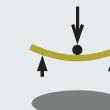
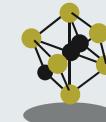
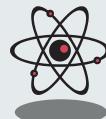
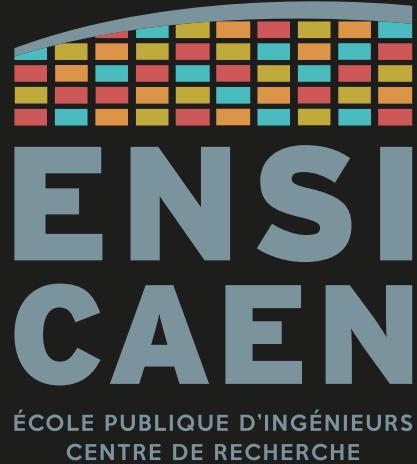


# Architectures pour le calcul

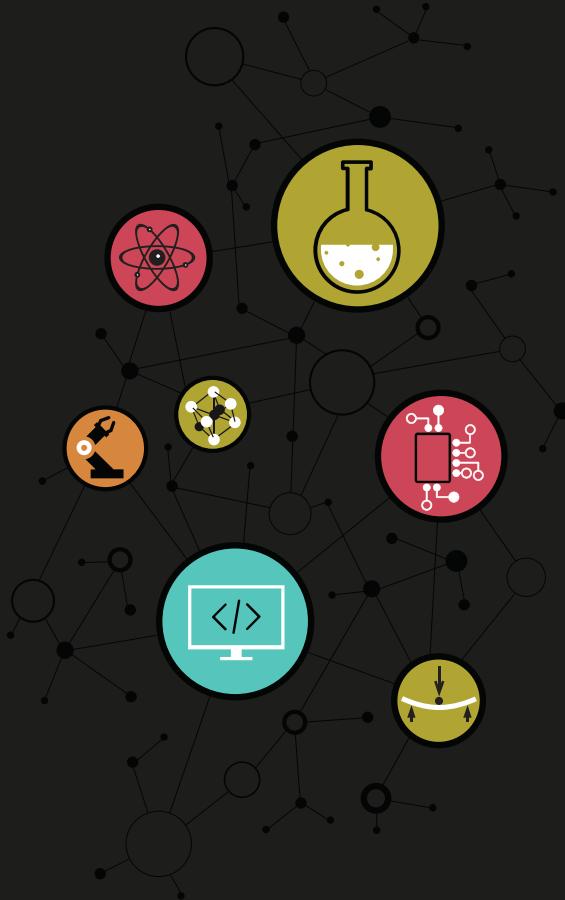
## 2A Informatique



Dimitri Boudier – PRAG ENSICAEN  
[dimitri.boudier@ensicaen.fr](mailto:dimitri.boudier@ensicaen.fr)

Avec l'aide précieuse de :

- Hugo Descoubes (PRAG ENSICAEN)



# Historique de l'électronique et présentation des architectures de processeurs

Puis, plus en profondeur :

- MCU MicroController Unit
- GPP General Purpose Processor
- AP Application Processor
- GPU Graphics Processing Unit
- DSP Digital Signal Processor
- NPU Neural Processing Unit
- FPGA Field Programmable Gate Array
- ASIC Application-Specific Integrated Circuit

Distinguer et comprendre les différentes architectures de processeurs  
et pourquoi elles sont plus adaptées à une situation ou une autre.

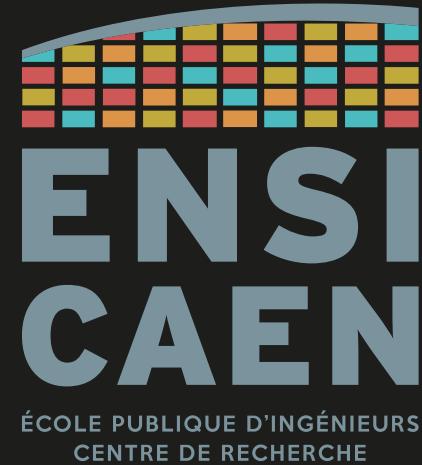
Culture générale uniquement → Pas d'évaluation !

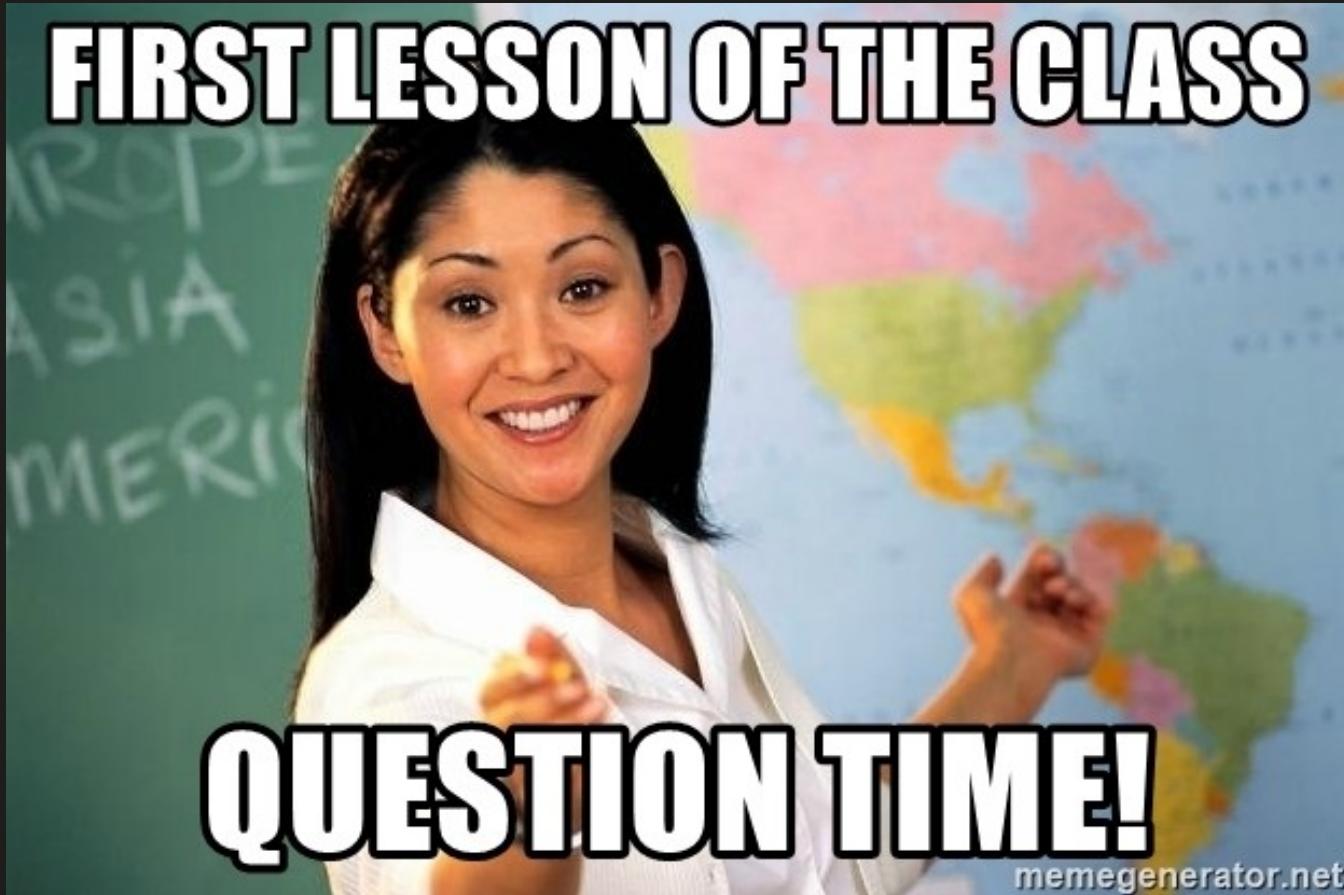
# RETOUR VERS LE FUTUR

Un peu d'histoire avant d'étudier les technologies actuelles

Puis classification des processeurs selon :

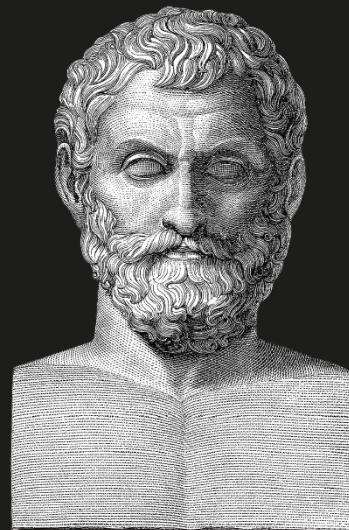
- leur architecture
- leur champ d'application





Les effets de l'électricité statique et du magnétisme ont été découverts pendant l'antiquité.

Thalès de Milet découvrit ces effets vers -600 grâce l'ambre, dont le nom grec (ἤλεκτρον ou éléktron) donnera son nom au phénomène physique vers 1600.



Thalès ~(-625, -545)



Ambre contenant un moustique

## Formalisation et premières inventions

La distinction entre phénomènes électriques et magnétiques se fait en 1600 (Gilbert).

En 1827 les travaux de Georg Ohm mènent à une théorie générale de l'électricité, complétée en 1865 par les équations de Maxwell qui traitent de l'électro-magnétisme.

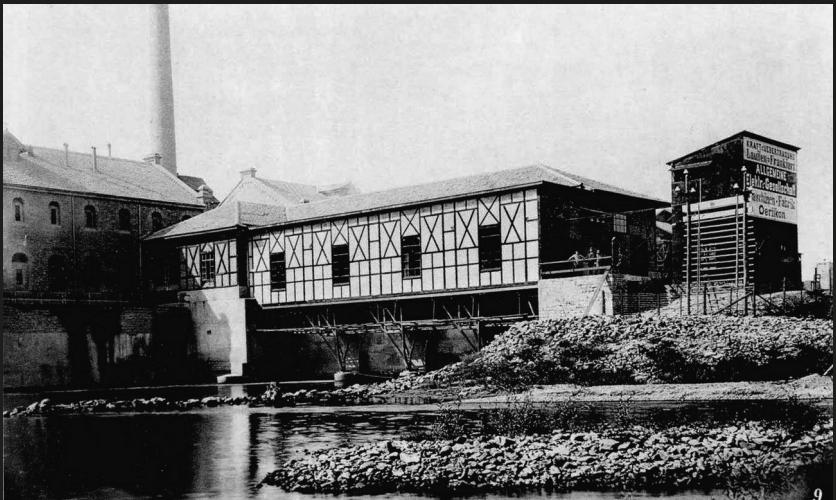
Pendant ce temps, les inventions basées sur l'électro-magnétisme se développent.



Pile électrique  
A. Volta, 1799



Machine à induction  
H. Pixii, 1832



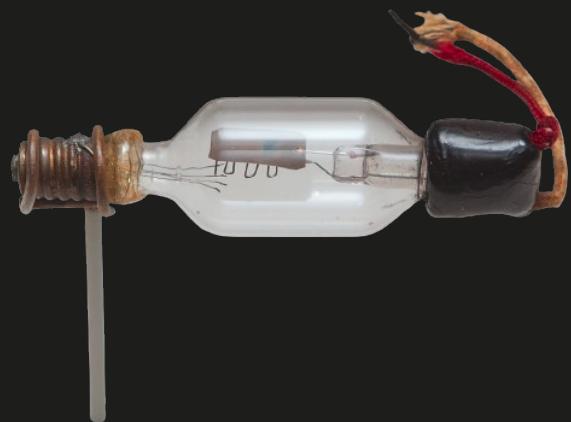
Ligne haute tension (Mühlgraben-Francfort)  
Usine de Mühlgraben, 1891

La découverte de l'électron en 1897-1897 par Joseph John Thomson ouvre un champ d'application largement utilisé pour l'électronique du début du XX<sup>e</sup> siècle.

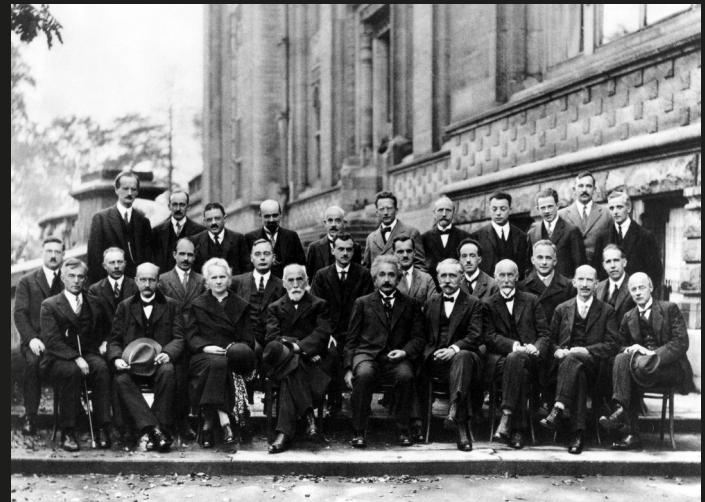
Les tubes à vide sont à cette époque les seuls éléments capables de contrôler le flux électrique dans l'optique de traiter de l'information.



John Ambrose Fleming  
Inventeur du tube à vide  
(diode, 1904)

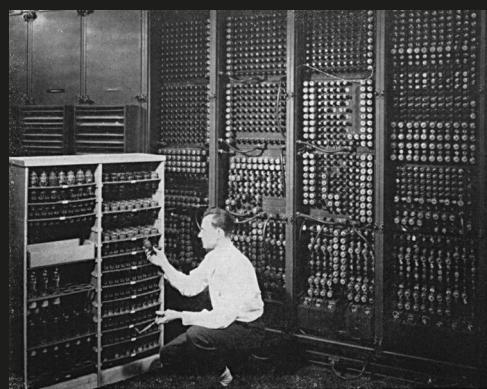
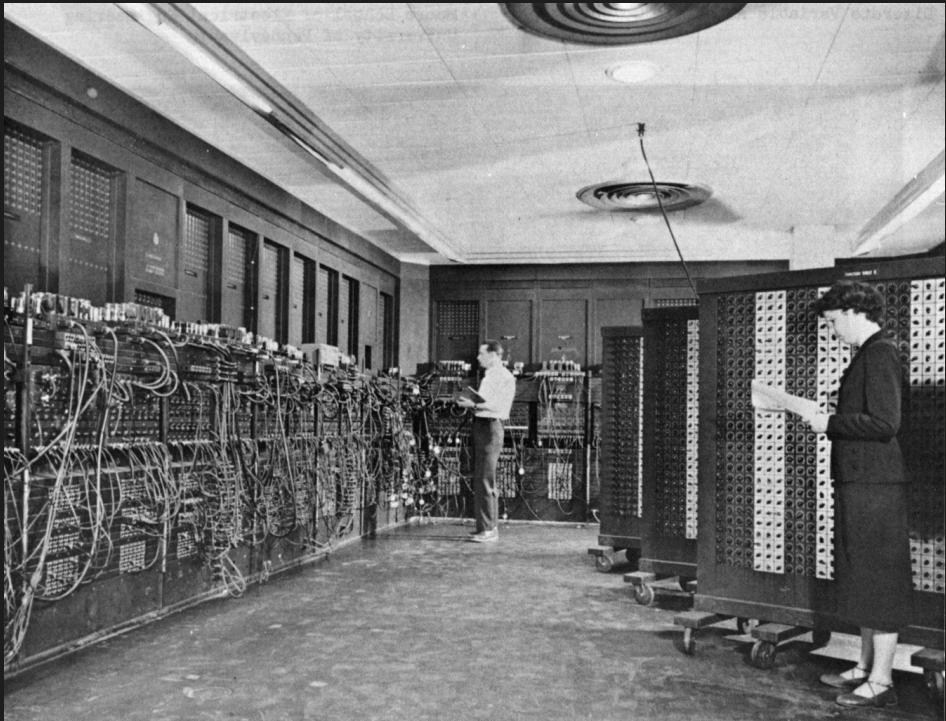


Audion (triode)  
Lee de Forest (1906)



5ème congrès de Solvay, 1927  
Thème « électrons et photons »  
Sur 29 participants, 17 prix Nobel 9

L'apothéose des tubes à vide fut la constitution de l'**ENIAC** (*Electronic Numerical Integrator And Computer*) en 1945, premier ordinateur **entièlement électronique**.



Les six premières personnes à programmer l'**ENIAC** sont des mathématiciennes (entre 1944 et 1955).

17 468 tubes à vide  
7 200 diodes  
70k résistances + 10k condo  
5M de soudures à la main  
167 m<sup>2</sup>, 30 tonnes  
150 kW

100 000 add/s  
357 mul/s  
38 div/s  
116 h : plus longue période sans défaillance

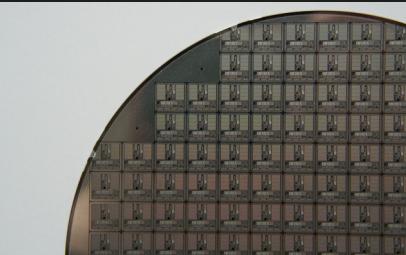
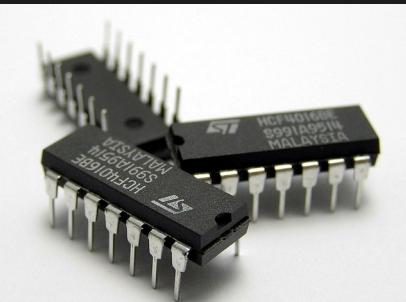
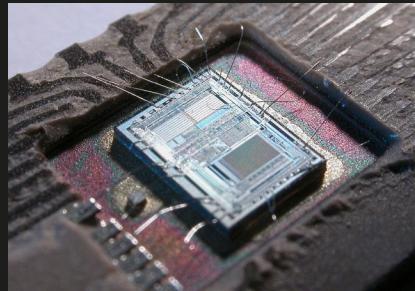
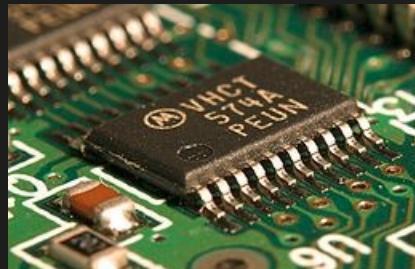
## L'avènement des semi-conducteurs

La découverte et la maîtrise technique des semi-conducteurs ont sûrement mené à l'une des révolutions majeures de l'humanité : après l'agriculture et l'énergie, voici l'information à la portée de tous.

Le **transistor bipolaire (1947)** et le **transistor à effet de champ (1960)** sont la clé de voûte de tout système numérique, notamment grâce aux **circuits intégrés** (TI, 1958).



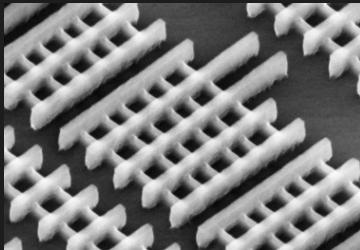
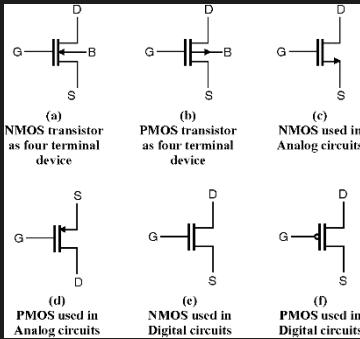
Premier transistor bipolaire (1947)  
Bardeen<sup>2</sup>, Schokley, Brattain (Bell Labs), Nobel en 1956



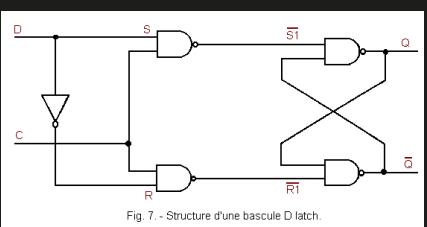
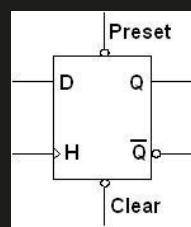
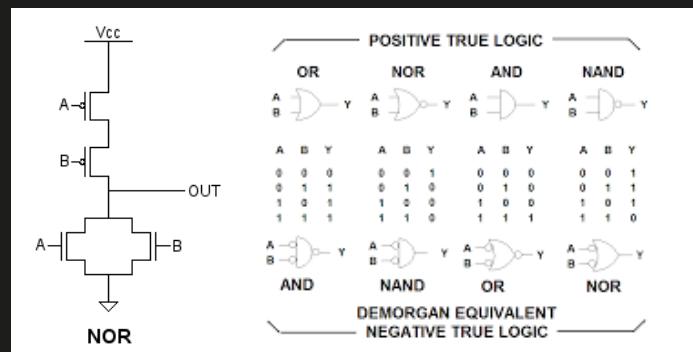
Les circuits numériques s'appuient sur l'utilisation de **transistors en commutation**.

Les transistors sont assemblés pour construire des portes logiques et bascules, elles-mêmes combinées pour créer des fonctions spécialisées.

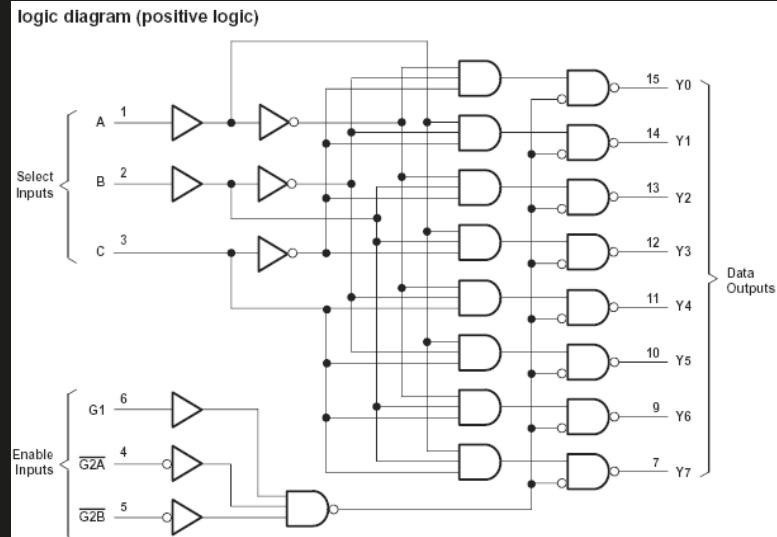
### MOS Transistors



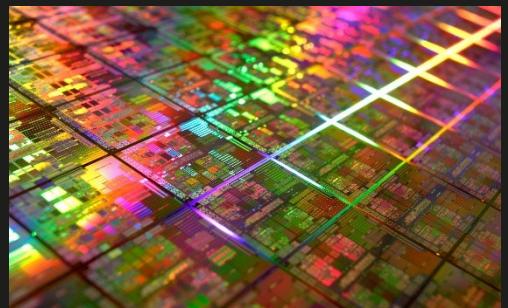
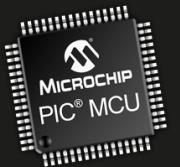
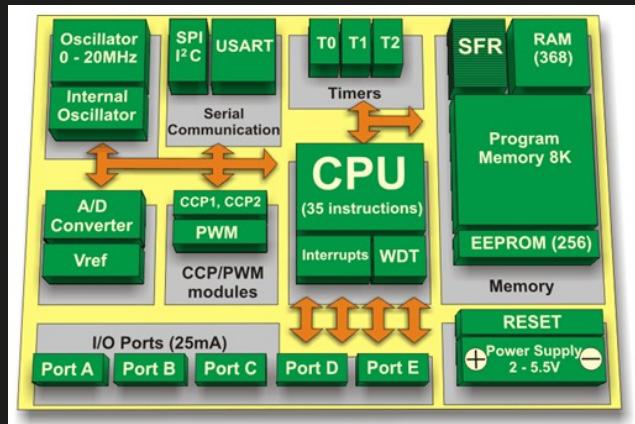
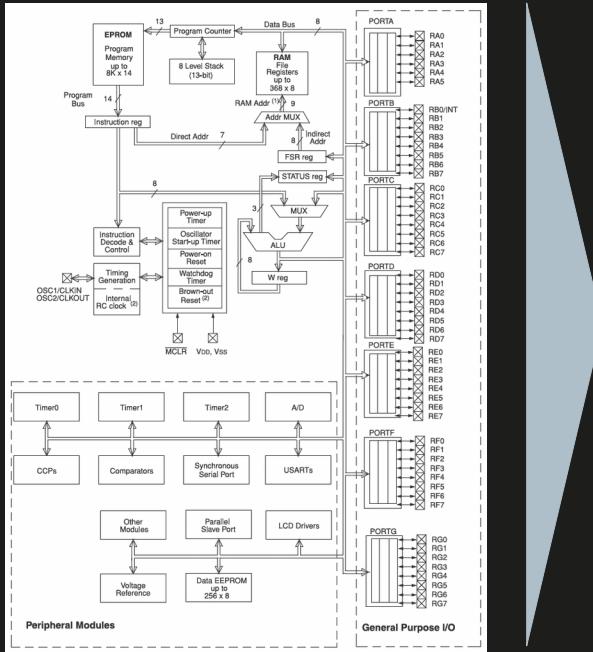
### Logic gates, latches



### Specialized circuit



Les blocs spécialistes peuvent être construits dans un **Circuit Intégré (Integrated Circuit, IC)**. Associés à d'autres fonctions via des bus de communication, ils peuvent former des circuits encore plus complexes, comme des processeurs.



## Premier processeur

Le premier processeur commercial est le **4004**, annoncé par **Intel** le 15 novembre **1971**.

En réalité, l'armée américaine avait déjà développé un processeur en juin 1970, gardé secret pour le F-14.

À titre de comparaison, la mission Apollo 11 s'est déroulée deux ans avant !

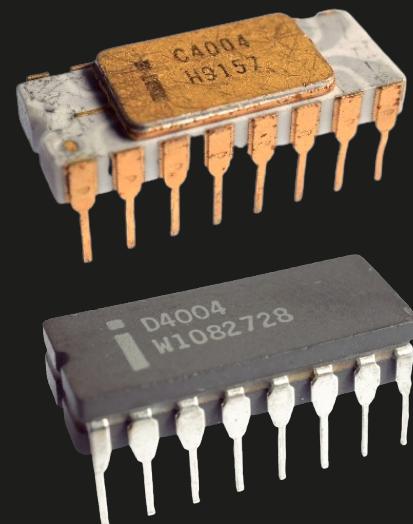
Le 4004 possède 2 300 transistors gravés en 10 µm.

C'est un processeur 4 bits, à 16 broches.

Son ISA compte 45 instructions,  
dont du saut conditionnel et de l'appel de fonction.

Cadencé à 740 kHz, il peut alors réaliser 90 kIPS.

Le tout pour la modique somme de 60 \$ !



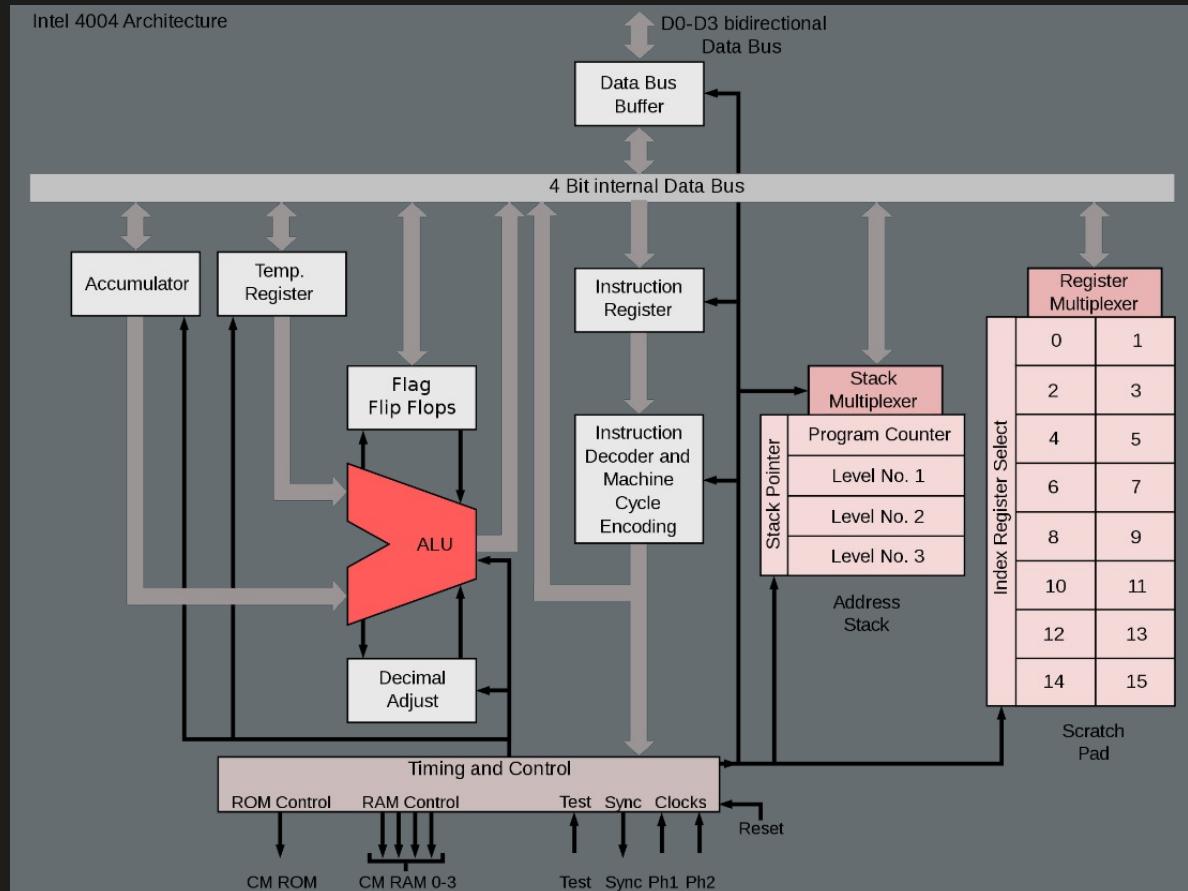
## Intel 4004

L'architecture du 4004 reste la base de tous les processeurs modernes.

À comparer avec le Microchip PIC18 vu en première année !

Pour les fans de transistors, le schéma est visible ici :

<https://www.framboise314.fr/le-micro-processeur-a-50-ans-intel-4004/>



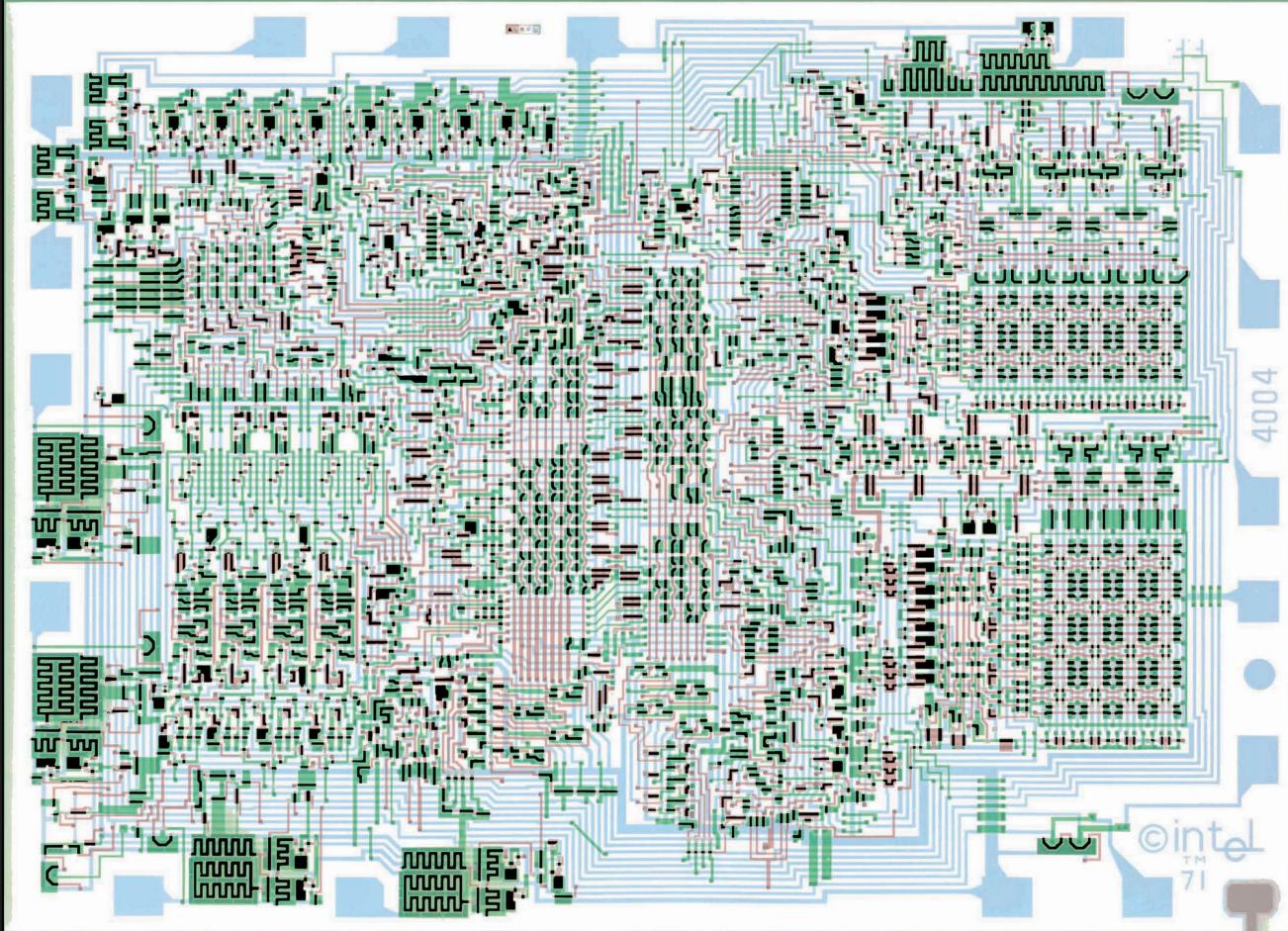


Schéma d'implantation.

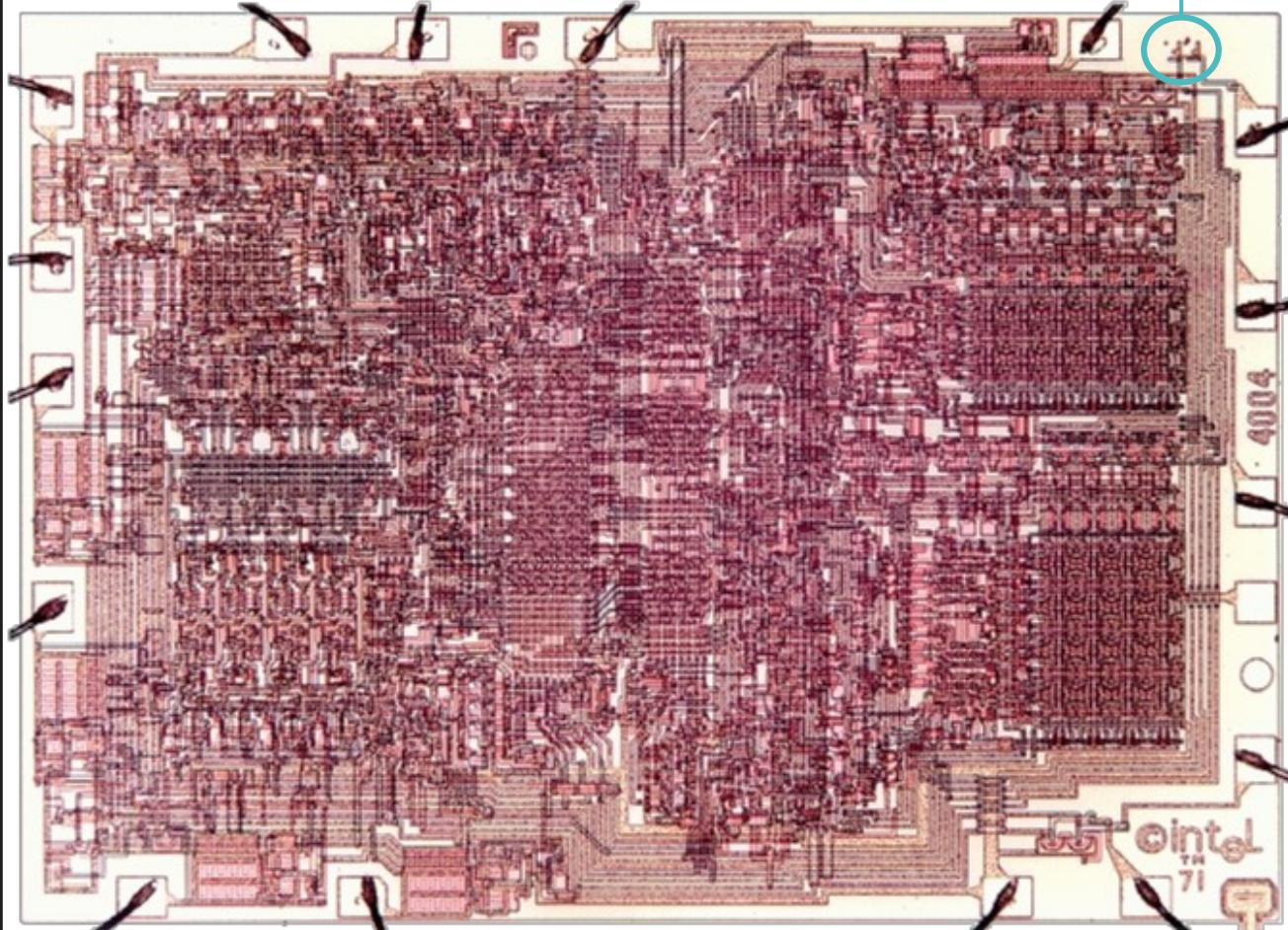
Objectif :

Polariser, assembler les 2300 transistors pour réaliser les différentes fonctions logiques.

À l'époque, pas d'outil informatique : tout ce travail est fait à la main !

# DIVERSITÉ DES ARCHITECTURES PROCESSEUR

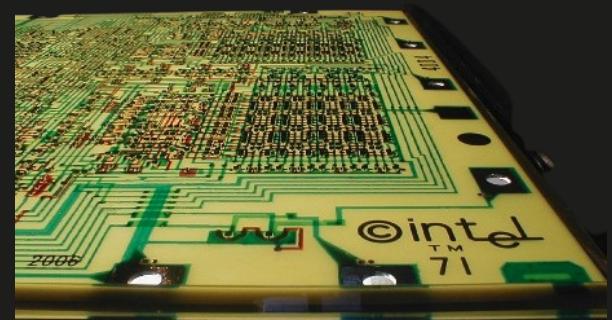
Intel 4004



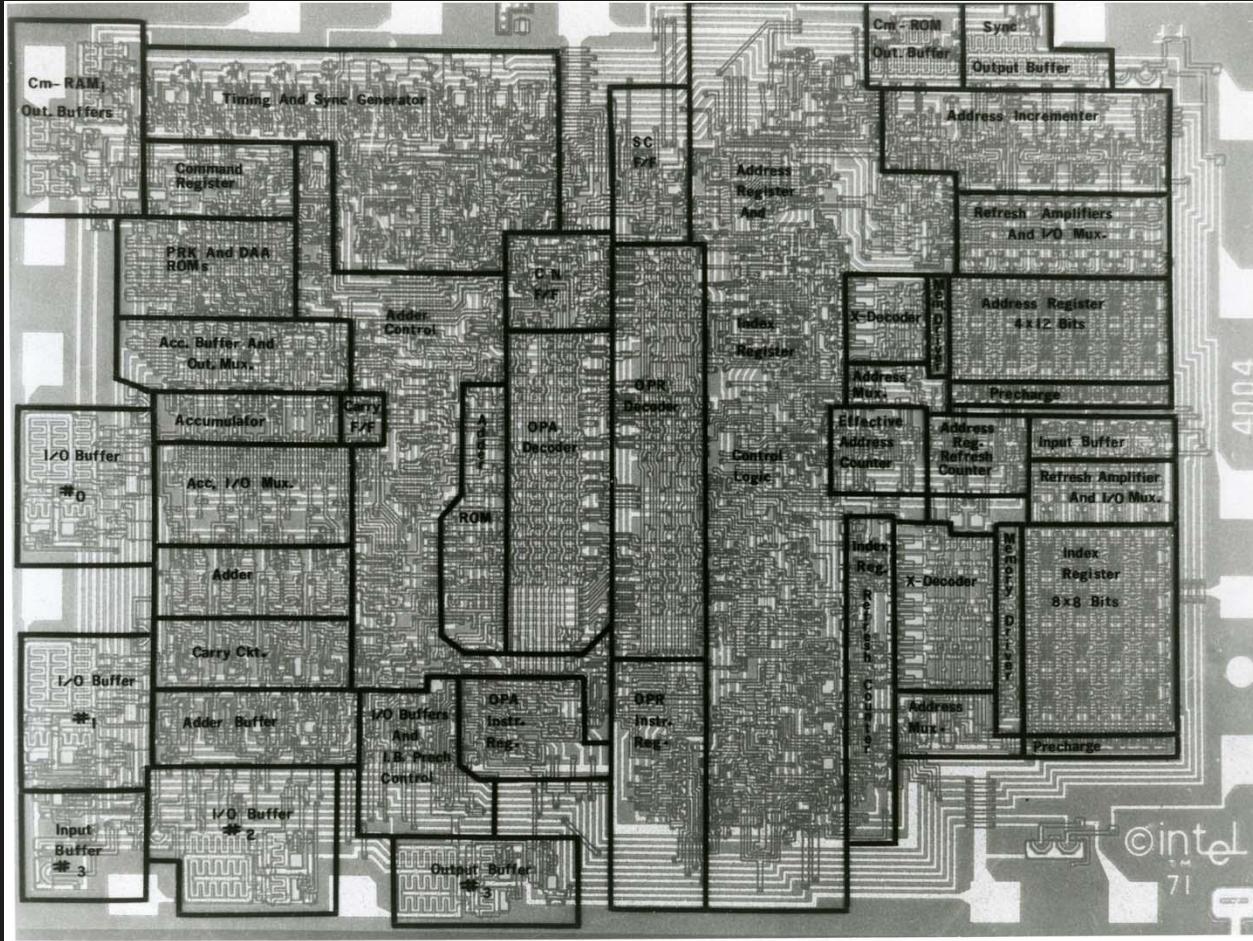
Initiales de Federico Faggin,  
concepteur du 4004, 8008,  
4040 et 8080 !

Photographie par MEB.

On voit encore les fils  
d'or reliant les *pads* du  
*die* aux broches du  
boîtier.



## Intel 4004



Découpage du schéma d'implantation en blocs fonctionnels.

Simulation à 6 cycles par seconde (90 kIPS IRL) :

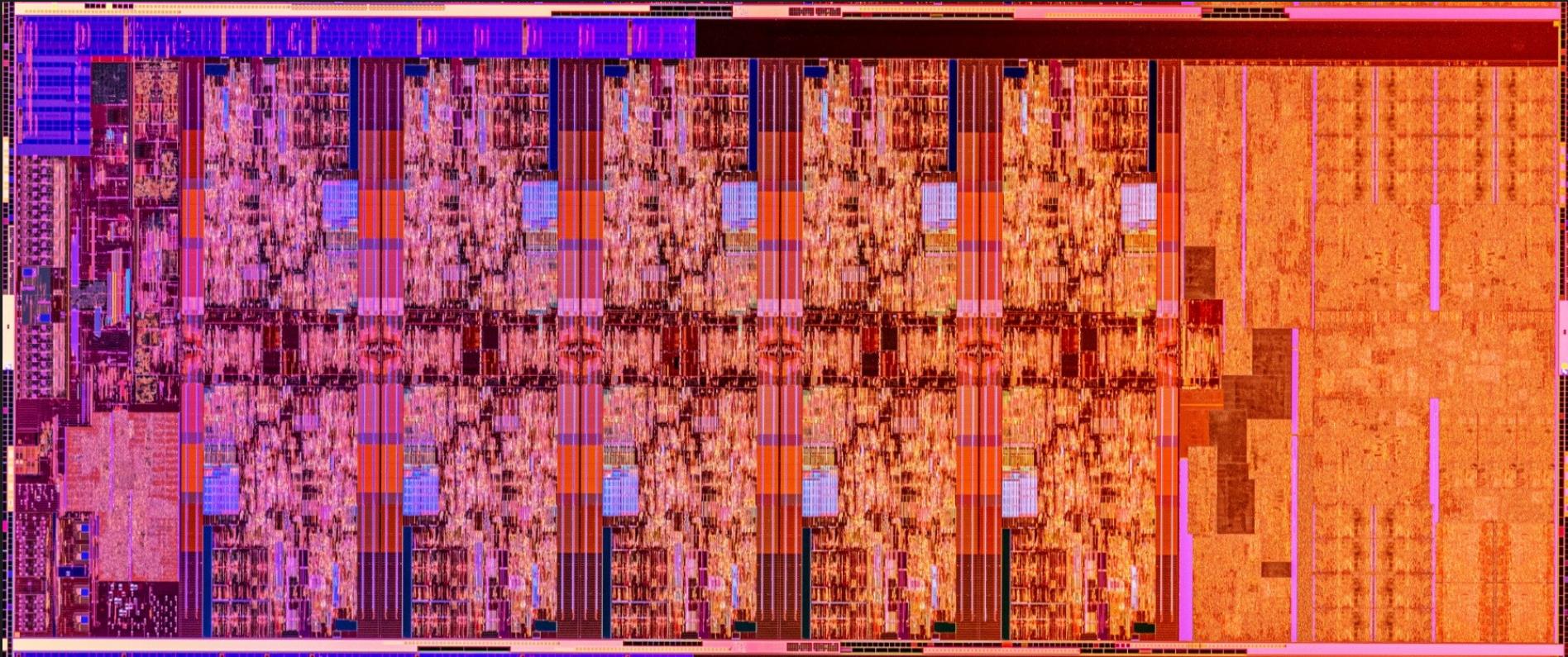
<https://www.youtube.com/watch?v=0Fixr39X8S4>  
(48 s de vidéo = 0.4 ms IRL)

## Intel Core 10<sup>e</sup> génération

<https://www.intel.com/content/www/us/en/products/sku/199332/intel-core-i910900k-processor-20m-cache-up-to-5-30-ghz/specifications.html>

### Intel® Core™ i9-10900K Processor

Q2'2020, 10<sup>th</sup> generation 'Comet Lake', ~\$ 540  
206 mm<sup>2</sup> die, 14 nm (estimated 7 billions transistors)  
10 cores, 20 threads, 3.7/5.3 GHz, 460.8 Gflops, 125 W



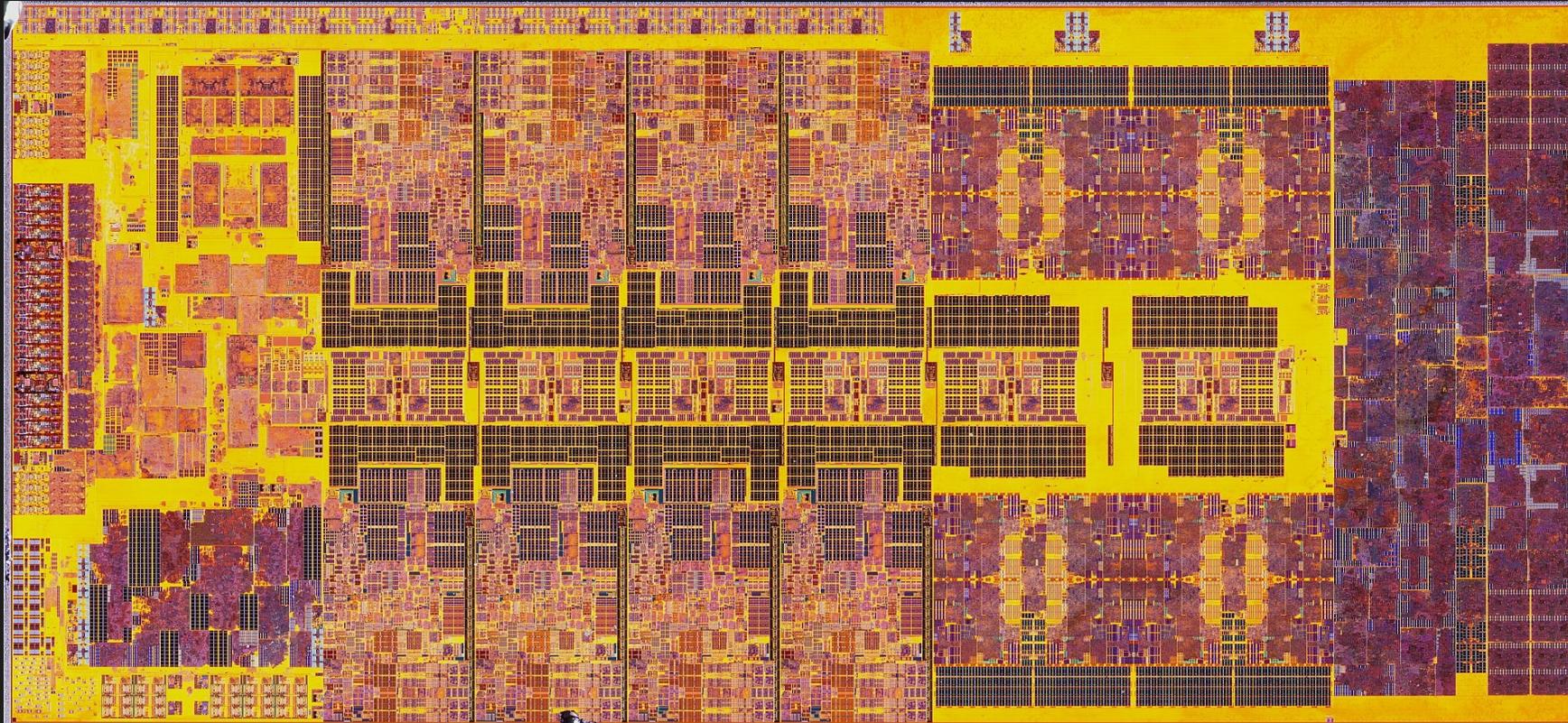
# DIVERSITÉ DES ARCHITECTURES PROCESSEUR

## Intel Core 13<sup>e</sup> génération

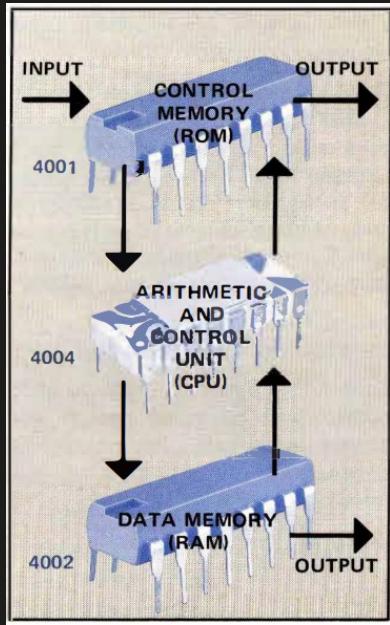
<https://www.intel.com/content/www/us/en/products/sku/230496/intel-core-i913900k-processor-36m-cache-up-to-5-80-ghz/specifications.html>

### Intel® Core™ i9-13900K Processor

Q3'2022, 13<sup>th</sup> generation 'Raptor Lake', ~\$ 589  
257 mm<sup>2</sup> die, Intel 7 nm lithography  
24 cores, 32 threads, 3.0/5.8 GHz, ? Gflops, 125 W



Le 4004 a été conçu pour une machine à calculer de *Busicom Corporation (la 141-PF)*. Il est alors associé à d'autres composants pour former le **chipset Intel MCS-4**.

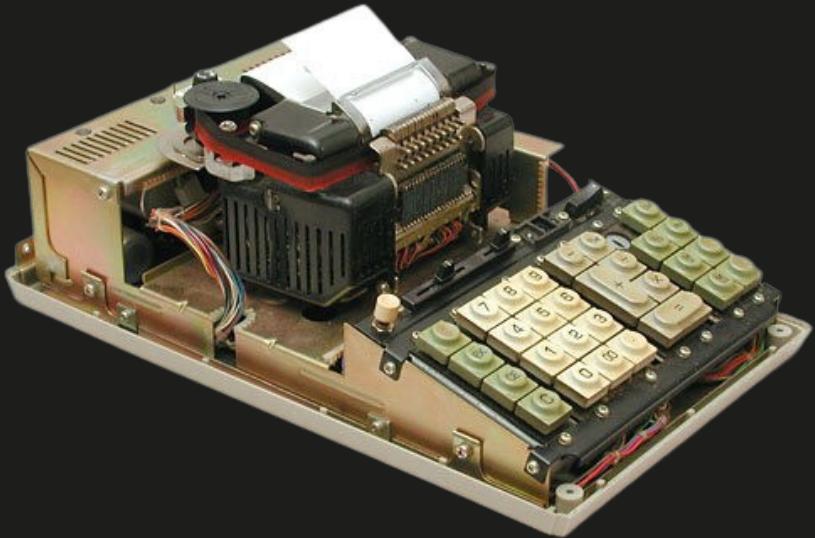


- 4001 : 256 x 8 bit ROM
- 4002 : 320 bit RAM
- 4003 : 10 bit shift register
- 4004 : 4 bit CPU



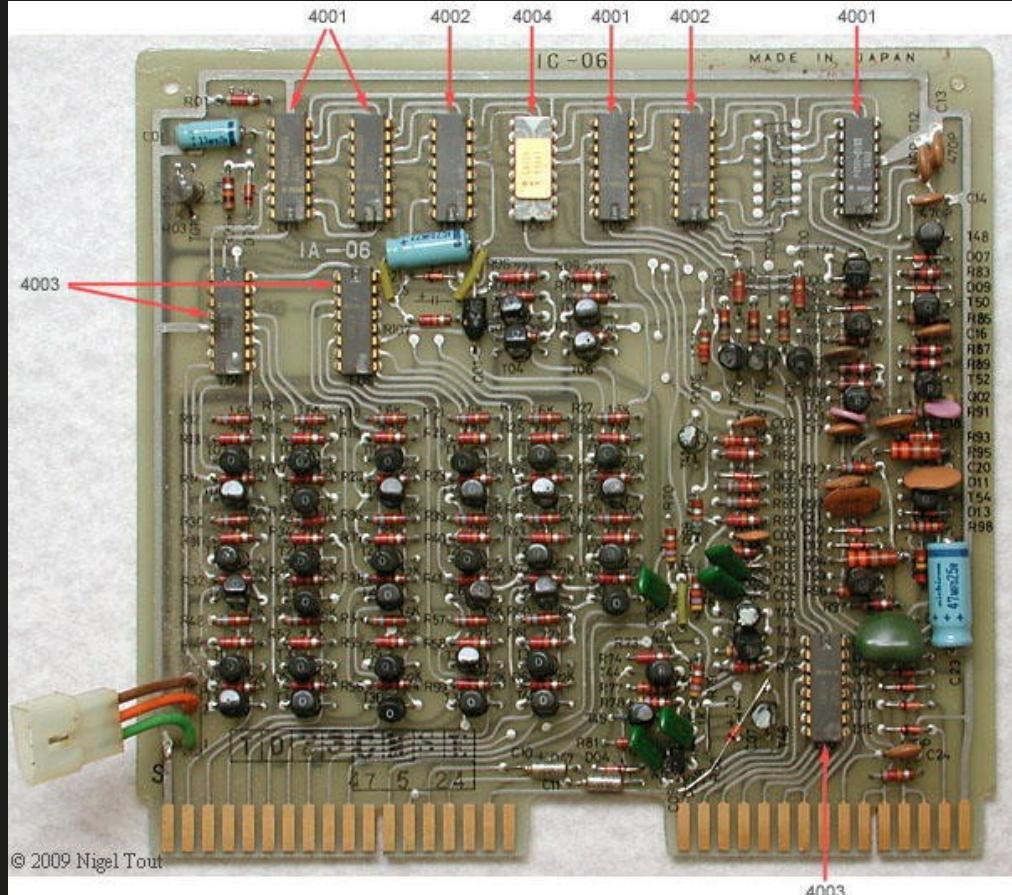
# DIVERSITÉ DES ARCHITECTURES PROCESSEUR

Intel 4004



Busicom 141-PF (ou NCR-18-36)

Bloc d'alimentation au fond,  
PCB unique en dessous.



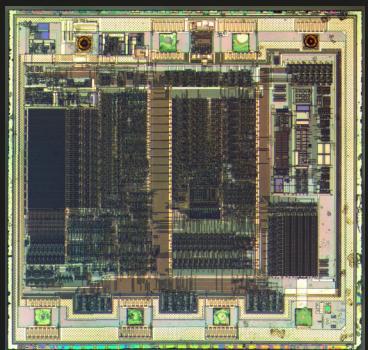
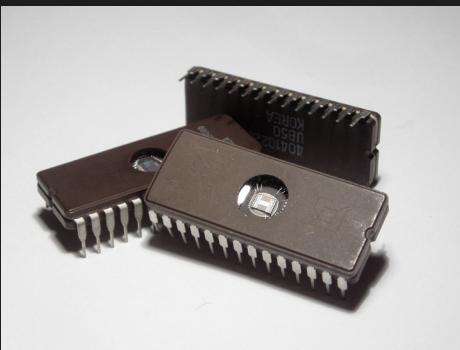
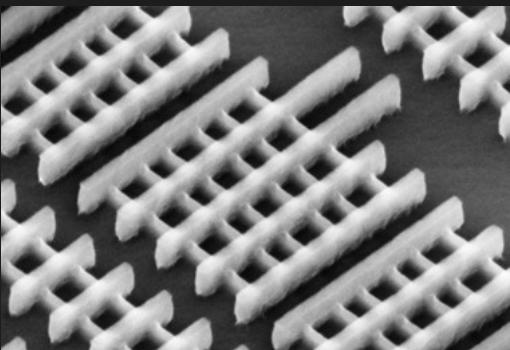
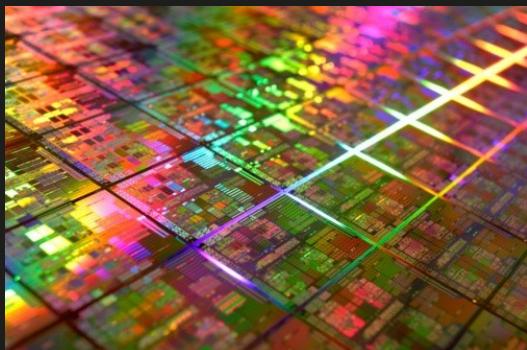
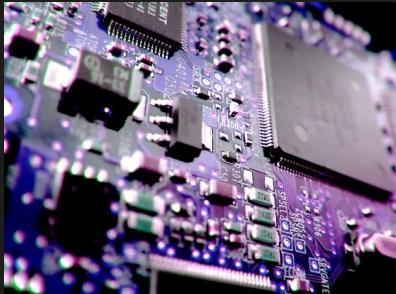
© 2009 Nigel Tout

Source : [http://www.vintagecalculators.com/html/busicom\\_141-pf.html](http://www.vintagecalculators.com/html/busicom_141-pf.html)

## Histoire du processeur

Depuis, les processeurs ont évolués suivant un processus de sélection naturelle.

Ceux répondant à des besoins spécifiques se sont développés et améliorés, tandis que d'autres ont disparu des marchés et laboratoires de recherche.



## Histoire du processeur

Comme pour le domaine du vivant, le processus d'évolution sera toujours en cours.  
De nouvelles architectures de processeur pourraient apparaître dans le futur proche !



En attendant, regardons les principales architectures utilisées aujourd'hui.

## Architectures généralistes

*Processeurs de contrôle*

MCU	AP	GPP
Micro Controller Unit	Application Processor	General Purpose Processor

**CONTROL**

Computer →

## Architectures hybrides

### SoC / SoB

System  
on  
Chip / Board

- FPGA-AP
- FPGA-MCU
- GPP-GPU
- AP
- MCU-analog

## Architectures spécialisées

*Coprocesseurs ou processeurs de calcul*

FPGA	DSP	(GP) GPU
Field Programmable Gate Array	Digital Signal Processor	Graphics Processing Unit

**CALCULUS**

## Architectures généralistes

*Processeurs de contrôle*

## Architectures hybrides

## Architectures spécialisées

*Coprocresseurs ou processeurs de calcul*

**MCU**

Micro  
Controller  
Unit

**AP**

Application  
Processor

**GPP**

General  
Purpose  
Processor

**SoC / SoB**

System  
on  
Chip / Board

**FPGA**

Field  
Programmable  
Gate Array

**DSP**

Digital  
Signal  
Processor

**(GP) GPU**

Graphics  
Processing  
Unit

**CPU**

- FPGA-AP
- FPGA-MCU
- GPP-GPU
- AP
- MCU-analog

**LOGIC**

**CPU**





# - MCU -

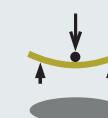
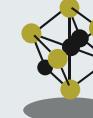
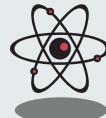
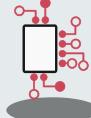
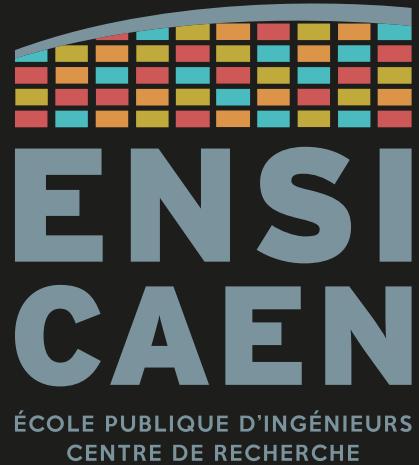
## MICROCONTROLLER UNIT

Applications

Architectures

Fabricants et produits

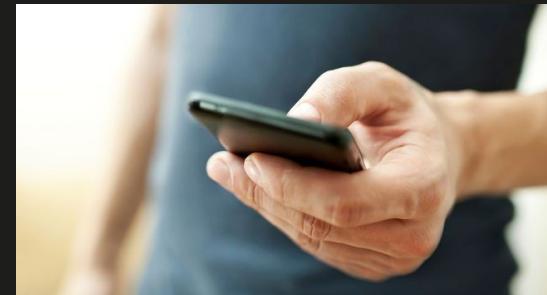
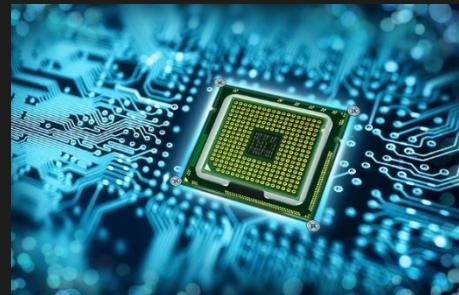
Parts de marché



## Applications

Les micro-contrôleurs (MCU, *Microcontroller Units*) sont les processeurs les plus répandus dans notre environnement.

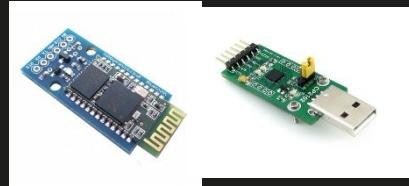
De près ou de loin, nous utilisons environ 200 processeurs par jour !



## Applications

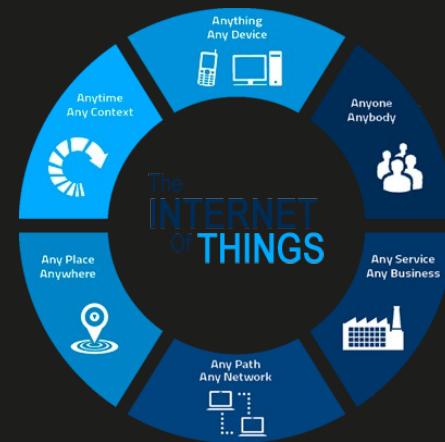
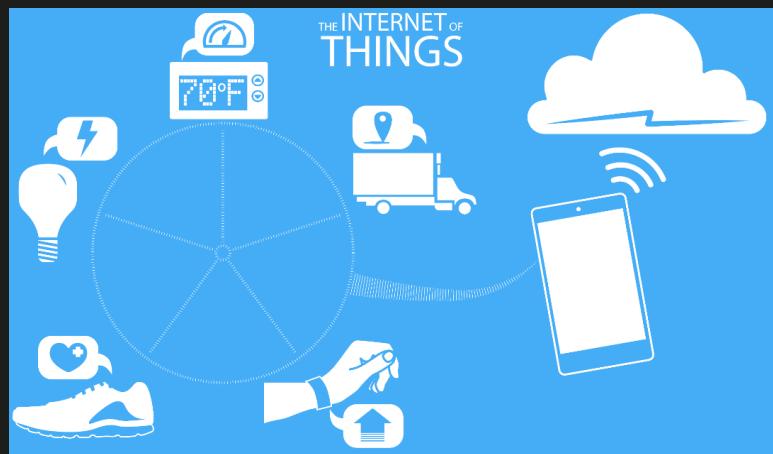
Les micro-contrôleurs sont des processus dédiés à la supervision des systèmes électroniques. Ils contrôlent leur environnement via leurs interfaces et leur firmware embarqué développé pour une application spécifique.

Ils ciblent des marchés où les applications sont faible coût, faible consommation, faible encombrement et gros volumes de production.



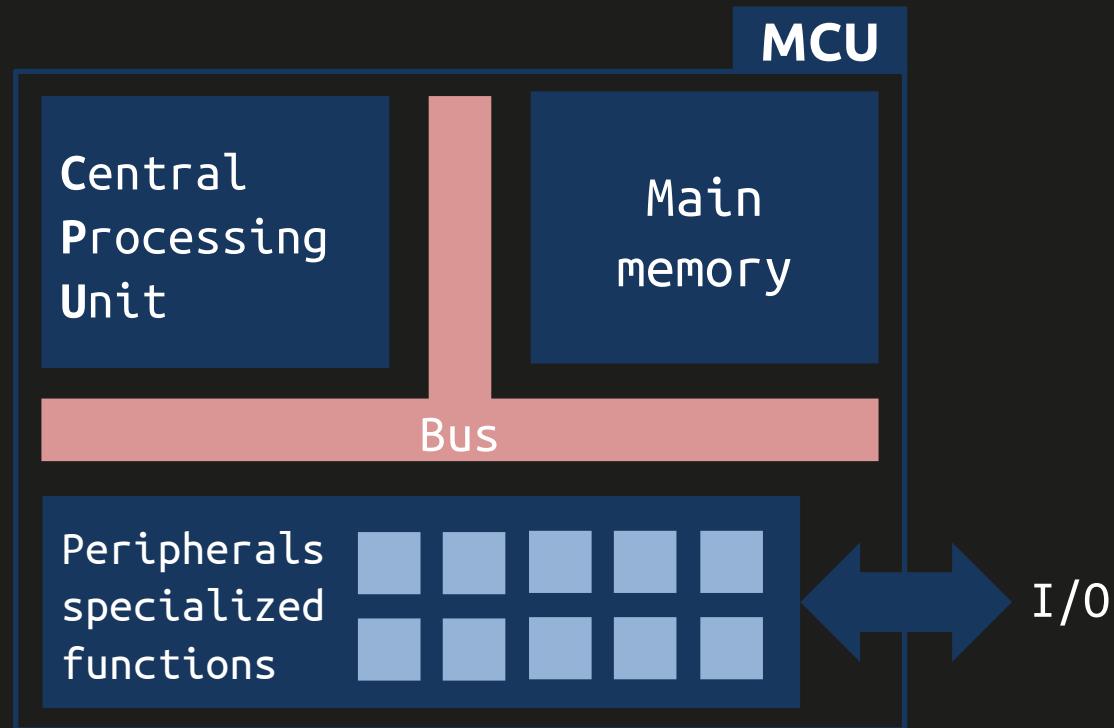
L'un des marchés phares actuels des MCU est celui des objets connectés ([IoT ou \*Internet of Things\*](#)). L'IoT représente l'extension d'Internet à des objets et lieux du monde physique. Il est considéré comme la troisième évolution d'Internet et, à ce titre, a été baptisé « Web 3.0 ».

Avec 3,6 milliards de connexions actives en 2015, 11,7 milliards en 2020 et 30 milliards prévues en 2025, l'IoT représentait 18 % des MCU en 2019 et 29 % en 2025.



Ces processeurs sont des systèmes numériques intégrés sur puce.

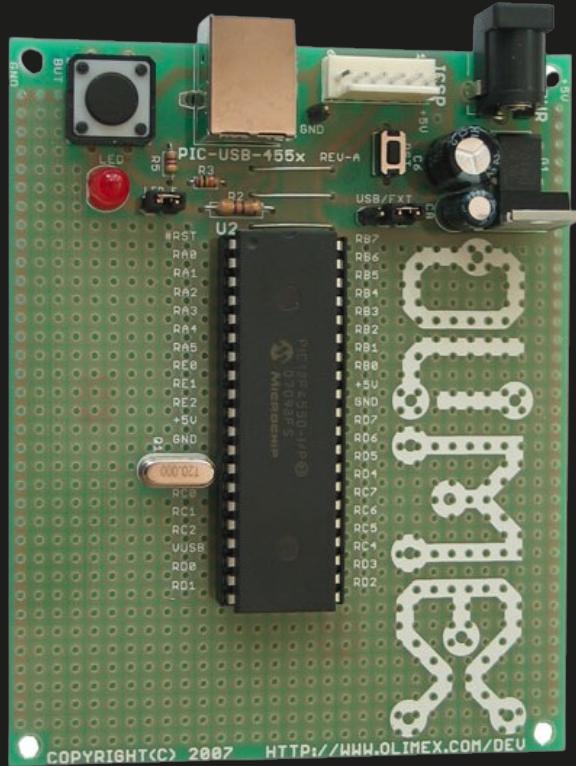
Ils sont pensés pour être autonomes (pas besoin de RAM, de HDD, ...).



## Schéma et carte

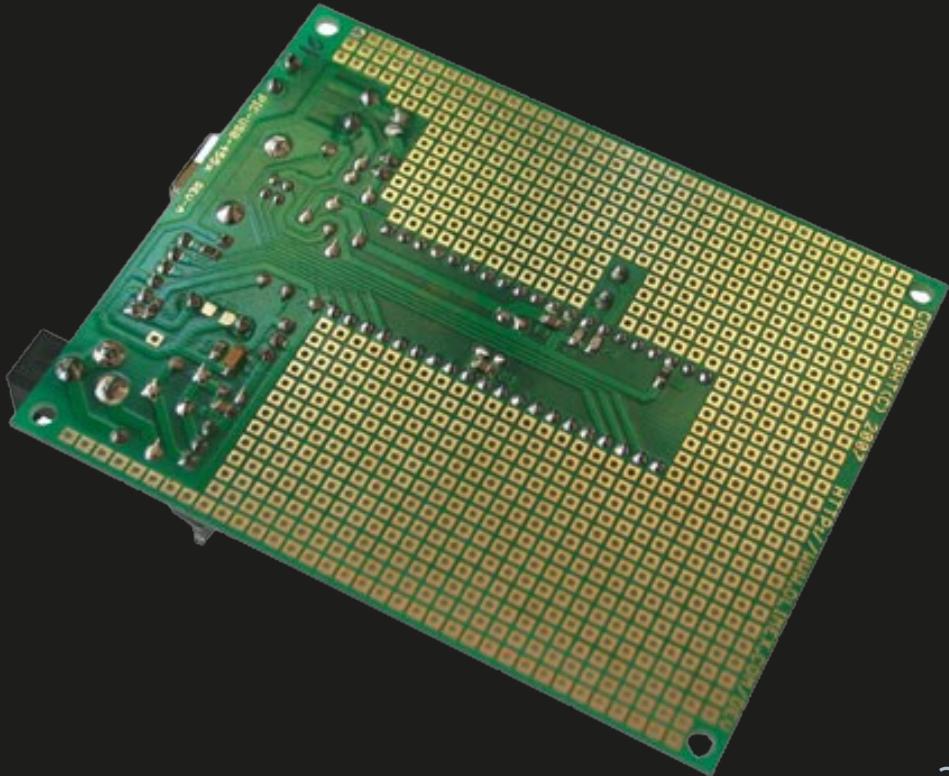
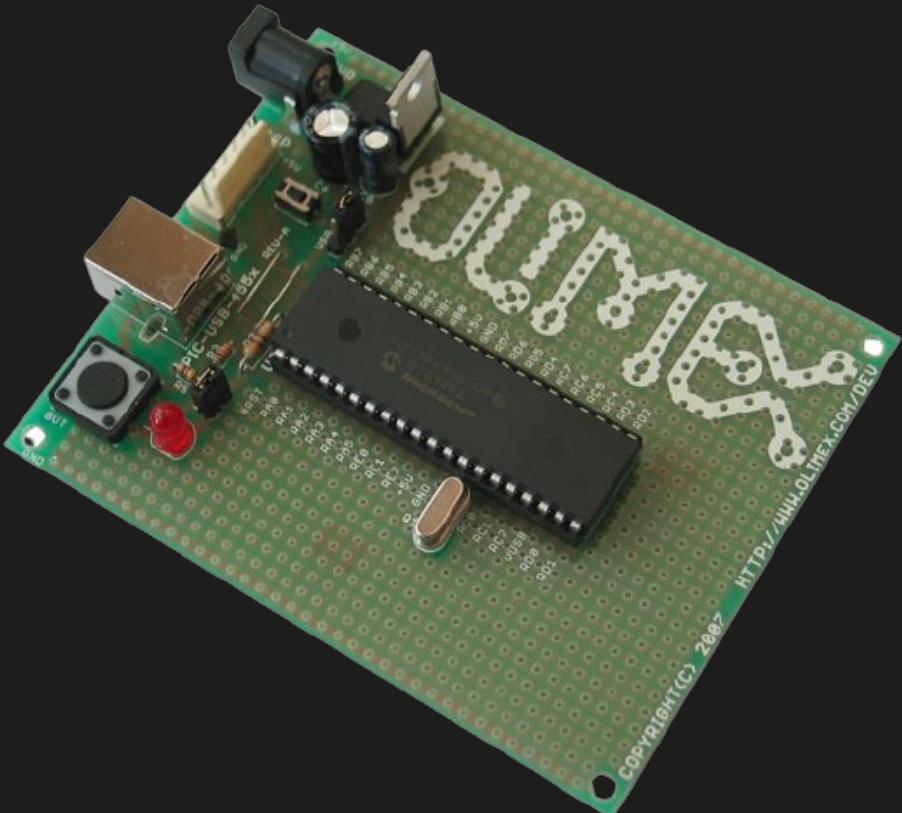
Exemple de schéma utilisant un PIC18 de Microchip.

Olimex PIC-USB-4550 board.



## Schéma et carte

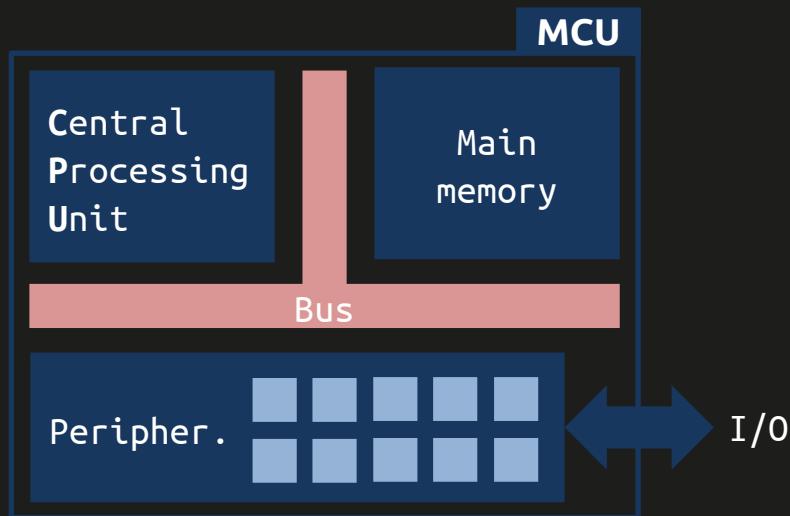
**Exercice : repérez les composants du schéma précédent sur les photos ci-dessous.**



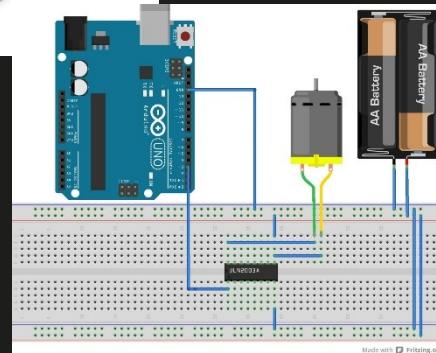
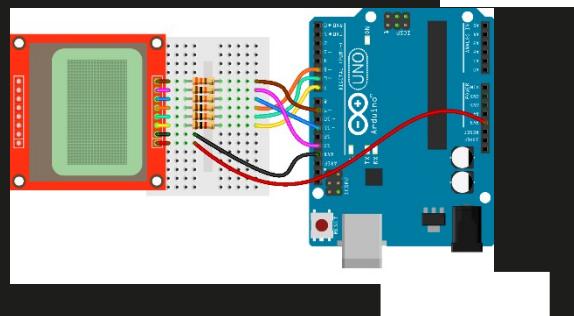
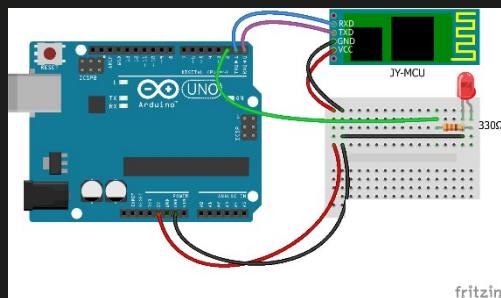
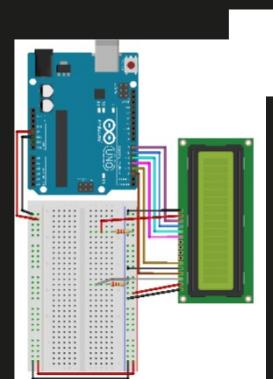
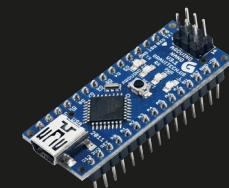
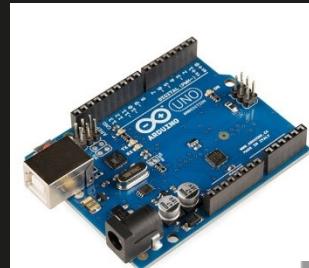


Il existe un très grand nombre de solutions MCU chez différents fondeurs, permettant de résoudre un cahier des charges.

Les MCU d'une même famille sont caractérisés par le même CPU et bus associés. Le **jeu d'instructions (ISA, Instruction Set Architecture)** et donc les outils de compilation sont similaires. Ce qui différencie les MCU d'une même famille sera le jeu de périphériques associés et les ressources mémoire disponibles.

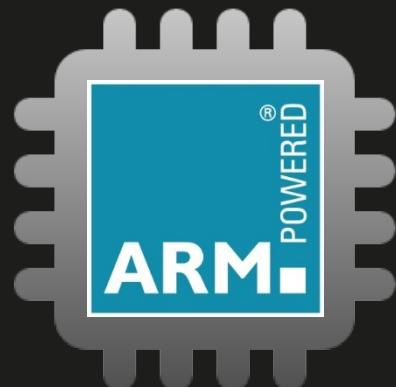


Sûrement le plus populaire des projets électroniques basés sur un MCU, il reste déprécié en enseignements ingénieurs pour son côté trop *friendly/maker* et sa non-application aux marchés en sortie d'école.



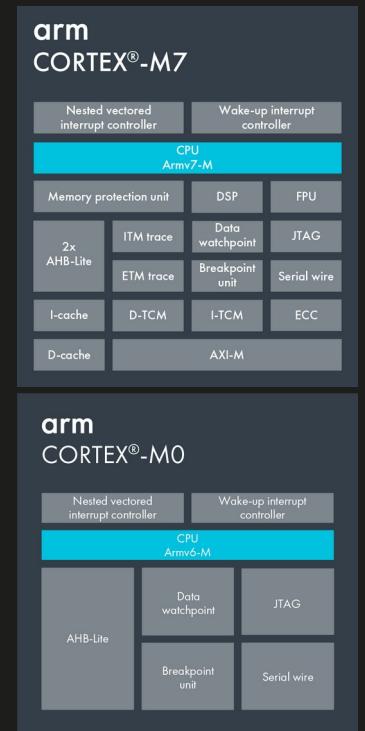
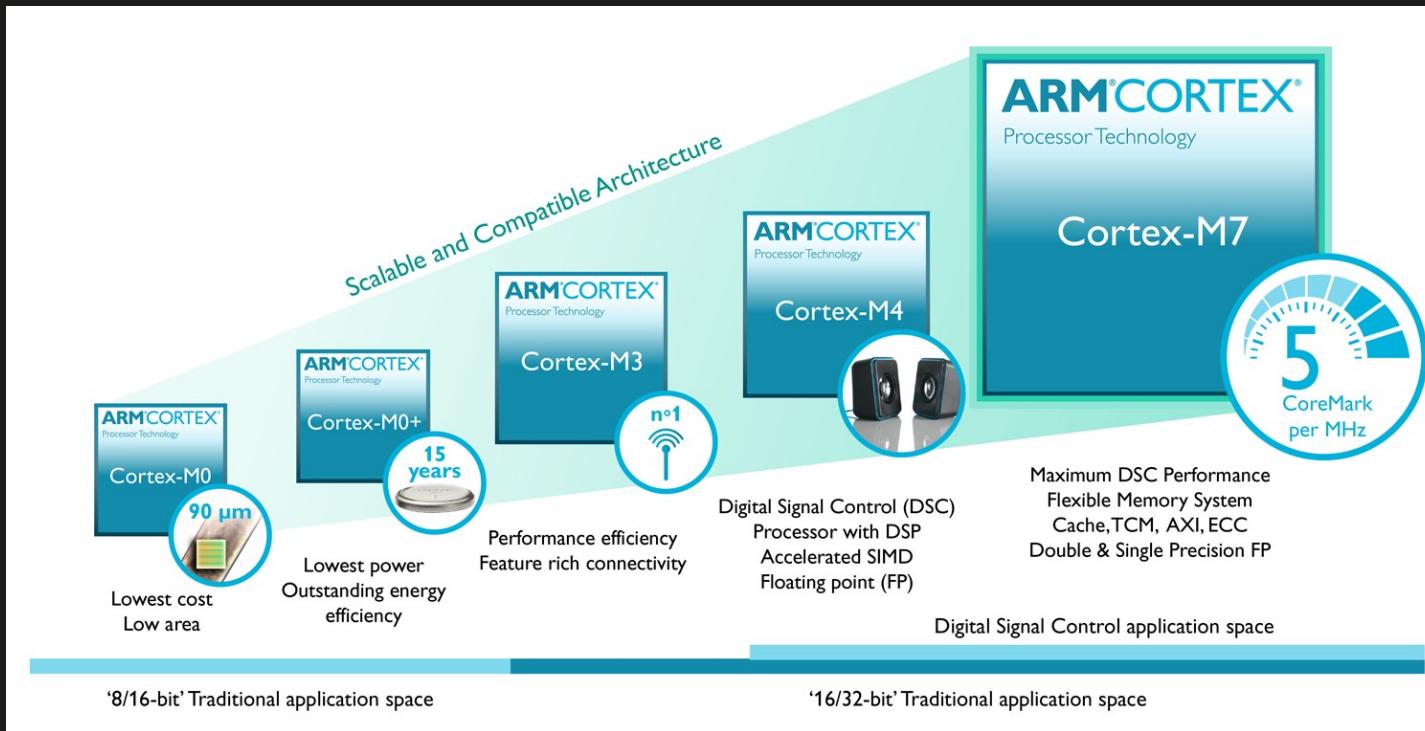
Même si le marché des MCU reste concurrentiel, la grande majorité des fondeurs de MCU (STMicro, Renesas, Texas Instruments, NXP, ...) utilisent des architectures CPU similaires, toutes proposées par la société ARM : la famille des **Cortex-M**.

Cela garanti un accès à des outils de développement, bibliothèques et services logiciels fiables, pouvant être libres et open-source (IP / *Graphical* / USB / Bluetooth, *stack*, RTOS, ...).



ARM propose la série des processeurs Cortex-M, où M signifie MCU.

Cette série comporte toute une famille de cœurs pour MCU adaptée à un large choix d'application.



Observons à titre d'illustration les gammes des STM32, qui sont des MCU 32-bits basés sur un cœur ARM Cortex-M.

Ils sont proposés par la société STMicroelectronics, société franco-italienne et principal fondeur européen.



Common core peripherals and architecture:

Communication peripherals:  
USART, SPI, I<sup>2</sup>C

Multiple general-purpose timers

Integrated reset and brown-out warning

Multiple DMA

2x watchdogs  
Real-time clock

Integrated regulator PLL and clock circuit

External memory interface (FSMC)

Up to 3x 12-bit DAC

Up to 4x 12-bit ADC (Up to 5 MSPS)

Main oscillator and 32 kHz oscillator

Low-speed and high-speed internal RC oscillators

-40 to +85 °C and up to 105 °C operating temperature range

Low voltage 2.0 to 3.6 V or 1.65/1.7 to 3.6 V (depending on series)

Temperature sensor

STM32 F4 series - High performance with DSP (STM32F405/415/407/417)

168 MHz Cortex-M4 with DSP and FPU	Up to 192-Kbyte SRAM	Up to 1-Mbyte Flash	2x USB 2.0 OTG FS/HS	3-phase MC timer	2x CAN 2.0B	SDIO 2x I <sup>2</sup> S audio Camera IF	Ethernet IEEE 1588	Crypto/ hash processor and RNG
------------------------------------	----------------------	---------------------	----------------------	------------------	-------------	--	--------------------	--------------------------------



STM32 F3 series - Mixed-signal with DSP (STM32F302/303/313/372/373/383)

72 MHz Cortex-M4 with DSP and FPU	Up to 48-Kbyte SRAM & CCM-SRAM	Up to 256-Kbyte Flash	USB 2.0 FS	2x 3-phase MC timer (144 MHz)	CAN 2.0B	Up to 7x comparator	3x 16-bit ΣΔ ADC	4x PGA
-----------------------------------	--------------------------------	-----------------------	------------	-------------------------------	----------	---------------------	------------------	--------



STM32 F2 series - High performance (STM32F205/215/207/217)

120 MHz Cortex-M3 CPU	Up to 128-Kbyte SRAM	Up to 1-Mbyte Flash	2x USB 2.0 OTG FS/HS	3-phase MC timer	2x CAN 2.0B	SDIO 2x I <sup>2</sup> S audio Camera IF	Ethernet IEEE 1588	Crypto/ hash processor and RNG
-----------------------	----------------------	---------------------	----------------------	------------------	-------------	--	--------------------	--------------------------------



STM32 F1 series - Mainstream - 5 product lines (STM32F100/101/102/103 and 105/107)

Up to 72 MHz Cortex-M3 CPU	Up to 96-Kbyte SRAM	Up to 1-Mbyte Flash	USB 2.0 OTG FS	3-phase MC timer	Up to 2x CAN 2.0B	SDIO 2x I <sup>2</sup> S audio	Ethernet IEEE 1588	
----------------------------	---------------------	---------------------	----------------	------------------	-------------------	--------------------------------	--------------------	--



STM32 F0 series – Entry level (STM32F050/051)

48 MHz Cortex-M0 CPU	Up to 12-Kbyte SRAM	Up to 128-Kbyte Flash	3-phase MC timer	Comparator	CEC			
----------------------	---------------------	-----------------------	------------------	------------	-----	--	--	--



STM32 L1 series - Ultra-low-power (STM32L151/152/162)

32 MHz Cortex-M3 CPU	Up to 48-Kbyte SRAM	Up to 384-Kbyte Flash	USB FS device	Up to 12-Kbyte EEPROM	LCD 8x40 4x44	Comparator	BOR MSI VScal	AES 128-bit
----------------------	---------------------	-----------------------	---------------	-----------------------	---------------	------------	---------------	-------------

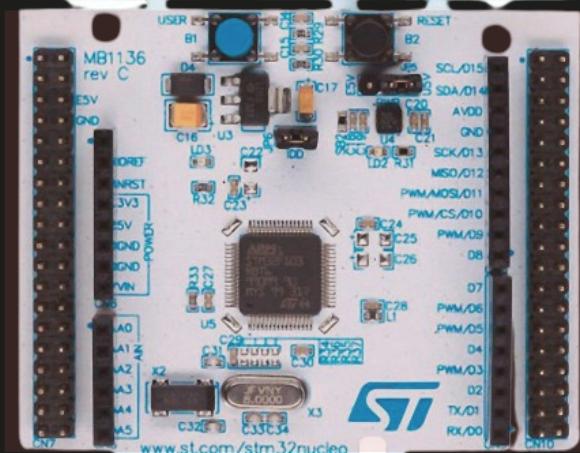
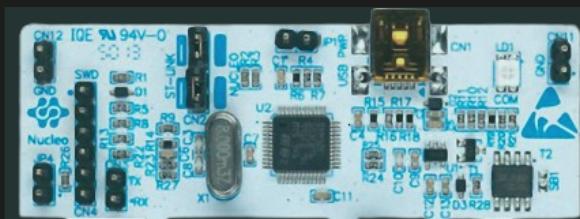
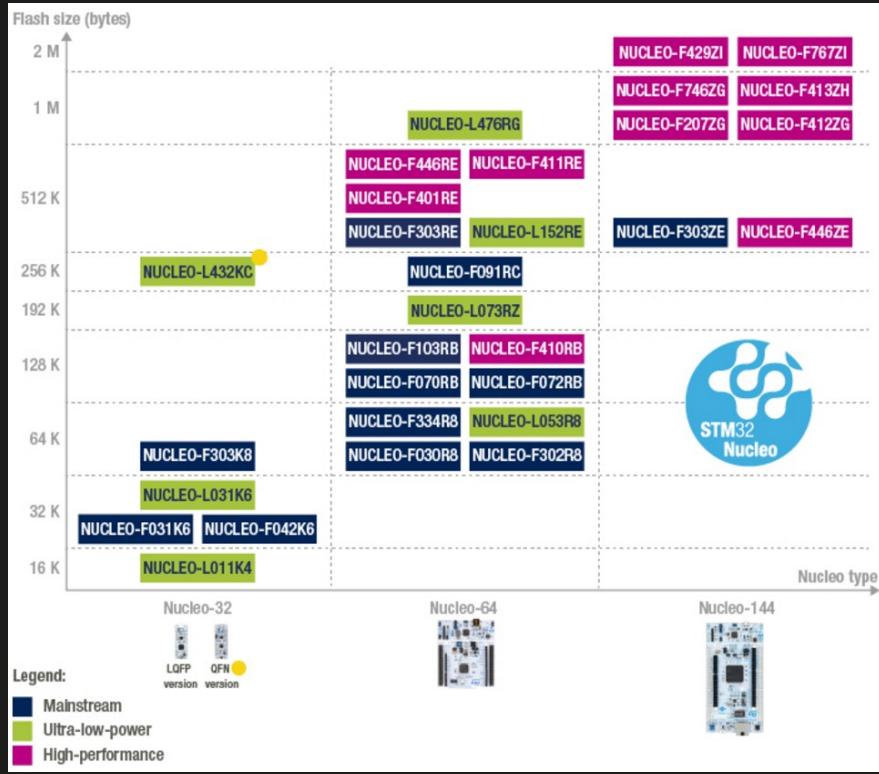


STM32 W series - Wireless (STM32W108)

24 MHz Cortex-M3 CPU	Up to 16-Kbyte SRAM	Up to 256-Kbyte Flash	2.4 GHz IEEE 802.15.4 Transceiver	Lower MAC Digital baseband	AES 128-bit	
----------------------	---------------------	-----------------------	-----------------------------------	----------------------------	-------------	--



Le projet Nucleo propose des maquettes d'évaluation à bas coût utilisant des solutions MCU et outils de développement de l'industrie ( $\approx 10$  €).



Nucleo-64

- Power supply
- Programmer (JTAG emulator)
- Target MCU
- Switch and LED
- External ports
- Shields connectors
- Arduino shield connectors

Observons les résultats d'une étude de marché réalisée chaque année.



**ASPENCORE**

**2019 Embedded Markets Study**

**Integrating IoT and Advanced Technology Designs,  
Application Development & Processing Environments**

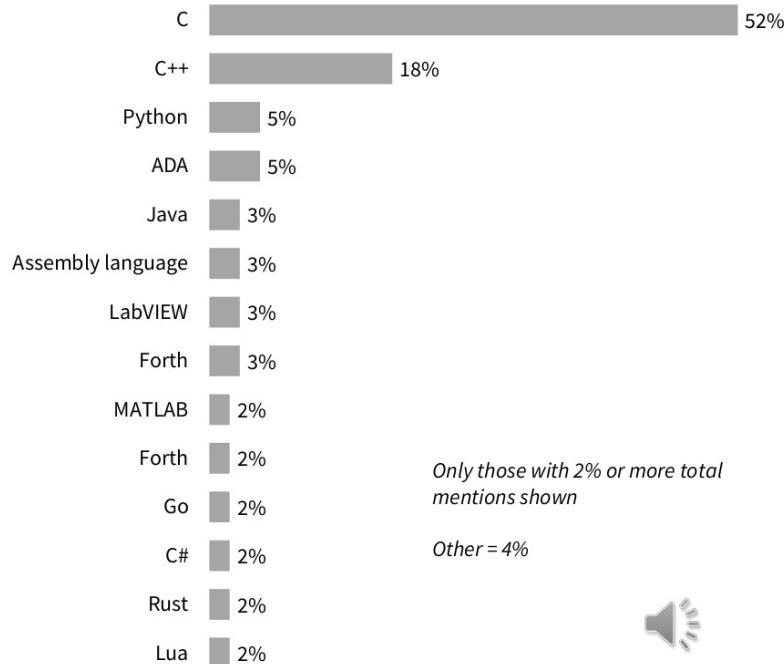
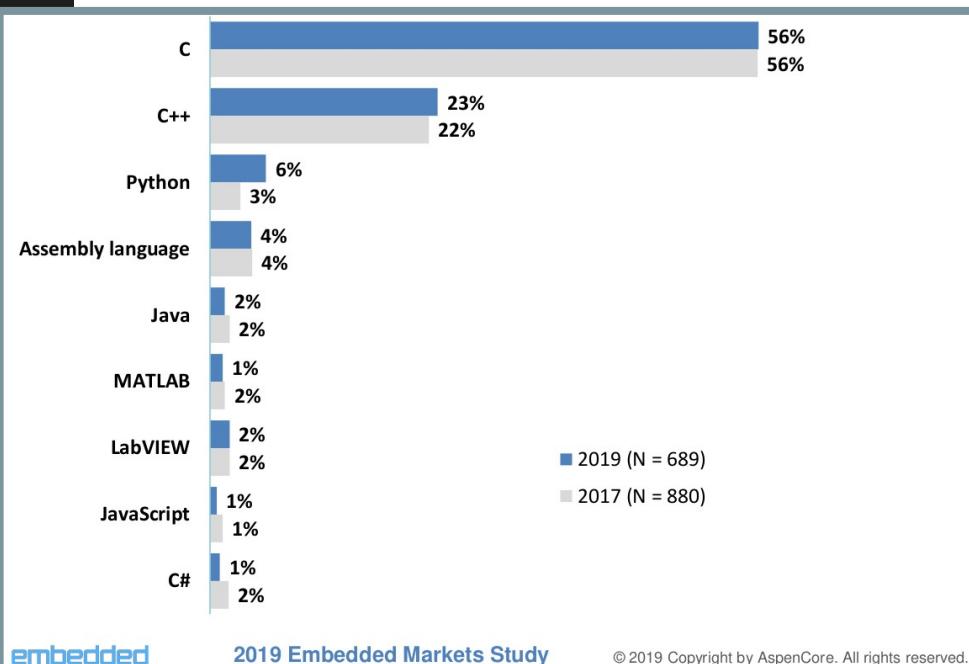
**March 2019**

Presented By: **EE Times** **embedded**

© 2019 AspenCore All Rights Reserved

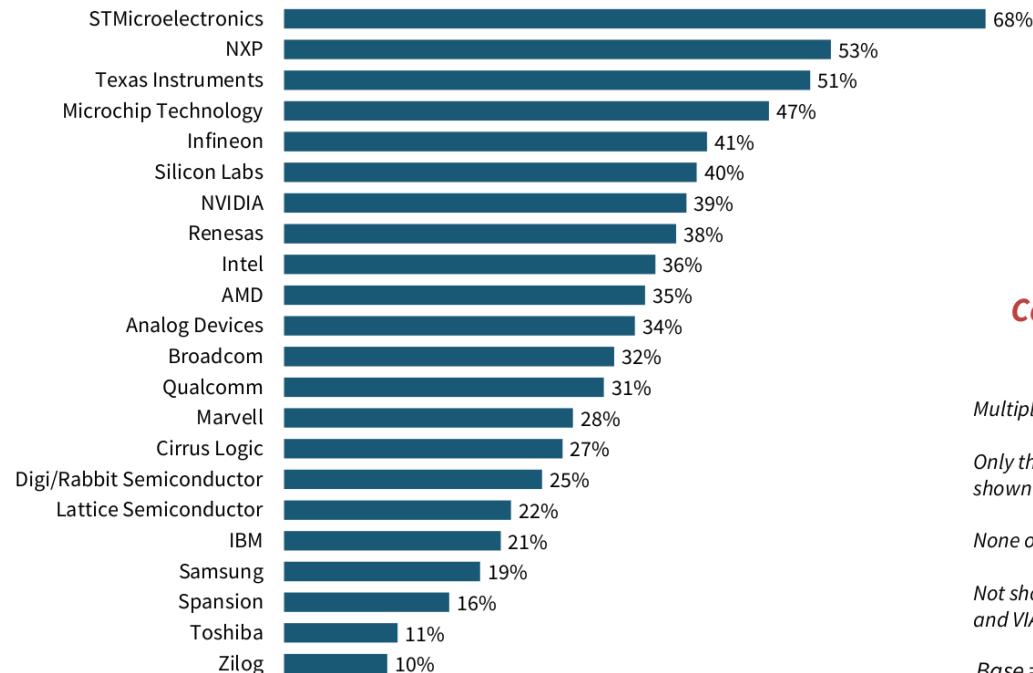
## Software development requires more cycle time

“C” dominates other languages for embedded software programming



## Future consideration of MPU/MCU vendors

STMicro, NXP, TI, and Microchip are the most efficient at converting familiarity into consideration for their processor solutions



**'Aided'  
Consideration**

Multiple responses allowed

Only those with 4% or more total mentions shown

None of the above = 5%

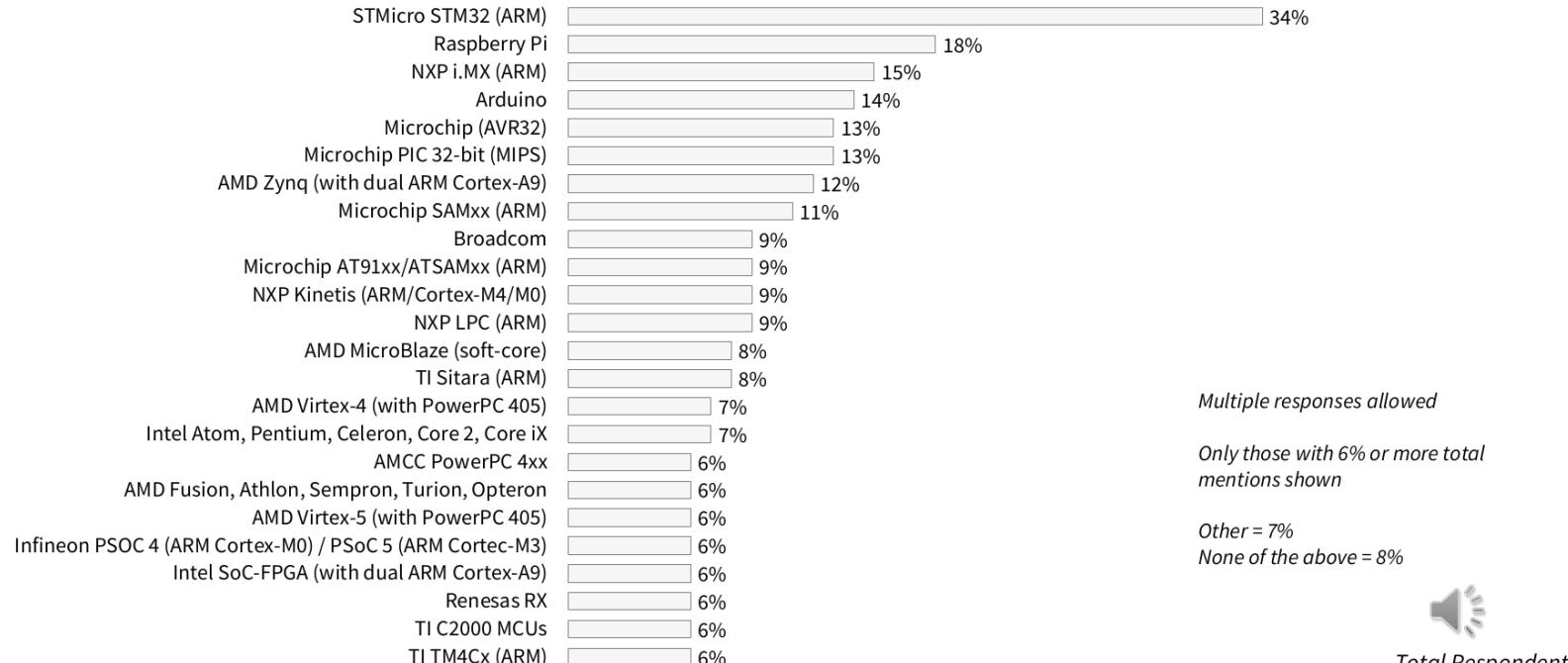
Not shown due to small sample size: Stretch and VIA



Base = Those familiar with each vendor

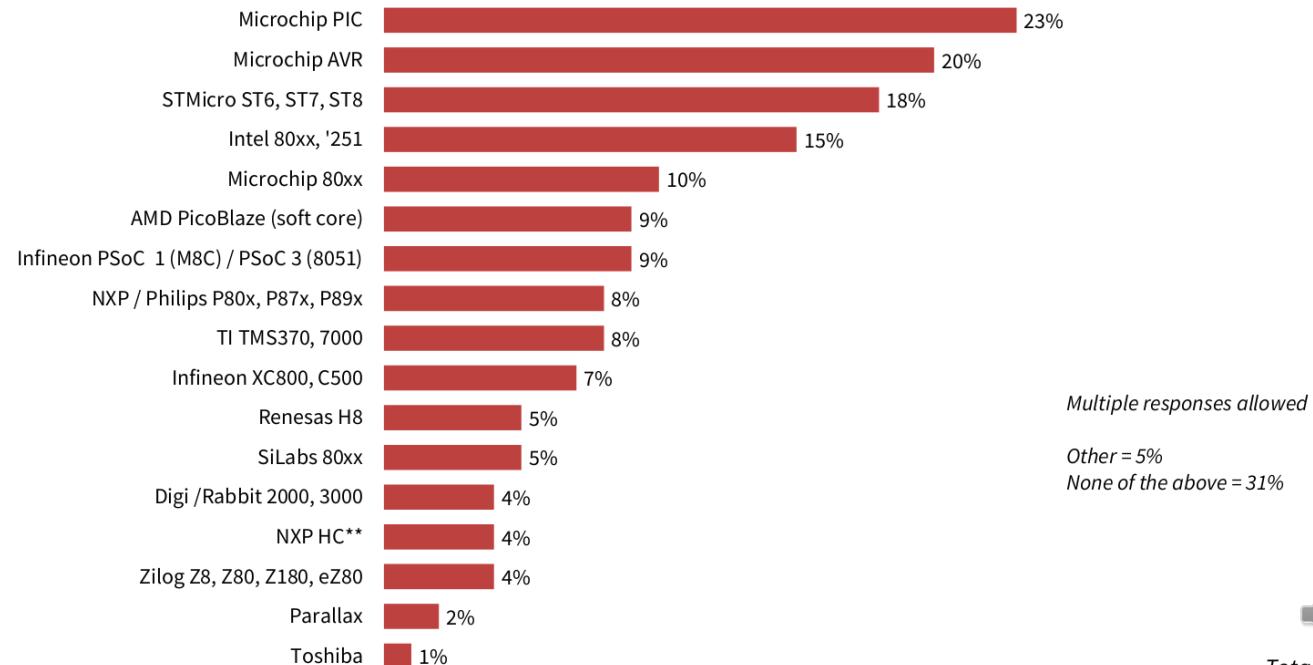
## Future consideration of 32-bit processor families

STMicro's STM32 is most widely considered, followed by Raspberry Pi, NXP's i.MX, Arduino and Microchip's AVR32



## Future consideration of 8-bit processor families

Microchip's PIC and AVR, STMicro's ST6, ST7 and ST8 and Intel's 80xx 8-bit processors are the most popular



# - GPP -

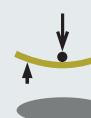
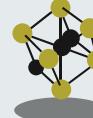
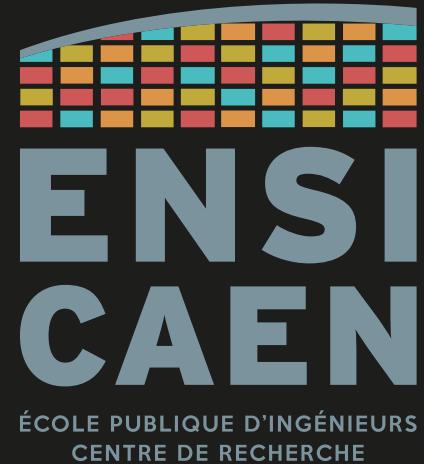
# GENERAL PURPOSE PROCESSOR

Applications

Architecture

Carte mère

Processeur superscalaire



## Applications

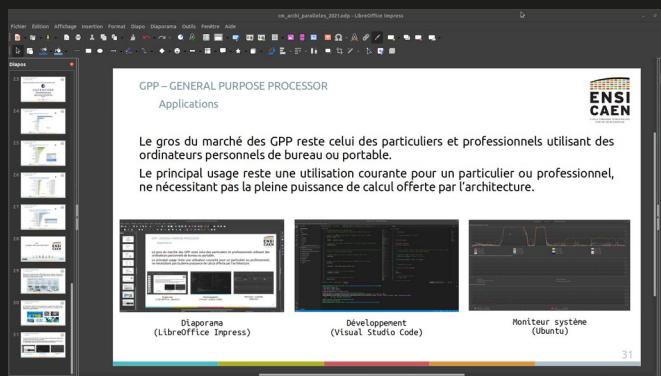
Les **General Purpose Processors (GPP)** possèdent une architecture CPU complexe leur offrant une **grande polyvalence**, notamment à l'exécution de code faiblement optimisé.

Il s'agit par exemple de programmes de contrôle offrant un code séquentiel avec un grand nombre de tests et d'appels de fonctions. Codes difficiles à accélérer.

```
444     prev = NULL;
445     for (mpnt = oldmm->mmap; mpnt; mpnt = mpnt->vm_next) {
446         struct file *file;
447
448         if (mpnt->vm_flags & VM_DONTCOPY) {
449             vm_stat_account(mm, mpnt->vm_flags, -vma_pages(mpnt));
450             continue;
451         }
452         charge = 0;
453         if (mpnt->vm_flags & VM_ACCOUNT) {
454             unsigned long len = vma_pages(mpnt);
455
456             if (security_vm_enough_memory_mm(oldmm, len)) /* sic */
457                 goto fail_nomem;
458             charge = len;
459         }
460         tmp = kmem_cache_alloc(vm_area_cachep, GFP_KERNEL);
461         if (!tmp)
462             goto fail_nomem;
463         *tmp = *mpnt;
464         INIT_LIST_HEAD(&tmp->anon_vma_chain);
465         retval = vma_dup_policy(mpnt, tmp);
466         if (retval)
467             goto fail_nomem_policy;
```

Le gros du marché des GPP reste celui des particuliers et professionnels utilisant des ordinateurs personnels de bureau ou portable.

Le principal usage reste une utilisation courante pour un particulier ou professionnel, ne nécessitant pas la pleine puissance de calcul offerte par l'architecture.



Diaporama  
(LibreOffice Impress)

Développement  
(Visual Studio Code)

## Moniteur système (Ubuntu)

## Applications

On peut également citer les applications de traitement du son, de traitement d'image, de traitement du signal, de développement logiciel ou de montages de médias.

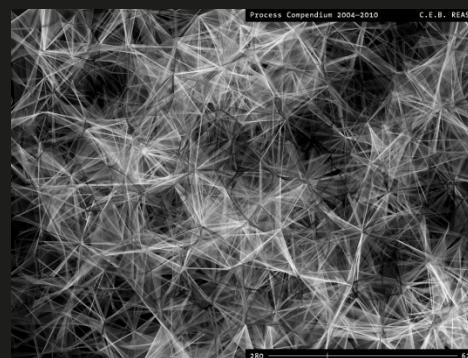
Celles-ci sont plus contraignantes au regard des ressources et exploitent souvent le plein potentiel du matériel.



Montage audio (Ableton)



Traitement du son



Traitement d'image

## Applications

Les applications industrielles sont également un terrain historique des GPP.

Ils sont typiquement rencontrés sur des tâches de contrôle ou des fonctions de calculs spécialisés. Ce marché tend à utiliser des solutions intégrées (AP, SoC, DSP, FPGA).



Radar GM400  
(Thalès)



Rafale  
(Dassault)

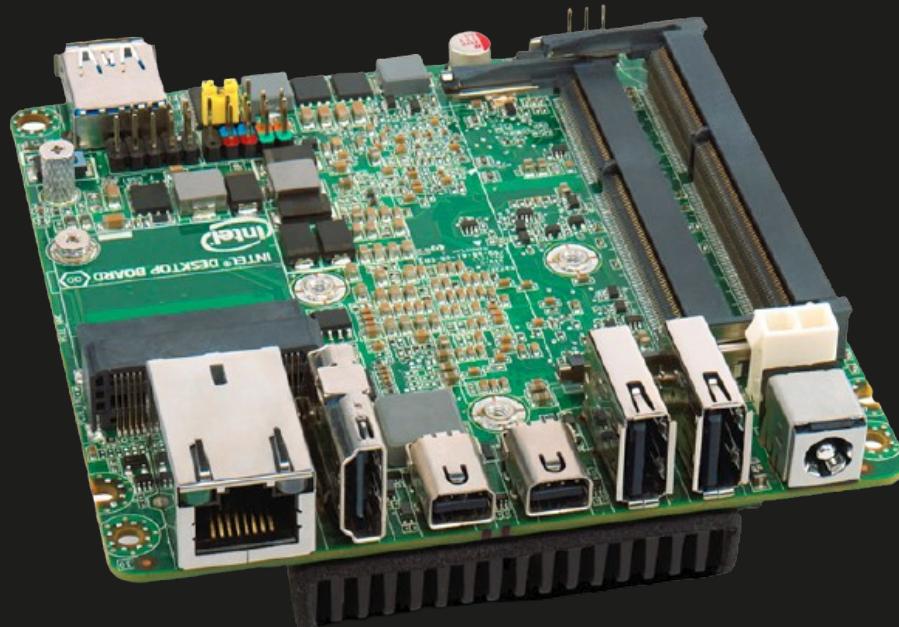


Borne automatique  
Box j200

## Applications

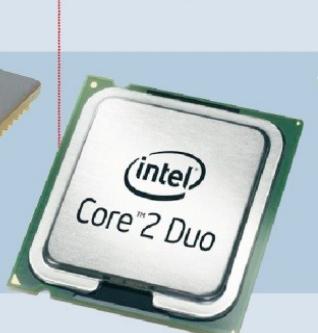
Notons que les GPP peuvent également être exploités par des applications rattachées au domaine des systèmes embarqués.

Voici par exemple la carte mère NUC Core i5 de Intel.



Observons les architectures phares d'Intel, leader actuel et historique du marché des GPP (*General Purpose Processor*) ou MPU (*MicroProcessor Unit*) mais également du marché des semi-conducteurs au sens large.

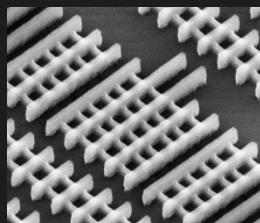
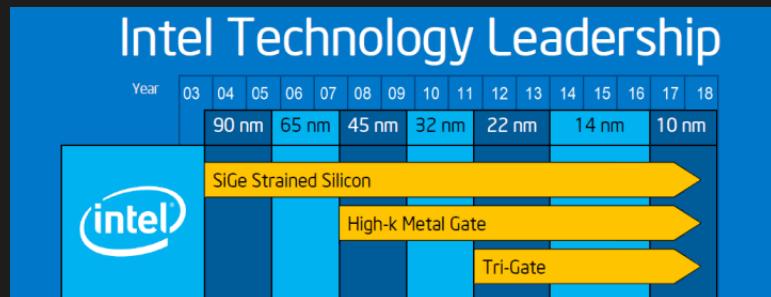
### 40 ANS DE COURSE À L'INNOVATION

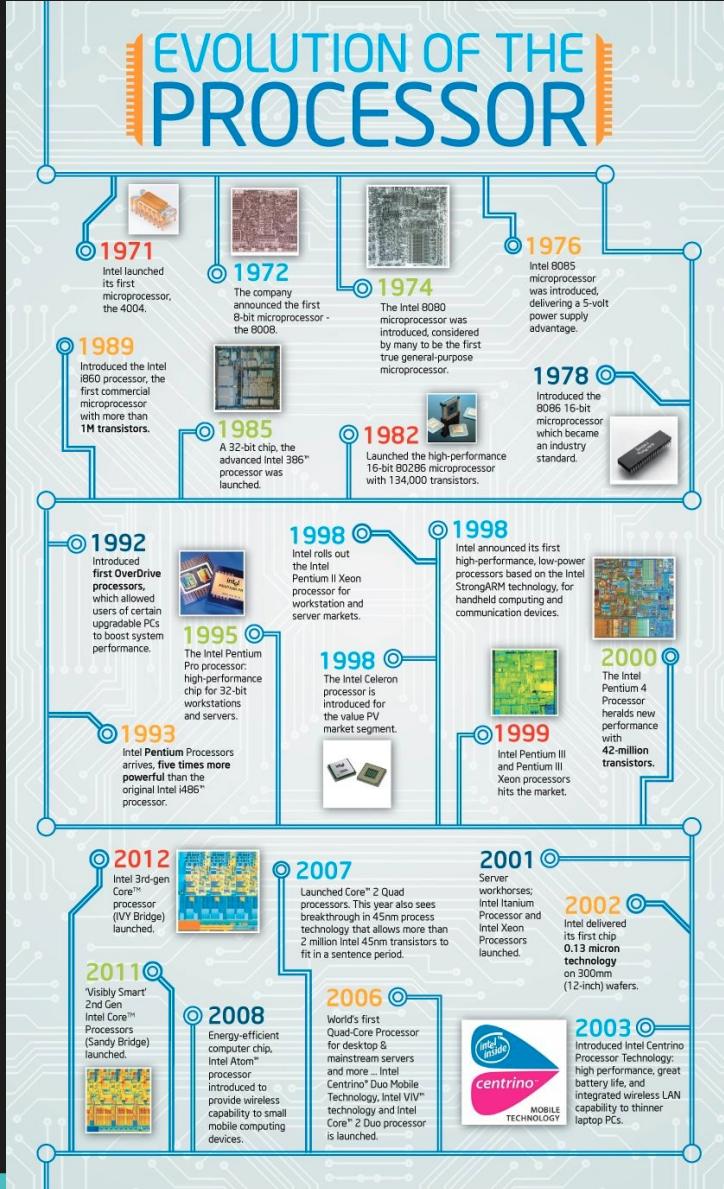
1971	1981	1993	2006	2012
<b>PROCESSEUR 4004 D'INTEL</b> Nombre de transistors : <b>2.300</b> Puissance : <b>108 kilohertz</b> <b>10 microns</b>  	<b>PROCESSEUR 8088</b> Introduit dans les PC d'IBM Nombre de transistors : <b>29.000</b> Puissance : <b>5 megahertz</b> <b>3 microns</b>  	<b>PENTIUM</b> Nombre de transistors : <b>3.1 millions</b> Puissance : <b>66 megahertz</b> <b>0,8 micron</b>  	<b>INTEL CORE 2 DUO</b> Nombre de transistors : <b>291 millions</b> Puissance : <b>2.93 gigahertz</b> <b>65 nanomètres</b>  	<b>PROCESSEURS IVY BRIDGE</b> Nombre de transistors : <b>1.400 millions (3D)</b> Puissance non communiquée <b>22 nanomètres</b>  

IDÉ / SOURCE ET PHOTOS : INTEL

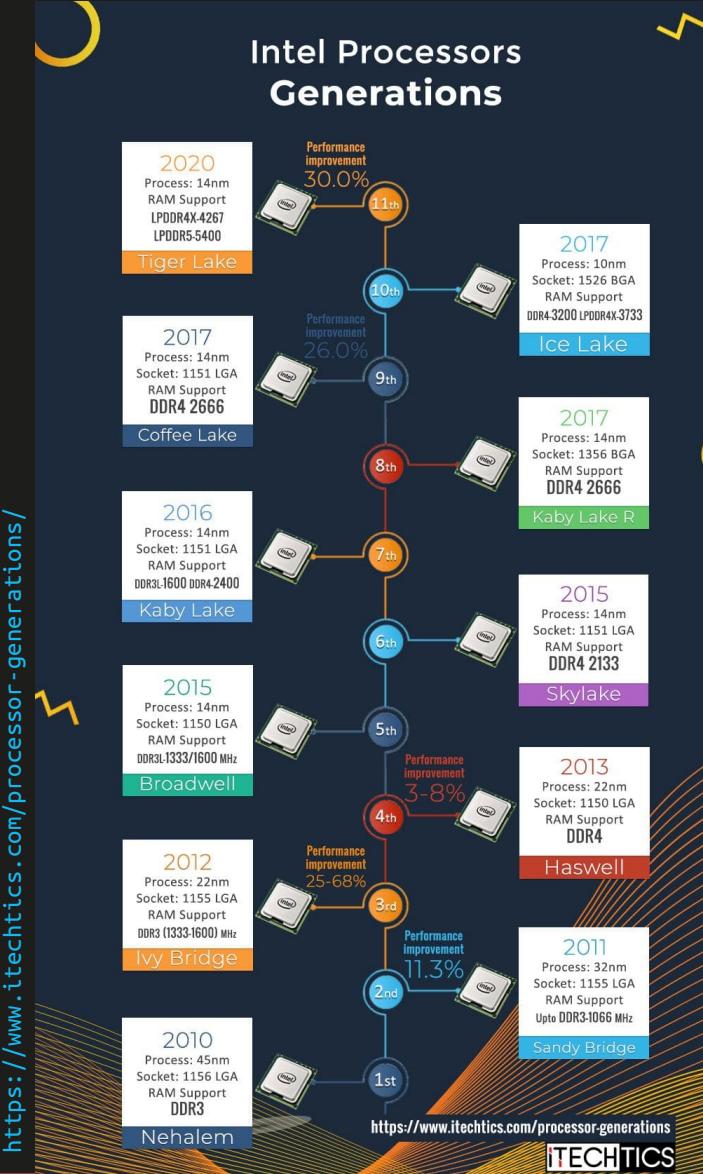
## Architectures Intel

Les architectures GPP phares à notre époque sont les familles Core i3/i5/i7 de Intel. Mais prudence, il existe un grand nombre d'autres architectures et fondeurs de GPP ciblant divers marchés différents.



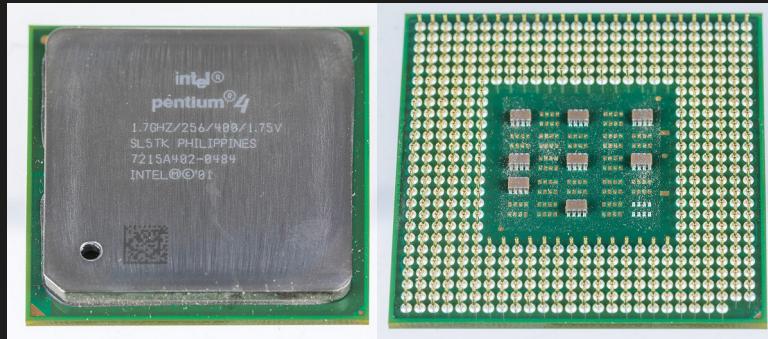


<https://www.itechtics.com/processor-generations/>



## Architectures Intel

4004	(1971)	Processeur 4 bit
8008	(1972)	Processeur 8 bit
<b>8086</b>	(1978)	Processeur 16 bit

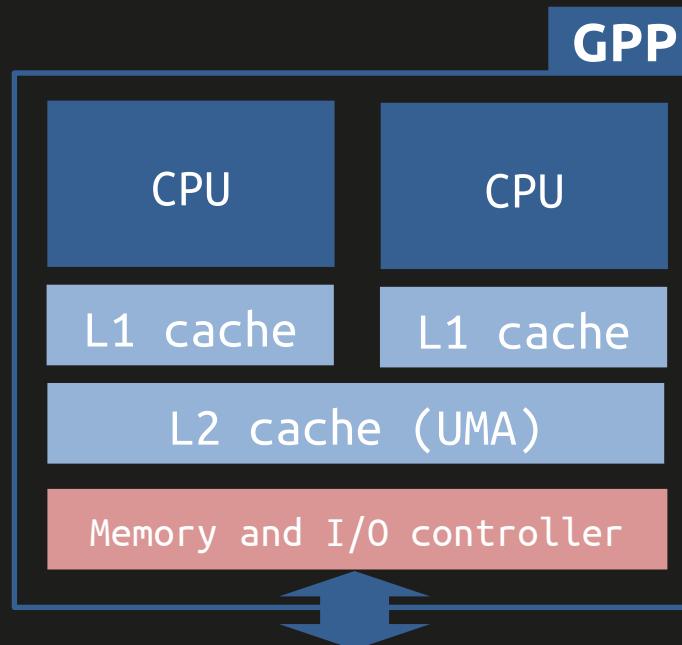


→ Premier CPU x86 (ISA x86-16)

<b>80386</b>	(1985)	Processeur 32 bit	→ ISA x86-32, rétro-compatible x86-16
Pentium	(1993)	Processeur 32 bit	→ Premier superscalaire commercialisé
<b>Pentium 4</b>	(2000)	Processeur 32 bit	→ 2 cœurs logiques (2 threads)
Core 2 Duo	(2006)	Proc. 32/64 bit	
		→ Apparition du multi-core chez Intel	
		→ Naissance de l'ISA x86-64 (calé sur celui d'AMD), rétro-compatible x86-32 et x86-16 !	
Core	(2008)	12 générations se succèdent jusqu'à aujourd'hui (2022)	

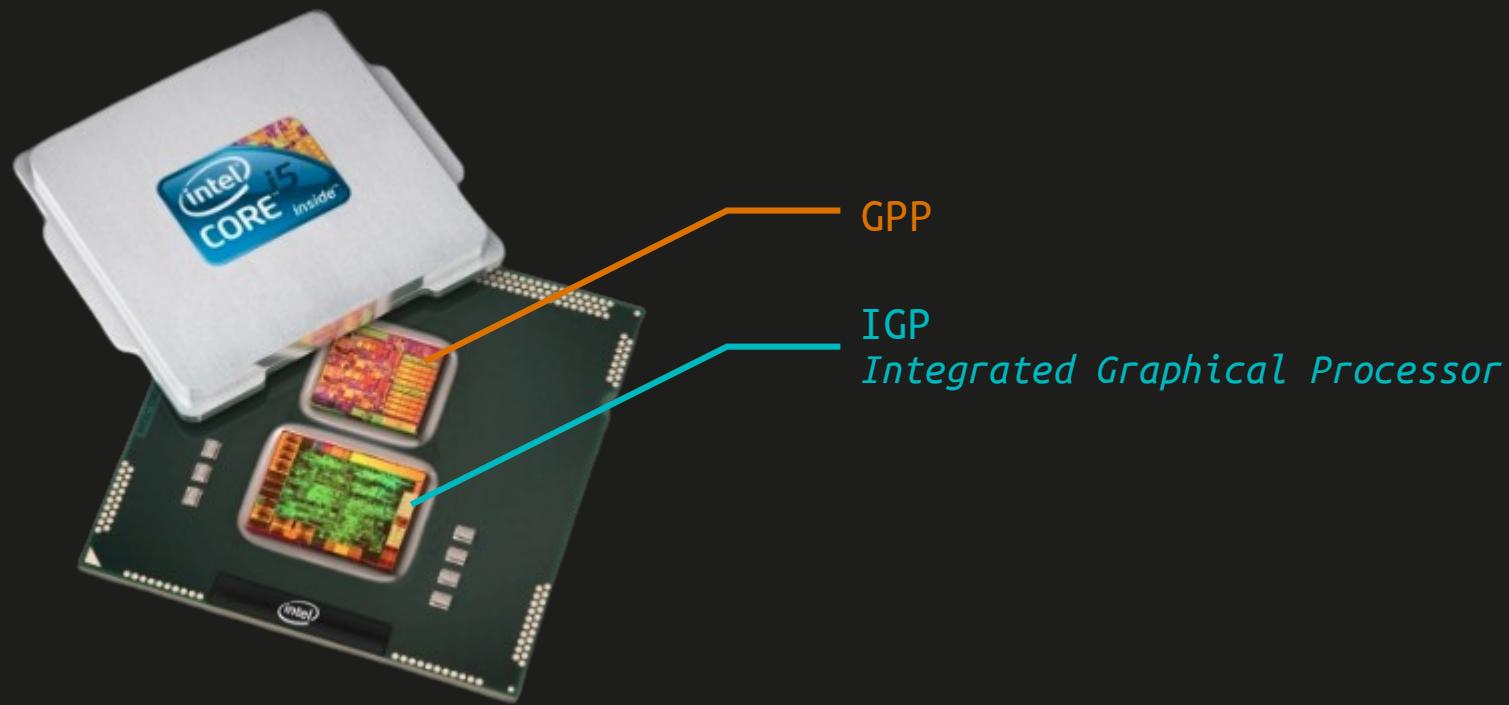
Processeur de traitement nu, dépourvu de mémoire principale.

Il embarque un ou plusieurs CPU (architecture homogène) mariés avec leurs caches, possède un modèle mémoire uniforme (UMA) et embarque un contrôleur d'interfaces.



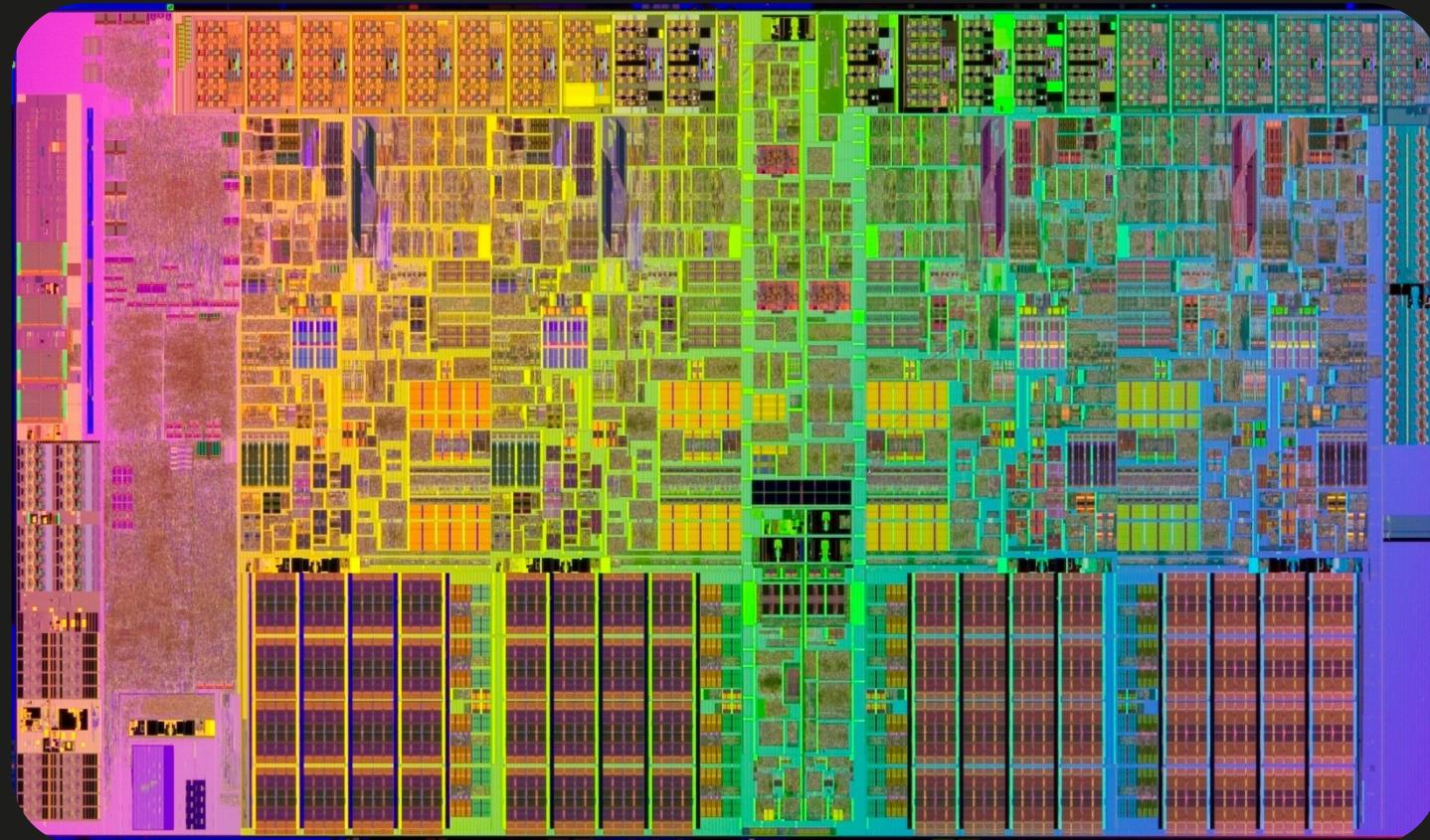
Exemple : Intel Core i5

Exemple de la famille Core i5 de Intel.



# GPP – GENERAL PURPOSE PROCESSOR

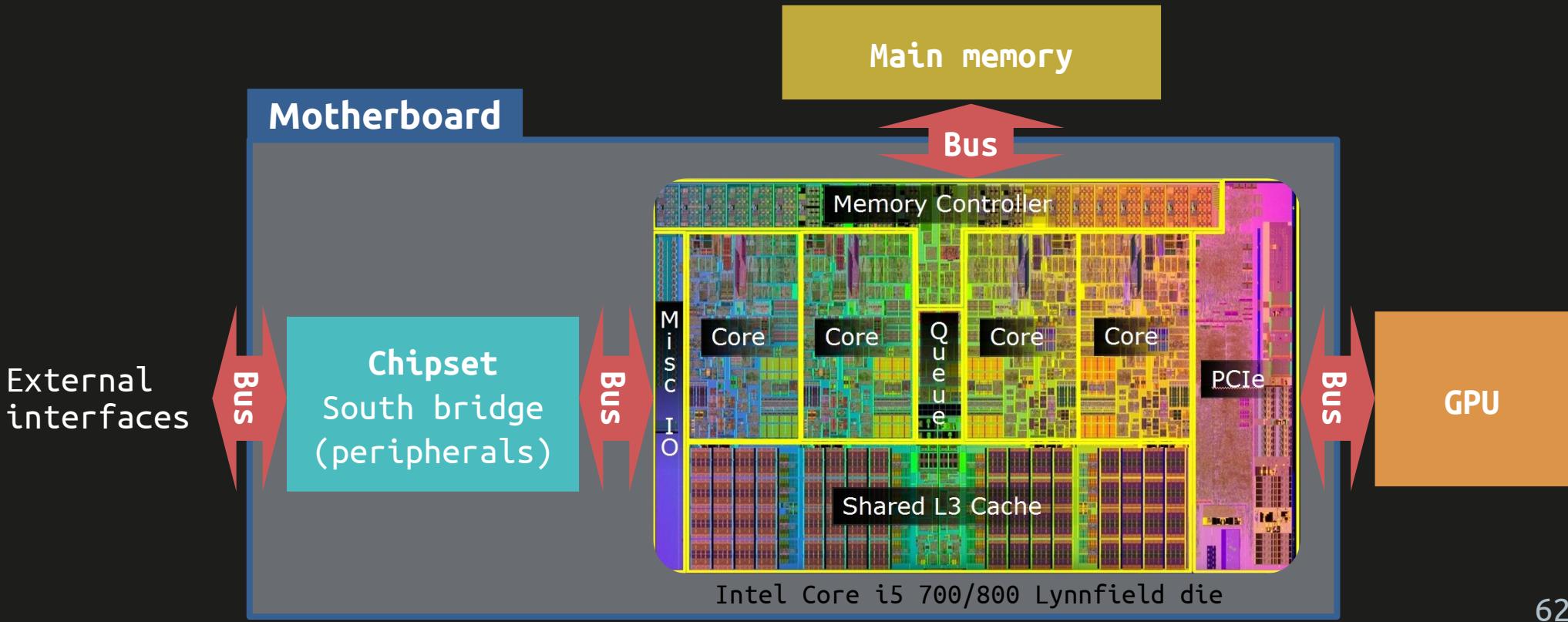
Exemple : Intel Core i5



Intel Core i5 700/800 Lynnfield die

Exemple : Intel Core i5

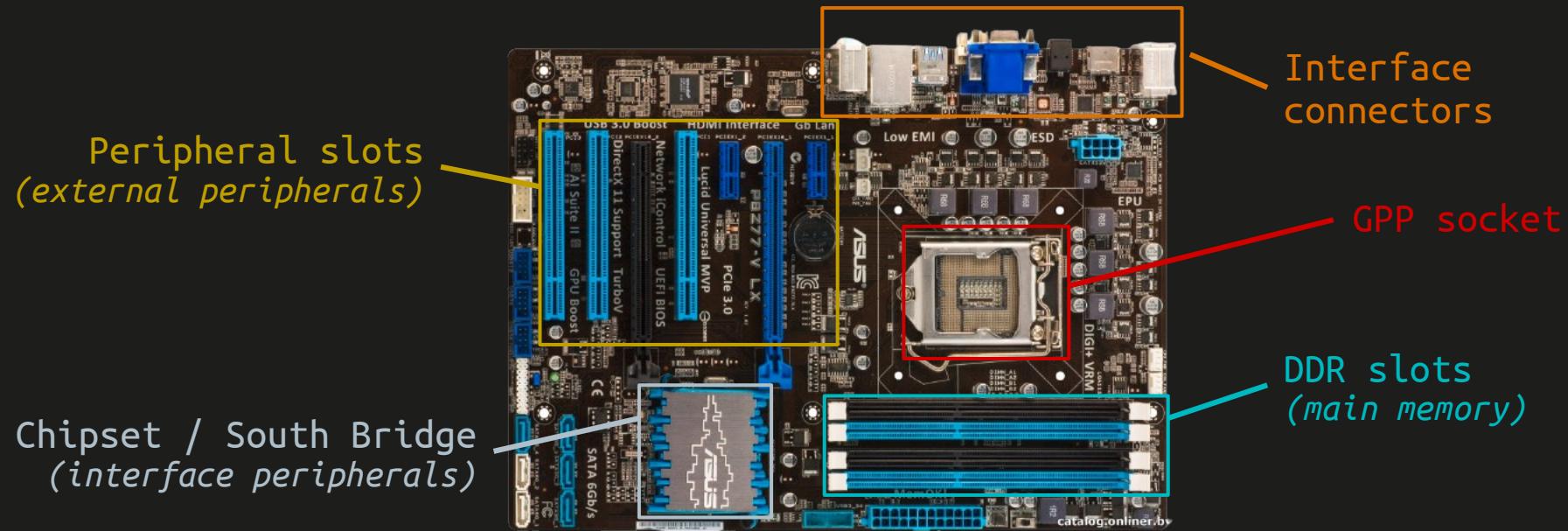
Intégration dans le système (carte mère)



## Carte mère

Un GPP doit forcément être porté sur une carte mère avec mémoire principale et périphériques d'interfaces externes déportés.

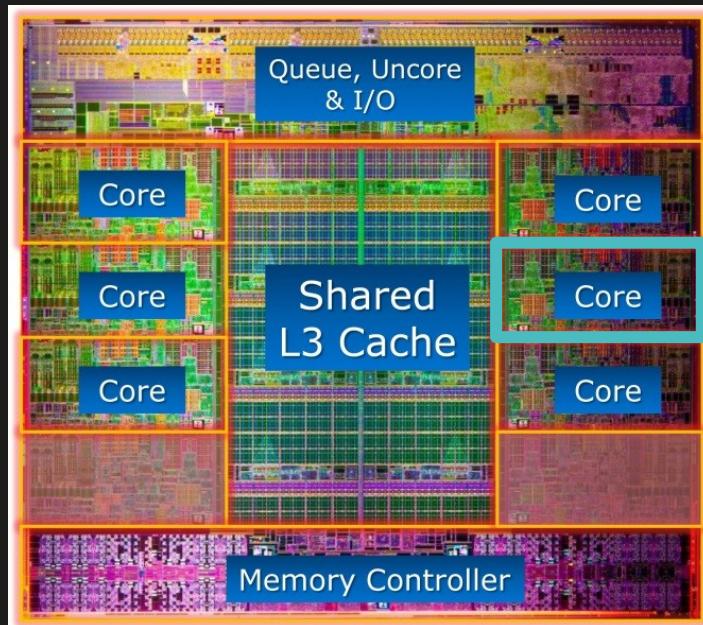
Exemple de carte mère ASUS, n°2 du marché mondial en 2016.



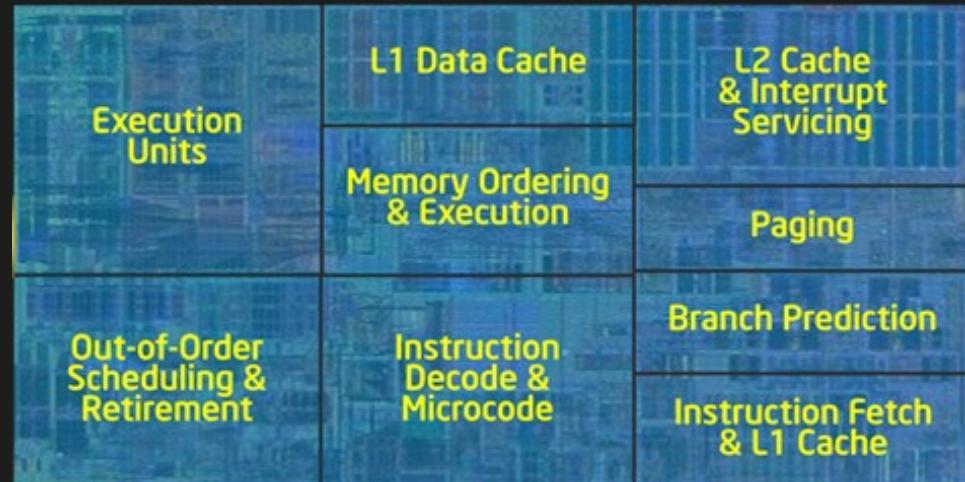
Les GPP possèdent un CPU dit **superscalaire**. Les processeurs possédant ce type de pipeline CPU se caractérisent le plus souvent par le déploiement des mécanismes d'accélération matériels suivants :

- **Étage d'exécution *Out Of Order*** : Exécution des instructions dans le désordre. Ordonnanceur matériel gérant les dépendances fonctionnelles et sur les données, étages de renommage des registres (résultats intermédiaires) et de réordonnancement
- **Étage de prédiction au branchement**
- **Étage d'exécution *RISC-like***, même si l'ISA est CISC

*Die d'un CPU de la génération Sandy Bridge de Intel, illustré pour un Core i7.*



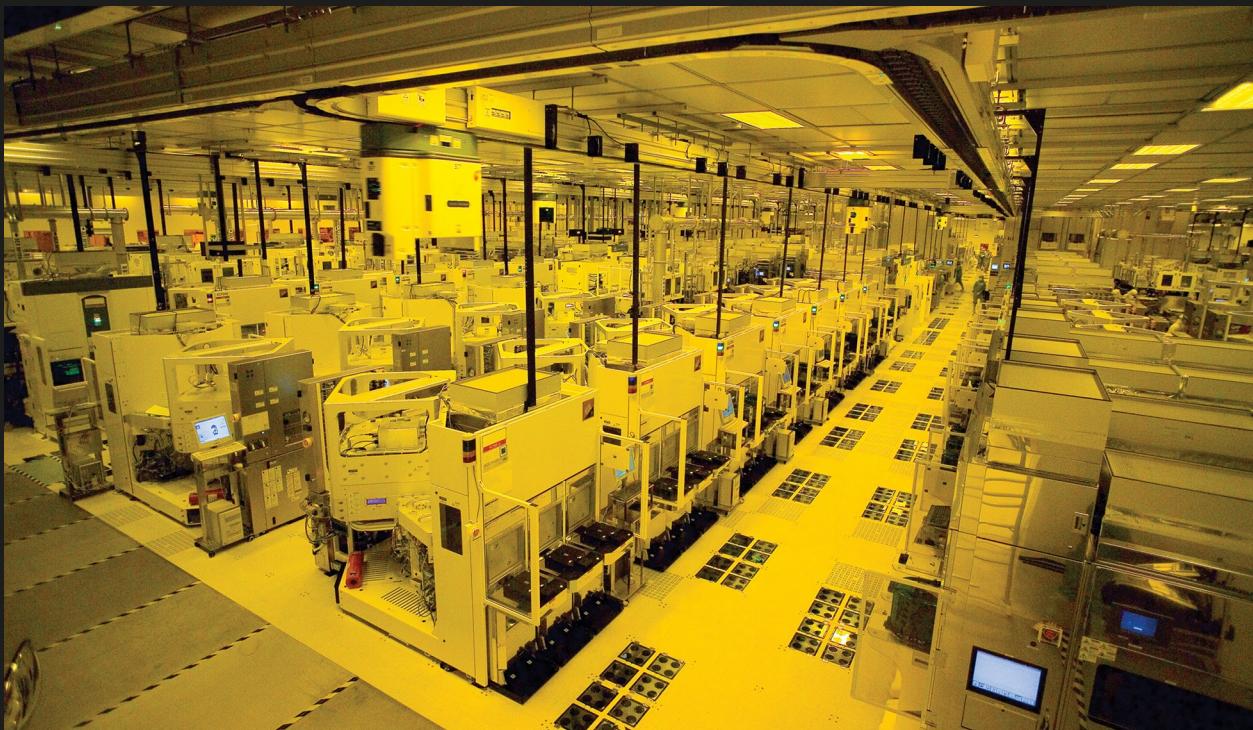
Intel Core i7



Sandy Bridge CPU/Core

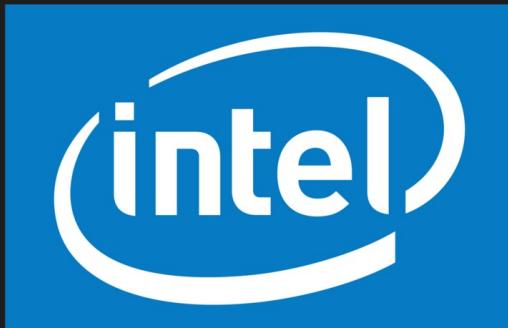
## Fabrication des processeurs

Le plus gros fondeur mondial de processeurs est TSMC (*Taiwan Semiconductor Manufacturing Company*). Il s'agit d'un *pure-player* (une seule activité industrielle et économique) ne faisant que de la fabrication de circuits intégrés.



En août 2023, l'Allemagne et TSMC s'associent pour lancer la construction d'une usine à +10 milliards €.  
Infineon, NXP et Bosch participent au projet.

Intel, NXP, Samsung, TI, ... sont des IDM (*Integrated Device Manufacturer*), qui ont des activités de conception, de fabrication et de vente de circuits intégrés. Ces mêmes compagnies sous-traitent une partie de leur production auprès de TSMC ou encore GlobalFoundries.



On peut également compter dans l'écosystème les entreprises *fabless* (sans usine de production), parmi lesquelles Nvidia, Qualcomm, Broadcom, AMD, MediaTek, Xilinx, Altera ... Ces entreprises reposent sur les IDM ou *pure-players* pour assurer la production de leurs circuits intégrés.

## Fabrication des processeurs

How are Microchips Made?

Branch Education

CPU Manufacturing Process Steps

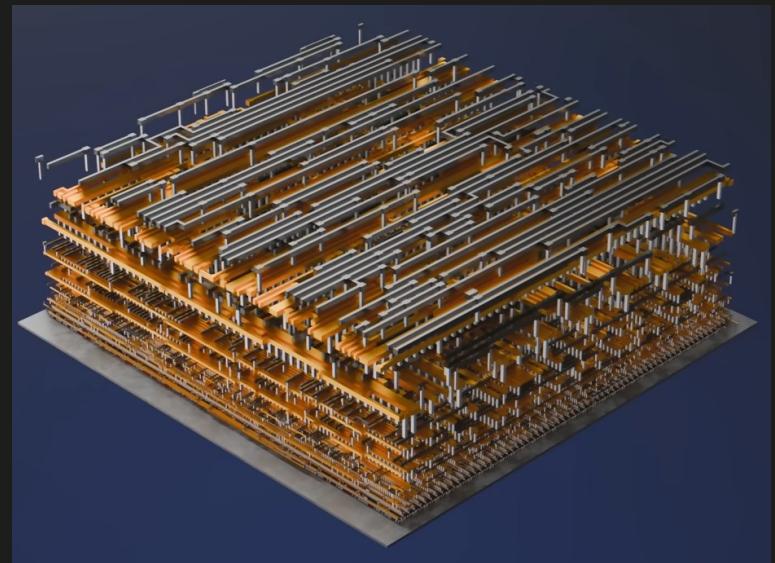
<https://www.youtube.com/watch?v=dX9CGRZwD-w>

00:00 – 1:55 - How are Transistors Manufactured?

02:34 – 4:51 - What's inside a CPU?

07:51 – 10:15 - 3D Animated Semiconductor Fabrication Plant Tour

22:18 – 23:42 - Silicon Wafer Manufacturing / Wafer Testing / Binning



Zoom Into a Microchip – NISENet

<https://www.youtube.com/watch?v=Fxv3JoS1uY8>

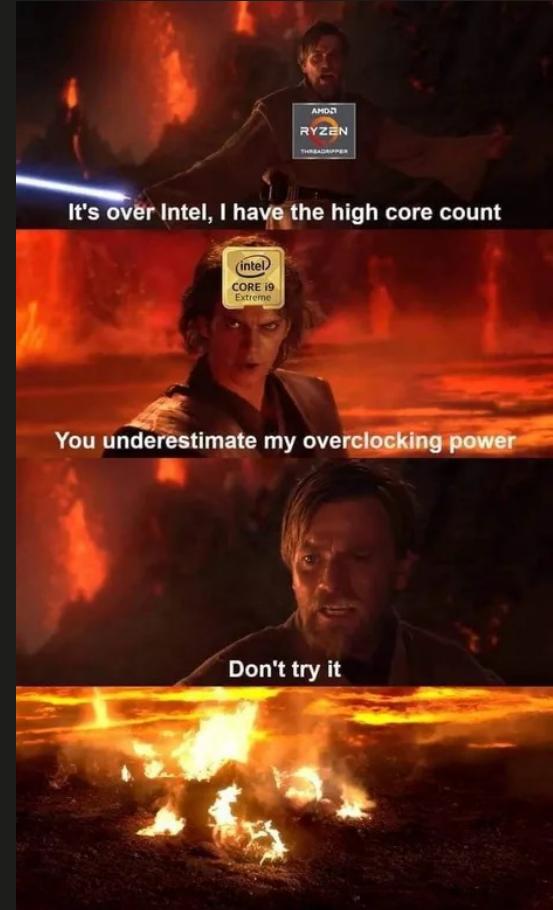


## Architecture superscalaire

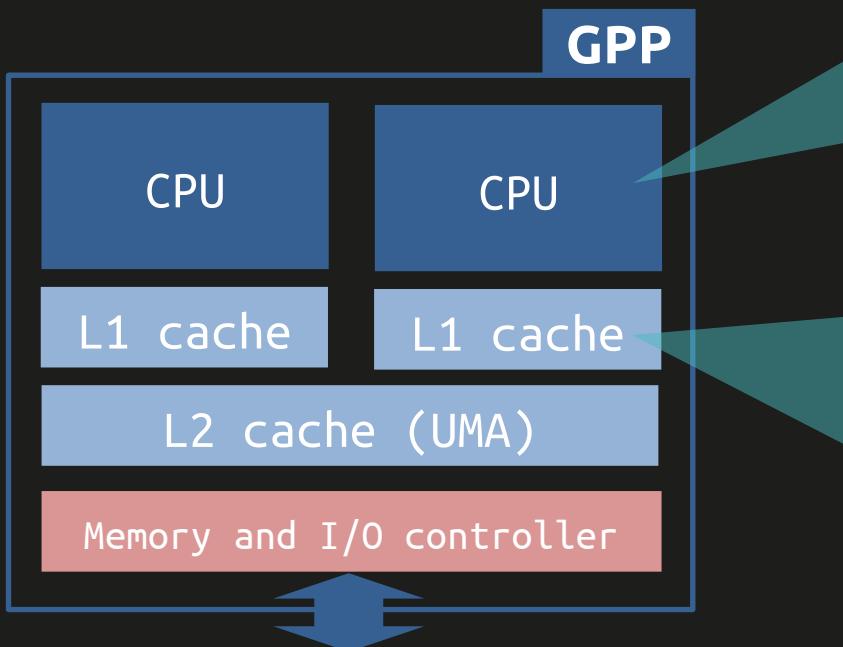
Attention, cette grande polyvalence et complexité matérielle se paye par un manque de déterminisme voire de performance à l'exécution sur des traitements algorithmiques spécifiques.

**Les GPP offrent un ratio performance de calcul ramené au coût et au Watt peu intéressant.**

Ils sont pensés pour porter un OS (*Operating System*) évolué et exécuter du code applicatif. Prenons les exemples des applications de traitement du son, traitement d'image, traitement vidéo, traitement d'antenne ... pour lesquels ils ne sont pas spécialisés.



À savoir



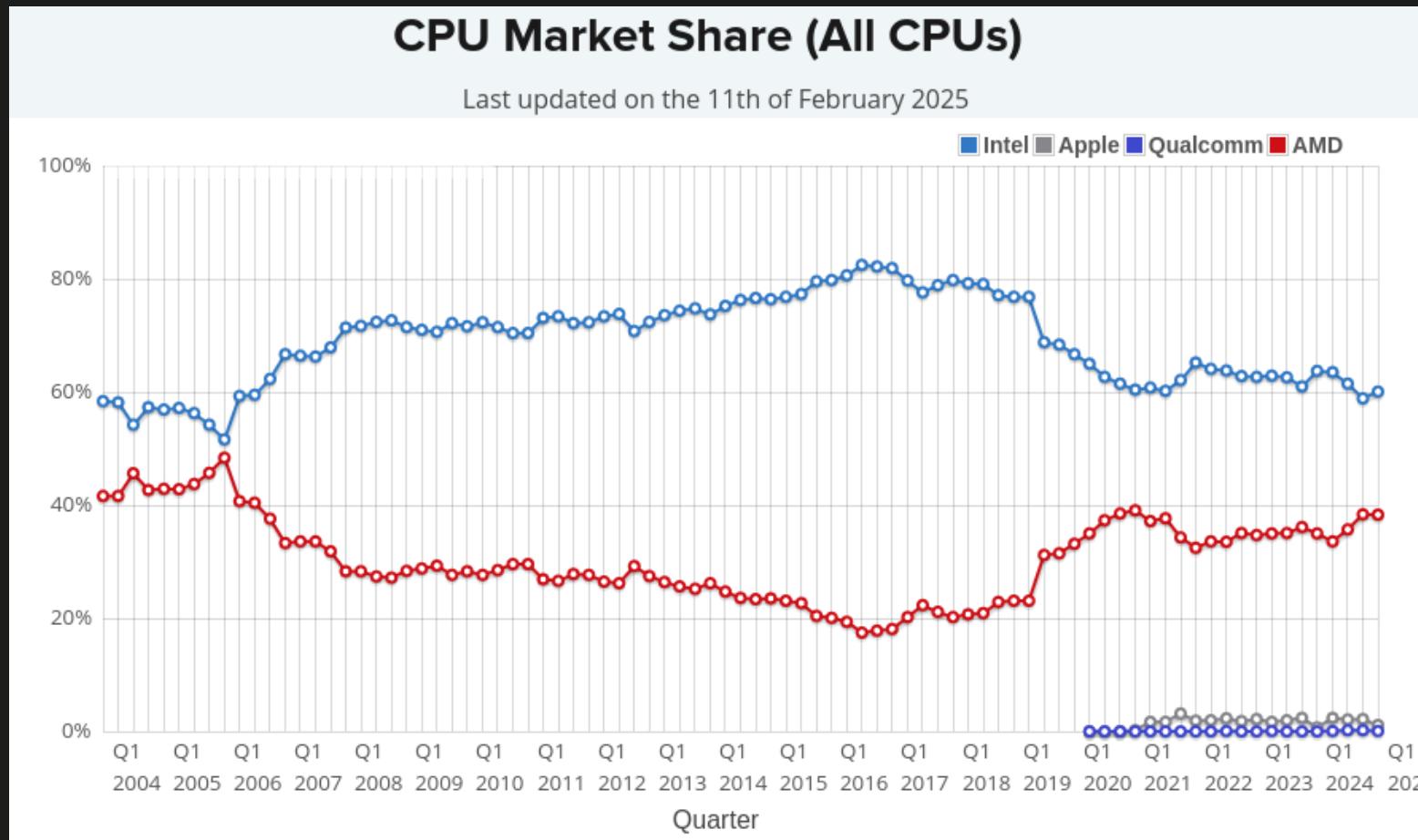
## CPU superscalaire

- exécution Out Of Order
- prédiction de branchement
- non déterministe
- mauvais ratio (puissance calcul) / (Watt x Coût)

## Mémoire

- Modèle mémoire uniforme (UMA)
- Cache processeur
  - Technologies de transfert rapides
  - Copies d'informations depuis la mémoire principale (DATA ou INST.)
  - Intelligence déportée dans les contrôleurs de caches (LRU)
  - Non déterministe

## Parts de marché : Intel vs. AMD



- AP -

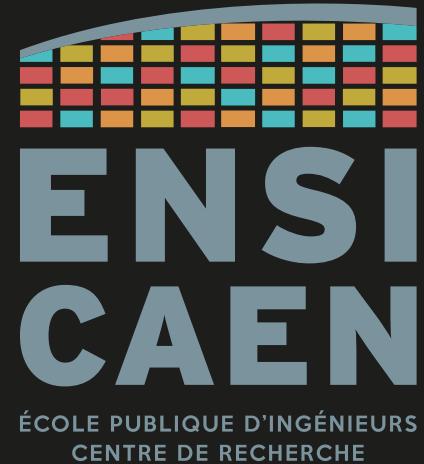
# APPLICATION PROCESSOR

Applications

Architecture

Solution Qualcomm

Solution ARM



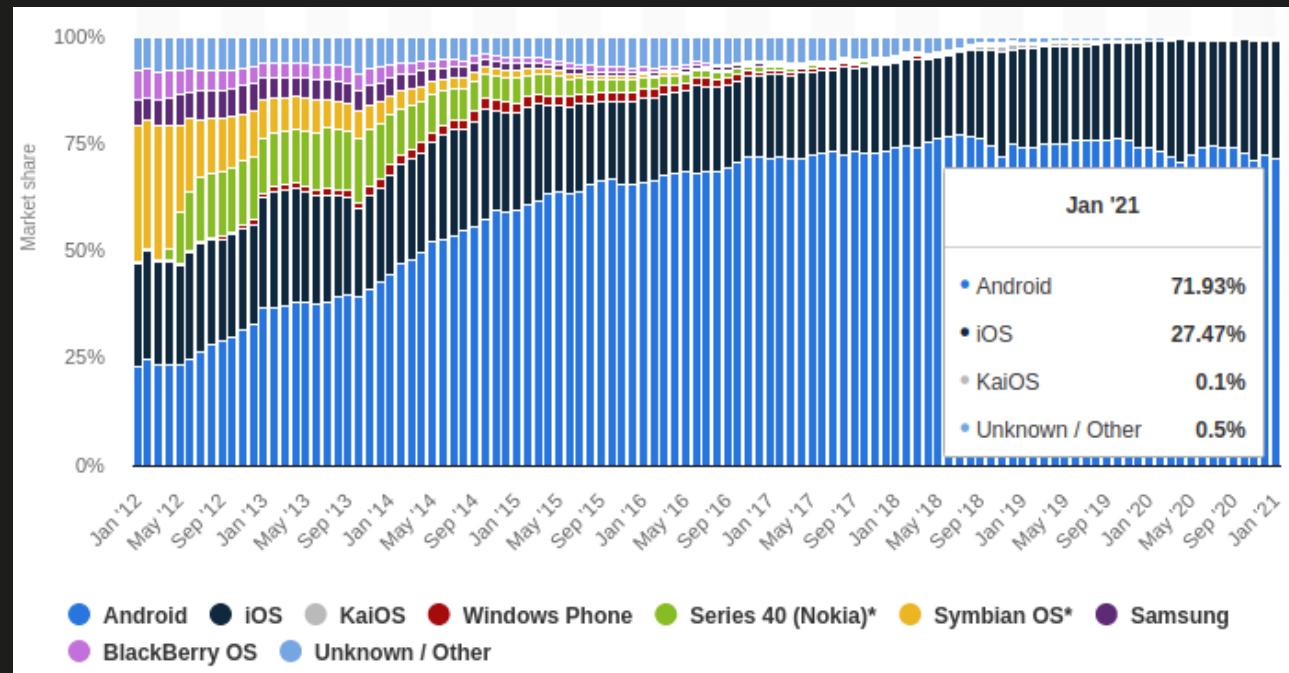
## Applications

Le marché des AP (*Application Processor*), processeurs riches en fonctionnalités et services matériels de type SoC (*System on Chip*), reste un marché récent qui a vu son envole avec celui des terminaux mobiles (smartphone, phablette et tablette).



Le principal marché des AP en terme de parts reste donc celui des terminaux mobiles.

Ce marché voit une utilisation écrasante du système d'exploitation Android en 2016, système basé sur un noyau Linux.



## Applications

Néanmoins les processeurs applications sont très rencontrés dans les systèmes embarqués au sens large, tous domaines confondus : *consumer*, défense, transport ...  
Ces systèmes embarquent généralement un OS et une interface graphique.



Freebox Revolution



Télévision 4K X94C Sony



Tablette Cook  
(fait à Caen par EOLANE)

Dans la majorité des cas, ces processeurs sont exploités par des systèmes évolués.

Sur ce marché les systèmes GNU/Linux (très souvent customisés) règnent en maîtres.



Exemple de plateforme industrielle durcie EOLANE (Français n°2 Européen) travaillant autour de SoC/AP iMX6 proposé par Freescale sur système GNU/Linux.

SOM SOLO



SOM QUAD



SBC



STARTER KIT



UN MODULE EMBARQUÉ  
OPTIMISÉ POUR VOS PRODUITS

UN MODULE MULTIMÉDIA  
PERFORMANT POUR VOS PRODUITS

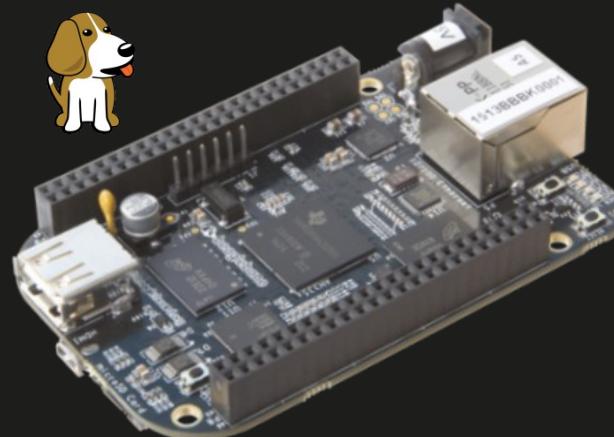
UNE SOLUTION PC  
INDUSTRIEL INTÉGRÉ

UNE PLATEFORME D'EVALUATION  
POUR VOS MAQUETTES

Voici les deux plateformes non-durcies à bas coût qui dominent le marché :  
les projets **Raspberry Pi** et **Beaglebone** (SoC AM335x TI).

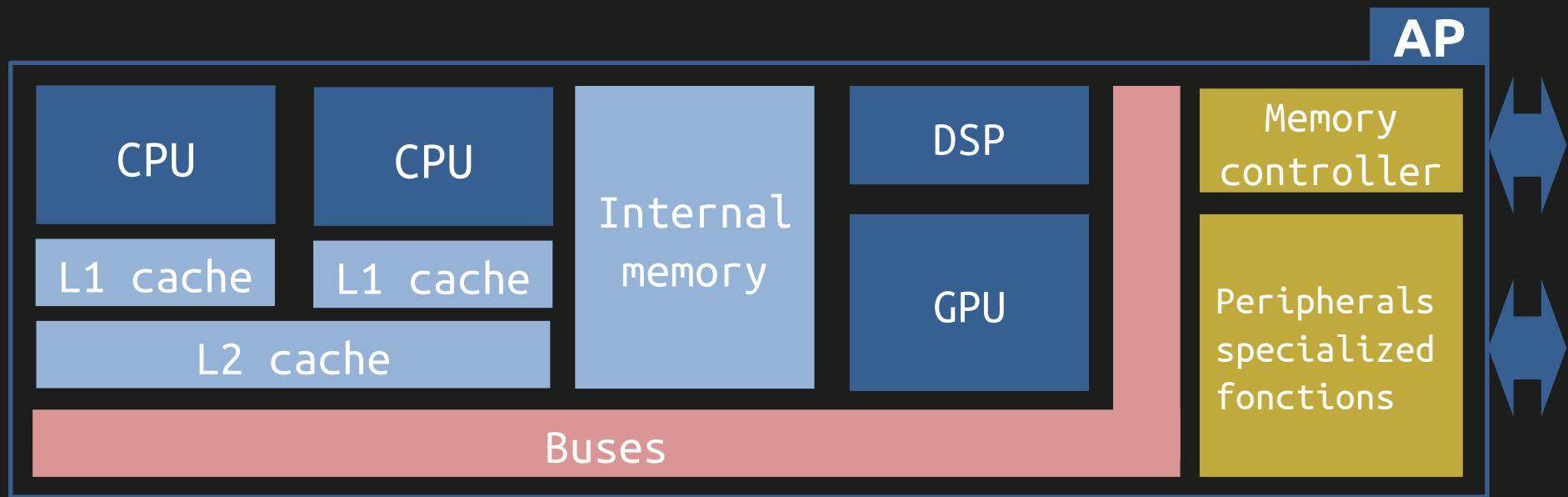
Ces solutions sont également basées sur des systèmes GNU/Linux

Elles sont très rencontrées durant les phases de prototypage ou en milieu universitaire,  
mais ne peuvent être industrialisées. Néanmoins des versions durcies existent.



Les AP sont des systèmes numériques complets intégrés dans une puce (architecture hétérogène).

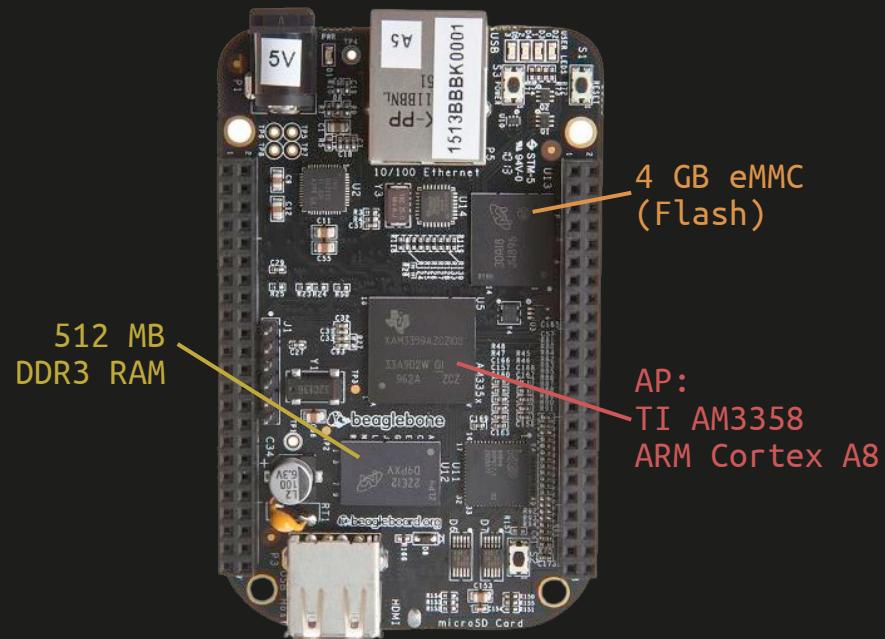
Néanmoins, la mémoire principale doit être ajoutée en externe.



Un **processeur application** embarque toujours un voire plusieurs CPU généralistes superscalaires. Ils sont dédiés à l'exécution du ou des systèmes d'exploitation évolués (virtualisés ou réels) ainsi que des applicatifs.

Un **AP** contient également une voire plusieurs fonctions spécialisées de calcul (GPU, DSP, crypto ...), un jeu de périphériques évolués complet et une mémoire interne ne permettant pas d'accueillir le système (*bootloader*).

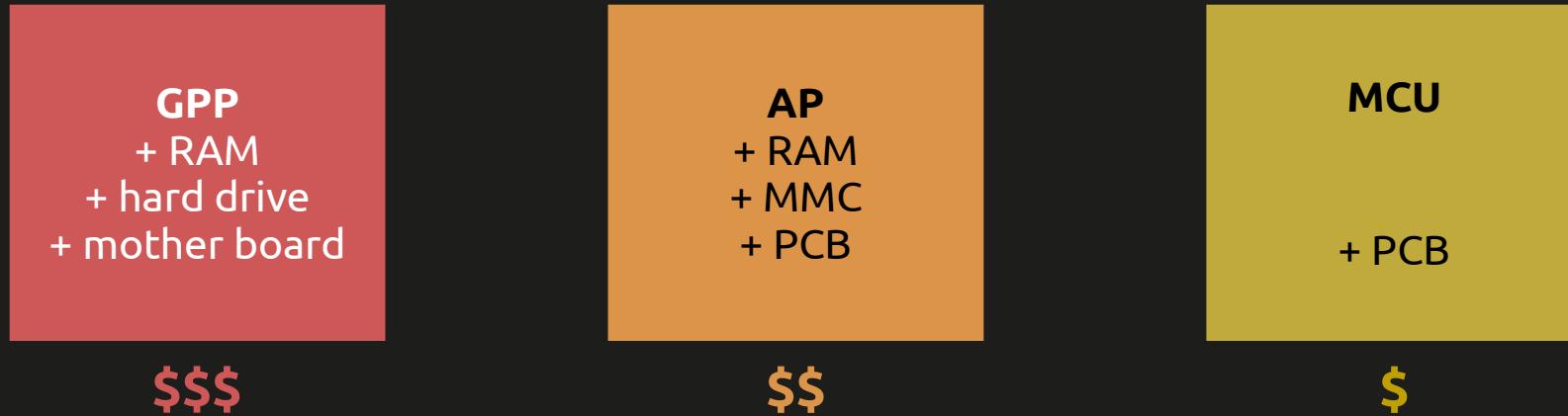
Par conséquent, une **mémoire principale** (DDR volatile) et une mémoire non-volatile de **stockage de masse** (MMC, eMMC, SDCard ...) externes doivent lui être ajoutées.



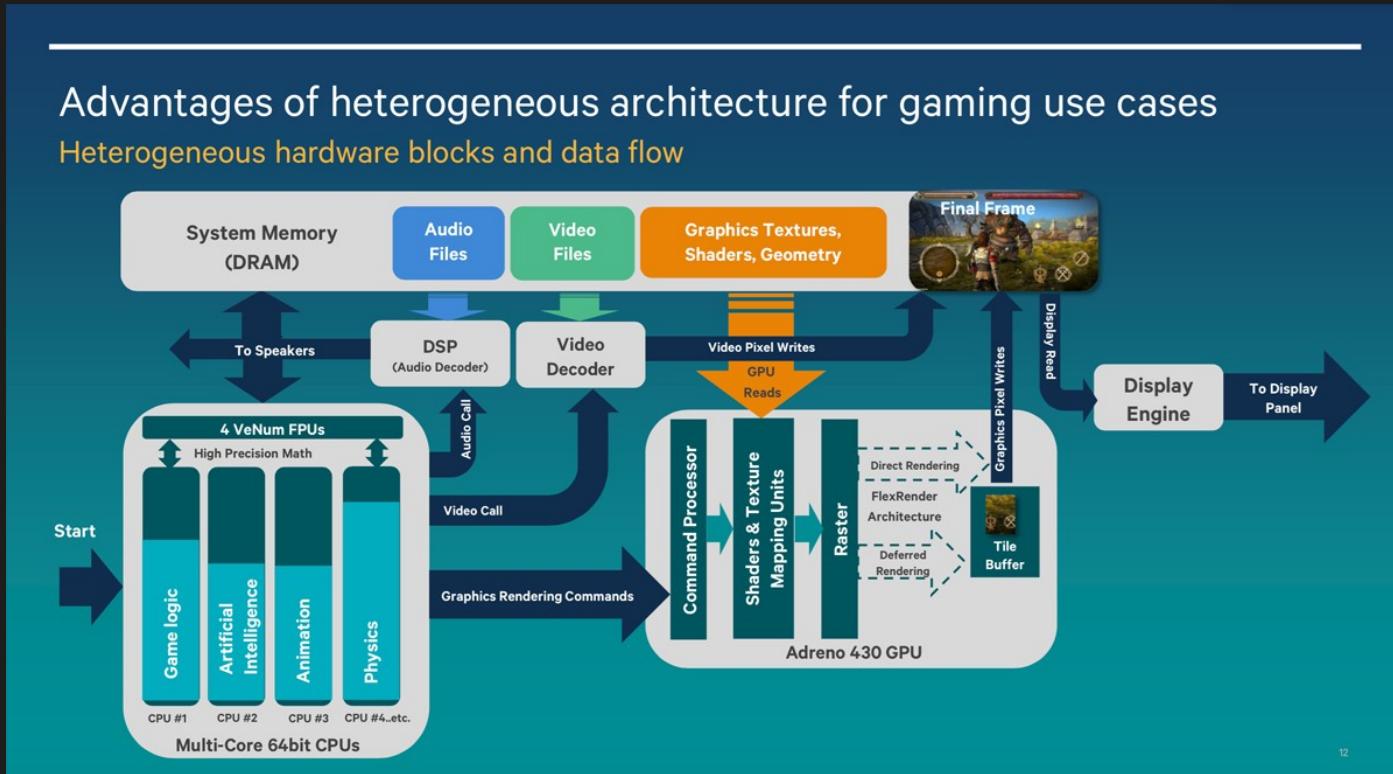
## Comparaison des processeurs généralistes

Contrairement aux MCU embarquant tous les services matériels sur la puce afin de contrôler un système (*on chip*), les AP exigent un coût unitaire non négligeable et restent dépréciés pour les applications à faible coût et fort volume.

Ils sont alors utilisés si il y a nécessité d'une interface et/ou de connectivités évoluées dans l'application.



Observons l'intérêt d'une architecture hétérogène pour une application aux jeux vidéos



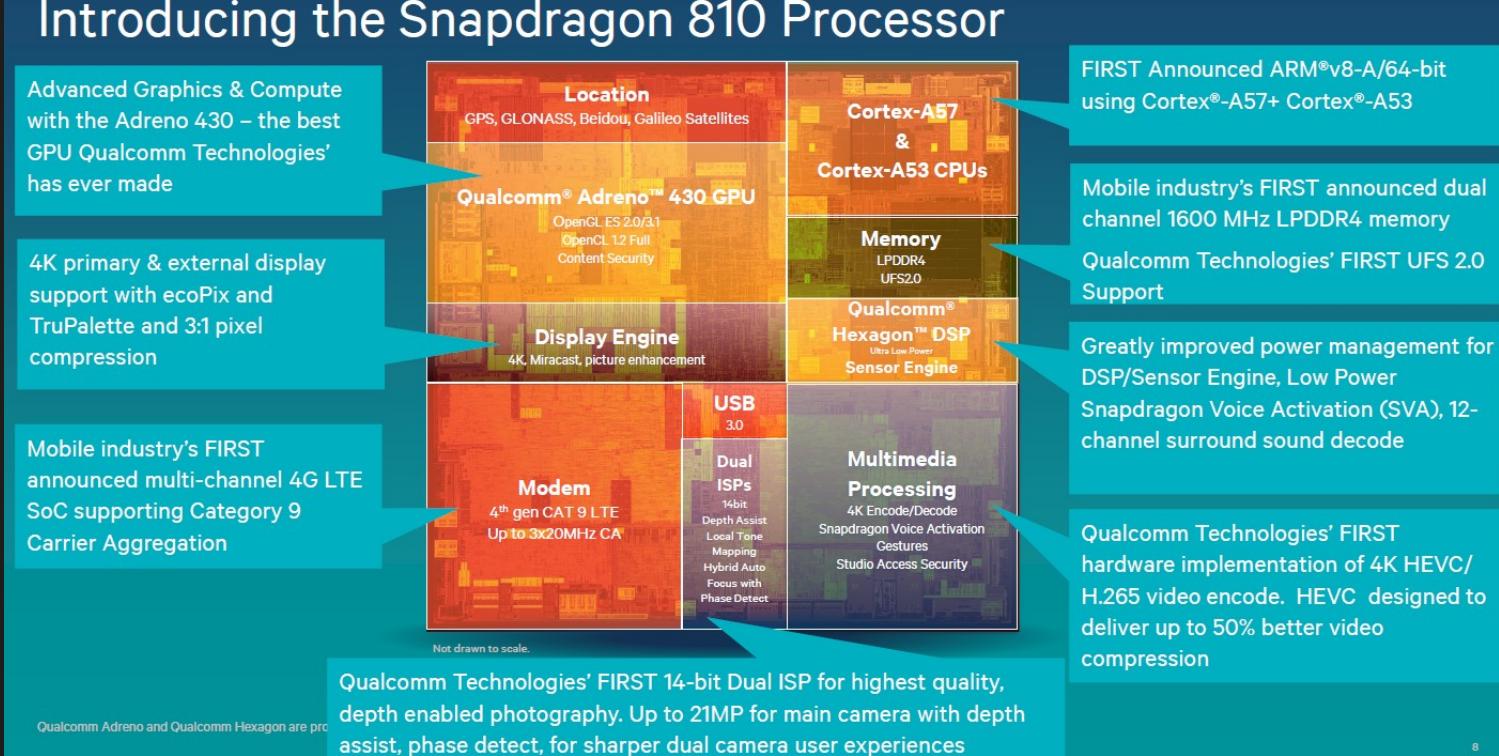
## Solution Qualcomm Snapdragon

Le leader du marché en terme de part de marché est Qualcomm, grâce à sa famille Snapdragon dédiée au marché des terminaux mobiles.



## Fonctions matérielles de l'architecture interne de la famille Qualcomm Snapdragon 810

**Introducing the Snapdragon 810 Processor**



The diagram illustrates the internal architecture of the Qualcomm Snapdragon 810 Processor, featuring a central processing unit (CPU) and various supporting components:

- Location:** GPS, GLONASS, Beidou, Galileo Satellites
- Cortex-A57 & Cortex-A53 CPUs:** FIRST Announced ARM®v8-A/64-bit using Cortex®-A57+ Cortex®-A53
- Memory:** LPDDR4 UFS2.0
- Qualcomm® Hexagon™ DSP (Ultra Low Power Sensor Engine):** Mobile industry's FIRST announced dual channel 1600 MHz LPDDR4 memory, Qualcomm Technologies' FIRST UFS 2.0 Support
- Display Engine:** 4K, Miracast, picture enhancement
- Modem:** 4<sup>th</sup> gen CAT 9 LTE, Up to 3x20MHz CA
- USB 3.0:** Dual ISPs 14bit Depth Assist, Local Tone Mapping, Hybrid Auto Focus with Phase Detect
- Multimedia Processing:** 4K Encode/Decode, Snapdragon Voice Activation, Gestures, Studio Access Security

**Advanced Graphics & Compute with the Adreno 430 – the best GPU Qualcomm Technologies' has ever made**

**4K primary & external display support with ecoPix and TruPalette and 3:1 pixel compression**

**Mobile industry's FIRST announced multi-channel 4G LTE SoC supporting Category 9 Carrier Aggregation**

**Qualcomm Technologies' FIRST 14-bit Dual ISP for highest quality, depth enabled photography. Up to 21MP for main camera with depth assist, phase detect, for sharper dual camera user experiences**

Not drawn to scale.

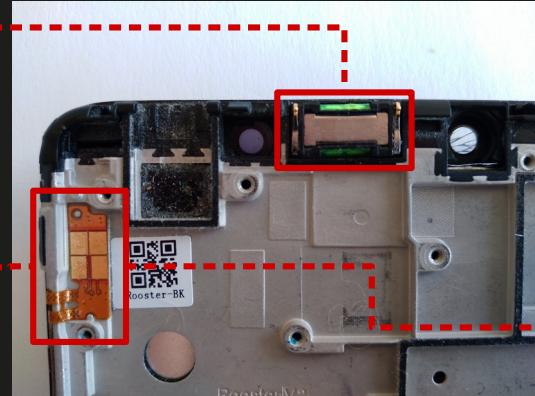
Qualcomm Adreno and Qualcomm Hexagon are pro

# AP – APPLICATION PROCESSOR

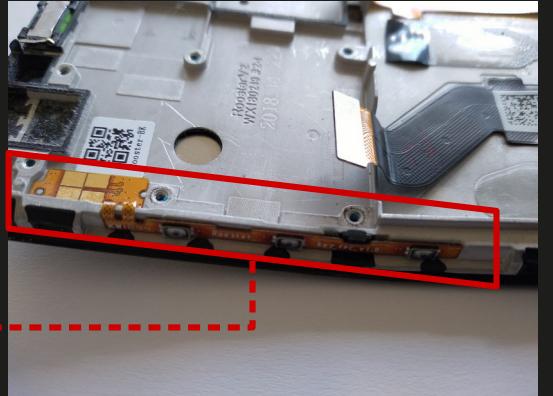
Exemple smartphone (Nokia 3.1 Plus, 2018)



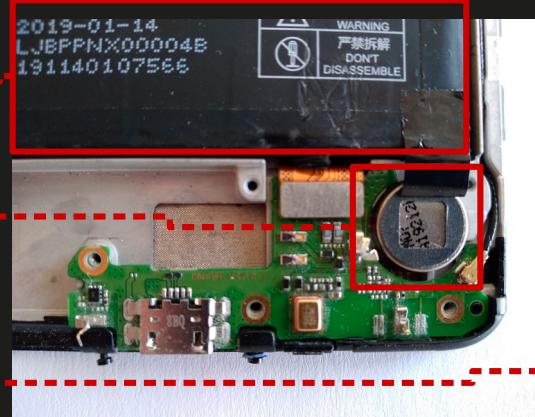
Speaker



Switches connector



Battery  
3400 mAh



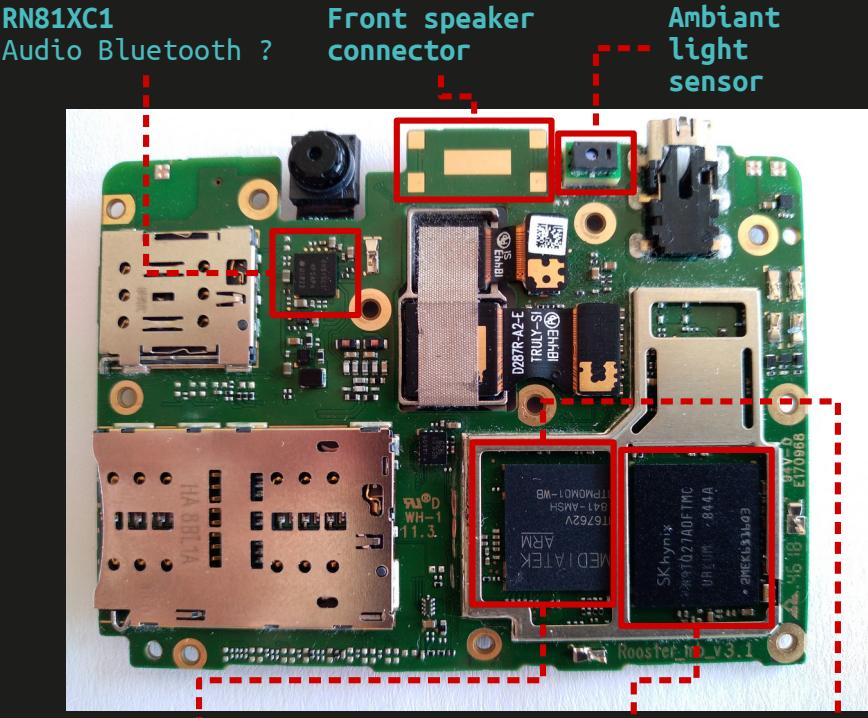
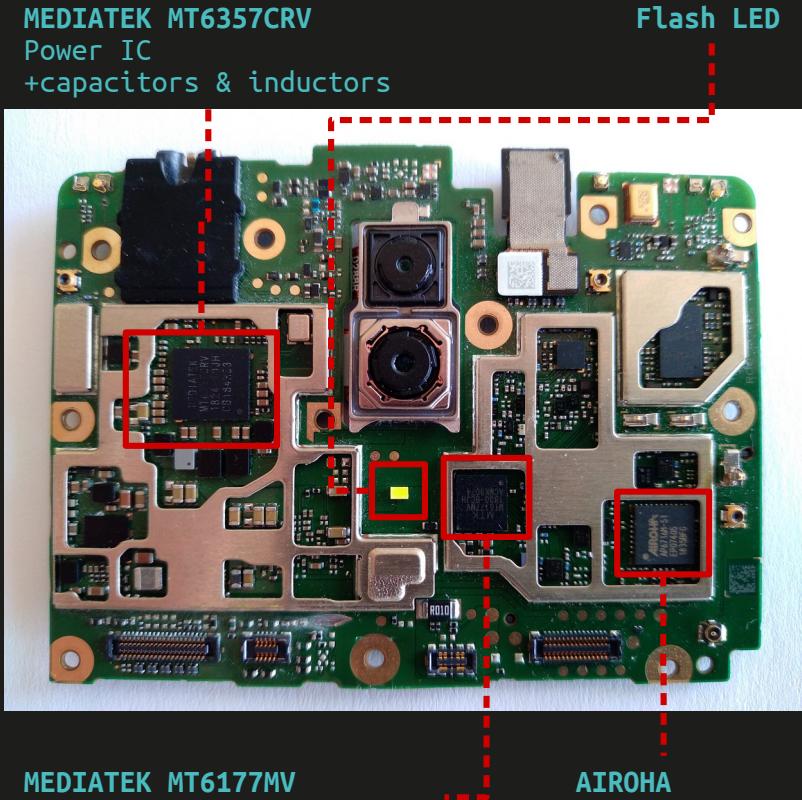
Loudspeaker



Microphone

# AP – APPLICATION PROCESSOR

Exemple smartphone (Nokia 3.1 Plus, 2018)



**MEDIATEK MT6762V (Helio P22)**  
Qualcomm SDM439 Snapdragon (2018)  
ARM v8-A (64-bit), Cortex-A53  
8 cores, 2 GHz, TSMC FinFET 12 nm  
2-core GPU, DSP

**MEDIATEK MT6177MV**  
Intermediate Frequency IC

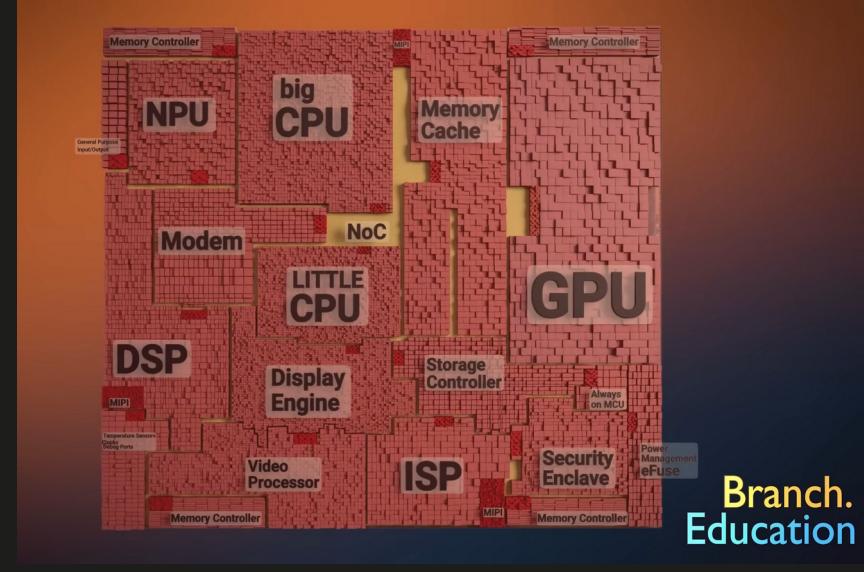
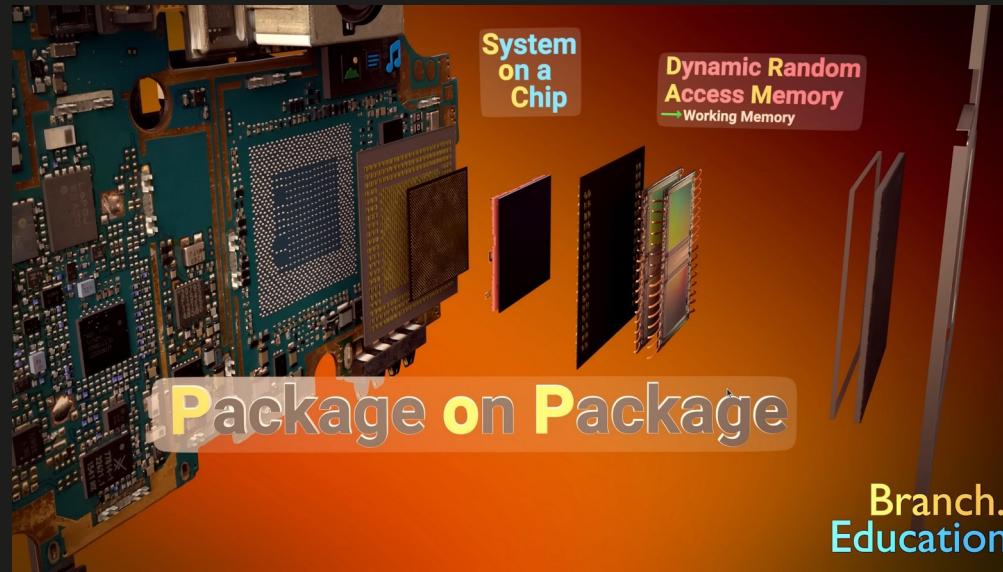
**AIROHA AP6716M-51**  
RF IC

**RAM**  
2 GB  
LPDDR3  
(Low-Power)

# How do Smartphone CPUs Work? || Inside the System on a Chip Branch Education

<https://www.youtube.com/watch?v=NKfW8ijmRQ4&t=136s>

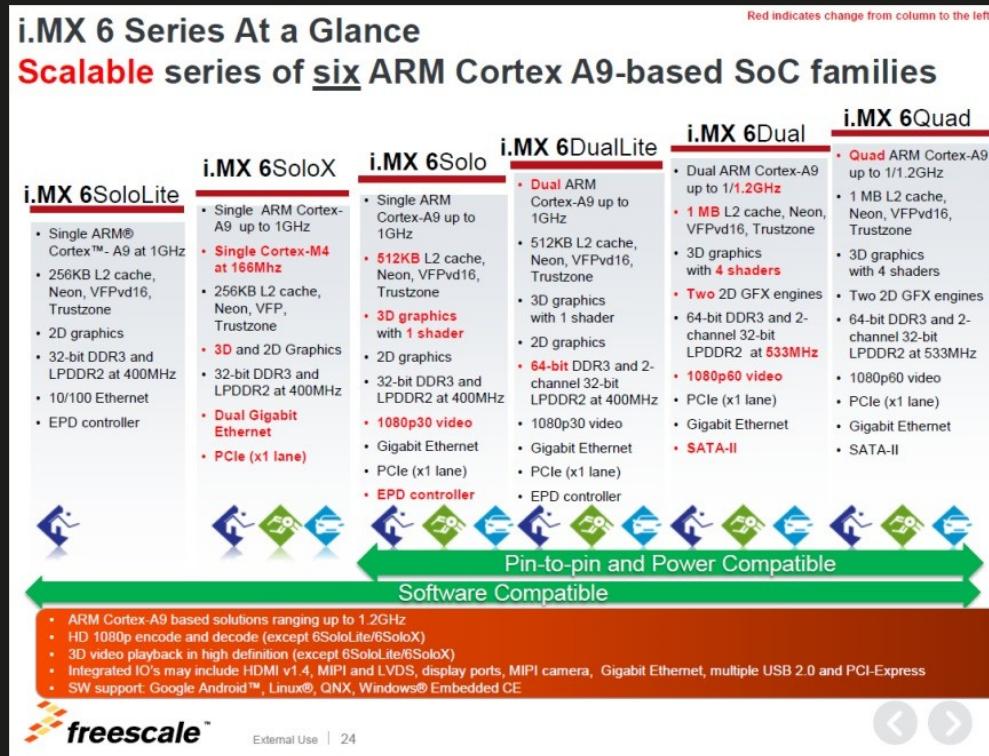
<https://www.youtube.com/watch?v=NKfW8ijmRQ4&t=1001s>



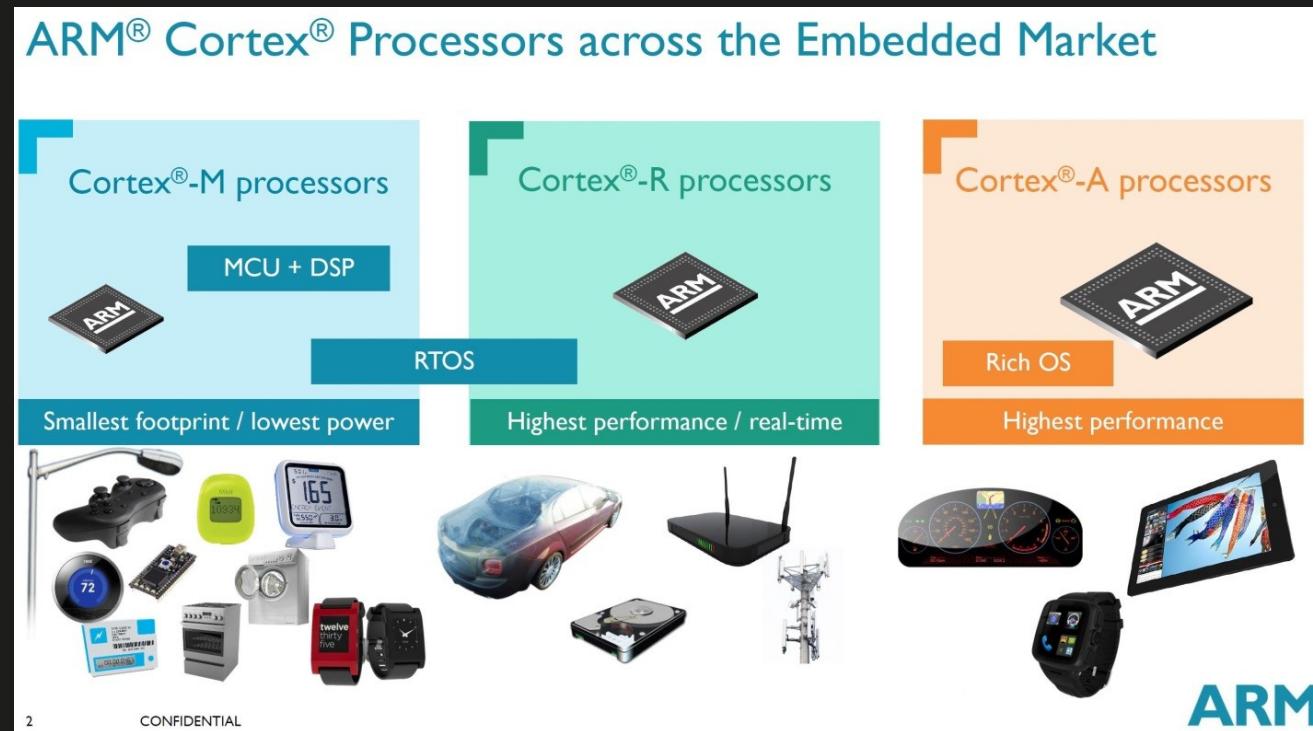
## Solution ARM Cortex-A

Les deux leaders du marché hors terminaux mobiles sont Texas Instruments et Freescale, deux fondeurs offrant de larges communautés d'utilisateurs.

Observons la famille i.MX6 de Freescale :



Hors marché des terminaux mobiles, sur le marché de l'embarqué les architectures Cortex-A de ARM sont également reines. Le « A » signifie *Application*.



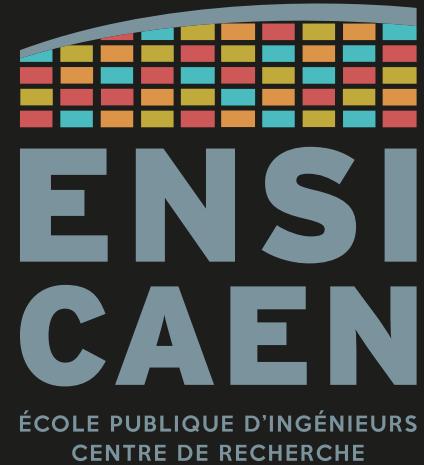
# - DSP -

# DIGITAL SIGNAL PROCESSOR

Applications

Architecture

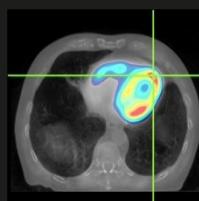
Texas Instruments



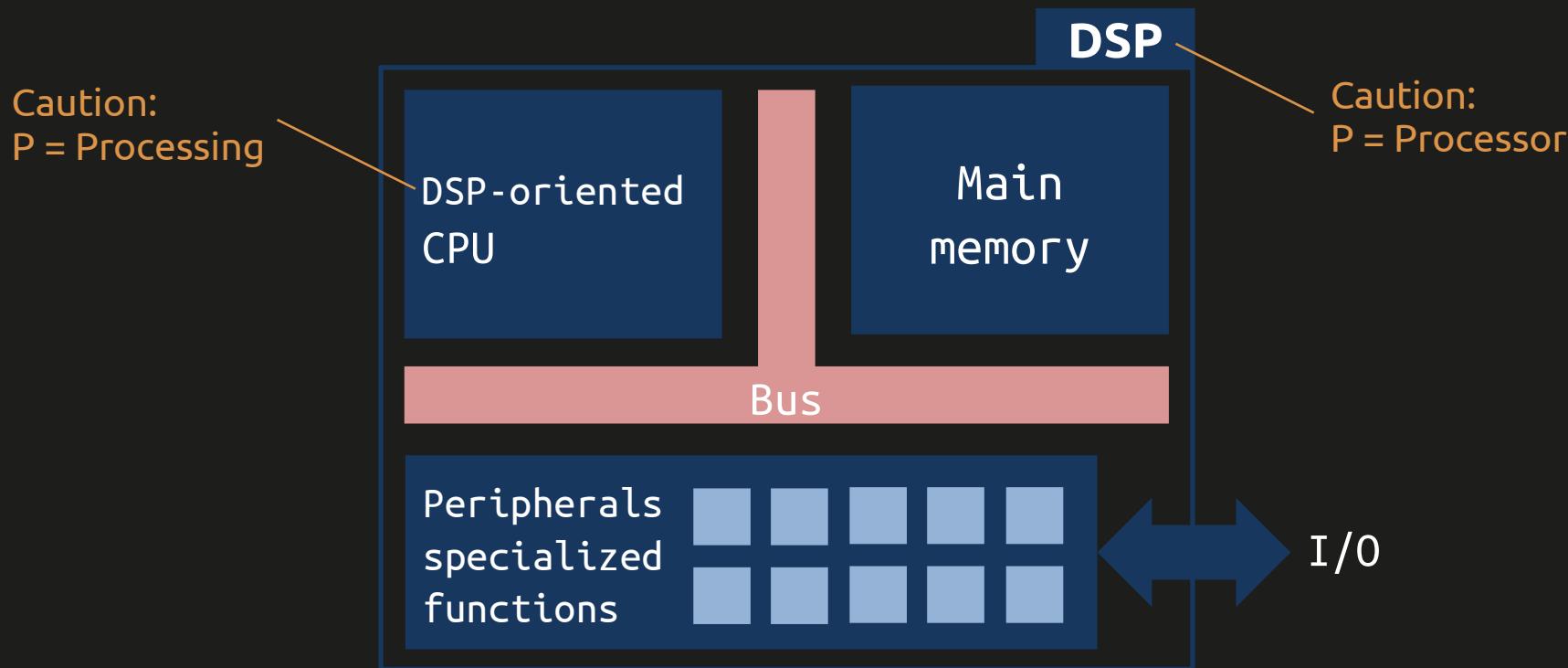
# DSP – DIGITAL SIGNAL PROCESSOR

## Applications

Les **DSP (Digital Signal Processor)** sont dédiés aux applications impliquant du Traitement Numérique du Signal (TNS ou DSP ou *Digital Signal Processing*).



Les DSP sont très proches des MCU : ce sont des systèmes autonomes.  
Leur CPU est néanmoins spécialisé pour le calcul numérique.

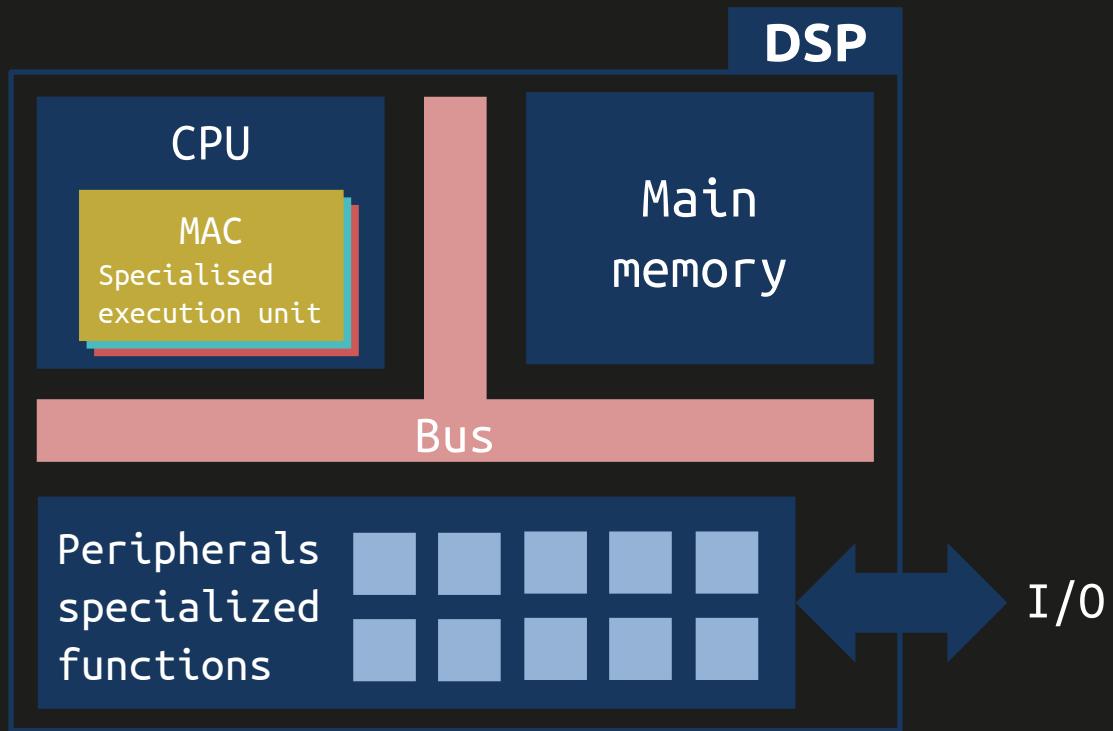


Leur CPU possède des extensions d'instructions et unités d'exécution dédiées au calcul de **MAC (Multiply Accumulate)** ou **SOP (Som Of Products)**. Il s'agit des opérations élémentaires rencontrées dans tout algorithme de Traitement Numérique du Signal.

Expansion of the Danielson-Lanczos Lemma to 8 terms:

$$\begin{aligned}
 F(n) = & \sum_{k=0}^{N/8-1} x(8k)e^{\frac{-j2\pi kn}{8}} + W_{\frac{N}{4}}^n \sum_{k=0}^{N/8-1} x(8k+4)e^{\frac{-j2\pi kn}{8}} + \\
 & W_{\frac{N}{2}}^n \sum_{k=0}^{N/8-1} x(8k+2)e^{\frac{-j2\pi kn}{8}} + W_{\frac{N}{2}}^n W_{\frac{N}{4}}^n \sum_{k=0}^{N/8-1} x(8k+6)e^{\frac{-j2\pi kn}{8}} + \\
 & W_N^n \sum_{k=0}^{N/8-1} x(8k+1)e^{\frac{-j2\pi kn}{8}} + W_N^n W_{\frac{N}{4}}^n \sum_{k=0}^{N/8-1} x(8k+5)e^{\frac{-j2\pi kn}{8}} + \\
 & W_N^n W_{\frac{N}{2}}^n \sum_{k=0}^{N/8-1} x(8k+3)e^{\frac{-j2\pi kn}{8}} + W_N^n W_{\frac{N}{2}}^n W_{\frac{N}{4}}^n \sum_{k=0}^{N/8-1} x(8k+7)e^{\frac{-j2\pi kn}{8}}
 \end{aligned}$$

CPU avec unités d'exécution dédiées au calcul de MAC ou SOP. Le jeu d'instructions (ISA) dispose d'instructions spécifiques pour exploiter ces EU.

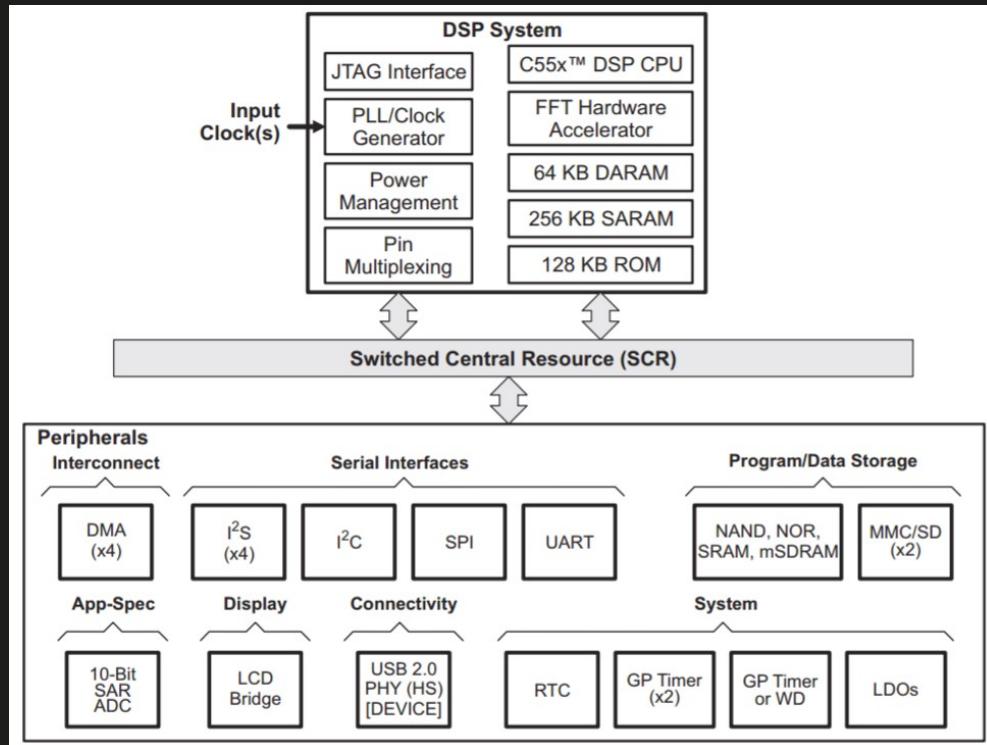
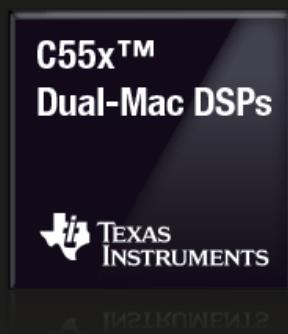


MAC = SOP

MAC : Multiply-Accumulate  
SOP : Som of Products

ISA : Instruction Set Architecture  
EU : Execution Unit

Observons la solution C5500 DSP proposée par Texas Instruments, l'une des solutions phares du fondeur américain.

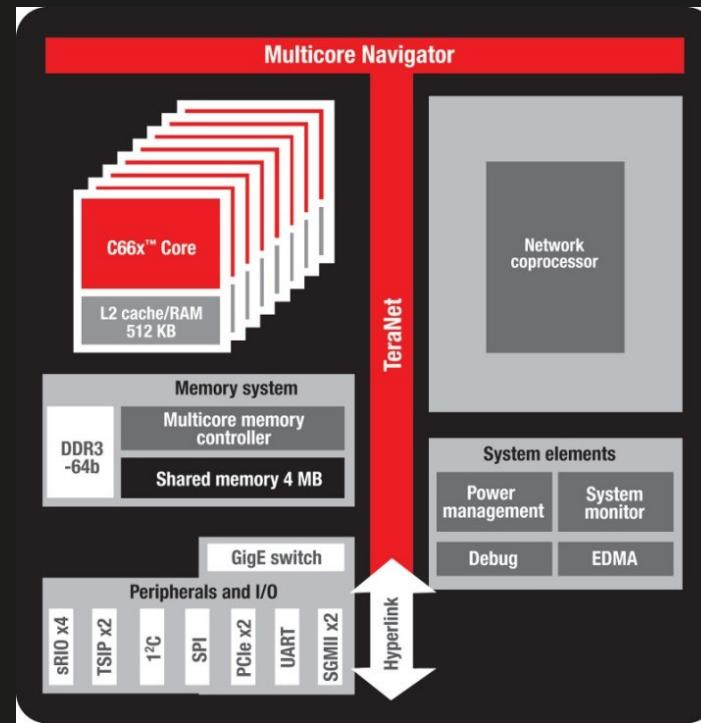
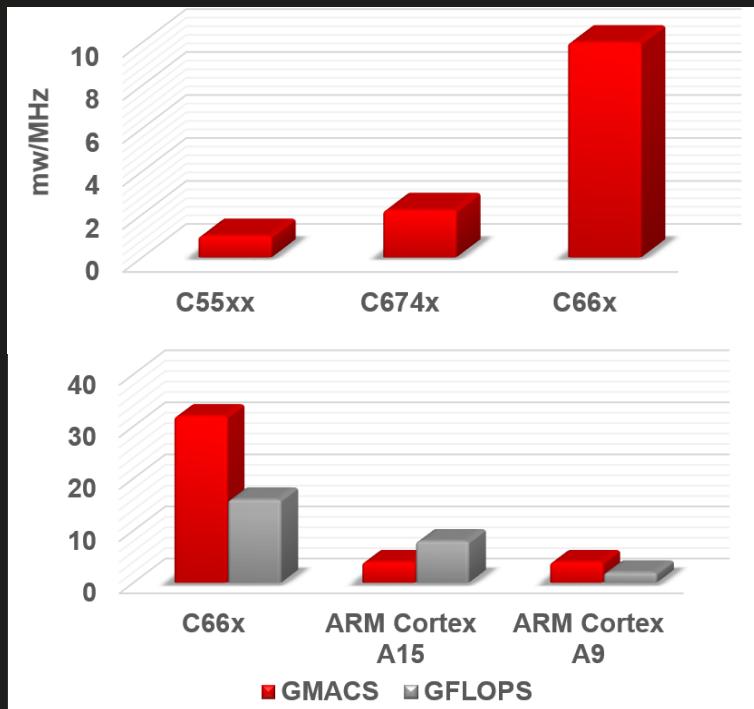


Étudions cet extrait de sa datasheet,  
le résumé des spécifications du CPU.

### 1.1 Features

- CORE:
  - High-Performance, Low-Power, TMS320C55x Fixed-Point Digital Signal Processor
    - 20-, 10-ns Instruction Cycle Time
    - 50-, 100-MHz Clock Rate
    - One or Two Instructions Executed per Cycle
    - Dual Multiply-and-Accumulate Units (Up to 200 Million Multiply-Accumulates per Second [MMACS])
    - Two Arithmetic and Logic Units (ALUs)
    - Three Internal Data and Operand Read Buses and Two Internal Data and Operand Write Buses
    - Software-Compatible with C55x Devices
    - Industrial Temperature Devices Available
  - 320KB of Zero-Wait State On-Chip RAM, Composed of:
    - 64KB of Dual-Access RAM (DARAM), 8 Blocks of 4K x 16-Bit
    - 256KB of Single-Access RAM (SARAM), 32 Blocks of 4K x 16-Bit
  - 128KB of Zero Wait-State On-Chip ROM (4 Blocks of 16K x 16-Bit)
  - Tightly Coupled FFT Hardware Accelerator

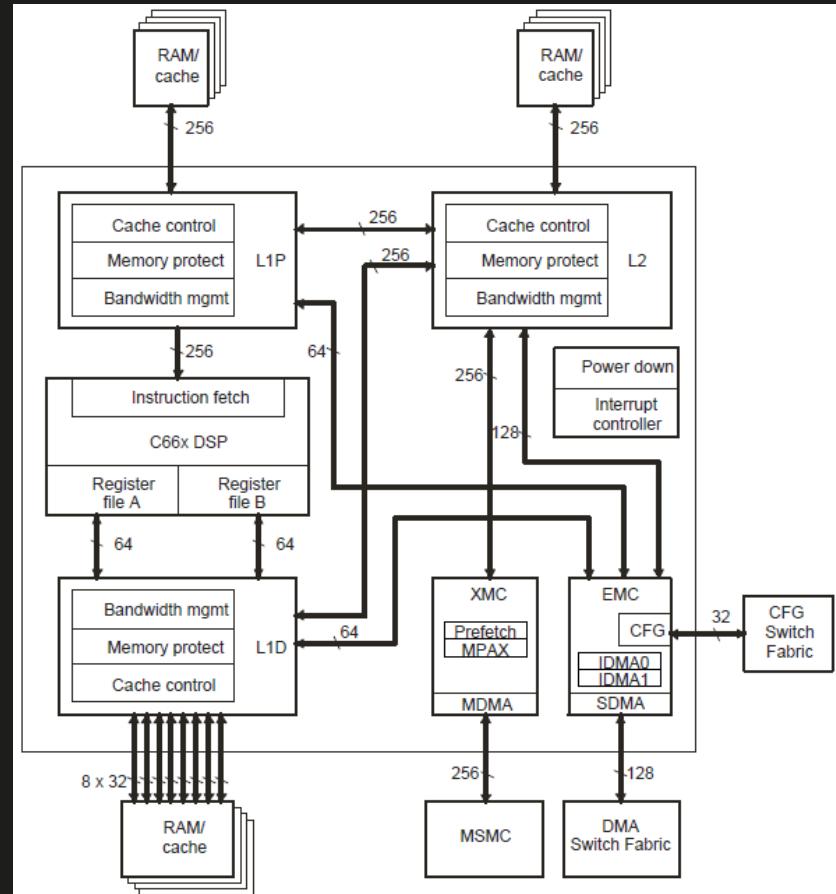
Passons maintenant à la gamme Keystone C6600 proposée par Texas Instruments. Cette architecture DSP est l'une des plus performantes du marché.



## Cœur (CorePac) C6600 développé par TI.

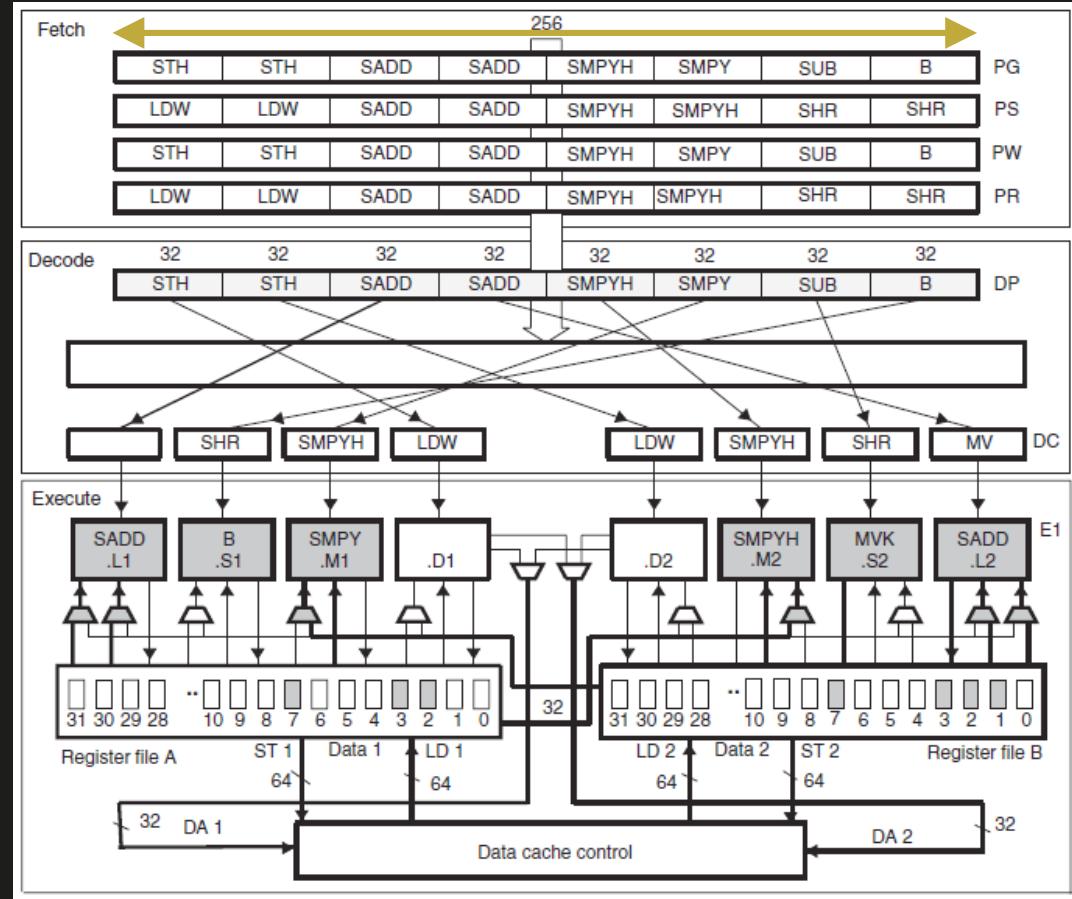
Hiérarchie mémoire configurable en cache ou SRAM adressable sans perte de bande passante.

Modèle UMA ou NUMA configurable pour chaque cœur.

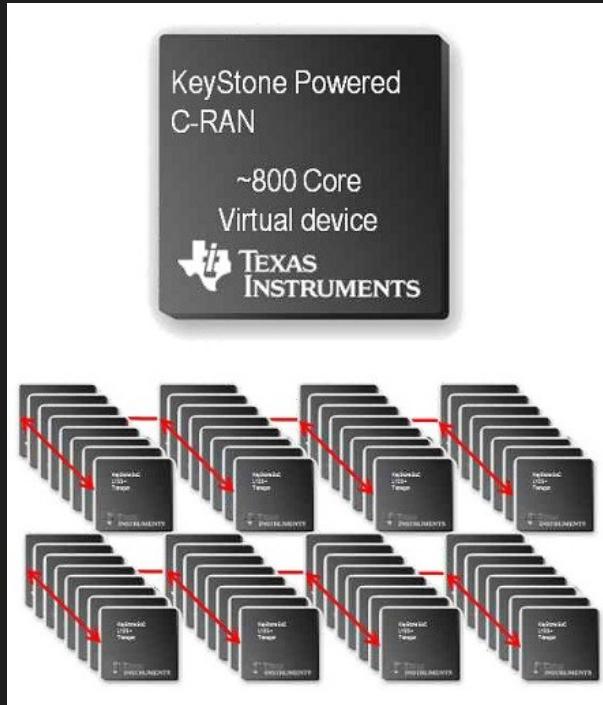


Cœur C6600 avec :

- pipeline matériel VLIW (*Very Long Instruction Word*) à 14 étages
- pipeline logiciel d'une largeur maximale de 8 instructions.



Ces DSP sont pensés aussi bien pour un usage en parallèle que pour être chaînés afin d'encaisser des chaînes des traitements profondes (reliés en *daisy-chain*).

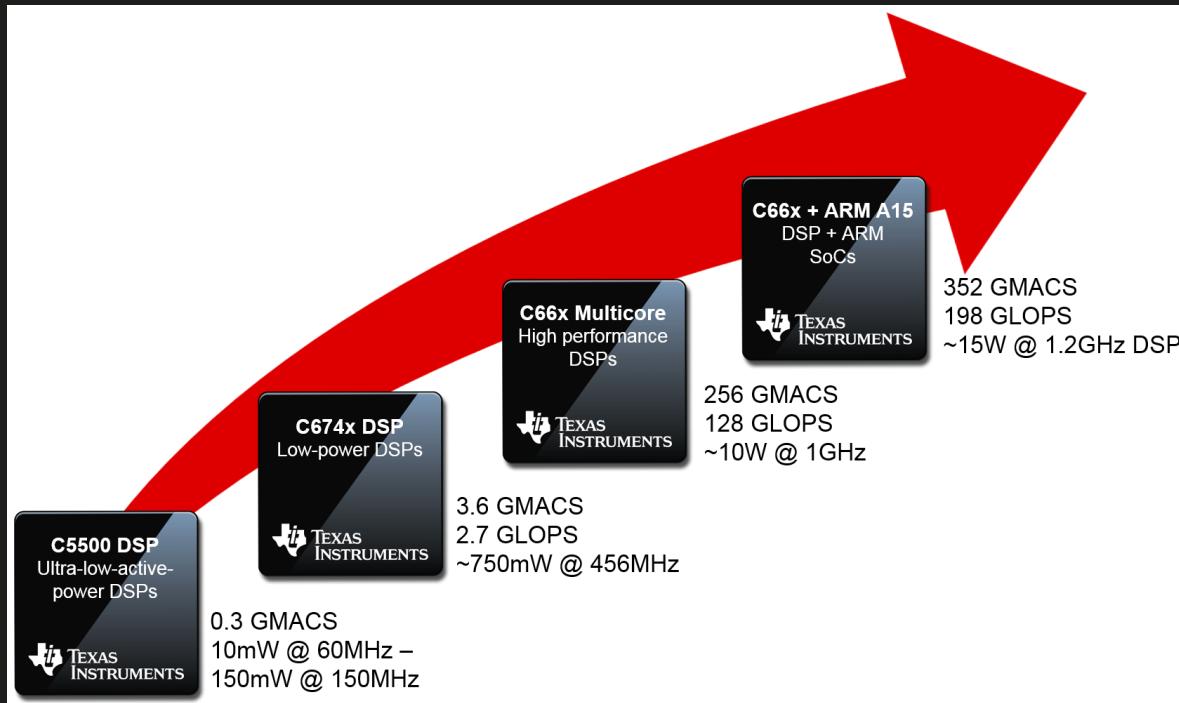


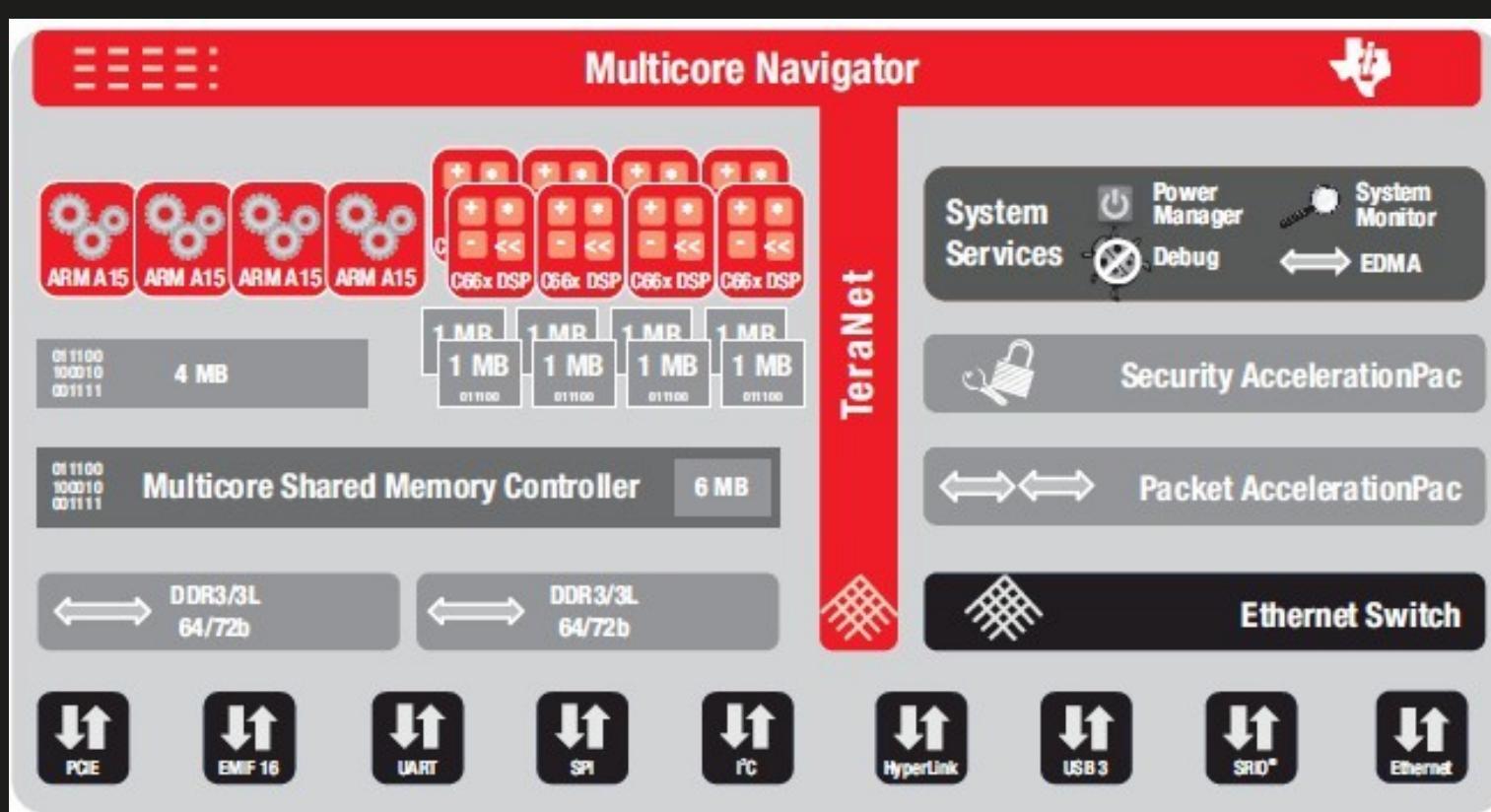
Avantage du *daisy-chain* :

## Exemple Texas Instruments : Keystone II

Mais ce n'est pas tout, TI propose la gamme Keystone II. Il s'agit de SoC de type AP spécialisés pour les applications DSP.

Principales applications visées : les télécommunications.



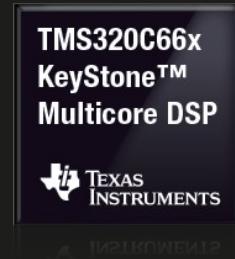


## Acteurs du marché

Le leader historique et actuel incontesté du marché est Texas Instruments.  
TI a été la première société à proposer des processeurs DSP en 1982.



TMS32020 (1982)  
Up to 8,77 MIPS



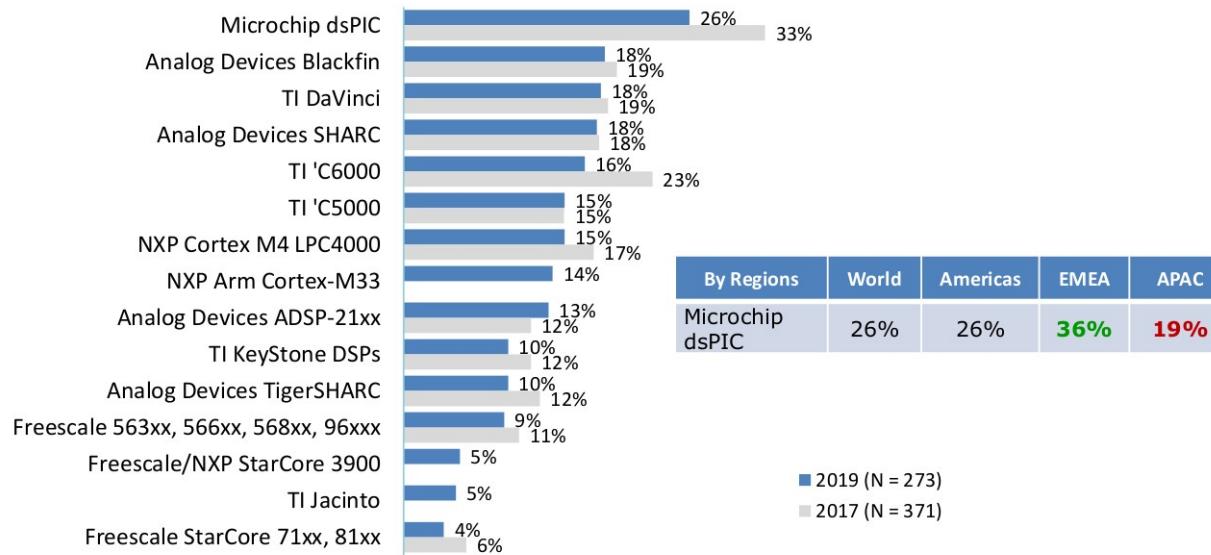
TMS320C6678 (2010)  
Up to 256 GMACS

Voici la gamme de processeurs proposée par Texas Instruments.

Microcontrollers (MCUs)		ARM®-based Processors			Digital Signal Processors		
16-bit Ultra Low Power MCU	32-bit Real-Time MCU	32-bit ARM MCU	32-bit ARM Processors for Performance Applications	Application Processors	Singlecore DSP	Multicore DSP	Ultra Low Power DSP
<ul style="list-style-type: none"> <li>• MSP430™ </li> <li>• C2000™</li> </ul>	<ul style="list-style-type: none"> <li>• TMS570 Cortex® R4</li> <li>• RM4 Cortex® R4F</li> <li>• TMS470M Cortex® M3 Automotive</li> </ul>	<ul style="list-style-type: none"> <li>• Sitara™ Cortex A and ARM9</li> <li>• KeyStone Cortex® A15 and Cortex® A15 + DSP</li> </ul>	<ul style="list-style-type: none"> <li>• OMAP™ Processors</li> <li>• DaVinci™ Video Processors</li> </ul>	<ul style="list-style-type: none"> <li>• C6000™ Power Optimized</li> </ul>	<ul style="list-style-type: none"> <li>• KeyStone Multicore DSP+ARM</li> <li>• C6000™ Multicore</li> </ul>	<ul style="list-style-type: none"> <li>• C5000™ </li> </ul>	



**Which of the following DSP chip families would you consider for your next embedded project?**



# - GPU -

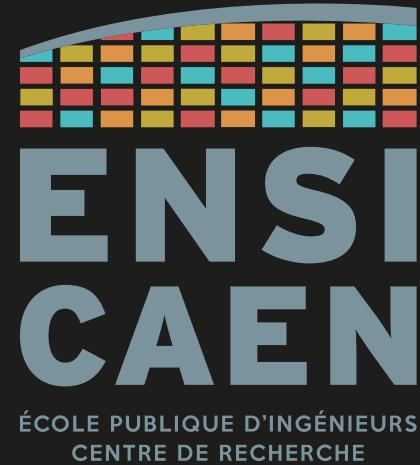
# GRAPHICS PROCESSING UNIT

Applications

Architecture

Produits Nvidia

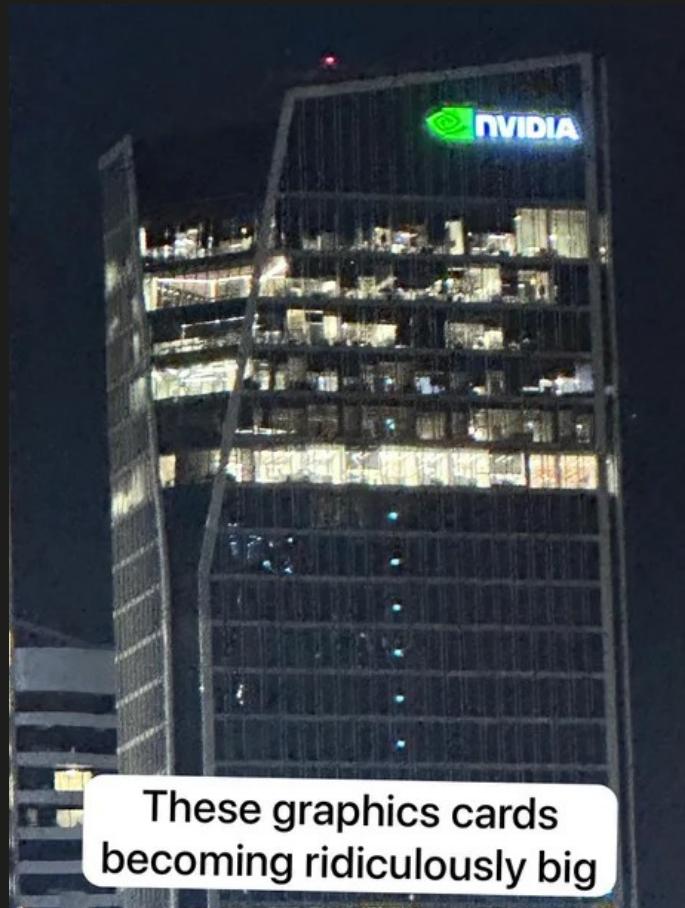
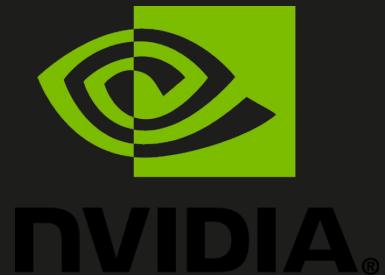
Marchés



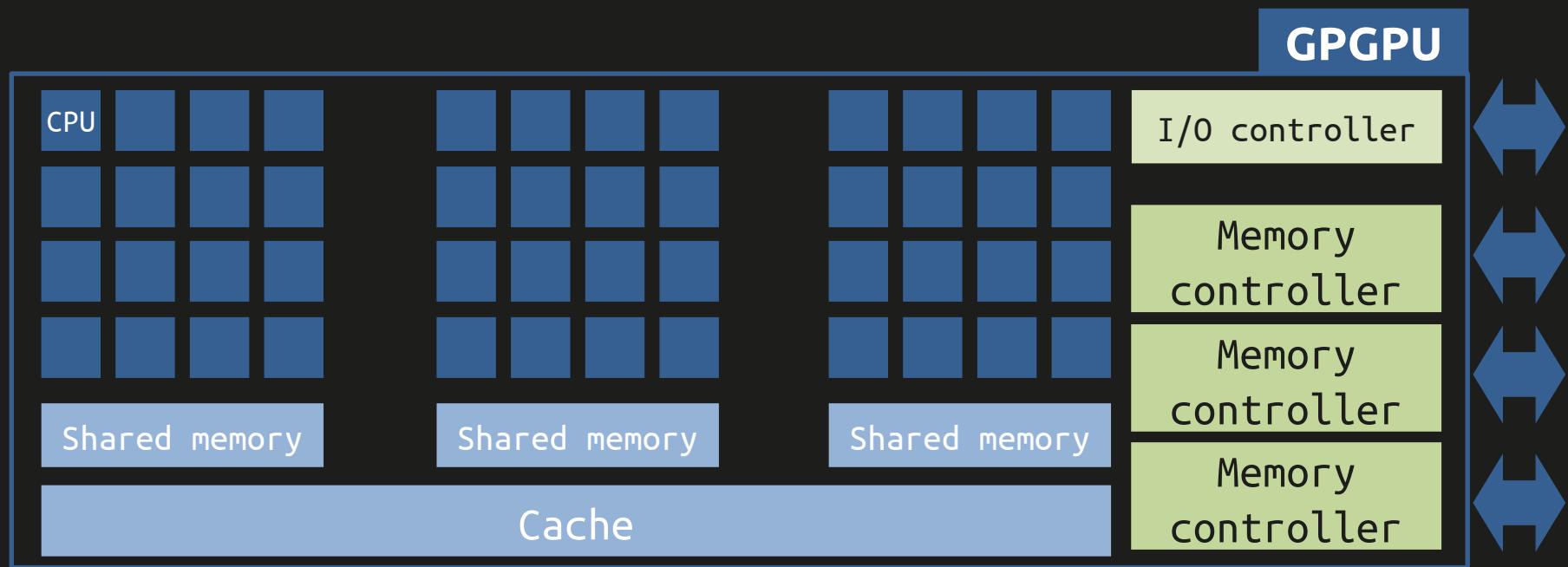
## Applications

Les **GPU (*Graphics Processing Unit*)** sont des **coprocesseurs** de traitement spécialisés pour le calcul intensif.

Depuis quelques années, nous parlons de GPGPU (*General Purpose GPU*), GPU dédié au calcul massif au sens large. Les applications sont multiples : finance, recherche et sciences, imagerie médicale, jeux vidéos ...



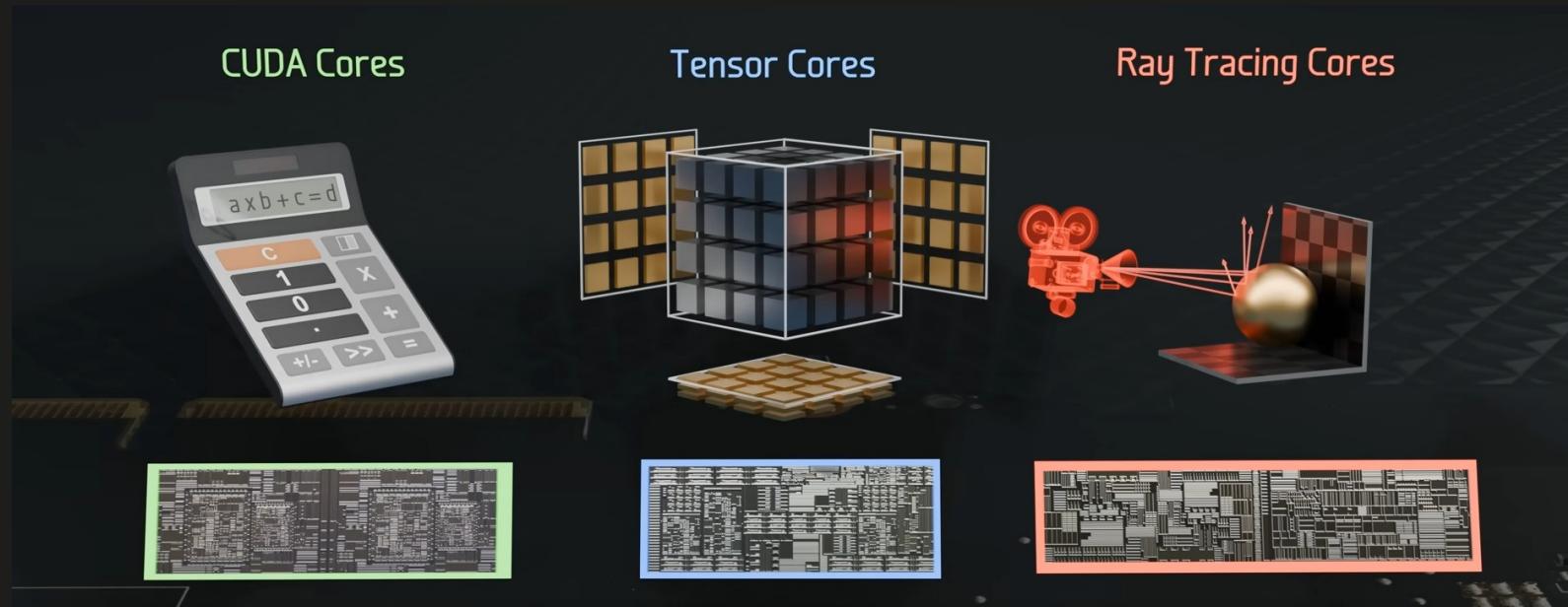
Les GPU possèdent un modèle mémoire réparti non uniforme de type NUMA (*Non Uniform Memory Access*), permettant un clonage des données à traiter et un parallélisme d'exécution. Ils intègrent une architecture massivement parallèle.



### How do Graphics Cards Work? Exploring GPU Architecture Branch Education

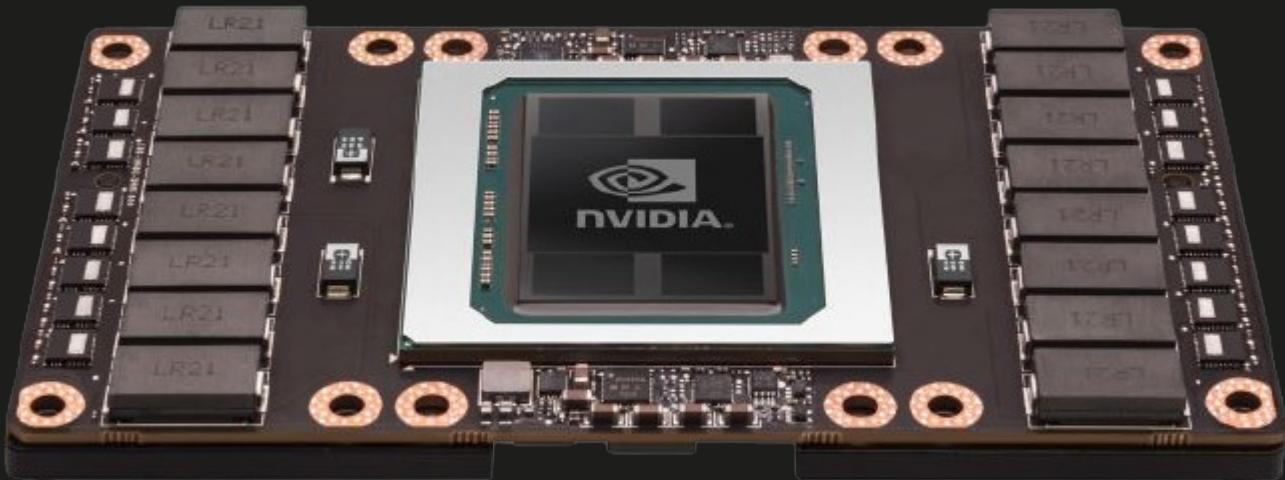
<https://www.youtube.com/watch?v=h9Z4oGN89MU>

04:56 - GPU GA102 Architecture



## Solution Nvidia : la carte Tesla P100

Observons le potentiel de la carte Tesla P100 proposée par Nvidia courant 2016 et dédiée aux data center les plus avancés du moment. Basée sur GPU GP100.



#### SPECIFICATIONS

GPU Architecture	NVIDIA Pascal
NVIDIA CUDA® Cores	3584
Double-Precision Performance	5.3 TeraFLOPS
Single-Precision Performance	10.6 TeraFLOPS
Half-Precision Performance	21.2 TeraFLOPS
GPU Memory	16 GB CoWoS HBM2
Memory Bandwidth	732 GB/s
Interconnect	NVIDIA NVLink
Max Power Consumption	300 W
ECC	Native support with no capacity or performance overhead
Thermal Solution	Passive
Form Factor	SXM2
Compute APIs	NVIDIA CUDA, DirectCompute, OpenCL™, OpenACC

TeraFLOPS measurements with NVIDIA GPU Boost™ technology

## Solution Nvidia : architecture Pascal

**12X TRAINING PERFORMANCE**

LEAP IN NEURAL NETWORK TRAINING PERFORMANCE WITH NEW **NVIDIA PASCAL ARCHITECTURE**

**150B TRANSISTORS**

FABRICATED WITH **16 NANOMETER FINFET** FOR UNPRECEDENTED ENERGY EFFICIENCY

**3X MEMORY BANDWIDTH**

WITH **CoWoS® WITH HBM2** COMPARED TO NVIDIA MAXWELL™ ARCHITECTURE FOR BIG DATA WORKLOADS

**5X INTERCONNECT BANDWIDTH**

WITH **NVIDIA NVLINK™** FOR MAXIMUM APPLICATION SCALABILITY

## Solution Nvidia : Architecture du GPU GP100

Les GPU intègrent un grand nombre de CPU à pipeline classique mais avec des EU vectorielles SIMD.

EU = Execution Unit

SIMD = Single Instruction Multiple Data

GPC = Graphics Processing Cluster

TCP = Texture Processing Cluster

SM = Streaming Multiprocessor  
(multithreaded processor)

Warp = thread of SIMD instructions

DP = Double Precision

LD/ST = Load/Store

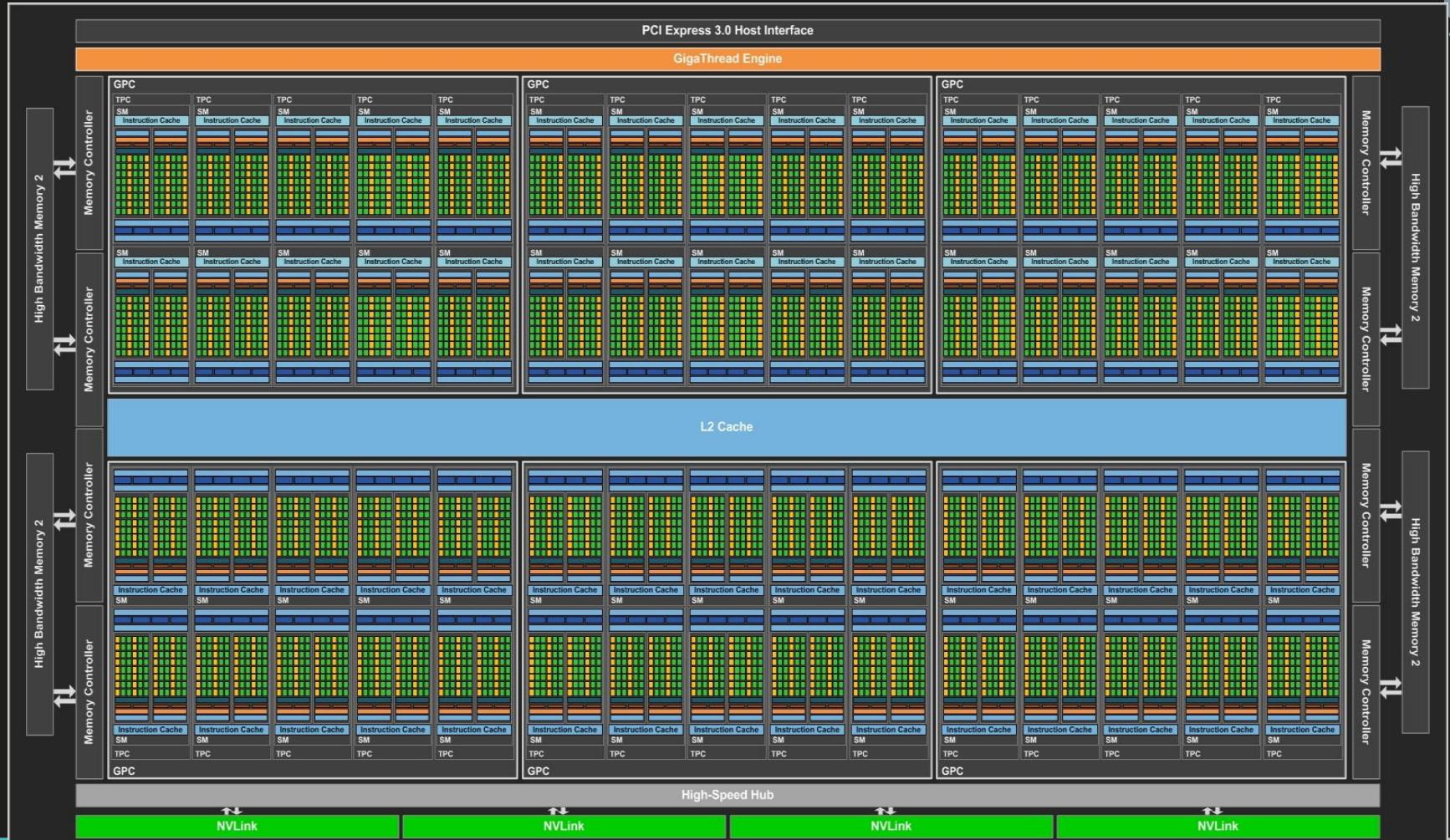
SFU = Special Function Unit

Tex = Texture



# GPU – GRAPHICS PROCESSING UNITS

## Solution Nvidia : Architecture du GPU GP100



## Le GPU Nvidia GP100 en chiffres

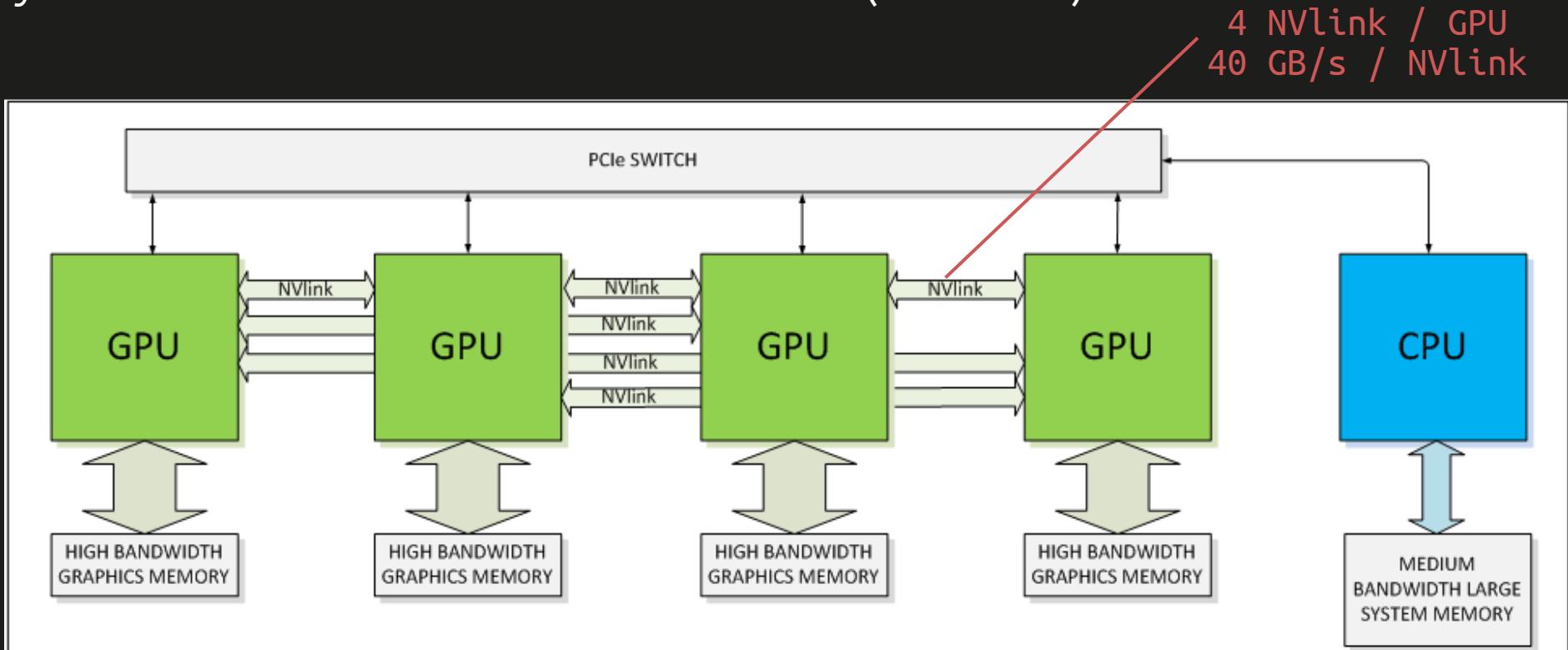
- 6 Graphics Processing Clusters
- 30 Texture Processing Clusters (5 / GPC)
- 60 Streaming Multiprocessors (2 / TPC)
- 3840 single precision cores (64 / SM)
- 1920 double precision unit (32 / SM)
- 240 texture units (4 / SM)
- 8 memory controllers
  - $8 \times 512 \text{ KB} = 4096 \text{ KB}$  L2 cache
  - 4 pairs that control HBM2 DRAM

Tesla Products	Tesla K40	Tesla M40	Tesla P100
<b>GPU</b>	GK110 (Kepler)	GM200 (Maxwell)	GP100 (Pascal)
<b>SMs</b>	15	24	56
<b>TPCs</b>	15	24	28
<b>FP32 CUDA Cores / SM</b>	192	128	64
<b>FP32 CUDA Cores / GPU</b>	2880	3072	3584
<b>FP64 CUDA Cores / SM</b>	64	4	32
<b>FP64 CUDA Cores / GPU</b>	960	96	1792
<b>Base Clock</b>	745 MHz	948 MHz	1328 MHz
<b>GPU Boost Clock</b>	810/875 MHz	1114 MHz	1480 MHz
<b>Peak FP32 GFLOPs<sup>1</sup></b>	5040	6840	10600
<b>Peak FP64 GFLOPs<sup>1</sup></b>	1680	210	5300
<b>Texture Units</b>	240	192	224
<b>Memory Interface</b>	384-bit GDDR5	384-bit GDDR5	4096-bit HBM2
<b>Memory Size</b>	Up to 12 GB	Up to 24 GB	16 GB
<b>L2 Cache Size</b>	1536 KB	3072 KB	4096 KB
<b>Register File Size / SM</b>	256 KB	256 KB	256 KB
<b>Register File Size / GPU</b>	3840 KB	6144 KB	14336 KB
<b>TDP</b>	235 Watts	250 Watts	300 Watts
<b>Transistors</b>	7.1 billion	8 billion	15.3 billion
<b>GPU Die Size</b>	551 mm <sup>2</sup>	601 mm <sup>2</sup>	610 mm <sup>2</sup>
<b>Manufacturing Process</b>	28-nm	28-nm	16-nm FinFET

<sup>1</sup> The GFLOPS in this chart are based on GPU Boost Clocks.

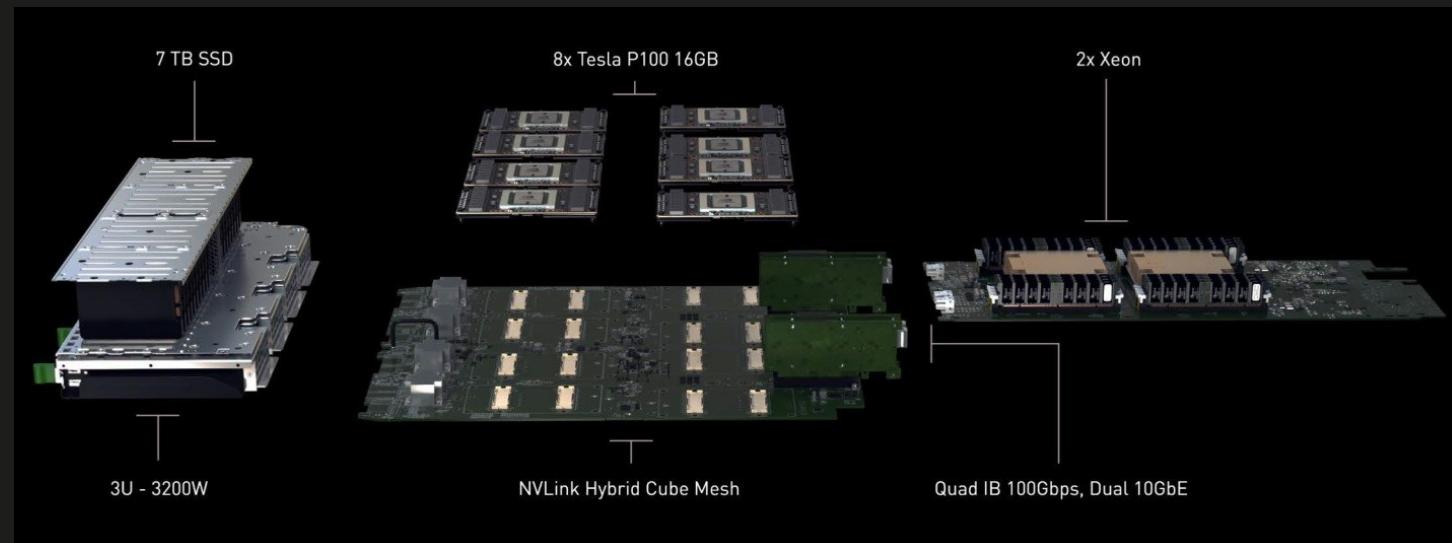
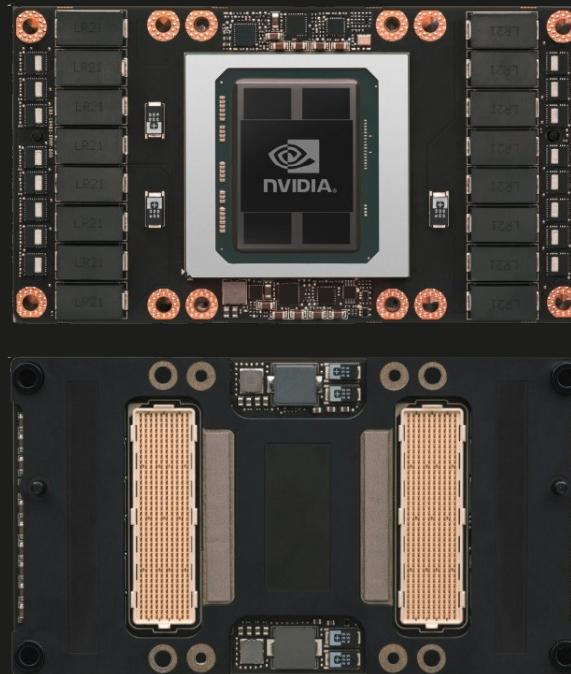
Note : la carte Tesla P100 exploite 56 SM sur les 60 SM disponibles dans le GP100.

## Système d'interconnexion et de communication (Tesla P100)

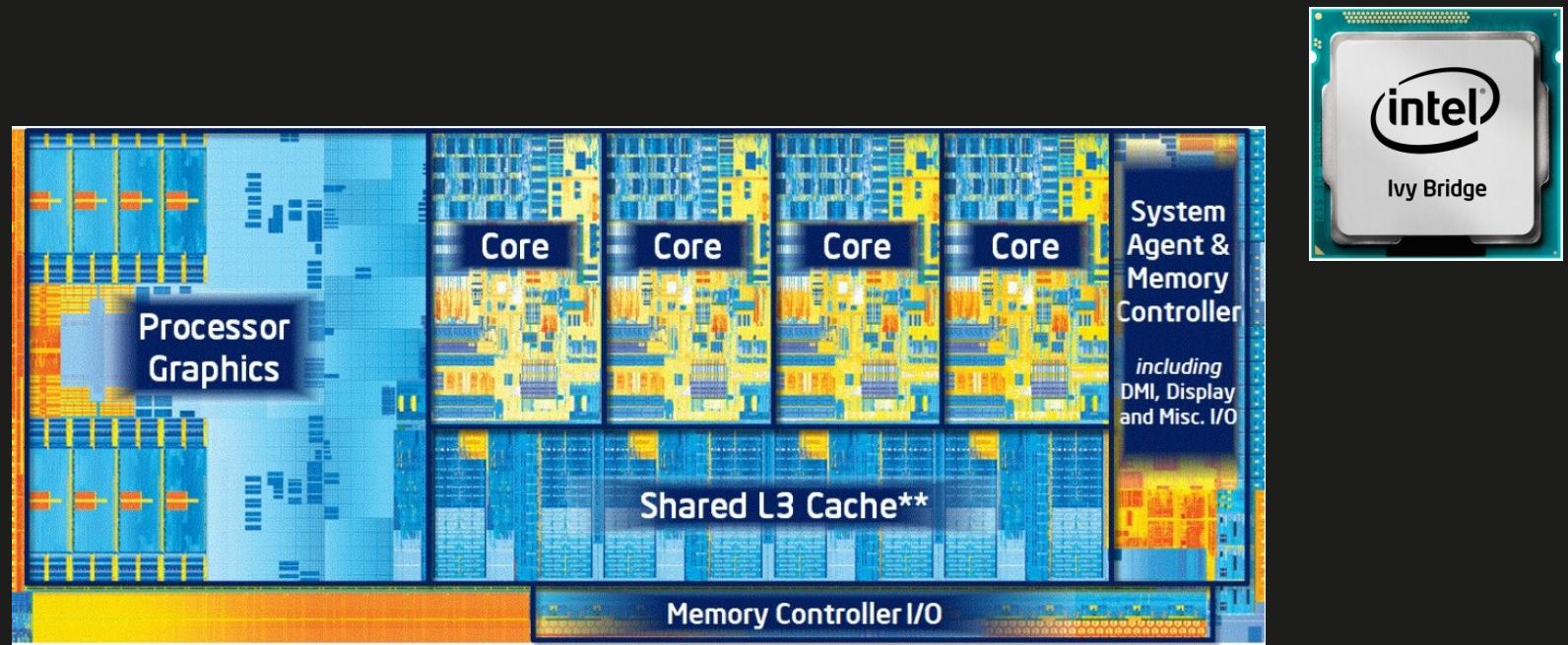


## Solution Nvidia : exemple d'application

Exemple d'application utilisant la carte Nvidia Tesla P100.



Le leader incontesté du marché des GPU/IGP en terme de part est Intel grâce aux coprocesseurs graphiques IGP (*Integrated Graphics Unit*) intégrés dans une grande partie de leurs gammes processeurs GPP (plus de 70% en 2016).



Toutefois, le leader des solutions hautes performances externes est l'américain Nvidia.



Tesla K20C



NVIDIA®

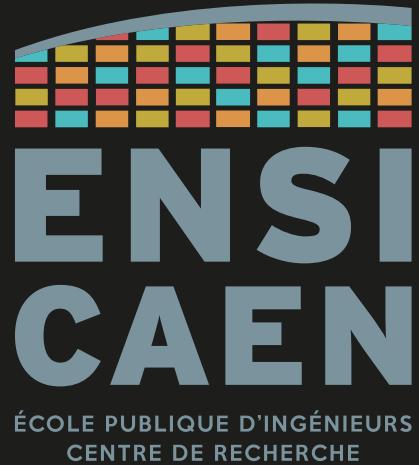


Tesla P100

GeForce  
RTX 4090



- NPU -  
NEURAL PROCESSING UNIT



Les **NPU (*Neural Processing Unit*)**, parfois appelées *AI accelerator*, sont des processeurs spécialistes.

Les GPU sont historiquement les premiers processeurs sur lesquels les algorithmes d'IA ont été exécutés. Leur architecture massivement parallèle et leur puissance de calcul ont permis de répondre avec succès aux attentes.

Toutefois, les algorithmes d'IA se sont rapidement avérés comme incontournables. C'est là qu'une architecture dédiée de processeur est apparue : le NPU.

En comparaison à un GPU, le NPU pourra gérer plus d'inférences (augmentations de la vitesse de calcul) tout en consommant beaucoup moins d'énergie.

## NPU dans un SoC : NXP eIQ Neutron

On retrouve les NPU sous forme de circuit intégré dédié, soit dans un SoC.

Classiquement le NPU fait partie du processeur principal (SoC) pour les applications orientées grand public, tandis que le NPU sera dans un processeur discret quand il s'agit d'applications industrielles.



Exemple :

NXP i.MX RT700

→ 2 x Cortex-M33

→ 2 DSP

→ 1 NPU eIQ Neutron

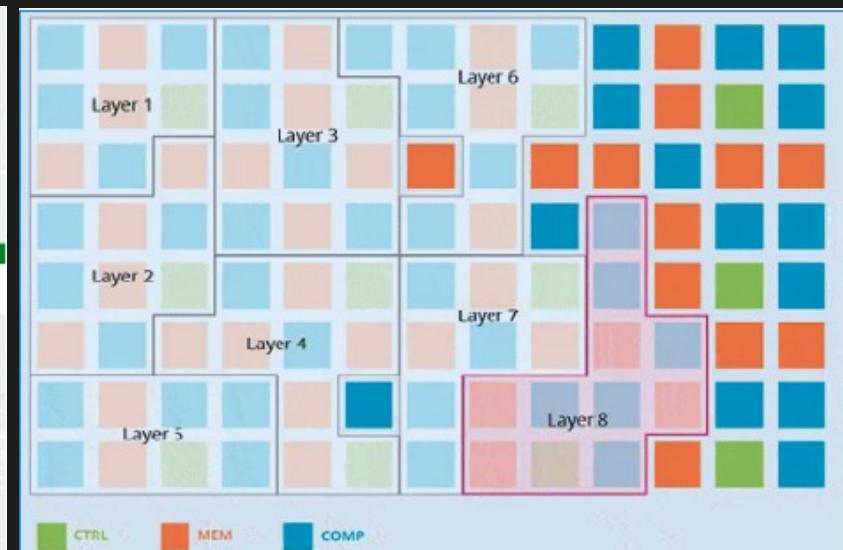
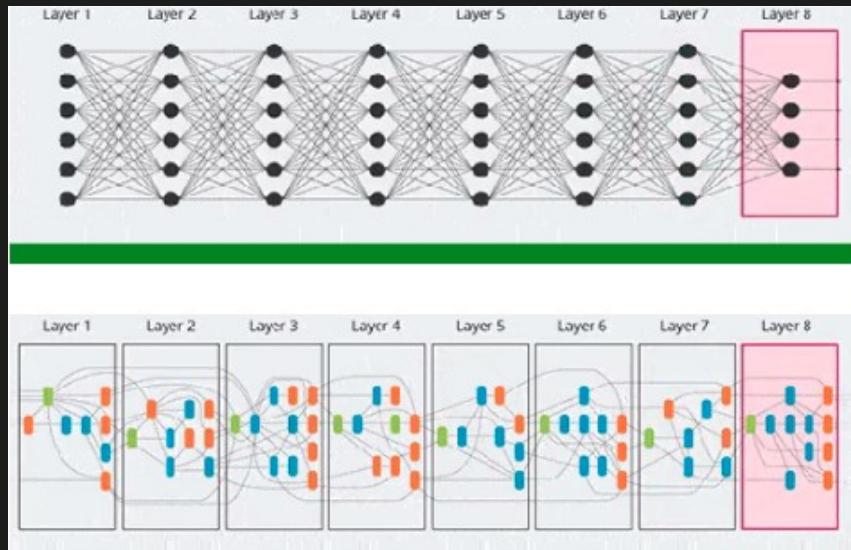
→ 1 GPU

## NPU discret : Hailo-8L

Dans d'autres cas, le NPU est un co-processeur discret.

Prenons ici l'exemple du Hailo-8L, NPU équipant le *Raspberry Pi AI Kit*.

Son architecture montre la correspondance entre couches du réseau de neurones et ressources de calcul du processeurs.



Leur architecture matérielle permet d'effectuer des opérations de réseaux de neurones ou liées à l'IA au sens large (reconnaissance vocale, traitement vidéo ou d'image, ...).

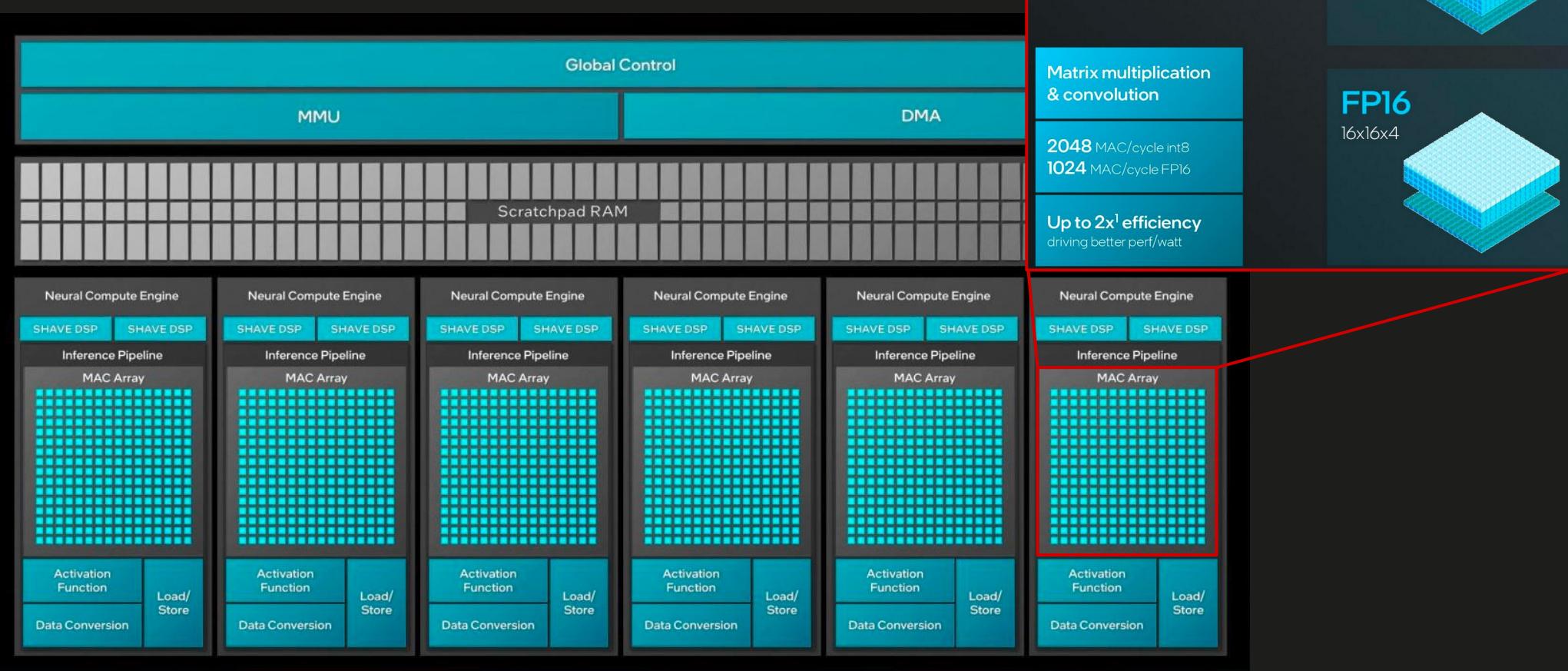
Operation Types Overview		
	Scalar	Vector
Complexity	1	N
Example functions	Conditional Looping	SoftMax Activation functions
Occurrence in AI	Low	Very high



**TOPs**

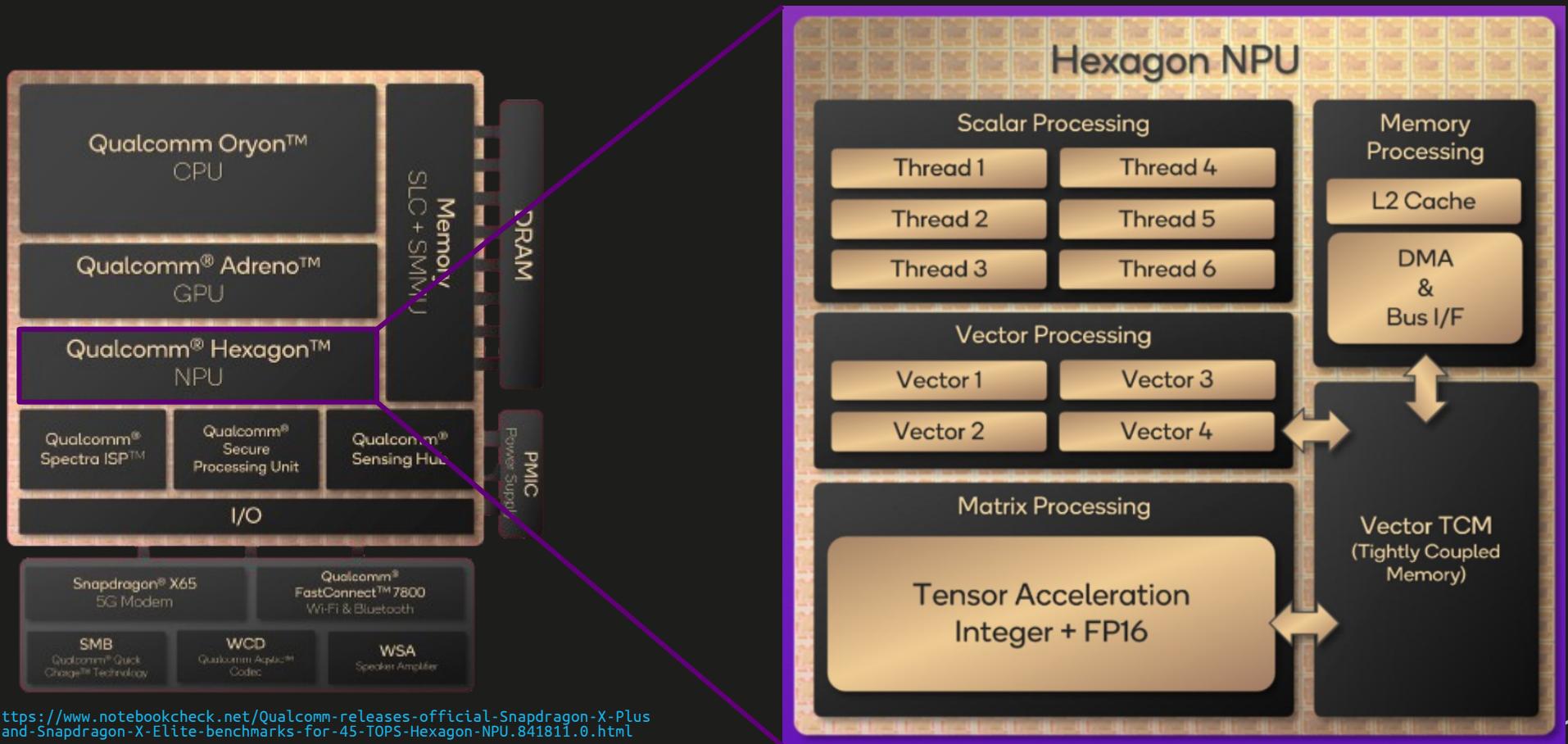
# NPU – NEURAL PROCESSING UNIT

## Intel's NPU: Intel NPU 4



# NPU – NEURAL PROCESSING UNIT

## Qualcomm's NPU: Hexagon

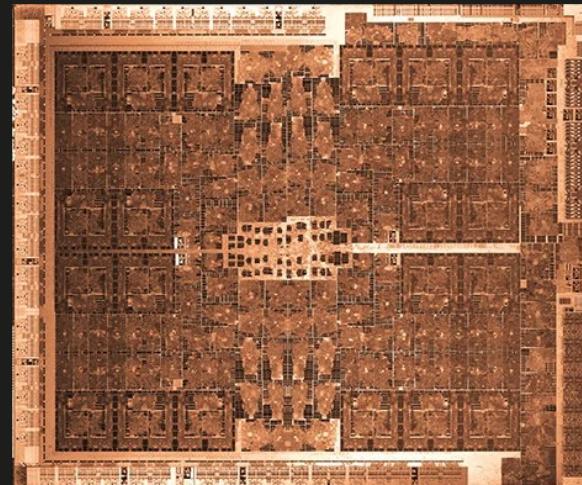
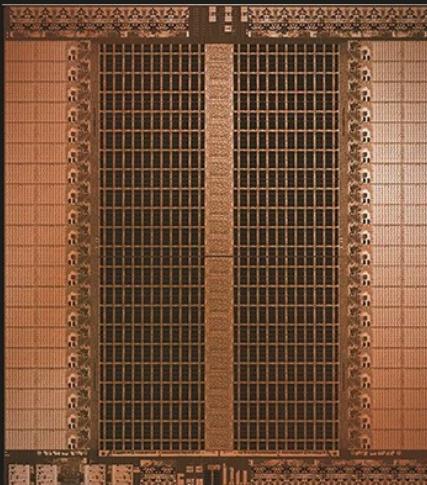


## Aparté sur le LPU

Dans la même tendance à concevoir des circuits spécialisés pour l'IA, les **LPU** (*Language Processing Unit*) sont apparus afin d'améliorer les performances d'utilisation de LLM.

Prenons le cas de Groq : en basant l'architecture de son processeur sur l'exécution d'inférence (opération de base des LLM), son LPU bat à plates coutures (vitesse, énergie, coût) la plus haute gamme de GPU NVidia.

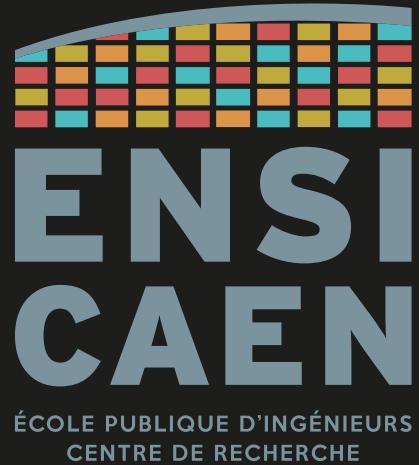
En revanche, le LPU n'est efficace que pour de l'IA générative de texte, rien d'autre.



À gauche :  
GroqChip 1 LPU

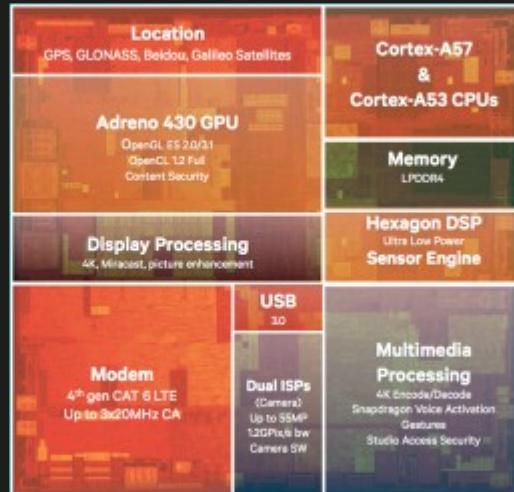
À droite :  
GPU Nvidia GTX 1070

# - SoC - SYSTEM ON CHIP

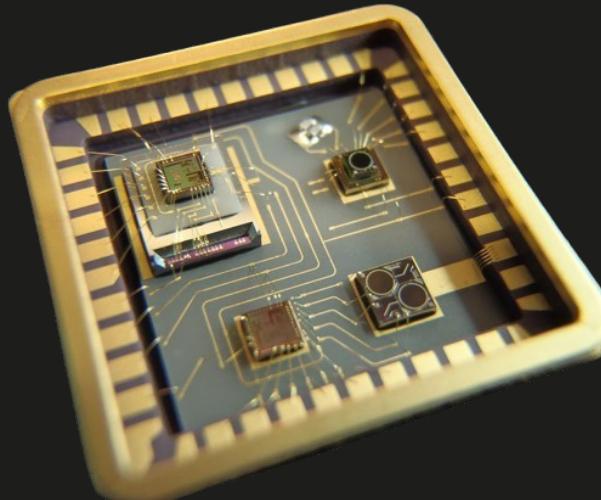


Un **SoC (System on Chip)** est un ensemble de composants hétérogènes assemblés sur dans un même et seul circuit intégré et théoriquement sur un seul et même *die (chip)*.

Dans la même idée, on trouve les **SiP (System in Package)** qui contiennent des composants hétérogènes intégrés dans un même et seul circuit intégré (*package*) mais réparti sur plusieurs *dies*.



The Qualcomm Snapdragon 810 is a SoC



Example of SiP

Par rapport à un *SoB (System on Board)* où toutes les fonctionnalités sont éclatées sur plusieurs circuits intégrés dispersés sur un circuit imprimé, le SoC et le SiP offrent :

- une meilleure efficacité énergétique
- une plus haute fréquence de fonctionnement (distance réduite = latence réduite)
- des coûts de fabrication réduits

En comparant un SoC et un SiP, les avantages précédemment mentionnés sont plus importants pour le SoC. Toutefois le SiP offre de plus grandes évolutivité et flexibilité.

Example : Intel Lunar Lake

## Intel Lunar Lake (Q3 2024)

SoC for mobile and premium laptops markets

### Why is this a SoC ?

- 8-Core hybrid design
- GPU
- NPU
- Memory on Package
- Connectivity and security units



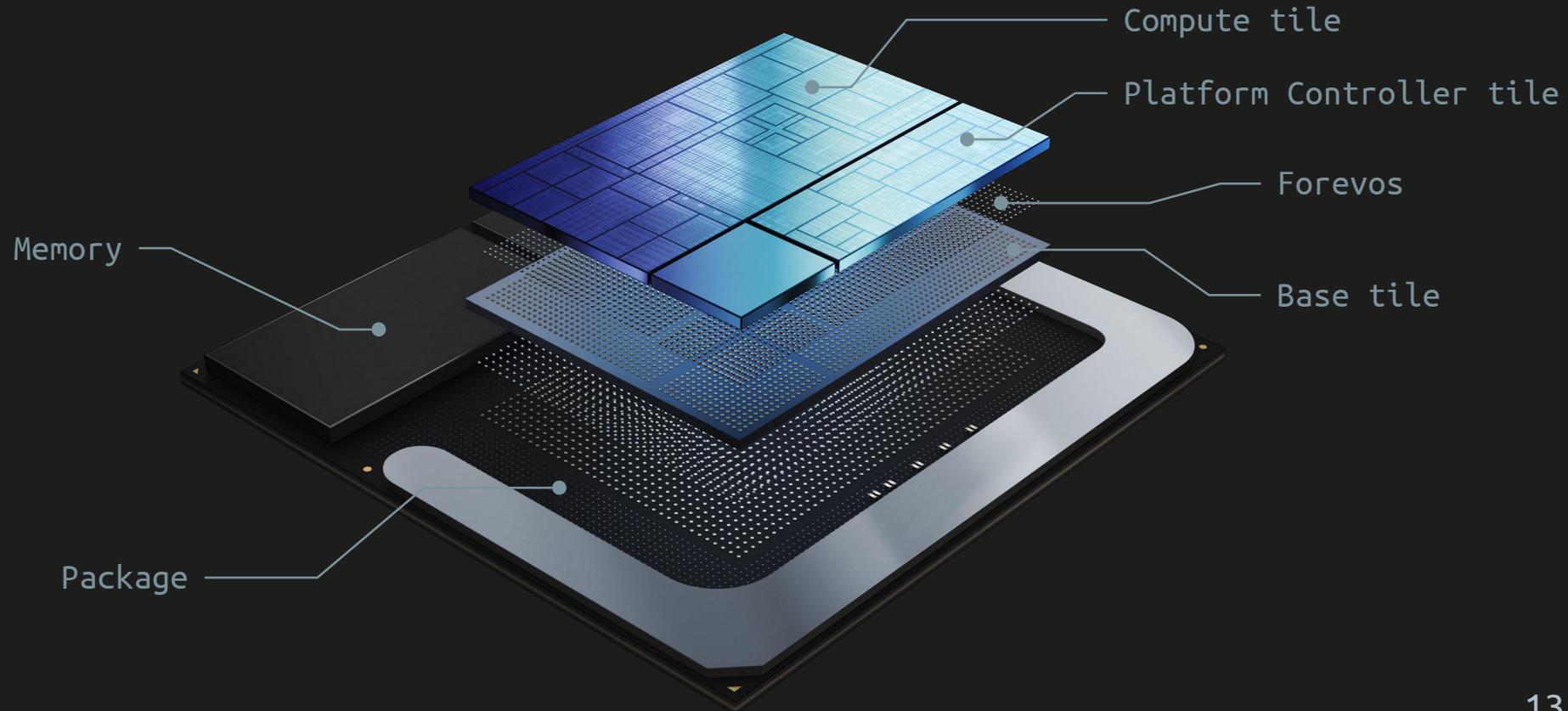
<https://www.youtube.com/watch?v=hmlDCAiD1bA&list=PL8t1FdN2Tj3a-H-K624FGvmaDGQVcqoXn>

<https://www.tomshardware.com/pc-components/cpus/intel-unwraps-lunar-lake-architecture-up-to-68-ipc-gain-for-e-cores-16-ipc-gain-for-p-cores>

<https://www.intel.com/content/www/us/en/content-details/824511/2024-intel-technology-tour-lunar-lake-power-management-intel-thread-director-innovations.html?wapkw=lunar%20lake>

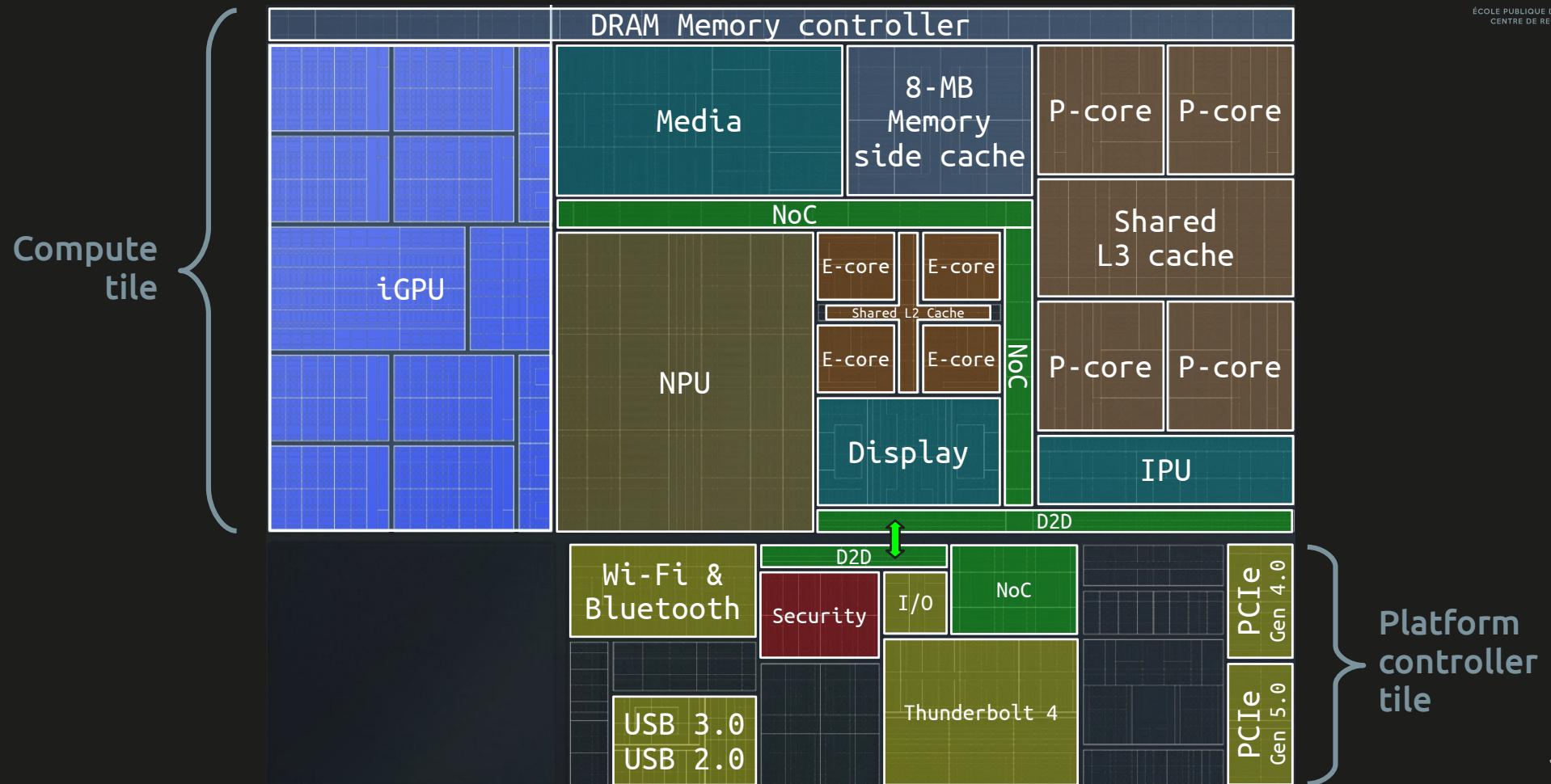
# SoC – SYSTEM ON CHIP

Example : Intel Lunar Lake



# SoC – SYSTEM ON CHIP

Example : Intel Lunar Lake



## Example : Intel Lunar Lake

### 4 E-cores (Efficiency)

→ 4-MB shared L2 cache

### 4 P-cores (Performance)

→ 12-MB shared L3 cache

→ 2.5-MB L2 cache per core

### GPU

→ 8 Xe2 cores

→ 8-MB cache

### NPU

→ 6 Neural compute engines

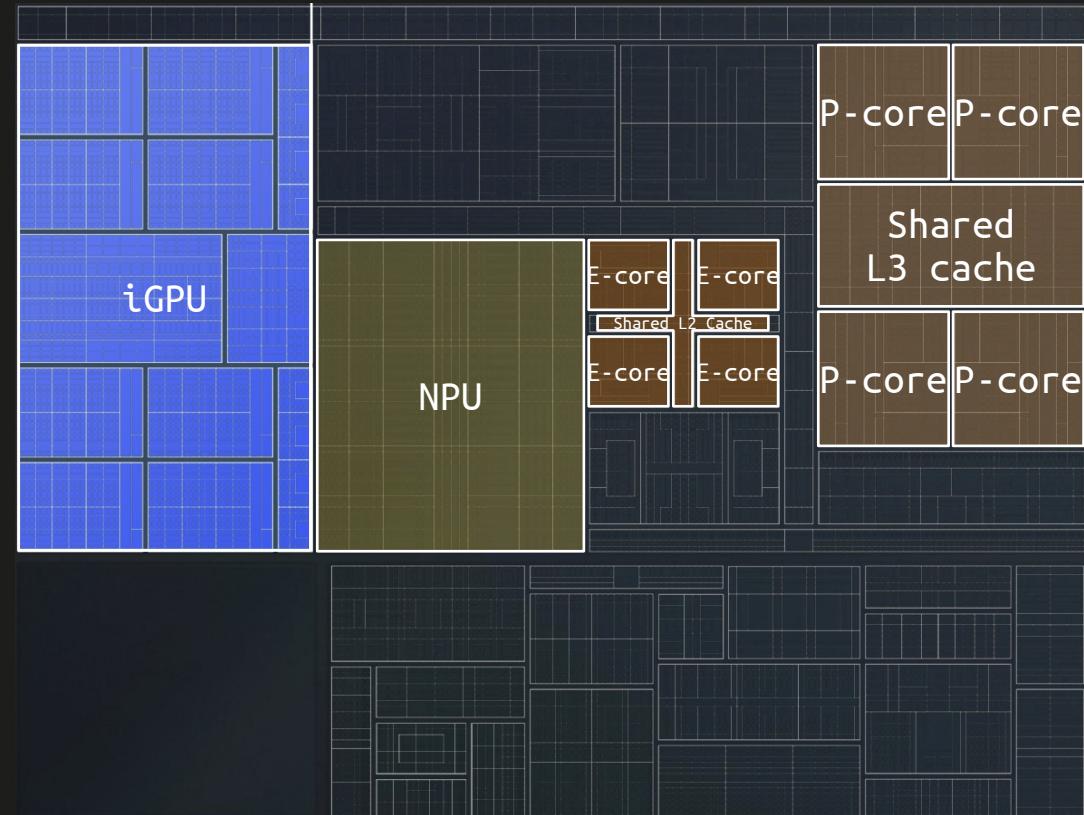
→ 12 Enhanced SHAVE DSPs

→ 9-MB cache

Up to  
5 TOPS

Up to  
67 TOPS

Up to  
48 TOPS



Example : Intel Lunar Lake

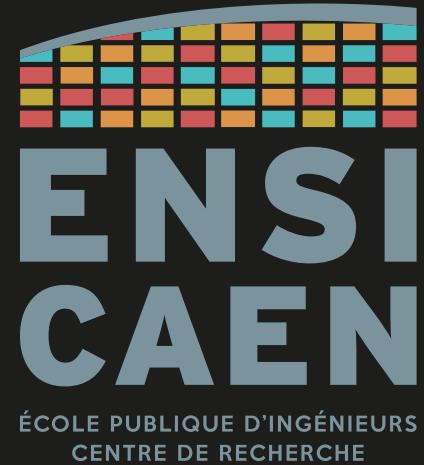
## Memory on Package

- Up to 32 GB
- LPDDR5X DRAM
- Up to 8.5 GT/s per chip
- 16 b x 4 channels

Plus a 8-MB memory side cache  
→ on the Compute tile



# - FPGA - FIELD PROGRAMMABLE GATE ARRAY



## Description

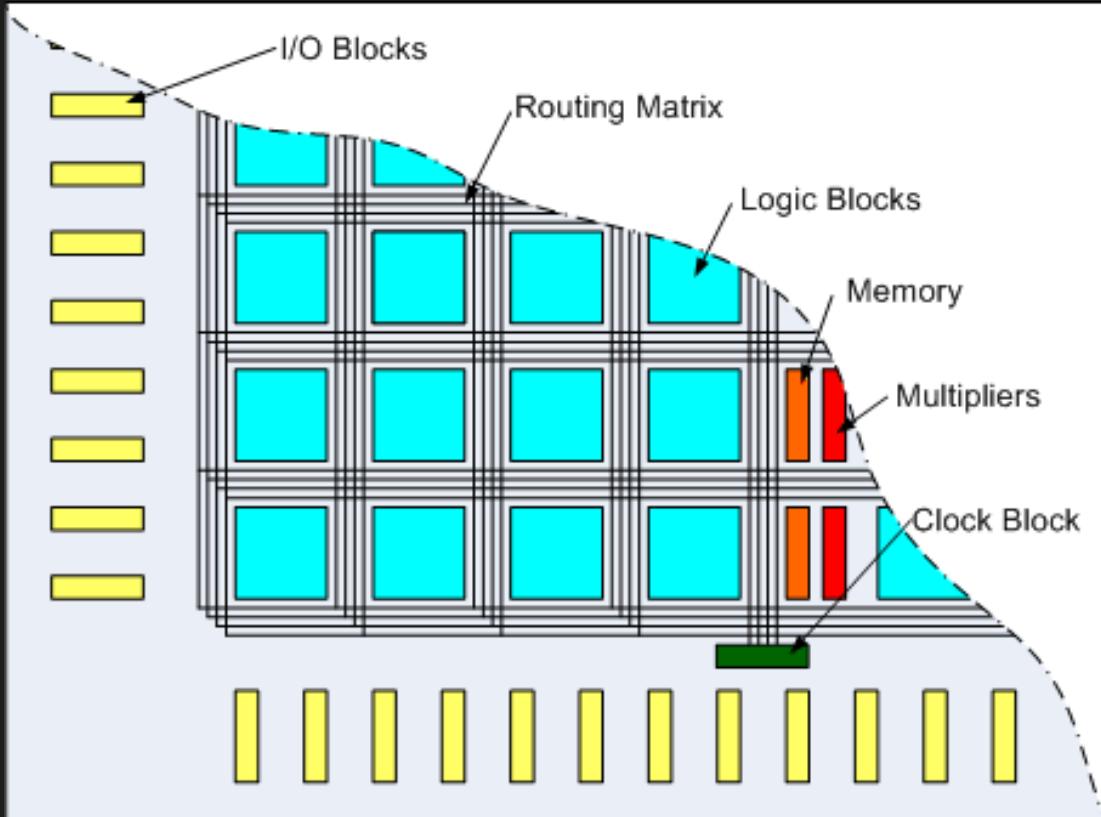
Le **FPGA (Field Programmable Gate Array)** est un composant purement logique :

Les Logic Blocks sont un ensemble de fonctionnalités logiques configurables (LUT, ALU, ...).

Les matrices d'interconnexion permettent d'associer les *Logic Blocks* à la guise du développeur

D'autres ressources spécifiques sont disponibles (RAM, multiplier, DSP, ...).

Le tout permet de créer une solution entièrement hardware, configurable à souhait.



### Avantages

Solution hardware = + rapide qu'un CPU

Consommation réduite

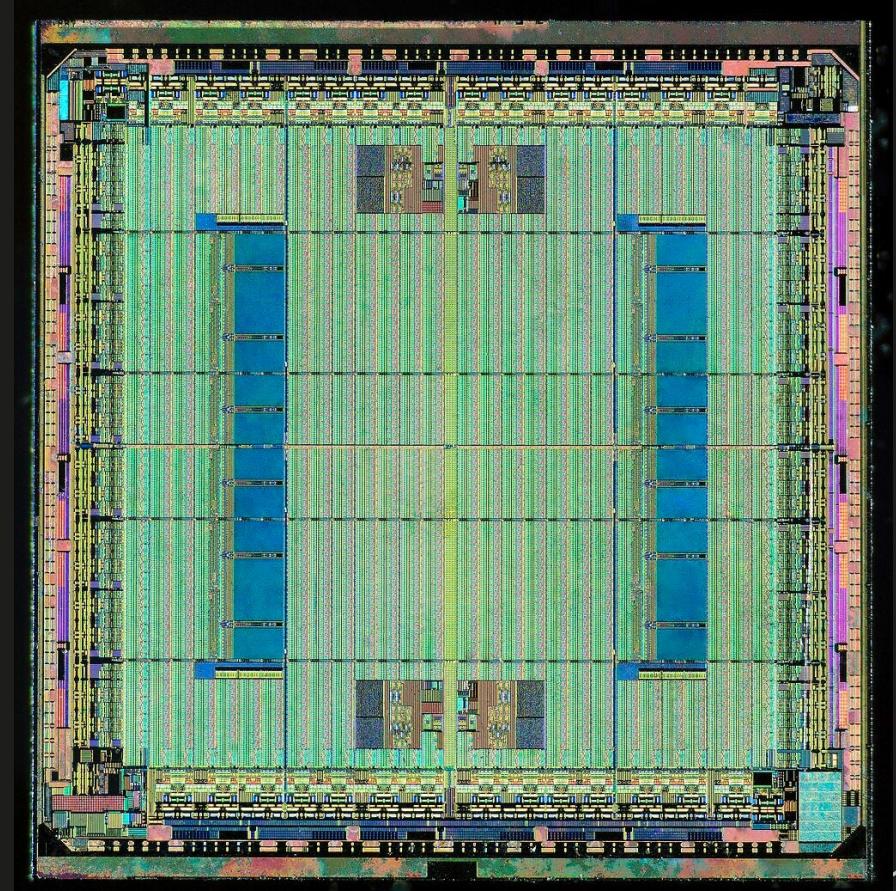
Adapté au traitement massivement parallèle

### Inconvénients

Coût du composant

Peu adapté au contrôle

(il faudrait implémenter un processeur à CPU)



## Langage

Un FPGA se synthétise en utilisant un langage de description (et non un langage de programmation).

Les deux plus connus sont le VHDL et le Verilog.

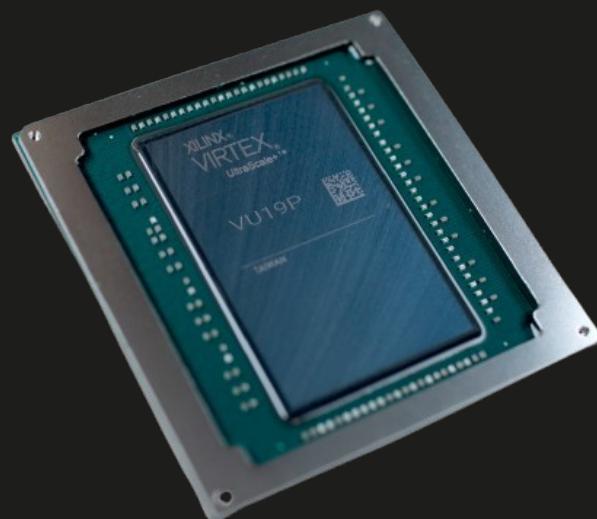
Il n'y a pas de programmation, juste une description du matériel à « fabriquer » :

- Mon bloc à telles entrées, telles sorties, la relation entrées-sorties est celle-ci
- Mon bloc A est relié au bloc B et C, ...
- Et ainsi de suite jusqu'à construire l'architecture matérielle souhaitée.

Les deux acteurs qui dominent largement le marché du FPGA sont :

- Xilinx (1984, USA), le créateur du FPGA (1985). Il a été racheté par AMD en 2022.

Device Name	VU19P
System Logic Cells (K)	8,938
CLB Flip-Flops (K)	8,172
CLB LUTs (K)	4,086
Max. Dist. RAM (Mb)	58.4
Total Block RAM (Mb)	75.9
UltraRAM (Mb)	90.0
DSP Slices	3,840
Peak INT8 DSP (TOP/s)	10.4
PCIe® Gen3 x16	0
PCIe Gen3 x16/Gen4 x8 <sup>(1)</sup>	8
150G Interlaken	0
100G Ethernet w/ KR4 RS-FEC	0
Max. Single-Ended HP I/Os	1,976
Max. Single-Ended HD I/Os	96
GTY 32.75 Gb/s Transceivers	80
GTM 58 Gb/s PAM4 Transceivers	–
100G / 50G KP4 FEC	–
Extended <sup>(2)</sup>	-1 -2
Industrial	–



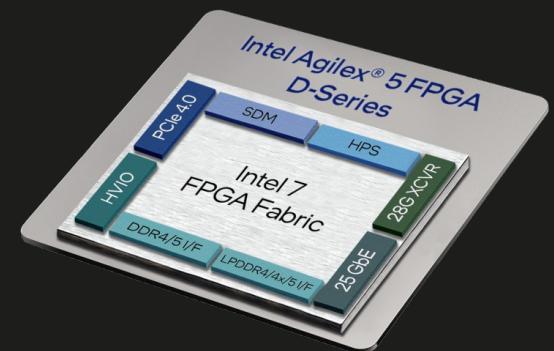
<https://www.amd.com/en/products/adaptive-socs-and-fpgas/fpga/virtex-ultrascale-plus-vu19p.html>

Acteurs : Intel Altera

Les deux acteurs qui dominent largement le marché du FPGA sont :

- Altera (1983, USA), racheté par Intel en 2015.

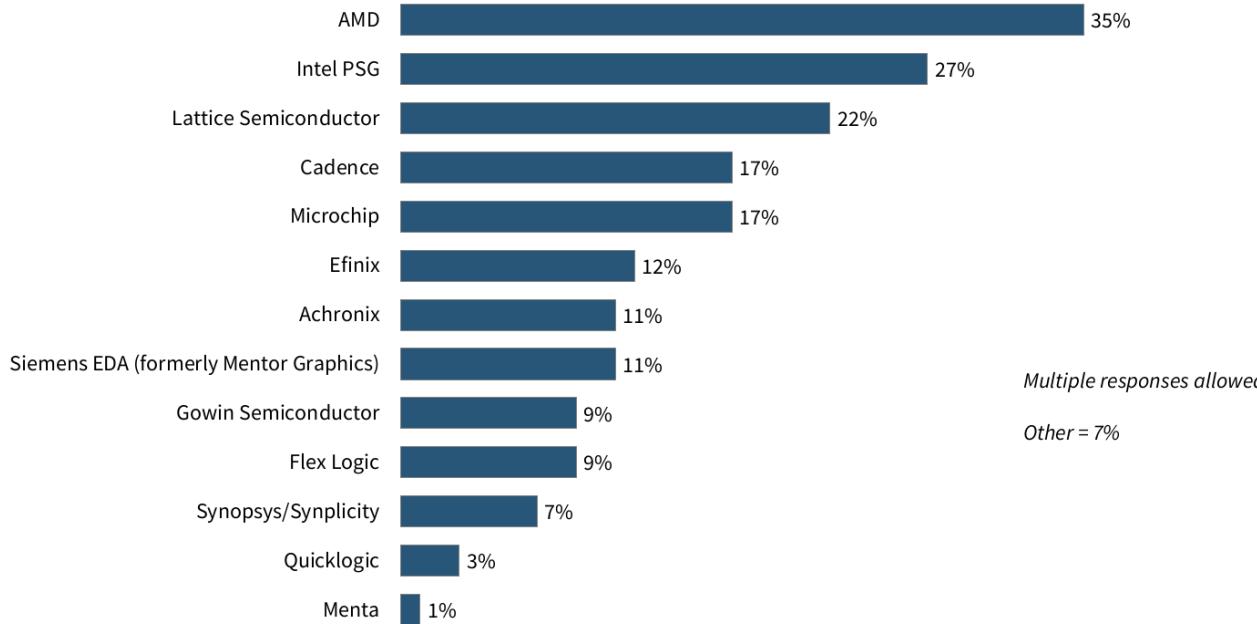
	Agilex™ 9 FPGAs	Agilex™ 7 FPGAs	Agilex™ 5 FPGAs	Agilex™ 3 FPGAs
Logic Capacity Range (logic elements)	Direct RF-Series	F-Series / I-Series / M-Series	E-Series / D-Series	C-Series
Memory (Max)	1.4M – 2.7M	573k – 4M	50k – 656k	25k – 135k
DSP Type	287 Mb	485 Mb (32 GB HBM2e option)	69 Mb	8.3 Mb
18x19 Multipliers (Max)	Variable-Precision DSP Blocks	Variable-Precision DSP Blocks	Enhanced DSP with AI Tensor Blocks	Enhanced DSP with AI Tensor Blocks
18x19 Multipliers (Max)	17,056	25,584	3,680	368
Hard Processor Options	Quad-Core Arm Cortex-A53	Quad-Core Arm Cortex-A53	Dual-Core Arm Cortex-A76 Dual-Core Arm Cortex-A55	Dual-Core Arm Cortex-A55
High-Speed Interfaces (max data rate)	58 Gbps XCVRs 64 Gsp/s ADC/DAC	116 Gbps XCVRs	28 Gbps XCVRs	12.5 Gbps XCVRs
Processor Interfaces	PCIe 4.0	PCIe 4.0/5.0, CXL	PCIe 4.0	PCIe 3.0
Memory Interfaces	DDR4, QDR IV	DDR4/5, LPDDR5, QDR IV	DDR4/5, LPDDR4/5, QDR IV	LPDDR4
I/O Count (Max)	660	768	444	344
XCVR count (Max)	32	120	32	4
Package Size (Min)	45x32mm	37.5x34mm	15x15mm	12x12mm
	Unprecedented Capabilities and Optimization for Target Applications	Higher Performance More Features and Capabilities Increasing Logic Capacity Greater IO Bandwidth	Lower Power More Cost Optimizations Less Logic Capacity Smaller Form Factors	



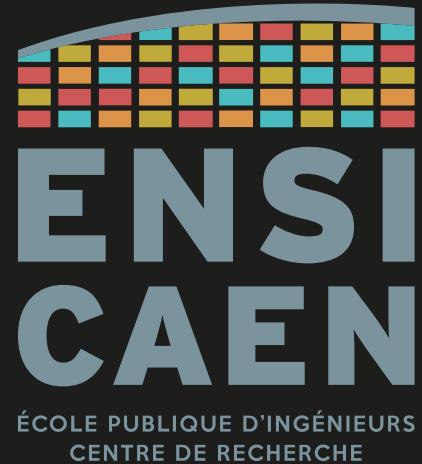
## Future consideration of FPGAs

AMD and Intel PSG are the most widely used vendors in the programmable logic space

**37% have incorporated FPGA chips in current project**



- ASIC -  
APPLICATION-SPECIFIC INTEGRATED CIRCUIT



## Définition

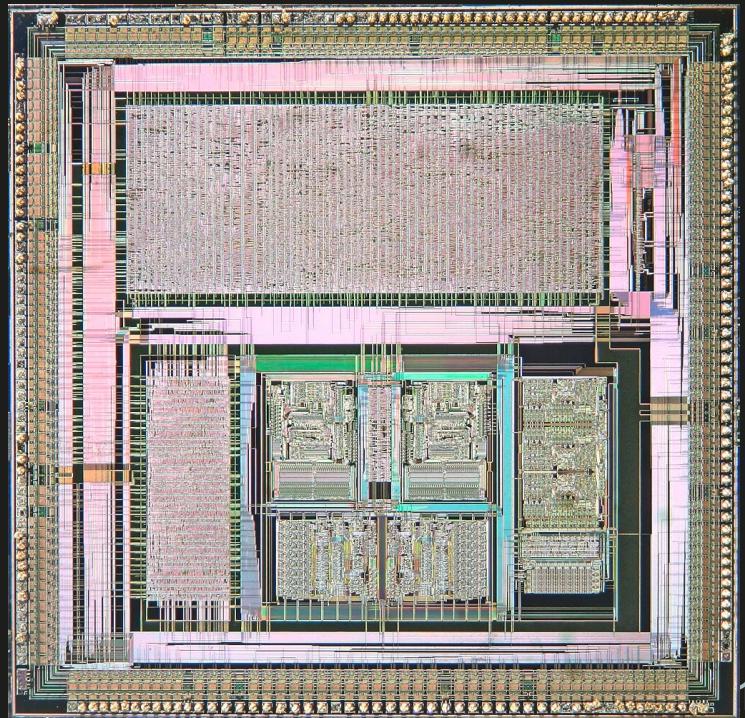
L'ASIC (*Application-Specific Integrated Circuit*) est un circuit intégré conçu sur mesure, pour une application précise.

La conception d'un ASIC est lourde (moyens humains) et coûteuses (humains et matériels).

On s'oriente donc vers un ASIC lorsque les fonctions souhaitées sont spécifiques et visent un grand volume de production :

Si les fonctions souhaitées sont relativement génériques, alors on utilisera des composants du marché.

Si le volume de production s'annonce réduit, alors on optera pour un FPGA.

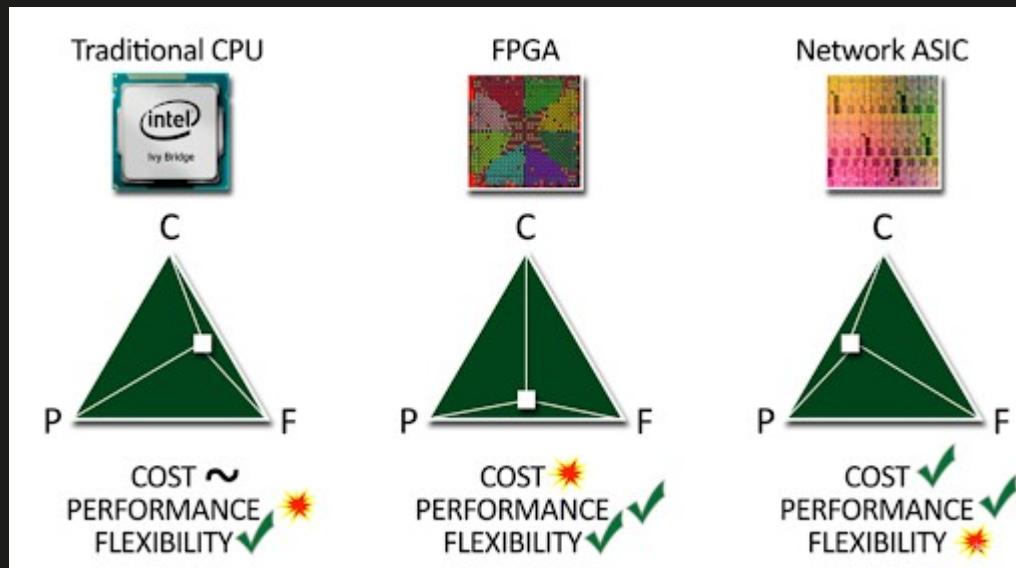


## ASIC ou composant générique ?

Un ASIC coûte plus cher qu'un FPGA : il faut du temps de développement, des outils et des moyens de production lourds.

En revanche une fois le premier composant fondu, le facteur d'échelle (investissement du développement d'une solution ramenée au nombre de composants) fait qu'un grand volume d'ASIC sera plus intéressant financièrement qu'un grand volume de FPGA.

Toutefois il reste un avantage indéniable pour le FPGA : celui-ci peut évoluer, même sur un produit déjà livré.



Et la crypto dans tout ça ?

Les ASIC ont vu un gros marché émerger ces dernières années : les cryptomonnaies !

Dans les entrailles de 2 calculateurs de minage de cryptomonnaie – Deus Ex Silicium

<https://www.youtube.com/watch?v=ccwTLWLeuwU>

## Pourquoi des ASIC ?

- Que de l'algorithmie, pas de contrôle → GPP
- Algorithmes ultra-spécifiques → GPU
- Pas besoin d'évolutivité → FPGA
- Gros volumes de production → ASIC
- Faible consommation → ASIC

## Algorithmes :

- Bitcoin → SHA256 (Proof of Work)
- Ethereum → Eth (Proof of Stake)
- Zcash → Equihash (Proof of Work)

Et la crypto dans tout ça ?

## Parenthèse (1/3) : les algorithmes Proof-of-Work sont extrêmement énergivores

En 2018, des chercheurs ont estimé la consommation d'énergie électrique liée au minage du Bitcoin à 48.2 TWh [1].

En 2021, le minage du Bitcoin seul consomme autant d'énergie électrique que la Thaïlande (>190 TWh) [2], soit x4 en deux ans ! En comparaison, cela représente ~40 % de la production électrique française cette année là.

Une transaction Bitcoin nécessite 700 kWh à elle seule, ce qui correspond la consommation énergétique moyenne d'un foyer américain sur une durée de deux mois [2].

Par conséquent, certains pays (dont la Chine en 2021) ont interdit le minage afin de garantir la stabilité du réseau électrique national. Parallèlement cela a provoqué une hausse de 7-8 % de la consommation électrique du Kazakhstan [3].

[1] Stoll, Klaaßen and Gallersdörfer. Center for Energy and Environmental Policy Research, MIT, 2018.

[2] « Preying on the poor? Opportunities and challenges for tackling the social and environmental threats of cryptocurrencies for vulnerable and low-income communities », Peter Howson, Alex de Vries. Energy Research & Social Science, 2022.

[3] <https://www.rts.ch/info/economie/13927740-le-minage-de-bitcoin-tilt-encore-un-avenir-au-kazakhstan.html>

Et la crypto dans tout ça ?

## Parenthèse (2/3) : algorithmes énergivores → rejet de CO<sub>2</sub>

Les références citées page précédente estiment l'empreinte carbone du Bitcoin oscillant entre 21.5 et 53.6 MT CO<sub>2</sub> en 2018 [1] (selon le mix énergétique des pays dans lesquels le Bitcoin est miné) et 90 Mt CO<sub>2</sub> en 2021 [2].

En comparaison, cela est bien supérieur à l'empreinte carbone de la filière d'extraction minière de l'or, tout en rapportant moins [2].

La création d'1 \$ de Bitcoin a engendré en moyenne un coût de 0.35 \$ de dommage climatique sur la période 2016-2021, avec un pic à 1.56 \$ en mai 2020. En comparaison l'extraction de 1 \$ d'or provoque 0.04 \$ de dommage climatique, la production de 1 \$ de véhicule SUV en cause 0.14 \$, et la production de 1 \$ de gazole provoque 0.41 \$ de dommage climatique [4].

Enfin, on peut évoquer les besoins en eau pour le refroidissement des fermes de minage, ou encore la durée de vie relativement courte des machines et donc la masse de déchets électroniques.

[4] « Economic estimation of Bitcoin mining's climate damages demonstrates closer resemblance to digital crude than digital gold », Benjamin A Jones, Andrew L Goodkind, Robert P Berrens. Scientific Reports, 2022. <https://www.nature.com/articles/s41598-022-18686-8>

Et la crypto dans tout ça ?

Parenthèse (3/3) : et on parlait de Bitcoin uniquement !

Certes Bitcoin est la cryptomonnaie la plus utilisée (et une des plus intrinsèquement énergivores), mais il ne faut pas oublier l'ensemble des autres cryptomonnaies !

En 2022, Ethereum « estime que Bitcoin et Ethereum brûlent plus d'un million de dollars de coûts d'électricité et de matériel par jour dans le cadre de leur mécanisme de consensus ».

Quelques liens pour explorer : Digiconomist

<https://digiconomist.net/bitcoin-energy-consumption>

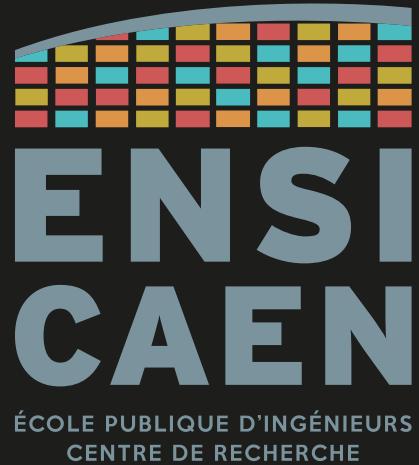
<https://digiconomist.net/ethereum-energy-consumption>

# TECHNIQUES D'ACCÉLÉRATION DES CALCULS

Accélération matérielle

Calcul vectoriel (SIMD)

VLIW (MISD)



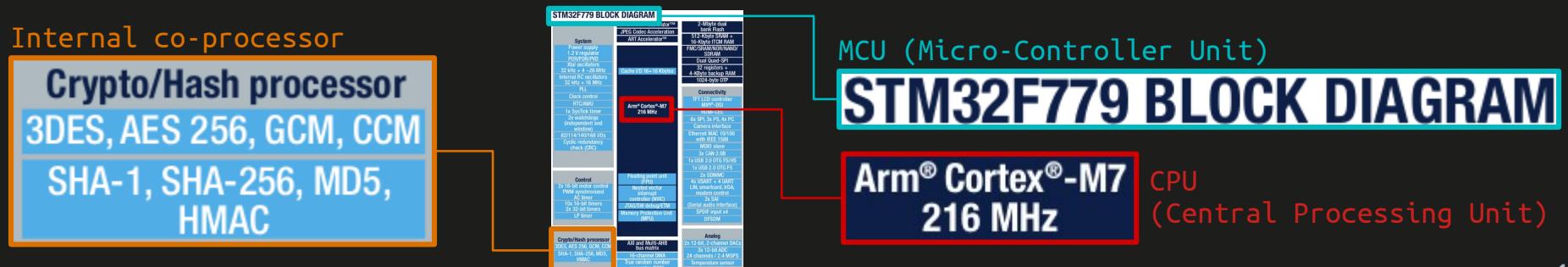
# Accélération matérielle

On appelle **accélération matérielle** (*hardware acceleration*) le fait de déléguer une partie des calculs logiciels (suite d'instructions) à un circuit câblé (matériel).

L'accélération matérielle peut s'effectuer au sein même d'un processeur, au travers des **périphériques** ou **unités de traitement dédiées**.

C'est le cas par exemple de la **FPU (Floating Point Unit)**, qui aujourd'hui équipe de série les GPP et GPU.

On peut aussi parler des périphériques **FFT (Fast Fourier Transform)** que contiennent certains MCU et DSP. Ou encore les périphériques de **chiffrement** (AES, SHA, ...) disponibles dans certains MCU hautes performances.



## Accélération matérielle

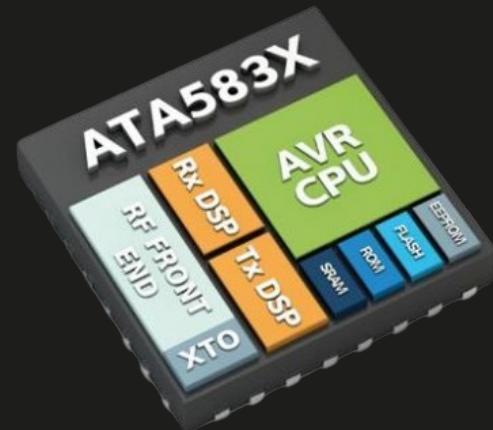
La fonction d'accélération matérielle peut aussi être confiée à un **co-processeur**.

C'est par exemple le cas des **GPU**, agissant en tant que co-processeur du GPP pour le traitement vidéo.

Dans le cas des cartes sons, c'est le **DSP** qui effectue les traitements (notamment les codecs).

Des **MCU** peuvent aussi jouer ce rôle, notamment dans les smartphones (modules radios (4G/5G/Wi-Fi)).

Les **FPGA** (*Field-Programmable Gate Array*) sont également très utilisés en temps que co-processeurs : leur fonctionnement étant basé sur de la logique câblée, ils sont extrêmement rapides (bien plus qu'un processeur).



## Accélération matérielle

Prenons pour exemple la **FPU** : premier traitement à être externalisé dans un objectif d'accélération.

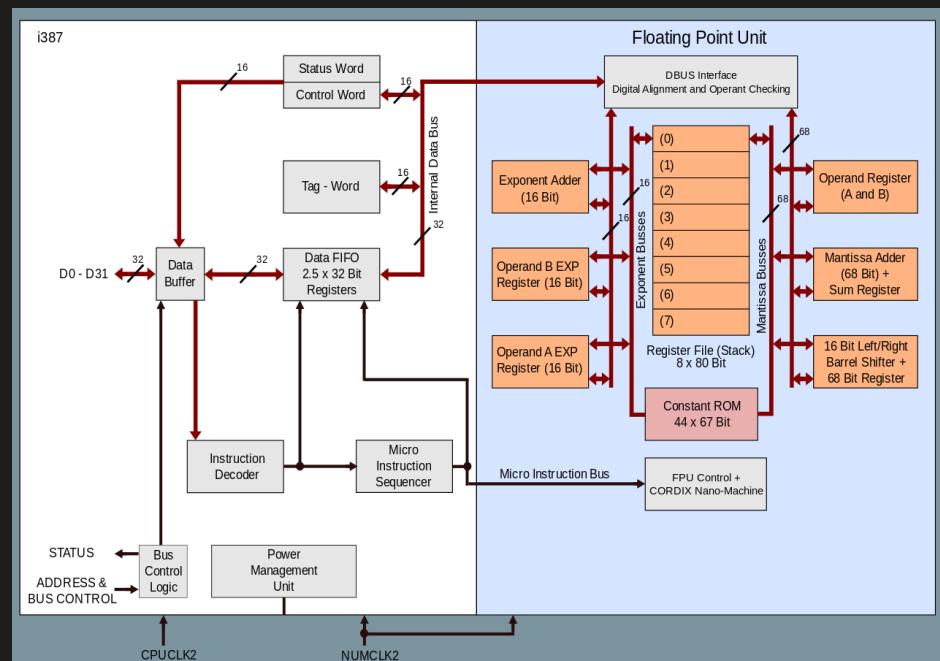
Historiquement, et jusqu'au milieu des années 1990, la FPU était réalisée par un co-processeur. Les plus connus sont les Intel x87 (8087, 80287, 80387, 80487) associés à leurs homologues x86 (du 8086 au 80486).

Mais depuis la FPU est directement intégrée au sein des GPP, des GPU, et même au sein de quelques MCU.

Parmi les principales fonctions réalisées, on trouve :

- *Fused multiply-add* (similaire à une MAC)
- Division (\*)
- Racine carrée
- Exponentielle (\*)
- Fonctions de trigonométrie (sin, cos, ...) (\*)

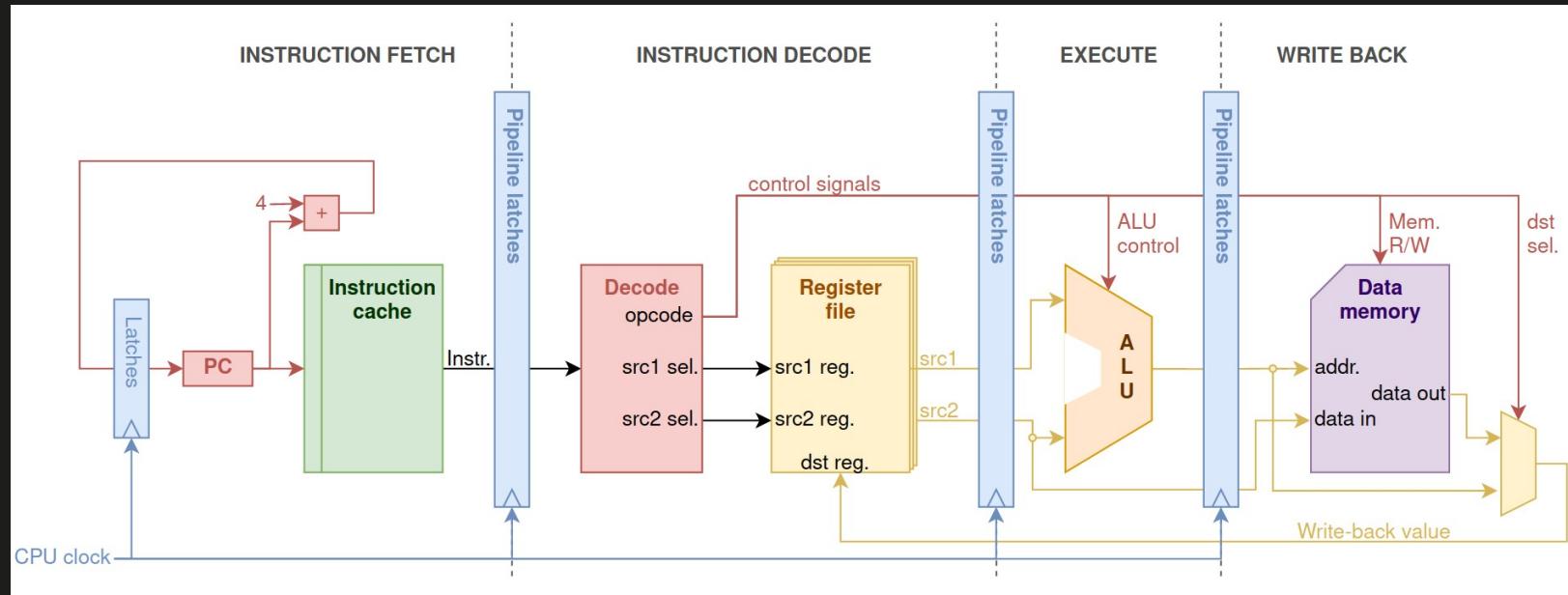
Certaines opérations sont pipelinées (*add, multiply, ...*).  
(\*) D'autres, plus complexes, sont parfois émulées et non câblées.



## Instruction pipelining

Le **Instruction pipelining** (ou *hardware pipeline*) est une technique d'implémentation du parallélisme d'instructions (*Instruction Level Parallelism* ou *ILP*).

L'idée est de décomposer la chaîne de traitement d'une instruction en plusieurs étapes successives, souvent regroupées en **Fetch/Decode/Execute/WriteBack** pour les processeurs RISC. L'objectif est d'utiliser la totalité du processeur en traitant simultanément plusieurs instructions, mais chacune à un étage différent.



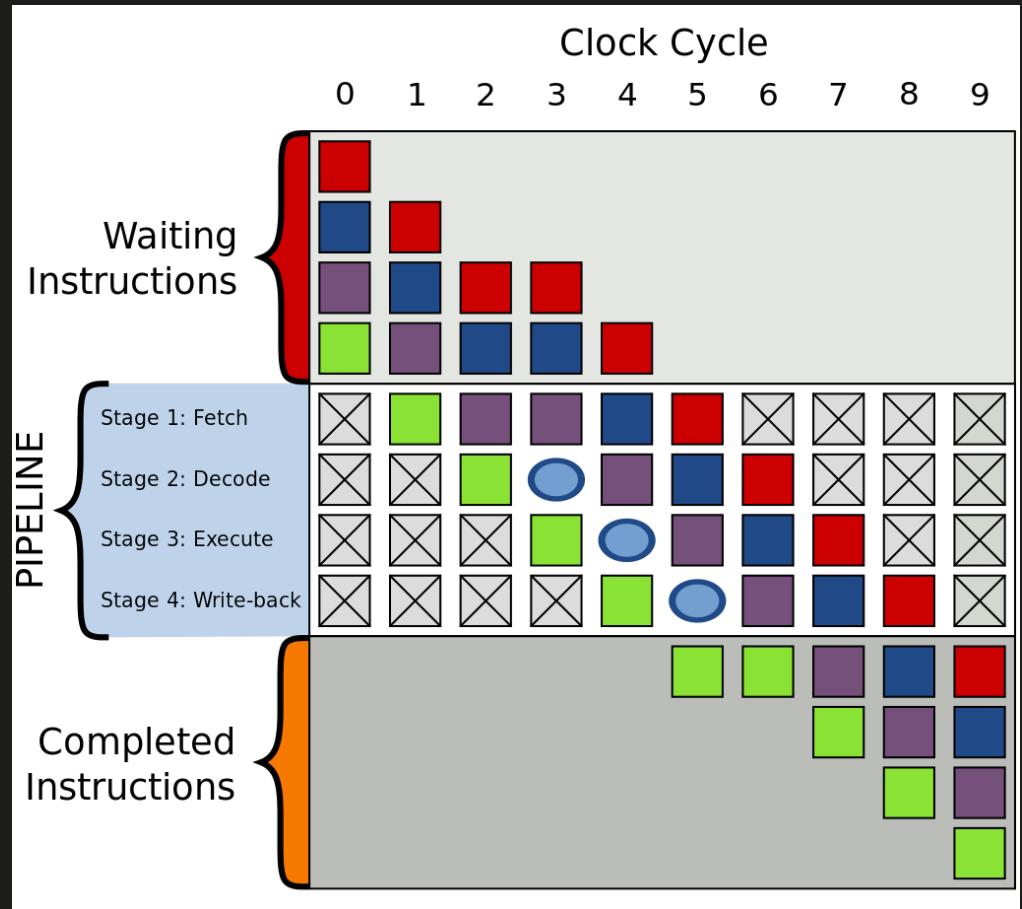
## Instruction pipelining

En apparence, il y a autant d'instructions en parallèle qu'il y a d'étages dans le pipeline.

Tous les étages étant pilotés par la même horloge, c'est l'étage le plus lent qui fixe la **cadence** de fonctionnement du pipeline.

Cependant, augmenter le nombre d'étages (autour de 31 pour les Pentium par exemple) peut provoquer des problèmes de dépendances de données, de chargement de mauvaises instructions en cas de mauvais branchement, d'attente de données si le cache ne contient pas la donnée attendue ...

Pour cela, différents mécanismes sont mis en place (bypass d'étage, insertion de bulle, prédiction de branchement, ...), ajoutant encore du matériel et de la complexité.

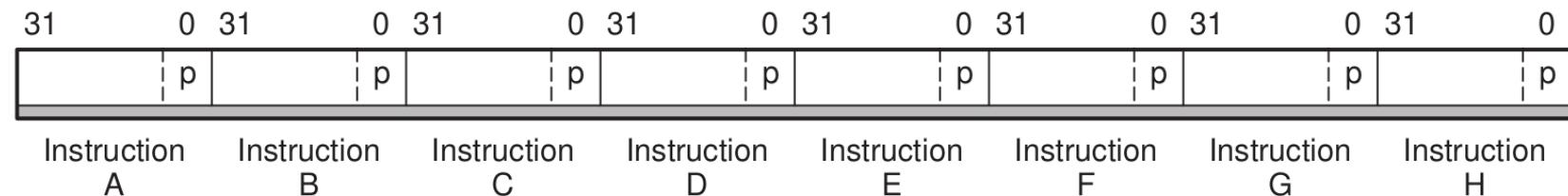


## Very Long Instruction Word

Les processeur à architecture **VLIW (Very Long Instruction Word)** implémentent aussi le parallélisme d'instructions, mais au sens strict du terme (contrairement au *pipelining*).

En l'occurrence, les bus d'instructions sont généralement de 128 ou 256 bits de large et permettent de transporter plusieurs instructions qui seront exécutées en parallèle.

**Figure 3-3 Basic Format of a Fetch Packet**

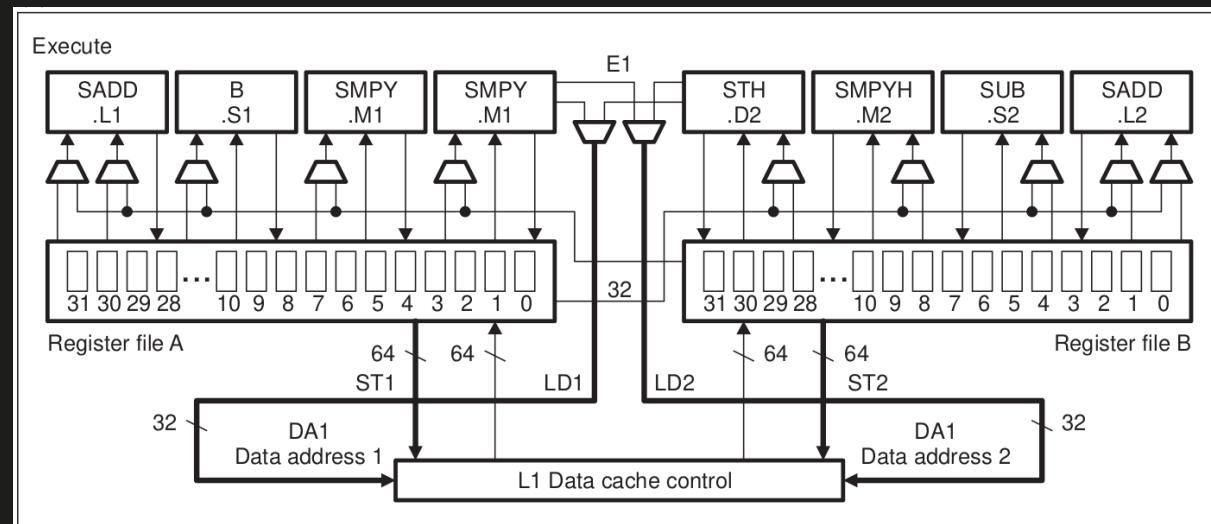


Exemple du DSP TI C6600, qui peut exécuter jusqu'à 8 instructions 32-bit simultanément. Le bit de poids faible (p) dans l'opcode d'une instruction indique à l'étage de Fetch s'il faut aller chercher l'instruction suivante pour la paralléliser, dans la limite de 8 instructions.

## Very Long Instruction Word

Pour exécuter simultanément plusieurs instructions, cela implique évidemment que l'étage d'exécution soit équipé de suffisamment d'unités d'exécutions.

De plus, la dépendance entre les données et entre les unités d'exécution doit être contrôlée. Cette tâche est généralement confiée à la chaîne de compilation.



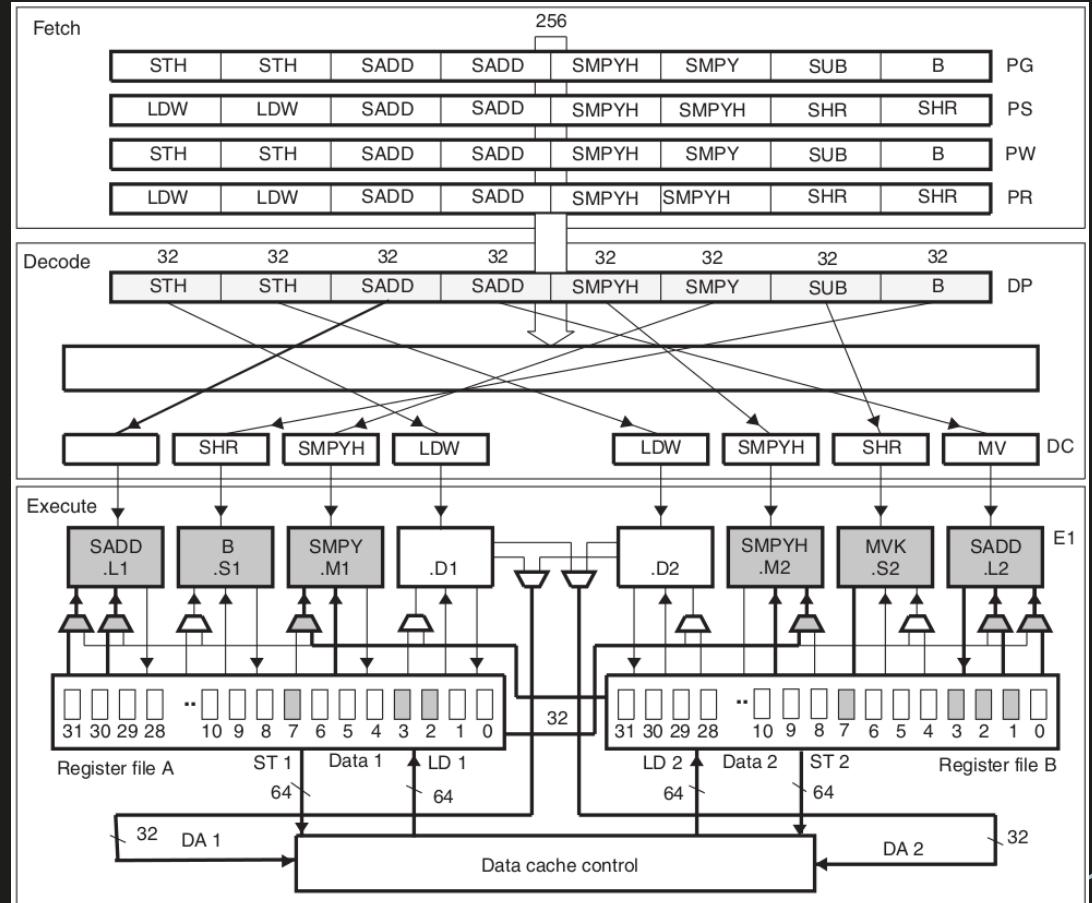
Exemple du DSP TI C6600 : étage d'exécution avec 8 unités d'exécution distinctes.

## Very Long Instruction Word

Il est évidemment possible de cumuler *hardware pipeling* et *VLIW*.

Le DSP TI C6600 dispose d'un pipeling à 16 étages associé à un VLIW de 8 instructions 32-bit.

Avec 8 cœurs cadencés jusqu'à 1.4 MHz, cette association mène à des performances de 44.8 GMAC ou 22.4 GFLOP par core.



## Vectorisation

Une autre manière de gagner du temps de calcul est de **paralléliser les données** (et non pas les instructions), à savoir opérer une instruction unique sur un lot de données. C'est le mécanisme de **vectorisation** (par opposition au calcul scalaire).

Calcul scalaire :       $a = b + c ;$

Calcul vectoriel :     $a[0:3] = b[0:3] + c[0:3] ; \quad // \text{pseudo-code}$

```
/** Équivalent à :  
 * a[0] = b[0] + c[0] ;  
 * a[1] = b[1] + c[1] ;  
 * a[2] = b[2] + c[2] ;  
 * a[3] = b[3] + c[3] ;  
 *  
 * Mais effectuées simultanément  
 */
```

La vectorisation correspond au paradigme **SIMD** de la classification de Flynn :

*SIMD = Single Instruction stream, Multiple Data stream :*

- le flot d'instructions est constitué d'instructions uniques (i.e. pas en parallèle)
- Le flot de données est constituée de données multiples (i.e. des vecteurs)

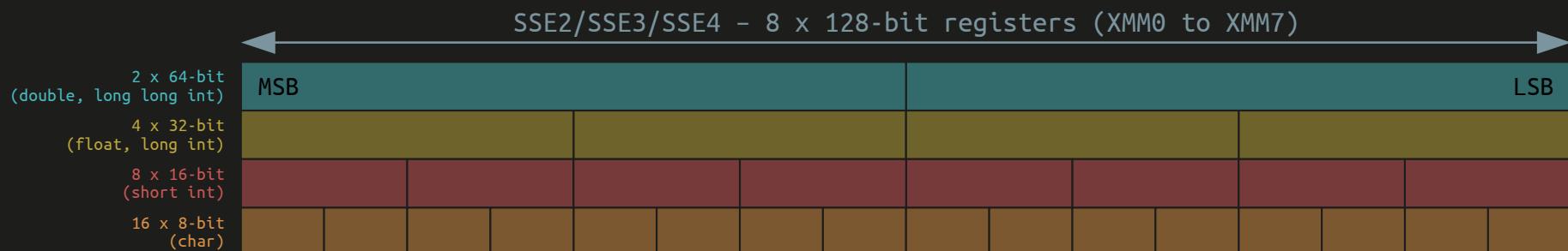
Pour avoir la possibilité de vectoriser un algorithme (ou une partie), l'architecture cible doit matériellement implémenter le mécanisme SIMD.

Très souvent, l'ISA (*Instruction Set Architecture*) du processeur contient des instructions permettant de manipuler directement ce matériel.

## Vectorisation : SSE, AVX

En 1999, Intel présente son **SSE** (*Streaming SIMD Extensions*), une extension de 70 instructions à son ISA. Ceci permet au Pentium III d'effectuer des instructions vectorielles à l'aide de 8 registres 128-bit (XMM0 à XMM7), contenant chacun 4 floats.

Un an après, **SSE2** fait son apparition sur Pentium 4 en ajoutant encore 144 instructions. Mais surtout, SSE2 offre désormais la possibilité de stocker tous types dans ses registres XMM. Il s'agit de la version SSE/AVX utilisée par défaut par GCC.



Puis virent SSE3, SSE4, suivi par **AVX** (*Advanced Vector Extensions*) en 2011. Les instructions AVX complètent SSE en offrant l'accès 16 registres 256-bit (YMM0 à YMM15).

La dernière version est actuellement **AVX-512**, parue en 2013 sur Skylake et offrant des registres 512 bits dédiés aux instructions vectorielles (ZMM0 à ZMM31).

## Vectorisation : SSE et fonctions intrinsèques

Ces registres dédiés au calcul vectoriels ne sont pas accessibles avec les instructions standards de l'ISA x86 ou x64, mais avec les instructions ajoutées par les extensions SSE ou AVX.

Au lieu de développer en assembleur, Intel fourni également un jeu de fonctions C permettant d'adresser directement ses instructions SIMD.

```
#include <stdio.h>
#include <x86intrin.h>

void main(){
    float a[4];
    __m128 vector_a = _mm_set_ps( 0.0, 1.0, 2.0, 3.0 );

    float b[4] = { 0.2, 0.4, 0.6, 0.8 };
    __m128 vector_b = _mm_load_ps( &b[0] );

    float c[4];
    __m128 vector_c = _mm_add_ps( vector_a, vector_b );

    _mm_store_ps( &c[0], vector_c );

    for( int i = 0; i < 4 ; i++ )
        printf("%f ", c[i]);
    printf("\n");
}
```

`__m128` : type conteneur 128-bit  
(mais on ne sait pas comment il est décomposé).

`_mm_set_ps()` : affecte les valeurs au conteneur, et précise qu'il s'agit de float (`ps` = *precision simple*).

`_mm_load_ps()` : charge à partir de l'adresse fournie, 4 données en `ps` (*single precision*) dans le conteneur.

`_mm_add_ps()` : additionne deux conteneurs, décomposés en 4 `float`, et retourne le résultat dans un conteneur.

`_mm_store_ps()` : stocke en mémoire (à l'adresse indiquée) 4 `floats` issus du conteneur.

## Vectorisation : SSE et fonctions intrinsèques

Très souvent, la vectorisation est utilisée dans les boucles, qui sont déroulées d'un facteur égal au nombre de données stockées dans les registres SIMD.

Attention à gérer les bords (nombre de données non multiples de la taille des vecteurs).

```
// 1. Canonical code
for( int i = 0 ; i < 4096 ; i++ ) {
    array[i] = i ;
}

// 2. Loop unrolling, radix 4
for( int i = 0 ; i < 4096 ; i+=4 ) {
    array[i] = i ;
    array[i+1] = i+1 ;
    array[i+2] = i+2 ;
    array[i+3] = i+3 ;
}

// 3. Vectorization (SSE2+)
for( int i = 0 ; i < 4096 ; i+=4 ) {
    // Set 4 int32_t into a 128-bit register
    __m128 vector = _mm_set_epi32( i, i+1, i+2, i+3 );
    // Store register into array
    _mm_store_ps( &array[i], vector );
}
```

Certes, ces fonctions permettent d'utiliser le SIMD et donc d'accélérer un algorithme, mais l'utilisation du SSE ou AVX a pour conséquence de rendre le code non-portable puisqu'il dépend des capacités du processeur !

## Vectorisation : activation par la toolchain

Il y a encore plus pratique que d'écrire du code optimisé : laisser la toolchain faire !

```
gcc [...] -O3 -march=native
```

**-O3** = Active les optimisations de niveau 3, et comprend entre autres :

- floop-interchange** → permutation d'imbrication de boucles
- floop-unroll-and-jam** → déroulement de boucles
- fpredictive-commoning** → prédiction de branchement

**-march=cpu-type** : « *Generate instructions for the machine type cpu-type. [...] -march=cpu-type allows GCC to generate code that may not run at all on processors other than the one indicated.* »

**'native'** : « *This selects the CPU to generate code for at compilation time by determining the processor type of the compiling machine. Using -march=native enables all instruction subsets supported by the local machine (hence the result might not run on different machines). [...]* »

### Processeur VLIW

- Intelligence (optimisation) portée par le développeur et la toolchain
- Code en mémoire programme dans le désordre
- Exécution du code dans l'ordre
- Déterministe
- Excellent ratio performance/Watt

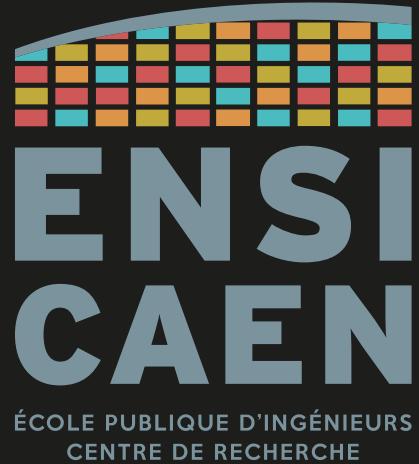
### Processeur superscalaire

- L'intelligence réside dans l'étage d'exécution du CPU
- Code en mémoire programme dans l'ordre
- Exécution dans le désordre (*OOO execution*)
- Non-déterministe
- Mauvais ratio performance/Watt

Les **CPU superscalaires** sont conçus pour exécuter du code générique faiblement optimisé, remplis de tests et de sauts. → **Objectif = généricité**

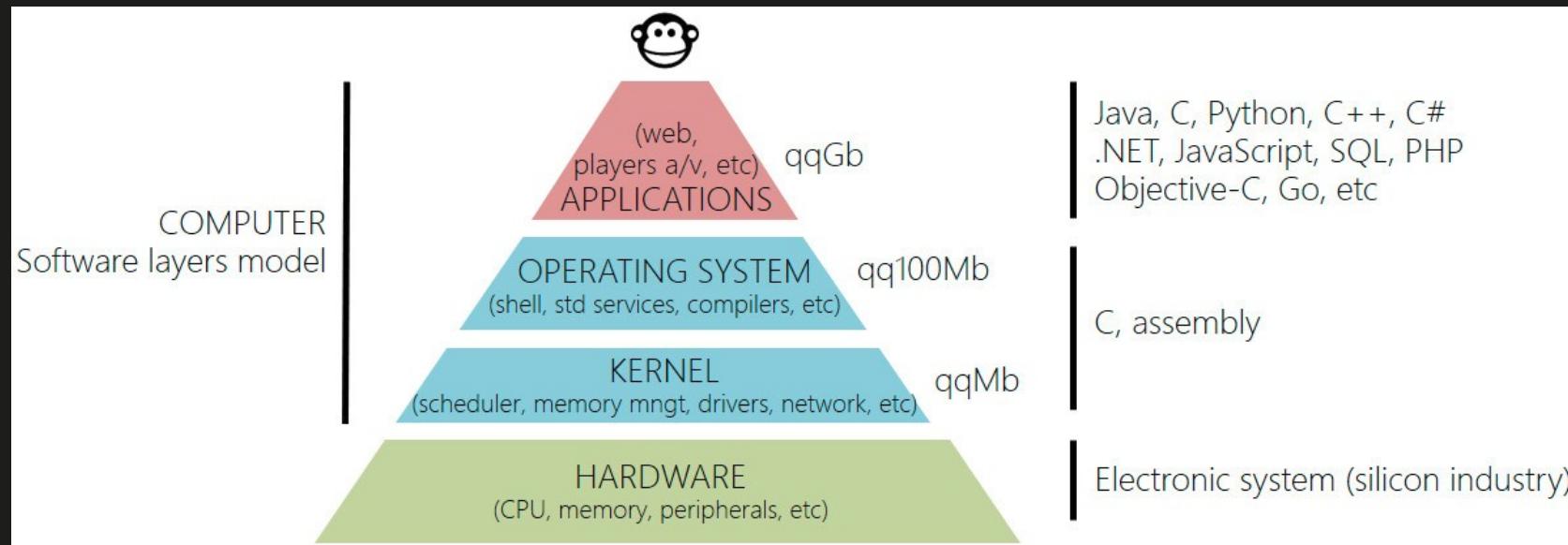
Les **CPU VLIW** doivent exécutés du code spécifique à une cible afin d'exploiter le maximum de ses capacités.  
→ **Objectif = code spécifique optimisé (non portable)**

# CHOISIR UN PROCESSSEUR



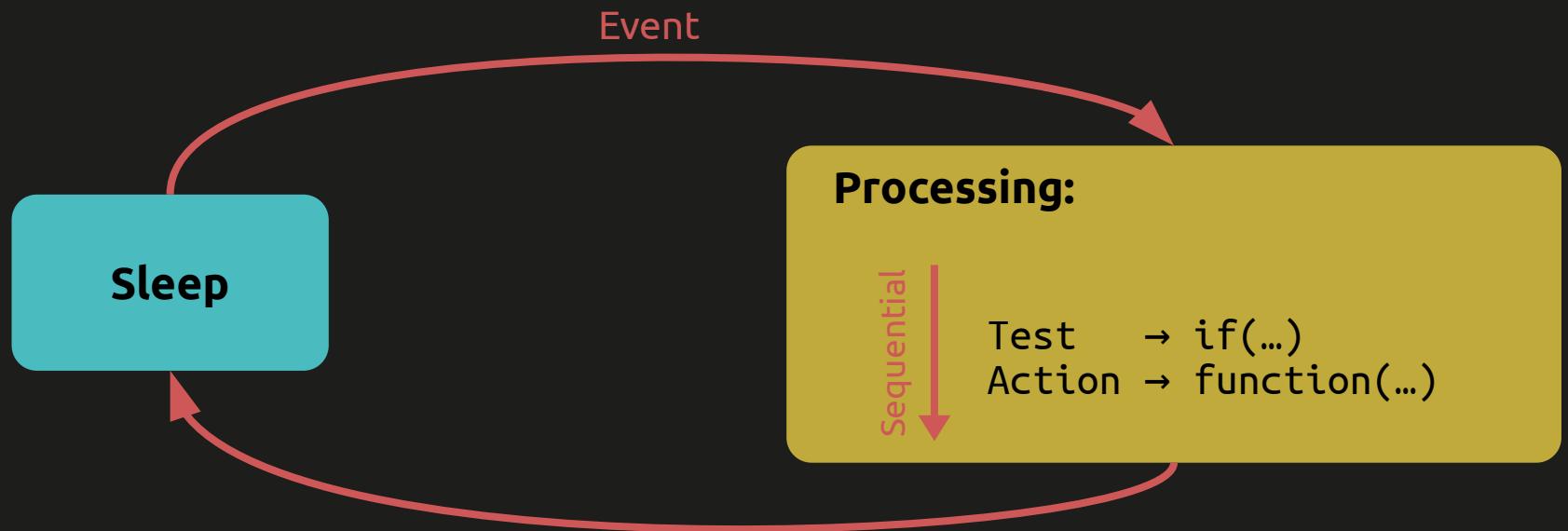
Système = Hardware + Application

Le système (composé du matériel et du logiciel) est développé dans l'optique de répondre à des spécifications ou exigences bien définies.



Si application == supervision

Dans 90 % des cas, le traitement logiciel consiste à effectuer une **simple supervision**.

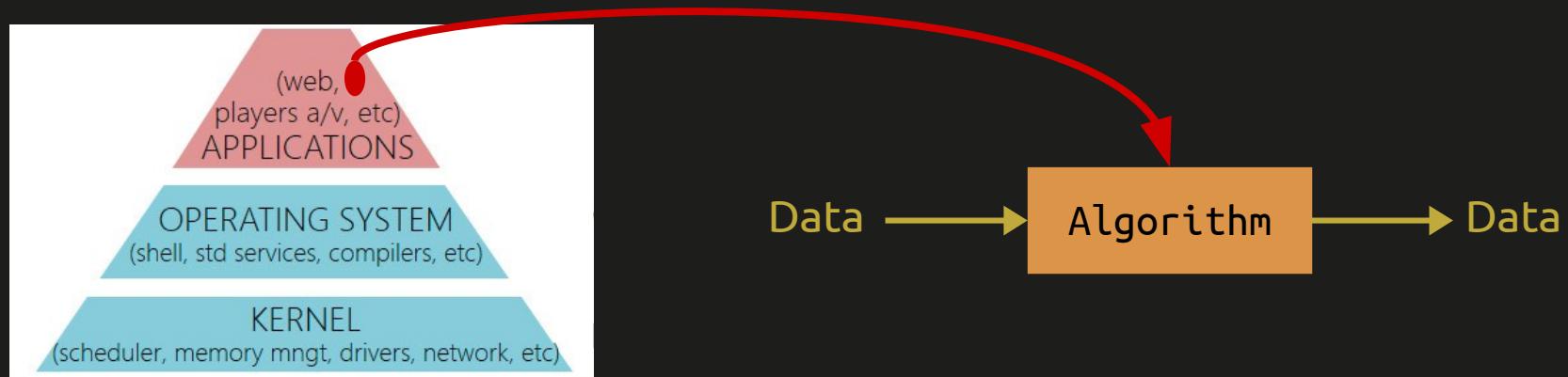


→ Dans ces cas, opter pour un MCU, AP ou GPP

En fonction du niveau de complexité des interfaces (réseau, IHM, ...) et de l'application (mono-/multi-tâches, ...)

Si application == algorithme

Mais de temps en temps, la fonction à exécuter peut être un **algorithme**, c'est-à-dire l'application d'un traitement pour une certaine quantité de données (informations).



Exemples d'algorithme : recherche, tri, traitement numérique du signal (audio, radar, communication, ...)

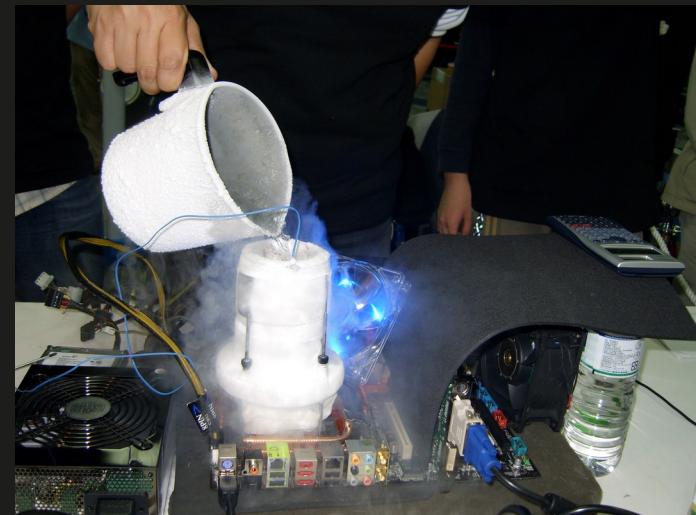
Si application == algorithme

Le premier choix de processeur devrait toujours être un processeur généraliste.

Cependant, quand ses performances ne permettent pas de répondre aux spécifications, il est sage de basculer vers une architecture spécialiste pour :

- Réduire le temps de traitement
- Réduire la taille du code et/ou son empreinte mémoire

Passer d'une architecture généraliste vers une architecture spécialiste devrait toujours être justifié par des mesures.



## Exemple d'algorithme

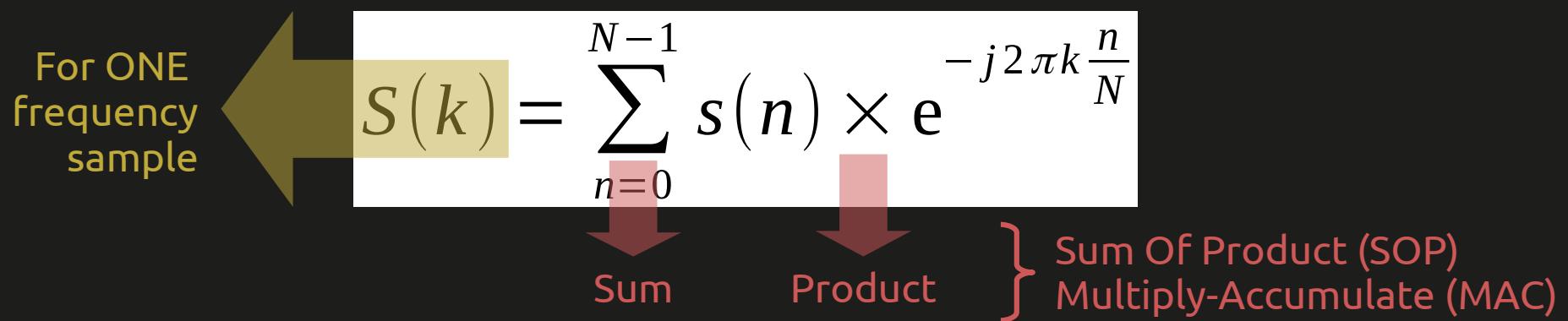
Prenons par exemple l'algorithme de DFT (*Discrete Fourier Transform*) :

For ONE frequency sample

$$S(k) = \sum_{n=0}^{N-1} s(n) \times e^{-j 2 \pi k \frac{n}{N}}$$

Sum      Product

} Sum Of Product (SOP)  
          Multiply-Accumulate (MAC)

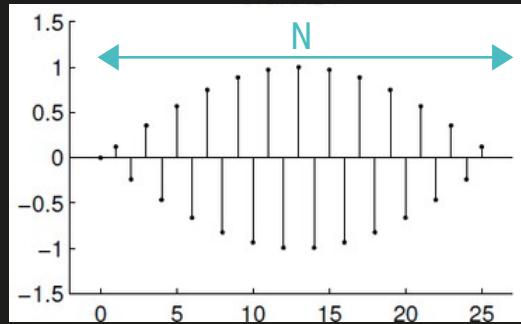


- Chaque produit est indépendant des autres
  - → Traitement en parallèle possible !
- Chaque échantillon de fréquence est indépendant des autres
  - → Traitement en parallèle possible !

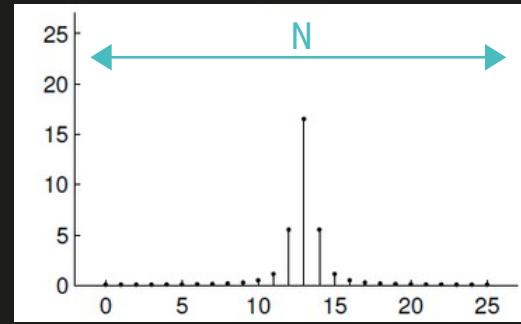
# CHOISIR UN PROCESSEUR

## Exemple d'algorithme

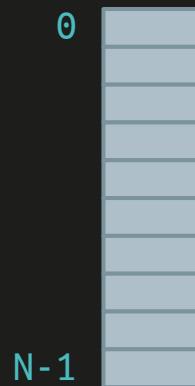
MATH



$$\sum \sum s \times e$$



SOFT



```
for( - )  
| for( - )  
| | s * e
```



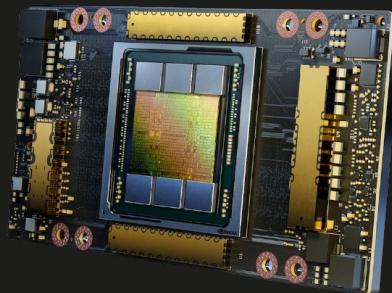
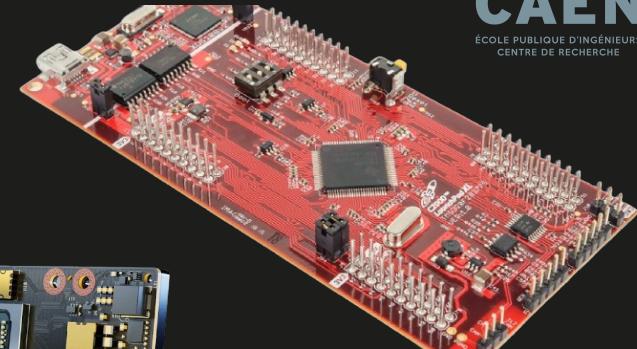
## Choix de l'architecture spécialiste

**DSP** : faible consommation, faible coût, développement bas niveau (C, asm)

**GPU** : hautes performances, prix fort, développement haut niveau (C++, OpenMP, Cuda, ...), gros potentiel de parallélisme.

**FPGA/ASIC** : très cher au développement, mais sur mesure (performances et conso)

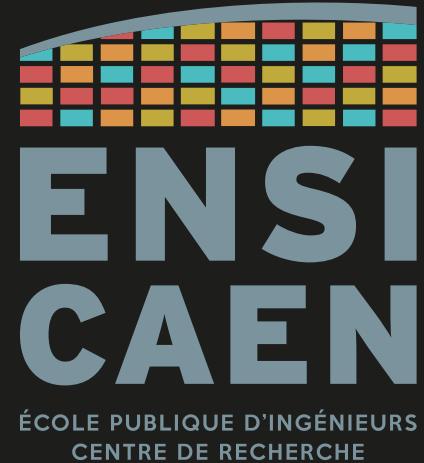
**MPPA** : *Massively Parallel Processor Array*, pas encore répandu, mais gros potentiel (répartir les coeurs pour des parties d'algorithme).



# MODÈLES D'EXÉCUTION

Classification des processeurs selon leur modèle d'exécution

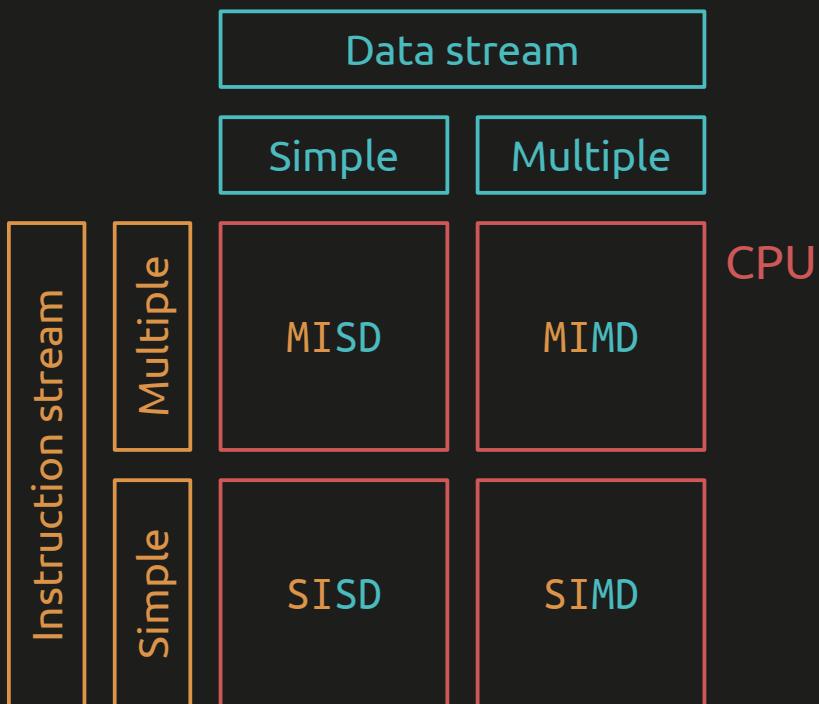
SISD – SIMD – MISD – MIMD



Les prochaines diapos concernent des termes que vous entendrez régulièrement. Elles n'ont pas pour vocation de donner des explications approfondies mais de vous donner une première idée sur leur définition.



## Classification de Flynn (1972)



*Simple data stream*: chaque opérande ne contient qu'une seule donnée (une case mémoire par opérande).

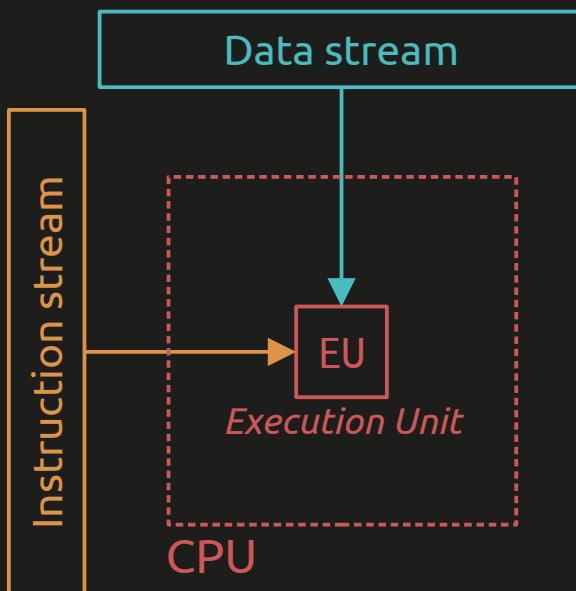
*Multiple data streams*: chaque opérande contient plusieurs données (un tableau fixe par opérande).

*Single instruction stream*: le CPU peut exécuter une instruction à la fois (exécution séquentielle).

*Multiple instruction streams*: le CPI peut exécuter plusieurs instructions simultanément, soit en utilisant le parallélisme de données (ex : boucle *forall*) ou le parallélisme de contrôle (ex : sections parallèles).



## SISD – Single Instruction stream, Single Data stream



Le processeur exécute une instruction à la fois, chaque instruction traitant une donnée.

Architecture mono-processeur classique :

→ Architecture Von Neumann

→ Micro-contrôleurs, anciennes générations de processeurs

→ Processeur séquentiel (pas de parallélisme)

→ Processeur scalaire

→ Une seule donnée (une seule case mémoire) par opérande

## SISD – Single Instruction stream, Single Data stream

*Example: TI C6600 assembly language  
Adding two floats*

```
; Single Precision ADD
ADDSP      A17, A5, A5

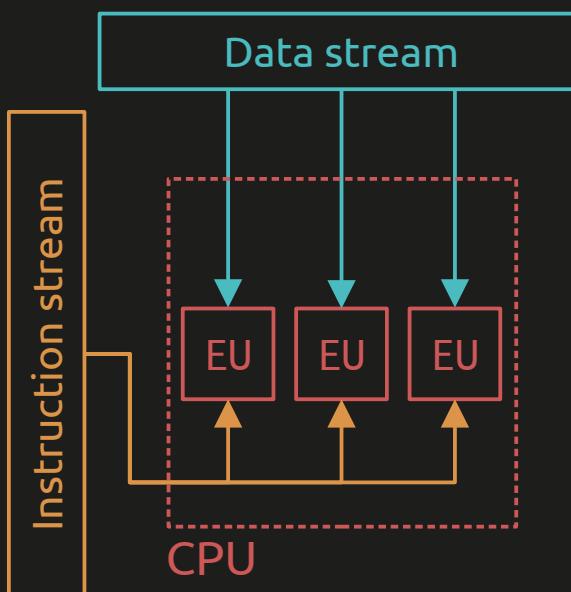
; Result:
; A5 = A5 + A17
```

*Example canonical C:  
Adding two floats*

```
float a, b ;
// Initialising a and b ...

a = a + b ;
```

## SIMD – Single Instruction stream, Multiple Data streams



La même instruction est exécutée par plusieurs EU, chaque EU traite sa propre donnée. Ainsi le CPU exécute la même instruction sur plusieurs données.

Archi. parallèle avec unité de contrôle centralisée :

→ Processeur vectoriel

→ GPU

→ Jeu d'instructions SSE et AVX d'Intel pour x86

SSE = Streaming SIMD Extension (SSE, SSE2, SSE3, SSE4)

AVX = Advanced Vector Extensions (AVX, AVX2, AVX-512)

## SIMD – Single Instruction stream, Multiple Data streams

*Example: TI C6600 assembly language  
Adding two couples of floats*

```
; Dual ADD Single Precision
DADDSP    A21:A20, A25:A24, A25:A24

; Result:
; A25 = A25 + A21
; A24 = A24 + A20
```

; Just like the SSE for Intel, the C6600  
; DSP has a C extension (C functions)  
; for vectorial instructions

*Example: x86 SSE C, adding four couples of floats*

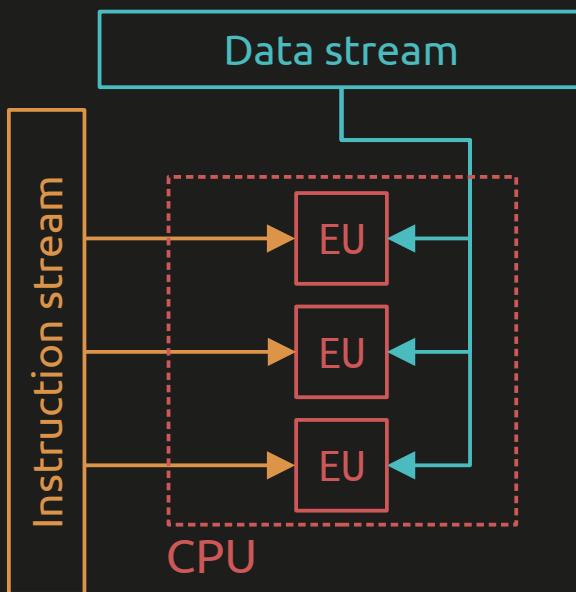
```
float A[N], B[N], C[N] ;

for( int i = 0 ; i < N ; i += 4 ) {
  __m128 reg_b = _mm_load_ps( &B[i] );
  __m128 reg_c = _mm_load_ps( &C[i] );
  __m128 reg_a = _mm_add_ps( reg_b , reg_c ) ;
  _mm_store_pd( &A[i] , reg_a );
}
```

Lanes per type in a 128-bit SIMD register

int8x16	s0	s1	s2	s3	s4	s5	s6	s7	s8	s9	s10	s11	s12	s13	s14	s15
int16x8	s0		s1		s2		s3		s4		s5		s6		s7	
int32x4 / float32x4	s0 (x)				s1 (y)				s2 (z)				s3 (w)			
float64x2	s0 (x)								s1 (y)							

## MISD – Multiple Instruction streams, Single Data stream



Chaque UE exécute une instruction propre, chaque EU traite une seule donnée.

Peu d'applications en pratique :

→ redondance de code (détection d'erreur d'exécution)

→ Processeur VLIW (*Very Long Instruction Word*)

Ex. C66xx Texas Instruments DSP

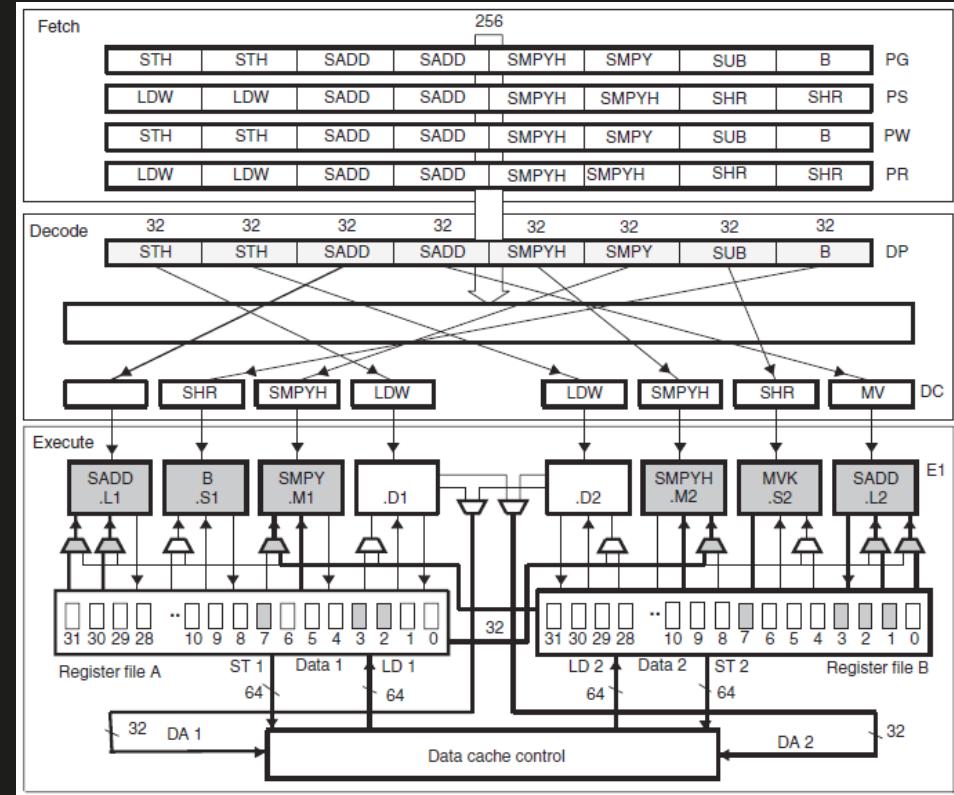
## MISD – Multiple Instruction streams, Single Data stream

*Example: TI C6600 assembly language  
Simultaneously adding and multiplying*

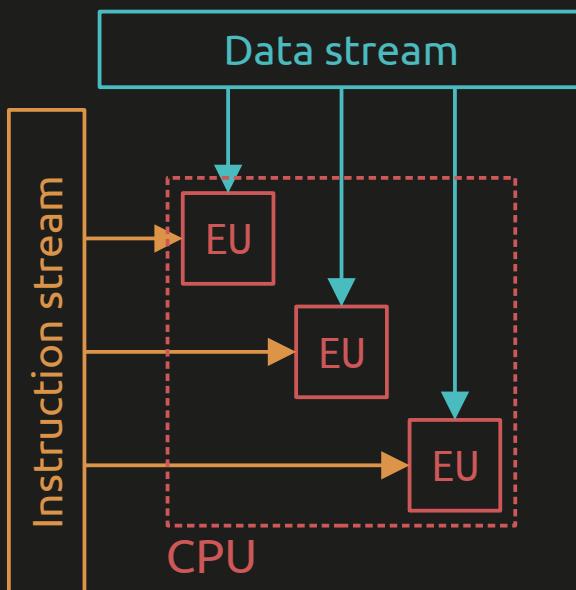
```
; ADD Single Precision
; MULTIPLY Single Precision
ADDSP      A3, A9, A3
|| MPYSP    B3, B9, B3
```

; The pipes (||) explicitly indicate that  
; instructions must be executed in parallel  
; (use of software pipeline)

; Result
; A3 = A9 + A3
; B3 = B9 + B3



## MIMD – Multiple Instruction streams, Multiple Data streams



Chaque EU exécute son propre lots d'instructions (les EU peuvent être groupées), en agissant chacun sur des données différentes.

Archi parallèle avec unités de contrôle indépendantes

→ Processeur super-scalaire

→ N'importe quel GPP moderne : x86-x64 (CISC), Cortex-A (RISC)

→ Inclut le SPMD (*Single Program, Multiple Data*)

## MIMD – Multiple Instruction streams, Multiple Data streams

*Example: TI C6600 assembly language  
Simultaneously adding and multiplying two different couples of data*

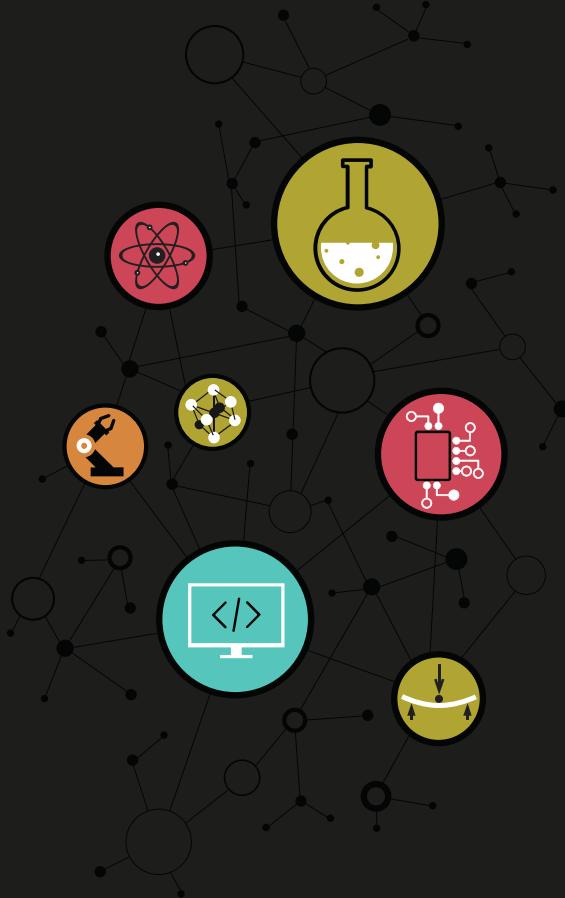
```
; Dual ADD Single Precision
; Dual SUBSTRACT Single Precision
    DADDSP    A21:A20, A25:A24, A25:A24
||    DSUBSP    B25:B24, B23:B22, B23:B22
```

; The pipes (||) explicitly indicate that  
; instructions must be executed in parallel  
; (use of software pipeline)

```
; Result
; A25 = A25 + A21
; A24 = A24 + A20
; B23 = B25 - B23
; B22 = B24 - B22
```

*Example: C and OpenMP  
Parallelisation of for loop*

```
#pragma omp parallel reduction(+:acc)
{
    #pragma omp for schedule(static)
    for( k = 0; k < size; k ++ )
    {
        acc += A[i * size + k] * x[k];
    }
}
```



Dimitri Boudier – PRAG ENSICAEN  
[dimitri.boudier@ensicaen.fr](mailto:dimitri.boudier@ensicaen.fr)

Avec l'aide précieuse de :

- Hugo Descoubes (PRAG ENSICAEN)



Except where otherwise noted, this work is licensed under  
<https://creativecommons.org/licenses/by-nc-sa/4.0/>