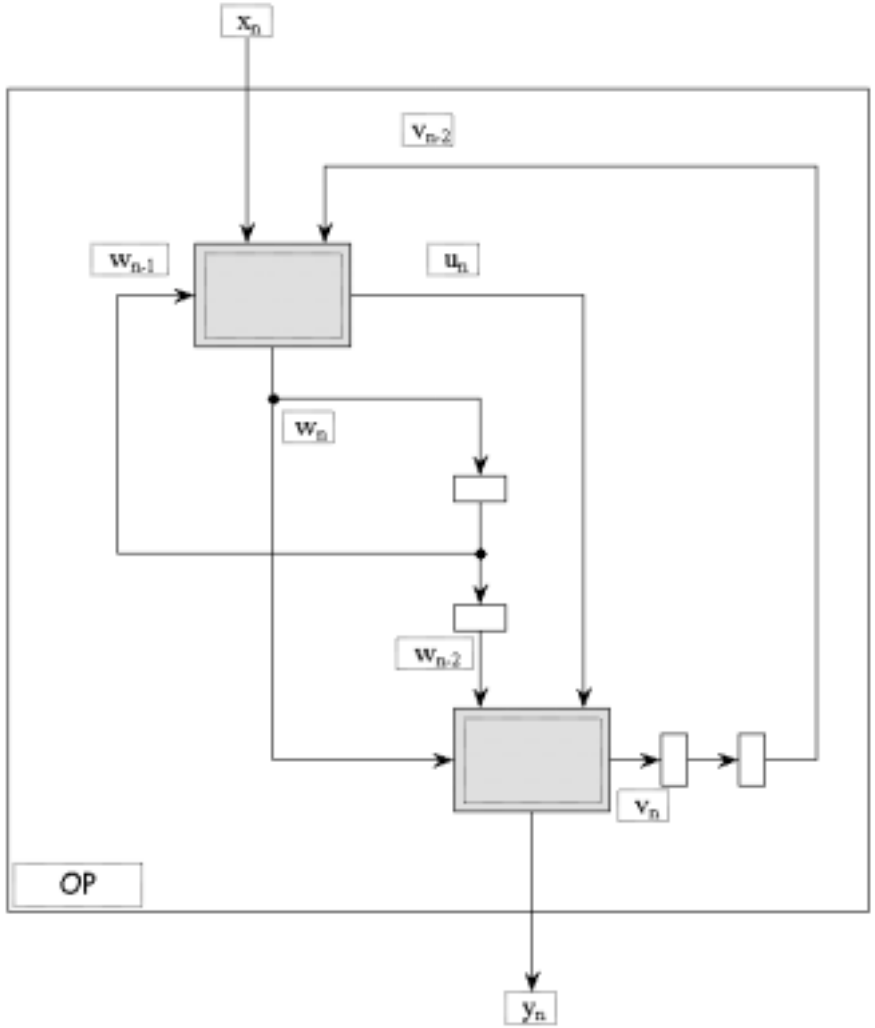


Figure 3

TD #2 – Exercice 2

Faire le schéma haut niveau (sans rentrer dans le détail) d'un opérateur réalisant le calcul de :

$$S = |A - B|$$

A et B sont 2 nombres positifs codés sur n bits

TD #2 – Exercice 3

On considère le calcul exprimé par les équations suivantes :

$y_n = OP2\ (x_n, w_{n-2})$

$w_n = OP2\ [\ OP1(w_{n-2}), OP1(z_{n-2})\ ]$

$z_n = OP2\ [\ OP1(y_n)\ , OP1(w_n)\ ]$

L'opérateur OP2 est commutatif : OP2 [A , B] = OP2 [B , A].

Ce calcul est appliqué à un signal échantillonné à une fréquence notée Fe, et dont les échantillons successifs sont notés x_n.

Questions

1) On cherche à implanter cette application à l'aide de l'architecture donnée par la figure 4. Les opérateurs OP1 et OP2 ont même temps de propagation égal à 25 ns.
   a) A quelle fréquence maximale peut fonctionner cet opérateur (on négligera dans ce qui suit les temps de propagation, de set-up et de hold des registres)?
   b) Comment augmenter ce débit pour traiter en temps réel un signal échantillonné à Fe = 20 MHz?
   c) Peut-on l'augmenter encore pour traiter en temps réel un signal échantillonné à Fe = 40 MHz?
2) On suppose maintenant le fréquence d'échantillonnage Fe du signal x(t) égale à 10 MHz. On cherche à développer une architecture série, constituée d'un seul opérateur OP1 et d'un seul opérateur OP2, permettant de traiter x(t) en temps réel. Proposer un séquencement possible des calculs.

TD #2 – Exercice 3

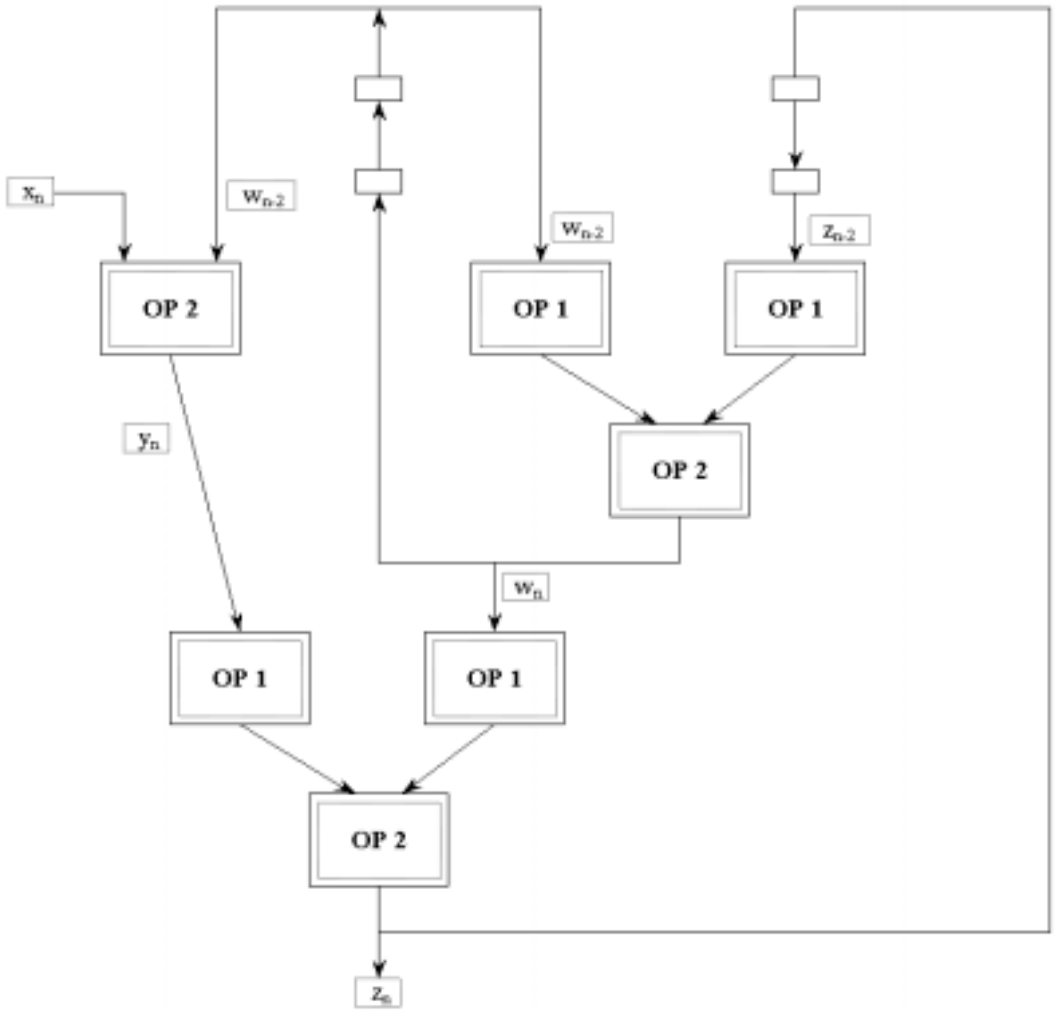


Figure 4

Plan

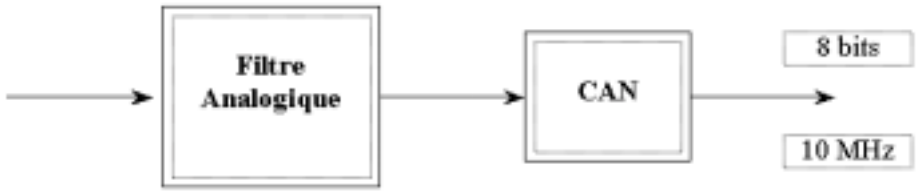
- **Chapitre IV Conception d'un filtre RII – ASIC et DSP**
  - Méthodologie de spécification de filtre
  - Filtre RII à correction de phase
  - Architecture et Conception d'un filtre RII – Approche Globale
  - Travaux Dirigés #3
  - Architecture et Conception d'un filtre RII – Approche Spécifique sur DSP
  - Travaux Dirigés #4

Synthèse de Filtre Numérique

Cas pratique – Spécification d'un filtre

Choix du Type de Filtre : RIF ou RII
Gabarit du Filtre
Synthèse des Coefficients du Filtre
Optimisation des Coefficients pour Implémentation

Étude de Cas : Filtre Numérique RII

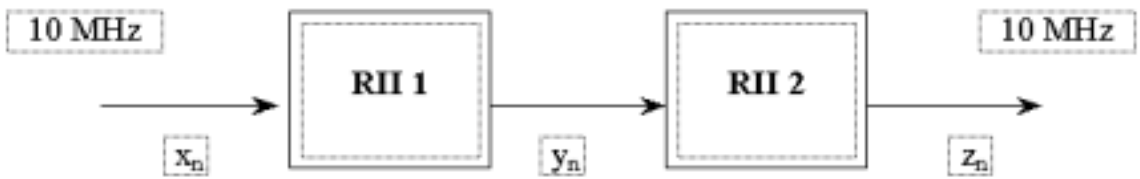


Filtre Numérique RII passe-tout à correction de phase

Problème Filtre Analogique à phase non-linéaire

Solution Correction de ce déphasage non linéaire à l'aide d'un filtre numérique passe-tout à correction de phase.

Fe = 10 MHz



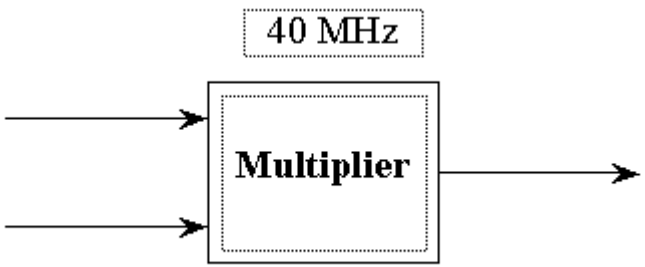
$$H_1(z) = \frac{z^{-2} + b\,z^{-1} + a}{a\,z^{-2} + b\,z^{-1} + 1} \qquad H_2(z) = \frac{z^{-2} + d\,z^{-1} + c}{c\,z^{-2} + d\,z^{-1} + 1}$$

Études Préliminaires – Approche ASIC

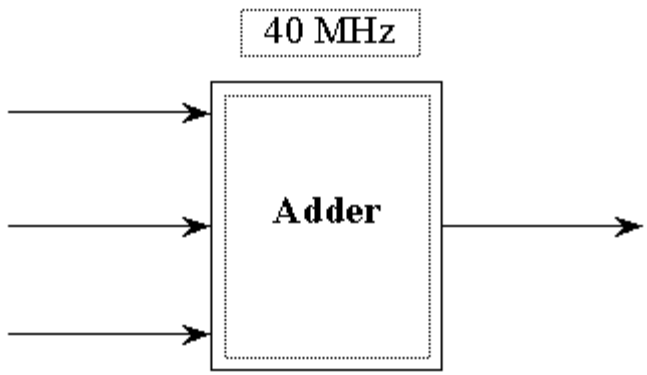
Hypothèses

On suppose que l'on dispose des opérateurs suivants fonctionnant à **40 MHz** :

1 multiplieur



1 additionneur/soustracteur (3 entrées – 1 sortie)



TD #3

Questions

1) Représenter la structure forme directe de $H_1(z)$. Exprimer le nombre d'opérateurs d'addition et de multiplications nécessaires pour réaliser cette structure ainsi que le nombre de registres de décalage z^{-1}. En déduire la puissance de calcul nécessaire de $H_1(z)$ x $H_2(z)$ (en Méga additions par seconde et Méga multiplications par seconde).
2) Optimisation : proposer une autre structure de $H_1(z)$ avec pour objectif de réduire le nombre d'opérateurs d'addition et de multiplication. Exprimer le nombre d'opérateurs d'addition et de multiplication ainsi que le nombre de registres à décalage nécessaires. En déduire la puissance de calcul de cette nouvelle structure $H_1(z)$ x $H_2(z)$.

A partir de maintenant, on ne considèrera que la structure proposée dans 2.

3) En déduire le nombre minimum d'opérateurs d'addition et de multiplication pour réaliser cette structure.
4) Il s'agit désormais d'étudier la durée de vie des variables. Identifier les différentes variables de $H_1(z)$ puis exprimer le nombre de registres nécessaires pour les sauvegarder.
5) Étudier et proposer un séquencement des données et variables.
6) En déduire une architecture de la structure $H_1(z)$ x $H_2(z)$.
7) On considère toujours les opérateurs de multiplication et d'addition à 40 MHz introduits précédemment. Exprimer la complexité en termes de nombres de multiplieurs et d'additionneurs nécessaires pour réaliser ce même filtre $H_1(z)$ x $H_2(z)$ mais aux fréquences d'échantillonnage suivantes :
   a) 20 MHz
   b) 40 MHz
   c) 60 MHz

Études Préliminaires – Approche DSP

Rappel sur la Structure du DSP TMS320VC5402

• Fréquence en MHz	100
• MIPS	100
• Temps Cycle (ns)	10
• Espace mémoire Données/Programme (Mots)	64K/1M
• RAM (Mots)	16K
• ROM (Mots)	4K

- Architecture Avancée Multi-bus avec 3 bus différents de 16 bits pour mémoire Données et 1 bus pour la mémoire Programme

- ALU de 40 bits incluant 1 barrel shifter de 40 bits et 2 accumulateurs de 40 bits.
- Multiplieur 17x17 bits couplé à un additionneur dédié (Opération MAC en un seul cycle d'horloge)

- Compare, Select and Store Unit CSSU pour l'opération élémentaire de Add, Compare, Select ACS du Viterbi
- Encodeur Exponentiel pour le calcul d'une valeur exponentielle d'un accumulateur de 40 bits en une seul cycle d'horloge.

TD #4

Questions

1) Quelle est la puissance de calcul disponible sur le DSP TMS320 C5402 ?
2) Est-il possible de différencier les opérations de multiplication et d'addition?
3) En déduire si l'on peut ou non implémenter la structure #2 vu dans le TD #3 sur le DSP TMS320 C5402 pour les fréquences d'échantillonnage suivantes :
   a) 10 MHZ
   b) 20 MHz
   c) 40 MHz
   d) 60 MHz
4) Dans les cas ci-dessus où ce n'est pas possible, proposer des solutions/modifications matérielles pour améliorer les performances du DSP.

Plan

- **Chapitre V Travaux Pratiques sur FPGA**
  - TP # 1 : Prise en main des Outils et du langage VHDL
  - TP # 2 : Création de Librairies d'Opérateurs Numériques (Blocs Altera)
  - TP # 3 : Exemple de Machine à Etats à base de Registre d'Etats
  - TP # 4 : Exemple d'Etude Architecturale
  - TP # 5 : Exemple de Machine à Etats à base de Machine de Moore
  - TP # 6 : Mini-Projet de Maximum Ratio Combining

Plan

- **VHDL Tutorial**

Plan

- **FPGA FAQ**

Plan

- **Machines d’états de Moore et de Mealy**

Plan

- **Circuits Logiques Programmables**

  « Bien concevoir avec un FPGA »