TD N°3

Exercice 1

Quel est le nombre minimum de bit nécessaire pour représenter :

1) 141 000 symboles ?
2) Des nombres entre 0 et 16384 ?

Solution

1) $\lceil log_2 \ 141\ 000 \rceil = 18$
2) $\lceil log_2 \ 16\ 385 \rceil = 15$

Exercice 2

Un microprocesseur a des lignes d'adresses mémoire encodées sur 28 bits. Chaque ligne mémoire contient 8 bits (1 octet). L'espace mémoire total est définit comme l'ensemble de la mémoire adressable.

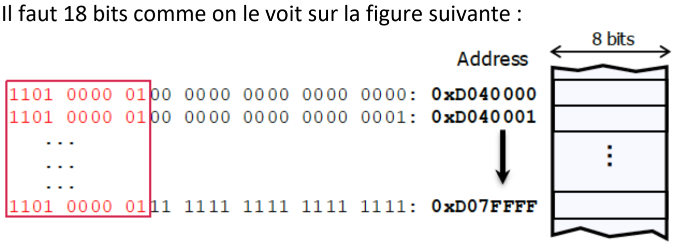
1) Quelle est l'intervalle des adresses mémoires accessible de ce processeur en hexa décimal ? (La plus petite et plus grande adresse)
2) Quelle est la taille (en octet, Ko, Mo) de l'espace mémoire ? (1Ko = 2^{10} octets, 1 Mo = 2^{20} octets, 1 Go = 2^{30} octets)

Un périphérique est connecté au microprocesseur et le processeur lui a attribué les adresses 0XD040000 à 0XD07FFFF de l'espace d'adressage mémoire.

3) Quel est le nombre minimum de bit nécessaire pour représenter les adresses manipulées par ce périphérique ?
4) Quelle est la taille mémoire accessible par ce périphérique ?

Solution

1) $0X0000000$ à 0XFFFFFFF
2) $2^{28} octet = 2^8 2^{20} = 256$Mo d'espace d'adressage
3) Il faut 18 bits comme on le voit sur la figure suivante :

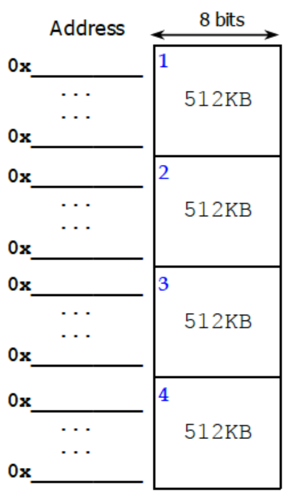


4) 2^{18}=256 Ko

Exercice 3

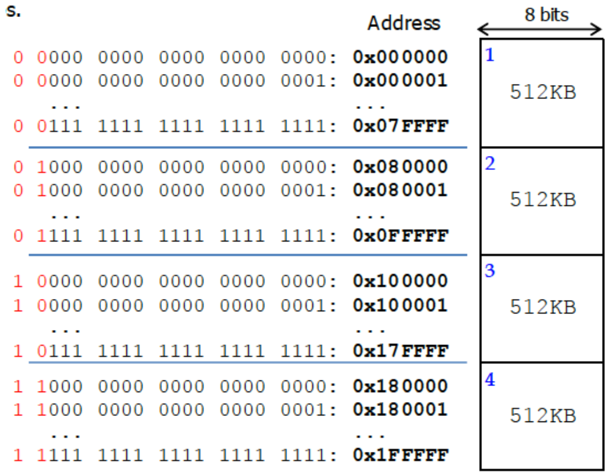
Un processeur a une mémoire de 2Mo et l'unité adressable fait 1 octet.

1) Quelle est la taille des adresses mémoire ?
2) Quelle est l'intervalle des adresses mémoires accessible de ce processeur en hexa décimal ? (La plus petite et plus grande adresse)
3) La figure suivante présente 4 modules mémoires placés l'un à la suite de l'autre dans l'espace d'adressage. Complétez les intervalles pour chaque module (en hexa).



Solution

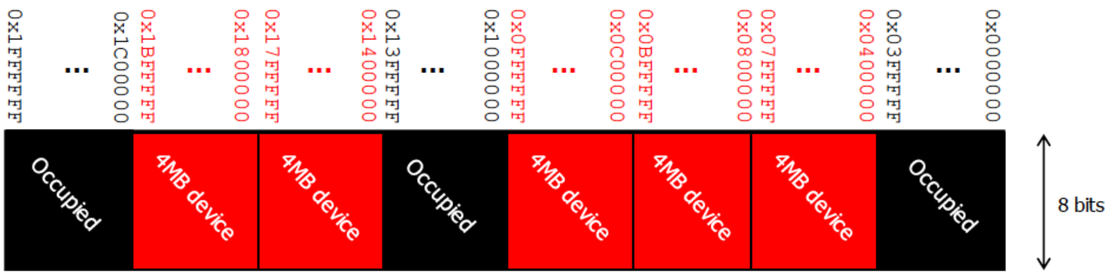
1) 2 Mo = 2^{21} octets donc chaque adresse fait 21 bits.
2) 0X00000 à 0X1FFFF
3)



Exercice 4

La figure suivante décrit la totalité de l'espace mémoire d'un processeur. Sachant que l'unité adressable fait 1 octet.

1) Quel est la taille (en octet, Ko ou Mo) de l'espace mémoire ? Qu'elle est la taille des adresses mémoire ?
2) Si l'on utilise un module mémoire de 4 Mo, combien faudrait-il de bit d'adresse ?
3) On souhaite connecter ce module de 4 Mo au microprocesseur. Dans un souci d'optimisation, on doit positionner ces 4 Mo dans un espace d'adressage de tel sorte que chaque adresse d'un même module ait les bits de poids fort identique (ex : 0X1C00000 à 0X1FFFFFF). Donnez la liste de tous les intervalles d'adresse de 4 Mo que ce module peut occuper.



Solution

1) 0X0000000 à 0X1FFFFFF. Il faut 25 bits pour représenter les adresses mémoires et la taille est donc de 2^{25}=32 Mo.
2) 4 Mo = 2^{22} octets. Il faudrait donc 22 bits d'adresse.
3) 0X0400000 à 0X07FFFFF ;
   0X0800000 à 0X0BFFFFF ;
   0X0C00000 à 0X0FFFFFF ;
   0X1400000 à 0X17FFFFF ;
   0X1800000 à 0X1BFFFFF

Exercice 5

Dans ces exercices vous devez donner le détail des conversions.

1) Convertissez les nombres décimaux suivants en représentation en complément à 2 (binaire et hexadécimal).
   -137.625 ; 92.3125 ; -128.6875 ; -37.65625
2) Complétez la table suivante. Les nombres décimaux sont non signés.

Décimal	BCD	Binaire	Code de Gray
137			
		10101011	
			1101101010
		1011100	
			110001101
	100101010111		

3) Complétez le tableau suivant en utilisant le plus petit nombre de bits dans chaque cas.

Décimal	Signe-magnitude	Complément à 1	Complément à 2
-237			
			1001000
		1011111	
	110101		
		01010001	
-128			

Solution

1)

- 137.625 = 010001001.1010 → -137.625 = 101110110.0110 = 0xF76.6
- 92.3125 = 01011100.0101 = 0x5C.5
- 128.6875 = 010000000.1011 → -128.6875 = 101111111.0101 = 0xF7F.5
- 37.65625 = 0100101.10101 → -37.65625 = 1011010.01011 = 0xDA.58

2)

Decimal	BCD	Binary	Reflective Gray Code
137	000100110111	10001001	11001101
171	000101110001	10101011	11111110
588	010110001000	1001001100	1101101010
92	10010010	1011100	1110010
265	001001100101	100001001	110001101
957	100101010111	1110111101	1001100011

3)

	REPRESENTATION		
Decimal	Sign-and-magnitude	1's complement	2's complement
-237	111101101	100010010	100010011
-56	1111000	1000111	1001000
-32	1100000	1011111	100000
-21	110101	101010	101011
81	01010001	01010001	01010001
-128	110000000	101111111	10000000

Exercice 6

Réalisez les additions et soustractions suivantes des nombres non-signés suivants. Vous utiliserez le plus petit nombre de bits pour représenter les 2 opérateurs, donnerez le détail des retenues (*carry* pour l'addition et *borrow* pour la soustraction) et indiquerez si un dépassement de capacité (*overflow*) s'est produit : 191+201 ; 210+69 ; 130-142 ; 241-36

Example (n=8):

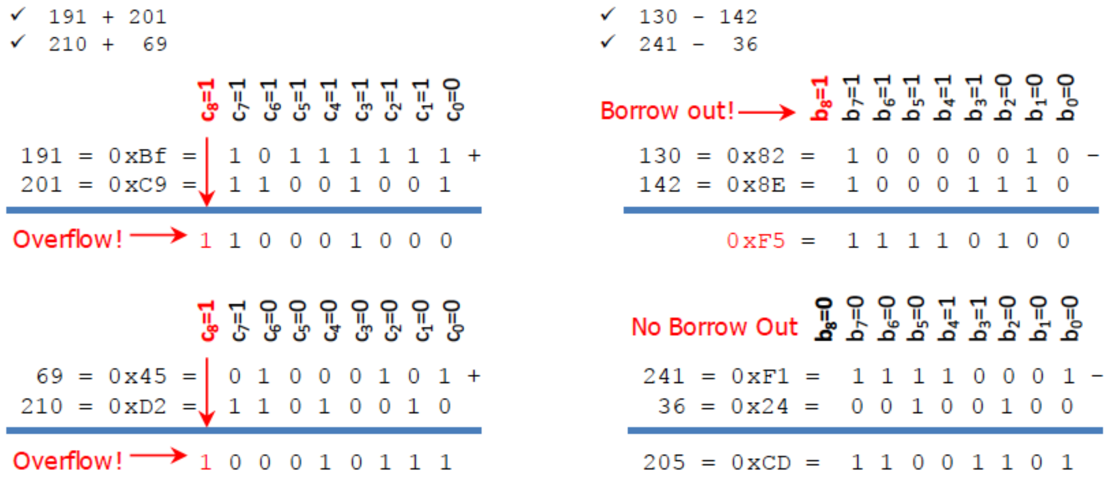
✓ 54 + 210

```
                 c8=1 c7=1 c6=1 c5=1 c4=0 c3=1 c2=1 c1=0 c0=0
 54 = 0x36 =          0    0    1    1    0    1    1    0   +
210 = 0xD2 =          1    1    0    1    0    0    1    0
--------------------------------------------------------------
Overflow! →      1    0    0    0    0    1    0    0    0
```

✓ 77 - 194

```
Borrow out! →    b8=1 b7=0 b6=0 b5=0 b4=0 b3=0 b2=1 b1=0 b0=0
 77 = 0x4D =          0    1    0    0    1    1    0    1   -
194 = 0xC2 =          1    1    0    0    0    0    1    0
--------------------------------------------------------------
                      0    0    0    0    1    0    1    1
```

Solution

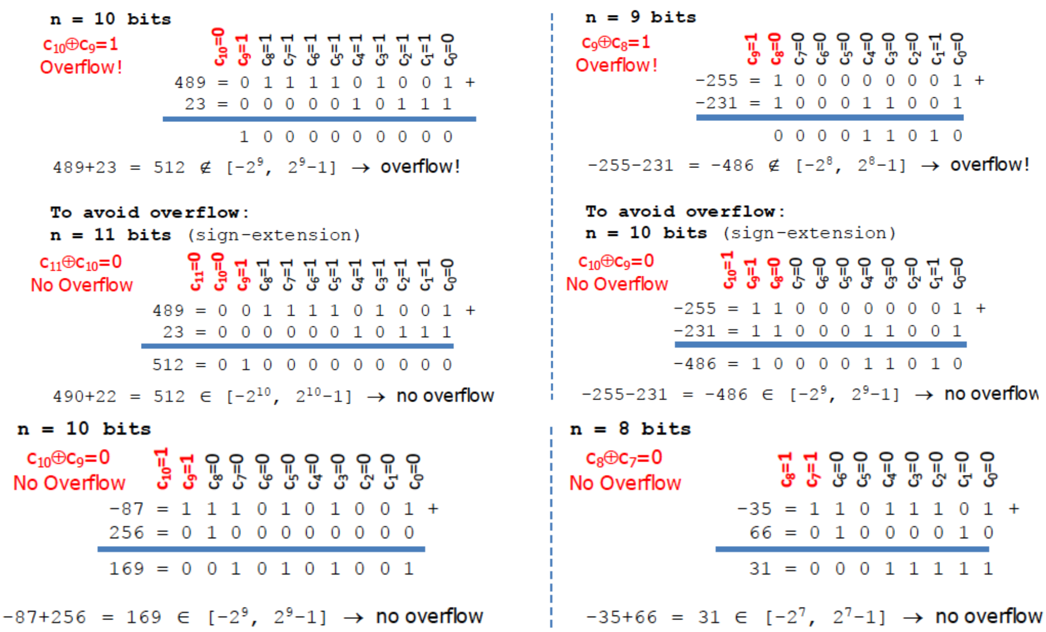


Exercice 7

On souhaite réaliser les opérations suivantes où les nombres sont représenté en complément à 2 :
489+23 ; 256 -87 ; (-129)+126 ; (-255)-231 ; (-35)+66 ; 985+122
Dans chaque cas :

- Déterminez le nb minimum *n* de bit pour représenter les 2 opérandes d'entrées. Attention il vous faudra faire une extension de signe de l'une des opérandes pour que l'opération soit correcte en complément à 2.
- Vous réaliserez impérativement les opérations en complément à 2 et le résultat sera représenté sur le même nombre de bit que les opérandes.
- Vous déterminerez si un overflow s'est produit en utilisant c_n et c_{n-1} (*carries*), en vérifiant votre résultat en décimal et si celui-ci est représentable sur *n* bits. Enfin, dans le cas d'un overflow, vous donnerez le nb minimum de bit pour représenter les 2 opérandes ainsi que le résultat.

Solution



n = 9 bits

$c_9 \oplus c_8 = 0$ No Overflow

```
c9=0 c8=0 c7=1 c6=1 c5=1 c4=1 c3=1 c2=1 c1=0 c0=0
-129 = 1 0 1 1 1 1 1 1 1 +
 126 = 0 0 1 1 1 1 1 1 0
------------------------
  -1 = 1 1 1 1 1 1 1 0 1
```

$-129+126 = -3 \in [-2^8, 2^8-1] \rightarrow$ no overflow

n = 11 bits

$c_{11} \oplus c_{10} = 1$ Overflow!

```
c11=0 c10=1 c9=1 c8=1 c7=1 c6=1 c5=1 c4=1 c3=0 c2=0 c1=0 c0=0
985 = 0 1 1 1 1 0 1 1 0 0 1 +
122 = 0 0 0 0 1 1 1 1 0 1 0
----------------------------
      1 0 0 0 1 0 1 0 0 1 1
```

$985+122 = 1107 \notin [-2^{10}, 2^{10}-1] \rightarrow$ overflow!

To avoid overflow:
n = 12 bits (sign-extension)

$c_{12} \oplus c_{11} = 0$ No Overflow

```
c12=0 c11=0 c10=1 c9=1 c8=1 c7=1 c6=1 c5=1 c4=1 c3=0 c2=0 c1=0 c0=0
 985 = 0 0 1 1 1 1 0 1 1 0 1 0 +
 122 = 0 0 0 0 0 1 1 1 1 0 1 1
------------------------------
1107 = 0 1 0 0 0 1 0 1 0 0 1 1
```

$98+122 = 1107 \in [-2^{11}, 2^{11}-1] \rightarrow$ no overflow

Exercice 8

Trouvez le résultat des multiplications suivantes dont les nombres en entrée sont représentés en complément à 2 sur 4 bits.
0101*0101 ; 1001*0110 ; 1000*1010

Solution

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

```
1 0 0 1 x   ➡   0 1 1 1 x
0 1 1 0         0 1 1 0
                -------
                0 0 0 0
              0 1 1 1
            0 1 1 1
          0 0 0 0
          -------------
        0 0 1 0 1 0 1 0
               ⬇
        1 1 0 1 0 1 1 0
```

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