2

Le microprocesseur et son architecture

L'unité Centrale (U.C.) se compose en partie du microprocesseur ou C.P.U. (Central Processing Unit), générant des impulsions au rythme de son horloge, et de cases-mémoire. Les mémoires peuvent être mortes (R.O.M. ou MEM) ou vives (R.A.M. ou MEV), comme nous l'avons déjà vu.

L'ordinateur communique avec la périphérie (clavier, écran, L.E.P., etc.), et on nomme périphérique tout ce qui est relié au système central.

Le processeur est en relation constante avec les mémoires de l'ordinateur (tableau numéro 1).

Le microprocesseur peut aller chercher ou écrire des données dans une case-mémoire (R.A.M. pour les écriture et lecture, R.O.M. pour la lecture seule). Pour ce faire, il s'adresse à cette case-mémoire par un bus de 16 bits (tableau numéro 1). En effet, rappelons qu'une adresse de case-mémoire est codée sur deux octets (16 bits), et que cette case contient 8 bits.

Soit l'adresse &HE800* sur TO7(70) et TO9 : cette case-mémoire ne peut être que lue, car se situe dans la R.O.M. (tableau numéro 2). Lorsqu'en basic vous faites : ? PEEK (&HE800), vous obtenez la valeur de son contenu en décimal : 126. Cet octet (126) transite par le bus de données, bus de 8 bits.

*&HE800 : 1re adresse du moniteur-système sur TO7(70) et TO9. Sur MO5 : &HF000.

Retenez qu'en R.A.M. vous avez la mémoire-écran, des parties libres, quelques adresses de P.I.A.*, les pages 0 et la mémoire-utilisateur. Le reste, c'est de la R.O.M..

Votre microprocesseur est un 6809 de la firme Motorola, appelé faux 16 bits car intermédiaire entre les processeurs 8 bits et les 16 bits de la génération suivante. Il est assez performant.

Ce 6809 comporte des cases-mémoire intégrées, alors que le reste de la mémoire de l'ordinateur lui est externe. Ces cases-mémoire du processeur, un peu particulières, travaillant sans relâche, sont appelées registres du processeur. On y trouve :

1) Les accumulateurs A et B (8 bits chacun), concaténables en un accumulateur de 16 bits.
2) Les registres d'index X et Y (16 bits chacun).
3) Les registres-pointeur de pile, S et U (16 bits chacun).
4) Le registre d'état ou de Code-condition CC (8 bits).
5) Le compteur de programme PC (16 bits).
6) Le registre de page directe DP (8 bits).

1) Les accumulateurs A et B, de 8 bits chacun, permettent le transfert de données, des opérations arithmétiques, logiques, comparatives.

Ils peuvent être associés, pour former UN SEUL accumulateur de 16 bits qui prend alors le nom de D. Cette possibilité de concaténation vaut au processeur l'appellation de : faux 16 bits. ''A'' représente dans ce cas les 8 bits de poids fort, et ''B'' les 8 bits les moins significatifs ou de poids faible.

L'accumulateur B sert plus particulièrement lors de l'accès à certaines routines de la R.O.M..

2) Les registres d'index X et Y, de 16 bits chacun, sont aussi appelés registres d'adresses. Leur fonction principale est en effet de représenter une adresse.

3) Les registres S et U contiennent chacun l'adresse d'une pile. Une pile est une portion de R.A.M., permettant la sauvegarde de données. Il existe une pile S, appartenant au système mais pouvant éventuellement servir au programmeur, et une pile U, réservée à la programmation. Toute donnée peut être poussée dans une pile par un registre du processeur, et retirée ensuite par lui-même, ou par un autre. On dit alors que l'on empile, ou que l'on dépile, ce registre. Les registres-

*P.I.A. = système permettant les entrées et les sorties (tableau n° 13).

pointeur S et U ne peuvent cependant pas être poussés dans la pile qu'ils définissent respectivement. Ces piles sont de type LIFO (LAST IN, FIRST OUT), c'est-à-dire qu'elles font sortir en premier la donnée entrée la dernière dans la pile. Les deux pointeurs de pile peuvent aussi fonctionner comme registres d'index.

4) Le registre CC est essentiel et un peu plus long à décrire ; nous resterons cependant simple dans sa présentation. C'est dans ce registre que se trouvent les indicateurs dont il a été question précédemment. Voici le schéma de ce registre de 8 bits :

E	F	H	I	N	Z	V	C

"C" : retenue ou CARRY. Il s'agit d'un FLAG (drapeau) ou indicateur qui peut être à l'état 0 ou 1. Ceci a déjà été vu. En testant cet indicateur après une quelconque opération, et selon son état, on peut se brancher ou non à un sous-programme, par exemple.

"V" : OVERFLOW. Ce drapeau indique le dépassement de capacité d'un registre pour une opération en arithmétique signée.

"Z" : ZERO FLAG. Il se positionne à 1 lors d'une manipulation de donnée égale à zéro, ou pour un résultat nul.

"N" : NEGATIVE FLAG. Il prend la valeur 1 lors d'une opération manipulant ou produisant des nombres négatifs. Il ne sert donc qu'aux nombres signés.

"I" : masque d'interruption IRQ. Se reporter au chapitre concernant les interruptions.

"H" : HALF-CARRY ou indicateur de « demi-retenue », déjà cité. Il se positionne à 1 lorsqu'il y a retenue d'un bit numéro 3 vers un bit numéro 4.

"F" : masque d'interruption FIRQ. Voir le chapitre sur les interruptions.

"E" : ENTIRE FLAG. Il est mis à 1 pour signaler la sauvegarde intégrale des registres du processeur, lors d'une interruption de type SWI, IRQ ou NMI.

Ainsi, si "E", "N" et "V" sont positionnés à 1, "CC" vaudra en hexadécimal : 8A. En tapant R et ENTRÉE dans le moniteur-binaire vous aurez, après exécution d'un programme, la valeur des registres du processeur, dont "CC".

Le tableau numéro 8 donne la signification des mnémoniques et leur action sur le registre d'état, ou de code-condition, CC.

5) Le compteur-programme PC : il s'agit du pointeur indiquant au processeur l'instruction à lire. Après cette lecture, le ''PC'' se trouve en avance d'un octet sur l'opération que le processeur est en train d'exécuter. Lors d'un break, il pointe sur l'adresse de celui-ci.

Ce registre peut fonctionner comme un registre d'index (= registre d'adresse), mais attention l'index est ici fonction du déroulement du programme ; il permet néanmoins dans ce cas la réalisation de programmes translatables (nous étudierons cela à la fin de l'ouvrage).

D'une manière plus simple, disons qu'un programme qui se réfère au ''PC'' peut s'implanter n'importe où, car l'adresse contenue dans le ''PC'' varie avec l'implantation du programme, alors qu'une programmation avec des adresses fixes est rigide.

Exemple :

Un programme en 32000 (''PC''), ayant une valeur à charger en 32020, ne fonctionnera pas en 40000 (''PC'').

Par contre, un programme en 32000 (''PC''), ayant une valeur à charger en ''PC'' + 20, pourra fonctionner en 40000 (''PC''). Le chargement se fera en effet, dans ce cas, en 40000 (''PC'') + 20 = 40020.

6) Le registre de page directe DP : Nous avons vu qu'une page comprenait 256 octets. Imaginons que nous travaillions en page $7D, et que notre programme débute en $7D00 sans excéder 256 octets, c'est-à-dire sans dépasser $7DFF. Plutôt que de coder les adresses sur 2 octets, nous pourrions signaler au registre DP que nous travaillons en page $7D : un stockage en $7D05, par exemple, deviendrait alors stockage en ''<05''. Le processeur comprendrait, quant à lui, que ce stockage doit se faire en : ''DP'' = $7D + ''<05'' = $7D05. Ceci nous ferait gagner un octet par adresse et donc de la place en mémoire, ainsi que du temps en cycles-machine (la rapidité d'un programme étant fonction du nombre de cycles-machine qu'il doit exécuter).

Quelques remarques :

a) Le signe : ''<'' est plus restrictif que ''>''.

''>'' signifie adressage étendu à toute la mémoire.

''<'' signifie adressage direct limité à la page définie par ''DP''. Pour sortir éventuellement de la page, on peut utiliser le signe ''>'' (on dit alors que l'on force l'adressage étendu).

b) A noter que l'on écrit ''$'' au début d'un nombre pour signifier : nombre hexadécimal. Il est possible d'utiliser le suffixe H également, mais cela peut créer des confusions.

$7D00 = 7D00H

De même, pour les nombres décimaux : &32000 = 32000T ou 32000.

Nos « petits » assembleurs ne travaillent pas directement en binaire et la représentation binaire ''%'', utilisée par d'autres assembleurs, est interdite. Nous ferons donc nous-mêmes les conversions du binaire à l'hexadécimal ou au décimal.

c) A propos du temps-machine, il faut savoir que le 6809, 8 bits, travaille à la fréquence de 1 MHZ (mégahertz). Ceci signifie qu'il peut traiter un million d'informations, c'est-à-dire faire un million de cycles-machine, à la seconde.

Une instruction demande de un à cinq octets, et peut faire de deux à parfois vingt cycles-machine. A nous de choisir les instructions les plus performantes. Il est par exemple plus judicieux de travailler avec le registre X que le registre Y, quand on le peut, car les instructions demandent un cycle-machine de moins avec ''X''.

Exemple :

L'instruction LDA #$24 requiert deux octets et deux cycles-machine pour être exécutée (un octet pour LDA en mode immédiat (#) et un octet pour $24).

Le tableau numéro 8 donne la signification des mnémoniques et le nombre d'octets, ainsi que le nombre de cycles-machine, nécessaires pour chacun d'eux.