

E302A Électronique embarquée et laboratoire

## Séance 1

# Introduction aux systèmes embarqués

*Sébastien Combéfis*

*2017–2018*



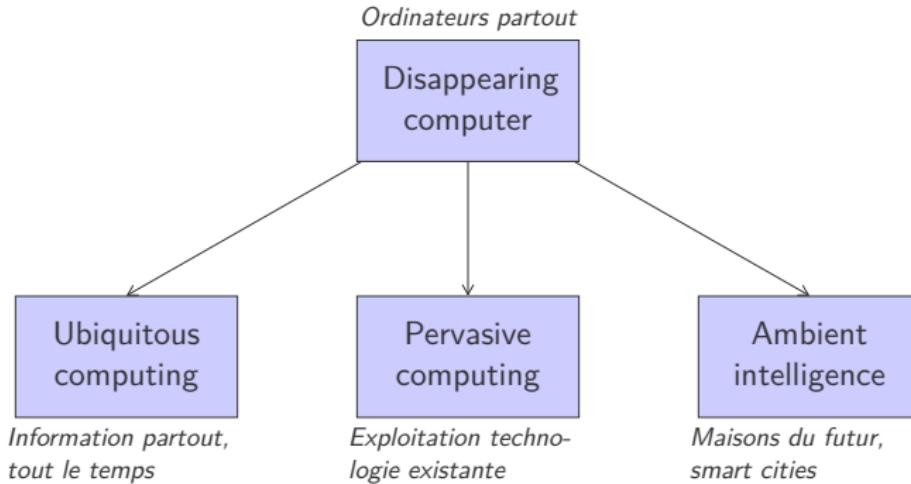
Ce(tte) œuvre est mise à disposition selon les termes de la Licence Creative Commons  
Attribution – Pas d'Utilisation Commerciale – Pas de Modification 4.0 International.

# Objectifs

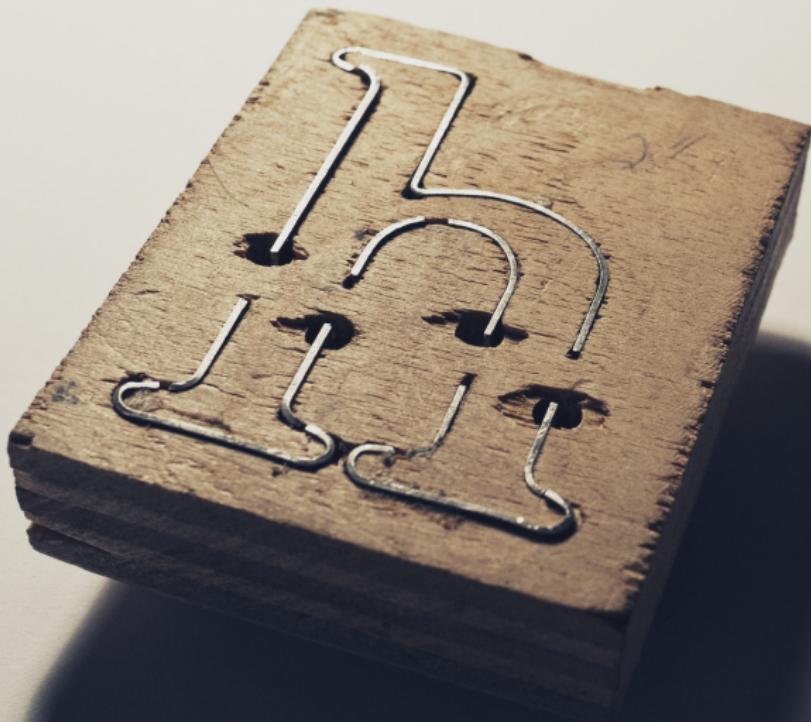
- Définir et comprendre la notion de **système embarqué**
  - Définition et composants d'un système embarqué
  - Types, caractéristiques et contraintes
  - Système connecté et objets intelligents
- Passer en revue les aspects **hardware et software**
  - Type de processeurs, unités hardware, interruption et énergie
  - Programmation, chaîne de compilation, système d'exploitation
- Vue de l'**ingénieur et du client**

*Exigences et aspects multi-disciplinaires*

# Technologie d'information du futur



# Système embarqué



# Système embarqué



- Présence grandissante d'appareils **intelligents et connectés**

*Constitué d'un « ordinateur de bord », le système embarqué*

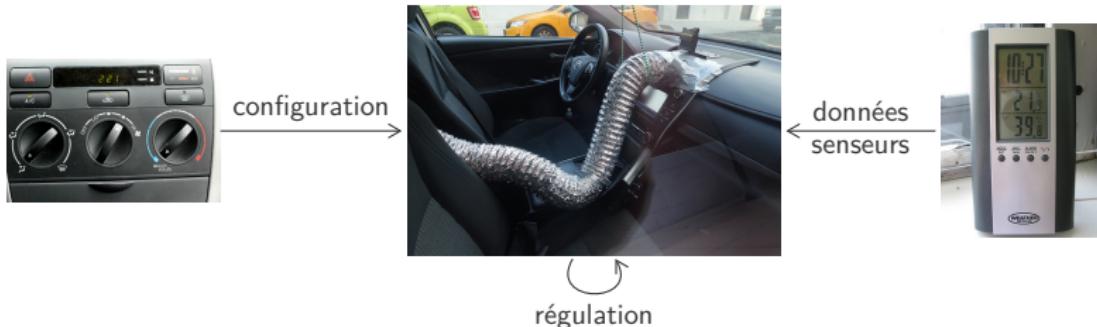
- Système embarqué est **électronique et informatique**

*Il est embarqué au sein d'un produit plus large*



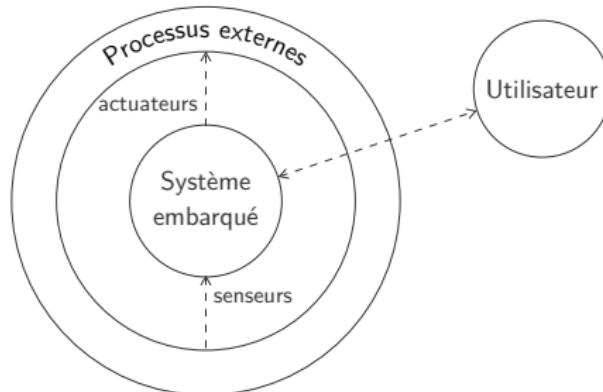
# Traitement d'information (1)

- Système embarqué est un **intermédiaire**
  - Réception d'information provenant de son environnement
  - Traitements de cette information
  - Action à travers le produit hôte qui l'héberge



# Traitement d'information (2)

- **Système embarqué** se trouve au sein d'un système plus large
  - **Input** en interrogant le système via des senseurs
  - **Output** en agissant sur le système via des actuateurs
- Possibilité d'interaction directe avec l'**utilisateur**



# Vue orientée-produit

- Système embarqué implémente un **processus**

*Collecte d'inputs pour produire des outputs*

- Inputs produits par des **senseurs** ou capteurs

*Température, gyroscope, configuration, communication réseau...*

- Outputs produits par des **actuateurs** ou actionneurs

*Moteur, vérin, baffle, LEDs, écran, communication réseau...*



# Définition de l'IEE

- Définition de l'**Institution of Electrical Engineers** (IEE)

*"the devices used to control, monitor or assist the operation of equipment, machinery or plant. 'Embedded' reflects the fact that they are an integral part of the system. In many cases their embeddedness may be such that their presence is far from obvious to the casual observer [...]"*

- Système embarqué souvent **associé à une machine**

*Fait partie d'un système plus large, et est souvent caché*

# Traditionnel versus embarqué

- Un système embarqué n'a **pas de raison d'être seul**

*Composant d'un système plus large, parfois invisible*

- **Situation** dans les années 2000

- Informatique focus sur  $150 \cdot 10^6$  d'ordinateurs « traditionnels »
- Processeurs se trouvent dans  $8 \cdot 10^9$  de systèmes embarqués

- Système embarqué exécute une **tâche spécifique**

- Moins de ressources et moins performants
- Software spécifique
- Fonctionnement en autonomie, pas de clavier, affichage réduit

# Exemple (1)

- Système embarqué **seul ou à plusieurs**

*Isolé et cloisonné dans l'hôte/collaborant avec d'autres systèmes*

- Simple **gestion des états** de fonctionnement d'une machine

*Machine à laver, four à micro-onde, toaster...*

- **Traitements de signaux** audio et vidéo

*Appareil photo et caméra numérique, console de jeu...*

- Système de **communication et d'information**

*Smartphone, modem-routeur, fax, antenne-relais, satellite...*

## Exemple (2)

- Plus que **40% de la valeur** d'une voiture moderne

*Multiples systèmes embarqués qui peuplent la voiture*



# Type d'application visé

- **Usage général** avec applications similaires aux « traditionnels »  
*Embarqué dans package de petites tailles comme PDA, ATM...*
- Boucle rétroactive fermée de **contrôle** de système temps-réel  
*Moteur de voiture, centrale nucléaire, contrôle aérien...*
- Calcul sur gros flux de données comme **traitement de signal**  
*Traitement audio et vidéo, radar, sonar...*
- **Communication et réseaux** pour transmission de données  
*Utilisés en téléphonie et sur internet*

# Type de fonction (1)

- Assurer qu'une **loi de contrôle** est respectée

*Contrôleur PID pour assurer stabilité vol stationnaire*

- Représentation d'une **logique séquentielle**

*Alternation entre modes comme machine à laver*

- Calculs et opérations en **traitement de signaux**

*Compression/décompression de contenu multimédia, filtrer son...*

# Type de fonction (2)

- Rôle d'**interface** pour des applications spécifiques  
*Gérer des alarmes sonores, contrôleur disque SSD...*
- Réaction à des **fautes** ou gestion d'exceptions  
*Détection, diagnostic et correction de la faute*

# Taille et complexité

- Système de **petite échelle** avec un microcontrôleur 8–16 bits  
*Niveau d'une carte, limitation ressources pour puissance*
- **Moyenne échelle** avec microcontrôleurs 16–32 bits, DSP, RISC  
*Hardware de type ASSPs et IPs et software avec des RTOS*
- Système **sophistique** avec processeurs configurables et PLA  
*Co-design et intégration software/hardware très important*

# Fiabilité

- Un système embarqué doit avoir une grande **fiabilité**

*Car il doit très souvent pouvoir fonctionner en autonomie*
- Plusieurs **caractéristiques mesurables** de fiabilité
  - **Robustesse**, proba de fonctionnement en  $t$ , si ok en  $t_0$
  - **Maintenabilité**, proba de fonctionnement  $d$  après une erreur 
  - **Disponibilité**, proba de fonctionner en  $t$
  - **Sûreté**, il ne cause pas de nuisances 
  - **Sécurité**, communications confidentielles et authentifiées
- Aspects à prendre en compte au moment de la **conception**

*Et compromis à prendre car impossible de tous les maximiser*

# Efficacité

- Un système embarqué doit être **efficace** pour sa fonction

*Plus qu'un ordinateur « traditionnel », justifier son développement*

- Cinq **aspects mesurables** d'efficacité

- **Consommation énergétique** en Watts
- **Taille du code** en octets, lignes de code
- **Exécution** du programme en MIPS, R/W par seconde
- **Poids et taille** en kg et cm
- **Cout de fabrication** et conception en \$, €, ¥, B...



# Contrainte

- Système **spécifique** dédié à une tâche déterminée

*Connaissances sur les conditions et à l'environnement d'utilisation*

- Contraintes de **temps-réel** sur le temps de réaction 

- **Strict**, réaction garantie endéans temps de réponse fixé
  - **Souple**, réaction au mieux endéans temps de réponse fixé

*"A real-time constraint is called hard, if not meeting that constraint could result in a catastrophe."*

- Alimentation du système embarqué en **énergie**

*Batterie évolue moins vite que besoins en puissance de calcul*

# Système connecté

- **Intelligence ambiente**, troisième ère de l'informatique

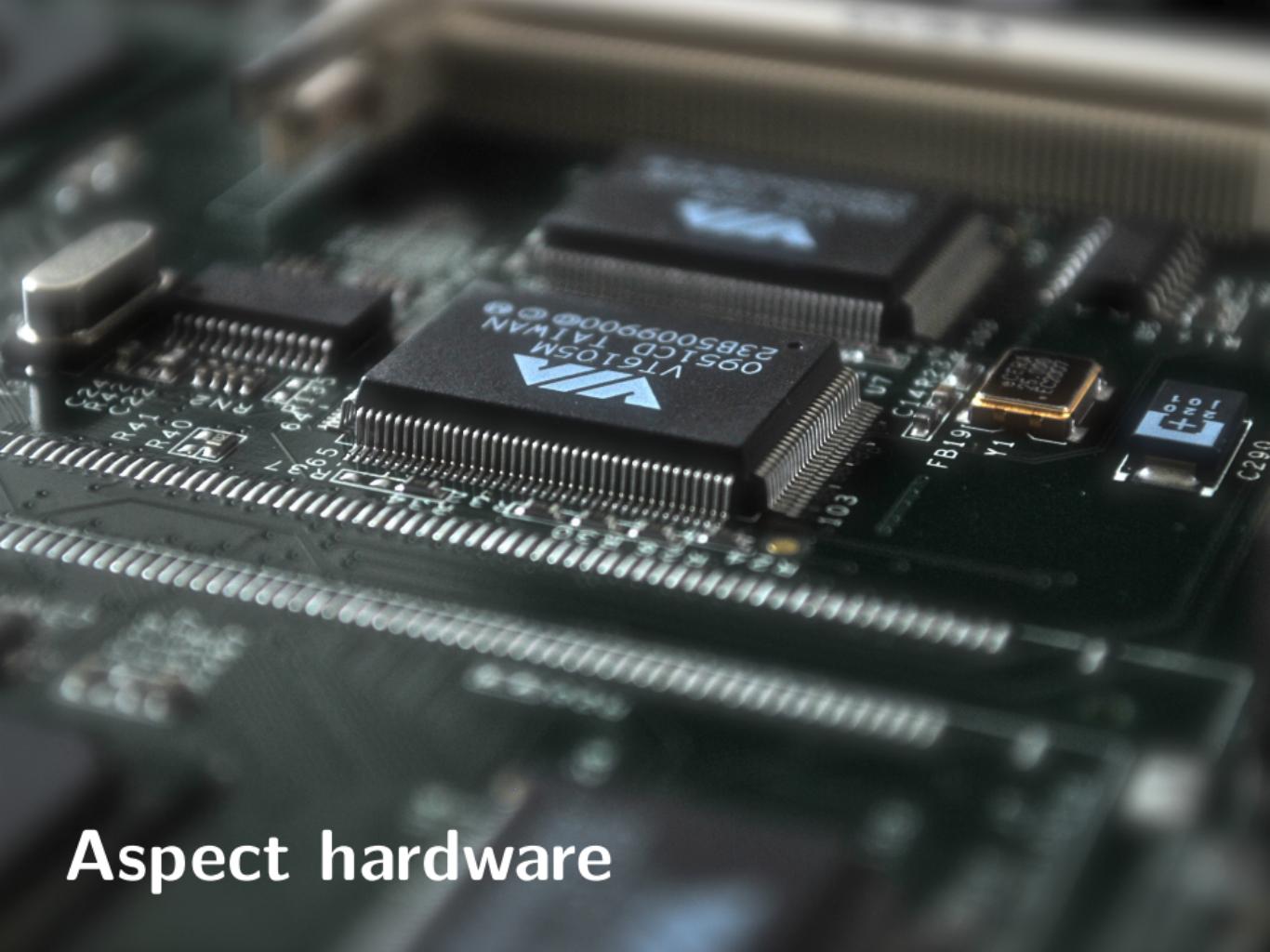
*Large gamme de petits appareils « intelligents »*

- Système embarqué doté de **moyens de communication**

*Récupération/échange d'informations, réseaux de senseurs*

- Apparition d'**applications distribuées** pour données et calculs

*Domotique et maison intelligente, smart cities, e-health...*

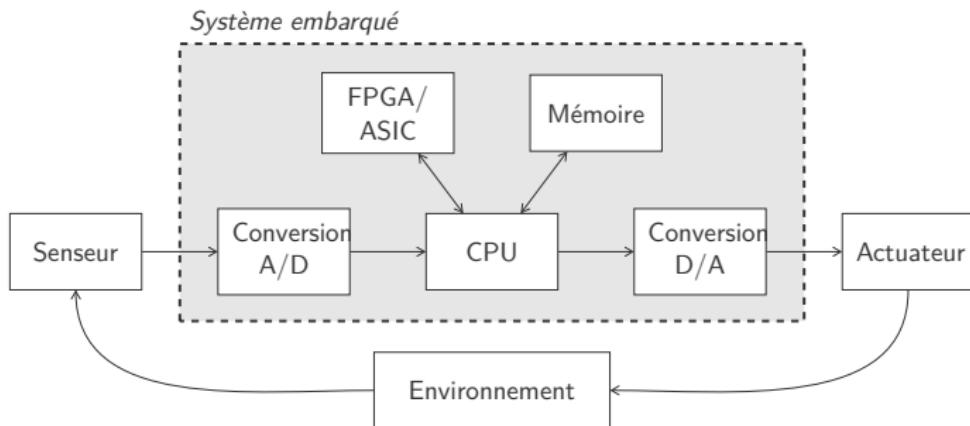


# Aspect hardware

# Boucle hardware

- Hardware d'un système embarqué en "*Hardware in a loop*"

*Méthode de test par envoi de stimuli et mesure de la réponse*



*environnement → senseur → système embarqué → actuateur → environnement → ...*

# Intel 4004

- Premier microprocesseur sur une seule puce est l'**Intel 4004**  
*Créé en 1971 suite à une demande de la firme japonaise Busicom*
- Microprocesseur à **usage général** programmé pour les produits  
*Sur 4 bits, fréquence CPU de 740 kHz max, 2300 transistors*

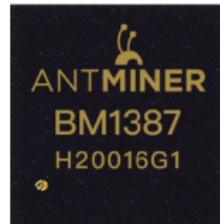
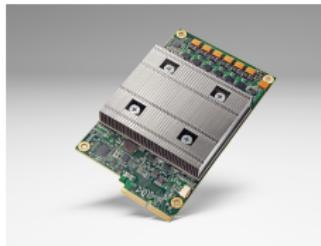


# Unité de calcul (1)

- **Processeurs** exécutent une séquence d'instructions
  - *General Purpose Processor* (GPP)  
*Jeu d'instructions large et varié*
  - *Application-Specific Instruction set Processor* (ASIP)   
*Jeu d'instructions conçu pour un ensemble d'opérations*
- **Circuits spécifiques** implémentent une fonction en hardware
  - *Reprogrammable hardware*  
*Fonction modifiable après fabrication, comme FPGA*
  - *Application-Specific Integrated Circuit* (ASIC)   
*Circuit intégré réalisant les fonctions d'une application*

# ASIC

- Google **Tensor Processing Unit** (TPU)
  - Accélérateur pour le neural network machine learning
  - Utilisé conjointement avec le framework TensorFlow
  - Bande passante mémoire 600 Gio/s, performance 56 TFLOPS
- **Bitcoin Miner** effectue des opérations cryptographiques   
*Groupe des transactions en blocs et calcule un hash du précédent*



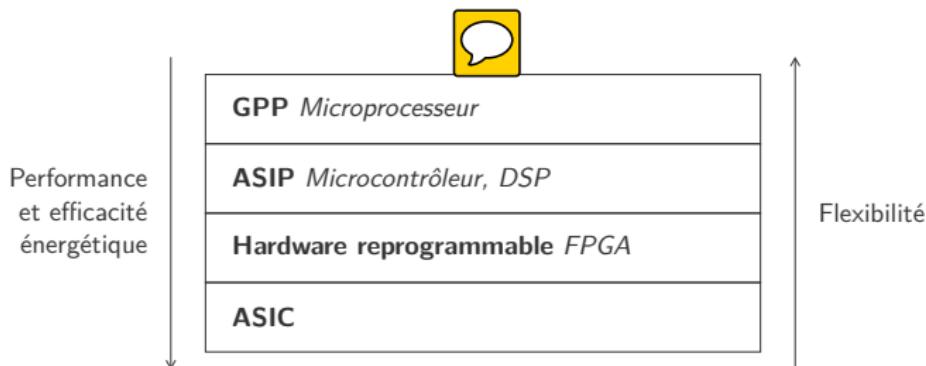
# Unité de calcul (2)

- Comparaison de la **flexibilité** par rapport aux **performances**

*Générique et flexible moins performant et efficace en énergie*

- **Grand cout** de conception et développement des ASIC

*Pas de chaîne de production à grande échelle, un ASIC par besoin*



# Processeur

- Processeur (CPU) permet de réaliser des calculs

*Cœur d'un système qui représente la puissance de calcul*

- Composé de plusieurs unités fonctionnelles

*Program Flow Control Unit (CU), Execution Unit (EU)...*

- Plusieurs considérations pour choisir un processeur

- Jeu d'instructions (plutôt RISC, CISC, etc.)
- Nombre de bits pour un opérande (8, 16, 32...)
- Fréquence d'horloge (Hz), vitesse de calcul (MIPS), Dhrystone

# Microprocesseur

- **Microprocesseur** circuit intégré avec uniquement processeur

*Partie plus importante d'un système de calcul*

- Circuit n'offrant que de la **puissance de calcul**
  - Peut contenir cache, FPU, pipelining, unité superscalaire...
  - Dépend d'éléments externes : RAM, ROM, périphériques...
- Intel Pentium 4, Intel Core i7, AMD Opteron...

# Microcontrôleur

- **Microcontrôleur** circuit intégré avec plusieurs éléments

*Partie plus importante d'un circuit de contrôle/communication*

- Faible charge de calcul et **échange** avec le monde extérieur
  - Possède des unités fonctionnelles : RAM, ROM, E/S...
  - Micro-ordinateur complet dans une seule chip
- Intel 8051, Atmel AVR, Microchip PIC...

# Microprocesseur versus microcontrôleur

- Plusieurs différences entre microprocesseur et microcontrôleur  
*Essentiellement calcul performant ou micro-ordinateur complet*

	Micropcesseur	Microcontrôleur
RAM, ROM, EEPROM, périphérique	Externe	Interne
Prix	Faible	Élevé
Vitesse	~ 8–50 MHz	> 1 GHz
Consommation énergétique	Élevée	Faible
Système économie d'énergie	Non	Oui
Taille	Volumineux	Compact
Tâches exécutées	Limitées et simples	Complexes
Architecture	von Neumann	Harvard



# ASSP et FPGA

- Application Specific Standard Product (ASSP) 
  - Version plus générale des ASICs, contenant une seule fonction
  - Vise un large marché et peut être produit en masse
  - À l'opposé des ASICs qui combinent plusieurs fonctions
- Prototypage d'une ASIC réalisé à l'aide de FPGA
  - À l'instar de la breadboard qui prototype un PCB

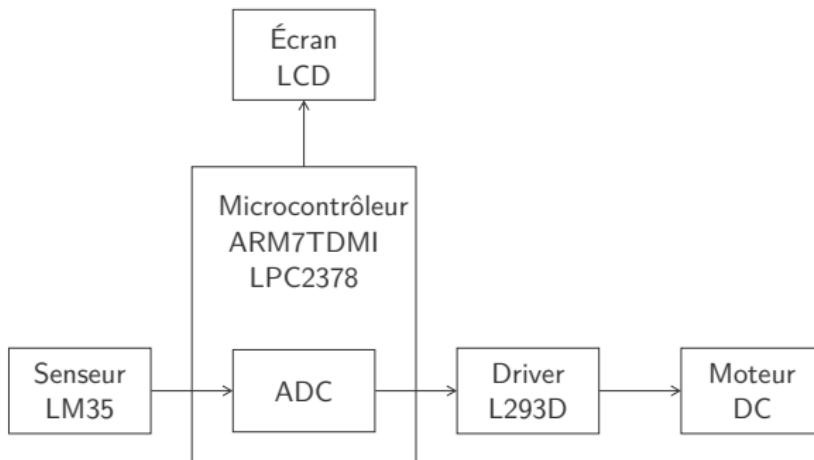
# Architecture matérielle (1)

- **Architecture matérielle** décrit structure système embarqué  
*Plusieurs représentations avec degrés d'abstraction différents*
- Description possible par un **diagramme blocs**
  - Blocs pour composants, unités, sous-systèmes...
  - Liens représentent échange d'information, commande...

# Architecture matérielle (2)

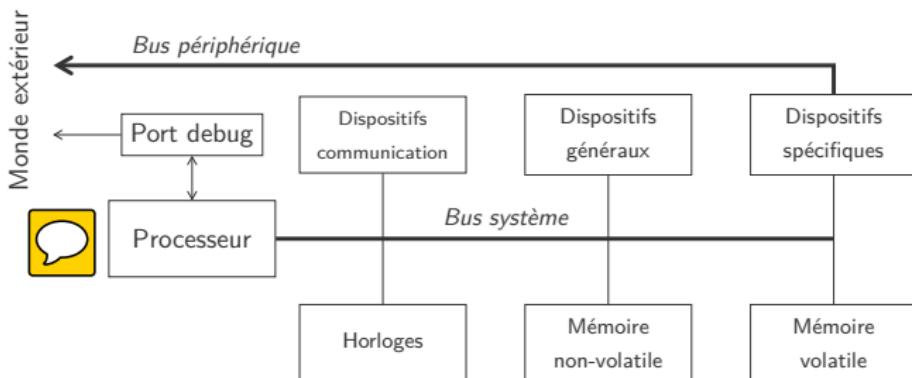
- Exemple d'un ventilateur contrôlé par la température

*Système embarqué construit avec microcontrôleur ARM7TDMI*



# Architecture type

- **Architecture type** d'un système embarqué avec processeur  
*Horloge, mémoire, dispositif général/spécifique/communication*
- Deux ouvertures vers le **monde extérieur**  
*Port debug et bus de périphériques*



# Alimentation (1)

- Alimentation en énergie d'un système embarqué importante

*Adaptateur secteur AC/DC, batterie, hôte, charge pump*

- Plusieurs plages possibles selon les besoins des composants

$5\text{ V} \pm 0.25\text{ V}$ ,  $3.3\text{ V} \pm 0.3\text{ V}$ ,  $2.0\text{ V} \pm 0.2\text{ V}$ ,  $1.5\text{ V} \pm 0.2\text{ V}$



- Construits à partir de **LVC MOS** et **LV TTL**

- Tension inversement proportionnelle délais de propagation
- Passer de  $5\text{ V}$  à  $3.3\text{ V}$  divise dissipation puissance par deux
- Passer à  $3.3\text{ V}$  génèrent moins de chaleur, plus petits packs

# Alimentation (2)



- **Caractéristiques** d'un bon sous-système d'alimentation
  - Tension stable et lisse, et constante pour ADC par exemple
  - Courant en suffisance pour système embarqué et composants
  - Efficace et stable selon conditions (température, circulation air)
- Offrir un **découplage** entre différents composants
  - Horloge et circuit de reset sans interférence radioélectrique
  - Ports d'entrée/sortie dissipent beaucoup de puissance
  - Timers dissipent une puissance constante

# Horloge

- Horloge cadence fonctionnement du processeur

*Rythme le cycle fetch-decode-execute*

- Trois principales technologies utilisées pour des oscillateurs

- Crystal externe plus stable (doit être proche du processeur)
- Résonateur céramique interne (dérive 10min/mois versus 1–5)
- Oscillateur externe en circuit intégré (plus contrôlable)

- Présence de circuit de timer Real-Time Clock (RTC)

*Ordonnanceur de tâche, système temps réel*

# Mémoire

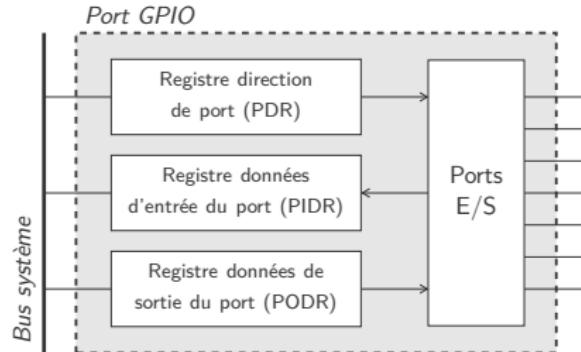
- Plusieurs **mémoires** permettant de stocker de l'information  
*Permanente ou volatile, interne ou externe*
- **Trois principaux types** de mémoires utilisés
  - *Read-Only Memory (ROM)*  
*Permanente en lecture seule, programme exécuté*
  - *Random-Access Memory (RAM)*  
*Volatile, travail en cours, variable, tampon*
  - *Electrically Erasable Programmable ROM (EEPROM)*  
*Permanente modifiable, copie instructions, calcul rapide*

# Entrée/sortie

- Communication avec **dispositifs externes** identifiés par port
  - Port d'entrée : senseur, bouton, circuit transducteur...
  - Port de sortie : LED, écran LCD...
- **Port** est un dispositif physique pour transiter des données
  - Recevoir données de périphérique, processeur, contrôleur
  - Envoyer données vers l'extérieur
  - Connecté vers le processeur grâce à un bus

# Port GPIO

- Port **General Purpose Input/Output** (GPIO)
  - Connexion directe du processeur avec le monde extérieur
  - Deux valeurs de tension pour chaque pin GPIO (valeur binaire)
  - Utilisé comme entrée (bouton) ou comme sortie (LED)
- Fonctionnement du port GPIO grâce à des **registres**



# Communication

- Transmission des informations en **série ou parallèle**
  - Parallèle historiquement plus rapide car plusieurs bits à la fois
  - Série plus léger niveau hardware (câble, émetteur/récepteur)
  - Parallèle sur distances courtes, série longue distance
- **Rythme** de la communication et messages échangés
  - Synchrone suit une horloge, à partager entre les bouts
  - Asynchrone transmet des balises, surcharge de communication
  - Choix à faire selon régularité des messages
- Plusieurs **types de bus** : RS-232, I<sup>2</sup>C, SPI, CAN, USB, PCI...

# Support analogique

- **Système hybride**, discret/numérique et continu/analogique
  - Lire une entrée analogique comme un son d'un micro
  - Produire une sortie analogique pour contrôler un moteur
- Génération d'un signal analogique avec **DAC (PWM)**  
*Construction d'un signal continu à partir d'états discrets*

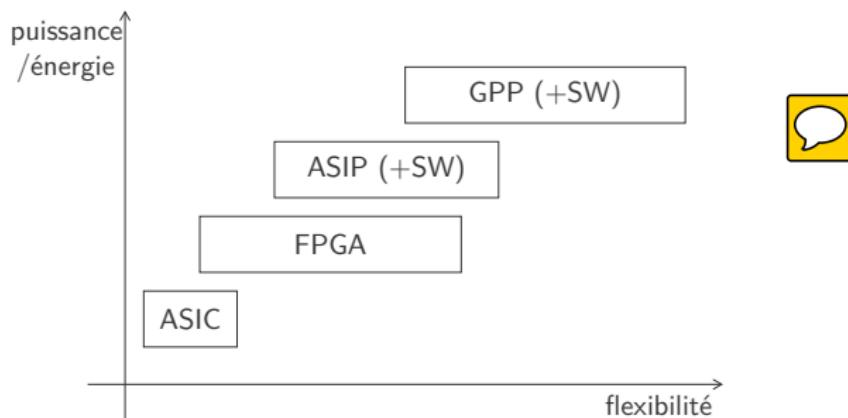
- Acquisition d'un signal analogique avec **ADC**  
*Discrétisation d'un signal continu en séquence binaire*

# Choix technologique

- Démarrage avec **technologie générique** moins couteux
  - Implémentation des fonctions désirées en software
  - Performances moins bonnes et nécessité d'optimisation
- Remplacement **processeur-software** par un ASIC
  - Implémentation des fonctions au niveau hardware
  - Solution plus chère et avec un grand temps de développement

# Flexibilité versus puissance/énergie

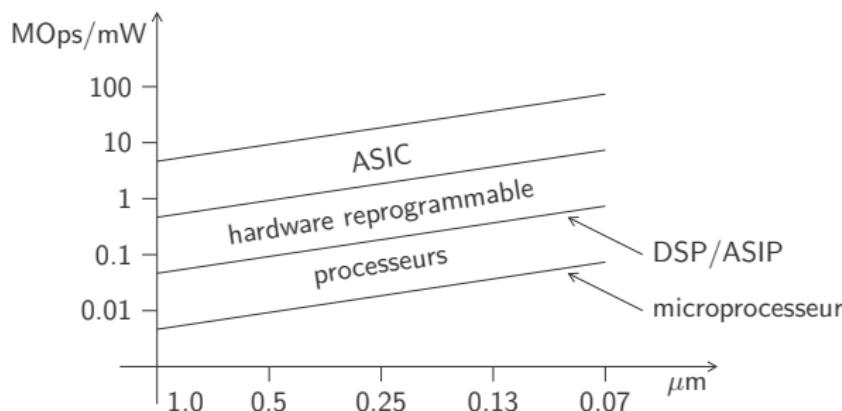
- Augmentation de la flexibilité coûte en performance/énergie  
*Fonction software coûte 10 fois plus que équivalent hardware*



# Taille versus opérations/Watt

- Plus d'opérations par Watt en **diminuant les tailles** des CI

*Hardware câblé pur offre un gain d'un ordre de grandeur supérieur*



*"Computer programming is an art, because it applies accumulated knowledge to the world, because it requires skill and ingenuity, and especially because it produces objects of beauty."*

Donald Knuth 1974

**Aspect software**

# Software

- Software partie importante système embarqué, son « *brain* »

*Tout du moins pour ceux qui sont basés sur un processeur*



- Software développé pour un système embarqué **pas portable**

*Car le lien avec le hardware peut être très fort*

- Développement différent de celui d'un software « *classique* »

*Pas facile de directement coder sur le système embarqué*

# Système réactif

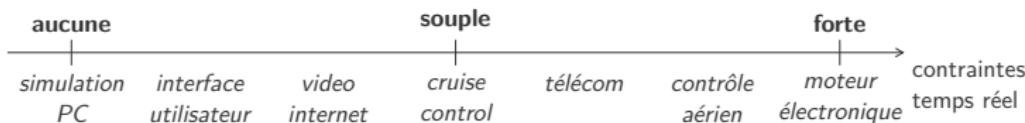
- Comportement exécuté en **réaction à des évènements**  
*L'environnement produit des évènements sur le système embarqué*
- **Interaction continue** à un rythme imposé par l'environnement  
*"A reactive system is one that is in continual interaction with its environment and executes at a pace determined by that environment."*
- Deux **types d'évènements** possibles sur le système embarqué  
*Périodique (boucle de contrôle) et non périodique (bouton)*

# Temps réel

- Système temps réel doit réagir **endéans une deadline**

*Pas une question de vitesse d'exécution, mais respect de deadline*

- Deux niveaux** de contraintes selon conséquence de non respect
  - Temps réel souple, pas grave si une deadline est manquée
  - Temps réel strict, conséquences très graves



# Polling et interruption

- Software doit être avertit de la **disponibilité de données**  
*Par des bus, périphériques, etc.*
- Deux mécanismes principaux pour avertir le CPU
  - **Polling**, CPU va vérifier régulièrement dans un registre d'état
  - **Mécanisme d'interruption** du CPU par le périphérique 
- Choix très critique dans le cas de **systèmes temps réel**  
*On ne peut pas toujours se permettre d'attendre un périphérique*

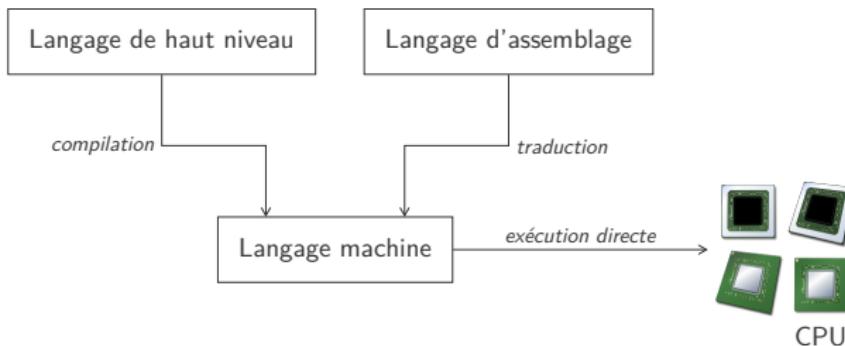
# Programmation

- Software se présente sous la forme de **code machine**

*Image ROM du software d'un système embarqué*

- Utilisation des opérations du **jeu d'instructions** du processeur

*Placement en mémoire en fonction de ses caractéristiques*



# Niveau de programmation

- Directement en **code machine**

*Beaucoup de temps pour comprendre jeu d'instructions et coder*

- **Langage d'assemblage** traduit en code machine

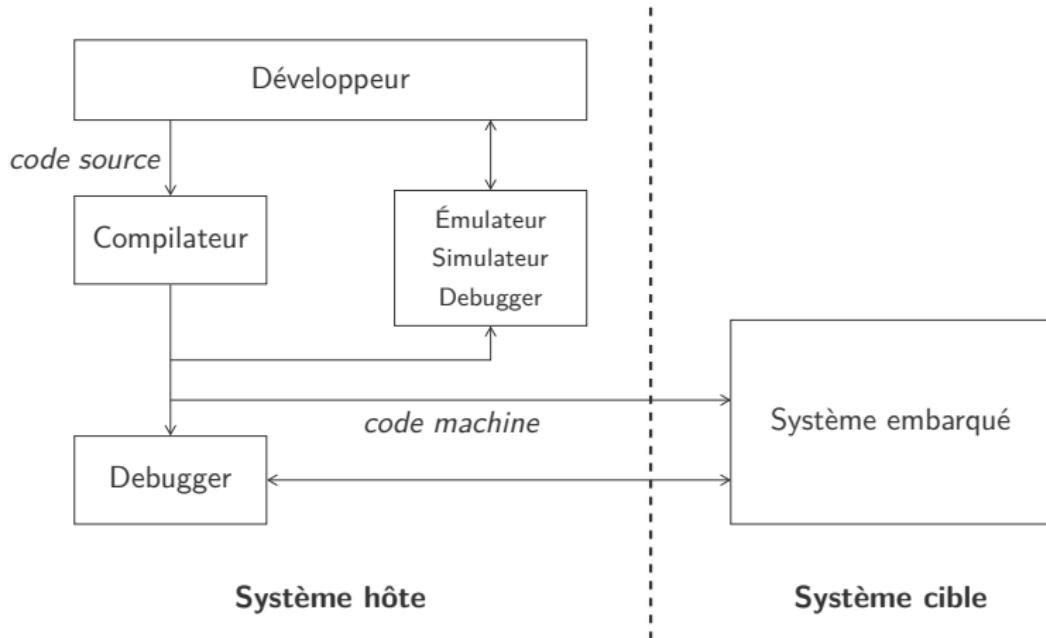
*Raccourci textuels pour l'être humain, et routines*

- **Langage de haut niveau** compilé en code machine



*Abstraction plus proche de la façon de penser du programmeur*

# Développement (1)



# Développement (2)

- Pas toujours possible de développement **sur système embarqué**

*Utilisation d'un système hôte pour programmer une cible*

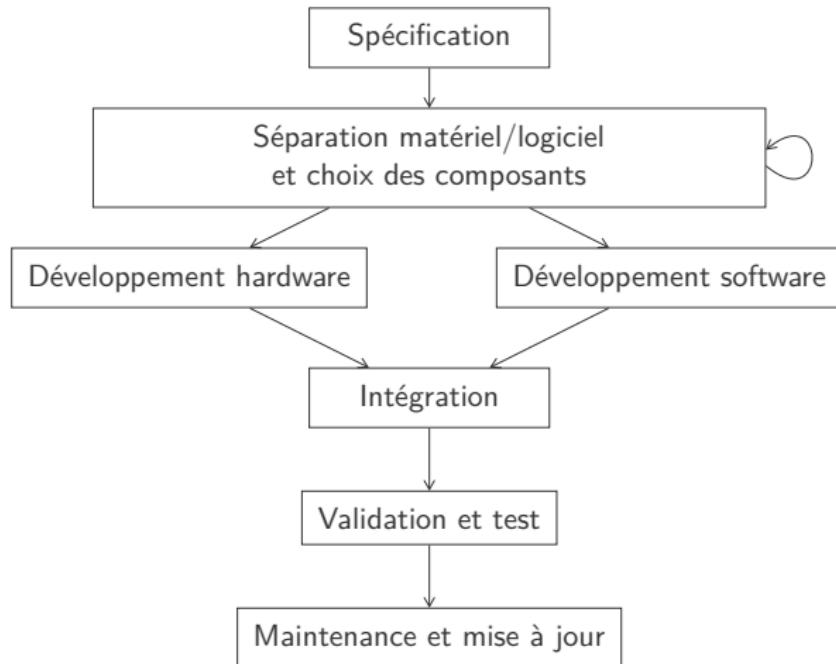
- **Deux options** avec un code source compilé en code machine

- Envoi vers le système cible pour exécution directe
  - Exécution sur le système hôte avec émulateur ou simulateur

- Possibilité de **debugger** directement depuis le système cible

*Avec les standards JTAG, par exemple*

# Développement (3)



# Développement (4)

- Pas une unique **procédure de développement**  
*Mais coordination entre les aspects hardware et software*
- **Spécification** puis séparation matériel/software  
*Plusieurs itérations et point de non retour*
- **Développement parallèle** software/hardware, mise en commun  
*Intégration, validation et tests, maintenance et mise à jour*

# JTAG

- Programmation et debugging via **interface JTAG** 
  - Accès aux fonctions de debug et d'émulation des processeurs
  - Accès aux fonctions de programmation des FPGAs et CPLDs
- **Protocole de communication** Joint Test Action Group (JTAG)
  - Méthode de test de bas niveau sans accès physique
  - Accès aux pins E/S via la technique du Boundary-Scan

# Système d'exploitation

- Manipulation de hardware à l'aide d'un **système d'exploitation**
  - Couche logicielle par dessus un hardware
  - Relais entre le hardware et les applications utilisateurs
  - Offre des services aux applications pour utiliser le hardware
- **Logiciel de bas niveau** s'exécutant directement sur le hardware
  - Exécuté avec des privilèges et droits directs sur le hardware
  - Dépend largement du hardware, et est plus ou moins portable

# Vue ingénieur et client



# Disparition de l'ordinateur

- L'ordinateur « traditionnels » **disparaît** lentement

*Relativement à l'explosion des systèmes embarqués*
- Trois grandes tendances d'**informatique omniprésente**
  - **Ubiquitous computing**

*But à long terme d'"information anytime, everywhere"*
  - **Pervasive computing**

*Exploitation de technologies déjà existantes*
  - **Ambient intelligence**

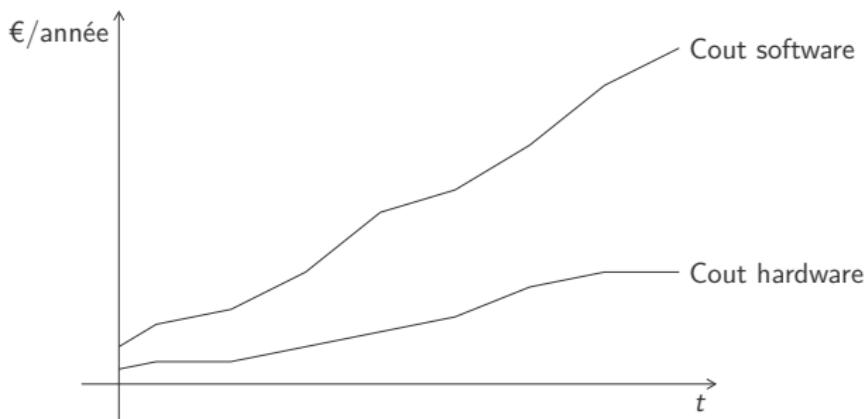
*Communication pour maison future et smart cities*



# Hardware versus software

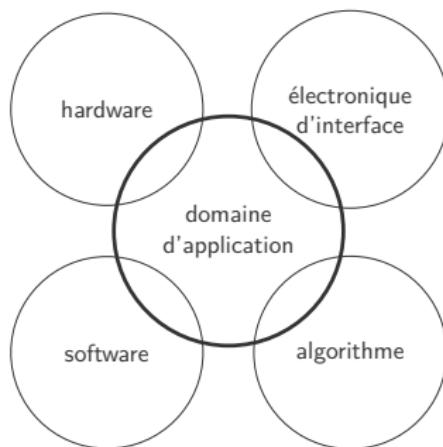
- Hardware stabilisé, usinage grande échelle et **cout récurrent**

*Cout software non récurrent, conception pour chaque application*



# Compétence

- Compétences **transversales**, dépassent frontières des disciplines  
*Importance de comprendre le domaine d'application*



# Compromis (1)

- Choix et compromis entre les différentes disciplines

*Comprendre les contraintes des différents domaines*

- Plusieurs activités menées par l'ingénieur

- Concevoir algorithmes de contrôle, traitement de signal...
- Concevoir microprocesseurs pour des applications embarquées
- Concevoir logiciels (e.g. RTOS) pour systèmes embarqués
- Faire partie d'équipes des domaines d'application
- Développer senseurs/actuateurs (e.g. MEMS)

# Compromis (2)

- **Optimisation** de plusieurs critères outre la vitesse d'exécution  
*Processeur, mais tout ce qui gravite autour*
- Trois **catégories d'éléments** à prendre en compte
  - **Multi-objectifs**, fiabilité, abordabilité, sécurité, sûreté, évolutivité et mise à l'échelle, ponctualité
  - **Multi-discipline**, hardware et software, algorithme de contrôle, interface électronique, domaine d'application
  - **Cycle de vie**, exigences, conception, usinage, déploiement, logistique et retraite

# Client

- Diminution des couts pour le client, plus petits et moins cher  
*Appareil spécialisé pour ce qu'il désire*
- Nouvelles fonctionnalités part intégration dans d'autres objets  
*Fonctions pas possibles avec appareils plus « traditionnels »*
- Améliorer les performances du système  
*Consommation plus contrôlée, meilleure batterie, etc.*

# Crédits

- <https://www.flickr.com/photos/splorp/8165771383>
- <https://www.flickr.com/photos/tamaki/7568407>
- [https://www.flickr.com/photos/code\\_martial/2445612018](https://www.flickr.com/photos/code_martial/2445612018)
- <https://www.flickr.com/photos/sozialhelden/9397521833>
- <https://www.flickr.com/photos/nicknormal/30419435085>
- <https://www.flickr.com/photos/gcollazo/65244956>
- <https://www.flickr.com/photos/davegray/6373321029>
- <https://www.flickr.com/photos/npobre/4484863605>
- [https://en.wikipedia.org/wiki/File:Intel\\_D4004.jpg](https://en.wikipedia.org/wiki/File:Intel_D4004.jpg)
- [https://en.wikipedia.org/wiki/File:Unicom\\_141P\\_Calculator\\_3.jpg](https://en.wikipedia.org/wiki/File:Unicom_141P_Calculator_3.jpg)
- [https://3.bp.blogspot.com/-Pv1QyUVIX20/Vz\\_iPo-qnQI/AAAAAAAACq8/mgLCTGT5M3QeM4nHZZBeiZp78GmuTWYowCLcB/s1600/tpu.png](https://3.bp.blogspot.com/-Pv1QyUVIX20/Vz_iPo-qnQI/AAAAAAAACq8/mgLCTGT5M3QeM4nHZZBeiZp78GmuTWYowCLcB/s1600/tpu.png)
- <https://en.bitcoin.it/wiki/File:BM1387.jpg>
- <https://www.flickr.com/photos/bradmontgomery/8007012137>
- <https://openclipart.org/detail/100267/cpu-central-processing-unit>
- <https://www.flickr.com/photos/128474566@N03/15504868300>