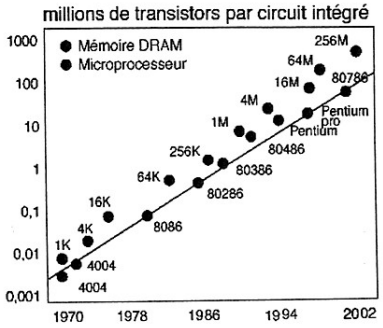
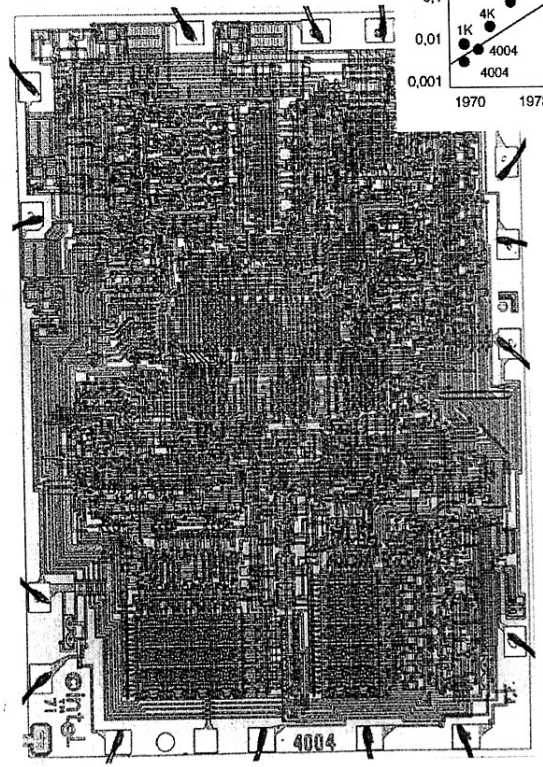
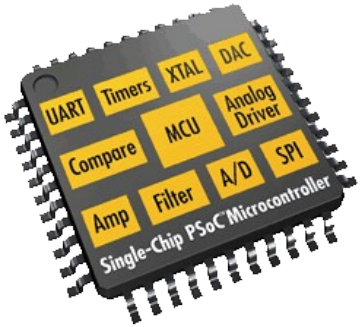
Cours « microcontrôleurs »







Plan du cours

- **Introduction : définitions, sites Internet et biblio., historique, bref rappel sur les technologies et familles logiques, mémoires, …**
- **Codage de l'information (rappels)**
- **Architecture interne des microcontrôleurs et fonctionnement**
- **Différents types et grandes familles de microcontrôleurs**
- **Applications des microcontrôleurs, futur (TCP-IP, CAN, …)**
- **Architecture classique d'une carte à microP ou microcontrôleur**
- **Langages et outils de programmation des microP (introduction)**
- **Interfaçage des périphériques et décodage, plan mémoire (*mapping*)**
- **Périphériques : liaisons série synchrone et asynchrone, ports parallèles, timers, protocoles (introduction), …**
- **Principe et gestion des interruptions**
- **Instructions assembleur et modes d'adressage**
- **Langages évolués**
- **Critères de choix des microcontrôleurs**
- **Les tendances actuelles :**
  - **DSP, circuits logiques programmables FPGA, mixtes DSP-FPGA, ASIC, analogique programmable, mixtes analog/num programmables, IP microcontrôleur sur FPGA**
  - **Exemples : PSoc (Cypress), PicoBlaze (Xilinx), …**

Microprocesseur / Microcontrôleur

Définitions

- **Microprocesseur :**

« Le microprocesseur est organisé pour extraire des instructions de sa mémoire de programmes au cours de cycles d'acquisition et exécuter ces instructions lors de cycles d'exécution »

« C'est le circuit qui se charge de manipuler les données (i.e. calculer) mais c'est aussi le << chef d'orchestre >> de la machine. Il pilote et distribue les tâches. Il est caractérisé par la cadence maximale à laquelle il est capable de travailler, par la taille et le nombre de données qu'il peut manipuler. »

« Le microprocesseur est un circuit intégré logique commandé par un programme. Il est capable :

- **d'organiser des transferts d 'informations**
- **de réaliser des opérations arithmétiques et logiques**

- **Microcontrôleur :**

Sur la même puce que le microprocesseur, divers périphériques et mémoires ont été intégrés. Peut être « généraliste » ou « spécialisé » .

Microprocesseur = orienté « système », modes « utilisateur » et « superviseur »

Microcontrôleur = orienté « contrôle - commande », périphériques spécialisés

Quelques sites Internet

- Les sites des principaux fabricants de composants :

  Intel, Motorola, Texas Instruments, Microchip, National Semiconductor, Zilog, NEC, ….

- Autres sites intéressants :

  - Cours d'électronique, logique, …

    http://www.multimania.com/xcotton/electron/coursetdocs.htm

    - Un cours d'initiation aux microcontrôleurs (Université de Lille) :

    http://www.univ-lille1.fr/~eudil/oscr/sc00a.htm

  - Un cours sur les microprocesseurs (SUPELEC Rennes) :

    http://www.supelec-rennes.fr/ren/perso/jweiss/microp/

  - Un cours sur l'électronique (BTS Nançy) :

  http://www.ac-nancy-metz.fr/pres-etab/lycom/Electro_cours/Electro_coursmenu.htm

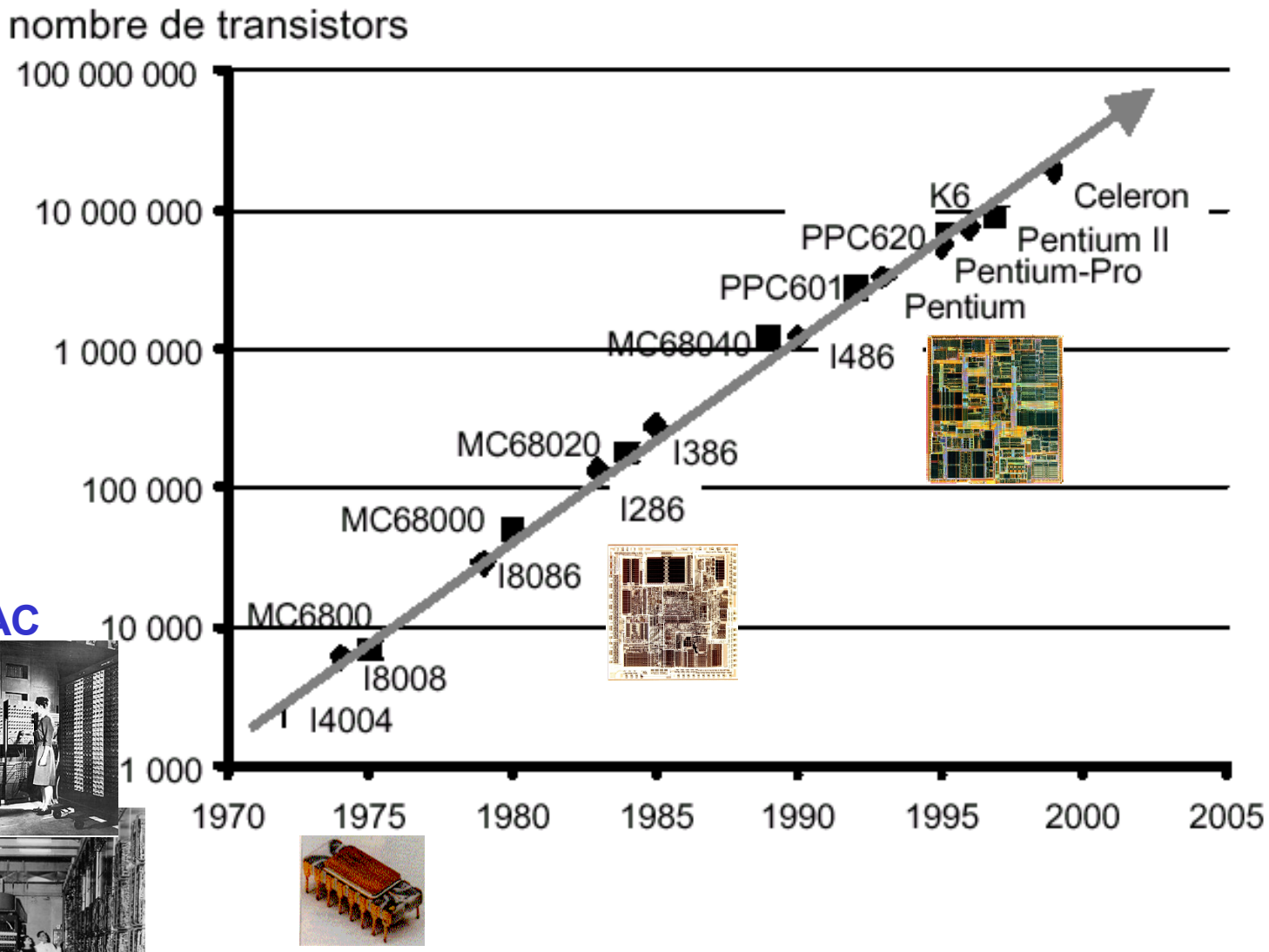
  - Présentations orales de manifestations organisées par le CRESITT

    www.cresitt.com

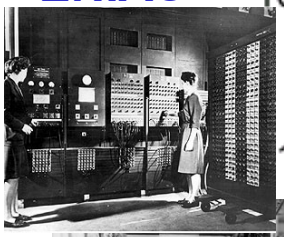
Quelques livres

- « Les microcontrôleurs 4 et 8 bits » C. Tavernier, DUNOD
  - Microcontrôleurs, mise en œuvre matérielle, développement et tests
  - Microcont. INTEL 8051, Motorola 68HC05 et 68HC11, NEC, SGS Thomson ST6, Texas Instruments TMS370, Microchip PIC et autres ouvrages de C. Tavernier chez DUNOD
- « Le microprocesseur et son environnement », Robert Du Bois, DUNOD *Tech*
  - Général. Bonne présentation des réseaux et des périphériques
- « Computer organization and architecture », William Stallings, Prentice Hall, une des références en langue anglaise
- Indispensable : « Le langage C » de B.W. Kernighan et D.M. Ritchie, MASSON
  - La référence sur le langage C, avec des exemples
- « Les microprocesseurs 16 bits à la loupe » R. Dubois, D. Girod, EYROLLES
  - Très détaillé sur 8086, 6800, Z8000 avec leurs périphériques, coprocesseurs, …
- « Microprocesseurs 16 bits », M. Aumiaux, MASSON
  - 8086 et 68000. Très détaillé sur l'interfaçage, les interruptions
- « Le microprocesseur 68000 et sa programmation », P. Jaulent, EYROLLES
- Consulter les « *Techniques de l'ingénieur* » (B.U. / accessible via internet) : structures des ordinateurs, microprocesseurs, microélectronique
- Les documentations des fabricants et les notes d'applications
- Lisez (au moins une fois !) les revues spécialisées *Electronique*, *Elect. Internat. Hebdo.*, *Elektor*, etc

Bref historique des microprocesseurs

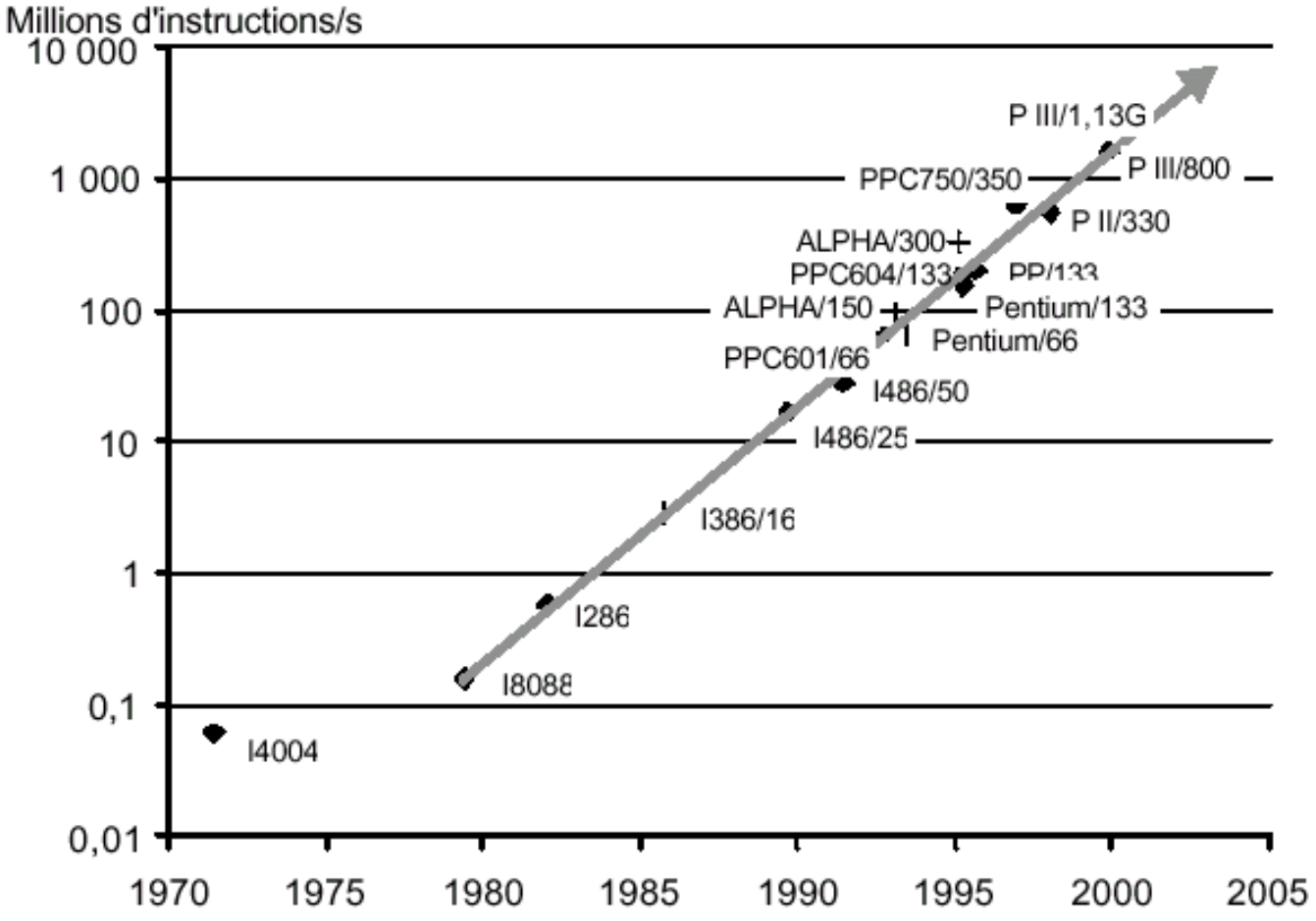


ENIAC



Évolution des microprocesseurs et mémoires

Evolution de la complexité des microprocesseurs



Evolution de la performance des microprocesseurs

Historique : quelques dates

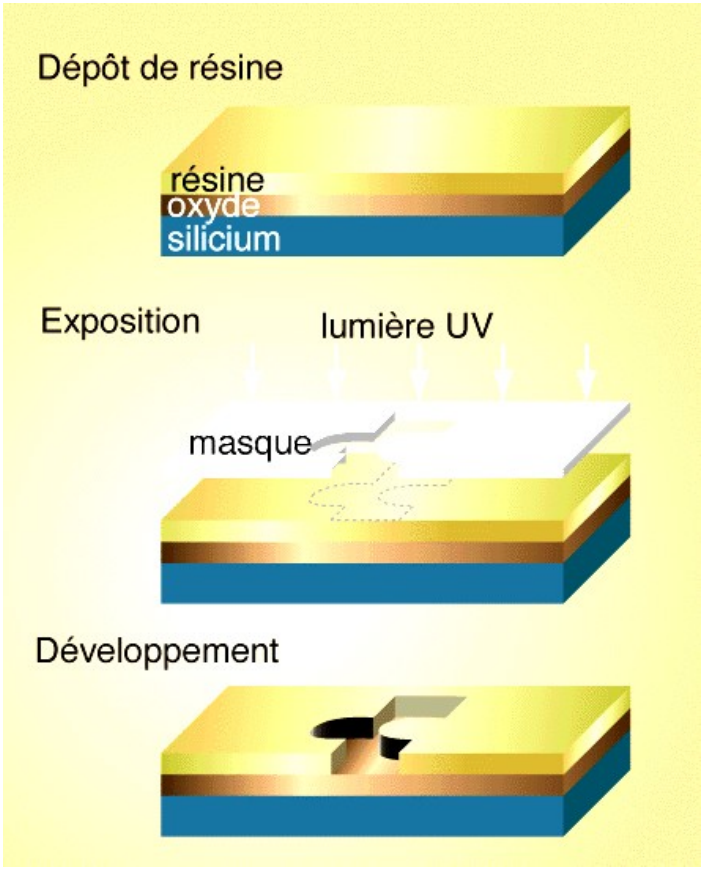
- **Premiers « calculateurs »**
  - **Machine mécanique de Blaise Pascal (1642)**
  - **Premier langage de communication « universel » : Morse 1837**
  - **Calculateurs mécaniques**
  - **Puis calculateurs électromécaniques (relais)**
  - **Puis calculateurs électroniques : tubes à vide**
    - **ENIAC (1946) : 18000 tubes, 60000 instructions/seconde, 30 tonnes, 75 m2 et 170 kW !**
  - **1940 : le circuit imprimé**
  - **1946 : architecture de Von Neumann**
  - **1947 : invention du transistor (Shockley, Brattain, Bardeen)**
  - **1950 : mémoires à tores de ferrite**
  - **1951 : premier ordinateur (UNIVAC)**
  - **1955 : le langage FORTRAN**
  - **1958 : premier circuit intégré (Jack Kilby,Texas Instruments)**
  - **1959 : la technologie « planar » de FAIRCHILD (photolithographie)**
  - **1960 : le langage COBOL**
  - **1964 : l'IBM 360 (1er ordinateur à circuits intégrés), code ASCII**
  - **Années 1969-70 : premières RAM et DRAM**
  - **1969 : le langage PASCAL**
  - **1970 : IBM 370 (la « mini » informatique)**

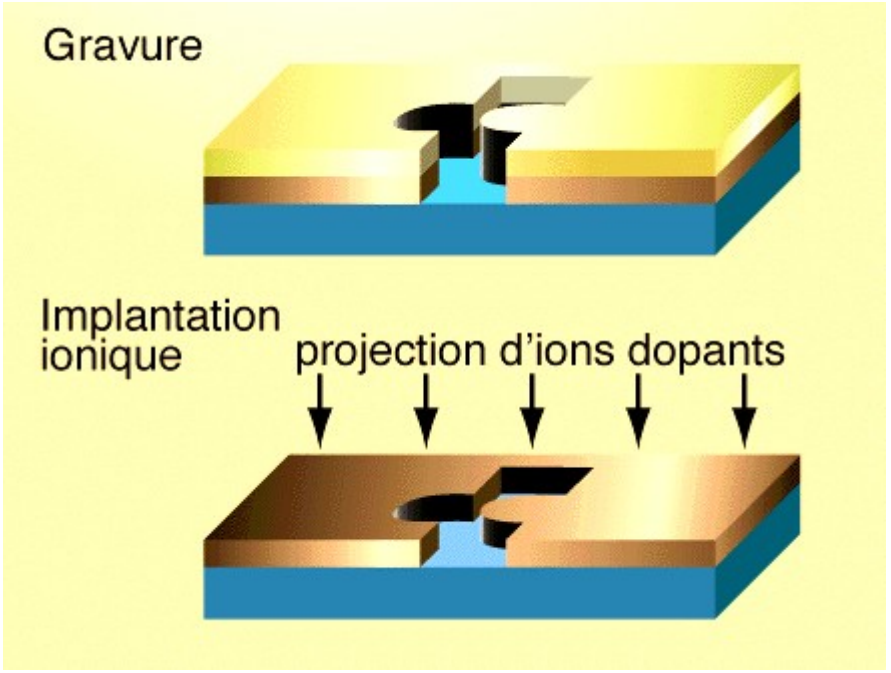
Historique : quelques dates (suite)

- 1970 : le langage C
- 1972 : premier microprocesseur 4 bits (Intel 4004) : 2800 transistors et 60000 instructions/seconde
- 1973 : le « Micral » (France, R2E)
- 1974 : microprocesseur 8 bits 6800 de Motorola
- 1974 : la carte à puce (Roland Moreno)
- 1975 : le 8080 d'Intel, le 6800 de Motorola
- 1975 : L'APPLE 2
- 1978 : le réseau TRANSPAC
- 1979 : le 68000 de Motorola (environ 70000 transistors)
- 1979 : le langage ADA
- 1981 : l'IBM PC
- 1983 : 1ère puce VLSI
- 1986 : microP 16 bits, la station SUN
- 1993 : le Pentium

Fabrication : technologie microélectronique

tendance : augmentation de la taille des puces et diminution de la largeur des interconnexions sur la puce





Évolution de la taille des puces

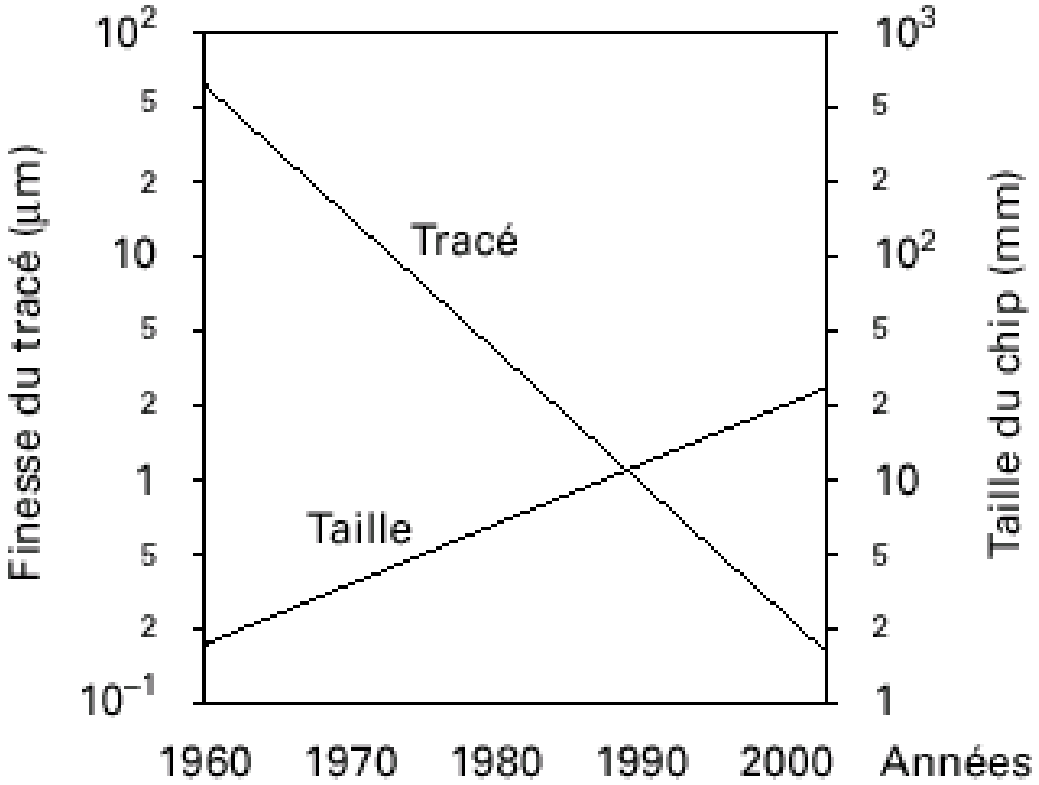


Figure 3 – Évolution de la technologie des microprocesseurs

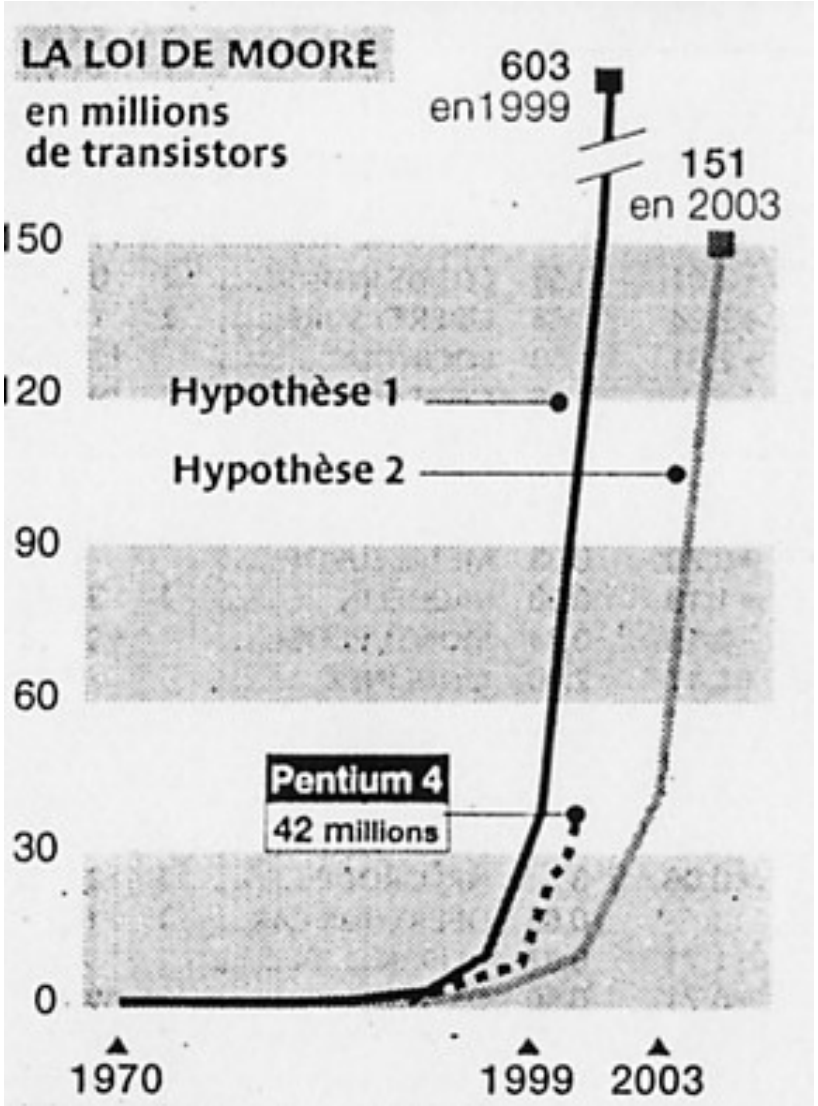
Évolution de la taille des puces

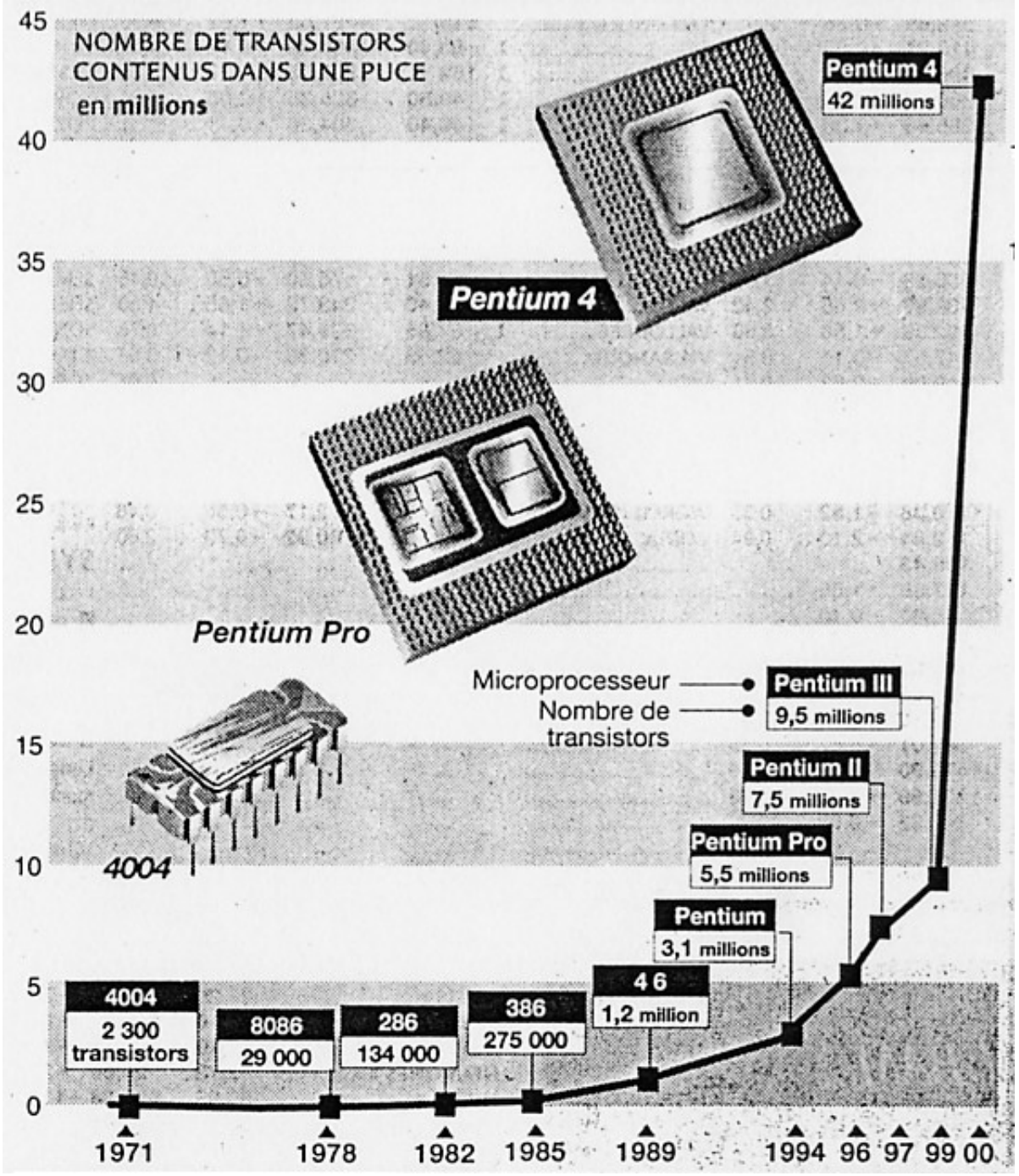
- Loi de Moore (1965),

Gordon Moore co-fondateur d'Intel :

  - Hypothèse 1 : nombre des transistors sur une puce doublé tous les 18 mois
  - Hypothèse 1 : nombre des transistors sur une puce doublé tous les 24 mois

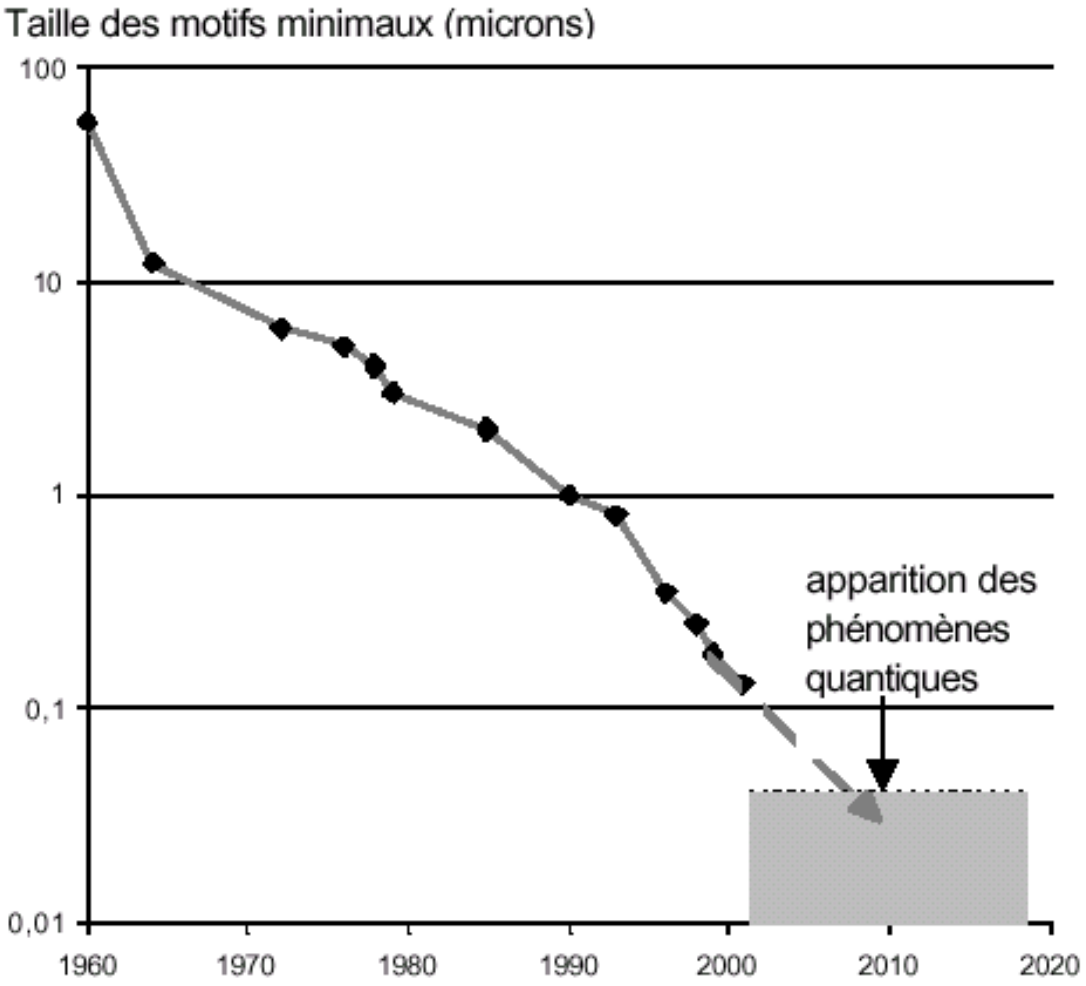
  Pour le moment, cette loi s'est toujours vérifiée





Évolution de la taille des puces

Évolution de la taille des puces



Evolution de la taille minimale des motifs en microélectronique

Evolution Technologique 1950 → 2000

Tableau I. — Les générations d'ordinateurs.

Génération	Dates	Technologie	Nouveau Produit	Nouvelles sociétés et machines
1	1950-59	Tubes à vide	Ordinateurs électroniques commercialisés	IBM 701, Univac I
2	1960-1968	Transistors	Ordinateurs meilleur marché	Burroughs 6500, NCR, CDC 6600 Honeywell
3	1969-1977	Circuits intégrés	Mini-ordinateurs	50 nouvelles compagnies DEC PDP-11, Data general Nova
4	1978-199?	LSI et VLSI	Ordinateurs personnels et stations de travail	Apple II, IBM-PC, Appolo DN300, Sun 2
5	199?-????	Traitement parallèle	Multiprocesseurs Multi-ordinateurs	Sequent Thinking Machine Co.

Tableau II. — Niveaux d'intégration et fonctionnalités.

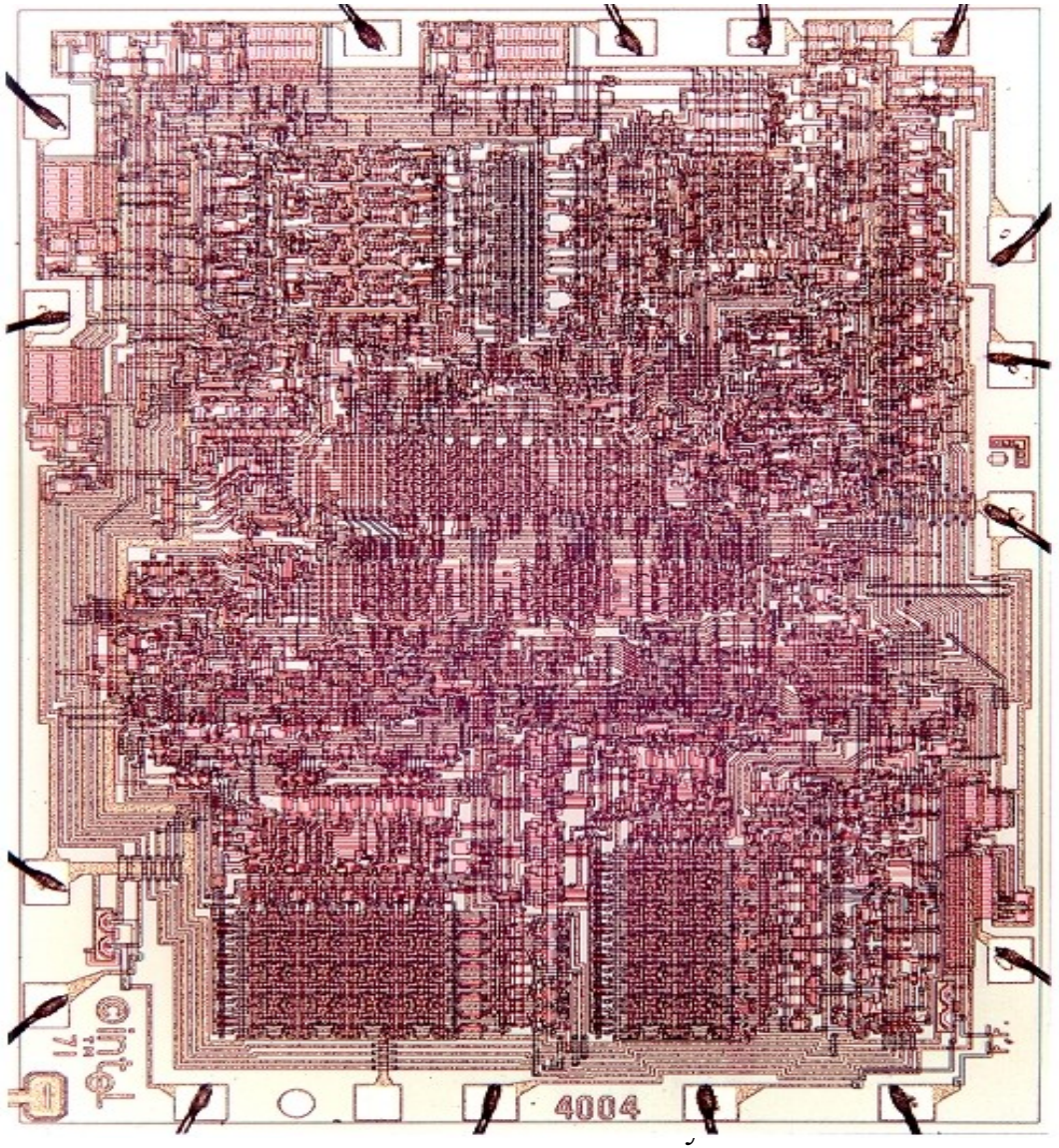
Année	Niveau d'intégration	Nombre de composants (processeur)	Fonctions	Technologie dominante	Dimension minimale de la lithographie en production µm	Taille de tranche pouce
1960	SSI	qq 10	Bascule	Bipolaire	10	1,5
1965	MSI	qq 100	Compteur	Bipolaire	8	2
1970	LSI	qq 1000	Microprocesseur	PMOS	6	2
1975	LSI*	6000	Microprocesseur	NMOS	5	3
1980	VLSI	40000	Micro-ordinateur	NMOS	3	4 5
1985	VLSI*	150000	Micro-ordinateur	CMOS	2	6
1990	VLSI**	10^6	Mini-ordinateur	CMOS	1	6
1992		$2,5 \times 10^6$	Mini-ordinateur	CMOS	0,8	6

* Les étoiles correspondent aux générations successives des technologies.

> 2000 > 10^7 CMOS < 0,2

2001 Pentium 4 = $42 \cdot 10^6$ transistors

Premier microprocesseur : le 4004 Intel en 1971



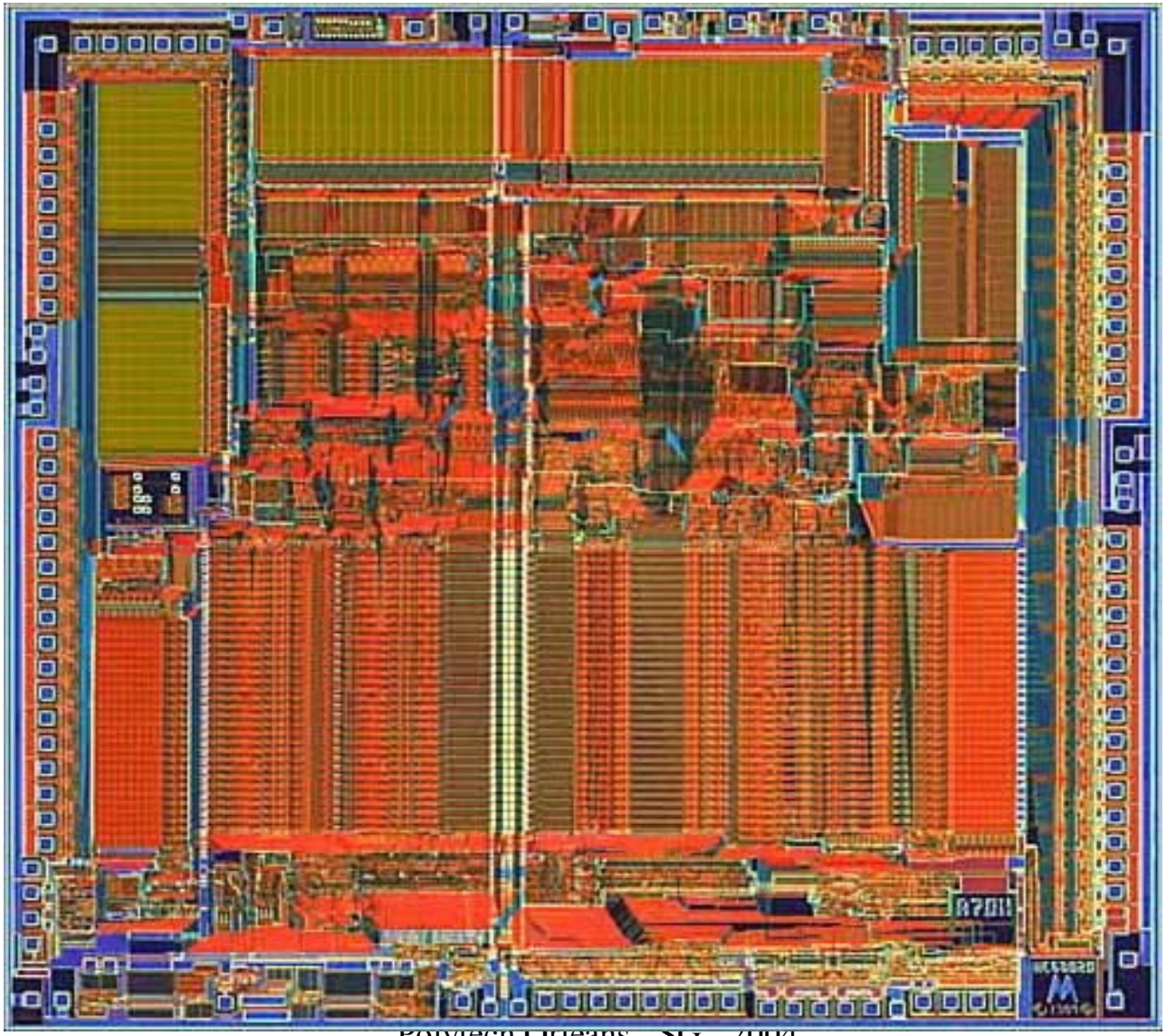
1971:
Le 4004 d'Intel fut le premier microprocesseur du monde (4 bits) :
2800 transistors et 60000 instructions/seconde

Microprocesseurs

Petit historique des microprocesseurs :

- **Les premiers (4 bits) :**
  - **PACE de NATIONAL SEMICONDUCTOR (N.S.)**
  - **4004 d' INTEL**
- **La première génération de microprocesseurs :**
  - **TMS9900 TEXAS INSTRUMENTS (T.I.)**
  - **CP16000 GENERAL INSTRUMENTS (G.I.)**
  - **9440 FAIRCHILD**
  - **Mn602 DATA GENERAL**
  - **8080 INTEL**
  - **6800 MOTOROLA**
  - **Z80 ZILOG**
- **La deuxième génération : années 1980**
  - **MC68000 MOTOROLA**
  - **8086 INTEL**
  - **Z8000 ZILOG**
  - **NS16000 NATIONAL SEMICONDUCTOR**

1984 : le microprocesseur 68020 de Motorola



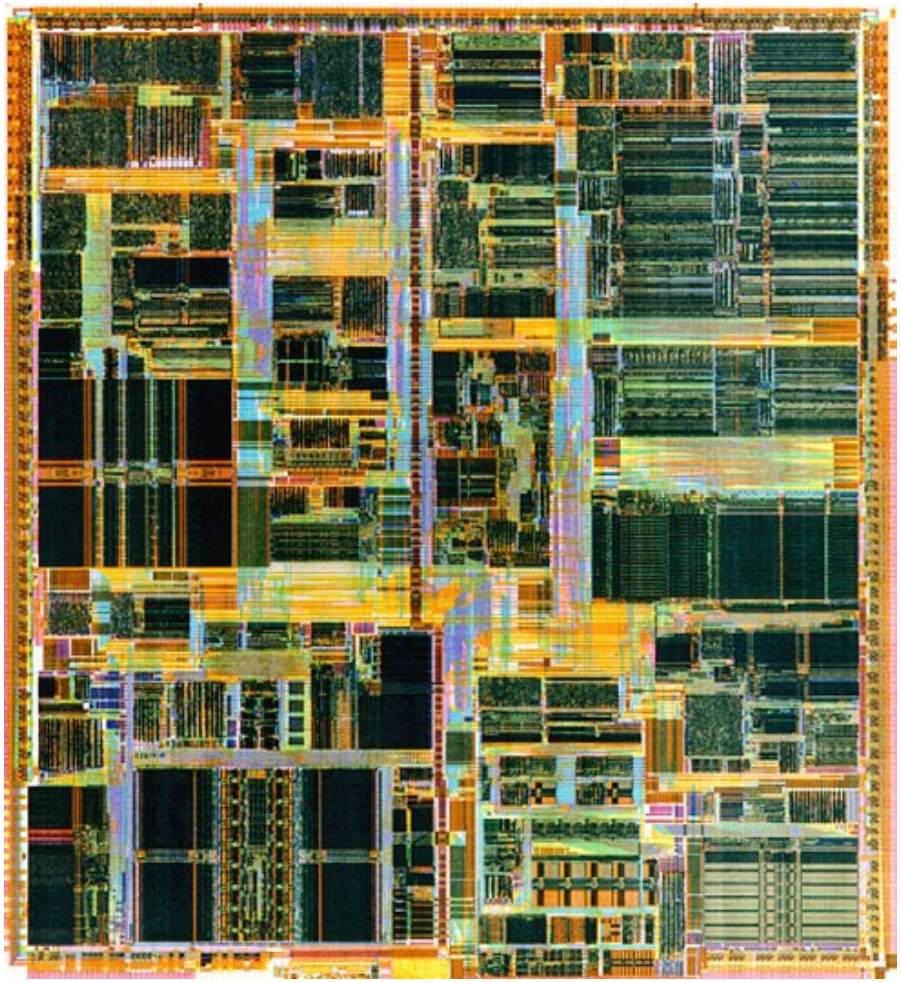
Exemple du microprocesseur 68000 de Motorola

- **Présentation, historique :**
  - né en 1979, technologie VLSI HMOS , environ 68000 transistors (existe en version CMOS MC68HC000)
  - environ 1,6 MIPS (100 MIPS pour 68060 à 66 MHz)
  - versions 8 à 16 MHz, gammes de températures diverses
  - microprocesseur 16 bits ‘ pseudo 32 bits ‘ :
  - bus de données 16 bits, bus adresse 24 bits
  - conçu pour opérer sur des mots de 32 bits : 16 registres généraux de 32 bits
  - architecture classique
  - modes utilisateur ou superviseur, instructions de haut niveau
  - gestion des interruptions matérielles et logicielles (*traps*) trés élaborée
  - configuration multi-processeur
  - secondes sources : THOMSON , ROCKWELL , HITACHI , PHILIPS
  - alimentation : 5V / 1,4W à 10 Mhz et 20°C

Exemple du microprocesseur 68000 de Motorola

- Bus de données et adresses séparés, compteur ordinal de 24 bits :
  ==>16 millions d'octets adressables sans multiplexage (8 mégamots de 16 bits)
  56 instructions seulement mais 14 modes d'adressage
  ==> plus de 1000 combinaisons possibles
- Adressage supportant 5 types de données :
  bits, digits BCD (4bits), octets, mots de 16 et 32 bits (longs)
- Adressage des périphériques comme de la mémoire (*memory mapped)*
- Boitier standard : 64 broches DIL ou 68 broches PLCC
- Tout type d'applications : calcul, gestion, ordinateurs individuels (ATARI ST, AMIGA, Mac), applications industrielles temps réel (système temps réel OS9, bus VME)
- Coupleurs spéciaux : MC68451 unité de gestion de la mémoire (MMU), MC68120 processeur d'entrée / sortie, MC68122 processeur de communication, MC68450 circuit d'accès direct à la mémoire (DMAC) MC68454 contrôleur de disque, MC68681 double contrôleur série, asynchrone MC68881 coprocesseur arithmétique flottant format IEEE, MC68230 interface parallèle / timer (PI/T)

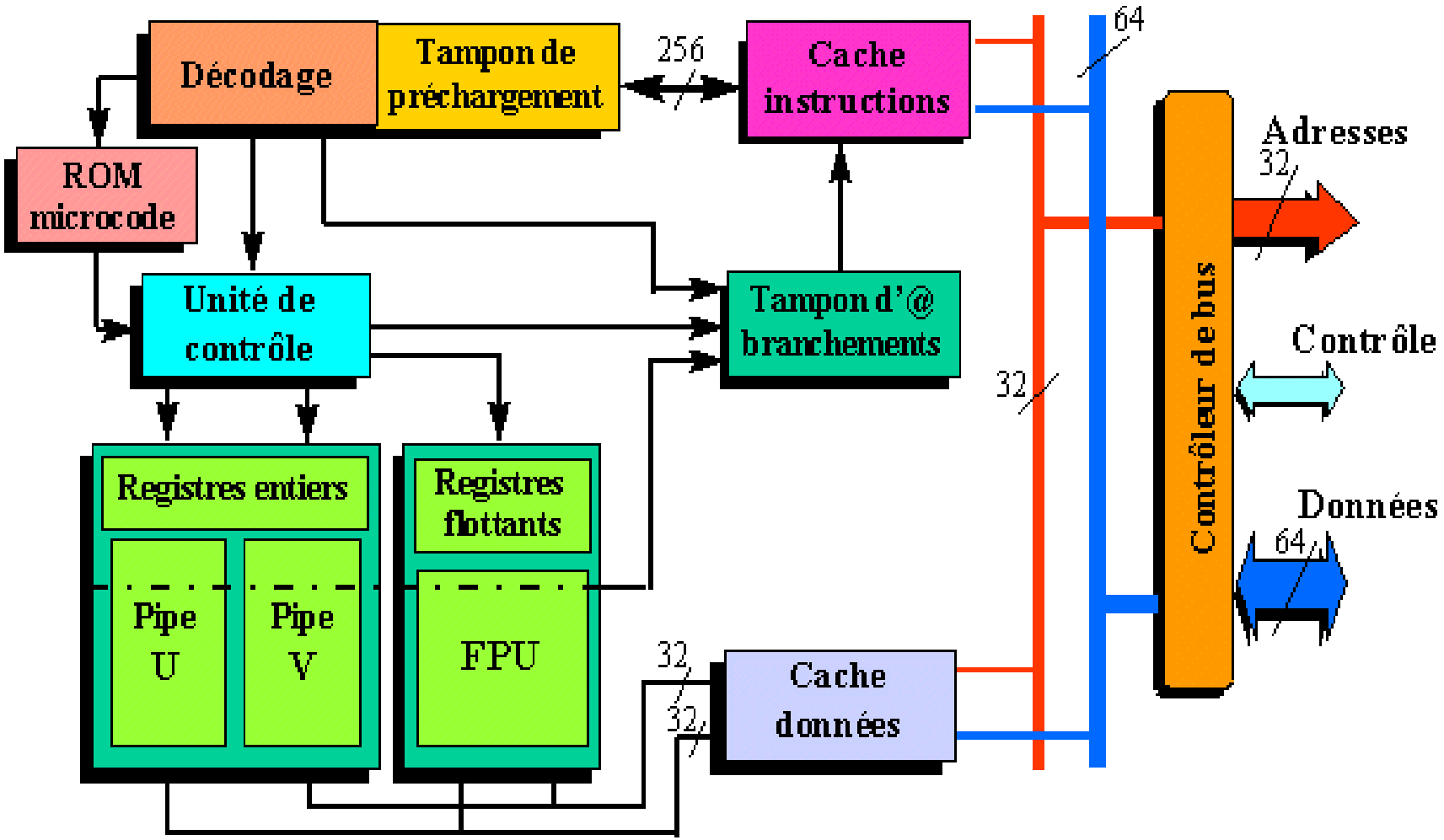
1997 : le Processeur Pentium 2 d'Intel



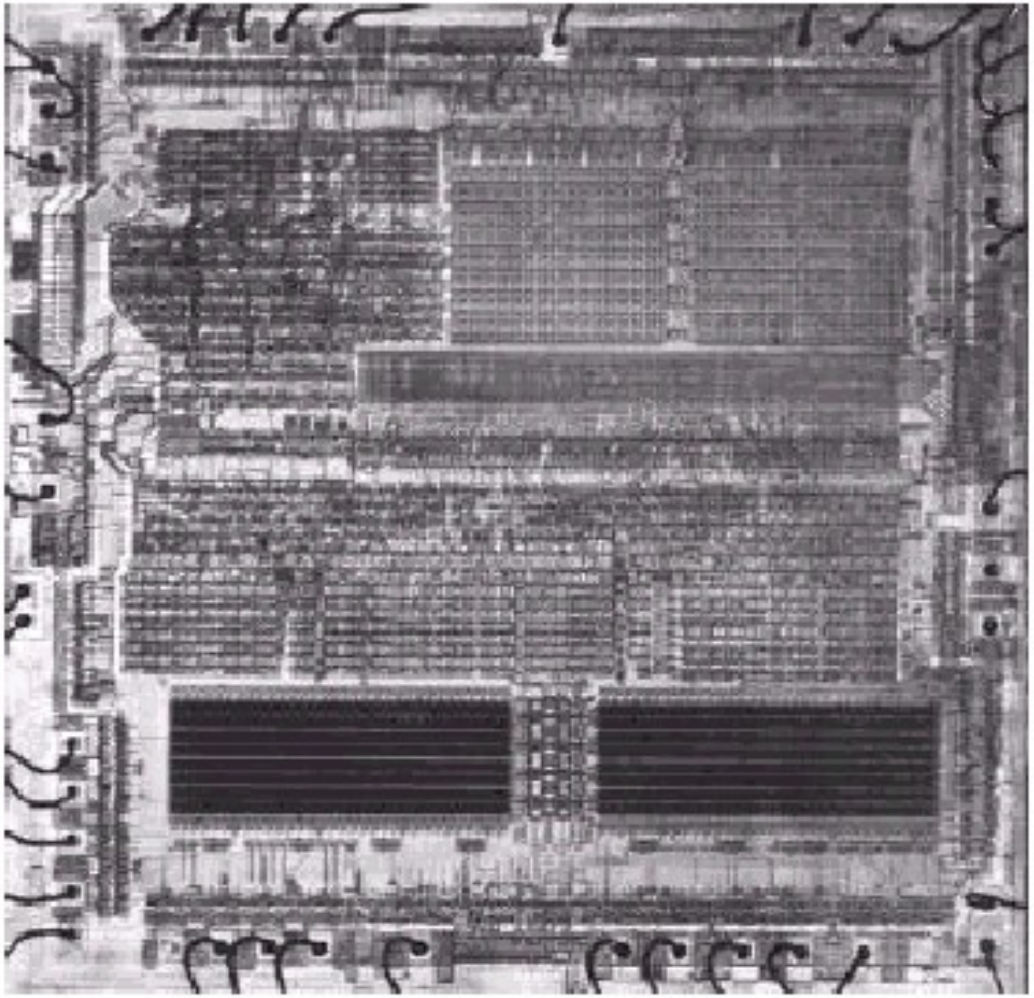
1997: Processeur Pentium® II

Le processeur Pentium® II (7.5 millions de transistors), intègre la technologie MMX™ d'Intel spécialement conçue pour améliorer le traitement de la vidéo, du son, et des données graphiques. Il est placé avec une mémoire cache rapide dans une cartouche S.E.C. (Single Edge Contact) connectée à la carte mère par un connecteur à une seule face et non à brochage multiple.

Le Processeur Pentium 2 d'Intel : Architecture

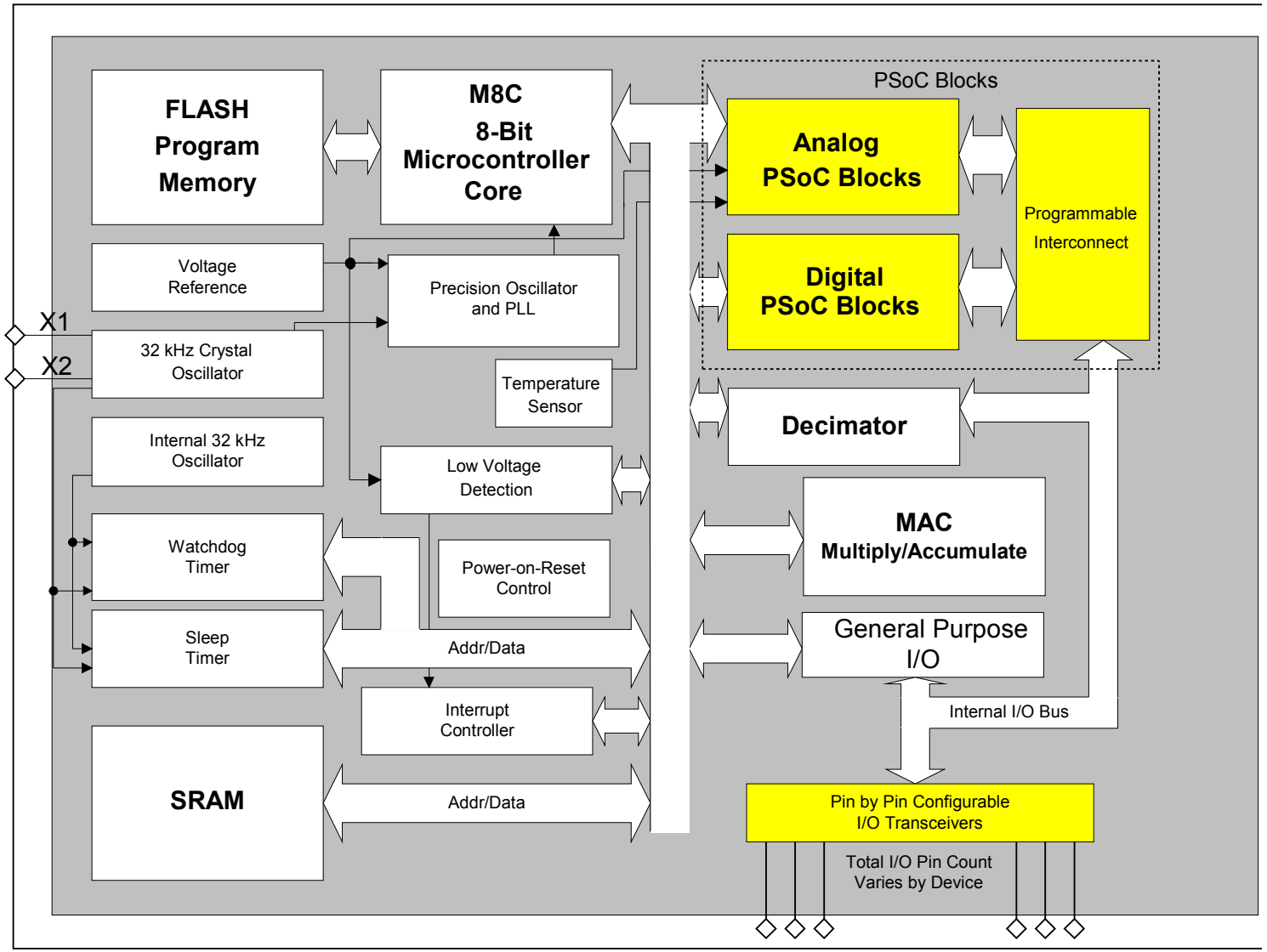


Microcontrôleur : le 80C51 d'Intel



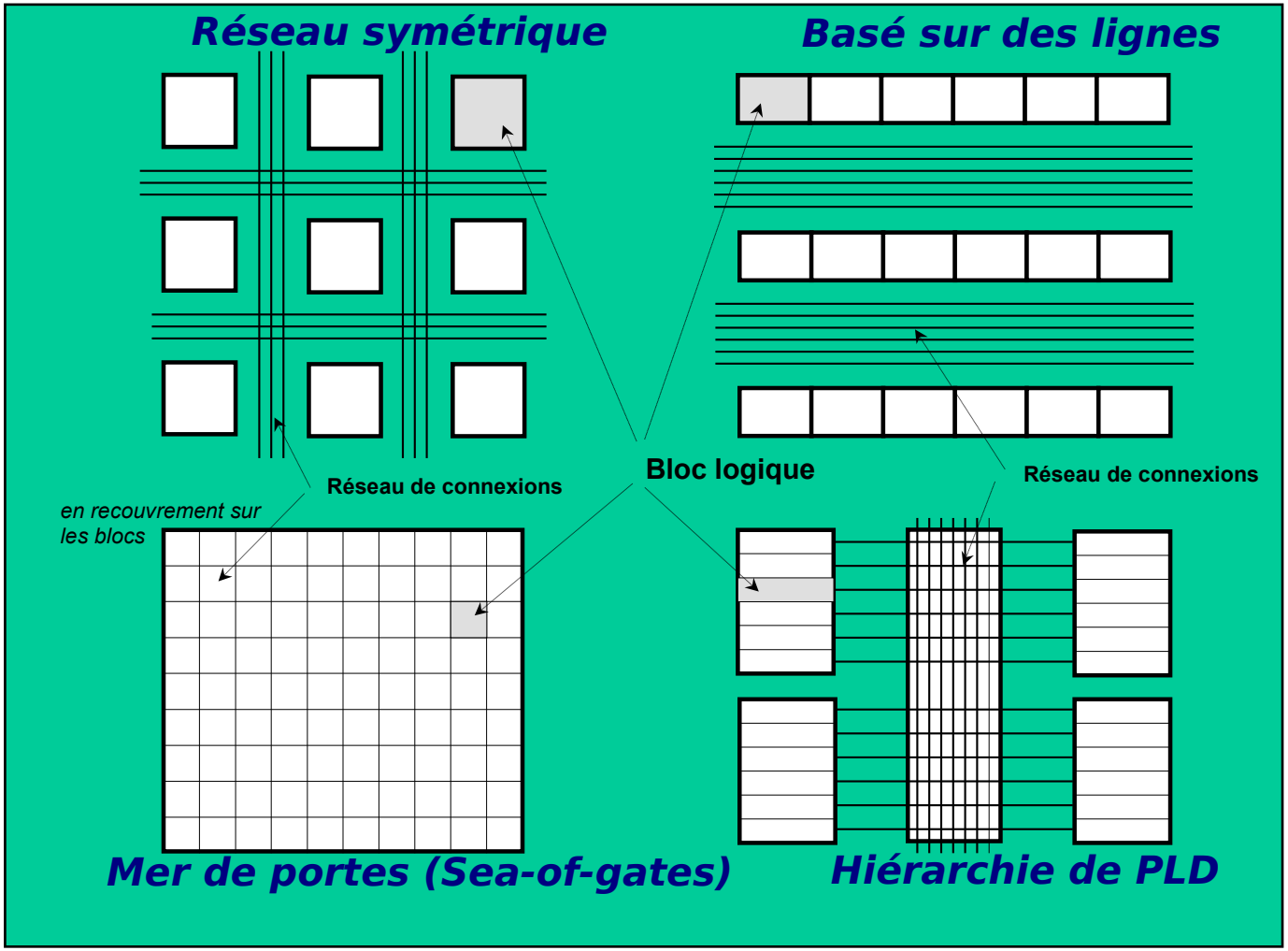
Microcontrôleur Intel 80C51

Composants mixtes: PSoc de Cypress



Conception logique moderne : utilisation de circuits logiques programmables

4 architectures de base, 2 grands types
(CPLD : décodage, contrôle, FPGA : traitement)



Les familles logiques (rappels)

- L'ordinateur ne comprend que deux valeurs : 0 et 1 (codage « binaire ») correspondant aux niveaux logiques de sa technologie interne et n'accepte que certains caractères.

  Technologie la plus basique : interrupteur ouvert ou fermé.
- Etats logiques « haut » et « bas » : en logique positive (0, 5 volts) TTL
  - « 0 » logique (« FAUX » en logique booléenne) = niveau bas (0 volt)
  - « 1 » logique (« VRAI » en logique booléenne)= niveau haut (5 volts)
- Selon le type de technologie logique utilisée on aura différentes plages de tension d'entrée (valeurs admissibles) correspondant à des niveaux logiques 1 ou 0
- Selon le type de technologie logique utilisée on aura différents niveaux d'immunité aux bruits (niveau maximal de bruit pouvant se superposer aux niveaux logiques sans déclancher intempestivement)
- Selon le type de technologie logique utilisée on aura différentes sortances (nombre d'entrées logiques pouvant être connectées sur une sortie).
- Selon le type de technologie logique utilisée on aura différentes plages de tension d'alimentation (les niveaux d'entrée et sortie dépendront bien sur de cette tension d'alimentation).

Les familles logiques (rappels)

- **Selon le type de technologie logique utilisée on aura différentes fréquences de fonctionnement (« vitesse ») maximales et différentes consommations électriques (typiques)**
- **Selon le type de technologie logique utilisée on aura différentes impédances d'entrée (capacité d'entrée de quelques pF et capacité de sortie) et courant de sortie maxi**
- **Attention à la compatibilité entre les différentes familles logiques / différentes tensions d'alimentation, …**
- **Vous pourrez également commuter des signaux analogiques (portes analogiques type 4066, multiplexeurs analogiques par exemple)**
- **Conception de cartes (interface mémoire / décodage) :**
  - **Lire les *data sheet* ! : valeur Max et Min, valeur typique**
  - **Utiliser les valeurs typiques (ou le pire cas) pour les conception des cartes (par exemple temps d'accès typique des circuits mémoires ou périphériques)**
  - **Vérifier que le tout fonctionne dans les cas les plus défavorables (min et max)**

Caractéristiques des familles logiques

Table 1: Transition rise/fall rates of logic circuits

Series	V_{cc} (V)	Vilmax (V)	Vihmin (V)	Vt (V)	dt/dv (ns/V)
SN74	4.75 - 5.25	0.8	2.0	1.4	100
SN74LS	4.75 - 5.25	0.8	2.0	1.4	50
SN74S	4.75 - 5.25	0.8	2.0	1.4	50
SN74ALS	4.5 - 5.5	0.8	2.0	1.4	15
SN74AS	4.5 - 5.5	0.8	2.0	1.4	8
SN74F	4.5 - 5.5	0.8	2.0	1.4	8
SN74HC	2.0	0.3	1.5	1.4	625
	4.6	0.9	3.15	2.25	110
	6	1.2	4.2	3.0	80
SN74HCT	4.5 - 5.5	0.8	2.0	1.4	125
74AC	3.0	0.9	2.1	1.5	10
	4.5	1.35	3.15	2.25	10
	5.5	1.65	3.85	2.75	10
74ACT	4.5 - 5.5	0.8	2.0	1.4	10
SN74BCT	4.5 - 5.5	0.8	2.0	1.4	10
SN74ABT	4.5 - 5.5	0.8	2.0	1.4	5/10
SN74LV	2.7 - 3.6	0.8	2.0	~ 1.5	100
SN74LVC	2.7 - 3.6	0.8	2.0	~ 1.5	5/10
SN74LVT	3.0 - 3.6	0.8	2.0	1.4	10

Table 3: Capacitances of digital circuits

		Output capacitance (pF)	
Series	Input capacitance (pF)	open collector output *)	Bus driver
SN74	3.0	5.0	-
SN74LS	3.5	3.5	5.0
SN74S	3.5	3.5	9.0
SN74ALS	2.0	4.0	5.0
SN74AS	4.0	-	10.0
SN74F	5.0	5.0	9.0
SN74HC	3.0	3.0	9.0
74AC	4.0	-	10.0
SN74BCT	6.0	-	12.0
SN74ABT	4.0	-	8.0
SN74LV	3.0	-	8.0
SN74LVC	4.0	-	8.0
SN74LVT	4.0	-	8.0

*) open collector output of gates and other circuits with low drive capability e.g. SN74xx03. Outputs of bus drivers with open collector have the same output capacitance as totem-pole (3-state) outputs.

Rappel : les principales familles logiques

Technologie TTL (Transistor - Transistor - Logic) à transistors bipolaires

- TTL standard 74XX
- TTL low power (faible consommation) 74LXX
- TTL schottky 74SXX
- TTL fast 74FXX
- TTL low power schottky 74LSXX
- TTL advanced schottky 74ASXX
- TTL advanced low power schottky 74ALSXX

Technologie CMOS (Complémentary - Métal - Oxide - Semiconducteur) à transistors Mos :

- CMOS rapides 74HCXX ou 74HCTXX
- CMOS classiques 74CXX ou 4000UB
- 4000B bufférisées (sorties amplifiées)

Rappel : les principales familles logiques

Tensions d'alimentation

74, 74L, 74S, 74F, 74LS : 5V (5%)

74ALS, 74AS, 74HCT : 5V (10%)

74HC : 2V à 6V

74C et série 4000 (B et UB) : 3V à 15V

Tendance pour les microprocesseurs = 2,5 V ou moins

Avantages des circuits CMOS

- faible puissance dissipée (< 10 mW par opérateur)
- large plage de tension de fonctionnement
- gamme de température de fonctionnement (de - 40 à 85°C)
- sortance élevée
- pour la série B : sorties bufférisés, vitesse de fonctionnement plus élevée que la série UB (CMOS classique), excellente immunité aux bruits

Rappel : les principales familles logiques

Fréquence maximale

- La vitesse de fonctionnement de la série CMOS est plus faible que celle de la série TTL 74LS.
- La fréquence maximale croît avec la tension, ce qui augmente d'une manière importante la dissipation de puissance dynamique.
- En CMOS la consommation en fonction de la fréquence de fonctionnement

Sortance : Correspond à la charge maximale que peut commander un opérateur.

Temps de propagation : Ce temps représente le temps que met le signal à franchir l'opérateur logique (retard). Ce temps ainsi que la fréquence maximale de l'horloge permet de juger de la rapidité des circuits, beaucoup plus faible en CMOS 4000 qu'en TTL.

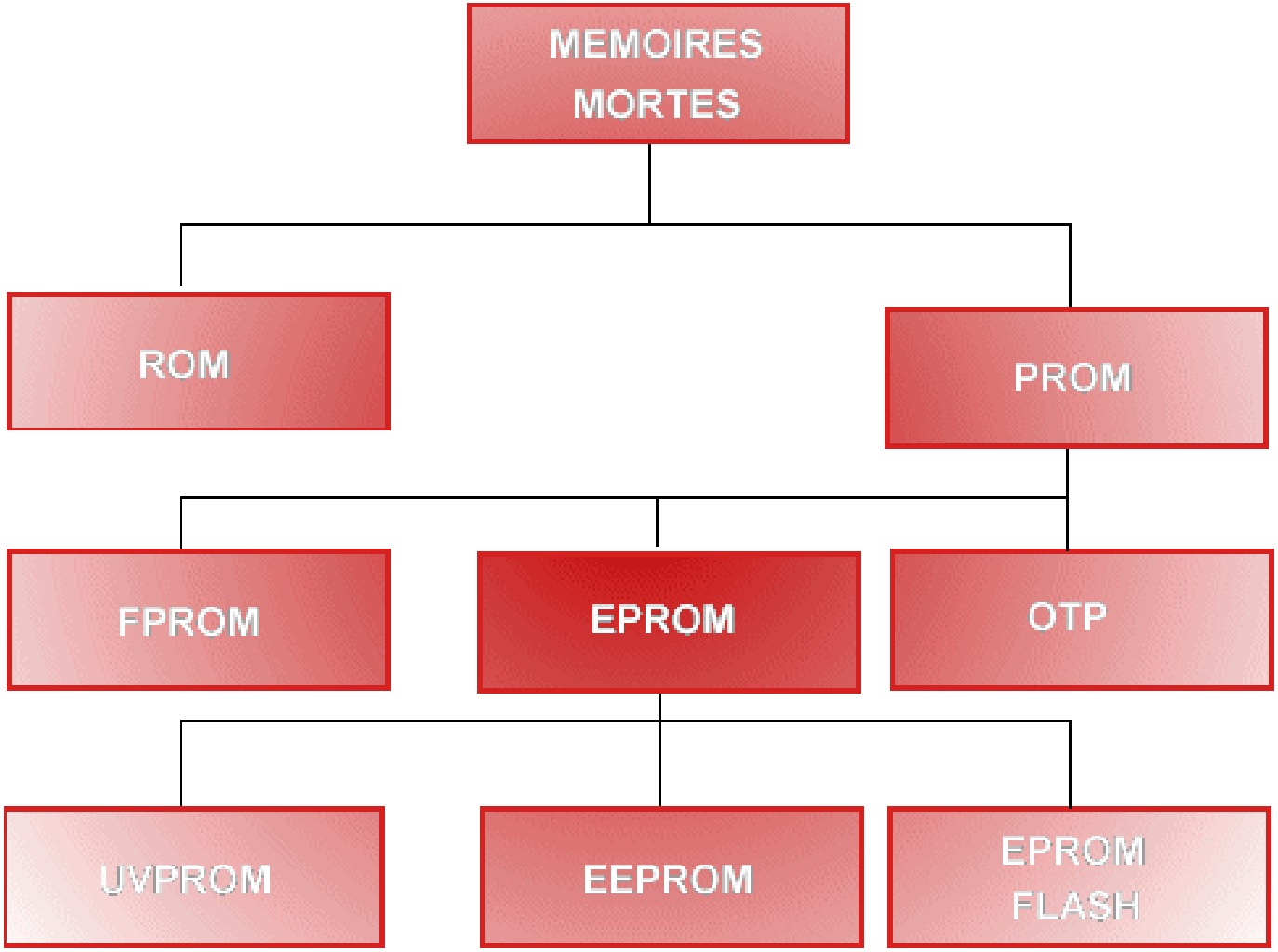
Pour connecter des familles différentes (TTL et CMOS par exemple), deux conditions sont nécessaires :

- à l'état haut : la tension VOH (voltage output hight), doit être supérieure ou égale à la tension d'entrée VIH (voltage input hight), du circuit commandé.
- à l'état bas : la tension VOL (voltage output low), doit être inférieure ou égale à la tension d'entrée VIL (voltage input low), du circuit commandé.

Les grands types de mémoires

- **Mémoires vives (RAM) « volatile » : *Random Access Memory***
  - **RAM statiques SRAM :**
    - Bascules D = mémoire élémentaire
    - Signaux de lecture (RD), écriture (WR), sélection du boîtier CS, adresse (A0 à An) et données (D0 à Dx)
    - Attention aux temps d'accès pour le décodage (adress access time maximum)
  - **RAM dynamiques DRAM** : capacité plus grande que les RAM statiques mais mise œuvre plus complexe
    - Stockage des informations sous forme capacitive (capa drain-source d'un transistor MOS) : nécessite un rafraichissement périodique (cycles RAS/CAS)
  - cf mémoire sur cartes à puces, RAM vidéo, RAM sauvegardée, …
- **Mémoires mortes (ROM) « permanente » : *Read Only Memory***
  - **ROM** : programmées à la fabrication (par masque, pour grandes séries)
  - **PROM** : programmable
    - Une fois : One Time Programmable OTP (faible coût, petites séries ou prototypes)
    - Plusieurs fois : programmable et effaçable
      - » **UVPROM** : effaçable par exposition aux UV, programmation hors carte
      - » **EEPROM** : Effacement (octet par octet) et pro. *In situ* (sur la carte) électriquement
      - » **FLASH** : Effacement (totalité) et pro. *In situ* (sur la carte) électriquement
  - Utilisation d'un « programmateur de PROM » (par ex. DataIO) avec algorithme de programmation intelligent

Les type de mémoires : mémoires mortes



Codage de l'information

- **L'unité de codage est le BIT (binary digit) : 0 ou 1**
- **Codage : Binaire**
  - Binaire naturel : 0 ou 1
    - ***Ex: 27d devient 0001 1011 en binaire (25d = 16d + 8d + 1d = $2^4 + 2^3 + 2^1 + 2^0$)***
  - Le « BCD » Binary Coded Decimal : 0 à 9 codés sur 4 bits
    - ***Ex: 27d devient 0010 0111 en BCD***
  - Code octal (base 8)
    - ***Ex: 27d devient 33o (25d = $3x8^1 + 3x8^0$)***
  - Hexadécimal (base 16) : 0 à 15 (en décimal) codés de « 0 » à « F » (en hexa)
    - ***Ex: 27d devient 1bH (27d = 1x16 + 11)***
- **Le traitement de l'information utilise le MOT**
  - mot de 4 bits « quartet »
  - mot de 8 bits « octet » (byte) : de 0 à 255, codage des caractères en code ASCII, par ex. : RC (retour chariot) : 0Dhexa, Line Feed (saut de ligne) 0Ahexa, nombres de 0 à 9 codes 30h à 39h, ….
  - mot de 16 bits : codage de nombres entiers non signés de 0 à 65535, par ex.

binaire	décimal	hexadécimal
0000	0	0
0001	1	1
0010	2	2
0011	3	3
0100	4	4
0101	5	5
0110	6	6
0111	7	7
1000	8	8
1001	9	9
1010	10	A
1011	11	B
1100	12	C
1101	13	D
1110	14	E
1111	15	F

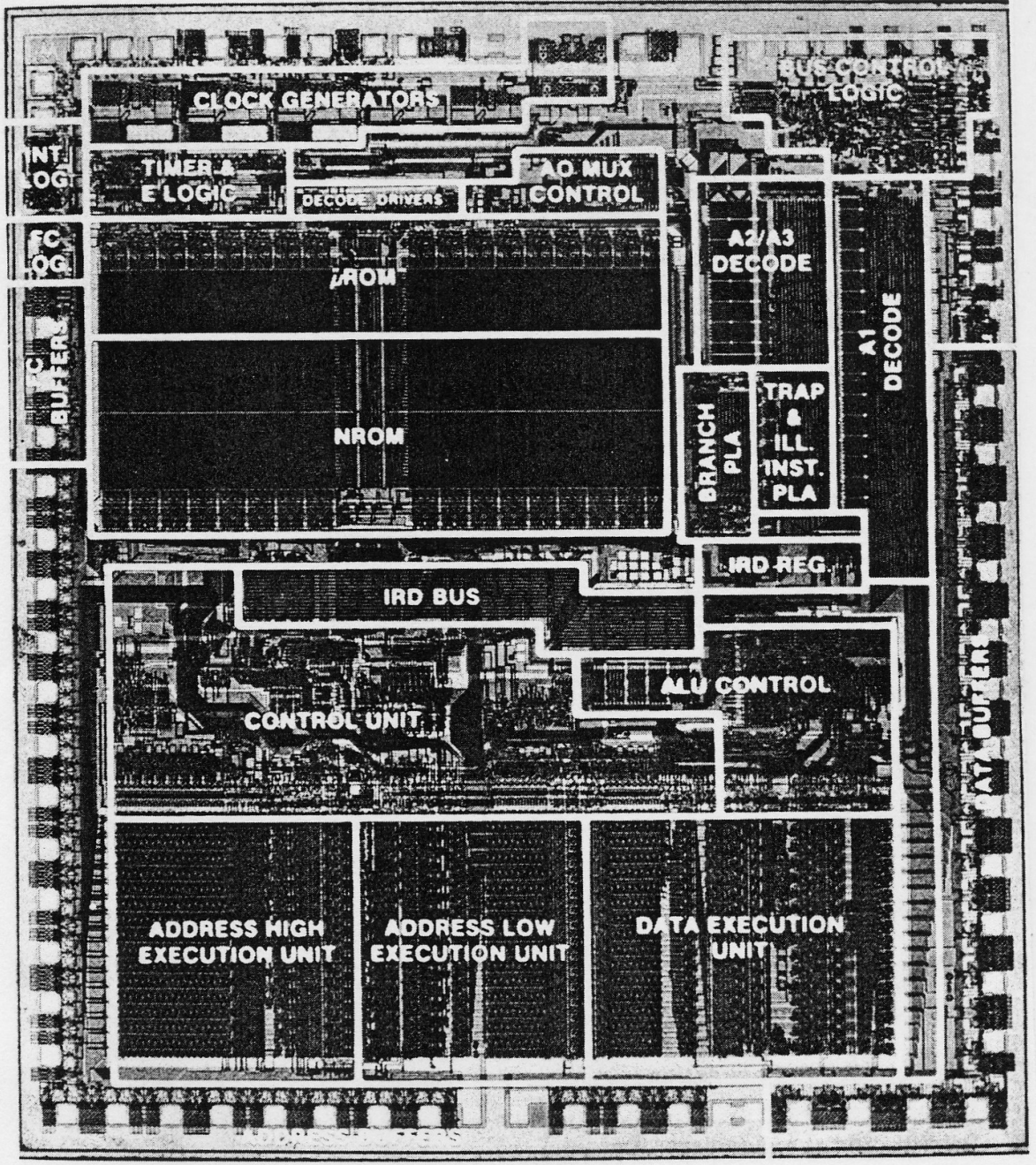
Codage des
nombres non signés :
binaire (base 2)
octal (base 8)
décimal (base 10)
hexadécimal (base 16)

Codage de l'information (suite)

- **Codage des nombres entiers signés**
  - Le plus simple : 1 bit de signe + valeur absolue
  - Complément à 2 :
    - Changement de signe en C2 : complémenter tous les bits et ajouter 1
  - Binaire décalé (sortie des convertisseurs analogiques-numériques, par ex.)
    - Passage du C2 en binaire décalé : complémenter le bit de poids fort (par exemple OU exclusif en 8 bits : XOR 80H)
- **Codage des nombres flottants (signés)**
  - 32 bits (4 octets) : 1 Bit de signe + 7 bits mantisse + 24 bits exposant
  - Format 32 bits IEEE (float) ou format long 64 bits

Codage des nombres signés en binaire

entier relatif	valeur absolue	binaire	complément à 1	complément à 2
-8	8	1000	0111	1000
-7	7	0111	1000	1001
-6	6	0110	1001	1010
-5	5	0101	1010	1011
-4	4	0100	1011	1100
-3	3	0011	1100	1101
-2	2	0010	1101	1110
-1	1	0001	1110	1111
0	0	0000	0000	0000
1	1	0001	0001	0001
2	2	0010	0010	0010
3	3	0011	0011	0011
4	4	0100	0100	0100
5	5	0101	0101	0101
6	6	0110	0110	0110
7	7	0111	0111	0111



Structure interne du Microprocesseur 68000
(documentation MOTOROLA)

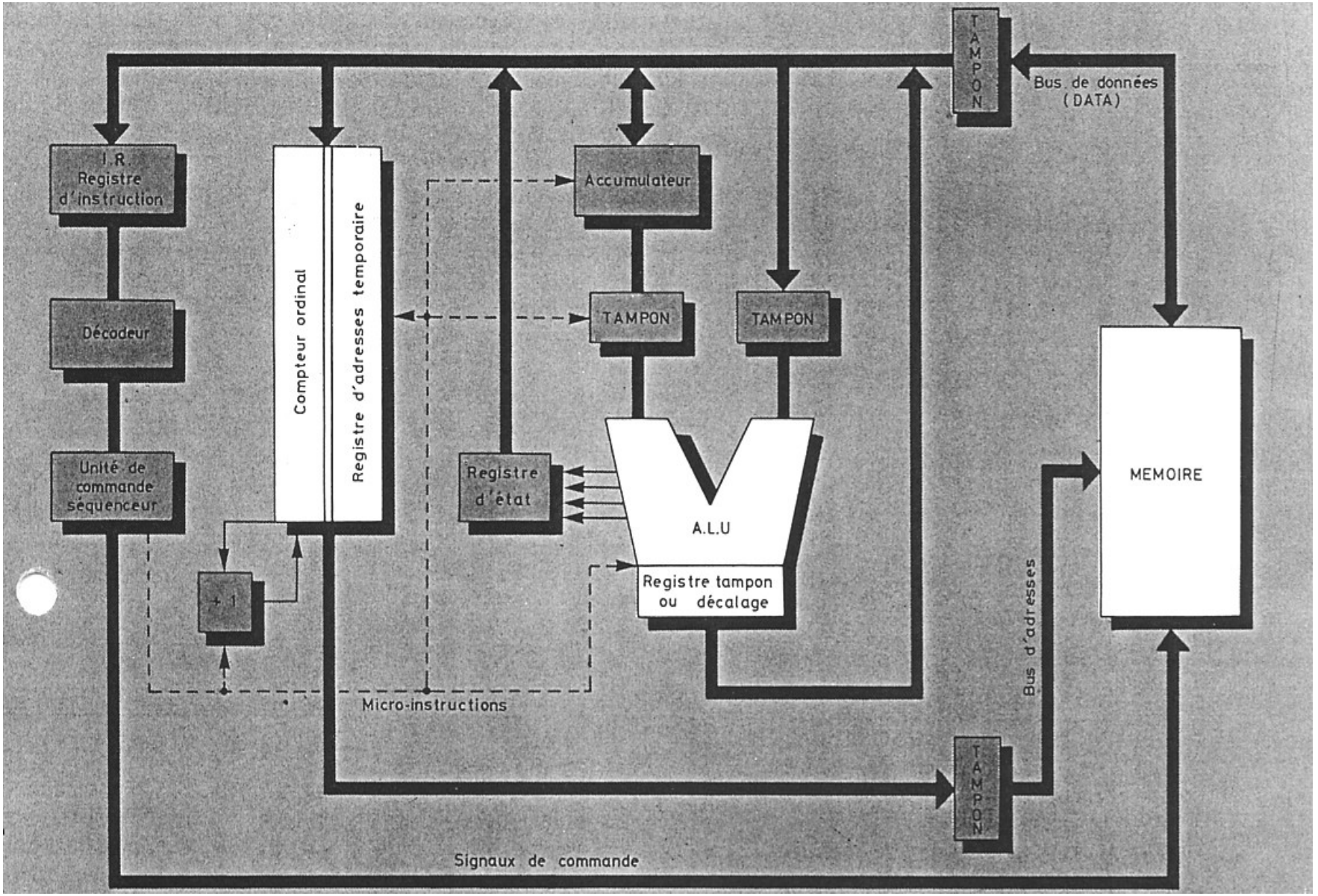
Exemple de la structure interne du microprocesseur « 68000 » de Motorola

Architecture interne du microprocesseur

- Unité centrale (UC, CPU) cadencée par une horloge
  - Unité Arithm. et Logique, unité de commande, registres
- 3 bus distincts pour véhiculer les informations :
  - le **BUS DONNEES** (*data bus*) : la taille du bus données permet de distinguer les microprocesseurs 8 bits, 16 bits , ...
    - Transfert des données de la mémoire programme (ROM) vers l 'UC
    - Transfert des données de ou vers la mémoire vive (RAM)
    - Transfert des données de ou vers les ports d 'entrée-sortie
  - le **BUS ADRESSES** (*adress bus*) : la taille de ce bus détermine l 'espace mémoire accessible directement
    - adresse de l 'instruction à lire en mémoire programme
    - adresse de la données à lire ou écrire en mémoire donnée
    - adresse du périphérique à lire (entrée) ou à écrire (sortie, commande)
  - le **BUS CONTROLE** (*control* bus):
    - sélectionne et oriente l 'accès aux mémoires ou périphériques (RD, WR, ALE, …)
    - interruptions, signaux DMA, reset, ..

Architecture interne (suite)

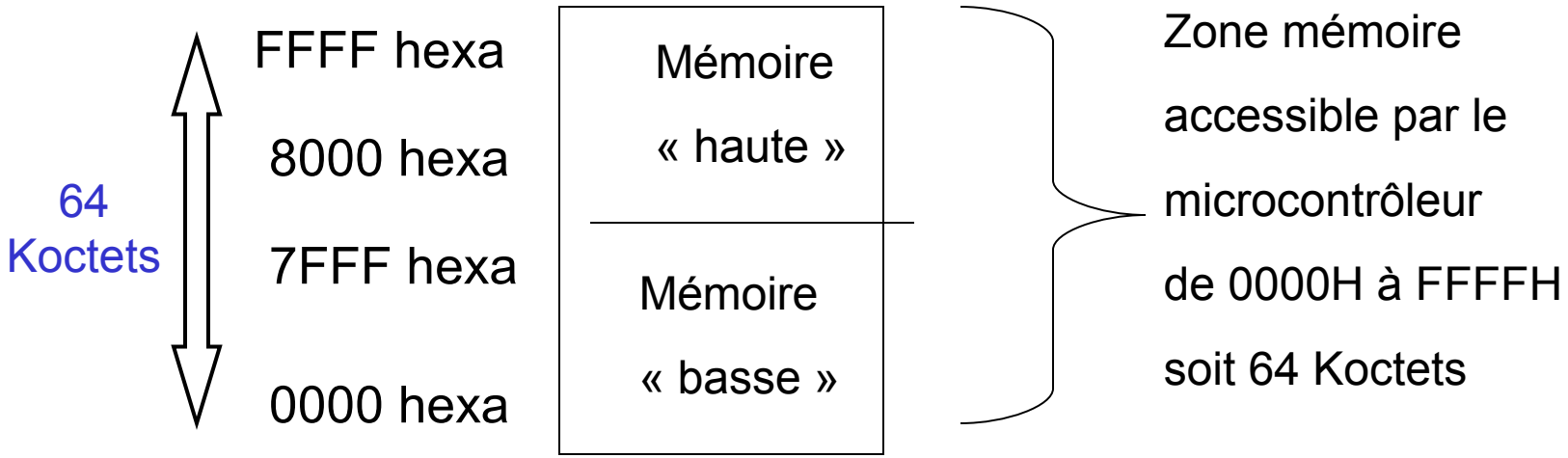
- Certains bus et certaines lignes du bus contrôle utilisent des buffers trois-états (tri-state) : lecture (in), écriture (out), haute impédance (Z)
  - bus données : trois états (« tri-state »), bi-directionnel
  - bus adresse : mono-directionnel
- **L'état haute impédance** est un état électrique, et ne peut en aucun cas être considéré comme un état logique. Dans un état haute impédance noté Z (niveau haute impédance), les sorties peuvent être considérées comme électriquement déconnectées, (circuit ouvert) du reste de la structure. La mise en état « haute impédance » se fait à l'aide d'une entrée spécifique du circuit (validation). Les circuits 3 états sont surtout utilisés dans la gestion des BUS.


I.R.
Registre d'instruction
Décodeur
Unité de commande séquenceur
Compteur ordinal
Registre d'adresses temporaire
+1
Accumulateur
TAMPON
TAMPON
Registre d'état
A.L.U
Registre tampon ou décalage
Micro-instructions
TAMPON
Bus de données (DATA)
MEMOIRE
Bus d'adresses
TAMPON
Signaux de commande

Architecture interne (suite)

- Pour limiter le nombre de broches du composant électronique, certains constructeurs utilisent des **bus adresses et données multiplexés.** Les mêmes « fils » véhiculent des signaux AD ou DATA, et un démultiplexage (74HC573 par exemple) doit être réalisé sur la carte en utilisant des signaux de contrôle (*Adress Latch Enable ALE*).

- On représente le **plan mémoire** (*memory MAP*) de la carte microprocesseur ou microcontrôleur, par exemple :

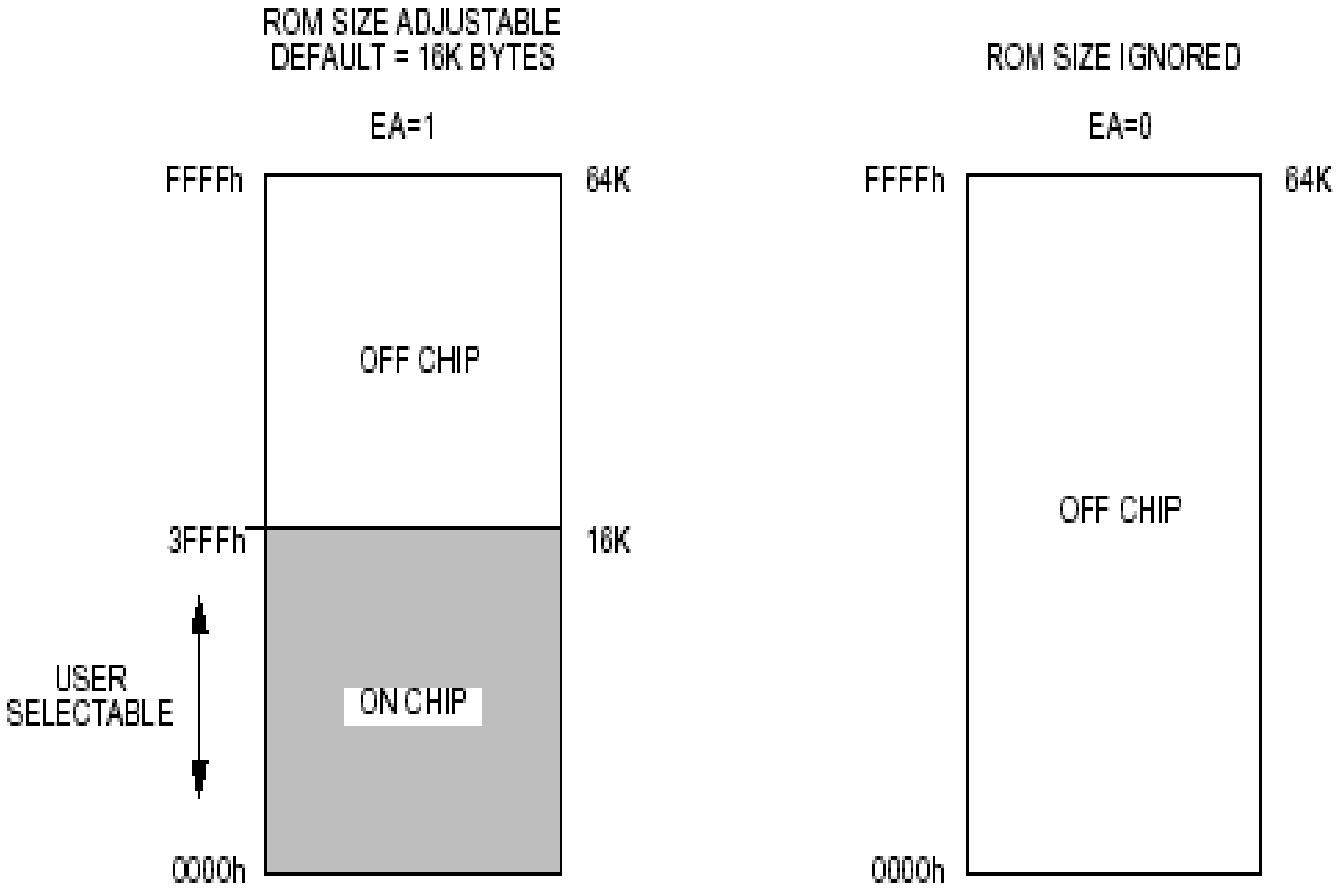


Architecture interne (suite)

- Chaque microprocesseur possède des registres internes :
  - l ’**accumulateur** ACC, ou ACCU : il joue un rôle fondamental dans les opérations arithmétiques et les transferts de données, notamment pour les microP ou microC 8 bits
  - les **registres temporaires et registres de travail** : en nombre très différent selon le type de microP
  - des **registres d ’état** (status register SR) : bits pour interruptions, dépassement (overflow), retenue (carry, C), parité (P), signe (S), ..;
  - le **pointeur de programme ou compteur ordinal** (Program Counter, PC) : pointe l ’adresse de la prochaine instruction à exécuter (à rechercher en mémoire)
  - le **pointeur de pile** (Stack Pointer, SP) : permet de gérer la pile (stack) :
    - en général la pile est une zone mémoire vive qui permet de stocker des informations (retour de sous-programmes, sauvegardes de registres, interruptions).
    - la pile est gérée en mode LIFO (Last In First Out)

Exemple de mapping mémoire

ROM MEMORY MAP Figure 2



Le registre d'état SR du 68000

→ Registre sur 16 bits :
- 1 octet SUPERVISEUR
- 1 octet UTILISATEUR (CCR)

→ Bits indicateurs d'état (CCR) :
- → sont positionnés par l'instruction qui précède (qui vient d'être exécutée)
- → sont utilisés lors de TESTS, COMPARAISONS, SAUTS CONDITIONNELS, CALCULS, ...

bit C : CARRY retenue / bit poids fort
bit Z : ZERO Z=1 si résultat nul
bit N : NEGATIF N=1 <0 (en C2)
N=0 ≥0
bit V : Overflow débordement
valeur du résultat > valeur maxi codable
bit X : indicateur d'extension
(opérations en multiprécision)

→ 3 bits de codage de priorité des interruptions
- ⇒ 7 niveaux de codage d'IT
- ⇒ "Masque" d'interruption

→ bit S : Mode Superviseur / Utilisateur
S=1 Superviseur (après Reset ou interrupt.)

→ bit T : Mode Trace
Single Step (Pas à Pas)

Registre d'état (status register SR) Exemple du 68000

Architecture interne du microprocesseur
Unité Centrale

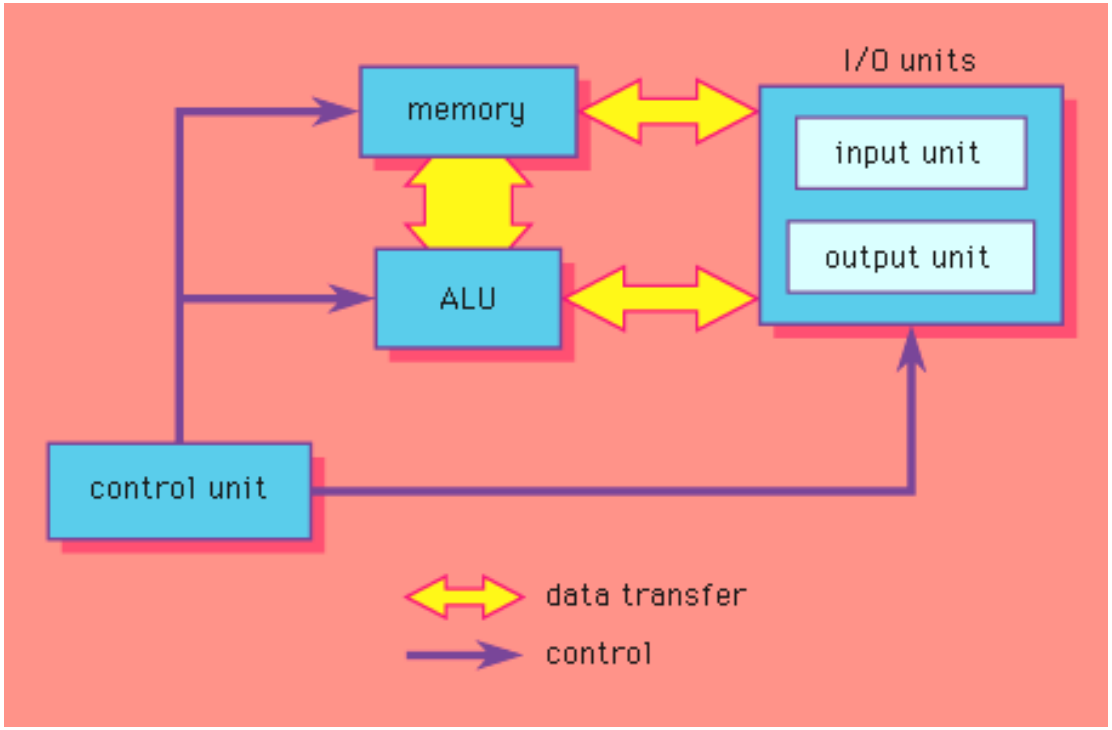
- Le compteur programme (PC : *program counter*) contient à chaque instant l'adresse mémoire à laquelle se trouve l'instruction à exécuter
- Unité centrale (UC, CPU) cadencée par une horloge : lire / décoder / éxécuter une instruction, et préparer l'instruction suivante
  - Unité de commande : charge (lire) une instruction : cycle « fetch » et la décode : cycle « décode ». Elle séquence les instructions.
  - Unité Arithm. et Logique (ALU) : exécute l'instruction,
- Le CPU incrémente le PC pour pointer sur l'instruction suivante à exécuter et le cycle recommence.
- Registres : mémoires internes à accès très rapide permettant de stocker les informations ou les résultats temporaires.
  - accumulateur ACC, registres d'état SR, compteur de programme PC, compteur de pile SP , registres d'index, …

Architecture interne (suite)

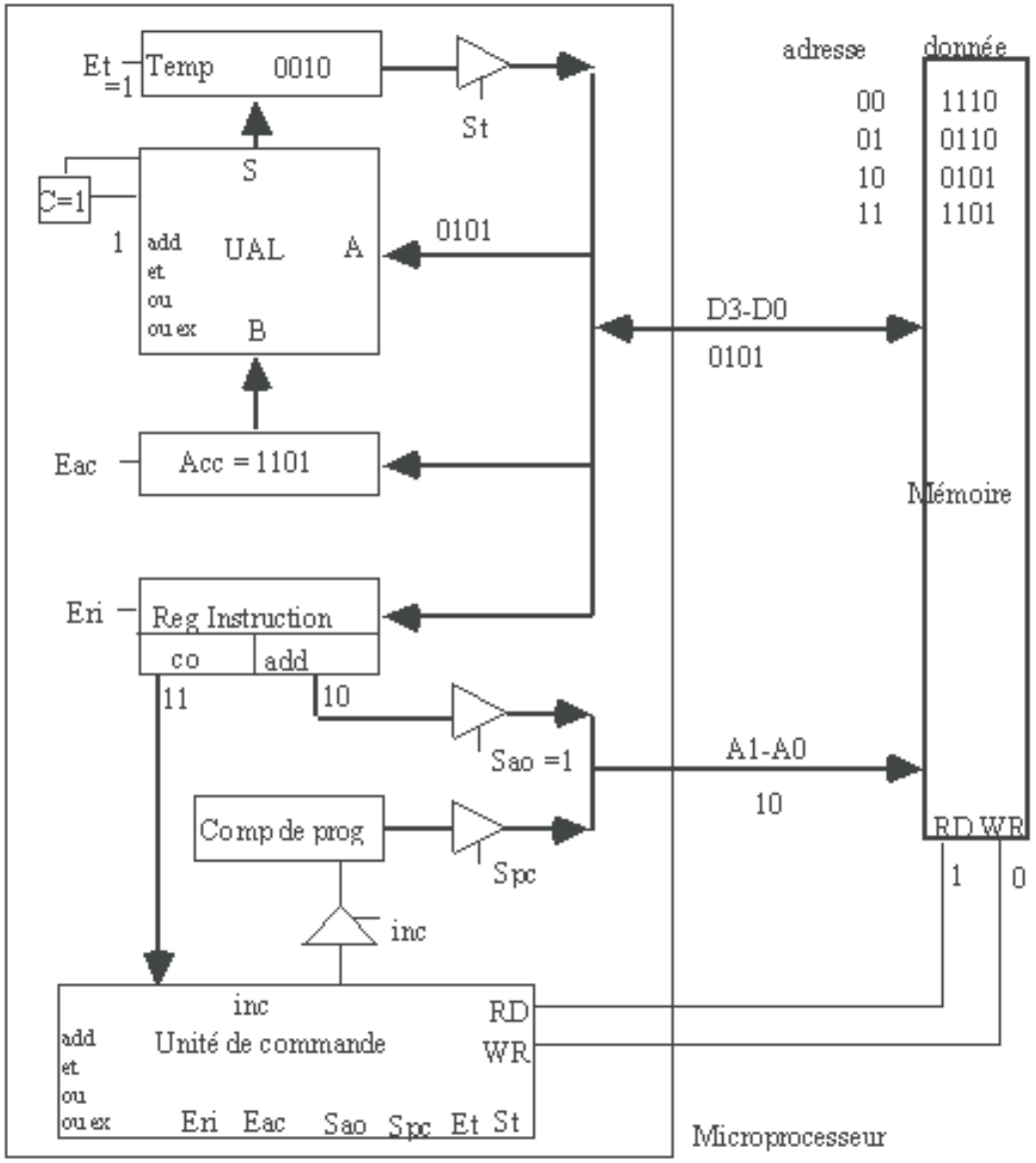
– l ’Unité Arithmétique et Logique (ALU) : effectue les opérations arithmétiques et logiques de base.

Elles est composée essentiellement de :

- Additionneur n bits, soustracteur
- portes et fonctions logiques (décalages, rotations)



Cycles d'exécution d'une instruction :

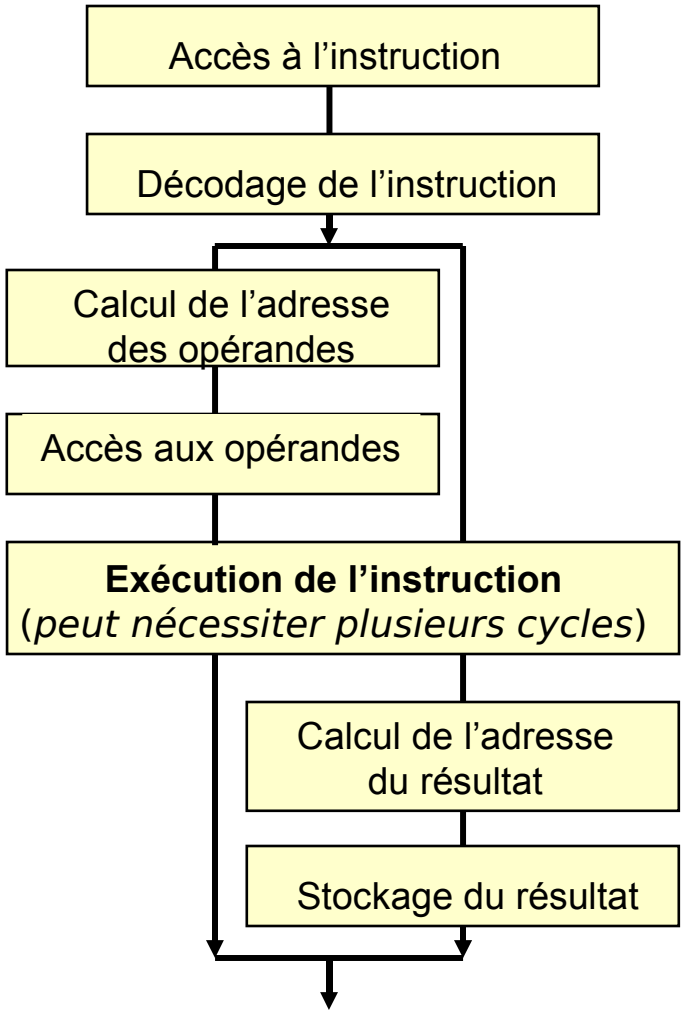


Cycles :

- Lecture (FETCH)
- Décodage (DECODE)
- Exécution (EXECUTE)
- Préparation de l'instruction suivante

Architecture interne du microprocesseur

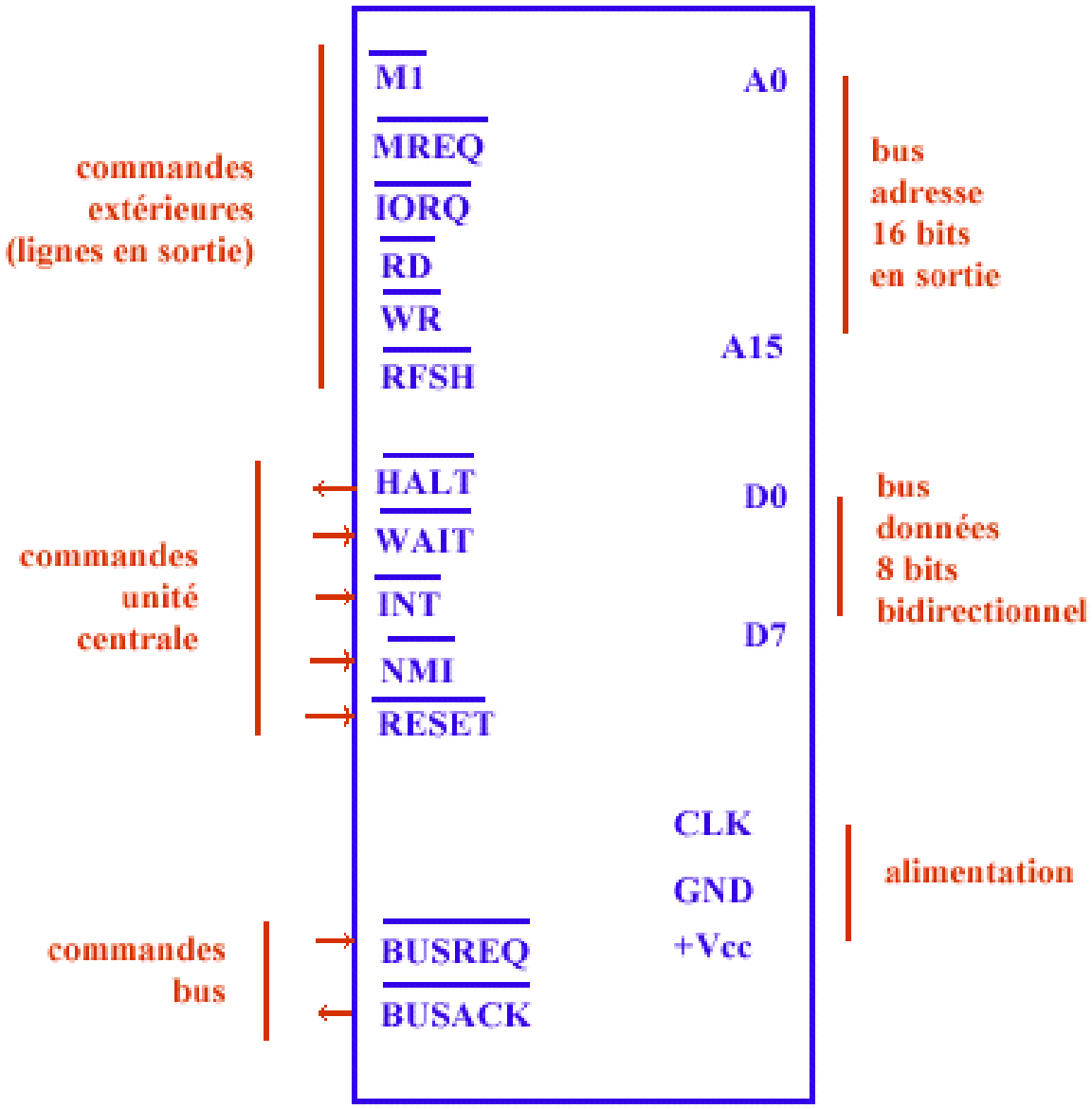
Décodage d'une instruction



Lecture

Décodage

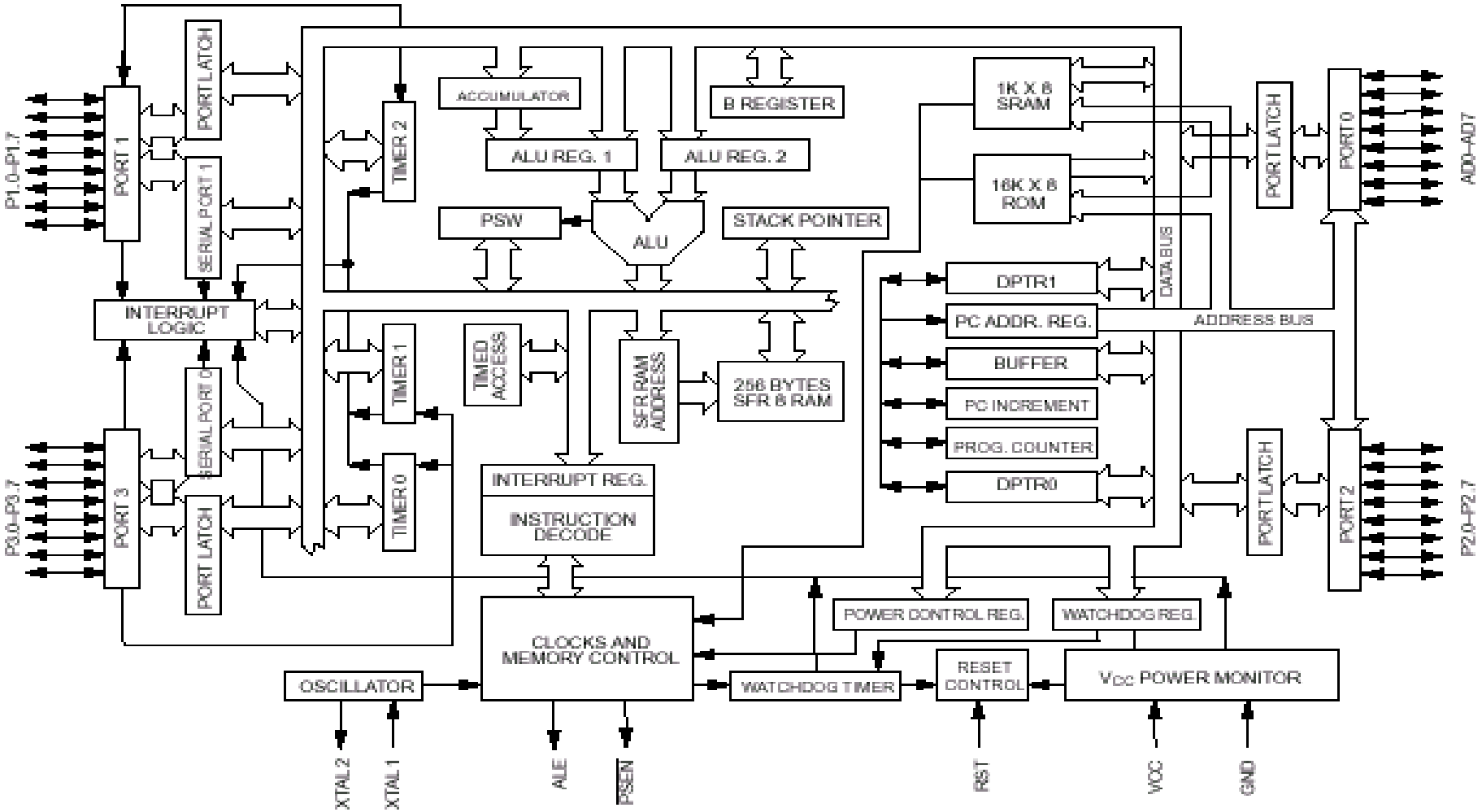
Exécution



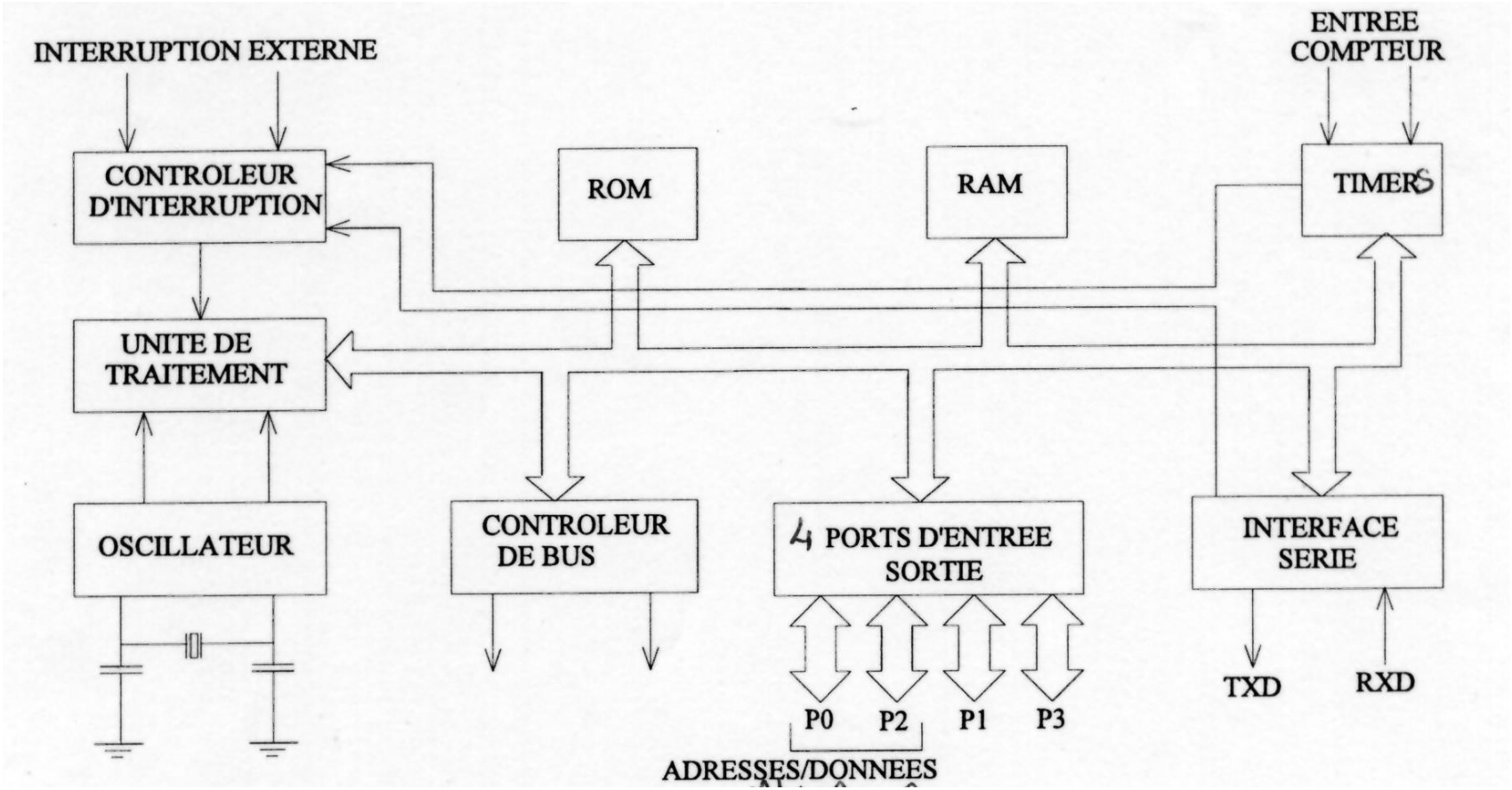
Micro processeur type

Architecture interne : exemple du 80C52

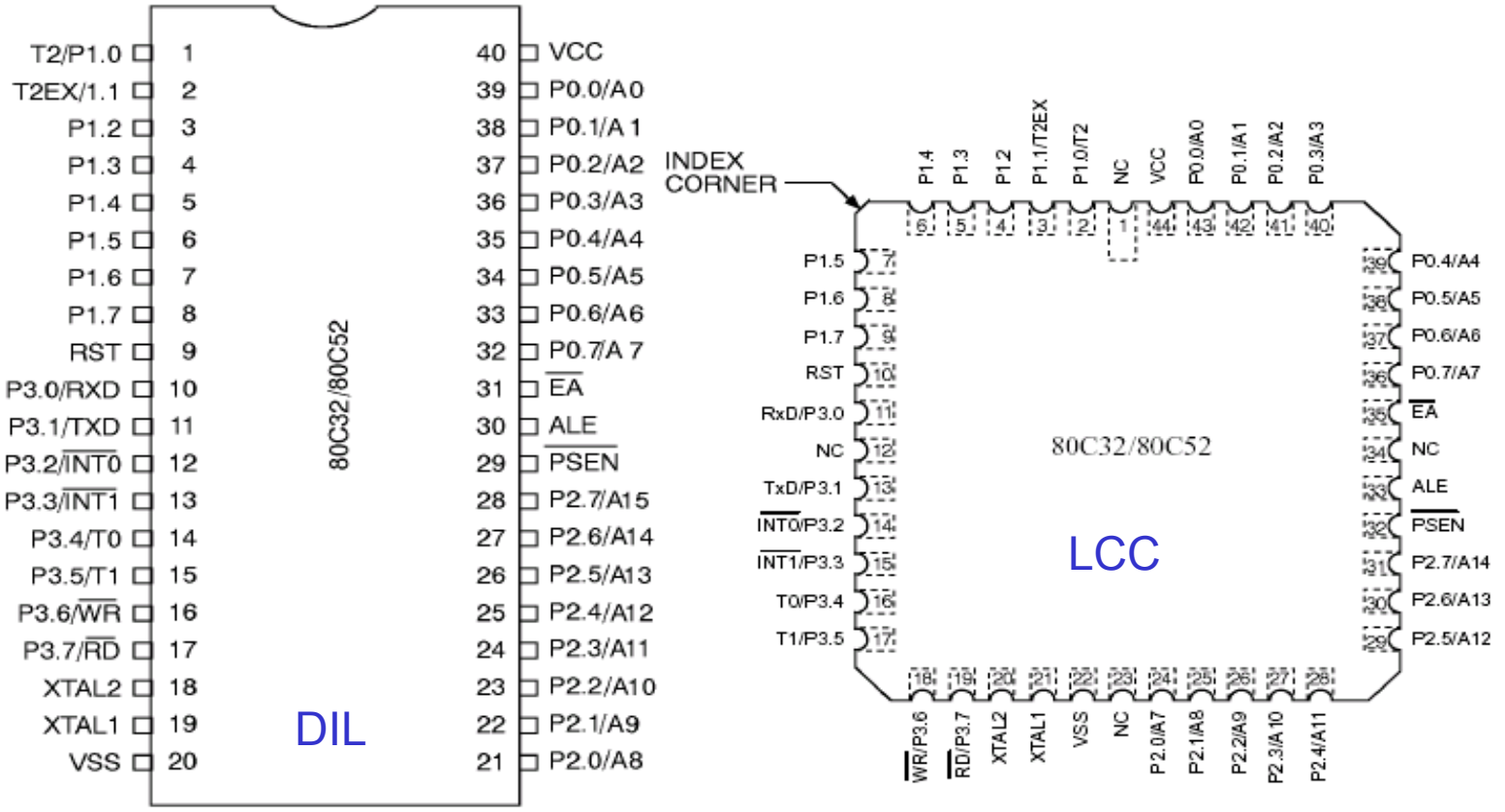
DS87C520/DS83C520 BLOCK DIAGRAM Figure 1



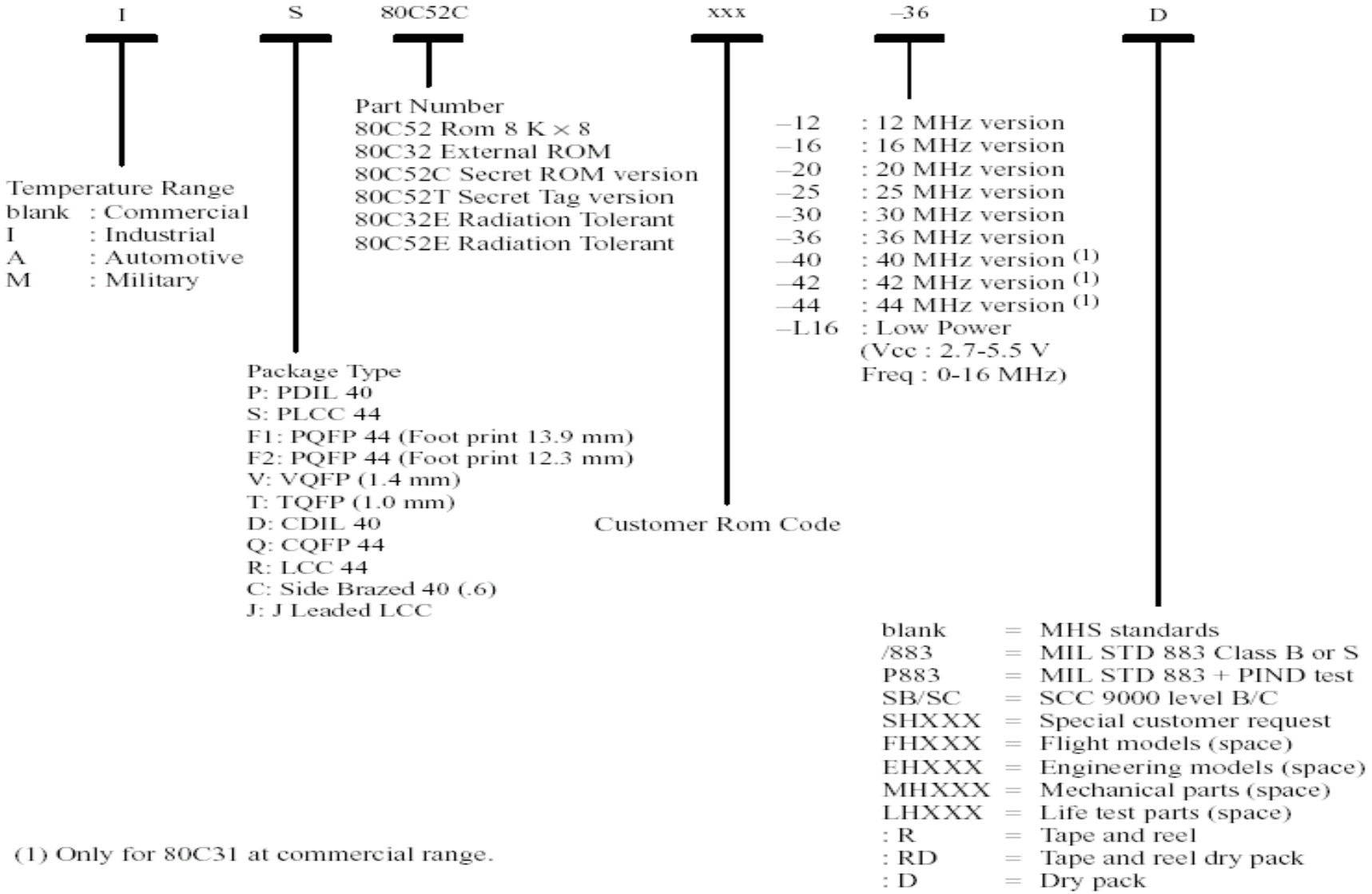
Architecture du 8051



Exemple : Intel 8032/8052



9. Ordering Information



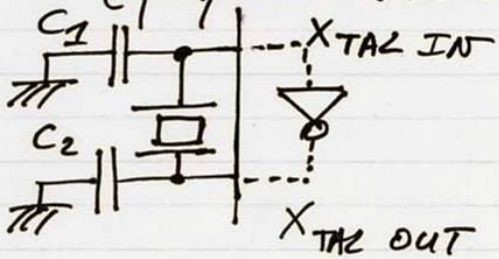
(1) Only for 80C31 at commercial range.

Introduction aux microprocesseurs

Le µC est cadencé par son HORLOGE : 99 MHz à 99 100 MHz

→ cf schéma procédé par le constructeur (data book, Web fabricant) :

→ en général utilisé un QUARTZ (fréquence standard)



C_1, C_2 quelques pF

→ Oscillateur interne (cf PIC Microchip)

→ Générateur externe (horloge cadée)

⇒ Couplage (synchro°) de plusieurs µC

Remarque : Vérifier la qualité, l'amplitude de l'horloge

En CMOS (et HCMOS) si $F_{Horloge}$ ↑ Consommation ↑

OSCILLATEUR µC

Pin Description (Continued)

When the AT89C51 is executing code from external program memory, PSEN is activated twice each machine cycle, except that two PSEN activations are skipped during each access to external data memory.

EA/VPP

External Access Enable. EA must be strapped to GND in order to enable the device to fetch code from external program memory locations starting at 0000H up to FFFFH. Note, however, that if lock bit 1 is programmed, EA will be internally latched on reset.

EA should be strapped to VCC for internal program executions.

This pin also receives the 12-volt programming enable voltage (VPP) during Flash programming, for parts that require 12-volt VPP.

XTAL1

Input to the inverting oscillator amplifier and input to the internal clock operating circuit.

XTAL2

Output from the inverting oscillator amplifier.

Oscillator Characteristics

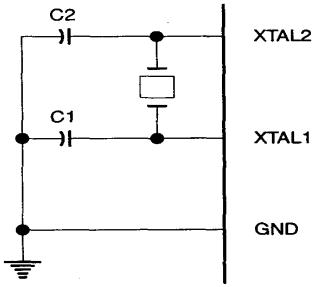
XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier which can be configured for use as an on-chip oscillator, as shown in Figure 1. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven as shown in Figure 2. There are no requirements on the duty cycle of the external clock signal, since the input to the internal clocking circuitry is through a divide-by-two flip-flop, but minimum and maximum voltage high and low time specifications must be observed.

Idle Mode

In idle mode, the CPU puts itself to sleep while all the on-chip peripherals remain active. The mode is invoked by software. The content of the on-chip RAM and all the special functions registers remain unchanged during this mode. The idle mode can be terminated by any enabled interrupt or by a hardware reset.

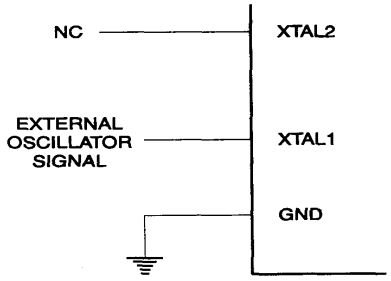
It should be noted that when idle is terminated by a hardware reset, the device normally resumes program execution, from where it left off, up to two machine cycles before the internal reset algorithm takes control. On-chip hard-

Figure 1. Oscillator Connections



Notes: C1, C2 = 30 pF ± 10 pF for Crystals
= 40 pF ± 10 pF for Ceramic Resonators

Figure 2. External Clock Drive Configuration



Status of External Pins During Idle and Power Down

Mode	Program Memory	ALE	PSEN	PORT0	PORT1	PORT2	PORT3
Idle	Internal	1	1	Data	Data	Data	Data
Idle	External	1	1	Float	Data	Address	Data
Power Down	Internal	0	0	Data	Data	Data	Data
Power Down	External	0	0	Float	Data	Data	Data

Oscillateur

CONSOMMATION = f (FREQ HORLOGE)

DC ELECTRICAL CHARACTERISTICS FOR PHILIPS NORTH AMERICA DEVICES (SC80C31 AND SC80C51)

T_{amb} = 0°C to +70°C or –40°C to +85°C, V_{CC} = 5V ±10%; V_{SS} = 0V

SYMBOL	PARAMETER	TEST CONDITIONS	LIMITS			UNIT
			MIN	TYP[1]	MAX	
V_{IL}	Input low voltage	4.5V < V_{CC} < 5.5V	–0.5		$0.2V_{CC}$–0.1	V
V_{IH}	Input high voltage (ports 0, 1, 2, 3, $\overline{EA}$)		$0.2V_{CC}$+0.9		V_{CC}+0.5	V
V_{IH1}	Input high voltage, XTAL1, RST		$0.7V_{CC}$		V_{CC}+0.5	V
V_{OL}	Output low voltage, ports 1, 2, 3[8]	V_{CC} = 4.5V I_{OL} = 1.6mA[2]			0.4	V
V_{OL1}	Output low voltage, port 0, ALE, $\overline{PSEN}$[8, 7]	V_{CC} = 4.5V I_{OL} = 3.2mA[2]			0.4	V
V_{OH}	Output high voltage, ports 1, 2, 3 [3]	V_{CC} = 4.5V I_{OH} = –30µA	V_{CC} – 0.7			V
V_{OH1}	Output high voltage (port 0 in external bus mode), ALE[9], $\overline{PSEN}$[3]	V_{CC} = 4.5V I_{OH} = –3.2mA	V_{CC} – 0.7			V
I_{IL}	Logical 0 input current, ports 1, 2, 3	V_{IN} = 0.4V	–1		–50	µA
I_{TL}	Logical 1-to-0 transition current, ports 1, 2, 3[6]	V_{IN} = 2.0V See note 4			–650	µA
I_{LI}	Input leakage current, port 0	0.45 < V_{IN} < V_{CC} – 0.3			±10	µA
I_{CC}	Power supply current (see Figure 8): Active mode @ 16MHz[5] Idle mode @ 16MHz[5] Power-down mode	See note 5 T_{amb} = 0 to +70°C T_{amb} = –40 to +85°C		11.5 1.3 3	32 5 50 75	µA mA µA µA
R_{RST}	Internal reset pull-down resistor		40		225	kΩ
C_{IO}	Pin capacitance[10] (except $\overline{EA}$)				15	pF

NOTES:

1. Typical ratings are not guaranteed. The values listed are at room temperature, 5V.
2. Capacitive loading on ports 0 and 2 may cause spurious noise to be superimposed on the V_{OL}s of ALE and ports 1 and 3. The noise is due to external bus capacitance discharging into the port 0 and port 2 pins when these pins make 1-to-0 transitions during bus operations. In the worst cases (capacitive loading > 100pF), the noise pulse on the ALE pin may exceed 0.8V. In such cases, it may be desirable to qualify ALE with a Schmitt Trigger, or use an address latch with a Schmitt Trigger STROBE input. I_{OL} can exceed these conditions provided that no single output sinks more than 5mA and no more than two outputs exceed the test conditions.
3. Capacitive loading on ports 0 and 2 may cause the V_{OH} on ALE and $\overline{PSEN}$ to momentarily fall below the (V_{CC}–0.7) specification when the address bits are stabilizing.
4. Pins of ports 1, 2 and 3 source a transition current when they are being externally driven from 1 to 0. The transition current reaches its maximum value when V_{IN} is approximately 2V.
5. See Figures 9 through 12 for I_{CC} test conditions.
   Active Mode: I_{CC} = 1.5 × FREQ + 8.0;
   Idle Mode: I_{CC} = 0.14 × FREQ +2.31; See Figure 8.
6. This value applies to T_{amb} = 0°C to +70°C. For T_{amb} = –40°C to +85°C, I_{TL} = –750µA.
7. Load capacitance for port 0, ALE, and $\overline{PSEN}$ = 100pF, load capacitance for all other outputs = 80pF.
8. Under steady state (non-transient) conditions, I_{OL} must be externally limited as follows:
   Maximum I_{OL} per port pin: 15mA (*NOTE: This is 85°C specification.)
   Maximum I_{OL} per 8-bit port: 26mA
   Maximum total I_{OL} for all outputs: 71mA
   If I_{OL} exceeds the test condition, V_{OL} may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test conditions.
9. ALE is tested to V_{OH1}, except when ALE is off then V_{OH} is the voltage specification.
10. Pin capacitance is characterized but not tested. Pin capacitance is less than 25pF. Pin capacitance of ceramic package is less than 15pF (except $\overline{EA}$ it is 25pF).

Conso = conso statique + conso dynamique (fonction de la fréquence)

La consommation dépend de la fréquence d'horloge

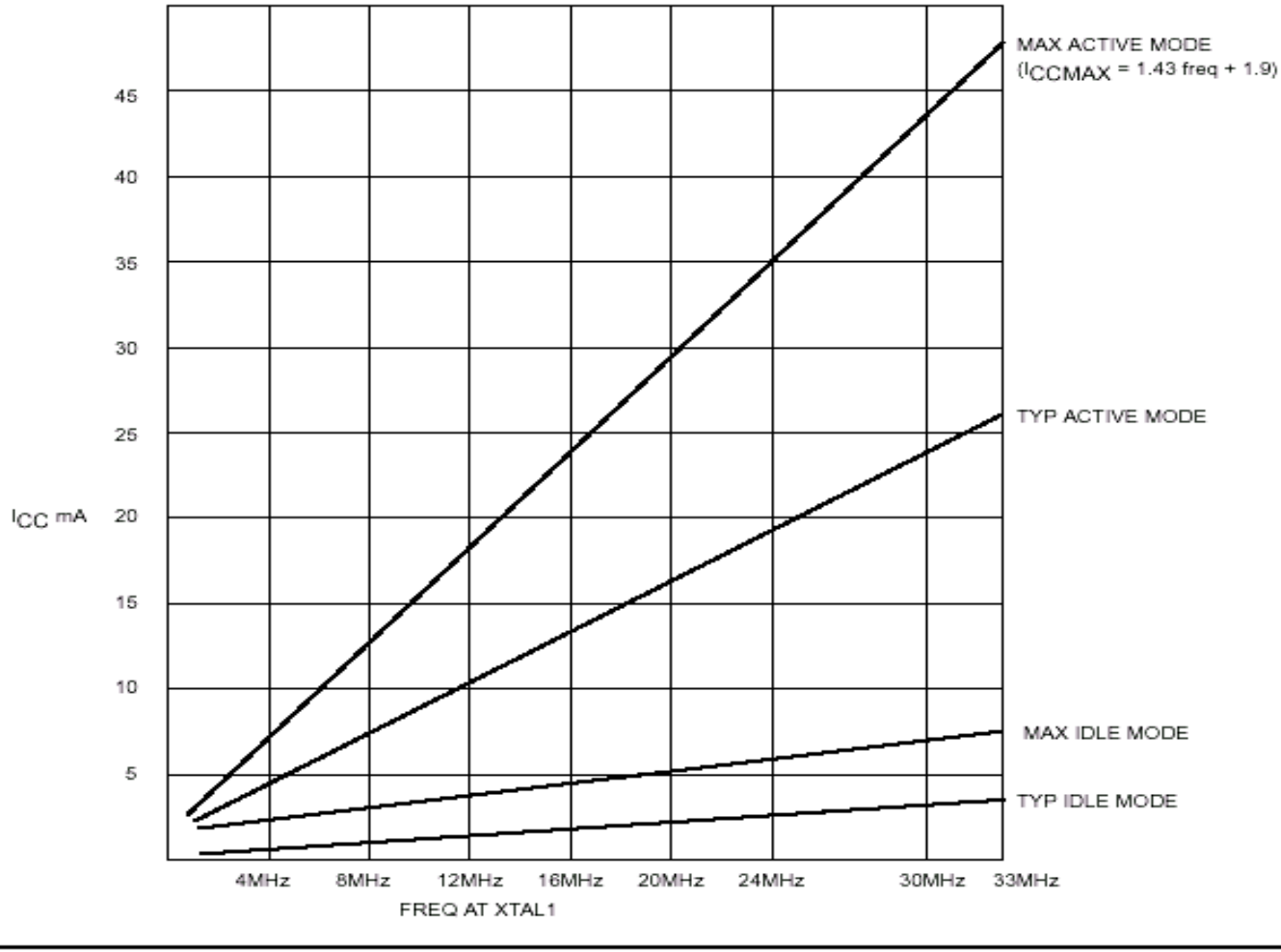
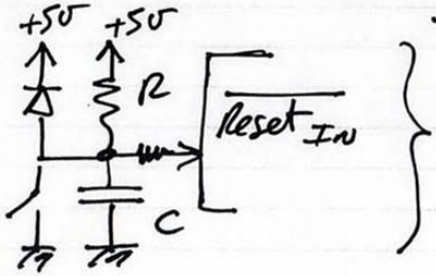


Figure 8. I_{CC} vs. FREQ
Valid only within frequency specifications of the device under test

Introduction aux microprocesseurs

INITIALISATION (RESET) du µC
↳ à la mise sous tension
↳ en fonctionnement (Reset)] cf notice constructeur
↳ cf NMI (Non Maskable Interrupt) en cas de coupure d'alimentation



Cas du reset [actif au niveau bas (Reset)
la constante (RC) détermine la durée de l'impulsion de Reset (cf notice constructeur)

→ cf Circuit de reset intégrés, superviseurs d'alimentation (MAXIM, Analog Devices, Motorola), sauvegarde de batteries, watchdog intégré

19-4333; Rev 3; 9/93

MAXIM

Microprocessor Supervisory Circuits

MAX690A/MAX692A/MAX802L/MAX802M/MAX805L

General Description

The MAX690A/MAX692A/MAX802L/MAX802M/MAX805L reduce the complexity and number of components required for power-supply monitoring and battery-control functions in microprocessor (µP) systems. They significantly improve system reliability and accuracy compared to separate ICs or discrete components.

These parts provide four functions:

1) A reset output during power-up, power-down, and brownout conditions.
2) Battery-backup switching for CMOS RAM, CMOS µP, or other low-power logic.
3) A reset pulse if the optional watchdog timer has not been toggled within 1.6sec.
4) A 1.25V threshold detector for power-fail warning or low-battery detection, or to monitor a power supply other than +5V.

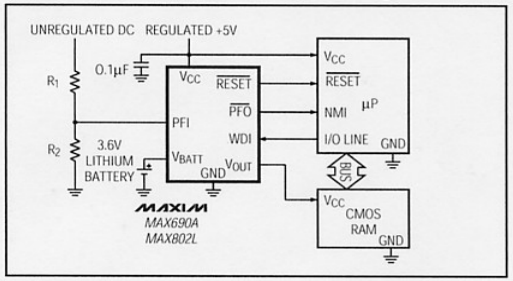
The parts differ in their reset-voltage threshold levels and reset outputs. The MAX690A/MAX802L/MAX805L generate a reset pulse when the supply voltage drops below 4.65V, and the MAX692A/MAX802M generate a reset below 4.40V. The MAX802L/MAX802M guarantee power-fail accuracies to ±2%. The MAX805L is the same as the MAX690A except that RESET is provided instead of RESET.

All parts are available in 8-pin DIP and SO packages. The MAX690A/MAX802L are pin compatible with the MAX690 and MAX694. The MAX692A/MAX802M are pin compatible with the MAX692.

Applications

Battery-Powered Computers and Controllers
Intelligent Instruments
Automotive Systems
Critical µP Power Monitoring

Typical Operating Circuit



Features

- **Precision Supply-Voltage Monitor:**
  4.65V for MAX690A/MAX802L/MAX805L
  4.40V for MAX692A/MAX802M
- **Reset Time Delay – 200ms**
- **Watchdog Timer – 1.6sec Timeout**
- **Battery-Backup Power Switching**
- **200µA Quiescent Supply Current**
- **50nA Quiescent Supply Current in Battery-Backup Mode**
- **Voltage Monitor for Power-Fail or Low-Battery Warning**
- **Power-Fail Accuracy Guaranteed to ±2% (MAX802L/M)**
- **Guaranteed $\overline{\text{RESET}}$ Assertion to V_{CC} = 1V**
- **8-Pin SO and DIP Packages**

Ordering Information

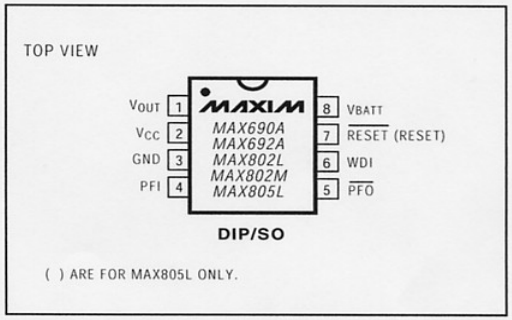
PART	TEMP. RANGE	PIN-PACKAGE
MAX690ACPA	0 °C to +70 °C	8 Plastic DIP
MAX690ACSA	0 °C to +70 °C	8 SO
MAX690AC/D	0 °C to +70 °C	Dice*
MAX690AEPA	-40 °C to +85 °C	8 Plastic DIP
MAX690AESA	-40 °C to +85 °C	8 SO
MAX690AMJA	-55 °C to +125 °C	8 CERDIP**

Ordering Information continued on last page.

* *Dice are specified at T_A = +25 °C*

** *Contact factory for availability and processing to MIL-STD-883.*

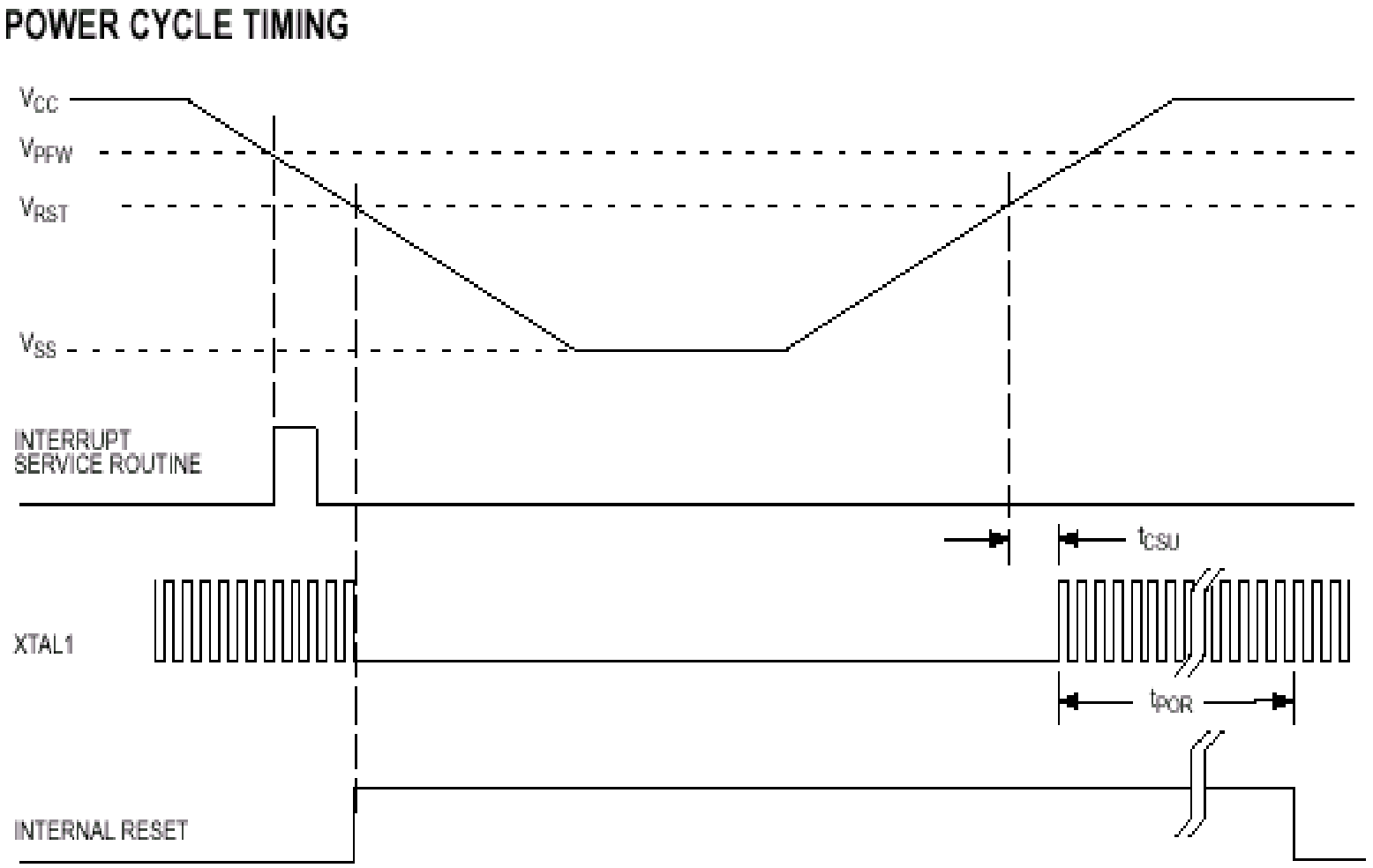
Pin Configurations



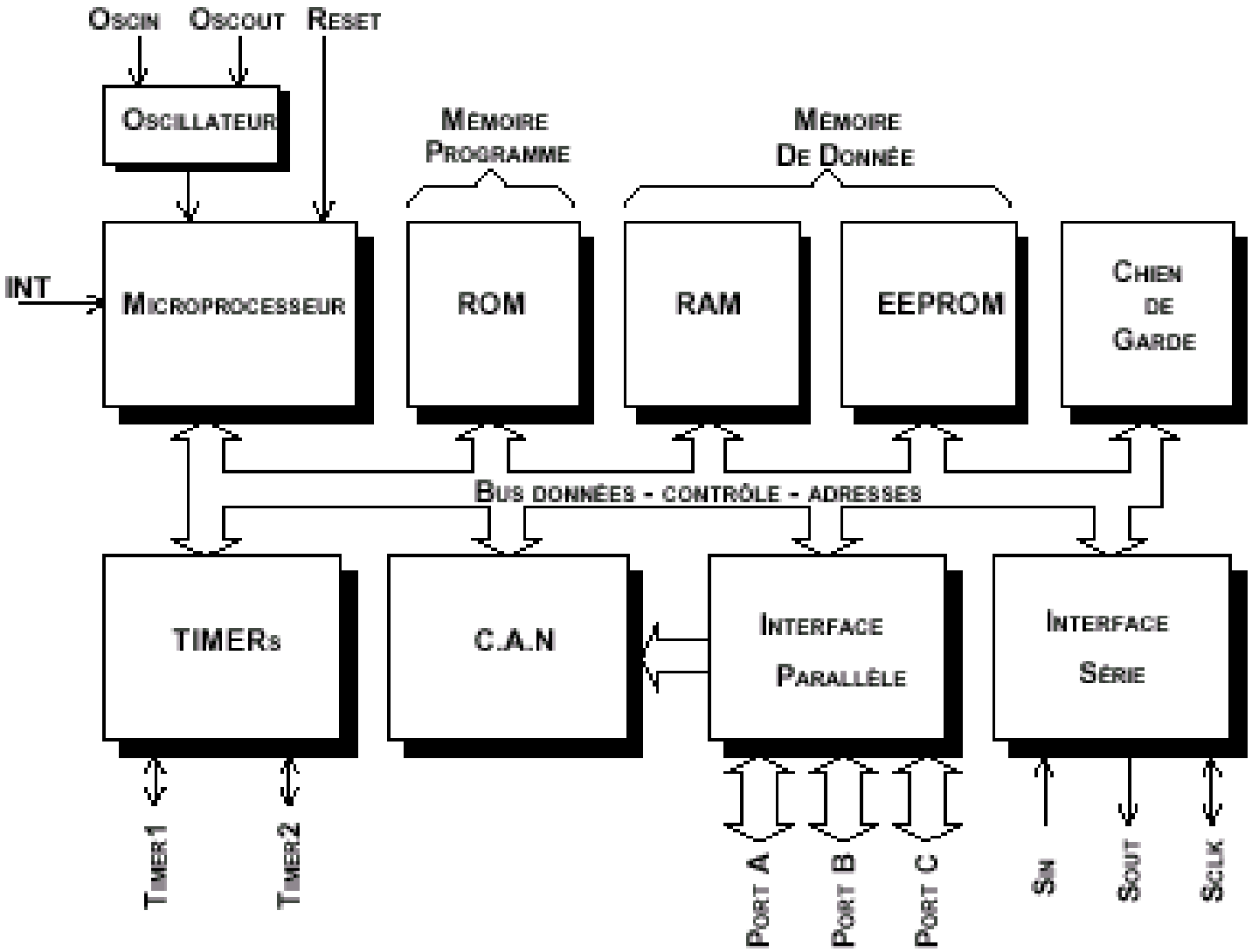
Call toll free 1-800-998-8800 for free samples or literature.

Circuit superviseur d'alimentation (et reset)

Timing microprocesseur : démarrage / reset



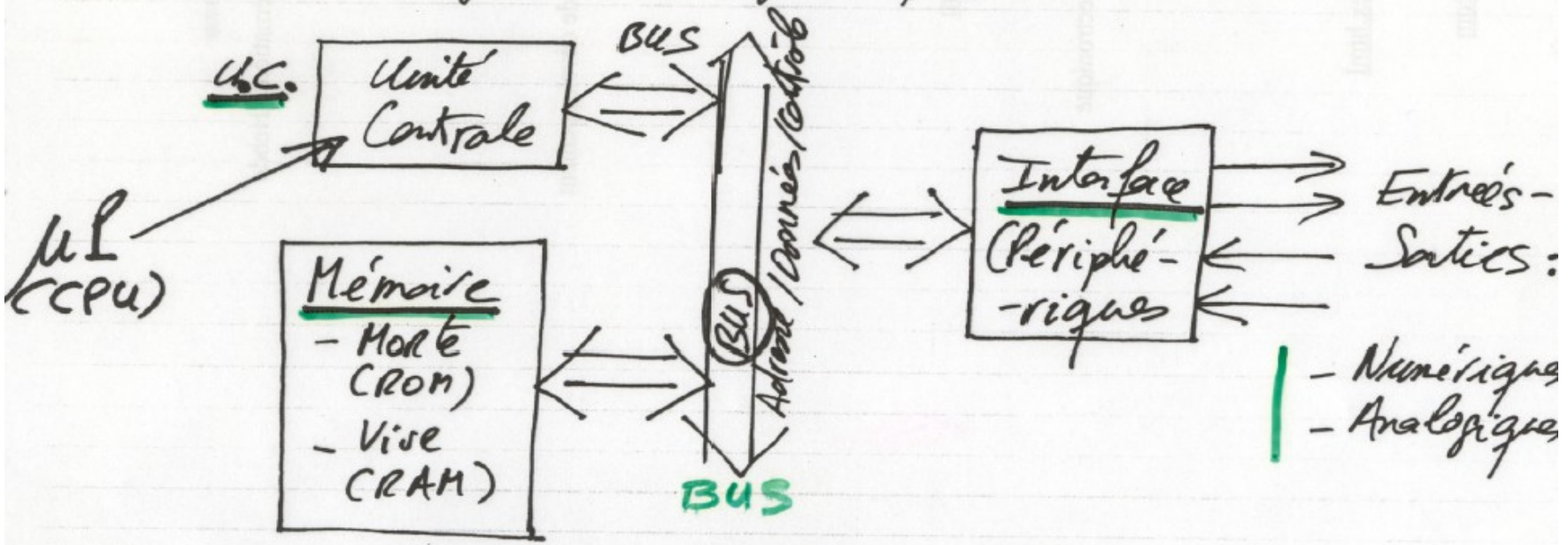
Architecture classique d'une carte à microprocesseur



Introduction aux microprocesseurs

MICROPROCESSEUR : composant programmable
"intelligent", "souple"

Architecture type d'un système programmable



Remarque : Attention à la qualité de l'alimentation et aux découplages des alimentations

Microcontrôleurs : quelques exemples

Référence	Fabricant	Vitesse	RAM	ROM / EPROM / FLASH	EEPROM	E / S Logiques	Timer	Entrées Analogiques	Particularité
8051	Intel	12 Mhz	128 o	4 Ko	X	32	2	0	
16C71	Microchip	20 Mhz	36 o	1Kx14	X	13	1	4	RISC
6805 S2	Motorola	4 MHz	64	1 Ko	X	16	2	8	
68HC11 A1	Motorola	8 MHz	256 o	X	512	22	1	8	Etendu
AT90S 8515	Atmel	20 MHz	512 o	4 Ko	512	32	3	8	RISC
ST 6265	Thomson	8 MHz	128 o	4 Ko	64 o	21	2	13	

Les grandes familles de microcontrôleurs

- **Microcontrôleurs « bas de gamme » :**
  - Exemple : PIC de MicroChip, 68HC05 de Motorola
    - Très faible coût (PIC 12CE67X quelques francs)
    - Oscillateur, convertisseur A/N, Timer, ... Intégrés
    - 8 broches
- **Microcontrôleurs « milieu de gamme » :**
  - Exemple MC68HC11K4 de Motorola, 8051 d 'Intel
- **Microcontrôleurs « haut de gamme » :**
  - Exemple : MPC555 de Motorola
    - Prix : plusieurs 100 F
    - Boîtier BGA 272 broches
    - Flash EPROM interne, A/D 32 voies, contrôleur bus CAN, ...
- **Les microcontrôleurs orientés « système »**

Architectures des microprocesseurs

Structure interne :

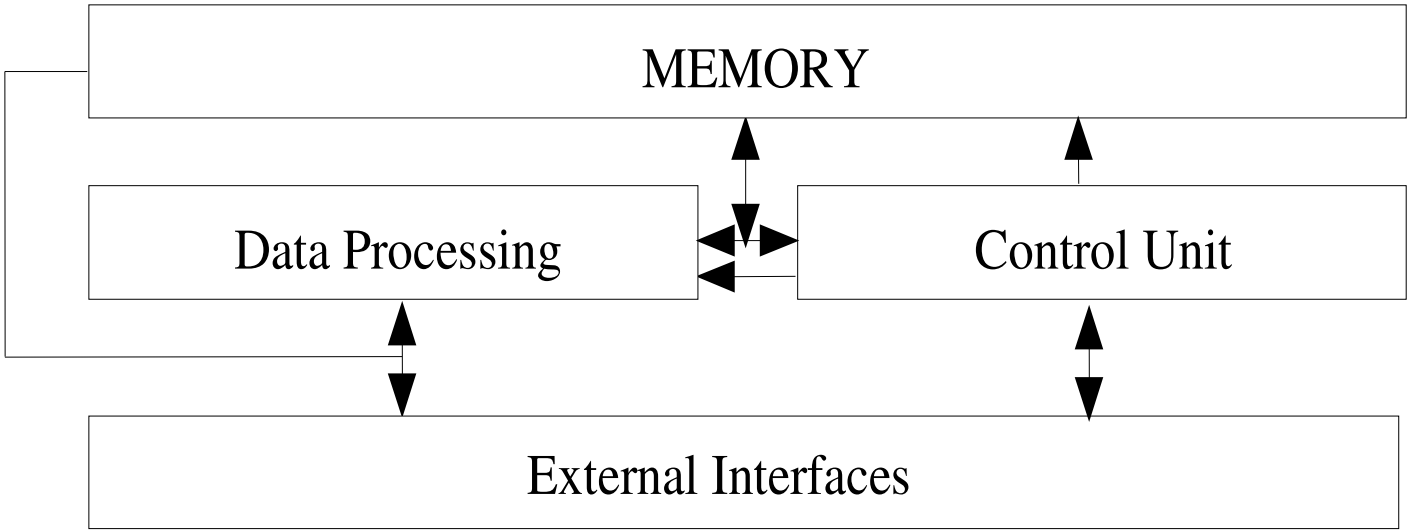
- type « **Von Neumann** » = Structure de base des microprocesseurs (modèle de Von Neumann de 1945) : 68HC11 par ex.
- type « **Harvard** » :
  - Mémoires programme et données séparées, plus rapide : PIC de Microchip par ex.
- type « **pipeline** » : type DSP (digital signal processor)

Jeu d'instructions :

- Architecture **CISC** : complex instruction set computer
- Architecture **RISC** : reduced instruction set computer,cf PIC
  - jeu d'instructions réduit et câblé (donc rapide)
  - une instruction par cycle
- Instructions spécialisées : instruction MAC des DSP par exemple

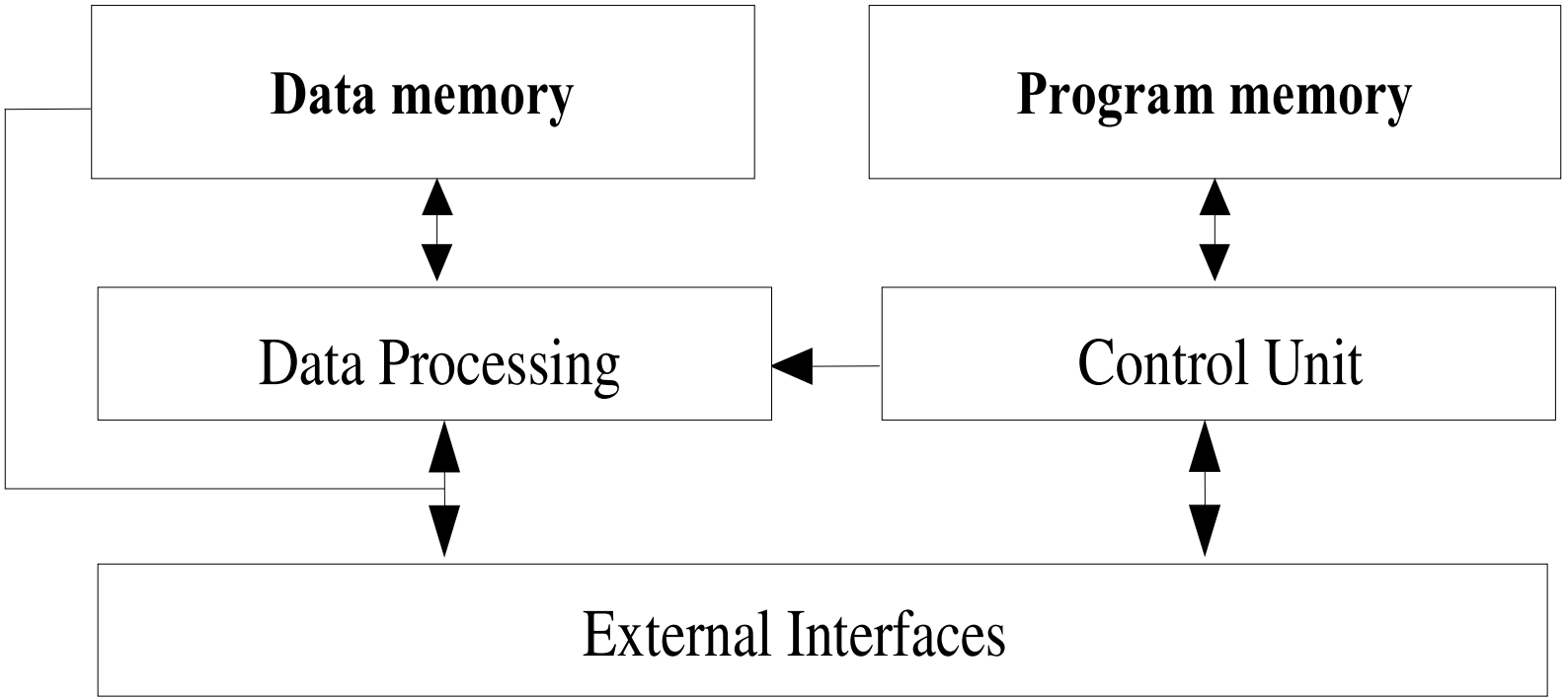
Architecture interne de microprocesseur de type « Von Neumann »

- Structure de base des microprocesseurs
  - la mémoire contient les instructions (programme) et les données
  - un seul bus interne



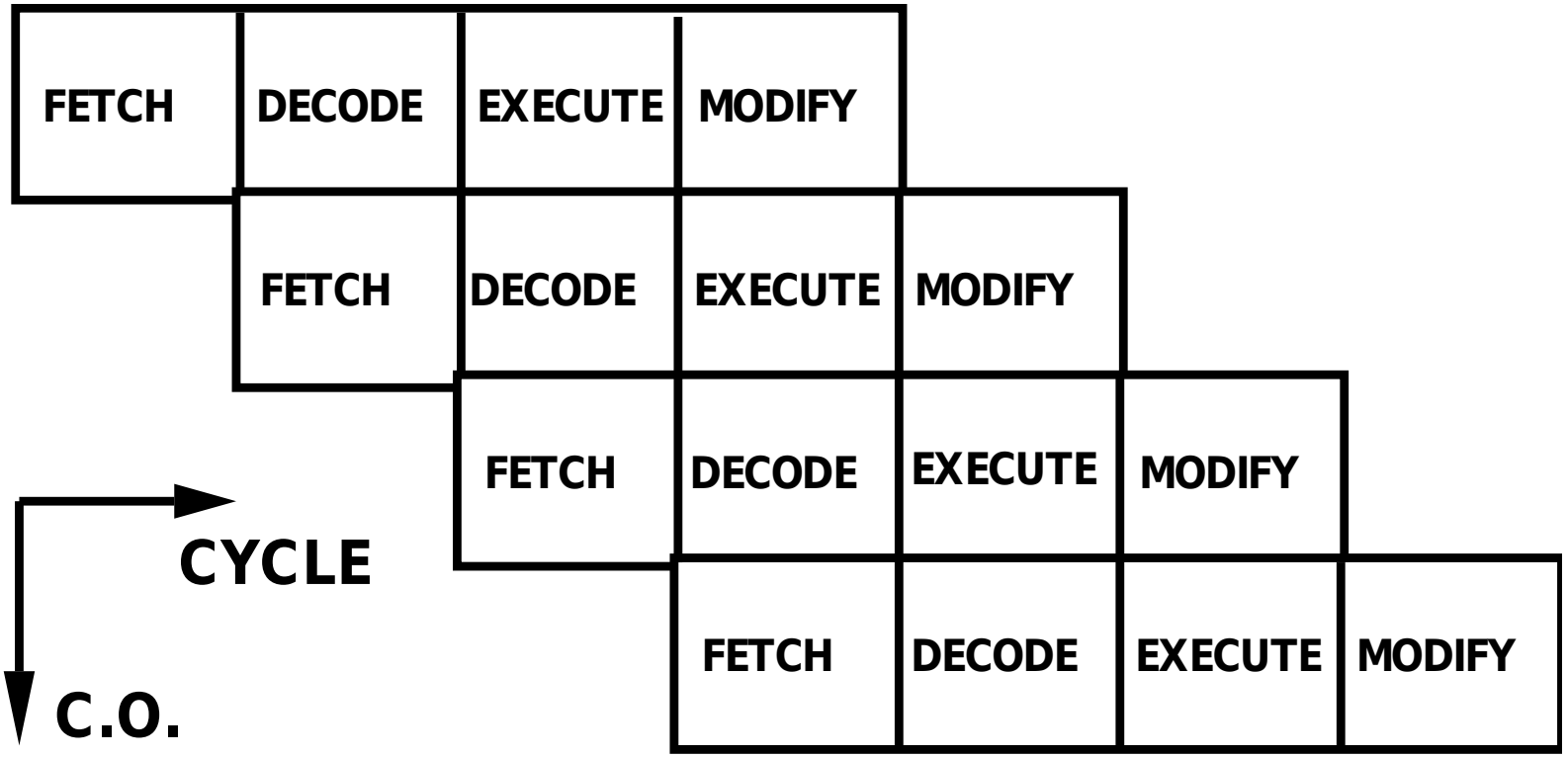
Architecture interne de microprocesseur de type « Harvard »

- Mémoires programme et données séparées, plus rapide



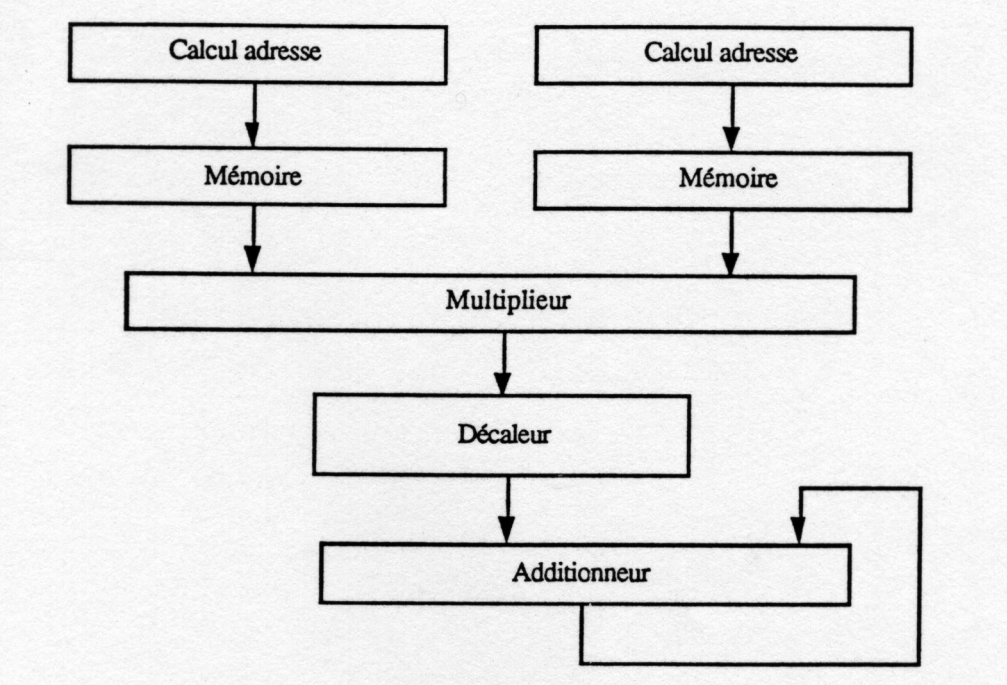
Structure interne de type « pipeline »

- processeurs de traitement du signal (digital signal processor DSP) et microprocesseurs rapides
- plusieurs bus internes

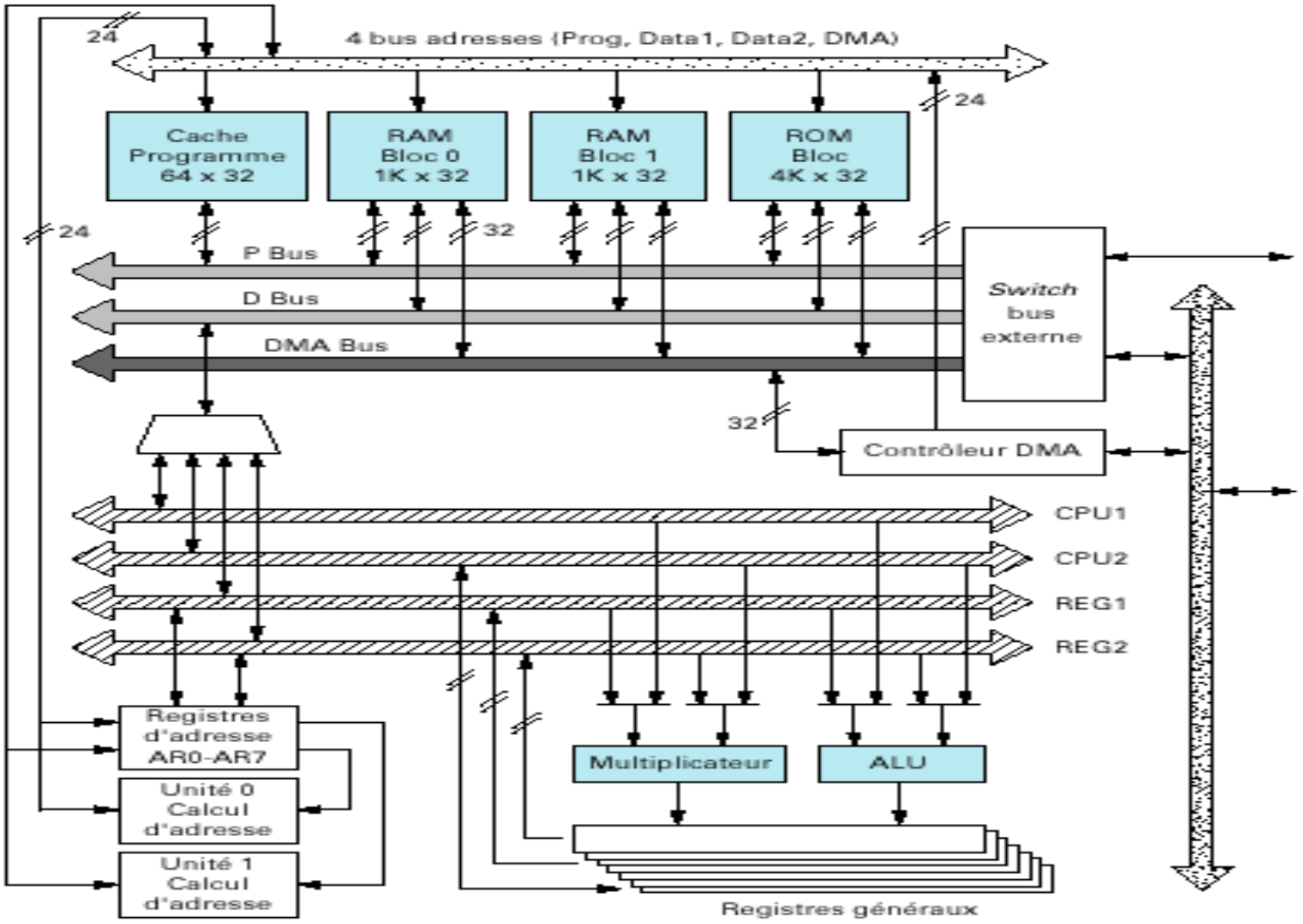


Instruction MAC (multiplication – accumulation) des DSP

- **Réalisation rapide de sommes pondérées (filtres numériques, FFT, …)**

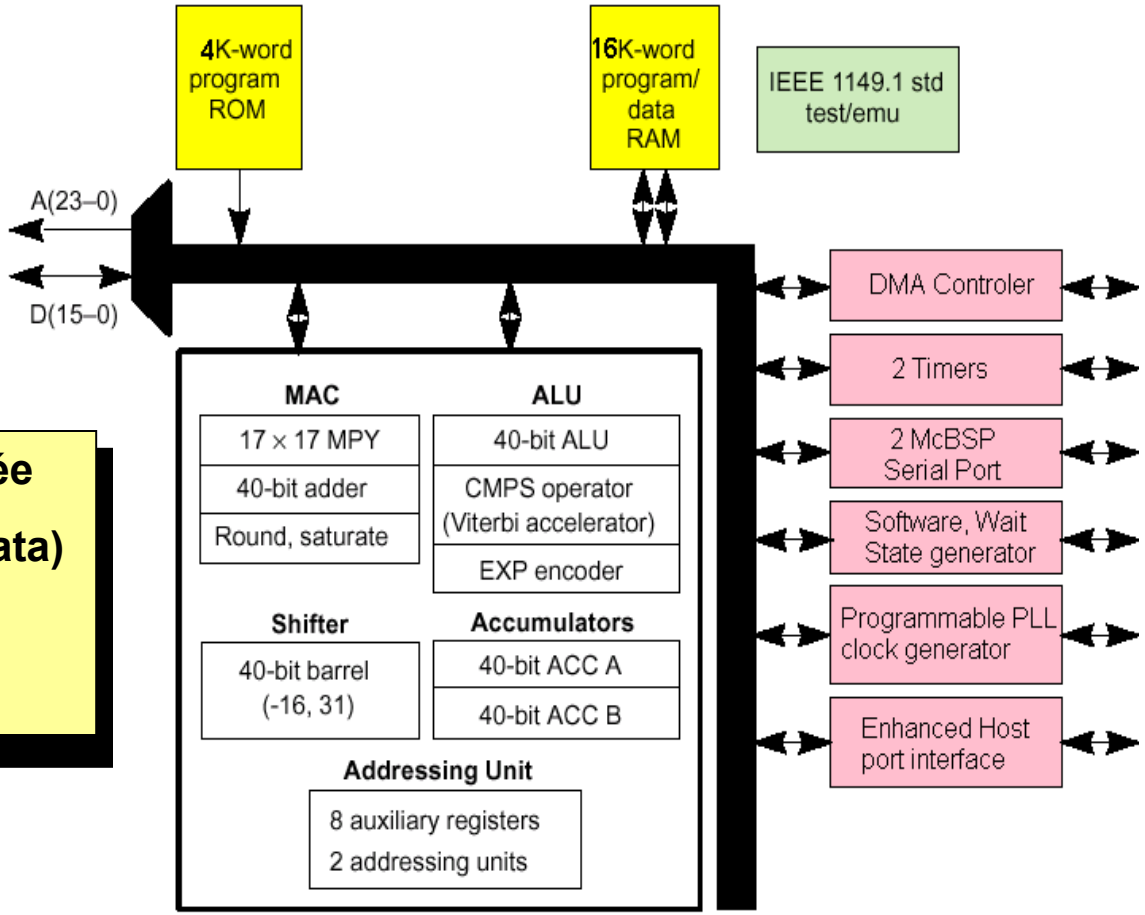


Structure interne de type « pipeline » : le TMS320C30



Présentation sommaire du DSP TMS320VC5402

- Architecture Harvard, arithmétique virgule fixe
- Instruction MAC (multiplication / accumulation)



Architecture Harvard Modifiée

- **bus programme (1 adr, 1 data)**
- **bus data (3 adr, 3 data)**
- **2 Read 1 Write en 1 cycle**

Architectures CISC / RISC

CISC : *complex instruction set computer*

RISC : *reduced instruction set computer*

Tableau 2 – Caractéristiques des architectures CISC et RISC

Caractéristiques	RISC	CISC
Nombre d'instructions	< 100	> 200
Nombre de modes d'adressage	1 à 2	5 à 20
Nombre de formats d'instructions	1 à 2	3 +
Nombre cycles/instruction	≈ 1	3 à 10
Accès à la mémoire	*load/store*	la plupart des instructions
Nombre de registres	32 +	2 à 16
Réalisation du microprocesseur	câblé	micropro-grammé
Logique pour le décodage des instructions	10 %	> 50 %

Tableau 1 – Classification des microprocesseurs en fonction des caractéristiques de l'architecture

Architecture	Caractérisation	Exemples
CISC classique	Jeu d'instructions avec un grand nombre d'instructions et un grand nombre de modes d'adressage. Architecture ayant évolué au cours du temps en cherchant à préserver la compatibilité	Famille IA-32 d'Intel Famille 680x0 de Motorola
RISC	Jeu d'instructions développé dans l'objectif d'optimiser le temps d'exécution des programmes	Alpha (DEC) MIPS (SGI) PowerPC (Apple/IBM/Motorola) *Precision Architecture* (HP) SPARC (Sun)
Traitement de signal (DSP)	Jeu d'instructions développé en vue d'optimiser les applications de traitement de signal, par exemple multiplication et accumulation...	Analog Devices ADSP-21020 et SHARC (ADSP-2106*x*) Motorola DSP96002 Texas Instruments TMS320C3*x* et TMS320C4*x*
En réseau	Architecture intégrant des moyens de communication rapides avec d'autres microprocesseurs permettant la mise en réseau. Structure fondée sur l'échange de messages	SGS Thomson (famille ex Transputer) Texas Instruments TMS320C4*x*

MICROCONTROLEURS RISC

CONCEPTION :

- Régularité du traitement des instructions (1 instr./cycle)
- Modèle à registres (opérations sur registres)
- Peu de modes d'adressage – formats d'instruction
- Utilisation intensive du pipeline

Bonne efficacité pour la programmation
en langage de haut niveau (C / C++)

Haute vitesse de traitement : 200 – 400 Mips

Évaluation de la puissance de calcul d'un processeur

- On peut tenter d'évaluer les performances d'un microprocesseur en comptant le nombre d'instructions qu'il est capable d'exécuter en une seconde. Les instructions bien qu'élémentaires peuvent s'exécuter en des temps assez différents suivant leur niveau de complexité, entre 1 et 10 cycles d'horloge pour les machines de type PC. Les unités pour ce type d'évaluation sont le **MIPS** (millions d'instructions par seconde) pour les calculs entiers et le **MFLOPS** (millions d'opérations flottantes par seconde) pour les calculs avec virgule.
- On utilise des programmes de tests « **Benchmarks** »
- Evaluation de la puissance « crête » (hors accès mémoire, périphériques, affichage des résultats)
- Puissance liée :
  - Au temps de cycle (vitesse d'horloge)
  - À l'architecture interne du microP (pipeline ?)

Évaluation de la puissance de calcul d'un processeur

	Horloge	Puissance	Nb transistors	Bus Interne	Date
	MHz	MIPS		bits	
8086	5	0,33	29000	16	78
80286	8	1,2	136000	16	82
80386DX	16	6	275000	32	85
80486DX	25	20	1200000	32	89
80486DX2	50	40	1200000	32	92
Pentium	66	112	3100000	64	93

Langages et outils de programmation des microprocesseurs

- Le *microcontrôleur est un composant électronique*, donc il faut connaître :
  - la notice technique du constructeur (« data-sheet ») et les notes d'applications principales
  - technologie (alimentation, compatibilité) et architecture interne
  - famille logique / interface
  - caractéristiques électriques
  - jeu d'instructions / adressage / format interne
  - vitesse / temps de cycle
  - différentes versions :
    - gammes de température : commerciale, étendue, militaire
  - vitesse / consommation / performances
  - type de boîtiers DIL, PLCC, CMS... (Attention à la CAO !)
  - versions EEPROM, UVPROM, FLASH, OTP, ...
  - prix / disponibilité (quantité) / délais

Langages et outils de programmation des microprocesseurs

- « secondes sources » , prix, disponibilité, obsolescence
- les circuits périphériques facilement utilisables :
  - de la même famille : séries 6800, 68000
  - d'autres constructeurs : INTEL, N.S., T.I.

- *C'est un composant programmable*, donc il faut connaître :
  - les outils de programmation à sa disposition :
    - assembleur : lié au jeu d'instructions du composant
    - compilateurs : langage C, C++, PASCAL, FORTRAN, BASIC compilé , ...
    - interpréteurs : BASIC interprété
    - simulateurs
  - la documentation et notes d 'applications du constructeur

Langages et outils de programmation des microprocesseurs

- - les outils de développement à sa disposition :
  - debuggers
  - simulateurs
  - kit, carte d'évaluation (EVMxxx)
  - **émulateurs**

- Pour le développement matériel (hardware) il faut utiliser :
  - l'oscilloscope
  - l'émulateur temps réel
  - l'analyseur logique
  - le programmateur d'EPROM, de FLASH, …

Emulateur temps réel

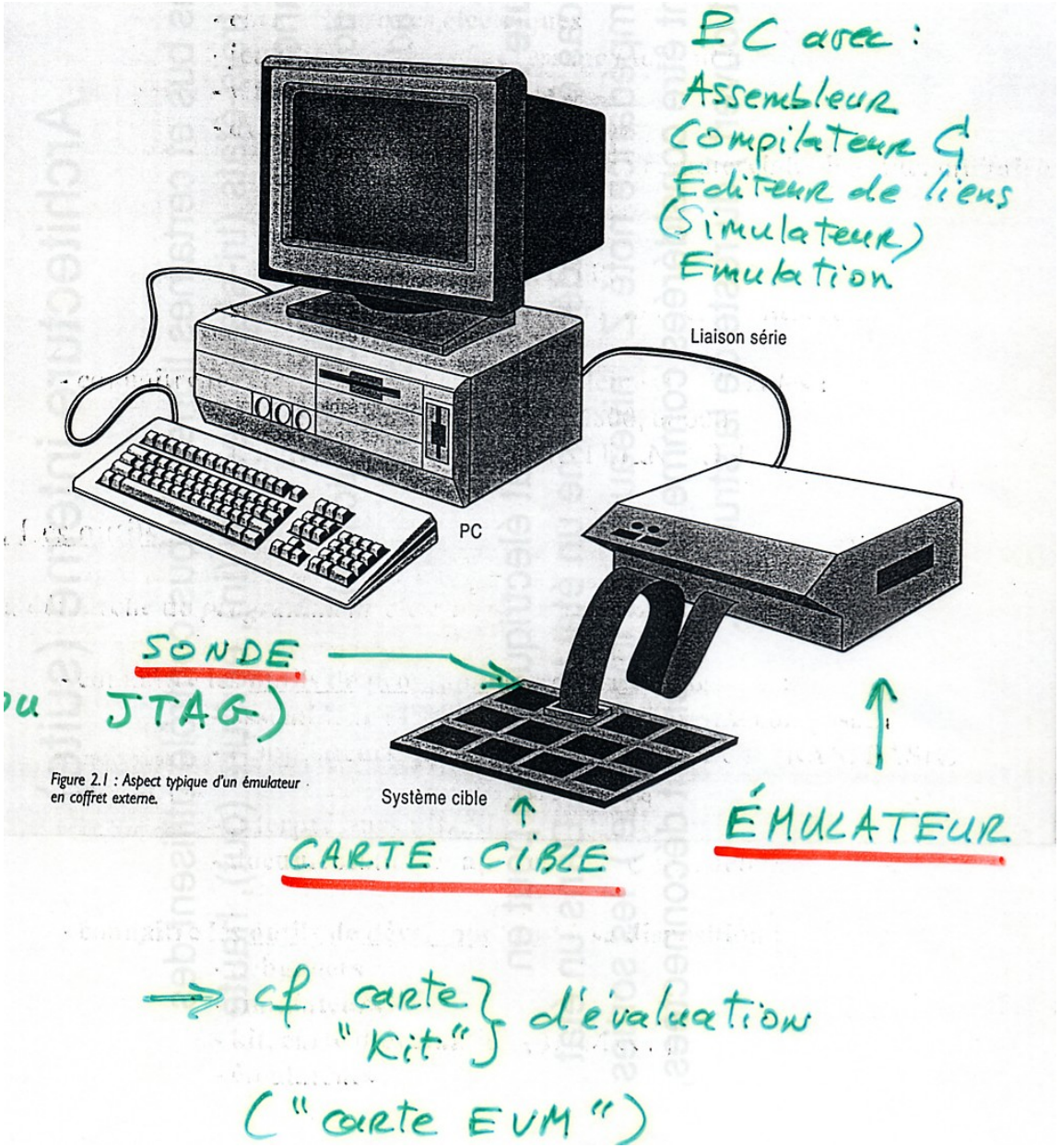
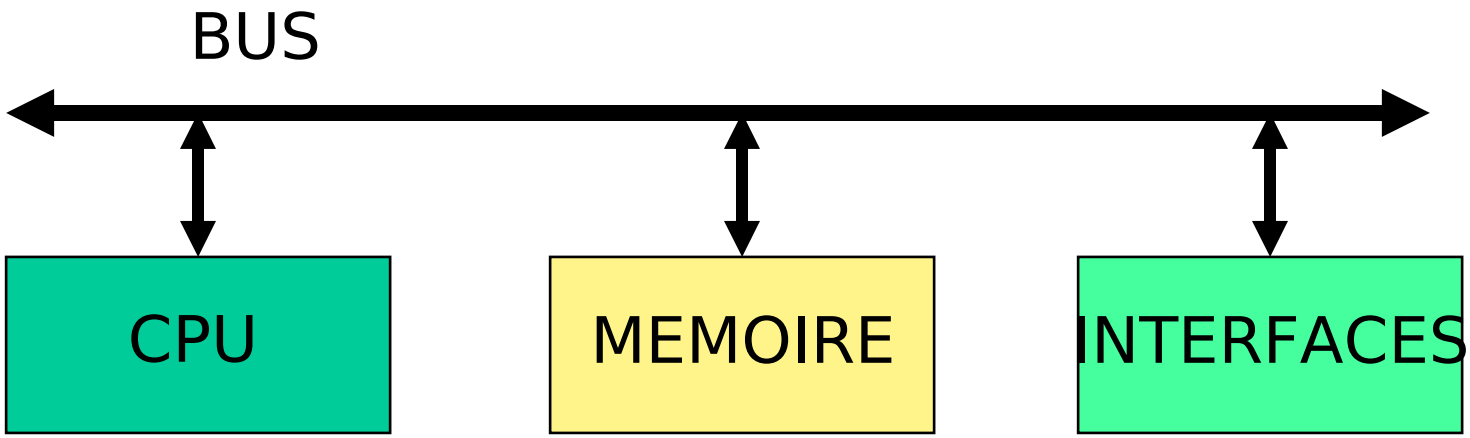


Figure 2.1 : Aspect typique d'un émulateur en coffret externe.

Applications des microcontrôleurs



- Simple
  - CISC
  - RISC

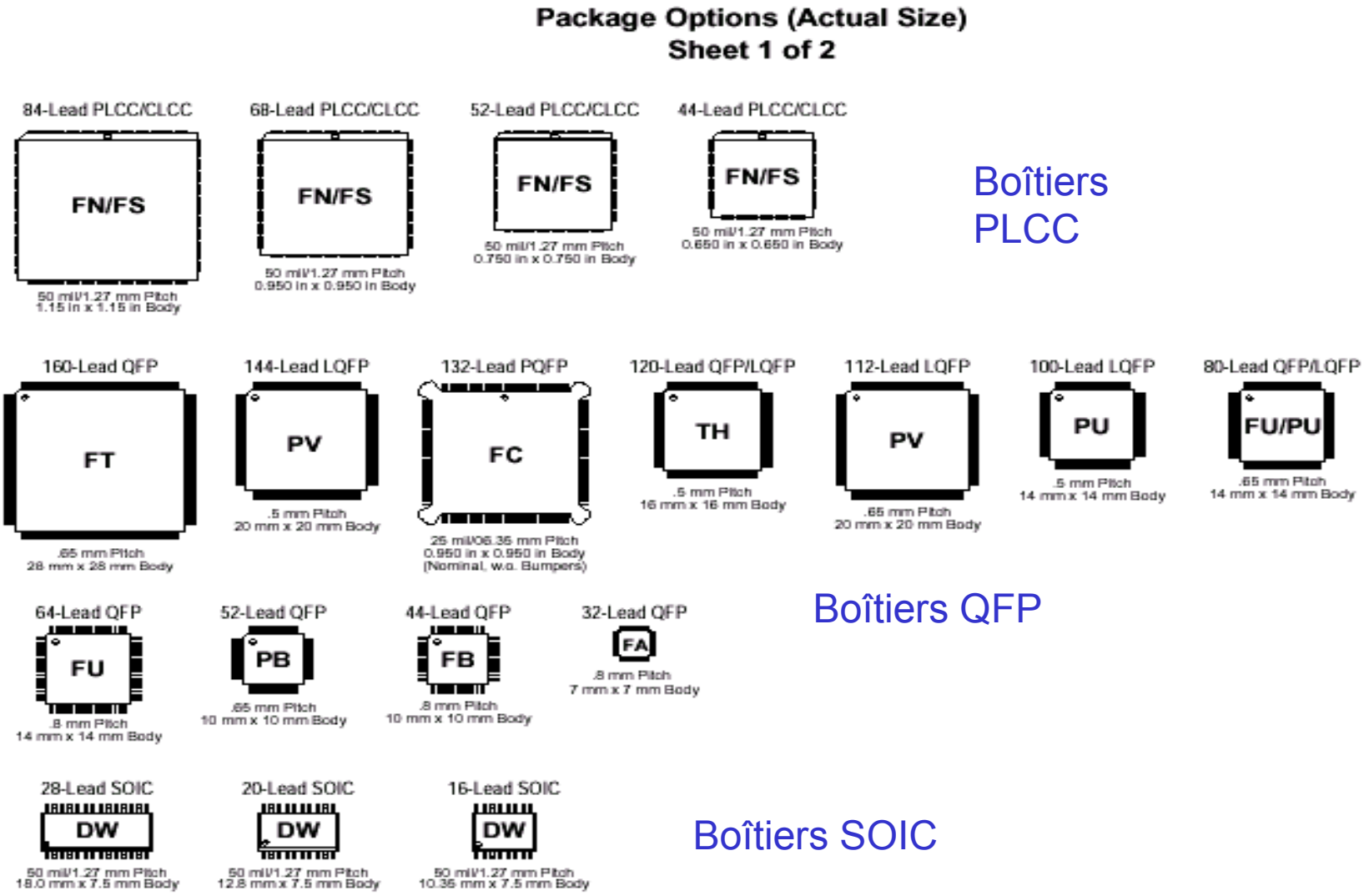
Programme
- EPROM Flash

Données
- RAM
- EEPROM

Génériques

Spécifiques
- Réseaux
- Analogiques
- Pilotage (PWM)

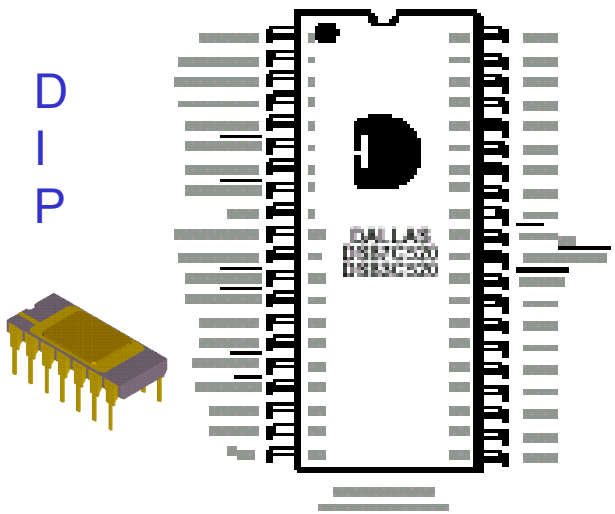
Un élément important : le type de boîtier



Un élément important : le type de boîtier

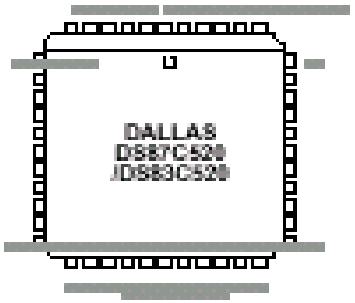
PACKAGE OUTLINE

DIP

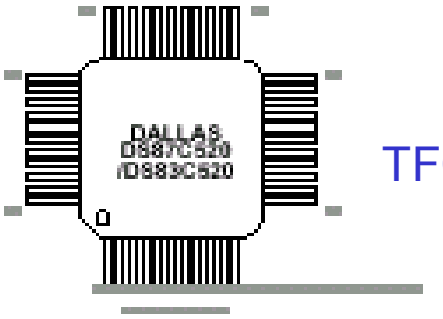


Exemple du microcontrôleur Dallas (compatible Intel 80C52)

- Available in 40–pin PDIP, 44–pin PLCC, 44–pin TQFP, and 40–pin windowed CERDIP



PLCC



TFQP

Selon le boîtier :

- CAO et routage (penser à la CEM dès la conception)
- support nécessaire ou câblage
- disponibilité / coût
- câblage manuel ou automatisé
- Maintenance / programmation

Un élément important : le type de boîtier, la vitesse et la gamme de température

ORDERING INFORMATION: Exemple du microcontrôleur Dallas compatible 80C52

PART NUMBER	PACKAGE	MAX. CLOCK SPEED	TEMPERATURE RANGE
DS87C520–MCL	40–pin plastic DIP	33 MHz	0°C to 70°C
DS87C520–QCL	44–pin PLCC	33 MHz	0°C to 70°C
DS87C520–ECL	44–pin TQFP	33 MHz	0°C to 70°C
DS87C520–MNL	40–pin plastic DIP	33 MHz	–40°C to +85°C
DS87C520–QNL	44–pin PLCC	33 MHz	–40°C to +85°C
DS87C520–ENL	44–pin TQFP	33 MHz	–40°C to +85°C
DS87C520–WCL	40–pin windowed CERDIP	33 MHz	0°C to 70°C
DS83C520–MCL	40–pin plastic DIP	33 MHz	0°C to 70°C
DS83C520–QCL	44–pin PLCC	33 MHz	0°C to 70°C
DS83C520–ECL	44–pin TQFP	33 MHz	0°C to 70°C

Ordering Information

Exemple du microcontrôleur Atmel 89C51

Speed (MHz)	Power Supply	Ordering Code	Package	Operation Range
12	5 V ± 20%	AT89C51-12AC AT89C51-12JC AT89C51-12PC AT89C51-12QC	44A 44J 40P6 44Q	Commercial (0°C to 70°C)
		AT89C51-12AI AT89C51-12JI AT89C51-12PI AT89C51-12QI	44A 44J 40P6 44Q	Industrial (-40°C to 85°C)
		AT89C51-12AA AT89C51-12JA AT89C51-12PA AT89C51-12QA	44A 44J 40P6 44Q	Automotive (-40°C to 125°C)
	5 V ± 10%	AT89C51-12DM AT89C51-12LM	40D6 44L	Military (-55°C to 125°C)
		AT89C51-12DM/883 AT89C51-12LM/883	40D6 44L	Military/883C Class B, Fully Compliant (-55°C to 125°C)
16	5 V ± 20%	AT89C51-16AC AT89C51-16JC AT89C51-16PC AT89C51-16QC	44A 44J 40P6 44Q	Commercial (0°C to 70°C)
		AT89C51-16AI AT89C51-16JI AT89C51-16PI AT89C51-16QI	44A 44J 40P6 44Q	Industrial (-40°C to 85°C)
		AT89C51-16AA AT89C51-16JA AT89C51-16PA AT89C51-16QA	44A 44J 40P6 44Q	Automotive (-40°C to 125°C)
20	5 V ± 20%	AT89C51-20AC AT89C51-20JC AT89C51-20PC AT89C51-20QC	44A 44J 40P6 44Q	Commercial (0°C to 70°C)
		AT89C51-20AI AT89C51-20JI AT89C51-20PI AT89C51-20QI	44A 44J 40P6 44Q	Industrial (-40°C to 85°C)

Décodage/interfaçage des périphériques

* INTERFAÇAGE AVEC LE MONDE EXTÉRIEUR :
  - ENTRÉES
  - SORTIES

  } LOGIQUES ou ANALOGIQUES

  ⇒ "PÉRIPHÉRIQUES" → à usage général
  → spécialisés

  → FONCTION D'ISOLATION
  (Tampons 3 état, "TRI-STATE")

  → EN SORTIE : FONCTION DE MÉMORISATION d'un état
  (d'un niveau logique)

  ⇒ typiquement : bascule D
  (Verrouillage d'un état)

  → ADRESSAGE NÉCESSAIRE

Décodage/interfaçage des périphériques

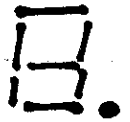
INTERFAÇAGE :

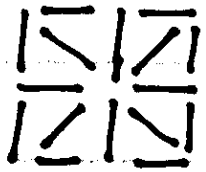
* SORTIES PARALLÈLES → COMMANDE
  → Commande de LED, afficheurs, puissance (moteurs)
* ENTRÉES PARALLÈLES → LECTURE
  → boutons poussoirs, claviers
* ENTRÉE / SORTIE SÉRIE
  → Asynchrone
  → Synchrone
* CONVERTISSEURS A/N et N/A
* TIMERS

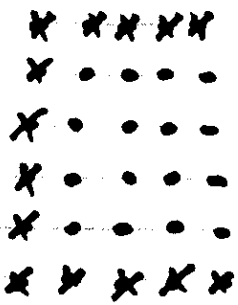
Décodage/interfaçage des périphériques

Grands types de périphériques (E/S) : pour µP et µC

→ les afficheurs

7 segments : 

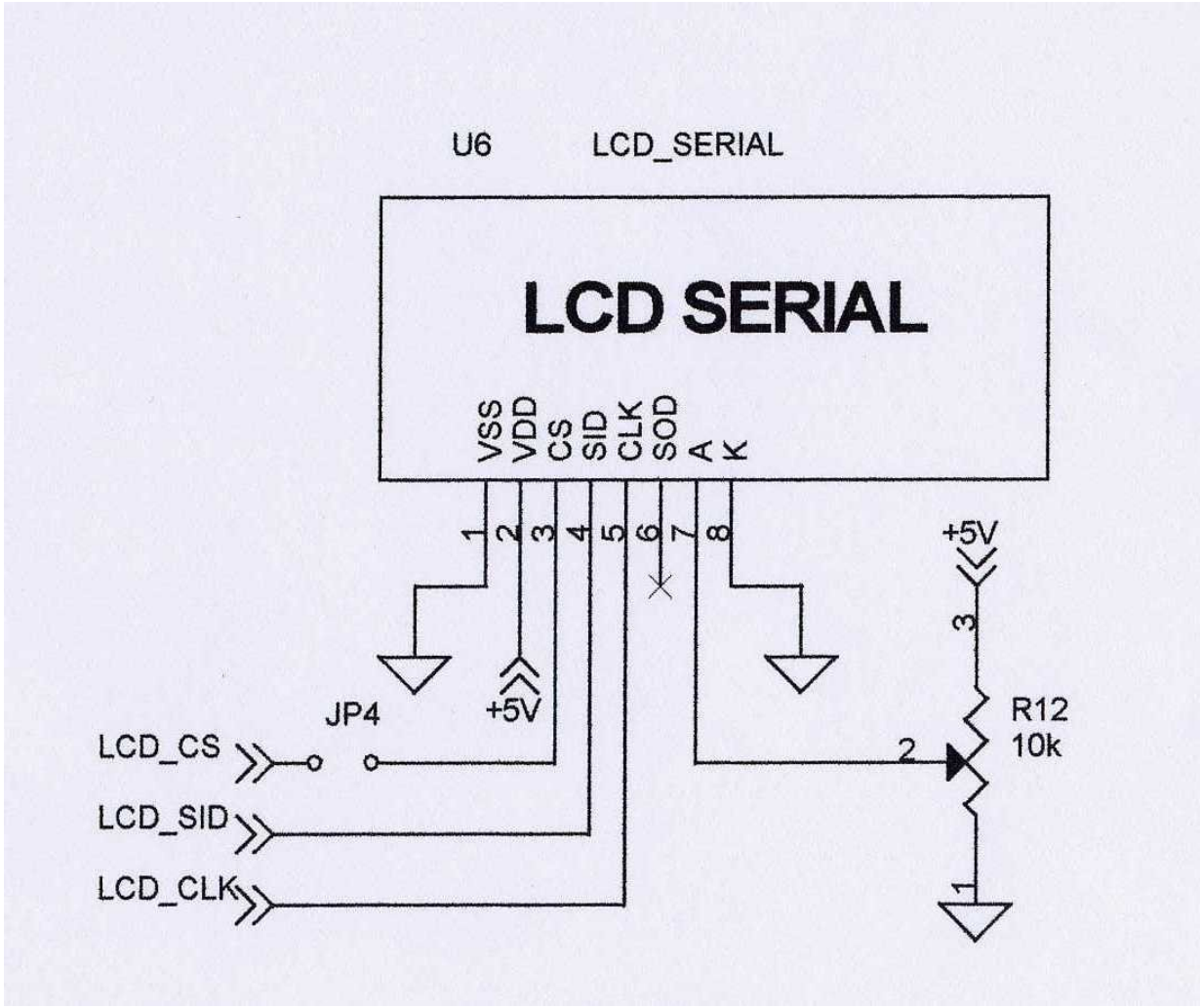
16 segments : 

Matriciel : 

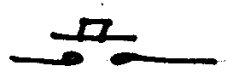
⇒ Controleur d'affichage LCD contrôle

Afficheurs
- → Numériques (7 segments par ex.)
- → Alphanumériques (1 ou plusieurs lignes de N caractères)
- → Graphiques : 400 × 640 points par ex.

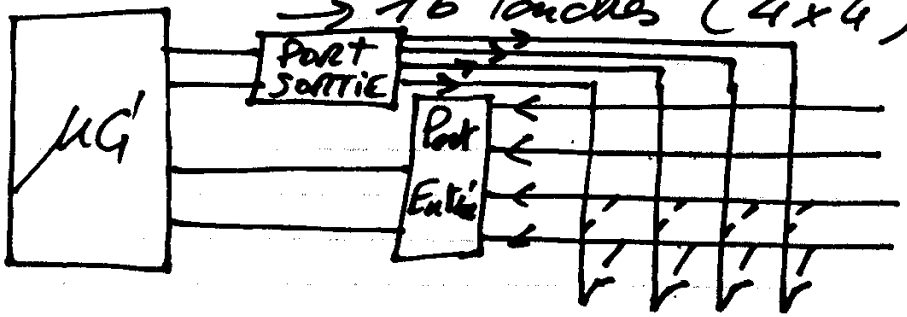
Exemple d'afficheur LCD (avec interface série)



Décodage/interfaçage des périphériques

→ les claviers : 

- l'interrupteur / le bouton poussoir
  - → cf circuit d'anti-rebond (bascule R,S)
- claviers industriels (rigides, souples, ...)
  - → 16 touches (4x4) par exemple



⇒ cf encodeur de clavier

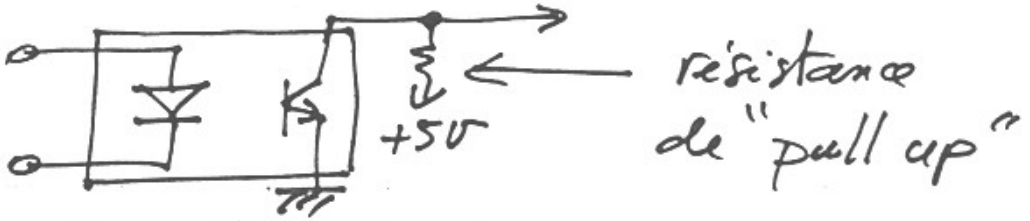
- → Interruption ↯ lors de l'appui sur une touche
- → anti-rebond interne
- → claviers 64 touches + shift + ...

Décodage/interfaçage des périphériques

Quelques éléments importants pour l'interfaçage :

→ entrées-sorties ISOLÉES

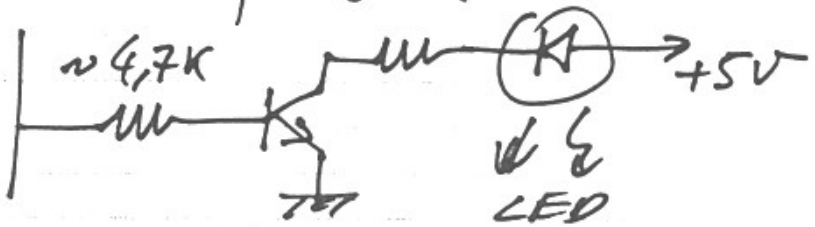
→ OPTO-COUPLEUR, OPTO-TRIAC, ...



→ dispositifs anti-rebonds

→ résistances de "pull up" et "pull down"

→ Commande de puissance



→ cf sorties "courant fort" de certains µC (ST6)

Interfaçage des périphériques

- Périphériques ou mémoires « compatibles microprocesseur » : accès aux registres internes, mémoires, ports, directement via les bus adresses/données :
  - Nécessitent un décodage classique
  - Gestion par « pulling », par interruption ou par DMA
- Périphériques ou mémoires sur bus série :
  - Interface série synchrone SPI : liaison 3 fils
    - MISO (Master In Slave Out) : esclave vers maître
    - MOSI (Master Out Slave In) : maître vers esclave
    - SCK (Serial Clock) : horloge du maître vers les esclaves
  - Bus I2C (*Inter Integrated Circuit, Philips*) : bus bifilaire
    - SDA (Serial Data) : IN ou OUT
    - SCL (Serial Clock)
    - Et la masse
  - Ethernet …
  - Voir les protocoles associés

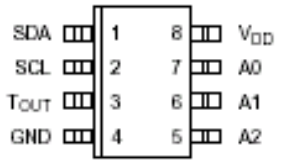
DALLAS SEMICONDUCTOR

DS1621
Digital Thermometer and Thermostat

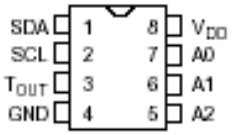
FEATURES

- Temperature measurements require no external components
- Measures temperatures from –55°C to +125°C in 0.5°C increments. Fahrenheit equivalent is –67°F to 257°F in 0.9°F increments
- Temperature is read as a 9–bit value (two byte transfer)
- Wide power supply range (2.7V to 5.5V)
- Converts temperature to digital word in 1 second
- Thermostatic settings are user definable and nonvolatile
- Data is read from/written via a 2–wire serial interface (open drain I/O lines)
- Applications include thermostatic controls, industrial systems, consumer products, thermometers, or any thermal sensitive system.
- 8–pin DIP or SOIC package (150 MIL and 208 MIL)

PIN ASSIGNMENT



DS1621S 8–PIN SOIC (150 MIL)
DS1621V 8–PIN SOIC (208 MIL)
See Mech. Drawings Section



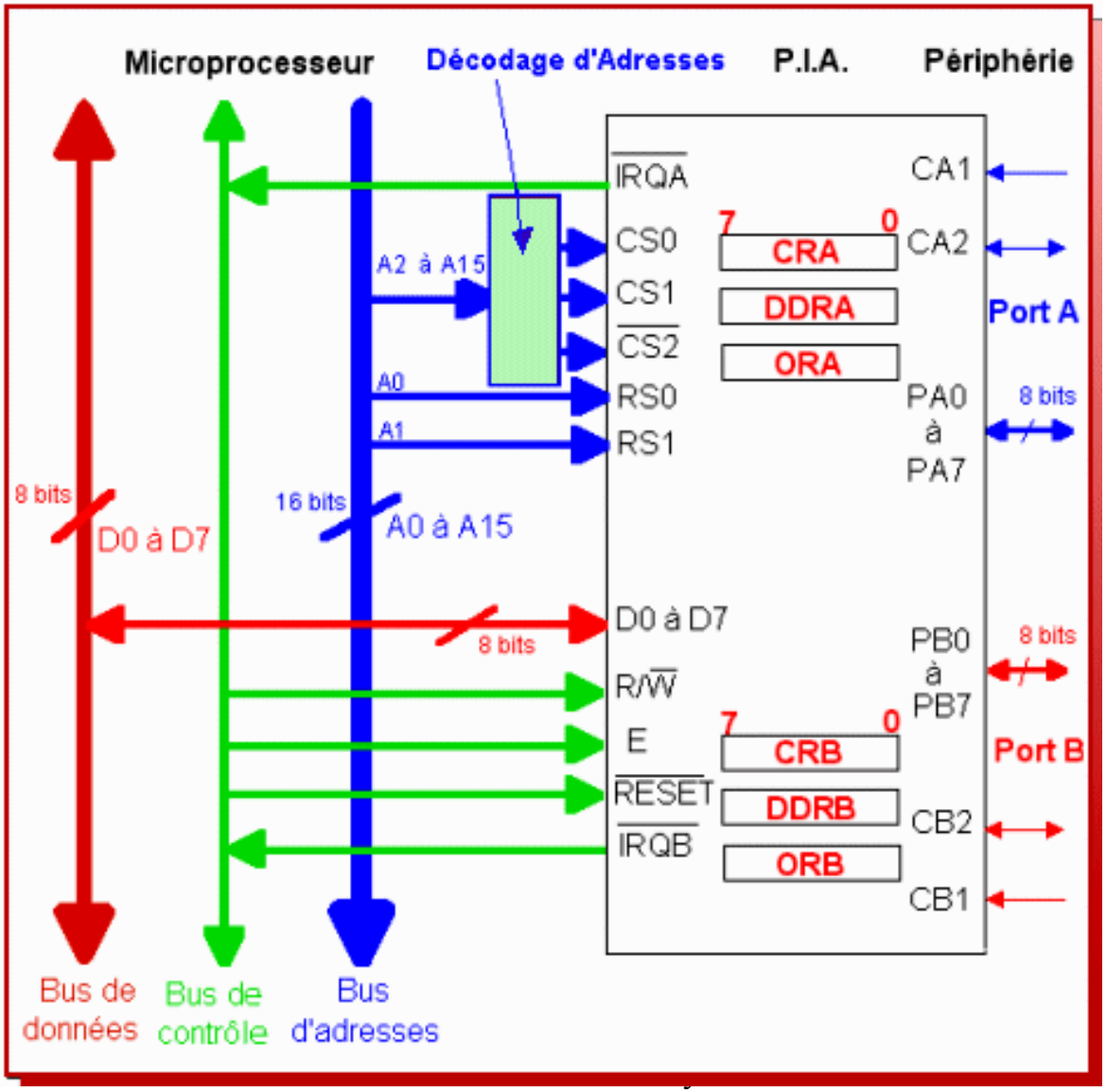
DS1621
8–PIN DIP (300 MIL)
See Mech. Drawings Section

PIN DESCRIPTION

SDA	–	2–Wire Serial Data Input/Output
SCL	–	2–Wire Serial Clock
GND	–	Ground
T_{OUT}	–	Thermostat Output Signal
A0	–	Chip Address Input
A1	–	Chip Address Input
A2	–	Chip Address Input
V_{DD}	–	Power Supply Voltage

Exemple de périphérique avec interfaçage de type I2C (thermomètre)

Exemple de périphérique : port parallèle (PIA)



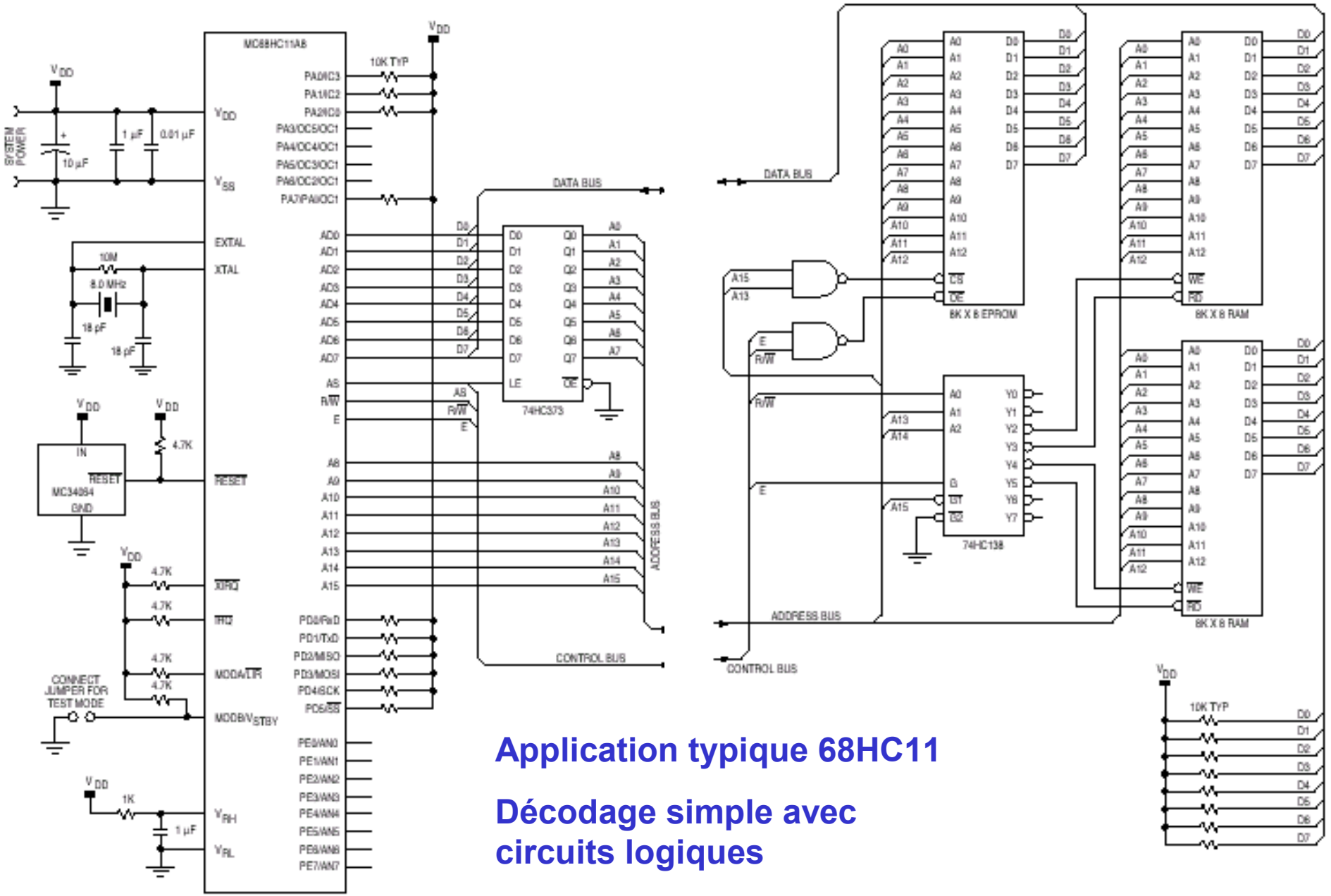
Type « port parallèle »

du PC

- registres internes accessibles en lecture ou écriture
- « interfaçable microP » : bus D0 – D7, R/W, Enable, Chip Select
- sorties interruption IRQ

Décodage

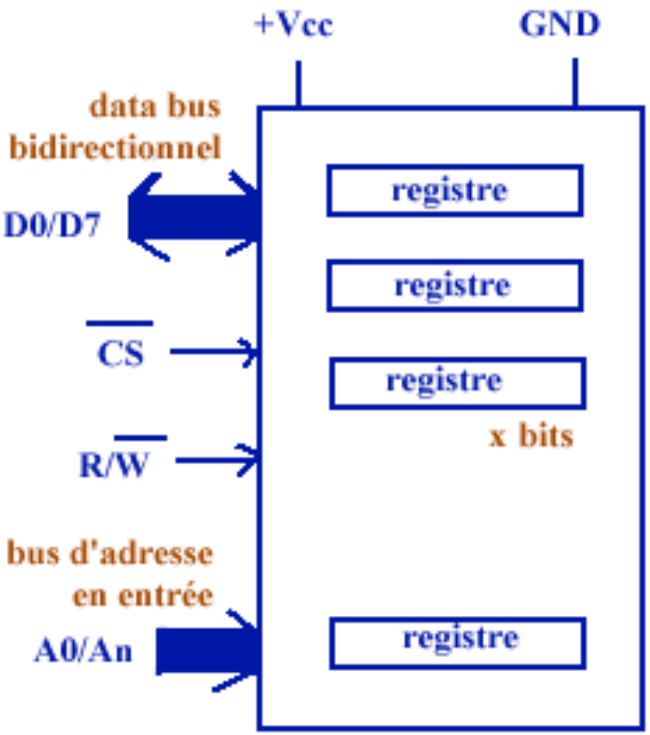
- **Pour accéder** (sélectionner en lecture ou en écriture à une adresse physique donnée) **à un périphérique** (interface série, //, convertisseurs A/D ou D/A, …) ou à un boîtier mémoire (ROM, RAM), **il faut réaliser un décodage** :
  - soit avec des circuits logiques classiques (portes)
  - soit avec des circuits logiques spécifiques (décodeurs)
  - soit avec des circuits logiques programmables (PAL, CPLD, …) = solution la pus moderne et la plus souple
- Le décodage doit prendre en compte le « timing » du microprocesseur :
  - Timing des signaux d'écriture externe (WR), lecture externe (RD), signaux de contrôle (adress latch enable ALE) : doc. constructeur
  - Temps d'accès des boîtiers périphériques ou mémoire externe : doc. constructeur
  - Prise en compte des retards dus aux portes, décodeurs
  - Utiliser des cycles d'attente (« wait state ») si nécessaire
- Penser à l'état du décodage à la mise sous tension et lors de la coupure d'alimentation
- Attention aux conflits de bus (plusieurs circuits sélectionnés en même temps) !

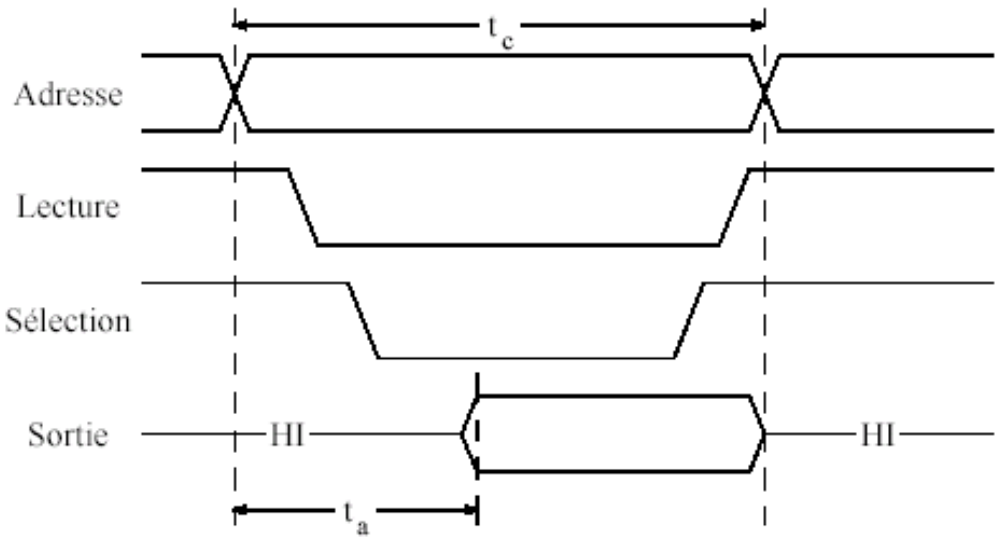


Application typique 68HC11

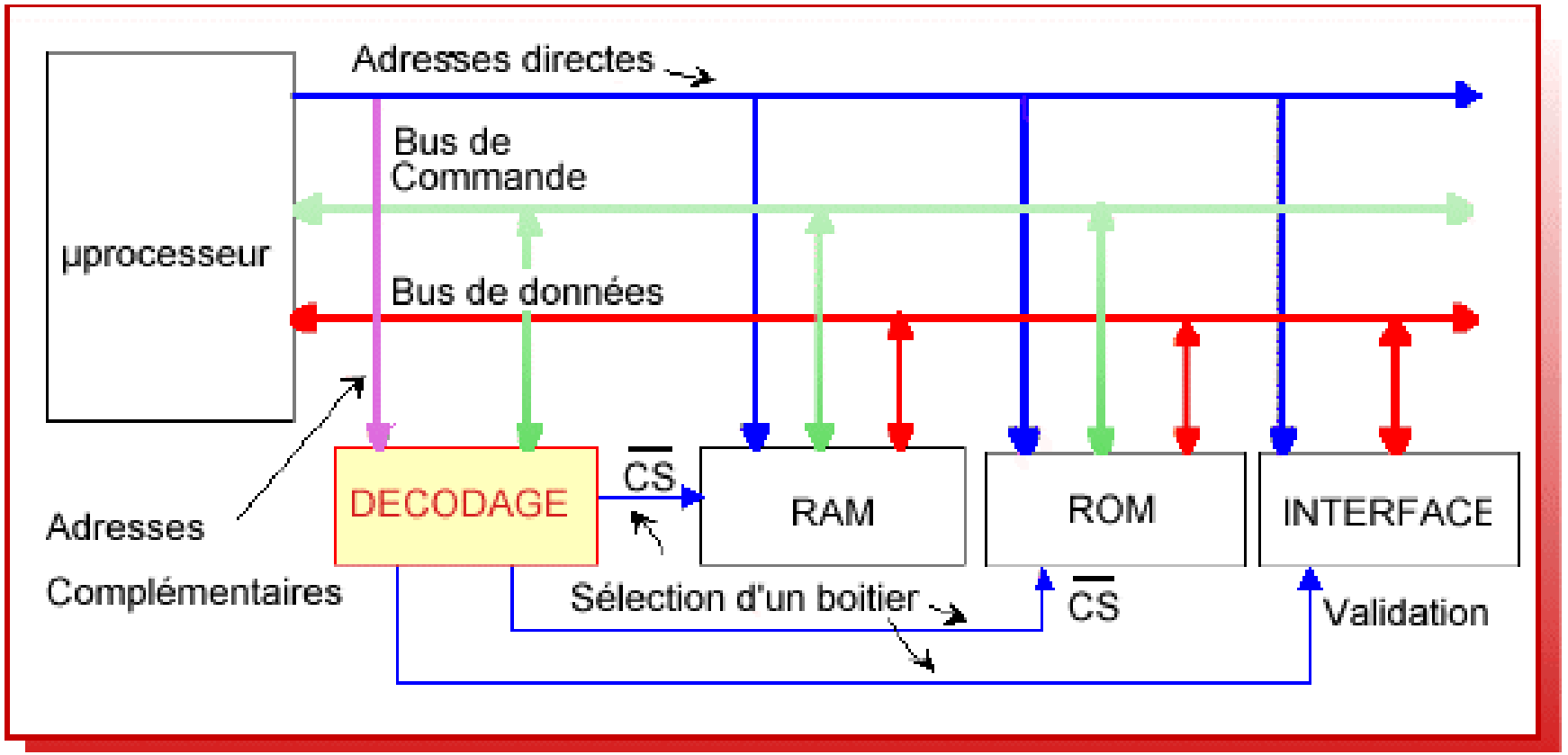
Décodage simple avec circuits logiques

Exemple de signaux : mémoire volatile type RAM





Principe du décodage



Interfaçage des périphériques des microprocesseurs

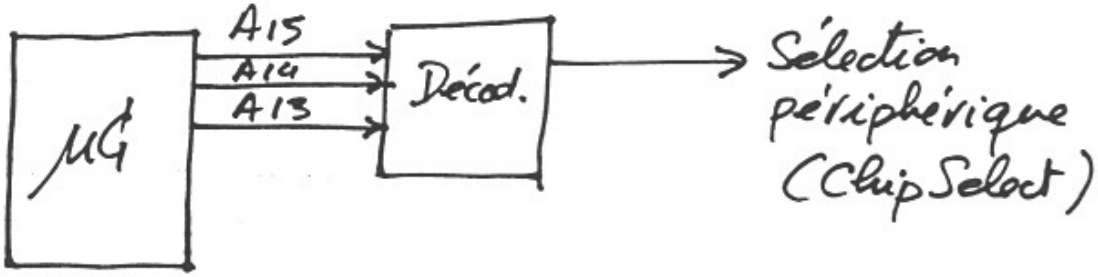
- Il y a 2 grands types d'adressage pour les périphériques :
  - *Memory mapped I/O* : pas de différence entre un périphérique (ou un de ses registres internes) et une adresse mémoire externe

    => attention au plan mémoire (mapping), recouvrements possibles
  - *I/O mapped I/O : possible seulement avec certains microprocesseurs disposant d'instructions spécifiques pour l'accès (lecture / écriture) aux entrées-sorties (I/O) : 8085 Intel, 8088, 20286, …. Un signal de contrôle spécifique (IO/M) permet de décoder la mémoire ou les périphériques, donc pas de recouvrement possible : 2 plans mémoires distincts (mémoire et I/O)*
- *Attention aux conflits de bus !*

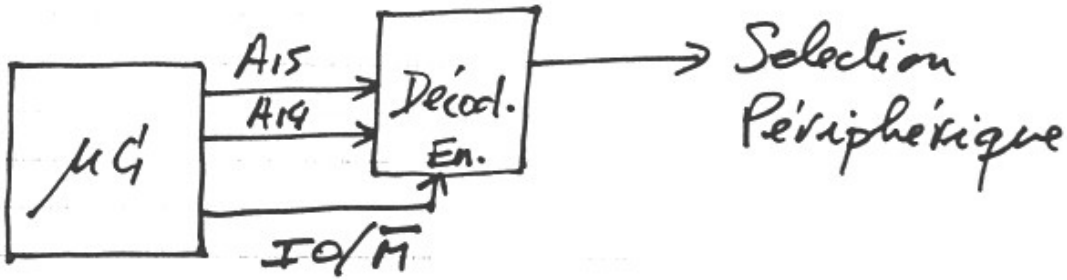
Décodage/interfaçage des périphériques

Exemples : L'Adressage / Le décodage diffère selon le mode (Memory Mapped ou I/O Mapped)

Memory Mapped



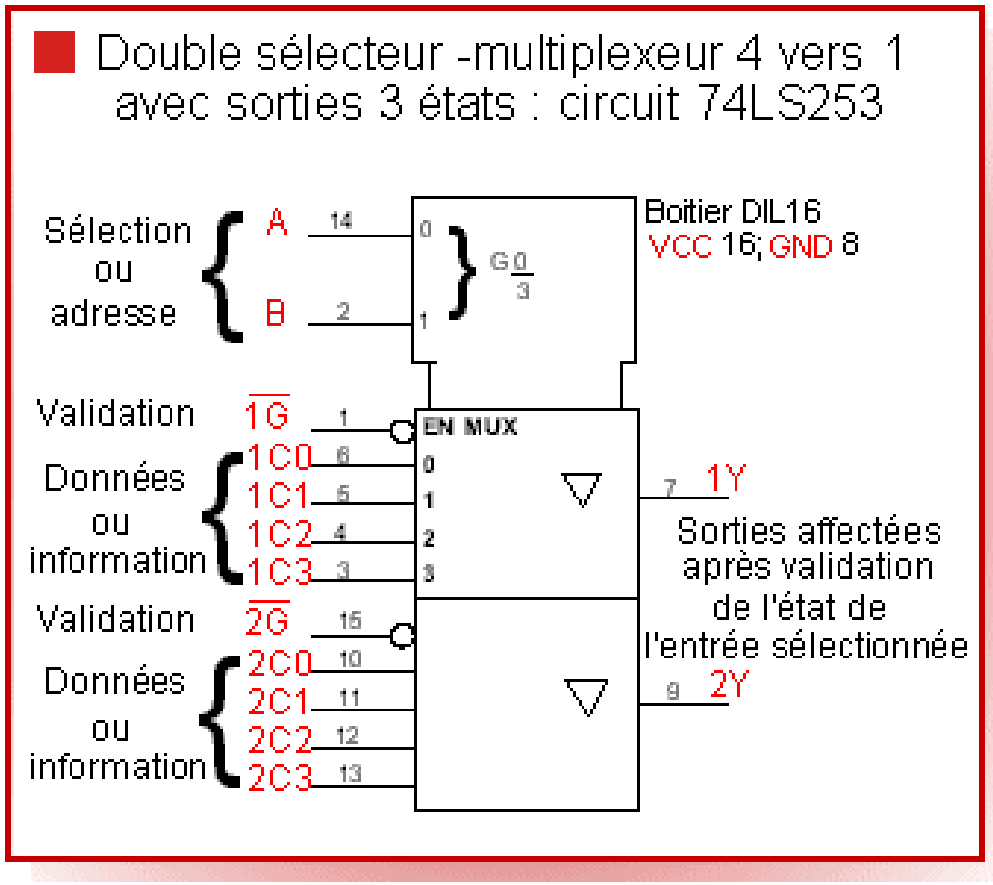
I/O Mapped (type Z 80)



→ ÉVITER LES CONFLITS DE BUS (plusieurs périphériques décodés en même temps)

Décodage des boîtiers mémoire externes :
exemple d'utilisation d'un décodeur

Décoder = sélectionner un périphérique (composant) et un seul
= écrire (ou lire) dans ce périphérique



Décodage utilisant un circuit « décodeur »

Solution plus moderne pour le décodage de périphériques :

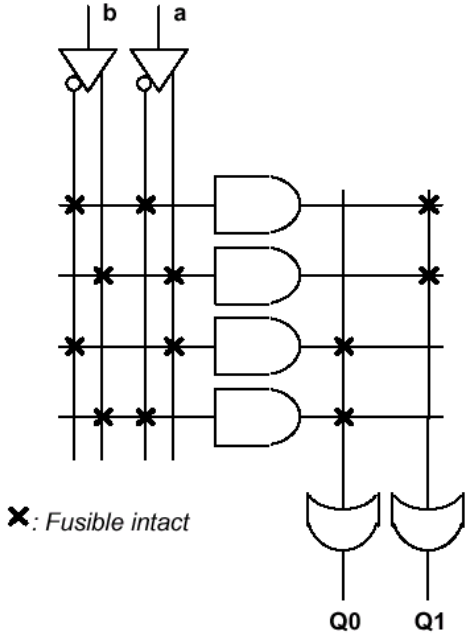
Utilisation de circuits logiques programmables (PAL, PLD)

Décodage des boîtiers mémoire externes : exemple d'utilisation d'un circuit programmable

Solution plus moderne pour le décodage de périphériques :

Utilisation de circuits logiques programmables (PAL, PLD)

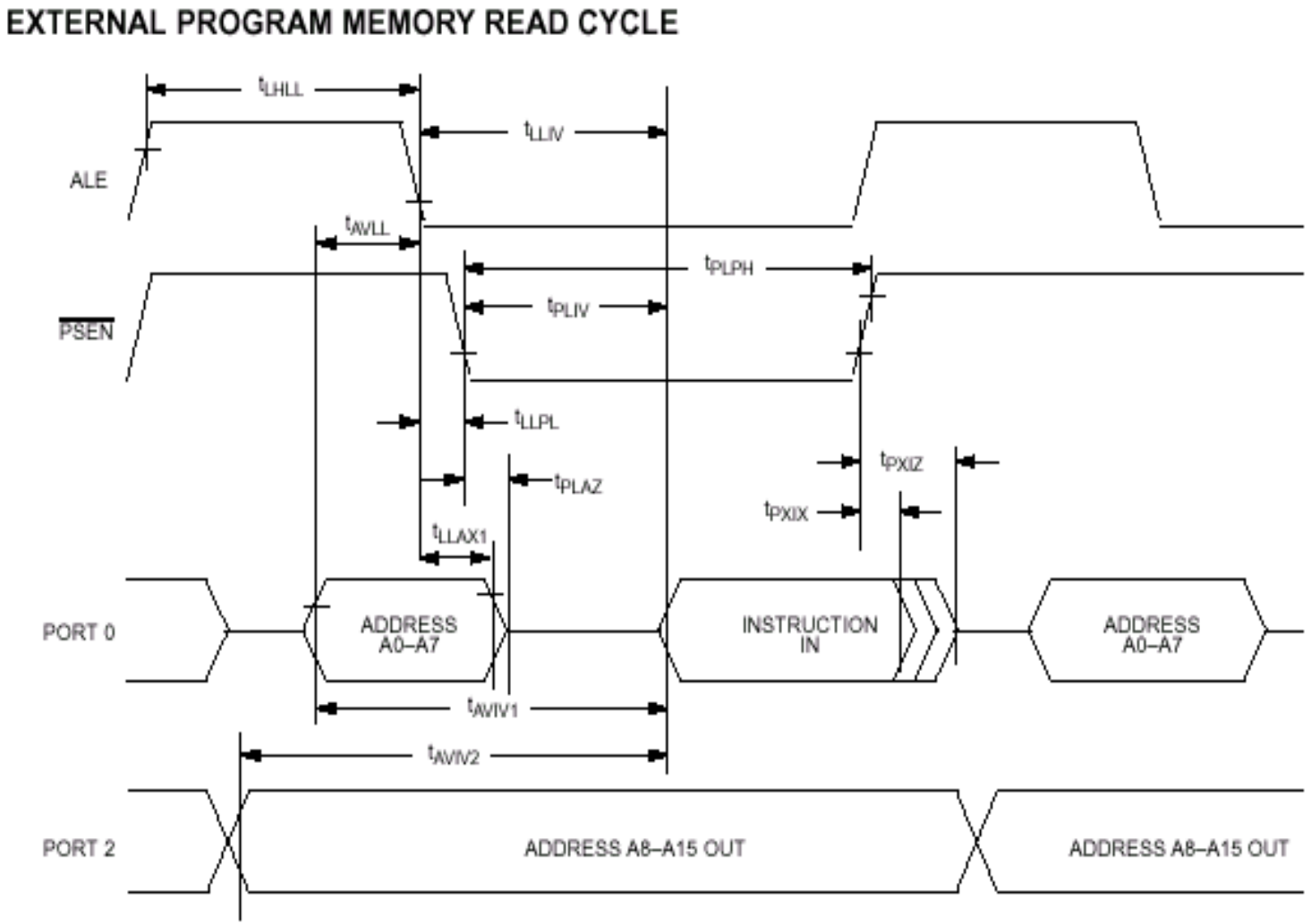
TYPE	Nombre de portes intégrées	Matrice ET	Matrice OU	Effaçable
PROM	2 000 à 500 000	Fixe	Programmable	Non
PAL	10 à 100	Programmable	Fixe	Non
GAL	10 à 100	Programmable	Fixe	Electriquement
EPLD	100 à 3000	Programmable	Fixe	Aux U-V
FPLA	2000 à 3000	Programmable	Programmable	Electriquement



- Figure 4 : PLD programmé -

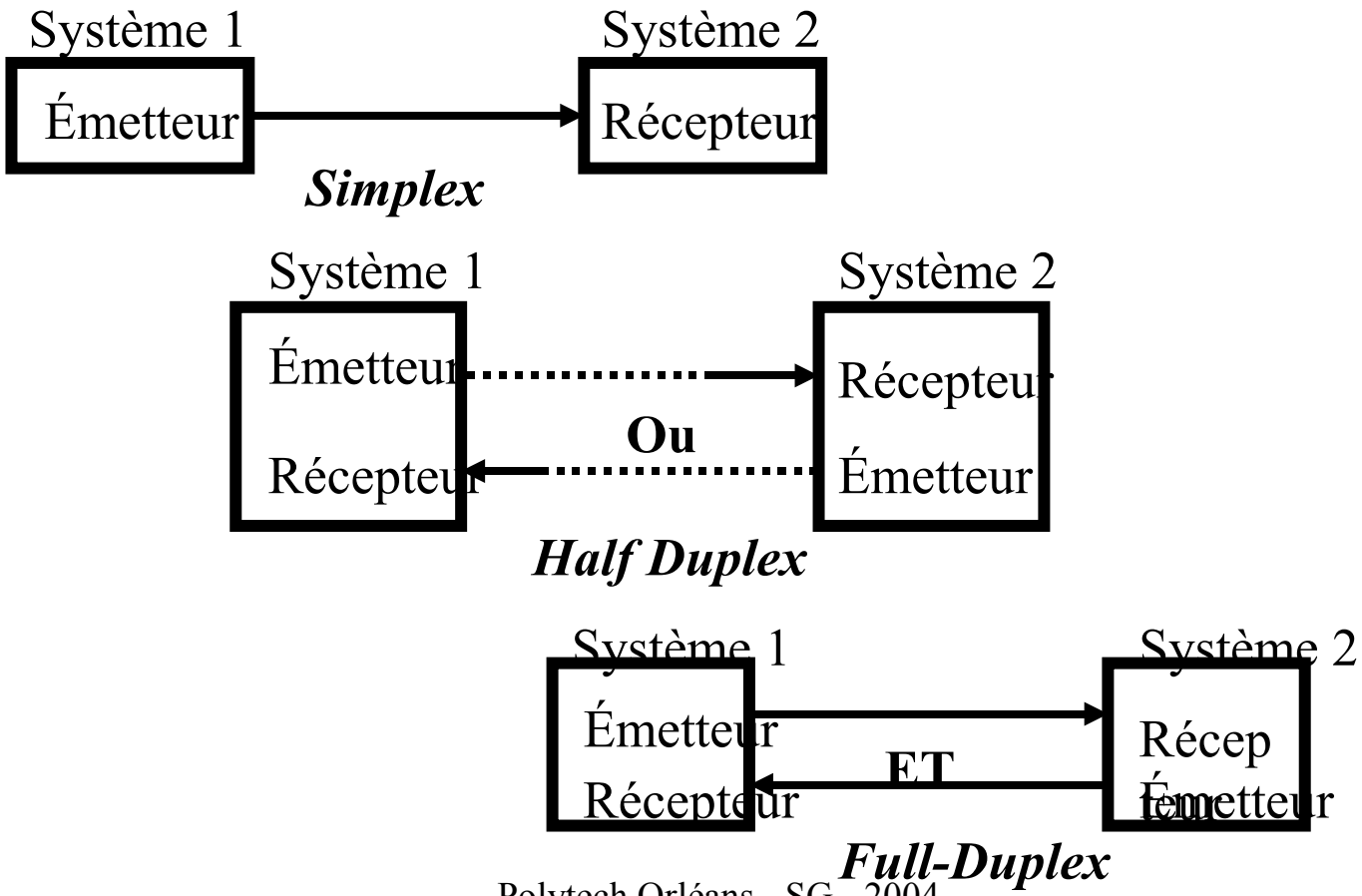
Avantages : reprogrammables (souplesse), gain de place (moins de composants), fiables

Timing microcont (8051) : accès mémoire externe
Décodage / accès aux périphériques et mémoires



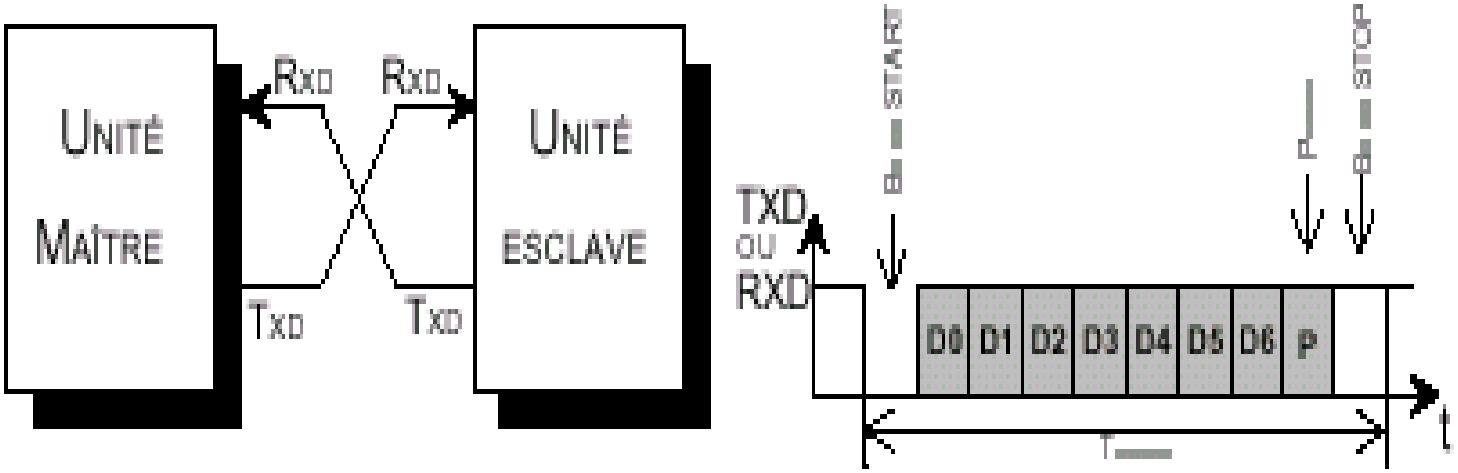
Liaisons séries : modes d'échanges

- Simplex / half duplex / full duplex



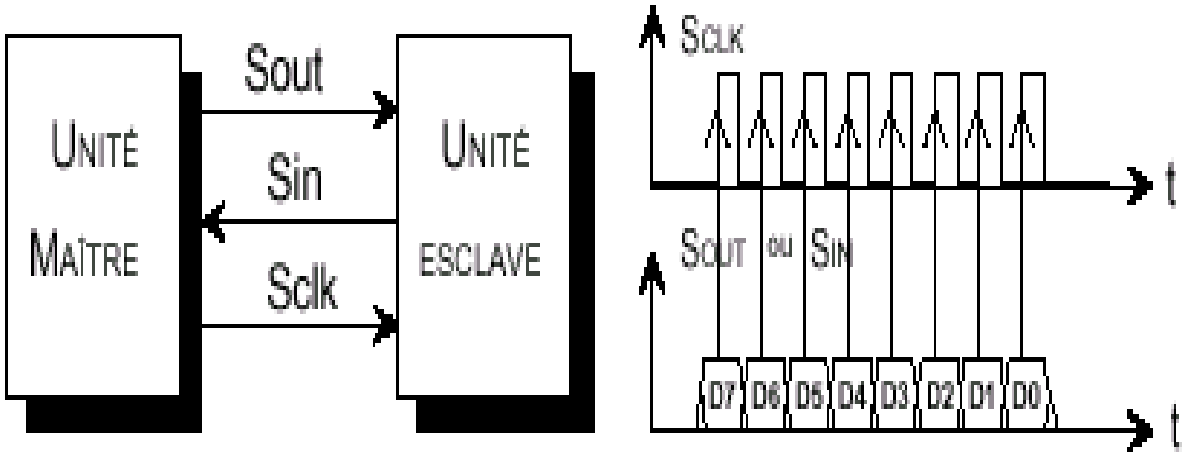
Liaison série asynchrone

- **type « RS232 » (COM1 du PC par exemple), horloges Rx et Tx # égales**
- **start bit / stop bit : permettent de détecter l'arrivée d'un caractère**
- **bit de parité éventuel (somme des bits à 1 : si nombre paire parité paire, ..). Permet la détection des erreurs de transmission.**
- **plus lent que liaison synchrone, plus simple**
- **les caractères peuvent arriver de façon aléatoire, à des intervalles quelconques**



Liaison série synchrone

- horloge de transmission (Tx) = horloge de réception (Rx)
  - transmission de l'horloge (par fil séparé)
  - ou synchro. du récepteur par le signal (utilisation de PLL)
- données émises par blocs (trames) et suivant un protocole de transmission (les caractères se suivent sans séparation)
- caractères de synchronisation en début et fin de blocs
- Détection d'erreur par bloc avec CRC (cyclic redundancy check)
- utilisation d'une pile d'émission FIFO
- Exemple : bus SPI



Autres modes d'échanges

- Transmissions parallèles : bus des ordinateurs ISA, PCMCIA, VME, …
- Transmissions de type « réseau » :
  - Point à Point
  - Multipoint
  - Topologies de réseau en étoile, en anneau, …

Liaison série

* LIAISONS SERIE → asynchrone : intervalle de temps entre l'envoi de 2 octets quelconques

→ Synchrone : données émises par bloc

⇒ codage (ASCII), protocoles, normes, erreurs

* Série asynchrone : registre à décalage + horloge

Registre à décalage de 10 bits
1 START bit + 8 bits + 1 STOP bits

(* cf bit de parité optionnel)

ENTRÉE :
- horloge de réception ≃ horloge d'émission (5% max)
- détection d'un START bit, puis de chaque bit

Liaison série

* liaison série synchrone : par blocs (trame) → plus rapide, plus complexe
  ⇒ horloge de réception = horloge d'émission
  → pas de start et stop bits, mais des caractères de synchronisation en début et fin de bloc
  ⇒ soit l'horloge est transmise (fil spécifique)
  ⇒ soit les horloges récept° et émiss° sont synchronisées (par PLL, par radio, ...)
  → utilisation de piles FIFO (First In First Out)
  → détection d'erreur performante en général (CRC : Cyclic Redundancy Check)

Tableau 2 – Caractéristiques électriques des normes RS 232C, RS 422, RS 423

Configuration	RS 232C Entrée et sortie référencées par rapport à la masse	RS 423 Entrée et sortie référencées par rapport à la masse	RS 422 Structure différentielle
Longueur max du câble	50 fe et (15,2 m)	4 000 fe et (1 200 m)	4 000 fe et (1 200 m)
Vitesse max de transfert de données	20 kbit/s	100 kbit/s à 40 feet	10 Mbit/s à 40 feet
Tension de sortie (sortie ouverte)	± 3 V à ± 25 V	± 4 V à ± 6 V	≤ 6 V entre les bornes A et B
Tension de sortie en charge (V_t)	± 5 V à ± 15 V charge de 3 à 7 kΩ	$\|V_t\| \geq 9 \|V_0\|$ charge de 450 Ω	2 V entre A et B en charge équilibrée de 100 Ω
Courant de sortie en court-circuit (I_s)	$\|I_s\| \leq$ ± 500 mA	$\|I_s\| \leq$ ± 150 mA	$\|I_s\| \leq$ ± 150 A
Vitesse de balayage (dV/dt)	30 V/µs max		Contrôle non nécessaire
Mark (*off* = 1) du récepteur	≤ – 3 V	A négatif par rapport à B'	A < B
Space (*on* = 0) du récepteur	≥ + 3 V	A positif par rapport à B'	A > B
Seuil de tension d'entrée du récepteur	± 3 V	200 mV mode différentiel	200 mV mode différentiel
Impédance d'entrée du récepteur	3 à 7 kΩ, 2 500 pF	≥ + 4 kΩ	≥ 4 kΩ
Dynamique de tension d'entrée du récepteur	– 25 V à + 25 V	– 12 V à + 12 V	– 12 V à + 12 V

Caractéristiques principales des normes de liaison série

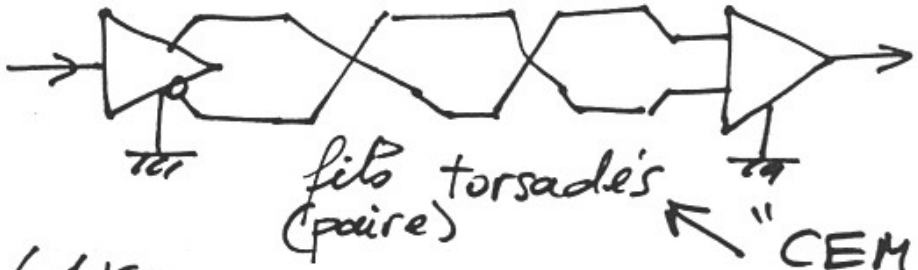
- RS232C
- RS423
- RS422 (différentiel)

Liaison série asynchrone : normes « RS »

Liaisons série : Normes et protocoles

NORME
* RS-232C : entre ±5V et ±15V
  ↳ "la" liaison série type LC (COM1, COM2...)

NORME
* RS-422 : plus rapide, mode différentiel



≈ 10 Mb / 1 Km

NB : isolation galvanique simple (opto coupleur, fibre optique plastique ou silice)

* Boucle de courant : "4-20 mA" ou "0-20 mA"

niveau 0 : 0 mA
niveau 1 : 20 mA
} 0-20 mA

- longues distances
- forte immunité au bruit

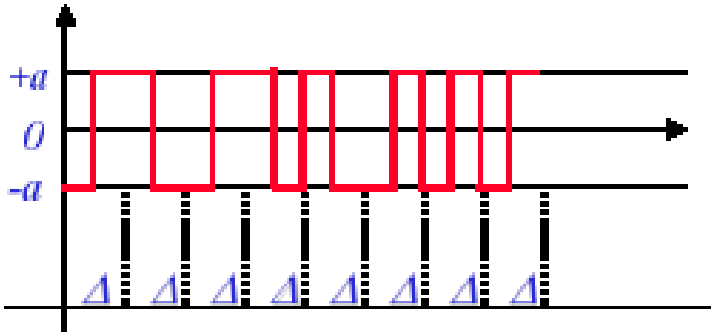
Notions de codage de l'information

- Nécessite de coder l'information pour adapter le signal émis au support de transmission (fil, fibre, radio, IR, …) : tenir compte de la bande passante du canal (câble), du bruit (radiofréquences), des impédances (téléphonie), …

Types	Bande Passante	Utilisation
Paire Torsadée (TP)	> 100 kHz	Téléphonie, LAN (UTP, STP)
Câble coaxial	> 100 MHz	Télévision, LAN, (MAN ?)
Fibre Optique	> 1 GHz	LAN, MAN et WAN (monomode #60 km, Xmode #2 km)
Faisceaux Hertziens	Variable (nature et fréquence)	MAN, LAN
Satellites	X canaux > 10 MHz	WAN
LAN: Local Area Network	*MAN: Metropolitan Area Netw.*	*WAN: Wide Area Network*

Notions de codage de l'information

- **Coder le signal binaire (0,1) :**
  - en **2 niveaux** (-V,+V) :
    - **codage NRZ** (0,1) => (-V,+V)
    - **Codage Manchester** : on introduit des transitions au milieu de chaque bit pour synchroniser l'horloge de réception. Codage de 01011000 :



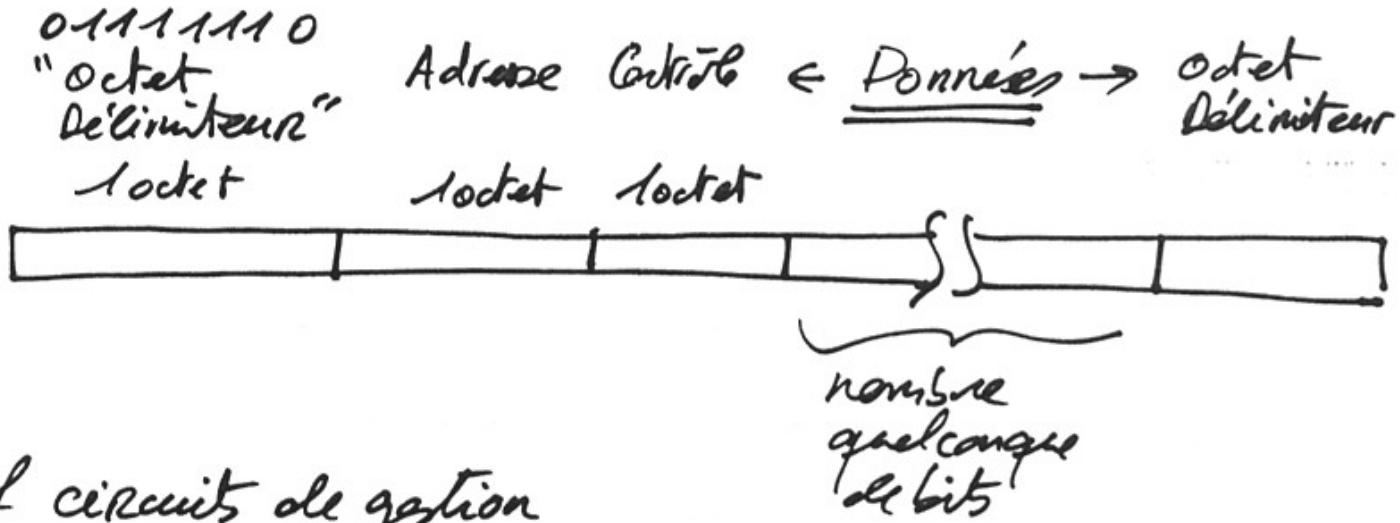
  - en **3 niveaux** (-V,0 volt,+V) :
    - Codage bipolaire simple : « 0 » => 0 volt
      et « 1 » vaut alternativement –V ou +V

Liaisons série : protocoles

Liaisons série : PROTOCOLES de transmission

protocoles → orienté bits (SDLC, HDLC) : Synchr.
↳ orienté octets (DDCMP) : Asyn. ou Sync

* HDLC : Trame HDLC



cf circuits de gestion de protocole (Intel, Motorola, ...)

Format interne des données : langage machine

- Etats logiques « haut » et « bas » : en logique positive (0, 5 volts)
  - **« 0 » logique = niveau bas (0 volt généralement)**
  - **« 1 » logique = niveau haut (5 volts généralement)**
- Les **pseudo-instructions** sont des instructions que l'assembleur doit exécuter. Pour cela, il utilise ses **registres internes.**
- Dans le **programme** (le « code »), on distingue trois types de données :
  - les données immédiates (400h, ...)
  - les constantes : définir l'identificateur et le type de la variable
  - les zones de mémoire réservées (pour des variables, des tableaux de variables par exemple)
- Une **donnée** peut être :

  Identificateur

  Mnémonique de l'instruction

  Opérande

  Commentaire

Le langage assembleur

NOTIONS DE BASE :

- Le « langage» est un code qui permet de converser avec la machine (le microprocesseur ou le microcontrôleur ici).

On peut distinguer **trois niveaux** d'utilisation :

langage machine, langage assembleur, langage évolué

- ***niveau 0 : langage machine*** , une instruction est codée en binaire et stockée sur un ou plusieurs octets (dans la mémoire programme).
  - Il est lié au fonctionnement réel (interne) de la machine .
  - Il dépend donc du type de machine (ou microprocesseur) .
    Ex 1 : 1 1 0 1 0 0 0 0 1 0 1 1 1 0 0 0

- ***niveau 1 : langage assembleur***, les instructions utilisent des **mnémoniques** plus faciles à mémoriser et interpréter par l'opérateur humain.
  - Il dépend encore de la machine, mais certaines ressemblances existent (ADD, LD, …)
  - L'assembleur va traduire ce langage de niveau 1 en code machine
    Ex 2 : ADD # 66,D0 équivaut à l'exemple 1

Exemple de code machine et assembleur 68HC11

Machine Code	Label	Operation	Operand	Comments
	CAT	EQU	7	CAT SAME AS 7
		ORG	$1000	SET LOCATION COUNTER
	RBGS	EQU	*	ADDR(RBGS) IS $1000
86 16		LDAA	#22	DECIMAL 22 ⇒ ACCA ($16)
CB 34		ADDB	#$34	XOR ($34,ACCB) ⇒ ACCB
81 24		CMPA	#%100100	RIGHT ALIGNED BINARY
86 07		LDAA	#CAT	7 ⇒ ACCA
CC 12 34		LDD	#$1234	
CC 00 07		LDD	#7	7 ⇒ ACCA:ACCB
86 12		LDAA	#@22	OCTAL
86 41		LDAA	#'A	ASCII
CE 10 00		LDX	#RBGS	ADDR(RBGS) ⇒ X

Le langage assembleur (suite)

- ***niveau 2 : langage évolué***, ou langage de haut niveau.

  Par exemple PASCAL, FORTRAN, C, C++, COBOL,JAVA, ...

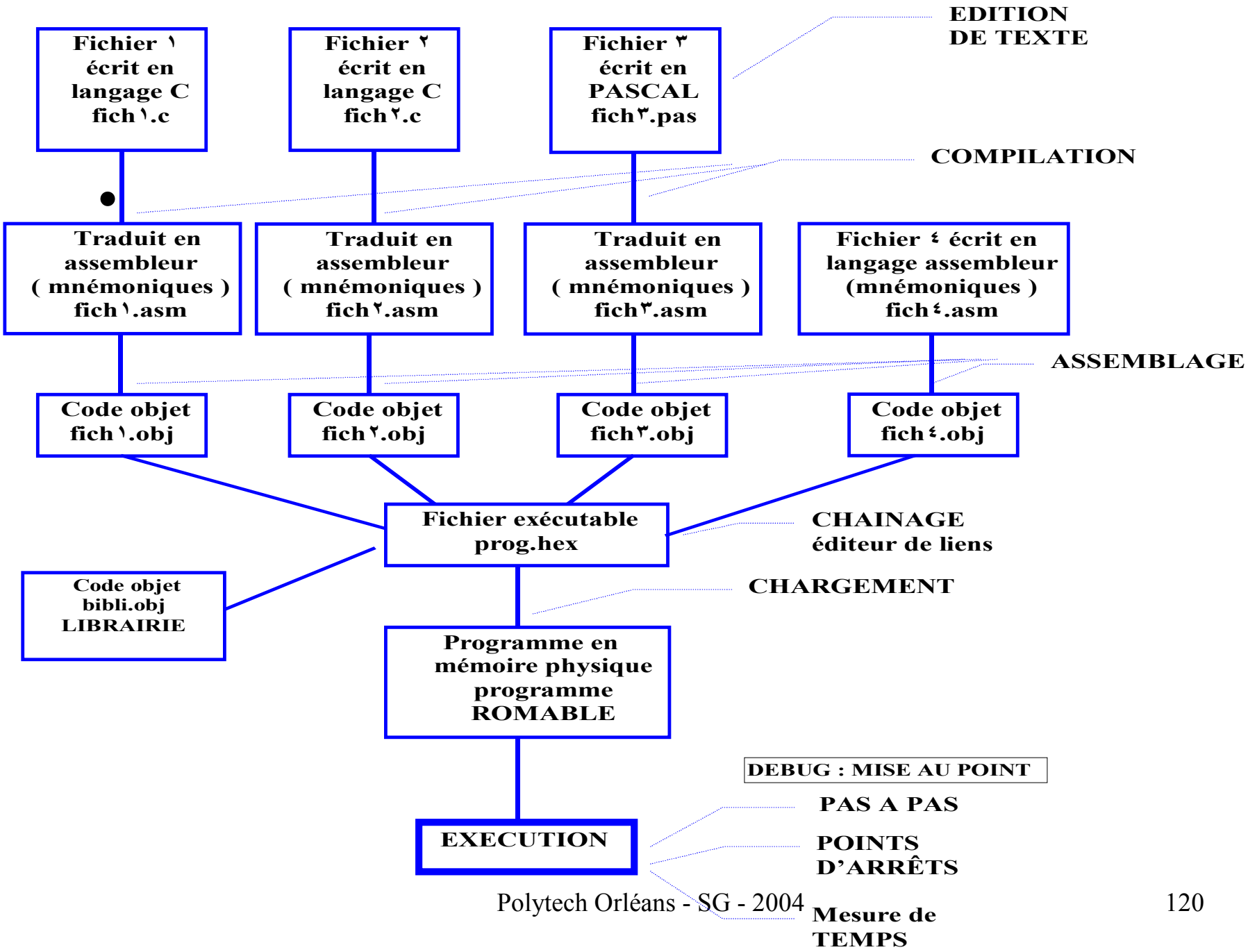
  - Il est quasiment standardisé (C de Kernighan et Richie, norme ANSI),
  - Une ligne de programmation en C, par exemple, va être traduite par le compilateur en une suite d'instructions assembleur, donc comprises par l'outil assembleur.
  - Il existe de très bonnes bibliothèques (librairies) en assembleur ou langage évolué : calcul, traitement du signal, compression d'images, …

- Un programme réalisant les mêmes fonctions, les mêmes calculs (avec les mêmes résultats) peut être écrit en langage machine, en langage assembleur ou en langage évolué. Les différences seront essentiellement :
  - Taille du programme
  - Rapidité du programme
  - Portabilité
  - Maintenabilité
  - Temps de développement
  - Documentation

*Le **compilateur*** **est donc spécifique à la machine et lié au langage** évolué, mais plusieurs compilateurs peuvent générer un code compatible .

C'est ***l'éditeur de liens*** (*linker*)qui va, entre autre, permettre de fabriquer un seul **programme exécutable** à partir de codes générés par ces compilateurs différents.

Phases de traduction d'un programme



Instructions assembleur

- L ’instruction assembleur comprend :
  - un **code opératoire** (type d ’instruction) : (op)
  - un ou des **opérandes** (source ou destination, donnée ou adresse)

- Les instructions assembleurs comportant en général 2 opérandes, l'instruction est en fait de type :

(opérande destination) (op) (opérande source) -> (opérande destination)

ou pour les instructions d'échange de données :

(opérande source) -> (opérande destination)

- La façon de spécifier la source et la destination des données s ’appelle **le mode d ’adressage.** Un microprocesseur puissant possédera des modes d’adressages complexes (échanges de données rapides, boucles, indexation).

Equivalence langage évolué - assembleur

langage C : Compilateur GCC (GCC ANSI)

```
x=0;
do {
      x++;
      if (x<5000)
            {
            *lights = red;      /* Commentaire */
            continue;
            }
      if (x>7000)
            {
            *lights = green;
            continue;
            }
      *lights = 0;
      }while(1);
```

Assembleur 68000 : Assembleur 68000 (* commentaire)

```
          clr.l     D0
loop      add.l     #1,D0
          cmp.l     #5000,D0
          blt       toolow
          cmp.l     #7000,D0
          bgt       toohigh
          move.b    #0,lights
          bra       loop
toohigh   move.b    #red,lights
          bra       loop
toolow    move.b    #green,lights
          bra       loop
```

Exemple du PIC (Microchip)

Processeur RISC

PIC18FXX2

TABLE 20-2: PIC18FXXX INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word MSb	LSb	Status Affected	Notes
BYTE-ORIENTED FILE REGISTER OPERATIONS							
ADDWF	f, d, a	Add WREG and f	1	0010 01da	ffff ffff	C, DC, Z, OV, N	1, 2
ADDWFC	f, d, a	Add WREG and Carry bit to f	1	0010 00da	ffff ffff	C, DC, Z, OV, N	1, 2
ANDWF	f, d, a	AND WREG with f	1	0001 01da	ffff ffff	Z, N	1,2
CLRF	f, a	Clear f	1	0110 101a	ffff ffff	Z	2
COMF	f, d, a	Complement f	1	0001 11da	ffff ffff	Z, N	1, 2
CPFSEQ	f, a	Compare f with WREG, skip =	1 (2 or 3)	0110 001a	ffff ffff	None	4
CPFSGT	f, a	Compare f with WREG, skip >	1 (2 or 3)	0110 010a	ffff ffff	None	4
CPFSLT	f, a	Compare f with WREG, skip <	1 (2 or 3)	0110 000a	ffff ffff	None	1, 2
DECF	f, d, a	Decrement f	1	0000 01da	ffff ffff	C, DC, Z, OV, N	1, 2, 3, 4
DECFSZ	f, d, a	Decrement f, Skip if 0	1 (2 or 3)	0010 11da	ffff ffff	None	1, 2, 3, 4
DCFSNZ	f, d, a	Decrement f, Skip if Not 0	1 (2 or 3)	0100 11da	ffff ffff	None	1, 2
INCF	f, d, a	Increment f	1	0010 10da	ffff ffff	C, DC, Z, OV, N	1, 2, 3, 4
INCFSZ	f, d, a	Increment f, Skip if 0	1 (2 or 3)	0011 11da	ffff ffff	None	4
INFSNZ	f, d, a	Increment f, Skip if Not 0	1 (2 or 3)	0100 10da	ffff ffff	None	1, 2

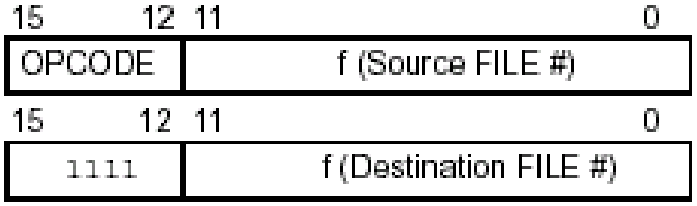
FIGURE 20-1: GENERAL FORMAT FOR INSTRUCTIONS

Byte-oriented file register operations — Example Instruction

ADDWF MYREG, W, B

d = 0 for result destination to be WREG register
d = 1 for result destination to be file register (f)
a = 0 to force Access Bank
a = 1 for BSR to select bank
f = 8-bit file register address

Byte to Byte move operations (2-word)



MOVFF MYREG1, MYREG2

f = 12-bit file register address

Bit-oriented file register operations

BSF MYREG, bit, B

b = 3-bit position of bit in file register (f)
a = 0 to force Access Bank
a = 1 for BSR to select bank
f = 8-bit file register address

20.1 Instruction Set

ADDLW — **ADD literal to W**

Syntax:	[*label*] ADDLW k
Operands:	$0 \le k \le 255$
Operation:	$(W) + k \rightarrow W$
Status Affected:	N, OV, C, DC, Z

Encoding:

0000	1111	kkkk	kkkk

Description: The contents of W are added to the 8-bit literal 'k' and the result is placed in W.

Words: 1

Cycles: 1

Q Cycle Activity:

Q1	Q2	Q3	Q4
Decode	Read literal 'k'	Process Data	Write to W

Example:

```
ADDLW   0x15
```

Before Instruction

W = 0x10

After Instruction

W = 0x25

Le langage assembleur (suite)

- Une adresse peut être spécifiée de multiples façons, et ce pour l'ensemble des instructions : *mode d'adressage*
- Instructions assembleur : on peut distinguer (exemple du 68000 Motorola) :
  - les instructions de base (LD, MOV)
  - les instructions arithmétiques (DEC, ADC, ADD, SBB, SUB, MUL, IMUL, DIV, IDIV, NEG, INC)
  - les instructions logiques (AND, OR, XOR, NOT, TEST)
  - les instructions de décalage et de rotation (SHL, SHR, SAR, SAL)
  - les instructions de pile (PUSH, POP, PUSHF, POPF)
  - les instructions de manipulation de chaînes de caractères (CLD, STD, MOVSB, MOVSW, REP ...,

Le langage assembleur (suite)

- **Les instructions de saut conditionnel et inconditionnel (JMP, INT)**
- **Les instructions de test :**
  - **Les tests d'identificateur (JZ, JE, JNZ, JNE, JC, JNC, JS, JNS, JO, JNO)**
  - **Les tests arithmétiques (CMP)**
  - **Les tests de nombres non signés (JA, JAE, JC, JNA, JNAE, JNBE, JNC, JNZ, JZ)**
  - **Les tests de nombres signés (JE, JG, JGE, JL, JLE, JNE, JNL, JNLE, JNG, JNGE, JNZ, JZ)**
- **Les instructions de boucle (LOOP, LOOPE, LOOPNE)**
- **Les procédures (PROC, CALL, RET) et leurs paramètres**
- **Les instructions de gestion des adresses (LEA, LES, LDS)**

Exemple de modes d ’adressage (type 68HC11)

- Modes d ’adressage classiques :
  - **adressage direct**
    - l ’adresse réelle est indiquée dans l’instruction : 2 octets (16 bits) permettent d ’accéder directement à 64 Koctets de mémoire (2^{16} = 65536)
    - direct relatif : un seul octet ajouté au PC permet d ’accéder à une page de la mémoire (2^{8} = 256 octets). Code plus court, instruction plus rapide
  - **adressage indirect** : fonctionnement type « pointeur »
  - **adressage indexé** : la valeur du registre d ’index est ajoutée à l ’adresse calculée (déplacement)

Modes d ’adressage du 68HC11 (extraits)

Function	Mnemonic	IMM	DIR	EXT	INDX	INDY	INH
Clear Memory Byte	CLR			X	X	X	
Clear Accumulator A	CLRA						X
Clear Accumulator B	CLRB						X
Load Accumulator A	LDAA	X	X	X	X	X	
Load Accumulator B	LDAB	X	X	X	X	X	
Load Double Accumulator D	LDD	X	X	X	X	X	
Pull A from Stack	PULA						X
Pull B from Stack	PULB						X
Push A onto Stack	PSHA						X
Push B onto Stack	PSHB						X
Store Accumulator A	STAA	X	X	X	X	X	
Store Accumulator B	STAB	X	X	X	X	X	
Store Double Accumulator D	STD	X	X	X	X	X	
Transfer A to B	TAB						X
Transfer A to CCR	TAP						X
Transfer B to A	TBA						X
Transfer CCR to A	TPA						X
Exchange D with X	XGDX						X
Exchange D with Y	EGDY						X

ADC Add with Carry ADC

Operation: ACCX ⇐ (ACCX) + (M) + (C)

Description: Adds the contents of the C bit to the sum of the contents of ACCX and M and places the result in ACCX. This instruction affects the H condition code bit so it is suitable for use in BCD arithmetic operations (see DAA instruction for additional information).

Condition Codes and Boolean Formulae:

S	X	H	I	N	Z	V	C
—	—	Δ	—	Δ	Δ	Δ	Δ

H $X3 \cdot M3 + M3 \cdot R3 + R3 \cdot X3$
Set if there was a carry from bit 3; cleared otherwise.

N R7
Set if MSB of result is set; cleared otherwise.

Z $\overline{R7} \cdot \overline{R6} \cdot \overline{R5} \cdot \overline{R4} \cdot \overline{R3} \cdot \overline{R2} \cdot \overline{R1} \cdot \overline{R0}$
Set if result is $00; cleared otherwise.

V $X7 \cdot M7 \cdot \overline{R7} + \overline{X7} \cdot \overline{M7} \cdot R7$
Set if a two's complement overflow resulted from the operation; cleared otherwise.

C $X7 \cdot M7 + M7 \cdot \overline{R7} + \overline{R7} \cdot X7$
Set if there was a carry from the MSB of the result; cleared otherwise.

Source Forms: ADCA (opr); ADCB (opr)

Addressing Modes, Machine Code, and Cycle-by-Cycle Execution:

Cycle	ADCA(IMM)			ADCA (DIR)			ADCA (EXT)			ADCA (IND,X)			ADCA (IND,Y)		
	Addr	Data	R/W	Addr	Data	R/W	Addr	Data	R/W	Addr	Data	R/W	Addr	Data	R/W
1	OP	89	1	OP	99	1	OP	B9	1	OP	A9	1	OP	18	1
2	OP+1	ii	1	OP+1	dd	1	OP+1	hh	1	OP+1	ff	1	OP+1	A9	1
3				00dd	(00dd)	1	OP+2	ll	1	FFFF	—	1	OP+2	ff	1
4							hhll	(hhll)	1	X+ff	(X+ff)	1	FFFF	—	1
5													Y+ff	(Y+ff)	1

Cycle	ADCB (IMM)			ADCB (DIR)			ADCB (EXT)			ADCB (IND,X)			ADCB (IND,Y)		
	Addr	Data	R/W	Addr	Data	R/W	Addr	Data	R/W	Addr	Data	R/W	Addr	Data	R/W
1	OP	C9	1	OP	D9	1	OP	F9	1	OP	E9	1	OP	18	1
2	OP+1	ii	1	OP+1	dd	1	OP+1	hh	1	OP+1	ff	1	OP+1	E9	1
3				00dd	(00dd)	1	OP+2	ll	1	FFFF	—	1	OP+2	ff	1
4							hhll	(hhll)	1	X+ff	(X+ff)	1	FFFF	—	1
5													Y+ff	(Y+ff)	1

Exemple du jeu d'instructions du PIC (Microchip)

- addlw ajoute une constante à W. Résultat dans W
- addwf ajoute W à F. Résultat dans W (si d=0) ou dans F
- (si d=1)
- andlw effectue un ET entre une constante et W. Résultat dans W
- andwf effectue un ET entre W et F. Résultat dans W (si d=0) ou dans F (si d=1)
- bcf fait passer un bit de F à 0
- bsf fait passer un bit de F à 1
- btfsc teste certains bits de F. Incrémente PC s'il sont à 0
- btfss teste certains bits de F. Incrémente PC s'il sont à 1
- call place dans PC l'adresse d'un sous programme
- clrf place zéro dans F
- clrw place zéro dans W et fait passer Z à 1
- clrwdt initialise le timer du chien de garde
- comf complémente F à 1

Exemple du jeu d'instructions du PIC (Microchip)

- decf décrémente F
- decfsz décrémente F. Si 0, Incrémente PC
- goto charge une adresse dans PC
- incf Incrémente F
- incfsz Incrémente F. Si 0, Incrémente PC
- iorlw effectue un OU inclusif entre une constante et F
- iorwf effectue un OU inclusif entre W et F
- movf si d=0, charge F dans W , sinon réécrit F dans F
- movlw charge une cste 8 bit dans W
- movwf stocke W vers F
- nop ne fait rien en gaspillant une période d'horloge
- retfie retourne d'une interruption
- retlw retourne d'un sous programme et place une constante dans W
- return retourne d'un sous programme

Exemple du jeu d'instructions du PIC (Microchip)

- rlf effectue une rotation de bits à gauche à travers Carry
- rrf effectue une rotation de bits à droite à travers Carry
- sleep fait passer le PIC en mode veille
- sublw soustrait W d'une constante
- subwf soustrait W de F
- swapf permute les deux quartets de F
- xorlw effectue un OU exclusif entre une constante et W.
- Résultat dans W
- xorwf effectue un OU exclusif entre W et F.

  Résultat dans W (si d=0) ou dans F (si d=1)

Les langages évolués pour microP et microC
C : adapté aux microcontrôleurs, C++

- Langage C :
  - Inventé par Dennis Ritchie sur système UNIX (écrit en C), en 1972
  - Normalisé en 1983 (le « C ANSI ») par l'American National Standards Institute
  - Très concis
  - Conçu pour être efficace (code généré très court) et portable facilement sur différentes machines, à condition de respecter les règles ANSI
  - Le langage de référence pour tous les petits ou moyens processeurs

- C++ :
  - Extension du C
  - Classes, objets, constructeur/destructeur, allocateur/ desallocateur, …
  - Moyens ou gros microcontrôleurs

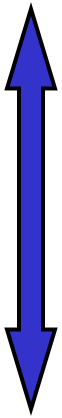
- JAVA…

Le langage C : adapté aux microcontrôleurs

#include <sdtio.h> // Directive de compilation indiquant d'inclure la bibliothèque E/S standard
#include <reg_uc.h> // Directive de compilation indiquant d'inclure la biblio. spécifique au µC
#define clear=0x00 // Directive de compilation indiquant des équivalences

```
char val1=0xA5;     // Déclaration d'une variable "caractère" avec valeur initiale
int val2;           // Déclaration d'une variable "nombre entier"

void tempo(char temps)
{
    corps du sous programme tempo
}
     void main(void) // Programme principal
     {
     DDRBA=0xFF // initialisation et configuration
     while (1) // Boucle principale
         {
         ...
         }
     }
void nmi(void)interrupt 0 // Sous programme d'interruption
{
}
```



Programme principal (main)

Le langage C : adapté aux microcontrôleurs (suite)

Déclarations de types du langage C : char, int, float

Type	Définition	Taille	Domaine non signé	Domaine signé
char	Variable de type caractère ascii mais utilisée pour les nombres entiers	8 bits	0 à 255	-128 à +127
int	Variable de type nombre entier	16 bits	0 à 65536	-32768 à 32767
float	Variable de type nombre réel	32 bits	$+/- 3,4.10^{-38}$ à $3,4.10^{38}$	X

Déclarations de types spécifiques aux microcontrôleurs :

Mot clé	Signification
at	Permet de définir l'adresse de la variable
code	Permet de réservé un bloc dans la mémoire programme
data	Permet de réservé un bloc dans la mémoire de données
register	Permet de réservé un bloc dans les registres de la CPU si cela est possible

Types structurés à maîtriser : chaînes de caractères, tableaux et pointeurs

Le langage C : adapté aux microcontrôleurs (suite)

Il existe dans tous les compilateurs “C” des bibliothèques de fonctions prédéfinies. La plus utilisée est **<stdio.h>** qui est propre au organes d’entrées / sorties standards. Dans le cas des ordinateurs ces organes sont le clavier et l’écran et ces bibliothèques sont fournies. Dans le cas d’un microcontrôleur ces organes sont généralement les interfaces séries ou parallèles du composant.

Dans cette bibliothèque on trouve les fonctions suivantes :

Printf() : écriture formatée de données.

scanf() : lecture formatée de données.

Putchar() : écriture d'un caractère.

getchar() : lecture d'un caractère.

Ces fonctions doivent être écrites spécifiquement pour la cible matérielle (dépend du mapping de la carte, du décodage, des périphériques, …)

NOTIONS d'OS9 (MICROWARE)

- OS9 = système d'exploitation temps réel
- OS9 a été développé sur 68000
- orienté contrôle de processus, automatismes, supervision, gestion d'E/S (Informatique industrielle)
- OS9 est MULTI-TÂCHES
  MULTI-UTILISATEUR
- OS9 est ROMABLE
- comprend un superviseur de ressources
  - mémoire
  - temps CPU
  - entrées / sorties
- comprend des fonctions système
  - interface utilisateur
  - E/S
  - partage des tâches
  - exécution de programme
  - ....
- cartes au format et bus GESPAC
- Programmation des cartes OS9

Notion de système temps réel

Exemple d'0S9 (cartes Gespac 68000)

- Notions de multitâches, de processus, de séquenceur

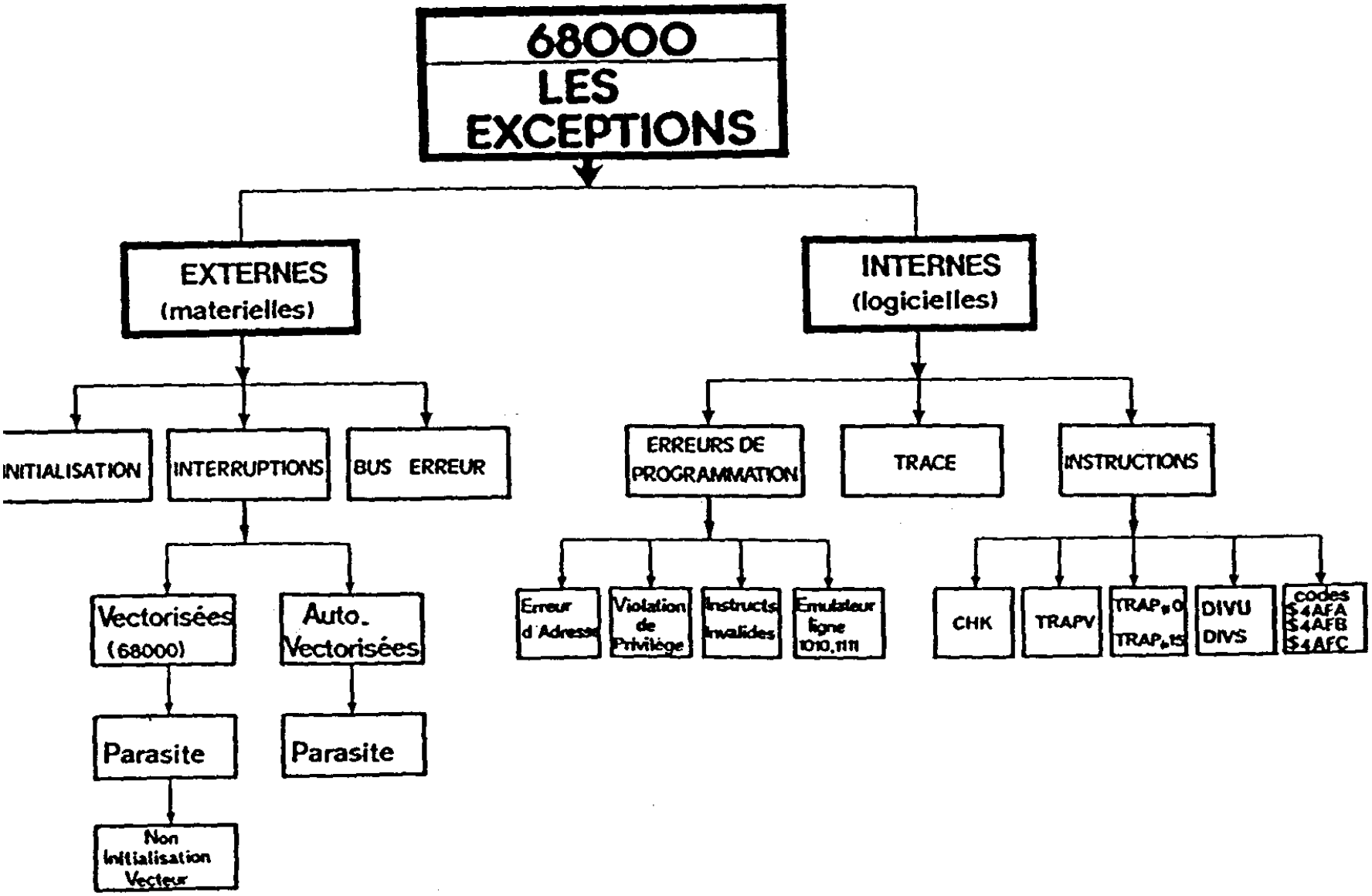
Interruptions

- **Action/événement sur un périphérique (exemple : appui sur une touche clavier ou caractère reçu par l'UART)**
  - Soit le microP scrute (lit en boucle) en permanence les périphériques (lent) : **«pooling »**
  - Soit accès direct du périphérique à la mémoire ou aux ressources du système (par ex. le caractère série reçu par l'UART est placé en mémoire) sans intervention du microP : **« DMA »** (accès direct mémoire)
  - Soit le périphérique alerte le microP (**interruption**), celui-ci pouvant décider de prendre en charge la demande immédiatement (inerrupt. prioritaire), de la reporter (interrup. non prioritaire) ou de la rejetter (interrupt. non validée)
    - Pour autoriser ou interdire les interrup.(sauf NMI), fixer les priorité : gestion du **masque d'interruption** (en général positionnement de bits dans le registre d'état)
    - NMI (ou Reset) = interruptions non masquables

Interruptions (suite)

- Le périphérique alerte le microP (**interruption**), celui-ci pouvant décider de prendre en charge la demande immédiatement (inerruption prioritaire), de la reporter (interruption non prioritaire) ou de la rejetter (interrupt. non validée)
  - Si l'interruption est traitée, le microP :
    - termine l'instruction en cours,
    - sauvegarde le contenu du PC dans la pile (adresse de retour) et éventuellement les registres en cours d'utilisation (sauvegarde du contexte)
    - selon la ou les lignes d'interruption activées, le microP identifie le périphérique appelant (clavier ? UART ? NMI ?) et saute à une adresse spécifique (interruption vectorisée)
    - le microP exécute le sous-programme d'interruption spécifique à l'interruption activée (par ex. lire le contenu de l'UART et le placer dans la mémoire RAM)
    - le microP « dépile » (contenu de la pile dans le PC = adresse de retour au programme principal) et restaure éventuellement les registres (contexte) et continue donc son programme initial

Interruptions internes (traps) et externes

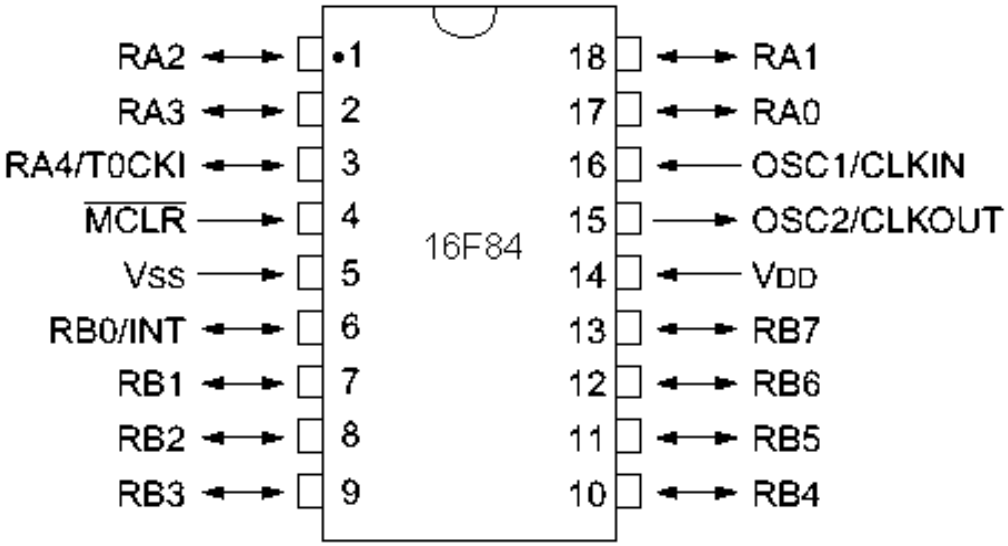


Interruptions (Intel 8051)

INTERRUPT SOURCES AND PRIORITIES Table 9

NAME	DESCRIPTION	VECTOR	NATURAL PRIORITY	8051/DALLAS
PFI	Power Fail Interrupt	33h	1	DALLAS
$\overline{\text{INT0}}$	External Interrupt 0	03h	2	8051
TF0	Timer 0	0Bh	3	8051
$\overline{\text{INT1}}$	External Interrupt 1	13h	4	8051
TF1	Timer 1	1Bh	5	8051
SCON0	TI0 or RI0 from serial port 0	23h	6	8051
TF2	Timer 2	2Bh	7	8051
SCON1	TI1 or RI1 from serial port 1	3Bh	8	DALLAS
INT2	External Interrupt 2	43h	9	DALLAS
$\overline{\text{INT3}}$	External Interrupt 3	4Bh	10	DALLAS
INT4	External Interrupt 4	53h	11	DALLAS
$\overline{\text{INT5}}$	External Interrupt 5	5Bh	12	DALLAS
WDTI	Watchdog Time–Out Interrupt	63h	13	DALLAS

Le microcontrôleur PIC de Microchip



PIC12CXXX/PIC12FXXX Family: 8-pin 12-bit/14-bit program word
PIC16C5X Family: 12-bit program word
PIC16CXXX/PIC16FXXX Family: 14-bit program word
PIC17CXXX Family: 16-bit program word
PIC18CXXX/PIC18FXXX Family: enhanced 16-bit program word

- **Vss** et **Vdd** : broches d'alimentation (3 à 5,5 V)
- **OSC1**, **OSC2** : signaux d'horloge. Peuvent recevoir un circuit (R,C) ou un résonateur. **CLKIN** peut être connectée à une horloge externe. (0 à 4 , 10 ou 20 MHz)
- **CLR** : Reset (Master Clear).
- **RA0..RA4** : 5 E/S du port A
- **RB0..RB7** : 8 E/S du port B
- **T0CKI** : Entrée d'horloge externe du timer TMR0.
- **INT** : Entrée d'interruption externe

Microcontrôleurs Motorola

MOTOROLA (6805 et) 68HC05 (8bits)

→ Technologie 6805 : HMOS ↗ dépassé (consommation)
→ 68HC05 : HCMOS (alim° 3 à 6V-)

→ Versions UVPROM, OTP PROM

→ 68HC05 : Timer 16bits très puissant

→ de très nombreuses versions (comme pour le PIC)

MOTOROLA 68HC11

→ Registres internes sur 16bits } "faux" 16bits
→ Bus données sur 8bits }

→ les Ports B et C peuvent être utilisés comme ports E/S ou Adresse / Données

Microcontrôleurs Intel

* MICROCONTRÔLEURS INTEL 8031 / 8051 / 80C51 (HCMOS).

→ 4 ports d'entrée-sortie de 8 broches

→ 2 compteurs-temporisateurs ("Timers") 8 bits

→ RAM interne + EPROM ou A/D selon modèles

→ interface série asynchrone full duplex, synchrone, ...

→ ORIGINALITÉ : 2 espaces mémoire différents pour le programme (ROM) et les données (RAM)

→ ROM Programme accès avec $\overline{PSEN}$

→ RAM accès avec $\overline{RD}$, $\overline{WR}$

→ Interruptions : broches INT0, INT1 + 3 ~~broches~~ 5 (2 timers, 1 série)] 5

→ certains 8051 (80C51, C52, ..) ont 2 modes de fonctionnement à puissance réduite (idle, power down)

→ cf 8052 BASIC (interpréteur en ROM interne)

Conclusion : Microcontrôleurs

OFFRE TRES VASTE :

8 bits :

- Architectures « anciennes » enrichies
- MOTOROLA (HC05 : 180 décl. – HC11 : 60 décl.)
- Reprises INTEL (8051 : 100 décl. chez Philips)
- Architectures modernes RISC (MicroCHIP – ATMEL, …)

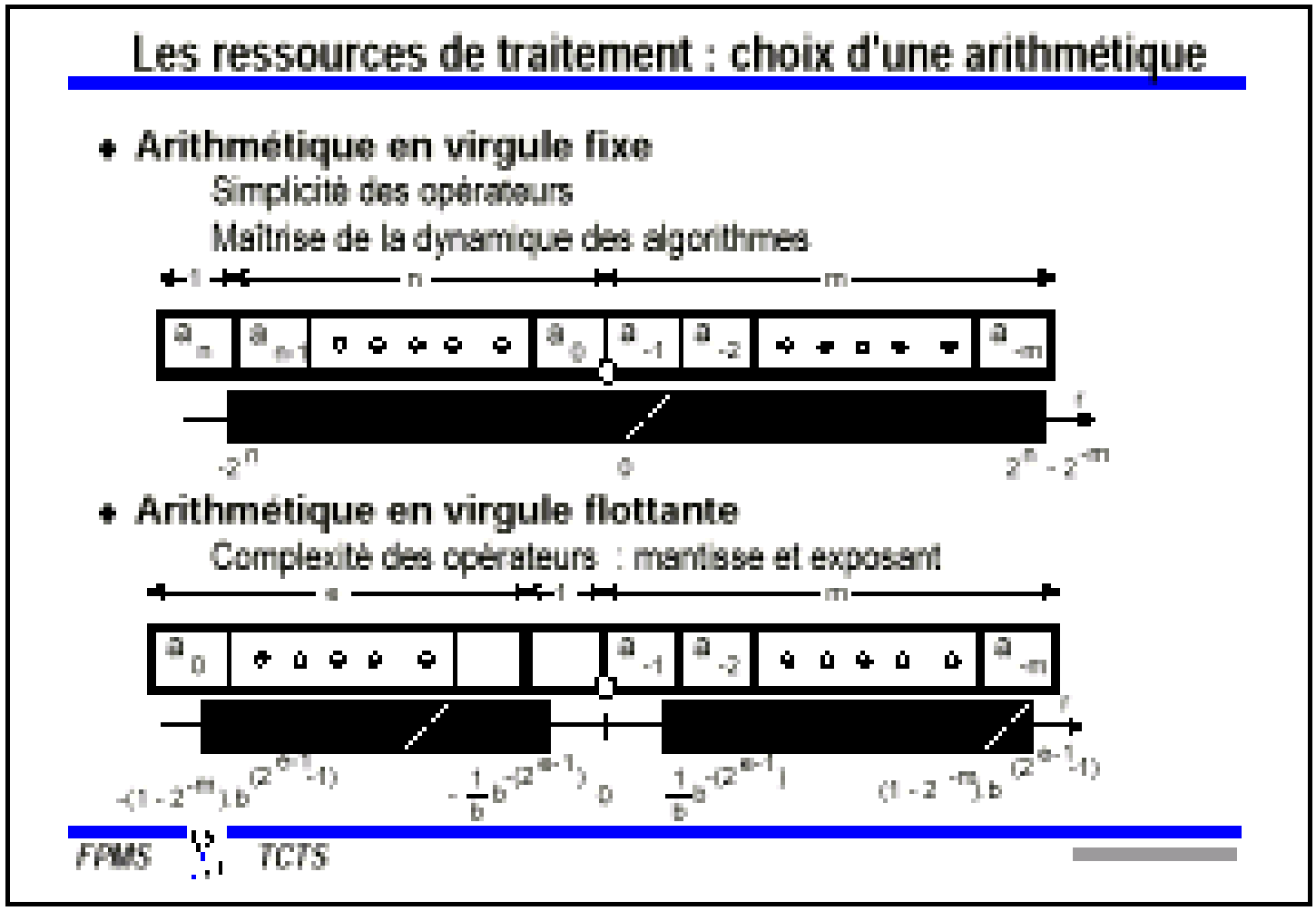
16 bits / 32 bits :

- Déclinaisons sur base de processeurs génériques
- PowerPC – MIPS – SPARC – X86 – ARM
- Architectures spécifiques évolutives : SH…, ColdFire

Microcontrôleurs : critères de choix

- Puissance de calcul (vitesse d'horloge maxi / mémoire adressable, mémoire interne, arithmétique interne)
- Consommation
- Technologie
- Gamme de température
- Instructions et modes d 'adressage
- Prix et disponibilité du microprocesseur
- Périphériques compatibles disponibles
- Prix du système de développement
- Logiciels et compilateurs disponibles
- Périphériques disponibles
- Support du fabricant ou distributeur (formation, *hot line,* ingénieurs d'applications)

Critères de choix des microprocesseurs : arithmétique virgule fixe ou virgule flottante



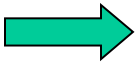
Conception d ’une carte à microprocesseurs

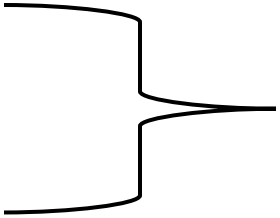
- **SOLUTIONS ACTUELLES :**
  - LES PROCESSEURS (microprocesseur et microcontrôleur)
  - LES COMPOSANTS PROGRAMMABLES (FPGA)
  - LES COMPOSANTS GENERIQUES DEDICACES
  - le mixage Processeur (calcul) + composants programmables (décodage / interface)
  - Les ASICs

- **Les microprocesseurs :**
  - Les microprocesseurs traditionnels
  - Les microcontrôleurs
  - Les microprocesseurs spécialisés : DSP

- **Vers des solutions mixtes :**
  - Les composants programmables (avec cœur de microP ou microC)
  - Les composants génériques dédicacés
  - Vers l'intégration spécifique

Microprocesseurs - Tendances

- Architectures de plus en plus sophistiquées ou mixtes
- Une application  un type de microcontrôleur ?
- Outils de développement et langages de haut niveau (l'assembleur est réservé au pilotage d'interface rapide, échantillonnage)
- Moins de différences entre microprocesseur, microcontrôleur et DSP
- Intégration des périphériques dans la puce microprocesseur
- Faible consommation
- Faible tension d'alimentation
- Haute fréquence d'oscillation



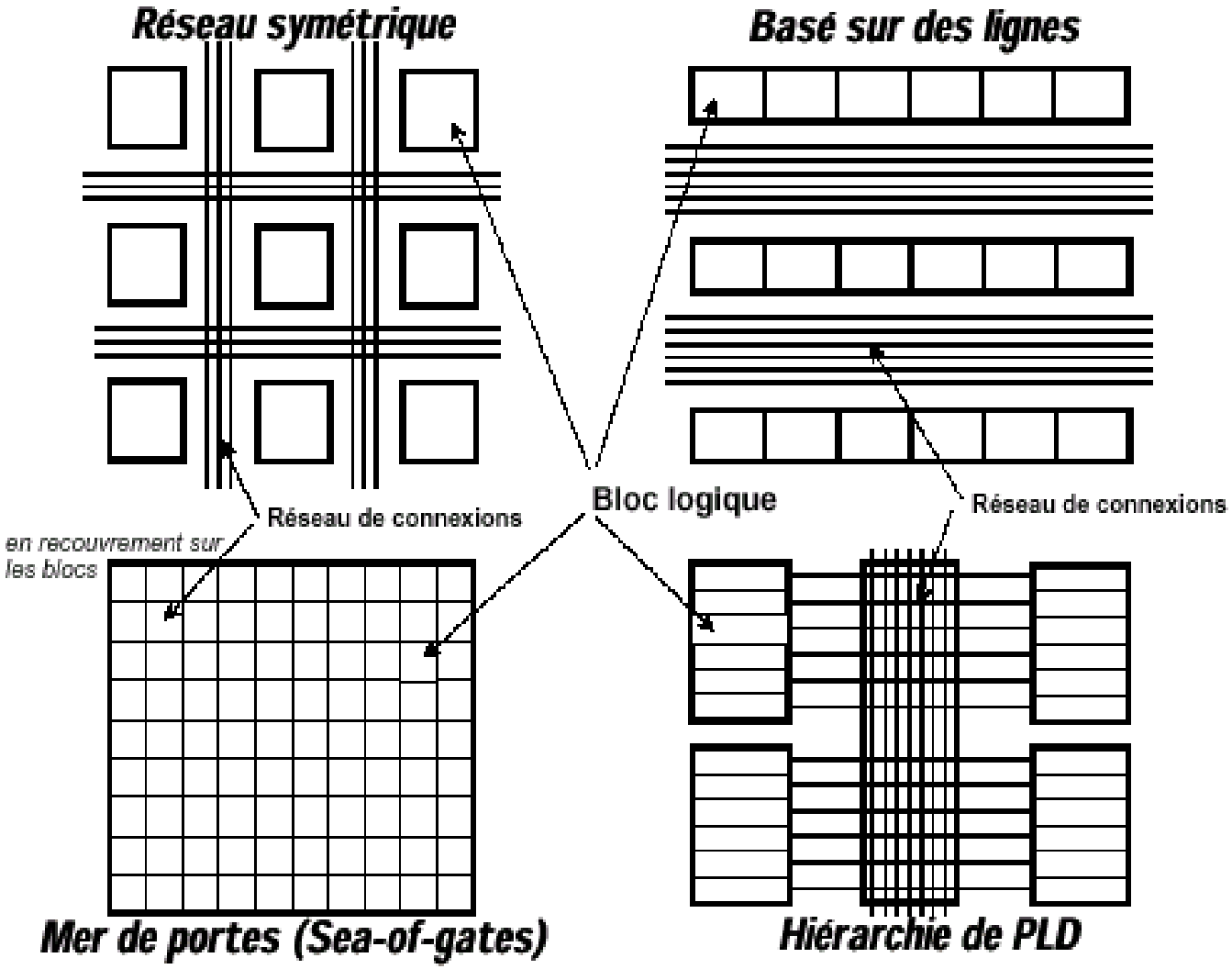
Le routage et la CAO (attention à la CEM) sont critiques

- « Concurrence » avec les circuits logiques programmables (FPGA) pour certaines applications (beaucoup d'E/S, vitesse de traitement élevée)

Microprocesseurs - Tendances

- **Intégration de gestionnaires de bus et réseaux spécifiques :**
  - **bus CAN (automobile)**
  - **Ethernet / Internet (pile TCP-IP interne)**
- **microprocesseurs RISC**
- **parallélisme**
- **cœurs de microprocesseurs ou microcontrôleurs « intégrables » dans des composants programmables ou ASICs (cf cœurs de DSP DSP_Group : Oak)**
- **« Système sur composant » System on chip pSOC (Cypress par exemple)**
  - **associations mixtes microcontrôleur-DSp, numérique-analogique, …**
  - **PSoc de Cypress, Anadigm, …**
- **émulateurs « in circuit » (chaîne JTAG par exemple)**
- **tendance vers l'unification de la programmation micro-FPGA : langage C ?**

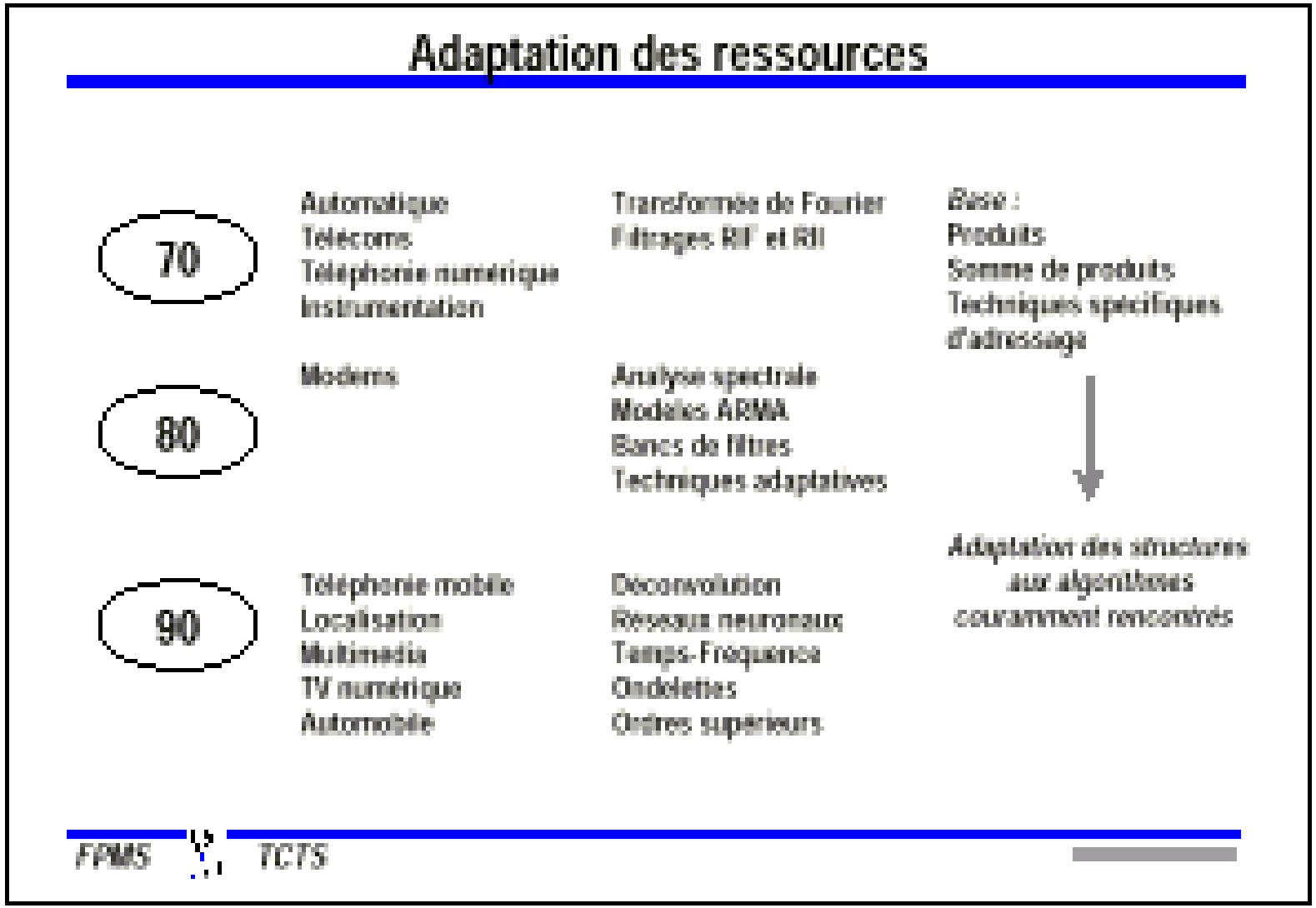
Les composants programmables



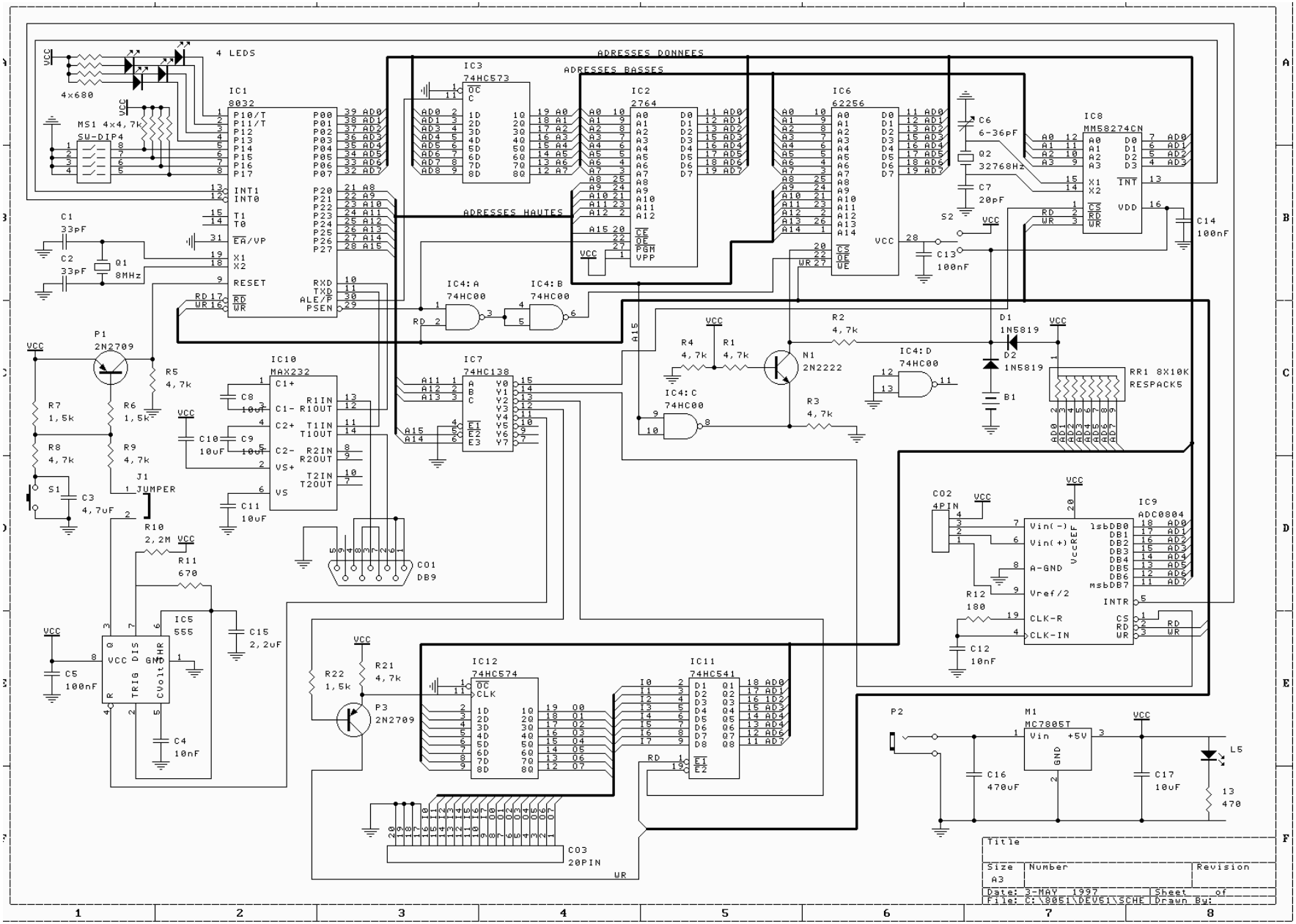
Leur futur

- Toutes les technologies : augmentation des vitesses
  RISC vers 2000 : 1 GHz
- Pour les composants programmables : augmentation des capacités
  projets d'un demi-million de portes
- Mixité des systèmes :
  - Systèmes mixtes : microcontrôleurs RISC + DSP
  - Microcontrôleurs RISC améliorés pour le traitement du signal
  - DSP modifié pour favoriser les fonctionnalités "microcontrôleur"
    exemple : TMS320C27x
- Noyaux intégrables : adjonction de mémoires / périphériques / FPGAs
- Importance des outils logiciels :
  - Structures mieux adaptées à la programmation en langage de haut niveau , compilateurs efficients (C *pour processeurs* / VHDL *pour les composants programmables*)
  - Outils de développement haut niveau : développement graphique de type "*flux de données*".

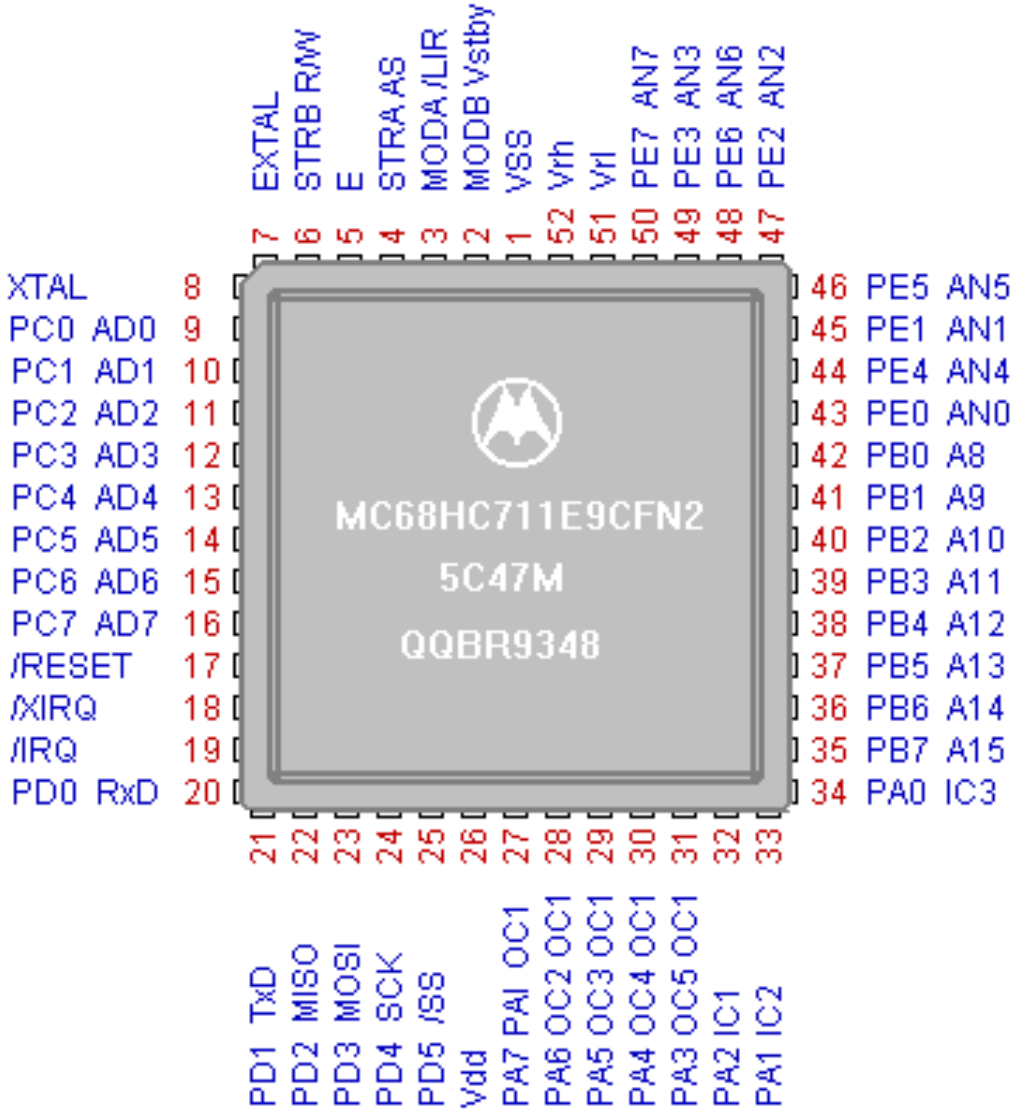
Applications des microprocesseurs



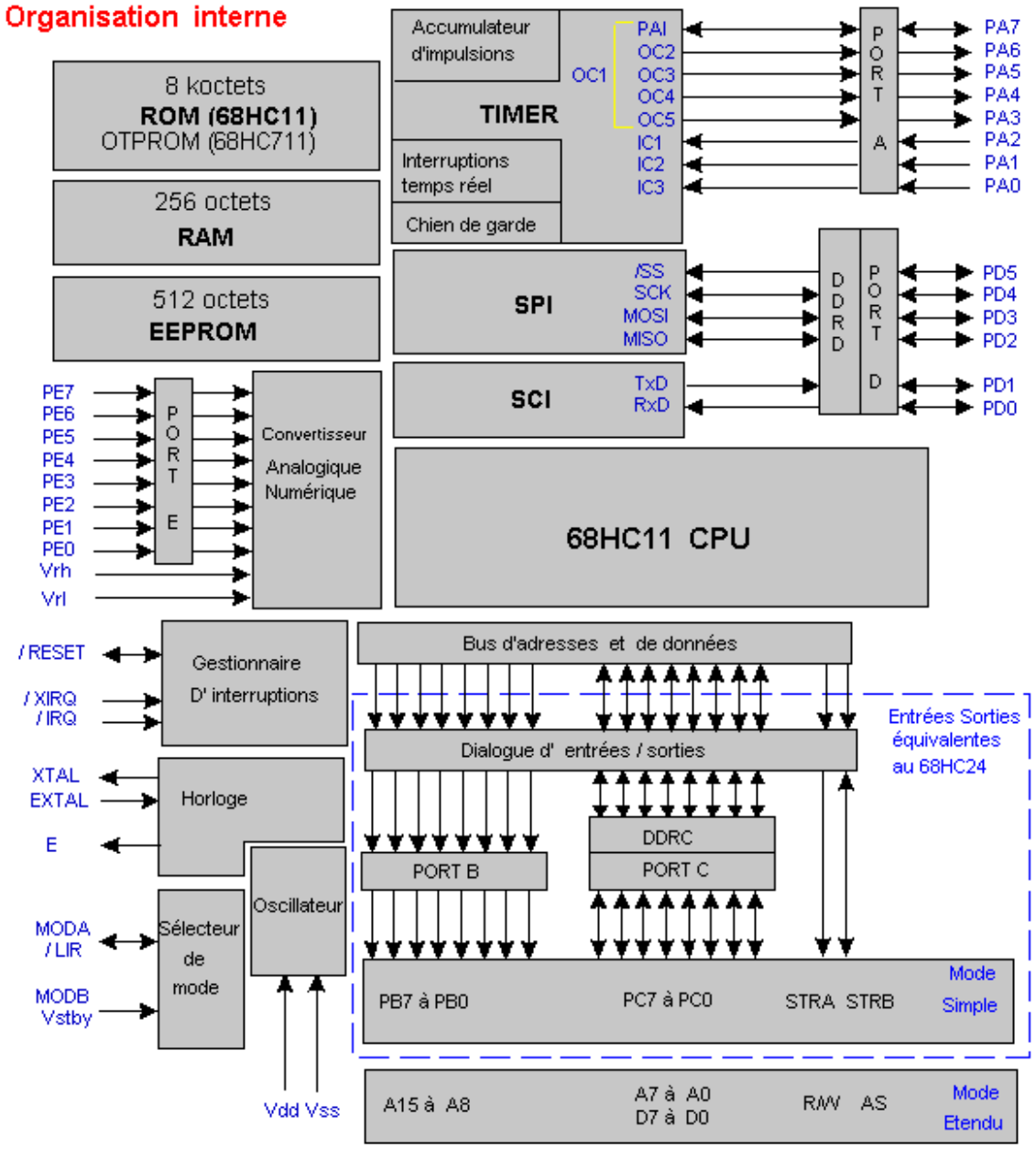
TD microcontrôleur 80C32



Le microcontrôleur 68HC11de Motorola



Le microcontrôleur 68HC11de Motorola



68HC11

Table 1-1 M68HC11 Family Members

Part Number	EPROM	ROM	EEPROM	RAM	CONFIG[2]	Comments
MC68HC11A8	—	—	512	256	$0F	Family Built Around This Device
MC68HC11A1	—	—	512	256	$0D	'A8 with ROM Disabled
MC68HC11A0	—	—	—	256	$0C	'A8 with ROM and EEPROM Disabled
MC68HC811A8	—	—	8K + 512	256	$0F	EEPROM Emulator for 'A8
MC68HC11E9	—	12K	512	512	$0F	Four Input Capture/Bigger RAM 12K ROM
MC68HC11E1	—	—	512	512	$0D	'E9 with ROM Disabled
MC68HC11E0	—	—	—	512	$0C	'E9 with ROM and EEPROM Disabled
MC68HC811E2	—	—	2K[1]	256	$FF[3]	No ROM Part for Expanded Systems
MC68HC711E9	12K	—	512	512	$0F	One-Time Programmable Version of 'E9
MC68HC11D3	—	4K	—	192	N/A	Low-Cost 40-Pin Version
MC68HC711D9	4K	—	—	192	N/A	One-Time Programmable Version of 'D3
MC68HC11F1	—	—	512[1]	1K	$FF[3]	High-Performance Non-Multiplexed 6B-Pin
MC68HC11K4	—	24K	640	768	$FF	> 1 Mbyte memory space, PWM, C_S, 84-Pin
MC68HC711K4	24K	—	640	768	$FF	One-Time Programmable Version of 'K4
MC68HC11L6	—	16K	512	512	$0F	Like 'E9 with more ROM and more I/O, 64/68
MC68HC711L6	16K	—	512	512	$0F	One-Time Programmable Version of 'L4

68HC11

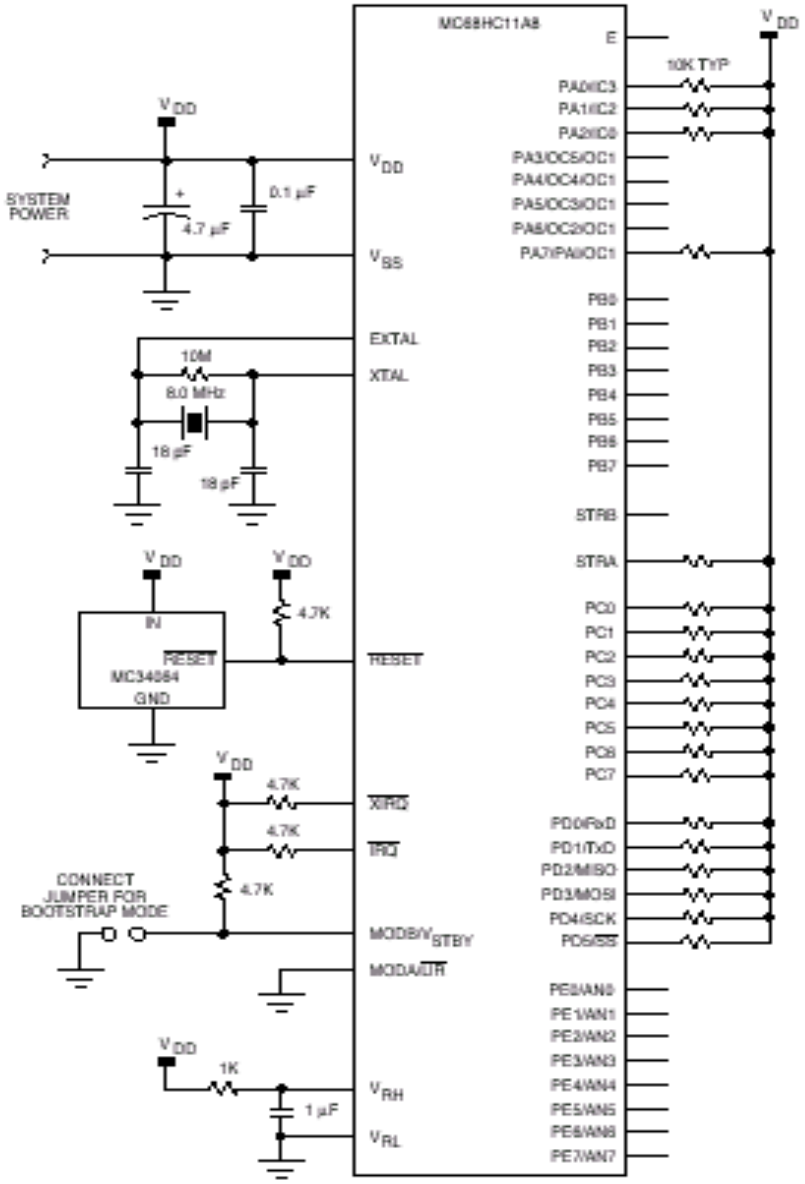
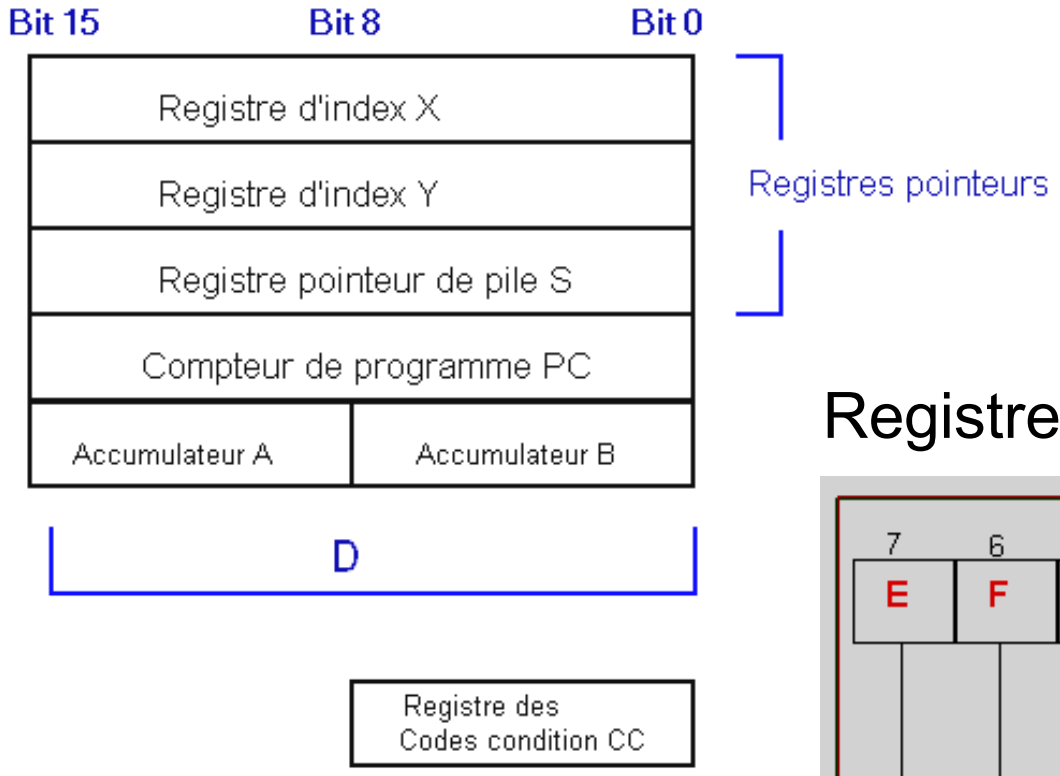


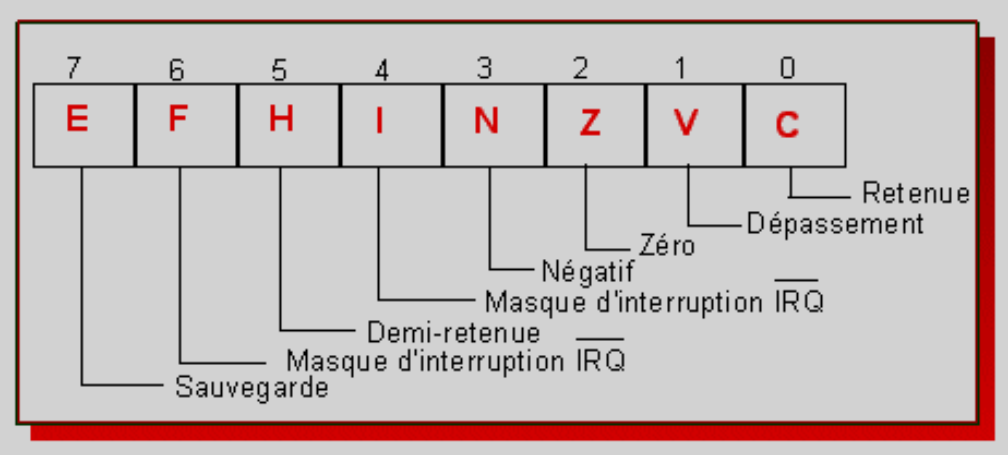
Figure 2-22 Basic Single-Chip-Mode Connections

Le microcontrôleur 68HC11de Motorola

Registres internes



Registre d'état du microcontrôleur



Modes d ’adressage du 68HC11

- Modes d ’adressage classiques :
  - **adressage direct**
    - l ’adresse réelle est indiquée dans l’instruction : 2 octets (16 bits) permettent d ’accéder directement à 64 Koctets de mémoire (2^{16} = 65536)
    - direct relatif : un seul octet ajouté au PC permet d ’accéder à une page de la mémoire (2^{8} = 256 octets). Code plus court, instruction plus rapide
  - **adressage indirect** : fonctionnement type « pointeur »
  - **adressage indexé** : la valeur du registre d ’index est ajoutée à l ’adresse calculée (déplacement)

Modes d ’adressage du 68HC11 (extraits)

Function	Mnemonic	IMM	DIR	EXT	INDX	INDY	INH
Clear Memory Byte	CLR			X	X	X	
Clear Accumulator A	CLRA						X
Clear Accumulator B	CLRB						X
Load Accumulator A	LDAA	X	X	X	X	X	
Load Accumulator B	LDAB	X	X	X	X	X	
Load Double Accumulator D	LDD	X	X	X	X	X	
Pull A from Stack	PULA						X
Pull B from Stack	PULB						X
Push A onto Stack	PSHA						X
Push B onto Stack	PSHB						X
Store Accumulator A	STAA	X	X	X	X	X	
Store Accumulator B	STAB	X	X	X	X	X	
Store Double Accumulator D	STD	X	X	X	X	X	
Transfer A to B	TAB						X
Transfer A to CCR	TAP						X
Transfer B to A	TBA						X
Transfer CCR to A	TPA						X
Exchange D with X	XGDX						X
Exchange D with Y	EGDY						X

ADC Add with Carry ADC

Operation: ACCX ⇐ (ACCX) + (M) + (C)

Description: Adds the contents of the C bit to the sum of the contents of ACCX and M and places the result in ACCX. This instruction affects the H condition code bit so it is suitable for use in BCD arithmetic operations (see DAA instruction for additional information).

Condition Codes and Boolean Formulae:

S	X	H	I	N	Z	V	C
—	—	Δ	—	Δ	Δ	Δ	Δ

H $X3 \cdot M3 + M3 \cdot R3 + R3 \cdot X3$
Set if there was a carry from bit 3; cleared otherwise.

N R7
Set if MSB of result is set; cleared otherwise.

Z $\overline{R7} \cdot \overline{R6} \cdot \overline{R5} \cdot \overline{R4} \cdot \overline{R3} \cdot \overline{R2} \cdot \overline{R1} \cdot \overline{R0}$
Set if result is $00; cleared otherwise.

V $X7 \cdot M7 \cdot \overline{R7} + \overline{X7} \cdot \overline{M7} \cdot R7$
Set if a two's complement overflow resulted from the operation; cleared otherwise.

C $X7 \cdot M7 + M7 \cdot \overline{R7} + \overline{R7} \cdot X7$
Set if there was a carry from the MSB of the result; cleared otherwise.

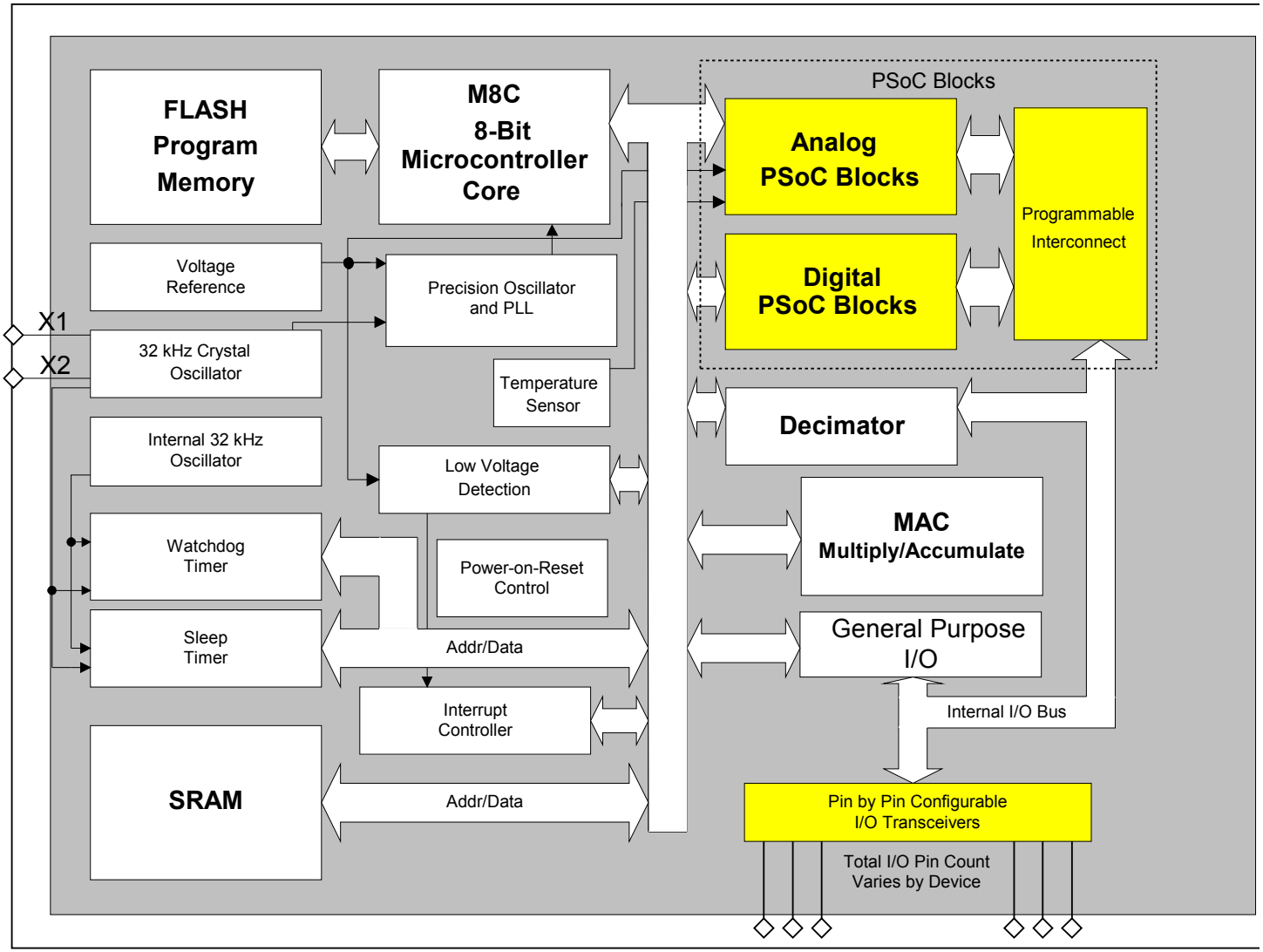
Source Forms: ADCA (opr); ADCB (opr)

Addressing Modes, Machine Code, and Cycle-by-Cycle Execution:

Cycle	ADCA(IMM)			ADCA (DIR)			ADCA (EXT)			ADCA (IND,X)			ADCA (IND,Y)		
	Addr	Data	R/W	Addr	Data	R/W	Addr	Data	R/W	Addr	Data	R/W	Addr	Data	R/W
1	OP	89	1	OP	99	1	OP	B9	1	OP	A9	1	OP	18	1
2	OP+1	ii	1	OP+1	dd	1	OP+1	hh	1	OP+1	ff	1	OP+1	A9	1
3				00dd	(00dd)	1	OP+2	ll	1	FFFF	—	1	OP+2	ff	1
4							hhll	(hhll)	1	X+ff	(X+ff)	1	FFFF	—	1
5													Y+ff	(Y+ff)	1

Cycle	ADCB (IMM)			ADCB (DIR)			ADCB (EXT)			ADCB (IND,X)			ADCB (IND,Y)		
	Addr	Data	R/W	Addr	Data	R/W	Addr	Data	R/W	Addr	Data	R/W	Addr	Data	R/W
1	OP	C9	1	OP	D9	1	OP	F9	1	OP	E9	1	OP	18	1
2	OP+1	ii	1	OP+1	dd	1	OP+1	hh	1	OP+1	ff	1	OP+1	E9	1
3				00dd	(00dd)	1	OP+2	ll	1	FFFF	—	1	OP+2	ff	1
4							hhll	(hhll)	1	X+ff	(X+ff)	1	FFFF	—	1
5													Y+ff	(Y+ff)	1

PSoc de Cypress



PSoc Cypress

Composant mixte avec Microcontrôleur 8 bit + mémoire + composants analogiques + composants à capacités commutées …

- One 8-Bit Counter
- One 16-Bit Timer
- One Full-Duplex UART w/ Baud Rate Generator
- One SPI Slave (Full Duplex)
- One 4-Input 8-Bit Delta-Sigma A/D
- One 6-Bit D/A
- One 8-Bit D/A
- Two Low-Pass Filters

- One 16-Bit Counter
- One 8-Bit PWM
- One Half-Duplex UART
- One SPI Master
- One 12-Bit Incremental A/D
- One Low-Pass Filter
- One 8-Bit D/A
- Two Instrumentation Amplifiers

Composant « virtuel » microcontrôleur

Figure 1 is a block diagram of a PicoBlaze module. The Spartan PicoBlaze modules require no external support and provide a flexible environment for other logic connections into the PicoBlaze module.

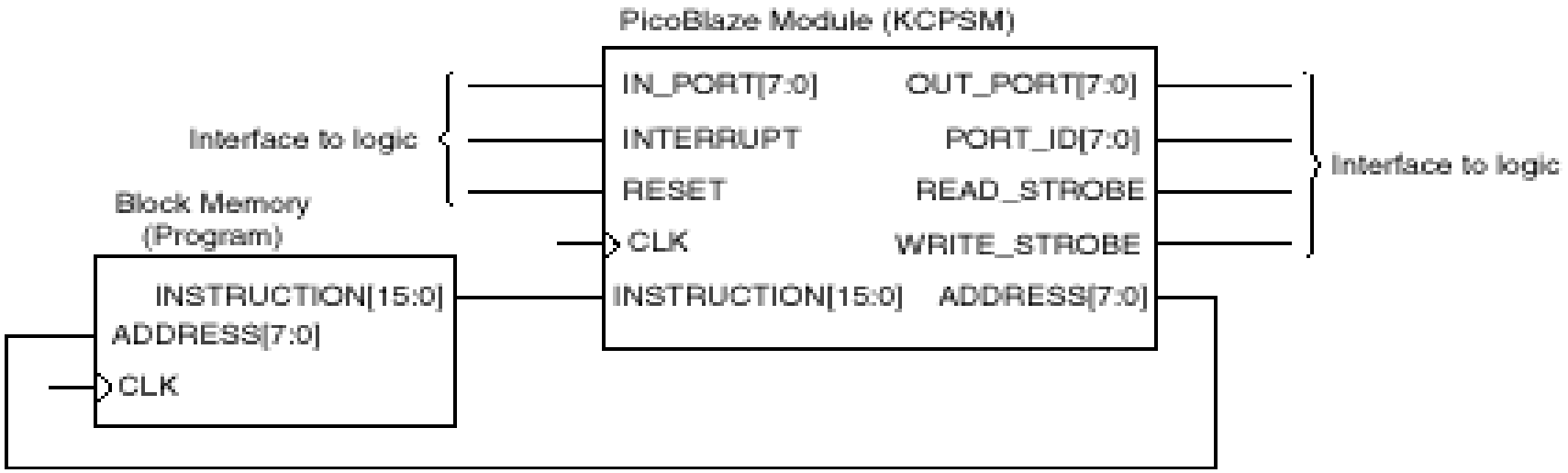


Figure 1: **PicoBlaze Module Block Diagram**

The PicoBlaze module is supplied as VHDL and as a precompiled soft macro that is handled by the place and route tools to merge with the logic of a design. This plot (Figure 2) from the FPGA Editor viewer shows the macro in isolation within the smallest Spartan-IIE device.

PicoBlaze Architecture

Figure 5 shows the PicoBlaze architecture.