PRÉSENTATION DE L'ORDINATEUR DANS LES COLLÈGES PROJET DE COURS À L'INTENTION DES DÉBUTANTS EN INFORMATIQUE

Jean-Marie GADAT

INTRODUCTION

Les programmes de technologie des collèges (6e, 5e, 4e, 3e) et les compléments précisent les points suivants (BO de juillet 87) :

INFORMATIQUE INDUSTRIELLE ET AUTOMATIQUE...

... Les caractéristiques et le fonctionnement de l'ordinateur au sein du système informatique, dont L'ETUDE FONCTIONNELLE devra obligatoirement être faite.....

... En ce qui concerne l'analyse fonctionnelle du système, on insistera sur les points suivants :

- Le partage des tâches entre les différents constituants physiques du système : unité centrale, clavier, écran...etc.
- L'importance des liaisons et des interfaces ;
- Le cheminement des informations ;
- Le dialogue avec l'utilisateur à travers les différents niveaux de langage, qui font du micro-ordinateur une MACHINE PROGRAMMABLE, en insistant sur la notion de code ;
- La nécessité d'un minimum d'informations, présentes en permanence dans la machine, pour assurer ses fonctions de mise en service ; la notion de langage résident ; le rôle du traducteur (compilateur ou interpréteur) ;
- le rôle du système d'exploitation ;
- la nécessité de sauvegarder les programmes et données introduites dans l'ordinateur ;

...

Or, les ouvrages qui traitent de ces questions, sont, soit des livres de haut niveau, donc inaccessibles aux débutants, soit des livres qui présentent l'ordinateur vu de l'extérieur, morcelé en ses constituants physiques comme l'unité centrale, le clavier, les mémoires mortes, les mémoires vives, etc.. et qui ayant décrit le système, montrent ensuite ce que l'on peut faire avec lui. Il en résulte des schémas architecturaux généralement assez complexes qui masquent en fait la relative simplicité de principe et de fonctionnement d'un micro-ordinateur.

Il me semble que - au moins au niveau d'une classe de premier cycle ou d'une classe de seconde - il serait préférable d'adopter la démarche inverse et de présenter l'ordinateur comme un outil qui répond à un besoin, qui résout un problème, que l'on pourrait définir comme l'automatisation de processus répétitifs.

PRÉSENTATION DE L'ORDINATEUR DANS LE PREMIER CYCLE

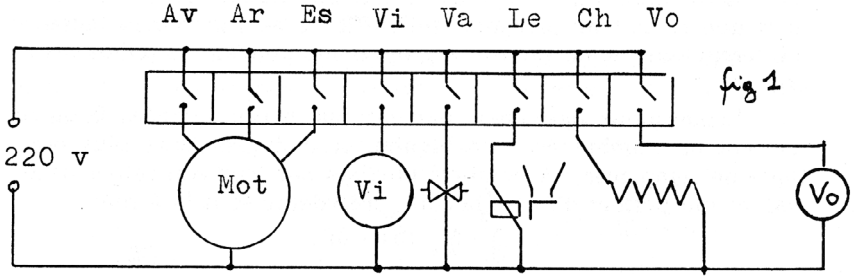
Les textes (* *) sont des commentaires et ne font pas partie de l'exposé...

1 - Un problème de machine à laver

Un constructeur a bâti une machine à laver le linge à faible coût sur le modèle suivant :

La machine comporte : un moteur qui a trois modes de marche, avant (Av), arrière (Ar), rapide pour essorage (Es), un moteur de vidange (Vi), une vanne de remplissage (Va), Un clapet pour produits lessiviels (Le), un circuit de chauffage (Ch), et un voyant de mise sous tension (Vo).

Chaque élément (actionneur) est commandé par un interrupteur à deux positions marche et arrêt qui permet la mise sous tension de l'élément. Les commandes sont regroupées sur une tablette. L'ensemble est représenté figure 1.



La machine est conçue pour être utilisée par un opérateur sans compétence en matière de lavage, la suite des opérations à effectuer se limitera pour lui à la fermeture ou l'ouverture des interrupteurs dans un ordre indiqué sur une feuille annexe, que nous appellerons "feuille programme". L'exemple ci-dessous indique une suite d'instructions que pourrait contenir cette feuille programme :

Remplissage : fermer 20 secondes Va,Vo

Lessive : fermer 5 secondes Le,Vo

Lavage : fermer 15 secondes Av,Ch,Vo
: fermer 15 secondes Ar,Ch,Vo
: etc.

(* Ce fonctionnement "en boucle ouverte", ne prend pas en compte les capteurs comme ceux de température ou niveau, mais la simulation doit rester très élémentaire *)

Pour uniformiser la rédaction, et simplifier la lecture par l'opérateur, on conviendra que chaque interrupteur, suivant qu'il est ouvert ou fermé sera indiqué sur la feuille programme par un code très simple : 0 s'il est ouvert et 1 s'il est fermé, ce qui donne pour les instructions précédentes :

20 s 0 0 0 0 1 0 0 1
5 s 0 0 0 0 0 1 0 1
15 s 1 0 0 0 0 0 1 1
15 s 0 1 0 0 0 0 1 1 etc.

Remarquons que l'opérateur n'a pas besoin de comprendre quoi que ce soit au lavage, il lui suffit de positionner la tablette d'interrupteurs selon sa feuille programme pendant le temps prescrit.

Remarquons également que les instructions codées 0 ou 1 forment un nombre écrit en binaire, et qu'il peut être plus pratique de remplacer ces nombres binaires par leur équivalent décimal, ce qui permet d'écrire la suite précédente sous la forme :

- 9 - 5 - 131 - 67...

2 - Automatisation du procédé

Notre opérateur ayant remarqué lui aussi que son travail ne réclamait aucune réflexion ou choix, imagine le dispositif suivant, où 8 contacts à lame souple remplacent les interrupteurs sur la tablette :

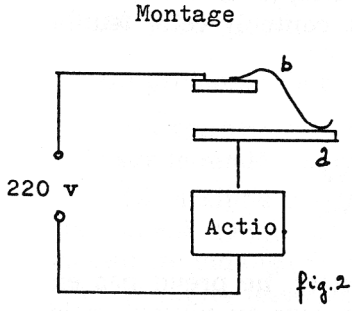


fig.2

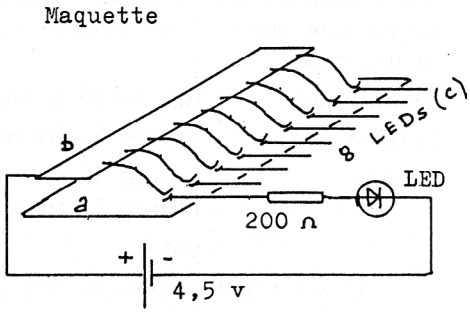


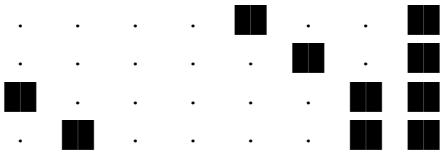
fig.3

(* Une maquette est réalisée en circuit imprimé, les contacts étant par exemple récupérés sur des piles de 4,5 volts. On peut quelquefois se procurer des éléments de lecteurs de cartes perforées sur des anciennes machines comptables.

Des diodes électroluminescentes (LED) figurent l'état des actionneurs. Elles sont sur un module séparé *)

Il suffit maintenant pour mettre à 0 l'interrupteur correspondant à l'instruction en cours d'exécution, de glisser une feuille isolante entre la lame flexible et le contact du bas, la mise à 1 se faisant par une perforation dans l'isolant.

Les instructions n'ont plus besoin d'être écrites, il suffit que la "fenêtre" corresponde aux codes à 1 directement dans la feuille programme, comme l'indique la figure ci-dessous :



Il faudra éventuellement utiliser un relais si le courant demandé par l'actionneur est trop important.

L'opérateur se borne maintenant à faire avancer le programme d'un pas en fonction du temps réclamé par l'instruction. Il peut d'ailleurs aller plus loin dans l'automatisation s'il a pris la peine d'écrire le programme pour que chaque instruction ait une durée constante (que l'on pourra appeler "cycle machine"), en recopiant éventuellement plusieurs fois les lignes qui durent plusieurs cycles. Le programme modifié devient, avec un cycle machine de 5 secondes :

- 9 - 9 - 9 - 9 - 5 - 131 - 131 - 131 - 67 - 67 - 67 -....

Et enfin, l'opérateur installe une horloge qui, toutes les 5 secondes fait avancer le programme d'une ligne.

L'automatisation de la machine est maintenant complète, le changement de programme se faisant par changement de la "feuille programme". Il faut évidemment que tous les programmes utilisés admettent la même horloge.

(* les ordinateurs ont fonctionné effectivement sur ce modèle - sans remonter au métier à tisser de Jacquard - ils sont à ma connaissance à peu prés abandonnés aujourd'hui *)

3 - Réalisation : micro-ordinateur

Les trois éléments que nous venons de décrire sont l'image d'un micro-ordinateur :

La partie maquette, qui interprète les instructions est appelée "microprocesseur", celui-ci "lit" les codes qui lui sont présentés et les exécute.

La partie qui contient les codes ou instructions à exécuter, notre "feuille programme" est appelée "mémoire". Comme dans l'exemple, les instructions y sont inscrites sous forme de lignes de nombres 0 ou 1. Chaque nombre 0 ou 1 étant appelé un BIT (BInary digiT)

La partie qui permet l'avancement du programme est aussi appelée "horloge", mais à la différence de notre horloge de période 5 secondes, elle génère des impulsions beaucoup plus rapides : quelques millions d'impulsions par seconde ;

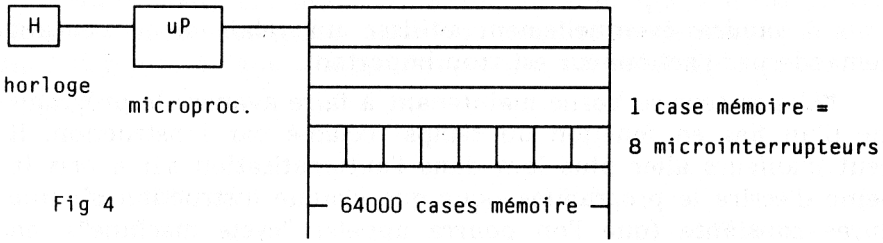


Fig 4

Nous voyons donc, que pour faire fonctionner un micro-ordinateur, il nous faut écrire dans la mémoire une suite de codes qui seront lus et exécutés par le microprocesseur.

Comme certaines de ces instructions reviennent tout le temps, elles ont été écrites par le constructeur qui les fournit avec la machine : on les appelle "programme résident" elles se trouvent dans une partie de la mémoire appelée "mémoire morte", car on ne peut plus les modifier, et les

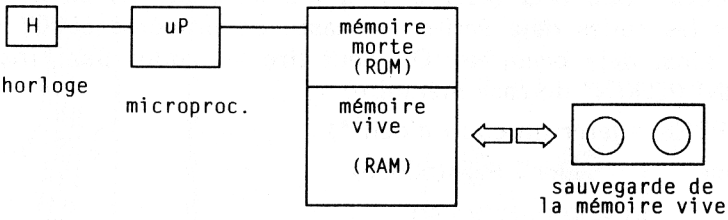
instructions écrites par l'utilisateur se trouvent dans l'autre partie de la mémoire appelée "mémoire vive". En reprenant notre exemple, tout se passe comme si l'utilisateur possédait trois types de "feuilles programmes" à sa disposition :

- Un type vendu avec la machine, ce sont le ou les programmes résidents (mémoire morte). Le constructeur indique sur la notice comment utiliser les instructions du (ou des) programme(s) résident(s).

- Un type vierge : la mémoire vive qui pourra, une fois écrite par l'utilisateur, être modifiée si besoin est (puis être effacée).

- Les feuilles programmes écrites par l'utilisateur (ou par un fabricant de programmes) et qu'il a voulu conserver pour pouvoir les réutiliser, ils sont rangés à part, dans une bibliothèque de programmes que l'on appelle disquettes ou cassettes.

Le fonctionnement est représenté ci-dessous :



Quelque soit l'ordinateur, on retrouve toujours ces éléments, et à eux seuls ils suffisent à faire fonctionner un système informatique. Evidemment une machine qui ne posséderait que cela ne serait pas très pratique à utiliser. Aussi, lui ajoute-t-on d'autres organes appelés "périphériques" qui permettent la liaison avec l'extérieur, et le dialogue avec l'utilisateur.

(* On peut expliquer à partir de là, mais dans une leçon ultérieure, que l'ensemble des instructions cohérentes pour utiliser une machine s'appelle un langage et indiquer la différence entre langage binaire, suite d'instructions numériques sans signification évidentes, et langage évolué, suite d'instructions (presque) compréhensibles par l'usager, mais qui ont besoin de toute façon d'être traduites en langage binaire *)

(* On peut ensuite introduire les notions habituelles de périphériques : clavier, écran, imprimante, etc.. niveaux de langages, compilateur, système d'exploitation, en montrant que finalement la partie qu'il convient de bien gérer - quand on veut comprendre le fonctionnement d'un micro - est la mémoire *).

(* D'autres scénarios sont évidemment utilisables : il suffit que les opérations puissent être gérées par une horloge *)

4 - Travaux pratiques

On peut montrer cette architecture minimum sur n'importe quel micro-ordinateur : sur un MO5 par exemple :

Le clignotement du curseur est obtenu à partir de l'horloge par division de la fréquence (facteur de division de l'ordre de 100000).

Les cases mémoire sont au nombre de 64000 environ, elles portent un numéro (de 0 à 65535 exactement), le constructeur indique quelles cases il a réservé à la mémoire morte, où sont donc rangés les codes déjà écrits. les cases de numéros 50000 à 65000 sont ainsi déjà occupées. On peut lire les codes par l'instruction `PRINT PEEK`(N° de case mémoire) :

(* to PEEK = jeter un coup d'oeil *)

Frapper
```
PRINT PEEK(50000)
```

ou
```
10 FOR I = 1 TO 50
20 PRINT PEEK(50000+I)
30 NEXT I
```

On voit défiler à l'écran les codes d'un programme résident.

Par contre si on fait les mêmes opérations en remplaçant 50000 par 13000, on trouve des cases vides.

(* on peut trouver 0 ou 255 - ce qui correspond à 8 "1"- parce qu'à l'initialisation les cases de RAM se remplissent de 0 ou de 1, cela ne change rien au fait qu'une case ne contient aucune information préalable utilisable *)

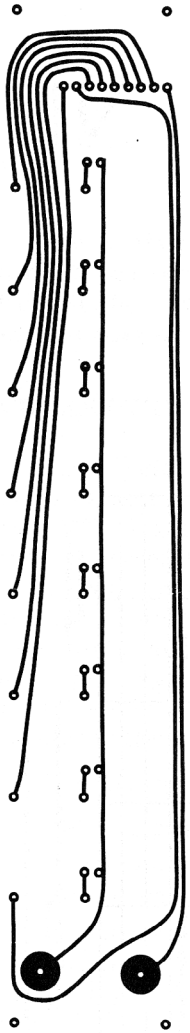
On peut écrire dans une case par l'instruction :

`POKE` N° de case ,valeur

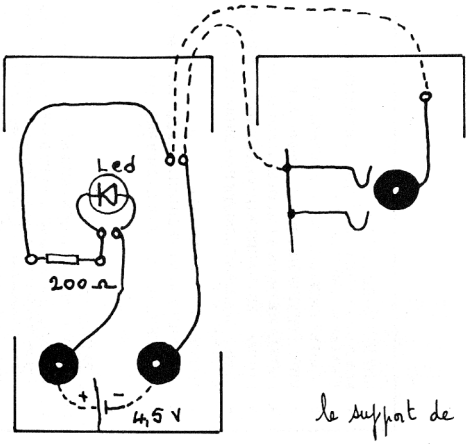
(* to POKE = pointer, montrer du doigt *)

On vérifie que l'écriture dans la case 50000 par exemple est impossible : l'instruction `POKE 50000,32` est sans effet, c'est à dire que la lecture `PRINT PEEK(50000)` donne toujours le même résultat. alors que la même opération avec la case 13000 permet bien de remplir la case 13000 avec le code 32.

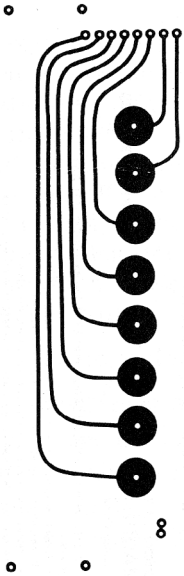
(*La carte des zones mémoire vives ou mortes est généralement donnée par le constructeur *)



support des LEDs
(figure 3 c)



le support de Leds et le "lecteur" sont reliés par une nappe de 9 fils



support du "lecteur"
(figure 2 a)

J.M. GADAT
Professeur de physique appliquée
Lycée Marie Curie - CREIL