ARCHI – Architecture des ordinateurs

Sylvain Brandel
2025 – 2026
sylvain.brandel@univ-lyon1.fr

Partie 1

INTRODUCTION
VUE D'ENSEMBLE DE L'ORDINATEUR

Fonctionnement

- CM : 18h – 12 x 1h30
  - Jeudi après-midi
  - Cela parait évident mais … Silence dans l'amphi. S'il vous plait.
- TD : 18h – 4 x 3h et 4 x 1h30
  - Mardi 8h – 9h30 ou 8h – 11h15
  - Début des TD mardi 9 septembre 2025
- TP : 24h – 8 x 3h
  - Mardi 8h – 11h15 ou 9h45 – 13h
  - Début des TP mardi 23 septembre 2025

- Fin des enseignements mardi 9 décembre 2025 (hors seconde chance)

Planning global

	MARDI			JEUDI		
Date	8:00-09:30	9:45-11:15	11:30-13:00	14:00-15:30	15:45-17:15	17:30-19:00
01/09/2025				CM1 vue ens	CM2 logique	
08/09/2025	TD vue ens / logique			CM3 codage 1	CM4 codage 2	
15/09/2025	TD codage			CM5 combi 1		
22/09/2025	TD combi	TP combi 1		CM6 combi 2		
29/09/2025	TD combi	Interro 1 (rés)		CM7 séq 1		
06/10/2025	TP combi 2			CM8 séq 2		
13/10/2025	TD séq			CM9 ass 1		
20/10/2025	TD ass	TP séq		CM10 ass 2		
27/10/2025	Pas d'enseignements en licence					
03/11/2025	Interro 2 (rés)	TP ass 1				
10/11/2025	Férié			CM11 ébauche		
17/11/2025	TD ass	TP ass 2		CM12 cache		
24/11/2025	TP LC3 1					
01/12/2025	TP LC3 2					
08/12/2025	Interro finale	TP noté				
15/12/2025						
22/12/2025	Vacances					
29/12/2025						
05/01/2026						

Evaluation

- http://sylvain.brandel.pages.univ-lyon1.fr/archi/

- UE en CCI (Contrôle Continu Intégral)
  - 4 épreuves :
    - Une interrogation d'1h, en séance de TD mardi 30 septembre 25 (à confirmer) 25%
    - Une interrogation d'1h, en séance de TD mardi 4 novembre 25 (à confirmer) 25%
    - Un TP noté d'1h, mardi 9 décembre 2025 25%
    - Une interrogation finale d'1h, mardi 9 décembre 2025 également 25%
  - Une épreuve de seconde chance (E2C)
    - Ouvert à tous
    - La note remplacera la moins bonne des notes obtenues aux interrogations, si elle est supérieure (le TP noté ne peut pas être remplacé par l'E2C)

- ATTENTION : Depuis 2023 – absences aux contrôles
  - ~~ABJUS => note neutralisée ; ABINJ => zéro~~
  - Plus de distinction pédagogique entre ABJUS et ABINJ
    - Un absent n'a pas été évalué, l'absence justifiée ou non a le même impact sur la note
    - En cas d'absence, justifiée ou non : zéro à l'épreuve
    - La note d'E2C remplace une absence, justifiée ou non
    - En cas d'absences justifiées à 2 épreuves ou plus, *possibilité* d'épreuves de substitution

De votre côté

- Travail personnel conséquent
- Se préparer à l'avance
- Ne pas attendre que les réponses viennent toutes seules

- Lisez vos mails …

- Contactez-moi par mail … précisez ARCHI et allez droit au but. Svp.

Sources et références

- **Cours de X. Urbain** et cours de N. Louvet

- Hennessy-Patterson : *Computer Architecture: A Quantitative Approach*
- Nisan & Schocken : *The Elements of Computing Systems*
  - http://www.nand2tetris.org

ARCHI – Architecture des ordinateurs

Sylvain Brandel
2025 – 2026
sylvain.brandel@univ-lyon1.fr

CM 1

VUE D'ENSEMBLE DE L'ORDINATEUR

Ce cours

- Super-calculateurs
  - Actuellement : plusieurs millions de cœurs, plusieurs MW
  - Motivations ? Par exemple, programme *Simulation*

    http://www-lmj.cea.fr/fr/programme_simulation

- Physique
  - Actuellement : électricité

Ordinateurs

- Programme
  - Grosso modo expression finie, et si possible succincte, d'une façon d'obtenir un résultat effectivement

- Ordinateur : dispositif physique pour
  - Réaliser des calculs …
  - Sur des données …

  décrits par le programme

Ordinateurs
Omniprésence

Type	Prix	Exemples d'utilisation
Supercalculateurs	~ 100M€	Simulations physiques de grande ampleur, météo
Ordinateurs centraux (mainframe)	~ 1M€	Banques, blockchains
Grappes de calcul	~ 100k€	Simulations physiques, météo
Serveurs	~ 10k€	Serveurs réseau
Micro-ordinateurs	~ 1k€	Ordinateurs de bureau, de jeu, portables ~ 150 millions en 2000
Processeurs embarqués	~ 100€	Téléphones, véhicules ~ 6 milliards en 2000
Microcontrôleurs	~ 5€	Électroménager
Puces jetables	~ 1€	Cartes à puce, RFID

https://www.top500.org

Ordinateurs

Une vieille histoire

- Moyens de calcul purement manuels
  - Systèmes de numération
- Moyens de calcul mécaniques (XVIIe – XIXe)
  - Pascaline (1623-1662)
    - Additions et soustractions en décimal
  - Machine de Leibniz (1646-1716)
    - Multiplications et divisions en décimal
  - Machine analytique de Babbage (1792-1871)
    - Mémoire, instructions sur cartes perforées
  - Développements jusqu'au XXe siècle
- Machines électromécaniques (début XXe)
  - Relais électromécaniques
  - Machines construites par Konrad Zuse entre 1930 et 1944 (Allemagne)
    - Calcul binaire
  - Mark I, Harvard (Cambridge, MA) en 1944
    - Mémoire de 72 mots de 23 chiffres décimaux, 6s pour exécuter une instruction

Ordinateurs

Une vieille histoire

- Tubes à vide (1945 – 1955)
  - Ancêtre du transistor
  - COLOSSUS en 1943 (GB)
  - ENIAC en 1946 (Philadelphie, PA)
    - 18000 tubes à vide, 1500 relais, 30t, 72m^2, 140kW
    - 20 registres de 10 chiffres décimaux, 350 multiplications par seconde
- Transistors (1955 – 1965)
  - Inventé / découvert en 1948, laboratoire Bell (NJ)
  - Plus petit et plus fiable que le tube à vide
  - Fabriqué à partir de matériaux semi-conducteurs, généralement le silicium
  - IBM 7094 en 1964
    - 32536 mots de 36 chiffres binaires, programmé en COBOL et FORTRAN
  - CDC, Honeywell en 1960-69

Ordinateurs
Une vieille histoire

- Circuits intégrés (1965 – 1980)
  - Procédé découvert en 1958
  - Graver plusieurs transistor sur une même plaque de silicium
  - IBM 360 en 1970-77
    - Temps de cycle 250ns, mémoire ~ 500000 mots de 8 chiffres décimaux
  - DEC en 1969-77
  - Intel 4004 en 1971
    - Premier microprocesseur commercialisé
    - Processeur 4 chiffres binaires, cycle de 10.6μs, 2300 transistors sur 10mm2
- LSI / VLSI (1980 –)
  - Large / Very Large Scale Integration
  - CMOS (*Complementary Metal Oxyde Semiconductor*) en 1980
  - IBM PC, Apple II en 1978
- Parallélisme
  - Micro-ordinateurs actuels, plusieurs cœurs, ~ 100W, 64 chiffres, milliards d'op./s
  - Super-calculateurs

Ordinateurs

Machine de Von Neumann

- John Von Neumann (1903 – 1957)
- Ordinateur : calculs / données

- Machine de Von Neumann :
  - Centre pour calculs — Unité Centrale
  - Centre pour données et programme — Mémoire Centrale

  qui communiquent — Bus

- Programmes en mémoire, briques de base : instructions

- Pas d'évolution depuis 20 ans

- Autre modèle : architecture de Harvard

Ordinateurs
Mémoires

- Mémoires à accès direct RAM : temps d'accès identique partout

- SRAM : registres, caches rapide, chères
  - Peu
  - Nommées
  - Certains spécialisés : CP, RI, résultat de comparaisons …

- Hiérarchies de mémoires cache assez rapides assez chères

- DRAM : mémoire principale lentes, peu chères
  - Beaucoup
  - Par adresse

Ordinateurs
Représentation

- Niveau programmeur : très évolué, très compliqué
- Niveau physique : très peu de choses
  - Représenter l'information, et
  - Traiter l'information représentée

- Information
  - Distinction d'un état donnée parmi plusieurs, à un instant donné
  - Physique : on sait faire avec deux états : chargé (1V) ou pas
  - 2 états → codage binaire
    - 1 bloc (bit) : 2 états
    - 2 blocs : 4 états
    - n blocs ?
  - Valeur d'un bit : connaissance d'un état parmi deux
  - Bloc de 8 bits : octet (Byte)
  - Du coup, quand on parle de kilo octet ?

Ordinateurs
Représentation

Préfixe	Symbole	Puissance de 10
nano	n	10^{-9}
micro	μ	10^{-6}
milli	m	10^{-3}
-	-	10^{0}
kilo	k	10^{3}
méga	M	10^{6}
giga	G	10^{9}
téra	T	10^{12}
péta	P	10^{15}
exa	E	10^{18}
zetta	Z	10^{21}
yotta	Y	10^{24}

Ordinateurs
Représentation

- Du coup, quand on parle de kilo octet ? Binaire, n'oublions pas.
- 1024 octets souvent abusivement noté 1ko
- Norme CEI-60027-2 de préfixes binaires

Préfixe courant	Symbole courant	Préfixe standardisé	Symbole standardisé	Puissance de 2		Puissance de 10 la plus proche
-	-	-	-	2^0	1	10^0
kilo	k	kibi	ki	2^{10}	1024	10^3
méga	M	mébi	Mi	2^{20}	1048576	10^6
giga	G	gibi	Gi	2^{30}	1073741824	10^9
téra	T	tébi	Ti	2^{40}		10^{12}

- 1024 octets : 1kio
- 1 milliard d'octets = 0,9313Go …
- Des fois, 1 « téra » = 1 000 000 000 000 d'octets = 931,32Go !

Ordinateurs
Représentation

- Instructions en mémoire stockées sous forme d'un mot binaire
  - opcode : code opération
  - opérandes (zéro, une ou plusieurs) : valeurs ou emplacement des sources, emplacement du résultat

- Exemple sur une machine 16 bits

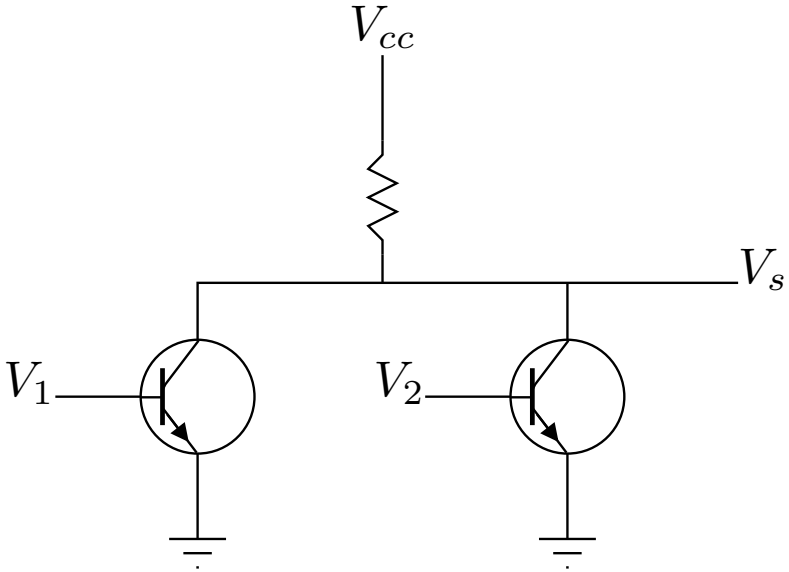
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
opcode				adresse											

- Exemples d'opcodes :
  - 0001 : charger le mot dont l'adresse est stockée dans ACC
  - 0010 : stocker le mot contenu dans ACC à l'adresse donnée

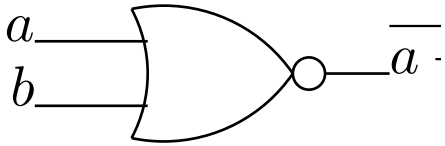
Niveau 0
Portes logiques

- Traitement de données élémentaires
- Il y a 50 ans : tubes à vide
- Aujourd'hui : transistors (nMOS, pMOS, CMOS)

- Entrées : 2 valeurs distinctes (0 ou 1, Vrai ou Faux …)
- Sortie : application fonction logique simple NON, ET, OU …



V1	V2	Vs
0	0	1
0	1	0
1	0	0
1	1	0



Niveau 1
Micro-architecture

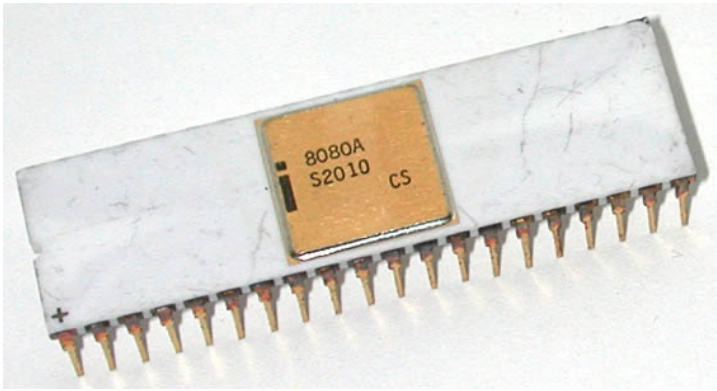
- Organisation des portes → circuits spécialisés

- Micro-architecture : matériel pour exécuter du langage machine
  - Circuits pour opérations de base (LM), logique et calcul : UAL
  - Chef d'orchestre : circuits de contrôle
  - Métronome : horloge

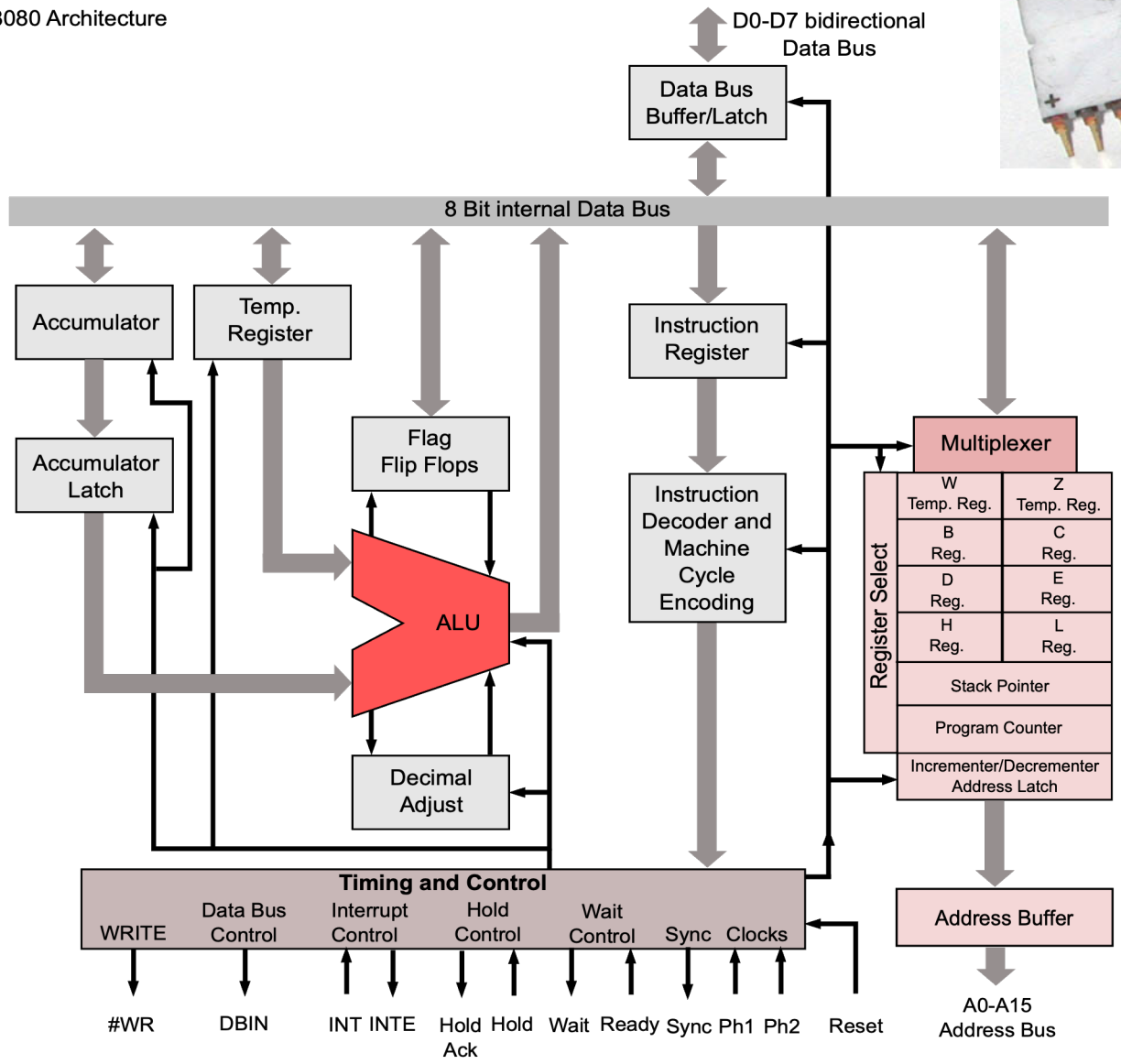
  composent le processeur

- Cycle d'exécution d'une instruction
  - Exemple : Lec I | Lec O | Exé | Ran (4 cycles d'horloge)

Niveau 1
Micro-architecture



Intel 8080 Architecture



Niveau 2
Architecture

- Informations sur le matériel si on veut programmer à ce niveau

- Architecture logicielle : ensemble des informations visibles au programme en LM
  - Ensemble des instructions (jeu d'instructions) + format
  - Organisation de la mémoire (adressage)
  - Représentations élémentaires (entiers, flottants …) (on est en binaire)

- Même architecture logicielle pour différentes micro-architectures
  - Architecture x86 : depuis 1978 jusqu'à maintenant
  - Micro-architectures 8086, 80386, Pentium I, Pentium IV très différents avec la même architecture x86

Niveau 2
Architecture

instruction	action	nzp	codage en langage machine															
			opcode				arguments											
			F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
NOT DR,SR	DR <- not SR	*	1	0	0	1	DR			SR			1	1	1	1	1	1
ADD DR,SR1,SR2	DR <- SR1 + SR2	*	0	0	0	1	DR			SR1			0	0	0	SR2		
ADD DR,SR1,Imm5	DR <- SR1 + SEXT(Imm5)	*	0	0	0	1	DR			SR1			1	Imm5				
AND DR,SR1,SR2	DR <- SR1 and SR2	*	0	1	0	1	DR			SR1			0	0	0	SR2		
AND DR,SR1,Imm5	DR <- SR1 and SEXT(Imm5)	*	0	1	0	1	DR			SR1			1	Imm5				
LEA DR,label	DR <- PC + SEXT(PCoffset9)	*	1	1	1	0	DR			PCoffset9								
LD DR,label	DR <- mem[PC + SEXT(PCoffset9)]	*	0	0	1	0	DR			PCoffset9								
ST SR,label	mem[PC + SEXT(PCoffset9)] <- SR		0	0	1	1	SR			PCoffset9								
LDR DR,BaseR,Offset6	DR <- mem[BaseR + SEXT(Offset6)]	*	0	1	1	0	DR			BaseR			Offset6					
STR SR,BaseR,Offset6	mem[BaseR + SEXT(Offset6)] <- SR		0	1	1	1	SR			BaseR			Offset6					
LDI DR,label	DR <- mem[mem[PC + SEXT(PCoffset9)]]	*	1	0	1	0	DR			PCoffset9								
STI SR,label	mem[mem[PC + SEXT(PCoffset9)]] <- SR		1	0	1	1	SR			PCoffset9								
BR[n][z][p] label	Si (cond) PC <- PC + SEXT(PCoffset9)		0	0	0	0	n	z	p	PCoffset9								
NOP	No Operation		0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
JMP BaseR	PC <- BaseR		1	1	0	0	0	0	0	BaseR			0	0	0	0	0	0
RET (JMP R7)	PC <- R7		1	1	0	0	0	0	0	1	1	1	0	0	0	0	0	0
JSR label	R7 <- PC ; PC <- PC + SEXT(PCoffset11)		0	1	0	0	1	PCoffset11										
JSRR BaseR	R7 <- PC ; PC <- BaseR		0	1	0	0	0	0	0	BaseR			0	0	0	0	0	0
RTI	cf. interruptions		1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TRAP Trapvect8	R7 <- PC ; PC <- mem[Trapvect8]		1	1	1	1	0	0	0	0			Trapvect8					
Réservé			1	1	0	1												

Niveau 2
Architecture

- Format des instructions ?
  - Longueur fixe ou variable
  - Nombre d'opérandes par instruction
  - Nombre d'opérandes mémoire par instruction

- RISC : Reduced Instruction-Set Computer
  - Taille d'instruction fixe, 3 opérandes, 0 mémoire (sauf lecture et écriture)
  - Cycle d'exécution simple et prévisible
  - Ex. MIPS, Alpha, PowerPC, Sparc

- CISC : Complex Instruction-Set Computer
  - Taille d'instruction variable
  - Instructions plus complexes à disposition
  - Ex. x86

Niveau 2
Architecture

- Adressage : accéder aux opérandes

- Mode registre : l'opérande est la valeur contenue dans un registre
  - `ADD DR, SR1, SR2` effectue DR <- SR1 + SR2
  - 0100 0111 0001 0010 : range dans R7 la somme des contenus de R1 et R2

- Mode immédiat : l'opérande est directement la valeur
  - `ADDI DR, SR, IMM` effectue DR <- SR1 + IMM
  - 0101 0111 0001 0010 : range dans R7 la somme du contenu de R1 et 2

- Mode direct : adresse dans l'instruction
  - `STRD SR1, ADR` range le contenu de SR1 à l'adresse ADR
  - 1101 0111 0001 0010 : range à l'adresse 18 le contenu de R7

Niveau 3
Système d'exploitation

- Pénible si on programme pour plusieurs architecture

- Système d'exploitation : abstraction et outils pour gérer architecture et matériel
  - Uniformisation des commandes et accès — visible au programmeur
  - Spécialisation pour architecture — invisible

- Ex. Linux
  - Même interface pour les versions AMD64, ARM, PowerPC, IA-32 (x86) …

Niveau 4
Langage d'assemblage

- Jusqu'ici que des 0 et des 1 …

- Langage d'assemblage : langage de niveau intermédiaire
  - Très très simple
  - Traduit vers OS + LM à l'aide d'un assembleur

Niveau 5
Langages de haut niveau

- Langage d'assemblage un peu primitif quand même

- Langages de haut niveau : Caml, C++ … COBOL …
  - Fonctions et opérations complexes décomposes en instructions
  - Traduits vers LA + OS à l'aide de compilateurs

Niveau 5 → Niveau 1
Exemple

- Un programme en langage C (langage de haut niveau)

```
#include <stdio.h>

char car;
int main(void) {
  printf("Hi!\n"); // appel à une primitive de l'OS
  printf("Entrez un caractere...\n");
  car = getchar(); // appel à une primitive de l'OS
  printf("Vous avez entre : ");
  putchar(car); // appel à une primitive de l'OS
  putchar('\n');
  printf("Bye!\n");
  return(0);
}
```

- On le compile avec un compilateur

  → On obtient un programme écrit dans un langage d'assemblage (ici LC3) :

Niveau 5 → Niveau 1
Exemple

- On le compile avec un compilateur

  → On obtient un programme écrit dans un langage d'assemblage (ici LC3) :

```
        LEA R0,msg0 ; charge l'adresse effective désignée par msg0 dans R0
        TRAP x22 ; affiche la chaine pointée par R0
        LEA R0,msg1 ;
        TRAP x22 ; affiche la chaine à l'adresse msg1
        TRAP x20 ; lit un caractère et le place dans R0
        ST R0,car    ; stocke le caractère lu à l'adresse car
        LEA R0,msg2 ;
        TRAP x22 ; affiche la chaine à l'adresse msg2
        LD R0,car    ; charge le caractère stocké à l'adresse car dans R0
        TRAP x21 ; affiche le caractère qui a été lu
        LEA R0,ret   ;
        TRAP x22 ; affiche un retour à la ligne
        LEA R0,msg3 ;
        TRAP x22 ; affiche la chaine à l'adresse msg3
        TRAP x25 ; termine le programme (rend la main à l'OS)
car:    .BLKW #1 ; case mémoire pour stocker un caractère lu
msg0:   .STRINGZ "Hi!\n"
msg1:   .STRINGZ "Entrez un caractere...\n"
msg2:   .STRINGZ "Vous avez entre : "
msg3:   .STRINGZ "Bye!\n"
ret:    .STRINGZ "\n"
```

Niveau 5 → Niveau 1
Exemple

- On le traduit (on l'assemble) avec un assembleur

  → On obtient un programme écrit en langage machine :

```
E00F F022 E012 F022 F020 3009 E026 F022 2006 ...
```

  → Plus lisiblement :

langage machine	langage machine hexa	langage d'assemblage
1110 0000 0000 1111	xE00F	LEA R0,msg0
1111 0000 0010 0010	xF022	TRAP x22
1110 0000 0001 0010	xE012	LEA R0,msg1
1111 0000 0010 0010	xF022	TRAP x22
1111 0000 0010 0000	xF020	TRAP x20
0011 0000 0000 1001	x3009	ST R0,car
1110 0000 0010 0110	xE026	LEA R0,msg2
1111 0000 0010 0010	xF022	TRAP x22
0010 0000 0000 0110	x2006	LD R0,car

Ordinateurs
Problématiques

- Performances ?
- En général ?
- Spécialisé ?
  - Calcul scientifique
  - BD
  - Traitement du signal (DSP)
  - Graphique
    - Aujourd'hui massivement parallèle

- OPS ? Communication ? Efficacité énergétique ?
  - En fonction de la gravure :

	90nm	65nm	45nm	32nm	22nm
GOPS	2	14	77	461	2458
kbps	384	2304	13824	82944	497664
mW max	100				

Ordinateurs
Problématiques

- Calcul
- Métronome : horloge

  Lect. Instr Lect. Op. Exécution Rangement
- Formule fondamentale :

$$T_{ex} = NI \times CPI \times T_c = \frac{NI}{IPC \times F}$$

  - T_{ex} : Temps d'exécution
  - NI : Nombre d'Instructions
  - CPI : Cycles Par Instruction = 1 / IPC
  - IPC : Instruction Par Cycle
  - T_c : Temps par cycle = 1 / F
  - F : Fréquence d'horloge

- De 1987 à 2004
  - Horloge : + 25% / an
  - Calcul : + 60% / an

Ordinateurs
Problématiques

- Mémoire
- Métronome : horloge
- De 1987 à 2004
  - Bande passante : RAM + 20% / an secondaire +40% / an
  - Latence : RAM + 6% / an secondaire + 7% / an
  - Capacité : RAM + 60% / an secondaire + 60% / an
  - Calcul : + 60% / an

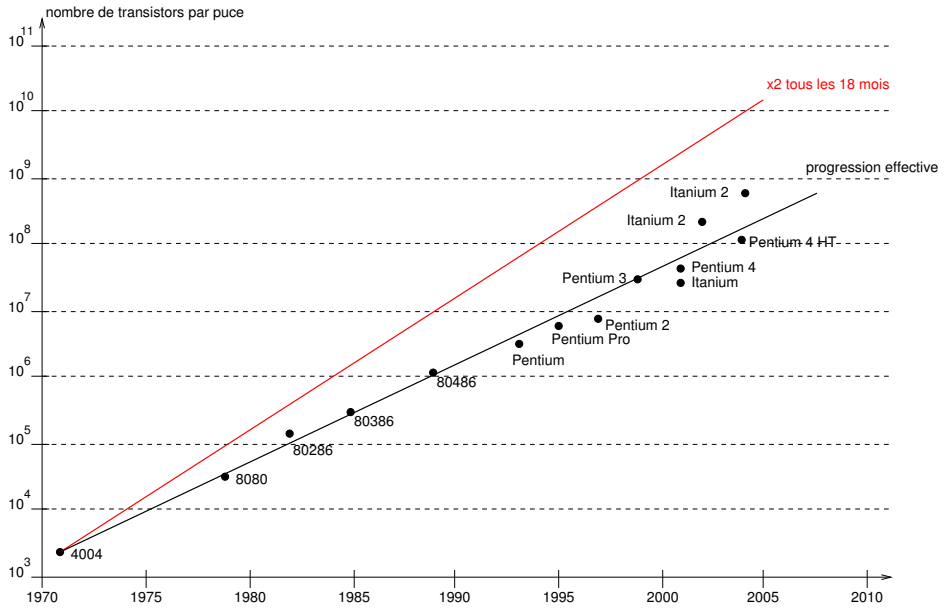
Ordinateurs
Technologie

- Engrenages
- Tubes à vide
- Transistors
  - nMOS, pMOS
  - CMOS
- Gravure
  - 1985 : 1000nm
  - …
  - 2002 : 90nm
  - 2006 : 65nm
  - 2008 : 45nm
  - 2010 : 32nm
  - 2012 : 22nm

Ordinateurs
Technologie

- Loi de Moore
  - Gordon Moore, Intel en 1965
  - Le nombre de transistors que l'on peut intégrer sur une puce, avec la technologie la plus économique, double tous les 18 mois

- En fait x 2 tous les 2 ans
  - 1985 : 80386 → 275 000
  - 1989 : 80486 → 1 000 000
  - 1995 : Pentium pro → 5 500 000
  - 2003 : K8 → 100 000 000
  - 2011 : Core i7 → 995 000 000
  - 2014 : Power8 → 4 200 000 000



- Fonctionnement : 1V → courant

  → chaleur

Ordinateurs
Chaleur

- Densité puissance : exponentielle (mono-processeur)
  - Pentium pro : ~ 10W/cm2
  - Pentium IV : ~ 50W/cm2
  - Cœur centrale nucléaire : ~ 200W/cm2

→ architectures parallèles

- Puissance dissipée

$$P_d = P_s + \alpha \sum_i C_i V^2 F$$

  - P_d : puissance dissipée
  - P_s : puissance statique
  - α : pourcentage de composants actifs
  - C_i : capacité
  - V : tension
  - F : fréquence de l'horloge

Ordinateurs
Chaleur

- Puissance dissipée

$$P_d = P_s + \alpha \sum_i C_i V^2 F$$

- Lorsque F augmente
  - Baisser P_s ? → non, fuites
  - Baisser V ? → non, seuil technologique
  - Baisser C_i ? → non, augmentation Nb. Transistors

- Puissance dissipée linéairement proportionnelle à F
  - Reste → difficile d'y toucher
- Du coup limite d'horloge : ~ 4GHz (depuis 2000)
- Gravure plus fine → densité
  - Util. F_{45} : 100% à 45nm 40% à 22nm 20% à 11nm
  - Util. F_{max} : 100% à 45nm 25% à 22nm 10% à 11nm

Ordinateurs
Problématiques

- Calcul

$$T_{ex} = NI \times CPI \times T_c = \frac{NI}{IPC \times F}$$

- Augmenter F ? → non, puissance dissipée
- Augmenter IPC ? → oui, pipelines, superscalaires
  - Séquentiel :

Lec I	Lec O	Exé	Ran	Lec I	Lec O	Exé	Ran

  - Pipeline :

Lec I	Lec O	Exé	Ran				
	Lec I	Lec O	Exé	Ran			
		Lec I	Lec O	Exé	Ran		
			Lec I	Lec O	Exé	Ran	
				Lec I	Lec O	Exé	Ran

Ordinateurs
Problématiques

- Calcul

$$T_{ex} = NI \times CPI \times T_c = \frac{NI}{IPC \times F}$$

- Augmenter IPC ? → oui, pipelines, superscalaires
  - Séquentiel :

Lec I	Lec O	Exé	Ran	Lec I	Lec O	Exé	Ran

  - Superscalaire (quand possible) :

Lec I	Lec O	Exé	Ran				
Lec I	Lec O	Exé	Ran				
	Lec I	Lec O	Exé	Ran			
	Lec I	Lec O	Exé	Ran			
		…					

Ordinateurs
Problématiques

- Calcul

$$T_{ex} = NI \times CPI \times T_c = \frac{NI}{IPC \times F}$$

- Du coup (encore) augmenter IPC ?

→ pas évident :

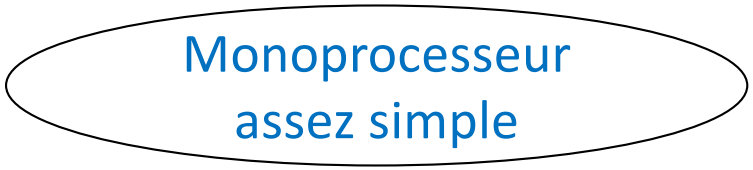
Pentium Pro → 3

i7 4ème génération → 4

- Diminuer NI ? → oui
  - Compilateurs +
    - SIMD (Single / Multiple Instruction / Data) → vecteur 512 bits
    - SIMT (Multithread) → GPU
    - Architectures parallèle

Ce cours

- Super-calculateurs
  - Actuellement : plusieurs millions de cœurs, plusieurs MW
  - Motivations ? Par exemple, programme *Simulation*



(mais entièrement)

- Physique
  - Actuellement : électricité