AEROMARITIME
ELECTRONIQUE
57, Avenue d'Iéna
Paris-16e 553 67-00

le calculateur électronique de bureau LOCI-2 et sa programmation

par Jacques BOIVIN

Le calcul automatique doit être nécessairement mis aujourd'hui à la disposition des bureaux d'études sous une forme souple et un encombrement réduit et cependant, avec de puissantes possibilités de calcul et une mise en œuvre simple. Un calculateur électronique de bureau doit donc se présenter comme la réduction d'un grand ensemble de calcul numérique : il doit en posséder les fonctions essentielles, utiliser des supports d'informations semblables, tant à l'entrée qu'à la sortie, et enfin posséder une programmation qui, tout en étant le plus proche possible de la formulation mathématique écrite, doit cependant porter en elle les rudiments des méthodes de langages de programmation des grandes machines informatiques. Sous cette forme un calculateur de bureau constitue non seulement un précieux outil de travail quotidien, mais également permet aux ingénieurs qui l'utilisent une approche facile des grands ensembles de traitements de l'information.

Conçus selon ces principes de base, les calculateurs électroniques de bureau LOCI*, réalisés par AERomaritime Electronique possèdent un fonctionnement logarithmique et leur programmation est assurée par une carte perforée unique ; leur taille et leur consommation sont comparables à celles d'une machine à écrire électrique. Tout comme un calculateur numérique, ces calculateurs de bureau possèdent :

— des entrées-sorties par affichage direct sur tubes « Nixie », par imprimante genre Télétype ou par bande perforée ;

— des mémoires au nombre de 16 plus la carte-programme à 80 niveaux d'instructions ;

— une unité arithmétique et même logarithmique ;

— un bloc de contrôle comportant notamment un compteur d'ordres et un registre d'index.

Sur le plan programmation, le LOCI-2 ne nécessite aucun apprentissage spécial et son langage reste aussi proche que possible de l'écriture des formules mathématiques à traiter. De plus, chaque instruction du programme correspond à la pression sur une touche du clavier.

CONSTITUTION DU LOCI-1.

Le modèle le plus simple des calculateurs LOCI, dénommé LOCI-1, comporte les quatre éléments constitutifs de base de tout calculateur électronique (voir schéma synoptique de la figure 1).

1) **L'ENTRÉE ET LA SORTIE DES INFORMATIONS** s'effectuent par affichage en clair sur une rangée de tubes « Nixie », avec positionnement automatique de la virgule et du signe, dans les limites de capacités de 10^{-10} à 10^{10}. Cette technologie procure au calculateur un fonctionnement silencieux, une très bonne fiabilité (aucune pièce en mouvement) et une absence d'ambiguité dans l'interprétation des résultats, qui sont visibles de loin et apparaissent toujours à la même place.

---

* LOCI : LOgarithmic Computing Instrument.

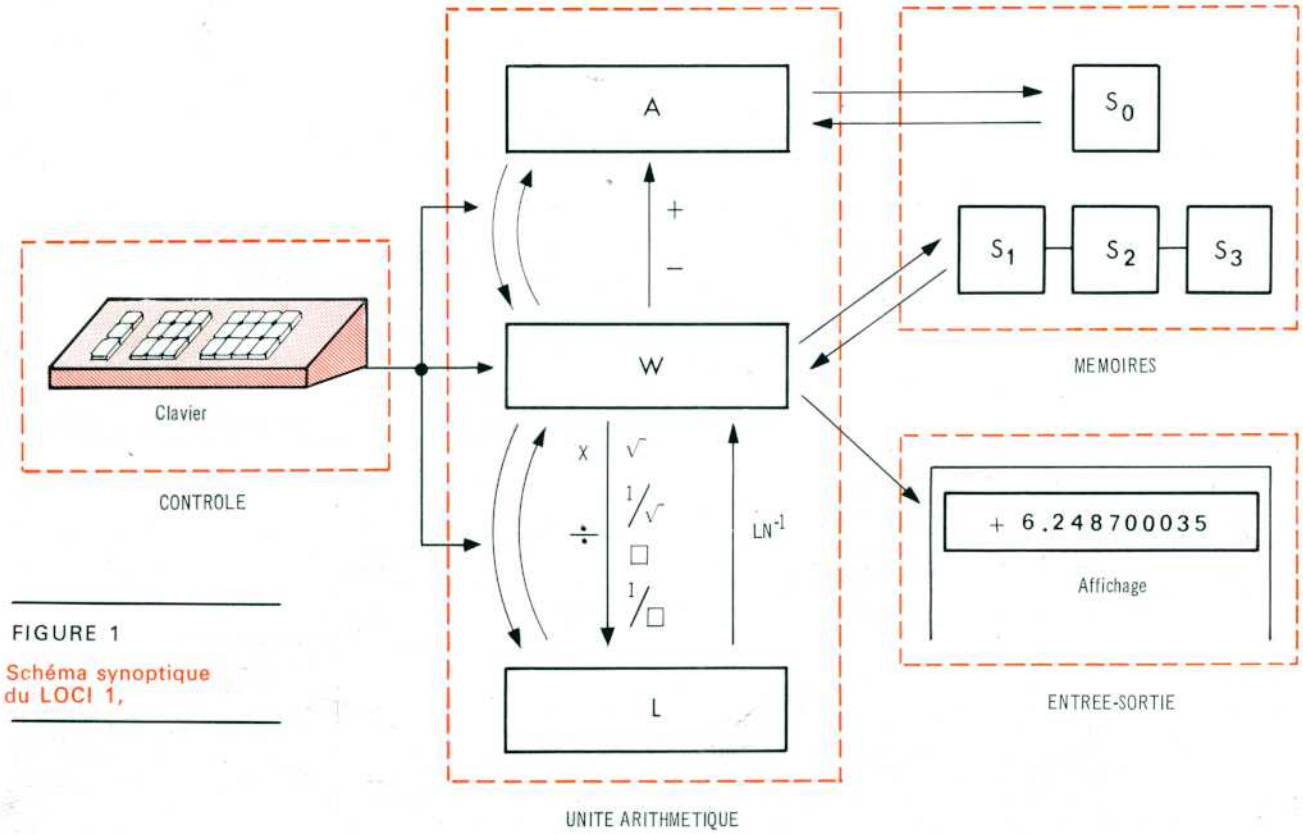


FIGURE 1

Schéma synoptique du LOCI 1,

2) **L'UNITÉ ARITHMÉTIQUE** (qu'on devrait plutôt nommer unité **logarithmique**) se compose principalement de trois registres d'une capacité de dix chiffres décimaux, plus le signe et la virgule (au cours d'un calcul, si la capacité d'un registre est dépassée, le voyant « erreur » s'allume) :

a) le **registre entrée-sortie W,** dont le contenu est affiché sur les tubes « Nixie », reçoit tout nombre composé sur le clavier et tout résultat doit y être ramené pour sa lecture.

b) **l'accumulateur A** est affecté à la somme algébrique des nombres entrés dans W, que les opérateurs + et — ajoutent et retranchent de son contenu, avec alignement automatique des décimales.

c) **le registre logarithmique L** est en réalité un **accumulateur de logarithmes népériens** qui confère au LOCI toute sa puissance et en fait l'originalité. **Les logarithmes népériens** sont générés par simple pression sur les touches suivantes :
(X) génère le logarithme du nombre contenu dans W et l'ajoute au contenu de L, ce que nous symboliserons :

$$(X) \curvearrowright L + \text{Log } (W) \rightarrow L$$

Nous symboliserons de même :

$(\div) \curvearrowright L - \text{Log } (W) \rightarrow L$
$(\sqrt{\ }) \curvearrowright L + 1/2 \text{ Log } (W) \rightarrow L$
$(1/\sqrt{\ }) \curvearrowright L - 1/2 \text{ Log } (W) \rightarrow L$
$(□) \curvearrowright L + 2 \text{ Log } (W) \rightarrow L$
$(1/□) \curvearrowright L - 2 \text{ Log } (W) \rightarrow L$

En plus des dix chiffres décimaux de la mantisse, L contient également :
- les deux chiffres décimaux de la caractéristique ;
- le signe de la caractéristique (+ ou — selon que la valeur absolue du nombre représenté est supérieure ou inférieure à 1) ;
- le signe du résultat du calcul lors de son affichage dans W.

Le résultat N du calcul est donc contenu dans L sous la forme de son logarithme. La connaissance de ce résultat est assurée par le calcul de l'antilog de N et son envoi dans W, opération symbolisée par $e^{(L)} \rightarrow W$ et réalisée par pression sur la touche (LN^{-1}) qui peut être assimilée au signe = pour toutes les opérations utilisant les logarithmes.

Transferts directs. En plus des opérateurs arithmétiques déjà rencontrés, il existe également des touches repérées (W → A), (A → W), (W → L), (L → W), commandant le transfert direct du contenu du premier registre dans le second, tout en laissant inchangé le contenu du premier (sauf L → W, qui remet L à zéro après le transfert).

L'utilisation des deux premières touches est évidente (elles assurent la mise en place du premier terme d'une somme et le rappel de cette somme dans W pour affichage) ; les deux autres servent à l'obtention des logarithmes et des exponentielles (voir § « utilisation »).

3) **Les mémoires** ont, comme les registres précédents, une capacité de 10 chiffres décimaux, plus le signe et la position de la virgule. Leur nombre maximal est de quatre dans le LOCI-1, réparties comme suit :

— une mémoire So, associée à l'accumulateur A ;

— trois mémoires, S_1, S_2, S_3, associées au registre W.

Les touches de transfert correspondantes (So ➛ A), (A ➛ So), etc., possèdent des fonctions suffisamment explicites. Notons seulement que ces transferts laissant inchangé le contenu du registre-origine, le seul moyen de remettre une mémoire à zéro est d'y transférer le contenu d'un registre préalablement remis à zéro (par action sur l'une des touches (CL W), (CL A), pour annuler respectivement W et A ou (PRIM) qui remet simultanément A, W et L à zéro).

4) **Le contrôle de LOCI-1** s'effectue exclusivement à partir du clavier (fig. 2 a) comprenant 40 touches repérées en clair et correspondant à autant d'instructions distinctes, deux voyants (« erreur » et « réponse » dont le bref éclair indique que l'instruction a été exécutée) et un inverseur qui, dans la position « AFF.AUTO ramène automatiquement le contenu de A dans W après chaque instruction + ou — (affichage des sommes partielles).

UTILISATION DU LOCI-1

Entrée des données.

Tout nombre composé au clavier entre dans W et est affiché en même temps. Pour composer un nombre, il suffit de presser les touches correspondantes dans l'ordre où l'on écrirait les chiffres, de gauche à droite, **y compris la virgule.**
Ainsi pour entrer 19,726 et 0,01, il suffit d'appuyer dans cet ordre, sur (1) (9) (,) (7) (2) (6) et (0) (,) (0) (1) (ou, tout simplement (,) (0) (1), les zéros précédant la virgule étant ignorés par le LOCI).

En cas d'erreur de frappe, trois touches de correction peuvent être utilisées :

(←) efface le dernier chiffre composé,

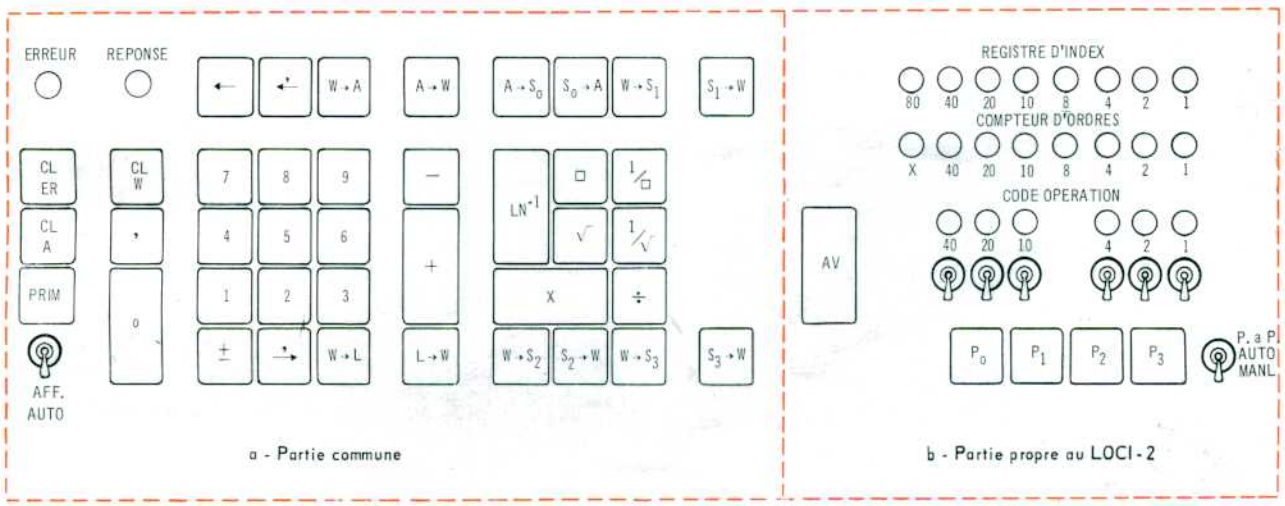
(➛) et (←) déplacent la virgule d'une décade dans le sens indiqué et peuvent être utilisées à tout moment pour multiplier ou diviser le contenu de W par une puissance de 10 quelconque.

Dès que le nombre entré dans W a été affecté d'un opérateur (voir § suivant), la première pression sur une touche numérique remet automatiquement W à + O avant l'entrée du nouveau nombre ; d'où la règle **d'entrée de nombres négatifs :** la pression sur la touche (±) (qui change le signe du contenu de W) doit **suivre** l'entrée des chiffres (ou au moins du premier).

Opérateurs.

Comme précédemment pour l'entrée des nombres négatifs, la pression sur la touche de l'opérateur affectant le contenu de W doit toujours **suivre** l'entrée du nombre dans W.

FIGURE 2.
Clavier des calculateurs LOCI-1 et 2



Exemples.

— **Addition 12 + 0,25**

Action sur les touches	Opérations réalisées
(1) (2)	W contient alors + 12
(W → A)	A contient + 12
(,) (2) (5)	W contient + 0,25
(+)	A contient (12 + 0,25)
(A → W)	W contient le résultat + 12,25

Notons que les deux dernières opérations se ramènent à une seule, (+), dans la position AFF.AUTO.

— **Multiplication (12) × (—5).**

(1) (2)	W contient (+ 12)
(×)	L contient Log (+ 12)
(5) (±)	W contient (—5)
(×)	L contient Log (+12)+Log (+5) et le signe — du résultat
(LN $^{-1}$)	W contient e^{L} = (—60)

— **Racine carrée $\sqrt{25}$.**

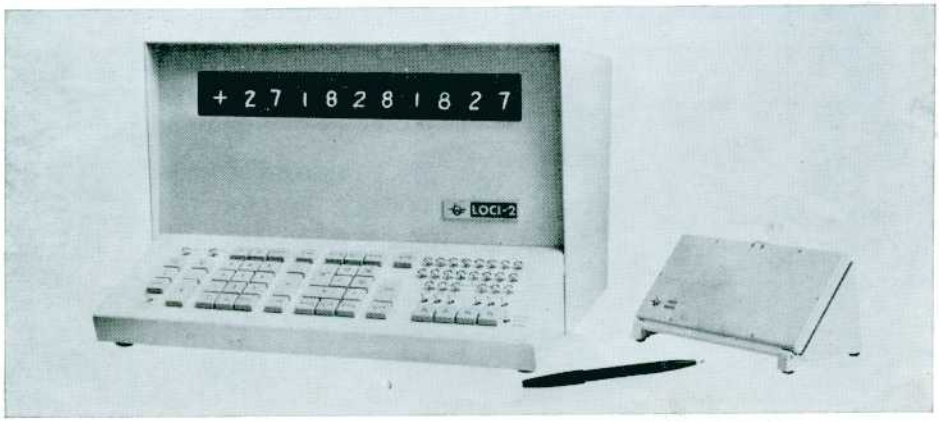
(2) (5)	W contient 25
($\sqrt{}$)	L contient 1/2 Log (25)= Log ($\sqrt{25}$)
(LN $^{-1}$)	W contient $\sqrt{25}$ = 5

Il n'est évidemment pas nécessaire d'effectuer l'opération LN-^{1} pour chaque produit partiel, en cas de produits de plus de deux termes. Il est ainsi possible d'effectuer des sommes de produits ou des produits de sommes (comprenant également des carrés ou racines carrées tant au numérateur qu'au dénominateur) **sans mobiliser de mémoires pour les résultats intermédiaires,** les registres A et L en faisant office.

Transferts directs.

L'utilisation des touches de transfert direct pour l'obtention des résultats (A → W) ou la mise en mémoire et le rappel de résultats intermédiaires ou de constantes (W → S_1, S_1 → W, etc.) est évidente. Une attention particulière doit être portée sur W → L et L → W qui permettent d'obtenir facilement exponentielles et logarithmes.

a) **Exponentielles.** L'opérateur LN^{-1} effectuant en réalité $e^{(L)}$ → W, e^{N} est donc obtenu en entrant N dans W, puis en effectuant W → L ; N est dès lors considéré comme le logarithme d'un nombre, d'où la séquence pour obtenir par exemple $e^{-2,5}$:
(2) (,) (5) (±) (W → L) (LN^{-1}).



Calculateur LOCI-2 avec lecteur de carte-programme.

b) **Logarithmes, puissances.** L'opérateur X effectuant en réalité Log (W) ➝ L, la séquence pour obtenir Log 13 est donc la suivante : (1) (3) (×) (L ➝ W).

Contrairement aux autres transferts, L ➝ W remet L à zéro, le rendant disponible pour un nouveau produit. Ceci rend très aisé le travail sur les logarithmes et permet d'obtenir des racines et des puissances d'ordre quelconque.

Exemple : $\sqrt[10]{1024.}$

(1) (0) (2) (4)(×) (L ➝ W) (×)	L contient Log (1 024) L contient Log (Log 1 024)
(1) (0) (÷)	L contient $\text{Log}\left(\frac{\text{Log } 1\,024}{10}\right) = \text{Log}\left(\text{Log}\sqrt[10]{1\,024}\right)$
(LN^{-1})	W contient $\text{Log}\left(\sqrt[10]{1\,024}\right)$
(W) ➝ L)	L contient $\text{Log}\left(\sqrt[10]{1\,024}\right)$
(LN^{-1})	W contient $\sqrt[10]{1024} = 2$

c) **Exemples de calculs enchainés :**

1°) $X = \dfrac{a - \sqrt{b^2 + cd}}{4\ e^2}$ pour a = 2 ; b = 4,3 ; c = 0,68 ; d = — 5 ; e = 3,27

par (,) (6) (8) (×) (5) (±) (×) (LN^{-1}) (W ➝ A) (4) (,) (3) (□) (LN^{-1}) (+) (A ➝ W) ($\sqrt{\ }$) (LN^{-1}) (±) (W ➝ A) (2) (+) (A ➝ W) (×) (4) (÷) (3) (,) (2) (7) (1/□) (LN^{-1})

2°) $Y = \text{SH}x = \dfrac{e^x - e^{-x}}{2}$ pour x = 1,25

par (1) (,) (2) (5) (W ➝ L) (LN^{-1}) (W ➝ A) (÷) (LN^{-1}) (—) (A ➝ W) (×) (2) (÷) (LN^{-1})

Ces deux types de calculs peuvent se représenter fréquemment au cours d'un certain travail (par exemple : calcul d'un tableau de valeurs de X pour différentes valeurs des paramètres a, b,...). Si l'on excepte les opérations d'entrée des variables dans W, indispensables dans tous les calculateurs, électroniques ou non, il a fallu 19 pressions de touches pour obtenir X et 11 pour obtenir Y. Le nombre de ces opérations peut devenir prohibitif dans le cas de calculs répétitifs où seule une variable change à chaque fois. D'où l'idée d'emmagasiner la séquence d'opérations nécessaire dans une **mémoire** spéciale, qui retiendra ainsi les **instructions** à donner au calculateur **dans un certain ordre** pour l'exécution du calcul demandé, l'opérateur n'ayant plus qu'à entrer les variables et à déclencher le calcul. D'où la notion de mémoire-programme.

CHOIX d'un PRINCIPE de mémoire-programme.

Au constructeur de calculateurs désirant réaliser une machine ainsi automatisée s'impose alors deux choix primordiaux : quel sera le support de cette mémoire-programme, et quel sera son moyen d'accès pour la constitution de nouveaux programmes ou leur modification ? Des réponses choisies pour ces questions, découlent les trois principaux types de programmation suivants :

1) Programme direct.

On peut d'abord imaginer une mémoire électronique (bascules bi-stables ou tores magnétiques) à accès direct par le clavier : dans ce cas, chaque pression sur une touche-opérateur génère un code spécial qui vient occuper un certain nombre de positions dans cette mémoire. Une logique câblée assure le transfert de ces instructions au calculateur dans l'ordre où elles ont été entrées. Ce principe, bien que séduisant au premier abord (accès rapide, pas de nécessité pour le programmeur de connaître le « langage » codé de la machine), présente de gros inconvénients :

a) pour les **longs programmes,** une telle mémoire est **onéreuse** et elle est de toutes façons d'une capacité limitée à la construction de l'appareil ;

b) une étape importante de toute programmation est la **vérification** du programme et sa **correction** éventuelle. Or, quand bien même un mode de fonctionnement pas à pas permettrait de localiser l'erreur, il serait impossible de modifier localement le programme sans le recomposer en entier, d'où de nouvelles sources d'erreurs possibles.

c) enfin et surtout, un tel programme sans support matériel durable est **extrêmement fragile :** l'introduction d'un nouveau programme détruit définitivement le précédent, de même que l'arrêt du calculateur en fin de journée. En cas d'utilisation de la même machine par plusieurs personnes, on ne sait jamais exactement dans quel état la mémoire a été laissée.

Tous ces facteurs rendent la **fiabilité** d'un tel programme direct extrêmement précaire.

2) Programme câblé.

Les inconvénients précédents peuvent être palliés en **câblant** les programmes ; mais comme l'utilisation de panneaux de connexions amovibles est impossibles sur les calculateurs de bureau (poids, encombrement et prix de revient prohibitifs) ce câblage doit être réalisé définitivement à la construction du calculateur.

La simplicité d'emploi en est grandement améliorée : une touche par programme, repérée (sin x) ou ($\sqrt{x^2 + y^2}$) par exemple, qu'il suffit de presser après introduction des variables dans les mémoires. Mais cette simplicité se paye de deux nouveaux inconvénients :

a) chaque programme occupant **définitivement** un certain nombre de positions de mémoire, le **nombre de programmes** disponibles sur une machine donnée est forcément **limité** (une dizaine en général) par de nombreuses considérations : prix de revient, volume et poids des composants, encombrement du panneau de commande du calculateur, etc. ;

b) la modification d'un programme ou son remplacement par un nouveau ne sont possibles qu'en usine et donc exceptionnels, Mais l'évolution constante des diverses techniques rend très difficile pour l'utilisateur la prévision à longue échéance de ses besoins en programmes de calcul. Le choix du modèle à acquérir est encore plus difficile lorsque le même calculateur doit être utilisé par les différents ingénieurs d'un bureau d'études ou les différents services d'une entreprise.

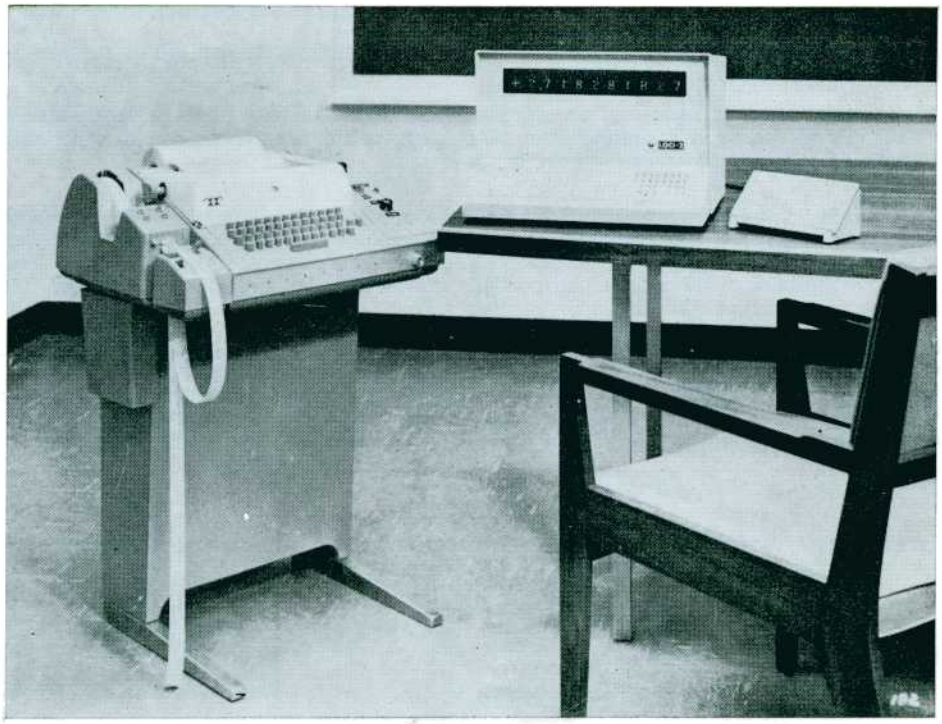
3) Programme extérieur.

Pour supprimer les inconvénients inhérents aux précédents principes de programmation : fragilité, limitation en nombre et longueur de programmes, extension onéreuse et malaisée, tous les calculateurs ont recours à des **supports** de programmes extérieurs au calculateur proprement dit ; ce sont les supports habituels au traitement numérique de l'information : cartes et bandes perforées, bande magnétique, etc. Ils sont durables, faciles à stocker et à manipuler, leur densité d'écriture permet d'inscrire de longs programmes sous un volume réduit.

Leur lecture met évidemment en jeu des machines électro-mécaniques encombrantes et onéreuses (lecteurs de cartes ou de bandes, unités magnétiques) mais nullement disproportionnées pour les gros calculateurs. Leur inconvénient majeur réside en ce que le programme ne peut être enregistré dans un code directement compréhensible par la machine. Malgré l'introduction de divers langages symboliques (FORTRAN, ALGOL, etc.) qui tendent à se rapprocher de la formulation mathématique usuelle, la nécessité de passer par divers transcodeurs ou compilateurs pour l'assemblage d'un programme impose le recours à des **programmeurs** spécialisés. Cette servitude acceptable dans le cas de gros ensembles en raison du volume de travail, est prohibitive pour un calculateur de bureau qui doit être utilisable et programmable par tout ingénieur ou technicien appelé à s'en servir.

PRINCIPES DE PROGRAMMATION DU LOCI-2.

Etant à **vocation scientifique universelle,** le LOCI devait avoir recours à la **programmation extérieure** pour éviter les limitations inhérentes aux autres principes. Un pont a dû tout d'abord être établi entre le **langage de la machine** et la **formulation mathématique usuelle** pour permettre à l'utilisateur non spécialisé de construire n'importe quel programme. A cet effet, examinons à nouveau la suite d'opérations constituée par les pressions sur les touches indiquées précédemment au paragraphe « transferts directs » Le principe choisi pour le LOCI-2 est de faire correspondre à chaque touche un **code** facilement décomposable en numération binaire ; ces codes sont ensuite perforés dans un **support** commode (une carte pré-perforée IBM) dans l'ordre de leur exécution. Une **logique** (bloc-contrôle du calculateur) « lit » chaque code, le transfère à l'unité arithmétique pour l'exécution de l'instruction correspondante, puis avance au niveau suivant pour lire la nouvelle instruction, jusqu'à ce que soit rencontré un code « stop » qui arrête le processus.



Calculateur LOCI-2 avec lecteur de carte-programme et Télétype servant de lecteur-perforateur de bande et d'imprimante.

La confection d'un programme se réduit donc à trois opérations simples :

— **imaginer** la suite des touches à enfoncer sur le clavier d'un LOCI pour effectuer le calcul et **écrire leurs symboles** sur une feuille de papier. (Les exemples précédents ont montré combien cette écriture symbolique est proche de la symbolique mathématique usuelle) ;

— **inscrire** à côté de chaque symbole le **code correspondant** fourni par le « dictionnaire » d'instructions du LOCI ;

— **perforer manuellement** les codes, dans l'ordre ci-dessus, dans une carte en suivant les repères imprimés.

Toutes ces opérations sont d'une réalisation élémentaire et il ne reste plus qu'à insérer la carte dans le lecteur pour effectuer le calcul.

ELÉMENTS DE PROGRAMMATION

1) **Codes Instruction :** en plus des 40 touches du clavier du LOCI-1, des instructions supplémentaires sont nécessaires en raison de l'automaticité du calcul (comme le code « stop » par exemple qui vient d'être rencontré). Le LOCI-2 dispose au total de **64 instructions** distinctes, repérées chacune par un **nombre octal de deux chiffres.**

2) **Séquence simple-entrée des variables.** Une carte-programme comprend 80 colonnes de 6 bits explorées séquentiellement ; chaque **niveau** est numéroté dans l'ordre de 00 à 79.

Exemple :

Le programme calculant Sh x, en supposant que x est déjà dans W, s'écrit alors :

Niveau d'ordre	Instruction	Code opération	Niveau d'ordre	Instruction	Code opération
00	$W \rightarrow L$	46	06	$A \rightarrow W$	45
01	LN^{-1}	14	07	X	12
02	$W \rightarrow A$	44	08	2	22
03	$\frac{\cdot}{\cdot}$	17	09	$\frac{\cdot}{\cdot}$	17
04	$\dot{LN}^{-1}$	14	10	$\dot{LN}^{-1}$	14
05	—	15	11	stop	37

Si le calcul porte sur **plusieurs variables,** deux méthodes peuvent être utilisées :

a) entrer manuellement chaque variable dans une mémoire distincte, où le programme ira la rechercher au moment voulu ;

b) intercaler dans la séquence une instruction « stop » à chaque fois qu'une nouvelle valeur est nécessaire. L'opérateur entre alors cette valeur dans W lorsque le calculateur s'arrête et fait repartir manuellement le programme.

3) **Transferts inconditionnels.** Le transfert en séquence d'un niveau d'ordre au suivant permet de construire des programmes de 80 instructions au maximum. Ce nombre serait insuffisant pour des calculs un peu compliqués, tels que sommations de séries ou résolutions d'équations, s'il n'était pas possible d'effectuer des **boucles d'itération.** Ceci suppose le **transfert** d'un niveau de programme à un autre.

Le bloc contrôle du LOCI-2 (voir figure 3) comporte un **compteur d'ordres** (PC) dont la valeur indique en permanence l'ordre du **prochain niveau d'instruction** à atteindre ; son contenu avance automatiquement d'une unité après exécution de chaque instruction. Un transfert s'effectue à l'aide de l'instruction W ➝ PC qui fait passer dans le compteur d'ordres les deux premiers chiffres du contenu de W. De cette façon la séquence :

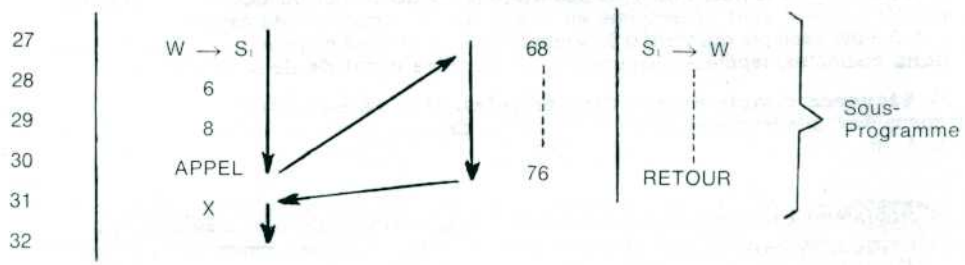
Niveau d'ordre	Instruction	Code opération
26	W ➝ S_1	52
27	4	24
28	2	22
29	W ➝ PC	40
42	S_1 ➝ W	53

fait « sauter » le programme de l'instruction n° 26 à l'instruction n° 42 à partir de laquelle le transfert en séquence reprend.

Il suffit donc de **trois niveaux** du programme pour effectuer un transfert inconditionnel Si le nouveau niveau atteint **précède** le niveau de départ dans la séquence, **une boucle** a été construite, ce qui condense l'écriture du programme dans le cas de calculs répétitifs ; la « sortie » d'une telle boucle sera explicitée plus bas.

4) **Appel-Retour.** Dans certains calculs, des « **sous-programmes** » (lignes trigonométriques par exemple) doivent pouvoir être utilisés chaque fois que cela est nécessaire. Ce rôle est assuré par les instructions APPEL (n° 64) et RETOUR (n° 65) ; la première est équivalente à W ➝ PC, mais auparavant elle transfère dans deux mémoires spéciales (PCS et DCS, figure 3) les contenus du compteur d'ordre PC et du registre d'index DC. La seconde reprend la séquence au niveau où elle avait été interrompue lors de l'appel du sous-programme, en retransférant respectivement dans PC et DC les contenus de PCS et DCS.

Exemple :



5) **Transferts conditionnels-tests.** Nous avons vu comment « boucler » un programme ; encore faut-il sortir de cette boucle à un moment donné pour terminer le calcul. C'est pourquoi le LOCI-2 est capable de **décisions logiques,** basées sur le **résultat de tests** portant sur le contenu des différents registres ou compteurs. Ces tests sont des questions posées au calculateur (par exemple : « le contenu de W est-il nul ? ») noté symboliquement (W = 0 ?). La règle appliquée par le LOCI-2 est la suivante :

— si la réponse est « NON », **le transfert s'effectue en séquence** à l'instruction suivant le test ;

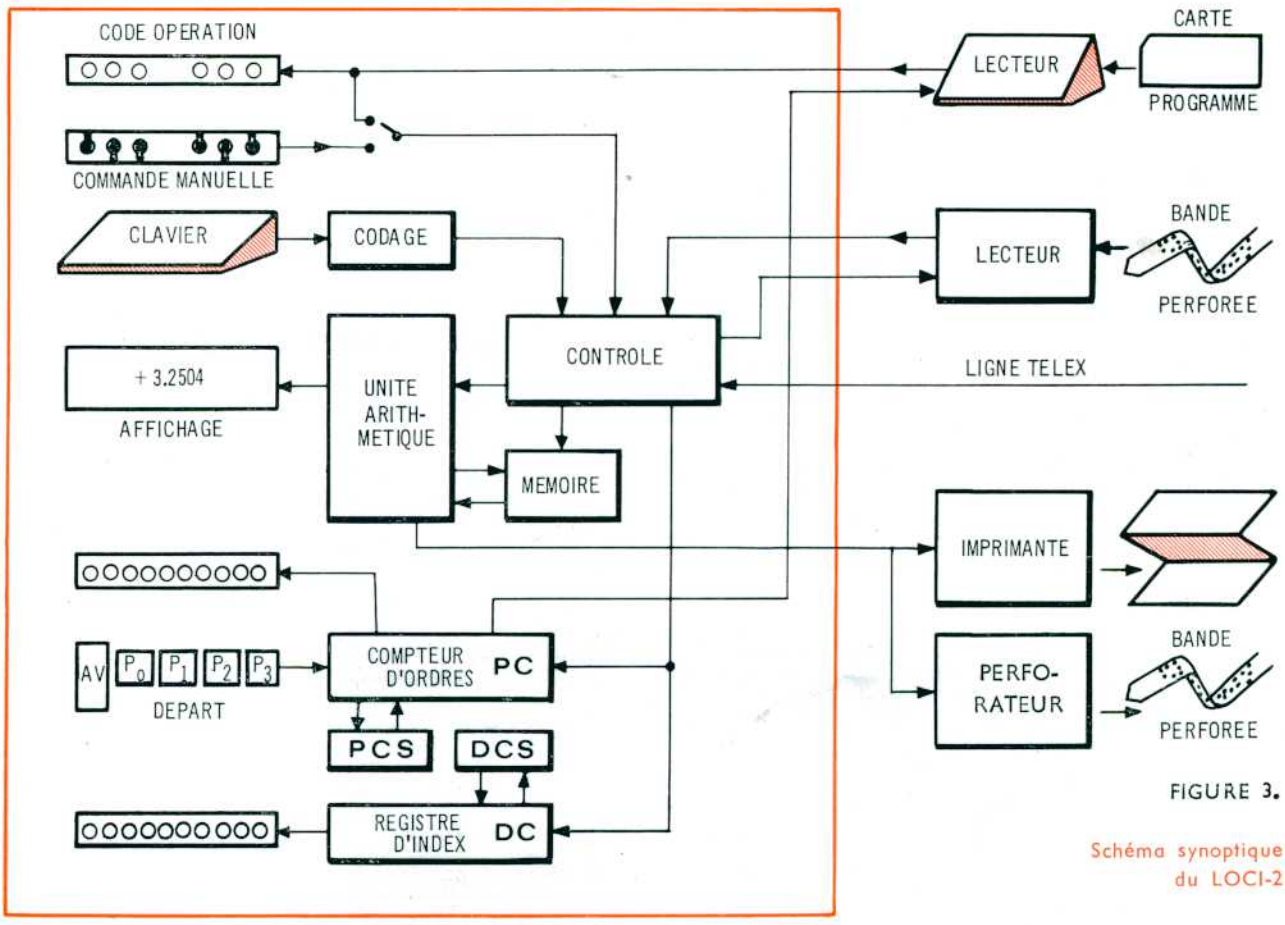


FIGURE 3.

Schéma synoptique du LOCI-2

— si la réponse est « OUI », **le transfert s'effectue à la quatrième instruction** suivant le test, sautant ainsi trois niveaux du programme. Or, ces trois niveaux suffisent exactement à effectuer un transfert vers un autre niveau ; par exemple la séquence suivante :

43	W = 0 ?	72
44	2	22
45	5	25
46	W → PC	40
47	stop	37

est interprétée par le LOCI-2 comme « si le contenu de W est nul, aller en 47 (et, ici, arrêter le calcul) sinon, aller en 25 ».

Le LOCI-2 dispose de six tests qui sont :

W = 0? W<0 ? A = 0 ? DC = 0? Erreur ? L < 0?

Le dernier test sert à déterminer l'ordre de grandeur d'un nombre ; en effet, s'il est inférieur à 1, son logarithme est négatif. Supposons que nous voulions arrêter un calcul, commençant au niveau 15, lorsque le nombre contenu dans W est inférieur à 10^{-4}, nous utiliserons la séquence suivante :

23	X	12	28	L < 0 ?	74
24	1	21	29	1	21
25	0	20	30	5	25
26	0	20	31	W → PC	40
27	□	06	32	stop	37

6) **Séries finies.** Dans le cas où la même séquence doit être recommencée un nombre connu de fois, on utilise un **registre d'index** (DC, figure 3). Le nombre d'itérations à effectuer y est transféré au moyen de l'instruction W → DC (nº 42) et son contenu est diminué de 1 au moyen de l'instruction DECrémenter (nº 66). Voici à titre d'exemple un programme calculant N ! (N étant contenu dans W au début du calcul) :

00	W → DC	42	05	0	20
01	DC → W	43	06	1	21
02	X	12	07	W → PC	40
03	DEC	66	08	LN^{-1}	14
04	DC = 0 ?	70	09	stop	37

RÉALISATION PRATIQUE DU PROGRAMME.

Examinons maintenant comment ces programmes sont inscrits sur leur support, communiqués au calculateur et exécutés par celui-ci.

1) **Langage machine.** La grande simplicité du système choisi vient de ce que le «**mot machine** » correspondant à chaque code est son équivalent en **octal codé binaire** ; donc un mot de 6 bits de poids respectifs 1, 2, 4, 10, 20 et 40 (on remarquera l'analogie avec les codes alphanumériques à 6 moments des bandes perforées, du télétype, etc, dont l'utilité sera explicitée plus bas).

2) **La carte-programme.** Le support utilisé est la carte pré-perforée IBM à 40 colonnes de 12 positions chacune (soit 80 colonnes de 6 positions, en deux rangées numérotées respectivement 00 à 39 et 40 à 79). La perforation s'effectue manuellement en passant simplement une pointe (d'un crayon par exemple) dans les emplacements poinçonnés et repérés lorsque le bit correspondant doit être un 1. La décomposition du code opération octal en son équivalent codé binaire est évidente. A titre d'exemple, la figure 4 représente à l'échelle 1 le fragment de carte occupé par le programme Shx déjà rencontré.

La carte ainsi préparée est glissée dans un **lecteur statique** s'ouvrant comme un livre ; un « feuillet » porte autant de palpeurs conducteurs que de perforations possibles (soit 480), son vis-à-vis portant autant de surfaces de contact imprimées. Une fois le lecteur refermé sur la carte, chaque « 1 » se traduit par une continuité électrique à l'endroit correspondant, sans aucune pièce en mouvement, sans manipulation ni usure possible de la carte.

3) **Constitution du bloc-contrôle** (voir figure 3). Le LOCI-2 comprend, outre les circuits de codage du clavier et de commande de l'unité arithmétique et des mémoires déjà décrits à propos du LOCI-1, les éléments suivants :

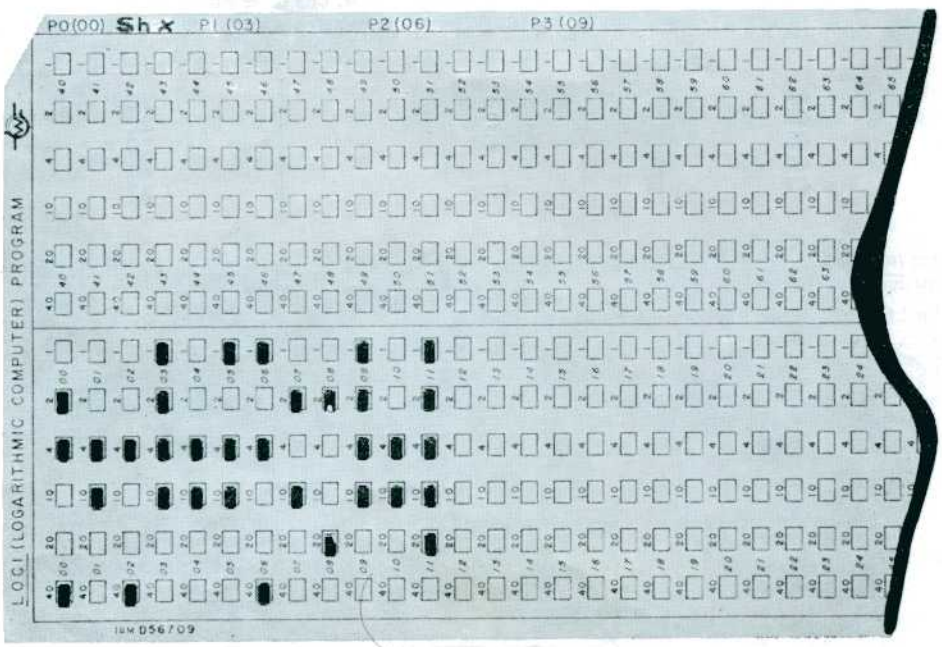


FIGURE 4.
Carte-programme de Sh x.

a) **Un compteur d'ordre PC** qui assure la « lecture » de la carte par exploration séquentielle des différentes colonnes. Un **affichage** associé par tubes néon, indique à tout moment le niveau d'ordre de l'instruction à exécuter et le code opération correspondant **avant son exécution ;** il sert au contrôle du programme. Une mémoire PCS' d'une capacité de deux chiffres décimaux, lui est associée pour assurer le RETOUR après APPEL d'un sous-programme ;

b) **Un registre d'index DC,** sert principalement à mémoriser le nombre de fois qu'une opération a été effectuée (parcours d'une boucle du programme, itération, etc.). Un **affichage** de son contenu par tubes néon, ainsi qu'une **mémoire** DCS, lui sont associés comme au compteur d'ordres.

4) **Déroulement du programme.** Le compteur d'ordres avance automatiquement d'une unité après l'exécution de chaque instruction et s'arrête dès qu'il rencontre l'instruction « stop ». Encore faut-il lui fixer un point de départ ; nous avons vu que l'instruction W➛ PC en fournissait un moyen. Mais le départ d'un programme se fait généralement par pression sur l'une des quatre touches repérées P_0 à P_3 ; celles-ci remettent le contenu de PC respectivement à 00, 03, 06 et 09, permettant ainsi de commencer à volonté la lecture de la carte programme en quatre points différents. Remarquons que trois niveaux séparant chacun de ces points, ils peuvent être utilisés comme les **adresses de quatre niveaux quelconques** du programme. Il est ainsi possible de faire tenir jusqu'à quatre programmes différents sur une même carte.

Enfin une touche repérée AVance permet de faire repartir le programme en séquence lorsqu'il a rencontré une instruction « stop » (par exemple, après l'entrée manuelle d'une variable dans W).

5) **Exemples de programmes.** Une idée de la capacité de calcul du LOCI-2 peut être donnée par la liste suivante de quelques programmes (ou groupes de programmes) n'utilisant chacun qu'une carte, soit 80 niveaux.:

Sin x et Cos x	$I = 1/12\ (Y_i - Y_{i+1})\ (Z_i + Z_{i+1})\ (Z_i^2 + Z_i^2 + 1)$
Arc Tg x	Sh x, Ch x, Arg. Sh x et Arg. Ch x
Ecart-type	Droite aux moindres carrés
N ! et Γ (x)	Exponentielle aux moindres carrés
$x^2 - e^{Kx} = 0$	Résolution des triangles quelconques
$ax^2 + bx + c = 0$	Fréquences de résonnances
$x\ e^x + K = 0$	Intégration par méthode de Simpson

Tous ces programmes en outre n'utilisent que quatre mémoires au maximum ; cette liste n'est évidemment pas limitative et l'intérêt principal du LOCI-2 réside précisément dans la possibilité illimitée d'extension de la bibliothèque des programmes par échanges entre les utilisateurs.

VÉRIFICATION DES PROGRAMMES.

Aussi importante que la confection d'un nouveau programme est sa **vérification ,**tous les programmeurs le savent bien. Celle-ci est prévue, grâce aux voyants d'affichage décrits plus haut et aux modes de fonctionnement suivants :

1) **Pas à Pas.** Dans cette position de l'inverseur l'avance automatique du compteur d'ordre après exécution de chaque instruction est supprimée et remplacée par la pression sur la touche AV, à chaque niveau, après vérification par l'opérateur du code-opération prévu au niveau suivant et éventuellement du contenu des différents registres et mémoires par transfert manuel dans W pour affichage. Une erreur éventuelle dans la conception logique du programme ou dans sa perforation, est ainsi aisément localisée. Trois cas de corrections peuvent se présenter :

a) instruction à **supprimer :** il suffit de recouvrir recto-verso les perforations du niveau considéré avec un simple ruban adhésif (l'instruction 00 étant ignorée par le LOCI) ;

b) instruction à **modifier :** il est aisé de modifier le code par perforation ou obturation des positions correspondantes comme précédemment. Pour éviter de sortir la carte sur le champ ou pour essayer l'instruction de remplacement, il est possible de passer en mode « **commande manuelle** » exposé ci-après ;

c) instruction à **ajouter :** la suite du programme étant décalée d'un ou plusieurs niveaux, il faut alors perforer une autre carte ; cependant la suite du programme peut être vérifiée en introduisant manuellement à l'aide du clavier l'instruction manquante.

2) **Commande manuelle :** dans ce mode de fonctionnement, le compteur d'ordres avance toujours manuellement par la touche AV, mais l'instruction exécutée au niveau atteint est, non pas celle perforée sur la carte, mais celle composée à l'aide des inverseurs « commande manuelle » située à droite du clavier. L'opérateur peut ainsi corriger son programme à tout niveau, en substituant une instruction mnauelle à celle qui a été perforée sur la carte et ne modifier celle-ci qu'en fin de vérification.

Dans la configuration qui vient d'être décrite, le LOCI-2 possède un champ d'utilisation très large pour les ingénieurs de bureaux d'études, les chercheurs de laboratoires, etc et il n'est limité que par le nombre des mémoires disponibles (16 au maximum). Dans certains cas particuliers, cependant, des extensions permettent de reculer encore les limites d'utilisation.

1) **Second lecteur de carte.** Si un problème nécessite plus de 80 niveaux de programme (la résolution des équations du 3e degré complètes en demande par exemple 135) il est aisé de le scinder en deux cartes (ou plus) puisque l'ouverture du lecteur et le remplacement de la carte ne perturbent en rien le contenu des compteurs ou registres. Mais s'il s'agit de faire fréquemment appel à des sous-programmes un peu longs (lignes trigonométriques par exemple) il est possible de connecter au LOCI-2 un second lecteur de cartes. Le bit supplémentaire repéré X dans l'affichage du compteur d'ordres avec la valeur « 1 » au lieu de « 0 » permet de désigner un niveau du second lecteur au lieu du premier. Ce passage est réalisé, soit par une instruction spéciale W ➤ XPC (nº 41), soit en donnant à W le signe — avant l'instruction W ➤ PC, qui transfère alors au niveau correspondant du second lecteur.

2) **Entrées-sorties supplémentaires.** A chaque instruction ou touche du clavier correspond un mot-machine de 6 bits ; celui-ci peut être reçu de l'extérieur par une interface électrique à 6 fils permettant le raccordement du LOCI-2 à tout organe périphérique utilisant un code à 6 moments : imprimante, lecteur de bandes perforées, machine à écrire électrique et leur combinaison représentée par les machines perforatrices-imprimantes-lectrices genre « Télétype » ou « Flexowriter ». Citons rapidement quelques exemples d'application. :

a) **impression des résultats :** l'instruction « ECRIRE » (nº 34) commande le transfert successif à l'imprimante (et/ou à la perforatrice de bande) des chiffres du contenu de W, avec le signe et la position de sa virgule. Des instructions supplémentaires sont disponibles pour l'espace et le retour du chariot ;

b) **lecture des données :** au cas où les variables à traiter sont des résultats de mesures numériques, il suffit de les perforer sur une bande dans l'ordre de leur introduction dans le LOCI. L'instruction « LIRE » arrête le compteur d'ordre et commande leur transfert dans W chiffre après chiffre (signe et virgule compris) jusqu'à lecture du code « AVance » qui rend le contrôle à la carte-programme et libère le compteur d'ordres ;

c) **programme extérieur :** le même processus peut être employé pour enregistrer sur bande perforée un très long programme ; les boucles et sous-programmes restant perforés sur la (ou les) carte-programmes ;

d) **travail « on line ».** Le calculateur LOCI-2 peut être utilisé avec rentabilité dans une boucle d'asservissement pour contrôle industriel par exemple. En effectuant des calculs statistiques, en résolvant des équations implicites ou transcendantes plus rapidement et avec plus de précision que des circuits analogiques, il peut éviter d'avoir recours à des calculateurs spécialisés.

Dans le cas d'informations arrivant à cadence relativement lente (par lignes télétype dans le cas de la météorologie par exemple) le LOCI peut également résoudre en temps réel de nombreux calculs préliminaires, diminuant d'autant le temps d'utilisation d'un ordinateur central (surtout si les résultats intermédiaires obtenus sont enregistrés sur bande magnétique au format IBM à l'aide d'un enregistreur incrémental tel que le DSR-1420 de DIGI-DATA).

Ces derniers exemples ouvrent des perspectives d'utilisation quasi-illimitées à un calculateur électronique de bureau tel que LOCI-2. N'oublions pas cependant que les caractéristiques uniques qui les rendent possibles — modes de calcul LOGARITHMIQUE et PROGRAMMATION souple et complète — sont maintenant à la portée de tous les bureaux d'études et laboratoires sous une forme pratique.

Un utilisateur moyen apprend à l'utiliser en une demi-heure, à le programmer en une demi-journée. Une fois familiarisé avec sa méthode de programmation, il lui suffira de bien peu — quelques notions de FORTRAN ou d'ALGOL — pour s'attaquer à la programmation de plus gros calculateurs ; ceci ajoute encore l'utilisation didactique aux possibilités déjà vastes du LOCI-2.

En voici assez pour vaincre les réticences au calcul électronique de trop d'ingénieurs qui se disaient : « cela » nous gagnerait un temps précieux, mais « cela » n'est pas pour nous : il faut recourir à des programmeurs qui, seuls, savent communiquer avec de telles machines, puis il faut attendre un temps considérable qu'elles soient libres pour prendre nos calculs, et tout ceci coûte très cher». Voici maintenant « cela » sur leur table, à leur disposition immédiate sans apprentissage spécial.

Jacques BOIVIN
Chef de la Division Télémesure
d'AERomaritime ELectronique.

Extrait de la revue Electro Calcul, 28, rue Emile-Landrin, 92-Boulogne - 825 46-20