2. Processeurs Embarqués pour les Systèmes Intelligents et Connectés

Les processeurs embarqués pour les systèmes intelligents et connectés peuvent prendre des formes physiques diverses. Cela peut aller du minuscule microcontrôleur pour les systèmes exécutant des tâches simples où de faibles ressources de calcul sont requises, au gros microprocesseur pour les systèmes devant exécuter des tâches complexes. Lorsque l'on développe un système intelligent, un choix devra être fait entre utiliser un :

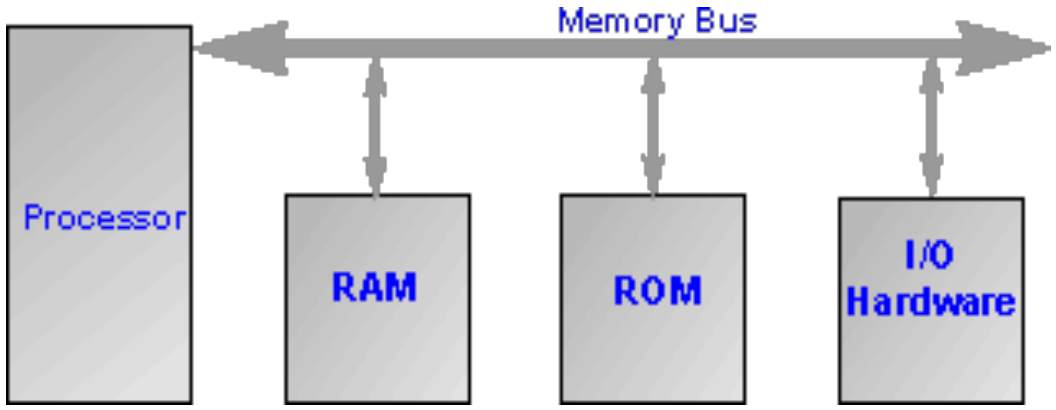
→ **Système à base de microprocesseur**

→ **Système à base de microcontrôleur**

Peu importe les exigences du système, les processeurs embarqués auront plus ou moins la même structure et les mêmes blocs de circuit de base.

❑ Architecture de base

Le processeur embarqué est constitué de plusieurs composants, chacun exécutant des fonctions spécifiques dans le but de permettre le fonctionnement global du système.



Typiquement, les programmes sont stockés dans une mémoire non volatile (ROM) mémoire morte , et au démarrage, sont chargés dans une mémoire volatile (RAM) pour être exécuté par le CPU. Les données sont échangées

❑ Programmation

Le logiciel embarqué dans les systèmes intelligents s'appelle firmware. Les programmes sont écrits pour exécuter des fonctions bien spécifiques normalement stockées sur la puce (chip).

Un langage de haut niveau est souvent utilisé lors de la programmation, puis compilé afin de fournir un code machine (bas niveau) compréhensible par le processeur embarqué. L'opération de chargement du code vers le processeur est appelée flashage.

❑ Microprocesseur

Un microprocesseur contient essentiellement un CPU (Central Processor Unit) et quelques composants additionnels, mais utilise des puces externes pour la mémoire et les interfaces de périphériques.

Les microprocesseurs ont pour tendance, être adaptés pour les calculs de hauts niveaux, où :

- **La performance est le facteur clé**
- **La consommation énergétique, la miniaturisation et le coût sont des facteurs moins importants**

La scalabilité est un autre atout, en effet, on peut étendre les capacités de mémoire et le nombre de périphériques en fonction des besoins de l'application.

❑ Microcontrôleur

Un microcontrôleur est constitué d'une puce unique contenant essentiellement un CPU (Central Processor Unit), une mémoire et inclut tous les supports et circuits périphériques requis. Il peut fonctionner de façon autonome après programmation.

Les microcontrôleurs ont pour tendance, être adaptés pour les petits systèmes, ou les systèmes à usage spécifique où :

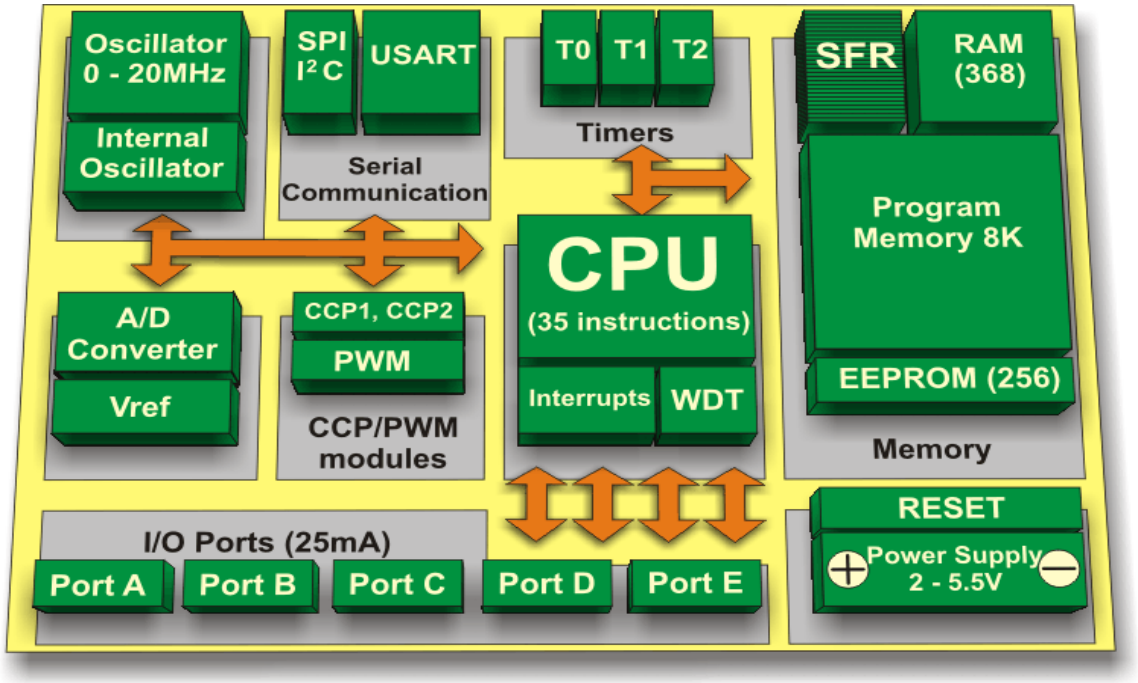
- **La taille, la faible consommation d'énergie et le coût sont des facteurs clés**
- **Un compromis doit être fait avec la performance**

La scalabilité et la flexibilité posent défaut, car on ne peut pas étendre les capacités du système ou ajouter/enlever des composants.

❑ Composants essentiels d'un processeur embarqué

Peu importe le type de processeur embarqué, les composants de base sont les mêmes, et inclus :

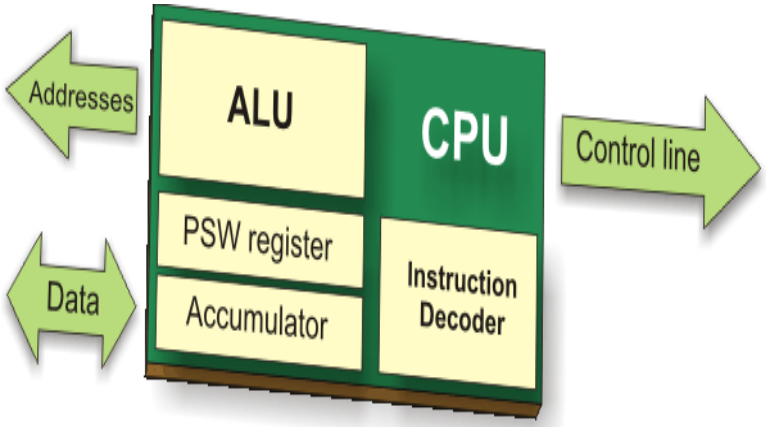
- CPU
- Mémoires
- Bus
- Interfaces I/O
- Horloge



- ***USART (Universal Asynchronous Receiver Transmitter***) est un émetteur-récepteur asynchrone universel. la liaison entre l'ordinateur et le port série.

❑ Central Processor Unit (CPU)

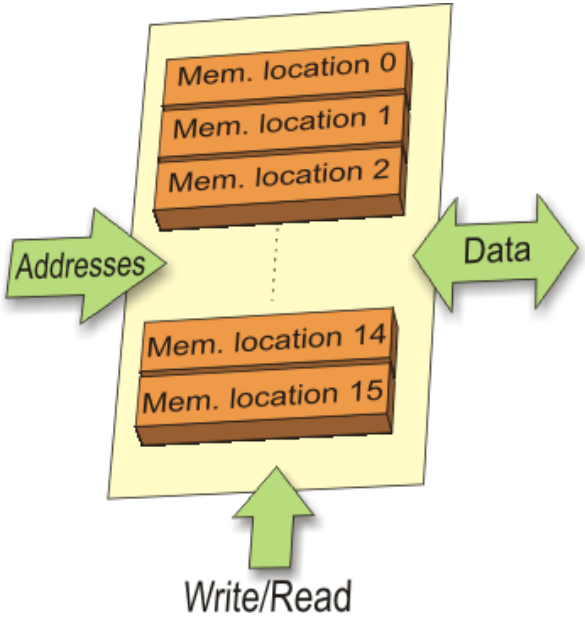
Le CPU est le cœur de n'importe quel système intelligent, car l'ensemble du traitement de l'information s'effectue à son niveau.



- **Unité de Commande :** cherche les instructions en mémoire, les décode et coordonne le reste du processeur pour les exécuter.
- **Unité Logique et Arithmétique (ALU) :** exécute les instructions arithmétiques et logiques demandées par l'unité de commande.
- **Accumulateur et registre d'état:** sont des cellules mémoires internes d'accès très rapide. Ils servent à stocker des variables, les résultats intermédiaires d'opérations ou encore des informations de contrôle du

❑ Mémoires

La mémoire est utilisée pour le stockage d'instructions, de données...



→**Mémoires de données (RAM) :** est l'endroit où les données temporaires utilisées par le programme sont stockées. Ces mémoires est habituellement volatiles.

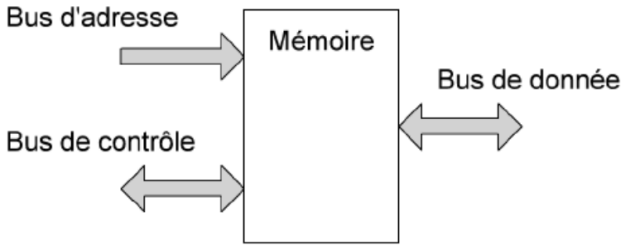
→**Mémoires de programme (ROM) :** conservent le programme écrit par l'utilisateur et ces dernières sont habituellement non volatiles.

Désormais, il est possible d'enregistrer des informations dans certaines mémoires de type ROM

- **EEPROM** (Electrically Erasable Programmable Read Only Memory)
- **Flash EEPROM :** une version de l'EEPROM plus performante.

❑ Bus

Le bus est un ensemble de lignes de liaison qui assurent la communication entre les différents composants du processeur embarqué.



→**Bus de données (bidirectionnel) :** permet de transférer entre composants des données

→**Bus d'adresses (unidirectionnel) :** permet de transférer entre composants des adresses

→**Bus de contrôle :** permet l'échange entre les composants d'informations de contrôle

- **Volume d'information :** largeur de bus = nombre de bits (lignes) du bus
- **Vitesse du bus :** nombre de paquets de données envoyés ou reçus par seconde (en Hertz)
- **Bande passante :** débit de données = largeur de bus de données * vitesse du bus
- **Profondeur** : nombre de mots mémoire = $2^{largeur\ de\ bus\ d'adresse}$

❑ Interfaces d'entrée/sortie

Les interfaces I/O permettent au processeur embarqué d'accéder au monde extérieur via un cycle de lecture (capteur) et d'écriture (actionneur).

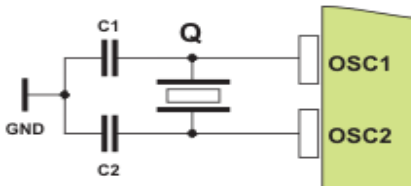
Les applications des systèmes intelligents ont en commun le fait qu'ils ne nécessitent que très rarement des calculs complexes alors qu'ils sont très friands de manipulations d'informations d'entrées/sorties.

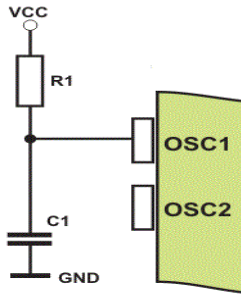
Il existe aussi des périphériques, on peut en citer :

- **Horloge :** tous les microcontrôleurs ont besoin d'une horloge pour fonctionner, l'horloge est généralement fournie par un circuit externe.
- **Timers :** pour compter le temps ou des événements, il est constitué d'un compteur dont on peut ajuster l'horloge, lire la valeur et écrire la valeur.
- **Convertisseurs Analogiques Numériques (CAN) :** utilisés pour convertir une grandeur analogique (tension) en donnée numérique.

❑ Horloge

L'horloge est utilisée par pratiquement tous les composants du processeur embarqué notamment le CPU pour cadencer l'exécution du programme.





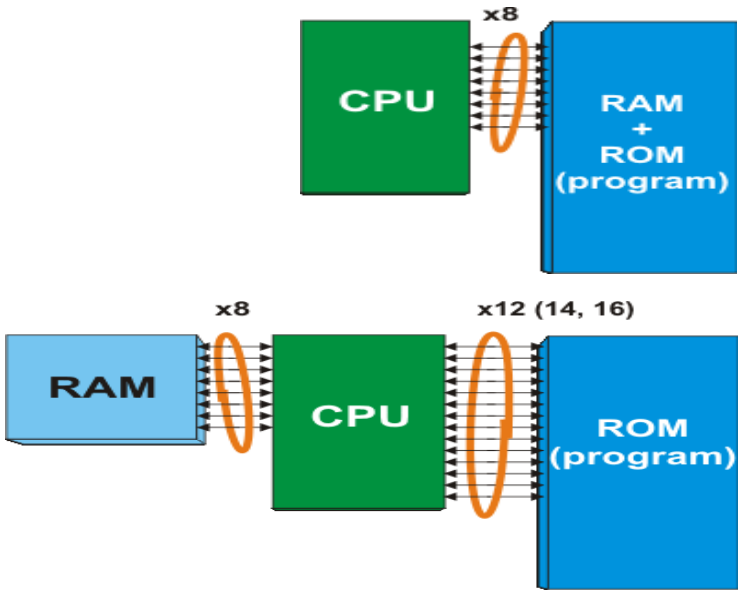
→**Oscillateur à quartz :** est utilisé pour la stabilisation de fréquence, fonctionne avec une fréquence bien précise qui n'est pas affectée par les perturbations.

→**Oscillateur RC :** est utilisé lorsque la fréquence de fonctionnement du système n'a pas besoin de composants additionnels couteux pour la stabilisation.

- **Cycle d'horloge :** correspond à une période de l'horloge.
- **Fréquence d'horloge :** nombre de cycles effectués par une horloge en une seconde.

❑ Modèle de conception

Tous les processeurs embarqués utilisent l'un des deux modèles de conception de base appelés architectures de :



→**von-Neumann :** seulement un bloc de mémoire pour le programme et les données, et partageant un même bus.

→**Harvard :** séparation de la mémoire de programme et la mémoire de données, chacune accessible via des bus distincts.

La structure de Harvard permet de transférer données et instructions simultanément, ce qui permet un meilleur gain de performances.

❑ Jeux d'instructions

Toutes les instructions que peut comprendre le processeur embarqué constituent ce que l'on appelle jeux d'instructions.

→ **CISC (Complex Instruction Set Computer) :** jeu d'instructions étendu conçu pour reconnaitre plusieurs types d'instructions notamment ceux complexes qui nécessitent plusieurs cycles horloge pour s'exécuter (lent).

→ **RISC (Reduced Instruction Set Computer) :** jeu d'instructions réduites ne reconnaissant et n'exécutant que des opérations de base ne nécessitant qu'un seul cycle d'horloge (rapide). La complexité est reportée niveau logiciel (programmation), donc un compilateur plus puissant est requis.

L'architecture RISC est celle étant la plus utilisée dans les processeurs embarqués,

❑ Jeux d'instructions : classification

Chaque instruction indique une tâche bien spécifique à exécuter, on peut en citer les instructions :

- → **ALU :** addition, soustraction, opérations logiques (AND, XOR)
- → **Orientées Registre :** manipulent un octet se trouvant dans la RAM
- → **Orientées bits :** manipuler directement un bit d'un registre
- → **Opérant sur une valeur :** entre l'accumulateur W et une valeur K précisée dans l'instruction. Le résultat va dans W.
- → **De saut et appel de procédures :** sauter à une autre position dans le programme et de continuer l'exécution à partir de cette position;

❑ Exécution des instructions

Une instruction est exécutée en deux phases :

→ **Phase de recherche** du code binaire de l'instruction stocké dans la mémoire de programme.

→ **Phase d'exécution** où le code de l'instruction est interprété par le processeur puis exécuté.

On définit le cycle instruction (**Tcy**) comme le temps nécessaire à l'exécution d'une instruction. Attention de ne pas confondre cette notion avec le cycle d'horloge (**Tosc**) qui correspond au temps nécessaire à l'exécution d'une opération élémentaire (soit un coup d'horloge). On note **MIPS** (Million Instruction Per Second) le nombre de millions d'instructions exécutables par seconde.

Par exemple, un microcontrôleur capable d'exécuter 1 instruction par cycle d'horloge CPU pourra exécuter 20MIPS si il est cadencé par un oscillateur à quartz à 20MHz.

❑ Exécution d'un programme

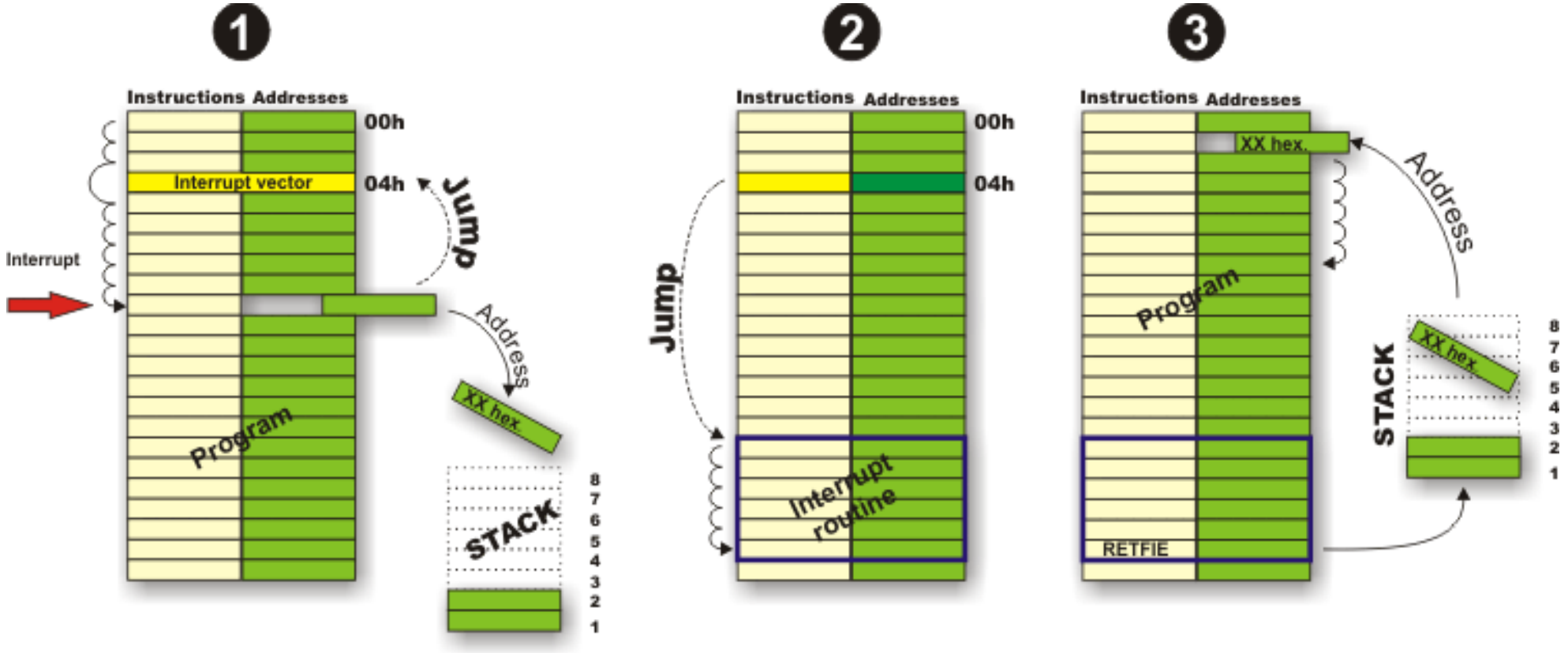
A la mise sous tension, le processeur va chercher la première instruction qui se trouve à l'adresse 0000 de la mémoire de programme, l'exécute puis va chercher la deuxième instruction à l'adresse 0001 et ainsi de suite (sauf cas de saut ou d'appel de sous-programme).

On parle de fonctionnement séquentiel

Une instruction occupe un emplacement qui est défini par une adresse. Le processeur peut alors sélectionner l'emplacement souhaité grâce au bus d'adresse et il peut lire son contenu (ici l'instruction) grâce à son bus d'instruction.

❑ Interruptions

Une interruption est un événement qui provoque l'arrêt du programme principal pour aller exécuter une procédure d'interruption.



(1) Génération d'une interruption (2) Exécution de l'interruption (3) Retour au programme principal

❑ Watchdog

Le chien de garde (watchdog) est un dispositif matériel et logiciel qui permet de se prémunir contre les plantages accidentels. On utilise souvent un timer avec une horloge RC intégrée indépendante de l'horloge système. Lorsque le timer atteint une valeur seuil (time out) on réinitialise (RESET) le système.

L'idée de base est :

- Si l'application fonctionne correctement, alors le timer du watchdog n'est pas incrémenté.
- Par contre s'il y a anomalie, le timer du watchdog s'incrémente continuellement jusqu'à atteindre une valeur seuil qui provoque un RESET du système.

❑ Présentation de quelques processeurs embarqués

Actuellement, il existe plusieurs familles de processeurs embarqués parmi lesquelles on peut citer les plus populaires :

- → **PIC :** c'est la famille de microcontrôleurs la plus populaire. Ils sont peu chers et on les utilise généralement pour l'apprentissage de la programmation de microcontrôleurs.
- → **AVR :** famille de microcontrôleurs popularisée grâce à l'Arduino, utilisée essentiellement pour l'apprentissage de l'embarqué et pour le développement de nombreux projets amateurs.
- → **ARM :** *architecture* de microprocesseur popularisé grâce à la Raspberry Pi, il fait partie des architectures les plus utilisées dans les systèmes embarqués.

❑ Les microcontrôleurs PIC

Les PIC (Programmable Integred Circuit) forment une famille de microcontrôleurs de la société *Microchip Technology*.

- → Architecture : **Harvard**
- → Jeux d'instructions : **RISC** (35 instructions)
- → Famille : **8 bits, 16 bits, 32 bits**
- → Rapidité : **Tcy=4*Tosc (8bits), Tcy=2*Tosc (16bits), Tcy=1*Tosc (32bits)**
- → Largeur de bus : **8 bits**
- → Mémoire : **SRAM, FLASH**
- → Adressage RAM : **par bloc de 256 octets**
- → Communication : **UART/USART, SPI, I2C…**

❑ Les microcontrôleurs AVR

Les microcontrôleurs AVR forment une famille de microcontrôleurs de la société *Atmel*.

- → Architecture : **Harvard**
- → Jeux d'instructions : **RISC** (131 instructions)
- → Famille : **Atmega, Tiny, Xmega**
- → Rapidité : **Tcy=1*Tosc**
- → Largeur de bus : **8 bits**
- → Mémoire : **SRAM, FLASH, EEPROM**
- → Adressage RAM : **peuvent adresser toute la RAM**
- → Communication : **UART/USART, SPI, I2C…**

❑ Les microcontrôleurs ARM

Les architectures ARM (Acorn RISC Machine) forment une famille d'architectures développées (ne fabrique pas) par ARM Holdings (ARCON).

- → Architecture : **von Neumann**
- → Jeux d'instructions : **RISC** (58 instructions)
- → Famille : **série ARMv4,5,6,7**
- → Rapidité : **Tcy=1*Tosc**
- → Largeur de bus : **32 bits, 64 bits (rare)**
- → Mémoire : **SRAM, FLASH, EEPROM**
- → Adressage RAM : **peuvent adresser toute la RAM**
- → Communication : **UART/USART, SPI, I2C, USB, Ethernet…**

Les Cortex sont des gammes de processeurs 32 bits comprenant le jeu d'instruction ARMv7 très utilisés dans les smartphones modernes.