Introduction à l'architecture ARM

GIF-1001 Ordinateurs: Structure et Applications, Hiver 2015

Jean-François Lalonde

Aujourd'hui

- Rappel:
  - TP1: dû mardi prochain
  - atelier: TP1 + micro-instructions
- Processeurs ARM
- (très rapidement) Processeurs de la famille x86

Deux approches différentes

- Tandis qu'Intel produit des microprocesseurs 8086 et 80286, des employés d'une compagnie anglaise inspirés par les travaux de l'heure travaillent à la conception d'un ordinateur

- En 1985, la compagnie ACORN produit un premier processeur qui résulte de ces travaux: le Acorn RISC Machine (ARM)

- La complexité du ARM est réduite par choix et par faute de moyens et le jeu d'instructions est simplifié.

- Alors que le 8086 gagne en popularité, ACORN et Apple collaborent pour fonder la compagnie ARM ltd en 1990

- ARM et Intel feront évoluer leurs produits en parallèle dans des marchés différents

ARM

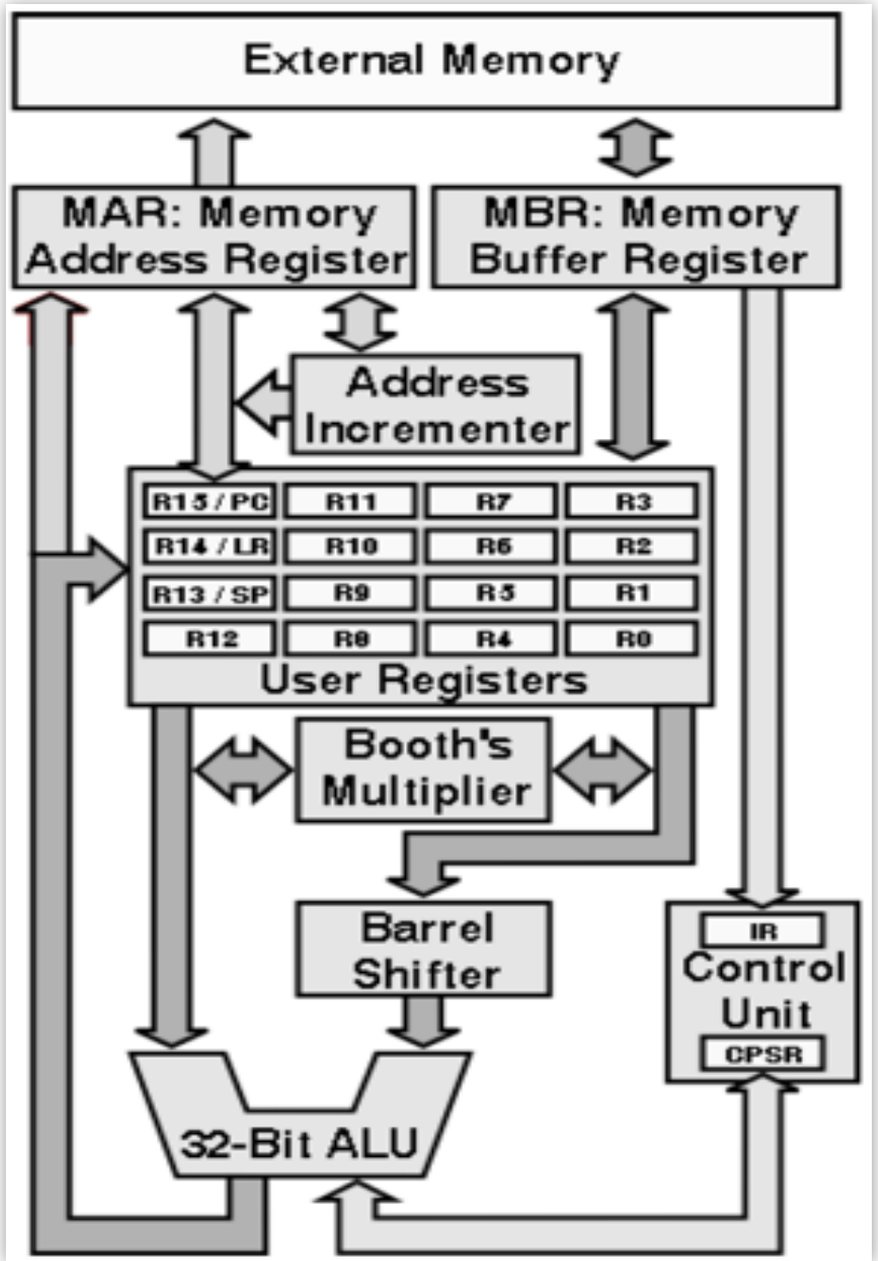
- ARM Holdings:
  - développe des architectures de micro-processeurs et des jeux d'instructions
  - ne construit aucun micro-processeur comme tel! La compagnie licencie la technologie à d'autres qui l'implémentent à leur façon en hardware.
    - clients: Apple, Nvidia, Samsung, Texas Instruments, etc…
- Micro-processeurs ARM
  - Supportent 32 et 64 bits
  - L'architecture **la** plus utilisée au monde
    - 10 milliards produits en 2013
    - 98% de téléphones portables contiennent au moins 1 processeur ARM

Processeurs ARM

- Plusieurs versions de processeurs, utilisées partout!
  - ARM7TDMI(-S): Nintendo DS, Lego NXT
  - ARM946E-S: Canon 5D Mark ii (caméra)
  - ARM1176JZ(F)-S: Raspberry Pi
  - Cortex-A9: Apple iPhone 4S, iPad2
  - Cortex-A15: Nexus 10
- Beaucoup d'autres!
  - http://en.wikipedia.org/wiki/List_of_applications_of_ARM_cores

Architecture des ARM (32 bits)

- Architecture RISC pour laquelle tout passe par des registres
- Registres
  - 16 registres généraux
  - PC = 32 bits
    - CPSR (statut) = 32 bits
- Instructions de 32 bits
- Accès à 2^{32} octets (4GB) de mémoire
- “Little ou Big endian” au choix
- Opère sur 8, 16 et 32 bits



Big vs Little Endian

- La taille minimum d'une donnée stockée en mémoire est habituellement 1 octet
- Les données ayant plusieurs octets (integer, long, float) peuvent être stockées de 2 façons:
  - "Little endian": l'octet le moins significatif est placé à la plus petite adresse dans la mémoire
    - Intel: x86
  - "Big endian": l'octet le moins significatif est placé à la plus haute adresse dans la mémoire
    - Motorola: 68000

Architecture des ARM (suite)

- Implémentation de petite taille qui favorise une *consommation réduite d'énergie*
- Accès à la mémoire régit par le contenu des registres et par les instructions
- Instructions qui combinent décalages et opérations logiques et arithmétiques
- Possibilité d'impliquer plusieurs registres dans une opération à l'aide d'une seule instruction
- Exécution conditionnelle possible pour la plupart des instructions

ARM vu par le programmeur

- Différents modes (7) d'opération sont possibles dont le:
  - User mode: mode normal d'opération
  - Supervisor mode: mode protégé pour les systèmes d'exploitation
  - Privileged mode: mode privilégié pour certaines tâches des systèmes d'exploitation
- Deux jeux d'instructions sont supportés:
  - Les instructions ont toutes 32 bits sauf si le jeu d'instructions Thumb est actif
  - Les instructions du mode Thumb ont 16 bits pour économiser de la mémoire au besoin

ARM vu par le programmeur (suite)

- Format de variables:
  - Octets de 8 bits (Bytes)
  - Mots de 32 bits (Words)
  - Demi-mots de 16 bits (Halfwords)
  - Mots doubles de 64 bits (Doublewords)
- Contenu possible des registres de 32 bits:
  - Adresse de 32 bits
  - Nombres entiers (signés ou non) de 32 bits
  - Nombres entiers (signés ou non) de 8 ou 16 bits
    - Deux nombres entiers de 16 bits par registre
    - Quatre nombres entiers de 8 bits par registre
  - Nombres sans signe de 64 bits utilisant 2 registres

Les registres

- 16 registres de 32 bits sont disponibles:
  - R0 à R12: usage général
  - R13 à R15: registres “spéciaux” (voir pages suivantes)
- Le “Current Program Status Register” (CPSR) est utilisé pour mémoriser les résultats d’opération

Le registres spéciaux R13 à R15

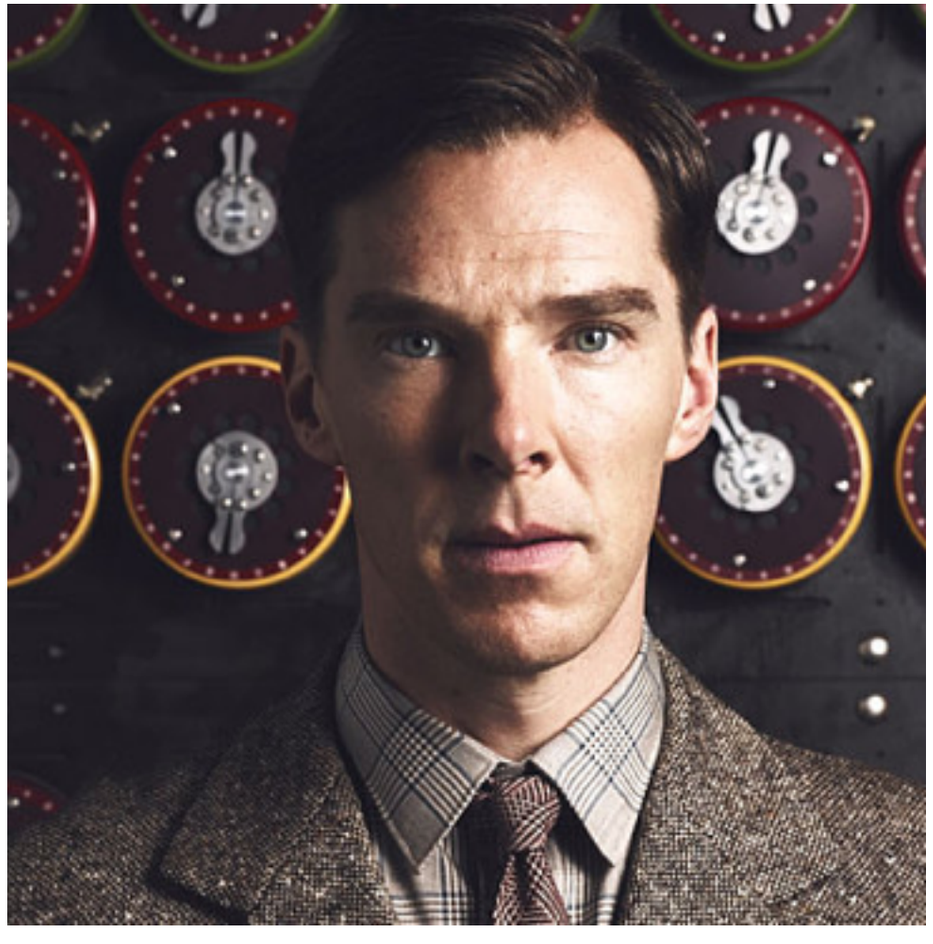
- R13: Pointeur de pile (Stack Pointer ou SP)
  - Indique où trouver des valeurs temporaires
  - À ne pas utiliser pour autre chose que pour la pile!
- R14: Registre de liens (Link Register ou LR)
  - Indique à partir d'où un appel de fonction a été fait
  - Sa copie dans R15 permet de poursuivre l'exécution du programme après celle d'une fonction
- R15: Compteur de programme (Program Counter ou PC)
  - Indique où se trouve l'instruction en cours
  - Changer son contenu permet de faire un "branchement" dans le programme ou un retour de fonction

Piles (“stacks”)

- Structure de données très utilisée en informatique
- Inventée par Alan Turing (1946)

Alan Turing

Imitation Game
(cette fois-ci c’est vrai!)



Piles (“stacks”)

- Structure de données “LIFO”
  - LIFO = Last In, First Out
- Deux opérations principales:
  - PUSH: rajoute un élément sur “le dessus” la pile
  - POP: enlève un élément du “dessus” de la pile

Piles (“stacks”)

- Exemples?
  - Notation Polonaise Inverse dans les calculatrices (“RPN”)
  - Algorithme pour naviguer un labyrinthe

Le “Stack Pointer” R13

- Par convention, on définit un bloc de mémoire qu’on nomme “pile” (stack)
- La pile est utilisée pour sauvegarder temporairement des valeurs de travail
  - exemples: variables locales, paramètres et valeurs de retour des fonctions
- Le Stack Pointer contient une adresse de la pile à laquelle on peut écrire ou lire une valeur
- Comme changer la valeur de SP directement peut être dangereux (il ne faut pas perdre la trace de la pile!), on procède plutôt en l’augmentant (“push”) ou en la diminuant (“pop”) à l’aide d’instructions dédiées
- Un usage hors-norme de SP ne peut que causer des problèmes si on n’a pas le plein contrôle du programme

Appel de fonction simple (sans paramètres)

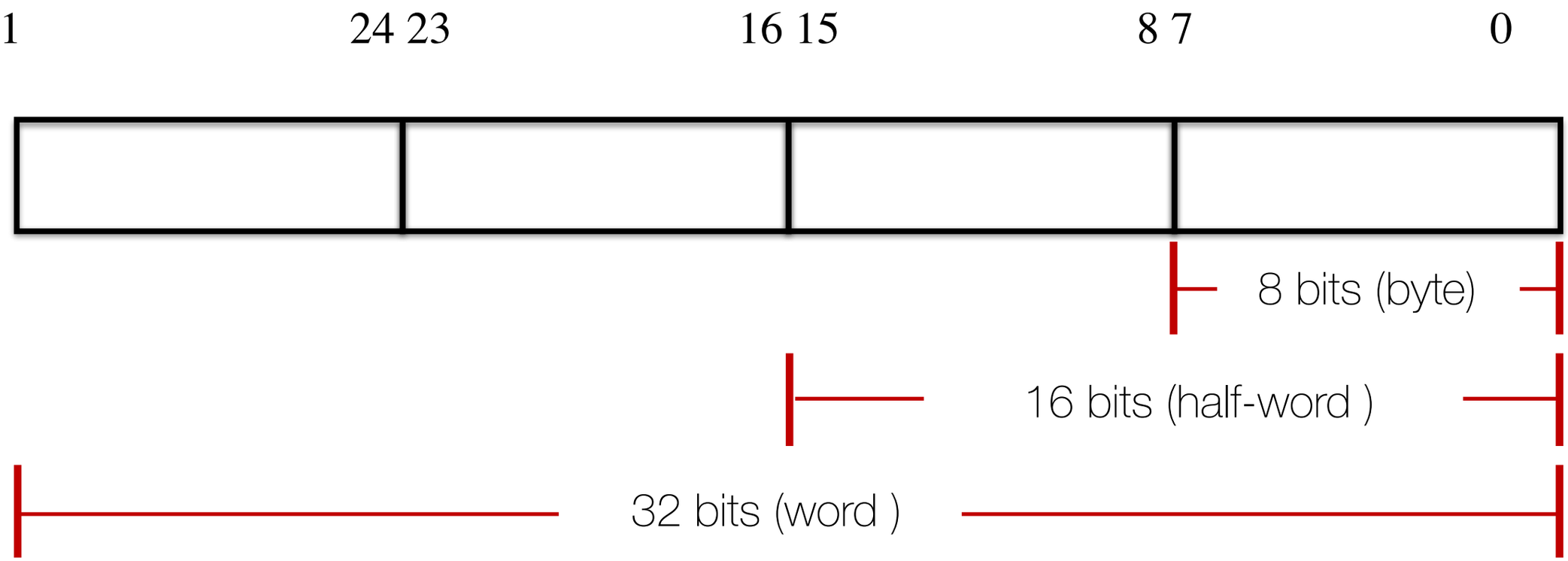
- Mnémonique assembleur indique un appel de fonction (ex: CALL maFonction)
  - maFonction:
    - INST 1 ; première instruction
    - … ; liste d'instructions
    - RET ; "return", indique que la fonction est terminée
- Il suffirait de modifier le PC pour le mettre à l'adresse correspondant aux instructions de "maFonction"
- Cependant, que faire lorsque l'on parvient à RET? Où doit-on revenir?

Le Link Register R14

- Un appel de fonction à l'aide d'instructions dédiées entraîne un changement de la valeur du Program Counter
- Un retour de fonction ne peut se faire que si PC reprend sa valeur d'avant l'appel de la fonction
- Par convention, le Link Register est utilisé pour sauvegarder l'adresse de retour pendant l'exécution de la fonction
- Mettre la valeur de LR dans PC commande un retour de fonction
- Utiliser LR à d'autres fins ne peut que causer des problèmes si on n'a pas le plein contrôle du programme
- L'appel d'une fonction par une fonction commande une gestion de LR qui se fait avec la pile par convention

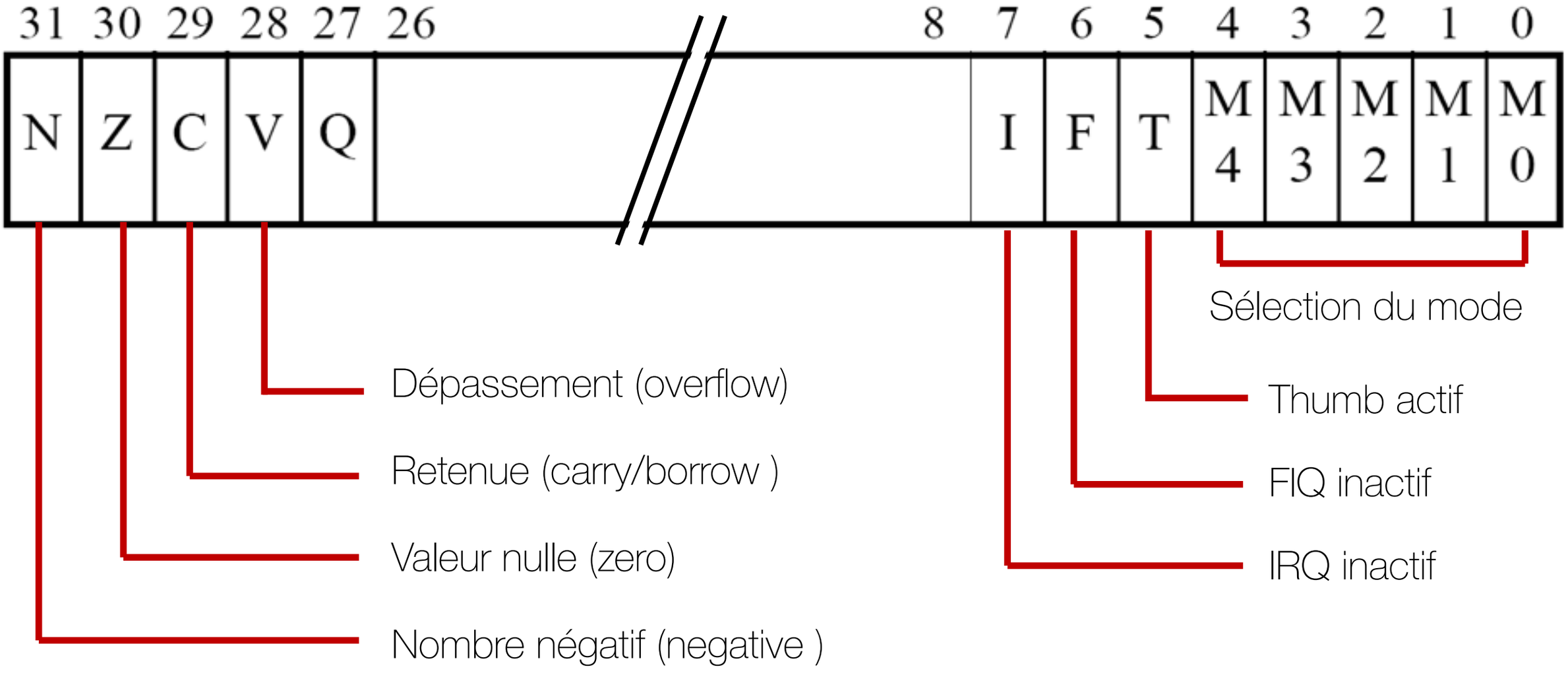
Registres d'usage général

- Gestion des nombres signés ou non à 8, 16 ou 32 bits
- Les calculs sont faits à 32 bits et la gestion des valeurs plus petites est logicielle



Registre de statut (CPSR)

- Un registre de statut décrit l'état du processeur



CPSR: Détection de conditions

- N: Détection de signe négatif
  - 1 si résultat < 0, 0 autrement
- Z: Détection de zéro
  - 1 si résultat = 0, 0 autrement
  - Souvent utilisé pour détecter les égalités
- C: Détection de retenue (“carry)” ou d’emprunt (“borrow”)
  - 1 si l’opération a impliqué une retenue
  - Ex. retenue d’addition de nombres positifs
- V: Détection de dépassements (overflow)
  - 1 si l’opération a impliqué un dépassement
  - Ex. dépassement signé lors d’une addition

Registre de statut (CPSR) (suite)

- Plusieurs mode peuvent être définis
- Les instructions permises varient selon le mode en cours

Table B1-1 ARM processor modes

Processor mode[a]		Mode encoding[b]	Privilege	Description
User	usr	10000	Unprivileged	Suitable for ~~most application code~~
FIQ	fiq	10001	Privileged	Entered as a result of a fast interrupt.[c]
IRQ	irq	10010	Privileged	Entered as a result of a normal interrupt.[c]
Supervisor	svc	10011	Privileged	Suitable for running most kernel code. Entered on Reset, and on execution of a Supervisor Call (SVC) instruction.
Monitor[d]	mon	10110	Privileged	A Secure mode that enables change between Secure and Non-secure states, and can also be used to handle any of FIQs, IRQs and external aborts.[c] Entered on execution of a Secure Monitor Call (SMC) instruction.
Abort	abt	10111	Privileged	Entered as a result of a Data Abort exception or Prefetch Abort exception.[c]
Undefined	und	11011	Privileged	Entered as a result of an instruction-related error.
System	sys	11111	Privileged	Suitable for ~~application code that requires privileged access~~

Les instructions du ARM

- Instructions de 32 bits sauf dans le mode Thumb où elles ont toutes 16 bits
- Exécution des instructions conditionnelles dans tous les cas
- Possibilité d'affecter le contenu de plusieurs registres en une seule opération
- Possibilité de combiner des opérations logiques et arithmétiques en une seule instruction

Exemple d'instructions

	31 30 29 28	27 26 25	24 23 22 21	20	19 18 17 16	15 14 13 12	11 10 9 8	7	6 5	4	3 2 1 0
Data processing immediate shift	cond [1]	0 0 0	opcode	S	Rn	Rd	shift amount		shift	0	Rm
Miscellaneous instructions: See Figure 3-3	cond [1]	0 0 0	1 0 x x	0	x x x x x x x x x x x x				x x	0	x x x x
Data processing register shift [2]	cond [1]	0 0 0	opcode	S	Rn	Rd	Rs	0	shift	1	Rm
Miscellaneous instructions: See Figure 3-3	cond [1]	0 0 0	1 0 x x	0	x x x x x x x x x x x x			0	x x	1	x x x x
Multiplies, extra load/stores: See Figure 3-2	cond [1]	0 0 0	x x x x	x	x x x x x x x x x x x x			1	x x	1	x x x x
Data processing immediate [2]	cond [1]	0 0 1	opcode	S	Rn	Rd	rotate	immediate			

Les jeux d'instructions ARM (1/2)

- Plusieurs jeux d'instructions sont supportés en fonction du processeur (il y a plusieurs processeurs ARM: ARM7, ARM Cortex M0, ARM Cortex M3…):
  - ARM: 32 bits, 58 instructions principales, accès simultané à l'unité de calcul et à l'unité de décalage des mots, 15 registres d'usage général
  - Thumb: 16 bits (supporte aussi les instructions ARM sur 32 bits), 30 instructions principales, pas d'accès simultané aux unités de calcul et de décalage, 8 registres d'usage général, pas d'accès direct au CPSR

Les jeux d'instructions ARM (2/2)

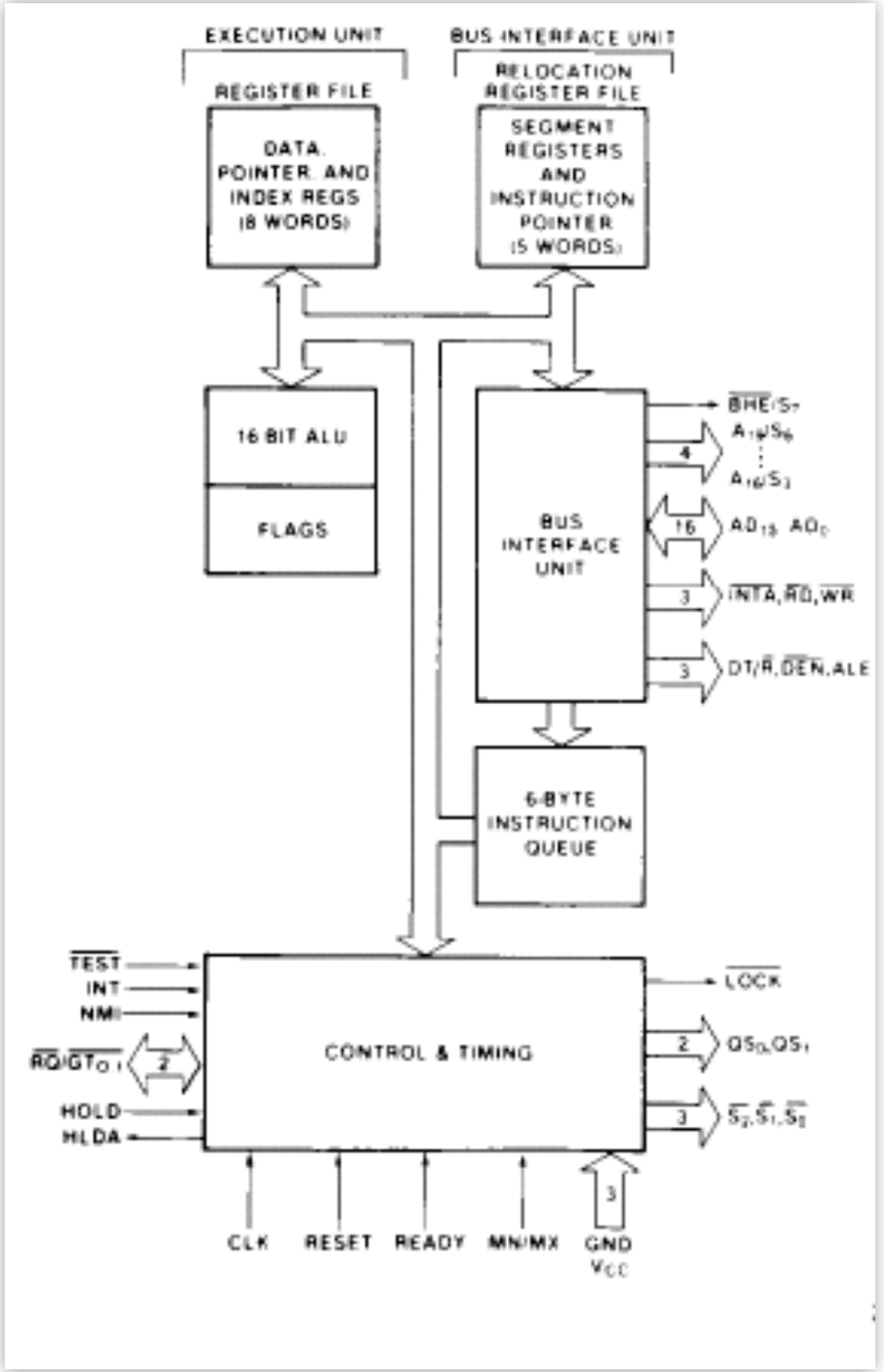
- Thumb2
  - Un mélange de Thumbs et d'ARM.
  - Supporte des instructions Thumbs 16 bits et des instructions ARM 32 bits.
    - L'opcode à l'intérieur des 16 premiers bits détermine la longueur de l'instruction.
- Jazelle
  - 8 bits, instructions reconnues et traitées par une partie du processeur conçue pour accélérer l'exécution du code écrit en Java

Questions?

Architecture x86

Architecture Intel

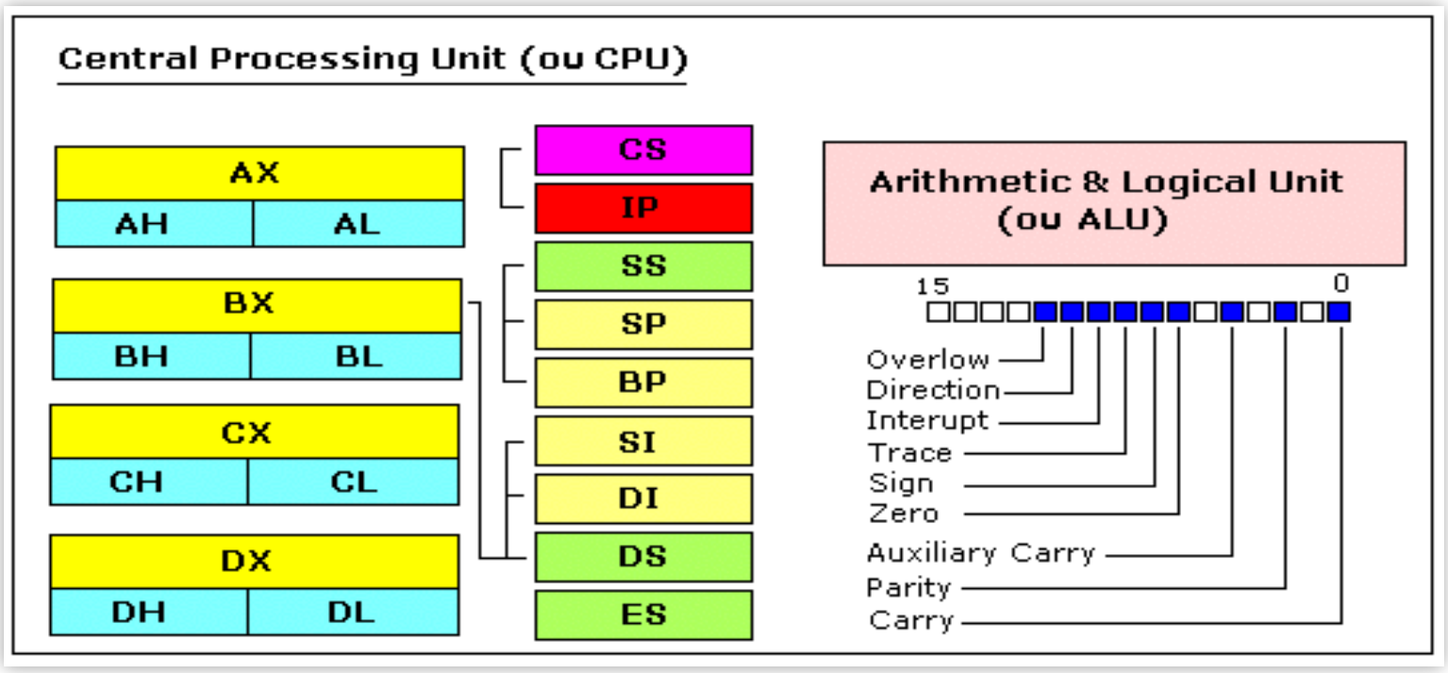
- Le 8086 est représentatif des principes qui orientent la conception des processeurs Intel
- Registre de 16 bits
- Combinaison du contenu décalé d'un registre de 16 bits à celui d'un autre pour créer des adresses de 20 bits
- Gère 64k à la fois à l'aide des concepts de segments et de pages
- Architecture CISC à instructions de longueur variable



Registres et statut

- De la même manière que les instructions sont conçues pour répondre aux besoins, les registres sont conçus en fonction des instructions et des besoins

- Des registres de 8 bits pourront être combinés pour faire des registres de 16 bits



Les principaux registres

- AX: Registre accumulateur de 16 bits (AH et AL)
- BX: Registre d'adresses de base de 16 bits (BH et BL)
- CX: Registre compteur de 16 bits (CH et CL).
- DX: Registre de données de 16 bits (DH et DL).
- SI: Registre d'index source de 16 bits.
- DI: Registre d'index destination de 16 bits.
- BP: Registre pointeur de base de 16 bits.
- SP: Registre pointeur de pile de 16 bits.
- IP: Registre pointeur d’instructions de 16 bits

Le registre de statut

- Carry Flag (CF): indique s'il y a retenue ou pas
- Zero Flag (ZF): indique si l'opération a donné zéro
- Sign Flag (SF): indique si le résultat est négatif
- Overflow Flag (OF): indique s'il y a eu dépassement
- Parity Flag (PF): indique si le nombre de 1 est pair
- Auxiliary Flag (AF): indique s'il y a retenue sur 4bits
- Interrupt enable Flag (IF): indique si les interruptions sont permises
- Direction Flag (DF): se rapporte à la façon de traiter les chaînes de caractères dans certains cas

Les successeurs du 8086

- La compatibilité avec le matériel déjà existant a toujours fait l'objet d'attentions chez Intel
- L'architecture des processeurs modernes reprend donc plusieurs éléments du 8086
- Des ajouts ont été faits pour améliorer les performances
- Les principes et concepts liés aux accumulateurs et aux pages et segments de données demeurent en vigueur

Sommaire

- L'architecture des processeurs ARM reprend les principes qui s'appliquent aux RISC tandis que les processeurs dérivés du 8086 d'Intel sont de type CISC

- La tendance est d'opter pour la simplicité de l'architecture, des registres et des jeux d'instructions quand il est question de RISC tandis que le format et la nature des registres des CISC sont d'une complexité qui va de pair avec celles de leurs jeux d'instructions

- La simplicité des RISC facilitera la mise en place de solutions qui ne consomment pas beaucoup d'énergie