Chapter 15

Instruction logic of the Soviet Strela (Arrow)[1]

John W. Carr III

A typical general purpose digital computer using three-address instruction logic is the Strela (Arrow) constructed in quantity under the leadership of Iu. Ia. Basilewskii of the Soviet Academy of Sciences, and described in detail by Kitov [1956]. This computer uses a (35, 6, 0)[2] binary floating point number system. Its instruction word, of 43 digits, contains a six-digit operation code, and three 12-digit addresses, with one breakpoint bit. In octal notation, two digits represent the operation, four each the addresses, and one bit the breakpoint. This machine operates with up to 2048 words of high-speed cathode ray tube storage.

Input-output is ordinarily via punched cards and punched paper tape. A "standard program library" is attached to the computer as well as magnetic tape units (termed "external accumulators" below). *Note.* This computer is different from both the BESM described by Lebedev [1956] and the Ural reported by Basilewskii [1957]. Apparently, it is somewhat lower in performance than BESM.

Since all arithmetic is ordinarily in floating point, "special instructions" perform fixed point computations for instruction modifications.

Ordinarily instructions are written in an octal notation, but external to the machine operation symbols are written in a mnemonic code. The two-digit numerals are the octal instruction equivalent.

Arithmetic and logical instructions

01. $+ \ \alpha \ \beta \ \gamma$. Algebraic addition of (α) to (β) with result in γ.

02. $+_1 \ \alpha \ \beta \ \gamma$. Special addition, used for increasing addresses of instructions. The command (α) or (β) is added to the number (β) or (α) and the result sent to the cell with address γ. As a rule, the address of the instruction being changed corresponds to the address γ.

03. $- \ \alpha \ \beta \ \gamma$. Subtraction with signed numbers. From the number (α) is subtracted the number (β) and the result sent to γ.

04. $-_1 \ \alpha \ \beta \ \gamma$. Difference of the absolute value of two numbers $|(\alpha)| - |(\beta)| = (\gamma)$.

05. $\times \ \alpha \ \beta \ \gamma$. Multiplication of two numbers (α) and (β) with result sent to γ.

06. $\wedge \ \alpha \ \beta \ \gamma$. Logical multiplication of two numbers in cells α and β. This instruction is used for extraction from a given number or instruction a part defined by the special number (β).

07. $\vee \ \alpha \ \beta \ \gamma$. Logical addition of two numbers (α) and (β) and sending the result to cell γ. This instruction is used for forming numbers and commands from parts.

10. Sh $\alpha \ \beta \ \gamma$. Shift of the contents of cell α by the number of steps equal to the exponent of the (β). If the exponent of the (β) is positive then the shift proceeds to the left, in the direction of increasing value; if negative, then the shift is right. In addition, the sign of the number, which is shifted out of the cell, is lost.

11. $-_2 \ \alpha \ \beta \ \gamma$. Special subtraction, used for decreasing the addresses of instructions. In the cell α is found the instruction to be transformed, and in cell β the specially selected number. Ordinarily addresses α and γ are identical.

12. $\not\equiv \ \alpha \ \beta \ \gamma$. Comparison of two numbers (α) and (β) by means of digital additions of the numbers being compared *modulo* two. In the cell γ is placed a number possessing ones in those digits in which inequivalence results in the numbers being compared.

Control instructions

13. C $\alpha \ \beta$ **0000.** Conditional transfer of control either to instruction (α) or to instruction (β), depending on the results of the preceding operation. With the operations of addition, subtraction, and subtraction of absolute values, it appraises the sign

[1] In E. M. Crabbe, S. Ramo, and D. E. Wooldridge (eds.), "Handbook of Automation, Computation, and Control," vol. 2, chap. 2, pp. 111–115, John Wiley & Sons, Inc., New York, 1959.

[2] Carr's triplet notation for: fractional significant digits, digits in exponent, and digits to left of radix point.

of the result: for a positive or zero result it transfers control to the command (α), for negative results to the command (β).

The result of the operation of multiplication is dependent on the relationship to unity. Transfer is made to the command (α) in the case where the result is greater than or equal to one, and to command (β), if it is smaller than one.

For conditional transfer after the operation of comparison, transfer to the instruction (α) is made in the case of equality of binary digits, and to (β) when there is any inequivalence.

After the operation $\wedge$ (logical sequential multiplication) the conditional transfer command jumps to the instruction (α) when the result is different from zero, and to instruction (β) when it is equal to zero.

A forced comparison is given by

C α α 0000

The third address in this command is not used and in its place is put zero.

14. I-O α 0000 0000. This instruction is executed parallel with the code of the other operations, and guarantees bringing into working position in good time the zone of the external accumulator (magnetic tape unit) with the address α.

15. H 0000 0000 0000. This instruction executes an absolute halt.

Group transfer instructions

Special instructions for group transfer serve for the accomplishment of a transfer of numbers to and from the accumulators. In the second address in these instructions stands an integer, designating the quantity of numbers in the group which must be transferred. Group transfers always are produced in increasing sequence of addresses of cells in the storage.

16. T_1 0000 n γ. The instruction T_1 guarantees transfer from a given input unit (with punched cards, perforated tape, etc.) into the storage. In the third address γ of the instruction is indicated the initial address of the group of cells in the storage where numbers are to be written. With punched paper tape or punched cards the variables are written in sequence, beginning with the first line.

17. T_2 0000 n γ. The instruction T_2 guarantees transfer of a group of n numbers from an input unit into the external accumulator in zone γ.

20. T_3 α n γ. This instruction guarantees a line-by-line sequence of transfers of n numbers from zone α of the external accumulator into the cells of the storage beginning with the cell with address γ.

21. T_4 α n 0000. This instruction guarantees the transfer to the input-output unit (to punched paper tape or punched cards) of a group of n numbers from the storage, beginning with address α. The record on punched paper tape or punched cards as a rule will begin with the first line and therefore a positive indication of the addresses of the record is not required.

22. T_5 α n γ. Instruction T_5 guarantees transfer of a group of n numbers from one place in the storage with initial address α into another place in the storage with initial address γ.

23. T_6 α n γ. Instruction T_6 guarantees transfer of a group of n numbers from the storage with initial address α into the external accumulator with address γ.

24. T_7 α n 0000. Instruction T_7 serves for transfer of n numbers from the zone of the external accumulator with address α into the input-output unit.

Instructions T_2 and T_7 cannot be performed concurrently with other machine operations.

Standard subroutine instructions

Certain instructions in the Strela, although written as ordinary instructions, are actually "synthetic" instructions which call on a subroutine for computation of the function involved. The amount of machine time (number of basic instruction cycles) for an iterative process depends on the required precision of the computed function. The figures given below are based on approximately ten-digit decimal numbers with desired precision one in the tenth place.

25. D α β γ. This standard subroutine serves for execution of the operation of division: The number (α) is divided into the number (β) and the quotient is sent to cell γ.

The actual operation of division is executed in two steps: the initial obtaining of the value of the inverse of the divisor, by which the dividend is then multiplied. The computation of the inverse is given by the usual Newton formula, originally used with the EDSAC [Wilkes et al., 1952].

$$y_{n+1} = y_n(2 - y_n x)$$

For $x = d \cdot 2^p$, where $\frac{1}{2} < d < 1$, the first approximation is taken as 2^{-p}. The standard subroutine takes 8 to 10 instructions and can be executed in 18–20 machine cycles (execution time for one typical command).

26. $\sqrt{\ }$ α 0000 γ. This instruction guarantees obtaining the value $\sqrt{x}$ from the value $x = (\alpha)$ and sending the result to cell γ. Initially $1/\sqrt{x}$ is computed by the iteration formula

$$y_{n+1} = \tfrac{1}{2}y_n(3 - xy_n^2)$$

where the first approximation is taken as

$$y_0 = 2^{[p/2]}$$

the bracket indicating "integral part of." After this the result is multiplied by x to obtain $\sqrt{x}$. This standard subroutine contains 14 instructions and is executed in 40 cycles.

27. e^x α 0000 γ. This instruction guarantees formation of e^x for the value $x = (\alpha)$ and sending the result to cell γ. The computation is produced by means of expansion of e^x in a power series. The standard subroutine contains 20 instructions and is executed in 40 cycles.

30. ln x α 0000 γ. This instruction guarantees formation of the function $\ln x$ for the value $x = (\alpha)$ and sending the result to location γ. Computation is produced by expansion of $\ln x$ in series. The subprogram contains 15 instructions and is executed in 60 cycles.

31. sin x α 0000 γ. This instruction guarantees execution of the function $\sin x$ and sending the result to location γ. The computation is produced in two steps: initially the value of the argument is translated into the first quadrant, then the value of the function is obtained by a series expansion. The subroutine contains 18 instructions and is executed in 25 cycles.

32. DB α n γ. This instruction performs conversion of a group of n numbers, stored in locations $\alpha, \alpha + 1, \ldots$ from binary-coded decimal into binary and sending of the result to locations $\gamma, \gamma + 1, \ldots$. The subroutine contains 14 instructions and is executed in 50 cycles (for each number).

33. BD α n γ. This instruction performs the conversion of a group of n numbers stored in locations $\alpha, \alpha + 1, \ldots$ from the binary system into binary-coded decimal and sends them to locations $\gamma, \gamma + 1, \ldots$. The subroutine contains only 30 instructions and is executed with 100 cycles (for each number).

34. MS α n γ. This is an instruction for storage summing. This instruction produces the formal addition of numbers, stored in locations beginning with address α, and the result is sent to location γ. Numbers and instructions are added in fixed point. This sum may be compared with a previous sum for control of storage accuracy.

References

BasiI57; KitoA56; LebeS56; WilkM52.