AN IMPLEMENTATION OF A LOW-PASS DIGITAL FILTER

by 632

A. P. VISWANATHAN

B. S., Punjab Engineering College,

Chandigarh, India, 1969

A MASTER'S REPORT

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Electrical Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1970

Approved by:

Dale E. Kaufman

Major Professor

TABLE OF CONTENTS

CHAPTER I

INTRODUCTION

Digital filtering is a term for which there can be no precise definition. Many attempts have been made to give it a definition and broadly it can be defined as follows. A digital filter is a device or computational process which consists of certain arithmetic calculations, like multiplication and addition, delays and storage elements to transform one sequence of numbers into another sequence. In general, it can be said that a digital filter does not depend on such elements as inductors, capacitors or resistors. Instead, it does the job with adders, multipliers, delay, and storage elements. The coefficients associated with the transfer function of a digital filter are numerical inputs to the multipliers and are thus not sensitive to component variations. In the case of an analog filter these coefficients are realized in terms of elements like inductors, capacitors, and resistors which are sensitive to physical conditions.

The digital filter offers performance in frequency selection and rejection which may be quite difficult to attain with analog filters composed of RLC elements. Other features of a digital filter are high degree of attenuation and better Q. Moreover, the performance characteristics and the frequencies of interest can be changed without changing any component.

Since a digital filter contains no reactive components it offers a very accurate drift-free operation. At lower frequencies, size and weight of a digital filter are considerably smaller than an analog filter.

Recent integrated circuit accomplishments, such as the Large Scale Integration (LSI), have made it possible to perform the digital filter operations in a small number of integrated circuit chips. The developments in LSI promise to include a second-order digital filter in two or three chips.

Digital Filter Operation

A basic digital filter operation can be represented by the block diagram in Fig. 1.1. As shown in the diagram, the

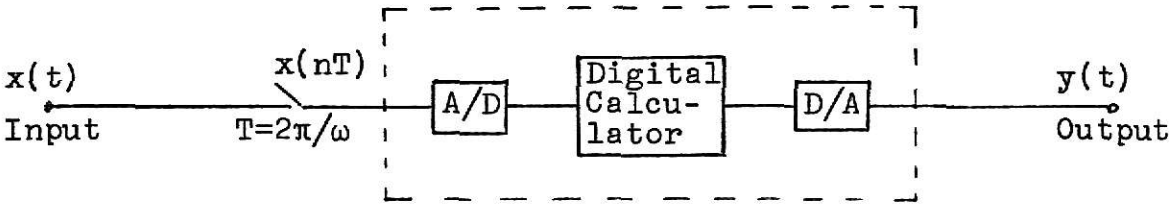


Fig. 1.1. Block diagram of a digital filter.

digital filter is comprised of three units: (1) An analog-to-digital (A/D) converter, (2) a digital calculator or the arithmetic unit, and (3) a digital-to-analog (D/A) converter. The analog signal at the input is sampled every T seconds so that at the time t = nT, the continuous input signal is momentarily sampled, and the pulse x(nT) appears at the input to the A/D converter. In the A/D converter, this pulse amplitude is converted to a digital word. This digital word is a set of binary

digits (bits) which is coded to represent the amplitude of x(nT). The accuracy of the word depends on the number of bits used to represent it. Mathematical operations such as multiplication, addition, and delays are performed numerically with this coded input and other coefficients in the digital calculator. The output word from the calculator is fed to a D/A converter which converts the digital coded word into an analog amplitude. If the input and/or the output to the digital filter is in binary bits, then the sampler, A/D converter and D/A converter are not needed and only the arithmetic unit is needed to perform the calculations.

The operation of a digital filter is defined by a difference equation. This equation defines the output pulse amplitude y(nT) as a function of the input pulse x(nT), and the past output pulses and input pulses. The past input and output pulses are stored in shift registers. A general formula for the difference equation is

$$y(nT) = \sum_{i=0}^{N} a_i x\left[(n - i)T\right] + \sum_{i=1}^{M} b_i y\left[(n - i)T\right] \qquad (1.1)$$

If y(nT) is a function of only the present and past input pulses, the filter is termed nonrecursive. If the past output pulses are also included, then it is a recursive type of filter.

Remark. In the case of analog filters differential equations are used to define the operation and Laplace transforms are used to study the frequency characteristics and related topics. In digital filters, difference equations are used to

define the operation, as stated earlier, and z-transforms are used for analysis.

A simple case of equation (1.1) is taken to demonstrate the realization of a digital filter. As an example, let a first-order linear difference equation be taken.

$$y(nT) = Ky(nT - T) + x(nT) \tag{1.2}$$

In order to perform the operations required by equation (1.2), three storage registers are needed, one for x(nT), one for y(nT), and one for K. The product Ky(nT - T) is obtained, the contents of the register containing x(nT) are added to this product, and the result is stored in the register containing y(nT).

A simple block diagram representative of equation (1.2) would be as shown in Fig. 1.2.

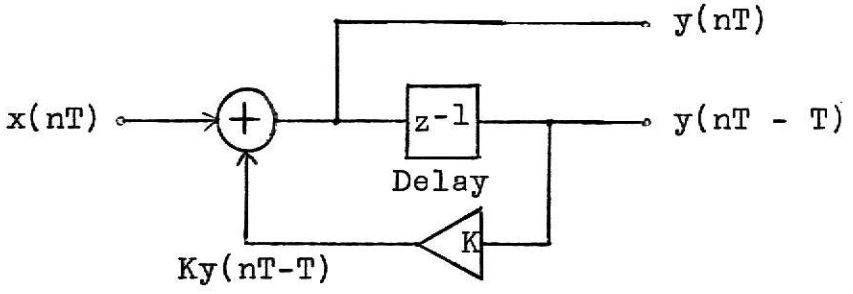


Fig. 1.2. Simple first-order digital filter.

The various notations used in the above figure are described as follows. The rectangular figure with z^{-1} inside denotes a unit delay of duration equal to the sampling interval T. The circle with + denotes addition of all signals with arrows pointing to the circle and the triangular figure with K

inside denotes the multiplication of the signal with a fixed coefficient K.

The next chapter describes the different forms of realization of a digital filter, their advantages and disadvantages. In later chapters, a low-pass second-order Butterworth filter is designed using z-transforms and the implementation of the transfer function obtained is also shown; also, the principle of A/D and D/A converters is described.

CHAPTER II

FORMS OF REALIZATION OF A DIGITAL FILTER

As stated in the first chapter, the operation of a digital filter is defined by the difference equation. The analysis of a digital filter and study of its frequency characteristics are done with z-transforms which permit algebraic manipulation of difference equations much as Laplace transforms are manipulated for differential equations. It is thus apparent that a transfer function must be realized which will be described in z-transforms, just as in the continuous case where the realization is obtained when the transfer function is described in Laplace transforms.

To illustrate the different forms of realization let a system be considered which is described by the following difference equation:

$$y(nT) = K_1 y(nT-T) + K_2 y(nT-2T) + x(nT) - Lx(nT-T) \qquad (2.1)$$

Taking the z-transform of equation (2.1) and assuming zero initial conditions, i.e., $y(-T) = y(-2T) = x(-T) = 0$, the following is obtained.

$$Y(z) = K_1 Y(z) z^{-1} + K_2 Y(z) z^{-2} + X(z) - LX(z) z^{-1}$$

or $$Y(z)\left[1 - K_1 z^{-1} - K_2 z^{-2}\right] = X(z)\left[1 - Lz^{-1}\right]$$

or $$\frac{Y(z)}{X(z)} = \frac{1 - Lz^{-1}}{1 - K_1 z^{-1} - K_2 z^{-2}} \quad .$$

Thus

$$H(z) = \frac{1 - Lz^{-1}}{1 - K_1 z^{-1} - K_2 z^{-2}} \quad . \tag{2.2}$$

A direct form of realization of equation (2.1) is shown in Fig. 2.1.

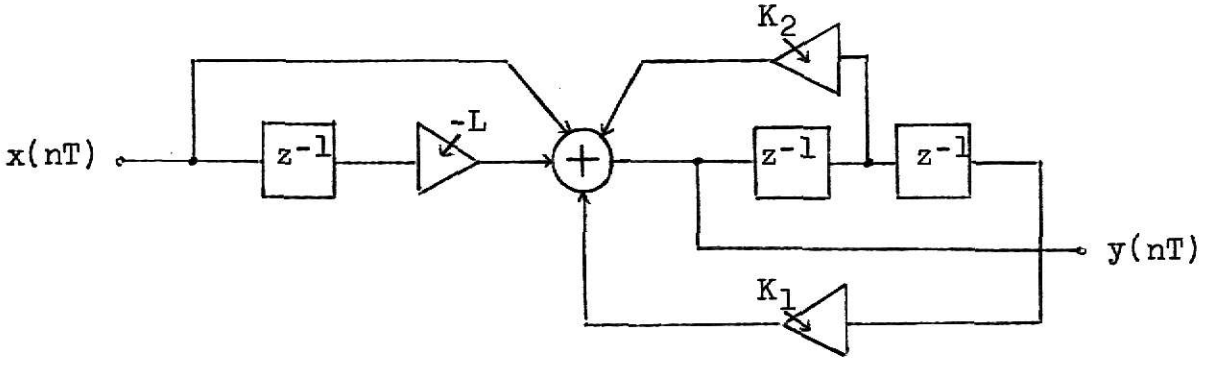


Fig. 2.1. Direct form of realization of equation (2.1).

The network describing the transfer function for H(z) given by equation (2.2) has a zero and two poles. Figure 2.1 shows that the zero is created by the delay of the input whereas poles are created by the delays of the feedback output. An alternative form of obtaining (realizing) H(z) is by reversing the order of feed forward and feedback delays as shown in Fig. 2.2.

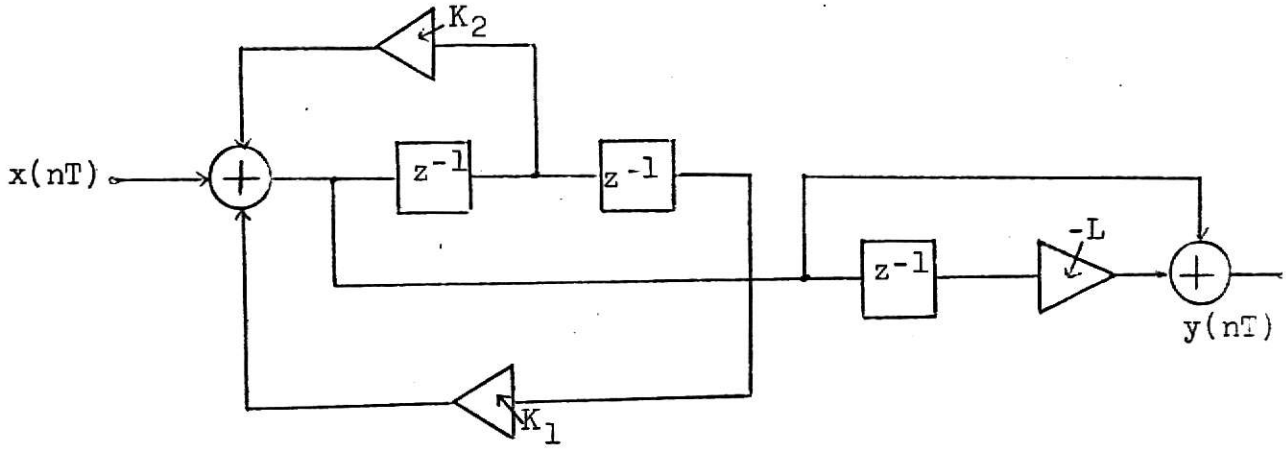


Fig. 2.2. Alternative form of realization of equation (2.1).

The above two forms of realization are called the direct forms of realization since the number of delay elements are equal to the number of zeros and poles in the transfer function irrespective of the order of the transfer function. The following realization employs as many delay elements as the order of the difference equation, and hence it is called a canonic form of realization. It is obtained as follows:

$$Y(z) = \frac{X(z)\left[1 - Lz^{-1}\right]}{1 - K_1 z^{-1} - K_2 z^{-2}} \tag{2.3}$$

Let

$$W(z) = \frac{X(z)}{1 - K_1 z^{-1} - K_2 z^{-2}} \tag{2.4}$$

Then

$$Y(z) = W(z)\left[1 - Lz^{-1}\right] . \tag{2.5}$$

From equation (2.4), taking the inverse z-transform, we obtain

$$w(nT) = x(nT) + K_1 w(nT - T) + K_2 w(nT - 2T) \ . \qquad (2.6)$$

Equation (2.5) yields

$$y(nT) = w(nT) - Lw(nT - T) \ . \qquad (2.7)$$

Equations (2.6) and (2.7) are two simultaneous difference equations which lead to the canonic structure shown in Fig. 2.3.

Figure 2.3 employs only two delay elements and hence it is a canonic form. But the above method is not the best method of realization of a digital filter. It has been shown by Kaiser[1] that in using the direct form, the accuracy requirements on the coefficients L_i and K_i are severe. These restrictions arise

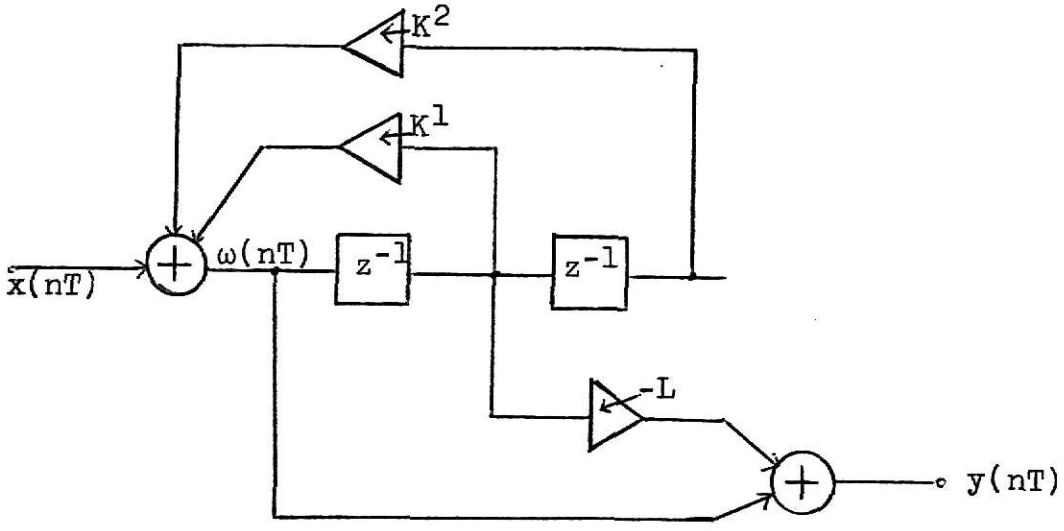


Fig. 2.3. Canonic form of realization of equation (2.1).

---

[1]J. F. Kaiser, "Some practical considerations in realization of linear digital filters," 1965 Proc. 3rd Allerton Conference on Circuit and System Theory.

from the noise created by quantization and multiplication round off. In order to limit the noise to a minimum the coefficients have to be very accurate. But in the case of serial or cascade form of realization, which will be described below, the accuracy requirements are not so severe. In order to demonstrate the widely used cascade form the following transfer function is considered.

$$H(z) = \frac{1 + L_1 z^{-1} + L_2 z^{-2}}{1 - (K_1 z^{-1} + K_2 z^{-2} + K_3 z^{-3})} \tag{2.8}$$

H(z) is written as a product of quadratic polynomials as follows:

$$H(z) = H_1(z)H_2(z) = \left[\frac{(1+L_{11}z^{-1}+L_{12}z^{-2})}{(1+K_{11}z^{-1}+K_{12}z^{-2})}\right]\left[\frac{(1+L_{21}z^{-1}+L_{22}z^{-2})}{(1+K_{21}z^{-1}+K_{22}z^{-2})}\right] \tag{2.9}$$

where the coefficients L_{ij} and K_{ij} are real. Equation (2.9) is obtained by the factorization of numerator and denominator of equation (2.8) to a quadratic form. In the above case if $L_{11} = L_1$ and $L_{12} = L_2$, then $L_{21} = L_{22} = 0$. Again, either K_{12} or K_{22} must equal zero. If $K_{22} = 0$, then K_{11}, K_{12}, and K_{21} must be solved for by the process of elimination. Thus H(z) is realized as shown in Fig. 2.4.

Since L_{21}, L_{22}, and K_{22} are zero in the example considered, their multiplication is shown dotted in Fig. 2.4. But in general they may not be zero.

It is noted that the above realization has four delay elements although equation (2.8) represents a third-order difference equation. But each quadratic $H_i(z)$ is realized in a

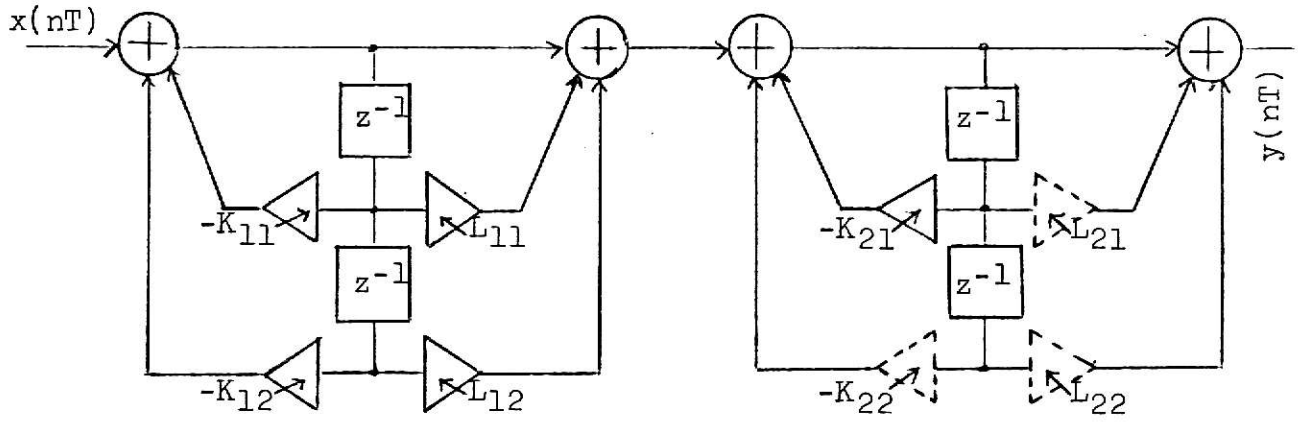


Fig. 2.4. Alternative canonic form of realization.

canonic form. This realization also has the advantage that it is amenable to multiplexing.

The canonic form shown in Fig. 2.3 can be used only for first- or second-order transfer functions. For higher order transfer functions noise plays an important role and the filter characteristics are degraded. The next chapter describes the implementation of a second-order Butterworth filter using the canonic form of Fig. 2.3.

CHAPTER III

LOW-PASS BUTTERWORTH FILTER

The second-order Butterworth transfer function in the s-plane is given by

$$H(s) = \frac{1}{s^2 + \sqrt{2}s + 1} \quad . \tag{3.1}$$

The above expression is the normalized transfer function. If the cutoff frequency is ω_c, then the transfer function becomes

$$H(s) = \frac{1}{(s/\omega_c)^2 + \sqrt{2}\,(s/\omega_c) + 1} \tag{3.2}$$

or

$$H(s) = \frac{\omega_c^2}{s^2 + \sqrt{2}\,\omega_c s + \omega_c^2} \tag{3.3}$$

The impulse invariance approach will be used to find H(z). In this approach the impulse response h(t) will be found and from the corresponding h(nT), the sampled signal of h(t), the z-transform will be obtained.

Now, h(t) is given by the inverse Laplace transform of H(s), equation (3.3), i.e.,

$$h(t) = \sqrt{2}\, e^{-\omega_c t/\sqrt{2}} \sin(\omega_c t/\sqrt{2}) \quad . \tag{3.4}$$

The corresponding z-transform is given by

$$H(z) = \sqrt{2}\ \omega_c \left[\frac{z\ e^{-\omega_c T/\sqrt{2}}\ \sin\ (\omega_c T/\sqrt{2})}{z^2 - 2z\ e^{-\omega_c T/\sqrt{2}}\ \cos\ (\omega_c T/\sqrt{2}) + e^{-\sqrt{2}\ \omega_c T}} \right]$$

or

$$H(z) = \sqrt{2}\ \omega_c \left[\frac{z^{-1}\ e^{-\omega_c T/\sqrt{2}}\ \sin\ (\omega_c T/\sqrt{2})}{1 - 2z^{-1}\ e^{-\omega_c T/\sqrt{2}}\ \cos\ (\omega_c T/\sqrt{2}) + z^2\ e^{-\sqrt{2}\ \omega_c T}} \right] \tag{3.5}$$

Let

$$\begin{aligned} \alpha_1 &= \sqrt{2}\ e^{-\omega_c T/\sqrt{2}}\ \sin\ (\omega_c T/\sqrt{2}) \\ \beta_1 &= 2\ e^{-\omega_c T/\sqrt{2}}\ \cos\ (\omega_c T/\sqrt{2}) \\ \beta_2 &= -e^{-\sqrt{2}\ \omega_c T}\ . \end{aligned} \tag{3.6}$$

It is noted that these coefficients are dependent on the cutoff frequency ω_c as well as the sampling time T. They can be kept within a small range for a wide range of cutoff frequencies if the sampling time T is also reduced correspondingly.

Substituting equation (3.6) in equation (3.5) gives

$$H(z) = \frac{\omega_c\ \alpha_1\ z^{-1}}{1 - \beta_1 z^{-1} - \beta_2 z^{-2}}$$

Now ω_c can be combined with α_1 to form one coefficient for small values of ω_c but for larger values the coefficient becomes very large and cannot be handled digitally; hence ω_c is scaled before the signal enters the arithmetic unit.

$$\frac{Y(z)}{X(z)} = \frac{\omega_c \alpha_1 z^{-1}}{1 - \beta_1 z^{-1} - \beta_2 z^{-2}} \quad . \tag{3.7}$$

Let

$$W(z) = \frac{\omega_c X(z)}{1 - \beta_1 z^{-1} - \beta_2 z^{-2}} \tag{3.8}$$

so that

$$w(nT) = \omega_c x(nT) + \beta_1 w(nT - T) + \beta_2 w(nT - 2T) \quad .$$

Substituting equation (3.8) in equation (3.7) gives

$$Y(z) = W(z)\, \alpha_1 z^{-1} \tag{3.9}$$

or

$$y(nT) = \alpha_1\, w(nT - T) \tag{3.10}$$

Equation (3.10) in the canonic form is realized as in Fig. 3.1.

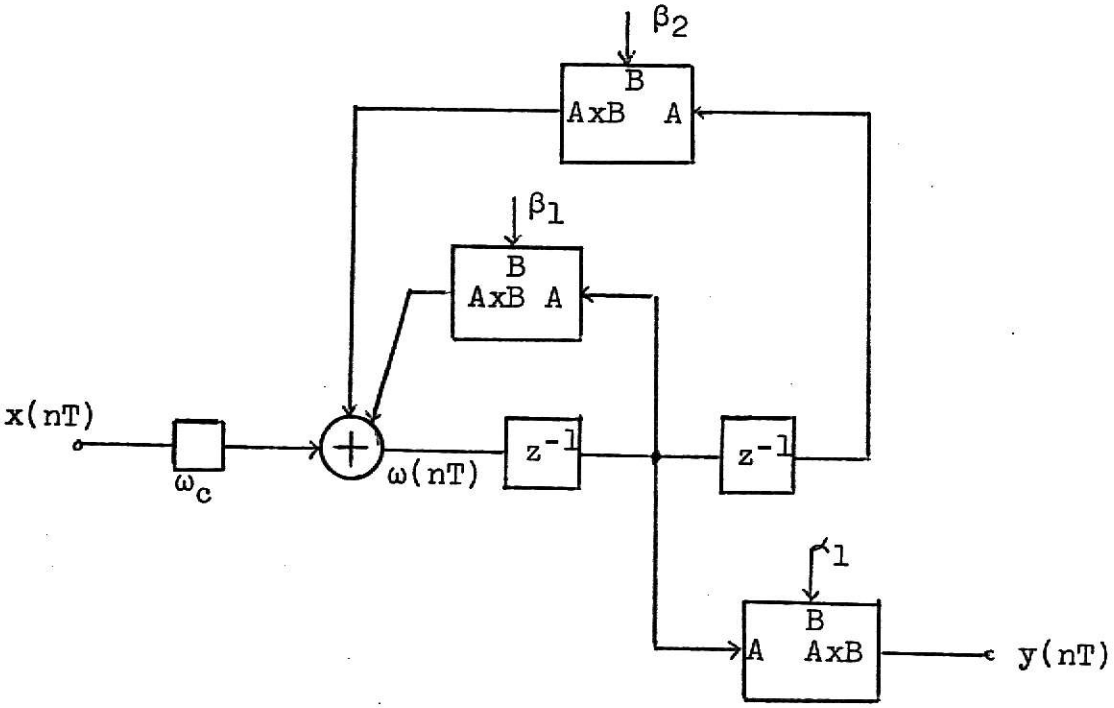


Fig. 3.1. Realization of a second-order Butterworth filter.

For the value of cutoff frequency, f_c, selected the sampling frequency, f, should be such that $f \geq 2 f_c$ (Shannon's Sampling Theorem).

For the sake of simplicity, let $\omega_c = 1$ rad/sec, or $f_c = 1/2\pi$ Hz. This means that f should be such that

$$f \geq 2/2\pi \text{ Hz}$$

$$\geq 1/\pi \text{ Hz} .$$

Let $f = 1$ Hz .

Then $T = 1$ sec.

The coefficients α_1, β_1, and β_2 are then calculated as follows.

$$\alpha_1 = \sqrt{2}\, e^{-1/\sqrt{2}} \sin (1/\sqrt{2})$$

$$= \sqrt{2} \times 0.494 \times 0.64685$$

$$= 0.4515 \qquad (3.11)$$

$$\beta_1 = 2\, e^{-1/\sqrt{2}} \cos (1/\sqrt{2})$$

$$= 2 \times 0.494 \times 0.76261$$

$$= 0.968 \times 0.76261$$

$$= 0.7375.$$

$$\beta_2 = -e^{-\sqrt{2}} = -0.244.$$

Arithmetic

For the implementation of the filter the arithmetic unit is the most important part and so it will be described in detail. If the incoming signal is in digital form $[x(nT)]$, then A/D conversion is not necessary and the signal will be the input to the arithmetic unit. On the other hand, if the signal is a continuous one, $[x(t)]$, then it has to be converted into digital form and coded in terms of bits after sampling. It is assumed that each value of x(nT) is described by an 8-bit word.

The arithmetic unit consists of the following: (1) An adder, (2) delay elements, and (3) multipliers.

In the description of the arithmetic unit, the 2's complement form of notation and parallel arithmetic are used. One of the main advantages of parallel arithmetic is the fast computation time which is best suited for time multiplexing.

2's Complement Notation

The 2's complement representation of binary numbers is most appropriate for digital filter implementation because additions may be performed without any previous knowledge of the signs or magnitude and with no corrections necessary later as in 1's complement notation. It will be assumed that the signal has a magnitude less than 1, or

$$-1 \leqslant x \leq 1.$$

If in binary 2's complement notation, the signal is represented by an eight-bit word, $x_0\ x_1\ .\ .\ .\ x_7$, then the digitized

value of x, $\hat{x}$, is given by

$$\hat{x} = x_0 + \sum_{i=1}^{7} x_i 2^{-1} ,$$

where each x_i is either a 0 or a 1. The sign of $\hat{x}$ is given by x_0. If x_0 is 1, $\hat{x}$ is a negative number, and if x_0 is 0, $\hat{x}$ is a positive number.

A very useful property of 2's complement representation is that in the addition of more than two numbers, if the final sum is less than 1 in magnitude so that it can be represented by the 8 bits, then the correct sum will be obtained regardless of the order of addition and even if overflow occurs in some partial sums. The following example illustrates the point.

Suppose the following numbers are to be added: 0.8, 0.6, and -0.7.

In the above example the correct total sum is 0.7 but it is noticed that an overflow occurs when 0.8 and 0.6 are added as the sum equals 1.4 which cannot be represented by the 8 bits. If the 2's complement notation is used and the addition performed, then the following result will be obtained.

Decimal Notation	2's Complement Notation
0.8	0.1100110
0.6	0.1001101
-0.7	1.0100110

Rounding off in the 2's complement notation is done in the following manner. A 1 is added to the ninth bit and then the resulting eight bits are taken for arithmetic.

Addition:

0.8	⟹	0.1100110
+0.6		0.1001101
1.4	⟹	1.0110011
+(-0.7)		1.0100110
0.7		(1)0.1011001

↑ Disregard

Another example is taken to illustrate the case of negative numbers. Let it be the addition of -0.8, -0.6, and +0.7. Addition of -0.8 and -0.6 gives -1.4 which cannot be represented by the 8 bits and will cause an overflow. But addition of -1.4 and +0.7 gives -0.7 which can be represented by the 8 bits and thus the correct result is obtained by performing the addition in 2's complement notation.

Decimal Notation	2's Complement Notation
-0.8	1.0011010
-0.6	1.0110011
+0.7	0.1011010

Rounding off is done in the same manner as was previously done.

Addition:

-0.8	⟹	1.0011010
+(-0.6)		1.0110011
-1.4		(1)0.1001101

↑ Disregard

	⟹	
+(+0.7)		0.1011010
-0.7		1.0100111

Delay Elements

The delay z^{-1} of the signal is obtained by using a shift register. Two cascaded chains of JK flipflops are employed for this purpose, the output of the first producing a delay of one word and the output of the second producing a delay of two words. For the 8-bit word, eight flipflops are connected in parallel and operated by the same clock.

Operation

Supposing a 1 is placed on one clocked input and a 0 is placed on the other of a JK flipflop, as shown in Fig. 3.2.

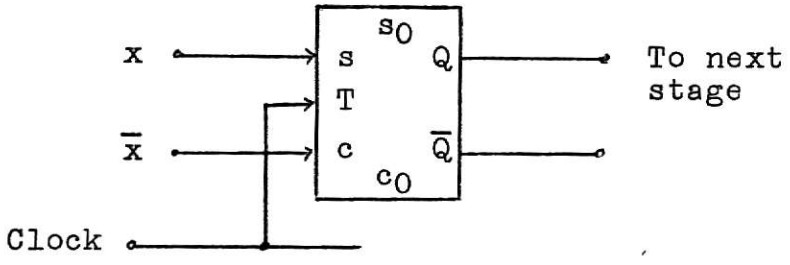


Fig. 3.2. Basic flipflop connection.

If the flipflop is clocked with a negative or a positive going transition pulse, depending upon the logic used, the same 1 and 0 now appear at the output, both signals apparently "passing through" the flipflop from input to output. The truth table of a JK flipflop is shown in Fig. 3.3.

A schematic of the shift register used for delay is shown in Fig. 3.4. The clock pulse is obtained from the 2^9 counter of the A/D converter at $(2^8 + 1)T_c$ of every sample which is

Set s	Clear c	Q	$\overline{Q}$
0	0	Q_n	$\overline{Q}_n$
1	0	1	0
0	1	0	1
1	1	$\overline{Q}_n$	Q_n

Fig. 3.3. Truth table of a JK flipflop.

immediately after the A/D conversion is complete. More details on the clock are presented in the section on A/D conversion.

Parallel Adder

As stated before, the parallel adder is more compatible in the filter than the serial adder. It has no control signals to control the carry input and no feedback of the carry output. Moreover, there is no synchronization needed between the input bits as all the bits enter the adder at the same time.

A parallel binary adder is shown in Fig. 3.5. It shows the way of adding two 8-bit binary numbers. The binary numbers are the input signal and the output of one of the multipliers.

The rightmost adder in the circuit in Fig. 3.5 is a half adder since it handles only the least significant bits (LSB) and requires no carry input. The carry output of the LSB is connected to the input of the next significant bit, and so on. The output is taken out in parallel as $s_7\ s_6\ \ldots\ s_1\ s_0$. The same configuration can be used to subtract if the negative number is represented in 2's complement notation. In that case

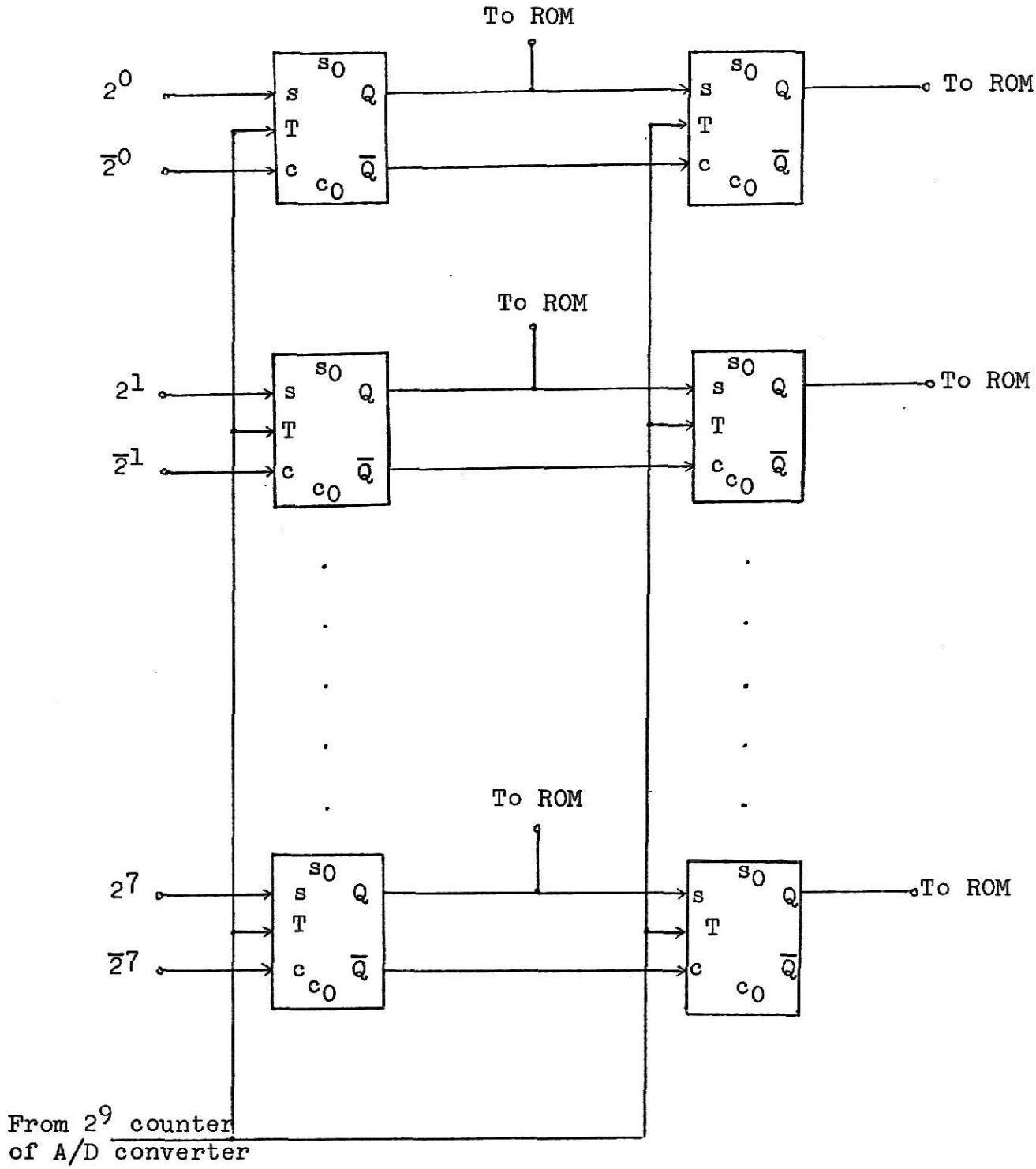


Fig. 3.4. Shift register.

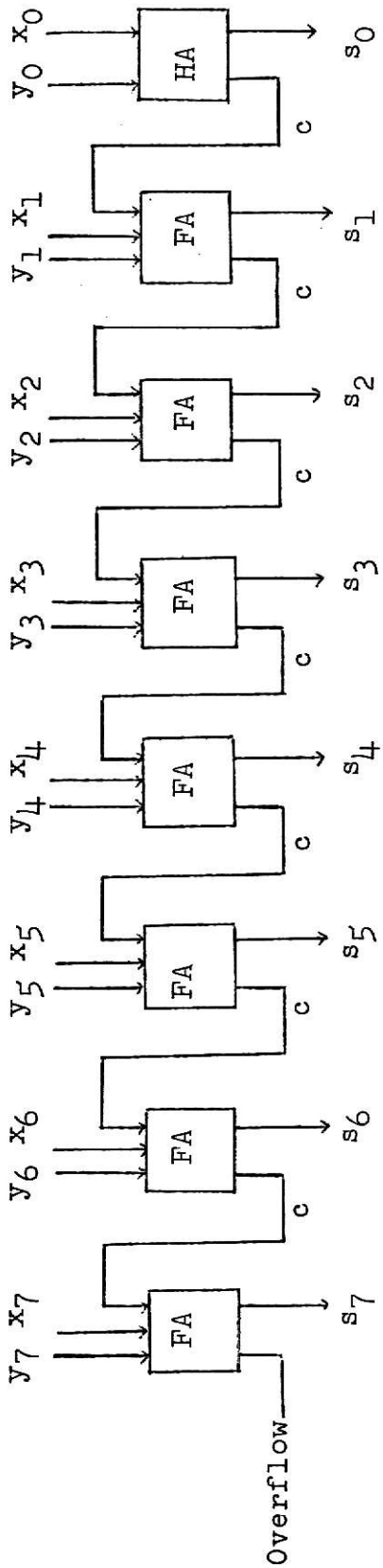
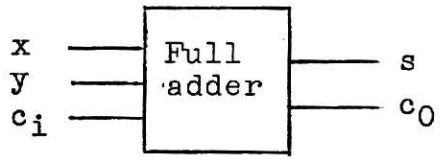


Fig. 3.5. Parallel adder.

the output will also be a number in 2's complement notation.

Many ways are used to implement a full adder and one of the ways is to use two half adders, as is shown in Fig. 3.6.



Block diagram

$$s = \bar{x}\,\bar{y}\,c_i + \bar{x}\,y\,\bar{c}_i + x\,\bar{y}\,\bar{c}_i + x\,y\,c_i$$

$$c_o = \bar{x}\,y\,c_i + x\,\bar{y}\,c_i + x\,y\,\bar{c}_i + x\,y\,c_i$$

or

$$c_o = x\,c_i + x\,y + y\,c_i$$

Boolean expressions.

Input			Output	
x	y	c_i	s	c_o
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Truth table

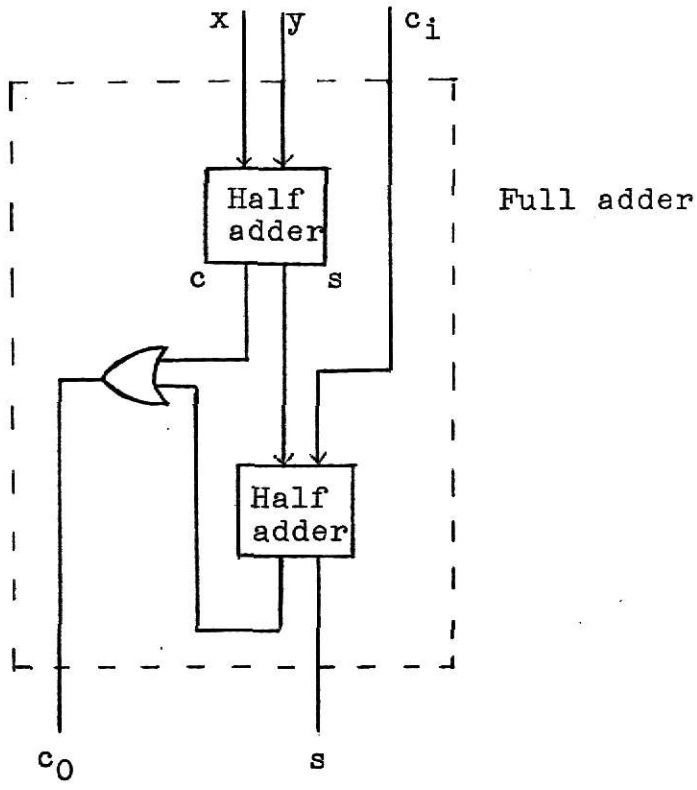


Fig. 3.6. Implementation of a full adder using two half adders.

A normal half adder has about four gates so that in a full adder there are approximately 9 gates. Hence for the implementation of an 8-bit parallel adder, the total number of gates required will be 9 x 7 + 4 = 67 gates. These gates can be implanted on a single chip, thanks to the developments being made in Large Scale Integration. Thus economically such an adder would not be very costly and would occupy an area less than a square inch.

As shown in Fig. 3.1, the adder has three inputs but the implementation shown in Fig. 3.5 has only two inputs. Thus in order to add the third signal, another adder must be connected. This has inputs, one of which is the output of the first adder and the other is the output of the second multiplier. The connection between them is shown in Fig. 3.7.

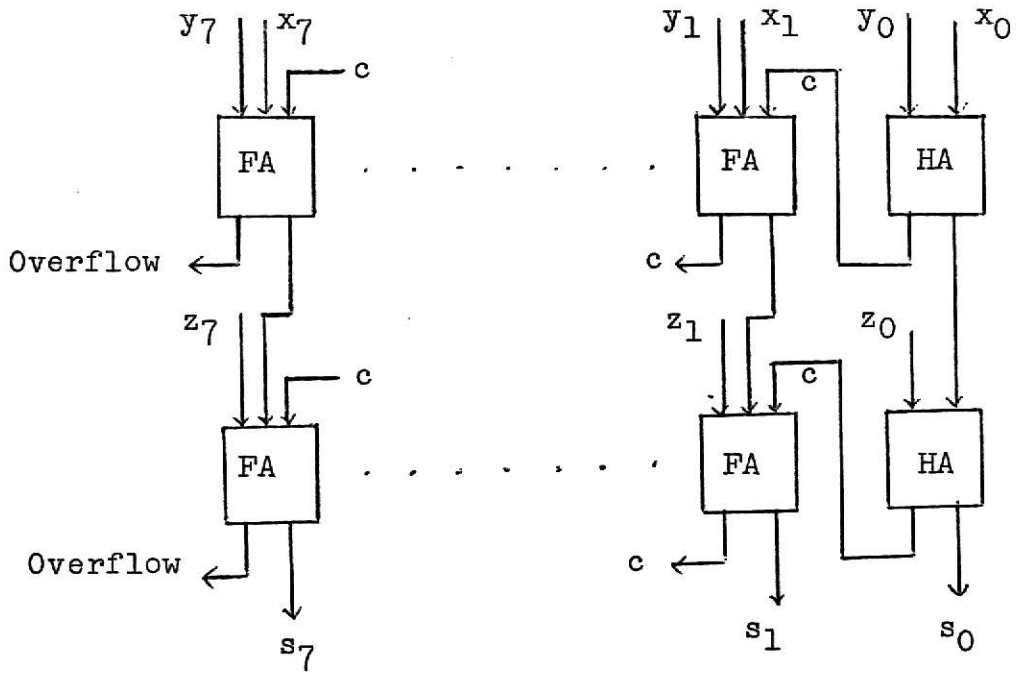


Fig. 3.7. Cascaded adders.

Should a larger number of inputs (e.g., 7 to 10) enter the adder, an alternative arrangement must be made. In that case two input signals are added first and their output stored in an accumulator. Then the third signal is added to the contents of the accumulator and the sum is again stored in the accumulator. The procedure is continued until all the input signals are added. This procedure is time consuming and requires many control (timing) signals. However, for higher order filters the cascade form of realization is often used which has at the most three signal inputs to the adder and thus the circuit of Fig. 3.7 may be used.

Multipliers

The multiplier section is used for the multiplication of the coefficients with the signal. A multiplier can be realized in hardware and implemented very simply using full adders and shift registers, if the two numbers are positive or if they are represented in signed magnitude notation. But if the signal and coefficients are represented in 2's complement notation, then the hardware becomes very complex and costly. In this report ROM's (Read Only Memories) are used as multiplier sections. A look up table is set up to show the input-output relationship. Every combination of the input signal is multiplied by the required coefficient and then shown on the look up table. The Read Only Memory is then constructed in such a way that when a particular signal appears at the input of the ROM, the corresponding multiplied signal appears at the output. A sample of the look up table for the three coefficients is given in Tables 1, 2, and 3.

Referring to equation (3.11), the coefficients α_1, β_1, and β_2 are given in decimal notation. They can be represented in binary form and their representation in binary bits is shown below.

$$\alpha_1 = 0.4515 \Longrightarrow \alpha_1 = 0.0111010$$
$$\beta_1 = 0.7375 \Longrightarrow \beta_1 = 0.1011110$$
$$\beta_2 = -0.2440 \Longrightarrow \beta_2 = 1.1100001 \ .$$

These coefficients are represented in 2's complement notation.

Table 1. Look up table showing multiplication of input data with $\alpha_1 = 0.0111010$.

Input	Output
0.0000000	0.0000000
0.0000001	0.0000000
0.0000010	0.0000001
0.0000011	0.0000001
0.0000100	0.0000010
.	.
.	.
.	.
0.1000001	0.0011101
0.1000010	0.0011110
0.1000011	0.0011110
0.1000100	0.0011111
0.1000101	0.0011111
.	.
.	.
.	.
1.0000000	1.0000000
1.1111111	1.0000000
1.1111110	1.1111111
1.1111101	1.1111111
1.1111100	1.1111110
.	.
.	.
.	.
1.0111111	1.1100011
1.0111110	1.1100010
1.0111101	1.1100010
1.0111100	1.1100001
1.0111011	1.1100001
.	.
.	.
.	.

Table 2. Look up table showing multiplication of input data with $\beta_1 = 0.1011110$.

Input	Output
0.0000000	0.0000000
0.0000001	0.0000001
0.0000010	0.0000001
0.0000011	0.0000010
0.0000100	0.0000011
.	.
.	.
.	.
0.1000001	0.0110000
0.1000010	0.0110000
0.1000011	0.0110001
0.1000100	0.0110010
0.1000101	0.0110011
.	.
.	.
.	.
1.0000000	1.0000000
1.1111111	1.1111111
1.1111110	1.1111111
1.1111101	1.1111110
1.1111100	1.1111101
.	.
.	.
.	.
1.0111111	1.1010000
1.0111110	1.1010000
1.0111101	1.1001111
1.0111100	1.1001110
1.0111011	1.1001101
.	.
.	.
.	.

Table 3. Look up table showing multiplication of input data with $\beta_2 = 1.1100001$.

Input	Output
0.0000000	1.0000000
0.0000001	1.0000000
0.0000010	1.0000000
0.0000011	1.1111111
0.0000100	1.1111111
.	.
.	.
.	.
0.1000001	1.1110000
0.1000010	1.1110000
0.1000011	1.1110000
0.1000100	1.1110000
0.1000101	1.1101111
.	.
.	.
.	.
1.0000000	0.0000000
1.1111111	0.0000000
1.1111110	0.0000000
1.1111101	0.0000001
1.1111100	0.0000001
.	.
.	.
.	.
1.0111111	0.0010000
1.0111110	0.0010000
1.0111101	0.0010000
1.0111100	0.0010000
1.0111011	0.0010001
.	.
.	.
.	.

Since the data to be multiplied with the coefficients is a signal coded in binary bits (8-bit word) and is also represented in 2's complement notation, it can take on any possible value from 0.0000000 to 1.1111111. So each of these numbers is multiplied with the coefficients, either by hand calculations or programmed to do the job on a digital computer, and is stored in the Read Only Memories. When the data input to the Read Only Memory is, for example, 0.0011010, the output will be a word which is the product of this number and the coefficient. This selection is done by the decoding network within the Read Only Memory.

The input signal to the ROM can take any combination of the 8 bits and the number of combinations equals 2^8 or 256. Thus the capacity of the ROM should be 2048 bits, in order to obtain 8-bit output for 8-bit input. As the number of bits increases the capacity of the memory also increases. A ROM of this capacity is now available on a chip measuring 90 by 110 mils through the use of metal oxide semiconductor technology. The memory's access time is 1 microsecond, which is longer than that of some other forms of memory, but is adequate for the purposes of the second-order filter.

A simple decoding network, using p-channel devices, for a 4-word memory is shown in Fig. 3.8. In general, the number of bits in each word may be taken to be N, but here only two bits are required for selecting the word. The scheme is such that one bit selects a row and the other bit a column. These then lead to the selection of the word required. It can also be extended to select the word from a 256-word memory.

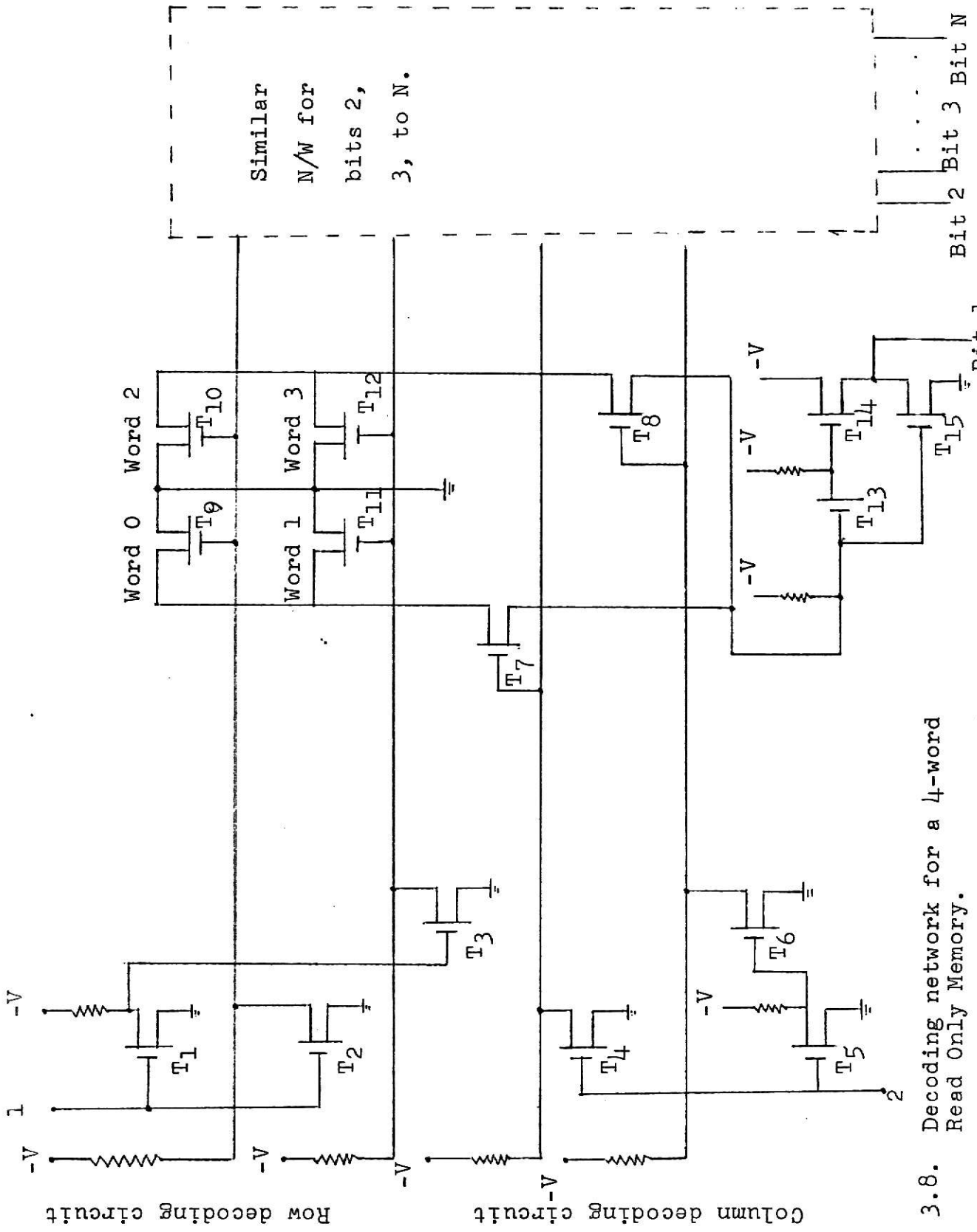


Fig. 3.8. Decoding network for a 4-word Read Only Memory.

Supposing the word 2 is to be read out of the memory; the address 2 in binary form is 10. The address line labeled 1 in the diagram is therefore at a zero potential and address line 2 is at a negative potential.

Remark: It is assumed that the logic used is

- volts - TRUE - 1

0 volts - FALSE - 0.

The zero potential at 1 will not allow the transistors T_1 and T_2 to switch on and they will remain at the cutoff state. Thus there will be no drop across the load resistor of T_1, and hence the gate of T_3 will be at a negative potential which will switch it to the saturation stage and put the row line that is its source at a zero potential. Thus the transistors controlled by this line (T_{11}, T_{12}, and others) are off. Since T_2 is off, the row line that is its source is negative. Thus transistors T_9 and T_{10} are gated on, together with transistors in other bit positions of words 0 and 2. Now the negative potential at 2 will switch transistors T_5 and T_4 on. The gate of T_7 will therefore be at zero potential, and hence it will not be switched on. In a similar manner, T_6 will be cut off so that the gate of T_8 will be at a negative potential which means it will be switched on. T_7 and T_8 have a common load resistor but since only T_8 is on, current will be supplied to the transistors in series with T_8, i.e., to T_{12} and T_{10}. But transistor T_{12} is already in the cutoff state, and hence T_{10} corresponding to one bit of word 2 is selected. Transistors T_{13}, T_{14}, and T_{15} buffer the voltage drop across the load resistor. Current passes

through T_{14} from the external circuit and this represents the reading out of a binary 1 from the memory. If a 0 is to be read from the memory, then the transistor corresponding to that bit in the word is coated with a thick oxide layer so that it will not be turned on even if its gate is negative. In that case current passes through T_{15} to the external circuit, representing a readout of a binary 0 from the memory.

MOS technology offers numerous advantages over bipolar technology. One of them is that bipolar memory cells exhibit parasitic capacitance and must therefore be isolated from one another. To provide this isolation together with suitable decoding circuits and buffers requires very difficult processing steps which increase cost and size of memory. In the case of a memory built with MOS devices, the entire storage matrix, input decoders, and output buffers are formed on a single monolithic substrate. Parasitic capacitance is greatly reduced, and therefore isolation is not necessary. This increases the component density which, in turn, improves the reliability and economy by permitting the packaging of complete functional entities together. Floyd Kvamme, of National Semiconductor, predicts that with the metal oxide silicon process, ROM's with capacities of 4096 bits per chip will be available soon. Memories with capacities of up to 2048 bits per chip are already in mass production. ROM's are among the least expensive forms of LSI because they comprise arrays of identical cells instead of collections of miscellaneous gates randomly connected. If a multiplier is implemented by the usual procedure, then it may

require complex logic circuits and full adders which may not be economically put on a chip. Thus ROM's offer a very economical and compact form of implementation of logic functions in digital filtering.

ROM's could have been used in place of adders but that involves lengthy look up tables which results in increase in memory capacity. For instance, in the addition of two 8-bit numbers, the input lines will be 16 and the different combinations of these 16 lines will be the total number of words which is equal to $2^{16} = 65,536$. Each word will be an 8-bit number, and hence the total capacity of the memory would be $65,536 \times 8 = 524,288$, which is impossible to attain on a single chip with the present technology.

The layout plan for the memory can be designed with the help of a computer; this helps in getting the most efficient plan for the implementation besides accommodating all the components in a minimum area.

CHAPTER IV

A/D AND D/A CONVERSION

As mentioned earlier, if the input to the digital filter is an analog or continuous signal, then it must be converted into digitized bits, which are coded binary bits, and after processing through the arithmetic unit the coded bits are reconverted to the original continuous form. These two operations are performed by the Analog to Digital (A/D) converter and Digital to Analog (D/A) converter, respectively.

D/A conversion is a straightforward process and is considerably easier than A/D conversion. In fact, a D/A converter is an integral part of an A/D converter. Hence a D/A converter will be considered first.

D/A Converter

The basic problem in converting a digital signal into an equivalent analog signal is to change the 8 digital voltage levels into an equivalent analog voltage. This can be most easily accomplished by designing a resistive network which will change each of the digital levels into an equivalent binary weighted voltage (or current).

A resistive ladder network is most suitable and a D/A converter using this principle is shown in Fig. 4.1 with a block diagram and a schematic logic diagram.

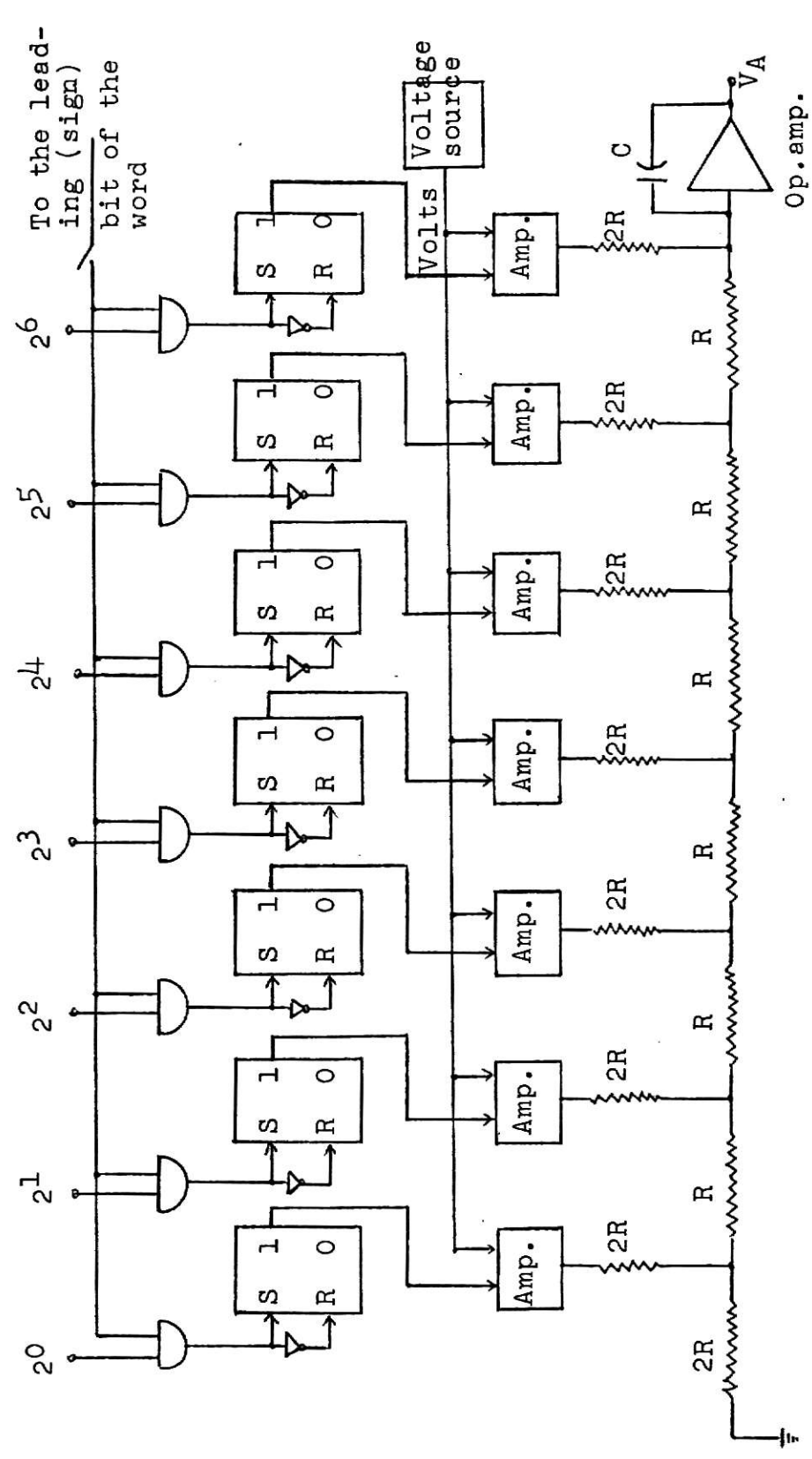


Fig. 4.1. Schematic diagram of 8-bit D/A converter.

An integral part of the D/A converter is the register which is used to store the digital information. Then there are level amplifiers between the register and the resistive network to insure that the digital signals presented to the network are all of the same level and are constant. Also there must be some form of gating on the input of the register such that flipflops can be set with the proper information from the digital system.

The level amplifiers work in such a way that when the input from the flipflop is high, the output of the amplifier is at V volts and when the input from the flipflop is low, the output is 0 volts.

The flipflop on the right represents the most significant bit (MSB) and the flipflop on the left represents the least significant bit (LSB). Each flipflop is of the RS type and requires a positive level at the R or S inputs to reset or set it. Finally in order to hold a voltage during the sampling intervals, a sample and hold circuit is used which can be approximated by the capacitor and the high-gain amplifier.

If the input signal is a negative signal represented in 2's complement notation, then a 2's complementer must be included. Since it is needed for both an A/D and a D/A converter it is described in the Appendix.

A/D Converter

The A/D conversion is a more complicated operation than the D/A conversion. Often a D/A converter is an integral part of the A/D converter.

The basic element in an A/D converter is a comparator which compares the analog voltage with some other reference voltage and switches other circuits to give the digital bits. Out of the different types of A/D converters available, the counter type is the most suitable as the reference voltage is continuously changed until it equals the analog voltage. In a simple realization of this type of A/D converter, an 8-bit binary counter is used and the digital output signals are taken from this counter. The output of this counter is connected to a binary ladder to form a D/A converter. If a clock is applied to the counter, then the output of the binary ladder will be a staircase type waveform which is the exact reference voltage required.

A block diagram of a counter type A/D converter is shown in Fig. 4.2.

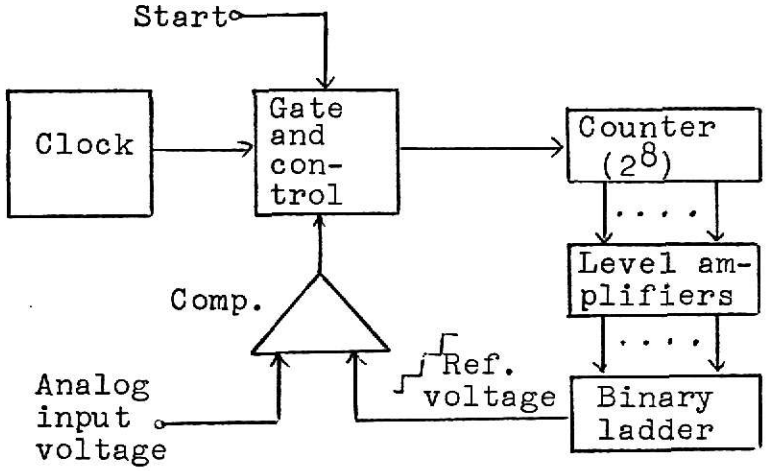


Fig. 4.2. Block diagram of A/D converter.

A detailed diagram showing the operation of the A/D converter, the clock pulses to toggle the shift registers and start the A/D converter each sample time, and different gates controlling the operation is shown in Fig. 4.3.

The figure shows a master clock, a 2^9 counter, and an A/D converter. All the flipflops are reset to zero except the control flipflop, which is set to 1, by the start-up pulse. When the control flipflop is set to 1, the AND gate is open and the 2^8 counter starts counting the clock pulses passing through the gate. The 2^8 counter advances through its normal binary count sequence and a staircase waveform is generated at the output of the ladder. This waveform is compared with the analog input voltage and when it equals (or exceeds) the input analog voltage, the control flipflop is reset to 0. As a consequence the AND gate is closed. The counter stops and the conversion is complete. The number stored is the digital equivalent of the analog input voltage. In the meantime, the clock pulses from the master clock advance the 2^9 counter through its binary sequence. The frequency of the master clock is selected in such a way that the sampling time of the signal equals the time required to advance the 2^9 counter through its full count. At the end of the count, a start pulse from the 2^9 counter initiates the conversion process again by setting the control FF to 1 and resetting the FF's of 2^8 counter to 0. The parallel output from the 2^8 counter is the input to a shift register operated by clock pulses from the 2^9 counter. This is used to insure that the system does not operate in an erratic manner. The

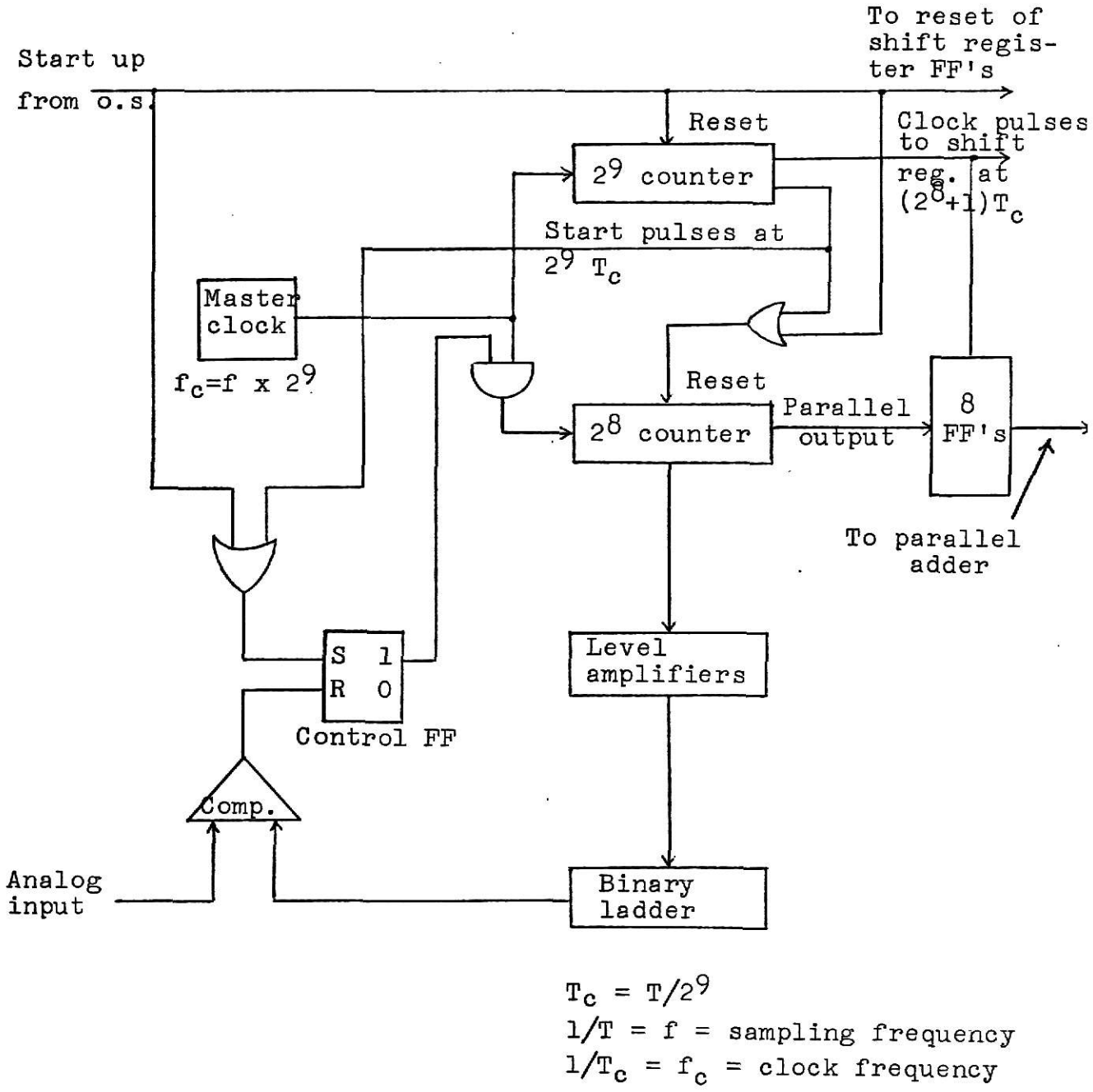


Fig. 4.3. Block diagram showing A/D converter and control signals.

output of this shift register is then connected to the parallel adder. The toggle pulses for this shift register and the other shift register, used as a delay element, are obtained from the 2^9 counter at $(2^8 + 1)\ T_c$ of each sample.

Figure 4.4 shows the complete block diagram of the digital filter with the control signals.

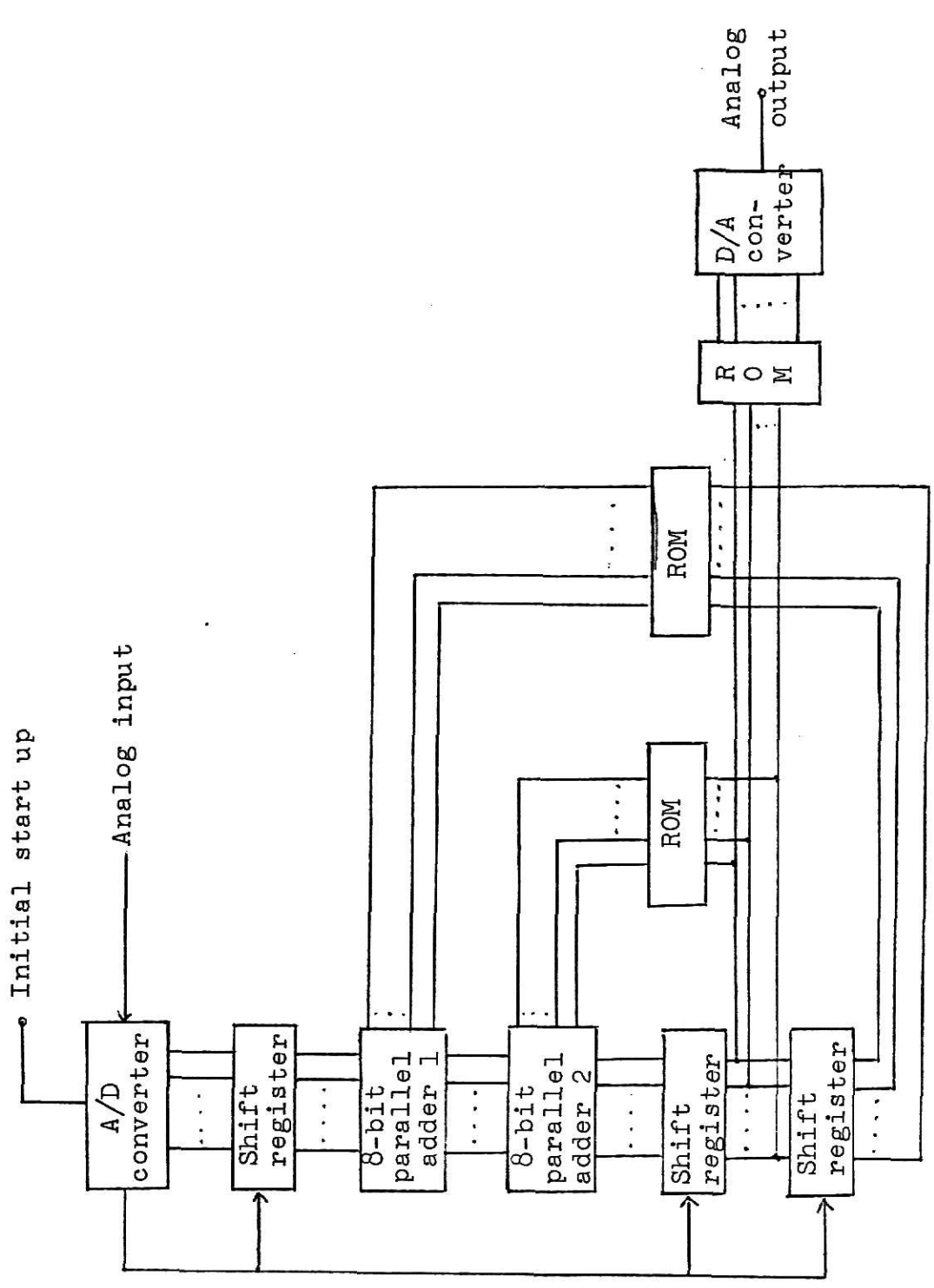


Fig. 4.4. Block diagram of the digital filter.

CHAPTER V

CONCLUSION

With the advent of digital electronics, where all the signal processing is done by digital bits, it is quite natural to assume that most of the analog processing devices have to be replaced by their digital counterparts. At the present time, digital filters are mainly used in the fields of seismography, oceanography, digital data processing, touch-tone and selective call systems, and medical electronics. It is also true that much research work is going on in the pulse-coded modulation of voice signals. This offers an extremely good signal to noise ratio and since the voice signal is translated in terms of digital bits, the filtering operation at the receiver end has to be done by digital filters.

The advancing LSI technology offers a very promising future for digital filters. It is now possible to accommodate a 2048-bit memory on a chip, or a full adder on a chip. This improvement in LSI technology very much reduces the size as well as the cost of the various components of the filter. Moreover, the digital filters are economical in the very low frequency band (0.01 to 1.0 Hz) where the size of the passive analog components becomes appreciable or impossible to realize.

Eventually it should be possible to design at least simple digital filters on a single LSI chip. More complex digital filters could be realized by interconnecting a relatively small number of LSI chips.

ACKNOWLEDGMENT

The author wishes to express his gratitude and appreciation to his major professor, Dr. Dale E. Kaufman, for his very valuable suggestions and comments during the preparation of this report.

LIST OF REFERENCES

1. Sydney C. Silver, "The Digital Filter: Potent Processing Tool." Electronic Products, March, 1970, pp. 32-37.

2. David J. Novak and Pierre E. Schmid, "Introduction to Digital Filters." IEEE Transactions on Electromagnetic Compatibility, Vol. EMC-10, No. 2, June, 1968, pp. 210-220.

3. Rader and Gold, "Digital Processing of Signals." McGraw-Hill Book Company, 1969.

4. J. F. Kaiser, "Some Practical Considerations in Realization of Linear Digital Filters." 1965 Proceedings 3rd Allerton Conference on Circuit and System Theory.

5. Jackson, Kaiser and McDonald, "An Approach to the Implementation of Digital Filters." IEEE Transactions on Audio and Electroacoustics, Vol. AU-16, No. 3, Sept., 1968.

6. Thomas C. Bartee, "Digital Computer Fundamentals." McGraw-Hill Book Company, 1960.

7. Y. Chu, "Digital Computer Design Fundamentals." McGraw-Hill Book Company, 1962.

8. Lee L. Boysel, "Memory on a Chip, A Step Toward Large Scale Integration." Electronics, Feb. 6, 1967.

9. Lee L. Boysel, "Adder on a Chip: LSI Helps Reduce Cost of Small Machines." Electronics, March 18, 1968.

10. Floyd Kvamme, "Standard Read-only Memories Simplify Complex Logic Design." Electronics, Jan. 5, 1970.

11. Albert Hemel, "Making Small ROM's Do Math Quickly, Cheaply, and Easily." Electronics, May 11, 1970.

12. Malvine and Leach, "Digital Principles and Applications." McGraw-Hill Book Company, 1969.

13. Donald E. Lancaster, "RTL Cookbook." Howard W. Sams and Co., Inc., 1969.

14. William Penny, "Doing Logic with M.O.S.". The Electronic Engineer, March, 1970.

APPENDICES

APPENDIX A

Doing Logic With MOS

In d-c logic with p-channel MOS devices, a relatively high impedance MOS device which needs very little chip area is used in place of a diffused load resistor which required much more area. This MOS load device is held conducting at all times by returning its gate to a negative potential sufficient to overcome its gate threshold voltage. The gate is usually returned to the drain supply voltage so that a single power supply suffices. The gate may also be returned to a supply which is more negative than the drain supply.

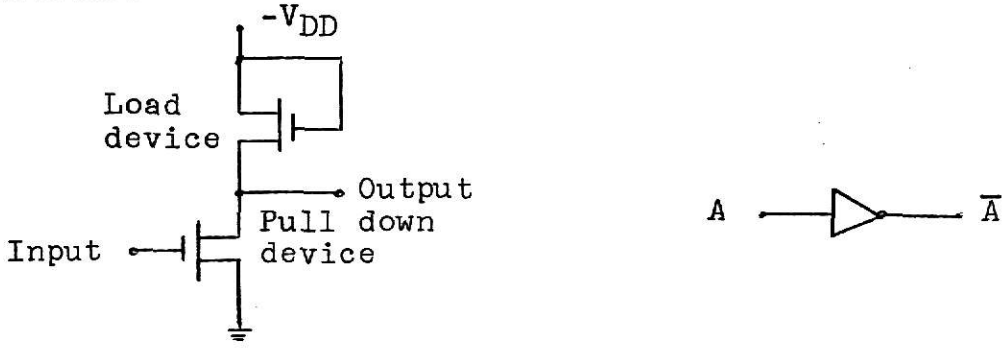
Operation of the Inverter

Logic used:

$-V_{DD}$ volts $\Longrightarrow$ 1

0 volts $\Longrightarrow$ 0.

Figure A.1 shows the schematic and logic diagrams of an inverter.



a. Schematic diagram. b. Logic diagram.

Fig. A.1. Inverter.

With the pull-down device on, the output node falls to ground. However, when the pull-down device is off, the output node goes to $-V_{DD}$. Using the logic defined above, it is seen that when the input is 1($-V_{DD}$), the output is 0 (ground), and when the input is 0 (ground) the output is 1 ($-V_{DD}$); thus the device operates as an inverter.

The load device is designed to have an impedance about 20 times that of the pull-down device. This ratio has been selected to assure that the output node goes to near ground when the inverter is conducting.

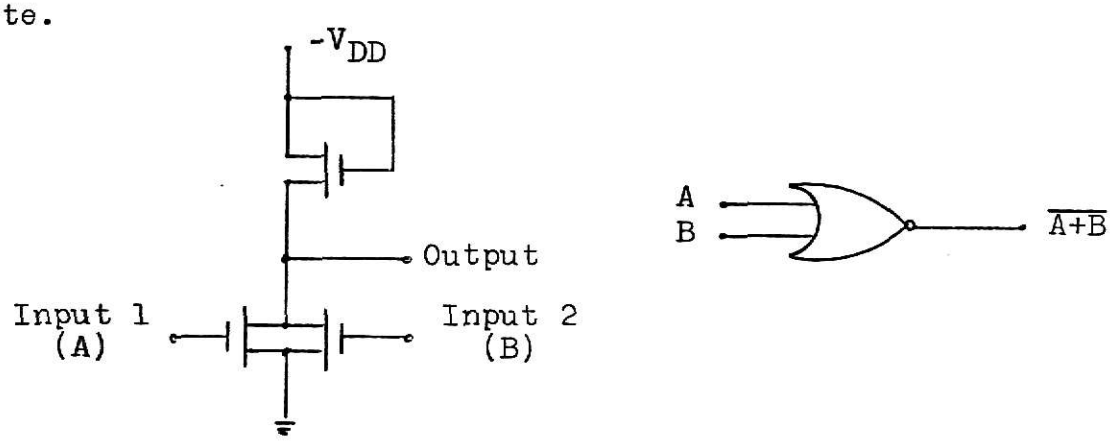
Operation of a NOR Gate

Logic used:

$-V_{DD}$ volts $\Longrightarrow$ 1

0 volts $\Longrightarrow$ 0.

Figure A.2 shows the schematic and logic diagrams of a NOR gate.



a. Schematic diagram. b. Logic diagram.

Fig. A.2. NOR gate.

From the diagram it is seen that when either A or B is 1, the output is 0, whereas if both A and B are 0 the pull-down devices are in the off state and the output is $-V_{DD}$ volts which is logic 1. Other requirements are the same as for the inverter.

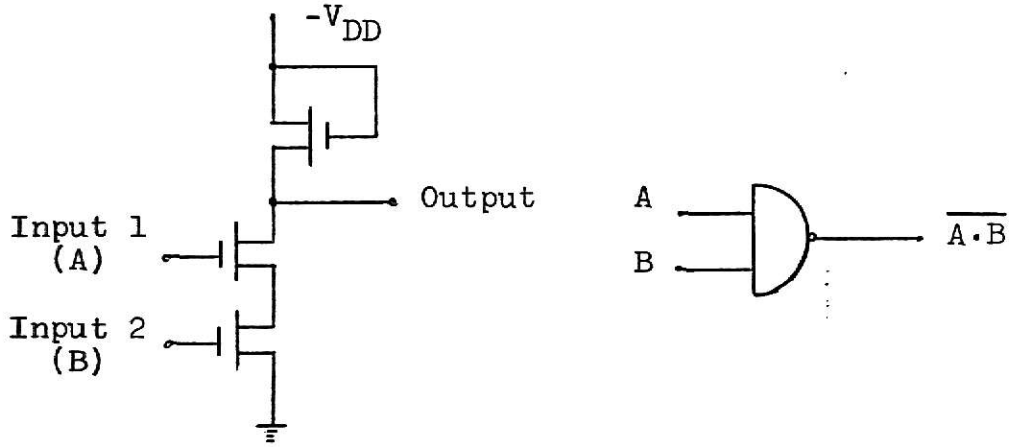
Operation of a NAND Gate

Logic used:

$-V_{DD}$ volts $\Longrightarrow$ 1

0 volts $\Longrightarrow$ 0.

Figure A.3 represents the schematic and logic diagrams of a NAND gate.



a. Schematic diagram. b. Logic diagram.

Fig. A.3. NAND gate.

The two pull-down devices are connected in series which makes the gate operate as a NAND gate. If either of the inputs A or B is 0 (ground), the corresponding transistor is off, which makes output voltage equal to $-V_{DD}$ volts (logic 1). On the other hand, if both the inputs are 1 ($-V_{DD}$ volts), both the transistors conduct, clamping the output to ground (logic 0). Thus the device operates as a NAND gate.

APPENDIX B

2's Complement Circuit

In the case of the input analog signals there must be a way of representing the negative signal by binary bits. This is accomplished by 2's complement representation. This 2's complement is usually obtained as follows.

No:	0 1 0 1
1's complement	1 0 1 0 (inversion)
Add	1
2's complement	1 0 1 1

Thus it is seen that in order to obtain the 2's complement of a number it is first inverted and a one is added to the resulting number. In hardware this is accomplished as shown in Fig. B.1.

The "1" at the input to the first half adder can be obtained from the leading bit of the signal. It is seen that for a parallel 2's complementer the implementation is quite expensive as the number of inverters and half adders are equal to the number of bits in the number. In the case of serial 2's complementer, the hardware implementation is quite simple and not so expensive.

The 2's complement circuit is needed both for the A/D converter and the D/A converter. In the case of the A/D converter the binary output for negative signals is passed through the

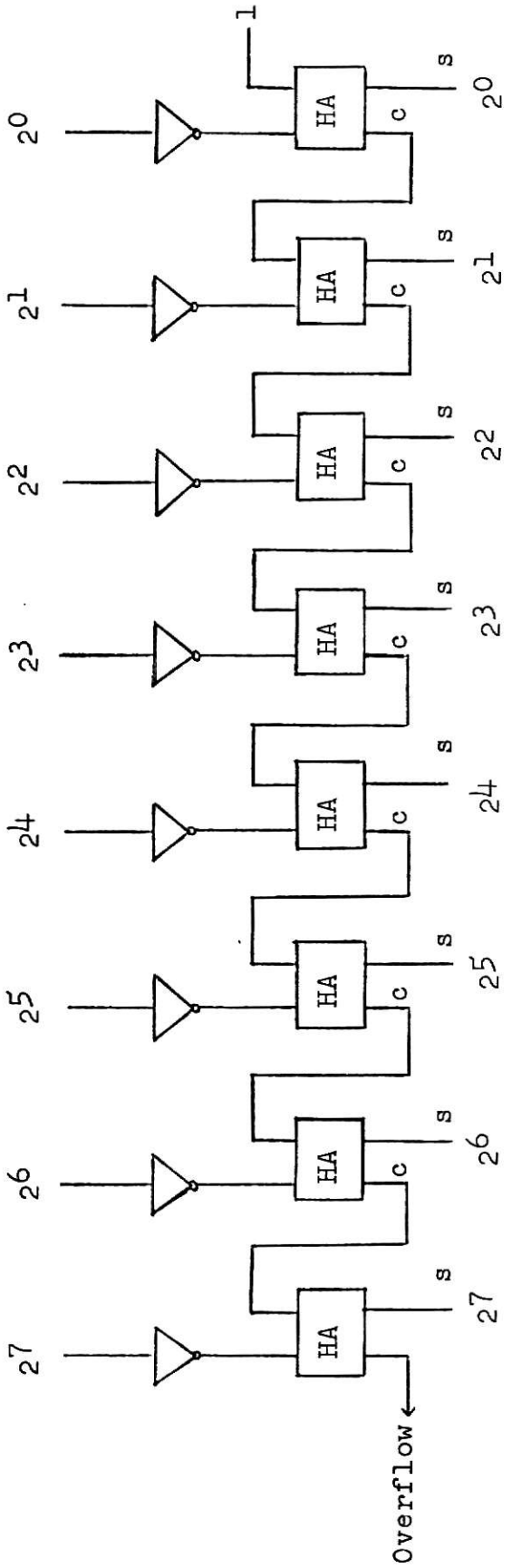


Fig. B.1. Two's complement circuit.

2's complementer to obtain the required 2's complement form for processing. In the case of a D/A converter the negative signals represented in 2's complement form are again passed through the circuit to obtain the true magnitude and consequently the negative signal.

AN IMPLEMENTATION OF A LOW-PASS DIGITAL FILTER

by

A. P. VISWANATHAN

B. S., Punjab Engineering College,

Chandigarh, India, 1969

AN ABSTRACT OF A MASTER'S REPORT

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Electrical Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1970

An implementation of a second-order Butterworth filter is shown using an A/D converter, parallel adders, JK shift registers, read only memories, and a D/A converter. The impulse response of the transfer function in s-domain is transformed to this z-domain using z-transforms and the corresponding transfer function is then implemented using digital circuits. Read only memories are used in place of multipliers whose implementation is very complex and costly if the incoming signals are represented in 2's complement notation. Advancing LSI technology promises to accommodate such a 4096-bit memory on a chip measuring less than a square inch. The clock pulses needed to toggle the shift registers are obtained from the A/D converter which has a master clock controlling the sampling rate. The clock pulses are obtained from a 2^9 counter which is toggled by the master clock. A counter type A/D converter is used and the resetting of the counter and setting of the control flipflop at the beginning of each sample are done by pulses from the 2^9 counter. A resistive ladder network is used for D/A conversion with an operational amplifier and a capacitor used as a hold circuit. Parallel adders are used in the filter for they require no control signals, act rapidly, and are amenable to multiplexing.