Initiation à l'Assembleur (6)

par André C. et tous ceux qui voudront bien y participer...

RAPPELS SUR LES OPERATIONS ARITHMETIQUES

L'ADDITION ADC

Addition à l'accumulateur A d'une valeur quelconque M avec la retenue précédente C.

ADC réalise A = A + M + C. Il est donc nécessaire de faire un CLC au préalable (sauf report d'une retenue dans le cas d'une addition de nombres sur plusieurs octets).

Exemple 1: ADC avec C = 0, A = #3A et M = #7C

Hexa :	Binaire :	Décimal :	Observations :
#3A	0011 1010	58	
+#7C	+0111 1100	+124	
+0	+000000000	+ 0	(Carry)
=#B6	=1011 0110	= 182	(pourait être lu comme le nombre signé -74)

Ce résultat étant non nul, Z = 0.

Le b7 est à 1 donc N = 1 (nombre > 127 si non signé ou négatif si signé).

Pas de "b8" donc C = 0 (pas de dépassement si arithmétique non signée, donc #B6 (182) est un résultat correct).

(Passage de retenue b6->b7 = 1) XOR (Carry = 0) donne V = 1 (dépassement si arithmétique signée = résultat -74 faux).

Exemple 2 : Addition sur 2 octets avec report de retenue, soit #ABCD + #1234 : Au départ le Carry sera mis à zéro. Puis on effectue #CD + #34 (addition des octets de poids faible) ce qui donne:

Hexa :	Binaire :	Décimal :	Observation :
#CD	1100 1101	205	(pourait être lu comme le nombre signé -51)
+#34	+0011 0100	+ 52	
+ 0	+0000 0000	+ 0	(Carry)
=#01	=0000 0001	= 1	(si arithmétique signée: -51 +52 = 1)

Z = 0, N = 0, C = 1 et V = 0 car (Passage b6->b7 = 1) XOR (Carry = 1).

Si arithmétique non signée : Il y a dépassement (C = 1). <u>Le résultat est #101 soit 257</u> (qui correspond bien à 205+52).

Si arithmétique signée : Pas de dépassement (V = 0) le résultat est #01 soit 1 (qui correspond bien à -51+52).

On poursuit avec les octets de poids fort en effectuant #AB + #12 + Carry :

Hexa :	Binaire :	Décimal :	Observation :
#AB	1010 1011	171	(pourait être lu comme le nombre signé -85)
+#12	+0001 0010	+ 18	
+ 1	+0000 0001	+ 1	(Carry)
=#BE	1011 1110	=190	(pourait être lu comme le nombre signé -66)

Z = 0, N = 1, C = 0 et V = 0 car (Passage b6->b7 = 0) XOR (Carry = 0).

Si arithmétique non signée : Il n'y a pas dépassement (C = 0) le résultat est #BE soit 190 (qui correspond bien à 171+18+1).

Si arithmétique signée : Pas de dépassement (V = 0) le résultat est #BE soit -66 (qui correspond bien à -85+18+1).

Le résultat final est #BE01 soit (256x190)+1 = 48641 en arithmétique non signée.

En arithmétique signée (un bit de signe + 15 bits significatifs) on a: #BE01 soit 1011 1110 0000 0001 qui est un nombre négatif exprimé en complément à 2 et qui vaut 0100 0001 1111 1111 soit -#41FF ou -16895. On retrouve que #ABCD (-21555) + #1234 (+4660) font bien -16895 ! Autre manière de calculer : Le résultat #BE01 peut se lire -(256x66)+1 = -16896+1 = -16895.

LA SOUSTRACTION SBC

Une valeur quelconque M et la retenue C sont soustraits de l'Accumulateur.

En fait au lieu de soustraire, on ajoute **le complément à 2 qui s'obtient en inversant un à un tous les bits puis en ajoutant 1**. Attention, la signification de Carry est inversée (Carry est en fait le complément à 1 de la retenue). SBC effectue A = A - M - (1 - C). Lorsqu'il n'y a pas de retenue à l'entrée, C = 1 et A = A - M. Lorsqu'il y a une retenue à l'entrée, C = 0 et A = A - M - (1 - 0) soit A = A - (M + 1).

Comme dans le cas de l'addition, SBC permet d'effectuer des opérations sur plusieurs octets avec report de Carry automatique. Il suffit de commencer par les octets de poids faible.

Au départ il faut effectuer un SEC qui en mettant C à 1 efface en fait toute vieille retenue !

Exemple 3 : SBC avec C = 1, A = #F6 et M = #18. On applique A = A - M

```
Hexa :      Binaire :      Décimal :      Observation :
 #18       0001 1000        24           (pour ôter #18, on ajoutera son complément à 2)
           1110 0111                     (inversion des bits)
+   1      0000 0001                     (et on ajoute 1)
---------------------------
=          1110 1000      - 24           (complément à 2 de #18)
+#F6      +1111 0110      +246           (pourait être lu comme le nombre signé -10)
---------------------------
=#DE      =1101 1110      =222           (pourait être lu comme le nombre signé -34)
```

Ce résultat étant non nul, Z = 0.

Le b7 est à 1 donc N = 1.

On obtient un "b8" donc C = 1 c'est à dire pas de retenue = pas de dépassement si arithmétique non signée.

V = 0 car (Passage b6->b7 = 1) XOR (Carry = 1) (pas de dépassement si arithmétique signée).

Résultat : #F6 - #18 = #DE.

Si arithmétique non signée, 246 - 24 = 222.

Si arithmétique signée, -10 - 24 = -34.

Exemple 4: SBC avec C = 1, A = #14 et M = #34. On applique A = A - M :

```
Hexa :      Binaire :      Décimal :      Observation :
 #34       0011 0100        52           (pour ôter #34, on ajoutera son complément à 2)
           1100 1011                     (inversion des bits)
+   1      0000 0001                     (et on ajoute 1)
---------------------------
=          1100 1100      - 52           (complément à 2 de #34)
+#14      +0001 0100      + 20
---------------------------
=#E0      =1110 0000      =224           (pourait être lu comme le nombre signé -32)
```

Ce résultat étant non nul, Z = 0.

Le b7 est à 1 donc N = 1.

Pas de "b8" donc C = 0 (qui signifie en fait une retenue, c'est à dire dépassement si arithmétique non signée = attention résultat faux).

V = 0 car (Passage b6->b7 = 0) XOR (Carry = 0) (pas de dépassement si arithmétique signée).

Résultat : #14 - #34 = #E0.

Si arithmétique non signée, il y a eu dépassement car 20 < 52, il faut interpréter le résultat comme 224 - 256 = -32.

Si arithmétique signée il n'y a pas eu dépassement et on a bien #E0 nombre négatif valant -32.

Exemple 5: Soustraction sur deux octets avec report de retenue, soit #ABCD - #12FF. On commence avec les octets de poids faible. Au départ le Carry sera mis à 1 (rappel : ça signifie pas de retenue). On applique A = A - M soit #CD - #FF ce qui donne:

```
Hexa :      Binaire :     Décimal :    Observation :
 #FF      1111 1111      255         (pourait être lu comme le nombre signé -1)
          0000 0000                  (inversion des bits)
+   1     0000 0001                  (et on ajoute 1)
---------------------------
=         0000 0001     -255         (complément à 2 de #FF)
+#CD      1100 1101      205         (pourait être lu comme le nombre signé -51)
---------------------------
=#CE     =1100 1110     =206         (pourait être lu comme le nombre signé -50)
```

Avec Z = 0, N = 1, C = 0 (il y a une retenue) et V = 0 car (Passage b6->b7 = 0) XOR (Carry = 0).

Si arithmétique non signée : Il y a dépassement (C = 0) le résultat réel serait #CE soit 206 - 256 = -50.

Si arithmétique signée : Pas de dépassement (V = 0) le résultat est bien #CE (-50).

On poursuit avec les octets de poids fort en effectuant A = A -(M + 1), soit #AB - (#12 + 1) ou #AB - #13, car maintenant il y a une retenue.

```
Hexa :        Binaire :     Décimal :    Observation :
 #13       0001 0011     + 19         (pour ôter #13, on ajoutera son complément à 2)
           1110 1100                  (inversion des bits)
+   1      0000 0001                  (et on ajoute 1)
----------------------------
=          1110 1101     - 19         (complément à 2 de #13)
+#AB       1010 1011     +171         (pourait être lu comme le nombre signé -85)
----------------------------
=#98       1001 1000     =152         (ou -104 si nombre signé)
```

Avec Z = 0, N = 1, C = 1 (pas de retenue) et V = 0 car (Passage b6->b7 = 1) XOR (Carry = 1).

Si arithmétique non signée : Il n'y a pas dépassement (C = 1) le résultat est #98 soit 152 (qui correspond bien à 171-19).

Si arithmétique signée : Pas de dépassement (V = 0) le résultat est #98 soit -104 (qui correspond bien à -85-19).

Le résultat final est #ABCD - #12FF = #98CE avec Z = 0, N = 1, C = 1 et V = 0 (ni retenue ni dépassement).

En arithmétique non signée 43981 (#ABCD) - 4863 (#12FF) = 39118 (#98CE). On peut vérifier que #98CE = (256x#98) + #CE = (256x152) + 206 = 38912 + 206 = 39118.

En arithmétique signée (un bit de signe + 15 bits significatifs) le résultat vaut #98CE soit 1001 1000 1100 1110 qui est un nombre négatif exprimé en complément à deux et qui vaut 0110 0111 0011 0010 soit -#6732 ou -26418. On retrouve que #ABCD (-21555) - #12FF (+4863) font bien -26418 !

à suivre...