SPECIAL-PURPOSE TERNARY COMPUTER FOR DIGITAL FILTERING

Tatsuo HIGUCHI and Hisamitsu HOSHI

Department of Electronic Engineering, Faculty of Engineering
Tohoku University, Aoba, Aramaki, Sendai, Japan

ABSTRACT

Real-time digital filters which have several advantages over analog filters are known as a very important digital-signal processor. This paper provides a summary of the design features of a special-purpose microprogram-controlled ternary computer suited for realizing the real-time digital filters by programming. Many advantages of the ternary number system are effectively employed for the ternary computer in hardware and software. Particular emphasis is placed on the use of the ternary adder with three inputs and the microprogram structure based on ternary logic. An example of the use of the ternary computer proposed for the real-time digital filters is presented to illustrate how microprogramming and user programming can easily be done and execution steps can substantially be reduced, unlike the corresponding binary computer.

I. INTRODUCTION

It is well known that digital filtering is a fundamental signal-processing operation of great utility in many branches of science and technology, and digital filters for on-line, real-time applications have several advantages over analog filters [1-4]. Digital-filter computational algorithms obtained by merely a suitable combination of standard mathematical operations such as addition, subtraction, multiplication and delay may be carried out by general-purpose digital computers or special-purpose hardwares. For some applications, however, the formers are too expensive and the latters are inflexible. One solution to these problems is to use microprocessors [5,6].

On the other hand, it is said that a symmetric ternary-number representation which has many advantages can simplify the construction of digital systems [7-10]. The authors have proposed a ternary digital filter as a special-purpose hardware based on the symmetric ternary-number representation using digits +1, 0 and -1, and established that the ternary digital filter thus obtained is suited for carrying out digital filtering as compared with the case of the binary digital filter [11]. In the ternary digital filter as the special-purpose hardware, however, arbitrary characteristics and structures have not been obtained because of inflexibility. Therefore, it is expected that arbitrary digital filters may be realized by programming on the special-purpose ternary computer.

The SETUN and TERNAC are known as the full scale implementations of ternary computers [12]. However, little is known as to how the advantages of the symmetrical ternary number system are applied to the digital-filter oriented ternary computer in hardware and software.

The purpose of this paper is to design and implement a special-purpose microprogram-controlled ternary computer which is suitable for digital filtering. In the computer, an adder with three inputs is used considering the many-input addition which is frequently required for digital filtering. The adder consists of only one stage of ternary full adders unlike the case of the binary adder. The feature can effectively be employed for detecting overflow in the adder which causes overflow oscillation in digital filters [13]. In multiplication, microprogram steps and execution steps needed are reduced utilizing the following advantages : the economy of digits, the unnecessity of a special rounding off technique and a sign conversion, and 3-valued branching. Moreover, many microinstruction fields are defined by one trit. This can simplify a control unit in the ternary computer because the microinstruction fields are carried out by only one output signal.

The practical implementation is realized using the ternary threshold gates as a basic building block [11] except for the ternary ROM and RAM realized based on the binary ROM and RAM. An example for the cannonic realization of a 2nd-order digital filter is presented to demonstrate how microprogramming and user programming can easily be done. In this example, it is established that the required execution steps can substantially be reduced as compared with the case of the corresponding binary computer.

II. ARCHITECTURE

In this section, the architecture is considered as the machine as seen by the programmer. The machine is a special-purpose computer operating with 10-trit (ternary digit) ternary numbers corresponding to 16-bit binary numbers. A 10-trit instruction word consists of a 4-trit operator and a 6-trit operand.

Instruction Set

In general, many differences between the binary and ternary instructions will not be recognized in the machine-language level. However, the instruction set used here includes several features not found in other binary computers. The most frequently used instructions are ADD, MULTIPLE, LOAD, STORE and JUMP. In addition operation, three-input addition which is frequently required for digital filtering can be done by only one stage of the ternary full adders, so that machine-language programming becomes more elegant and simple in detecting overflow which is very important for realizing real-time digital filters. Fruthermore, both addition and subtraction are programmed without considering signs. In multiplication operation, the problem of rounding off noise is an important design consideration [1-4]. In machine-language programming of the ternary computer, special rounding off considerations are unnecessary.

Address Space

By using 6 trits, one can directly address 3^6 = 729 memory locations. The main memory is composed of 729 words, each of them having 10 trits. In general, digital filtering can be carried out without large capability of memory. Therefore, the digital filters will easily be implemented using the special-purpose computer. In the cannonic form for the nth-order digital filter, the number of memory locations required is shown in Table 1.

Table 1

Items	Number of memory locations
Multiplier coefficients	2n + 1
Variable data	4n + 2
Programming area	
Multiplication	3(2n+1)
3-input addition	3n
Delay	2n
Total	17n + 6

Multiplier coefficients determine the characteristics of digital filter obtained by programming. The number of memory locations for the variable data corresponds to the number of nodes in the block diagram representing the digital filters. Under the present capability of memory, this ternary computer can implement about 42nd-order digital filters.

III. ORGANIZATION

Fig. 1 shows a block diagram of the ternary computer proposed here. The structure of the computer consists of three main units : the register and arithmetic unit, the main memory, and the control unit.

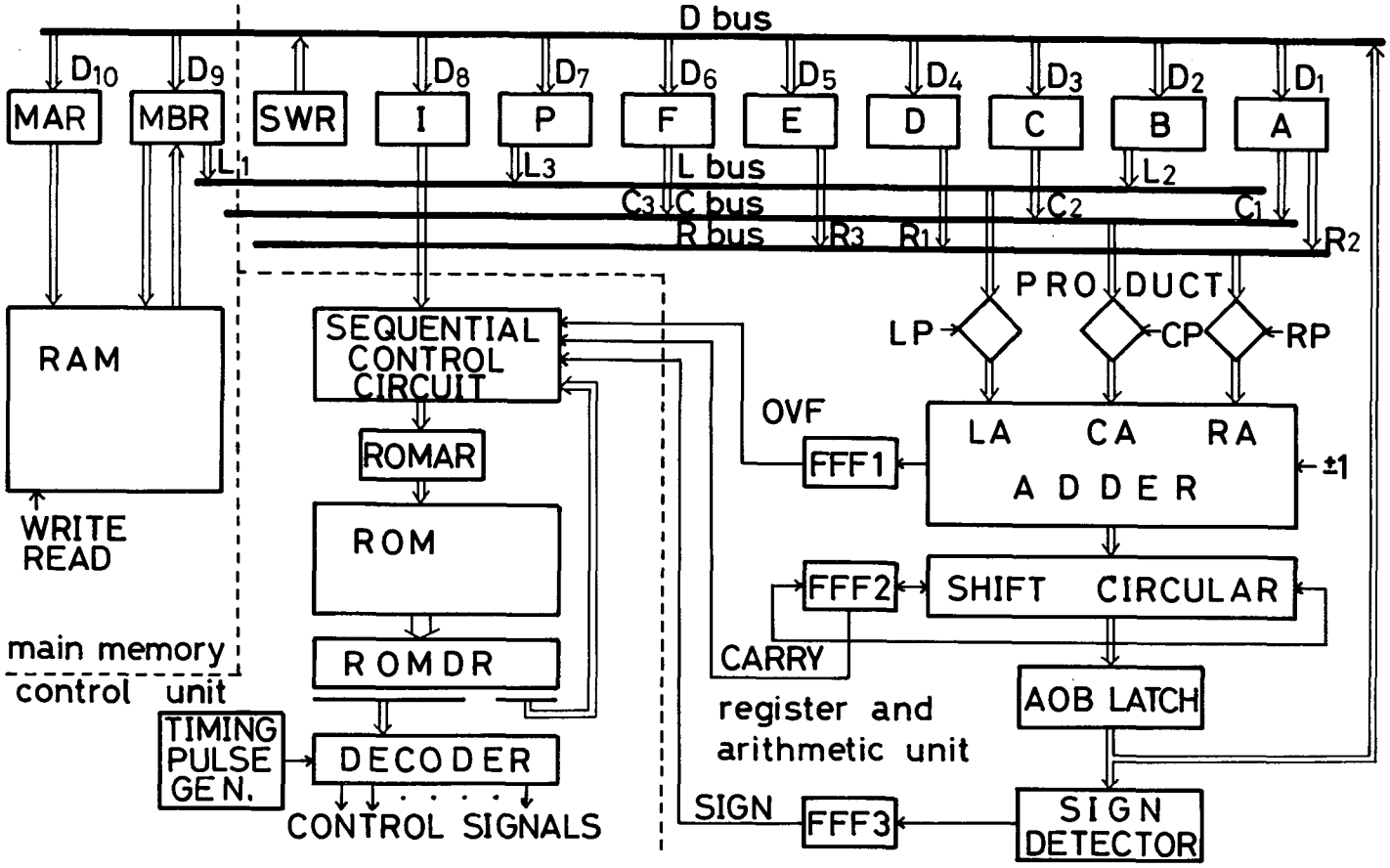


Fig. 1

Register and Arithmetic Unit

This unit has several ternary registers which consist of six 10-trit general-purpose registers (A-F), one 6-trit register for program counter (P), and one 4-trit instruction register (I). Signal fed from the register to the bus line L, C, or R is multiplied by +1, 0 or -1 every trit at the product circuit where the product operation is defined by Table 2, and then is entered to an adder with three inputs. Addition and subtraction can easily be done using the product circuit.

In carrying out digital filtering, many-input addition is frequently required. The three-input parallel adder which consists of ten ternary full adders with four inputs including a carry input as shown in Table 3 is very useful for digital filtering. Unlike the case of the binary adder [14], it should be noted that the three-input addition can be done by only one stage of the ternary full adders. In the adder, a control signal +1, 0 or -1 is entered to the ternary full adder which has the smallest weight as a carry input. As a result, the up-down counting can easily be done by the control signal.

Table 2

Y \ X	1	0	-1
1	1	0	-1
0	0	0	0
-1	-1	0	1

Table 3

$\sum_{i=1}^{3}$ X_i + C	4	3	2	1	0	-1	-2	-3	-4
Carry	1	1	1	0	0	0	-1	-1	-1
Sum	1	0	-1	1	0	-1	1	0	-1

The output of the adder is transfered to a latch circuit for the adder output buffer (AOB) through a shift circular circuit which operates as left and right shifting. In addition, a sign detector, and flags for overflow, carry and sign (FFF1, FFF2, and FFF3) are included. By testing the values stored in these 1-trit ternary registers, 3-valued branching can be executed by one microprogram step. A switch register (SWR) is used for entering program and data into the ternary computer.

Main Memory

The main memory is composed of 3^6 = 729 words, each of them having 10 trits. One word can be used to store one ternary number or one instruction of the programs, coded in ternary form. Some words are allocated to the I/O registers. The memory address register (MAR) and the memory buffer register (MBR) are 6-trit and 10-trit widths, respectively. The content of the memory location specified by the MAR is read-out to the MBR. On the other hand, the content of the MBR is stored into the memory location specified by the MAR.

Control Unit

This computer which may be classified as a horizontal machine employs the microprogrammed control. The microinstruction width is 23 trits. The ternary ROM which serves as the store for the microprogram is composed of 3^5 = 243 words, each of them having 23 trits. The content of the memory location specified by the read-only memory address register (ROMAR) is read-out into the read-only memory data register (ROMDR) every machine cycle. The ROMAR and ROMDR are 5-trit and 23-trit widths, respectively.

IV. MICROPROGRAMMING

Microinstruction Format

The microinstruction format is depicted in Fig. 2. The microinstruction as defined here requires 23 trits. The microinstruction fields have two groups : the static and sequential control fields.

Static Control Fields												Sequential Control Fields				
W/R	SL	SC	SR	LP	CP	RP	±1	CC	SH	C	D	BT	FB	NA		

Fig. 2

The static control field can be devided into the following mutually exclusive fields : W/R, SL, SC, SR, LP, CP, RP, ±1, CC, SH, C, D. By using ternary logic, many fields are defined by only one trit, so that a control unit can be simplified.

The sequential control fields specify the address of the next microinstruction. If the conditional branch test is required, then the operation for modifying the sequencing scheme based on the results of the branch test specified is carried out. Three fields are used here : BT, FB, NA.

Static Control Field

(1) Write/Read field, W/R (1 trit)

This field controls the main memory :

1	Write
0	No operation
-1	Read.

(2) Data selection for the L, C, and R bus lines, SL, SC, and SR (1 trit for each)

These fields select the registers for the L, C, and R bus lines.

(3) Product fields, LP, CP, and RP (1 trit for each)

These fields control the product circuit and transfer the data source to the inputs of the adder :

LP	CP	RP
1 L bus→LA	1 C bus→CA	1 R bus→RA
0 "0" →LA	0 "0" →CA	0 "0" →RA
-1 L bus→LA	-1 C bus→CA	-1 R bus→RA

(4) Lowest carry-input field, ±1 (1 trit)

The up-down counting can easily be performed by this field :

1	Add with "+1" carry-in to the lowest full adder
0	Add with " 0" carry-in to the lowest full adder
-1	Add with "-1" carry-in to the lowest full adder.

(5) Carry control field, CC (2 trits)

This field controls the carry register FFF2.

(6) Shift circular control field, SH (1 trit)

A 1-trit field is sufficient to encode the three possibility :

1	Shift right one trit
0	No shift
-1	Shift left one trit.

(7) Overflow flag field, C (1 trit)

This field controls the OVF register FFF1.

(8) Data bus destination, D (3 trits)

Information on the data bus is routed to the destination specified by this field.

Sequential Control Fields

(1) Branch test field, BT (2 trits)

Conditions from which to determine the lowest address trit for the next micro-instruction address are specified by this field. Using this field, 3-valued branching can easily be executed. One must designate the lowest trit of the next address to be "0" :

0	0	No test	
1	1	OVF test	Replace the address trit with the value of FFF1.
1	0	Carry test	Replace the address trit with the value of FFF2.
1	-1	Sign test	Replace the address trit with the value of FFF3.
-1	1	OVF 1 test	Set the address trit to "1" if the value of FFF1 is "1", and otherwise set "0".
-1	0	Carry 1 test	Set the address trit by the value of FFF2 as mentioned above.
-1	-1	Sign 1 test	Set the address trit by the value of FFF3 as mentioned above.
0	-1	Sign 0 test	Set the address trit to "0" if the value of FFF3 is "0", and otherwise set "-1".

(2) Functional branch field, FB (1 trit)

Machine-language instructions can be connected with the microprogram by this field.

(3) Next address field, NA (5 trits)

These trits directly supply the address for the next microinstruction to be executed.

Microprogramming Timing

Fig. 3 shows the time chart in execution of a microinstruction. One microinstruction is executed in one machine cycle. The content of the ROM specified by the ROMAR is read-out by a pulse C_1. The microinstruction decoded controls the corresponding gates. Let the decoding time be t_1, the delay time of the product circuit be t_2, the time required for the adder be t_3, and the delay time of the shift circular circuit be t_4. If the propagation delay time of the gates used here is assumed to be τ sec, then $t_1 = 8\tau$, $t_2 = 2\tau$, $t_3 = 21\tau$, and $t_4 = 2\tau$. After a elapsed time ($t_1+t_2+t_3+t_4+\Delta t$), a set pulse C_2 must be sent to the AOB latch circuit.

The elapsed time determined by t_3 is substantially decreased as compared with the binary case because of the reduction of the digits and the stage of gates needed. For example, if the propagation delay time of binary AND, OR and NOT gates is assumed to be τ sec as well as the case of the ternary gates, then the time required for the corresponding 16-bit 2-input adder will be 35τ. A pulse C_3 set the registers, and then a pulse C_4 the ROMAR.

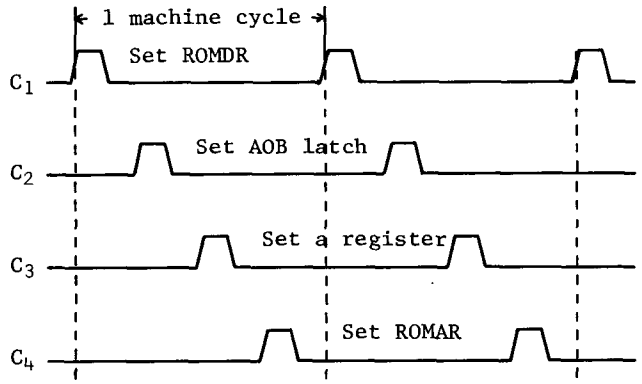


Fig. 3

Microprogramming Example

The machine instruction consists of a 4-trit operator and a 6-trit operand. Fig. 4 represents the flow chart for stage I (fetch cycle). The microprogramming of stage II (execution cycle) is given by the machine instruction used. As an example of microprogramming, consider the case of multiplication which determines the operating time of digital filters. The flow chart is shown in

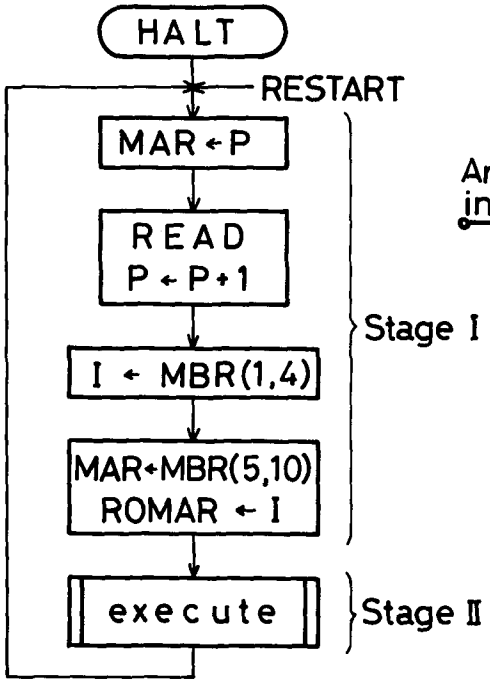


Fig. 4

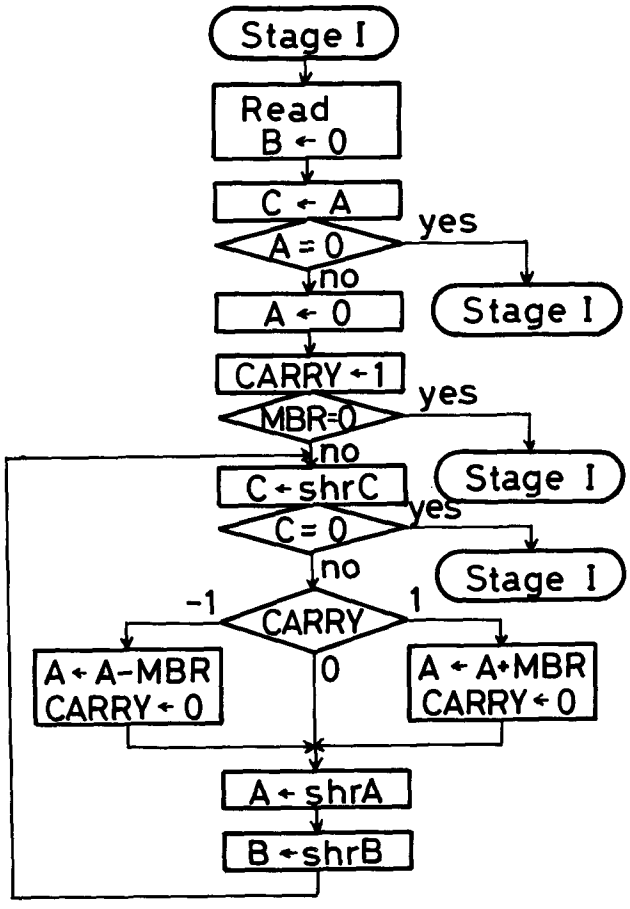


Fig. 5

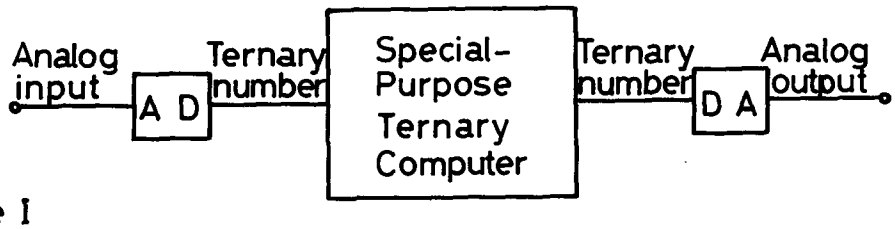


Fig 6

Fig. 5. From this figure it follows that micro-programming for the ternary multiplication can simple be done without a sign conversion and complex overflow detection. The multiplier (MBR) and multiplicant (A register) are 10 trits, and the product obtained will be 20 trits, (A-B registers). In rounding off the result into 10 trits, no special round off technique is required. The microprogramming steps necessary for the ternary multiplication are 13 steps. On the other hand, 27 microprogram steps are required for the case of the corresponding binary computer which must use more complex algorithms.

V. PRACTICAL IMPLEMENTATION

Fig. 6 shows a diagram of the real-time digital filter under consideration. In the digital filter, the signal to be processed is an analog signal, while the processing is performed digitally. Therefore, the practical implementation needs the A/D and D/A converters as well as the special-purpose ternary computer.

First, consider the ternary computer. The practical implementation is realized using the ternary threshold gates as a basic building block [11] except for the ternary ROM and RAM realized based on the binary ROM and RAM. The basic circuits utilized for the implementation are the D FFF [15], the four-input full adder including a carry input [11], the product circuit, and the T-gate as shown in Figs. 7,8,9 and 10, respectively.

The ternary A/D converter used here has been realized on the basis of the commercially available IC operational amplifier [17]. Fig. 11 shows the flow chart representing the algorithms for the A/D converter where X is the analog input and a_I is the ith-trit. Fig. 12 represents the circuit realization based on the flow chart of Fig. 11. Similarly, the D/A converter is realized based on ladder networks [18].

The ternary ROM and RAM are implemented based on the commercially available binary ROM and RAM, but it is ommited here.

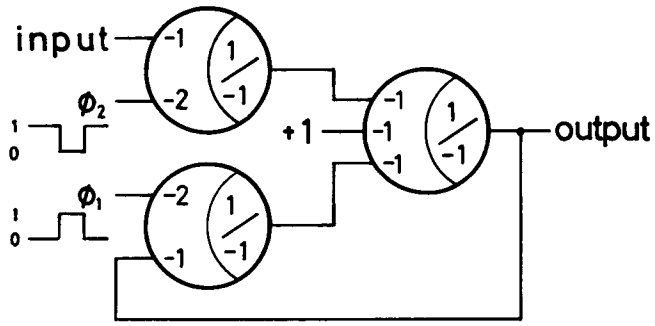


Fig. 7

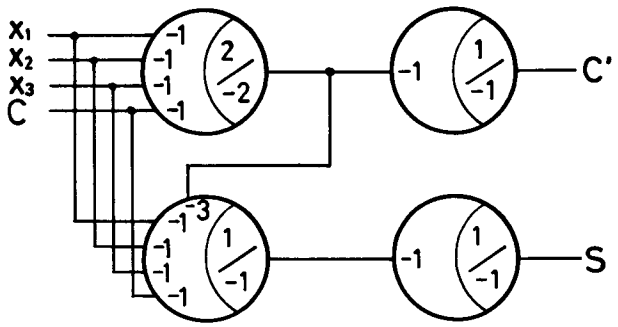


Fig. 8

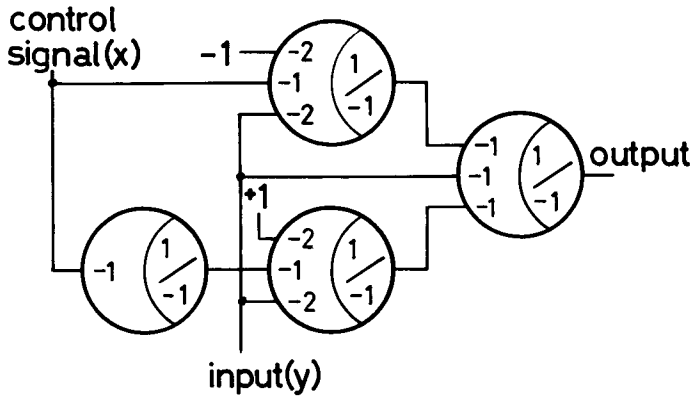


Fig. 9

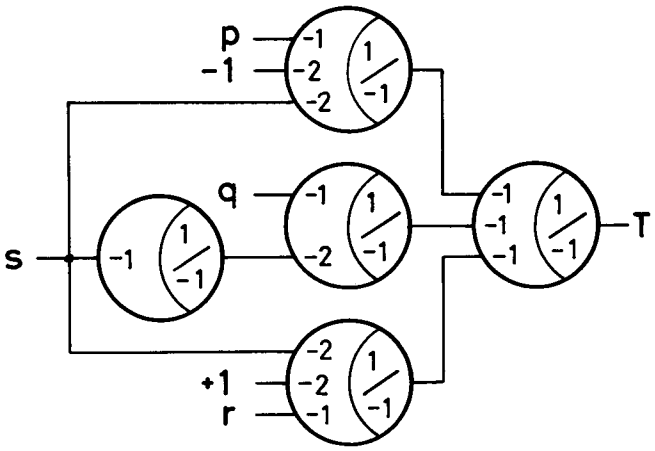


Fig. 10

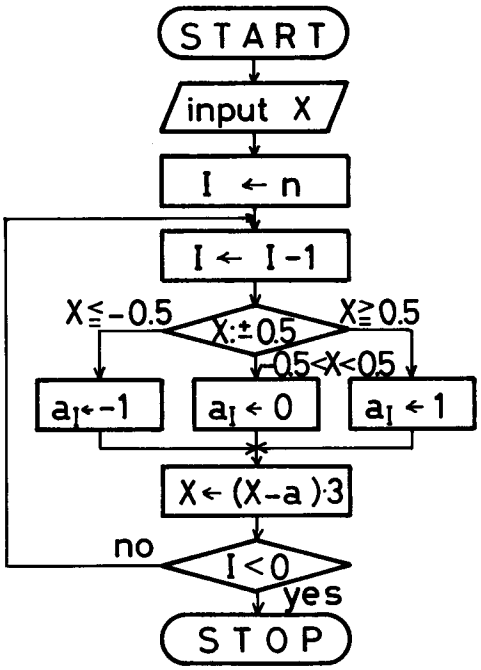


Fig. 11

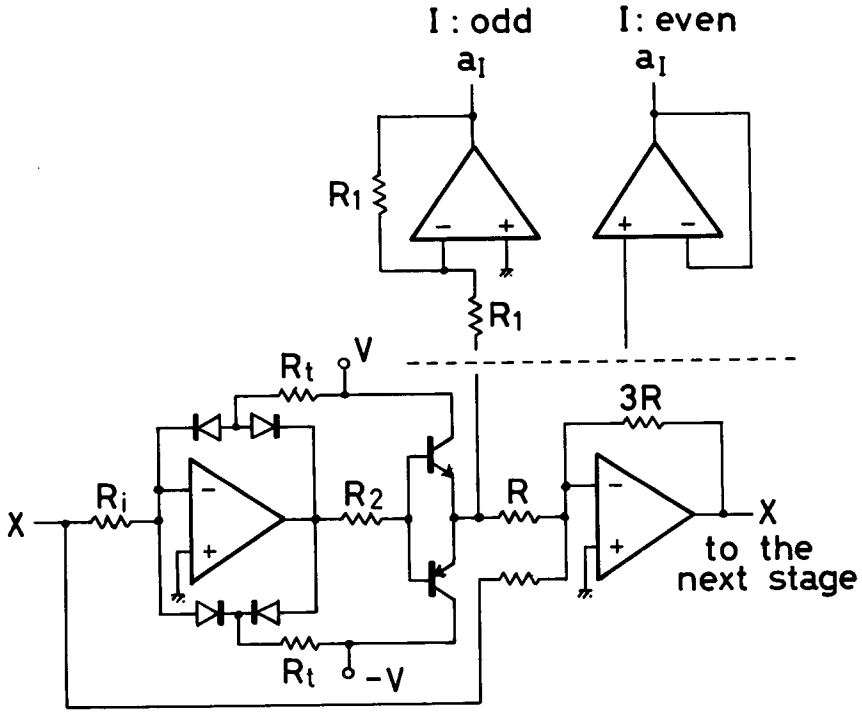


Fig. 12

VI. APPLICATION TO DIGITAL FILTERING

As a simple example, consider an implementation of the cannonic form of the 2nd-order digital filter of Fig. 13. This structure forms the basic building block for the development of further and more complicated cannonic structures. Any nth-order digital filter can be represented as either a cascade or a parallel (or combination of both) of this structure.

Digital filtering considered here must carry out two 3-input additions, five multiplications, and two delay operations. The 3-input addition can be carried out in one machine cycle as well as the 2-input addition. The detection of overflow and underflow is more clegant and simple because the ternary addition can be done in one step, and needs no complement representation. The flow chart for the 3-input addition is illustrated in Fig. 14. If overflow or underflow occurs, the result is set to the maximum or minimum value, so that overflow oscillation can be avoided [13]. In the binary case, it is practically impossible to detect strictly overflow in the 3-input addition because of awful complexity.

The microprogramming steps for the multiplication mentioned earlier are reduced to about a half of the ones for the binary multiplication. The execution steps which determine the operating time of the digital filter are also decreased. In the ternary and binary multiplication, the machine cycles needed are 5t + 8 and 5b + 21, respectively where t is the number of trits and b is the number of bits. The relation between t and b becomes t = 0.63b. Hence, the execution steps in the ternary case are reduced to 3.15b + 8. For example, the 10 x 10-trit multiplication can be carried out by only 58 execution steps, while the corresponding 16 x 16-bit multiplication needs 101 steps. Fig. 15 represents the execution steps required for the binary and ternary multiplication. This figure shows that the required execution steps can substantially reduced as compared with the case of the corresponding binary multiplication.

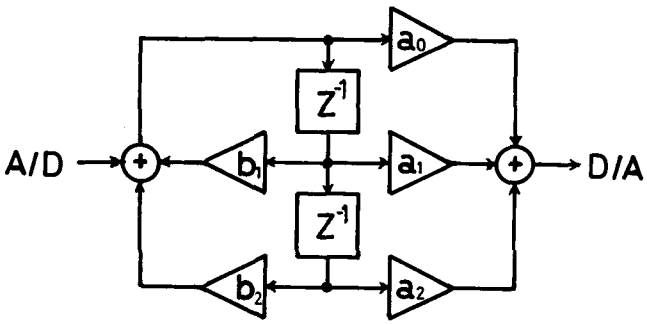


Fig. 13

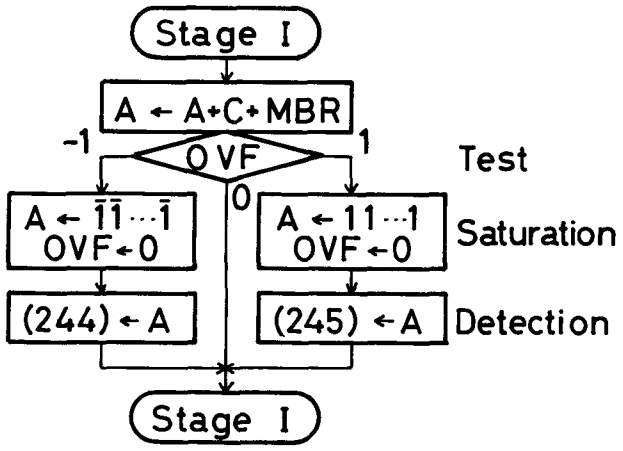


Fig. 14

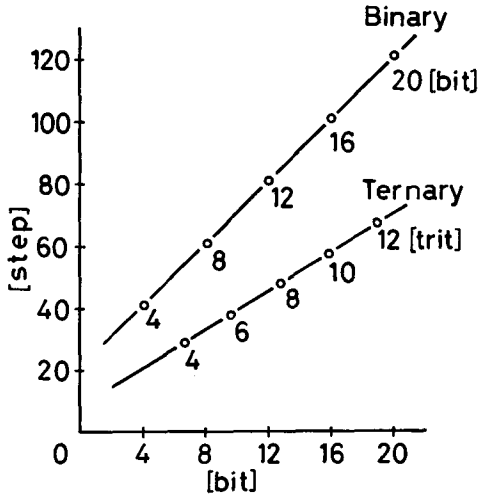


Fig. 15

VII. CONCLUSION

This paper has described the design and implementation of a special-purpose microprogram-controlled ternary computer for digital filtering. It has been demonstrated that many advantages of the ternary number system can effectively be employed for the computer in hardware and software. In particular, it has been shown that the ternary number system simplifies the arithmetic operations required for digital filters, so that the hardware structure and programming for the ternary computer can easily be carried out.

ACKNOWLEDGMENT

The authors wish to thank Prof. T. Anayama for many helpful discussions.

REFERENCES

[1] B. Gold and C. M. Rader, Digital processing of Signals. New York : McGraw-Hill, 1969.

[2] L. R. Rabiner and C. M. Rader, Digital Signal Processing. New York : IEEE Press, 1972.

[3] Digital Signal Processing Committee, Digital Signal Processing, II. New York : IEEE Press, 1976.

[4] L. R. Rabiner and B. Gold, Theory and Application of Digital Signal Processing. Prentice-Hall, 1975.

[5] N. Ahmed and J. P. Jayapalan, " On digital filter implementation via microprocessors," IEEE Trans. Ind. Electron. Contr. Instrum., vol. IECI-23, pp. 249-253, Aug. 1976.

[6] T. Higuchi, T. Saito and A. Kanomata, " A microprocessor-based digital filter programmed in a block diagram language," IEEE Trans. Ind. Electron. Contr. Instrum., vol. IECI-24, pp. 231-234, Aug. 1977.

[7] I. Halpern and M. Yoeli, " Ternary arithmetic unit," Proc. IEEE, vol. 115, pp. 1385-1388, Oct. 1968.

[8] D. I. Porat, " Three-valued digital systems," Proc. IEEE, vol. 116, pp. 947-954, June 1969.

[9] Z. G. Vranesic and K. C. Smith, " Engineering aspects of multi-valued logic systems," IEEE Computer, vol. 7, pp. 34-41, Sep. 1974.

[10] L. Ojala and P. Yrjölä, " Design of incremental signal processing devices using binary coded ternary logic," Conference Record of the 1973 International Symposium on Multiple-valued Logic, Toronto, May 24-25, 1973.

[11] T. Higuchi and K. Kobayashi, " Hardware implementation of digital filter based on ternary threshold gates," Proceedings of the 7th International Symposium on Multiple-valued Logic, Charlotte, May 24-27, 1977.

[12] D. C. Rine, Computer Science and Multiple-valued Logic, theory and applications. North-Holland Publishing Company, 1977.

[13] P. M. Ebert, J. E. Mazo and M. G. Taylor, " Overflow oscillations in digital filter," Bell Syst. Tech. J., vol. 38, pp. 2999-3020, Nov. 1969.

[14] G. L. Kratz, W. W. Sproul and E. T. Walendziewicz, " A microprogrammed approach to signal processing," IEEE Trans. Computers, vol. C-23, pp. 808-817, Aug. 1974.

[15] T. Higuchi and M. Kameyama, " Static-hazard-free T-gate for ternary memory element and its application to ternary counters," IEEE Trans. Computers, vol. C-26, pp. 1212-1221, Dec. 1977.

[16] T. Higuchi and M. Kameyama, " Ternary logic system based on T-gate," Proceedings of the 5th International Symposium on Multiple-valued Logic, Bloomington, May 13-16, 1975.

[17] K. Kobayashi and T. Higuchi, " A/D converter for ternary digital filter," 1977 National Convention Record of the Institute of Electronics and Communication Engineers of Japan, 31, Aug. 1977 (in Japanese).

[18] L. Ojala and P. Yrjölä, " Design of A/D converters for binary coded ternary systems," Conference Record of the 1972 Symposium on the Theory and Applications of Multiple-valued Logic Design, New York, May 25-26, 1972.