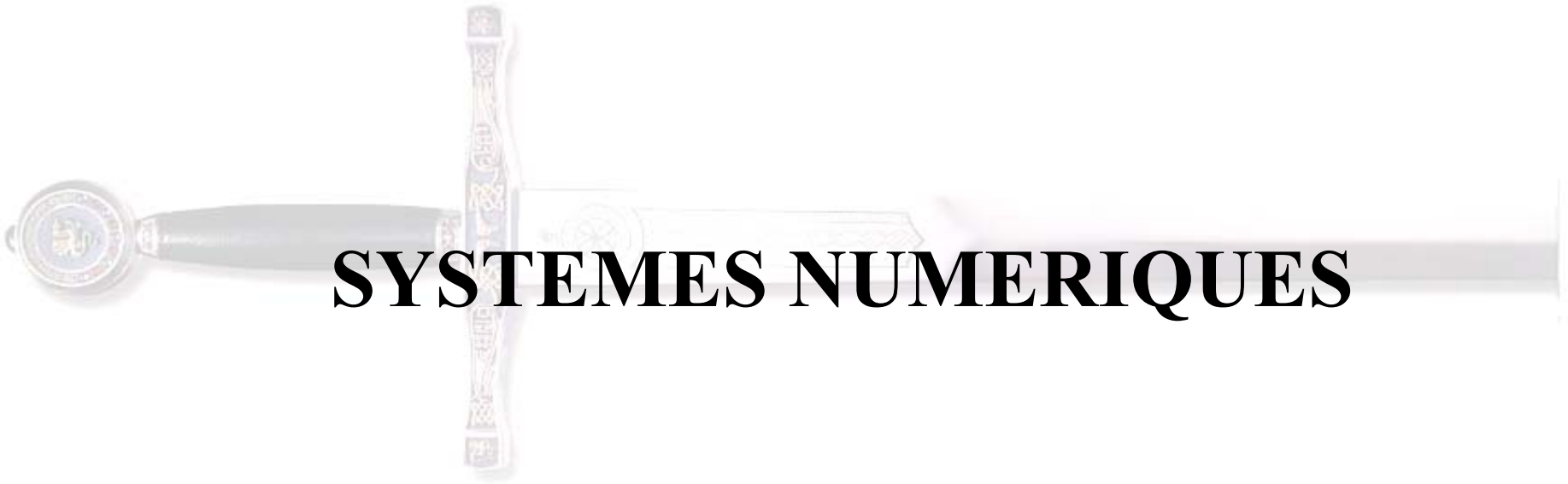
ÉCOLE NATIONALE SUPÉRIEURE
ÉLECTRONIQUE, INFORMATIQUE & RADIOCOMMUNICATIONS
BORDEAUX



# SYSTEMES NUMERIQUES

**Patrice KADIONIK**
kadionik@enseirb.fr
http://www.enseirb.fr/~kadionik

# PLAN DE LA FORMATION

# 1. OBJECTIFS DE LA FORMATION

• Renforcer les bases de microinformatique de l 'UV E4 A
• Permettre aux étudiants de réaliser une carte numérique complète à base de processeur (du matériel au logiciel ☺)

## 1.1. Contenu général de l'UV

• Projet : 40 h (coefficient : 4 note englobant le travail, le rapport avec soutenance/démonstration)
• Cours : 20 h (coefficient : 2 un examen)
• TP : 16 h (coefficient : 1,5 compte rendus de TP notés)

## 1.2. Les intervenants

- P. Kadionik
- V. Lebret
- G. Morizet
- P. Nouel

## 1.3. Le projet

♦Etude et réalisation d'une carte numérique à base de microcontrôleurs 8 bits 68HC11 et PIC (PCB fait à l 'ENSEIRB)

♦Développement du logiciel applicatif en langage C principalement

♦Lieu : salle E119

♦Moyens : PC, analyseurs logiques, émulateurs, compilateurs croisés, équipements de laboratoire courants, CAO de l'école

## 1.4. Le cours

20 h soit 20 séances de 1h

| Partie du cours | Responsable Intervenant | Nombre de séances (1 h) |
|---|---|---|
| Introduction : Les systèmes numériques aujourd'hui : systèmes embarqués, systèmes embarqués temps réel, codesign, OS, modules IP. Quand le software rejoint le hardware… | pk | 1 |
| Périphériques intégrés des processeurs :<br>Mémoires (ROM, RAM, FLASH, EEPROM…)<br>ports d'E/S<br>port série<br>Input Capture, Out Capture, timer, watchdog, Pulse Accumulator<br>Convertisseur A/N<br>PWM<br>Application au PIT Motorola | gm | 3 |

| | | |
|---|---|---|
| Périphériques intégrés des processeurs :<br>Liaison RS.232, bus I2C, bus SPI | vl | 2 |
| Périphériques intégrés des processeurs :<br>Interface réseau de terrain : interface CAN<br>Interface Ethernet : problème de la connectivité IP | pk | 4 |
| Présentation du microcontrôleur 68HC11 (fonctionnalités, périphériques intégrés…) | gm | 2 |
| Présentation du microcontrôleur PIC | vl | 2 |
| SoPC : Concepts. Mise en œuvre d'un module IP microcontrôleur 8 bits sur composant programmable | pn | 3 |

## 1.5. Les TP

16 h soit 4 séances de 4 h

| Thème du TP | Nombre de séances (4 h) |
|---|---|
| Microcontrôleur 68HC11 | 1 |
| Microcontrôleur PIC | 2 |
| Processeur NIOS : introduction au SoPC | 1 |

Vos Questions ?

# 2. INTRODUCTION. L'IMPORTANCE DES SYSTEMES NUMERIQUES

## 2.1. L'avènement du processeur

Les systèmes numériques ont vu leur importance progresser au rythme de l'importance prise par les microprocesseurs.

- 1971 : premier microprocesseur 4 bits 4004 d'Intel à 92,5 kHz vendu 200 $. Le succès a été là tout de suite.

- Motorola, Zilog, TI ont emboîté le pas…

Le marché des microprocesseurs est un marché qui croît de façon exponentielle.

Deux lois empiriques vérifiées depuis 30 ans sont vérifiées (en plus de la loi de Moore) :

- Loi de JOY : la puissance CPU en MIPS double tous les 2 ans.

- Loi de RUGE : on a besoin d'une Bande Passante de 0,3 à 1 Mb/s par MIPS.

Le marché du microprocesseur a donc aussi tiré le marché des télécommunications et des systèmes embarqués !

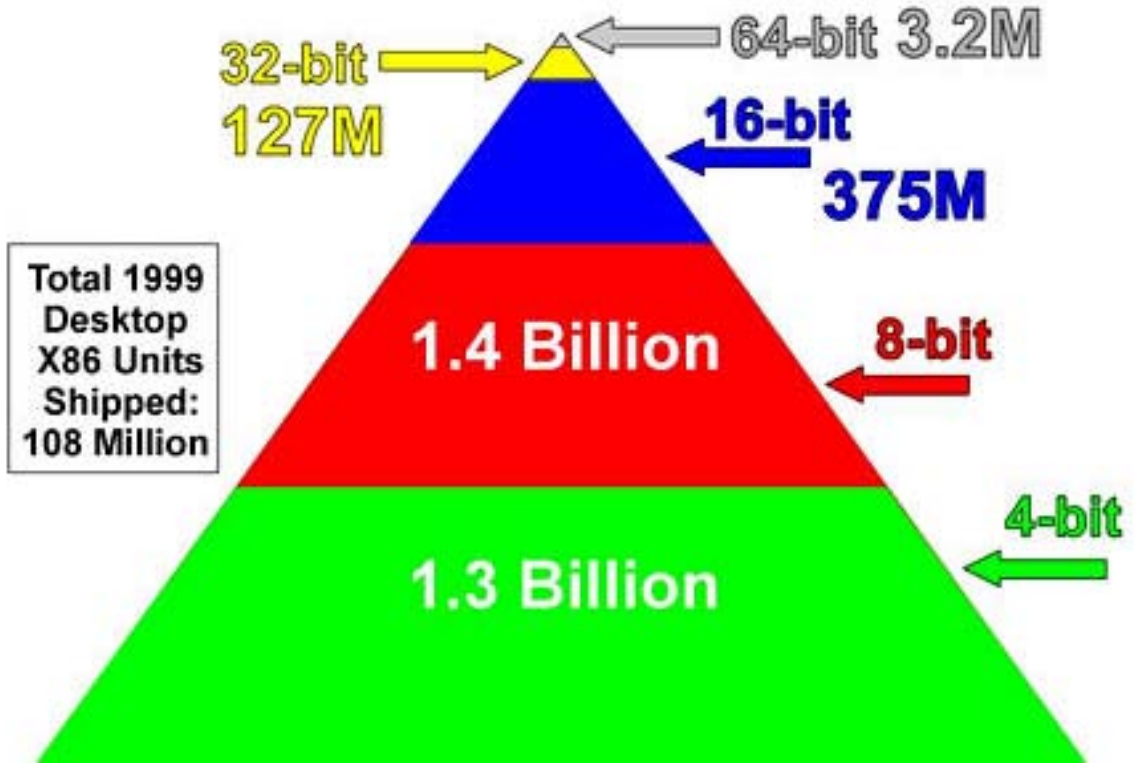
Grâce aux progrès de l'intégration sur silicium, on est passé rapidement du processeur 4 bits au :

- processeur 8 bits.
- processeurs 16 bits.
- processeurs 32 bits.
- processeurs 64 bits.

La taille d'un processeur correspond à la taille de ses registres de données (et généralement à la taille de son bus de données).

Il ne faut pas croire que le marché du microprocesseur se résume à celui du PC via les processeurs x86.

La figure suivante démontre le contraire (année 1999) :




.

Il a été vendu 108 millions de processeurs x86 pour le marché du PC contre 1,4 milliard de processeurs 8 bits pour le marché des systèmes embarqués (appelé aussi marché de l'embarqué) !

On voit ainsi que 5 % des processeurs vendus sont pour le marché du PC. Dans 85 % des cas, Microsoft Windows est utilisé.

Cela veut dire que pour 95 % des autres processeurs vendus, on utilisera un autre système d'exploitation (OS : *Operating System*). On trouvera ici dans 60 % des cas un OS propriétaire ; ce qui a poussé bon nombre à opter pour des OS libres comme Linux pour limiter les coûts…

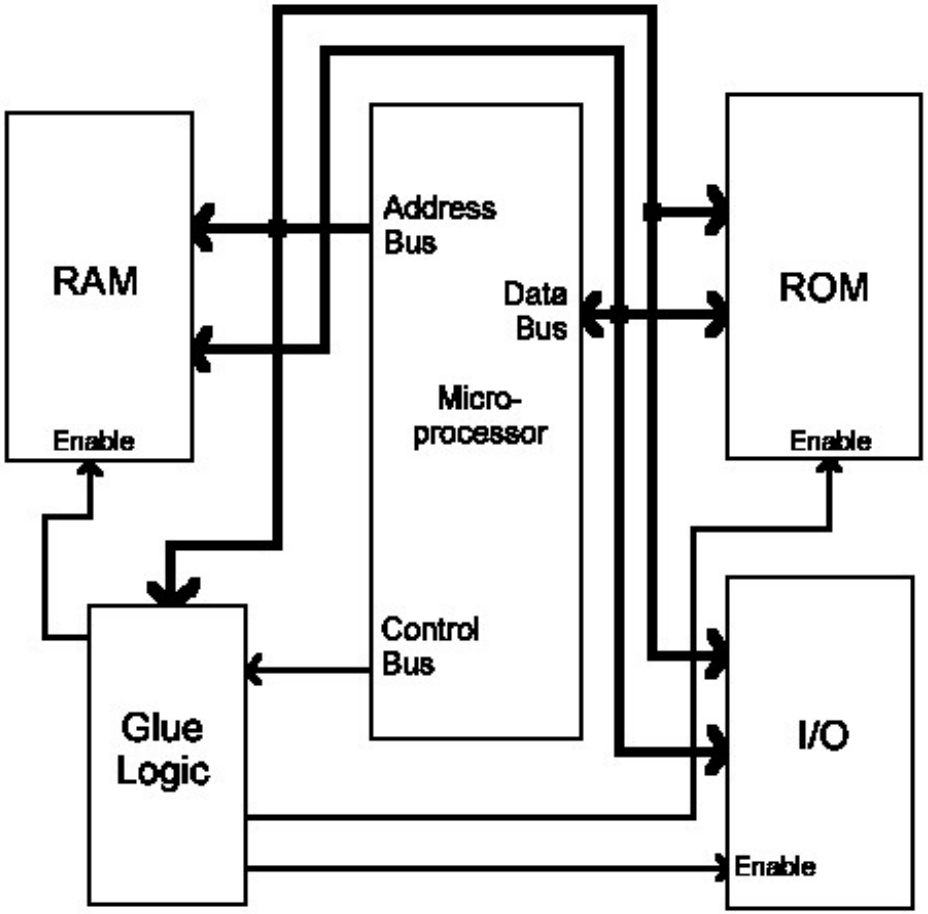
Les systèmes embarqués sont donc prépondérants !

## 2.2. Microprocesseur, microcontrôleur, DSP : à chacun son rôle !

Un microprocesseur est un processeur intégrant sur le silicium aucune mémoire, aucun périphérique…

On doit donc les rajouter en externe au microprocesseur en intercalant une « glue logique ».

Une conséquence est un coût final important surtout pour les grosses productions. C'est en fait la première génération de processeurs ('70 et '80) .



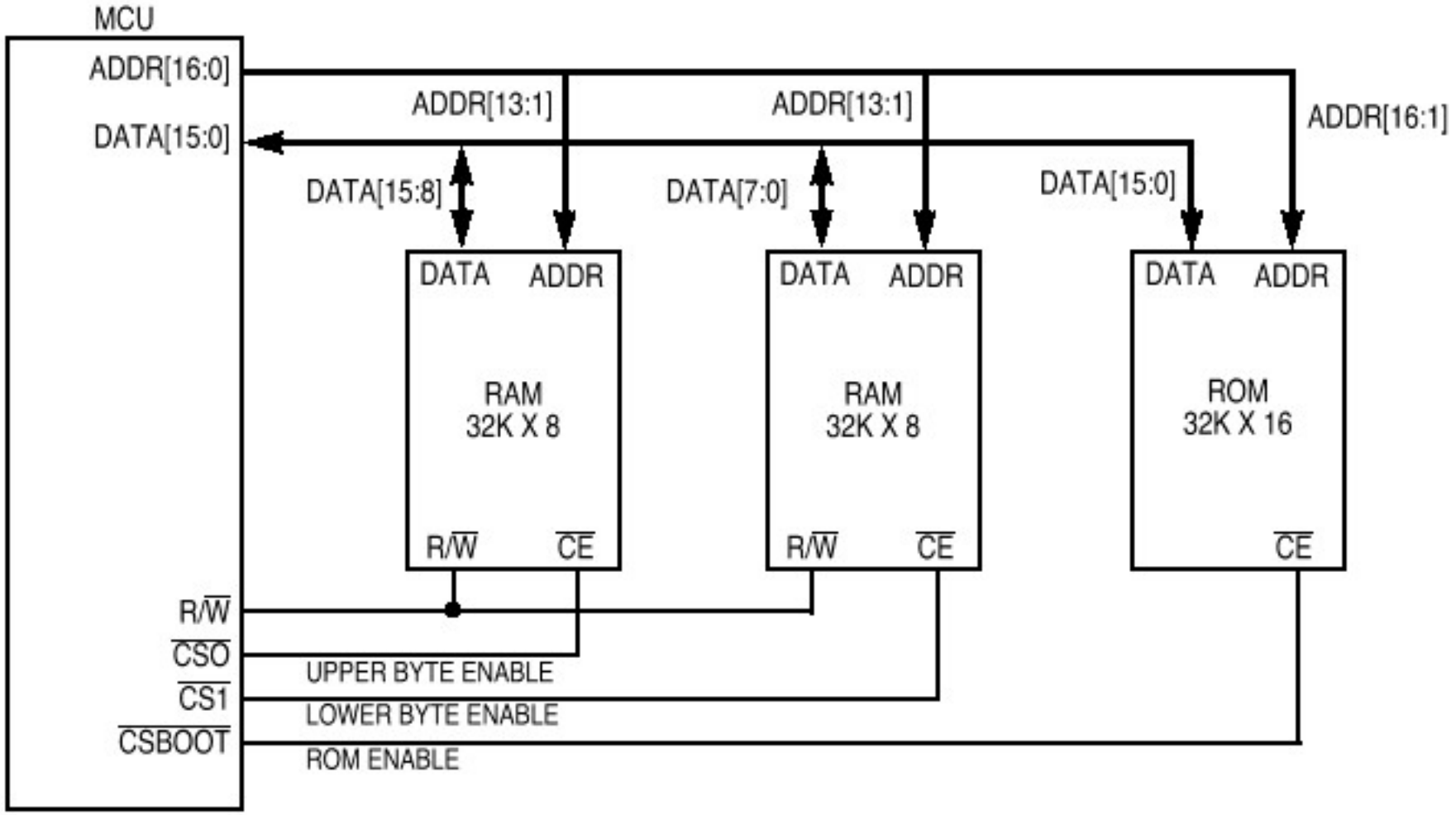

Un microcontrôleur est un processeur intégrant sur le silicium de la mémoire, des périphériques, de la glue logique :

- SRAM, PROM, ROM (*One Time Programming*), EEPROM
- E/S, liaison RS.232, ADC, réseau de terrain, Timer, PWM…

- Lignes de *Chip Select* avec paramétrage du nombre d'états d'attente

Cela permet de limiter le nombre de composants et donc le prix du système.

C'est le concept du *Single Chip*.



Un DSP (*Digital Signal Processor*) est un processeur dédié au traitement du signal. Il intègre dans son jeu d'instructions une instruction optimisée en temps d'exécution qui permet de réaliser à la fois une multiplication et une addition (instruction MAC *Multiply And Accumulate*). Cette opération est très importante et on la retrouve utilisée dans le filtrage numérique…

Un DSP peut intégrer des périphériques si bien que la frontière microcontrôleur / DSP est de plus en plus flou :
Le microcontrôleur Motorola ColdFire successeur du processeur Motorola 68K intègre dans certaines versions des instructions DSP !

| Embedded Processor | System Requirement | Feature | Benefit |
|---|---|---|---|
| **Microcontroller** | I/O Control | I/O Ports with bit-level control | Efficient control of external devices<br>Direct interface to actuators, switches and digital status signals |
| | Peripheral Communication | Serial Ports : SPI, I²C, Microwire, UART, CAN | Hardware support for expansion & external device networking and communications |
| | Precision control of motors and actuators | Sophisticated timers and PWM peripherals | Low software overhead |
| | Quickly resolve complex software program control flow | Conditional jumps<br>Bit test instructions<br>Interrupt priority control | Efficiently implement control oriented algorithms |
| | Fast response to external events | External interrupts with multiple priority levels | Program control immediately redirected on event occurrence with minimal overhead |
| | Conversion of sensor data | Analog-to-Digital (A/D) Converters | Hardware support for external sensors |

| Embedded Processor | System Requirement | Feature | Benefit |
|---|---|---|---|
| DSP | Software Filters | Multiply/Accumulate Unit Zero-overhead loops | Digital filtering in few cycles |
| | Interface to codecs | High-speed serial ports | Hardware support for translation of analog signals |
| | High data Throughput from serial ports | Peripheral DMA | Less wasted cycles fetching data from serial ports |
| | Fast data access | Harvard architectures and variants | Fast execution of signal processing algorithms |

## 2.3. Les systèmes embarqués. Importance du Temps Réel

Un système embarqué peut être défini comme un système électronique et informatique autonome ne possédant pas des entrées/sorties standards comme un clavier ou un écran d'ordinateur.

Le système matériel et l'application sont intimement liés et noyés dans le matériel et ne sont pas aussi facilement discernables comme dans un environnement de travail classique de type PC.

On peut citer comme exemples de systèmes embarqués :

- Un four à micro ondes.
- Une télécommande de TV.
- Une fusée.
- Un missile.

Généralement, un système embarqué doit respecter :

- des contraintes temporelles fortes *(Hard Real Time*)
- on y trouve enfoui un système d'exploitation ou un noyau Temps Réel (*Real Time Operating System,* RTOS).

Le Temps Réel est un concept un peu vague :

On pourrait le définir comme : "*Un système est dit Temps Réel lorsque l'information après acquisition et traitement reste encore pertinente*".

Cela veut dire que dans le cas d'une information arrivant de façon périodique (sous forme d'une interruption périodique du système), les temps d'acquisition et de traitement doivent rester inférieur à la période de rafraîchissement de cette information.

Pour cela, il faut que le noyau ou le système Temps Réel soit *déterministe* et *préemptif* pour toujours donner la main durant le prochain *tick* à la tâche de plus forte priorité prête.

Une confusion classique est de mélanger Temps Réel et rapidité de calcul du système donc puissance du processeur (microprocesseur, microcontrôleur, DSP). On entend souvent :
« *Etre temps Réel, c'est avoir beaucoup de puissance : des MIPS, des MFLOPS…* ».

Ce n'est pas toujours vrai. En fait, être Temps Réel dans l'exemple donné précédemment, c'est être capable d'acquitter l'interruption périodique (moyennant un temps de latence de traitement d'interruption imposé par le matériel), traiter l'information et le signaler au niveau utilisateur (réveil d'une tâche, libération d'un sémaphore…) dans un temps inférieur au temps entre deux interruptions périodiques consécutives.

On est donc lié à la contrainte durée entre deux interruptions.

Si cette durée est de l'ordre de la seconde (pour le contrôle d'une réaction chimique par exemple), il ne sert à rien d'avoir un système à base de Pentium III ! Un simple processeur 8 bits du type microcontrôleur Motorola 68HC11 ou Microchip PIC ou même un processeur 4 bits fera amplement l'affaire ; ce qui permettra de minimiser les coûts sur des forts volumes de production.

Il faut toujours garder à l'esprit cette donne spécifique de l'embarqué où économiser une simple résistance à 0,20 Francs (0,03 Euros) permet d'économiser 200000 Francs (30 490 Euros) sur un volume d'un million de pièces !

Si ce temps est maintenant de quelques dizaines de microsecondes (pour le traitement des données issues de l'observation d'une réaction nucléaire par exemple), il convient de choisir un processeur nettement plus performant comme un processeur 32 bits Motorola 68K ou ColdFire.

Il est à noter que contrairement à l'environnement grand public PC où les processeurs Intel et AMD sont rois, ce sont plutôt les processeurs Motorola 68K, ColdFire, PowerPC… qui sont utilisés dans l'embarqué sous forme de cartes au standard VME (*Versa Module Eurocard*).

Ces processeurs sont réputés pour leur rapidité de traitement des interruptions et pour la prise en charge d'interruptions avec un niveau de demande d'interruption…

L'exemple donné est malheureusement idyllique (quoique fréquent dans le domaine des télécommunications et réseaux) puisque notre monde interagit plutôt avec un système électronique de façon apériodique.

Il convient donc avant de concevoir ledit système de connaître la durée minimale entre 2 interruptions ; ce qui est assez difficile à estimer voire même impossible. C'est pour cela que l'on a tendance à concevoir dans ce cas des systèmes performants (entre terme de puissance de calcul CPU et rapidité de traitement d'une interruption) et souvent surdimensionnés pour respecter des contraintes Temps Réel mal cernées à priori. Ceci induit en cas de surdimensionnement un surcoût non négligeable…

Outre les contraintes Temps Réel que l'on retrouve souvent dans un système embarqué, il existe d'autres contraintes importantes à prendre en compte :

- L'encombrement.
- L'environnement extérieur.
- L'aspect mécanique.
- La consommation.
- La tolérance aux fautes.

## 3. LE CHOIX D'UN SYSTEME D'EXPLOITATION : LE PHENOMENE LINUX

Linux depuis presque 3 ans est en train de conquérir un domaine où on ne l'attendait pas vraiment : l'univers des systèmes embarqués.

Pourquoi retrouve-t-on Linux dans l'embarqué ?

Tout d'abord pour ses qualités qu'on lui reconnaît maintenant dans l'environnement plus standard du PC grand public :

- Libre, disponible gratuitement au niveau source : pas de royalties à reverser.
- Ouvert.
- Différentes distributions proposées pour coller au mieux à un type d'application.
- Stable et efficace.
- Aide rapide en cas de problèmes par la communauté Internet des développeurs Linux.
- Nombre de plus en plus important de logiciels disponibles.
- Connectivité Internet en standard.

Linux a aussi d'autres atouts très importants pour les systèmes embarqués :

- Portage sur processeurs autres que x86 : PowerPC, ARM, MIPS, 68K, ColdFire…
- Taille du noyau modeste compatible avec les tailles de mémoires utilisées dans un système embarqué (500 Ko pour Hard Hat Linux de MontaVista).

- Différentes distributions proposées suivant le domaine : routeur IP, PDA, téléphone…
- Support du chargement dynamique de modules qui permet d'optimiser la taille du noyau.
- Migration rapide et en douceur pour un spécialiste Linux à Linux embarqué ; ce qui réduit les temps de formation (et les coûts…).

On retrouve généralement aussi un certain nombre de suppressions de fonctionnalités dans les distributions Linux embarqué :

- Pas de gestion de la MMU (*Memory Management Unit*) pour ne pas pénaliser les performances globales du système.
- Utilisation de systèmes de fichiers en mémoire RAM (*RAM Disk*) ou en mémoire FLASH *(FLASH Disk)*.

On a en fait entendu parler pour la première fois officiellement de Linux embarqué à une exposition *Linux World* en 1999 où les sociétés Motorola, Force et Ziatech ont présenté un système CompactPCI fonctionnant sous Linux.

En 2000 a été créé le consortium Linux embarqué (*Embedded Linux Consortium*) dont le but est de centraliser et de promouvoir les développements de solutions Linux embarqué. Ce consortium regroupe des éditeurs de distribution Linux, des éditeurs de systèmes Temps Réel propriétaires (comme WindRiver pour VxWorks) et des fabricants de composants. Il compte actuellement plus de 100 membres.

Les distributions Linux embarqué se sont rapidement imposées face à des distributions propriétaires généralement Temps Réel comme VxWorks, pSOS, QNX… où l'on est d'abord obligé de payer pour accéder à la plateforme de développement puis de payer des royalties pour chaque système (ou cible) que l'on commercialise ensuite.

Il est à noter que l'on observe une évolution de ce système à péage de certains face à la « menace » Linux.

Linux embarqué supporte aussi différentes extensions Temps Réel qui mettent en place une couche d'abstraction logique entre matériel, interruptions et Linux. Linux et

l'ensemble des processus sont généralement considérés comme la tâche de fond exécutée quand il y  a rien de Temps Réel à faire…

On peut citer comme extensions Temps Réel :

- La distribution RTLinux et sa distribution Mini RTLinux pour l'embarqué.
- La distribution RTAI.

Le tableau suivant donne une image du marché des systèmes embarqués :

| **Besoin** | **Miniature** | **Petit** | **Moyen** | **Haut de gamme** | **PC embarqué** | **Embarqué haute disponibilité** |
|---|---|---|---|---|---|---|
| **Taille RAM** | <0,1 Mo | 0,1-4 Mo | 2-8 Mo | 8-32 Mo | 16-64 Mo | > x Mo |
| **Taille ROM/FLASH** | 0,1-0,5 Mo | 0,5-2 Mo | 2-4 Mo FLASH | 4-16 Mo FLASH | xx Mo | Go-To |
| **Processeurs** | DragonBall 68K<br>Mcore<br>ColdFire<br>ARM | | MIPS<br>Hitachi SH<br>x86<br>PowerPC | | | Pentium<br>PowerPC |
| **Caractéristiques matérielles** | MMU optionnelle | | Ardoise Internet<br>Carte unité centrale<br>System on Chip (*SoC*) | | | CompactPCI |
| **Exemples d'applications** | Caméra numérique<br>PDA<br>Téléphone | | Routeur<br>Décodeur<br>Stockage en réseau<br>Imprimante en réseau | | | Commutateur téléphonique<br>Routeur haute performance<br>Serveur central |

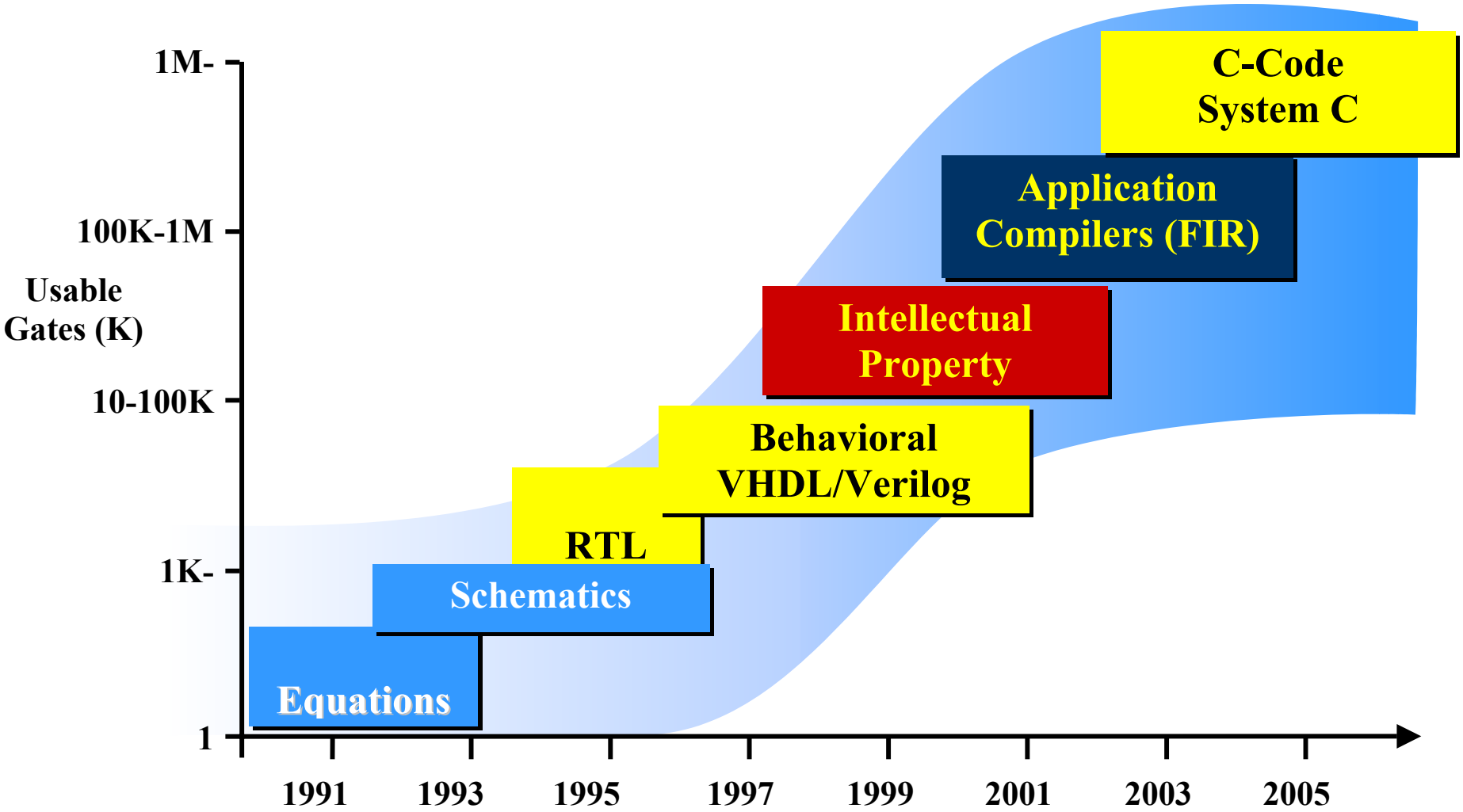
# 4. CODESIGN : QUAND LE MATERIEL REJOINT LE LOGICIEL

La capacité de conception des systèmes numériques permet aujourd'hui de tout intégrer dans un même composant.

On travaille donc au niveau système et non plus au niveau porte élémentaire ou schématique. On parle de système sur silicium SoC (*System on Chip*) ou SoPC *(System on Programmable Chip*).

Ceci est dû à la loi empirique de Moore qui dit que pour une surface de silicium donné, on double le nombre de transistors intégrés tous les 18 mois !

| | 1998 | 1999 | 2001 |
|---|---|---|---|
| Technologie | 0,25 µm | 0,18 µm | 0,15 µm |
| Complexité | 1 M de portes | 2-5 M | 5-10 M |
| | Loi de Moore → | | |



On utilise maintenant des langages de description du matériel (VHDL, Verilog) pour synthétiser et aussi tester les circuits numériques. On a ainsi une approche logicielle pour concevoir du matériel.

Avec l'augmentation de l'intégration, les systèmes numériques se sont complexifiés et que la mise sur le marché doit être la plus rapide possible :

- Prise en compte du *Time To Market* (TTM).
- Réutilisation de choses déjà réalisées (*Design Reuse*).

On a ainsi vu apparaître la notion de blocs IP (*Intellectual Property*) qui est possible par l'utilisation des langages de description du matériel.

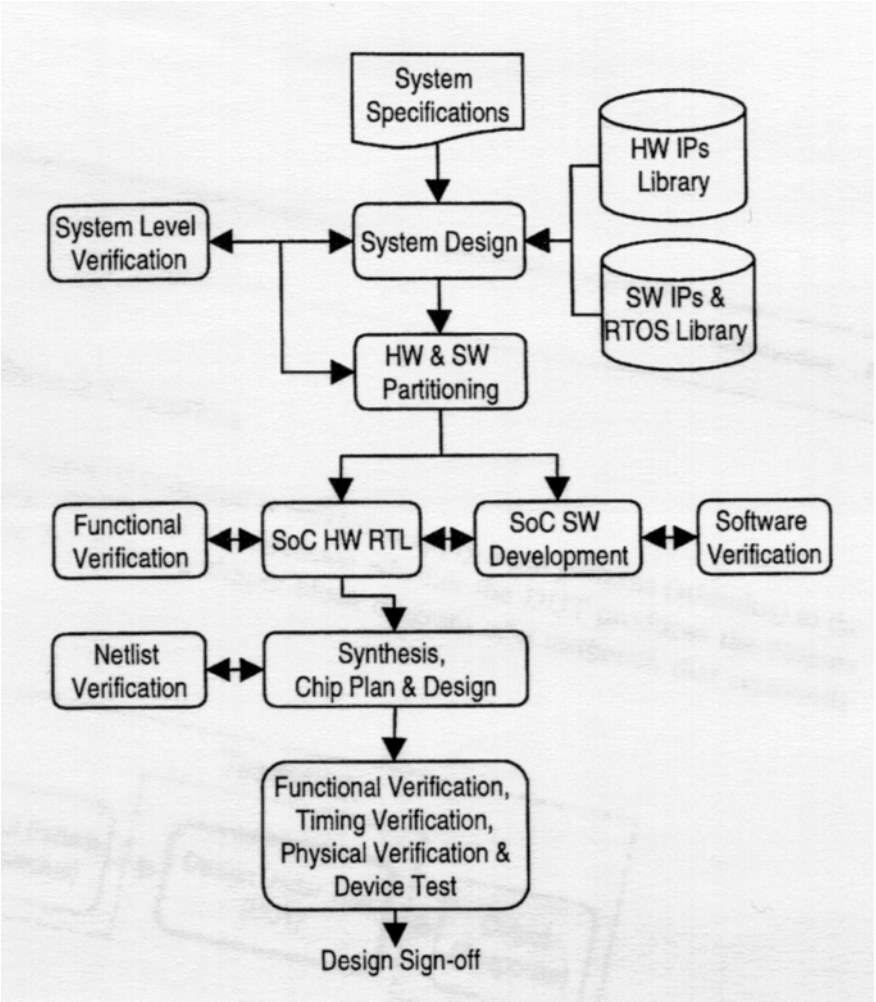
On achète des blocs IP (interface CAN, DCT…) comme on achète un circuit intégré.

La majeure partie du temps de développement est consommée par la phase de vérification et de tests (estimée à 40 à 70 %) !

(D'après ASIC Technology & News) :

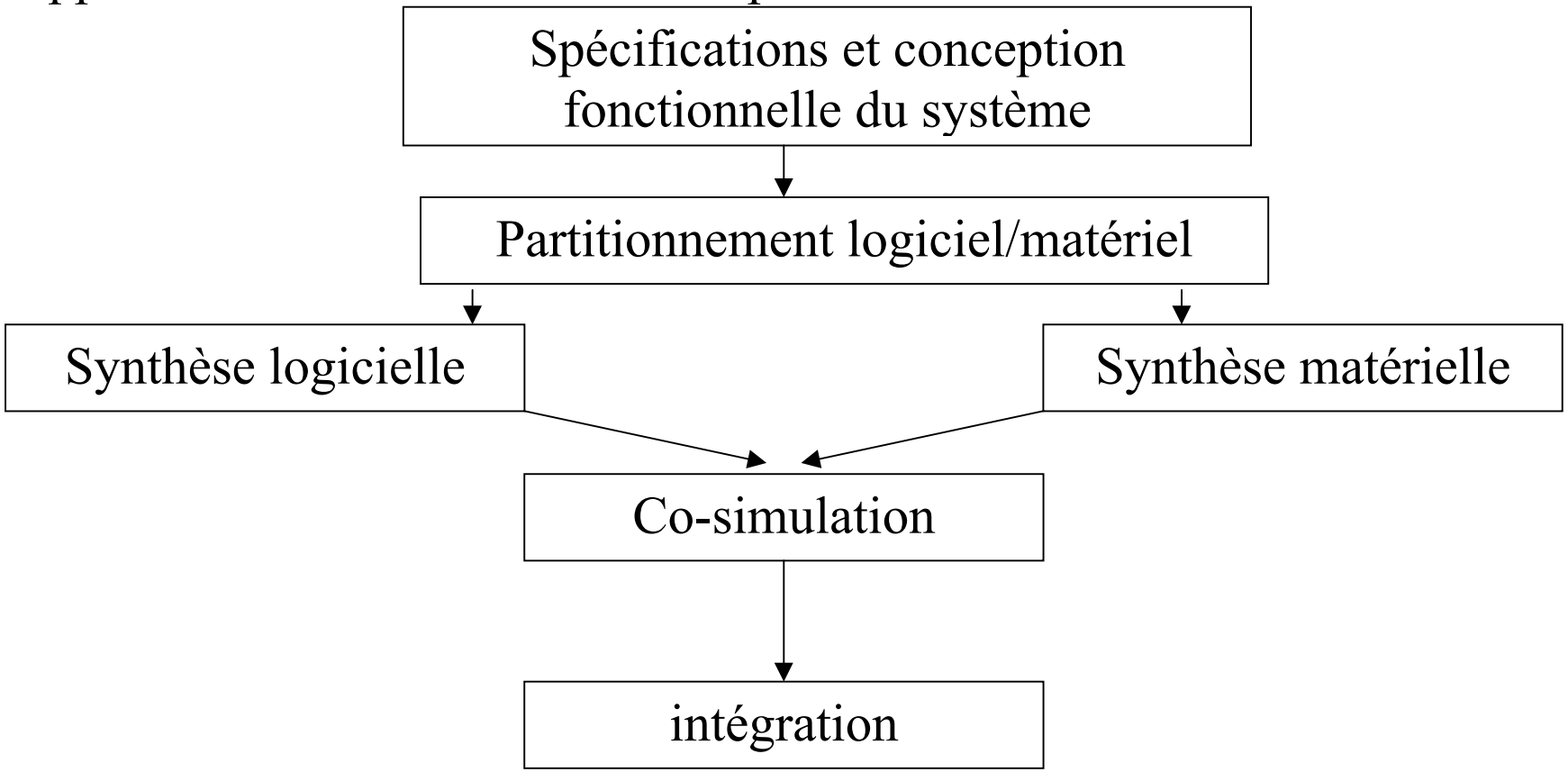
| | |
|---|---|
| Spécifications : | 5% du temps |
| Capture schéma : | 10 % |
| Vérification design : | 25 % |
| Génération vecteurs de test : | 40 % |
| Test prototype : | 10 % |
| Interface client : | 10 % |

On voit l'importance des *testbenchs* (VHDL)…



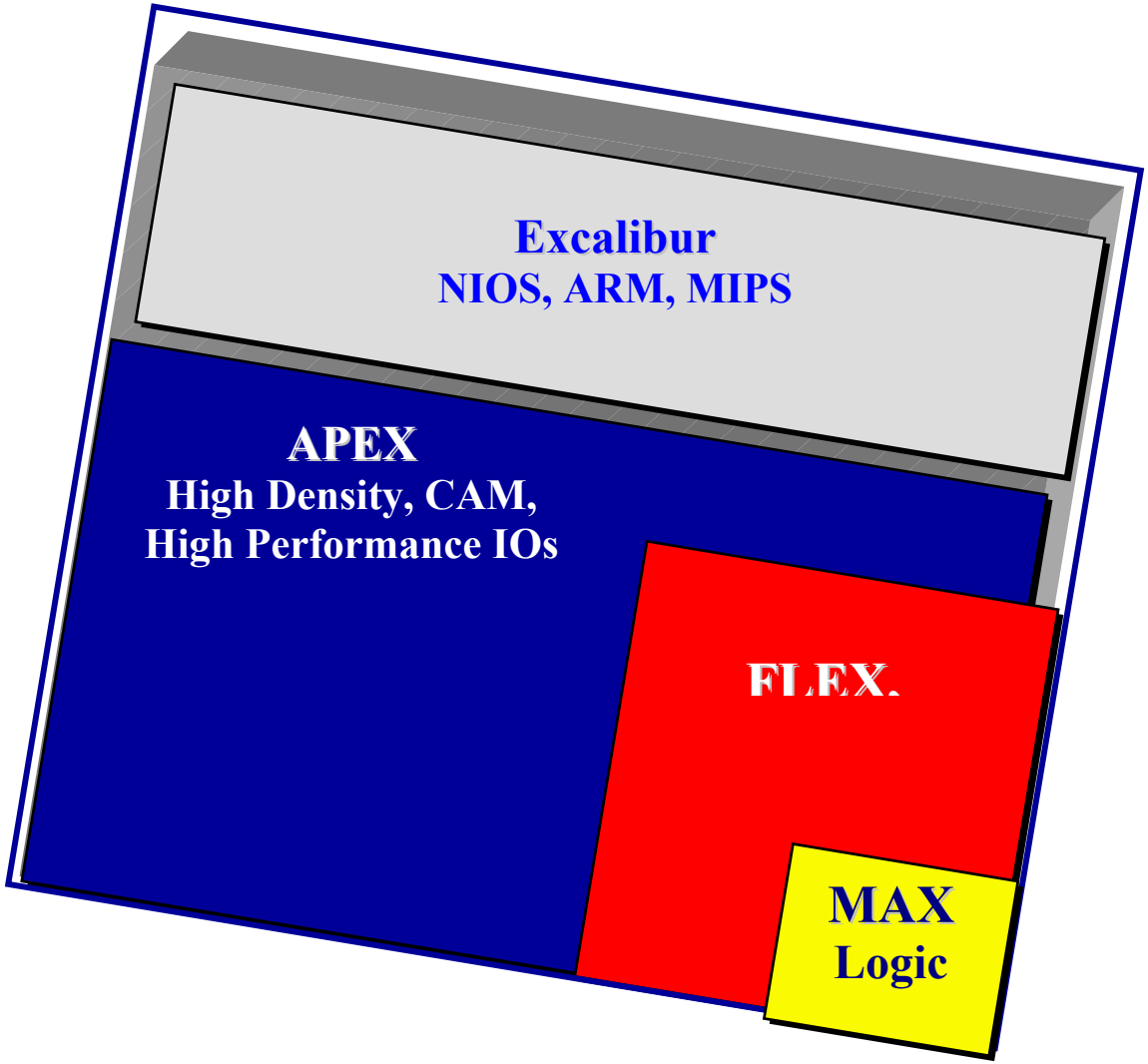
La difficulté est maintenant de savoir si l'on doit réaliser une fonctionnalité par logiciel ou par matériel pour des raisons de performance, de coût…

C'est l'approche *codesign* quand le matériel rejoint le logiciel…
Cette approche nouvelle est maintenant possible !



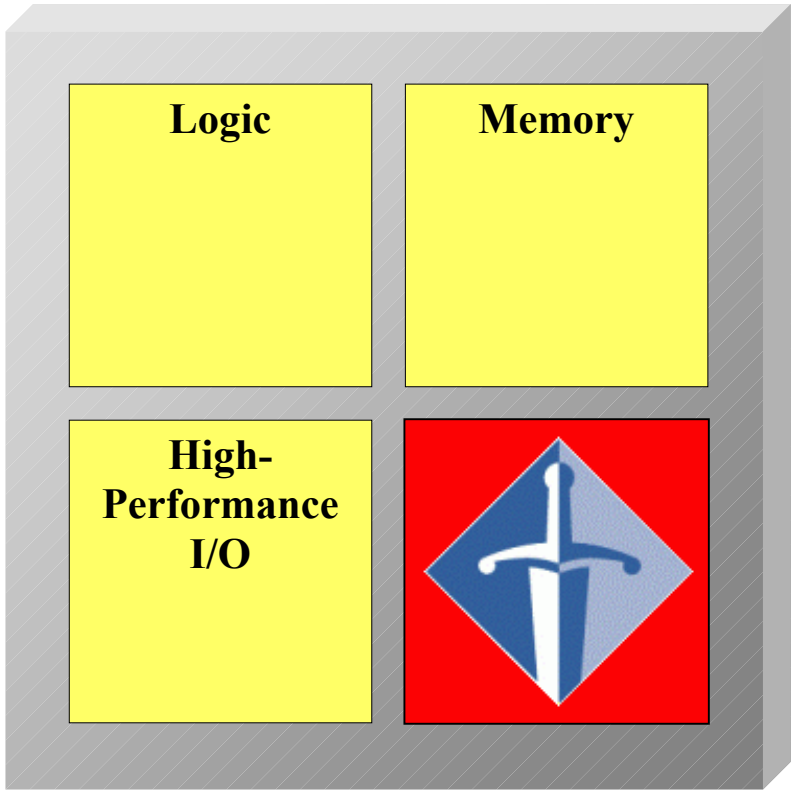
# 5. UN EXEMPLE DE SOLUTION DE CODESIGN : L'OFFRE NIOS D'ALTERA

**System On a Programmable Chip**



L'offre SoPC Excalibur d'Altera permet la flexibilité de programmation des PLD (*Programmable Logic Device*) avec les performances de temps de traitement d'un processeur embarqué sur silicium pour répondre au besoin d'un court TTM.

***A Complete SOPC Solution***

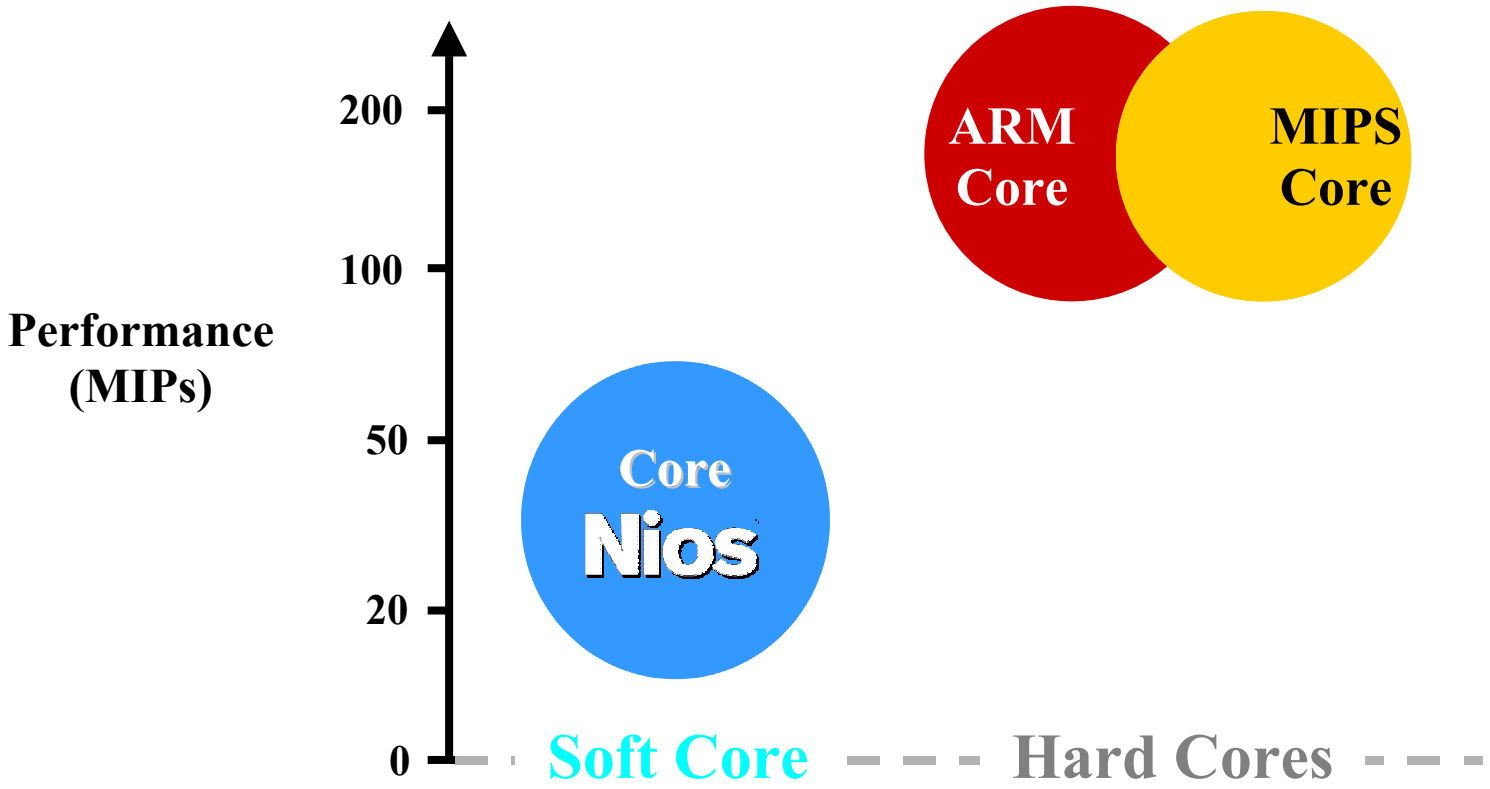


# Cœurs de processeur Hard et Soft embarqués sur silicium

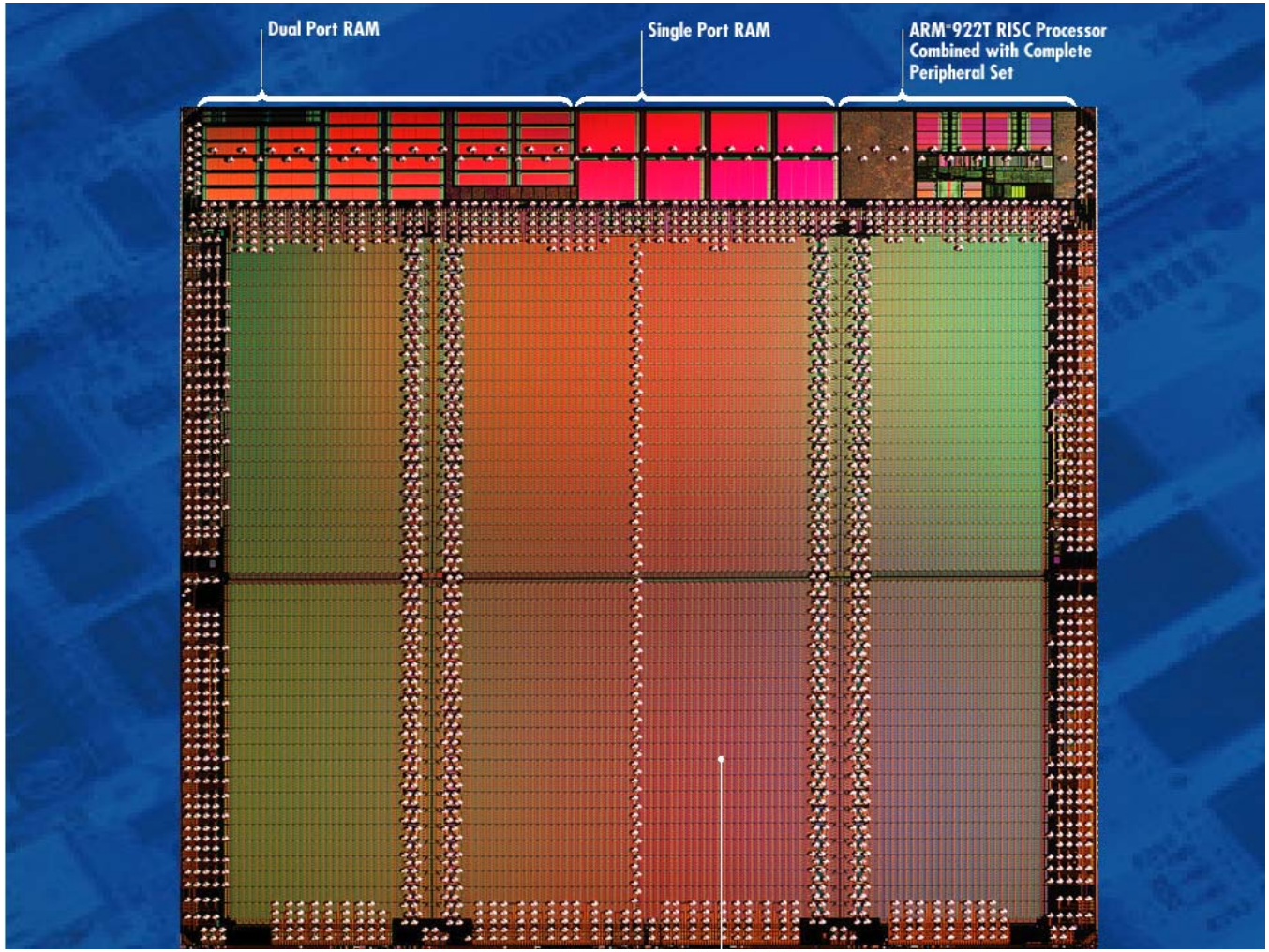
- **Soft Cores**
  - ✓ Maximum Flexibility
  - ✓ Easy Changes & Upgrades
  - ✓ Scalability
  - ✓ Fast Process Migrations
- **Hard Cores**
  - ✓ Maximum Performance
  - ✓ Optimized Die Size
  - ✓ Guaranteed Timing
  - ✓ Established Architectures

*Need Both for Complete Solution*

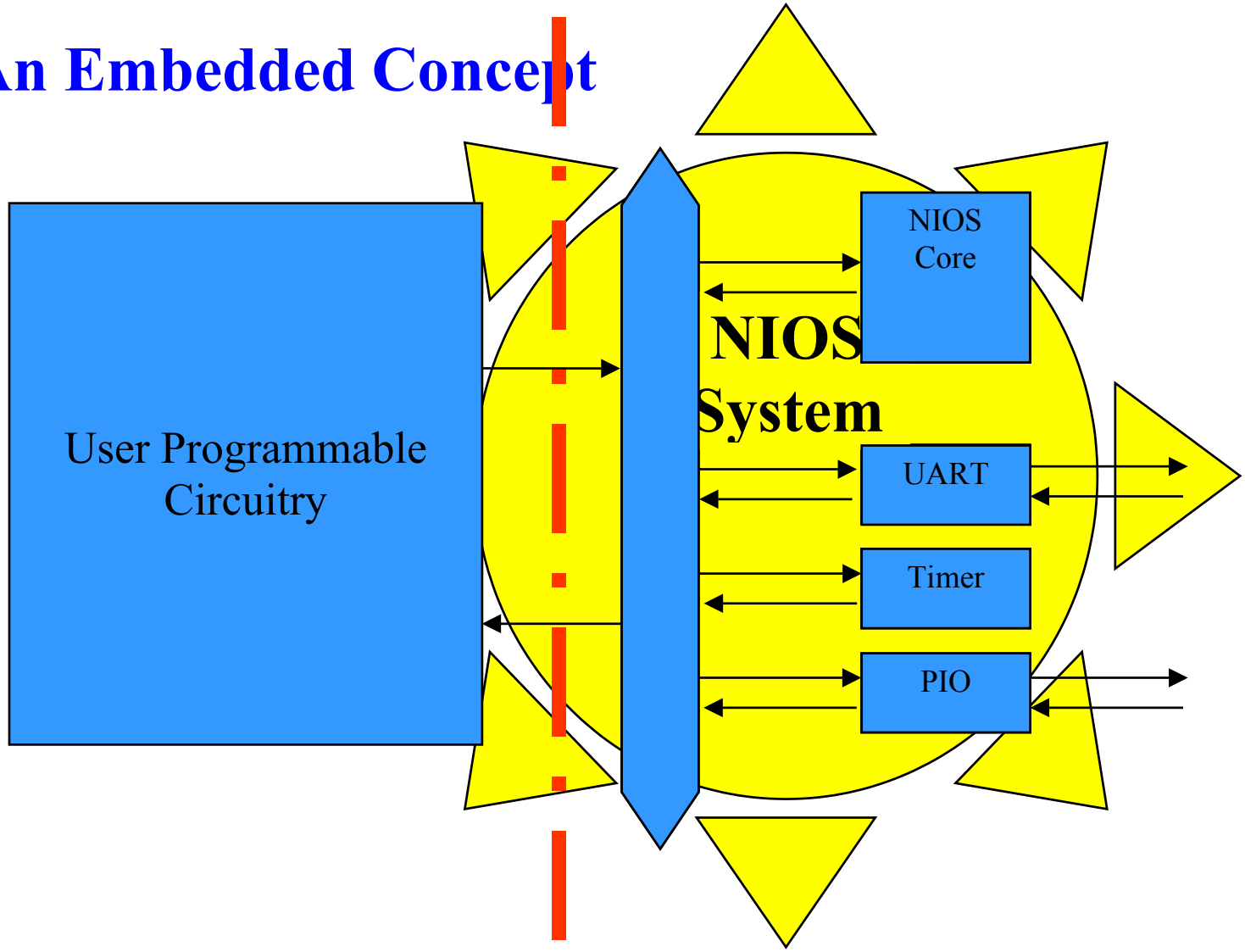
# Excalibur Solutions Provide Flexibility & Horsepower for Broad Market Coverage



# Excalibur et son cœur Hard ARM

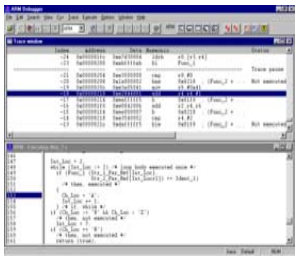


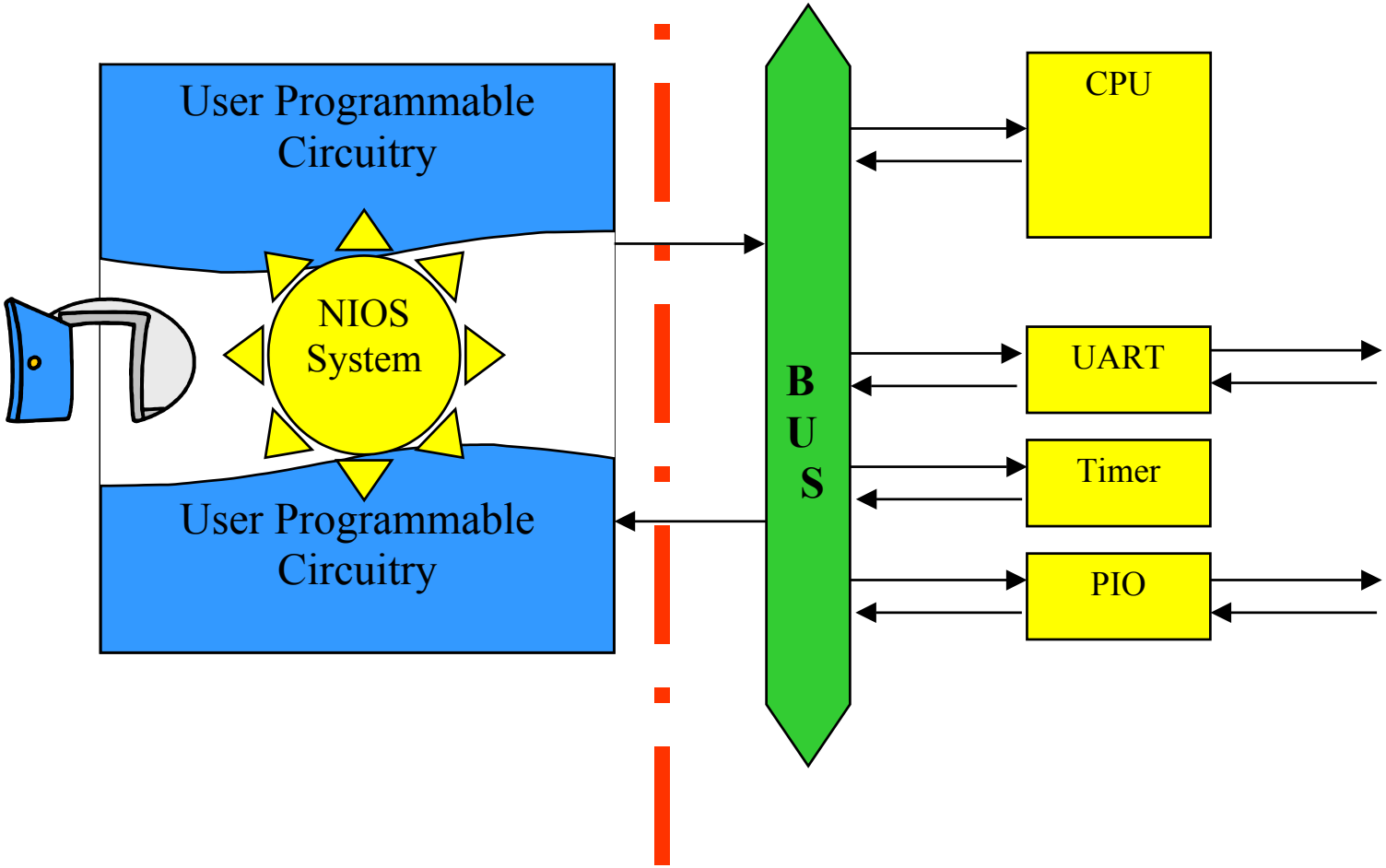
# Nios : An Embedded Concept



# Nios An Embedded Concept

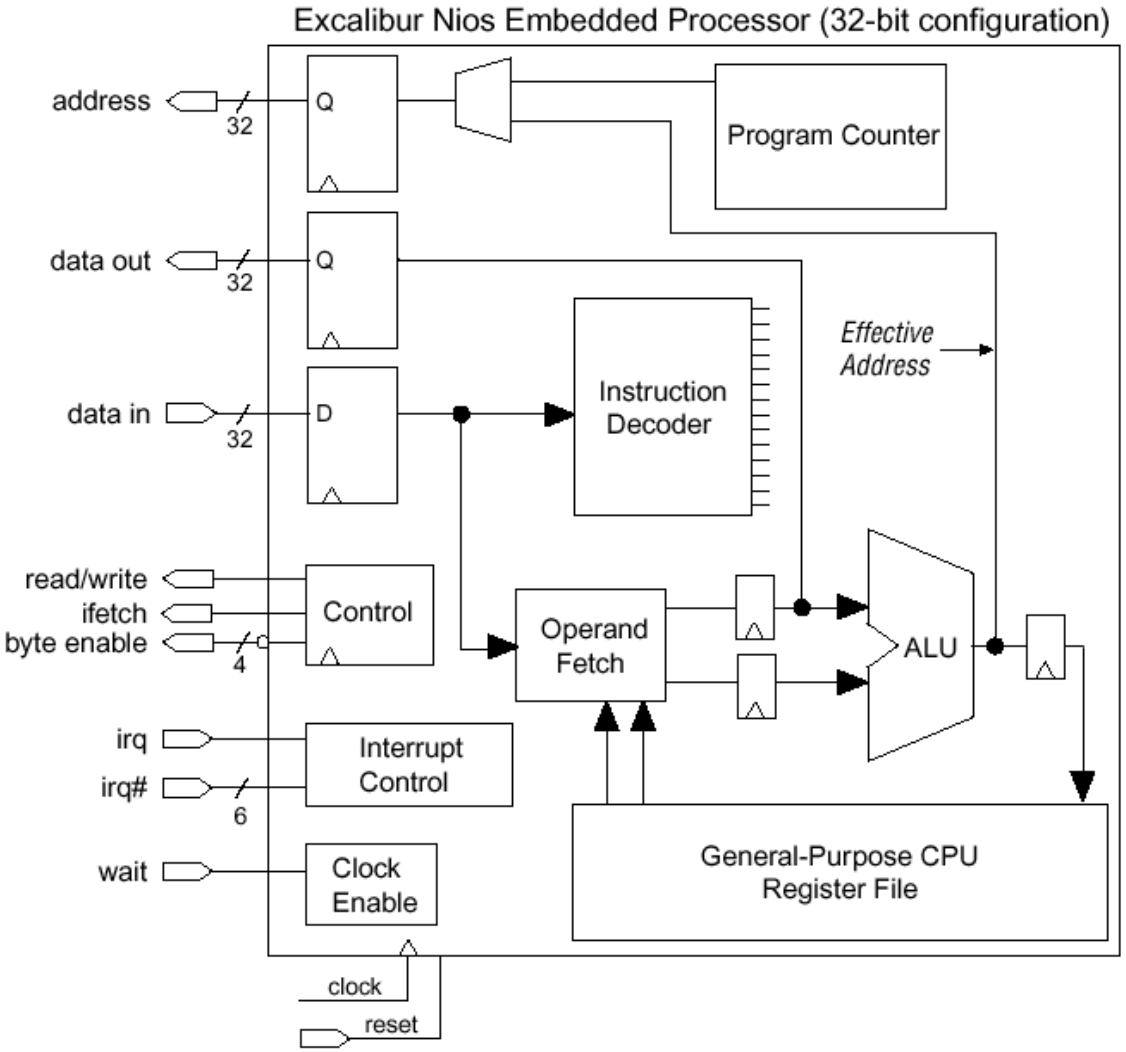
Software Side





# Nios RISC Processor Block Diagram

- Standard RISC Components
- Fully-Synchronous Interface



# Nios Flexibility & Scalability

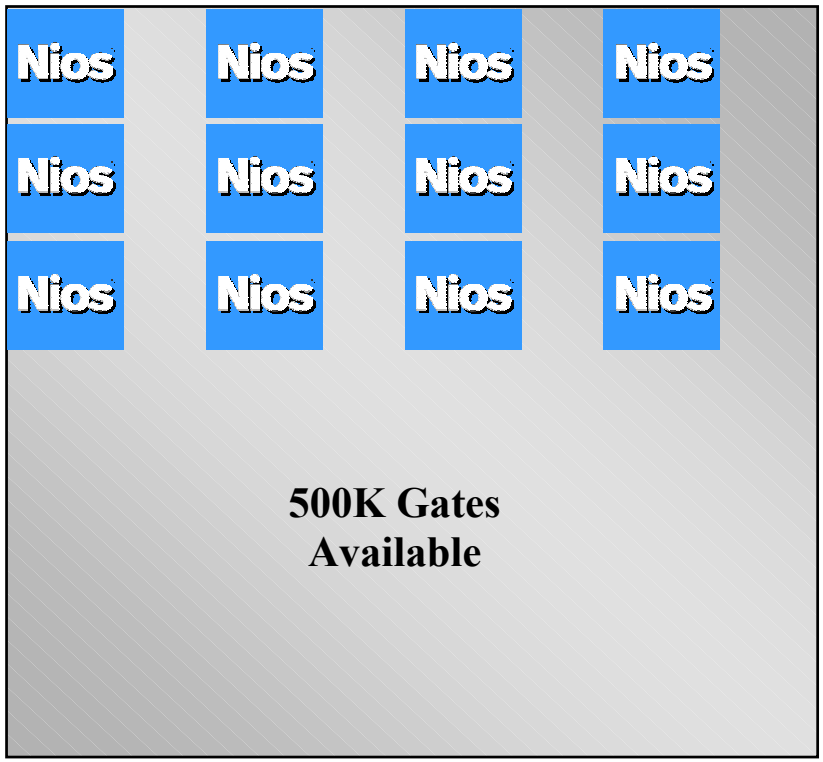
**High-Performance Embedded Processor**


**APEX EP20K100E**

**Custom DSP**

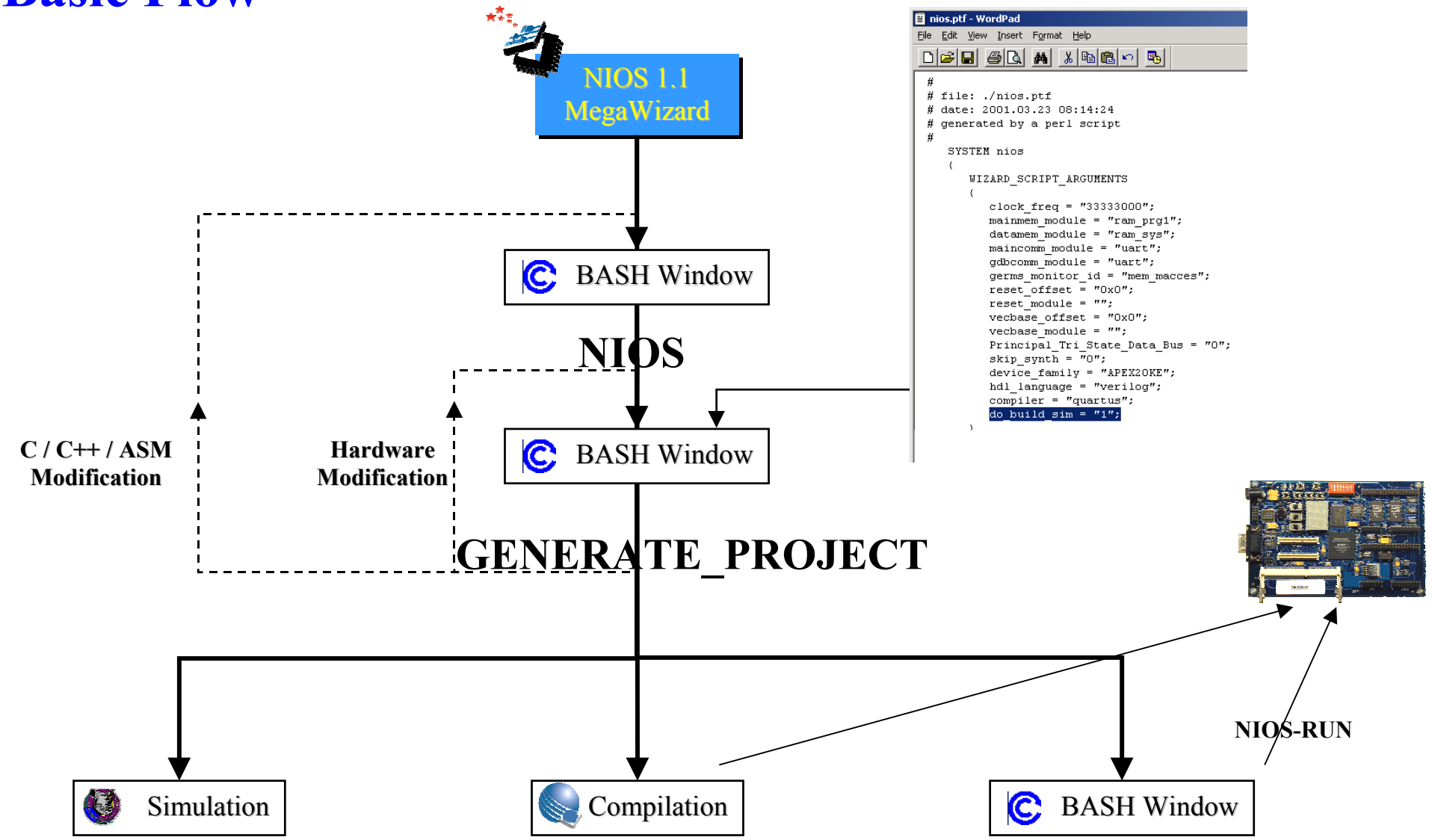

**APEX EP20K200E**

**Multi-Processor Micro-Coded System**



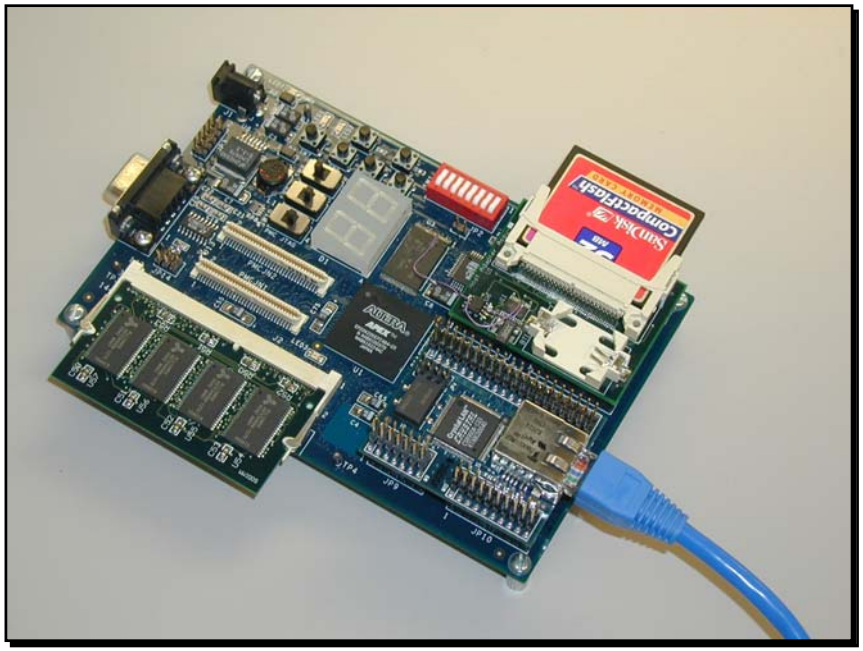

**APEX EP20K1000E**

# Basic Flow



# Linux Development Kit

- Open-Source µCLinux Operating System
- Development Kit Contents
  - µCLinux Source Code
  - Ethernet Development Board
  - SDRAM / Flash Memory Module
  - SDRAM Controller Core
  - IDE Interface
  - Compact Flash Interface
  - Real Time Clock
  - Reference Design
    - Quartus Project
    - Web Server Application
- Price $2495 (www.microtronix.com)



# Development Tools

- Software Development Tools
  - RedHat GNUPro Toolkit (Compiler, Debugger)
  - Nios Ethernet Development Kit (TCP/IP Stack)


- Operating System Support
  - Linux Development Kit
  - ATI Nucleus
  - KROS (Shugyo Designs)
  - µC OS II