Systèmes à Microprocesseurs

Cycle Ingénieur Troisième Année

Yoann Charlon

Plan

- Ch1 – Représentation de l’information
- Ch2 – ARM Instruction Set Architecture
- Ch3 – Accès aux données
- Ch4 – Programmation structurée
- Ch5 – Cycle d’exécution
- Ch6 – Codage binaire
- Ch7 – Microcontrôleur ARM Cortex-M

Représentation de l’information

- Représentation de l’information
  - Opérations sur les données

Représentation de l'information

- Types de données de base
  - Nombres entiers (0, -23, 10, 6, 65635)
  - Nombres réels (0.3, -3.2, 5, 10e3)
  - Booléens (true, false)
  - Caractères ('a', 'b', '?', '&', '0')
- Systèmes de numération
  - Décimal: 0 … 9, décomposition en puissances de 10
  - Hexadécimal: 0 à 9 et A…F, décomposition en puissances de 16
  - Binaire: 0 ou 1, décomposition en puissances de 2
- Décimal et hexadécimal sont utilisables pour l'être humain
- La représentation binaire est utilisable par l'ordinateur

Représentation de l'information

- Mot binaire
  - Bit (Binary digit)
  - Quartet (nibble) 4 bits
  - Octet (byte) 8 bits
- Terminologie utilisée sur les processeurs 32-bit (y compris ARM)
  - Demi-mot (half-word) 16 bits
  - Mot (word) 32 bits
  - Double-mot (double word) 64 bits
- /!\ la terminologie peut changer d'une famille à l'autre
  - Motorola, Intel
    - word 16 bits, long word 32 bits, quad word 64 bits

Représentation des nombres entiers

- Binaire pur sur n bits
  - Représentation des entiers entre 0 et $2^n - 1$

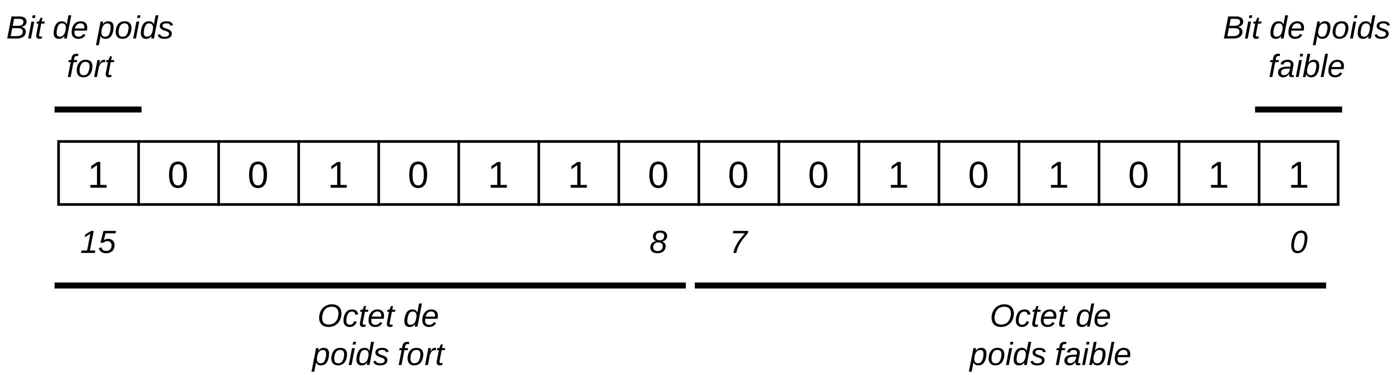
- Exemples (sur 8 bits):
  - $(00000000)_{bin} = (0)_{dec}$
  - $(11111111)_{bin} = (2^8-1)_{dec} = (255)_{dec}$
  - $(11001001)_{bin} = 1x\ 2^7 + 1x\ 2^6 + 0x\ 2^5 + 0x\ 2^4 + 1x\ 2^3 + 0x\ 2^2 + 0x\ 2^1 + 1x\ 2^0$

    $= (128 + 64 + 8 + 1)_{dec}$

    $= (201)_{dec}$

Représentation des nombres entiers

- Vocabulaire
  - Bit de poids fort: bit le plus significatif (MSB, Most Significant Bit)
  - Bit de poids faible: bit le moins significatif (LSB, Least Significant Bit)



Représentation des nombres entiers signés

- Nombre entier signé : signe = bit de poids forts
  - Poids fort à 0 => nombre positif
  - Poids fort à 1 => nombre négatif

- Représentation des entiers signés entre -2^{n-1} et $2^{n-1}-1$
  - Nombres positifs: correspondance directe avec le binaire pur
  - Nombres négatifs: conversion nombre positif vers négatif avec **complément A2**

- Méthode pour prendre l'opposé d'un nombre en complément à deux
  - Complémenter (=inverser) bit à bit (complément A1)
  - Puis on ajoute 1 au résultat (complément A2)

Représentation des nombres entiers

- Complément à deux sur n bits
  - Exemples (sur 8 bits)
    - $(10000000)_{bin} = (-128)_{dec}$
    - $(11111111)_{bin} = (-1)_{dec}$
    - $(00000000)_{bin} = (0)_{dec}$
    - $(01111111)_{bin} = (127)_{dec}$

    - $(5)_{dec} = (00000101)_{bin}$
    - $(-5)_{dec} = (\overline{00000101})_{bin} + 1 = (11111010)_{bin} + 1 = (11111011)_{bin}$

Représentation des nombres entiers

Décimal	Binaire pur	Complément à deux
255	11111111	
…	…	
129	10000001	
128	10000000	
127	01111111	01111111
…	…	…
3	00000011	00000011
2	00000010	00000010
1	00000001	00000001
0	00000000	00000000
-1		11111111
-2		11111110
-3		11111101
…		…
-127		10000001
-128		10000000

Représentation des nombres entiers

- Correspondance entre binaire et hexadécimal
  - Un chiffre hexadécimal pour 4 bits
  - Exemples:
    - $(53)_{hex} = (0101\ 0011)_{bin}$
    - $(FF)_{hex} = (1111\ 1111)_{bin}$
  - Nombres négatifs
    - En décimal on fait figurer le signe –
    - En hexadécimal par convention, on traduit directement à partir du binaire
  - Exemple en complément à deux sur 8 bits
    - $(-1)_{dec} = (1111\ 1111)_{bin} = (FF)_{hex}$

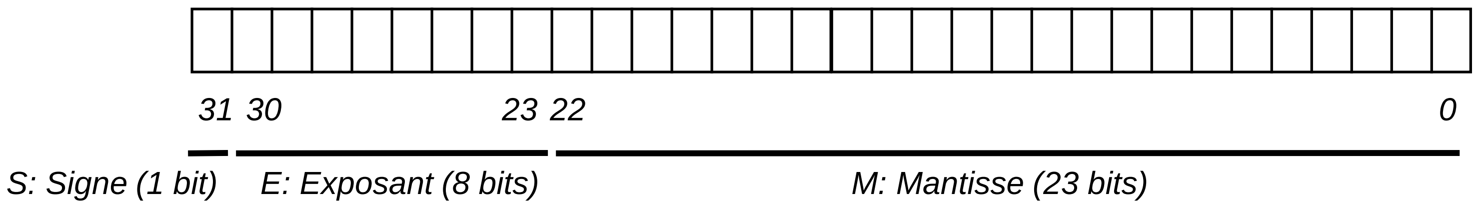
Représentation des caractères

- Le code ASCII
  - American Standard Code for Information Interchange
  - 1 caractère = 1 octet
  - Exemples:

Décimal	Caractère	Décimal	Caractère	Décimal	Caractère
32	ESPACE	48	0	91	[
33	!	49	1	92	\
34	‘’	…	…	63	]
35	#	57	9	…	
36	$	…	…	…	
37	%	65	A	97	a
38	&	66	B	98	b
39	’	…	…	…	…
40	(	90	Z	122	z

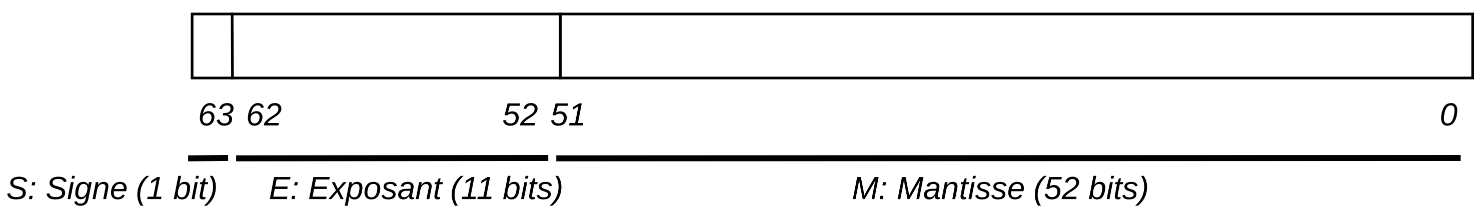
Représentation des nombres réels

- Le standard IEEE 754 (32-bit, simple précision)



$$(-1)^{S} * (1 + M/2^{23}) * 2^{E-127}$$

- Le standard IEEE 754 (64-bit, double précision)



$$(-1)^{S} * (1 + M/2^{52}) * 2^{E-1023}$$

Représentation des nombres réels

- Exemple avec standard IEEE 754 :

  0 1000 1000 100 0001 0101 0100 0100 0101

  - Signe: 0 positif
  - Exposant: 10001000 = $(136)_{dec}$
  - Mantisse: 1000001 01010100 01000101 = $(4281413)_{dec}$

  - Résultat: $(1 + 4281413/2^{23}) * 2^{136-127} =$

    $= 1.5103842 * 2^{9} = 773.31671$

Représentation de l’information

- Représentation de l’information
  - Opérations sur les données

Opérations arithmétiques

- Addition n bits
  - Exemple

$$
\begin{array}{cccccccccc}
 & & & 1 & 1 & 1 & & 1 & 1 & \\
 & & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\
+ & & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \\
\hline
 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0
\end{array}
$$

- A chaque étape, on réalise la somme de trois bits
  - Un bit de chacun des deux opérandes
  - La retenue de l'étape précédente
- Bit d'état (=drapeau): C=1

Opérations arithmétiques

- Addition de nombres signés
  - Utilisation du complément à deux

$$
\begin{array}{rl}
-5 & \rightarrow 0000\,0101 \rightarrow 1111\,1010 + 1 \rightarrow \begin{array}{r} {\scriptstyle 1\ 1\ 1\ 1\ 1\ 1\ 1\ \ } \\ 1\,1\,1\,1\,1\,0\,1\,1 \end{array} \\
+\ 7 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad + \ 0\,0\,0\,0\,0\,1\,1\,1 \\
\hline
2 & \qquad\qquad\qquad\qquad\qquad\qquad\qquad \color{red}{1}\,0\,0\,0\,0\,0\,0\,1\,0
\end{array}
$$

- Indicateur d'état: C=1
  - La retenue sortante est ignorée dans le résultat
  - Risque de débordement

Opérations arithmétiques

- Addition de nombres signés
  - Risque de débordement

	A+B=S		n-1 n-2 0
			1 1 1 1 1
70	A		0 1 0 0 0 1 1 0
+ 95	+B	+	0 1 0 1 1 1 1 1
165	S		0 1 0 1 0 0 1 0 1

- bit d'état: C=0
  - La retenue sortante est ignorée dans le résultat
- bit de débordement: V=1
  - $V = \overline{A_{n-1}}.\overline{B_{n-1}}.S_{n-1} + A_{n-1}.B_{n-1}.\overline{S_{n-1}}$

Opérations arithmétiques

- Multiplication en binaire pur sur 8 bits

```
      110
  x   102
  -------
      220
     000
    110
  -------
    11220
```

```
                    0 1 1 0 1 1 1 0
                  x 0 1 1 0 0 1 1 0
          ---------------------------
                    0 0 0 0 0 0 0 0
                  0 1 1 0 1 1 1 0
                0 1 1 0 1 1 1 0
              0 0 0 0 0 0 0 0
            0 0 0 0 0 0 0 0
          0 1 1 0 1 1 1 0
        0 1 1 0 1 1 1 0
      0 0 0 0 0 0 0 0
      ---------------------------------
      0 1 0 1 0 1 1 1 1 0 1 0 1 0 0
```

Opérations arithmétiques

- Multiplication en binaire pur sur 8 bits
  - Résultat sur 2n bits
  - Combinaisons de décalages et d'additions
  - Opération coûteuse en temps de calcul

  - Si l'un des deux nombres contient beaucoup de 0 et peu de 1, l'algorithme de multiplication peut devenir très inefficace.

Opérations arithmétiques

- Multiplication en complément à deux

-5	1 0 1 1
x 7	x 0 1 1 1
- 35	1 0 1 1
	1 0 1 1
	1 0 1 1
	0 0 0 0
	1 0 0 1 1 0 1

Opérations arithmétiques

- Multiplication en complément à deux

  - On obtient 77 !!

  - Conséquence: les circuits multiplieurs traitent différemment les opérations en binaire pur (nombres non signés) et les opérations en complément à deux (nombres signés).

Opérations arithmétiques

- Décalages et rotations
  - Intérêts d'une opération de décalage
    - Multiplication ou division par des puissances de deux.
    - Multiplication rapide pour des nombres comprenant beaucoup de 0 et peu de 1.
  - Deux types de décalage
    - Décalage logique
    - Décalage arithmétique

Opérations arithmétiques

- Décalage logique
  - $(72)_{dec}$ >> 2 = $(01001000)_{bin}$ >> 2 = $(00010010)_{bin}$ = $(18)_{dec}$
  - $(-4)_{dec}$ >> 2 = $(11111100)_{bin}$ >> 2 = $(00111111)_{bin}$ = $(63)_{dec}$

- Décalage arithmétique
  - $(72)_{dec}$ >> 2 = $(01001000)_{bin}$ >> 2 = $(00010010)_{bin}$ = $(18)_{dec}$
  - $(-4)_{dec}$ >> 2 = $(11111100)_{bin}$ >> 2 = $(11111111)_{bin}$ = $(-1)_{dec}$

Opérations de comparaison

- Les quatre bits d'état:
  - C: Carry
  - V: oVerflow
  - Z: Zero
  - N: Negative

- Comment comparer deux nombres A et B ?
  - On calcule la différence A - B
  - On utilise les indicateurs N, Z, C, V
    - $Z = 1$, $A = B$
    - $Z = 0$, $A \neq B$

Opérations de comparaison

- Comparaison de nombres non signés
  - La soustraction génère une retenue lorsque A ≥ B.

C	
0	A < B
1	A ≥ B

C	Z	
0	-	A ≤ B
-	1	A ≤ B
1	0	A > B

Opérations de comparaison

- Comparaison de nombres signés
  - On test le signe du résultat (N) en tenant compte d'un éventuel débordement (V).

N	V	
1	0	$A < B$
0	1	
1	1	$A \geq B$
0	0	

N	V	Z	
1	0	-	$A \leq B$
0	1	-	
-	-	1	
1	1	0	$A > B$
0	0	0	

Opérations logiques

- AND
  - $(00000101)_{bin}$ AND $(00001111)_{bin}$ = $(00000101)_{bin}$
- OR
  - $(00000101)_{bin}$ OR $(00000111)_{bin}$ = $(00000111)_{bin}$
- Exclusive OR
  - $(00000101)_{bin}$ XOR $(00000111)_{bin}$ = $(00000010)_{bin}$
- Bit clear
  - Mise à 0 de bits
  - $(10000101)_{bin}$ BIC $(00000111)_{bin}$ = $(10000000)_{bin}$

and	0	1
0	0	0
1	0	1

or	0	1
0	0	1
1	1	1

xor	0	1
0	0	1
1	1	0

A BIC B	0	1
0	0	0
1	1	0

B