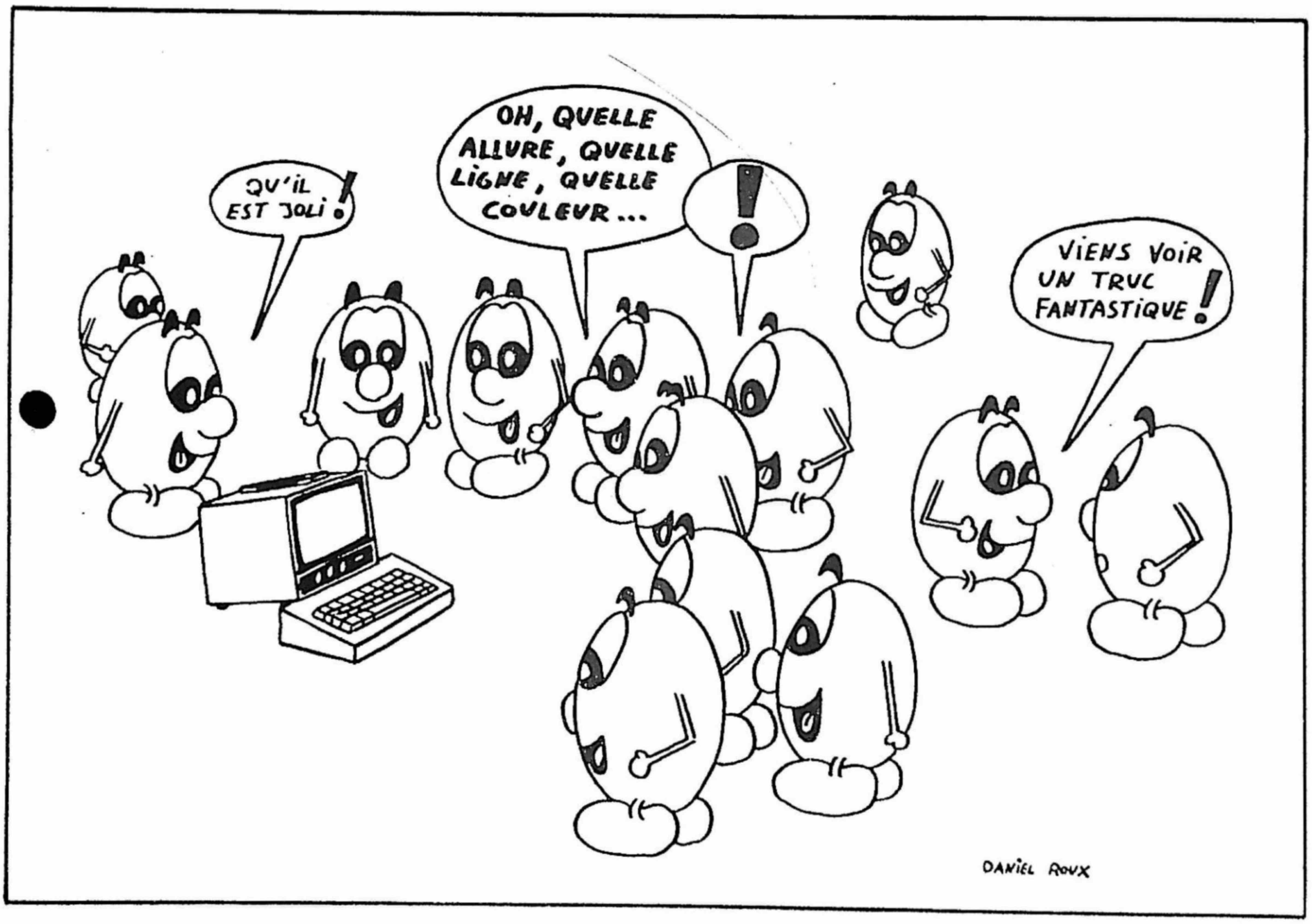
LA PROGRAMMATION DU SMAKY

I

Daniel Roux

Février 1979

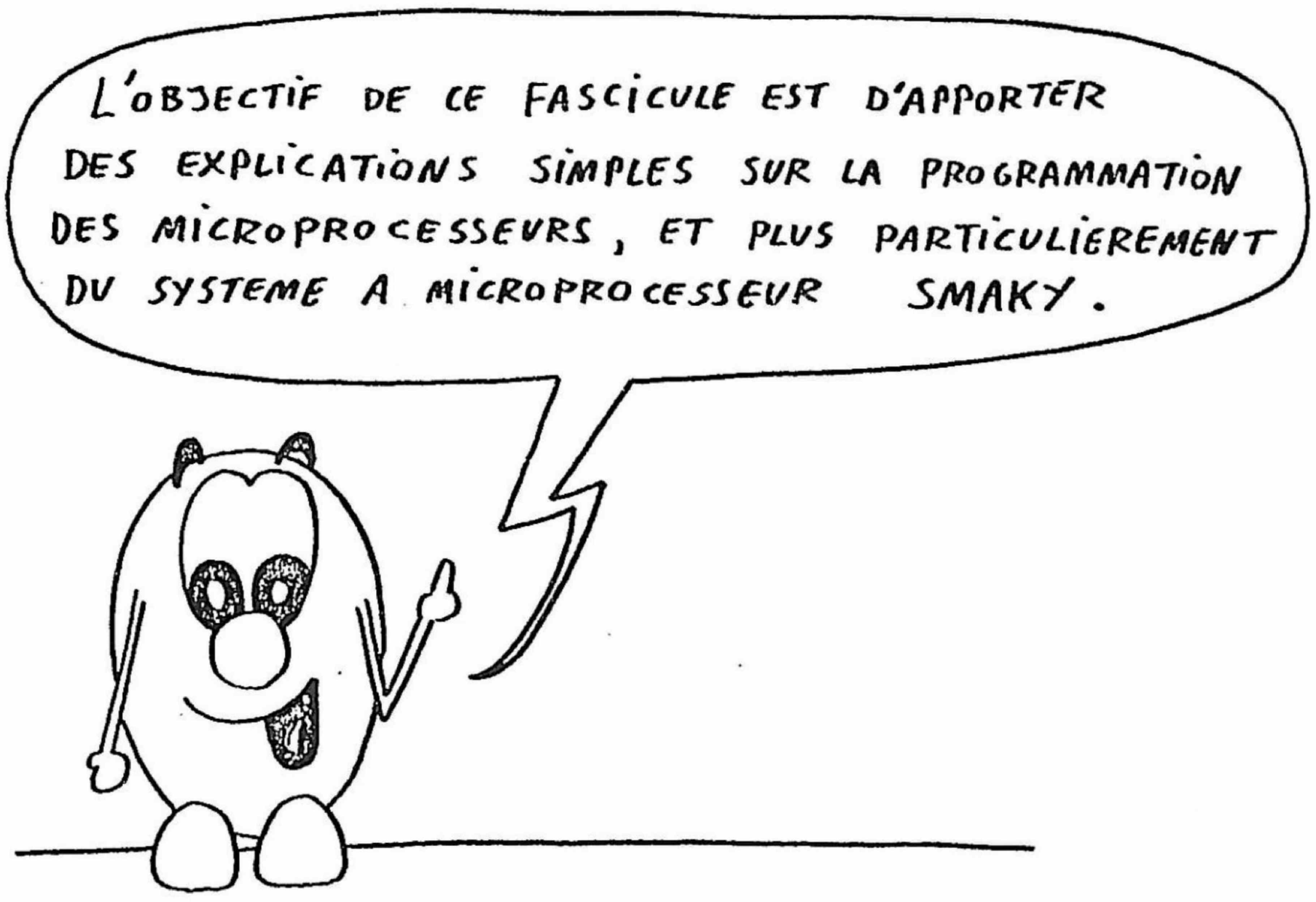


epsitec-system sa

LA PROGRAMMATION DU SMAKY6

lère partie

1. INTRODUCTION



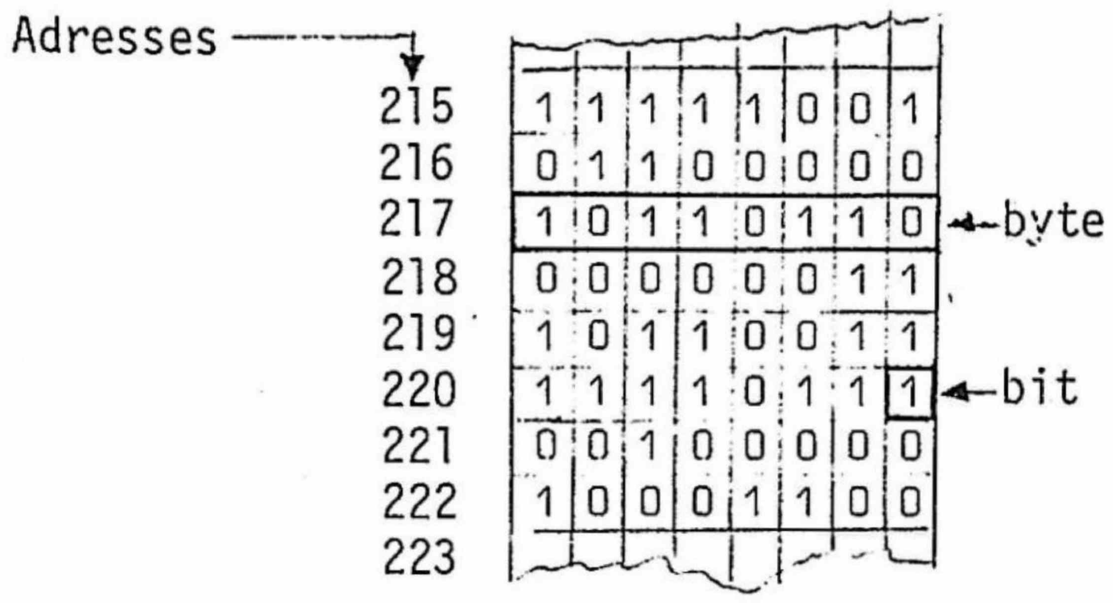
Nous nous efforcerons d'utiliser les termes techniques français, en mentionnant leurs équivalents anglais entre deux barres obliques.

2. SYSTEME MICROPROCESSEUR

Il faut bien distinguer un microprocesseur et un système à microprocesseur. Le microprocesseur se réduit généralement à une unité arithmétique et de contrôle; sa structure ne sera pas étudiée dans ce fascicule. Un système à microprocesseur comporte, en plus du processeur, la mémoire et des interfaces d'entrée/sortie, selon l'application visée.

2.1 MEMOIRE

La mémoire est un ensemble de cases numérotées, Le numéro est appelé adresse */address/* et le contenu est appelé mot, donnée,information, byte, code */data/*. Dans le SMAKY, le contenu est un mot binaire de 8 bits. C'est-à-dire qu'il possède 8 sous-cases pouvant contenir chacune un Ø ou un 1.



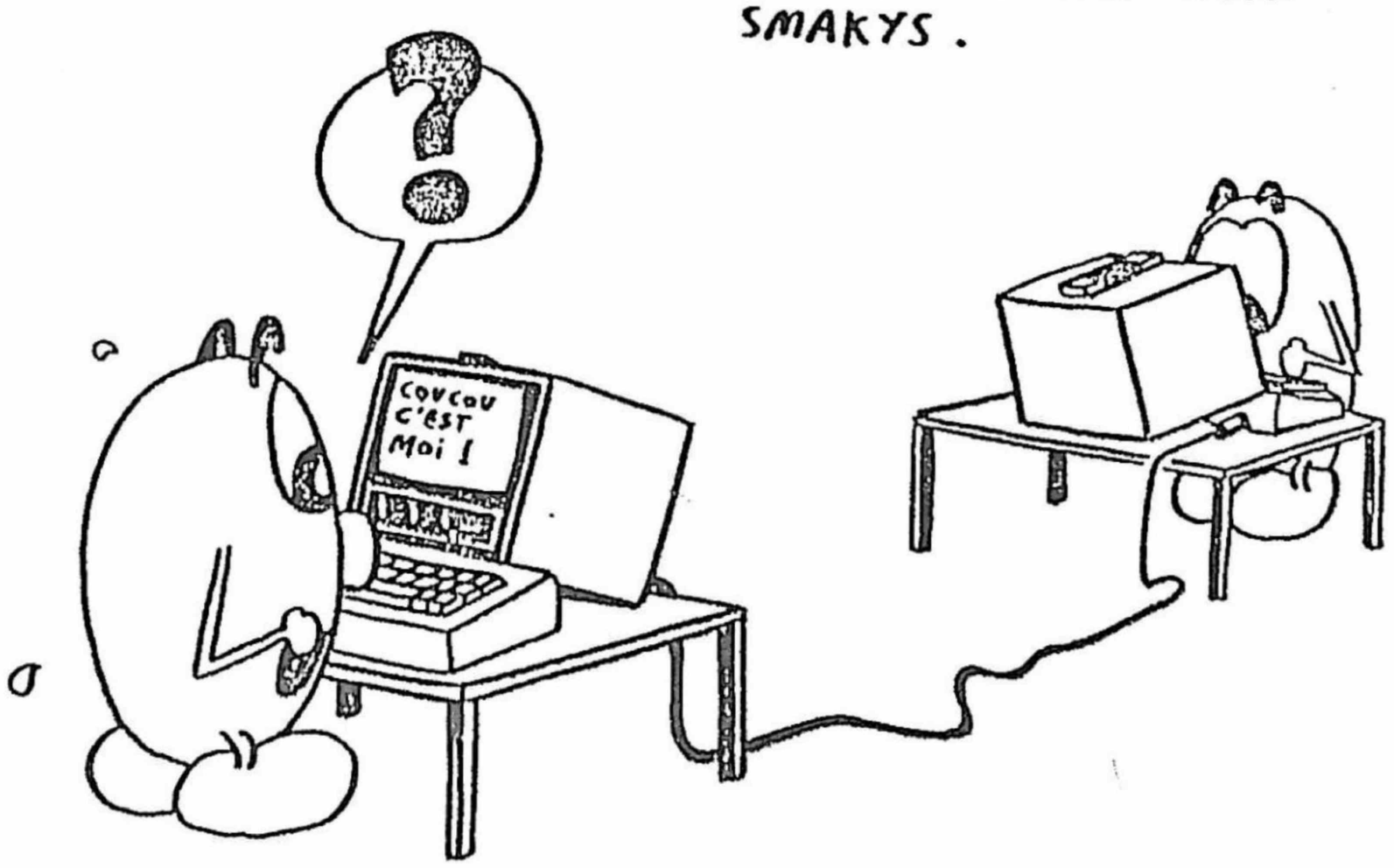
La mémoire contient le programme et les données; elle peut être à lecture et écriture (mémoire vive) */RAM: Random Access Memory/* ou à lecture seulement (mémoire morte) */ROM: Read Only Memory/*.

Le contenu d'une mémoire vive peut donc être constamment modifié. On l'utilisera, par exemple, pour stocker des données. Lorsque l'alimentation est coupée, tout le contenu est perdu.

Le contenu d'une mémoire morte ne peut être que lu. Il a quand même bien fallu l'écrire une fois, mais ceci se fait sur un appareil spécial appelé programmateur. Lorsque l'alimentation est coupée, le contenu est conservé.

2.2 INTERFACES

Les interfaces */interface/* permettent la commande des périphériques de dialogue avec l'utilisateur (périphériques clavier et écran par exemple) ou avec un autre système à microprocesseur (périphériques de transmission) ou encore avec une mémoire de masse (cassette, unité de disques, etc.).



A un périphérique correspondent une ou plusieurs adresses. Une seule adresse suffit pour lire un clavier: la donnée correspondante permet de savoir quelle touche est pressée.

L'écran du SMAKY6 est particulier. En effet, pour le processeur, il réagit comme une mémoire vive normale, mais cette mémoire est visualisée en permanence sur l'écran par un interface assez compliqué qui ne sera pas étudié ici.

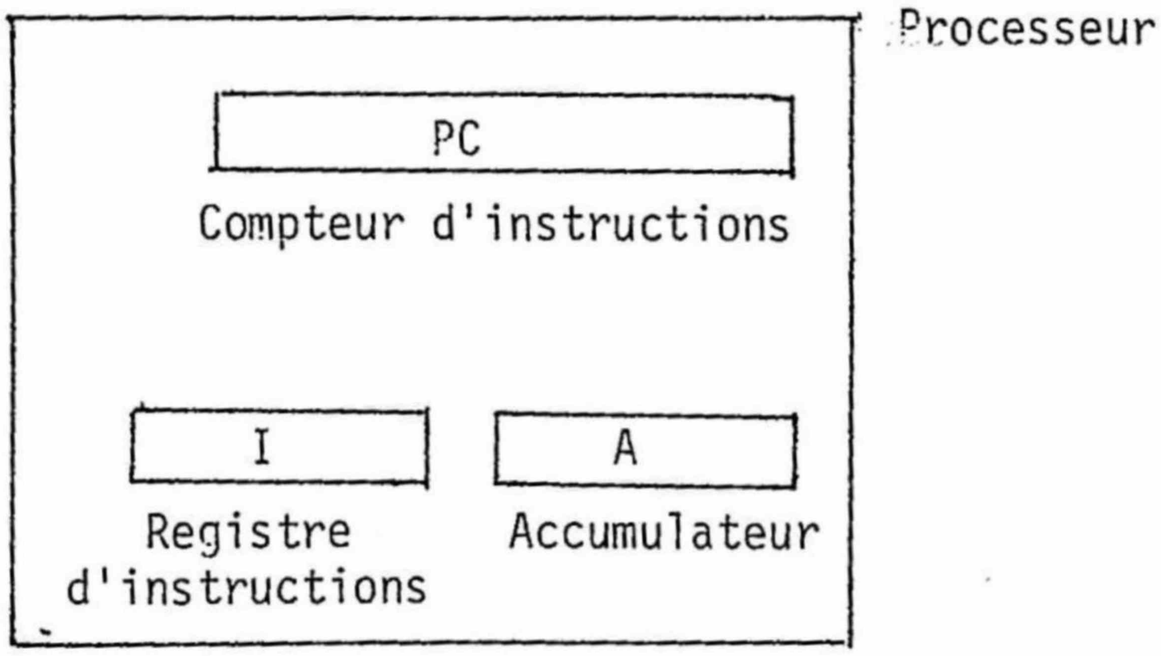
2.3 PROCESSEUR

Le processeur s'occupe de la gestion de la mémoire et des interfaces d'entrée/sortie. C'est lui qui va lire la mémoire pour exécuter le programme qui s'y trouve, qui va afficher des caractères sur l'écran, etc.

Pour faire cela, le processeur contient au moins trois registres:

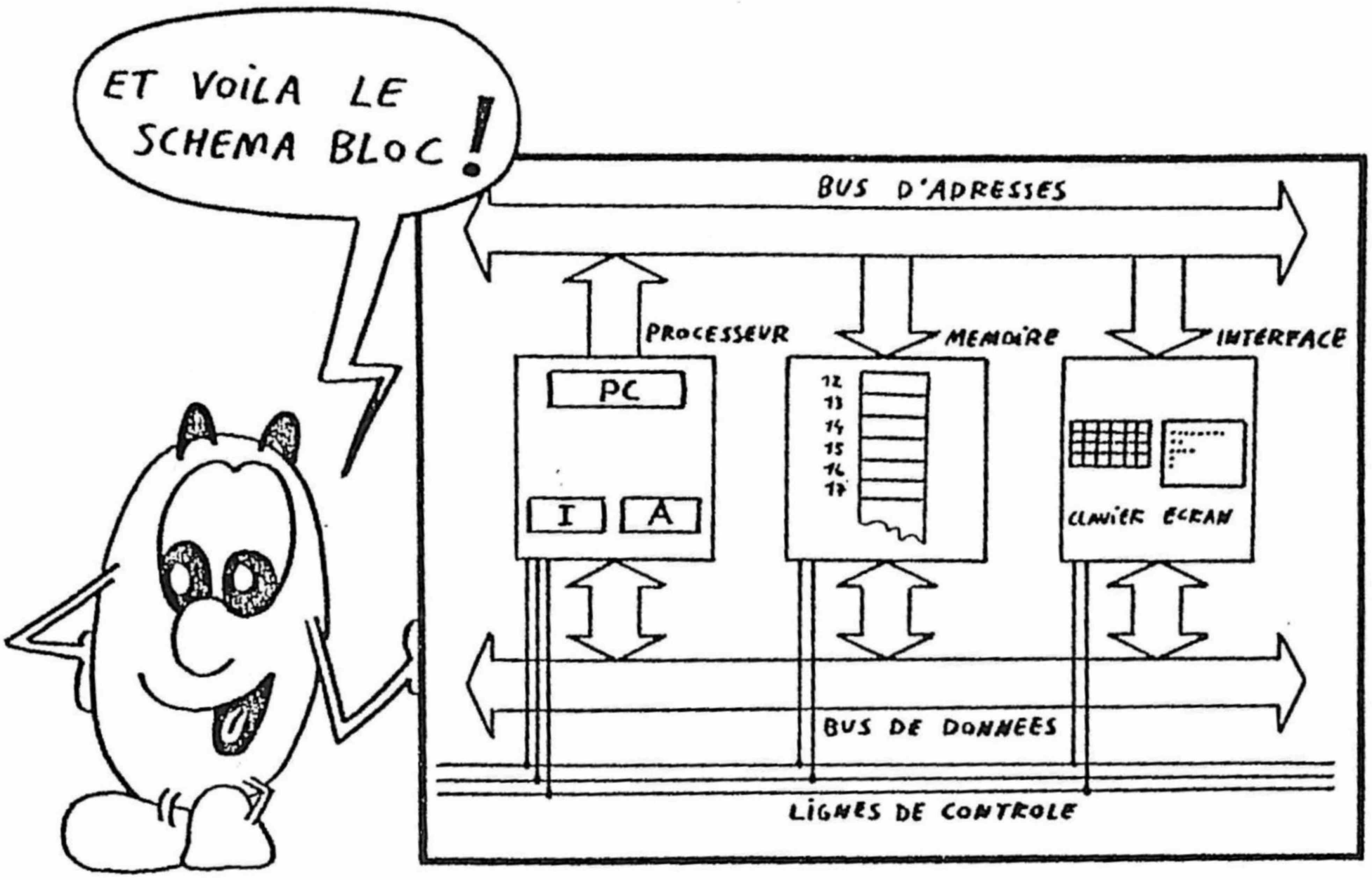
- le compteur de programme */PC: Program Counter/* qui pointe l'adresse de la position mémoire contenant l'instruction à éxécuter
- le registre d'instructions */I: Instruction Register/* qui mémorise l'instruction pendant son exécution
- le registre accumulateur */A: Accumulator/* qui est utilisé pour les transferts et opérations arithmétiques.

Il y a en général d'autres registres dans le processeur, dont nous parlerons par la suite.



Le processeur, la mémoire et l'interface sont reliés par des lignes de contrôle appelées bus */bus/*.

On distingue en général le bus d'adresse */address bus/*, qui transporte les adresses mémoires et les adresses des périphériques, le bus de données */data bus/*, qui transporte les contenus des positions mémoire et registres périphériques, et les lignes de contrôle */control bus/* qui synchronisent les transferts.



3 PROGRAMME

Le programme stocké en mémoire correspond à la succession d'ordres à donner au processeur pour qu'il effectue correctement les transferts et opérations entre ses registres, la mémoire et les périphériques. Un programme peut donc s'écrire sous forme d'une suite de nombres binaires associés à des adresses.

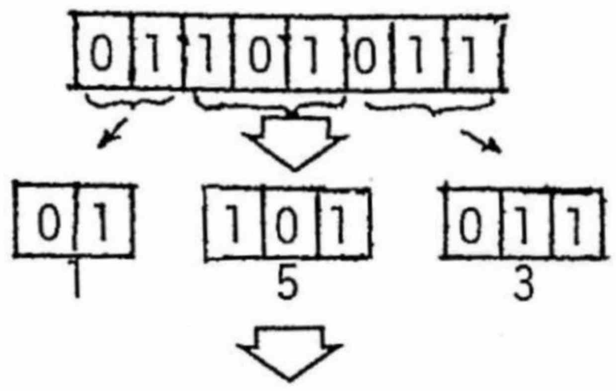
Cette méthode de travail est très longue et comporte beaucoup de risques d'erreur. C'est pourquoi, dans une première étape, nous n'écrirons plus les nombres en binaire, mais en octal. Les données seront aussi écrites en octal Cette transformation correspond simplement à faire un passage de la base 2 (binaire) à la base 8 (octal)

Décimal	Binaire	Octal
0	0	0
1	1	1
2	10	2
3	11	3
4	100	4
5	101	5
6	110	6
7	111	7
8	1000	10
9	1001	11
10	1010	12
...	...	...

On remarque qu'un nombre octal ne contient pas de 8, ni de 9.

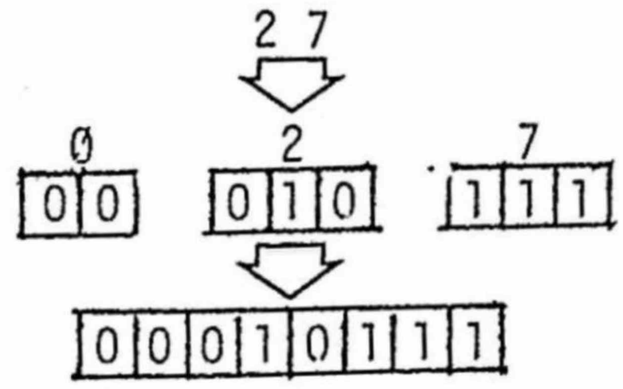
Pour faire rapidement cette transformation, il suffit de se rappeler la partie de la table entourée en gras.

Exemple de transformation binaire → octal:



d'où $(01101011)_2 = (153)_8$

Exemple de transformation octal → binaire:



d'où $(27)_8 = (00010111)_2$

E X E R C I C E :

1) Combien vaut $(252)_8$ en binaire ?
2) Combien vaut $(370)_8$ en binaire ?
3) Combien vaut (10111000) en octal ?
4) Combien vaut (00101010) en octal ?
5) Combien font $(27)_8 + (101)_8$ en bianire ?
6) Combien font $(00111001)_2 + (130)_8$ en octal ?

REPONSES:

1) $(10101010)_2$ 2) $(11111000)_2$ 3) $(270)_8$
4) $(52)_8$ 5) $(01011000)_2$ 6) $(221)_8$