TECHNICAL DEVELOPMENTS

The Incorporation of Subroutines into a Complete Problem on the NBS Eastern Automatic Computer

The construction of a modest-scale automatically-sequenced electronic digital computer at the National Bureau of Standards is nearing completion. This computer, called the NBS Eastern Automatic Computer (SEAC), is expected to be a useful tool both for numerical computation and for research in numerical analysis. In order to facilitate the exploitation of the SEAC, the Machine Development Laboratory of the Bureau has planned in advance key instruction routines for use on the machine. Groups of these instruction routines will be incorporated as subroutines in instruction programs governing the solution of complete problems. It is the purpose of this paper to show how subroutines prepared in advance of problem solution, selected from a library of permanent subroutines, will be properly inserted by the computer in the instruction program relevant to a problem at hand.

In order to follow this program it will be necessary to understand the principal features of the logical design of the SEAC.

1. Memory.—The present routine is written to fit the initial model of the machine which has 64 acoustic lines, or tanks, each line storing eight "words." A word consists of 48 binary digit positions, of which 45 are used to represent either a number or a four-address command, and the remaining three provide spacing between words. A number, N, of 44 binary digits followed by a sign digit, is stored as an absolute value with the proper sign attached. A plus sign is represented by "0" and a minus sign by "1." The binary point of the number is located between the second and third positions from the left so that $|N| < 4$.

In the operation representation, 10 binary positions are apportioned to each of four addresses, or memory locations. As there are only 512 memory locations in the initial model, nine binary digits are sufficient to specify any address (six digits to indicate the tank number and three digits to indicate the word within that tank). The first binary digit from the left of any address will always be zero and therefore will not be indicated here. The remaining nine binary digits will be represented by three octal digits. In addition, four binary digits represent, in coded form, the operations to be performed, designated herein by capital letters. The 45th digit from the left is again a sign indicator.

The following important feature of the acoustic-line memory should be kept in mind: information sent to an address will replace the previous content of that address.

2. Command Code.—In Table I under the headings of each of the four addresses α, β, γ, and δ the items which are located therein are listed. An address enclosed in parentheses indicates the content of the corresponding memory location; the address of the next command is abbreviated as "n.c."

The 45th binary digit from the left in a command is normally a zero (i.e., a plus sign); when a minus sign is coded in that position, the machine

TABLE I

Operation	α	β	γ	δ	Operation Symbol	Result of Operation
Addition	Augend	Addend	Sum	n.c.	A	$(\gamma) = (\alpha) + (\beta)$
Subtraction	Minuend	Subtrahend	Difference	n.c.	S	$(\gamma) = (\alpha) - (\beta)$
Multiplication (High-order)	Multiplicand	Multiplier	High-order Product	n.c.	M	(γ) = the more significant half of $(\alpha)(\beta)$
Multiplication (Low-order)	Multiplicand	Multiplier	Low-order Product	n.c.	N	(γ) = the less significant half of $(\alpha)(\beta)$
Multiplication (Rounded)	Multiplicand	Multiplier	Rounded Product	n.c.	R	(γ) = the more significant half of $(\alpha)(\beta)$ increased by the first digit of the less significant half of the product
Division	Divisor	Dividend	Quotient	n.c.	D	$(\gamma) = (\beta)/(\alpha)$, unrounded
Logical Transfer	Extractee	Extractor	Altered Number	n.c.	L	Digits in (γ) which correspond to 1's in (β) are replaced by corresponding digits of (α)
Algebraic Comparison	1st Comparand	2nd Comparand	n.c. or	n.c.	C	(γ) = n.c., if $(\alpha) < (\beta)$ (δ) = n.c., if $(\alpha) \geq (\beta)$
Absolute Value Comparison	1st Comparand	2nd Comparand	n.c. or	n.c.	K	(γ) = n.c., if $\lvert(\alpha)\rvert < \lvert(\beta)\rvert$ (δ) = n.c., if $\lvert(\alpha)\rvert \geq \lvert(\beta)\rvert$
Input Order	Any Number	Odd Number	Word from Input	n.c.	T	(γ) = next word from Input medium
Input Order	Any Number	Even Number	Word from Input	n.c.	T	(γ) to $(\gamma + 7)$ = next 8 words from Input medium
Output Order	Any Number	Odd Number	Word to be Recorded	n.c.	P	(γ) = next word on Output medium
Output Order	Any Number	Even Number	Word to be Recorded	n.c.	P	(γ) to $(\gamma + 7)$ = next 8 words on Output medium

will stop automatically after executing the indicated operation. Commands can be manipulated in the arithmetic unit of the machine, since, to this unit, they are indistinguishable from numbers.

3. Timing.—A minor cycle, the time needed for one word to pass a given point in the machine, is equal to 48 microseconds. The time consumed for a given operation, located at address ϵ, can be obtained from the following formulae (where an underscored symbol represents the last octal digit of the indicated address):

1. For operations A, S, and L, $t = (\underline{\epsilon} - \underline{\beta}) + (\underline{\beta} - \underline{\alpha}) + (\underline{\alpha} - \underline{\gamma}) + (\underline{\gamma} - \underline{\delta})$, where 8 minor cycles must be added to each difference which is algebraically less than $+1$.
2. For operations M, N, R, and D, the above sum must be increased by 40 minor cycles if $(\underline{\alpha} - \underline{\gamma}) \geq 5$; otherwise, the sum must be increased by 48 minor cycles.
3. For operations C and K, two timings are possible depending on the result of the comparison; the number of minor cycles is given by $(\underline{\epsilon} - \underline{\beta}) + (\underline{\beta} - \underline{\alpha}) + (\underline{\alpha} - \underline{\gamma})$, if $(\alpha) < (\beta)$, or $(\underline{\epsilon} - \underline{\beta}) + (\underline{\beta} - \underline{\alpha}) + (\underline{\alpha} - \underline{\delta})$, if $(\alpha) \geq (\beta)$. In this case, 8 minor cycles must be added to the first two differences whenever their value is less than $+1$ and to the third difference whenever its value is less than $+2$.

The following table indicates the range of execution time for each command:

Operation	Minimum Time minor cycles	microsec.	Maximum Time minor cycles	microsec.
C, K	4	192	25	1200
A, S, L	4	192	32	1536
M, N, R, D	48	2304	76	3648
T, P	About 2 seconds per word, at present			

4. Modification of Subroutines.—In 1947, SAMUEL LUBKIN considered the problem of modifying subroutines which were to be used in a given problem to fit the available machine storage. He proposed the use of a Base Number Command to simplify the task of adapting any subroutine to its position in the memory. GOLDSTINE & VON NEUMANN[1] have reported on the programming of such adaptation using the codes of the Institute for Advanced Study machine. A similar routine prepared for use on the SEAC is presented herein. Because of the restricted storage capacity of this machine, an effort has been made to pack frequently-used routines in the internal memory as economically as possible. The present routine, in which the assumption is made that n subroutines are to be incorporated within the given program, occupies only $17 + n$ cells. This number does not include, however, four temporary storage cells and the storage for nine constants which belong to a common pool (as explained in the next section).

A general explanation of the method used in the modification of subroutines for insertion in main instruction routines will perhaps be helpful to the reader. Each permanent subroutine is coded as though its first word were located at the address 0 100 000 000 (i.e., the binary equivalent of the octal address 400). All addresses in the subroutine which require modification, as introduced into the computer, are greater than 400. Thus, there will be a "1" present in the second position of every address which must be modified and a zero in the corresponding position of all other addresses which are less than 400.

In the present program, each of the required group of subroutines is inserted into the memory in the final locations it is to occupy. The modifying routine begins at position 050. All addresses of 400 or above contained within the subroutine are modified to fit the actual location of the subroutine in the memory. In substance, the four digits corresponding to the second digit from the left of each address within such words of the subroutine as need modification are extracted. The resulting number is multiplied by d, which represents the difference between the address occupied by the first word of the subroutine under consideration and the number 400 at which address this word was originally coded. The resulting product is added to the original word, and the addresses contained therein are thereby properly modified.

Consider a specific example illustrating this procedure. As stated above the addresses will be written in octal form, although they are stored in binary form. Suppose the following subroutine operation, originally coded at address 400, is placed in the memory at address 070:

Cell No.	α	β	γ	δ	Operation
070	401	046	401	415	A

In order to modify (070), subtract 400 from 070 octally with the result, $d = -310$. Next, extract the second binary digit from the left of each address giving a "1" in the case of α, γ, and δ and a "0" in the case of β. Multiply each extracted digit by $d = -310$, giving -310 in the α, γ, and δ positions of a memory cell and 000 in the β position, and add the result octally to (070). The modified operation is as follows:

Cell No.	α	β	γ	δ	Operation
070	071	046	071	105	A

After all the subroutines in a program are modified, the memory cells occupied by the modifying routine are available for other uses.

5. Conventions Used in Coding.—Before actually presenting the program, it will be necessary to explain some of the conventions used in the coding of this routine. An address enclosed within brackets is used to indicate the fact that the content of that storage location will vary. Braces are used in the ordinary algebraic sense.

The first two memory tanks (i.e., memory locations 000 through 017) are retained for temporary storage in the modification routine; cell 007 contains the instruction to be executed immediately after the routine is completed. Positions 020 through 047 serve as a common pool of frequently-used constants, which all subroutines employ. Of this pool, the present routine makes use of the following constants:

Memory Cell	Content
020	+ Zero
023	400 (in binary form) occupying the α space, zeros occupying the remaining binary positions
027	the number 2^{-8}, representing a unit in the last position of α
031	the number 2^{-22}
034	the number 2^{-28}, representing a unit in the last position of γ
041	the number 2^{-10}
042	$2^{2} - 2^{-8}$, representing 10 units occupying the α space
044	$2^{-18} - 2^{-28}$, representing 10 units occupying the γ space
047	Zero (at the start of the routine only)

The following notations will be used in the program:

Notation	Significance
a_{ij}	the address at which the j-th word of the i-th subroutine is located, where $i = 1, 2, 3, \cdots n$ and $j = 1, 2, 3, \cdots b_i$
b_i	the number of words to be modified in the i-th subroutine
$\overline{w}_{ij}$	the content of cell a_{ij} before modification
w_{ij}	the content of cell a_{ij} after modification
s_k	the binary digit of $\overline{w}_{ij}$ in position k from the left, where $k = 1, 2, 3, \cdots 45$

PROGRAM

Cell No.	α	β	γ	δ	Operation	No. of minor cycles	Result of Operation
050	070	044	055	062	L	14	(055) = 010 013 a_{i1} 061 A
062	070	031	012	063	N	55	(012) = $2^{-18}a_{i1}$ (see footnote 2)
063	041	012	012	067	D	52	(012) = $2^{-8}a_{i1}$
067	012	042	051	064	L	19	(051) = a_{i1} 020 010 066 A
064	012	023	012	051	S	11	(012) = $\{a_{i1} - 400\}2^{-8} = d$
051	[a_{ij}]	020	010	066	A	12	(010) = $\overline{w}_{ij}$

Cell No.	α	β	γ	δ	Operation	No. of minor cycles	Result of Operation
066	010	065	047	054	L	10	$(047) = s_2 2^0 + s_{12}2^{-10} + s_{22}2^{-20} + s_{32}2^{-30} = t$
054	047	012	013	055	M	63	$(013) = td = c$, correction
055	010	013	$[a_{ij}]$	061	A	14	$(a_{ij}) = \bar{w}_{ij} + c = w_{ij}$
061	051	070	052 / 056		K	15/11	Test as to whether b_i modifications have been made.
052	027	051	051	060	A	10	If b_i modifications have not been made, (051) and (055) are stepped up.
060	034	055	055	051	A	15	
056	[071]	020	070	057	A	16	All b_i modifications have been made; $(070) = [\{a_{i1} + b_i - 1\}2^{-8} + a_{i1}2^{-28}]$, where $i = 2, 3, \cdots, (n+1)$.
057	027	056	056	053	A	12	(056) is stepped up.
053	020	070	050 / 007		K	19/20	Test as to whether n subroutines have been modified.

Temporary and Constant Storage

065	$2^0 + 2^{-10} + 2^{-20} + 2^{-30}$
070	$\{a_{11} + b_1 - 1\}2^{-8} + a_{11}2^{-28}$; $[\{a_{i1} + b_i - 1\}2^{-8} + a_{i1}2^{-28}]$
071	$\{a_{21} + b_2 - 1\}2^{-8} + a_{21}2^{-28}$
. . .	. . . .
$067 + n$	$\{a_{n1} + b_n - 1\}2^{-8} + a_{n1}2^{-28}$
$070 + n$	Zero

In this program the commands are arranged in logical sequence. On the input medium, they would be arranged in numerical order of addresses of the memory cells containing them. The time consumed for each instruction to be modified is 6.5 milliseconds.

MDL Staff

NBSMDL

[1] H. H. GOLDSTINE, J. VON NEUMANN, *Planning and Coding of Problems for an Electronic Computing Instrument*, Part II, v. 3, Institute for Advanced Study, Princeton, N. J., 1948 [*MTAC*, v. 3, p. 541–542].

[2] Because the first two digits from the left in the result of low order multiplication on the SEAC are always zero, two instructions are required to put a_{i1} into the α position of memory cell 012.

DISCUSSIONS

A Note on "Is" and "Might Be" in Computers

Recent press reports have aired the disagreement between the "brain" and the "antibrain" factions in the computer fraternity. The term "brain" is a bit fanciful and perhaps smells slightly like commercial advertising; however, I feel that some of the stories given to the press have been more definitely misleading. I would like to enter a plea for careful distinction between facts and fancies by scientific people who write and speak for the general public.

A specialist in one field must be particularly careful in talking to the public about matters not in his own field of specialization. A careless or exaggerated remark made to experts in a field will do little harm because