MODULAR APPROACH TO THE DESIGN OF DIGITAL SIGNAL PROCESSOR

ENGR. (DR.) S. O. FALAKI
Industrial Mathematics & Computer Science Department
Federal University of Technology, Akure.

ABSTRACT

A new technique for a hardware implementation of digital signal processor is proposed. The technique is based on the mapping of differentials. The characteristic equation of the processor takes a modular form. This gives the possibility of designing a module which can be used to build any higher order digital processor. The output of the module is the first difference of its input, as well as the product of that difference with the coefficient. The modules are connected to realize the different table, which in turn, satisfies the characteristic equation of the processor.

More than one design for module is considered. A comparison between these different design from the viewpoint of computational time and equipment is also given.

INTRODUCTION

In general, the term, "digital filter", or "digital processing machine" refers to any device which operates on input number sequence to produce a second sequence of numbers by means of computational algorithm.

If a digital filter is part of a signal processing system, the input number sequence is usually the digital version of an analog signal. The output sequence is usually converted to the analog form. The advantages of coding a signal digitally are well known. Briefly, digital representation offers ruggedness, efficient signal regeneration, easy encryption, the possibility of combing transmission and switching functions, and the advantage of a uniform format for different types of signals. The price paid for the benefits is the need for increased bandwidth (1).

Until recently, signal processing has typically been carried out using analog equipment. Because of the flexibility of digital computers, it was often useful to simulate a signal processing system on a digital computer before implementing it in analog hardware. The rapid development in the technology of medium, large and very large scale integrated circuits is also making it possible to construct special - purpose hardware for real-time digital signal processing (2).

Higher - order digital filters were normally realized as either a cascade or parallel network of basic second - order section (3, 4), which in the former case, are ordered for minimum round-off noise and have outputs suitably scaled (7, 8). also, the actual hardware synthesis, reported in the literature, is usually at the discrete gate level, and structures proposed are mainly for specific configurations.

In this paper, modular approach to hardware implementation of digital processor is proposed. This approach is general and flexible. We proposed here the use of a simple mapping (mapping of differentials) procedure to transform a given continuous - time filter to its corresponding digital filter. The resulting characteristic equation takes the form of a summation of difference ($\Delta_1, \Delta_2, \ldots$). The hardware realization of such form of equations needs a number of similar modules. The output of the proposed module is $\Delta_{i+1}w(n)$ if its input is $\Delta_i w(n)$. The main advantage of this approach is the possibility of using the same module for building a digital system in which the full sample values are used for processing, as well as those systems in which incremental changes are used in processing. Another advantage is the possibility of using it for processing directly a differential pulse code modulated (DPCM) signal.

In section 2, the design approach using mapping of differentials will be considered. The proposed hardware implementation is the subject of section 3. More than one module will be given in this section. A comparison between the different structures is also

given. The hardware requirements as well as the computation time is taken as base of the comparison. Conclusion is given in section 4 and acknowledgments in section 5.

2. DESIGN APPROACH-MAPPING OF DIFFERENTIALS:

The problem of designing filters is one of finding filter coefficients such that some aspects of the filter's response (e.g. time response, frequency response, group delay, etc.) approximates a desired behaviour in a specified manner. If the approximation problem is solved in the z-plane, the resulting filter is a digital filter. If it is solved in the S-plane, the resulting filter is an analog filter. The theory and techniques of approximation in the S-plane is well established in the literature. Then, instead of redeveloping these theories for digital filters, simple mapping procedures can be used to transform filters from one domain to filters in the other domain. The technique of designing an appropriate continuous - time filter and digitizing the resulting design to give a digital filter is very useful for designing standard filters such as lowpass, handpass, stoppass, and highpass filters.

One of the simplest ways of digitizing a continuous system is to replace differentials in the differential equation of the continuous system with finite difference in order to obtain a difference equation that approximates the given differential equation (5). The differential equation of a filter is:

$$\sum_{i=0}^{N} a_i \frac{d^i y(t)}{dt^i} = \sum_{i=0}^{N} b_i \frac{d^i x(t)}{dt^i} \quad \text{.....(1)}$$

The corresponding difference equation will be:

$$\sum_{i=0}^{N} a_i \Delta_i[y(n)] = \sum_{i=0}^{N} b_i \Delta_i[x(n)] \quad \text{..... (2)}$$

where:

x(n) - is the input sequence to the filter;

y(n) - is the output sequence

$\Delta_i w(n)$ - is the i^{th} difference defined by:

$$\Delta_{i-1}[w(n)] = \Delta[\Delta_i[w(n)]] \quad \text{..... (3)}$$

and

$$\Delta_1[w(n)] = \begin{cases} \frac{1}{T}[w(n)-w(n-1)] & \textit{for backward difference} \\ \frac{1}{T}[w(n+1)-w(n)] & \textit{for forward difference} \end{cases} \quad (4)$$

The two desirable properties of any mapping from continuous to discrete space are:

1. The jΩ axis in the S-plane should be mapped to the unit circle in the Z-plane.
2. Points in the left-half of the S-plane {Re[s]<0} should be mapped inside the unit circle (|Z|<1).

The use of backward differences satisfies the second property while the first property is not satisfied [5]. Therefore stable analog filters map into stable digital filters using backward differences but their frequency selection properties are not maintained. Forward difference method also, does not satisfy both properties. The use of higher-order properties of mapping. For example, consider the case of defining Δ_1 of equation (4) by the expression:

$$\Delta_1[w(n)] = \frac{1}{T}\sum_{i=1}^{L} c_i[w(n+i) - w(n-i)] \quad \text{..... (5)}$$

where L is the order of difference to be used,

The mapping between S-plane and Z-plane then becomes:

$$S = \frac{1}{T}\sum_{i=1}^{L} c_i(z^i - z^{-i}) \quad \text{..... (6)}$$

by setting $Z=\exp^{(jwT)}$ in equation (6) we obtain

$$S = \frac{1}{T}\sum_{i=1}^{L} c_i(e^{jwTi} - e^{-jwTi}) = \frac{1}{T}\sum_{i=1}^{L} 2j.c_i \sin(wiT) = j\beta(w) \quad \text{..... (7)}$$

This result means that, points on the unit circle in the Z-plane ($Z=e^{jwT}$) are the mapping of points on the jΩ axis in the S-plane ($S=j\beta(w)$). By proper choice of the coefficients c_i, the function $\beta(w)$ can approximate almost any desired odd function in the S-plane which can be mapped monotonically to the unit circle in the Z-plane. Furthermore, the mapping of equation (6) can be shown to be conformal, thereby mapping the left-half of the S-plane to the inside of the unit circle in the Z-plane. For such cases both desirable properties of the mapping are preserved.

The details of selecting the coefficients c_i that make the function β_i (w) approximate a given frequency characteristics is out of the scope of this paper; since we are concerned here with the phase of the hardware implementation of the filters. Some information are given in ref. [5], [6].

In all the existing works, after calculating the filter coefficients and for the implementation purposes, the difference equation (2) is usually converted to the classical non-recursive form:

$$y(n) = \sum_{i=1}^{N} a_i x_{n-i} \quad \text{..... (8a)}$$

or to the recursive form:

$$y(n) = \sum_{i=0}^{N} a_i x_{n-i} - \sum_{i=1}^{M} b_i y_{m-i} \quad \text{..... (8b)}$$

by using either equation (4) or (5). This is not the method used in this paper, here, equation (2) will be used directly for the hardware implementation for the following reasons:

1. The aim of this paper is the design of multiplication free digital filter by using mainly the difference signal in processing instead of the full sample value. The multiplication free filter is realized actually by using one-bit to represent the difference word.

2. Equation (2) gives the possibility of using modular approach for realizing the digital filter. We can mention here that, if the classical form of the filter equation (Equation 8a) is suitable for processing pulse code modulated (PCM) signals, then the form of [2] is that suitable for implementing a filter whose input is a differential pulse code modulated signal (DPCM).

3. HARDWARE IMPLEMENTATION [8]

Equation (2) is the base of the proposed hardware implementation. This equation means that, the implementation of the Nth order digital filter, needs a number of similar modules. The output of the module is the first difference of its input signal.

If the input signal is:

$I(n) = \Delta_i i[w(n)]$ ..... (9a)

the output will be

$O(n) = \Delta_{i+1}[w(n)]$ .....(9b)

The hardware of the module depends on the method used to calculate the first difference Δ_i. In this section, some of the possible modules are examined.

3.1 THE USE OF BACKWARD DIFFERENCE:

The algorithm of operation of module is:

$O(n) = \Delta_i[I(n)] = [I(n) - I(n-1)]$ ..... (10)

The hardware realization of the module based on the above equation is given in Fig. 1. The summation in equation (2) consists of terms of the form $a_i\Delta_i[w(n)]$. To get these terms, the coefficient a_i is used as the second input to the module. In turn, the module has two outputs; one of them $a_i\Delta_{i+1}[w(n)]$ is used in the summation to get the final result, and the other output $\Delta_{i+1}[w(n)]$ is used as input to the preceeding module. The module consists of a delay element, an adder, and a multiplier.

3.2 THE USE OF FORWARD DIFFERENCE

The alogrithm of operation of the module is:

$O(n) = \Delta_i[I(n)] = [I(n+1) - I(n)]$ ..... (11)

In this case, a predictor is necessary to predict the value of I(n+1). The prediction can be for any order. Let us examine how to construct a predictor. Using Taylor Series expansion for y(t+h) we get:

$$y(t+h) = y(t) + hy'(t) + \frac{1}{2!}h^2y''(t) + \text{....} \quad (12)$$

$y(t+h) = a_0y_0 + a_1y_1 + a_2y_2 + a_3y_3 + \text{... .....}$ (13)

where

$y_0 = y(t)$

$y_1 = y(t-h)$

$y_2 = y(t-sh)$ etc. and

a_0, a_1, a_2 ..... are the coefficients of the prediction routine.

So equation (13) becomes:

$$y(t+h) = a_0y_0 +$$

$$+ a_1(y-hy_1+{}^{h2}/_{2!}y'' - {}^{1}/_{3!}h^3y''' +...)+$$

$$+ a_2(y-2hy'+4{}^{h2}/_{2!}y'' - {}^{1}/_{3!}8h^3y''' +...)+$$

$$+ a_3(y-3hy'+9{}^{h2}/_{2!}y''-{}^{1}/_{3!}27h^3y''' + ...) +...$$

$$+..... \quad (14)$$

To construct a predictor of any chosen order, we simply equate terms in equations (12) and (14). For instance, zero-order prediction is given by:

$$y(t+h) = y(t) \quad ..... (15)$$

A first-order predictor gives:

$$y + hy' + {}^{h2}/_{2!}y'' + ... \sim a_0y+a_1(hy' + ...) \quad ..... 16$$

and equating terms in y and y' in equation (16) we have:

$$\left.\begin{aligned} a_0 + a_1 &= 1 \\ a_1 &= -1 \end{aligned}\right\} \quad ..... 17$$

$\therefore a_0=2$

Thus

$$y(t+h) = 2y(t) - y(t-h) \quad ..... (18)$$

The second-order predictor gives:

$$y+hy'+{}^{1}/_{2}!h^2y''\sim a_0y_0 + a_1y_1 + a_2y_2$$

$$= y(a_0+a_1+a_2)+hy' (-a_1-2a_2)+{}^{1}/_{2}h^2h'' (a_1+4a_2) \quad .....(19)$$

Hence equation terms in equation (19)

$$\left.\begin{aligned} a_0+a_1+a_2 &= 1 \\ -a_1-2a_2 &= 1 \\ a_1+4a_2 &= 1 \end{aligned}\right\} \quad ..... 20$$

Therefore

$a_2 = 1$, $a_1 = -3$ and $a_0 = +3$

i.e

$$y(t+h) = 3y(t) - 3y(t-h) + y(t-2h) \quad ..... (21)$$

Hence in our case, for the first order prediction and using eq (18) we have

$$I(n+1) = 2I(n) - I(n-1) \quad ..... (22)$$

therefore

$$\Delta_i I(n) = I(n+1) - I(n) = I(n) - I(n-1) \quad ..... (23)$$

This is the same as the backward difference method. The module in turn, has the same shape as that given in Fig. 1.

For the second-order prediction and using equation (21) we have:

$$I(n+1) = 3I(n) - 3I(n-1) + I(n-2) \quad ..... (24)$$

and

$$\Delta_i I(n) = 21(n) - 31(n-1) + I(n-2) \quad ..... (25)$$

The hardware realization of the module, which is based on this equation is given in Fig. 2 The module has two input I(n) and a; and two outputs $\Delta_i I(n)$ and $a.\Delta_i I(n)$. Two delay element are used to get I(n-1) and I(n-2); and three multipliers are also needed to build up the module.

3.3 THE USE OF HIGHER ORDER DIFFERENCE

The algorithm of operation of the module is:

$$O(n) = \Delta_1[I(n)] = \sum_{i=1}^{L} c_i[I(n+i)-I(n-i)] \quad (26)$$

Where L is the order of difference. Also in this case, a prediction is necessary to predict the value of I(n+i). The prediction can be for any order.

i. For first order prediction we get:

$$\Delta_1[I(n)] = 2I(n)\sum_{i=1}^{L} c_i - 2\sum_{i=1}^{L} c_iI(n-i) \quad ..... (27)$$

ii. For second order prediction we have:

$$\Delta_1[I(n)] = \sum_{i=0}^{2L} k_iI(n-i) \quad ..... (28)$$

where,

$$k_o = 3\sum_{i=o}^{L} c_i \quad ..... (29)$$

and the relation between k_i (i=1,2,3,...) and L is given in Table 1.

L/k$_i$	k$_1$	k$_2$	k$_3$	k$_4$	k$_5$	k$_6$	k$_7$	k$_8$
1.	-4c$_1$	c$_1$						
2.	-4c$_1$	c$_1$-4c$_2$	0	c$_2$				
3.	-4c$_1$	c$_1$-4c$_2$	-4c$_3$	c$_2$		0	c^3	
4.	-4c$_1$	c$_1$-4c$_2$	-4c$_3$	c$_2$-4c$_4$	0	c$_3$	0	c$_4$

Table 1

The hardware realization of the module for the second order prediction and for L = 1 is given in Fig. 3. It consists of two delay elements, four multipliers, and one adder.

3.4 DIFFERENCE TABLE GENERATION METHOD

The right-hand side, as well as the left-hand side of the difference equation (2) consists of the difference situated in the upper line of the table [y(n),Δ_i y(n), Δ_2 y(n), ... Δ_N (n)].

To get the value of the summation at any instant (n), the difference table at that instance, must be calculated. As a matter of fact, there is no need to calculate all the terms of the difference table each time we need the summation. All the terms at instance n, except those appearing in the upper line, can be generated by simple down shift of the terms at instance (n-1). As a result, at any instance it is enough to calculate the new values of the terms of the upper line. This need a module which realizes the equation;

$$O(n) = I(n) - I(n-1) \quad \ldots\ldots (30)$$

if

$$I(n) = \Delta_{i+1}\, y(n),\ I(n-1) = \Delta_{i-1} y(n-1),$$

then

$$O(n) = \Delta_{i+1}\, y(n) - \Delta_{i+1}\ y(n-1) \quad \ldots\ldots (31)$$

The hardware realization of the module based on the above equation is given in Fig. 4. The module consists of two registers to store the values of I(n) and I(n-1). It consists also of one adder and a multiplier.

3.5 REALIZATION OF HIGHER ORDER DIGITAL FILTER

The realization of an nth order digital filter using any of the proposed modules is shown in Fig. 5 It needs (N+M) modules where N and M are as in equation (2). The decision to use any of the proposed modules is affected by the computation time of the filter and the required hardware comparison of the various implementation methods is given in Table 2.

Method	Hardware Requirements			Computation Time
	Adder	Multiplier	Registers or Delays	
Backward	(M+N+3)	(M+N)	(M+N)	(N+2)T$_a$+T$_m$ if (T$_m$<NT$_a$) or, 2T$_a$ + T$_m$ if (T$_m$>NT$_a$)
Forward (second order prediction)	(M+N+3)	2(M+N)	2(M+N)	N(2T$_a$+T$_m$) + T$_m$+2T$_a$
Higher Order Difference (second order prediction)	(N+M+3)	4(M+N)	2(M+N)	(2T$_a$+T$_m$)N + T$_m$ +2T$_a$
Difference table	(N+M+3)	(M+N)	2(M+N)	(N+2)T$_a$+T$_m$

Table 2

(T$_a$ - one addition time, T$_m$ - one multiplication time)

The use of the configuration of Fig. 5 needs (M+N) multipliers as given Table 2. This makes the given configuration impractical. This is more so if fast multiplier are to be used.

Considering the fact that in any difference table, the formation of the nth difference Δ_n comes only after the formation of the lower differences, which means that the systems is a serial system. As a result, it is possible to take out the multiplier from each module and use only one multiplier for the whole system. As a result the module will consist of delay elements (or a register) and adders. The number of delay elements and adders depends on the module used. If backward differences are used, the formation of the differences Δ_i (n), Δ_2(n),..., Δ_N(n) is shown in Fig. 6.

The block diagram needs N delay elements and [N + (N-1) + (N-2) + ... +2 +1] adders. The number of adders can be reduced to minimum if we take into consideration, as given in section 3.4 that equation (2) contains only the upper row of the difference table. As a result the configuration of Fig. 6 can be modified to give that given in Fig. 7. This system contains N delay elements and N adders only.

The corresponding digital filter is shown in Fig. 8 A memory unit is used to store the filter coefficients a_1, a_2, a_3, ...,a_n; a multiplier is used to multiply the coefficients with the corresponding differences and the output of the multiplier is accumulated to get final result Yn.

4. CONCLUSION

A method for designing a general digital filter has been presented. The resulting hardware structures are modular, and have uniform inter-connection patterns. The flexibility of the proposed technique should make possible, the economical design of special purpose digital filter hardware for any application including those requiring real-processing. It is significantly more efficient in terms of computation time than the existing methods.

5. ACKNOWLEDGEMENTS

The author wishes to thank **D. S. Dawoud and F. S. Atiya** for their comments on the earlier version of this paper.

REFERENCES

1. **N. S. Jayant,** "Digital Coding of speech waveforms: PCM, DPCM, and MN Quantizers," IEEE Proc. vol. 62 No. 5 May 1974. p. 611 - 632.

2. Transputer Topics No 5 Sept. 1991.

3. **B. Gold and C. M. Rader,** "Digital Processing of Signals", McGraw-hill Book Company New York, 1969

4. **L. R. Rabiner and C. M. Rader,** "Digital signal Processing", IEEE Press, New York, 1972

5. **L. R. Rabiner and B. Gold,** "Theory and Application of Digital Signal Processing", Prentice-Hall, Inc. Englewood Cliffs, New Jersey, 1975

6. **Alan V. Oppenheim and R. W. Schafer,** "Digital Signal Processing", Prentice-Hall, Inc. Englewood Cliffs, New Jersey, 1975.

7. **B. Liu, "Effect** of finite word length on the accuracy of digital Filters a review", IEEE trans. on circuit theory vol. CT-18, No. 6, Nov. 1971, pp670-677.

8. **A. V. Openheim and C. J. Weinstrin,** "Effects of finite Register Length in digital filtering and Fast Fourier Transform", Proc. IEEE, vol. 60, No. 8, pp. 957-976, August, 1972.

9. **Trevor J. Terrell,** "Introduction to Digital Filters" Macmillan Education Ltd, London 1988 (2nd Ed.)

10. **Rodger E. Ziemer** et. al, "Signal and System; Continuous and discrete", Macmillan Publishing Company N. Y. 1989 (2nd INEd)

11. **Falaki S. O.,** "A new approach to the implementation of High Speed Digital Filter", Journal of Nigerian Society of Engineers Vol. 19 No. 2 April - June 1984 pp. 69 - 80.

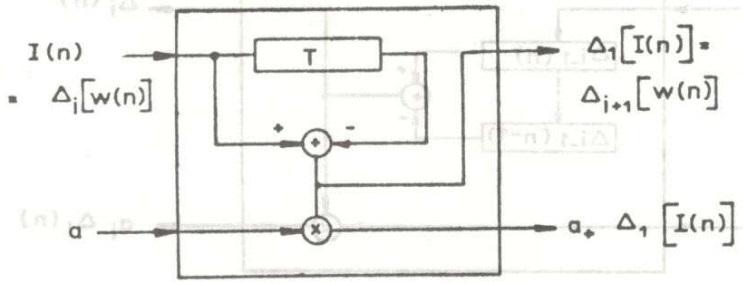


Fig. 1: THE MODULE USING BACKWARD DIFFERENCE.

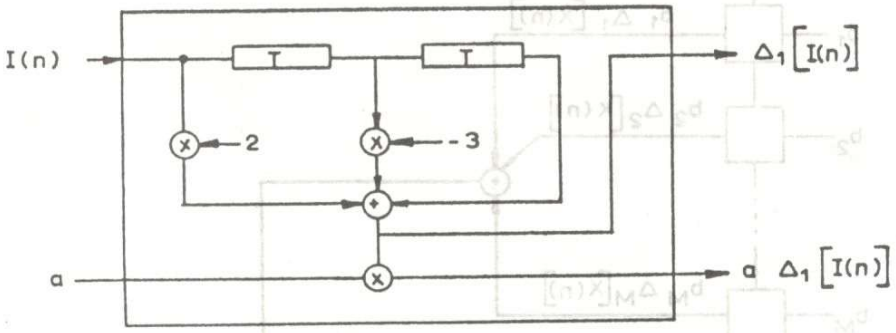


Fig. 2: THE MODULE USING FORWARD DIFFERENCE
(2^{nd} ORDER PREDICTION).

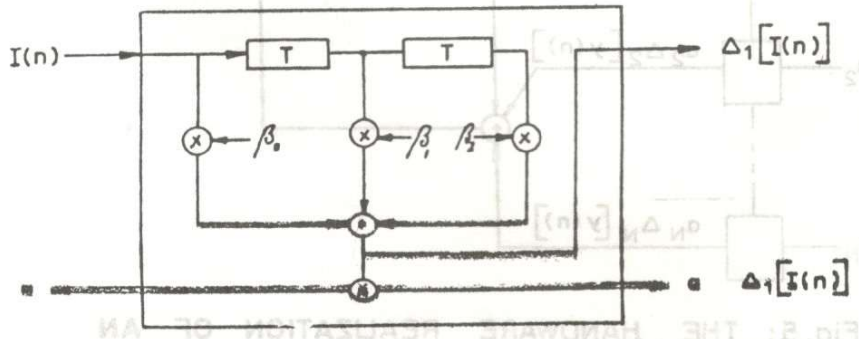


Fig. 3: THE MODULE USING HIGHER ORDER DIFFERENCE.

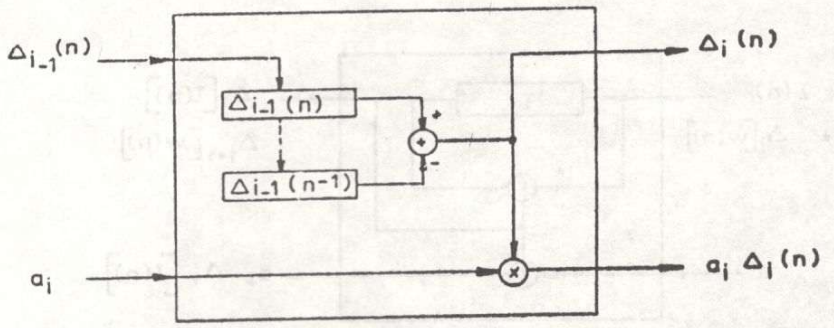


Fig. 4: THE MODULE IN CASE OF USING TABLE OF DIFFERENCES

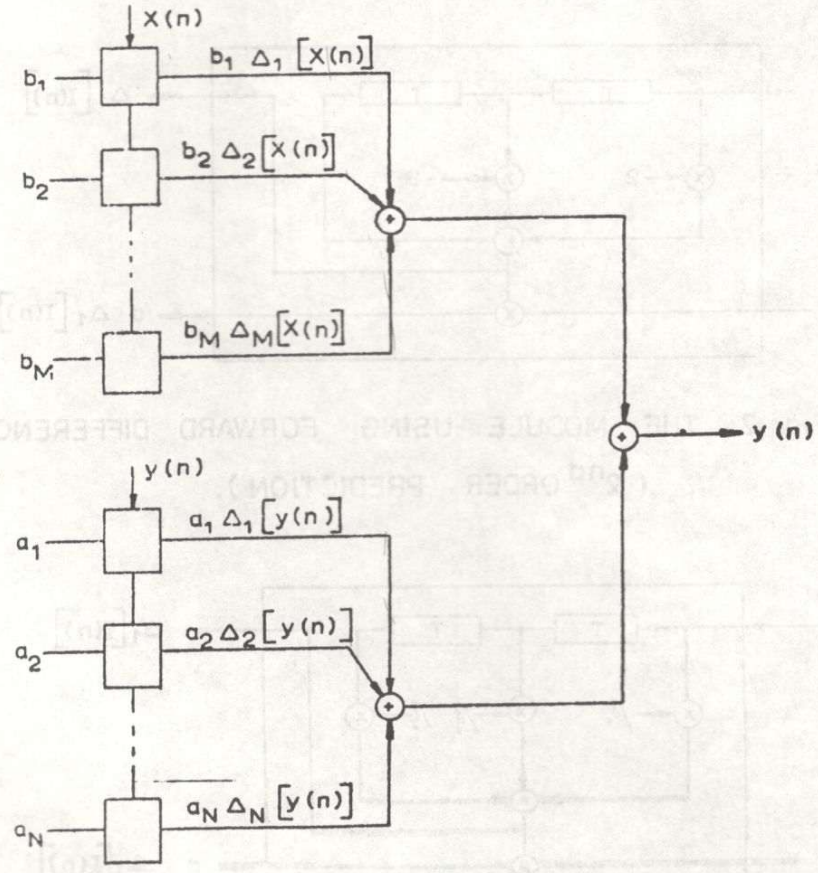


Fig. 5: THE HANDWARE REALIZATION OF AN N^{th} ORDER DIGITAL FILTER.

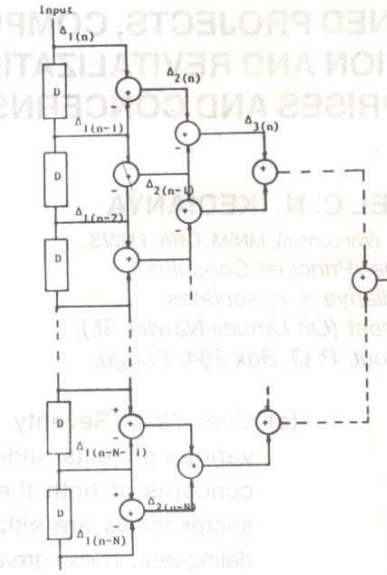


Figure 6. Formation of the difference table using delay elements and Adders.

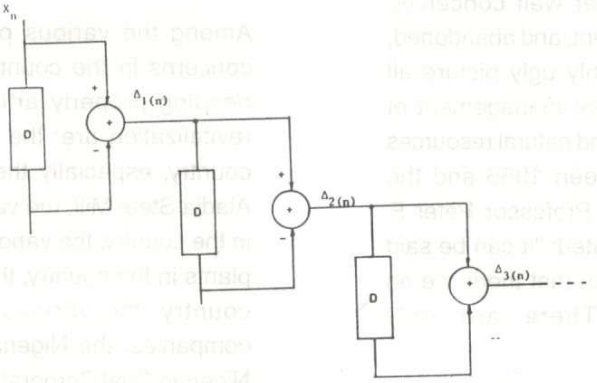


Figure 7. Generation of the first row of the difference table.

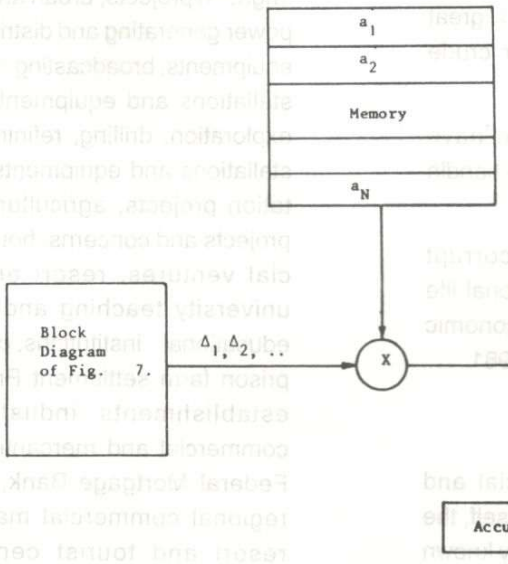


Figure 8. The Proposed Hardware Realization of an N^{th} Order Filter.