NPS-62-84-061

NAVAL POSTGRADUATE SCHOOL

Monterey, California

AD-A140 904

DTIC FILE COPY

1 0 1984

A

HIGH-SPEED RECURSIVE DIGITAL
FILTER REALIZATION

Herschel H. Loomis, Jr.
Bhaskar Sinha

6 February 1984

Approved for public release; distribution unlimited.

Prepared for:
Naval Electronic Systems Command
Washington, D.C. 20360

Reproduced From
Best Available Copy

84 05 09 014

20000803056

NAVAL POSTGRADUATE SCHOOL
Monterey, California

Commodore Robert H. Shumaker
Superintendent

David A. Schrady
Provost

This work was supported by the Naval Electronic Systems Command, Washington, D.C.

Reproduction of all or part of this report is authorized.

This report was prepared by:

HERSCHEL H. LOOMIS, JR., Adjunct Professor
Department of Electrical and Computer Engineering

Reviewed by:

ABRAHAM SHEINGOLD, Acting Chairman
Department of Electrical and Computer Engineering

Released by:

JOHN N. DYER, Dean
Science and Engineering

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
NPS-62-84-061

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
High-Speed Recursive Digital Filter Realization

5. TYPE OF REPORT & PERIOD COVERED
Technical

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
Herschel H. Loomis, Jr.
Bhaskar Sinha

8. CONTRACT OR GRANT NUMBER(s)

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Naval Postgraduate School
Monterey, CA 93943

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
62734N
N0003983WRDY078

11. CONTROLLING OFFICE NAME AND ADDRESS
Naval Electronic Systems Command
Washington, DC 20360

12. REPORT DATE
6 February 1984

13. NUMBER OF PAGES

14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)

15. SECURITY CLASS. (of this report)
UNCLASSIFIED

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)
Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES
This is a preprint of a paper accepted for publication by CIRCUITS, SYSTEMS, AND SIGNAL PROCESSING.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)
Digital filters, high rate computation, high-speed logic, pipelined logic, recursive digital filters.

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)
Pipeline techniques have been successfully applied to speeding up processing in both general and special purpose digital computers. Application of these techniques to non-recursive (FIR) filters has been suggested and is quite straightforward. Application to recursive (IIR) filters has not previously been shown. In this paper, the technique for applying pipeline techniques to recursive filters is shown and the advantages and disadvantages of the technique are discussed. Using these techniques, recursive digital filters

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

operating at hitherto impossibly high rates can be designed.

High-Speed Recursive Digital Filter Realization

Herschel H. Loomis, Jr.

Bhaskar Sinha

Pre-print of paper to appear in

CIRCUITS, SYSTEMS, AND SIGNAL PROCESSING

Accession For

A1

High-Speed Recursive Digital Filter Realization*

Herschel H. Loomis, Jr.**

Bhaskar Sinha***

ABSTRACT

Pipeline techniques have been successfully applied to speeding up processing in both general and special purpose digital computers. Application of these techniques to non-recursive (FIR) filters has been suggested and is quite straightforward. Application to recursive (IIR) filters has not previously been shown. In this paper, the technique for applying pipeline techniques to recursive filters is shown and the advantages and disadvantages of the technique are discussed. Using these techniques, recursive digital filters operating at hitherto impossibly high rates can be designed.

---

*The basic research for this idea was supported by Department of Electrical and Computer Engineering, University of California, Davis, California 95616. The basic idea is the subject of a pending patent application. Subsequent research was supported by the U.S. Naval Electronic Systems Command.

**Department of Electrical and Computer Engineering, Naval Postgraduate School, Monterey, California 93943. On leave from the Department of Electrical and Computer Engineering, University of California, Davis, California 95616.

***International Business Machines Corporation, Boca Raton, Florida 33432.

High-Speed Recursive Digital Filter Realization

ABSTRACT

Pipeline techniques have been successfully applied to speeding up processing in both general and special purpose digital computers. Application of these techniques to non-recursive (FIR) filters has been suggested and is quite straightforward. Application to recursive (IIR) filters has not previously been shown. In this paper, the technique for applying pipeline techniques to recursive filters is shown and the advantages and disadvantages of the technique are discussed. Using these techniques, recursive digital filters operating at hitherto impossibly high rates can be designed.

INTRODUCTION

Digital filters have an important place in the technology of processing signals. Because of the sample theorem, the sampling rate which is also the clock rate of the filter, must be higher than twice the highest frequency component in the signal to be filtered. Thus the use of digital filters in real time application involving high frequencies demands high sample rate.

Pipeline techniques are well known [1,2] ways to increase the clock rate (throughput) of a particular state-of-the-art realization of a logic module. For example, the Cray I supercomputer has a clock rate of 80 Mhz and requires a 7 stage pipeline multiplier to perform 64-bit floating point multiplication at the rate of one product every 12.5 ns. [3].

Other workers have investigated residue number techniques to obtain high rate digital filters [4]. However, difficulties in implementing division by a constant impose limitations on such realizations, making scaling awkward and precluding anything resembling floating point.

Pipeline techniques can thus be used to produce multipliers and adders that operate at the maximum clock rate possible for a given state of the art of logic devices. For example, TRW makes a non-pipeline 16 bit integer multiply chip with input and output registers that can be operated at the rate of about 9 MHz. Pipeline techniques could be applied to that design, producing a new chip that might have three stages and operate at near 40 MHz.

Discrete logic realization of the basic functions of multiplication and division can yield even better performance. At the extreme we have some of the integrated circuit ECL lines. Based on the experience gained with the pipeline supercomputers, it is reasonable to expect that a three stage 12-bit integer multiply and 1 to 2 stage 16-bit integer adder could be built to operate at around 80 MHz. Floating point operation could be obtained at the same rate

with a cost of perhaps only 2 additional stages for the multiplier and 3 more for the adder.

What finally limits the clock rate in a pipeline system is the net delay of all logic and the register delay between successive registers, however pipeline techniques will yield the highest throughput or clock rate possible with a given state of the art.

Now it should be clear why we should consider the application of pipeline techniques.

1) higher clock (sample) rates are possible for a given state of the art.

2) more complex operations (i.e. floating point) are possible for a given state of the art and clock rate.

Pipeline techniques can be easily applied to non-recursive systems, that is, systems without feedback. In the digital filter arena, this would correspond to non-recursive or Finite Impulse Response (FIR) filters. These techniques are straight forward and proceed as follows:

An FIR filter can be described by a transfer function in the delay operator (z^{-1} or D) as in (1)

$$\frac{y}{x} = H(z^{-1}) = a_0 + a_1 z^{-1} + \ldots + a_m z^{-m} \qquad (1)$$

Figure 1 shows the realization of such a filter using delayless add and multiply modules as well as unit delays. That realization is unrealistic and does not capitalize on the advantages of pipeline processors as discussed above.

Let us suppose that we have a 1-stage pipeline adder and a 3-stage pipeline multiplier.

Figure 2 shows a pipeline realization of the transfer function given in (1). In fact, (1) is not exactly realized because of the delay of the adders. Instead the output of the filter is y delayed by 7 clock periods. This, however, is not a serious difficulty.

Thus, pipeline realization of FIR filters is straight forward, costing only an added delay in the resulting output sequence rate, and perhaps some additional adders over the realization of figure 1, if the adder has more than one stage.

What then about doing this for IIR filters? Applying pipeline techniques to recursive calculations is not so straightforward as it is in the non-recursive case, and was first reported in [1] in connection with accumulator design.

In the next section we shall consider the design of recursive (IIR) filters. Following that, we discuss the general attributes of the solution and some of the unsolved problems associated with the technique. Finally, an example of a sixth order Butterworth filter is treated, to illustrate the technique.

PIPELINE RECURSIVE DESIGN

An nth order recursive or Infinite Impulse Response (IIR) filter can be characterized by a ratio of two nth degree polynomials in z^{-1}, the delay operator, as shown in (2)

$$y = \frac{a_0 + a_1 z^{-1} + \ldots + a_m z^{-m}}{1 - b_1 z^{-1} - b_2 z^{-2} - \ldots - b_n z^{-n}} \; x = \frac{A(z)}{B(z)} \; x \qquad (2)$$

This representation can be expressed using D, another notation for the delay operator, as shown in (3). Time is measured in integer steps by the index i.

$$y(i) = \frac{A(D)}{B(D)} \; x(i) = \frac{(a_0 + a_1 D + \ldots + a_m D^m) \; x(i)}{(1 - b_1 D - \ldots - b_n D^n} \qquad (3)$$

where

$$D^k x(i) = x(i-k)$$

Rearranging (3) yields the familiar recursive formulation of the filter equation, as shown in (4)

$$y(i) = (a_0 + a_1 D + \ldots + a_m D^m) \; x(i) + (b_1 D + \ldots + b_n D^n) \; y(i) \qquad (4)$$

Equation (4) leads directly to the canonical representation of the filter as shown in figure 3, assuming the availability of a multiply-add unit with unit delay (one clock pulse period delay). This realization actually does not yield y, but in fact produces Dy, y delayed by one clock pulse period. The canonical form shown in this figure can be realized, but unfortunately because the multiply-add unit is very complex, the realization will be quite slow. If we assume 12 bits of significance to the representation of the numbers in the filter, we find that the unit has two 1-input, 12-bit constant multipliers

feeding a three-input by 12-bit adder. All of this complexity forces the clock pulse period to be very long, and the clock or sampling rate as a consequence to be low. A typical value or clock rate using state-of-the-art Schottkey integrated circuit PROMs and adders would be on the order of 10 MHz. A slight improvement in the rate of operation is possible if we use the basic module shown in figure 4a. A delay of 1 clock pulse period is produced by the output register. This module could be built to operate at a clock rate of perhaps 12 MHz, compared with the realization of figure 3.

What is desired is some means to take advantage of higher clock rate pipeline realization of the basic multiply-add module, thus producing a higher sampling rate realization of the recursive digital filter.

In order to increase the clock rate of the multiply-add unit, we first use the two input version, and then insert several registers for intermediate results in the coefficient multiply and adder sections. For example, a 2-stage version is shown in Figure 4b that could be made to operate at 16 MHz. The net result is a $k_m + k_a$-stage (register) pipeline multiply-add unit as shown in figure 5, with k_m stages involved in the coefficient multiply and k_a stages in the adder.

When we attempt to use the module of figures 4 or 5, we find it impossible to insert it in the feedback loop, preserving a minimum delay of only one. So, if we are to use the pipeline module, some other way must be found.

For the purposes of the derivation which follows, we will restrict our consideration to second-order filters, that is m=2, n=2. This in no way limits the result, for the process used is not dependent on n, except for the value of the individual coefficients. On the other hand, higher order filters can also be realized as the cascade of $\lceil n/2 \rceil$ 2nd order filters, and at most one filter of lower order.

Now we let n=2 and consider that specific case of

$$\frac{Y}{X} = \frac{A(z^{-1})}{B(z^{-1})} = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2}}{1 - b_1 z^{-1} - b_2 z^{-2}} \tag{5}$$

or represented in terms of the delay operator D, the usual computer representation of delay.

$$\frac{x(i)}{y(i)} = \frac{a_0 + a_1 D + a_2 D^2}{1 - b_1 D - b_2 D^2} \tag{6}$$

Writing this in a different form we have

$$y(i) = (a_0 + a_1 D + a_2 D^2)x(i) + (b_1 D + b_2 D^2)y(i) \tag{7}$$

which can also be written in difference equation form:

$$y_i = a_0 x_i + a_1 x_{i-1} + a_2 x_{i-2} + (b_1 y_{i-1} + b_2 y_{i-2}) \tag{8}$$

For subsequent development we will use (7); furthermore, we will omit the (i) notation. We will relate that form to the other forms where appropriate.

Let us first ignore the A(D) polynomial by substituting

$$x' = A(D)x \tag{9}$$

into (7) yielding

$$y = x' + (b_1 Dy + b_2 D^2 y) \tag{10}$$

Delaying (10) and substituting for Dy in (9) we obtain

$$y = x' + b_1(Dx' + b_1 D^2 y + b_2 D^3 y) + b_2 D^2 y$$

or

$$y = (1 + b_1 D)x' + (b_1^2 + b_2)D^2 y + b_1 b_2 D^3 y \tag{11}$$

(11) is now a third order difference equation describing the <u>same</u> digital filter. That is to say, the filter described by (11) has the same transfer characteristic as the one described by (7) or (10). Delaying (10) by D^2 and substituting into (11) yields

$$\begin{aligned} y = {} & [1 + b_1 D + (b_1^2 + b_2)D^2]x' \\ & + (2b_1 b_2 + b_1^3)D^3 y + (b_2^2 + b_1^2 b_2)D^4 y \end{aligned} \tag{12}$$

In general, successively more delayed versions of (10) can be substituted, raising the order of the difference equation, but, more importantly, raising also the minimum delay associated with the feedback y terms. The general higher order difference equation has the following general form, provided we started from a second-order equation:

$$\begin{aligned} y = {} & [1 + \alpha_1^{(p)} D + \alpha_2^{(p)} D^2 + \ldots + \alpha_p^{(p)} D^p]x' \\ & + b_{p+1}^{(p)} D^{p+1} y + b_{p+2}^{(p)} D^{p+2} y \end{aligned} \tag{13}$$

In fact, if we began with an nth order difference equation, we would have an equation of p+n order resulting from the foregoing process as shown in (14)

$$\begin{aligned} y = {} & [1 + \alpha_1^{(p)} D + \ldots + \alpha_p^{(p)} D_p]x' \\ & + b_{p+1}^{(p)} D^{p+1} y + \ldots + b_{p+n}^{(p)} D^{p+n} y \end{aligned} \tag{14}$$

The coefficients $a_i^{(p)}$ and $b_i^{(p)}$ can be calculated by following the process described above in detail. For example, it should be clear from (12) that

$$\alpha_1^{(2)} = b_1 \; ; \; \alpha_2^{(2)} = b_1^2 + b_2$$

$$b_3^{(2)} = 2b_1b_2 + b_1^3 \; ; \; b_4^{(2)} = b_2^2 + b_1^2b_2 \tag{15}$$

where the original b coefficients are from (10). We will cover the process in detail a little later in this section.

Now let us recall what x' is in terms of the original IIR filter as described in (7). Substituting $x' = A(D)x$ into (13) yields:

$$y = [1 + \alpha_1^{(p)} D + \ldots + \alpha_p^{(p)} D^p] [a_0 + a_1D + a_2D^2]x$$

$$+ b_{p+1}^{(p)} D^{p-1} y + b_{p+2}^{(p)} D^{p+2} y \tag{16}$$

This equation when multiplied out for the case of p=2 yields (17)

$$y = [a_0^{(2)} + a_1^{(2)} D + \ldots + a_4^{(2)} D^4]x$$

$$+ [b_3^{(2)} D^3 y + b_4^{(2)} D^4 y] \tag{17a}$$

and

$$w' = D^4 D^\Delta [a_0^{(2)} + a_1^{(2)} D + \ldots + a_4^{(2)} D^4]x$$

$$+ [b_3^{(2)} D^3 w' + b_4^{(2)} D^4 w'] \tag{18a}$$

These two equations look very similar now, the only difference being that the non-recursive parts (18a) is the non-recursive part of (17a) delayed by $\Delta + 4$. If both sides of (17a) are delayed by that amount, (17b) results:

$$D^{\Delta+4} y = D^{\Delta+4} [a_0^{(2)} + a_1^{(2)} D + \ldots + a_4^{(2)} D^4]x$$

$$+ [b_3^{(2)} D^{\Delta+7} y + b_4^{(2)} D^{\Delta+8} y] \tag{17b}$$

Finally, it should be noted that $D^{\Delta+4}y$ in (17b) is the same as w' in (18a). substituting $D^{\Delta+4}y = w'$ in (17b) produces (17c):

$$w' = D^{\Delta+4}\,[a_0^{(2)} + a_1^{(2)} + \ldots + a_4^{(2)}D^4]x$$
$$+ [b_3^{(2)}D^3\,w' + b_4^{(2)}D^4\,w'] \qquad (17c)$$

which is the same as (18a).

What this all means is that the structure of figure 6 will produce an output w' which is simply the desired signal y, delayed by $\Delta+4$ sampling periods. This digital filter structure uses pipeline logic units in both the recursive and non-recursive portions to realize the desired output.

The design still needs to be completed, even though the heart of the structure has been developed. Figure 7 shows the structure of the non-recursive portion of the filter to complete the example.

The circuit of figure 7 produces $D^4[a_0 + a_1D + \ldots + a_4D^4]x$ which corresponds to the $D^{\Delta}[a_0 + a_1D + \ldots + a_4D^4]x$ term required as the output of the non-recursive part of figure 7. Therefore, the pipeline digital filter of figure 7 combined with figure 6 produces $D^{4+4}y$, that is y, delayed by 8 sample periods.

Now we return to the calculation of the $a_i^{(p)}$ and $b_j^{(p)}$ coefficients. More general relations than (15) can be developed in difference equation form with respect to p.

Starting with the p augmented order (p+2 order) difference equation (13) we can derive the p+1 augmented order equation by substituting for $D^{p+1}y$ [(10) delayed by D^{p+1}] into (13).

$$y = [1 + \alpha_1^{(p)} D + \ldots + \alpha_p^{(p)} D^p]x'$$

$$+ b_{p+1}^{(p)} [D^{p+1} x' + (b_1 D^{p+2} y + b_2 D^{p+3} y)]$$

$$+ b_{p+2}^{(p)} D^{p+2} y$$

$$y = [1 + \alpha_1^{(p)} D + \ldots + \alpha_p^{(p)} D^p + b_{p+1}^{(p)} D^{p+1}]x'$$

$$+ (b_{p+1}^{(p)} b_1 + b_{p+2}^{(p)}) D^{p+2} y + b_{p+1}^{(p)} b_2 D^{p+3} y \tag{19}$$

From (19) we can see that

$$\alpha_{p+1}^{(p+1)} = b_{p+1}^{(p)} \tag{20a}$$

$$b_{p+2}^{(p+1)} = b_{p+1}^{(p)} b_1 + b_{p+2}^{(p)} \tag{20b}$$

$$b_{p+3}^{(p+1)} = b_{p+1}^{(p)} b_2 \tag{20c}$$

Where $\alpha_0^{(p)} = 1$, $b_1^{(0)} = b_1$, and $b_2^{(0)} = b_2$.

Table 1 shows the values of the coefficients of (13) for the values of p from 0 through 5 as derived from (20). Furthermore, multiplication of the pth order polynomial

$$\alpha_0^{(p)} + \alpha_1^{(p)} D^1 + \ldots + \alpha_p^{(p)} D^p$$

by the original

$$a_0 + a_1 D + a_2 D^2$$

yields the polynomial

$$a_0^{(p)} + a_1^{(p)} D + \ldots + a_{p+2}^{(p)} D^{p+2}$$

as shown in (16) and (18). Table 2 shows the values of $a_j^{(p)}$ for p from 0 through 5.

The unique structure of this realization, which differs from conventional realizations of digital filters is contained in the recursive portion, where the basic feedback loop involving the longest delay passes through p+2 pipeline stages, for the realization of a 2nd order filter. In the example developed, p was 2, causing us to use a 4th order pipeline filter representation of the original 2nd order filter.

In general, we assume we have k_a stage pipeline add units and k_m stage pipeline multipliers. The structures of the ralization of the feedback portion of the filter would be as shown in figure 8. Since the maximum delay around the loop must match the p augmented difference equation order, we have

$$p = 2k_a + k_m - 2 \tag{21}$$

The FIR portion of the filter is shown in general terms in figure 14. From this figure, it should be clear that

$$\Delta = \lceil \log_2(k_a) \rceil k_a + k_a + k_m$$

and that the over all delay is therefore

$$\Delta + 2k_a = \lceil \log_2(k_a) \rceil k_a + 3k_a + k_m \tag{22}$$

STABILITY ISSUES

In the previous section we have seen the general process of realization of a digital filter transfer function using pipelinemultiply-add units with delay 2. We have also shown a realization using 2-stage multiply-add units. The essence of the techique is to represent a second-order section by means of a $(2+p)^{th}$ order section of a particular form, where p is determined by the number of stages in the pipeline multiply and add units.

We know from other considerations that the transfer function of the higher order realization must in fact be the same as the original and therefore must have the same poles and zeros in the z plane. Thus, the operation which transforms (7) into (17) can be thought of in the following way:

$$H'(z^{-1}) = \frac{(a_0 + a_1 z^{-1} + a_2 z^{-2})}{(1 - b_1 z^{-1} - b_2 z^{2})} \frac{(1 + \alpha_1^{(p)} z^{-1} + \ldots + \alpha_p^{(p)} z^{-p})}{(1 + \alpha_1^{(p)} z^{-1} + \ldots + \alpha_p^{(p)} z^{-p})} \tag{23}$$

$$= \frac{a_0^{(p)} + a_1^{(p)} z^{-1} + \ldots + a_{p+2}^{(p)} z^{-p-2}}{1 - b_{p+1}^{(p)} z^{-p-1} - b_{p+2}^{(p)} z^{-p-2}} \tag{24}$$

Where the coefficients of the α polynomial depend on the original b_1 and b_2 of (5) and (7). These α coefficients are governed by (20) and some were tabulated in Table 1b. Thus, we raise the order of the denominator, introducing p new poles and corresponding cancelling zeros. These new poles correspond to the roots of

$$0 = 1 + \alpha_1^{(p)} D + \ldots + \alpha_p^{(p)} D^p \tag{25}$$

The roots of $\alpha(D)$ of course are the poles in the z^{-1} plane and are the reciprocal of the poles in z plane.

One concern in the realization is that the filter be stable, that is, that it have an impulse response which decays to zero. A sufficent condition for the stability of a digital filter with transfer function $H(z^{-1})$ is that the poles of $H(z^{-1})$ in the z^{-1} plane lie <u>outside</u> the unit circle, and that the order of the numerator (-m) be less than or equal in magnitude to the order (-n) of the denominator. A more familiar sufficient condition is that the poles of the transfer function of z (H(z)) be <u>inside</u> the unit circle.

Now, if we start with a stable filter, we would like to be assured that the poles of the augmented order filter all be outside the unit circle in the z^{-1} plane. We desire this because, even though the added poles are cancelled by the zeros of $(1 + \alpha^{(p)}z^{-1} + \ldots + \alpha^{(p)}z^{-p})$, realization imperfections will prevent exact cancellation and the augmented filter would be unstable.

In order to examine the stability question, we will make use of Jury's stability test [6] as applied to the successively higher degree denominator polynomials in z as p increases.

We start with the denominator of (5), written as a polynomial in z

$$P(z) = z^2 - b_1 z - b_2 \tag{26}$$

To insure that the <u>original</u> poles of (5) are inside the unit circle and hence that the original filter is stable, we apply Jury's test with necessary conditions:

$$\left.\begin{array}{l} P(1) = 1 - b_1 - b_2 > 0 \\ (-1)^n P(-1) = 1 + b_1 - b_2 > 0 \\ \text{and sufficient condition:} \\ |b_2| < 1 \end{array}\right\} \tag{27}$$

Hence, for this second-order digital filter to be stable, the values of b_1 and b_2 must be in the shaded area as shown in Fig. 10. Note that this is the case when p = 0, i.e., no augmentation.

Now, assuming that the original filter transfer function (5) is stable, i.e., (27) is satisfied, it is desirable to determine conditions to assure

stability of the new p poles introduced in the system when the transfer function is p-augmented. For p = 1, the new polynomial introduced in the numerator and denominator is

$$F(z) = z + b_1 \tag{28}$$

Applying Jury's test, conditions for stability are

$$\left.\begin{aligned} F(1) &= 1 + b_1 > 0 \\ (-1)^n F(-1) &= 1 - b_1 > 0 \end{aligned}\right\} \tag{29}$$

This condition, $|b_1| < 1$, is shown graphically in figure 11. Similarly, for p = 2, the added polynomial is

$$F(z) = z^2 + b_1 z + (b_1^2 + b_2) \tag{30}$$

Jury's test gives conditions for stable roots. The necessary conditions are

$$\left.\begin{aligned} F(1) &= 1 + b_1 + (b_1^2 + b_2) > 0 \\ (-1)^n F(-1) &= 1 - b_1 + (b_1^2 + b_2) > 0 \\ \text{and the sufficient condition is} \\ |b_1^2 + b_2| &< 1 \end{aligned}\right\} \tag{31}$$

The region defined by conditions (31) is shown graphically in figure 12. Finally, for p = 3

$$F(z) = z^3 + b_1 z^2 + (b_1^2 + b_2)z + (b_1^3 + 2b_1 b_2)$$

and the conditions for stable poles are (figure 13):

$$\left.\begin{aligned}
&F(1) = 1 + b_1 + (b_1^2 + b_2) + (b_1^3 + 2b_1b_2) > 0 \\
&(-1)^n F(-1) = 1 - b_1 + (b_1^2 + b_2) - (b_1^3 + 2b_1b_2) > 0 \\
&\left| b_1^3 + 2b_1b_2 \right| < 1 \\
&\left| (b_1^3 + 2b_1b_2)^2 - 1 \right| > \left| b_1(b_1^3 + 2b_1b_2) - (b_1^2 + b_2) \right|
\end{aligned}\right\} \qquad (32)$$

This procedure can be continued for higher values of p; in fact general stability tests based on Jury's test have been developed [6]. Unfortunately, those tests are cumbersome to apply.

Instead, let us take another approach.

From figures 10-13 it may be observed that as the value of p increases, the area of stability (the shaded areas in the figures) tends to increase. It was seen graphically that as p increased, the stable region came to be closer and closer to the original shaded area for p = 0, i.e., figure 10. This leads to the obvious conjecture that all stable filters have a stable augmented filter for some large enough p.

We will follow the procedure suggested by Voelcker [7] for Block Filters to show that for a large value of augmentation, p, the pipelined version of the original filter will always be stable, provided that the original second order filter is stable. This is a very important and useful conclusion and implies that an augmented stable realization is always possible if the original transfer function is stable. The proof for this fact follows:

Since the process of augmentation implies multiplying the numerator and the denominator by the same polynomial, the original transfer function may be equated to the augmented transfer function as

$$\frac{N(z)}{1 - b_1 z^{-1} - b_2 z^{-2}} = \frac{N(z)(\alpha_0^{(p)} + \alpha_1^{(p)} z^{-1} + \ldots + \alpha_p^{(p)} z^{-p})}{1 - z^{-(p+1)}(b_{p+1}^{(p)} + b_{p+2}^{(p)} z^{-1})}$$

where $N(z)$ is the numerator of the original transfer function. Denoting the denominator of the original transfer function as $D(z)$, since this original filter is stable, the roots of $D(z)$ are inside the unit circle. Figure 14(a) shows the original filter where $H_p(z)$ is a pole only function representing the denominator, $(1/D(z))$ of the original filter.

The process of augmenting the difference equation suggests building the filter as shown in figure 14(b) where

$$\alpha(z) = \alpha_0^{(p)} + \alpha_1^{(p)} z^{-1} + \ldots + \alpha_p^{(p)} z^{-p}$$

and $$G(z) = b_{p+1}^{(p)} + b_{p+2}^{(p)} z^{-1}$$

Thus $$H_p(z) = \frac{1}{D(z)} = \frac{\alpha(z)}{1 - z^{-(p+1)} G(z)} \tag{33}$$

Hence, the nonrecursive transfer function $\alpha(z)$ may be written as

$$\alpha(z) = H_p(z) - z^{-(p+1)} \frac{G(z)}{D(z)} \tag{34}$$

Since $H_p(z)$ is a pole only transfer function

$$H_p(z) = \frac{1}{\prod_{i=1}^{2} 1 - \frac{z_i}{z}}$$

$$= \sum_{i=1}^{2} \frac{a_i}{1 - \frac{z_i}{z}}$$

Taking the inverse transform

$$h_p(n) = \sum_{i=1}^{2} a_i z_i^n$$

Since z_1 and z_2 are the initial poles, they are inside the unit circle. Hence $h_p(n)$ is an infinite sequence decreasing in magnitude. If the sequence is truncated at the (p + 1) term, the remaining portion of the sequence

$$\hat{h}_p(n) = \begin{cases} \sum_{i=1}^{2} a_i z_i^n & n = 0, 1, \ldots, p \\ 0 & \text{otherwise} \end{cases}$$

Taking the z-transform

$$\hat{H}_p(z) = \sum_{n=0}^{p} \hat{h}_p(n) z^{-n}$$

$$= \sum_{n=0}^{p} \sum_{i=1}^{2} a_i z_i^n z^{-n}$$

$$= \sum_{i=1}^{2} a_i \sum_{n=0}^{p} \left(\frac{z_i}{z}\right)^n$$

$$= \sum_{i=1}^{2} a_i \left[\frac{1 - (z_i/z)^{p+1}}{1 - (z_i/z)}\right]$$

$$= \sum_{i=1}^{2} \frac{a_i}{1 - (z_i/z)} - \sum_{i=1}^{2} \frac{a_i (z_i/z)^{p+1}}{1 - (z_i/z)}$$

Thus,

$$\hat{H}_p(z) = H_p(z) - z^{-(p+1)} \sum_{i=1}^{2} \frac{a_i z_i^{p+1}}{1 - (z_i/z)} \tag{35}$$

Comparing (34) and (35) and, since $\hat{H}_p(z)$ is a finite impulse response transfer function, equating $\hat{H}_p(z)$ and $\alpha(z)$

$$\frac{G(z)}{D(z)} = \sum_{i=1}^{2} \frac{a_i z_i^{p+1}}{1 - z_i/z}$$

Thus

$$G(z) = \sum_{i=1}^{2} \frac{a_i z_i^{p+1} D(z)}{1 - (z_i/z)} \tag{36}$$

The intent of this procedure is to prove that, for some high value of p, all roots of $\alpha(z)$ are inside the unit circle in the z-domain. From (33)

$$D(z)\alpha(z) = 1 - z^{-(p+1)} G(z) \tag{37}$$

Since D(z) is the denominator of the original transfer function, all roots of D(z) are inside the unit circle. Hence all roots of $\alpha(z)$ are also inside if, and only if, all roots of the right hand expression of (37) are inside the unit circle. To do this, let tilde (~) denote the result of the mapping $z^{-1} \rightarrow z$, i.e., $\tilde{f}(z) = f(z^{-1})$ and $\tilde{A}(z)$ is defined as

$$\tilde{A}(z) = 1 - z^{(p+1)} \tilde{G}(z) = \tilde{D}(z)\tilde{\alpha}(z) \tag{38}$$

All roots of $\tilde{D}(z)$ are exterior (outside the unit circle). Hence, all roots of $\alpha(z)$ are exterior if all roots of $\tilde{A}(z)$ are exterior. To prove this, Rouche's theorem is used which states that: "If f(z) and g(z) are analytic inside and on a closed contour C, and $|g(z)| < |f(z)|$ on C, then f(z) and f(z) + g(z) have

the same number of zeroes inside C." Clearly, $\tilde{A}$ in (38) is analytic within and on the unit circle. The constant "1" has no interior zeroes. Hence A(z) will have no interior zeroes if, on the unit circle,

$$\left| z^{(p+1)}\tilde{G}(z) \right| < 1 \quad , \quad z = e^{j\theta} \tag{39}$$

From (4.20),

$$\left| z^{(p+1)}\tilde{G}(z) \right|_{z=e^{j\theta}} = e^{j(p+1)\theta} \sum_{i=1}^{2} \frac{a_i z_i^{(p+1)}\tilde{D}(z)}{1 - z_i e^{j\theta}} \tag{40}$$

But $|z_i| < 1$ for all i because these are poles of the original filter. Thus, for some value of p greater than some critical value, equation (40) will satisfied. Note that (40) may be satisfied for smaller value of p also.

From the above discussion, it may be concluded that for some high value of augmentation, it is always possible to ensure that $\alpha(z)$ has roots inside the unit circle. Since these roots are the new additional poles of the augmented system, the new high order transfer function would be stable.

As a result of this fact, the following design method is suggested.

We assume that the designer is given a stable recursive digital filter transfer function H(z) and a desired sampling (clock) rate of operation, f_s.

1. Find k_m stage pipeline multiply and k_a stage pipeline add units that will operate at clock frequency f_s.

2. Factor H(z) into ℓ (ℓ - 1 if n is odd) 2nd order (and if n is odd, at most one 1st order) transfer functions.

$$H(z) = H_1(z) \cdot H_2(z) \ldots H_\ell(z)$$

where $\ell = \lceil n/2 \rceil$

3. For each $H_i(z)$, i = 1, 2, ... ℓ. Realize $H_i(z)$ as follows:

a. Set $p = 2k_a + k_m - 2$,

b. Determine the polynomial $1 - b_{p+1}^{(p)} z^{-p-1} - b_{p+2}^{(b)} z^{-p-2} = B_p(z^{-1})$,

c. Solve for the roots of $B_p(z^{-1}) = 0$ (41)

d. If all roots of $B_p(z^{-1}) = 0$ are outside the unit circle, go to e, else set $p = p + 1$ and return to step 3b.

e. $H_1(z)$ can be realized by a stable p augmented order difference equation

$$H_1(z^{-1}) = \frac{a_0^{(p)} + a_1^{(p)} z^{-1} + \ldots + a_{p+2}^{(p)} z^{-p-2}}{1 - b_{p+1}^{(p)} z^{-p-1} - b_{p+2}^{(p)} z^{-p-2}} \tag{24}$$

where $p \geq 2k_a + k_m - 2$

Because the last result, we know that we can always find a large enough p so that (24) will be stable.

In the next section, we will illustrate the method with an example filter taken from the literature.

EXAMPLE

We will treat the Butterworth filter considered by Oppenheim and Shafer [8]. This is a sixth order filter whose original transfer function H(s) is given by

$$H(s) = \frac{0.20238}{(s^2+0.396s+0.5871)(s^2+1.083s+0.5871)(s^2+1.4802s+0.5871)} \tag{42}$$

Applying the bi-linear transformation, a sixth order difference equation is obtained that can be factored in three second order filters.

$$H(z) = H_1(z) \cdot H_2(z) \cdot H_3(z) \tag{43}$$

where

$$H_1(z) = \frac{.0007378\ (1 + 2z^{-1} + z^{-2})}{(1 - 1.2686\ z^{-1} + 0.7051\ z^{-2})} \tag{44a}$$

$$H_2(z) = \frac{(1 + 2z^{-1} + z^{-2})}{(1 - 1.0106\ z^{-1} + 0.3583\ z^{-2})} \tag{44b}$$

$$H_3(z) = \frac{(1 + 2z^{-1} + z^{-2})}{(1 - 0.9044\ z^{-1} + 0.2155\ z^{-2})} \tag{44c}$$

The poles in the z^{-1}plane of these equations are as follows:

$$P_1 = 0.8996 \pm j.7804 \tag{45a}$$

$$|P_1| = 1.1909$$

$$P_2 = 1.4103 \pm j.8956 \tag{45b}$$

$$|P_2| = 1.6706$$

$$P_3 = 2.0984 \pm j.4870 \tag{45c}$$

$$|P_3| = 2.1542$$

Thus all three of the factoring second order transfer functions are stable.

p augmented order difference equations of the form of (16) were derived for each of the three sections, for values of p from 1 through 5. H_1 and H_2 were unstable for p=1 and stable for all values of p from 2 through 5. H_3 was stable for all values of p from 1 through 5.

If we had a 2 stage adder and a 2 stage multiplier, (21) would yield a p of 4, which for our example adds stable poles to each section. The general structure of each of the 2nd order (4-augmented) sections would be as shown in figure 15.

Finally, the values for the coefficient of each of the multipliers would be as shown in table 3.

Note that the $a^{(4)}$ coefficient for the first section are all on the order of .001 in magnitude. These coefficient could easily be computed to more significant figures, although realization of more significant figures in the coefficient of an actual filter would require either scaling or the use of a floating point number system.

This example shows how an IIR filter can be realized using high rate pipeline modules, with the ultimate objective of achieving a higher sample rate than possible with non-pipelined multiplier and adders.

SUMMARY AND CONCLUSIONS

In this paper we have developed a method for applying pipeline techniques to the design of high speed recursive digital filters. Using these techniques, recursive digital filters operating at rates hitherto impossible can be designed. The general structure of the filter and the method for calculating the multiplier coefficients is presented. The stability of the resulting realization has been investigated and a technique for ascertaining the stability of the realization is presented.

A significant example taken from the literature illustrates the technique and demonstrates the existance of a stable, high rate pipeline realization of a practically useful filter.

REFERENCES

1. Loomis, H. H. Jr., "The Maximum Rate Accumulator," IEEE TEC, EC-15, Number 4, Aug. 1966, pp. 628-639.
2. Chen, T. C., "Overlap and Pipeline Processing," INTRODUCTION TO COMPUTER ARCHITECTURE, H Stone Editor, Science Research Associate, Chicago, Chapter 9.
3. Cray Research, Inc., CRAY-1 HARDWARE REFERENCE MANUAL, Cray Research Publication No. 2240004, Rev E, May 15, 1979.
4. M. A. Soderstrand and E. L. Fields, "A High Speed Low-Cost Recursive Digital Filter Using Residue Number Arthmetic," PROC IEEE, July 1977.
5. Sinha, B., PIPELINED FINITE STATE MACHINE ARCHITECTURE APPLIED TO DIGITAL FILTERS, Ph.D. Dissertation, University of California, Davis, CA, September 1983.
6. Nagrath, I. J. and Gopal, M., CONTROL SYSTEMS ENGINEERING, Halsted Press, John Wiley and Sons, 1982.
7. Voelcker, H. B. and Hartquest, E. E., "Digital Filtering via Block Recursion," IEEE TRANSACTIONS on Audio and Electroacoustics, June 1970.
8. Oppenheim, A. and Shafer, R., DIGITAL SIGNAL PROCESSING," Prentice Hall, Englewood Cliffs, NJ, 1975.

	Coeff of D^{p+1}	Coeff of D^{p+2}
p	$b_{p+1}^{(p)}$	$b_{p+2}^{(p)}$
0	b_1	b_2
1	$b_1^2 + b_2$	$b_1 b_2$
2	$b_1^3 + 2b_1 b_2$	$b_1^2 b_2 + b_2^2$
3	$b_1^4 + 3b_1^2 b_2 + b_2^2$	$b_1^4 b_2 + 3b_1^2 b_2^2 + b_2^3$
4	$b_1^5 + 4b_1^3 b_2 + 3b_1 b_2^2$	$b_1^4 b_2 + 3b_1^2 b_2^2 + b_2^3$
5	$b_1^6 + 5b_1^4 b_2 + 6b_1^2 b_2^2 + b_2^3$	$b_1^5 b_2 + 4b_1^3 b_2^2 + 3b_1 b_2^3$

Table 1a. b's for p = 0, 1, ...5.

p	$\alpha_0^{(p)}$	$\alpha_1^{(p)}$	$\alpha_2^{(p)}$	$\alpha_3^{(p)}$	$\alpha_4^{(p)}$	$\alpha_5^{(p)}$
0	1	0	0	0	0	0
1	1	b_1	0	0	0	0
2	1	b_1	$b_1^2 + b_2$	0	0	0
3	1	b_1	$b_1^2 + b_2$	$b_1^3 + 2b_1 b_2$	0	0
4	1	b_1	$b_1^2 + b_2$	$b_1^3 + 2b_1 b_2$	$b_1^4 + 3b_1^2 b_2 + b_2^2$	0
5	1	b_1	$b_1^2 + b_2$	$b_1^3 + 2b_1 b_2$	$b_1^4 + 3b_1^2 b_2 + b_2^2$	$b_1^5 + 4b_1^3 b_2 + 3b_1 b_2^2$

Table 1b. α's for p = 0, 1, ... 5.

p	$a_0^{(p)}$	$a_1^{(p)}$	$a_2^{(p)}$	$a_3^{(p)}$	$a_4^{(p)}$	$a_5^{(p)}$	$a_6^{(p)}$	$a_7^{(p)}$
0	a_0	a_1	a_2	0	0	0	0	0
1	a_0	$a_0b_1+a_1$	$a_1b_1+a_2$	a_2b_1	0	0	0	0
2	a_0	$a_0b_1+a_1$	$a_0(b_1^2+b_2)$ $+a_1b_1+a_2$	$a_1(b_1^2+b_2)$ $+a_2b_1$	$a_2(b_1^2+b_2)$	0	0	0
3	a_0	$a_0b_1+a_1$	$a_0(b_1^2+b_2)$ $+a_1b_1+a_2$	$a_0(b_1^3+2b_1b_2)$ $+a_1(b_1^2+b_2)$ $+a_2b_1$	$+a_1(b_1^3+2b_1b_2)$ $+a_2(b_1^2+b_2)$	$+a_2(b_1^3+2b_1b_2)$	0	0
4	a_0	$a_0b_1+a_1$	$a_0(b_1^2+b_2)$ $+a_1b_1+a_2$	$a_0(b_1^3+2b_1b_2)$ $+a_1(b_1^2+b_2)$ $+a_2b_1$	$a_0(b_1^4+3b_1^2b_2+b_2^2)$ $+a_1(b_1^3+2b_1b_2)$ $+a_2(b_1^2+b_2)$	$+a_1(b_1^4+3b_1^2b_2+b_2^2)$ $+a_2(b_1^3+2b_1b_2)$	$+a_2(b_1^4+3b_1^2b_2+b_2^2)$	0
5	a_0	$a_0b_1+a_1$	$a_0(b_1^2+b_2)$ $+a_1b_1+a_2$	$a_0(b_1^3+2b_1b_2)$ $+a_1(b_1^2+b_2)$ $+a_2b_1$	$a_0(b_1^4+3b_1^2b_2+b_2^2)$ $+a_1(b_1^3+2b_1b_2)$ $+a_2(b_1^2+b_2)$	$a_0(b_1^5+4b_1^3b_2+3b_1b_2^2)$ $+a_1(b_1^4+3b_1^2b_2+b_2^2)$ $+a_2(b_1^3+2b_1b_2)$	$a_1(b_1^5+4b_1^3b_2+3b_1b_2^2)$ $+a_2(b_1^4+3b_1^2b_2+b_2^2)$	$+a_2(b_1^5+4b_1^3b_2+3b_1b_2^2)$

Table 2. $a^{(p)}$ Coefficients of equation (17)

	$a_0^{(4)}$	$a_1^{(4)}$	$a_2^{(4)}$	$a_3^{(4)}$	$a_4^{(4)}$	$a_5^{(4)}$	$a_6^{(4)}$	$a_5^{(4)}$	$a_6^{(4)}$
$H_1(z^{-1})$	.0007	.0023	.0031	.0023	.0008	-.0003	-.0002	-.5304	.2236
$H_2(z^{-1})$	1.0	3.0106	3.6842	2.6446	1.3525	.4552	.0736	-.0359	-.0264
$H_3(z^{-1})$	1.0	2.9044	3.4112	2.4592	1.4890	.7233	.1867	.0934	-.0402

Table 3. Coefficient of filter section in figure 15 to realize 6th order Butterworth filter.

LIST OF FIGURES

Figure 1. Simple FIR Realization

Figure 2. Realization of A(D)x using 3-stage pipeline multipliers, 2-stage pipeline adders and unit delays

Figure 3. Canonical Realization of a Recursive Filter

Figure 4a. One-stage pipeline multiply-add unit

Figure 4b. One-stage pipeline multiply feeding one-stage add

Figure 5. $k_a + k_m$ stage pipeline multiply-add module

Figure 6. Heart of the pipeline recursive filter realization

Figure 7. Non-recursive portion of pipeline filter realization

Figure 8. Heart of more general pipeline filter realization

Figure 9. Realization of FIR portion of filter

Figure 10. Condition for stability of the original filter

Figure 11. Condition for stability of the augmented filter: p=1

Figure 12. Condition for stability of the augmented filter: p=2

Figure 13. Condition for stability of the augmented filter: p=3

Figure 14. (a) Original transfer function (b) Equivalent augmented transfer function

Figure 15. Sixth-order pipeline realization of second-order recursive filter

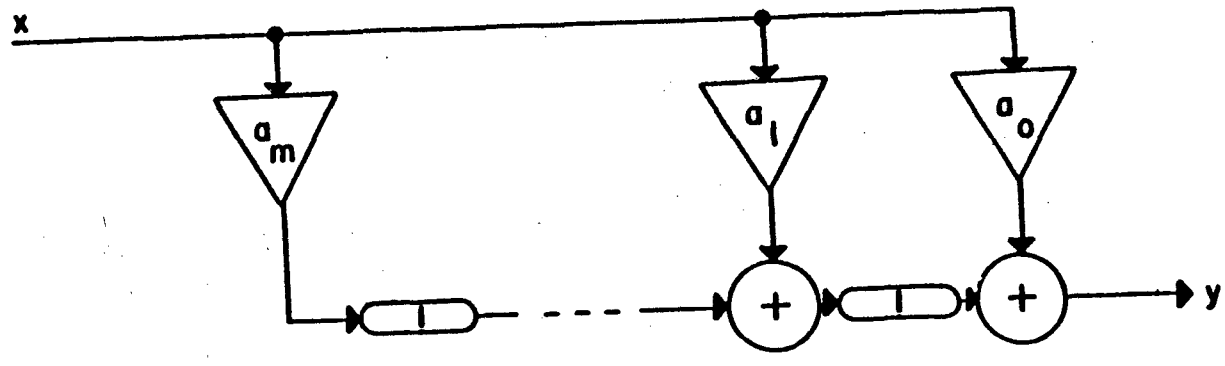


Figure 1. Simple FIR Realization

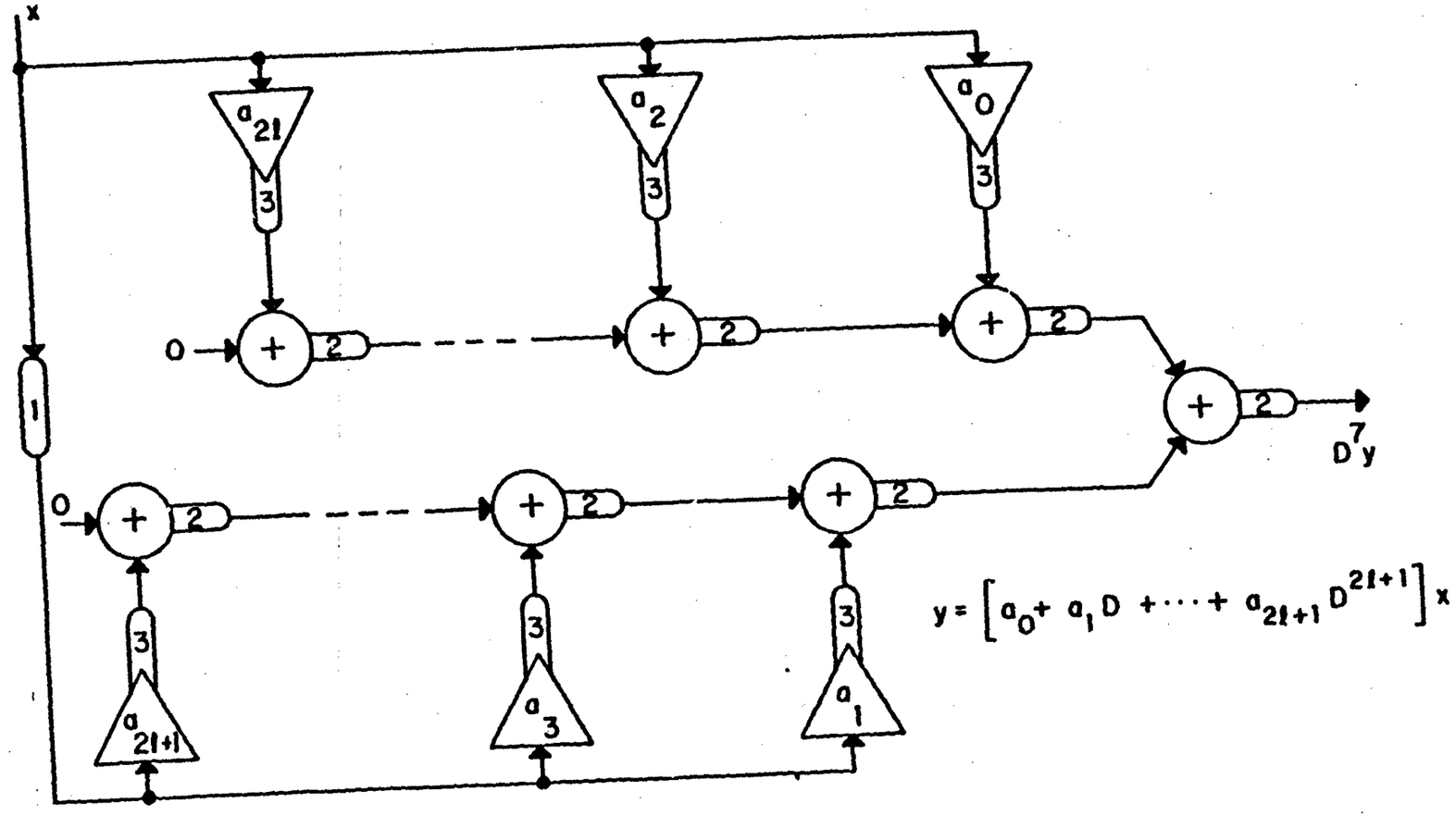


Figure 2. Realization of A(D)x using 3-stage pipeline multipliers, 2-stage pipeline adders and unit delays

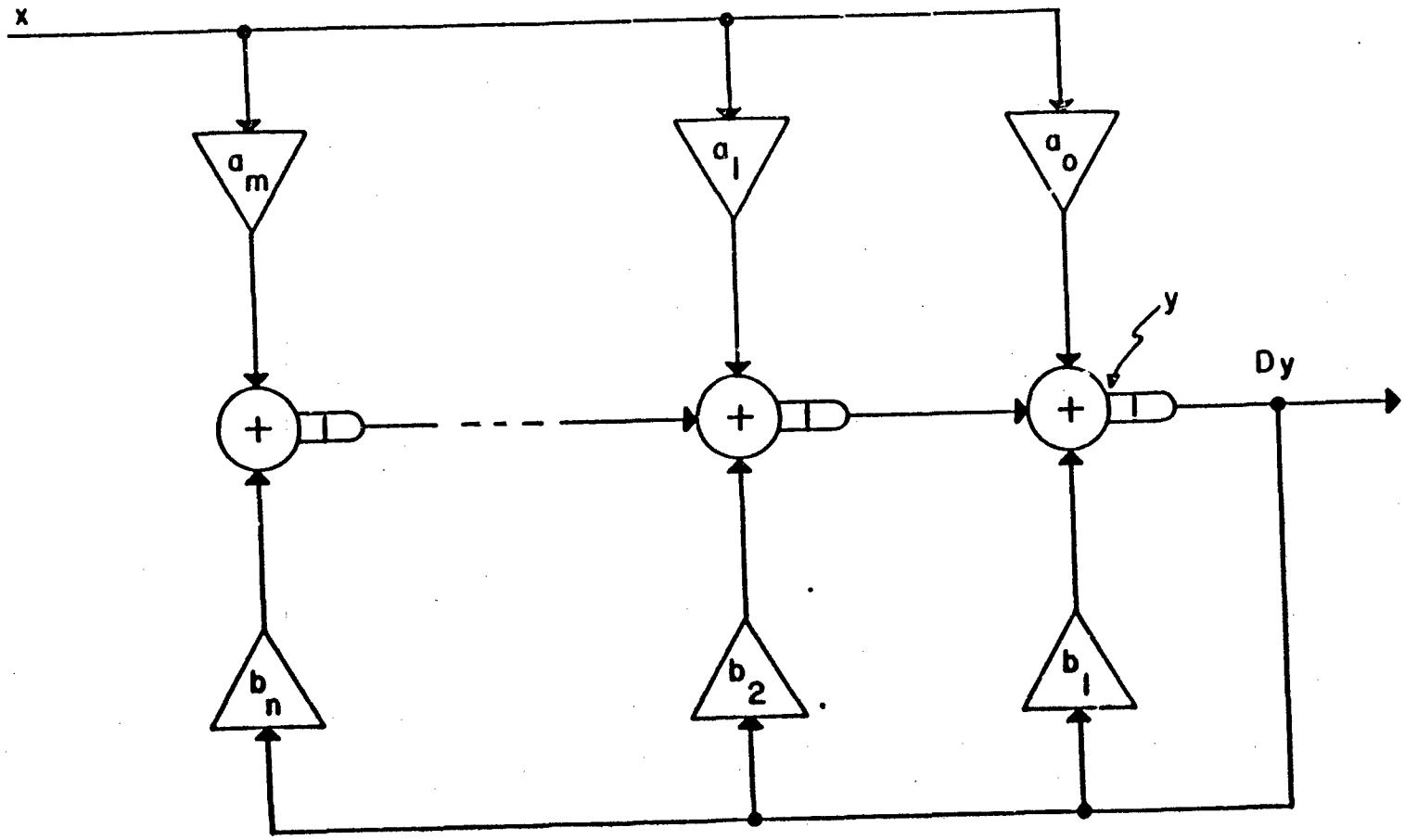


Figure 3. Canonical Realization of a Recursive Filter

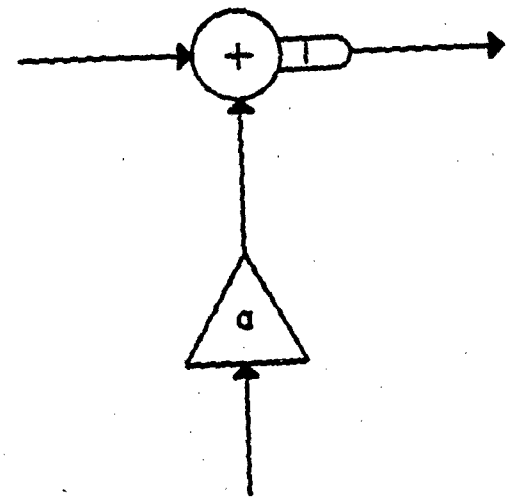


Figure 4a. One-stage pipeline multiply-add unit

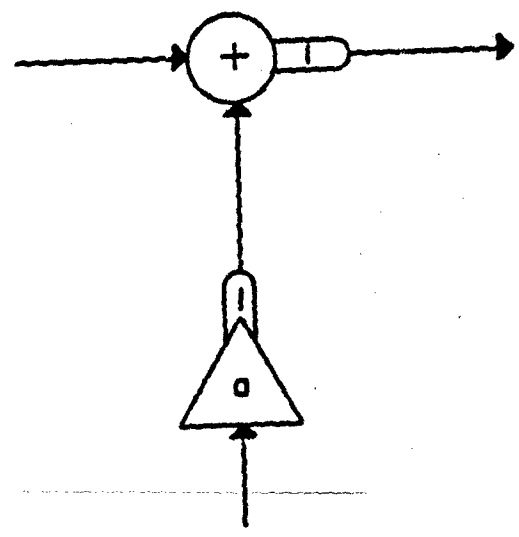


Figure 4b. One-stage pipeline multiply feeding one-stage add

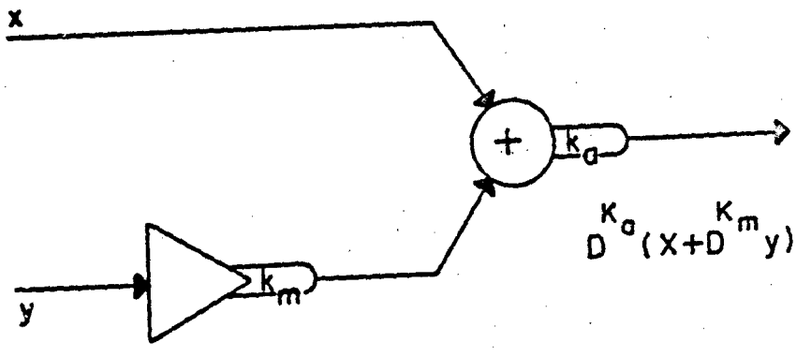


Figure 5. $k_a + k_m$ stage pipeline multiply-add module

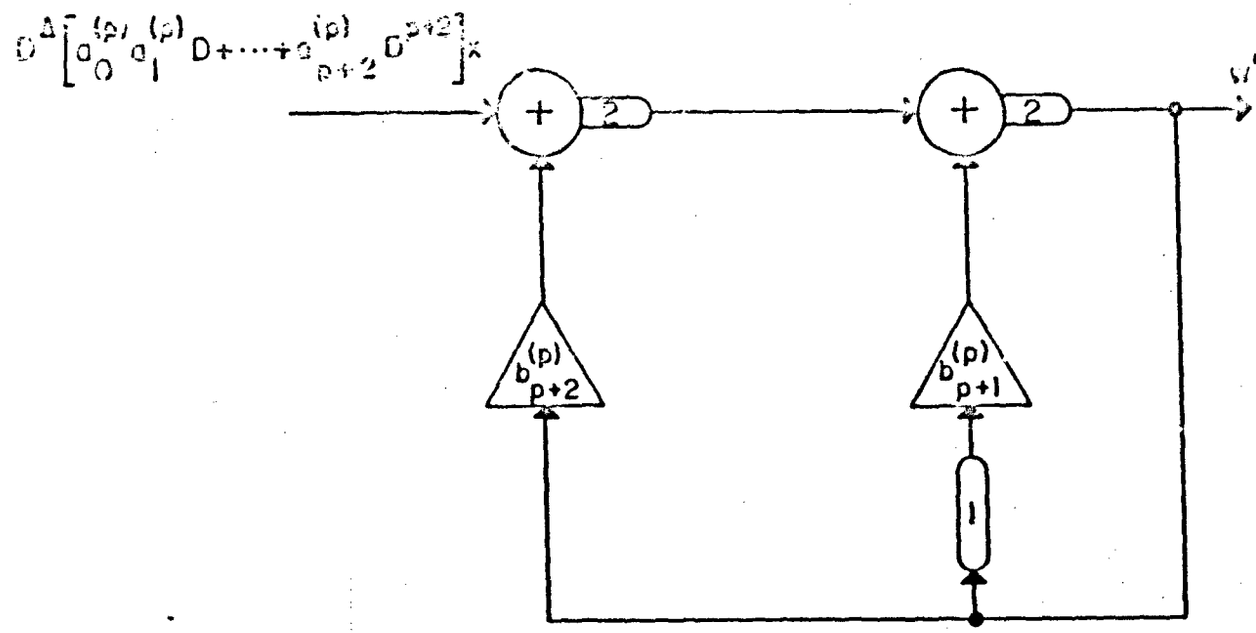


Figure 6. Heart of the pipeline recursive filter realization

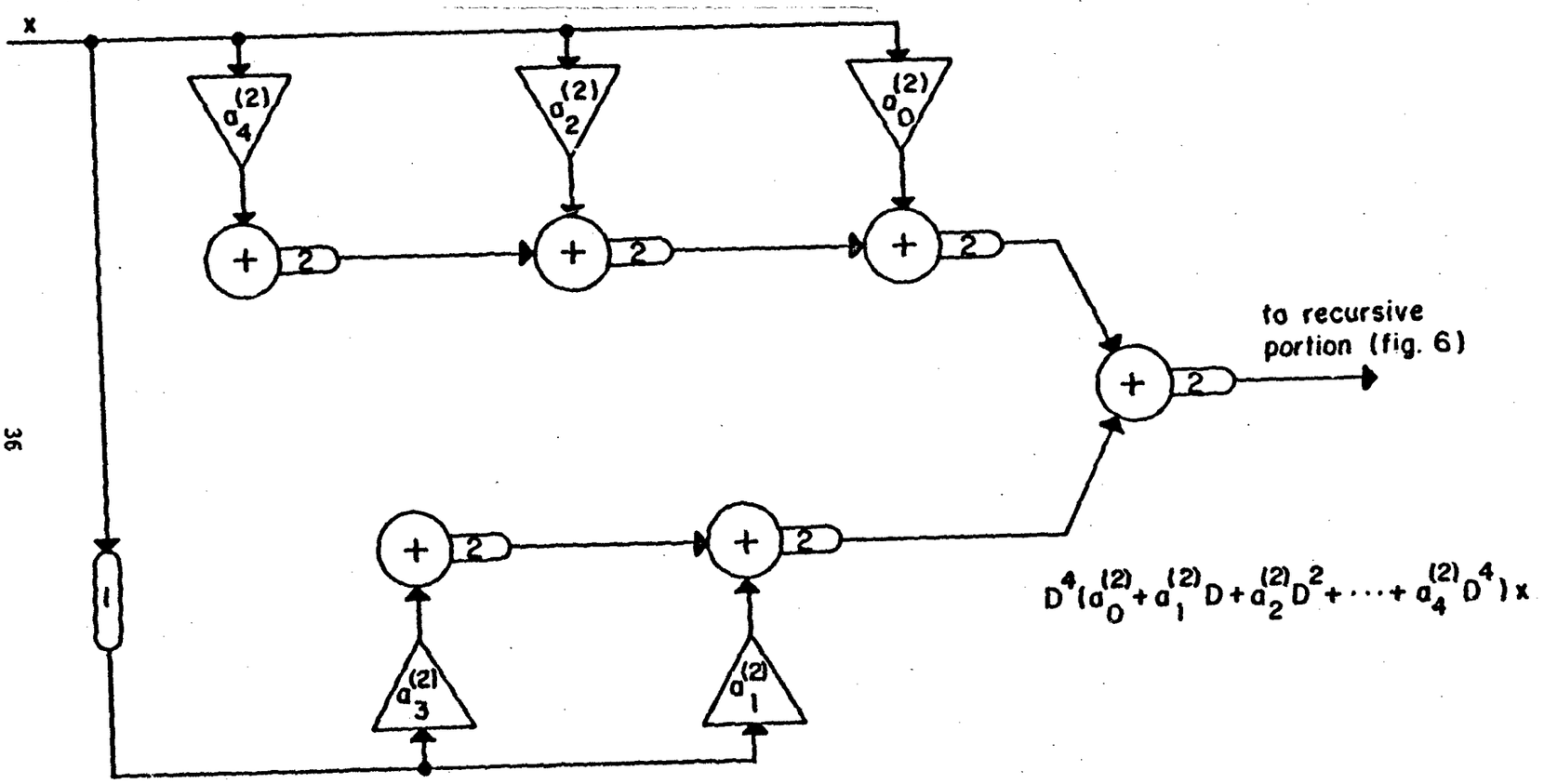


Figure 7. Non-recursive portion of pipeline filter realization

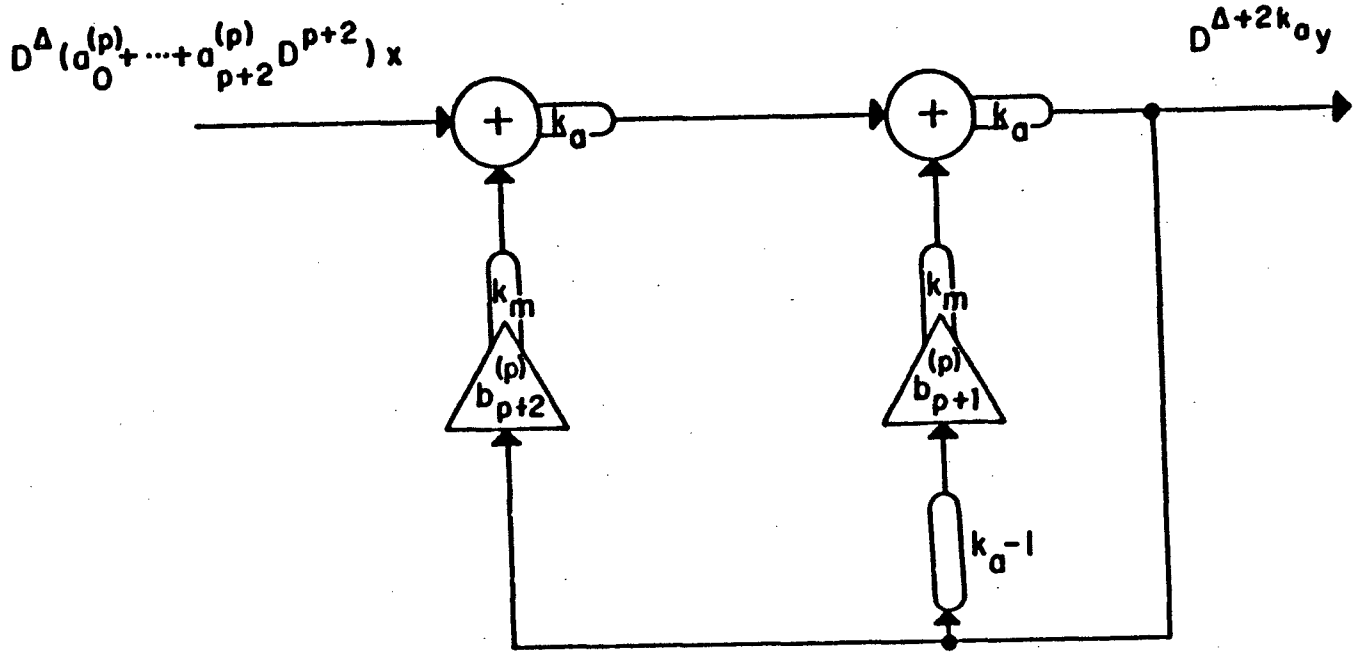
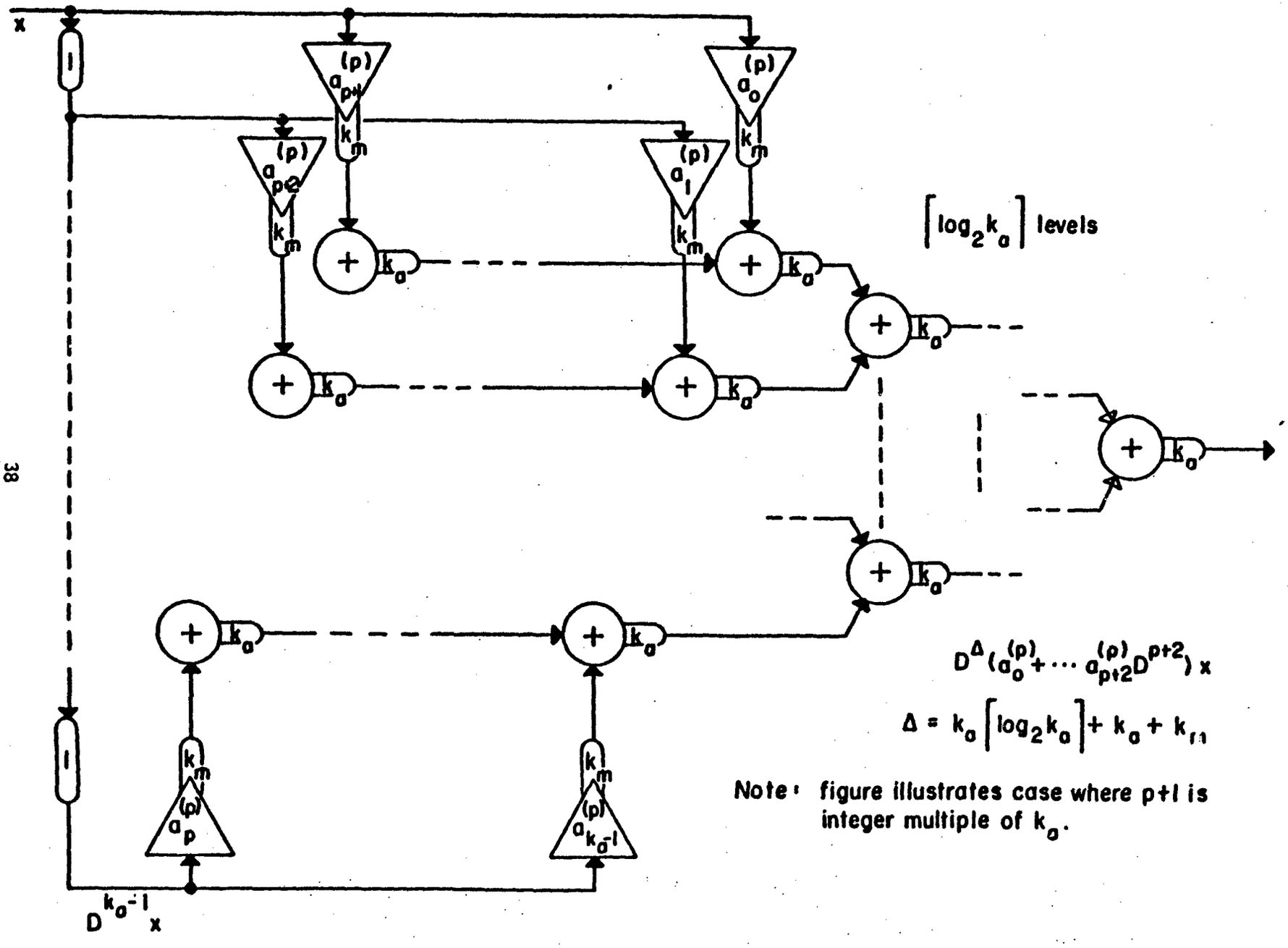


Figure 8. Heart of more general pipeline filter realization



Note: figure illustrates case where p+1 is integer multiple of k_a.

Figure 9. Realization of FIR portion of filter

Polynomial: $F(z) = z^2 - b_1 z - b_2 = 0$

Conditions for Stability:

$1 - b_1 - b_2 > 0$ ..... ①

$1 + b_1 - b_2 > 0$ ..... ②

$|b_2| < 1$ ..... ③

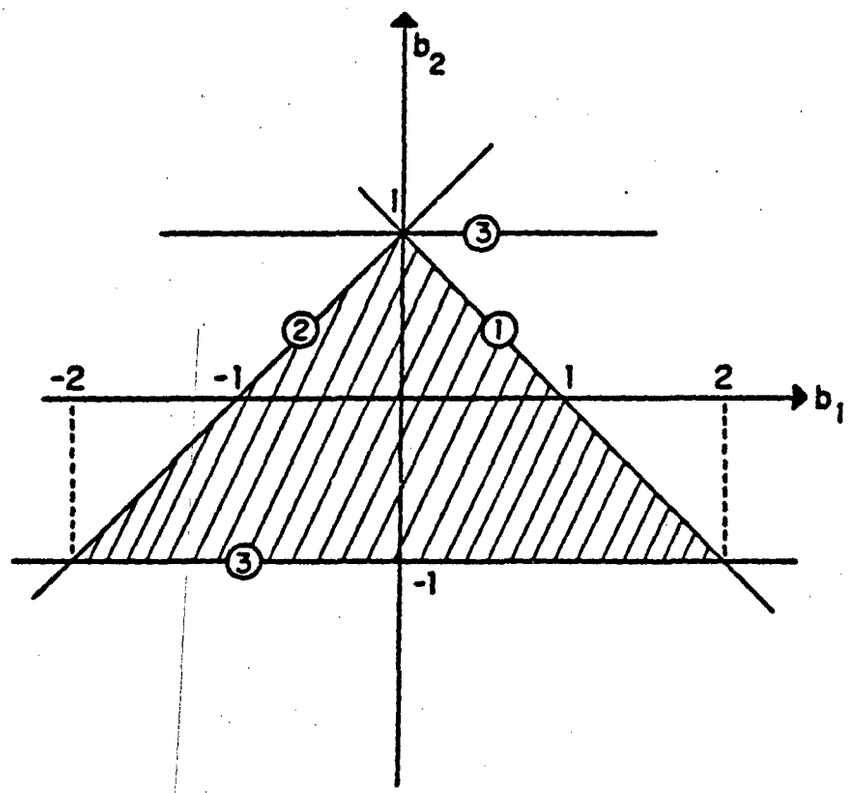


Figure 10. Condition for stability of the original filter

Polynomial: $F(z) = z + b_1 = 0$

Conditions for Stability:

$1 + b_1 > 0$ ..... ①

$1 - b_1 > 0$ ..... ②

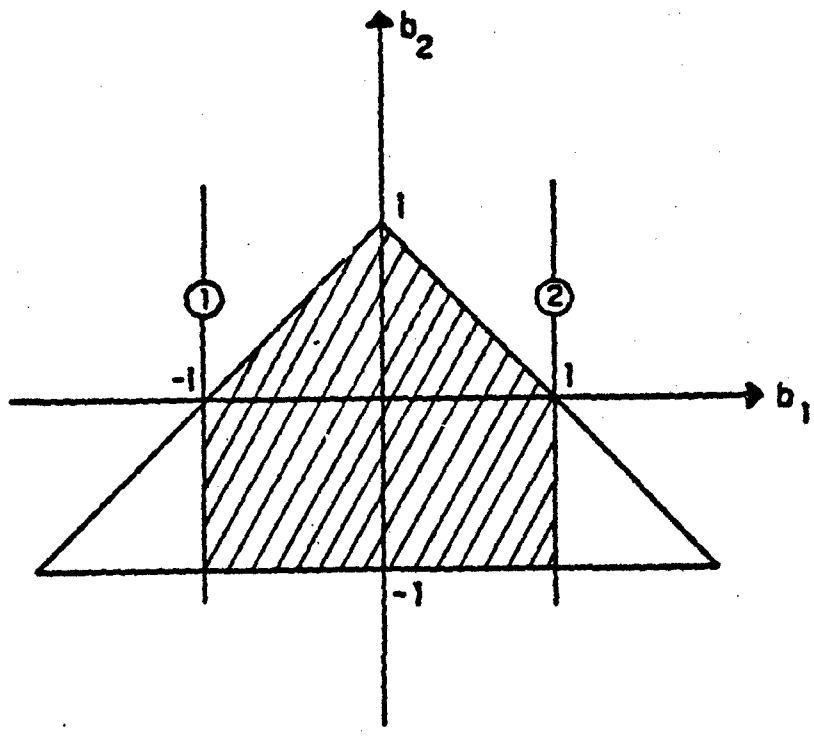


Figure 11. Condition for stability of the augmented filter: p=1

Polynomial: $F(z) = z^2 + b_1 z + (b_1^2 + b_2)$

Conditions for Stability:

$$1 + b_1 + (b_1^2 + b_2) > 0 \quad \cdots\cdots\cdots\cdots \quad ①$$

$$1 + b_1 + (b_1^2 + b_2 > 0 \quad \cdots\cdots\cdots\cdots \quad ②$$

$$\left| b^2 + b_2 \right| < 1 \quad \cdots\cdots\cdots\cdots \quad ③$$

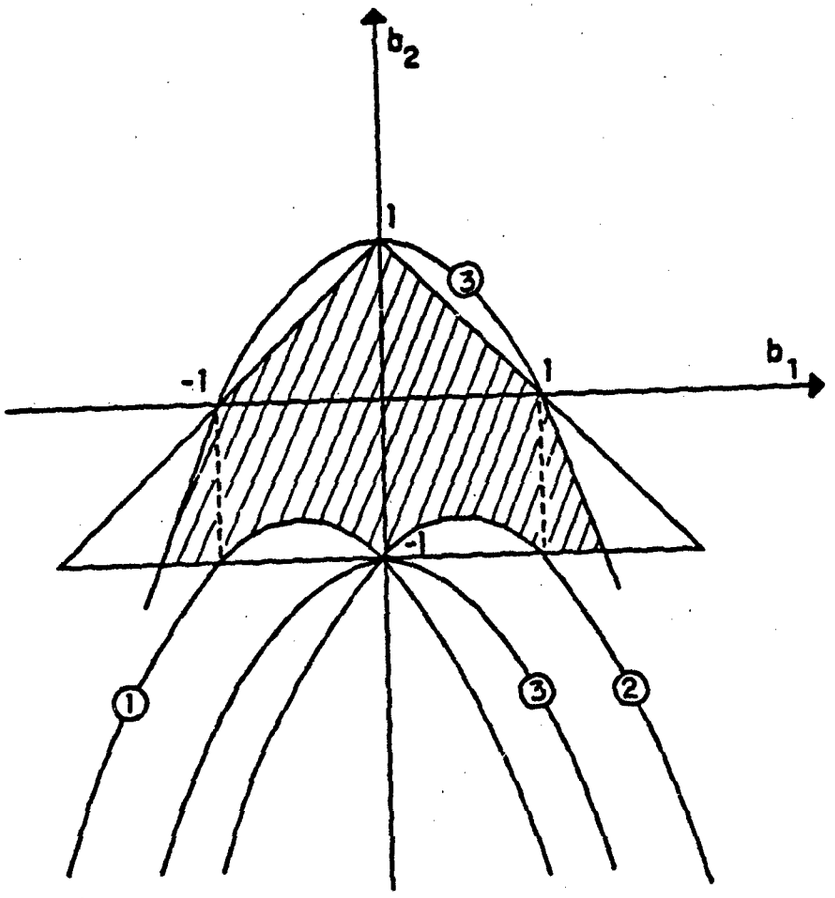


Figure 12. Condition for stability of the augmented filter: p=2

Polynomial: $F(z) = z^3 + b_1 z^2 + (b_1^2 + b_2) z + (b_1^3 + 2b_1 b_2)$

Conditions for Stability:

$$1 + b_1 + (b_1^2 + b_2) + (b_1^3 + 2b_1 b_2) > 0 \quad \ldots\ldots\ldots\ldots\ (1)$$

$$1 - b_1 + (b_1^2 + b_2) - (b_1^3 + 2b_1 b_2) > 0 \quad \ldots\ldots\ldots\ldots\ (2)$$

$$\left| b_1^3 + 2b_1 b_2 \right| < 1 \quad \ldots\ldots\ldots\ldots\ (3)$$

$$\left| (b_1^3 + 2b_1 b_2)^2 - 1 \right| > \left| b_1 (b_1^3 + 2b_1 b_2) - (b_1^2 + b_2) \right| \quad \ldots\ (4)$$

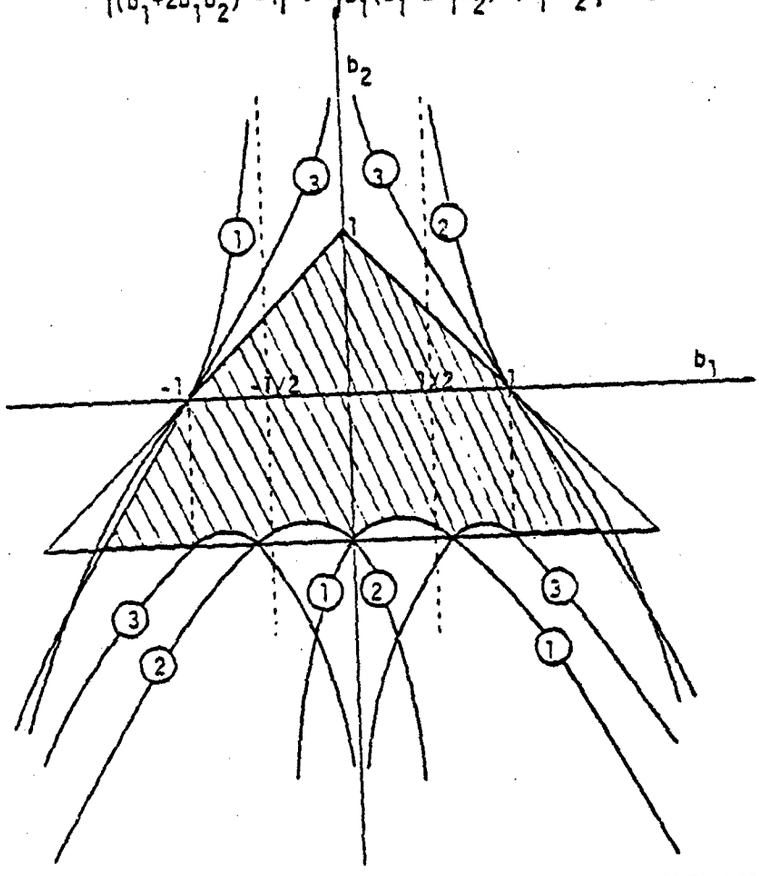
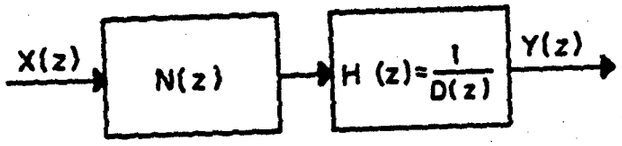
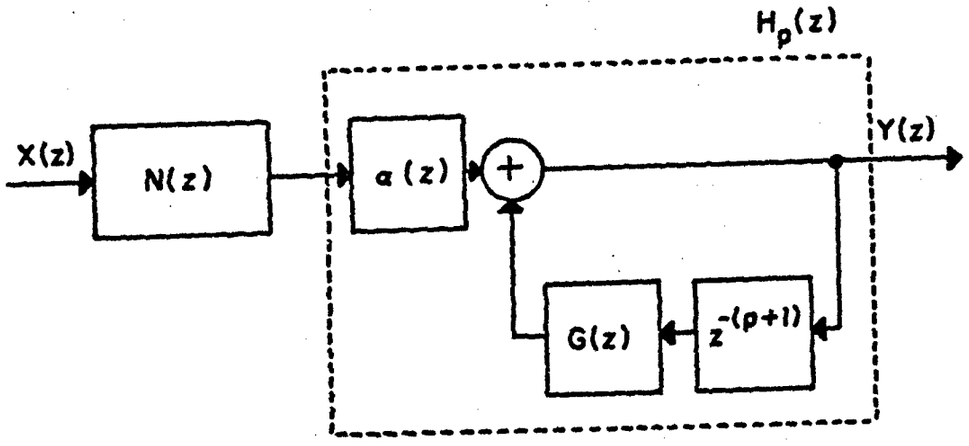


Fig. 13 Stability of augmented filter: p=3



(a)



(b)

Figure 14. (a) Original transfer function (b) Equivalent augmented transfer function

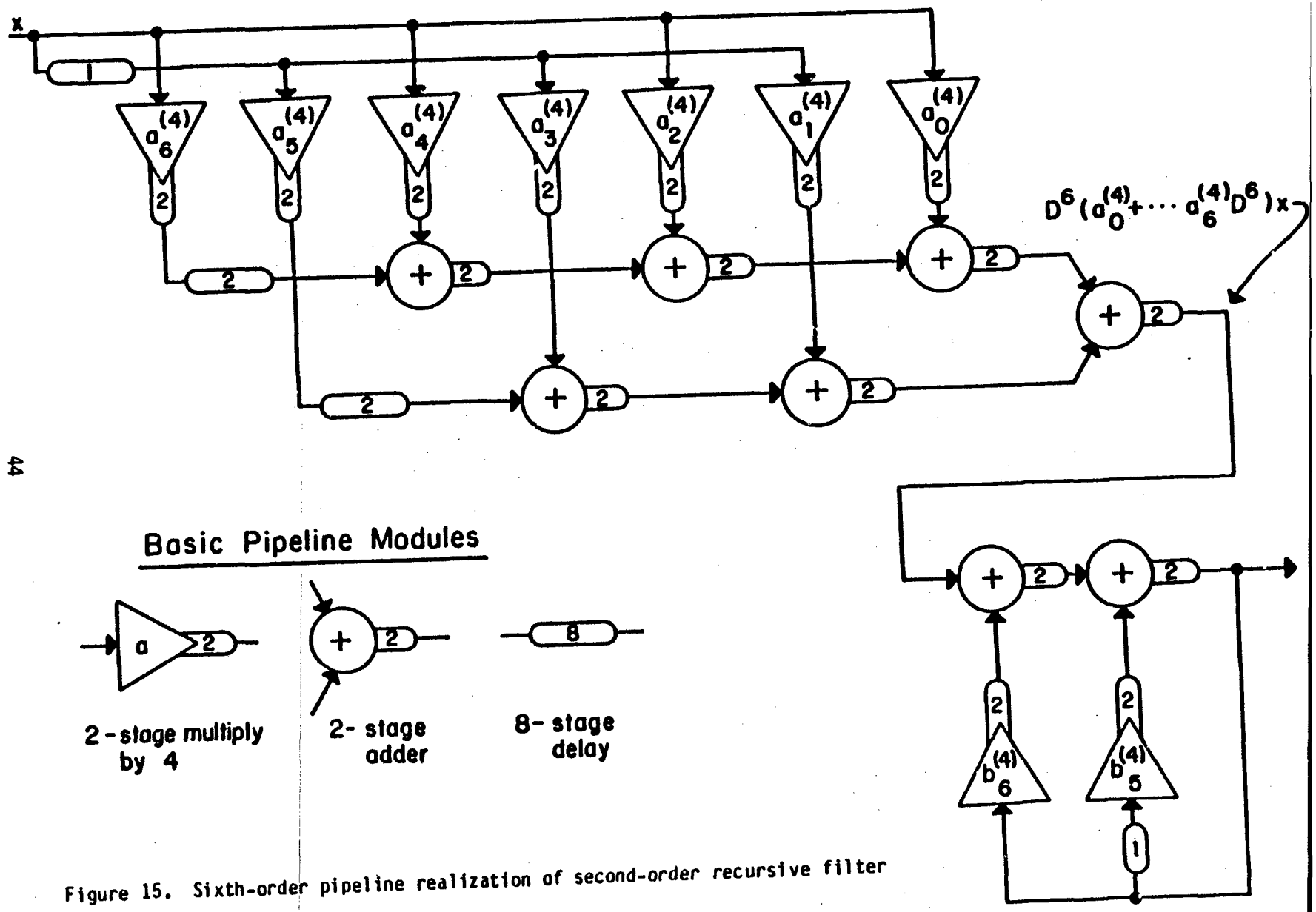


Figure 15. Sixth-order pipeline realization of second-order recursive filter

DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22214	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943	2
3.	Commander Attention: CAPT Glover, Code 61 Naval Electronic Systems Command National Center 1 Washington, D.C. 20360	1
4.	Dr. Sydney Parker, Code 62Px Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943	1
5.	Dr. Donald E. Kirk, Code 62Ki Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943	1
6.	Dr. Richard Twogood Mail Stop L-156 Lawrence Livermore National Laboratory P.O. Box 808 Livermore, California 94550	1
7.	Dr. Herschel H. Loomis, Jr., Code 62Lm Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943	6
8.	Dr. Michael Soderstrand Department of Electrical and Computer Engineering University of California Davis, California 95616	1
9.	Dr. Ralph Algazi Chairman Department of Electrical and Computer Engineering University of California Davis, California 95616	1

	No. Copies
10. Dr. Abraham Sheingold, Code 62 Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, California 93943	1
11. Dr. Bhaskar Sinha IBM Corporation 1000 NW 51 Street Boca Raton, Florida 33432	6
12. Director, National Security Agency Attention: Mr. Jerry Freidman, W3 Fort George G. Meade, Maryland 20755	1
13. Naval Research Laboratory Code 9110, PME 106-5 4555 Overlook Avenue SW Washington, D.C. 20375 Attention: Mr. Jim Morgan	1
14. Naval Research Laboratory Code 9110, PME 106-5 4555 Overlook Avenue SW Washington, D.C. 20375 Attention: CDR Richard Rowe	1
15. Director, National Security Agency Attention: Dr. Jeff Friedhoffer, R631 Fort George G. Meade, Maryland 20755	1