

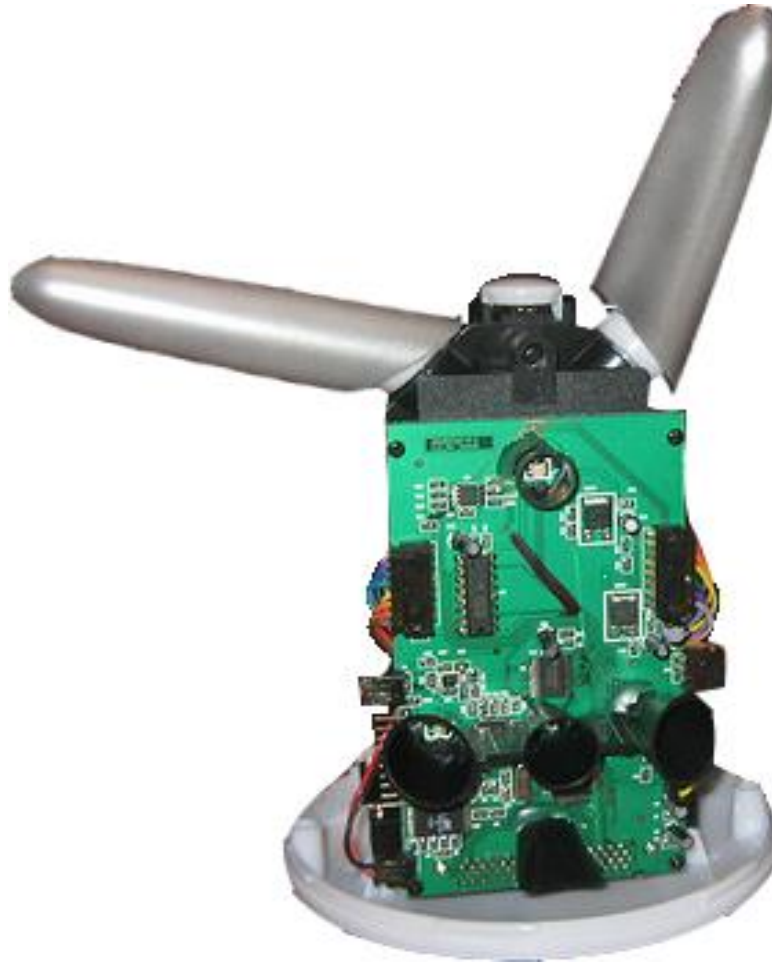


LES OBJETS COMMUNICANTS

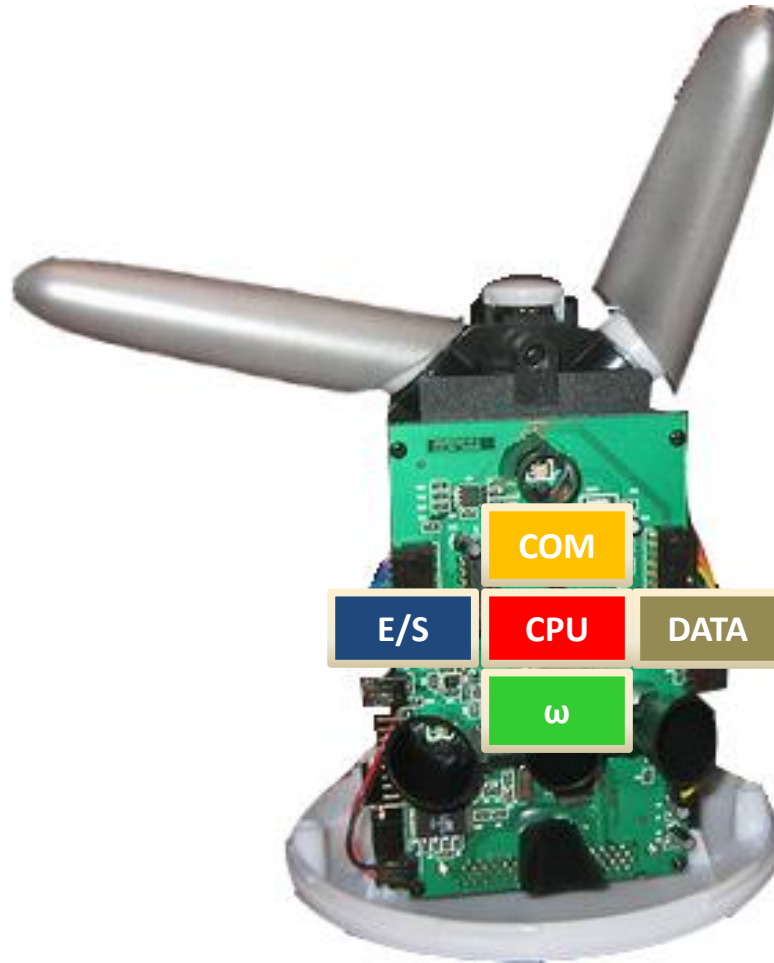
Jean-Paul Jamont, Lionel Médini, Michaël Mrissa



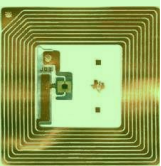
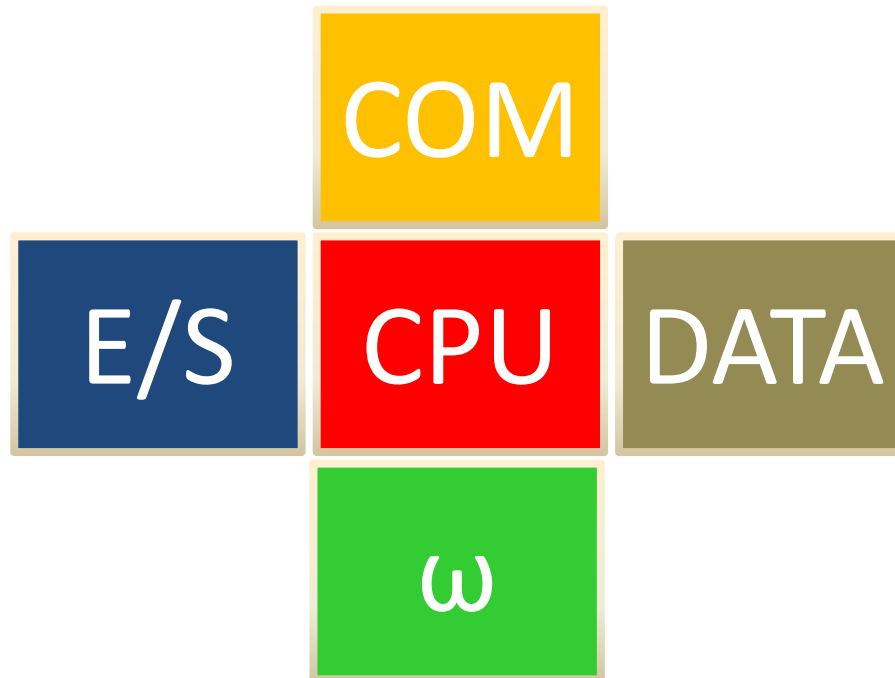
COMMENT FONCTIONNENT LES OBJETS COMMUNICANTS?



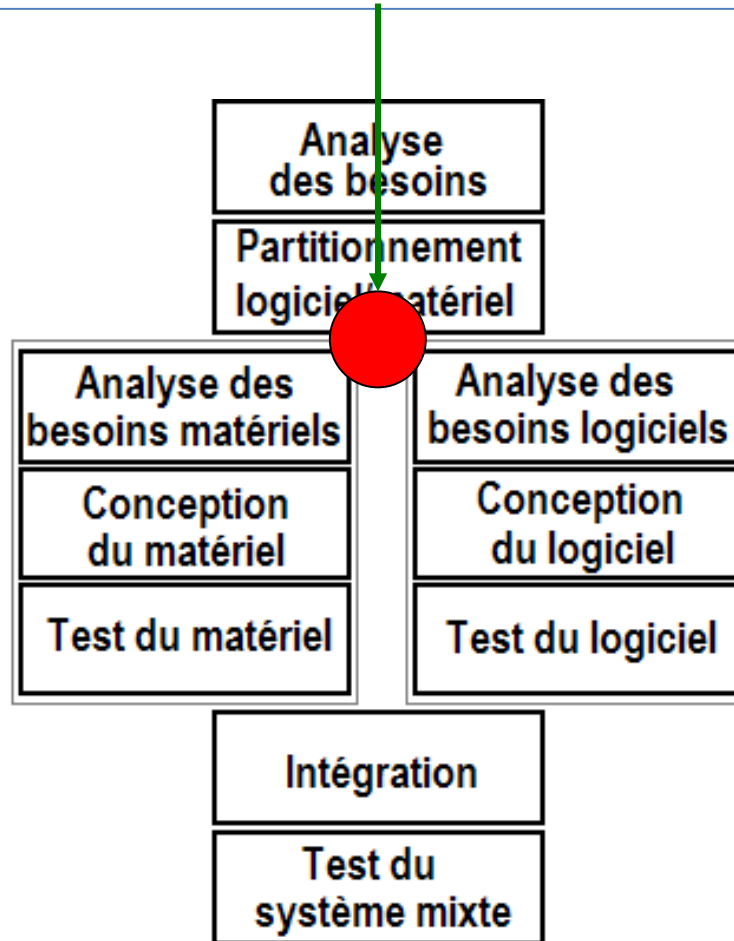
COMMENT FONCTIONNENT LES OBJETS COMMUNICANTS?



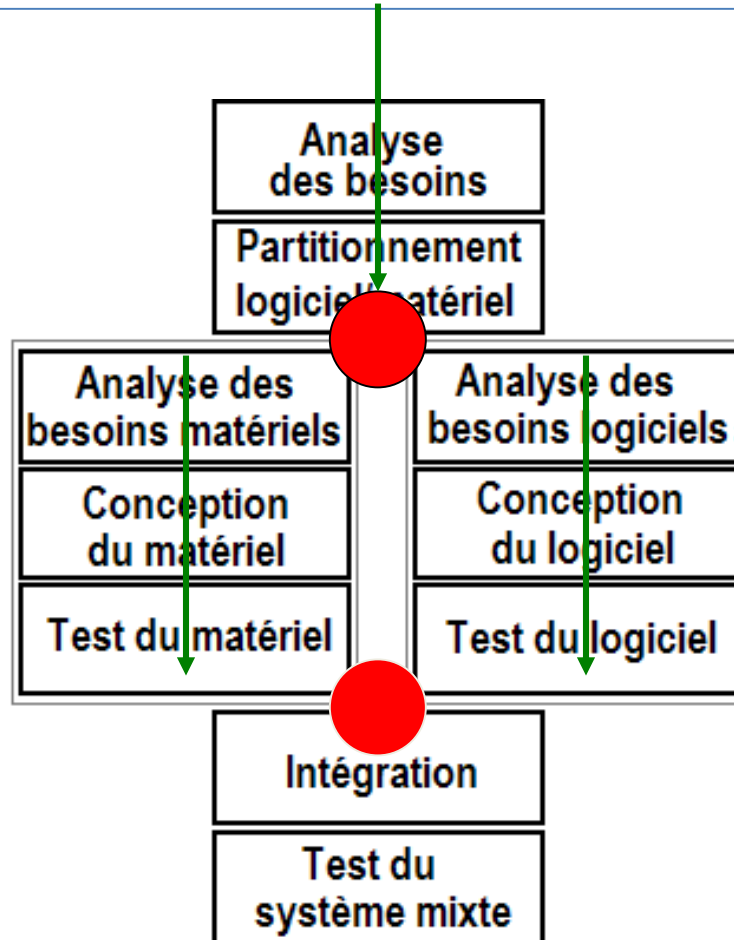
COMMENT CRÉER SES PROPRES OBJETS COMMUNICANTS?



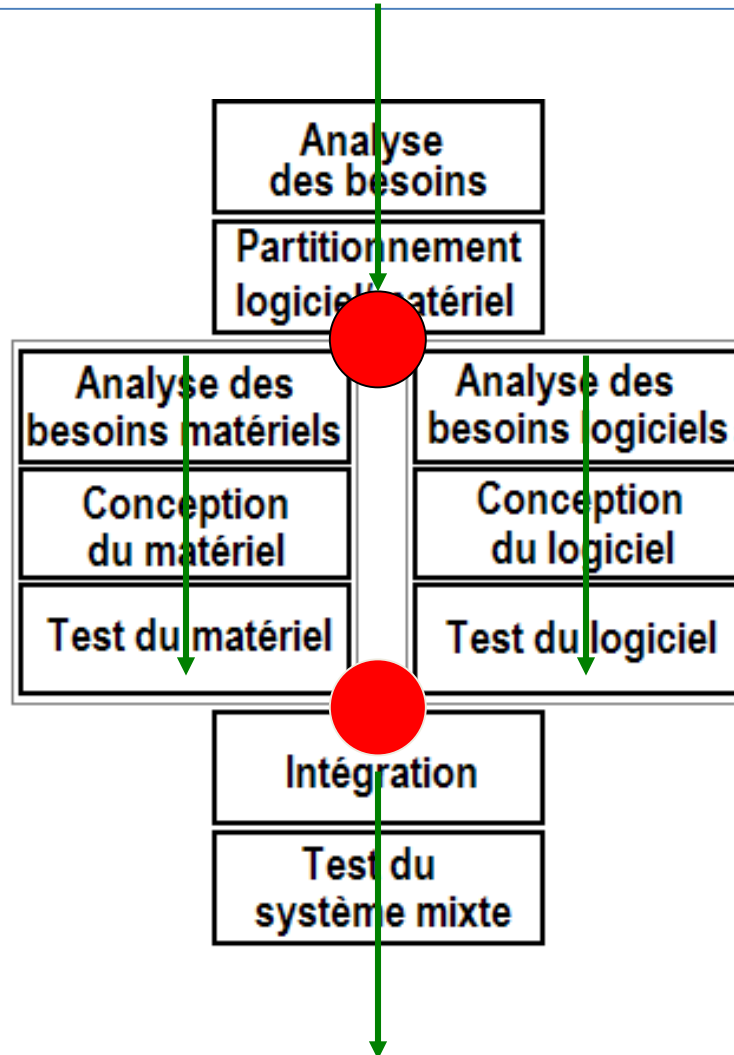
1. QUELLE DÉMARCHE?



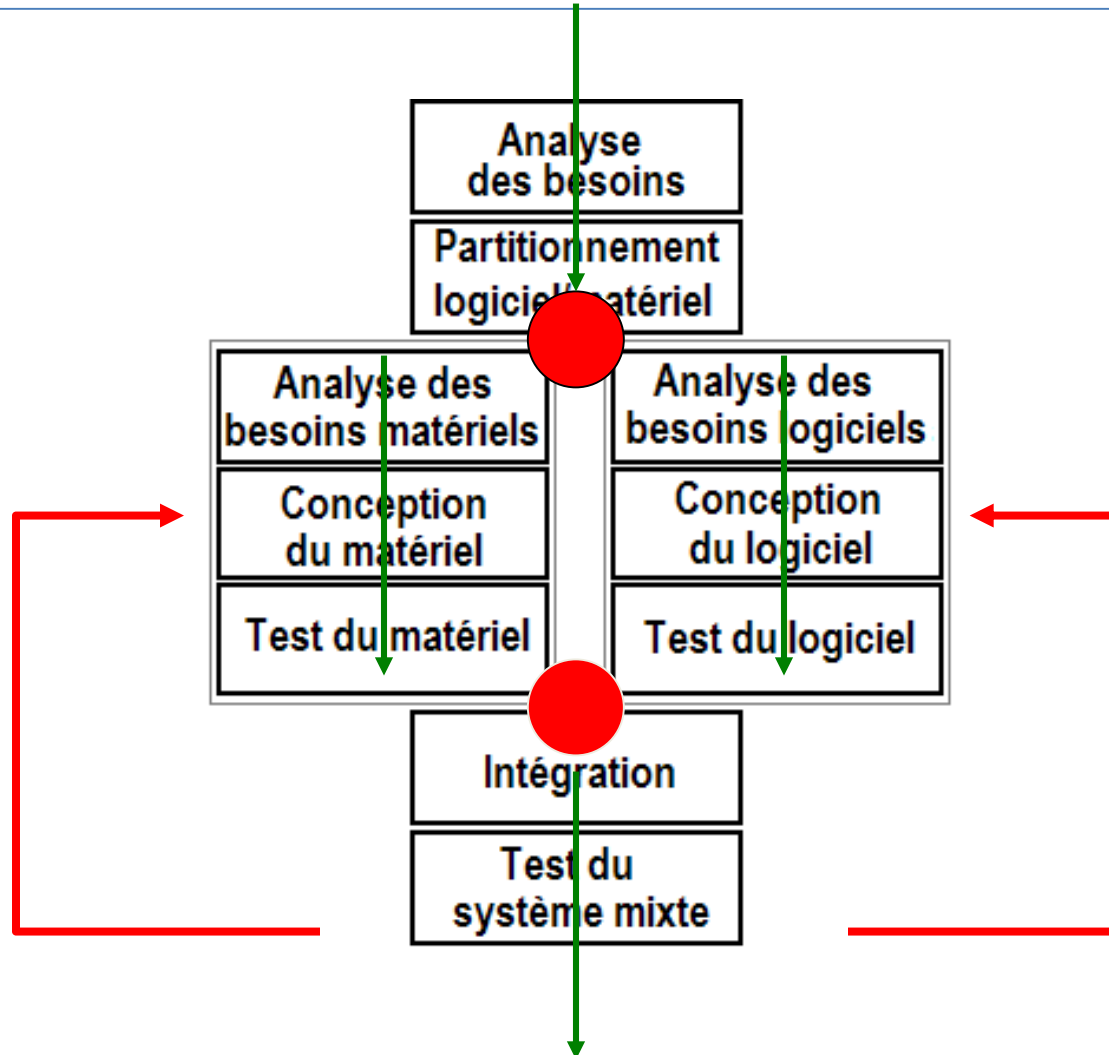
1. QUELLE DÉMARCHE?



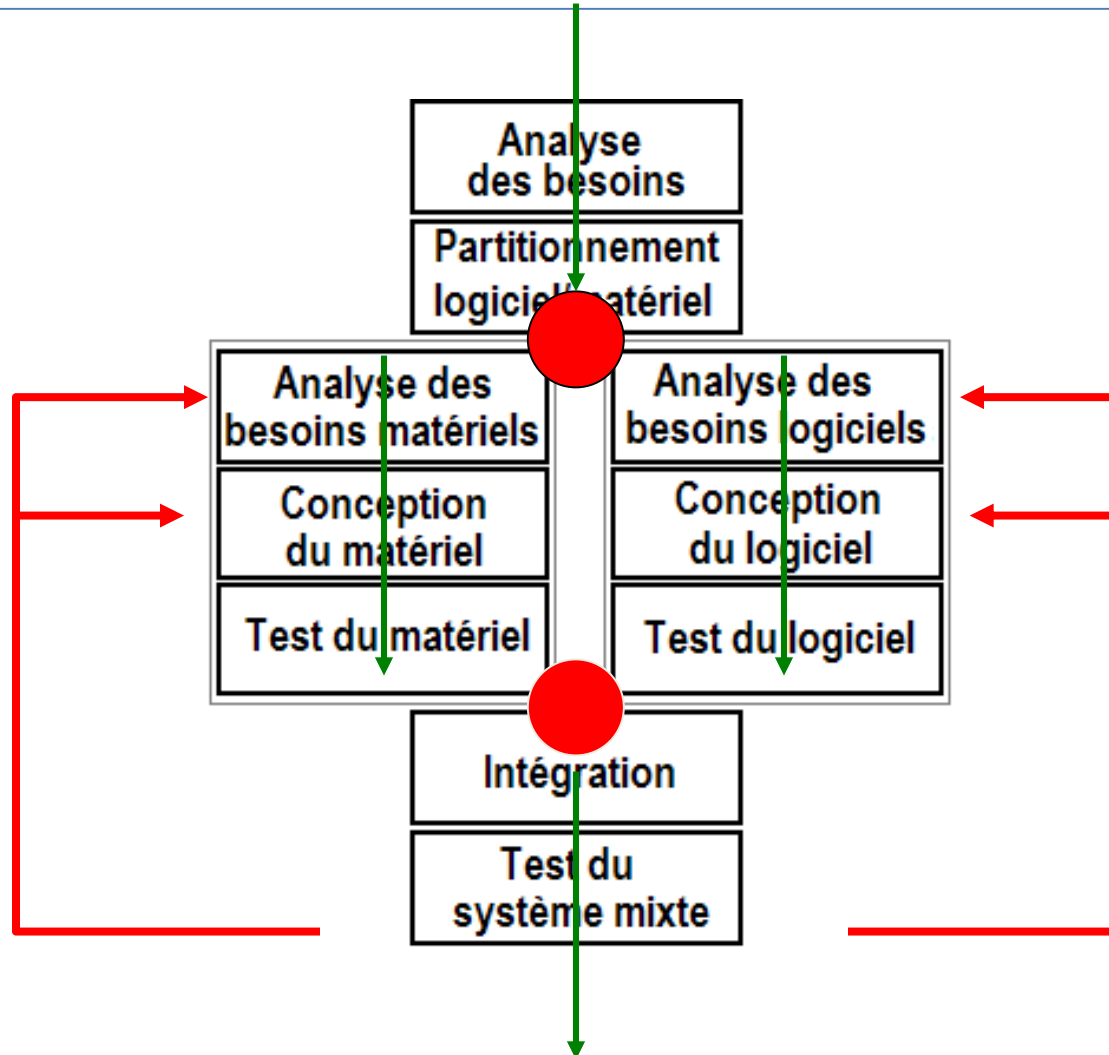
1. QUELLE DÉMARCHE?



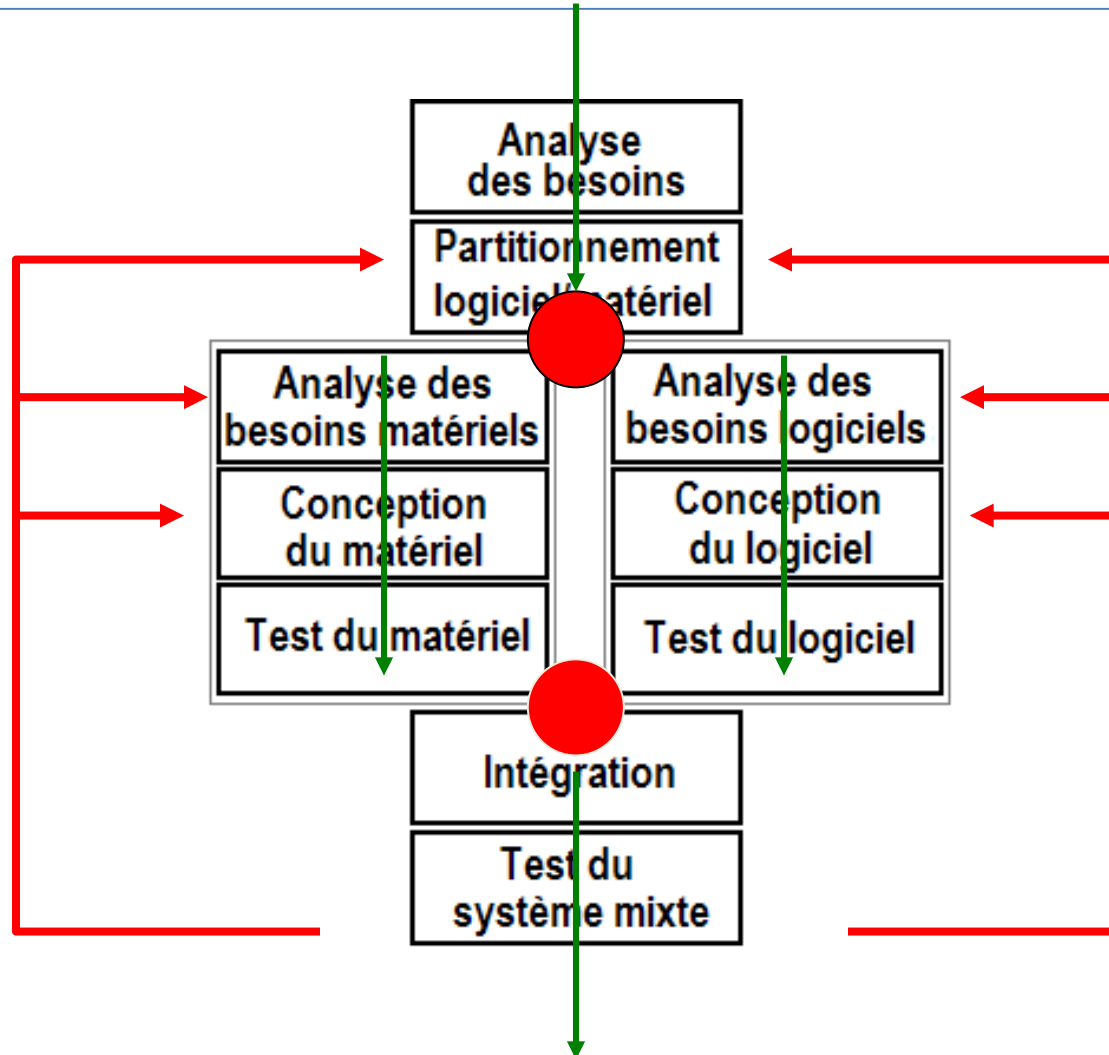
1. QUELLE DÉMARCHE?



1. QUELLE DÉMARCHE?



1. QUELLE DÉMARCHE?



1. QUELLE DÉMARCHE?

71.5% of traditionnal embedded system designs were not within **30%** of pre-design performance expectations. [1]

Changes in specifications

Complexity of the application

Inadequate specifications

Too few developers

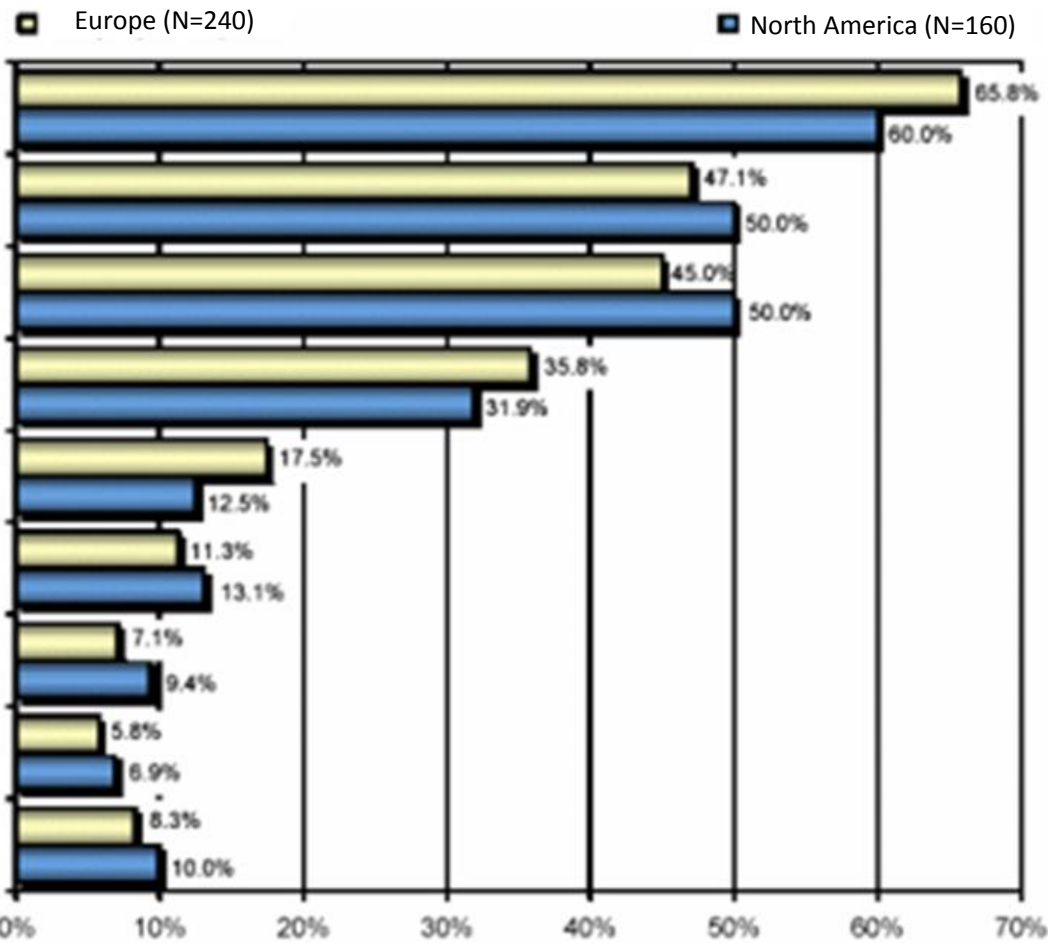
Too few testing personnel

Poor testing tools

Poor developpement tools

Inefficient
production/manufacturing

Others



1. QUELLE DÉMARCHE?

Recueil
des besoins

Analyse

Conception
générique

Partitionnement
logiciel/matériel

Test

1. QUELLE DÉMARCHE?

**SOLUTION MATERIELLE
CARTE A BASE D'EPLD/FPGA**



**SOLUTION LOGICIELLE
CARTE A MICROPROSSEUR/MICROCONTROLEUR**



SYNTHESE MATERIELLE

```

1  -- COMPONENT Gestion role
2  -- Version 0.95a
3  -- WARNING: Default parameter use = Synchronous version =
4  -- State translation :
5  -- S1 by vector "000"
6  -- S2 by vector "001"
7  -- S3 by vector "010"
8  -- S4 by vector "011"
9  -- S5 by vector "100"
10
11 library ieee;
12 use ieee.std_logic_1164.all;
13 use work.std_arith.all;
14
15 -- Component description
16 entity GestionRole is port(
17     reset : in std_logic;
18     NVR : in std_logic_vector(7 downto 0);
19     role : in out std_logic_vector(1 downto 0);
20     -- Call to function "election"
21     electionIn : in std_logic_bit;
22     electionOut : in std_logic_bit;
23 end GestionRole;
24
25 -- Component architecture
26 architecture arch_GestionRole is
27     signal pastState:std_logic_vector(2 downto 0);
28     signal state:std_logic_vector(2 downto 0);
29     signal futureState:std_logic_vector(2 downto 0);
30 begin
31     exec:process(clk)
32     begin
33         if(clk'event and clk='1') then
34             if(reset='1') then
35                 state<="000";
36             else
37                 case state is
38                     -- State S1
39                     when "000" =>
40                         if (NVR="00000001" then
41                             futurState = "001";

```

SYNTHESE LOGICIELLE

```

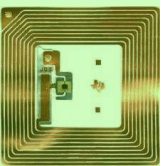
6 String futurState = "";
7
8
9 public void exec(){
10 //FIRST STATE ENTRY
11     for(;;){
12         switch(presentState){
13             case(S1):
14                 if(NVR == 1)
15                     futurState = "S2";
16
17                 if(NVR == 2)
18                     futurState = "S3";
19
20 //ACTION
21         if (presentState != futureState)
22             //EXIT
23         else if (presentState != pastState)
24             //ENTRY
25         else
26             role = awuncun;//STATE
27             break;
28         case(S2):
29             if(NVR==1)
30                 futurState = "D";

```


2. QUELS SONT MES BESOINS?

Pour quel type d'application l'objet que je dois construire va me servir?

1. Calcul généraliste
2. Contrôle de systèmes
3. Traitement du signal
4. Réseaux et communications





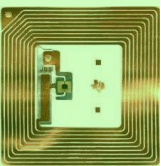
2. QUELS SONT MES BESOINS?

Quels sont mes **besoins** / mes **contraintes**?

- Puissance de **calcul**
- Capacité de **communication**
- Capacité de **stockage**
- Consommation d'**énergie**
- Temps de réponse
- Tolérances aux **pannes**
- Durée de vie
- Testabilité/débogage
- Nombre d'unités produites

3. QUELS CHOIX POUR LA PARTIE MATÉRIELLE?

1. PDA / Mini-PC / PC Industriels
2. Une carte de développement **existante**
3. From **scratch**



3. QUELS CHOIX POUR LA PARTIE MATÉRIELLE?

- PDA / Mini-PC / PC Industriels



Modèle: Samsung Galaxy S4GT-I9505

OS: Android 4.2.2 Jelly Bean

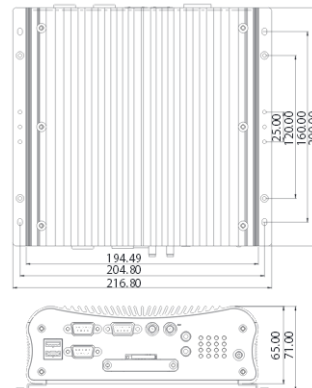
CPU,GPU: 4 cœurs Qualcomm Snapdragon 600 (ARMv7) à 1,9 GHz, Qualcomm Adreno 320

Mémoire vive: 2 Go de RAM LPDDR3

Stockage: 32 Go de mémoire flash intégrée

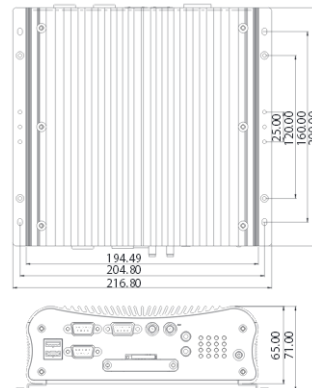
Connectique: HDMI, prise casque jack 3,5 mm, port micro USB 2.0 compatible MHL 2.0, WiFi 802.11 a/b/g/n/ac (HT80), NFC, Bluetooth 4.0 (LE), LED IR3

Capteurs: de proximité, de lumière, IR, thermomètre, hygromètre, baromètre, accéléromètre et gyroscope à 3 axes, magnétomètre



3. QUELS CHOIX POUR LA PARTIE MATÉRIELLE?

- PDA / Mini-PC / PC Industriels



Modèle: Android Mini PC MK802

OS: Android 4.2.2 Jelly Bean ou Ubuntu ou PicUntu

CPU,GPU: Cortex-A9 à 1.6 GHz, 400 MHz Mali GPU

Mémoire vive: 2 Go de RAM LPDDR3

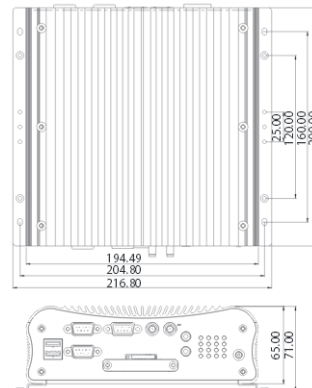
Stockage: 8 Go de mémoire flash intégrée

Connectique: HDMI, micro-USB 2.0, USB 2.0, microSD slot, alim. via micro-USB OTG, Wi-Fi 802.11 b/g/n, Bluetooth,

Capteurs:

3. QUELS CHOIX POUR LA PARTIE MATÉRIELLE?

- PDA / Mini-PC / PC Industriels



Modèle: PC industriel fanless NISE2200

OS: Linux, Windows ...

CPU,GPU: Intel® Atom™ Dual Core D2550 1.86GHz

Mémoire vive: 8Go DDR3 SODIMM

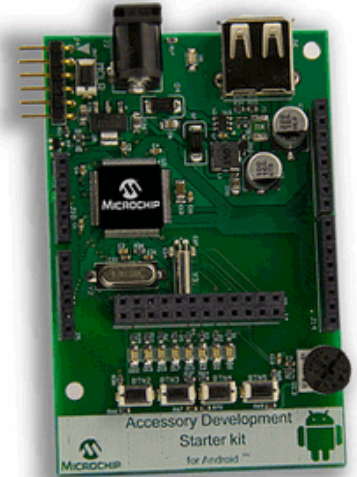
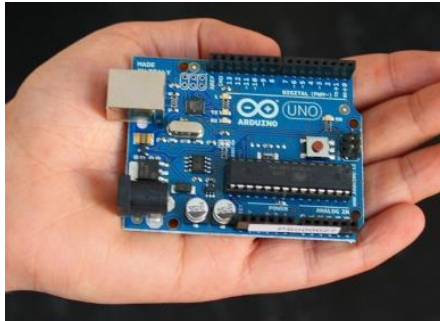
Stockage: Disque dur 2.5" SATA

Connectique: 6 ports USB2.0; 1 emplacement CFast; 1 emplacement carte SIM, 2 ports RS-232/422/485 isolés, 2 ports Ethernet Intel® 82574IT Gb, 1 port DB15 E/S , WiFi 802.11 a/b/g/n/ac or 3.5G (auto detected modules), Support 9-36V DV input, Audio Jack

Capteurs:

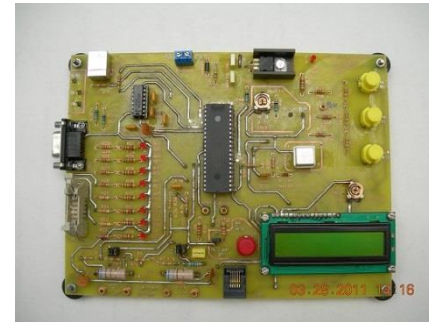
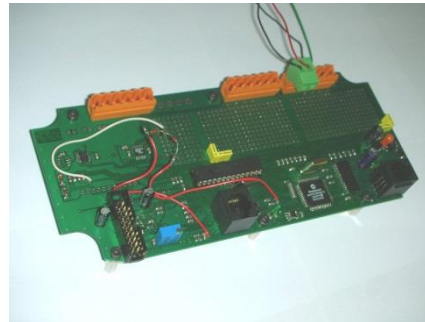
3. QUELS CHOIX POUR LA PARTIE MATÉRIELLE?

- Une carte de développement **existante**

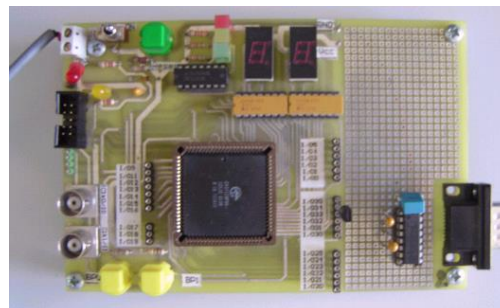


3. QUELS CHOIX POUR LA PARTIE MATÉRIELLE?

- From **scratch**
 - Architecture centrée **microcontrôleur**

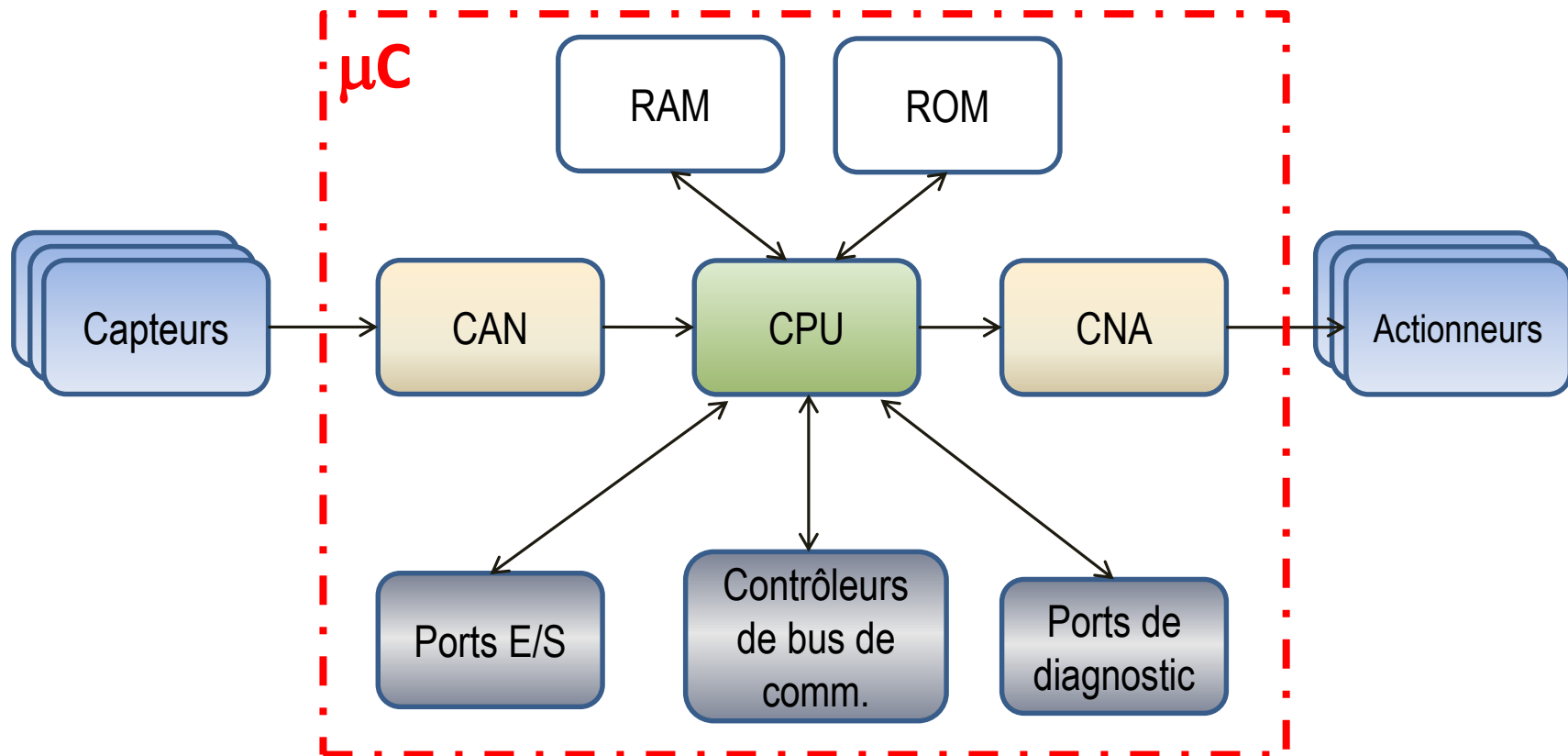


- Architecture centrée **FPGA/EPLD/ASIC?**



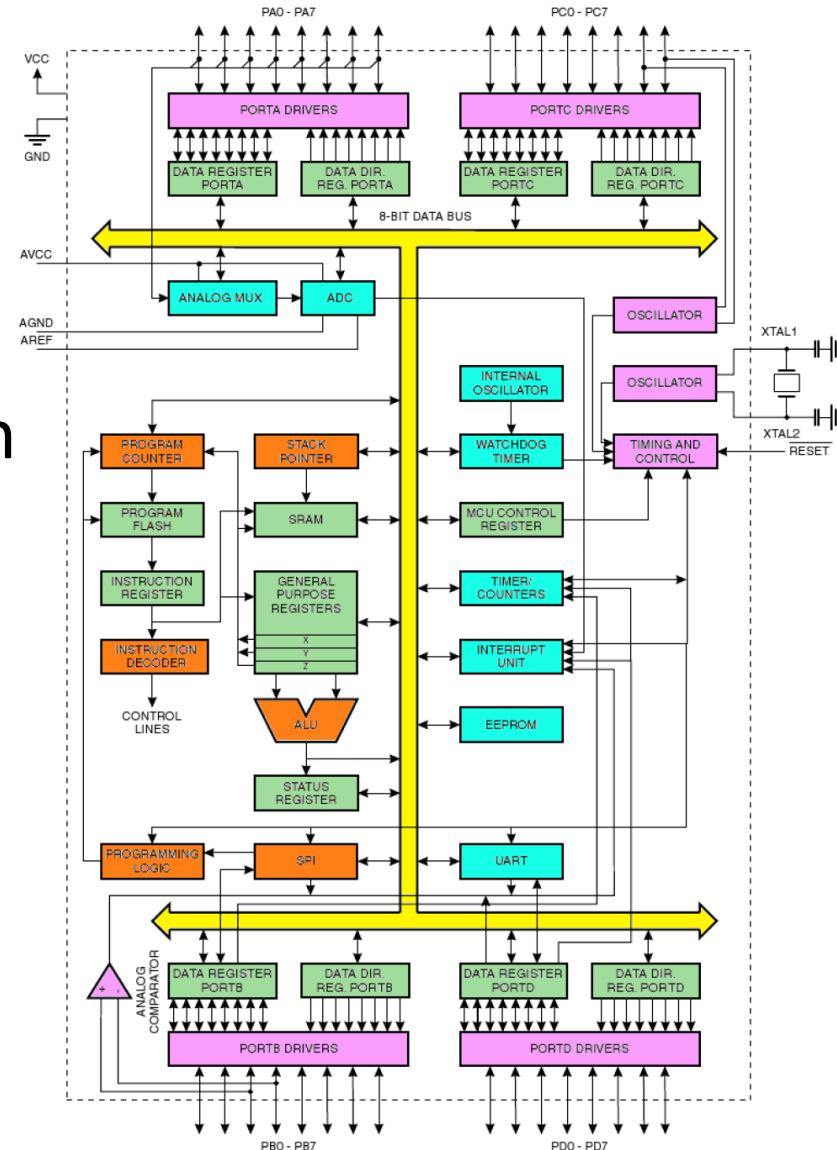
3. QUELS CHOIX POUR LA PARTIE MATÉRIELLE?

- Architecture d'objet communicant basée sur un microprocesseur ou un microcontrôleur



3. QUELS CHOIX POUR LA PARTIE MATÉRIELLE?

- Processeur
- Mémoire
- Périphériques
- Bus de communication
- Entrées/Sorties



D'après Julien DeAntoni

-
- The diagram illustrates the Airbus vision for RFID-enabled business processes. It shows a supply chain flow from Suppliers to Customers. Key components include:
- Suppliers:** Represented by icons of a factory and a truck.
 - Production:** A central factory icon with a large 'Z' shape.
 - Customers:** Represented by an icon of a person.
 - RFID Technology:** A central box labeled 'RFID' with a tag icon.
 - Business Processes:** A flowchart showing the sequence of events: 'Supplier sends goods to Production', 'Production sends goods to Customer', and 'Customer sends goods to Supplier'.
 - RFID-enabled Processes:** A flowchart showing the sequence of events: 'Supplier sends goods to Production', 'Production sends goods to Customer', and 'Customer sends goods to Supplier'.
 - RFID-enabled Business Processes:** A flowchart showing the sequence of events: 'Supplier sends goods to Production', 'Production sends goods to Customer', and 'Customer sends goods to Supplier'.

[illegible]

3. QUELS CHOIX POUR LA PARTIE MATÉRIELLE?

Utiliser les « Product Selector Guide »



Microchip Advanced Part Selector

Microchip Technology Inc.
2355 West Chandler Blvd.
Chandler, AZ 85224-1699
Ph: 480-792-7200 Fax: 480-899-9210
www.microchip.com

The information contained in this media regarding device specification or pricing and the like is intended through suggestion only and may be superseded by updates.

The content of the database is representative of various manufacturers' product specifications and contains the latest product specification information available to Microchip Technology Inc. at the time of MAPS last update. The product specifications in MAPS are subject to change without notice.

It is your responsibility to ensure that your product selection meets with your system specifications.

All trademarks mentioned herein are property of their respective companies.

Where would you like to start?

Analog Interface Memory Microcontroller Wireless OK

Rechercher : jamont

3. QUELS CHOIX POUR LA PARTIE MATÉRIELLE?

Microcontroller - Mozilla Firefox

Fichier Édition Affichage Historique Marque-pages Outils ?

Microcontroller

Page précédente Page suivante

www.microchip.com/maps/microcontroller.aspx

Actualiser Arrêter

microchip sele

Téléchargements Accueil Firebug

Les plus visités À la une Conferences acceptin... http://ade52-upmf.gr... Zimbra: Réception (18)

Parameter Search

Match ALL (AND) Match ANY

Expand

Family: -All- Prefix: -All-

8-bit 16-bit 32-bit

Program Memory: From Thru

P.Memory(Kbytes): -All- P.Memory(KWords): -All-

RAM(Bytes): -All- EEPROM(Bytes): -All-

I/O Pins: -All- Max CPU Speed: -All- Int. OSC: -All-

Comparator: -All- A/D Ch.: -All- A/D Bits: -All- D/A Ch.: -All-

UART Ch.: -All- SPI: -All- I²C™: -All-

CAN Ch.: -All- USB Ch.: -All-

Input Capture: -All- PWM Ch.: -All-

MtrCtrl PWM Ch.: -All- SMPS PWM Ch.: -All-

Search Results

819 MCHP Parts found

Reset Search

Sort Results by: Memory Size

To add MCHP part to side-by-side: Add dsPIC33EP512MU814

Go to side-by-side

dsPIC33EP512MU814 In Production

Specifications Dev Tools Technical Docs Budgetary Pricing

P.Memory (Kbytes) 512 Flash

P.Memory (KWords) 170

Self-Write Flash Yes

RAM (Bytes) 52K

EEPROM (Bytes) 0

DMA Ram 4096

Auxiliary/Boot Flash 24

I/O Pins 122

Max CPU Speed 140 MHz (70 MIPS)

Internal OSC 7.37 MHz, 32.768 kHz

Code Guard™ Security Basic

System Mgmt Features BDR, None, POR, WDT, WUR, 15-DMA, nanoWatt (Low Sleep, Fast Wake, Power Modes)

Analog Peripherals 3-Comparators, Bandgap - No, OpAmp; 2A/D, 32x12-bit @ 500kps; 1D/A, 0x4-bit @ kpsps

Digital Comm. Peripherals 4-UART, 4-SPI, 2-I2C, AC97, I2S, PPS

Connectivity 1-Full Speed (Device, Host, OTG)-USB 2.0 OTG, 2-CAN, None, LIN, IrDA

Capture/Compare PWM Peripherals 16-Output Comp. & Std. PWM, 16-bitPWM, 16-Input Capture

Digital Timers 9x16-bit, 4x32-bit

Application Peripherals 14-MtrCtrl, PWM, 14-PS_PWM, 2-QEI, PMP, EBI-No

Debug/Development Features JTAG-Boundary Scan, ICSP, ICDdebug - Yes

Package (Pins) LQFP, TQFP (144)

Operating Voltage (3V-3.6V)

Temperature Ranges (-40 to +125)

Min PCB Area (mm2)

Show Other Features

Not sure of complete part number?

Global Part Search

Have internet access?

Selection Tools Hom

Have a suggestion?

Send Email

More Links

Select a part in the search results, press one of the buttons below to view

dsPIC33EP512MU814

MICROCHIP

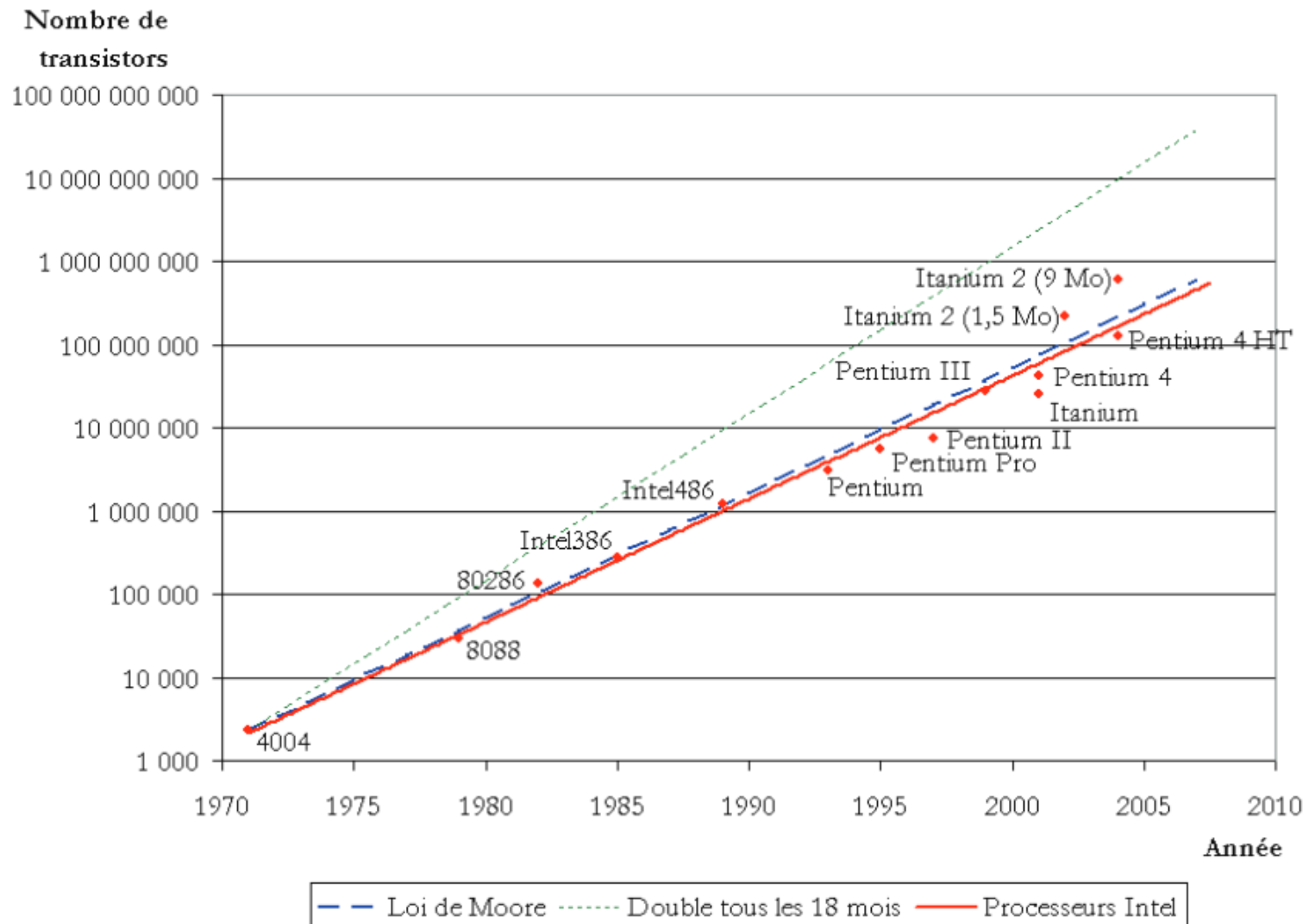
microchip DIRECT

Rechercher : jamont

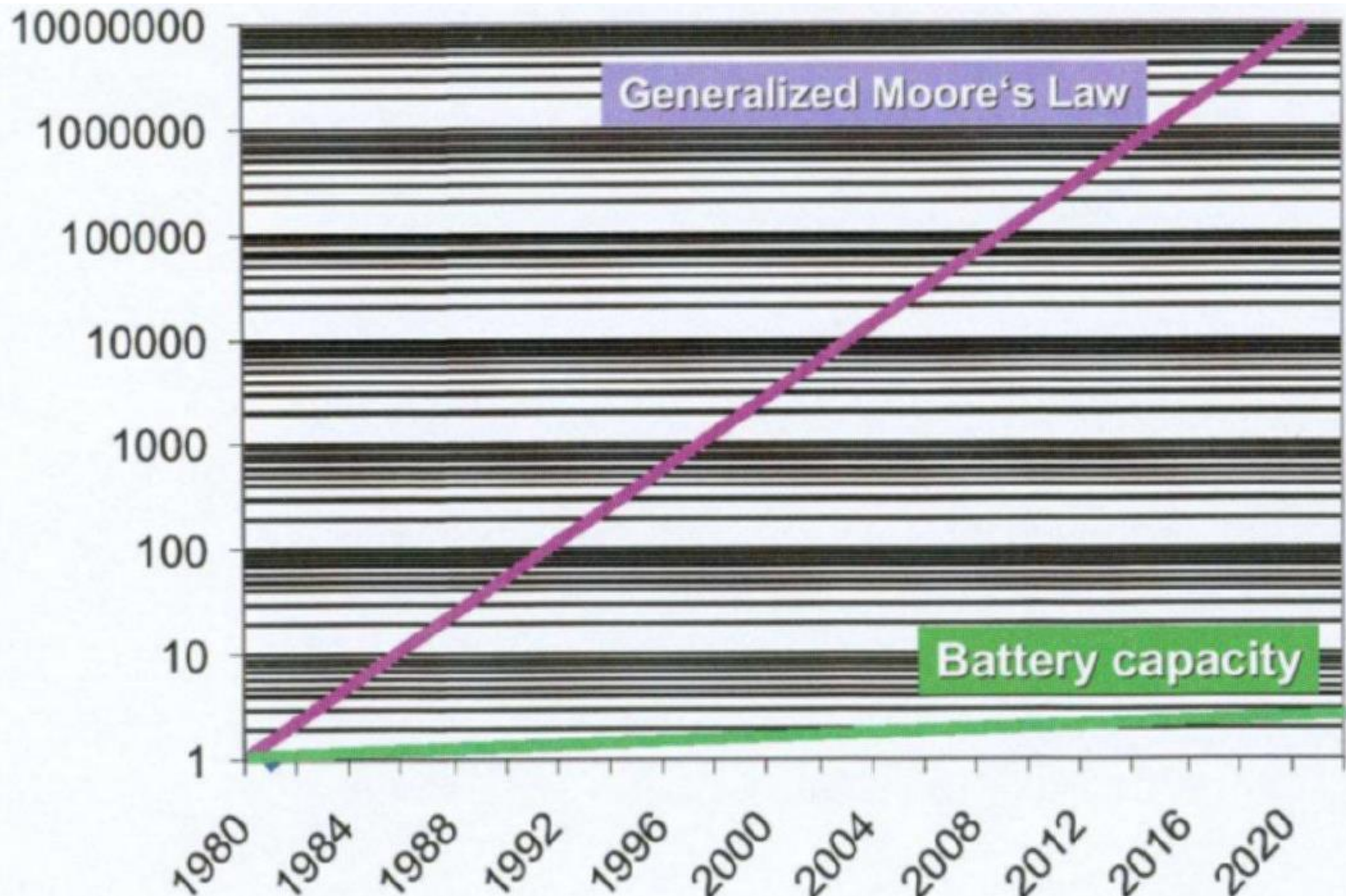
Suivant Précédent

Tout surligner Respecter la casse

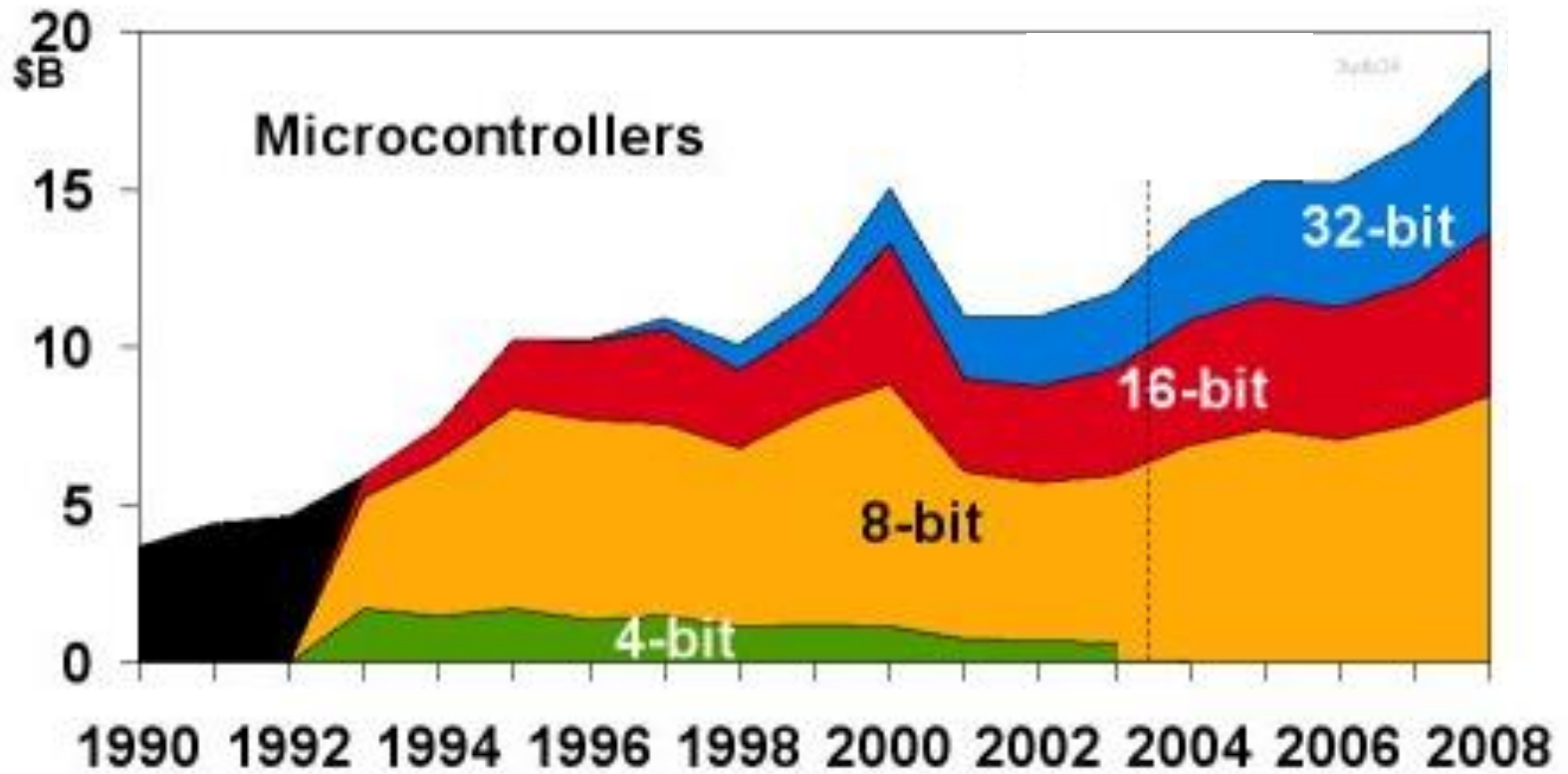
3. QUELS CHOIX POUR LA PARTIE MATÉRIELLE?



3. QUELS CHOIX POUR LA PARTIE MATÉRIELLE?

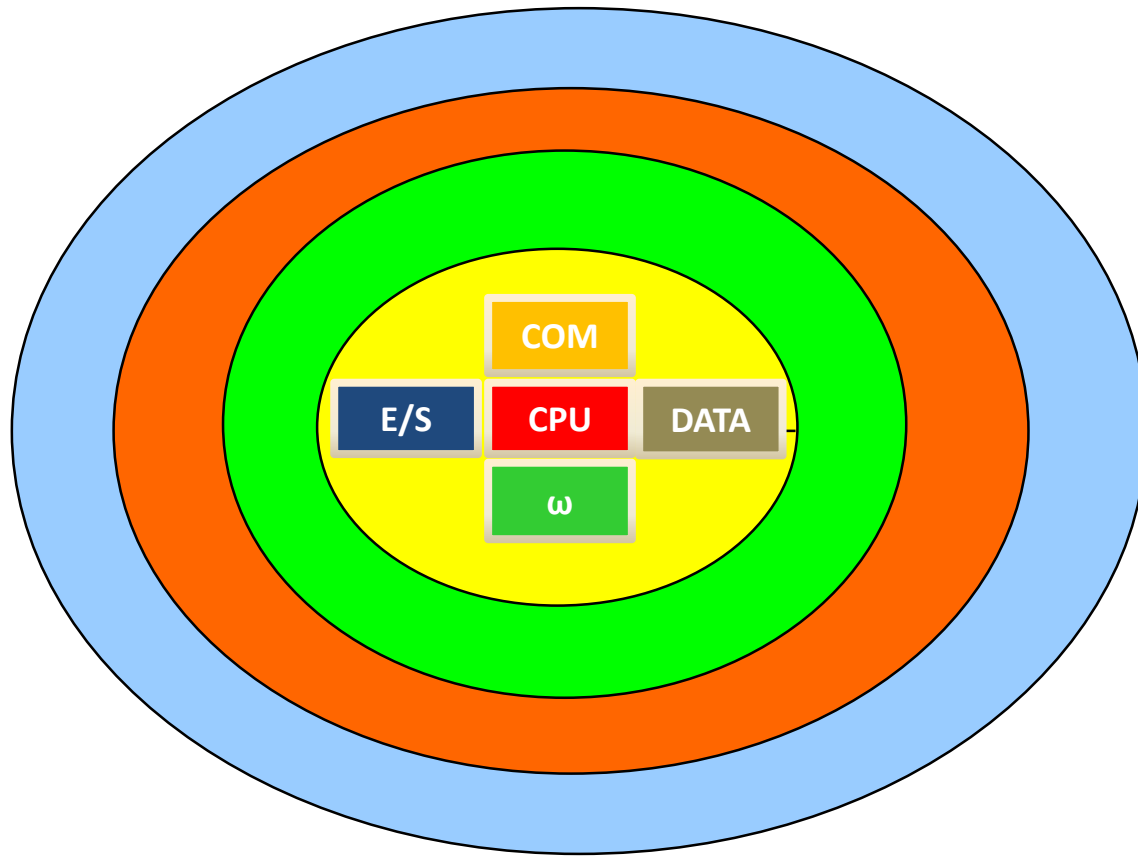


3. QUELS CHOIX POUR LA PARTIE MATÉRIELLE?



Source: Gartner Dataquest

3. VAIS-JE UTILISER UN OS?



3. VAIS-JE UTILISER UN OS?

Slicing the operating systems pie

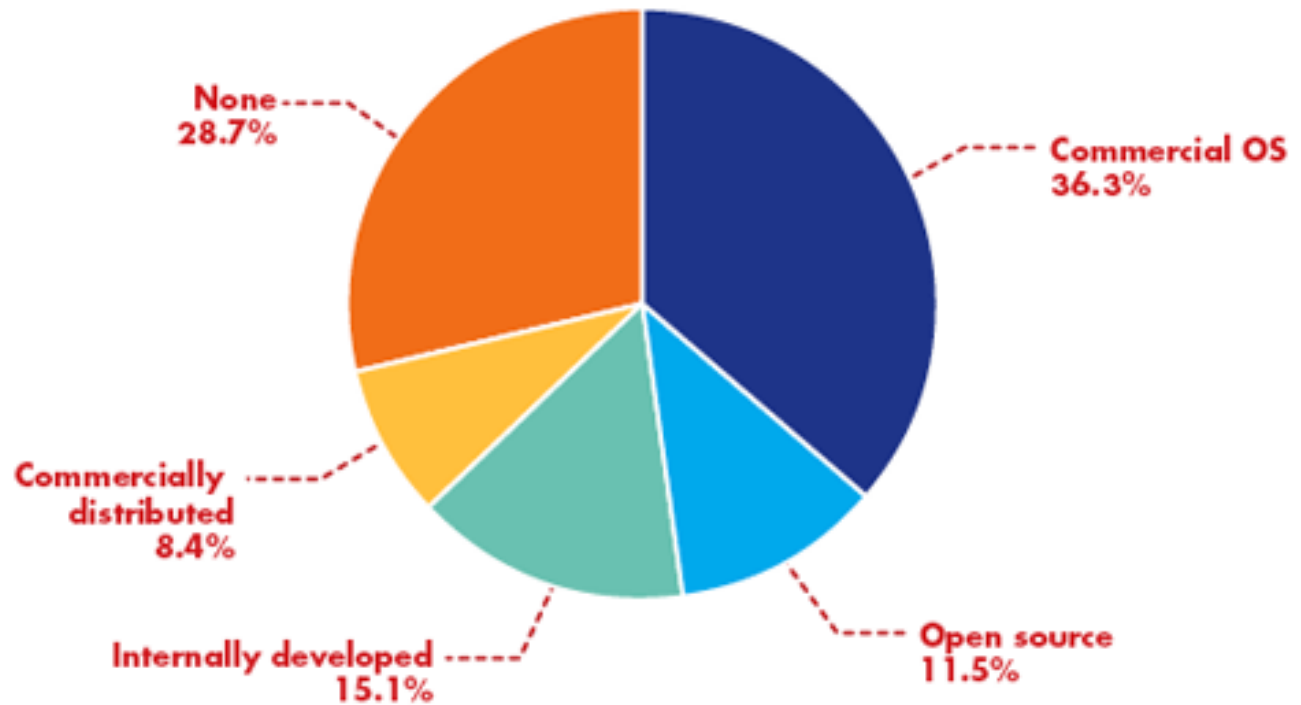


Figure 1

3. VAIS-JE UTILISER UN OS?

Quel type d'OS?

- **GPOS** (Normal General Purpose Operating System)
=> **Interruptible**
- **RTOS** (Real Time Operating System)
=> **Prédictabilité**

Critères	Temps partagé	Temps réel
But	Maximiser la capacité de traitement (débit) & utilisation des ressources	Etre prévisible (garantir les temps de réponse)
Temps de réponse	Bon en moyenne	Bon dans le pire des cas / moyenne non importante
Comportement à la charge	Confortable à l'utilisateur	Stabilité et respect des contraintes de temps

D'après J.Boukhobza , Systèmes d'exploitation embarqués

3. VAIS-JE UTILISER UN OS?

Operating systems evaluation criteria

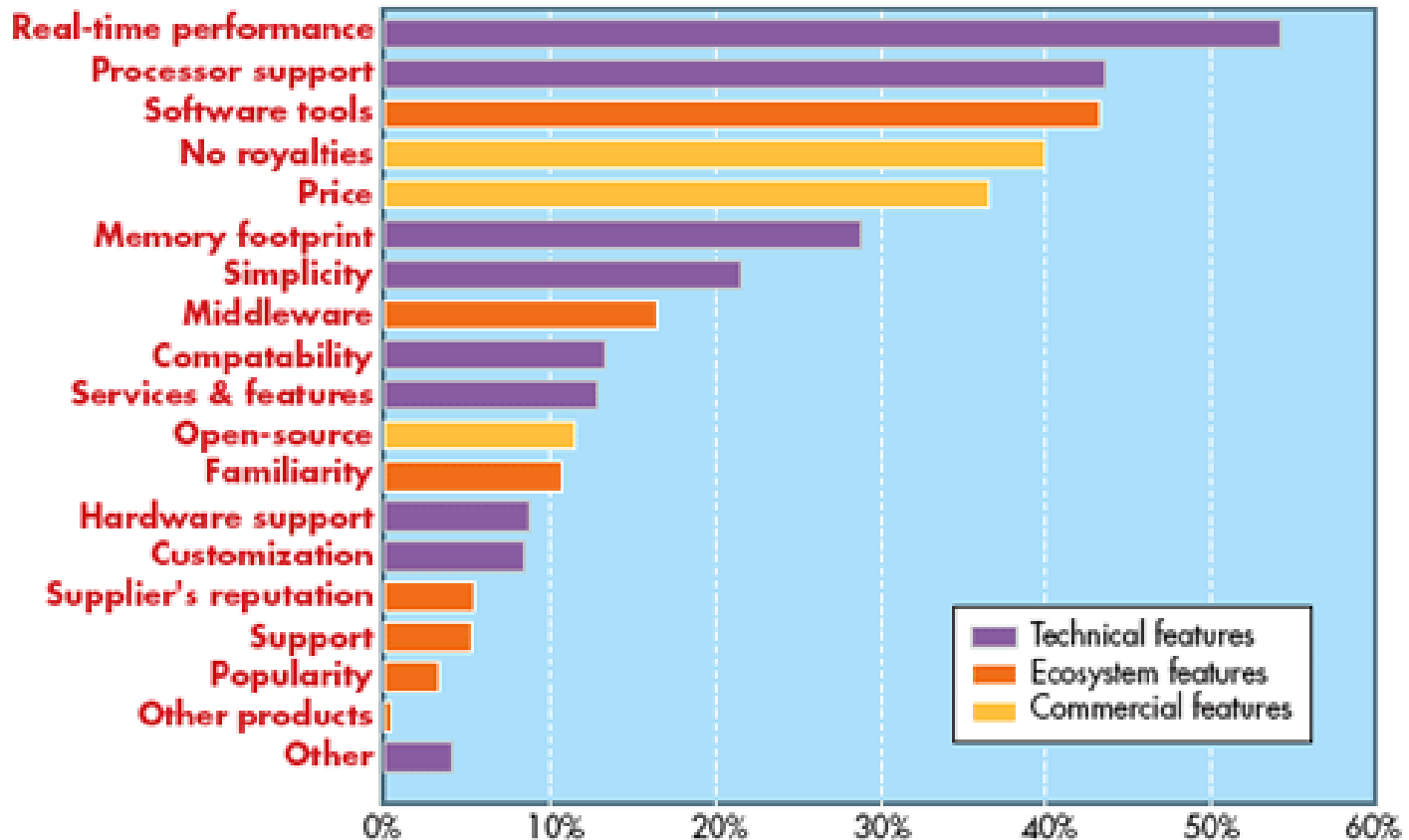


Figure 5

3. VAIS-JE UTILISER UN OS?

RTOS popularity now

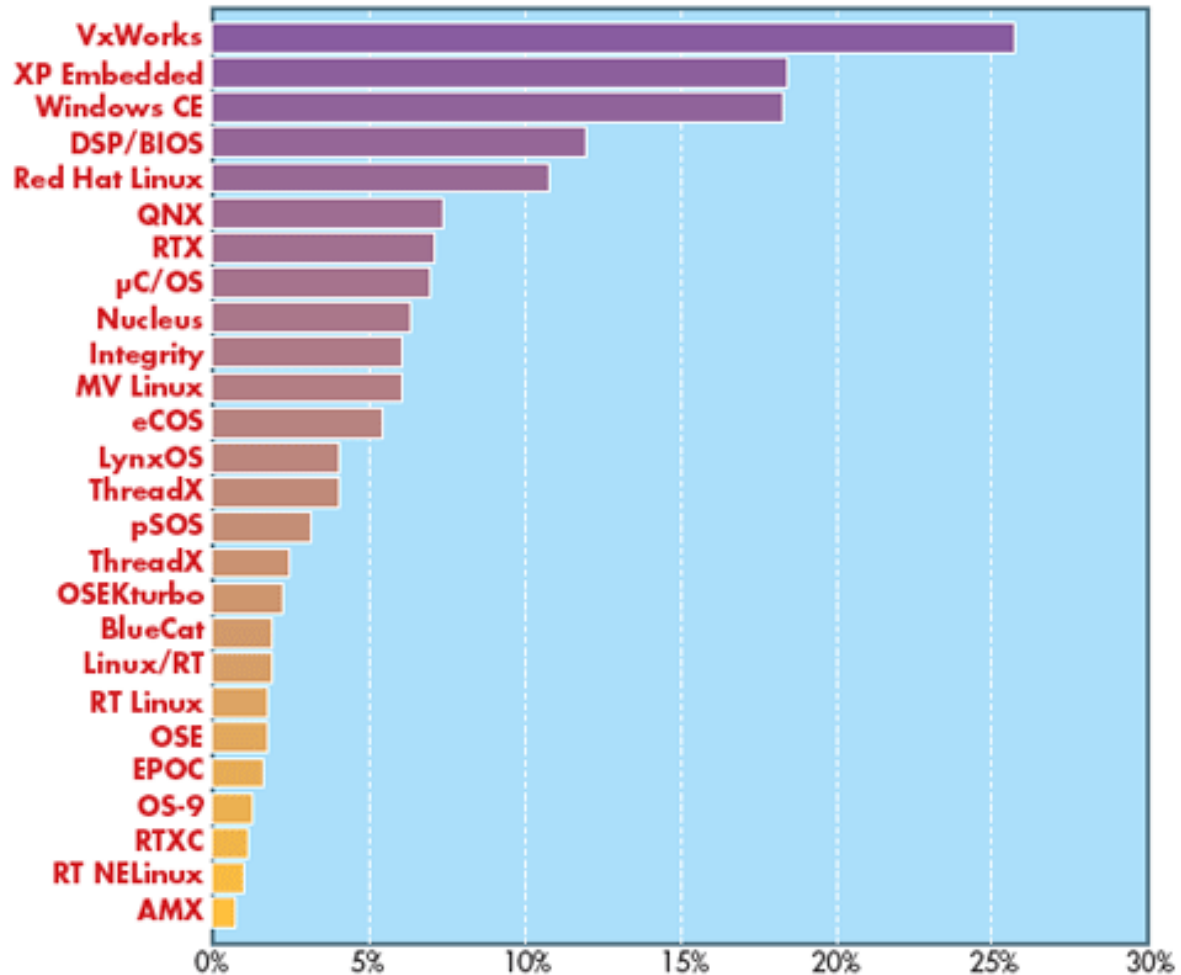


Figure 6

3. VAIS-JE UTILISER UN OS?

Pour créer&initaliser&activer une tâche avec VxWorks

```
int taskSpawn(  
    {Task Name},  
    {Task Priority 0-255, related to scheduling},  
    {Task Options - VX_FP_TASK, execute with floating point  
    coprocessor  
    VX_PRIVATE_ENV, execute task with private environment  
    VX_UNBREAKABLE, disable breakpoints for task  
    VX_NO_STACK_FILL, do not fill task stack with 0xEE}  
    {Stack Size}  
    {Task address of entry point of program in memory-initial PC  
    value}  
    {Up to 10 arguments for task program entry routine})
```

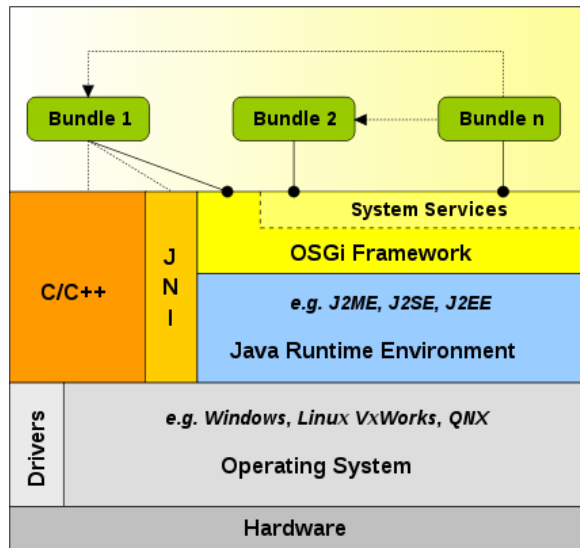
=> Après appel du taskSpawn, une image de la tâche est créée (Process Control Block , pile, programme)

3. VAIS-JE UTILISER UN OS?

```
// Tâche du parent qui active le l'horloge logicielle
void parentTask(void)
{
...
if sampleSoftware Clock NOT running {
newSWClkId = taskSpawn ("sampleSoftwareClock", 255,
VX_NO_STACK_FILL, 3000, (FUNCPTR) minuteClock, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0);
...
}
// Tâche exécutée par le programme fils
void minuteClock (void) {
integer seconds;
while (softwareClock is RUNNING) {
seconds = 0;
while (seconds < 60)
seconds = seconds+1;
}
.....
}
```

3. VAIS-JE UTILISER UN OS?

- Un **framework** au-dessus de l'OS?



Bundle = applications et/ou composants déployés

Services fournis:

- **journalisation**,
- gestion des **configurations**,
- le service **HTTP** (exec. servlets),
- l'analyse syntaxique **XML**, l'accès aux dispositifs (Device Access),
- l'administration de **paquetage** (Package Admin),
- l'administration des **permissions** (Permission Admin),
- le niveau de **démarrage** (Start Level),
- la gestion des **utilisateurs** (User Admin),
- le connecteur d'ES (IO Connector; IO = Input Output = Entrées Sorties),

- la gestion des **connexions** (Wire Admin),
- Jini, l'exportateur UPnP (UPnP Exporter),
- le **pistage applicatif** (Application Tracking),
- les paquets signés (Signed Bundles),
- les services **déclaratifs** (Declarative Services),
- la **gestion de l'énergie** (Power Management),
- la gestion des **dispositifs** (Device Management),
- les politiques de **sécurité** (Security Policies),
- **diagnostic/contrôle** et organisation en couches du cadrage (Diagnostic/Monitoring and Framework Layering).

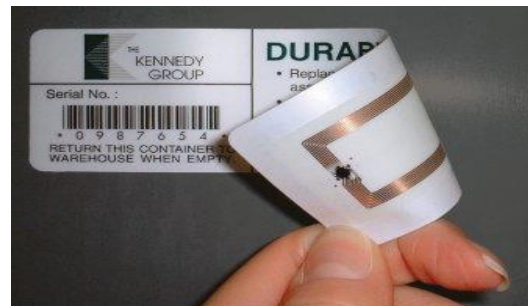
4. COMMENT VONT COMMUNIQUER LES OBJETS ENTRE EUX?

- Les réseaux sans fil sont des **briques de base** du *Web Intelligence* et de l'*Internet of Things*
 - ⇒ Il faut savoir identifier les critères pertinents qui vont conditionner le choix de ces briques
 - ⇒ Il faut comprendre la caractérisation de ces briques
- Il est indispensable de comprendre les couches basses d'un **système communicant** pour assurer une réelle **maîtrise d'ouvrage**.
- Les couches basses impactent profondément la **qualité de service** d'un système communicant! En effet, il existe un **fort couplage** entre les technologies utilisées pour les couches basses et :
 - La consommation d'énergie,
 - Le déterminisme (ou non) temporel,
 - Le débit (couplé avec l'environnement du système)
 - La précision d'une radiolocalisation
 - Le taux de perte
 - Le déséquencement
 - La latence
 - La gigue
 - ...

	Zigbee	Bluetooth	Wi-Fi
Besoins en mémoire	4-32 Kb	250 Kb	1 Mb
Autonomie avec pile	Années	Jours	Heures
Nombre de nœuds	65 000+	7	32
Vitesse de transfert	250 Kb/s	1 Mb/s	11-54-108-... Mb/s
Portée	10-100m	10-100 m	300 m

5. COMMENT MON OBJET ACCÈDE À INTERNET?

- Deux grands types d'objets physiques
 - Objets communicants et/ou intelligents
 - A base de micro-contrôleur(s) /FPGA équipés d'interface(s) de communication
 - Modèle de **COMPORTEMENT EMBARQUÉ** sur l'objet
 - Objets « chipless » tagués
 - Objets sur lesquels on a apposé une étiquette RFID (ou autre)
- Modèle de **COMPORTEMENT DÉPORTÉ** sur un serveur distant

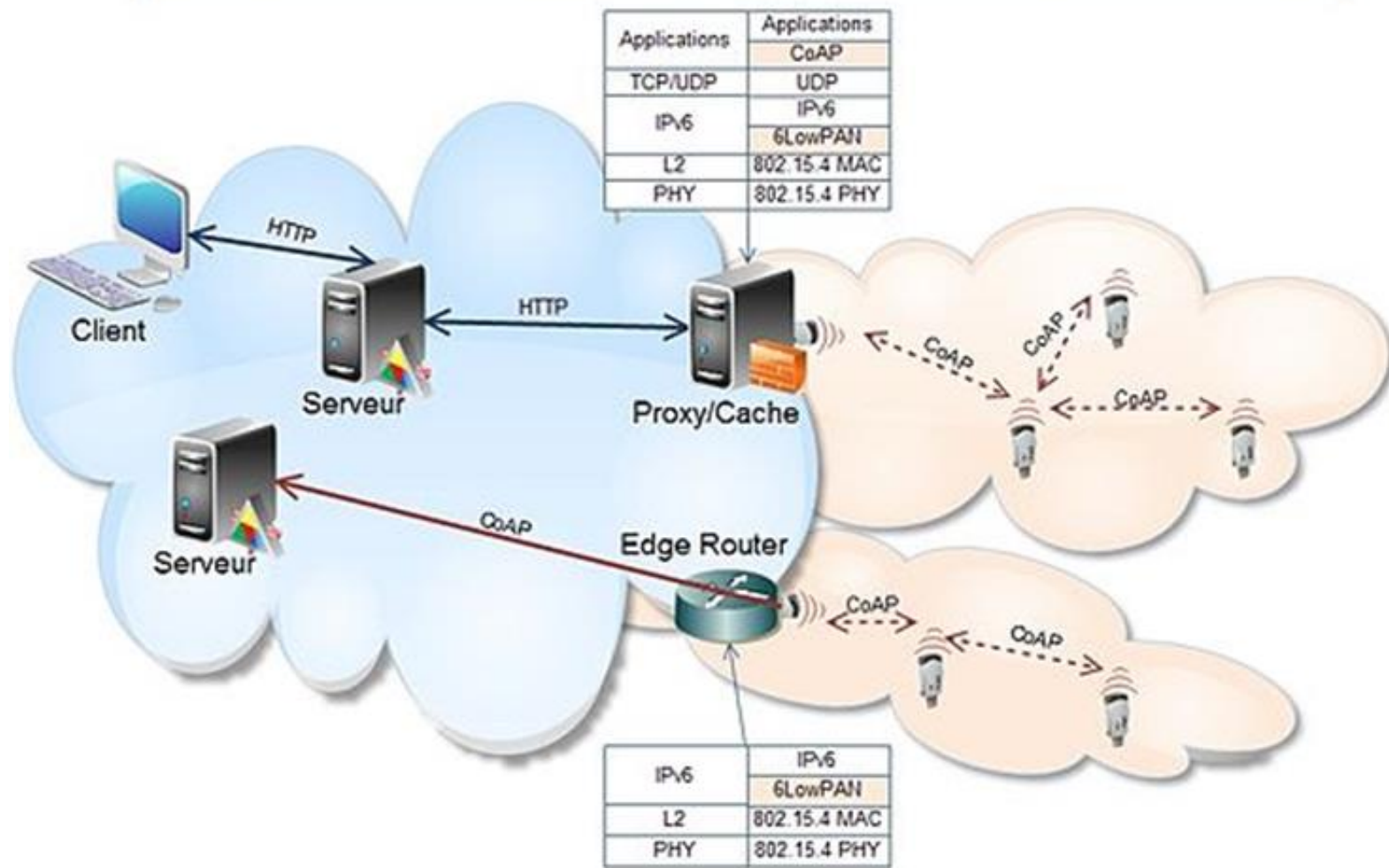


5. COMMENT MON OBJET ACCÈDE À INTERNET?

Comportement embarqué

Comportement déporté

Architecture REST

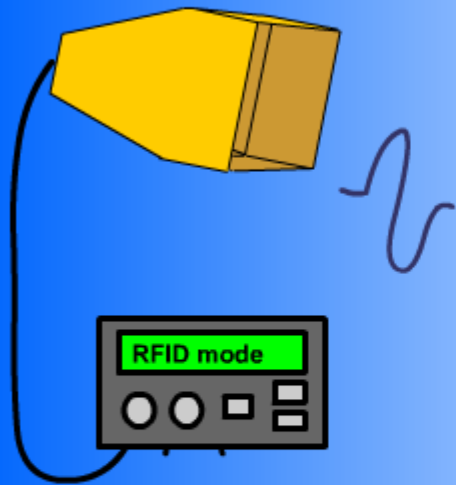


5. COMMENT MON OBJET ACCÈDE À INTERNET?

Comportement embarqué

Comportement déporté

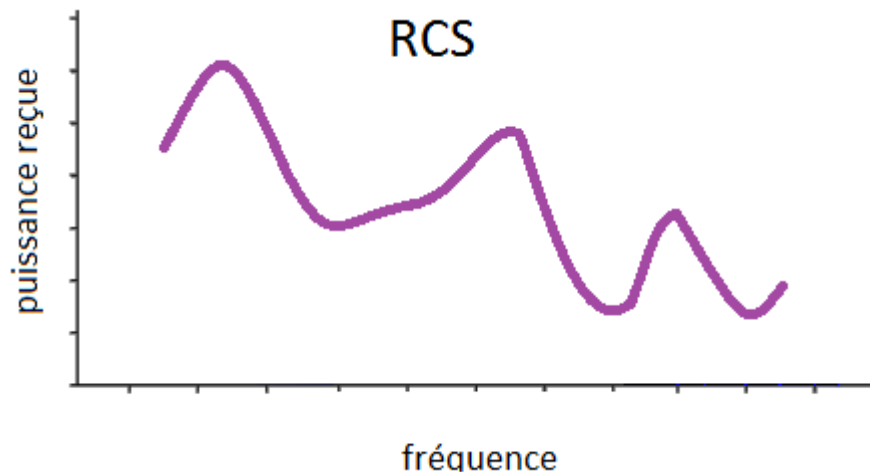
RFID chipless tag system



5. COMMENT MON OBJET ACCÈDE À INTERNET?

Comportement embarqué

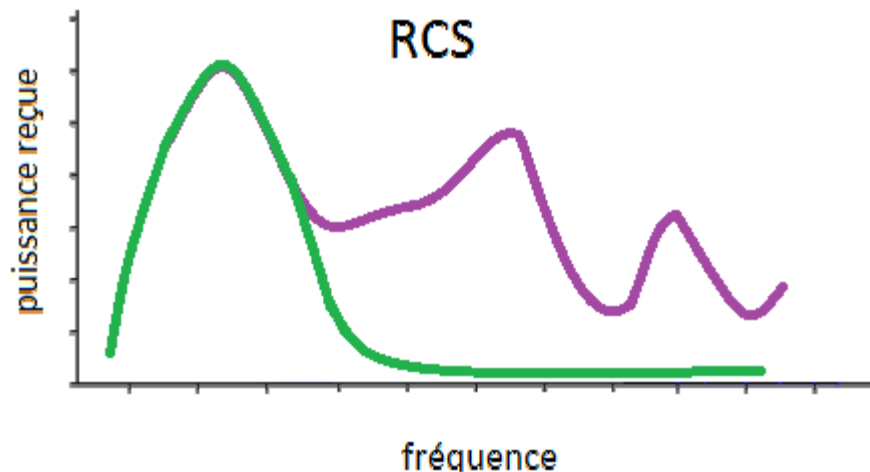
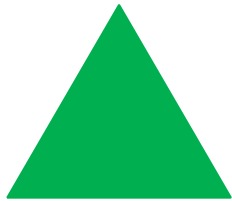
Comportement déporté



5. COMMENT MON OBJET ACCÈDE À INTERNET?

Comportement embarqué

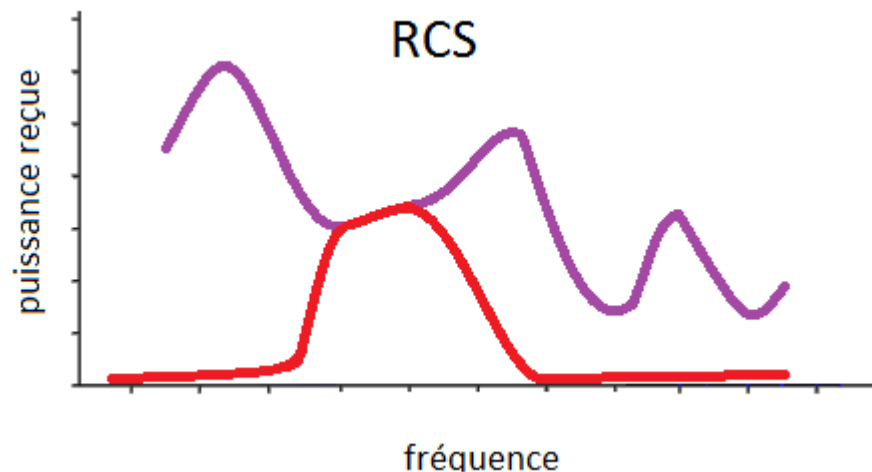
Comportement déporté



5. COMMENT MON OBJET ACCÈDE À INTERNET?

Comportement embarqué

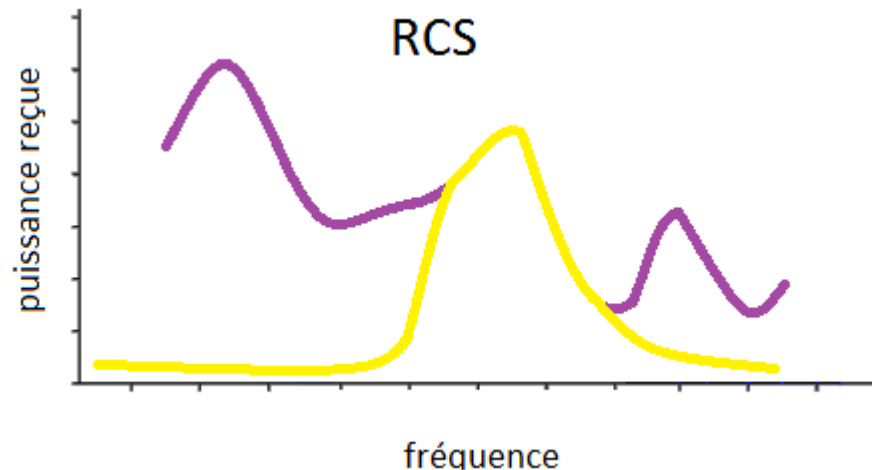
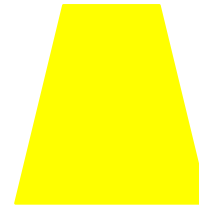
Comportement déporté



5. COMMENT MON OBJET ACCÈDE À INTERNET?

Comportement embarqué

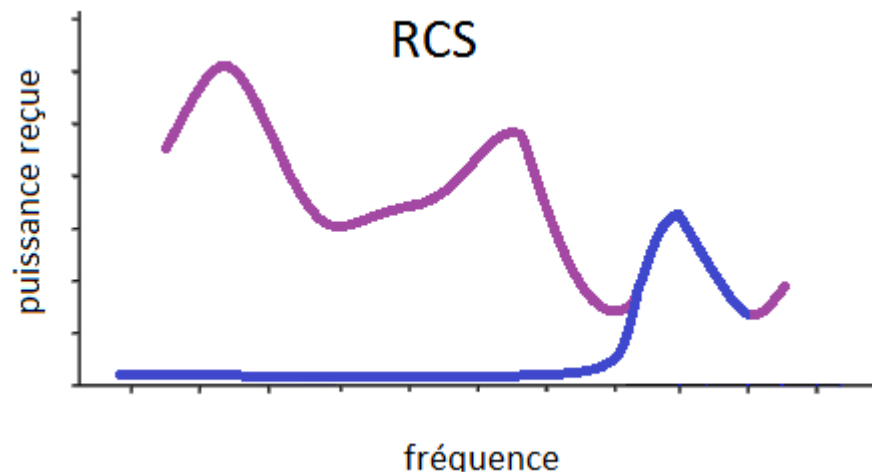
Comportement déporté



5. COMMENT MON OBJET ACCÈDE À INTERNET?

Comportement embarqué

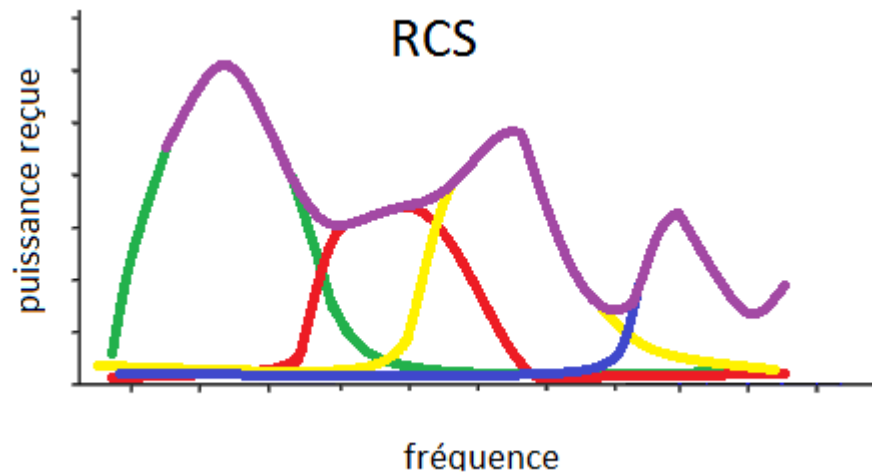
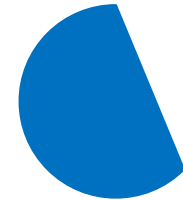
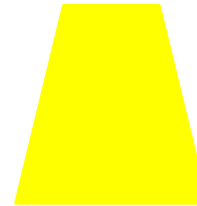
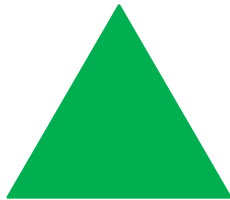
Comportement déporté



5. COMMENT MON OBJET ACCÈDE À INTERNET?

Comportement embarqué

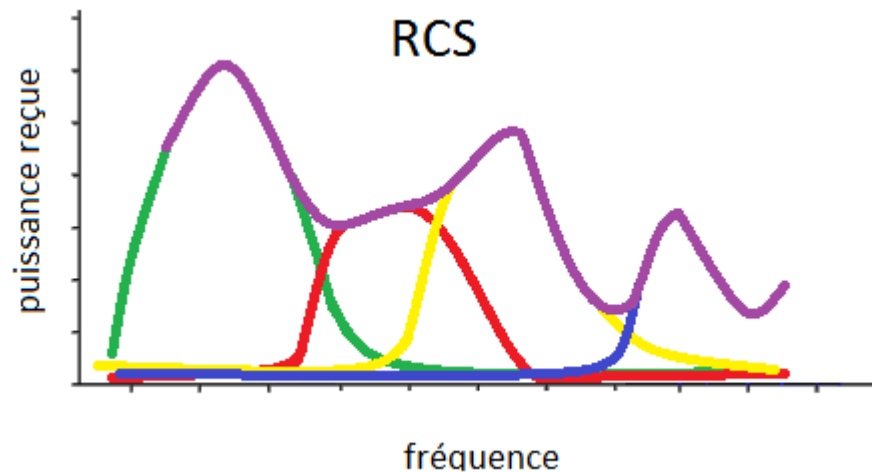
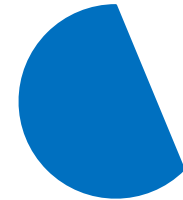
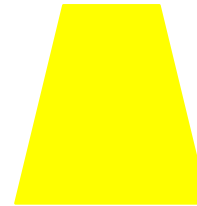
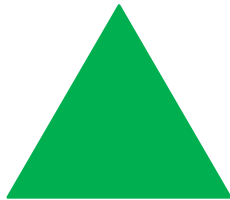
Comportement déporté



5. COMMENT MON OBJET ACCÈDE À INTERNET?

Comportement embarqué

Comportement déporté

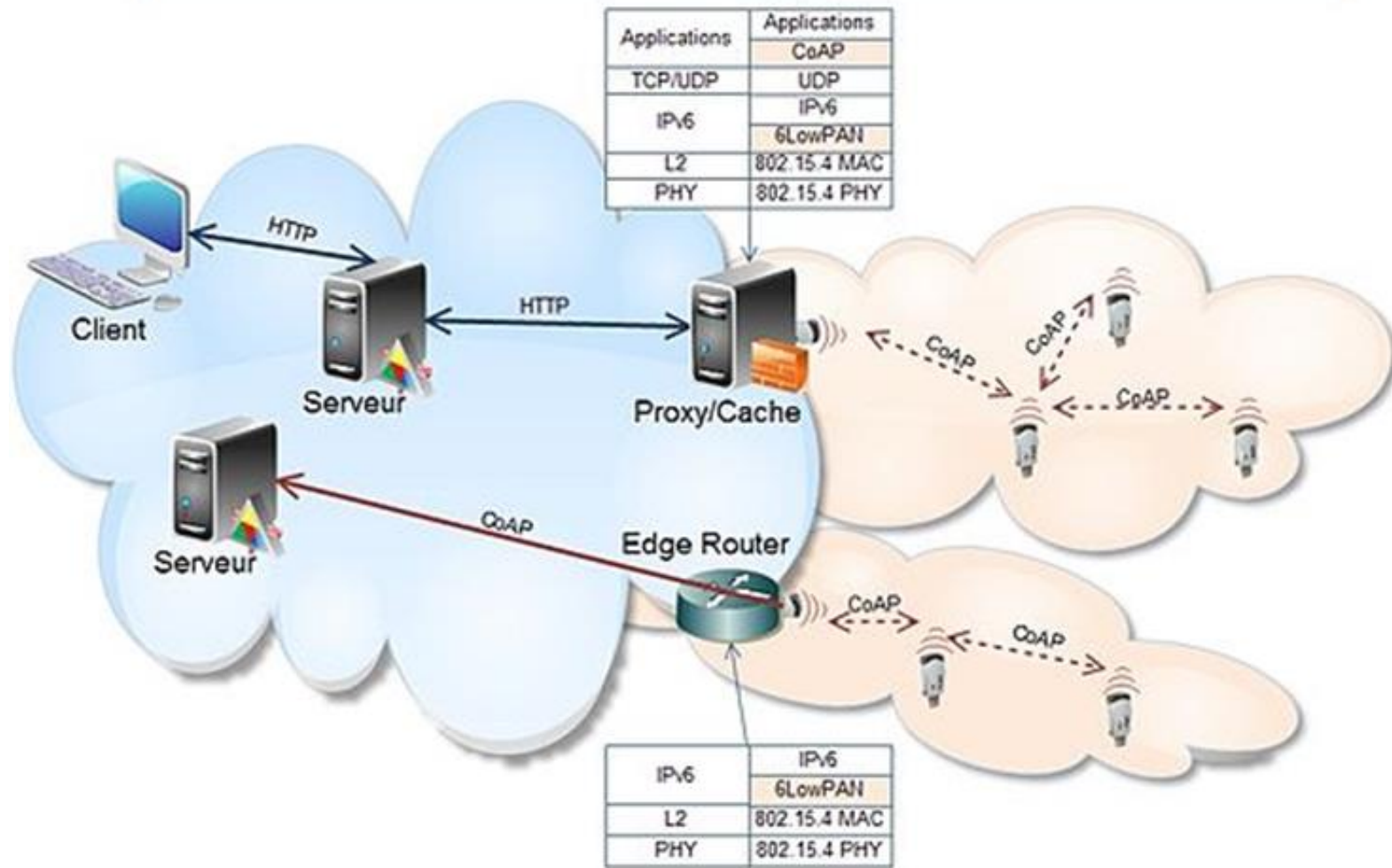


5. COMMENT MON OBJET ACCÈDE À INTERNET?

Comportement embarqué

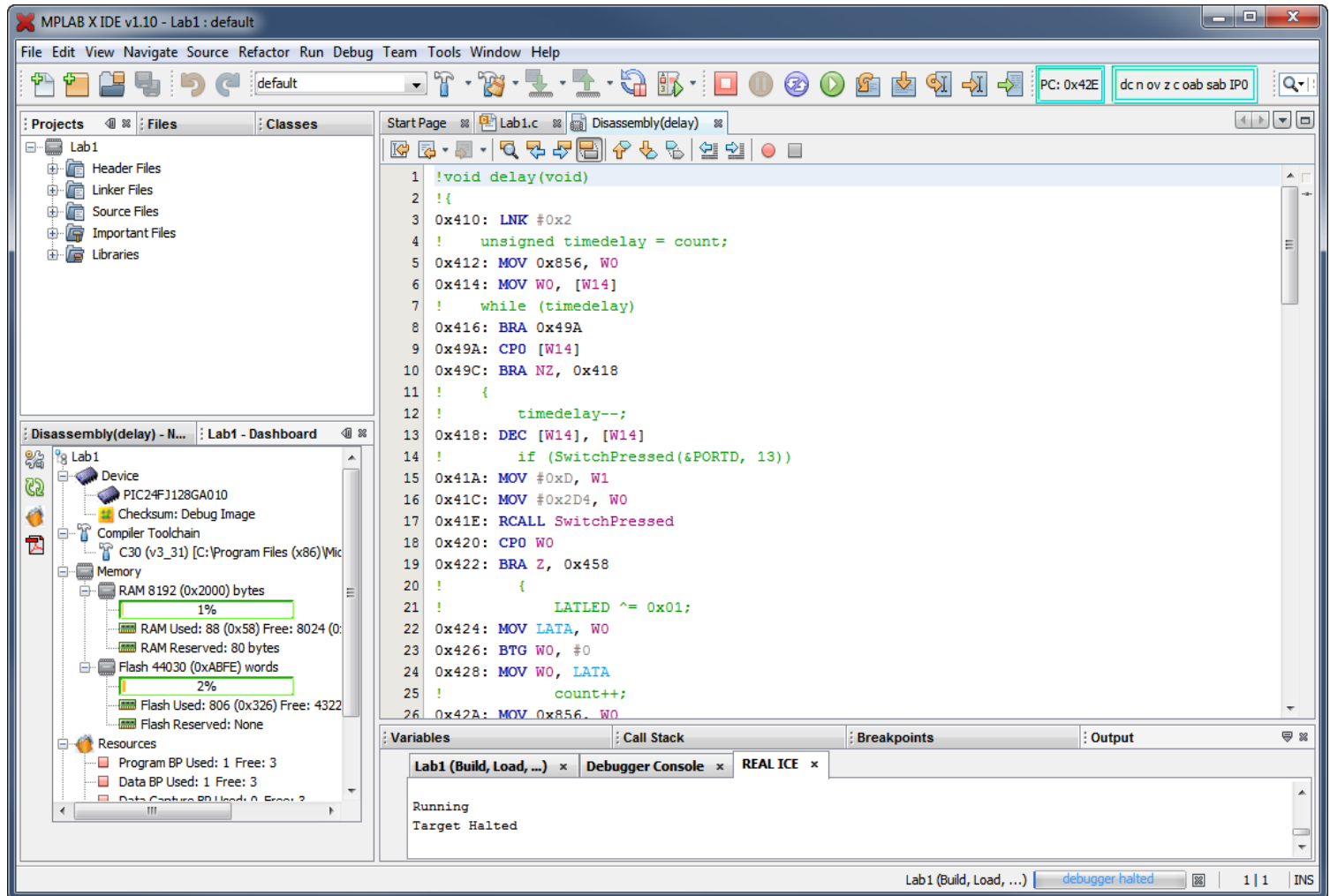
Comportement déporté

Architecture REST



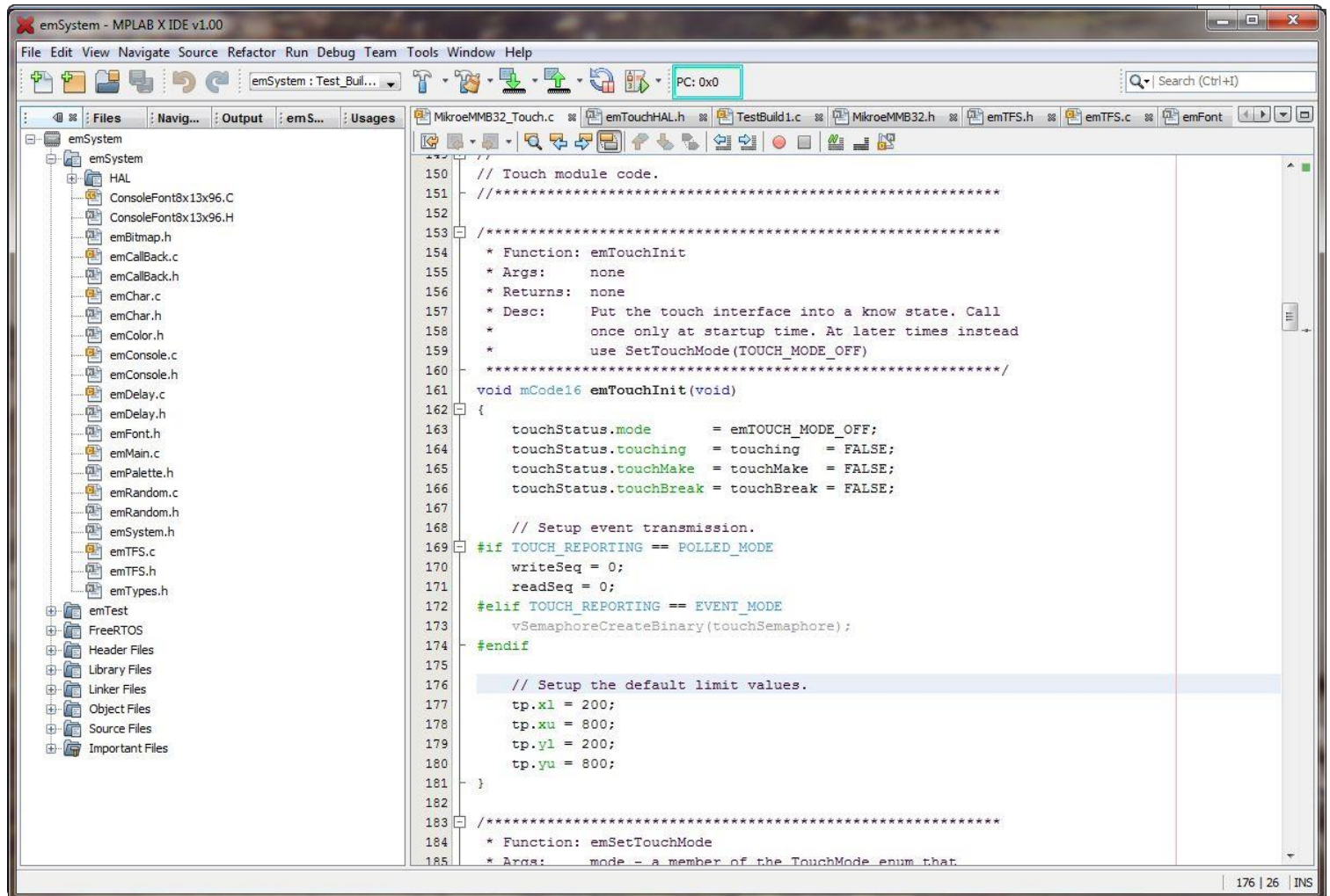
6. JE DÉVELOPPE MON APPLICATION

- Choisir la chaine de développement



6. JE DÉVELOPPE MON APPLICATION

- Choisir la chaine de développement



6. JE DÉVELOPPE MON APPLICATION

- Choisir la chaine de développement

The screenshot displays the MPLAB X IDE v1.20 interface for a PIC24 application. The main window shows the source code for `WORD GOLDrawCallback(void)`, which includes logic for updating a time display and processing events. The `main` function is also visible, showing the initialization of the board and the creation of the scheme.

Project Explorer: Shows the project structure, including source files, object files, and libraries. The `Graphics` folder is expanded, showing the `GOLDrawCallback` function.

Source Editor: Displays the `GOLDrawCallback` function, which updates the time display and processes events. The code includes comments and variable declarations.

Call Stack: Shows the current call stack, including `Delay10us`, `DelayMs`, and `GOLFindObject`.

File Registers: Displays the current state of the file registers, including the `0000` register.

Watches: Shows the current state of the watches, including the `tempBuffer` array.

main - Call Graph: A graph showing the call relationships between the `main` function and other functions in the application, such as `InitializeBoard`, `GOLCreateScheme`, `GOLDraw`, `UpdateMeter`, `DisplayPullDown`, `RTCCProcessEvents`, `GetECGSamples`, `GOLFindObject`, `GOLMsg`, and `TouchGetMsg`.

6. JE DÉVELOPPE MON APPLICATION

- Etudier les *datasheets* du micro-contrôleur

7.3 Reading the Data EEPROM Memory

To read a data memory location, the user must write the address to the EEADRH:EEADR register pair, clear the EEPGD control bit (EECON1<7>), clear the CFGS

control bit (EECON1<6>) and then set the RD control bit (EECON1<0>). The data is available for the very next instruction cycle; therefore, the EEDATA register can be read by the next instruction. EEDATA will hold this value until another read operation or until it is written to by the user (during a write operation).

EXAMPLE 7-1: DATA EEPROM READ

```
MOVLW DATA_EE_ADDRH ; Upper bits of Data Memory Address to read
MOVWF EEADRH
MOVLW DATA_EE_ADDR   ; Lower bits of Data Memory Address to read
MOVWF EEADR
BCF  EECON1, EEPGD    ; Point to DATA memory
BCF  EECON1, CFGS     ; Access EEPROM
BSF  EECON1, RD       ; EEPROM Read
MOVF  EEDATA, W       ; W = EEDATA
```

7.4 Writing to the Data EEPROM Memory

To write an EEPROM data location, the address must first be written to the EEADRH:EEADR register pair and the data written to the EEDATA register. Then the sequence in Example 7-2 must be followed to initiate the write cycle.

The write will not initiate if the above sequence is not exactly followed (write 55h to EECON2, write AAh to EECON2, then set WR bit) for each byte. It is strongly recommended that interrupts be disabled during this code segment.

Additionally, the WREN bit in EECON1 must be set to enable writes. This mechanism prevents accidental writes to data EEPROM due to unexpected code

execution (i.e., runaway programs). The WREN bit should be kept clear at all times except when updating the EEPROM. The WREN bit is not cleared by hardware.

After a write sequence has been initiated, EECON1, EEADRH, EEADR and EEDATA cannot be modified. The WR bit will be inhibited from being set unless the WREN bit is set. Both WR and WREN cannot be set with the same instruction.

At the completion of the write cycle, the WR bit is cleared in hardware and the EEPROM Write Complete Interrupt Flag bit (EEIF) is set. The user may either enable this interrupt or poll this bit. EEIF must be cleared by software.

EXAMPLE 7-2: DATA EEPROM WRITE

```
MOVLW DATA_EE_ADDRH ; Upper bits of Data Memory Address to write
MOVWF EEADRH
MOVLW DATA_EE_ADDR   ; Lower bits of Data Memory Address to write
MOVWF EEADR
MOVLW DATA_EE_DATA    ; Data Memory Value to write
MOVWF EEDATA
BCF  EECON1, EEPGD    ; Point to DATA memory
BCF  EECON1, CFGS     ; Access EEPROM
BSF  EECON1, WREN     ; Enable writes

; Disable Interrupts
BCF  INTCON, GIE
MOVLW 0x55
MOVWF EECON2
MOVLW 0xAA
MOVWF EECON2
BSF  EECON1, WR       ; Set WR bit to begin write
BSF  INTCON, GIE      ; Enable Interrupts

; User code execution
BCF  EECON1, WREN     ; Disable writes on write complete (WEIF set)
```

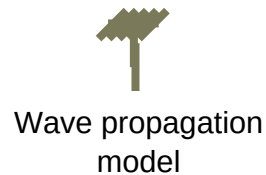
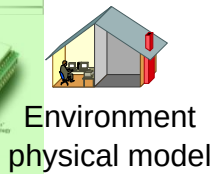
Risques acceptables / Satisfaction des utilisateurs

7. JE VALIDE MON SYSTÈME

---Des outils spécifiques---

MODELS INVOLVED IN THE SYSTEM SIMULATION

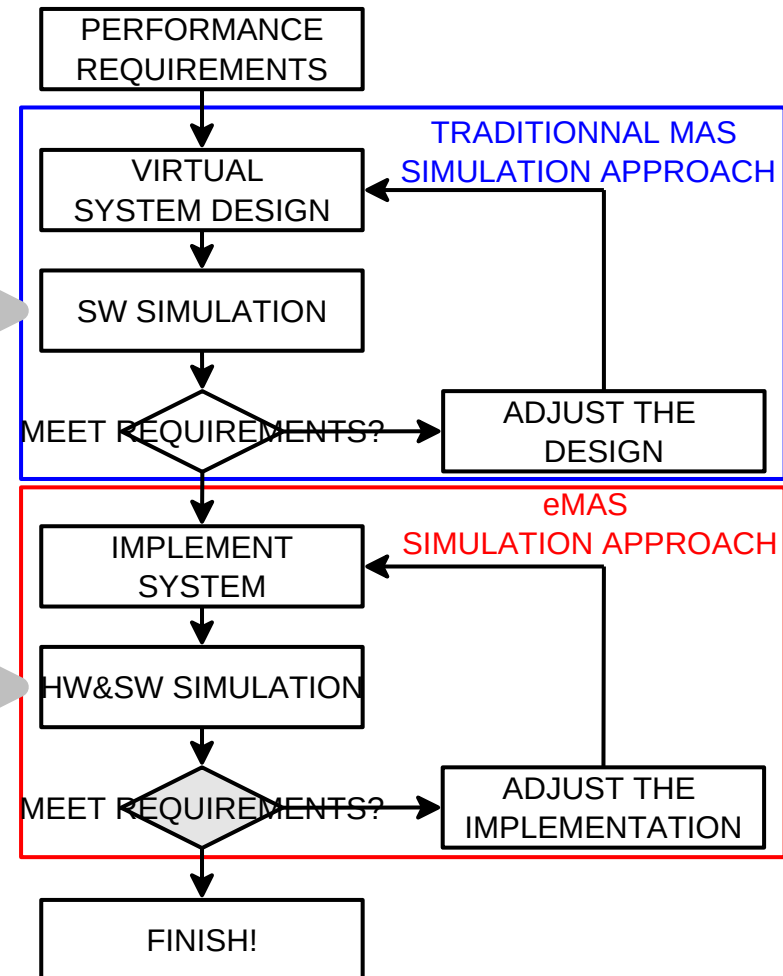
Environnement description models



Entities involved models



MASH [Jamont13]



7. JE VALIDE MON SYSTÈME

---Performances énergétiques---

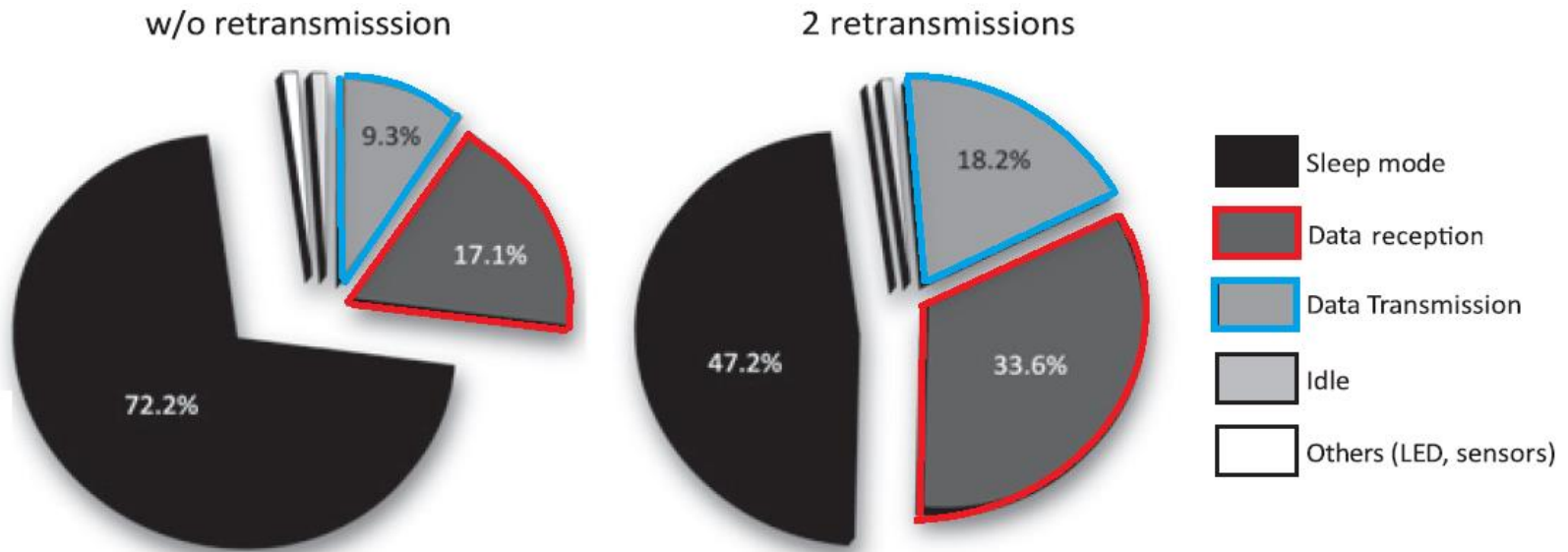


Fig. 6. Energy consumption repartition for different scenari.

[Fourty12]

MERCI POUR VOTRE ÉCOUTE



