BULLETIN DE L'ACADÉMIE
POLONAISE DES SCIENCES
Série des sciences techniques
Volume VII, No. 12 — 1959

COMPUTERS

An Electronic Digital Computer Based on the "— 2" System

by

Z. PAWLAK

Presented by P. SZULKIN on October 29, 1959

Introduction

The present paper describes the experimental digital computer denoted as EMC built in 1958 at the Warsaw Technical University. The fundamental concept of this computer as well as the draft logical schemes were elaborated in the Mathematical Institute of the Polish Academy of Sciences, in 1956.

The computer works at the rate of 100 operations per second, approximately, the drum memory has a capacity of 500 words of 36 bits. In the future an extension of memory is envisaged to a capacity of 4,000 words. The computer uses one-address type of instructions. As an input-output device a standard tele-type equipment is used. Dynamic technique is applied. The total number of electronic tubes is 350, approximately.

The computer is destined for designing offices, scientific institutes and universities. The computer construction was also undertaken with the aim of ascertaining the possibilities of application of a negative base number representation system to digital computers and certain experiments in the field of organization of digital computers.

Numbers and orders

Every real number may be represented in the form

$$x = \sum_{i=-\infty}^{m} (-1)^{F(i)} C_i g^i,$$

where m, C_i, g are integers such that $|g| > 1$, $0 \leqslant |C_i| \leqslant |g| - 1$, and $F(i)$ is a function defined on natural numbers.

If $F(i) = 0$, $g = -2$ and $0 \leqslant C_i \leqslant 1$ or if $F(i) = i$, and $g = 2$, $0 \leqslant C_i \leqslant 1$, we shall obtain the expansion given in [1], [2] *).

*) An identical principle of representing negative numbers was also given by M. V. Wilkes; it has not been published, however.

In the computer described here 34 digit numbers are used which are represented in the following form:

$$x = \sum_{i=-34}^{-1} C_i(-2)^i,$$

where $0 \geqslant C_i \geqslant -1$.

It can easily be seen that $-1/3 < x < 2/3$. The asymmetric interval presents, however, no difficulty in programming.

The order is composed of the address part (12 least significant positions) and the operational part (20 positions). The remaining positions are not used.

The operational part has the form:

Xo	*Do*	*Ad*	*Ro*	*Lo*	*Ns*	*Zl*	*Ao*	*Al*	*Ap*	*Nd*	*Sg*	*Wa*	*Mn*	*Xz*	*Dz*	*Az*	*Lz*	*Zp*	*Rz*

The meaning of each position is given in the next paragraph.

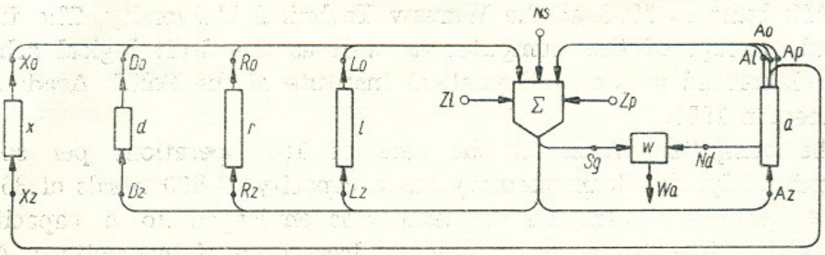

Fig. 1

Organization of the computer

A simplified organization diagram of the machine is shown in the figure.

The diagram shows the *registers*, *ways* and *arithmometer*.

The registers are as follows:

x_i — memory registers ($-1365 \leqslant i \leqslant 2730$),
a — accumulator register,
r — order register (or the multiplicand register),
l — order counter register (or the multiplier register),
d — input-output register,
s — control register,
$C\alpha$ denotes the number in the register α.

The address part of the content of a will be denoted by $\{C a\}$; the i-th digit of $C a$ is denoted by a^i.

The arithmometer is composed of an *adder* Σ and a device W for *sign* and *overflow registration*.

The adder operates according to the formulae

(1) $$c = (-1)^{Zl} \cdot a + (-1)^{Zp} \cdot b + Ns.$$

Zl, Zp, Ns may assume the values: 0, 1; *a, b* are the arguments, *c* — the result.

The device *W* computes the functions $Sg\,x$ and $Nd\,a$, defined thus

$$Sg\,x = \begin{cases} 0 \text{ if } x < 0, \\ 1 \text{ if } x \geqslant 0, \end{cases} \tag{2}$$

$$Nd\,a = a^0. \tag{3}$$

The following ways exist in the computer: *Xo, Xz, Ro, Rz, Po, Dz, Lo, Lz, Ao, Al, Ap, Az, Sg, Nd.* Every way may be closed or opened. If the way from the register α is opened, we say that the content of the register α is read out; if the way to the register α is opened, we shall say that a new value is recorded in the register α. If the way from α to β is opened we write $\alpha\beta$. If $\alpha\beta$, then $C\alpha = C\beta$.

The control register has 20 positions denoting the following elementary operations:

Xo	— do not read the memory register,
Do	— read out the input-output register,
Ad	— the address part,
Ro	— read out the order register,
Lo	— read out the order counter register,
Ns	— successor,
Zl	— do not change the sign of the left-hand argument,
Ao	— read out the accumulator register,
Al	— shift the content of the accumulator register by one position to the left,
Ap	— shift the content of the accumulator register by one position to the right,
Nd	— examine the overflow in the accumulator register,
Sg	— examine the sign,
Wa	— conditional order,
Mn	— multiplication,
Xz	— record in the memory register,
Dz	— record in the input-output register,
Az	— record in the accumulator register,
Lz	— record in the order counter register,
Zp	— do not change the sign of the right-hand argument,
Rz	— do not record in the order register.

The ways and the positions of the register (elementary operations) are denoted by the same symbols. This does not, however, lead to misunderstandings.

If a position in a register *s* has the value of 1, this denotes the performance of the elementary operation corresponding to this position; in case the value is 0, it means that the operation is not performed.

For instance,

$Ro = 1$ denotes the opening of the way from the register *r*,

$Xo = 0$ — the opening the way from the memory register *x* (with the number given in the address part of the order).

The meaning of *Zl, Zp, Ns* follows from Eq. (1).

Ar means that the computer operates with 12 digit numbers (the address part).

During multiplication the multiplicand is in the register r, the multiplier — in the register l, the product: the head — in the register a — and the tail are in the register l *).

The meaning of *Wa* will be explained in the description of the operation cycle of the computer. The meaning of the remaining elementary operations is obvious.

If in the register s we have a number, e.g. 0000 0000 0000 0000 0000, this means that the contents of the register x_i should be read out and recorded in the register r, $i = \{Cr\}$.

The number 1001 0011 0000 0000 1111 denotes the summation of the contents of the registers r and a and recording of the result in the registers a and l.

Symbolic form of orders

The symbolic form of the orders consists of three groups of symbols:

a) the letters x, r, d, l, a denoting the registers;

b) the symbols of operations of one and two arguments:

N	— successor,
Nd	— overflow,
Sg	— sign,
L	— accumulator shift to the left,
P	— „ „ „ „ right,
$+$	— register sign unchanged,
$-$	— „ „ changed,
$\cdot$	— multiplication;

c) special symbols,

$\#$	— the address part,
?	— condition,
$\cdot$	— stop.

In addition, parantheses are used in multiplying orders. The orders will be divided into:

Ps	— transfer and adding of orders,
Pa	— accumulator shifting orders,
Zr	— clear orders,
Sn	— register sign examination orders,
Nd	— accumulator overflow examining orders,
Wa	— conditional orders,
St	— orders with stop,
Ad	— orders concerning address parts,
Mn	— multiplying orders.

Instead of writing: a is the accumulator shifting order, for instance, we shall write $a \epsilon$ ***Pa***. Also instead of the phrase "If ...then..." we shall use

*) In its present form the computer has a separate register m for the multiplier.

the symbol $\supset$. Similarly, $\alpha \in \mathscr{L}$ means that α is one of the letters x, r, l, d, a. Instead of writing: α is an order (it belongs to one of the above groups), we write $\alpha \in \boldsymbol{R}$.

I. Transfer and adding orders (*Ps*)

a) If $\alpha \in \mathscr{L}$ and $\beta = \beta_1, \ldots, \beta_n \, (1 \leqslant n \leqslant 5)$, $\beta_i \in \mathscr{L}$, $\beta_i \neq x$, $\beta_i \neq \beta_j$, then $\alpha\beta \in \boldsymbol{Ps}$;

b) $\alpha \in \boldsymbol{Ps} \supset N\alpha \in \boldsymbol{Ps}$;

c) $\alpha \in \boldsymbol{Ps} \supset -\alpha \in \boldsymbol{Ps}$;

d) $\alpha \in \boldsymbol{Ps} \supset \pm a \pm \alpha \in \boldsymbol{Ps}$.

For instance,

xal	denotes	$Cx \rightarrow a, l$ *),
$Nxar$	,,	$(Cx + 1) \rightarrow a, r,$
$-xar$	,,	$-Cx \rightarrow a, r,$
$-Nrda$	,,	$(-Cr + 1) \rightarrow d, a,$
$-x + aa$	,,	$(-Cx + Ca) \rightarrow a,$
$r - Nalr$	,,	$(Cr - Ca + 1) \rightarrow l, r.$

II. Accumulator shifting orders (*Pa*)

a) If $\beta = \beta_1, \ldots, \beta_n \, (1 \leqslant n \leqslant 5)$, $\beta_i \in \mathscr{L}$, $\beta_i \neq x$ and $\beta_i \neq \beta_j$, then $La\beta \in \boldsymbol{Pa}$ and $Pa\beta \in \boldsymbol{Pa}$;

b) $\alpha \in \boldsymbol{Pa} \supset N\alpha \in \boldsymbol{Pa}$;

c) $\alpha \in \boldsymbol{Pa} \supset -\alpha \in \boldsymbol{Pa}$;

d) If $\alpha \in \boldsymbol{Pa}$ and $\beta = \beta_1, \ldots, \beta_n$, $(1 \leqslant n \leqslant 5)$, $\beta_i \in \mathscr{L}$, $\beta_i \neq x$, $(2 \leqslant i \leqslant 5)$, $\beta_1 \neq a$, $\beta_i \neq \beta_j$, then $\alpha \pm \beta \in \boldsymbol{Pa}$.

For instance,

Laa	denotes	$Ca \cdot (-2) \rightarrow a,$
Paa	,,	$Ca/(-2) \rightarrow a,$
$NLaar$	,,	$(Ca \cdot (-2) + 1) \rightarrow a, r,$
$-NPaard$	,,	$(-Ca/(-2) + 1) \rightarrow a, r, d,$
$-La + xar$	,,	$(-Ca \cdot (-2) + Cx) \rightarrow a, r.$

*) The symbol $\rightarrow$ denotes transfer.

III. Clearing orders (*Zr*)

If $\alpha = a_1, \ldots, a_n (1 \leqslant n \leqslant 5)$, $a_i \in \mathcal{L}$, and for $i > 1$, $a_i \neq x$,

then $0\alpha \in \boldsymbol{Zr}$ and $1\alpha \in \boldsymbol{Zr}$.

For instance,

$0a$	denotes	$0 \rightarrow a,$
$0arl$	,,	$0 \rightarrow a, r, l,$
$1xa$	,,	$1 \rightarrow x, r.$

IV. Sign examination orders (*Sn*)

a) $\alpha \in \mathcal{L} \supset Sg\,\alpha \in \boldsymbol{Sn}$;

b) $\alpha, \beta \in \mathcal{L}$ and $\alpha \neq \beta \supset Sg\,(\pm\,\alpha \pm \beta) \in \boldsymbol{Sn}$;

c) $\alpha \in \boldsymbol{R} \supset Sg\,\alpha \in \boldsymbol{Sn}$.

For instance,

$Sg\,a$	denotes	$Sg\,Ca \rightarrow w,$
$Sg\,(x + a)$	,,	$Sg\,(Cx + Ca) \rightarrow w,$
$Sg\,(-La + xar)$	,,	$(-Ca \cdot (-2) + Cx) \rightarrow a, r;$ $Sg\,(-Ca \cdot (-2) + Cz) \rightarrow w.$

V. Overflow examination (*Nd*)

a) $Nd\,a \in \boldsymbol{Nd}$; b) $\alpha \in \boldsymbol{R} - \boldsymbol{Sg} \supset Nd\,\alpha \in \boldsymbol{Nd}$.

For instance,

$Nd\,a$ denotes $a^0 \rightarrow w.$

VI. Conditional orders (*Wa*)

$$\alpha \in \boldsymbol{R} \supset \alpha ? \in \boldsymbol{Wa}.$$

For instance,

$$Sg\,(-La + xar)\,?$$

VII. Orders concerning the address part (*Ad*)

$$\alpha \in \boldsymbol{R} - \boldsymbol{Pa} \supset \#\,\alpha \in \boldsymbol{Ad}.$$

For instance,

$\#\,x + aa$ denotes $\{Cx\} + \{Ca\} \rightarrow a.$

VIII. Stop orders (*St*)

$$\alpha \in \boldsymbol{R} \supset \alpha . \in \boldsymbol{St}.$$

For instance,

$$Sg(-La + xar)?.$$

The full stop in the order means that the computer stops before the order is executed.

IX. Multiplication orders (*Mn*)

For these orders we can also give a general scheme, however, in view of the small number of these orders and for the sake of clearness the more important ones will be written in full form:

$$a \cdot x, \quad -a \cdot x, \quad a \cdot (a + x), \quad a \cdot (a - x),$$
$$(a + x)^2, \quad (a - x)^2, \quad a^2; \quad x^2.$$

These orders may also be conditional, that is, they may, for instance, take the form

$$a(a - x)?$$

X. Composite orders

For certain orders, if $\alpha \in \boldsymbol{R}$ and $\beta \in \boldsymbol{R}$, then $\alpha ; \beta \in \boldsymbol{R}$. It is difficult to give a general scheme of composite orders; we shall therefore confine ourselves to a few examples:

$$ax; 0a, \quad ax; Nra?, \quad ax; Naa, \quad r + al; 0a?, \quad Laa; \# Sg\, Nrr?$$

It is not difficult to read these orders. Thus, for instance, the last represents accumulator shifting to the left through n positions. The classes described do not exhaust all the possible orders. It would be of interest to give a general scheme of the list of orders, exhausting all the possible well formed orders; for a given organization, however, this seems to be difficult.

The list of orders

The described computer has not a fixed list of orders. This list may be, within certain limits, arbitrary. Below, we give an example of a list of orders which may be realized in the computer:

1. $-aa$ — accumulator sign change,
2. $x + aa$ — addition,
3. $x - aa$ — accumulator subtraction,
4. $-x + aa$ — memory subtraction,
5. Laa — accumulator shifting to the left,
6. Paa — ,, ,, ,, ,, right,
7. $L_n aa$ — ,, ,, by n positions to the left,
8. $P_n aa$ — ,, ,, ,, ,, ,, ,, ,, right
9. $x \cdot a$ — multiplication,

10.	x^2	— raising to powers,
11.	xa	— transfer from the memory,
12.	$-xa$	— ,, ,, ,, ,, with change of sign,
13.	$0a$	— clearing of the accumulator,
14.	ax	— transfer to the memory,
15.	$ax; 0a$	— ,, ,, ,, ,, with clearing of the accumulator,
16.	$\# rl\,?$	— conditional jump,
17.	$Sg\,a$	— sign examination,
18.	$Nd\,a$	— overflow examination,
19.	xr	— taking of order,
20.	$x + ar$	— taking of order with a modification,
21.	$\# rl$	— unconditional jump,
22.	$\# ra$	— taking a parameter,
23.	$\# r + aa$	— address modification,
24.	$\# Sg\,Nra$	— cycle counter,
25.	$\# ax$	— address transfer,
26.	da	— tape reading,
27.	$\# rd$	— printing.

Operation cycle

The operation cycle is composed of three steps:

a) execution of order,

b) taking the address of the next order and increasing the contents of the order counter by 1 (the order $Nllr$),

c) taking the order (order xr).

If r is a conditional order ($Wa = 1$) and $Cw = 1$, r is not executed: if $Cw = 0$, r is performed.

Pre-input program

The pre-input program causes the sending, without modification of the words from the tape to successive places of the memory starting with the place with number pre-set in the order register.

The pre-input program has the form:

$$Nd\,(L_4\,aa + da)\,?\,,$$
$$ax; Nrr; 0a\ .$$

The first order causes the completing of the word entering from the tape into the accumulator. The second order causes the sending of the completed order to the successive memory, and the modification of the transfer order, and clearing of the accumulator. Both orders constitute a fixed part of the computer in the form of a diode matrix.

The electronic systems were designed by Mr. Łazarkiewicz and Mrs. Wieruszowa. Detailed logical schemes were made by Mr. Balasiński; the principles of programming were elaborated by Mr. Kulikowski, the mechanical construction of the memory was designed by Mr. Künel and Mr. Terlecki. The input and output system was designed by Mr. Kacz-

marewicz. Assembling schemes were made by Mr. Braun, who also put the computer in operation.

Most of the assembling work was done by Mr. Wardak. The computer was built under the direction of Mr. Łazarkiewicz.

The author wishes to express his gratitude to the Director of the Institute, Prof. Kiliński for making possible the building of the computer.

DEPARTMENT OF TELE- AND RADIOPHONIC CONSTRUCTIONS, WARSAW TECHNICAL UNIVERSITY
(ZAKŁAD KONSTRUKCJI TELE- I RADIOFONII POLITECHNIKI WARSZAWSKIEJ)

REFERENCES

[1] Louis B. Wadel, *Negative base number systems,* IRE. Trans. Electronic Comput. **6** (1957), 123.

[2] Z. Pawlak, A. Wakulicz, *Use of expansions with a negative basis in the arithmometer of a digital computer,* Bull. Acad. Polon. Sci., Sér. sci. math., astr. et phys., **5** (1957), 233.