SU-SEL-68-084

Circuit Synthesis Utilizing Digital Variable-Precision-Integrating and Summing Elements

by

Chia-peir Yu

December 1968

Scientific Report No. 30

Prepared under
National Aeronautics and Space Administration
Research Grant No. NGL 05-020-014

RADIOSCIENCE LABORATORY

STANFORD ELECTRONICS LABORATORIES

STANFORD UNIVERSITY • STANFORD, CALIFORNIA

SEL-68-084

CIRCUIT SYNTHESIS UTILIZING DIGITAL VARIABLE-PRECISION-INTEGRATING AND SUMMING ELEMENTS

by

Chia-peir Yu

December 1968

Scientific Report No. 30

Prepared under

National Aeronautics and Space Administration
Research Grant NGL 05-020-014

Radioscience Laboratory
Stanford Electronics Laboratories
Stanford University Stanford, California

ABSTRACT

In the electrical network, transfer and driving-point functions have, in the past, been designed and implemented by using either passive elements or a combination of active and passive elements. These analog network functions possess such problems as drift, unsteady gain, and temperature sensitivity, causing variations from the desired response. In this report, an alternative digital approach is suggested which could overcome these problems.

General-purpose digital computers have been utilized to numerically generate transfer functions using sampled digital data provided by analog-to-digital converters. Because of the limitation on operating speed of the central processor, a typical general-purpose computer can realize only real-time digital filters with input frequencies up to a few kilohertz. In addition, this type of computer is too complex and costly to be widely used as a replacement for analog filters in most electronic systems.

To permit the implementation of both transfer and driving-point functions, a pair of integrated-circuit digital building blocks are proposed. These are a digital increment integrator module and a digital increment summing module to be used as the building blocks to replace all or the majority of all possible electrical components and networks. The choice of these two types of semi-universal modules rather than one universal module is justified by a minimum-cost criterion.

Three synthesis methods and their performance errors are presented and compared, from which one procedure is deduced to yield a network function with a minimum number of modules as well as best performance.

All data transfer among the digital modules is in digital increments. Therefore, the analog-to-digital incremental converter is used at the input terminal. This results in faster conversion and better accuracy than the analog-to-digital converter.

In the internal design of the digital modules, a signed-digit number system is used to eliminate the carry-propagation time, and a modified trapezoidal rule is introduced to perform high-speed integration. With the help of large-scale integration, the complexity of the two proposed

modules is on the order of a few hundred logical AND-OR gates, and variable precision can be achieved by cascading the modules together. With an accuracy of 0.1 percent, the input frequency can be handled up to 100 kHz at a clock rate of 10 MHz, or up to 1 MHz with a 1-percent accuracy.

CONTENTS

CONTENTS (Cont)

ILLUSTRATIONS

ILLUSTRATIONS (Cont)

ILLUSTRATIONS (Cont)

TABLES

ACKNOWLEDGMENT

The author wishes to express his sincere appreciation and indebtedness to Professor Allen M. Peterson, who supervised this research and whose invaluable suggestions, constant guidance, and encouragement made this work possible. The interest displayed by Professor R. W. Newcomb during this investigation and his comments after reading the manuscript are also highly appreciated. The suggestions of Professor F. W. Crawford concerning the manuscript are also gratefully acknowledged.

Thanks are due the IBM Corporation for the Residence-Study Fellowship granted the author for the year 1968, which enabled him to complete this study.

Chapter I

INTRODUCTION

Discrete signal processing by linear filters or weighting sequences originated in the early 1600's with the work of mathematicians who constructed mathematical tables and with astronomers who were concerned with the determination of the orbits of heavenly bodies. The works of Napier, Newton, the Bernoullis, Euler, Lagrange, Laplace, and Gauss are evident in the classical numerical analysis techniques used even today for numerical integration, interpolation, differentiation, and so on. The historical development of digital filtering was thoroughly described by Kaiser [Ref. 1]. In recent publications [Refs. 1,2,3, and 4], the basic principle in the design of real-time digital filters is that the analog input signal undergoes the process of spectrum shaping then feeds through digital hardware. In this hardware, a general-purpose computer is utilized to perform such operations as delay, storage, addition, subtraction, and multiplication in such a way as to satisfy a set of specified difference equations between input and output. In other words, the digital output of the computer, after a set of calculations, and the digital input to the computer satisfy a specified transfer function in z-transform, which can be obtained by direct transformation from the Laplace transform. From the transfer function, a set of difference equations can be found and executed on the digital computer; the data output can be collected as the final output, and the transfer function of a digital filter is realized. This technique of filtering by implementation of difference equations is sometimes referred to as "recursive filtering" (if the present value of the output depends not only on the present and past values of the input but also on the previous values of the output) or "nonrecursive filtering" (if the present value of the output does not depend on the previous values of the output). See Ref. 1.

Computer implementation has gained much attention [Refs. 1,3,4, and 5], and many programs have been written to implement digital filters. Because of the fairly slow operating speed of the central-processor unit, the numerical calculation performed by the arithmetic unit can

only realize those real-time digital filters that handle an input signal frequency up to a few kilohertz. Furthermore, certain spacecraft applications where small size and light weight are extremely important, computer simulation is impractical.

The motivation for this investigation is based on the following questions: Can we apply a digital technique other than computer simulation to realize analog networks? Can we construct a small, light-weight, integrated universal or semi-universal digital module to be used as the only building block for replacement of all or the majority of possible electrical components and networks? Can we develop a synthesis procedure which would result in a network having a minimum number of digital modules with best possible performance? How can the speed of the universal digital module be increased to the domain of real time? How can the accuracy of the network be improved, or can the precision be varied to fit our different needs?

A feasibility study, answering some of these questions, has been reported [Ref. 6], where it was shown that the realization of transfer and immittance functions by digital building blocks can be done. A digital integrator (a device containing digital building blocks such as registers, adders, and logic gates which perform digital integration) was used as the basic building block for the realization of the transfer and driving-point functions; therefore, the mathematical models were differential rather than difference equations. Although the Laplace transformation was used to specify the transfer and driving-point functions, the outputs were not precisely the same as those outputs specified by the transformation. However, they are essentially a very good approximation and are compatible with the continuous (analog) type of network.

Digital increments are the only data transferred between the digital integrators and the summing elements. By this means, much shorter time is needed, compared to transferring the full word. If an analog signal only is available at the input, a device called analog-to-digital incremental converter (ADIC) can be used, which converts the analog difference of the present and previous analog inputs to digital increments whereby higher accuracy and fast conversion time is achieved. More will be said concerning the ADIC in Appendix A.

If incremental analog-to-digital and digital-to-analog converters (DAC) are available, all driving-point functions can be realized by using digital elements only. It was also hoped that both ADIC and DAC could be integrated because all driving-point functions could then be integrated without difficulty, and the digital equivalent of high Q inductors or high-capacity capacitors could be made available.

Prince and Sendzuk [Ref. 7] presented a paper describing how digital filters can be built by using MOS (metal-oxide silicon) chips; however, the inherent slow speed of MOS makes these filters impossible for real-time application.

Recent progress in large-scale integration (LSI) techniques advances theoretical assumptions well along the way toward reality [Ref. 8]. (LSI refers to complexity levels greater than 100 circuits.) To take full advantage of these techniques, logic functions can be complicated to such an extent that as many gates as possible can be contained on a module, subject only to a minimum-cost constraint. With these considerations in mind, the idea of using universal modules as basic building blocks to realize any real-time circuits, such as circuits formed by any combination of R, L, C elements (and many others formed beyond the restrictions of R, L, C elements), can be accomplished.

To operate as a real-time system, the circuitries involved in the digital modules must be very high speed. Therefore, high-speed circuitry is needed, combined with a parallel-operated arithmetic unit that has minimum possible carry propagation. To perform this high-speed digital integration, and after the failure of all classical methods, a modified trapezoidal integration technique is derived. Parallel operation requires too many input-output pins, which will greatly increase the cost of the module; however, a minimum-cost criterion has been introduced as a justification. A nonconventional redundant number system, called the signed-digit number system [Refs. 9 and 10], is used in the design of the digital modules, where the carry-propagation chain is eliminated at the expense of using more complicated logic. The problem of the variable-precision scheme was also resolved. With a clock rate of 20 MHz, the input signal frequency can be handled up to a few hundred kilohertz with an accuracy of 0.1 percent, which is, of course, more than sufficient for networks.

As is known in network theory, variations of parameters, as individual amplifier gains, often result in small variations in the response of the cascade structure because the high number of feedback paths in the noncascade structure is reduced to one or two in cascade form [Ref. 11]. Consequently, the sensitivity performance is often improved by factoring the transfer function into terms, each having a degree of one or two. A performance study in the digital domain has been carried out by Knowles and Edwards [Ref. 12] and Knowles and Olcayto [Ref. 13]. Their results indicate that the cascade form of digital realization exhibits somewhat less performance error than the parallel and direct forms.

The synthesis method of digital realization of network functions using the two types of proposed semi-universal digital modules (digital-integrating and digital-summing elements) is given. Here, optimization, based on the minimum-cost criterion, indicates its choice, which coincides with the best-performance choice mentioned earlier.

With further advancement in LSI techniques, all R,L,C elements and their combined circuits could be replaced by the above two types of modules in which all electrical circuits, linear or nonlinear, time-variant or time-invariant, can be designed by using these two semi-universal elements only. The negative elements can be realized as well.

Chapter II describes the principles of digital integration and explains how the digital integrator is operated and implemented. Also included is a brief discussion of the classical numerical methods and their drawbacks under fast operating speed, resulting in the modification of the classical trapezoidal method of integration.

Chapter III discusses optimization of the digital modules with emphasis on their application to the network functions. This optimization is based on fast speed, variable precision, and minimum cost. The signed-digit number system and a modified scheme of trapezoidal integration are utilized to implement the digital modules. Two types of modules, i.e., digital variable-precision-integrating and summing elements, are proposed as the semi-universal modules for network-function realization.

Chapter IV covers the digital realization of the transfer function by using the two-element modules, and synthesis methods are discussed.

The synthesis procedure, which would result in a network realization containing a minimum number of digital modules with the best performance, is also given.

Chapter V illustrates some applications of the digital realization of transfer functions; in particular, digital filters, the digital spectrum analyzer, and digital filters with time-varying coefficients are discussed and examples are given.

Chapter VI presents the possibility of digital realization of driving-point immittance functions, i.e., impedance and admittance functions.

Chapter VII summarizes the results and suggests areas for further research.

Chapter II

DIGITAL INTEGRATION

A. Principles of Digital Integration

In the method of numerical integration, a given function can be approximated by some polynomial over a short interval T, following which the polynomial rather than the original given function is integrated.

1. Classical Numerical-Integration Methods

The most frequently used classical methods are rectangular and trapezoidal integration rules [Refs. 14 and 15].

a. Rectangular Integration

Suppose that the integral of a given function $y = f(x)$ needs to be found. This integral represents the area bounded by the curve $y = f(x)$ and the abscissa, but it can be approximated as the sum of the areas of the elementary rectangles. The height of each of the rectangles is the current ordinate y and the base is the increment Δx of the independent variable x; each increment is obtained by dividing the entire range of x into equal increments. As shown in Fig. 1, the integral z is found to be

$$z = \int_{x_o}^{x} y\,dx = \lim_{n \to \infty} \sum_{i=1}^{i=n} y_i \Delta x_i + R_o \approx \sum_{i=1}^{i=n} y_i \Delta x_i + R_o \qquad (2.1)$$

where $y = f(x)$, $y_i = f(x_i)$, and R_o is the initial value of the integral.

$$\Delta x_i = x_{i+1} - x_i \qquad (2.2)$$

is the increment of the independent variable x. If we make $\Delta x_i = T$, then

$$z \approx \sum_{i=1}^{i=n} y_i T + R_o \qquad (2.3)$$

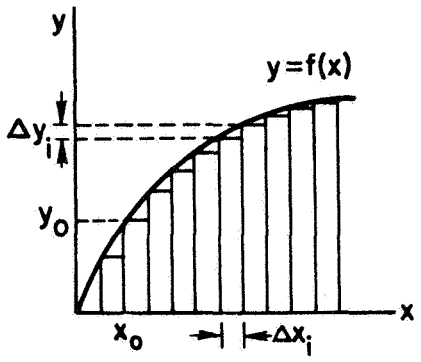


Fig. 1. RECTANGULAR INTEGRATION.

To compute the integral, it is sufficient to add all the ordinates corresponding to each of the elementary rectangles. Consequently, the integration process reduces to a summation of the members representing those ordinates. Each ordinate is equal to the preceding one, plus (or minus) the ordinate increment Δy_i for the interval $\Delta x_i = x_{i+1} - x_i$. Thus, the value of y_i can be computed by adding Δy_i to each preceding ordinate. For example,

$$y_1 = y_o + \Delta y_1$$

$$y_2 = y_1 + \Delta y_2$$

$$\vdots$$

$$y_n = y_{n-1} + \Delta y_n = y_o + \Delta y_1 + \Delta y_2 + \ldots + \Delta y_n \qquad (2.4)$$

Hence, it is seen that the current value of y_i can be obtained by accumulating all increments of the ordinate up to Δy_i.

If $\Delta x_i = T$, then in the difference equation, the rectangular integration performs

$$R_k = R_{k-1} + Ty_k \qquad (2.5)$$

where R_k = the area accumulated from x_o up to x_k

y_k = the ordinate at x_k

or in terms of the z-transform, the transfer function H(z) defined as the ratio of $Z[R_k]$ to $Z[y_k]$ is

$$H(z) = \frac{R(z)}{Y(z)} = \frac{T}{1 - z^{-1}} \qquad (2.6)$$

b. Trapezoidal Integration

The error in rectangular integration can be reduced by using the trapezoidal rule, where the curve $y = f(x)$ at each interval Δx is approximated by a chord. This rule is equivalent to a summation of the areas of "mean" rectangles; as shown in Fig. 2, each rectangle has a mean ordinate approximated by

$$y_o + \frac{1}{2} (\Delta y)$$

The increment of the integral (the area of an elementary rectangle) is

$$\Delta z = \left(y + \frac{1}{2} \Delta y\right) \Delta x$$

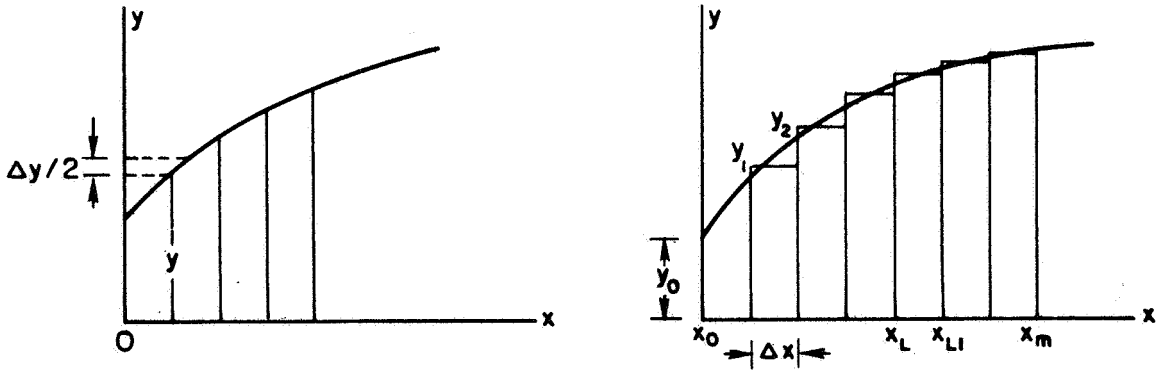


Fig. 2. TRAPEZOIDAL INTEGRATION.

If the trapezoidal-integration method is employed, the integral is approximated by

$$\int_{x=x_o}^{x_n} f(x)\, dx \approx \frac{x_n - x_o}{n} \left(\frac{y_o + y_1}{2} + \frac{y_1 + y_2}{2} + \ldots + \frac{y_{n-1} + y_n}{2}\right) \tag{2.7}$$

$$= \frac{x_n - x_o}{n} \left(\frac{y_o + y_n}{2} + y_1 + y_2 + \ldots + y_{n-1}\right)$$

If $\Delta x = T$, the trapezoidal integration performs

$$R_k = R_{k-1} + \frac{T}{2}(y_k + y_{k-1}) \tag{2.8}$$

or in terms of the z-transform,

$$H(z) = \frac{R(z)}{Y(z)} = \frac{T}{2}\,\frac{(1 + z^{-1})}{(1 - z^{-1})} \tag{2.9}$$

2. Modified Trapezoidal Integration

For a fast-integration algorithm, it is required that R_k and y_k of Eq. (2.8) be calculated at the same time, but y_k must be available when R_k is to be executed. Therefore, Eq. (2.8) states an unrealizable algorithm, and all available numerical-integration rules [Ref. 14] are unrealizable under the assumption of fast operation.

A modified trapezoidal-integration algorithm can provide a method of fast integration:

$$\Delta Y_k = y_k - y_{k-1} \tag{2.10a}$$

$$Y_k = Y_{k-1} + \Delta Y_{k-1} \tag{2.10b}$$

$$R_k = R_{k-1} + TY_k + \frac{T}{2}(\Delta Y_{k-1}) \tag{2.10c}$$

As indicated previously, T is the sampling interval (the period between two clock pulses). y_k and R_k are present values of the coordinate and the area, respectively, and Y_k is the content of the Y register at $t = kT$. It is seen also that R_k and Y_k can always be formed at the same time; therefore, the modified-trapezoidal integration algorithm is realizable. Modification is based on the condition that the trapezoidal correction is introduced at a 1-bit time later.

The integration performed by Eq. (2.10) is shown in Fig. 3, and the differences between it and Fig. 4, as shown by the shaded area, can be observed. If $y_o = 0$ or the Y register has been set to zero for unknown y_o prior to the start of integration, then the integration by the trapezoidal rule (tr) of Eq. (2.8) and by the modified trapezoidal rule (mtr) of Eq. (2.10) differs by a small triangular area $T(y_k - y_{k-1})/2$, or

$$(R_k)_{tr, y_o = 0} = (R_k)_{mtr, y_o = 0} \pm \left| \frac{T}{2} (y_k - y_{k-1}) \right| \qquad (2.11)$$

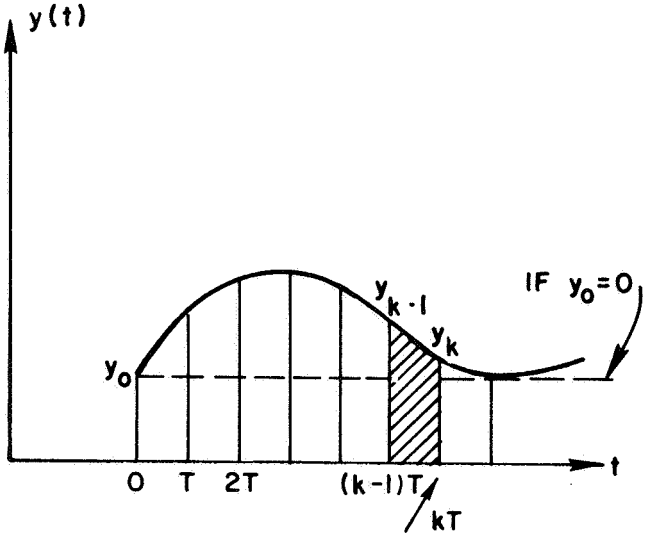


Fig. 3. MODIFIED TRAPEZOIDAL INTEGRATION, EQ. (2.10).

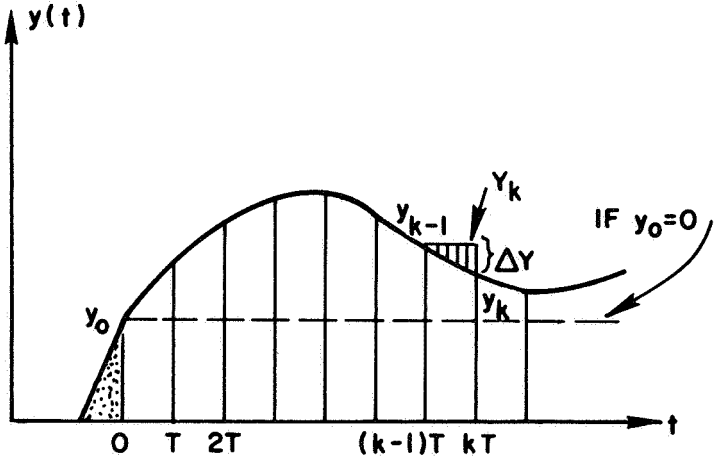


Fig. 4. TRAPEZOIDAL INTEGRATION, EQ. (2.8).

If the initial condition $y_o \neq 0$ is known at the beginning of integration, $(\text{Y register})_{t=0} = y_o \neq 0$ and $(\text{R register})_{t=0}$ will be

assumed zero. Under such an assumption, Eqs. (2.8) and (2.10) will produce one extra area, as shown in black in Figs. 3 and 4. Therefore, in general, Eq. (2.11) can be modified for any y_o as

$$(R_k)_{tr,\ Y_o = y_o} = (R_k)_{mtr,\ Y_o = y_o} \pm \left|\frac{T}{2}(y_k - y_{k-1})\right| - \frac{T}{2}y_o \qquad (2.12)$$

This discussion is based on the assumption that the calculation will start at $t_1 \neq t_o$, and at t_o only initial-value transfer is performed.

In the case where Y_o has been set to zero without y_o actually known beforehand, calculation starts from time t_o and Eq. (2.12) is applicable. For a real-time digital filter, the initial value Y_o is always equal to zero; therefore, no initial-value register is needed. Note that t does not necessarily denote time. Equation (2.10) yields

$$\begin{aligned} R(kT) &= R[(k-1)T] + TY(kT) + \frac{T}{2}\Delta Y[(k-1)T] \\ &= R[(k-1)T] + Ty[(k-1)T] + \frac{T}{2}\{y[(k-1)T] - y[(k-2)T]\} \\ &= R[(k-1)T] + \frac{3T}{2}y[(k-1)T] - \frac{T}{2}y[(k-2)T] \end{aligned} \qquad (2.13)$$

The z-transform of the above difference equation obtains

$$R(z) = z^{-1}R(z) + \frac{3T}{2}z^{-1}Y(z) - \frac{T}{2}z^{-2}Y(z) \qquad (2.14a)$$

or

$$R(z) = \frac{T(3z^{-1} - z^{-2})}{2(1 - z^{-1})}Y(z) \qquad (2.14b)$$

The error analysis of the modified trapezoidal integration and the comparison with the error in the classical trapezoidal integration are presented in Appendix G, where the modified scheme is found to be as good as the unmodified one.

B. The Digital Integrating Element

The operation of the digital integrator is more or less the same for all integration methods. Only those operations performed by

rectangular and modified trapezoidal rules are discussed in the following sections.

1. Operation of the Digital Integrating Element with the Rectangular-Integration Rule.

In Fig. 5, it is assumed that increments Δx, Δy, and Δz are in the form of individual pulses. The reversible counter 1 counts the incoming pulses. For each integration step, pulses from several input channels are accepted, and the number of pulses accumulated is regarded as an increment of the integrand, where that increment is the sum of several elementary increments, i.e., $\Sigma\Delta y$.

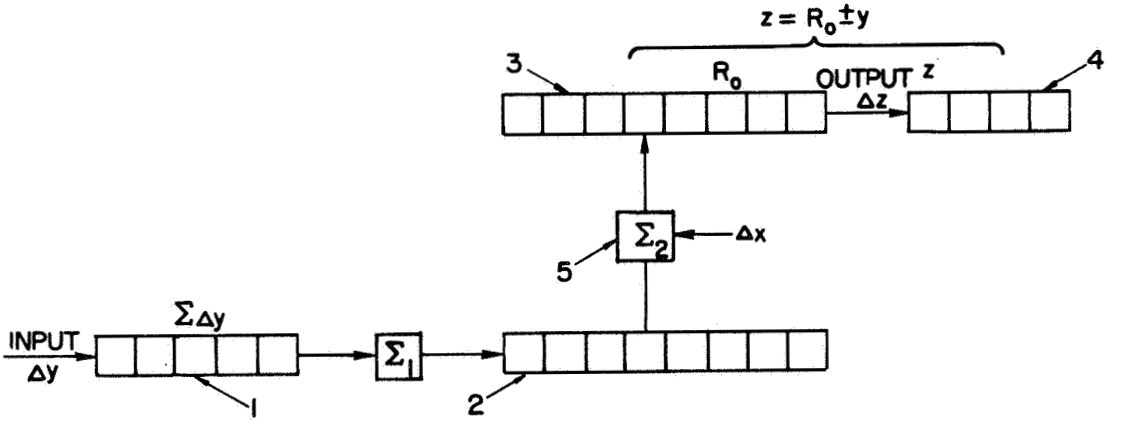


Fig. 5. BLOCK DIAGRAM OF DIGITAL INTEGRATING ELEMENTS WITH RECTANGULAR-INTEGRATION RULE.

For each integration step, $\Sigma\Delta y$ stored in counter 1 is summed with the number y_o stored in register 2 (y register) by means of the Σ_1 adder. As a result of addition (or subtraction), a new ordinate $y = y_o \pm \Sigma\Delta y$ is obtained for each step. During each step, y stored in register 2 is added to register 3 (R register), wherein the number corresponding to the sum of the ordinates (i.e., the value of the integral) is stored. Summation of y and R_o is achieved by the Σ_2 adder,

$$z = R_o \pm y$$

where R_o is the number initially stored in register 3. Summation is performed each time the input Δx of the integrator receives a pulse representing an increment of the independent variable x.

Let y_i be the content of the y register when the i^{th} Δx pulse occurs. If the R register has sufficient capacity, after the n^{th} Δx pulse, it will contain the sum

$$\sum_{i=1}^{n} y_i \Delta x_i$$

which is an approximation of

$$\int_0^{x_n} y \, dx$$

where

$$x_n = \sum_{i=1}^{n} \Delta x_i$$

However, if the R register has the same length (or less) as the y register and if register 4 (Δz register) is appended to the R register, the Δz register will be used to store the carriers generated in the R register. Hence, during summation, the R register may overflow, resulting in

$$\Delta z_i + R_i = R_{i-1} + y_i \Delta x_i \tag{2.15}$$

where R_i is the number in the R register after y_i has been added by the presence of the i^{th} Δx pulse. Therefore,

$$\sum_{i=1}^{n} \Delta z_i = \sum_{i=1}^{n} y_i \Delta x_i + R_o - R_n \tag{2.16}$$

and the sum of the Δz_i binary bits is an approximation to

$$\sum_{i=1}^{n} y_i \Delta x_i$$

with a round-off error $R_o - R_n$. Therefore, the combination of registers and adders in Fig. 5 is an approximate integrator. Registers 3 and 4 may be regarded as two parts of a single register having 2n bits; register 3 holds the less significant bits of integral z, and register 4 holds the most significant bits located at (n+1) to 2n-bit positions. In such an arrangement, z has, at the most, twice as many bits as the integrand register 2.

The integration process of accumulating the overflow pulses Δz from the integrator output by register 4 has been discussed. In general, the integral increment Δz is

$$\Delta z = ky\Delta x \tag{2.17}$$

where k is a constant scale factor. For binary numbers,

$$k = \frac{1}{2^n} \tag{2.18}$$

and n is the number of bits of register y or z, where R_o denotes the number initially in register 3.

The coefficient k is the scale factor of the digital integrator and signifies that, for $y = 1$ and $\Delta x = 1$, 2^n summations (or 2^n steps) are required to obtain one overflow pulse Δz at the output of register 3. If y is equal to 2^n and $\Delta x = 1$, there will be an overflow pulse for each of the integration steps.

Converting from increments Δx, Δy, and Δz to derivatives of x, y, and z in time, the formula $\Delta z = ky\Delta x$ can be rewritten as

$$\frac{dz}{dt} = ky\,\frac{dx}{dt} = k\left(\int \frac{dy}{dt}\,dt\right)\frac{dx}{dt} \tag{2.19}$$

or in the form of an integral

$$z = k\int y\,dx \tag{2.20}$$

2. Operation of the Digital Integrating Element with the Modified Trapezoidal Rule

The internal functional building block of a digital integrating element performing the modified trapezoidal rule is shown in Fig. 6. The following operations are performed in the presence of the Δx pulses:

$$\begin{aligned} \Sigma\Delta Y &\leftarrow \Delta Y_1 + \Delta Y_2 \\ Y &\leftarrow Y + \Sigma\Delta Y \\ R &\leftarrow R + Y + 3\ \Sigma\Delta Y/2 \end{aligned} \tag{2.21}$$

where Y and R are registers, and the arrow means "is replaced by." For example, the sum of the old content of the Y register and $\Sigma\Delta Y$ will be placed in the Y register as the new content.

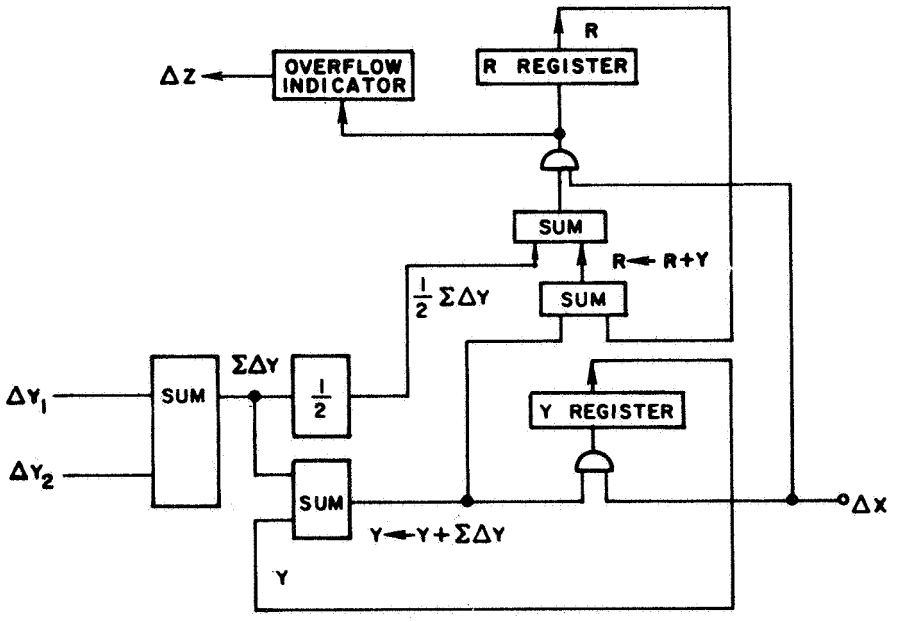


Fig. 6. FUNCTIONAL BUILDING BLOCK OF DIGITAL INTEGRATING ELEMENT PERFORMING EQ. (2.21).

In Fig. 6, the overflow indicator Δz can be expressed as

$$\begin{aligned} \Delta z &= +1 \quad \text{if} \quad R \quad \text{has overflow} \\ &= -1 \quad \text{if} \quad R \quad \text{has underflow} \\ &= \ \ 0 \quad \text{otherwise.} \end{aligned}$$

Chapter III

OPTIMAL DESIGN CONSIDERATIONS OF DIGITAL MODULES

Certain problems need to be considered during the optimization of digital modules:

1. By using digital modules as real-time building blocks for synthesizing network functions, speed of operation must be as high as possible. To achieve this high-speed operation, the digital integrator utilizes a fast integration algorithm (modified-trapezoidal method) and a special number system (signed-digit number system); both are operated in a parallel fashion.

2. With the specified speed of operation, the type and number of digital modules, for which a certain network function is to be constructed, are to be minimum.

3. Accuracy can be varied, and the network function realized should be better than the conventional method of realization. Hence, for applications in designing network functions, the internal register length of the digital integrator is assumed to have a single precision of 10^{-3}; i.e., the length of the register is equivalent to a three-decimal digit length. As will become clear later, for the case described, the precision is $(2 \times 666)^{-1}$ or 0.7575×10^{-3} rather than 10^{-3}.

A. The Signed-Digit Number System

In 1961, Avizienis [Refs. 9 and 10] introduced a class of number representations, called signed-digit representations, that limits the carry propagation to one position at the left during the operation of addition and subtraction in the arithmetic unit. Carry-propagation chains are eliminated by the use of redundant representations for the operands. In a conventional-number representation with an integer radix $r > 1$, each digit is allowed to assume exactly r values: 0,1,2, ..., r-1. In a redundant representation with the same radix r, each digit is allowed to assume more than r values; i.e., the allowed values q are

$$r + 2 \leq q \leq 2r - 1 \qquad (3.1)$$

For the application here, $r = 10$ and maximum redundancy is used; that is, a total of $2 \times 10 - 1 = 19$ numbers are allowed: 9, 8, 7, 6, 5, 4,

3, 2, 1, 0, $\bar{1}$, $\bar{2}$, $\bar{3}$, $\bar{4}$, $\bar{5}$, $\bar{6}$, $\bar{7}$, $\bar{8}$, $\bar{9}$, where the bar indicates the negative number. To satisfy the condition of totally parallel addition or subtraction [Ref. 9], the outgoing transfer digit and the interim sum digit, t_i and w_i, respectively, are restricted to be $\bar{4} \leq t_i \leq 4$ and $\bar{5} \leq w_i \leq 5$. Any addition can be done in two successive steps. If two operands are y_i and z_i, then

$$y_i + z_i = rt_{i-1} + w_i \tag{3.2}$$

and the sum digit is formed as

$$s_i = w_i + t_i \tag{3.3}$$

For example, the conventional radix-10 numbers 64, 72, 48, -39, 5, 9, and 279 are represented in signed-digit numbers as 64, 72, 48, $\overline{39}$, 5, 9, and 279. In performing addition or subtraction, these numbers are represented as $1\bar{4}4$, $1\bar{3}2$, $5\bar{2}$, $\bar{4}1$, 5, $1\bar{1}$, and $3\overline{2\bar{1}}$, respectively; for example, 64 = 100 - 40 + 4. An example is given to show the addition procedure.

Example 1.

Suppose the radix-10 signed-digit operands are:

augend z: $1.\bar{3}65\overline{1\bar{4}}$, the algebraic value Z = 0.76486

addend y: $0.\bar{4}053\bar{1}$, the algebraic value Y = -0.39471

The procedure of addition is as follows:

augend z:	1.	$\bar{3}$	6	5	$\bar{1}$	$\bar{4}$
addend y:	0.	$\bar{4}$	0	5	3	$\bar{1}$
step 1:	0+1.	$\overline{10}$+3	10+$\bar{4}$	10+0	0+2	0+$\bar{5}$
step 2:	$\bar{1}$.	1	1	0	0	
sum s:	0.	4	$\bar{3}$	0	2	$\bar{5}$

The sum is s: $0.4\bar{3}02\bar{5}$ and the algebraic value S = 0.37015.

Because radix-10 signed-digit numbers are used, each digit can be coded by 5-bit binary numbers. The code used here to identify the signed-digit numbers is:

9: 01001; 8: 01000; 7: 00111; 6: 00110; 5: 00101; 4: 00100;
3: 00011; 2: 00010; 1: 00001; 0: 00000; $\overline{1}$: 11111; $\overline{2}$: 11110;
$\overline{3}$: 11101; $\overline{4}$: 11100; $\overline{5}$: 11011; $\overline{6}$: 11010; $\overline{7}$: 11001; $\overline{8}$: 11000;
$\overline{9}$: 10111.

Note that 0 has a unique representation, as do all the others.

B. Minimum-Cost Criterion

The advances made by integrated-circuit technology, especially large-scale integration (LSI), make it possible to define a minimum cost for synthesizing a combinational switching function [Ref. 16]. Because the principal interest in this report is to synthesize network functions of variable precision by using digital elements, minimum cost can be defined as the cost incurred under the following three constraints:

1. The number of types of modules that make up the network function must be minimum.

2. The circuit chosen to realize any given network function must contain the smallest number of modules.

3. The number of input-output pins of a module must be minimum.

Variable precision means that the lengths of the Y and R registers can be varied; thus, the precision of the integration varies accordingly.

Subject to the first constraint, it is obvious that the minimum number of types of modules is one; i.e., one universal module best satisfies the first constraint. As noted earlier, the realization of network functions needs the operations of integration, addition, and multiplication; therefore, the universal module should contain digital integration and summing elements. The second constraint is also satisfied; however, the third may not be because more operations need more input-output pins.

Because multiplication by a constant requires one digital integrating element and because incremental multiplication can be performed by using two digital integrating elements and one digital summing element [Ref. 17], it is natural to break the block structures into two kinds of elements, namely, digital-integrating and summing. A comparison between the cost of using universal modules and the cost of using

two-element modules to realize network functions is made in Chapter IV.C. However, under certain assumptions, the two-element module is more economical, and, thus, it will be used here to realize all the networks.

C. Proposed Two-Element Module

1. Digital Variable-Precision Integrating-Element Module

The proposed digital integrating-element module with variable precision is shown in Fig. 7. The pin-number assignments are arbitrary; however, their functions are important. Referring to Fig. 7, this module has two sets of incremental ΔY inputs: pins 1 to 4 as a set that takes any incremental Δy between 7 to $\overline{7}$, and pins 5 and 6 as a set that takes Δy_{ti} between 1 and $\overline{1}$. Pins 7 and 8 are assigned to the Δx input, where

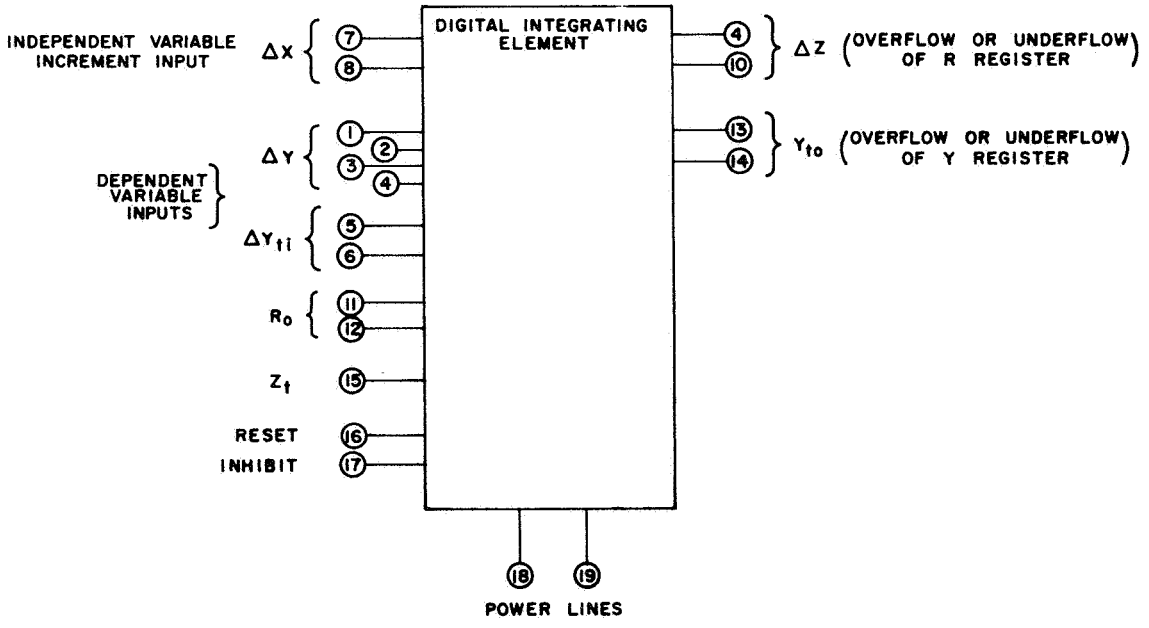


Fig. 7. DIGITAL VARIABLE-PRECISION INTEGRATING ELEMENT.

Δx is between 1 and $\overline{1}$. If the independent variable is time, pin 8 will be used as the clock input, while pin 7 is held at zero. Pins 9 and 10 are assigned to Δz, where $\Delta z = 01$ (+1) or $\Delta z = 11$ (-1), corresponding to overflow or underflow, respectively. The overflow or underflow of the Y register will be taken from pins 13 and 14. The reset, the inhibit, and the power-supply lines are assigned to pins 16, 17, 18, and 19. Pins 11, 12, and 13 will be held at zero if single precision is used. For more than double precision, pin 15 will be held at one level, while pins 11 and 12 will be connected to pins 9 and 10 of the previous stage of the cascading integrating element. In addition, pins 13 and 14 of the previous stage must be connected to pins 5 and 6 of the current integrating element. For cascade connection, refer to Fig. 8, where a double-precision cascade connection is shown.

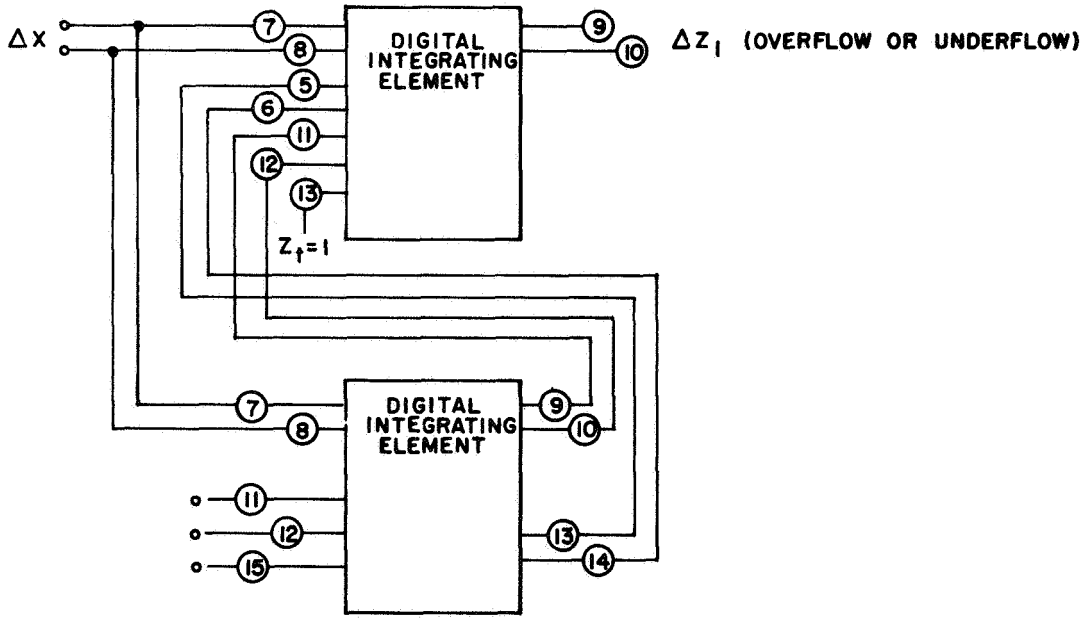


Fig. 8. CASCADE CONNECTION OF DIGITAL INTEGRATING ELEMENT FOR PURPOSE OF DOUBLE PRECISION.

The internal building structure and the complete logical design of the digital variable-precision integrating element are shown in Appendix B, where all the logics are combinational networks, except the Y and R registers. For fast operating speed, logic can be done by two level

AND-OR gates to minimize the propagation delay; however, the number of gates can be reduced significantly by increasing the levels of the AND-OR gates.

The overflow and underflow problem of the unconventional signed-digit number system is not as simple as the conventional one. In the conventional number system (for example, either the binary or decimal system), the overflow or underflow can be determined by the very leftmost digit and the sign digit. In other words, the overflow or underflow can be determined by the nonzero value of the digit-one position at the left of the most significant digit of the R register. For the case of three-decimal digits in the R register, if R has a value of +1 x x x or -1 x x x, not only an overflow or underflow occurs but also the remainder will be automatically left over in those x x x positions. However, in the unconventional signed-digit redundancy number system, the overflow or underflow is a function of all digits of the R register, and thus overflow or underflow can be detected, as is shown in Appendix B.

The remainder problem can also be solved by adding $\overline{1}334$ (decimal number -666) or $1\overline{334}$ (decimal number +666) to the R register if overflow or underflow occurs, respectively. This has been resolved in the integrator design. For example, at time t_{n-1}, the R register stored $1\overline{337}$ (decimal number 663) and no overflow or underflow occurred. Now, suppose at time t_n, number 5 is added into R and makes $R = 1\overline{332}$, which would result in an overflow (in decimal number $663 + 5 = 668 > 666$). As soon as the overflow indicator is turned on, number $\overline{1}334$ will be added to the contents of R, and the remainder $1\overline{332} + \overline{1}334 = 0002$ results.

2. Digital Summing-Element Module

The digital summing element is shown in Fig. 9 where, according to control (see Table 1), addition, subtraction, or multiplication is performed. Three sets of inputs, Δy_1, Δy_2, and Δy_3 are assigned to pins 1 to 3, 4 and 5, and 6 to 8, where $\overline{3} \leq \Delta y_1 \leq 3$, $\overline{1} \leq \Delta y_2 \leq 1$, and $\overline{3} \leq \Delta y_3 \leq 3$. Control signals c_1 and c_2 are assigned to pins 13 and 14. Their function is to control the sum S, such that

$$S = f(\Delta y_1, \Delta y_2, \Delta y_3, c_1, c_2)$$

$$= (c_1 \wedge c_2)(\Delta y_1 + \Delta y_2 + \Delta y_3) \vee (c_1 \wedge \bar{c}_2)(\Delta y_1 + \Delta y_2 - \Delta y_3)$$

$$\vee(\bar{c}_1 \wedge c_2)(\Delta y_1 - \Delta y_2 + \Delta y_3) \vee (\bar{c}_1 \wedge \bar{c}_2)(-\Delta y_1 - \Delta y_2 - \Delta y_3) \quad (3.4)$$

where

$\vee$ = logic OR

$\wedge$ = logic AND

thus, the range of S is $\bar{7} \leq S \leq 7$. The logic design of the digital summing element is shown in Appendix C, where only combinational logics are employed.

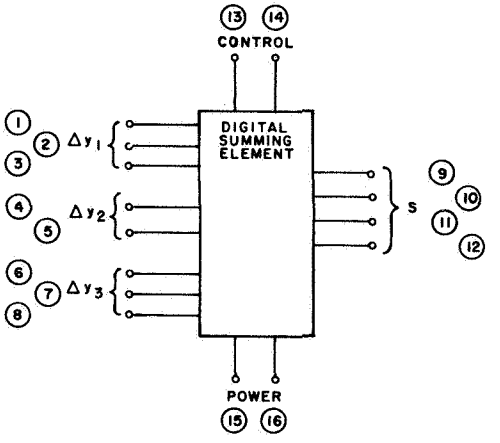


Fig. 9. DIGITAL-SUMMING ELEMENT.

Combinational logics are employed in the design of both the digital integrating and summing elements; therefore, a large number of AND-OR gates are used. However, with the advancement of large-scale-integration techniques, it has been predicted that to cram a thousand gates in one chip [Refs. 18, 19, and 20] will be no problem before 1970. The cost of the LSI module depends on the internal repeatability of the subcells. Hence implementation of the two types of modules utilizes the high standardization of the subcell. This is described in Appendixes B and C, where, without much simplification of the logic, the number of gates is roughly 900 and 200 for digital-integrating and summing elements, respectively. As was noted earlier, this number is well within the predicted limitation.

Table 1

CONTROL ASSIGNMENT OF SUMMING ELEMENT

$c_1 c_2$	11	10	01	00
S	$\Delta y_1 + \Delta y_2 + \Delta y_3$	$\Delta y_1 + \Delta y_2 - \Delta y_3$	$\Delta y_1 - \Delta y_2 + \Delta y_3$	$-\Delta y_1 - \Delta y_2 - \Delta y_3$

Chapter IV

REALIZATION OF TRANSFER FUNCTIONS BY USING DIGITAL-INTEGRATING AND SUMMING ELEMENTS

A. Solutions of Differential Equations Using Two-Element Modules

Functionally, a digital integrator is represented by a schematic, as shown in Fig. 10, where arrows indicate the direction of data flow. The inputs dx and dy are incremental inputs, and dx can be either an incremental time input dt (clock pulses) or a function of y. The output dz observes the relations

$$dz = ydx \tag{4.1a}$$

or after summing up the dz,

$$z = \int ydx \tag{4.1b}$$

Sometimes, more than one digital integrator is used to solve a certain problem, in which case they can be connected in such a way that the overflow of one integrator is connected to the input of the other. From time to time, scalar multiplication is required; in that event, an integrator also can be used. If the dy input terminal is left open and the content of the y register is set to a desired constant k, then dz = kdx; that is, the output dz is equal to k multiplied by the dx input (see Fig. 11).

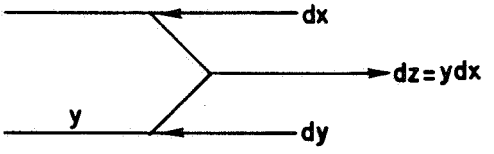


Fig. 10. FUNCTIONAL SCHEMATIC OF A DIGITAL INTEGRATOR.

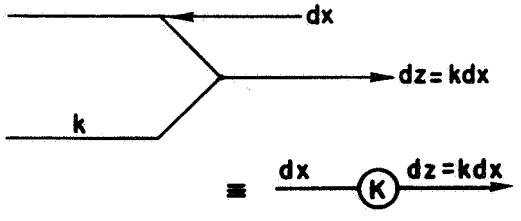


Fig. 11. SCHEMATIC OF A CONSTANT MULTIPLIER.

A set of digital integrators can be used to solve an ordinary differential equation of any order or degree, linear or nonlinear, or even a simultaneous set of such equations. Normally, for solving a differential equation, two steps (mapping and scaling) are involved.

Mapping specifies how the operational units (integrators and adders) should be interconnected so that the variable or variables of interest are generated within the system. Because a digital integrator has a limited capacity of registers, it is necessary to ensure that intermediate results stay within the specified ranges during the running of a problem, so that the estimated maximum values of each of the variables can be scaled to a meaningful range. This is amplitude scaling. Sometimes, frequency scaling is also employed to ensure proper operation. For a real-time device, the amplitude and frequency of the input must be specified in a workable range; therefore, no frequency scaling is permissible. However, if amplitude scaling is necessary, it can be done either by adjusting the ratio of the analog-to-digital or the digital-to analog converters or by using a multiplier to restore the scale factor.

Example 2.

Solve the following differential equation:

$$\frac{d^2y}{dt^2} - \frac{dy}{dt} - \sin y = 0$$

Solution. Because the digital integrator only deals with digital increments, differentiating the given differential equation once yields

$$d\ddot{y} = d\dot{y} + d(\sin y)$$

The solution y can be obtained by interconnecting the digital integrators, as shown in Fig. 12, where the independent variable input is dt. The initial conditions of $y(0)$, $\dot{y}(0)$, and $\ddot{y}(0)$ have not been considered; however, they can be treated by adding one extra register (the I register) to store the initial condition for each integrator. The data transfer from I register to y register will be done at the beginning of the operation cycle. Similarly, nonlinear differential equations can be solved without difficulty. For simplicity, a circle indicates the multiplication by a constant, with the desired constant noted.

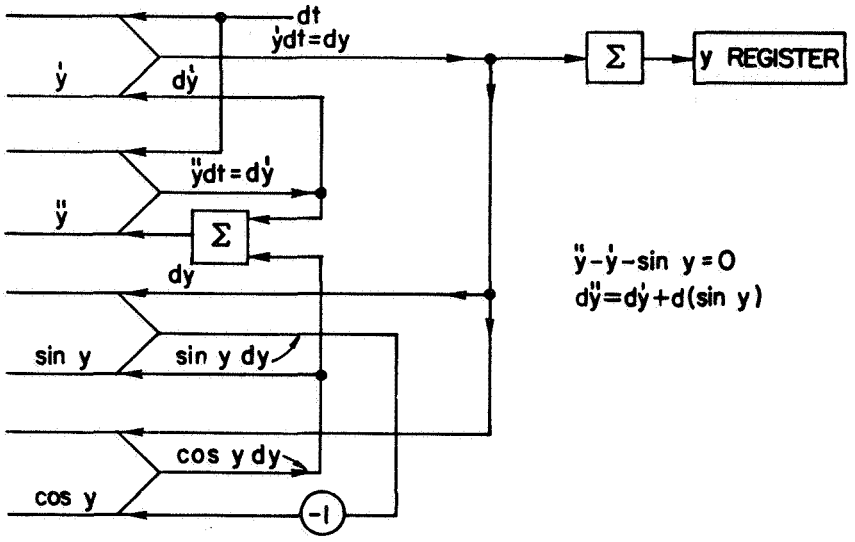


Fig. 12. CONNECTIONS FOR SOLUTION OF $\ddot{y} - \dot{y} - \sin y = 0$.

Example 3.

Solve the following nonlinear differential equation:

$$\ddot{y}(t) - y(t)\,\dot{y}(t) + y(t)^2 = 0$$

Solution. Differentiating the given equation once, yields

$$d\ddot{y} = \dot{y}dy + yd\dot{y} - 2ydy$$

The solution can be obtained by interconnecting digital integrators, as those shown in Fig. 13.

B. Synthesis Methods for Transfer-Function Realization

The transfer function is defined as the ratio of the Laplace transform of the output quantity to the Laplace transform of the input, with the restriction that the initial conditions appearing in the transformed differential equation (or equations) are all zero [Ref. 21].
Let

$$\text{Transfer function} = G(s) = \frac{\mathcal{L}[y(t)]}{\mathcal{L}[x(t)]} = \frac{Y(s)}{X(s)} \qquad (4.2)$$

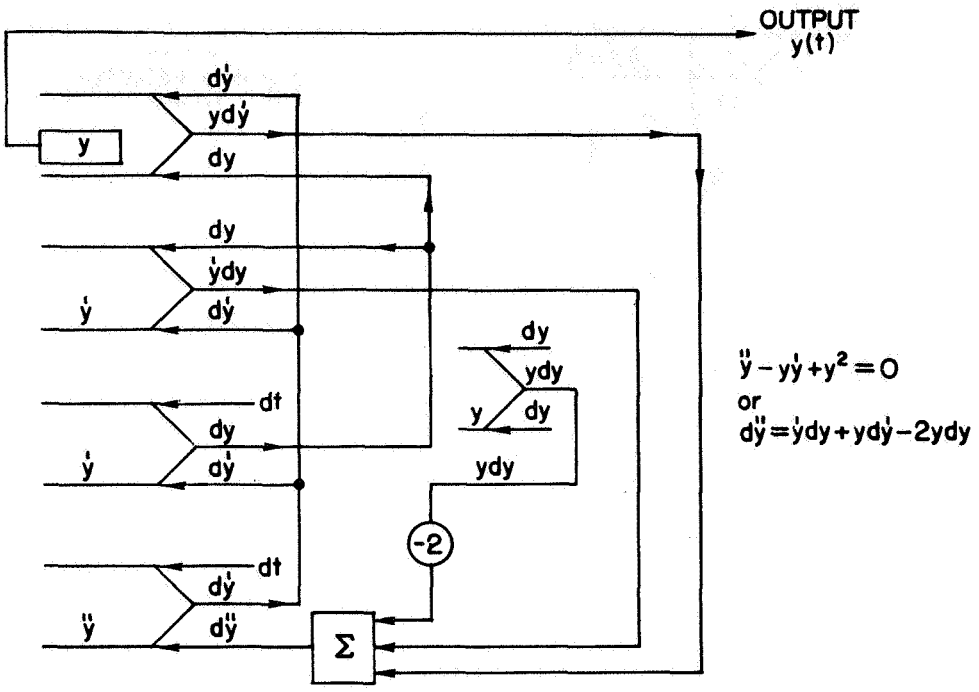


Fig. 13. CONNECTIONS FOR SOLUTION OF $\ddot{y} - y\dot{y} + y^2 = 0$.

where $y(t)$ and $x(t)$ are the output and input of a system, respectively. Suppose there is a black box with the transfer function $G(s)$; the purpose is to realize the black box with digital elements. The input and output quantities can be either electrical voltage or current; hence, the transfer function can be voltage-to-voltage or voltage-to-current, or vice versa. Generally, the transfer functions to be discussed are those whose degree of the numerator is equal to or less than the degree of the denominator, such that $G(s)\big|_{s\to\infty}$ is finite [Ref. 11].

Before the different methods of synthesis procedures are presented, a simple example will illustrate how the realization of the transfer function can be done by using digital integrating and summing elements.

Example 4.

Given a simple transfer function,

$$G(s) = \frac{1}{s+1} = \frac{Y(s)}{X(s)}$$

where s is the complex variable, realize G(s) with only digital elements, and show the steps of realization in detail.

Solution. First, transforming the given function back to the time domain, yields

$$\dot{y}(t) + y(t) = x(t)$$

or

$$d\dot{y} = dx - dy$$

The network having the above characteristics can be realized by interconnecting the digital building blocks, as shown in Fig. 14, where dx and dy are the digital input and output, respectively, and ADIC and DAC are not shown in connection with x(t) and y(t).

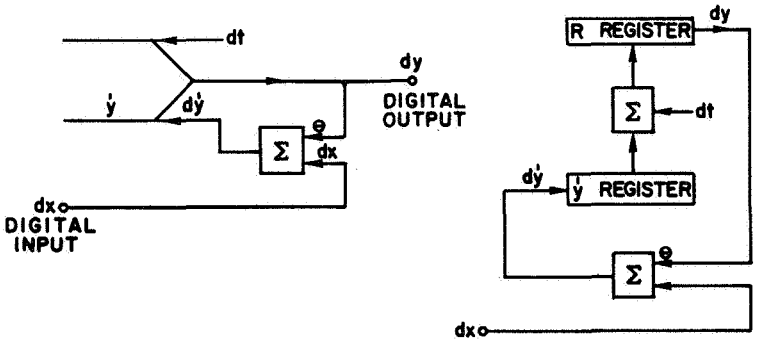


Fig. 14. REALIZATION OF TRANSFER FUNCTION 1/(s + 1).

For convenience, this network contains two registers of 8 bits each (see Fig. 15). The 8 whole-number bits have a capacity of $2^8 = 256$ maximum. The input dx can be obtained as the difference between the two consecutive samples of the input; that is, dx can be obtained by letting

$$dx_i = x(t_i) - x(t_{i-1})$$

During the presence of the dt pulse, the duration of dt can be roughly divided into three subtime intervals, T_1, T_2, and T_3. In T_1, the summation of dx and -dy is performed, where -dy comes from the

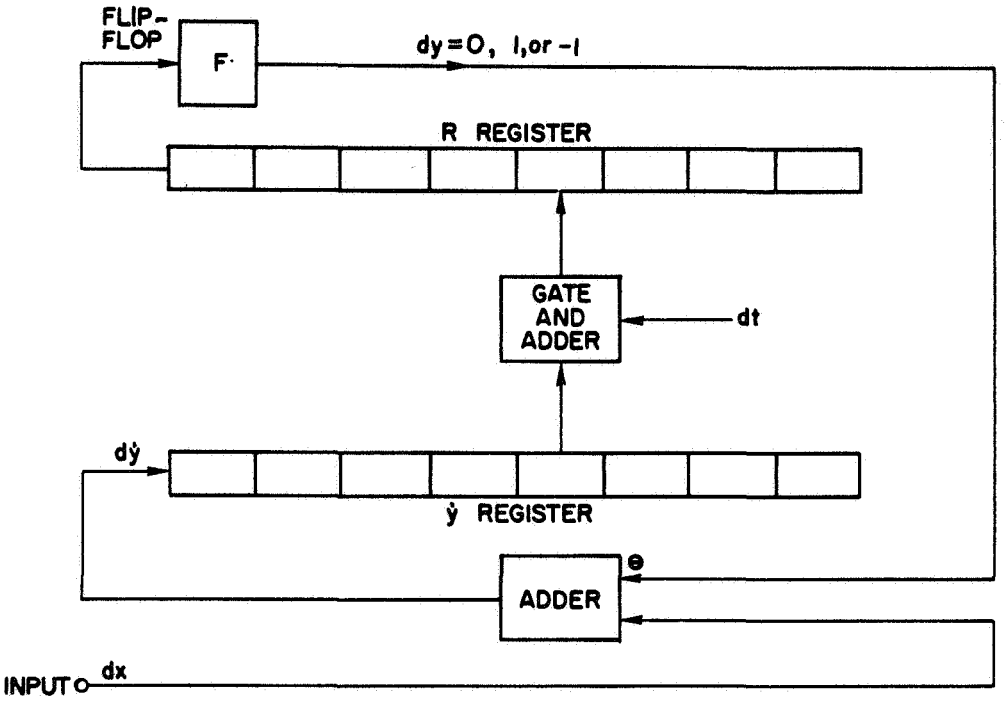


Fig. 15. TRANSFER FUNCTION $1/(s + 1)$ WITH 8-BIT REGISTERS.

flip-flop F, which contains the overflow of the R register. Within T_2, the sum of (dx-dy) plus the contents of the $\dot{y}$ register will again be placed in the y register. Meantime, the flip-flop F will be reset. In T_3, the contents of the $\dot{y}$ register are added to the contents of the R register, and the result is placed in the R register. If the R register has no overflow, the F output is zero, i.e., $dy = 0$. If there is an overflow, the output of F is one. In other words, $dy = 1$ or $dy = -1$; the choice depends on the sign of the R register. For the case in Example 4, $dy = -1$ if $R \leq -256$ and $dy = 1$ if $R \geq 256$.

The structure of the digital integrator (connected either in series or in parallel) is much simpler for the series-operated digital integrator, but the operation speed is slower.

The simulation of the problem in Example 4 was solved on the digital computer. The computer program of the sinusoidal and unit-step responses of the network are presented in Appendixes D and E. The step-by-step calculations are tabulated in Tables 2 and 3, and the graphical responses are shown in Figs. 16 and 17, where the method of rectangular integration with the ternary code is assumed.

Table 2

SINUSOIDAL RESPONSE OF $G(s) = 1/(s+1)$ RESULTS BY DIGITAL-ELEMENTS REALIZATION

Input: $x(t) = 255 \sin t$

Let $t = T/256$, $x_i(t) = 255 \sin (T/256\ i)$, $dx = x_i(t) - XREG_{i-1}$

$XREG_i = XREG_{i-1} + dx$ $\qquad$ $\dot{Y}REG_i = \dot{Y}REG_{i-1} + d\dot{Y}$

$d\dot{Y} = dx - dY$ $\qquad$ $RREG_i = RREG_{i-1} + \dot{Y}REG$

$dY = 1$ while RREG has overflow, $dY = -1$ while RREG has underflow, otherwise $dY = 0$

$YREG_i = YREG_{i-1} + dY$

Assume that all registers have 8-bit length, and that the method of triangular integration with trinary code is used.

T	XREG	$\dot{Y}$REG	RREG	dY	YREG	Real Solution y(t)
0	0	0	0	0	0	0
1	0	0	0	0	0	0.0019
2	1	1	1	0	0	0.0077
3	2	2	3	0	0	0.0175
4	3	3	6	0	0	0.0310
5	4	4	10	0	0	0.0485
6	5	5	15	0	0	0.0697
7	6	6	21	0	0	0.0948
8	7	7	28	0	0	0.1236
9	8	8	36	0	0	0.1563
10	9	9	45	0	0	0.1927
11	10	10	55	0	0	0.2329
12	11	11	66	0	0	0.2768
13	12	12	78	0	0	0.3244
14	13	13	91	0	0	0.3758
15	14	14	105	0	0	0.4308
16	15	15	120	0	0	0.4895
17	16	16	136	0	0	0.5519
18	17	17	153	0	0	0.6179
19	18	18	171	0	0	0.6876
20	19	19	190	0	0	0.7609
21	20	20	210	0	0	0.8377
22	21	21	231	0	0	0.9182
23	22	22	253	0	0	1.0022
24	23	23	20	1	1	1.0898
25	24	23	43	0	1	1.1809
26	25	24	61	0	1	1.2756
27	26	25	92	0	1	1.3737
28	27	26	118	0	1	1.4754
32	31	30	232	0	1	1.9166
33	32	31	7	1	2	2.0355
41	40	38	/ 27	1	3	3.1079
64	63	56	95	0	7	7.3334
96	93	78	209	0	15	15.7509
128	122	96	192	0	26	26.6718
160	149	110	175	0	39	39.6026
192	173	120	33	1	54	54.0565
224	195	126	153	0	69	69.5563
256	214	129	159	0	85	85.6381
288	230	129	203	0	101	101.8551
320	241	124	171	0	117	117.7813
352	250	118	216	0	132	133.0156
384	254	108	244	0	146	147.1856
416	254	95	157	0	159	159.9513
448	251	81	178	0	170	171.0087
480	244	65	210	0	179	180.0927
512	232	46	188	0	186	186.9797
576	199	6	71	0	193	193.4906
640	153	-37	-172	0	190	189.6576
704	98	-78	-292	0	176	175.3460
768	36	-116	-115	-1	151	151.1551
800	5	-131	-222	0	136	135.7300
832	-27	-146	-59	-1	118	118.3627
896	-89	-168	-194	0	79	78.8314
960	-145	-181	-163	-1	35	34.8820
992	-170	-183	-104	-1	12	12.0636
1000	-176	-183	-33	-1	7	6.3290
1024					-11	
1152					-97	
1280					-158	
1408					-180	

Table 3

UNIT-STEP RESPONSE OF $G(s) = 1/(s + 1)$ RESULTS BY DIGITAL-ELEMENTS REALIZATION

T	XREG	$\dot{Y}$REG	RREG	dY	YREG	Real Solution y(t)
0	0	0	0	0	0	0
1	255	255	255	0	0	0.9980
2	255	255	253	1	1	1.9922
3	255	254	251	1	2	2.9824
4	255	253	248	1	3	3.9689
5	255	252	244	1	4	4.9514
6	255	251	239	1	5	5.9302
7	255	250	233	1	6	6.9051
8	255	249	226	1	7	7.8762
9	255	248	218	1	8	8.8436
10	255	247	209	1	9	9.8072
11	255	246	199	1	10	10.7670
12	255	245	188	1	11	11.7230
13	255	244	176	1	12	12.6754
14	255	243	163	1	13	13.6240
15	255	242	149	1	14	14.5690
16	255	241	134	1	15	15.5102
17	255	240	118	1	16	16.4478
18	255	239	101	1	17	17.3817
19	255	238	83	1	18	18.3120
20	255	237	64	1	19	19.2387
21	255	236	44	1	20	20.1617
22	255	235	23	1	21	21.0811
23	255	234	1	1	22	21.9970
24	255	233	234	0	22	22.9093
25	255	233	211	1	23	23.8180
26	255	232	187	1	24	24.7232
27	255	231	162	1	25	25.6249
28	255	230	136	1	26	26.5230
29	255	229	109	1	27	27.4177
30	255	228	81	1	28	28.3088
31	255	227	52	1	29	29.1965
32	255	226	22	1	30	30.0807
33	255	225	247	0	30	30.9615
64	255				56	56.6269
96	255				80	80.0539
128	255				100	100.7281
160	255				118	118.9730
192	255				134	135.0741
224	255				149	149.2833
256	255				161	161.8228
288	255				172	172.8889
320	255				182	182.6547
352	255				191	191.2730
384	255				198	198.8786
416	255				205	205.5906
448	255				211	211.5138
480	255				216	216.7411
512	255				221	221.3541
576	255				228	229.0177
640	255				234	234.9862
704	255				239	239.6344
768	255				242	243.2545
800	255				244	244.7521
832	255				245	246.0738
896	255				247	248.2694
960	255				249	249.9794
992	255				250	250.6868

On the basis of the results of the unit-step and sinusoidal responses, a comparison between digital and analog realizations can be made. From the last two columns of Tables 2 and 3, it is seen that the contents of YREG (digital) are a good approximation to the real solution (analog). Note that the figures tabulated have been scaled 256 times. At most,

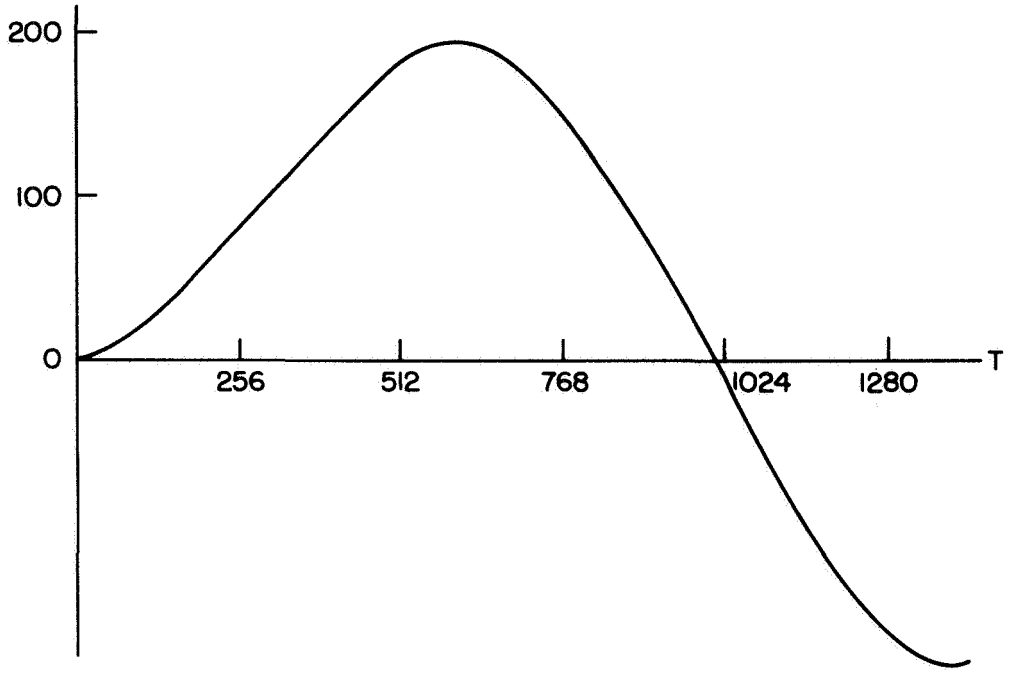


Fig. 16. SINUSOIDAL RESPONSE OF G(s) = 1/(s + 1) WITH INPUT x(t) = 255 sin (T/256).

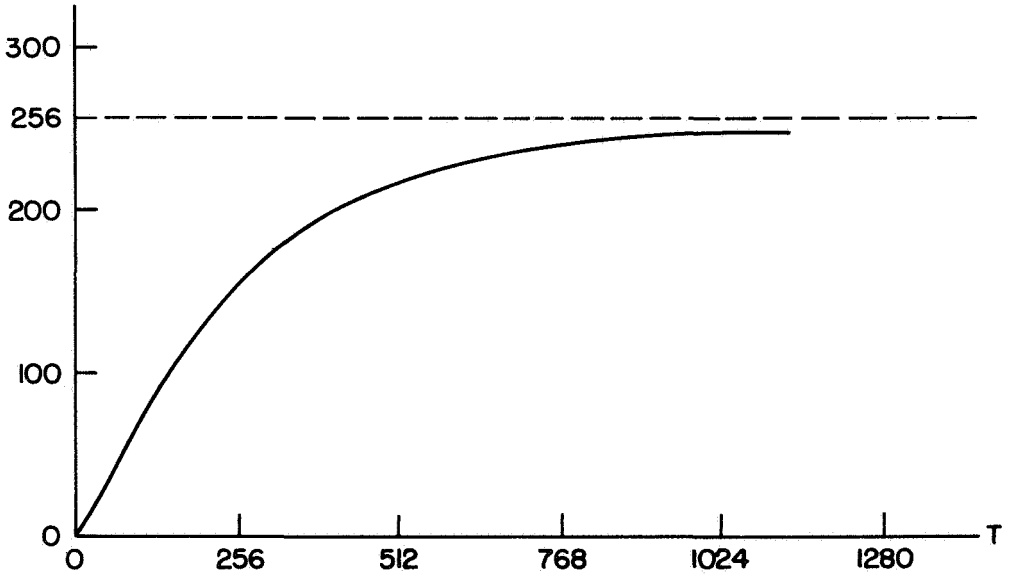


Fig. 17. UNIT-STEP RESPONSE OF G(s) = 1/(s + 1) WITH INPUT x(t) = 255.

they differ by one per 256 for registers of 8-bit lengths. If registers with more than 8 bits are used, accuracy will be proportionally increased.

The closeness of the two solutions (digital and analog) is shown in Fig. 18. If trapezoidal integration is employed in the design integrators, the accuracy of the digital solution will be further improved. It is noteworthy that even if the input is not started from zero at $t = 0$, as was assumed in Tables 2 and 3, the digital realization is still valid.

For better approximation, more register bits can be used. In the presence of fractional numbers, as well as whole numbers, a few fractional-number bits can be attached to the end of the whole numbers and a fixed decimal point can be assigned. For example, to maintain an accuracy of 10^{-3}, 10 bits, corresponding to the fractional number, will be used in addition to the whole-number bits because $2^{-10} = 1/1024 \approx 10^{-3}$; this illustrates how the given transfer function can be realized. For the realization of a complicated transfer function, more digital integrators could be utilized.

References 1, 2, 3, and 5 discuss digital-computer simulation of sampled-data systems; however, the realization of network functions by using digital elements can also be accomplished by direct, parallel, and cascade methods [Ref. 13]. Realization does not take the form of a computer program or a set of digital delay elements but is an interconnection of digital integrating elements and summing elements such as those differential-equation-solving problems encountered in the digital differential analyzer [Refs. 13 and 17]. The following example illustrates the realization procedures.

Example 5.

Suppose $dx(t)$ and $dy(t)$ are the input and output increments, respectively. Realize the transfer function

$$G(s) = \frac{k}{s^3 + fs^2 + gs + h} = \frac{Y(s)}{X(s)} \qquad (4.3)$$

by using digital elements, where $Y(s) = \mathcal{L}[y(t)]$ and $X(s) = \mathcal{L}[x(t)]$ are Laplace transforms of $y(t)$ and $x(t)$, respectively.

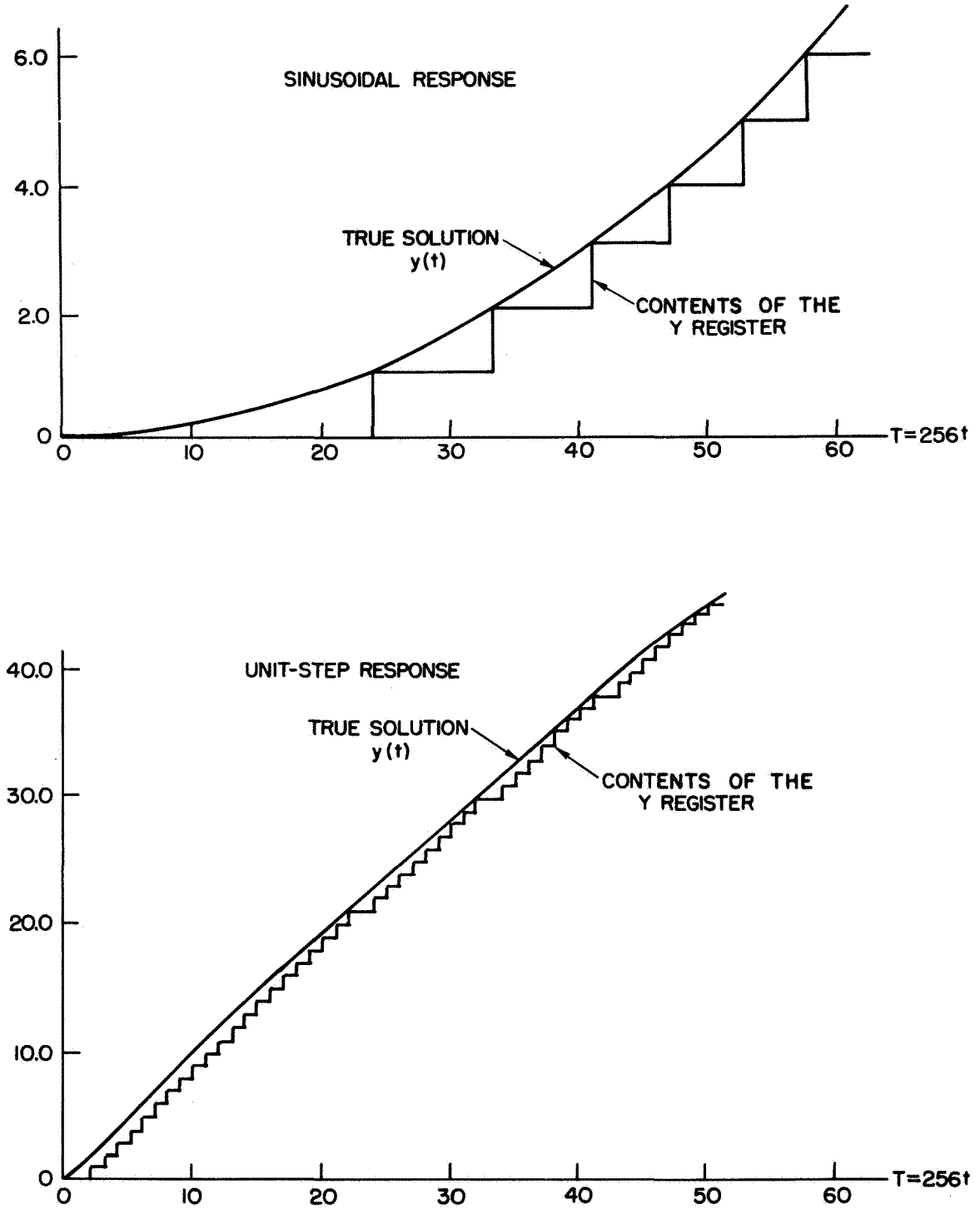


Fig. 18. COMPARISON BETWEEN THE TWO RESPONSES OF ANALOG AND DIGITAL REALIZATIONS.

Solution. The synthesis steps are more or less the same for the three methods.

1. Direct Method

(a) Crossmultiply Eq. (4.3) to obtain

$$(s^3 + fs^2 + gs + h)\ Y(s) = kX(s) \tag{4.4}$$

(b) Transform the given transfer function into a time-domain differential equation, with zero initial conditions:

$$\dddot{y}(t) + f\ddot{y}(t) + g\dot{y}(t) + hy(t) = kx(t) \tag{4.5}$$

(c) Differentiate Eq. (4.5) once:

$$d\dddot{y} + fd\ddot{y} + gd\dot{y} + hdy = kdx \tag{4.6}$$

(d) Rewrite Eq. (4.6) as

$$d\dddot{y} = kdx - fd\ddot{y} - gd\dot{y} - hdy \tag{4.7}$$

(e) Draw a solution diagram, as shown in Fig. 19, where the multiplication of a constant k is shown with a circle enclosing it.

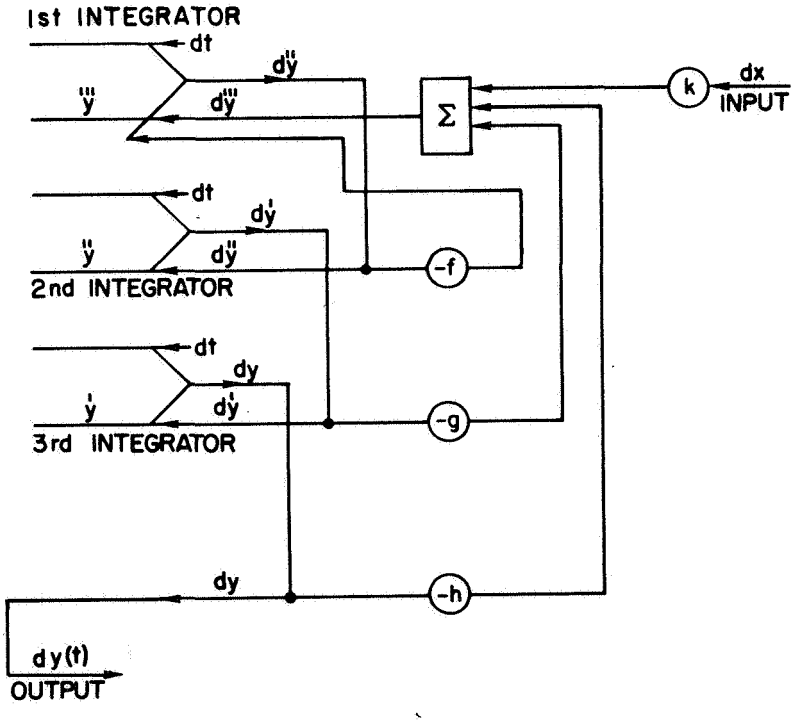


Fig. 19. DIRECT-METHOD REALIZATION OF THE TRANSFER FUNCTION $G(s) = k/(s^3 + f\,s^2 + g\,s + h)$.

2. Parallel Method

(a) Expand G(s) of Eq. (4.3) into partial-fraction form:

$$G(s) = \frac{k}{(s + a)(s^2 + bs + c)} = \frac{\ell}{s + a} + \frac{rs + q}{s^2 + bs + c}$$

$$= G_a(s) + G_b(s) \tag{4.8}$$

where

$$G_a(s) = \frac{\ell}{s + a} = \frac{Y_1(s)}{X(s)} \tag{4.9a}$$

$$G_b(s) = \frac{rs + q}{s^2 + bs + c} = \frac{Y_2(s)}{X(s)} \tag{4.9b}$$

$$Y_1(s) + Y_2(s) = Y(s) \tag{4.9c}$$

and a, b, c, k, ℓ, r, and q are real constants.

(b) Transform $Y_1(s)$ and $Y_2(s)$ back to the time domain:

$$\dot{y}_1(t) + ay_1(t) = \ell x(s) \tag{4.10a}$$

$$\ddot{y}_2(t) + b\dot{y}_2(t) + cy_2(t) = r\dot{x}(t) + qx(t) \tag{4.10b}$$

$$d\dot{y}_1 = \ell dx - ady_1 \tag{4.10c}$$

and

$$\dot{y}_2(t) + by_2(t) + c\int y_2(t)\,dt = rx(t) + q\int x(t)\,dt \tag{4.10d}$$

(c) Then, differentiate once,

$$d\dot{y}_2 = -bdy_2 + rdx + (qx\,dt - cy_2\,dt) \tag{4.11}$$

(d) Draw a solution diagram, as shown in Fig. 20, for $dy_1(t)$, $dy_2(t)$, and

$$dy(t) = dy_1(t) + dy_2(t) \tag{4.12}$$

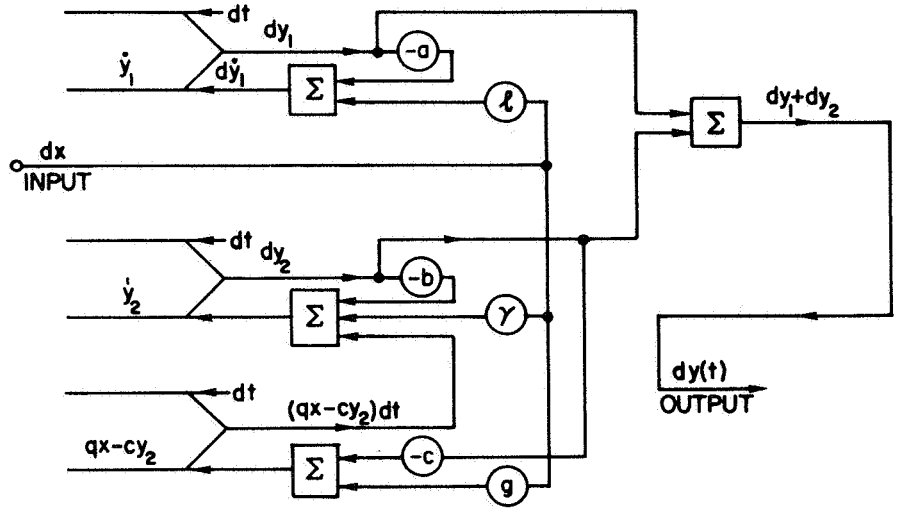


Fig. 20. PARALLEL-METHOD REALIZATION OF $G(s) = k/(s^3 + f s^2 + g s + h)$.

3. Cascade Method

(a) Write $G(s)$ into the product of two transfer functions:

$$G(s) = G_1(s)\, G_2(s) = \frac{Y(s)}{Y_2(s)} \frac{Y_2(s)}{X(s)}$$

$$= \frac{1}{s + a} \frac{k}{s^2 + bs + c} \tag{4.13}$$

(b) Transform $G_1(s)$ and $G_2(s)$ into time-domain differenital equations and differentiate once:

$$d\dot{y}(t) = dy_2(t) - ady(t)$$

$$d\ddot{y}_2(t) = kdx(t) - bd\dot{y}_2(t) - cdy_2(t) \tag{4.14}$$

(c) Draw a solution diagram, as shown in Fig. 21.

4. Performance Error and Cost Comparisons

By observing Figs. 19, 20, and 21, it is seen that the most economical way to realize a third-order transfer function is by either the direct or cascade method, where only six digital integrating elements (three as integrators and three as constant multipliers) and one digital summing elements are used. The question is: Which of the two synthesis

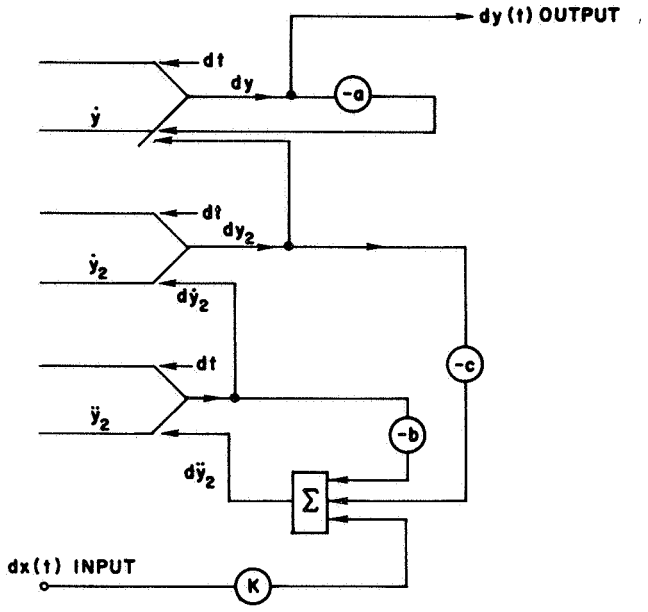


Fig. 21. CASCADE-METHOD REALIZATION OF THE TRANSFER FUNCTION $G(s) = k/(s^3 + f s^2 + g s + h)$.

methods utilizes a minimum number of modules? Consider a general third-order transfer function, which is more complicated than Eq. (4.3):

$$G(s) = \frac{ks^2 + ms + n}{s^3 + fs^2 + gs + h} = \frac{Y(s)}{X(s)} \tag{4.15}$$

The realizations of $G(s)$ by both direct and cascade methods are given below.

(a) Cascade Method

Equation (4.15) can be factored into

$$G(s) = \frac{1}{s + a} \frac{ks^2 + ms + n}{s^2 + bs + c} = \frac{1}{s + a} \left[k + \frac{rs + q}{s^2 + bs + c} \right]$$

$$= \frac{Y(s)}{Y_2(s)} \frac{Y_2(s)}{X(s)} = \frac{Y(s)}{Y_2(s)} \left[k + \frac{Y_3(s)}{X(s)} \right] \tag{4.16}$$

Transform $Y(s)/Y_2(s) = 1/(s + a)$ into a time-domain differential equation:

$$\dot{dy}(t) = dy_2(t) - ady(t) \tag{4.17}$$

Crossmultiply $Y_3(s)/X(s) = (rs + q)/(s^2 + bs + c)$ and divide both sides by s to obtain

$$\left(s + b + \frac{c}{s}\right)Y_3(s) = \left(r + \frac{q}{s}\right)X(s) \tag{4.18}$$

Transform Eq. (4.18) into the time domain:

$$dy_3(t) = - bdy_3(t) + rdx(t) - cy_3(t)\,dt + qx(t)\,dt \tag{4.19}$$

and

$$dy_2(t) = kdx(t) + dy_3(t) \tag{4.20}$$

Then draw a solution diagram, as shown in Fig. 22. Note that in Fig. 22 there are nine digital integrating and two summing elements, and six of the nine integrating elements serve as constant multipliers.

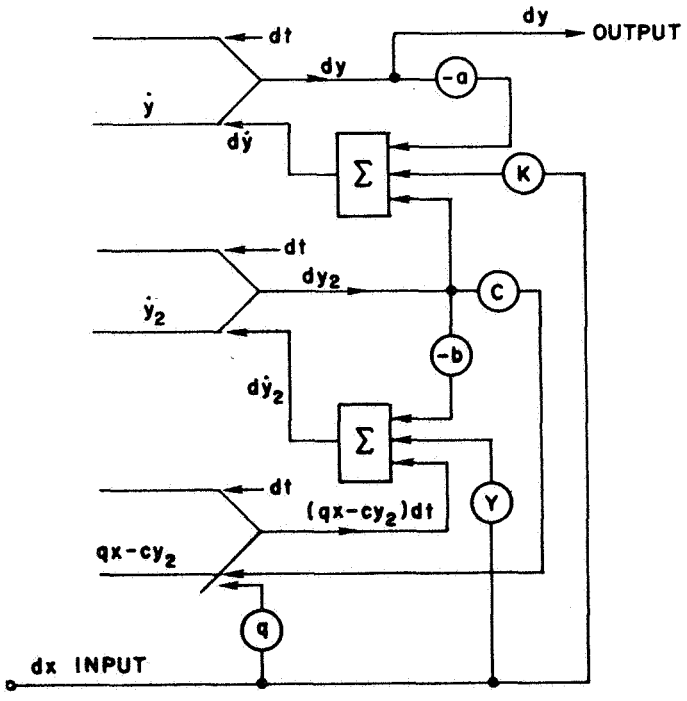


Fig. 22. CASCADE REALIZATION OF EQ. (4.15).

(b) Direct Method

Crossmultiply Eq. (4.15) and divide both sides by s^2:

$$\left(s + f + \frac{g}{s} + \frac{h}{s^2}\right) Y(s) = \left(k + \frac{m}{s} + \frac{n}{s^2}\right) X(s) \qquad (4.21)$$

Transform Eq. (4.21) into a time-domain differential equation:

$$d\dot{y}(t) = d[kx(t) - fy(t)] + [mx(t) - gy(t)]\, dt$$

$$+ \left\{\int [nx(t) - hy(t)]\, dt\right\} dt \qquad (4.22)$$

One of the direct-method realizations of Eq. (4.15) is shown in Fig. 23, where 10 digital integrating elements (six serving as constant multipliers) and two summing elements are utilized. Because this realization is not unique, other configurations can be obtained also; for example, the term $[mx(t) - gy(t)]\, dt$ in Fig. 23 can also be formed by first finding $mx(t)\, dt$ and $-gy(t)\, dt$ separately and then summing

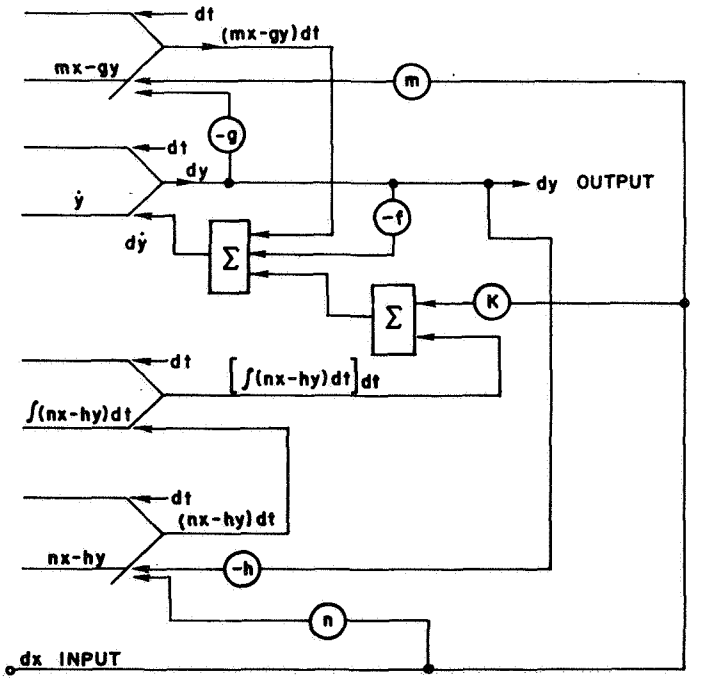


Fig. 23. DIRECT REALIZATION OF EQ. (4.15).

them together. Other configurations are found, requiring more integrators and summing elements.

Comparing Figs. 22 and 23, it becomes obvious that the cascade method requires less modules, even less than those required by the direct method as the order of function goes higher. Hence, the network that results from the cascade method will always have a minimum number of modules. Furthermore, a summing element with three sets of inputs is sufficient for the cascade method but is not enough for the other two methods.

Concerning performance errors, there are three different types of errors: quantization, round-off, and coefficient. Quantization error is the same for all three methods, but as was pointed out [Refs. 12 and 22], the round-off error resulting from the finite arithmetic operations of the recursive filter was found to be less for the cascade method. One way of checking which one of the three methods has the least coefficient error is to assume a double-precision wordlength for the coefficient of an errorless (ideal) case; at the same time, using a single-precision wordlength for the coefficient of an error (nonideal) case. The final result shows that the cascade method has less error [Ref. 13]; therefore, the most economical choice coincides with the best performance.

Among the three synthesis methods, the best one to follow for a given problem is summarized below.

For a given transfer function $G(s) = P_m(s)/P_n(s)$ where $P_m(s)$ and $P_n(s)$ are of order m and n, $n \geq m$, respectively, the numerator and denominator will be decomposed into products of first- and second-order real-coefficient polynomials such that the second-order polynomials cannot be further decomposed into first-order real coefficient polynomials. As a result, the most effective procedure is the cascade method to synthesize each first- and second-order real-coefficient polynomials to achieve minimum realization.

C. Comparison between Universal and Two-Element Modules Used in the Digital Realization of Network Functions

As a result of the minimum-cost criterion described in Chapter III, a comparison is made between the cost of using universal modules and the cost of two-element modules to realize network functions. Suppose

A = integrating-element module

B = summing-element module

U = universal module

Let

C(A) = cost of A

C(B) = cost of B

C(U) = cost of U

Then

$$C(U) = \frac{1}{k} [C(A) + C(B)] \tag{4.23}$$

As proposed in Chapter III, A and B are 19- and 16-pin modules, respectively, and because they are contained in U, U is a 33-pin module, which is two power pins less than A and B. If the cost of an integrated-circuit module is directly proportional to the number of input-output pins, then

$$C(A) = \frac{19}{16} C(B) \tag{4.24a}$$

$$C(U) = \frac{33}{16} C(B) \tag{4.24b}$$

From Eqs. (4.15) and (4.16),

$$C(U) = \frac{1}{k} \left(\frac{35}{16}\right) C(B) \qquad 1 \leq k \leq 35/19 \tag{4.25}$$

If $k < 1$, that is, if U is more expensive than the total cost of A and B, the choice of the two-element module over the universal module is obvious. If $k > (35/19)$, then C(U) is even cheaper than C(A); this is unlikely because A and B are contained in U. Comparing Eqs. (4.17) and (4.18), obtains

$$\frac{1}{k} \frac{35}{16} = \frac{33}{16} \qquad \text{or} \qquad k = \frac{35}{33} \tag{4.26}$$

Consider the case of a general second-order transfer-function realization, as shown in Fig. 24. It takes either 6 universal modules or 6A and 1B for a precision of .1 percent. Then

$$C(T_1) = C(6U) = 6C(U) = \frac{210}{16k}C(B) \qquad (4.27a)$$

$$C(T_2) = C(6A + B) = 6C(A) + C(B) = \frac{115}{16}C(B) \qquad (4.27b)$$

where $C(T_1)$ and $C(T_2)$ are the costs of realization by using universal and two-element modules, respectively. To find if $C(T_1) \geq C(T_2)$, compare Eqs. (4.26) and (4.27):

$$C(T_1) = \frac{210}{16k}C(B) = \frac{210 \cdot 33}{16 \cdot 35}C(B) = \frac{198}{16}C(B) > \frac{115}{16}C(B) = C(T_2) \qquad (4.28)$$

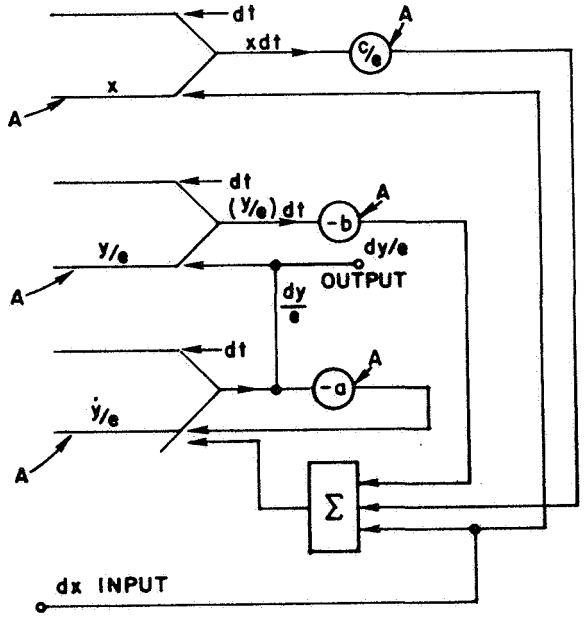


Fig. 24. REALIZATION OF A GENERAL SECOND-ORDER TRANSFER FUNCTION $G(s) = \dfrac{s + c/e}{\frac{1}{e}(s^2 + a\,s + b)}$, $|e| < 1$.

In LSI hybrid technology where one integrates to a certain level on the chip and then employs film technology to interconnect the chips within a package [Ref.8], the above assumed cost-pin linear relationship is approximately true.

For the case of n-tuple precision, i.e., error = $(10^{-3})^n$, n-cascaded digital integrating elements as in Fig. 8 are required. The cost would be

$$C_n(T_1) = 3(1 + n)\,C(U) \qquad (4.29a)$$

$$C_n(T_2) = 6n\,C(A) + C(B) \qquad (4.29b)$$

Substituting Eqs. (4.16) and (4.17) into (4.23) yields

$$C_n(T_1) = \frac{99n + 99}{16} C(B) \tag{4.30a}$$

$$C_n(T_2) = \frac{114n + 16}{16} C(B) \tag{4.30b}$$

If it is desired to make $C_n(T_1) > C_n(T_2)$, then from Eq. (4.30):

$$99n + 99 > 114n + 16 \tag{4.31}$$

or the maximum n that satisfies Eq. (4.31) is found to be

$$n = 5$$

In other words, if the cost-pin linear relationship holds, two-element modules used for building blocks will be least costly for precisions down to 10^{-15}, which is, of course, far more than enough in the application of network-function realization. Note that a precision of 10^{-15} is approximately equivalent to 50-binary-bits precision.

Chapter V

APPLICATION OF THE DIGITAL REALIZATION OF TRANSFER FUNCTIONS

A. Digital Filters

The term digital filter generally refers to the computational process or algorithm by which a sampled signal or sequence of numbers (acting as an input) is transformed into a second sequence of numbers termed the output signal. To date, real-time digital filters utilize digital computers to execute the difference equations in which the desired transfer functions in z-transform are satisfied [Refs. 1, 2, and 3].

The digital filters, proposed in this report, are somewhat different from the above definition and are also different in structure. Physically, they are composed of small digital building blocks (digital integrating and summing elements), and they deal with quantized signals rather than with discrete signals. Their functions are analogous to the continuous (analog) filters and are actually very good approximations to them. These proposed digital filters have definite advantages over the analog and sample-data filters because of their small size, accuracy, stability, and real-time controllability.

The design procedures for continuous filters, such as Butterworth and Chebyshev filters, are treated in standard texts [Refs. 23, 24, and 25] and are discussed briefly in Appendix F.

The technique in designing a digital filter by using digital integrators is fairly straightforward. After the designer decides which type of filter fits his needs, he follows the outlines listed in Appendix F until he obtains the desired transfer function which he then transforms back to the time domain, whereby a linear differential equation can be formed. He follows the synthesis procedures listed in Chapter IV, which results in a diagram of interconnected digital integrators. Thus the design of the digital filter with a minimum number of two-element modules is completed. The following examples are used to illustrate the design and implementation of these filters, using direct, parallel, and cascade methods.

Example 6.

Using the Butterworth transfer function

$$|G(j\omega)|^2 = \frac{1}{1 + \omega^{2n}}$$

design a low-pass filter having a magnitude characteristic such that at a frequency three times the cutoff frequency, the magnitude is at least 25 dB down from its value at zero frequency.

Solution. First, find the required value of n:

$$\left.\frac{1}{1 + \omega^{2n}}\right|_{\omega=3} = 10^{-2.5} \quad \text{or} \quad \log_{10}(1 + 9^n) = 2.5$$

$$9^n \approx 316 \quad \text{or} \quad n = \frac{2.5}{0.954} = 2.62$$

Thus, this required value is the next largest integer $n = 3$.

For $n = 3$, the third-order Butterworth polynomial is

$$B_3(s) = s^3 + 2\,s^2 + 2\,s + 1$$

hence, the transfer function $G(s)$ can be obtained as

$$G(s) = \frac{1}{s^3 + 2\,s^2 + 2\,s + 1} = \frac{Y(s)}{X(s)} \qquad (5.1)$$

Now, try the three methods of realization.

Direct Method:

Transforming $G(s)$ back to the time domain yields

$$\dddot{y}(t) + 2\ddot{y}(t) + 2\dot{y}(t) + y(t) = x(t) \tag{5.2a}$$

or

$$d\ddot{y} = dx - 2d\ddot{y} - 2d\dot{y} - 2dy \tag{5.2b}$$

The solution and the realized-network diagram are shown in Figs. 25 and 26, respectively, and the sinusoidal and step responses are plotted in Figs. 27 and 28.

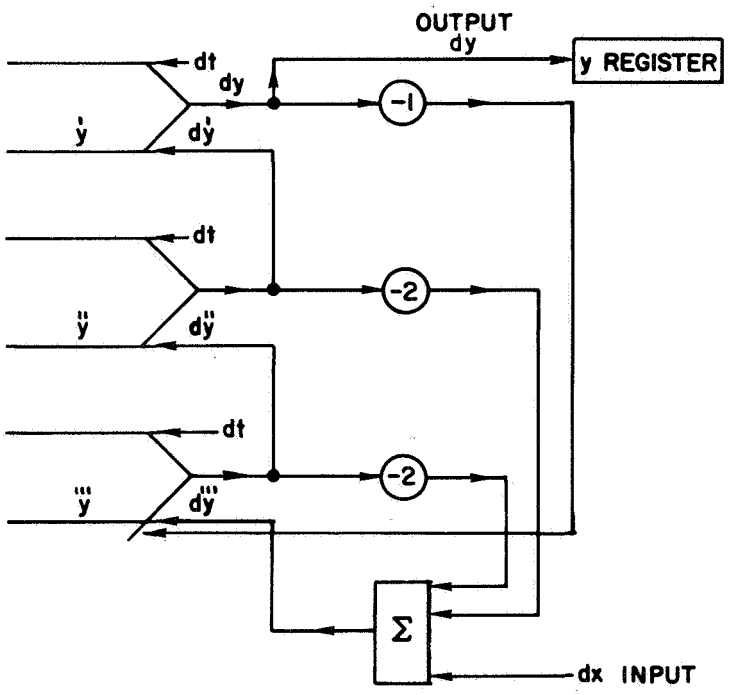


Fig. 25. SOLUTION MAPPING BY DIRECT METHOD.

Parallel Method:

Expanding Eq. (5.1) into partial-fraction form yields

$$G(s) = \frac{1}{s^3 + 2s^2 + 2s + 1} = \frac{Y(s)}{X(s)} = \frac{1}{s+1} - \frac{s}{s^2 + s + 1}$$

$$= G_1(s) - G_2(s)$$

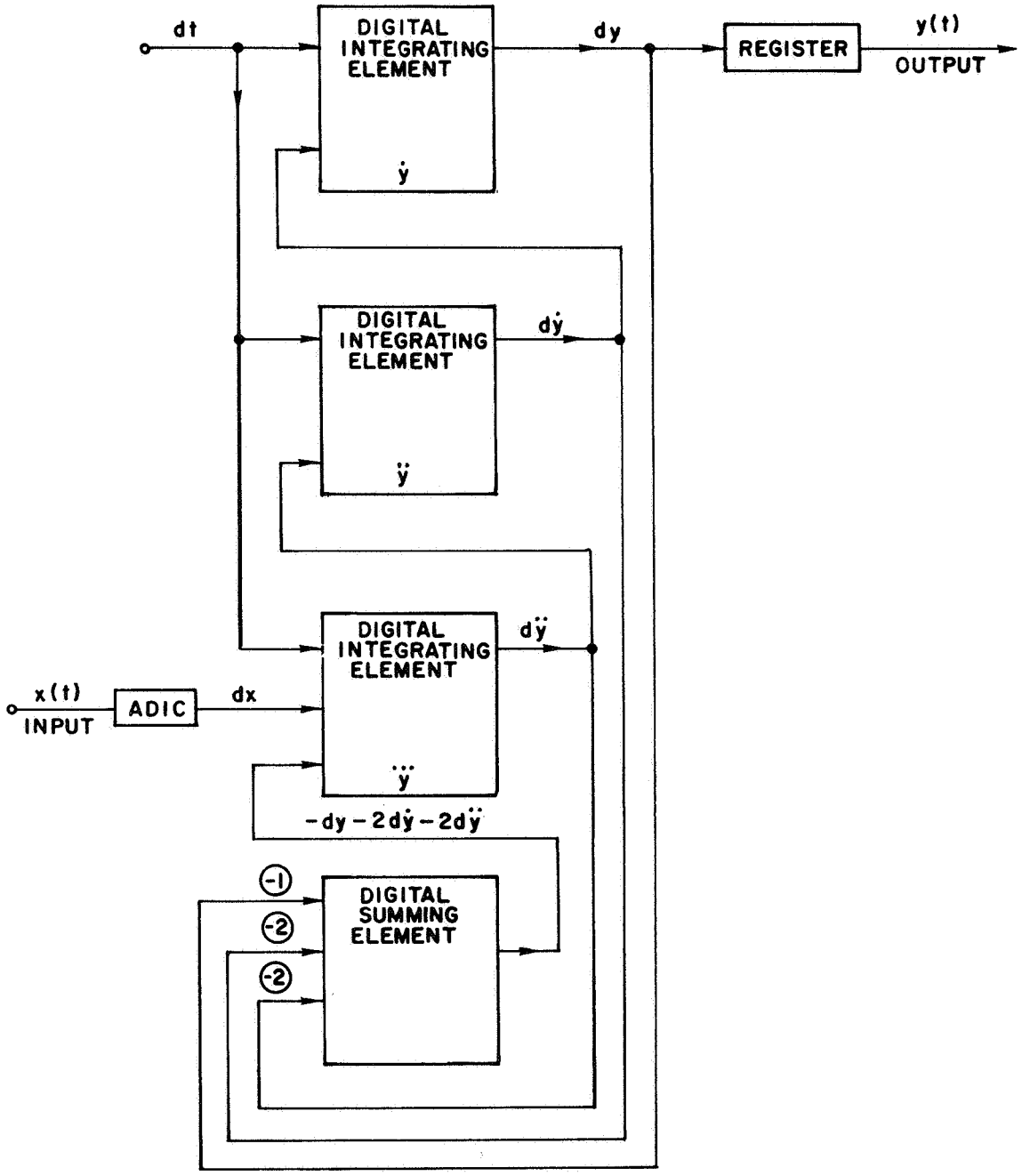


Fig. 26. REALIZED NETWORK OF $G(s) = 1/(s^3 + 2\,s^2 + 2\,s + 1)$ BY DIRECT METHOD.

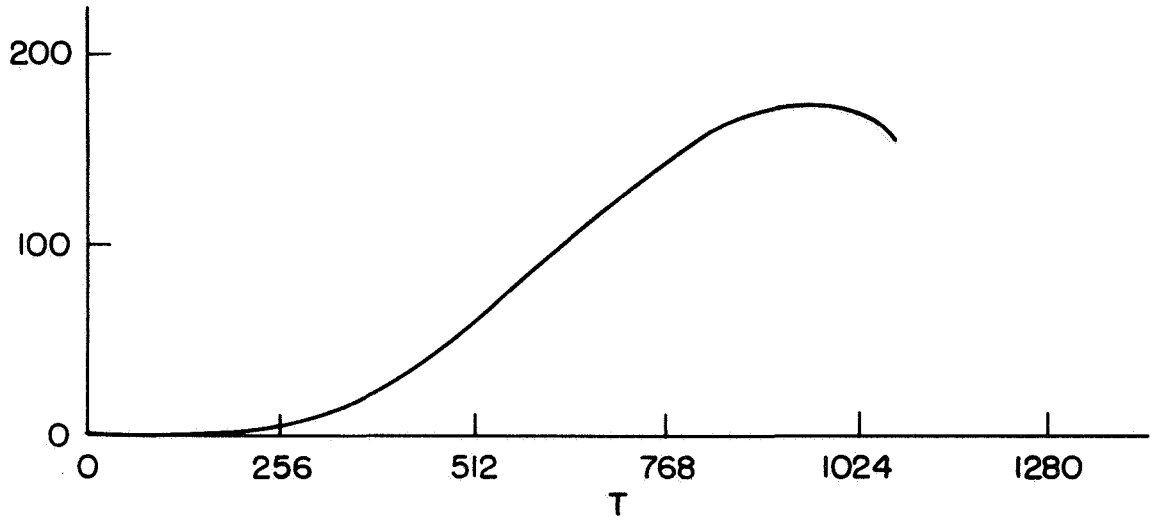


Fig. 27. SINUSOIDAL RESPONSE OF $G(s) = 1/(s^3 + 2\,s^2 + 2\,s + 1)$ WITH INPUT $x(t) = \sin(T/256)$.

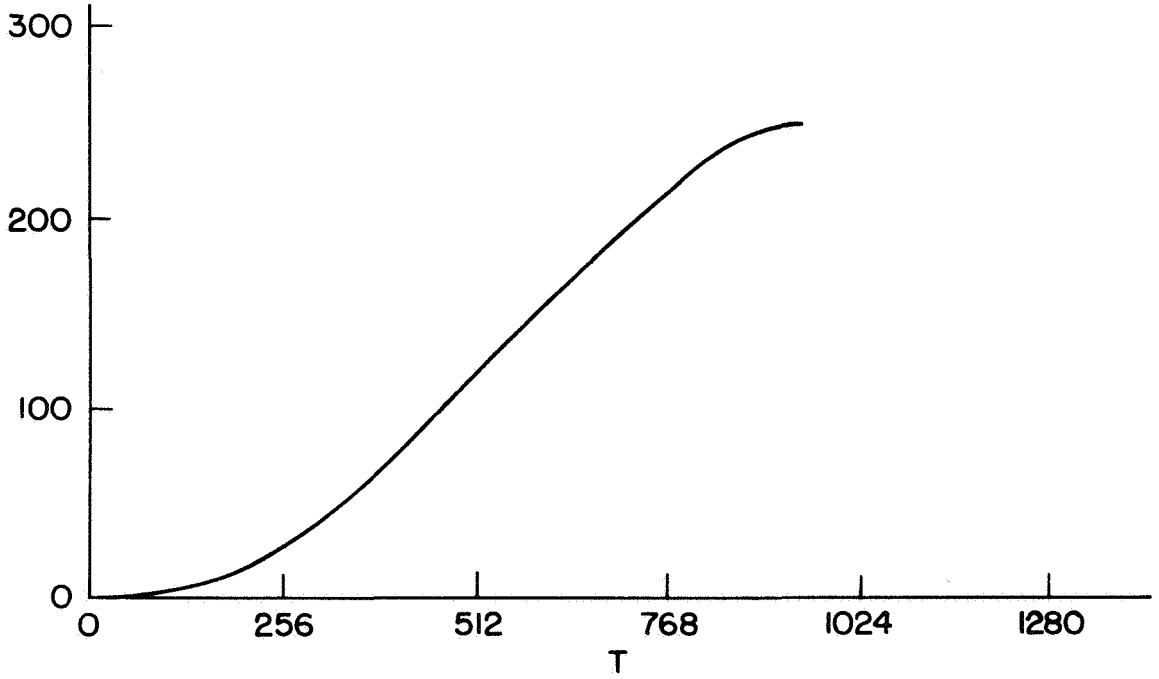


Fig. 28. STEP RESPONSE OF $G(s) = 1/(s^3 + 2\,s^2 + 2\,s + 1)$, WITH INPUT $x(t) = 255$.

$$G_1(s) = \frac{1}{s + 1} = \frac{Y_1(s)}{X(s)}$$

$$G_2(s) = \frac{s}{s^2 + s + 1} = \frac{Y_2(s)}{X(s)}$$

In the time domain:

$$d\dot{y}_1(t) = dx(t) - dy_1(t) \tag{5.3a}$$

and

$$d\dot{y}_2(t) = dx(t) - dy_2(t) - y_2(t)\, dt \tag{5.3b}$$

and the output signal $y(t)$ equals $y_1(t) - y_2(t)$.

The digital filter designed by this method is shown in Fig. 29.

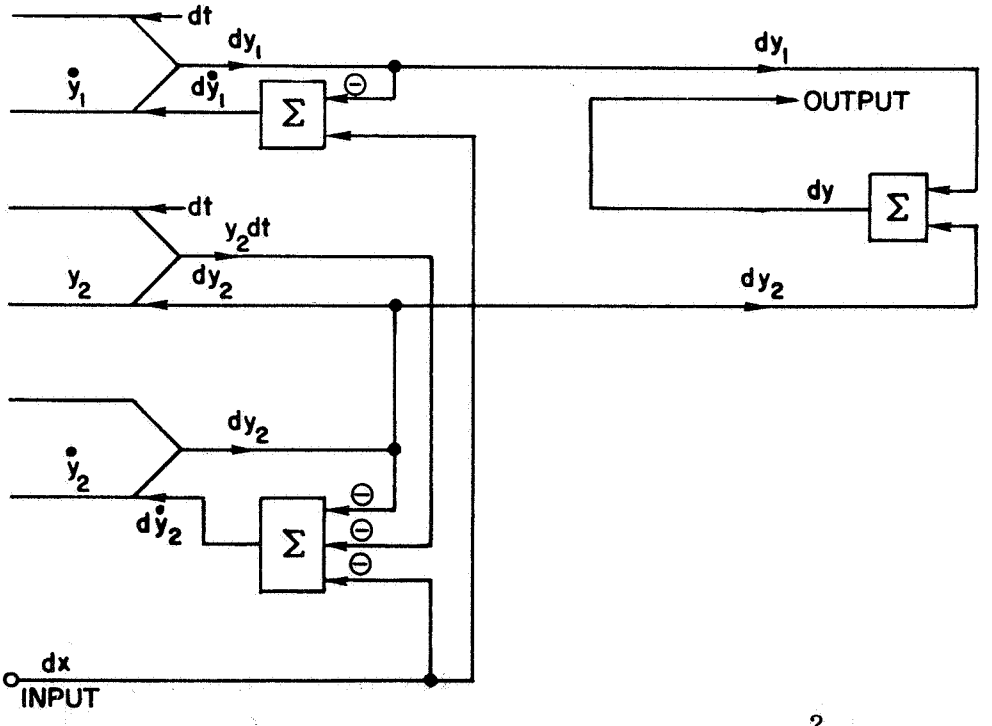


Fig. 29. REALIZATION OF $G(s) = 1/(s + 1) - s/(s^2 + s + 1)$.

Cascade Method:

Factor Eq. (5.1) into

$$G(s) = \frac{1}{s+1}\,\frac{1}{s^2+s+1} = \frac{Y(s)}{Y_2(s)}\,\frac{Y_2(s)}{Y(s)}$$

In the time domain:

$$d\dot{y}(t) = dy_2(t) - dy(t) \tag{5.4a}$$

$$d\ddot{y}_2(t) = dx(t) - d\dot{y}_2(t) - dy_2(t) \tag{5.4b}$$

This cascade realization of G(s) is shown in Fig. 30.

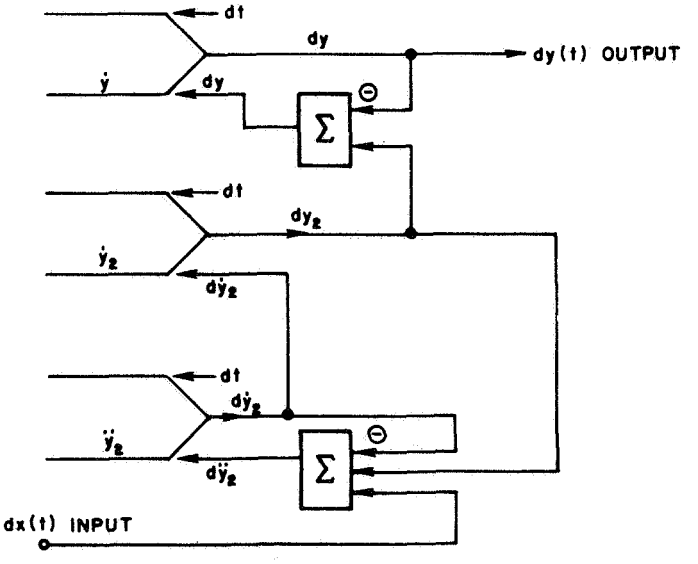


Fig. 30. CASCADE REALIZATION OF $G(s) = 1/(s^3 + 2s^2 + 2s + 1)$.

Example 7.

Design a low-pass filter utilizing the Chebyshev characteristic

$$|G(j\omega)|^2 = \frac{1}{1+\epsilon^2 C_n(\omega)}$$

so that

1. The peak-to-peak ripple in the squared magnitude characteristic does not exceed 15 percent of the maximum value.

2. The magnitude response is down at least 50 dB at $\omega = 4\omega_c = 4$.

Solution. First, it is necessary to calculate the required value of ϵ^2. At the trough of the ripple:

$$\frac{1}{1 + \epsilon^2 C_n(1)} = 1 - 0.15 = 0.85 \quad \text{or} \quad 1 + \epsilon^2 = \frac{20}{17}$$

thus $\epsilon^2 = 0.175$.

At $\omega = 4$:

$$\frac{1}{1 + \epsilon^2 C_n(4)} = 10^{-5} \qquad 1 + \epsilon^2 C_n^2(4) = 10^5$$

or

$$\epsilon^2 C_n^2(4) = 10^5$$

implying that $C_n(4) = 753$.

To find n:

$$\frac{1}{2}\left[\left(\omega + \sqrt{\omega^2 - 1}\right)^n + \left(\omega + \sqrt{\omega^2 - 1}\right)^{-n}\right]_{\omega=4} = 753$$

$$\left(\omega + \sqrt{\omega^2 - 1}\right)^n\Big|_{w=4} \approx 1506 \quad \text{or} \quad n = 3.58$$

Therefore, $n = 4$ will be very satisfactory.

The location of the poles for the fourth-order Chebyshev polynomial can be found at

$$S_k = \sigma_k + j\omega_k \qquad k = 1, 2, 3, 4 \tag{5.5}$$

where

$$\sigma_k = \pm \tanh a \sin [(2k - 1)/n](\pi/2)$$

$$\omega_k = \cos [(2k - 1)/2](\pi/2)$$

and $a = 1/2 \sinh^{-1} 1/\epsilon$

For the present case, $n = 4$, $\epsilon^2 = 0.176$,

$$a = \frac{1}{4} \sinh^{-1} \frac{1}{0.42} = \frac{1}{4} \sinh^{-1} (2.38) = 0.402$$

and

$$\tanh a = 0.38$$

Therefore,

$$s_1 = -0.144 + j0.924$$

$$s_2 = -0.144 - j0.924$$

$$s_3 = -0.351 + j0.383$$

$$s_4 = -0.351 - j0.383 \qquad (5.6)$$

Thus, the transfer function $G(s)$ can be obtained:

$$G(s) = \frac{Y(s)}{X(s)} = \frac{s_1 s_2 s_3 s_4}{(s - s_1)(s - s_2)(s - s_3)(s - s_4)}$$

$$= \frac{0.111}{(s^2 + 0.288 s + 0.416)(s^2 + 0.702 s + 0.268)}$$

$$= \frac{0.111}{(s^4 + 0.99 s^3 + 0.886 s^2 + 0.369 s + 0.111)} \qquad (5.7)$$

Equation (5.7) can be realized by the three methods:

Direct Method:

Transforming G(s) back to the time domain

$$\dddot{y}(t) + 0.99\,\ddot{y}(t) + 0.886\,\dot{y}(t) + 0.369\,\dot{y}(t) + 0.111\,y(t) = 0.111\,x(t) \tag{5.8a}$$

or

$$d\ddot{y} = 0.111\,dx - 0.99\,d\ddot{y} - 0.886\,d\dot{y} - 0.369\,d\dot{y} - 0.111\,dy \tag{5.8b}$$

The network realized, corresponding to the above equation, is shown in Fig. 31.

Parallel Method:

$$G(s) = -\frac{0.414\,s + 0.268}{s^2 + 0.288\,s + 0.416} + \frac{0.414\,s + 0.439}{s^2 + 0.702\,s + 0.268}$$

$$= -G_1(s) + G_2(s) \tag{5.9}$$

where

$$G_1(s) = \frac{0.414\,s + 0.268}{s^2 + 0.288\,s + 0.416} = \frac{Y_1(s)}{X(s)}$$

$$G_2(s) = \frac{0.414\,s + 0.439}{s^2 + 0.702\,s + 0.268} = \frac{Y_2(s)}{X(s)}$$

Transforming $G_1(s)$ and $G_2(s)$ back to the time domain yields

$$\ddot{y}_1(t) + 0.288\,\dot{y}_1(t) + 0.416\,y_1(t) = 0.414\,\dot{x}(t) + 0.268\,x(t) \tag{5.10a}$$

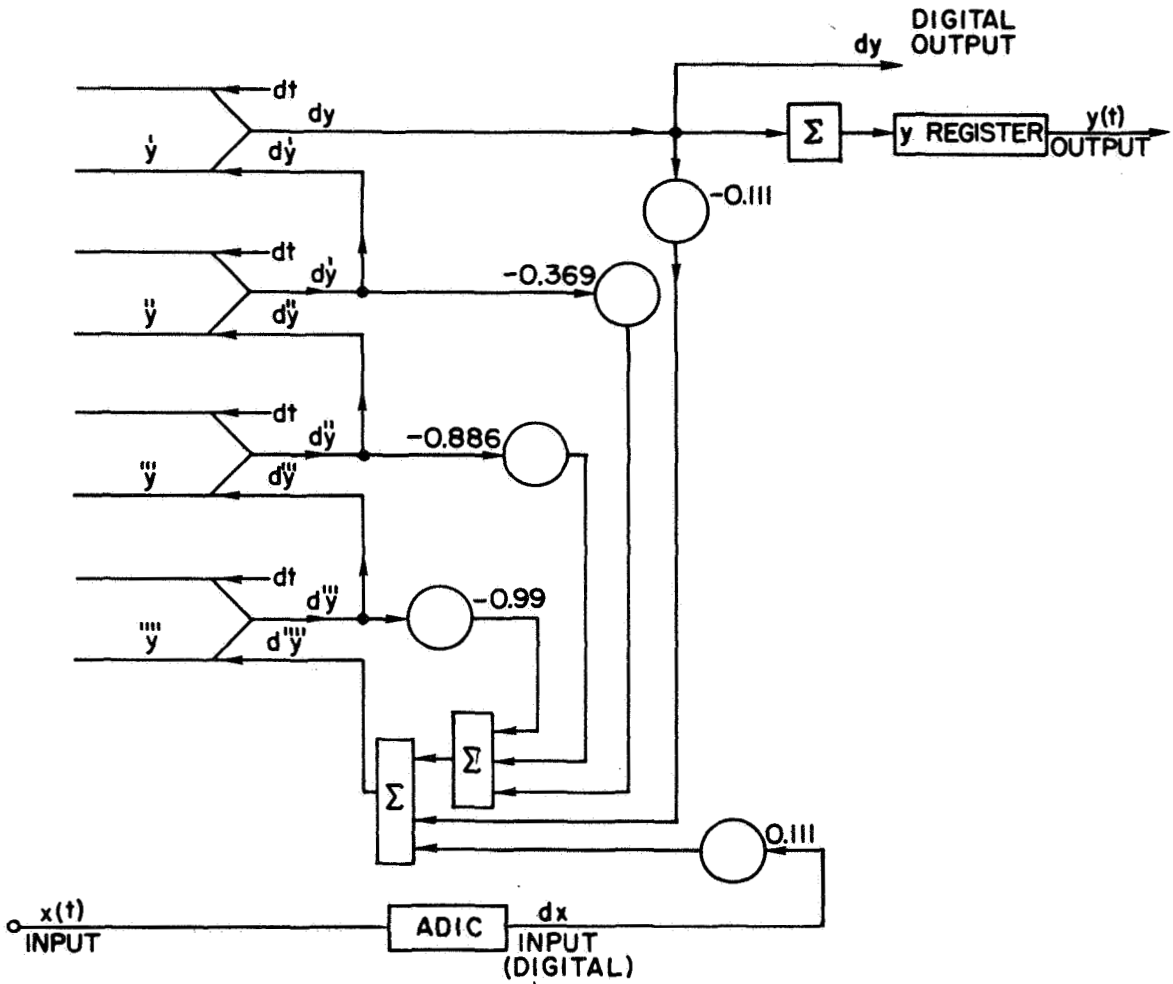


Fig. 31. REALIZATION OF
$G(s) = 0.111/(s^4 + 0.99\ s^3 + 0.886\ s^2 + 0.369\ s + 0.111)$
BY DIRECT METHOD.

and

$$\ddot{y}_2(t) + 0.702\ \dot{y}_2(t) + 0.268\ y_2(t) = 0.414\ \dot{x}(t) + 0.439\ x(t) \tag{5.10b}$$

or

$$d\dot{y}_1 = 0.414\ dx - 0.288\ dy_1 + 0.268\ xdt - 0.416\ y_1 dt \tag{5.10c}$$

and

$$d\dot{y}_2 = 0.414\ dx - 0.702\ dy_2 + 0.439\ xdt - 0.268\ y_2 dt \tag{5.10d}$$

and the overall output $y(t)$ is

$$y(t) = y_2(t) - y_1(t)$$

The network realized by this parallel method is shown in Fig. 32.

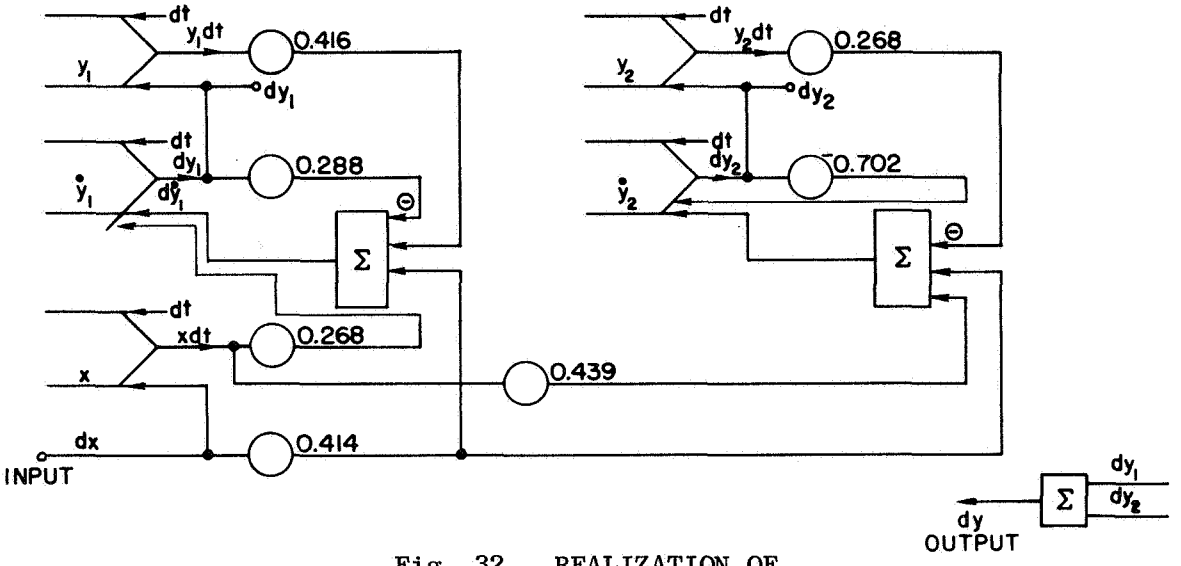


Fig. 32. REALIZATION OF
$G(s) = 0.111/(s^4 + 0.99\ s^3 + 0.886\ s^2 + 0.369\ s + 0.111)$
USING PARALLEL METHOD.

Cascade Method:

$$G(s) = \frac{1}{s^2 + 0.288\ s + 0.416} \quad \frac{0.111}{s^2 + 0.702\ s + 0.268} = \frac{Y(s)}{Y_2(s)} \quad \frac{Y_2(s)}{X(s)} \qquad (5.11)$$

In the time domain:

$$d\ddot{y}(t) = dy_2(t) - 0.288\ d\dot{y}(t) - 0.416\ dy(t) \qquad (5.12a)$$

$$d\ddot{y}_2(t) = 0.111\ dx(t) - 0.702\ d\dot{y}_2(t) - 0.268\ dy_2(t) \qquad (5.12b)$$

Network realization by this cascade method is shown in Fig. 33.

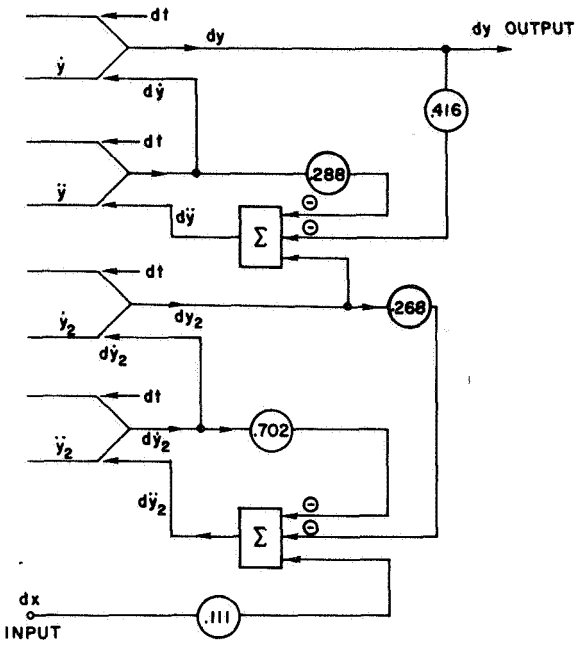


Fig. 33. CASCADE REALIZATION OF
$G(s) = 0.111/(s^4 + 0.99\ s^3 + 0.886\ s^2 + 0.369\ s + 0.111)$.

B. Digital Spectrum Analyzer

The spectrum analyzer is a device used to measure the distribution of energy at different frequencies of interest. One method of measuring the frequency spectrum is to measure the energy in the passbands of a bank of narrowband filters. The digital spectrum analyzer of interest here is realized by a bank of narrowband digital filters, each with a fixed bandwidth spanning the entire frequency range. An example of how to design a bank of bandpass filters is given below.

Example 8.

Design a bank of bandpass filters with a common input, each 400 cycles/s wide, covering the band 300 to 3100 Hz. Contiguous filters are required to cross at -3 dB of the midband gain. Use third-order maximally flat approximation.

Solution. The transfer function of each bandpass filter can be found from the frequency transformation of a low-pass filter. For a third-order maximally flat low-pass filter, the transfer function is

$$G_L(s) = \frac{1}{s^3 + 2\ s^2 + 2\ s + 1}$$

Make the following transformation:

$$s = \frac{\omega_o}{B}\left(\frac{p}{\omega_o} + \frac{\omega_o}{p}\right) \tag{5.13}$$

$$G_B(s) = \frac{B^3 p^3}{p^6 + 2Bp^5 + (3\omega_o^2 + 2B^2)p^4 + (B^3 + 4B\omega_o^2)p^3 + (3\omega_o^4 + 2B^2\omega_o^2)p^2 + 2B\omega_o^4 p + \omega_o^6} \tag{5.14}$$

where

B = bandwidth of the bandpass filter

ω_o = geometrical mean of the cutoff frequencies of the bandpass filter = $\sqrt{\omega_1 \omega_2}$

ω_1 = lower cutoff frequency of the bandpass filter

ω_2 = higher cutoff frequency of the bandpass filter

p = complex frequency

With the above transformation formula, the first bandpass filter, from 300 to 700 Hz, can be designed as $\omega_o = \sqrt{300 \times 700}$ Hz = 458.258 Hz and B = 400 Hz. Normalizing $B_n = 1$ and $\omega_{o1} = 458.258/400 = 1.1456$, the transfer function of the first bandpass filter is

$$G_1(p) = \frac{p^3}{p^6 + 2p^5 + 5.937p^4 + 6.25p^3 + 7.79p^2 + 3.446p + 3.26} \tag{5.15}$$

Without much difficulty, this transfer function can be realized by the syntheses found in Chapter IV.B, but an attempt will be made to realize it by the cascade method. There is no doubt that theoretically the denominator of $G_1(p)$ can be factorized as $(p^2+ap+b)(p^2+cp+d)(p^2+ep+f)$, but it is very difficult to factor it as the order of the polynomials increases. From another point of view, the denominator might be factorized directly from the transfer function of the lowpass filter because the pole locations of the transfer function are known. This can be demonstrated by utilizing both the parallel and cascade methods:

Parallel Method:

$$G_L(s) = \frac{1}{s^3 + 2s^2 + 2s + 1} = \frac{1}{s+1} - \frac{s}{s^2 + s + 1}$$

$$= \frac{1}{s+1} - \frac{s}{(s + 0.5 + j0.866)(s + 0.5 - j0.866)}$$

Now the transformation can be made:

$$s = \frac{\omega_o}{B}\left(\frac{p}{\omega_o} + \frac{\omega_o}{p}\right) = \frac{1}{B}\left(p + \frac{\omega_o^2}{p}\right)$$

then

$$G_B(p) = \frac{Bp}{p^2 + p + \omega_o^2} - \frac{p^3 + \omega_o^2 p}{\left[p^2 + (0.5 + j0.866)Bp + \omega_o^2\right]\left[p^2 + (0.5 - j0.866)Bp + \omega_o^2\right]} \qquad (5.16a)$$

$$G_B(p) = \frac{1}{1/B\left[p + \left(\omega_o^2/p\right)\right] + 1/B} - \frac{1/B\left[p + \left(\omega_o^2/p\right)\right]}{\left\{1/B\left[p + \left(\omega_o^2/p\right)\right] + 0.5 + j0.866\right\}\left\{1/B\left[p + \left(\omega_o^2/p\right)\right] + 0.5 - j0.866\right\}} \qquad (5.16b)$$

The poles of the second term of Eq. (5.15b) can be found by solving

$$p^2 + (0.5 + j0.866)Bp + \omega_o^2 = 0$$

and the solutions are

$$p = \frac{1}{2}\left[-(0.5 + j0.866)B \pm \sqrt{(0.5 + j0.866)^2 B^2 - 4\omega_o^2}\right]$$

$$= \frac{1}{2}\left[-(0.5 + j0.866)B \pm (\sqrt{u} + j\sqrt{v})\right] = \frac{1}{2}\left[(-0.5B \pm \sqrt{u}) - j(0.866\,B \mp \sqrt{v})\right]$$

where

$$u - v = B^2 - 4\omega_o^2$$

$$uv = 3B^2/16$$

$$u + v = + \sqrt{(u - v)^2 + 4uv}$$

These solutions can be written explicitly as

$$p_1 = (-0.25B + 0.5\sqrt{u}) - j(0.433 - 0.5\sqrt{v}) \tag{5.17a}$$

$$p_2 = (-0.25B - 0.5\sqrt{u}) - j(0.433 + 0.5\sqrt{v}) \tag{5.17b}$$

The other two poles can be found by solving

$$p^2 + (0.5 - j0.866)Bp + \omega_o^2 = 0 \tag{5.18a}$$

or

$$p_3 = (-0.25B + 0.5\sqrt{u}) + j(0.433 - 0.5\sqrt{v}) \tag{5.18b}$$

$$p_4 = (-0.25B - 0.5\sqrt{u}) + j(0.433 + 0.5\sqrt{v}) \tag{5.18c}$$

then $G_B(p)$ can be written in partial-fraction form as

$$G_B(p) = \frac{Bp}{p^2 + p + \omega_o^2} - \frac{a_1 p + b_1}{(p - p_1)(p - p_3)} - \frac{a_2 p + b_2}{(p - p_2)(p - p_4)} \tag{5.19}$$

where a_1, a_2, b_1, and b_2 are real constants and can be found in terms of B and ω_o from Eqs. (5.16). For the present design problem, $B = 1$, $\omega_{o1} = 1.1456$, $u = 0.0448$, $v = 4.2948$. Substituting these values into Eqs. (5.17) and (5.18) yields $p_1, p_3 = -0.144 \pm j0.603$; $p_2, p_4 = -0.356 \mp j1.469$. Substituting them into Eq. (5.19) obtains

$$G_1(p) = \frac{p}{p^2 + p + 1.312} - \frac{0.545\ p + 0.105}{p^2 + 0.288\ p + 0.384} - \frac{0.455\ p - 0.624}{p^2 + 0.712\ p + 2.284} \tag{5.20}$$

There is no doubt that the transfer function $G_1(p)$ of the bandpass filter can be realized by the parallel connection of three building blocks, each block containing digital integrators, such as those discussed in the previous section.

Similarly, the next bandpass filter of frequency range 700 to 1100 Hz can be designed with $\omega_{02}^2 = (700 \times 1100)/(400 \times 400) = 4.813$; $B_n = 1$, $u \approx 0$, and $v = 18.25$; therefore,

$$p_1, p_3 = -0.25 \pm j1.704$$

$$p_2, p_4 = -0.25 \mp j2.57$$

and

$$G_2(p) = \frac{p}{p^2 + p + 4.813} - \frac{0.498\ p + 0.351}{p^2 + 0.5\ p + 2.96} - \frac{0.502\ p - 0.851}{p^2 + 0.5\ p + 7.18} \tag{5.21}$$

The realization of $G_2(p)$ is similar to $G_1(p)$. The realized networks of $G_1(p)$ are shown in Figs. 34 and 35. By the same token, $G_2(p)$, $G_3(p)$, $G_4(p)$, ..., $G_7(p)$ can be designed by simply changing the contents of the constant registers, as in Fig. 36.

Cascade Method:

$$G_L(s) = \frac{1}{s^3 + 2\ s^2 + 2\ s + 1} = \frac{1}{s + 1} \frac{1}{s^2 + s + 1}$$

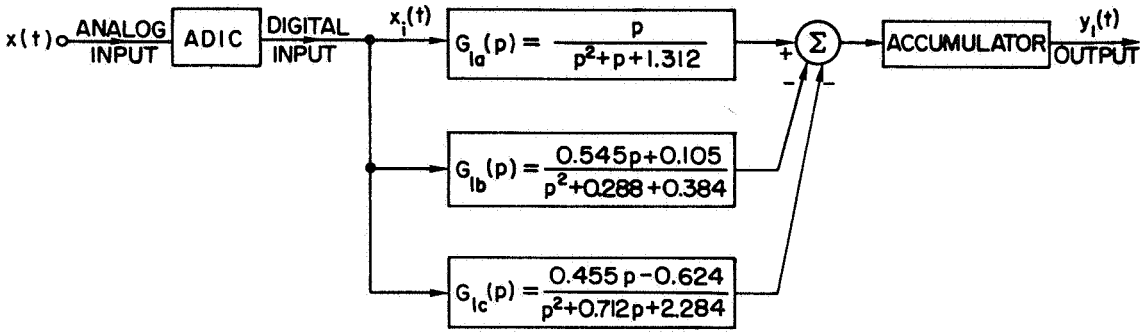
$$= \frac{1}{s + 1} \quad \frac{1}{s + 0.5 + j0.866} \quad \frac{1}{s + 0.5 - j0.866}$$

Now the transformation can be made:

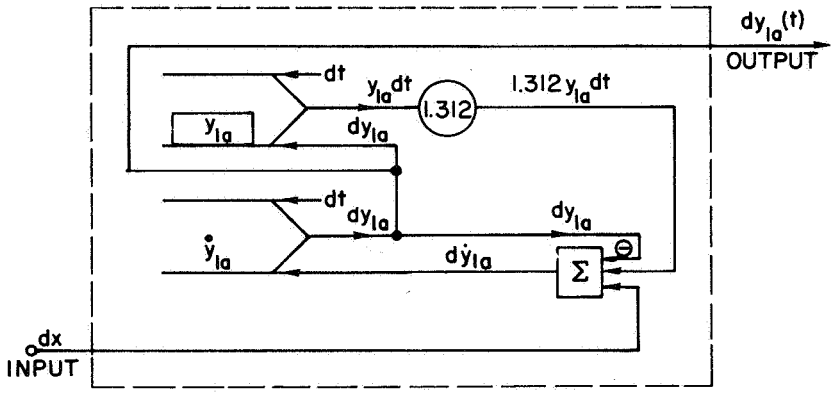
$$s = \frac{\omega_o}{B}\left(\frac{p}{\omega_o} + \frac{\omega_o}{p}\right) = \frac{1}{B}\left(p + \frac{\omega_o^2}{p}\right)$$

then

$$G_B(p) = \frac{Bp^2}{\left[p^2 + (0.5 + j0.866)Bp + \omega_o^2\right]\left[p^2 + (0.5 - j0.866)Bp + \omega_o^2\right]} \quad \frac{Bp}{p^2 + p + \omega_o^2} \tag{5.22}$$



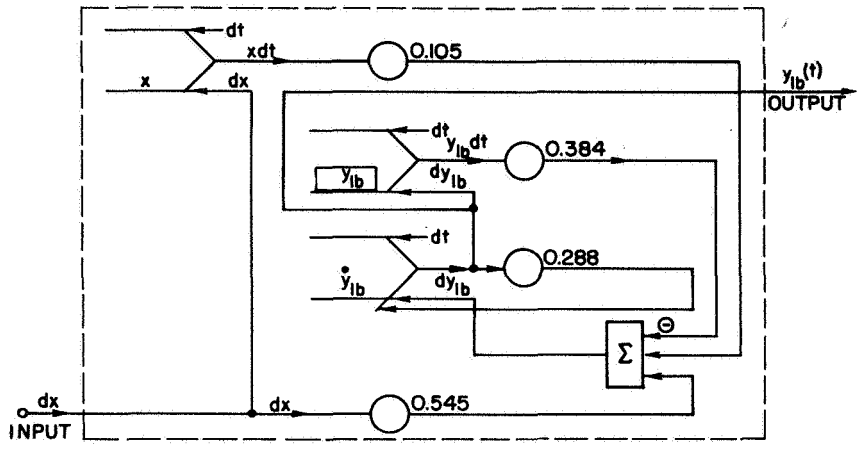
a. Realization of $G_1(p)$



b. $G_{1a}(p) = \dfrac{p}{p^2 + p + 1.312}$

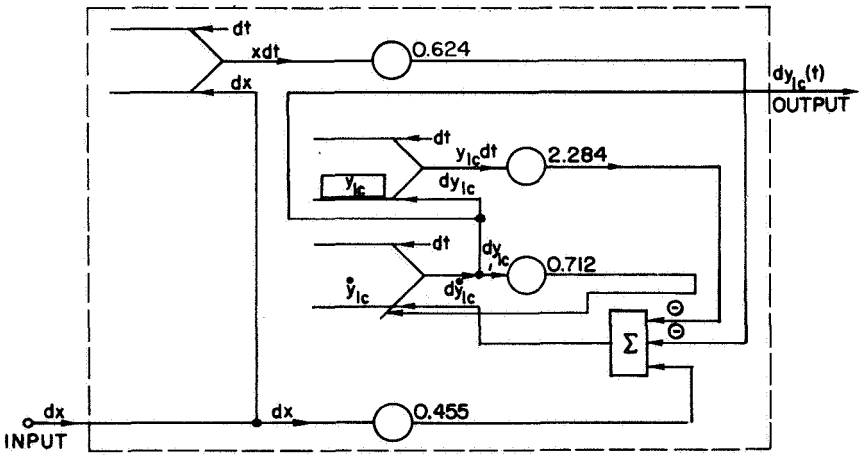
$$d\dot{y}_{1a} = dx - dy_{1a} - 1.312\ y_{1a}dt$$

Fig. 34. REALIZATION OF $G_{1a}(p)$.



a. $$G_{1b}(p) = \frac{0.545\ p + 0.105}{p^2 + 0.288\ p + 0.384}$$

$$d\dot{y}_{1b} = 0.105\ xdt - 0.384\ y_{1b}dt + 0.545\ dx - 0.288\ dy_{1b}$$



b. $$G_{1c}(p) = \frac{0.455\ p - 0.624}{p^2 + 0.712\ p + 2.284}$$

$$d\dot{y}_{1c} = 0.455\ dx - 0.712\ dy_{1c} - 0.624\ xdt - 2.284\ y_{1c}dt$$

Fig. 35. REALIZATION OF $G_{1b}(p)$ and $G_{1c}(p)$.

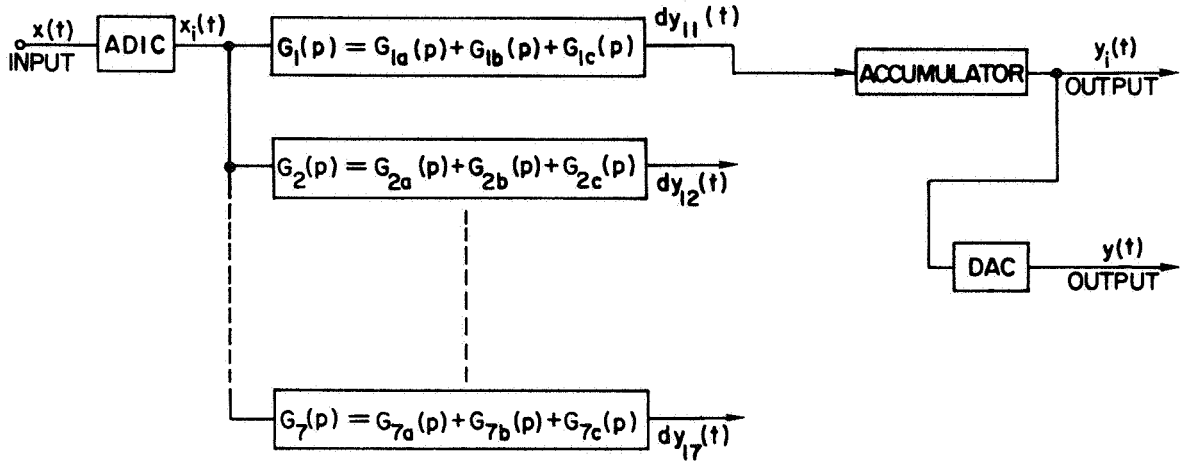


Fig. 36. REALIZATION OF A BANK OF DIGITAL FILTERS.

Following the same procedure as in the parallel method, poles p_1, p_2, p_3, and p_4 can be found as in Eqs. (5.14) and (5.15). Substituting $B = 1$, $\omega_o = 1.1456$, $u = 0.0448$, $v = 4.2948$ into Eqs. (5.14) and (5.15) and then into $G_B(p)$ yields

$$G_1(p)' = \frac{p}{p^2 + p + 1.312} \frac{p}{p^2 + 0.288\,p + 0.384} \frac{p}{p^2 + 0.712\,p + 2.284} \tag{5.23}$$

The digital realization of $G_1(p)'$ is shown in Fig. 37, where 14 modules are used; $G_2(p)'$, $G_3(p)'$, ..., $G_7(p)'$ can be similarly designed. Thus, the overall configuration is a combination of cascade and parallel realization.

It is seen from Eq. (5.15) that if the sixth-order polynomial with a small error produced at one pole location is factorized, the remaining five pole locations will be affected; that is, a small coefficient perturbation (or truncation) may result in a large shift in root location. On the other hand, the factorized two-pole filter combination will result in better performance because the coefficient perturbation of the first factorized term will not change the pole position in the second and/or third terms. Therefore, it is advisable to use the cascade and parallel combination in the design.

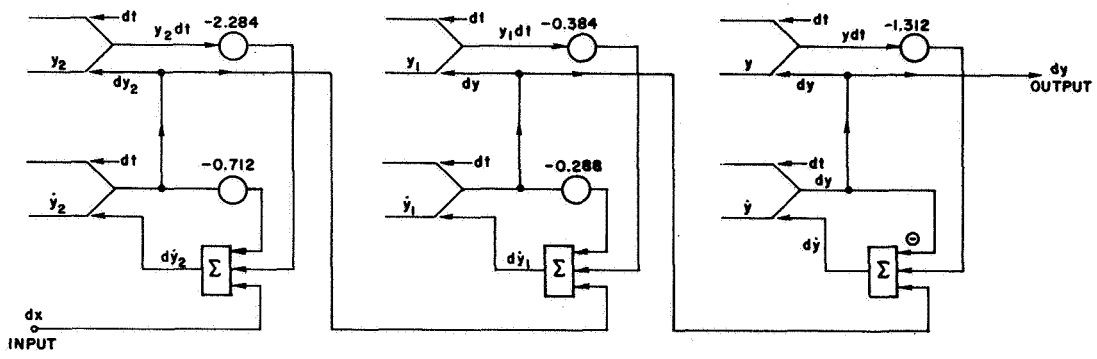


Fig. 37. THE DIGITAL REALIZATION OF $G_1(s)'$ OF EQ. (5.23).

C. Digital Filters with Time-Varying Coefficients

It is possible to design a digital filter with time-varying coefficients. Because the transfer function of this type of filter is not defined, as are those in Chapter IV.A, a time-domain synthesis will be presented. The following example illustrates the realizability of this filter.

Example 9.

Suppose $x(t)$ and $y(t)$ are the input and output of a digital filter in such a way that the following relation is satisfied:

$$\dddot{y}(t) + a(t)\,\ddot{y}(t) + b(t)\,\dot{y}(t) + c(t)\,y(t) = c(t)\,x(t) \qquad (5.24)$$

Design such a network.

Solution. It is seen that if $a(t)$, $b(t)$, and $c(t)$ are constants, the problem can be reduced to one of the typical digital-filter design problems. Now, with time-varying coefficients, it is necessary first to differentiate the equation,

$$d\ddot{y} = d(cx - cy) - d(a\ddot{y}) - d(b\dot{y}) \qquad (5.25)$$

and then to use digital integrators to generate the terms $d(a\ddot{y})$, $d(b\dot{y})$, $d(cy)$, and $d(cx)$. Note that

$$d(a\ddot{y}) = ad\ddot{y} + \ddot{y}da$$

which can be generated by integrators, as shown in Fig. 38, where $d\ddot{y}$ is an output of an integrator, and where da is the differential of the input $a(t)$ and is controllable from the outside. By using more integrators, this problem can be designed as in Fig. 39.

The advantages or any application of digital filters with nonconstant coefficients have not been investigated, but an interesting case concerning this particular type of filter will be discussed.

As is known, the location of the poles of the Chebyshev filters differs only slightly from those of the Butterworth filters [Ref. 24]. Therefore, if the time-varying coefficients are changing in such a way that the poles are shifting horizontally (in the s-plane) from the Butterworth-pole locus to the Chebyshev-pole locus, the digital filter will have a changing magnitude square characteristic from the maximum flat response to different equal-ripple responses.

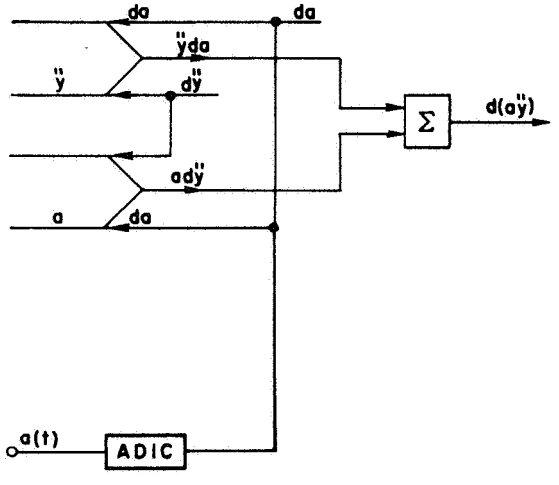


Fig. 38. GENERATION OF $d(a\ddot{y})$.

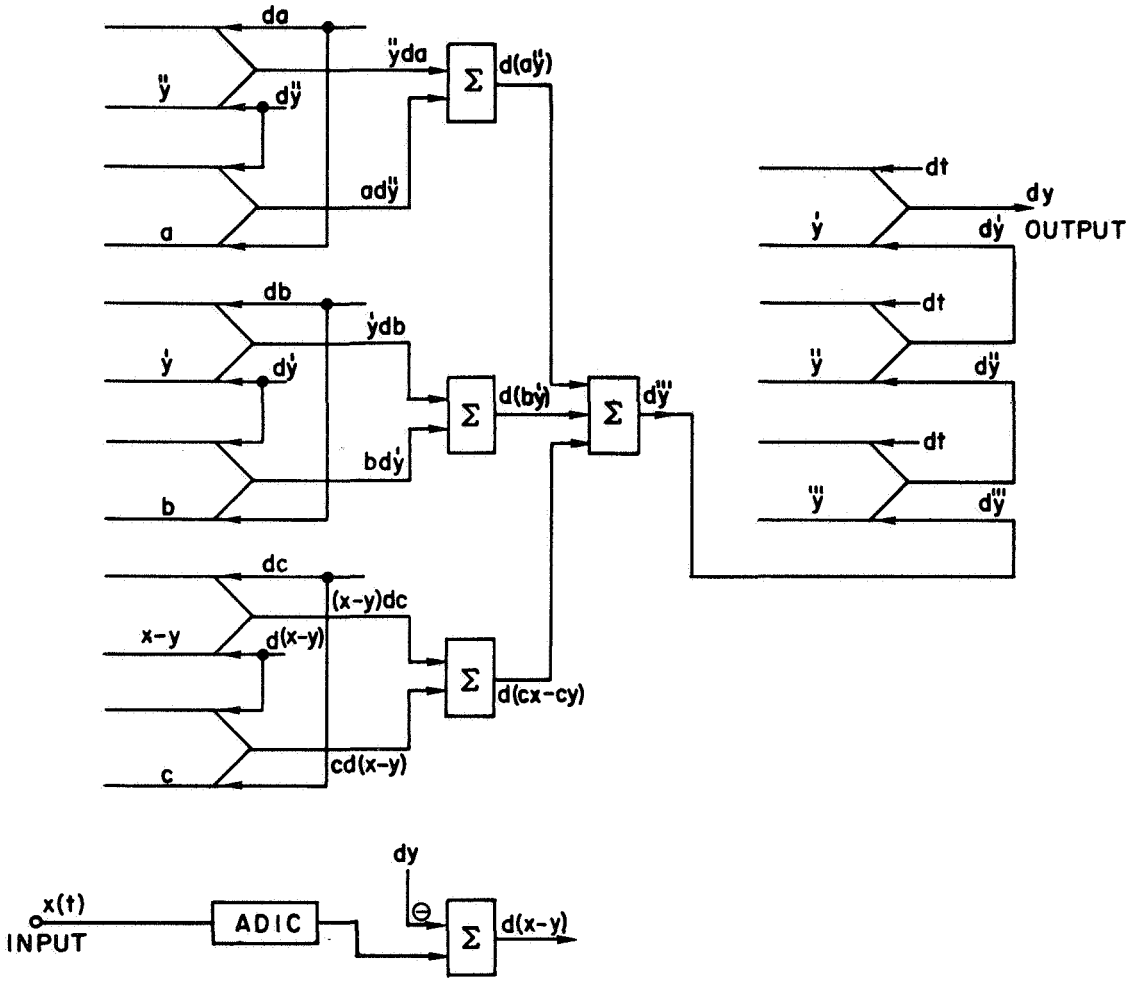


Fig. 39. REALIZATION OF DIGITAL FILTER WITH TIME-VARYING COEFFICIENTS.

Example 10.

For purposes of illustration, a low-pass filter with the following specifications has been designed [Ref. 26].

1. The allowable deviation from the ideal in the passband must be equal to or less than 1/2 dB. The passband extends from 0 to 100 Hz.
2. The attenuation must be at least 18 dB at frequencies higher than 200 Hz.

Solution. The actual configuration of the network designed [Ref. 26] is shown in Fig. 40, and the transfer function is

$$G(s) = \frac{V_2(s)}{V_1(s)} = \frac{0.36 \times 10^6}{s^3 + 125\ s^2 + 3.78 \times 10^6\ s + 0.72 \times 10^6} = \frac{Y(s)}{X(s)}$$

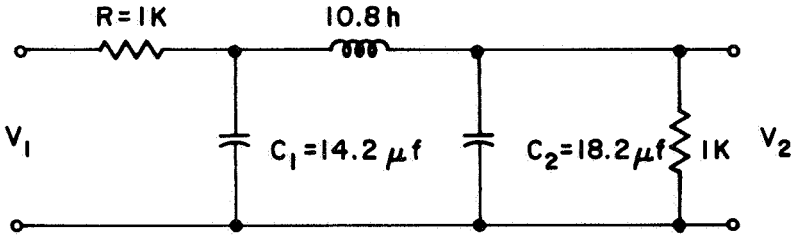


Fig. 40. LOW-PASS FILTER DESIGN OF EXAMPLE 10.

For simplicity, the above low-pass filter can be synthesized by using the direct method,

$$\dddot{y} + 125\ \ddot{y} + 3.78 \times 10^6\ \dot{y} + 0.72 \times 10^6\ y = 0.36 \times 10^6\ x$$

$$\frac{d\ddot{y}}{10^6} + 0.125\ \frac{d\ddot{y}}{10^3} + 3.78\ d\dot{y} + 0.72\ dy = 0.36\ dx$$

$$\frac{d\ddot{y}}{10^6} = 0.36\ dx - 0.125\ \frac{d\ddot{y}}{10^3} - 3.78\ d\dot{y} - 0.72\ dy \qquad (5.26)$$

and the equivalent digital realization is shown in Fig. 41.

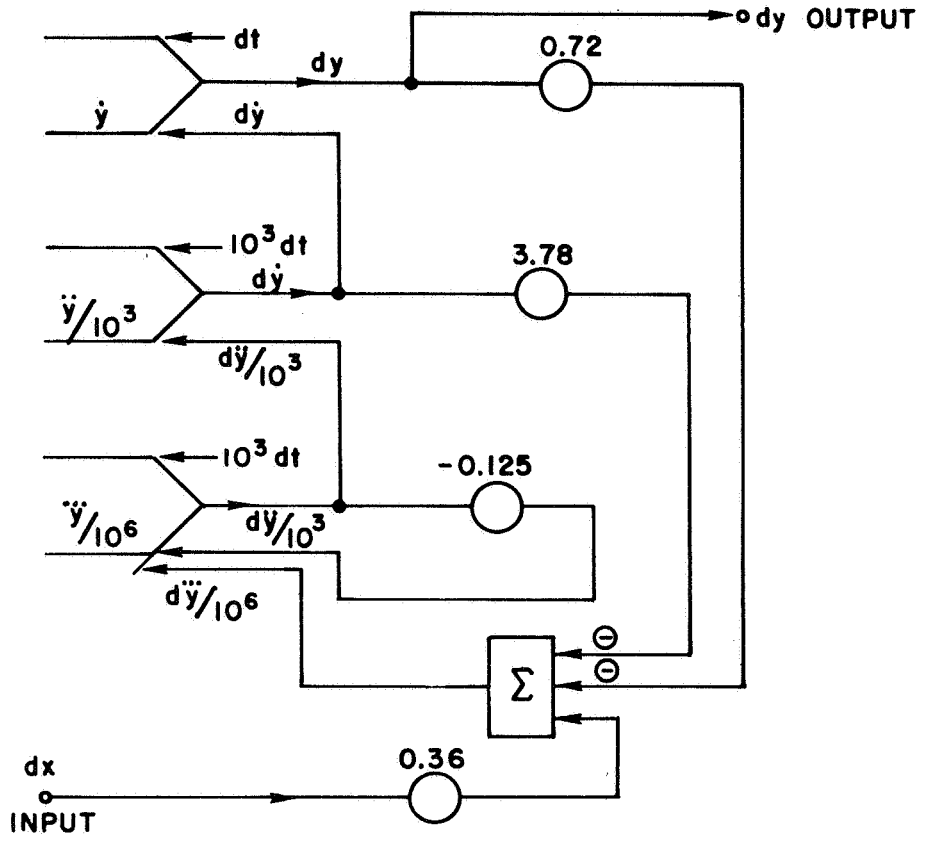


Fig. 41. EQUIVALENT CIRCUIT REALIZATION BY DIGITAL ELEMENTS.

Chapter VI

DIGITAL REALIZATION OF DRIVING-POINT IMMITTANCE FUNCTIONS

A. Driving-Point Impedance-Function Realization

The driving-point impedance function $Z_1(s)$ of a network is defined as the ratio of $V_{in}(s)$ to $I_{in}(s)$, where V_{in} and I_{in} are, respectively, the input voltage and input current in Laplace transform form; namely, $Z_1(s) = V_{in}(s)/I_{in}(s)$.

Consider a 1-port network, as shown in Fig. 42, with an internal current transfer function $G_I(s)$ defined as

$$G_I(s) = \frac{I_2(s)}{I_1(s)} \tag{6.1}$$

where $I_1(s)$ and $I_2(s)$ are the input and output currents, respectively, of the network function $G_I(s)$. Thus

$$Z_1(s) = \frac{V_{in}(s)}{I_{in}(s)} = \frac{V_{in}(s)}{I_1(s) - I_2(s)} = \frac{V_{in}(s)}{I_1(s)[1 - G_I(s)]} \tag{6.2}$$

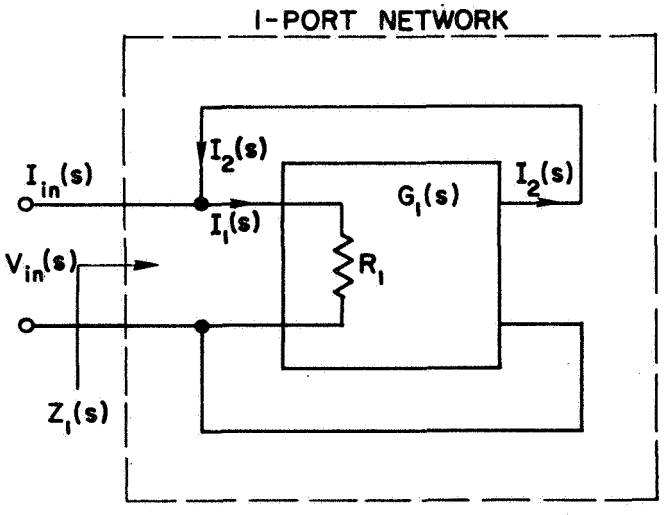


Fig. 42. FEEDBACK CONNECTION USED TO REALIZE THE DRIVING-POINT IMPEDANCE FUNCTION.

Solving for $G_I(s)$ yields

$$G_I(s) = 1 - \frac{V_{in}(s)}{I_1(s)Z_1(s)} \tag{6.3}$$

if

$$R_1 I_1(s) = V_{in}(s) \tag{6.4}$$

Then

$$G_I(s) = 1 - \frac{R_1}{Z_1(s)} \tag{6.5}$$

where R_1 is a constant (resistance).

From the last two chapters, it is known that once the transfer function of the network is specified, it can be realized by interconnecting the digital integrators. Therefore, once the desired driving-point function $Z_1(s)$ is specified, $G_I(s)$ can be found. In this way the 1-port network with the desired $Z_1(s)$ can be constructed, as in Fig. 42.

Example 11.

Realize the following driving-point impedance function by using the prescribed technique

$$Z_1(s) = \frac{s^2 + s + 2}{2s^2 + s + 1} = \frac{V_{in}(s)}{I_{in}(s)}$$

Solution. First, the transfer function $G_I(s)$ must be found. Using Eq. (6.5) obtains

$$G_I(s) = 1 - \frac{R_1}{Z_1(s)} = \frac{s^2(1 - 2R_1) + s(1 - R_1) + (2 - R_1)}{s^2 + s + 2}$$

$$= \frac{I_2(s)}{I_1(s)} = \frac{R_1 I_2(s)}{V_{in}(s)}$$

Let $Y(s) = R_1 I_2(s)$ [note, here, that $Y(s)$ is not the admittance function] and $X(s) = V_{in}(s)$, then:

$$(s^2 + s + 2)\, Y(s) = [s^2(1 - 2R_1) + s(1 - R_1) + (2 - R_1)]\, X(s)$$

or

$$\left(1 + \frac{1}{s} + \frac{2}{s^2}\right) Y(s) = \left[(1 - 2R_1) + \frac{(1 - R_1)}{s} + \frac{(2 - R_1)}{s^2}\right] X(s)$$

Transforming the above expression back to the time domain yields

$$y(t) = (1 - 2R_1)\, x(t) + \int [(1 - R_1)\, x(t) - y(t)]\, dt$$

$$+ \int\int [(2 - R_1)\, x(t) - 2y(t)]\, dt^2$$

or

$$dy = (1 - 2R_1)\, dx + [(1 - R_1)\, x - y]\, dt + \int [(2 - R_1)\, x - 2y]\, dt^2$$

The network realized for $G_I(s)$ is shown in Fig. 43. By a connection similar to Fig. 42, the network realization for $Z_1(s)$ is as shown in Fig. 44. The input current to ADC (analog-to-digital converter) has been assumed to be zero. This method needs a controllable current source at the output; this is not easy to obtain. The equivalent analog networks realized by the Brune method and the Bott and Duffin method are shown in Figs. 45 and 46 [Ref. 24].

An alternative method of realizing the driving-point impedance function is suggested below.

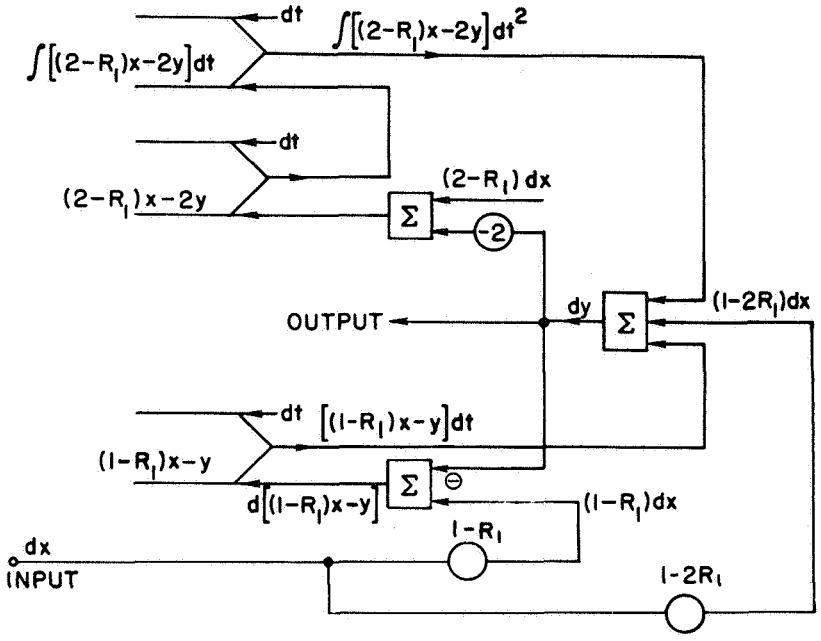


Fig. 43. REALIZATION OF
$G(s) = [s^2(1-2R_1) + s(1-R_1) + (2-R_1)] / (s^2 + s + 2)$.

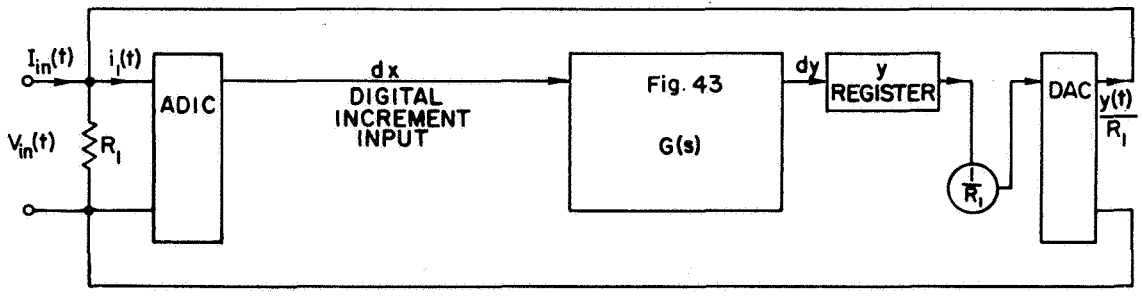


Fig. 44. REALIZATION OF $Z_1(s) = (s^2 + s + 2)/(2s^2 + s + 1)$.

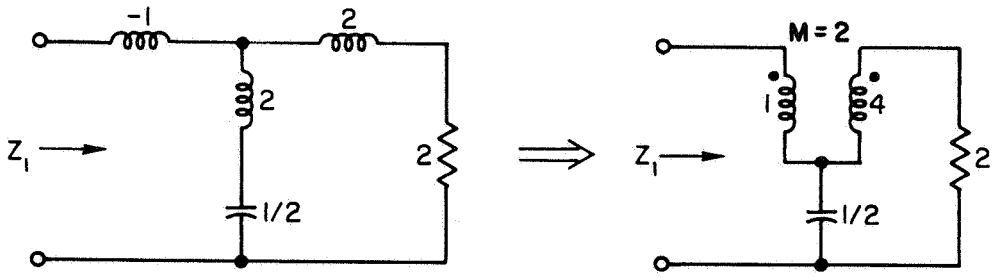


Fig. 45. BRUNE NETWORK REALIZATION OF $Z_1(s) = (s^2 + s + 2)/(2s^2 + s + 1)$.

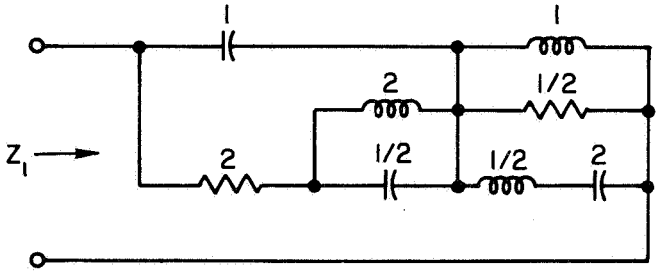


Fig. 46. THE BOTT AND DUFFIN REALIZATION OF $Z_1(s) = (s^2 + s + 2)/(2s^2 + s + 1)$.

Changing the configuration of Fig. 42 to Fig. 47, define the voltage transfer function as

$$G_v(s) = \frac{V_2(s)}{V_{in}(s)} \tag{6.6}$$

and assume that the output voltage has a very small output impedance such that

$$I_2(s) = \frac{V_2(s) - V_{in}(s)}{R_2} \tag{6.7}$$

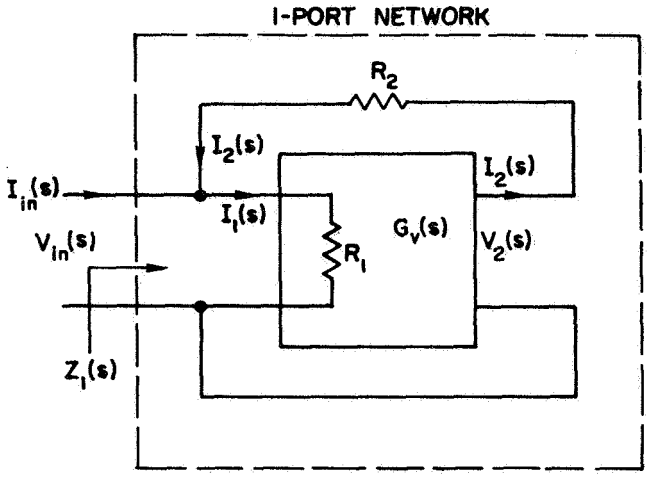


Fig. 47. EQUIVALENT OF FIG. 42.

From Eq. (6.5)

$$G_I(s) = 1 - \frac{R_1}{Z_1(s)} \tag{6.5}$$

then

$$G_I(s) = \frac{I_2(s)}{I_1(s)} = \frac{[V_2(s) - V_{in}(s)]}{V_{in}(s)/R_1} = \frac{R_1}{R_2}\left[\frac{V_2(s)}{V_{in}(s)} - 1\right]$$

$$= \frac{R_1}{R_2}[G_v(s) - 1] = 1 - \frac{R_1}{Z_1(s)} \tag{6.8}$$

Solving for $G_v(s)$ obtains

$$G_v(s) = \frac{R_2}{R_1} + 1 - \frac{R_2}{Z_1(s)} \tag{6.9}$$

With $Z_1(s)$ specified, $G_v(s)$ can be found, and $Z_1(s)$ can be realized without difficulty by interconnecting the digital integrators.

By this alternative method, the network of Example 11 can also be realized as follows:

$$G_v(s) = \frac{R_2}{R_1} + 1 - \frac{R_2(2\,s^2 + s + 1)}{s^2 + s + 2}$$

$$= 1 + \frac{R_2}{R_1} - 2R_2 + \frac{R_2 s + R_2}{s^2 + s + 2} = \frac{V_2(s)}{V_{in}(s)} = \frac{Y(s)}{X(s)}$$

$$y(t) = y_1(t) + y_2(t)$$

where

$$y_1(t) = \left(1 + \frac{R_2}{R_1} - 2R_2\right) x(t)$$

$y_2(t)$ = the solution of

$$\dot{dy}_2(t) = R_2\, d\dot{x}(t) + R_2\, x(t)\, dt - dy_2(t) - 2y_2(t)\, dt$$

The realized network is shown in Fig. 48.

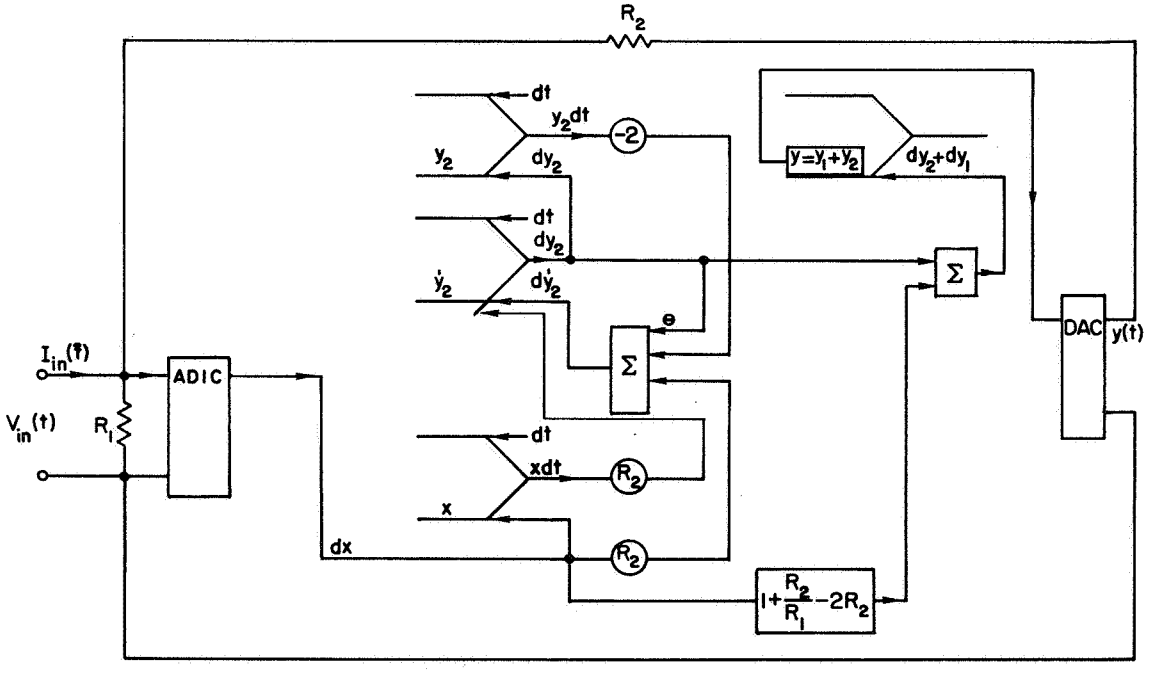


Fig. 48. ALTERNATE REALIZATION OF $Z_1(s)$.

B. Driving-Point Admittance-Function Realization

The driving-point admittance function $Y_1(s)$ of a network is defined as the ratio of $I_{in}(s)$ to $V_{in}(s)$, where $I_{in}(s)$ and $V_{in}(s)$ are, respectively, the input current and input voltage in Laplace transform form, namely,

$$Y_1(s) = \frac{I_{in}(s)}{V_{in}(s)} \tag{6.10}$$

In contrast to the last section, consider a 1-port network, now using the voltage feedback rather than the current feedback because the input is assumed to be a current-controlled source.

Referring to Fig. 48, define an internal-voltage transfer function H(s), such that

$$H(s) = \frac{V_2(s)}{V_1(s)} \tag{6.11}$$

Because $V_1(s) = V_{in}(s) + V_2(s)$,

$$Y_1(s) = \frac{I_{in}(s)}{V_{in}(s)} = \frac{I_{in}(s)}{V_1(s) - V_2(s)}$$

$$= \frac{I_{in}(s)}{V_1(s)} [1 - H(s)]$$

or

$$H(s) = 1 - \frac{I_{in}(s)}{V_1(s)\ Y_1(s)} \tag{6.12}$$

If a fixed resistor R_1' has been connected across terminals 1-2', then

$$\frac{I_{in}(s)}{V_1(s)} = \frac{1}{R_1'} \tag{6.13}$$

thus,

$$H(s) = 1 - \frac{1}{R_1' \, Y_1(s)} \tag{6.14}$$

which is similar to Eq. (6.5). By the methods described in the last section, the transfer function $H(s)$ can be easily realized by using digital integrators. With $H(s)$ realized, the desired driving-point admittance $Y_1(s)$ can be obtained by connecting $H(s)$, as shown in Fig. 49.

Example 12.

Realize the following driving-point admittance function

$$Y_1(s) = \frac{s^2 + s + 2}{2s^2 + s + 2} = \frac{I_{in}(s)}{V_{in}(s)}$$

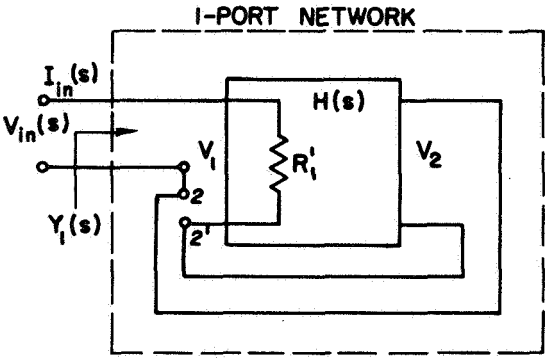


Fig. 49. FEEDBACK CONNECTION USED TO REALIZE DRIVING-POINT ADMITTANCE FUNCTION.

Solution. First, find the transfer function $H(s)$ corresponding to the given $Y_1(s)$:

$$H(s) = 1 - \frac{1}{R_1' \, Y_1(s)} = \frac{(R_1' - 2)s^2 + (R_1' - 1)s + (2R_1' - 1)}{R_1'(s^2 + s + 2)} = \frac{V_2(s)}{V_1(s)}$$

Let $V_2(s) = Y(s)$ and $V_1(s) = X(s)$, then

$$R_1'(s^2 + s + 2)\, Y(s) = \left[(R_1' - 2)s^2 + (R_1' - 1)s + (2R_1' - 1)\right] X(s)$$

or

$$\left(1 + \frac{1}{s} + \frac{2}{s^2}\right) Y(s) = \left[\left(1 - \frac{2}{R_1'}\right) + \left(1 - \frac{1}{R_1'}\right)\frac{1}{s} + \left(2 - \frac{1}{R_1'}\right)\frac{1}{s^2}\right] X(s)$$

Transforming back to the time domain yields

$$y(t) = x(t) + \int \left[\left(1 - \frac{1}{R_1'}\right)x(t) - y(t)\right] \, dt + \int \left[\left(2 - \frac{1}{R_1'}\right)x(t) - 2y(t)\right] \, dt^2$$

or

$$dy = dx + \left[\left(1 - \frac{1}{R_1'}\right)x - y\right] dt + \int \left[\left(2 - \frac{1}{R_1'}\right)x - 2y\right] dt^2$$

The realized network of $H(s)$ and $Y_1(s)$ are given in Figs. 50 and 51, respectively.

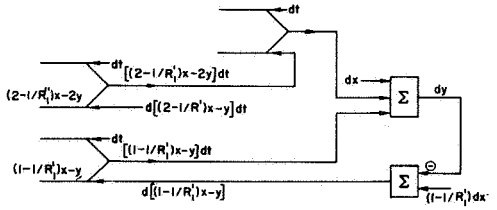


Fig. 50. REALIZATION OF $H(s) = \dfrac{(R_1'-2)s^2 + (R_1'-1)s + (2R_1'-1)}{R_1'(s^2 + s + 2)}$.

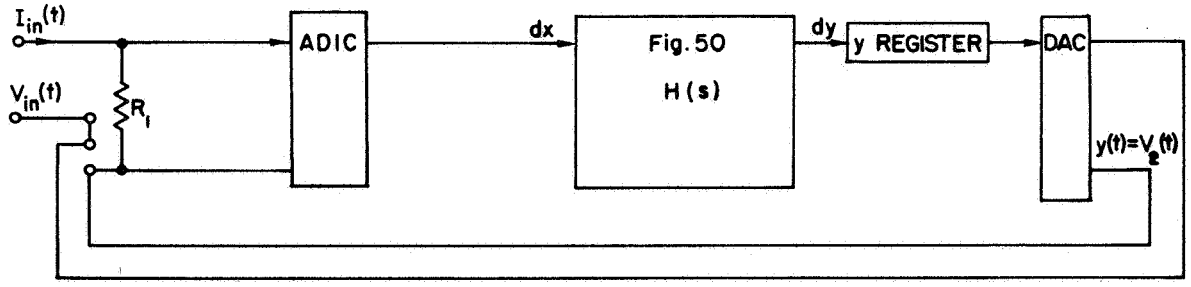


Fig. 51. REALIZATION OF $Y_1(s) = (s^2 + s + 2)/(2s^2 + s + 1)$.

An interesting result can be drawn from Examples 11 and 12, where it is seen that if the values of R_1 and R_1' are normalized or made $R_1 = R_1' = 1$ ohm, then $G(s)$ is exactly the same as $H(s)$. Therefore, the same function can be realized as either an admittance or an impedance function, depending on how the integrators are connected.

The following two examples show the realization of a single inductor and a single capacitor by use of digital building blocks.

Example 13.

Realize $Z_1(s) = sL$ by using the above techniques.

Solution. By using Eq. (6.5),

$$G(s) = 1 - \frac{R_1}{Z_1(s)} = 1 - \frac{R_1}{sL} = \frac{sL - R_1}{sL} = \frac{I_2(s)}{I_1(s)} = \frac{R_1 I_2(s)}{V_{in}(s)}$$

Let

$$Y(s) = R_1 I_2(s) \qquad X(s) = V_{in}(s)$$

then

$$Y(s) = \left(1 - \frac{R_1}{sL}\right) X(s)$$

In the time domain:

$$y(t) = x(t) - \frac{R_1}{L} \int x(t)\, dt$$

or

$$dy = dx - \frac{R_1}{L}\, x\, dt$$

The network of $G(s)$ can be realized, as in Fig. 52, and the single inductor of value L can be realized as in Fig. 53.

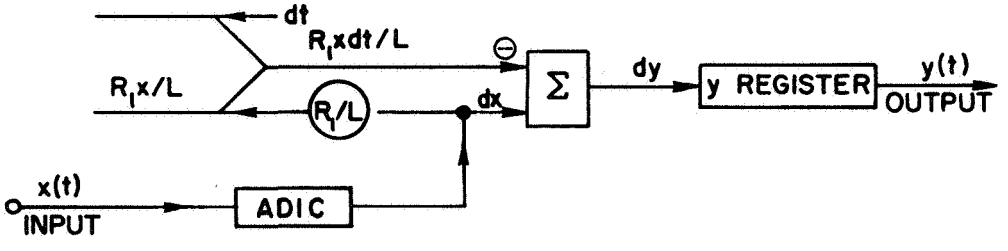


Fig. 52. REALIZATION OF $G(s) = (sL - R_1)/sL$.

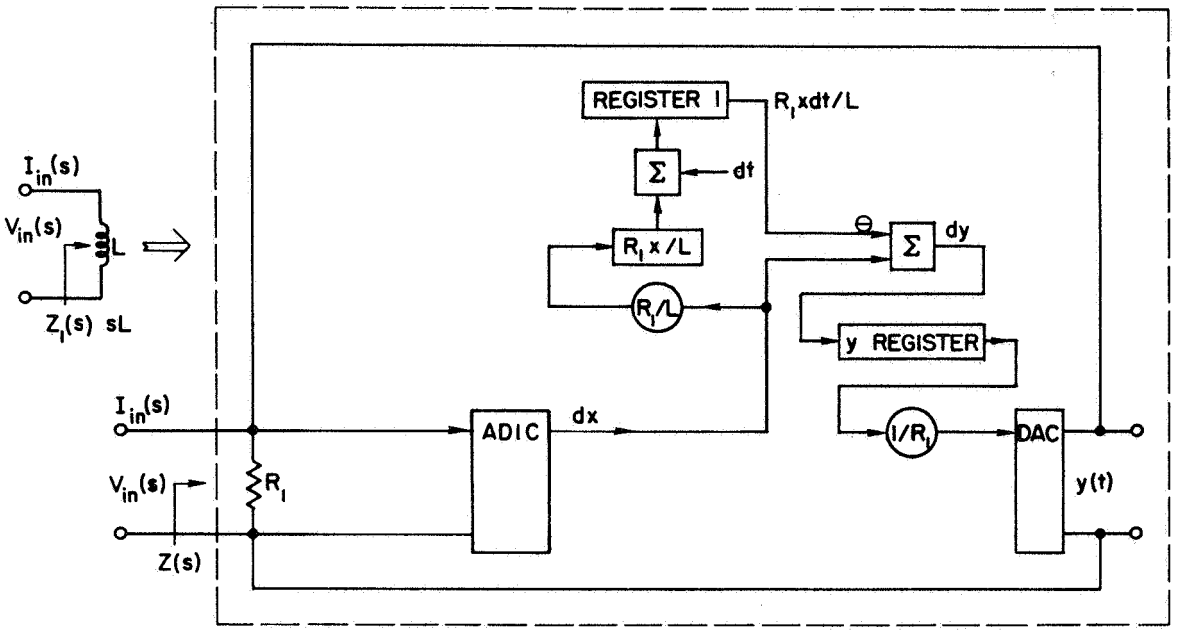


Fig. 53. REALIZATION OF A SINGLE INDUCTOR, $Z_1(s) = sL$.

Note that if $x(t)$ is a sinusoidal function whose time integral stays finite, the output $y(t)$

$$y(t) = x(t) - \frac{1}{R_1'C} \int x(t)\, dt$$

will remain finite; otherwise, the integral grows increasingly larger and eventually will cause overflow in the $y(t)$ register which, in turn, will not perform the correct operation. From another point of view, the current $y(t)/R_1$, flowing through the inductor, will increase indefinitely after a step voltage $x(t) = \text{constant}$ is applied:

$$i_L(t) = \frac{1}{L} \int v(t)\, dt$$

Example 14.

Realize a single capacitor $Y_1(s) = sC$.

Solution. By using Eq. (6.14),

$$H(s) = 1 - \frac{1}{R_1' Y_1(s)} = \frac{R_1' sC - 1}{R_1' sC} = \frac{V_2(s)}{V_1(s)} = \frac{Y(s)}{X(s)}$$

$$Y(s) = \left(1 - \frac{1}{R_1' sC}\right) X(s)$$

In the time domain:

$$y(t) = x(t) - \frac{1}{R_1' C} \int x(t)\, dt \quad \text{or} \quad dy = dx - \left(\frac{1}{R_1' C}\right) x\, dt$$

The network of $H(s)$ can be realized, as shown in Fig. 54. With $H(s)$ realized, the single capacitor C can be easily obtained, as shown in Fig. 55.

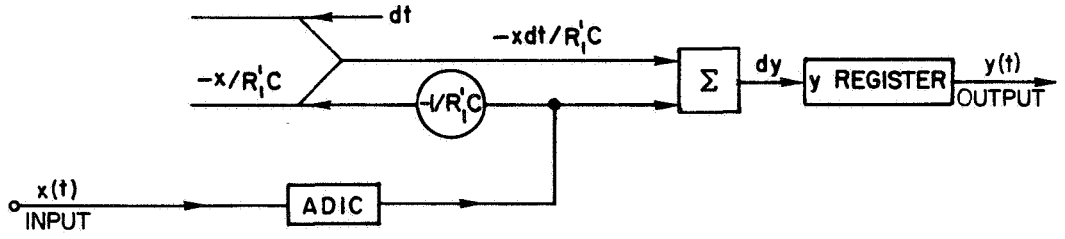


Fig. 54. REALIZATION OF $H(s) = (sR_1' - 1)/sR_1'C = Y(s)/X(s)$.

From Example 13 it is seen that with the length of the registers sufficiently long, an almost ideal inductor could be theoretically constructed; that is, a circuit with a very high Q can be obtained. From the realization techniques presented in this report, it is obvious that any driving-point immittance function with negative values or with poles in the right-half of the complex frequency plane can be realized without additional effort.

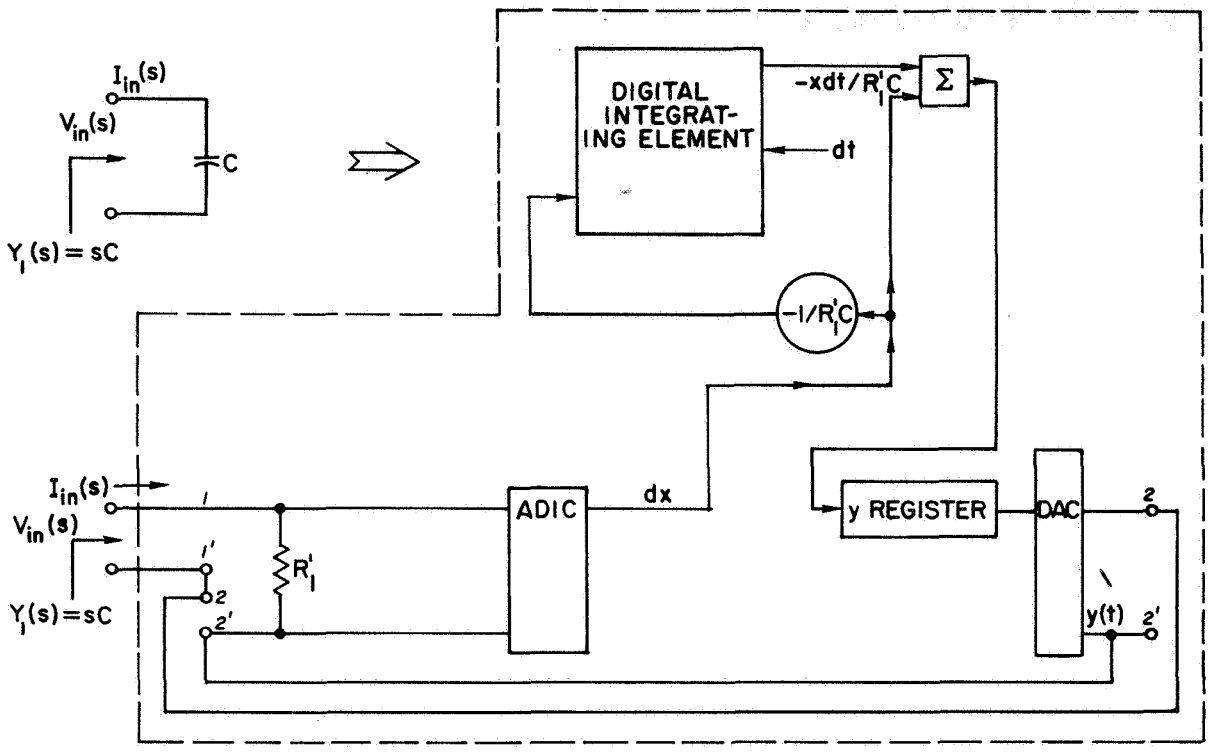


Fig. 55. REALIZATION OF A SINGLE CAPACITOR.

Example 15.

Realize the following immittance functions:

a. $Z_1(s) = -s$ (negative inductor, $L = -1$)

b. $Y_1(s) = s-1$ (tunnel diode, $C = 1$, $R = -1$)

Solution.

a. Using Eq. (6.9)

$$G_v(s) = \frac{R_2}{R_1} + 1 - \frac{R_2}{-s} = \left(1 + \frac{R_2}{R_1}\right) + \frac{R_2}{s} = \frac{V_2(s)}{V_{in}(s)} = \frac{Y(s)}{X(s)}$$

$$Y(s) = \left(1 + \frac{R_2}{R_1}\right) X(s) + \frac{R_2 X(s)}{s}$$

Transforming back to the time domain:

$$y(t) = \left(1 + \frac{R_2}{R_1}\right) x(t) + R_2 \int x(t)\, dt$$

or

$$dy = \left(1 + \frac{R_2}{R_1}\right) dx + R_2\, x\, dt$$

The network with the transfer function $G_v(s)$ can be designed easily, as shown in Fig. 56; with $G_v(s)$ realized, the negative inductor can be obtained, as shown in Fig. 57.

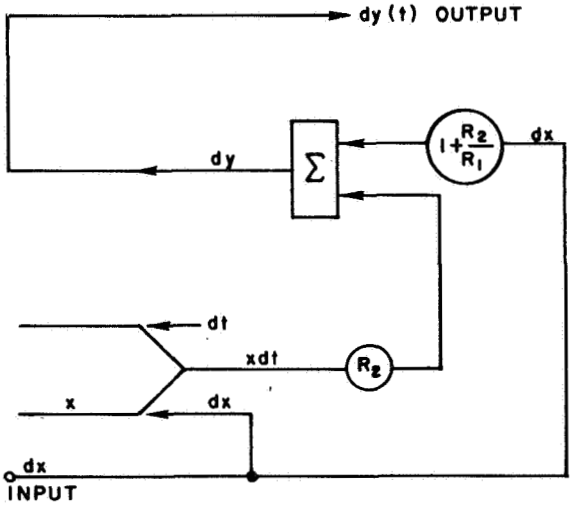


Fig. 56. REALIZATION OF $G_v(s) = (1 + R_2/R_1) + R_2/s$.

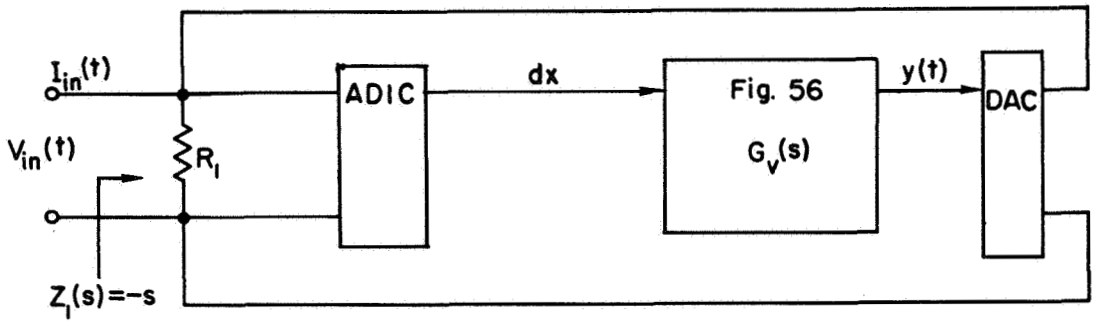


Fig. 57. REALIZATION OF $Z_1(s) = -s$.

b. Using Eq. (6.14),

$$H(s) = 1 - \frac{1}{R_1'(s-1)} = \frac{V_2(s)}{V_1(s)} = \frac{Y(s)}{X(s)}$$

then

$$Y(s) = \left(\frac{R_1's - R_1' - 1}{R_1's - R_1'}\right) X(s) = \frac{1 - (1/s)[1 + (1/R_1')]}{1 - (1/s)}$$

In the time domain:

$$y(t) = x(t) + \int \left[y(t) - \left(1 + \frac{1}{R_1'}\right) x(t)\right] dt$$

or

$$dy = dx + \left[y - \left(1 + \frac{1}{R_1'}\right) X\right] dt$$

The network with $H(s) = 1 - 1/[R_1'(s-1)]$ can be realized, as can the tunnel diode (see Figs. 58 and 59).

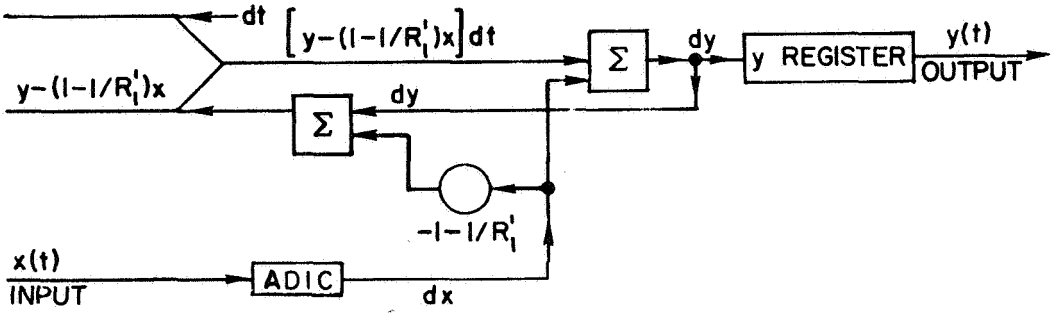


Fig. 58. NETWORK OF $H(s) = 1 - 1/[R_1'(s-1)]$.

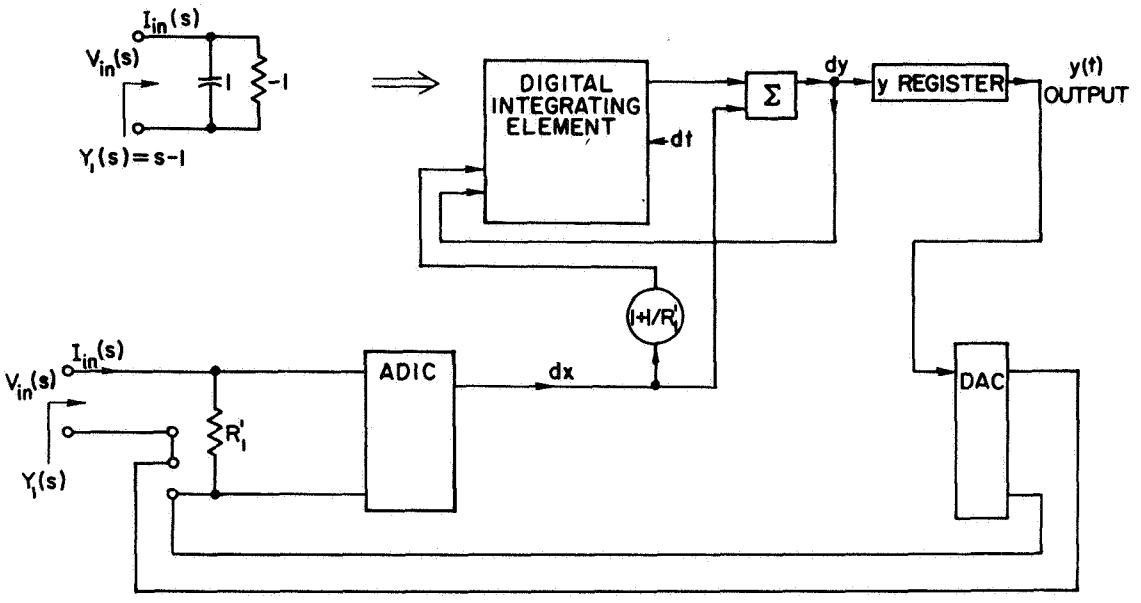


Fig. 59. NETWORK OF $Y_1(s) = s - 1$.

Chapter VII

CONCLUSION

A. Summary of Results

The use of integrated digital building blocks to realize network functions has been initiated, and the feasibility has been studied. Two types of modules (digital integrator and adder) have been proposed as the semi-universal building blocks to construct the network functions whose inputs and outputs are digital increments. Analog signals also can be handled by the analog-to-digital incremental converter and the digital-to-analog converter.

For high-speed operation, the proposed digital integrator has been implemented by the modified trapezoidal-integration method and by the signed-digit number system. A technique to obtain variable precision is also achieved.

Optimization is performed on the digital modules, subject to a minimum-cost criterion, resulting in a synthesis procedure for obtaining network realization with a minimum number of digital modules as well as with best performance. Essentially, the transfer function can be realized by the digital modules alone; however, the immittance function or the conventional-element replacement can only be made possible with the help of the analog-to-digital incremental converter and digital-to-analog converter.

The idea of using digital incremental data as the only information transfer in the system can be applied to the present prevailing digital-computer realization technique. The principal advantages are small round-off error and more accuracy. The application of incrementals to digital-computer realization of the transfer function is discussed in Appendix H.

B. Discussion

1. Advantages

The advantages of the realization of network functions by using digital elements are

a. small physical size (integrated circuits) in which size is independent of frequency, especially at $f < 1$ Hz

b. no inductor or capacitor

c. no realizability problems, such as negative elements or multiplicity of poles

d. easy implementation of linear and nonlinear networks

e. no drift problems (center frequency can be steadily maintained)

f. improvement of accuracy by increasing lengths of the registers

g. almost no difficulty in selecting, specifying, and storing components

h. easily made modifications to parameters of the circuit to fit individual needs with no physical change.

The advantages of the two-element module are

a. Digital integrating elements can be cascaded to increase precision.

b. Operation is high speed, handling signal input frequencies up to hundreds of kilohertz, and the integrator operates at one iteration per cycle.

c. The digital integrating element can be used as a constant multiplier and an incremental multiplier.

d. The summing element can also be used as an output multiplier (anywhere between -7 to 7 times).

e. There is a high repeatability of cells inside the module.

f. Only two kinds of elements are needed to realize the whole class of network functions.

g. There is a high gate-to-pin ratio.

2. Comparisons between Analog and Digital Realizations

It is of interest to compare the digital realization of network functions with those discussed in the newly published book, Active Integrated Network Synthesis [Ref. 11]:

a. The network functions in Ref.11 are realized primarily by using one of three types of integrated modules--integrated gyrator, operational amplifier, or negative impedance converter (excluding the distributed networks)--as the basic building blocks to deal with analog signals of frequencies ranging from dc to the lower part of the MHz region. In contrast, the basic building block in this research is the digital two-element module, and the frequencies of the analog signal range from dc to a few hundred kHz with the help of the ADIC (analog-to-digital incremental converter) and the DAC (digital-to-analog converter).

b. In comparison to analog realization in the operating frequency range of the digital hardware, digital realization has no parasitics problem, no need for temperature compensation, no critical component problem, no repeatability problem of the solution, no drift problem, and it is more accurate.

c. Optimum analog realization would involve minimization of the number of capacitors, whereas the optimum digital realization would involve a minimum number of digital modules.

d. Time is not the only independent variable in the digital realization; hence, nonlinear networks can also be realized. In the analog domain, time is the only independent variable.

e. To increase reliability, redundant circuits or majority voting logics can be very helpful, but none of these will help the analog realization.

f. Coefficient setting will be a problem in both analog and digital realizations. In analog realization, unless very accurate integrated potentiometers can be made, a large number of different-valued precision resistors will be required. In digital realization, the coefficient-setting problem is different and will be discussed in Section C.

g. In both analog and digital realizations, cascade synthesis offers the best performance and lowest sensitivity.

h. Variable precision can be achieved in digital realization.

i. Gyrator conductance must be changed from case to case. Identical (unique) gyrators cannot be achieved unless some external means can be applied to change the gyrator conductance. The same problem exists in the uniqueness of the capacitance value.

3. Cost

Cost is always a major concern. In microelectronics, a single chip can be used to realize a trivial or a complicated Boolean function at essentially the same expense; therefore, the Quine-McCluskey simplification method to reduce the number of gates in a Boolean function may not, in fact, reduce the cost of the chip. Concerning fabrication cost, the optimum number of gates per chip varies with the passage of time (50 gates/chip in 1965, 70 gates/chip in 1966, 1000 gates/chip in 1970, and perhaps 5000 gates/chip in 1972) [Refs. 18 and 19]. Thus, the few hundred gates in the proposed digital integrating element may seem awesome,

but as technology further advances, the two-element module will become cheaper. One factor that could help to cut down the price is the tremendous quantities of modules needed for network construction.

4. The Signed-Digit Number System

In the design of the digital two-element module, the reasons for choosing the signed-digit number system over the conventional one are threefold: (1) the carry-propagation chain is eliminated, thus speeding up the internal addition and subtraction operations; (2) the data that are transferred, external to the two-element module, are incremental data that can be easily and quickly converted to the conventional number system, if required; and (3) if higher precision is needed propagation takes no longer time, and the complexity of the logic gate structure is not increased.

5. Current-Mode Circuits

The current-mode switching circuitry can be used to implement logic-gate circuitry because current-mode circuits have the potential for nanosecond systems [Ref. 27].

6. Prewired Module Packages

It is also possible to use prewired (on a printed circuit board) modules to form any second-, third- or fourth-order transfer-function packages. The values of the constant multipliers (the coefficients of the transfer function) can be either fixed for specific applications or adjusted to fit any case.

C. Recommendations for Further Study

This investigation has been primarily a feasibility study concerned with how the realization and design of network functions utilizing variable-precision digital modules can be achieved. The proposed two-element module and the internal structure of the modules lay the groundwork for future hardware design. Development and testing of prototype digital two-element modules would be useful in deriving further information about this system.

Listed below are suggestions for future research, based on the studies described in this report:

1. Optimization on the structure of digital modules can be reached if more exact information and constraints concerning the relationships of the number of pins, gates, and cost can be found and specified. This optimization is subject also to the minimum delay in the modules.

2. A new technique is needed to discover how the coefficient setting of any transfer function can be done so that the setting will be fairly easy, and once set, the coefficient will remain unchanged at some later power-on-condition, regardless of how many times the power has been turned off. One feasible method is to connect more test-point pins (not input-output pins) to the Y-registers of the digital integrator, which can be externally accessible by a specially designed tool so that the setting of the coefficients can be easily performed after each power off-on sequence.

3. To build a powerful high-speed system, it is feasible to design either a hybrid system involving the proposed integrated digital modules and those integrated analog modules discussed in Ref. 11 or a combined digital system using the proposed digital modules and the existing general-purpose digital computer.

Appendix A

THE ANALOG-TO-DIGITAL INCREMENTAL CONVERTER

The analog-to-digital incremental converter (ADIC) is a device that converts the difference between two analog quantities to digital form. In particular, if an analog signal is applied to the input of the ADIC, the difference, or increment, of the analog quantities measured at two consecutive bit times is converted to digital form.

One implementation is shown in Fig. 60, where the operational amplifiers are employed to obtain the difference between the signals at two consecutive bit times. If available, a difference amplifier can be used to replace the two operational amplifiers. The gain of the amplifiers does not have to be unity. When necessary, gain adjustments can be made to fit the analog-to-digital converter (ADC) input levels.

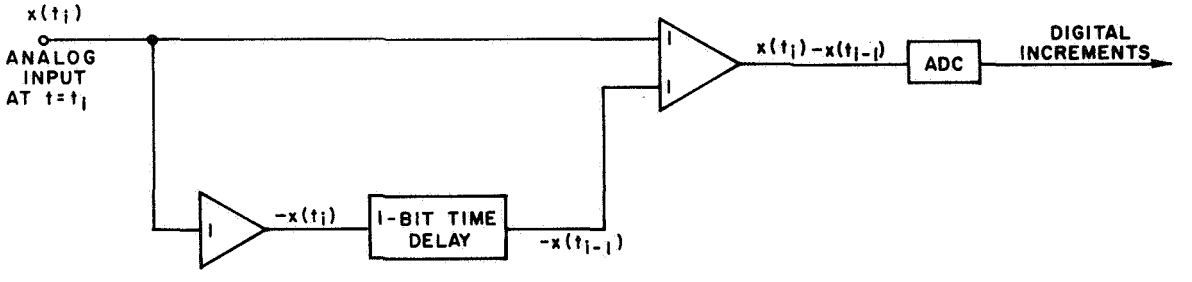


Fig. 60. ONE IMPLEMENTATION OF ADIC.

Similarly, another possible implementation is shown in Fig. 61, where the analog signal is converted to digital form first, then subtracted from the previous digital quantity by a digital adder-subtractor. The difference output is the digital increment.

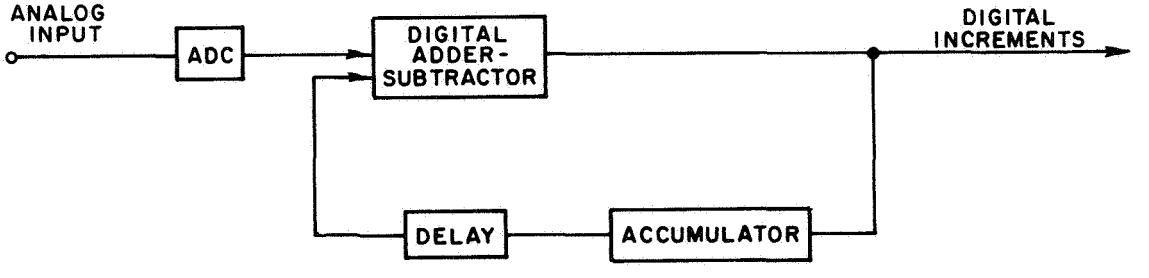


Fig. 61. ANOTHER IMPLEMENTATION OF ADIC.

Of the two implementations, the first (Fig. 60) is preferred and a few of its advantages can be noted. The input level to the ADC is limited; therefore, fewer comparators are needed and more accuracy can be achieved. The conversion time of the ADIC in Fig. 60 is much less than that of the second implementation because only increments are transferred rather than full words. Also, the difference between the two analog quantities can be obtained immediately from the difference amplifier, whereas digital subtraction takes time.

Appendix B

LOGIC DESIGN OF THE DIGITAL INTEGRATING ELEMENT

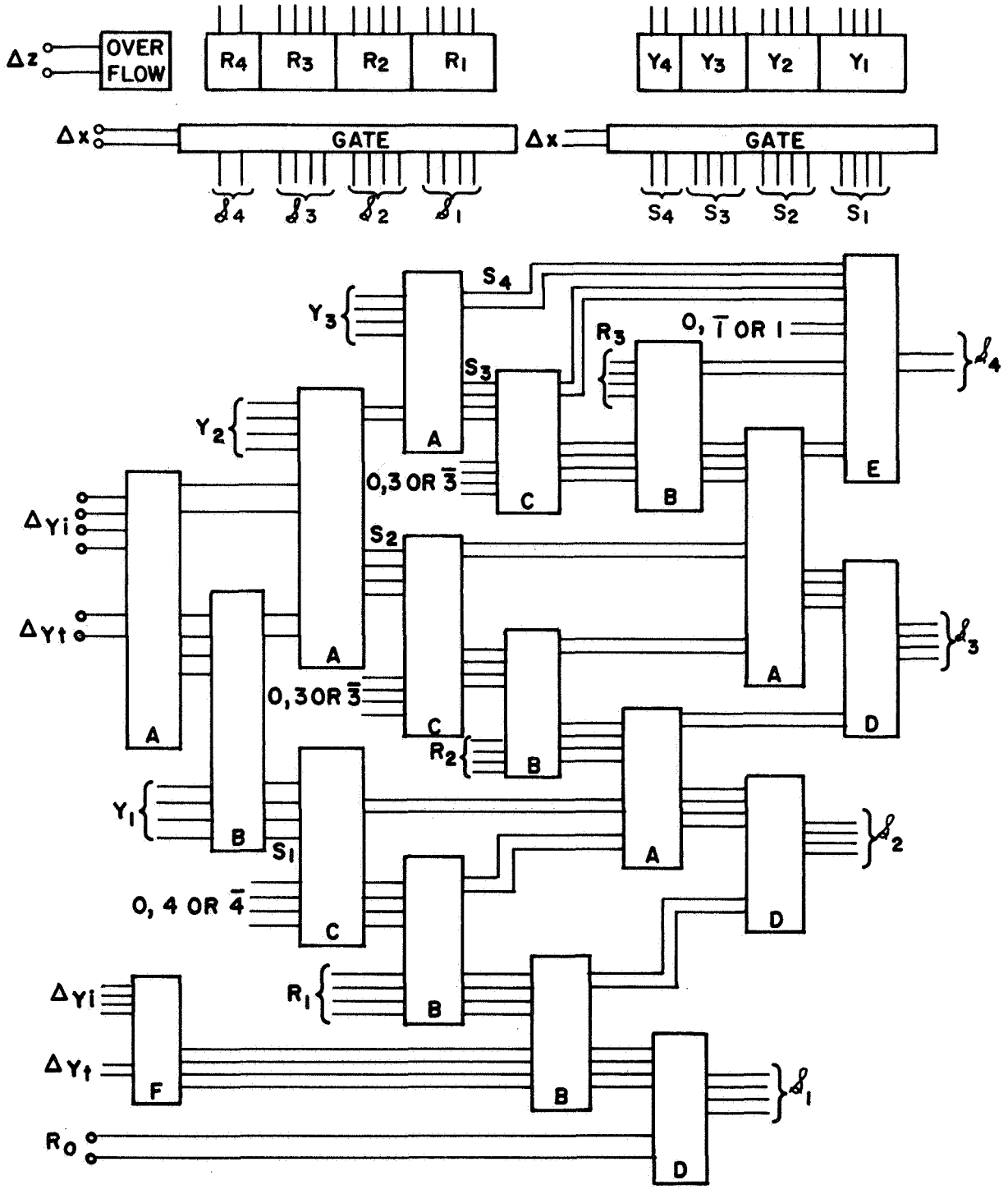


Fig. 62. INTERNAL BLOCK STRUCTURE OF DIGITAL INTEGRATING ELEMENT.

1. Logic Design for Cell A

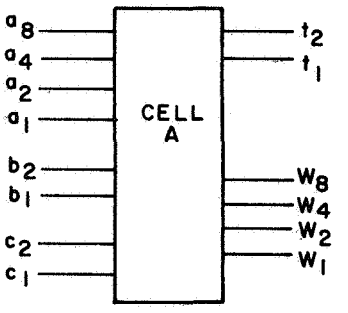


Fig. 63. I/O PINS OF CELL A.

TRUTH TABLE FOR CELL A

b_2	b_1	c_2	c_1	a_8	a_4	a_2	a_1	t_2	t_1	w_8	w_4	w_2	w_1
0	0	0	0	0	0	0	0	0	0	0	0	0	0
				0	0	0	1	0	0	0	0	0	1
				0	0	1	0	0	0	0	0	1	0
				0	0	1	1	0	0	0	0	1	1
				0	1	0	0	0	0	0	1	0	0
				0	1	0	1	0	0	0	1	0	1
				0	1	1	0	0	1	1	1	0	0
				0	1	1	1	0	1	1	1	0	1
				1	1	1	1	0	0	1	1	1	1
				1	1	1	0	0	0	1	1	1	0
				1	1	0	1	0	0	1	1	0	1
				1	1	0	0	0	0	1	1	0	0
				1	0	1	1	0	0	1	0	1	1
				1	0	1	0	1	1	0	1	0	0
				1	0	0	1	1	1	0	0	1	1
0	0	0	1	0	0	0	0	0	0	0	0	0	1
				0	0	0	1	0	0	0	0	1	0

b_2	b_1	c_2	c_1	a_8	a_4	a_2	a_1	t_2	t_1	w_8	w_4	w_2	w_1
0	0	0	1	0	0	1	0	0	0	0	0	1	1
				0	0	1	1	0	0	0	1	0	0
				0	1	0	0	0	0	0	1	0	1
				0	1	0	1	0	1	1	1	0	0
				0	1	1	0	0	1	1	1	0	1
				0	1	1	1	0	1	1	1	1	0
				1	1	1	1	0	0	0	0	0	0
				1	1	1	0	0	0	1	1	1	1
				1	1	0	1	1	1	1	1	1	0
				1	1	0	0	0	0	1	1	0	1
				1	0	1	1	0	0	1	1	0	0
				1	0	1	0	0	0	1	0	1	1
				1	0	0	1	1	1	0	1	0	0
0	0	1	1	0	0	0	0	0	0	1	1	1	1
				0	0	0	1	0	0	0	0	0	0
				0	0	1	0	0	0	0	0	0	1
				0	0	1	1	0	0	0	0	1	0
				0	1	0	0	0	0	0	0	1	1
				0	1	0	1	0	0	0	1	0	0
				0	1	1	0	0	0	0	1	0	1
				0	1	1	1	0	1	1	1	0	0
				1	1	1	1	0	0	1	1	1	0
				1	1	1	0	0	0	1	1	0	1
				1	1	0	1	0	0	1	1	0	0
				1	1	0	0	0	0	1	0	1	1
				1	0	1	1	1	1	0	1	0	0
				1	0	1	0	1	1	0	0	1	1
				1	0	0	1	1	1	0	0	1	0
0	1	0	0	0	0	0	0	0	0	0	0	0	1
				0	0	0	1	0	0	0	0	1	0
				0	0	1	0	0	0	0	0	1	1
				0	0	1	1	0	0	0	1	0	0
				0	1	0	0	0	0	0	1	0	1

b_2	b_1	c_2	c_1	a_8	a_4	a_2	a_1	t_2	t_1	w_8	w_4	w_2	w_1
0	1	0	0	0	1	0	1	0	1	1	1	0	0
				0	1	1	0	0	1	1	1	0	1
				0	1	1	1	0	1	1	1	1	0
				1	1	1	1	0	0	0	0	0	0
				1	1	1	0	0	0	1	1	1	1
				1	1	0	1	0	0	1	1	1	0
				1	1	0	0	0	0	1	1	0	1
				1	0	1	1	0	0	1	1	0	0
				1	0	1	0	0	0	1	0	1	1
				1	0	0	1	1	1	0	1	0	0
0	1	0	1	0	0	0	0	0	0	0	0	1	0
				0	0	0	1	0	0	0	0	1	1
				0	0	1	0	0	0	0	1	0	0
				0	0	1	1	0	0	0	1	0	1
				0	1	0	0	0	1	1	1	0	0
				0	1	0	1	0	1	1	1	0	1
				0	1	1	0	0	1	1	1	1	0
				0	1	1	1	0	1	1	1	1	1
				1	1	1	1	0	0	0	0	0	1
				1	1	1	0	0	0	0	0	0	0
				1	1	0	1	0	0	1	1	1	1
				1	1	0	0	0	0	1	1	1	0
				1	0	1	1	0	0	1	1	0	1
				1	0	1	0	0	0	1	1	0	0
				1	0	0	1	0	0	1	0	1	1
0	1	1	1	0	0	0	0	0	0	0	0	0	0
				0	0	0	1	0	0	0	0	0	1
				0	0	1	0	0	0	0	0	1	0
				0	0	1	1	0	0	0	0	1	1
				0	1	0	0	0	0	0	1	0	0
				0	1	0	1	0	0	0	1	0	1

b_2	b_1	c_2	c_1	a_8	a_4	a_2	a_1	t_2	t_1	w_8	w_4	w_2	w_1
0	1	1	1	0	1	1	0	0	1	1	1	0	0
				0	1	1	1	0	1	1	1	0	1
				1	1	1	1	0	0	1	1	1	1
				1	1	1	0	0	0	1	1	1	0
				1	1	0	1	0	0	1	1	0	1
				1	1	0	0	0	0	1	1	0	0
				1	0	1	1	0	0	1	0	1	1
				1	0	1	0	1	1	0	1	0	0
				1	0	0	1	1	1	0	0	1	1
1	1	0	0	0	0	0	0	0	0	1	1	1	1
				0	0	0	1	0	0	0	0	0	0
				0	0	1	0	0	0	0	0	0	1
				0	0	1	1	0	0	0	0	1	0
				0	1	0	0	0	0	0	0	1	1
				0	1	0	1	0	0	0	1	0	0
				0	1	1	0	0	0	0	1	0	1
				0	1	1	1	0	1	1	1	0	0
				1	1	1	1	0	0	1	1	1	0
				1	1	1	0	0	0	1	1	0	1
				1	1	0	1	0	0	1	1	0	0
				1	1	0	0	0	0	1	0	1	1
				1	0	1	1	1	1	0	1	0	0
				1	0	1	0	1	1	0	0	1	1
				1	0	0	1	1	1	0	0	1	0
1	1	0	1	0	0	0	0	0	0	0	0	0	0
				0	0	0	1	0	0	0	0	0	1
				0	0	1	0	0	0	0	0	1	0
				0	0	1	1	0	0	0	0	1	1
				0	1	0	0	0	0	0	1	0	0
				0	1	0	1	0	0	0	1	0	1
				0	1	1	0	0	1	1	1	0	0

b_2	b_1	c_2	c_1	a_8	a_4	a_2	a_1	t_2	t_1	w_8	w_4	w_2	w_1
1	1	0	1	0	1	1	1	0	1	1	1	0	1
				1	1	1	1	0	0	1	1	1	1
				1	1	1	0	0	0	1	1	1	0
				1	1	0	1	0	0	1	1	0	1
				1	1	0	0	0	0	1	1	0	0
				1	0	1	1	0	0	1	0	1	1
				1	0	1	0	1	1	0	1	0	0
				1	0	0	1	1	1	0	0	1	1
1	1	1	1	0	0	0	0	0	0	1	1	1	0
				0	0	0	1	0	0	1	1	1	1
				0	0	1	0	0	0	0	0	0	0
				0	0	1	1	0	0	0	0	0	1
				0	1	0	0	0	0	0	0	1	0
				0	1	0	1	0	0	0	0	1	1
				0	1	1	0	0	0	0	1	0	0
				0	1	1	1	0	0	0	1	0	1
				1	1	1	1	0	0	1	1	0	1
				1	1	1	0	0	0	1	1	0	0
				1	1	0	1	0	0	1	0	1	1
				1	1	0	0	1	1	0	1	0	0
				1	0	1	1	1	1	0	0	1	1
				1	0	1	0	1	1	0	0	1	0
				1	0	0	1	1	1	0	0	0	1

Boolean Equations:

$$w_1 = a_1 \oplus b_1 \oplus c_1$$

$$w_2 = \bar{b}_2\bar{b}_1\bar{c}_2\bar{c}_1(\bar{a}_8\bar{a}_4a_2 + a_8a_4a_2 + a_8\bar{a}_4a_1)$$

$$+ \bar{b}_2\bar{b}_1\bar{c}_2c_1(\bar{a}_8\bar{a}_4\bar{a}_2a_1 + \bar{a}_8\bar{a}_4a_2\bar{a}_1 + \bar{a}_8a_4a_2a_1 + a_8a_2\bar{a}_1 + a_8a_4\bar{a}_2a_1)$$

$$+ \bar{b}_2\bar{b}_1c_2c_1(\bar{a}_8\bar{a}_2\bar{a}_1 + a_8\bar{a}_4\bar{a}_2 + a_8\bar{a}_4\bar{a}_1 + a_8\bar{a}_2\bar{a}_1 + \bar{a}_8\bar{a}_4a_2a_1 + a_8a_4a_2a_1)$$

$$+ \bar{b}_2 b_1 \bar{c}_2 \bar{c}_1 (\bar{a}_4 a_2 \bar{a}_1 + \bar{a}_8 \bar{a}_4 + \bar{a}_8 a_4 a_2 a_1 + \bar{a}_8 \bar{a}_4 \bar{a}_2 a_1 + a_8 a_4 a_2 \bar{a}_1 + a_8 a_4 \bar{a}_2 a_1)$$

$$+ \bar{b}_2 b_1 \bar{c}_2 c_1 (\bar{a}_8 \bar{a}_4 \bar{a}_2 + \bar{a}_8 a_4 a_2 + a_8 a_4 \bar{a}_2 + a_8 \bar{a}_4 \bar{a}_2)$$

$$+ \bar{b}_2 b_1 c_2 c_1 (\bar{a}_8 \bar{a}_4 a_2 + a_8 a_4 a_2 + a_8 \bar{a}_4 a_1)$$

$$+ b_2 b_1 \bar{c}_2 \bar{c}_1 (\bar{a}_8 \bar{a}_2 \bar{a}_1 + \bar{a}_8 \bar{a}_4 a_2 a_1 + a_8 a_4 a_2 a_1 + a_8 a_4 \bar{a}_2 + a_8 \bar{a}_4 \bar{a}_1 + a_8 \bar{a}_4 \bar{a}_2)$$

$$+ b_2 b_1 \bar{c}_2 c_1 (\bar{a}_8 \bar{a}_4 a_2 + a_8 a_4 a_2 + a_8 \bar{a}_4 a_1)$$

$$+ b_2 b_1 c_2 c_1 (\bar{a}_8 \bar{a}_2 + a_8 \bar{a}_4 a_2 + a_8 a_4 \bar{a}_2 a_1)$$

$$w_4 = \bar{b}_2 \bar{b}_1 \bar{c}_2 \bar{c}_1 (a_4 + a_8 \bar{a}_4 \bar{a}_1) + \bar{b}_2 \bar{b}_1 \bar{c}_2 c_1 (\bar{a}_8 a_4 + \bar{a}_4 a_2 a_1 + a_8 a_4 \bar{a}_1 + a_8 \bar{a}_2)$$

$$+ \bar{b}_2 \bar{b}_1 c_2 c_1 (\bar{a}_8 \bar{a}_4 \bar{a}_2 \bar{a}_1 + a_4 a_2 + a_4 a_1 + a_8 a_2 a_1)$$

$$+ \bar{b}_2 b_1 \bar{c}_2 \bar{c}_1 (\bar{a}_8 a_4 + \bar{a}_8 a_2 a_1 + a_8 \bar{a}_4 a_1 + a_8 a_4 \bar{a}_2 + a_8 a_4 \bar{a}_1)$$

$$+ \bar{b}_2 b_1 \bar{c}_2 c_1 (\bar{a}_8 a_4 + \bar{a}_4 a_2 + a_4 \bar{a}_2) + \bar{b}_2 b_1 c_2 c_1 (a_4 + a_8 \bar{a}_1)$$

$$+ b_2 b_1 \bar{c}_2 \bar{c}_1 (\bar{a}_8 \bar{a}_4 \bar{a}_2 \bar{a}_1 + a_4 a_2 + a_4 a_1 + a_8 a_2 a_1) + b_2 b_1 \bar{c}_2 c_1 (a_4 + a_8 \bar{a}_1)$$

$$+ b_2 b_1 c_2 c_1 (\bar{a}_8 \bar{a}_4 \bar{a}_2 + a_4 a_2 + a_8 \bar{a}_2 \bar{a}_1)$$

$$w_8 = \bar{b}_2 \bar{b}_1 \bar{c}_2 \bar{c}_1 (a_8 a_4 + a_4 a_2 + a_8 a_2 a_1)$$

$$+ \bar{b}_2 \bar{b}_1 \bar{c}_2 c_1 (\bar{a}_8 a_4 a_2 + a_4 \bar{a}_2 a_1 + a_8 \bar{a}_4 a_2 + a_8 a_4 \bar{a}_1)$$

$$+ \bar{b}_2 \bar{b}_1 c_2 c_1 (\bar{a}_8 \bar{a}_4 \bar{a}_2 \bar{a}_1 + a_4 a_2 a_1 + a_8 a_4)$$

$$+ \bar{b}_2 b_1 \bar{c}_2 \bar{c}_1 (\bar{a}_8 a_4 a_2 + a_8 \bar{a}_4 a_2 + a_8 a_4 \bar{a}_2 + a_4 \bar{a}_2 a_1 + a_4 a_2 \bar{a}_1)$$

$$+ \bar{b}_2 b_1 \bar{c}_2 c_1 (\bar{a}_8 a_4 + a_8 \bar{a}_4 + a_4 \bar{a}_2) + \bar{b}_2 b_1 c_2 c_1 (a_8 a_4 + a_4 a_2 + a_8 a_2 a_1)$$

$$+ b_2 b_1 \bar{c}_2 \bar{c}_1 (\bar{a}_8 \bar{a}_4 \bar{a}_2 \bar{a}_1 + a_8 a_4 + a_4 a_2 a_1) + b_2 b_1 \bar{c}_2 c_1 (a_4 a_2 + a_8 a_4 + a_8 a_2 a_1)$$

$$+ b_2 b_1 c_2 c_1 (\bar{a}_8 \bar{a}_4 \bar{a}_2 + a_8 a_4 a_2 + a_8 a_4 a_1)$$

2. Logic Design for Cell B

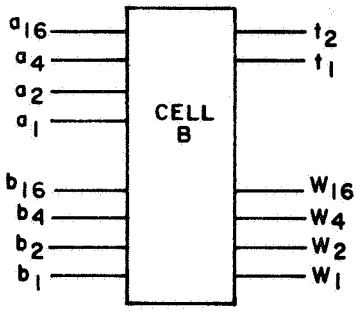


Fig. 64. I/O PINS OF CELL B.

TRUTH TABLE FOR CELL B

a_{16}	a_4	a_2	a_1	b_{16}	b_4	b_2	b_1	t_2	t_1	w_{16}	w_4	w_2	w_1
0	0	0	0	0	0	0	0	0	0	0	0	0	0
				0	0	0	1	0	0	0	0	0	1
				0	0	1	0	0	0	0	0	1	0
				0	0	1	1	0	0	0	0	1	1
				0	1	0	0	0	0	0	1	0	0
				0	1	0	1	0	0	0	1	0	1
				1	1	1	1	0	0	1	1	1	1
				1	1	1	0	0	0	1	1	1	0
				1	1	0	1	0	0	1	1	0	1
				1	1	0	0	0	0	1	1	0	0
				1	0	1	1	0	0	1	0	1	1
0	0	0	1	0	0	0	0	0	0	0	0	0	1
				0	0	0	1	0	0	0	0	1	0
				0	0	1	0	0	0	0	0	1	1
				0	0	1	1	0	0	0	1	0	0
				0	1	0	0	0	0	0	1	0	1
				0	1	0	1	0	1	1	1	0	0
				1	1	1	1	0	0	0	0	0	0
				1	1	1	0	0	0	1	1	1	1
				1	1	0	1	0	0	1	1	1	0
				1	1	0	0	0	0	1	1	0	1
				1	0	1	1	0	0	1	1	0	0

a_{16}	a_4	a_2	a_1	b_{16}	b_4	b_2	b_1	t_2	t_1	w_{16}	w_4	w_2	w_1
0	0	1	0	0	0	0	0	0	0	0	0	1	0
				0	0	0	1	0	0	0	0	1	1
				0	0	1	0	0	0	0	1	0	0
				0	0	1	1	0	0	0	1	0	1
				0	1	0	0	0	1	1	1	0	0
				0	1	0	1	0	1	1	1	0	1
				1	1	1	1	0	0	0	0	0	1
				1	1	1	0	0	0	0	0	0	0
				1	1	0	1	0	0	1	1	1	1
				1	1	0	0	0	0	1	1	1	0
				1	0	1	1	0	0	1	1	0	1
0	0	1	1	0	0	0	0	0	0	0	0	1	1
				0	0	0	1	0	0	0	1	0	0
				0	0	1	0	0	0	0	1	0	1
				0	0	1	1	0	1	1	1	0	0
				0	1	0	0	0	1	1	1	0	1
				0	1	0	1	0	1	1	1	1	0
				1	1	1	1	0	0	0	0	1	0
				1	1	1	0	0	0	0	0	0	1
				1	1	0	1	0	0	0	0	0	0
				1	1	0	0	0	0	1	1	1	1
				1	0	1	1	0	0	1	1	1	0
0	1	0	0	0	0	0	0	0	0	0	1	0	0
				0	0	0	1	0	0	0	1	0	1
				0	0	1	0	0	1	1	1	0	0
				0	0	1	1	0	1	1	1	0	1
				0	1	0	0	0	1	1	1	1	0
				0	1	0	1	0	1	1	1	1	1
				1	1	1	1	0	0	0	0	1	1
				1	1	1	0	0	0	0	0	1	0
				1	1	0	1	0	0	0	0	0	1
				1	1	0	0	0	0	0	0	0	0
				1	0	1	1	0	0	1	1	1	1
0	1	0	1	0	0	0	0	0	0	0	1	0	1

a_{16}	a_4	a_2	a_1	b_{16}	b_4	b_2	b_1	t_2	t_1	w_{16}	w_4	w_2	w_1
0	1	0	1	0	0	0	1	0	1	1	1	0	0
				0	0	1	0	0	1	1	1	0	1
				0	0	1	1	0	1	1	1	1	0
				0	1	0	0	0	1	1	1	1	1
				0	1	0	1	0	1	0	0	0	0
				1	1	1	1	0	0	0	1	0	0
				1	1	1	0	0	0	0	0	1	1
				1	1	0	1	0	0	0	0	1	0
				1	1	0	0	0	0	0	0	0	1
				1	0	1	1	0	0	0	0	0	0
0	1	1	0	0	0	0	0	0	1	1	1	0	0
				0	0	0	1	0	1	1	1	0	1
				0	0	1	0	0	1	1	1	1	0
				0	0	1	1	0	1	1	1	1	1
				0	1	0	0	0	1	0	0	0	0
				0	1	0	1	0	1	0	0	0	1
				1	1	1	1	0	0	0	1	0	1
				1	1	1	0	0	0	0	1	0	0
				1	1	0	1	0	0	0	0	1	1
				1	1	0	0	0	0	0	0	1	0
				1	0	1	1	0	0	0	0	0	1
1	1	1	1	0	0	0	0	0	0	1	1	1	1
				0	0	0	1	0	0	0	0	0	0
				0	0	1	0	0	0	0	0	0	1
				0	0	1	1	0	0	0	0	1	0
				0	1	0	0	0	0	0	0	1	1
				0	1	0	1	0	0	0	1	0	0
				1	1	1	1	0	0	1	1	1	0
				1	1	1	0	0	0	1	1	0	1
				1	1	0	1	0	0	1	1	0	0
				1	1	0	0	0	0	1	0	1	1
				1	0	1	1	1	1	0	1	0	0
1	1	1	0	0	0	0	0	0	0	1	1	1	0
				0	0	0	1	0	0	1	1	1	1

a_{16}	a_4	a_2	a_1	b_{16}	b_4	b_2	b_1	t_2	t_1	w_{16}	w_4	w_2	w_1
1	1	1	0	0	0	1	0	0	0	0	0	0	0
				0	0	1	1	0	0	0	0	0	1
				0	1	0	0	0	0	0	0	1	0
				0	1	0	1	0	0	0	0	1	1
				1	1	1	1	0	0	1	1	0	1
				1	1	1	0	0	0	1	1	0	0
				1	1	0	0	0	0	1	0	1	1
				1	1	0	0	1	1	0	1	0	0
				1	0	1	1	1	1	0	0	1	1
1	1	0	1	0	0	0	0	0	0	1	1	0	1
				0	0	0	1	0	0	1	1	1	0
				0	0	1	0	0	0	1	1	1	1
				0	0	1	1	0	0	0	0	0	0
				0	1	0	0	0	0	0	0	0	1
				0	1	0	1	0	0	0	0	1	0
				1	1	1	1	0	0	1	1	0	0
				1	1	1	0	0	0	1	0	1	1
				1	1	0	1	1	1	0	1	0	0
				1	1	0	0	1	1	0	0	1	1
				1	0	1	1	1	1	0	0	1	0
1	1	0	0	0	0	0	0	0	0	1	1	0	0
				0	0	0	1	0	0	1	1	0	1
				0	0	1	0	0	0	1	1	1	0
				0	0	1	1	0	0	1	1	1	1
				0	1	0	0	0	0	0	0	0	0
				0	1	0	1	0	0	0	0	0	1
				1	1	1	1	0	0	1	0	1	1
				1	1	1	0	1	1	0	1	0	0
				1	1	0	1	1	1	0	0	1	1
				1	1	0	0	1	1	0	0	1	0
				1	0	1	1	1	1	0	0	0	1
1	0	1	1	0	0	0	0	0	0	1	0	1	1
				0	0	0	1	0	0	1	1	0	0
				0	0	1	0	0	0	1	1	0	1

a_{16}	a_4	a_2	a_1	b_{16}	b_4	b_2	b_1	t_2	t_1	w_{16}	w_4	w_2	w_1
1	0	1	1	0	0	1	1	0	0	1	1	1	0
				0	1	0	0	0	0	1	1	1	1
				0	1	0	1	0	0	0	0	0	0
				1	1	1	1	1	1	0	1	0	0
				1	1	1	0	1	1	0	0	1	1
				1	1	0	1	1	1	0	0	1	0
				1	1	0	0	1	1	0	0	0	1
				1	0	1	1	1	1	0	0	0	0
1	0	1	0	0	0	0	0	1	1	0	1	0	0
				0	0	0	1	0	0	1	0	1	1
				0	0	1	0	0	0	1	1	0	0
				0	0	1	1	0	0	1	1	0	1
				0	1	0	0	0	0	1	1	1	0
				0	1	0	1	0	0	1	1	1	1
				1	1	1	1	1	1	0	0	1	1
				1	1	1	0	1	1	0	0	1	0
				1	1	0	1	1	1	0	0	0	1
				1	1	0	0	1	1	0	0	0	0
				1	0	1	1	1	1	1	1	1	1

3. Logic Design for Cell C

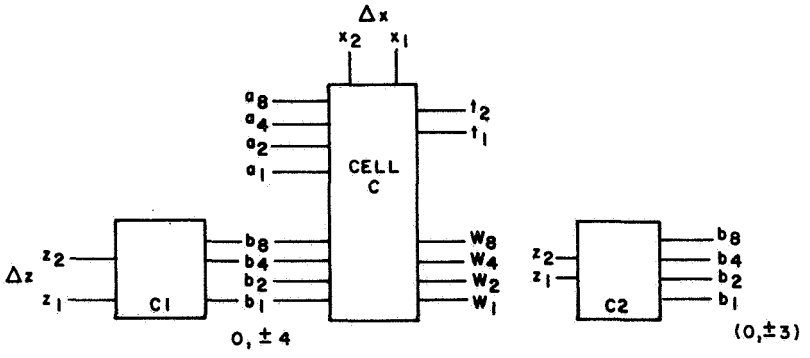


Fig. 65. I/O PINS OF CELL C.

Note that $b_8b_4b_2b_1$ has values of decimal 0, ±3, or ±4, and its value depends on z_2z_1.

TRUTH TABLE FOR CELL C-1

z_2	z_1	b_8	b_4	b_2	b_1
0	0	0	0	0	0
0	1	0	1	0	0
1	1	1	1	0	0

For Cell C-1

$$\begin{cases} b_8 = z_2 \\ b_4 = z_1 \\ b_2 = b_1 = 0 \end{cases}$$

TRUTH TABLE FOR CELL C-2

z_2	z_1	b_8	b_4	b_2	b_1
0	0	0	0	0	0
0	1	0	0	1	1
1	1	1	1	0	1

For Cell C-2

$$\begin{cases} b_8 = b_4 = z_2 \\ b_2 = \bar{z}_2 z_1 \\ b_1 = z_1 \end{cases}$$

TRUTH TABLE FOR CELL C

$x_2x_1 = 01$													$x_2x_1 = 11$					
b_8	b_4	b_2	b_1	a_8	a_4	a_2	a_1	t_2	t_1	w_8	w_4	w_2	w_1	t_2	t_1	w_8	w_4	w_2	w_1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
				0	0	0	1	0	0	0	0	0	1	0	0	1	1	1	1
				0	0	1	0	0	0	0	0	1	0	0	0	1	1	1	0
				0	0	1	1	0	0	0	0	1	1	0	0	1	1	0	1
				0	1	0	0	0	0	0	1	0	0	0	0	1	1	0	0
				0	1	0	1	0	0	0	1	0	1	0	0	1	0	1	1
				1	1	1	1	0	0	1	1	1	1	0	0	0	0	0	1
				1	1	1	0	0	0	1	1	1	0	0	0	0	0	1	0
				1	1	0	1	0	0	1	1	0	1	0	0	0	0	1	1
				1	1	0	0	0	0	1	1	0	0	0	0	0	1	0	0
				1	0	1	1	0	0	1	0	1	1	0	0	0	1	0	1
0	0	1	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	1
				0	0	0	1	0	0	0	1	0	0	0	0	0	0	1	0

$x_2x_1 = 01$														$x_2x_1 = 11$					
b_8	b_4	b_2	b_1	a_8	a_4	a_2	a_1	t_2	t_1	w_8	w_4	w_2	w_1	t_2	t_1	w_8	w_4	w_2	w_1
0	0	1	1	0	0	1	0	0	0	0	1	0	1	0	0	0	0	0	1
				0	0	1	1	0	1	1	1	0	0	0	0	0	0	0	0
				0	1	0	0	0	1	1	1	0	1	0	0	1	1	1	1
				0	1	0	1	0	1	1	1	1	0	0	0	1	1	1	0
				1	1	1	1	0	0	0	0	1	0	0	0	0	1	0	0
				1	1	1	0	0	0	0	0	0	1	0	0	0	1	0	1
				1	1	0	1	0	0	0	0	0	0	0	1	1	1	0	0
				1	1	0	0	0	0	1	1	1	1	0	1	1	1	0	1
				1	0	1	1	0	0	1	1	1	0	0	1	1	1	1	0
0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0
				0	0	0	1	0	0	0	1	0	1	0	0	0	0	1	1
				0	0	1	0	0	1	1	1	0	0	0	0	0	0	1	0
				0	0	1	1	0	1	1	1	0	1	0	0	0	0	0	1
				0	1	0	0	0	1	1	1	1	0	0	0	0	0	0	0
				0	1	0	1	0	1	1	1	1	1	0	0	1	1	1	1
				1	1	1	1	0	0	0	0	1	1	0	0	0	1	0	1
				1	1	1	0	0	0	0	0	1	0	0	1	1	1	0	0
				1	1	0	1	0	0	0	0	0	1	0	1	1	1	0	1
				1	1	0	0	0	0	0	0	0	0	0	1	1	1	1	0
				1	0	1	1	0	0	1	1	1	1	0	1	1	1	1	1
1	1	0	1	0	0	0	0	0	0	1	1	0	1	0	0	1	1	0	1
				0	0	0	1	0	0	1	1	1	0	0	0	1	1	0	0
				0	0	1	0	0	0	1	1	1	1	0	0	1	0	1	1
				0	0	1	1	0	0	0	0	0	0	1	1	0	1	0	0
				0	1	0	0	0	0	0	0	0	1	1	1	0	0	1	1
				0	1	0	1	0	0	0	0	1	0	1	1	0	0	1	0
				1	1	1	1	0	0	1	1	0	0	0	0	1	1	1	0
				1	1	1	0	0	0	1	0	1	1	0	0	1	1	1	1
				1	1	0	1	1	1	0	1	0	0	0	0	0	0	0	0
				1	1	0	0	1	1	0	0	1	1	0	0	0	0	0	1
				1	0	1	1	1	1	0	0	1	0	0	0	0	0	1	0

$x_2x_1 = 01$														$x_2x_1 = 11$					
b_8	b_4	b_2	b_1	a_8	a_4	a_2	a_1	t_2	t_1	w_8	w_4	w_2	w_1	t_2	t_1	w_8	w_4	w_2	w_1
1	1	0	0	0	0	0	0	0	0	1	1	0	0	0	0	1	1	0	0
				0	0	0	1	0	0	1	1	0	1	0	0	1	0	1	1
				0	0	1	0	0	0	1	1	1	0	1	1	0	1	0	0
				0	0	1	1	0	0	1	1	1	1	1	1	0	0	1	1
				0	1	0	0	0	0	0	0	0	0	1	1	0	0	1	0
				0	1	0	1	0	0	0	0	0	1	1	1	0	0	0	1
				1	1	1	1	0	0	1	0	1	1	0	0	1	1	0	1
				1	1	1	0	1	1	0	1	0	0	0	0	1	1	1	0
				1	1	0	1	1	1	0	0	1	1	0	0	1	1	1	1
				1	1	0	0	1	1	0	0	1	0	0	0	0	0	0	0
				1	0	1	1	1	1	0	0	0	1	0	0	0	0	0	1

Boolean Equations:

$$t_1 = \bar{x}_2 x_1 [b_2(\bar{a}_8\bar{a}_4 a_2 a_1 + \bar{a}_8 a_4 \bar{a}_2) + \bar{b}_8 b_4(\bar{a}_8\bar{a}_4 a_2 + \bar{a}_8 a_4 \bar{a}_2)$$

$$+ b_8 b_1(a_8 a_4 \bar{a}_2 + a_8 \bar{a}_4) + b_8 \bar{b}_1(a_8 \bar{a}_4 + a_8 \bar{a}_2)]$$

$$+ x_2 x_1 [b_2(a_8 \bar{a}_4 + a_8 a_4 \bar{a}_2) + \bar{b}_8 b_4(a_8 \bar{a}_4 + a_8 \bar{a}_2 + a_8 \bar{a}_1)$$

$$+ b_8 b_1(\bar{a}_8 a_2 a_1 + \bar{a}_8 a_4) + b_8 \bar{b}_1(\bar{a}_8 a_2 + \bar{a}_8 a_4)]$$

$$t_2 = \bar{x}_2 x_1 [b_8 b_1(a_8 a_4 \bar{a}_2 + a_8 \bar{a}_4) + b_8 \bar{b}_1(a_8 \bar{a}_4 + a_8 \bar{a}_2)]$$

$$+ x_2 x_1 [b_8 b_1(\bar{a}_8 a_2 a_1 + \bar{a}_8 a_4) + b_8 \bar{b}_1(\bar{a}_8 a_2 + \bar{a}_8 a_4)]$$

$$w_1 = b_1 \oplus a_1 = b_1 \bar{a}_1 + \bar{b}_1 a_1$$

$$w_2 = \bar{x}_2 x_1 [\bar{b}_8\bar{b}_4\bar{b}_2 a_2 + b_2(a_8 \bar{a}_4 + a_8 a_2 a_1 + a_8 a_4 \bar{a}_2 \bar{a}_1 + \bar{a}_8\bar{a}_4\bar{a}_2 + \bar{a}_8 a_4 a_1)$$

$$+ \bar{b}_8 b_4(\bar{a}_8 a_4 + a_8 a_4 a_2 + a_8 \bar{a}_4)$$

$$+ b_8 b_1 (\bar{a}_8 \bar{a}_2 a_1 + a_2 \bar{a}_1 + a_8 \bar{a}_4 + a_8 a_4 \bar{a}_1)$$

$$+ b_8 \bar{b}_1 (\bar{a}_8 \bar{a}_4 a_2 + a_8 a_4 a_1 + a_8 a_4 \bar{a}_2)]$$

$$+ x_2 x_1 [\bar{b}_8 \bar{b}_4 \bar{b}_2 (\bar{a}_2 a_1 + a_2 \bar{a}_1) + b_2 (\bar{a}_8 \bar{a}_2 + a_8 \bar{a}_4)$$

$$+ \bar{b}_8 b_4 (\bar{a}_8 \bar{a}_2 a_1 + \bar{a}_4 a_2 \bar{a}_1 + a_8 \bar{a}_4 + a_8 a_4 \bar{a}_2 \bar{a}_1)$$

$$+ b_8 b_1 (\bar{a}_4 a_2 + \bar{a}_8 a_4 + a_8 a_4 a_2)$$

$$+ b_8 \bar{b}_1 (\bar{a}_8 \bar{a}_4 a_1 + \bar{a}_8 a_4 \bar{a}_1 + a_8 \bar{a}_2 a_1 + a_8 a_2 \bar{a}_1)]$$

$$w_4 = x_2 x_1 [\bar{b}_8 \bar{b}_4 \bar{b}_2 (a_4 \bar{a}_2 \bar{a}_1 + \bar{a}_4 a_2 + \bar{a}_8 \bar{a}_4 a_1) + b_2 (a_8 + \bar{a}_8 a_4)$$

$$+ \bar{b}_8 b_4 (\bar{a}_4 \bar{a}_2 \bar{a}_1 + a_8 + a_4 \bar{a}_2 a_1) + b_8 b_1 (a_8 a_4 a_2 + \bar{a}_4 \bar{a}_2 + \bar{a}_8 \bar{a}_4 a_1)$$

$$+ b_8 \bar{b}_1 (\bar{a}_8 \bar{a}_4 \bar{a}_1 + a_8 a_4 a_2 + a_8 a_4 a_1)]$$

$$+ \bar{x}_2 x_1 [\bar{b}_8 \bar{b}_4 \bar{b}_2 a_4 + b_2 \bar{a}_8 (a_1 + a_2 + a_4) + b_2 (a_8 \bar{a}_4 + a_8 \bar{a}_2 \bar{a}_1)$$

$$+ \bar{b}_8 b_4 (\bar{a}_8 + \bar{a}_4) + b_8 b_1 (a_8 a_4 a_1 + \bar{a}_8 \bar{a}_4 \bar{a}_1 + \bar{a}_8 \bar{a}_4 \bar{a}_2)$$

$$+ b_8 \bar{b}_1 (\bar{a}_8 \bar{a}_4 + a_8 a_2 \bar{a}_1)]$$

$$w_8 = \bar{x}_2 x_1 [\bar{b}_8 \bar{b}_4 \bar{b}_2 a_8 + b_2 (\bar{a}_8 a_4 + \bar{a}_8 a_2 a_1 + a_8 \bar{a}_2 \bar{a}_1 + a_8 \bar{a}_4)$$

$$+ \bar{b}_8 b_4 (\bar{a}_8 a_4 + \bar{a}_4 a_2) + b_8 b_1 (a_8 a_4 a_2 + \bar{a}_8 \bar{a}_4 \bar{a}_2 + \bar{a}_8 \bar{a}_4 \bar{a}_1)$$

$$+ b_8 \bar{b}_1 (\bar{a}_8 \bar{a}_4 + a_8 a_2 a_1)]$$

$$+ x_2 x_1 [\bar{b}_8 \bar{b}_4 \bar{b}_2 \bar{a}_8 (a_4 + a_2 + a_1) + b_2 (\bar{a}_8 a_4 + a_8 \bar{a}_2 + a_8 \bar{a}_4)$$

$$+ \bar{b}_8 b_4 (a_4 \bar{a}_2 a_1 + a_8 \bar{a}_1 + a_8 \bar{a}_4) + b_8 b_1 (\bar{a}_8 \bar{a}_4 \bar{a}_2 + \bar{a}_8 \bar{a}_4 \bar{a}_1 + a_8 a_4 a_2)$$

$$+ b_8 \bar{b}_1 (\bar{a}_8 \bar{a}_4 \bar{a}_2 + a_8 a_4 a_2 + a_8 a_4 a_1)]$$

4. Logic Design for Cell D

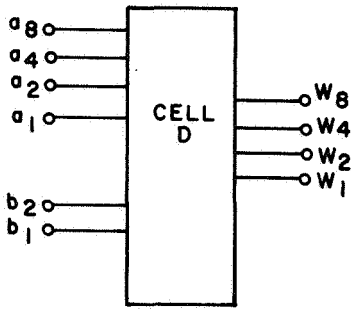


Fig. 66. I/O PINS OF CELL D.

TRUTH TABLE OF CELL D

a_8	a_4	a_2	a_1	b_2	b_1	w_8	w_4	w_2	w_1
0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	1
0	0	1	0	0	0	0	0	1	0
0	0	1	1	0	0	0	0	1	1
0	1	0	0	0	0	0	1	0	0
0	1	0	1	0	0	0	1	0	1
1	1	1	1	0	0	1	1	1	1
1	1	1	0	0	0	1	1	1	0
1	1	0	1	0	0	1	1	0	1
1	1	0	0	0	0	1	1	0	0
1	0	1	1	0	0	1	0	1	1
0	0	0	0	0	1	0	0	0	1
0	0	0	1	0	1	0	0	1	0
0	0	1	0	0	1	0	0	1	1
0	0	1	1	0	1	0	1	0	0
0	1	0	0	0	1	0	1	0	1
0	1	0	1	0	1	0	1	1	0
1	1	1	1	0	1	0	0	0	0
1	1	1	0	0	1	1	1	1	1
1	1	0	1	0	1	1	1	1	0
1	1	0	0	0	1	1	1	0	1
1	0	1	1	0	1	1	1	0	0
0	0	0	0	1	1	1	1	1	1
0	0	0	1	1	1	1	1	1	0
0	0	1	0	1	1	1	1	0	1
0	0	1	1	1	1	1	1	0	0
0	1	0	0	1	1	1	0	1	1
0	1	0	1	1	1	1	0	1	0
1	1	1	1	1	1	1	0	0	0
1	1	1	0	1	1	0	1	1	1
1	1	0	1	1	1	0	1	1	0
1	1	0	0	1	1	0	1	0	1
1	0	1	1	1	1	0	1	0	0

Boolean Equations:

$$w_1 = a_1\bar{b}_1 + \bar{a}_1 b_1$$

$$w_2 = \bar{b}_2\bar{b}_1 a_2 + \bar{b}_2 b_1[\bar{a}_8\bar{a}_2 a_1 + \bar{a}_8\bar{a}_4 a_2\bar{a}_1 + a_8 a_4(a_2\bar{a}_1 + \bar{a}_2 a_1)]$$

$$+ b_2 b_1(\bar{a}_8\bar{a}_2\bar{a}_1 + \bar{a}_4 a_2 a_1 + a_8 a_2 a_1 + a_4\bar{a}_2\bar{a}_1)$$

$$w_4 = \bar{b}_2\bar{b}_1 a_4 + \bar{b}_2 b_1(\bar{a}_8 a_4 + \bar{a}_4 a_2 a_1 + a_8 a_4\bar{a}_2 + a_8 a_4\bar{a}_1)$$

$$+ b_2 b_1(\bar{a}_8\bar{a}_4\bar{a}_2\bar{a}_1 + a_4\bar{a}_2 a_1 + a_8 a_4 a_2)$$

$$w_8 = \bar{b}_2\bar{b}_1 a_8 + \bar{b}_2 b_1(a_8 a_4\bar{a}_1 + a_8 a_4\bar{a}_2 + a_8\bar{a}_4 a_2 a_1) + b_2 b_1(\bar{a}_8\bar{a}_4\bar{a}_2\bar{a}_1 + a_8)$$

5. Logic Design for Cell E

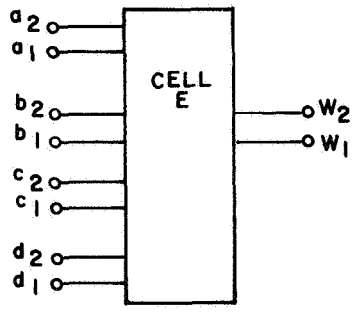


Fig. 67. I/O PINS OF CELL E.

TRUTH TABLE FOR CELL E

a_2 a_1	b_2 b_1	c_2 c_1	d_2 d_1	w_2 w_1
0 0	0 0	0 0	0 0	0 0
0 0	0 0	0 0	0 1	0 1
0 0	0 0	0 0	1 1	1 1
0 0	0 0	0 1	0 0	0 1
0 0	0 0	0 1	1 1	0 0
0 0	0 0	1 1	0 0	1 1

a_2 a_1	b_2 b_1	c_2 c_1	d_2 d_1	w_2 w_1
0 0	0 0	1 1	0 1	0 0
0 0	0 1	0 0	0 0	0 1
0 0	0 1	0 0	1 1	0 0
0 0	0 1	1 1	0 0	0 0
0 0	1 1	0 0	0 0	1 1
0 0	1 1	0 0	0 1	0 0
0 0	1 1	0 1	0 0	0 0
0 1	0 0	0 0	0 0	0 1
0 1	0 0	0 0	1 1	0 0
0 1	0 0	1 1	0 0	0 0
0 1	1 1	0 0	0 0	0 0
1 1	0 0	0 0	0 0	1 1
1 1	0 0	0 0	0 1	0 0
1 1	0 0	0 1	0 0	0 0
1 1	0 1	0 0	0 0	0 0

Boolean Equations:

$$w_1 = a_1 \oplus b_1 \oplus c_1 \oplus d_1$$

$$w_2 = a_1\bar{b}_1\bar{c}_1\bar{d}_1 + \bar{a}_1 b_1\bar{c}_1\bar{d}_1 + \bar{a}_1\bar{b}_1 c_1\bar{d}_1 + \bar{a}_1\bar{b}_1\bar{c}_1 d_1$$

6. Logic Design for Cell F (Half Divider)

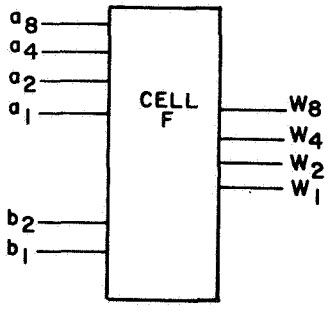


Fig. 68. I/O PINS OF CELL F.

TRUTH TABLE FOR CELL F

a_8	a_4	a_2	a_1	b_2	b_1	w_8	w_4	w_2	w_1
0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0	0
0	0	1	0	0	0	0	0	0	1
0	0	1	1	0	0	0	0	0	1
0	1	0	0	0	0	0	0	1	0
0	1	0	1	0	0	0	0	1	0
0	1	1	0	0	0	0	0	1	1
0	1	1	1	0	0	0	0	1	1
1	1	1	1	0	0	0	0	0	0
1	1	1	0	0	0	1	1	1	1
1	1	0	1	0	0	1	1	1	1
1	1	0	0	0	0	1	1	1	0
1	0	1	1	0	0	1	1	1	0
1	0	1	0	0	0	1	1	0	1
1	0	0	1	0	0	1	1	0	1
0	0	0	0	0	1	0	0	0	0
0	0	0	1	0	1	0	0	0	1
0	0	1	0	0	1	0	0	0	1
0	0	1	1	0	1	0	0	1	0
0	1	0	0	0	1	0	0	1	0
0	1	0	1	0	1	0	0	1	1
0	1	1	0	0	1	0	0	1	1
0	1	1	1	0	1	0	1	0	0

a_8	a_4	a_2	a_1	b_2	b_1	w_8	w_4	w_2	w_1
1	1	1	1	0	1	0	0	0	0
1	1	1	0	0	1	0	0	0	0
1	1	0	1	0	1	1	1	1	1
1	1	0	0	0	1	1	1	1	1
1	0	1	1	0	1	1	1	1	0
1	0	1	0	0	1	1	1	1	0
1	0	0	1	0	1	1	1	0	1
0	0	0	0	1	1	0	0	0	0
0	0	0	1	1	1	0	0	0	0
0	0	1	0	1	1	0	0	0	0
0	0	1	1	1	1	0	0	0	1
0	1	0	0	1	1	0	0	0	1
0	1	0	1	1	1	0	0	1	0
0	1	1	0	1	1	0	0	1	0
0	1	1	1	1	1	0	0	1	1
1	1	1	1	1	1	1	1	1	1
1	1	1	0	1	1	1	1	1	1
1	1	0	1	1	1	1	1	1	0
1	1	0	0	1	1	1	1	1	0
1	0	1	1	1	1	1	1	0	1
1	0	1	0	1	1	1	1	0	1
1	0	0	1	1	1	1	1	0	0

Boolean Equations:

$$w_1 = \bar{b}_2\bar{b}_1(\bar{a}_8a_2 + a_8a_2\bar{a}_1 + a_8\bar{a}_2a_1) + \bar{b}_2b_1(\bar{a}_8\bar{a}_2a_1 + \bar{a}_8a_2\bar{a}_1 + a_8a_4\bar{a}_2 + a_8\bar{a}_2a_1)$$

$$+ b_2b_1(a_2a_1 + a_8a_2\bar{a}_1 + \bar{a}_8a_4\bar{a}_2\bar{a}_1)$$

$$w_2 = \bar{b}_2\bar{b}_1(\bar{a}_8a_4 + a_8a_4\bar{a}_1 + a_4\bar{a}_2a_1 + a_8\bar{a}_4a_2a_1)$$

$$+ \bar{b}_2 b_1 (\bar{a}_4 a_2 a_1 + \bar{a}_8 a_4 \bar{a}_2 + a_8 a_4 \bar{a}_2 + a_8 \bar{a}_4 a_2 + \bar{a}_8 a_4 a_2 \bar{a}_1)$$

$$+ b_2 b_1 (a_8 a_4 + a_4 a_2 + \bar{a}_8 a_4 a_1)$$

$$w_4 = w_8 + \bar{b}_2 b_1 \bar{a}_8 a_4 a_2 a_1$$

$$w_8 = \bar{b}_2 \bar{b}_1 (a_8 \bar{a}_4 + a_8 \bar{a}_1 + a_8 \bar{a}_2) + \bar{b}_2 b_1 (a_8 a_4 \bar{a}_2 + a_8 \bar{a}_4) + b_2 b_1 a_8$$

7. Overflow (or Underflow) Detection

TRUTH TABLE FOR OVERFLOW INDICATION

	R_4		R_3				R_2				R_1				Δz_{i-1}		z_t
z_2 z_1	r_{42}	r_{41}	r_{38}	r_{34}	r_{32}	r_{31}	r_{28}	r_{24}	r_{22}	r_{21}	r_{18}	r_{14}	r_{12}	r_{11}	z_2	z_1	
	0	1	0	–	–	–	–	–	–	–	–	–	–	–	–	–	–
	0	1	1	1	1	–	–	–	–	–	–	–	–	–	–	–	–
	0	1	1	1	0	1	0	–	–	–	–	–	–	–	–	–	–
Δz_i=01	0	1	1	1	0	1	1	1	1	–	–	–	–	–	–	–	–
(+1)	0	1	1	1	0	1	1	1	0	1	0	–	–	–	–	–	–
	0	1	1	1	0	1	1	1	0	1	1	1	1	–	–	–	–
	0	1	1	1	0	1	1	1	0	1	1	1	–	1	–	–	–
	0	1	1	1	0	1	1	1	0	1	1	1	0	0	0	1	1
	0	1	1	1	0	1	1	1	0	1	1	1	0	0	–	–	0
	1	1	1	–	–	–	–	–	–	–	–	–	–	–	–	–	–
	1	1	0	0	0	–	–	–	–	–	–	–	–	–	–	–	–
	1	1	0	0	–	0	–	–	–	–	–	–	–	–	–	–	–
	1	1	0	0	1	1	1	–	–	–	–	–	–	–	–	–	–
Δz_i=11	1	1	0	0	1	1	0	0	0	–	–	–	–	–	–	–	–
(-1)	1	1	0	0	1	1	0	0	–	0	–	–	–	–	–	–	–
	1	1	0	0	1	1	0	0	1	1	1	–	–	–	–	–	–
	1	1	0	0	1	1	0	0	1	1	0	0	0	–	–	–	–
	1	1	0	0	1	1	0	0	1	1	0	0	1	–	–	–	–
	1	1	0	0	1	1	0	0	1	1	0	1	0	0	1	1	1
	1	1	0	0	1	1	0	0	1	1	0	1	0	0	–	–	0

Boolean Functions:

$$\Delta z_i = \bar{z}_2 z_1 = \bar{r}_{42} r_{41} \Big\{ \bar{r}_{38} + r_{38} r_{34} r_{32} + r_{38} r_{34} \bar{r}_{32} r_{31} \Big\{ \bar{r}_{28} + r_{28} r_{24} r_{22} + r_{28} r_{24} \bar{r}_{22} r_{21} \Big[\bar{r}_{18} + r_{18} r_{14} r_{12} + r_{18} r_{14} r_{11} + r_{18} r_{14} \bar{r}_{12} \bar{r}_{11} (\bar{z}_2 z_1 + \bar{z}_t) \Big] \Big\} \Big\}$$

$$\Delta z_i = z_2 z_1 = r_{42} r_{41} \Big\{ r_{38} + \bar{r}_{38} \bar{r}_{34} \bar{r}_{32} + \bar{r}_{38} \bar{r}_{34} \bar{r}_{31} + \bar{r}_{38} \bar{r}_{34} r_{32} r_{31} \Big[r_{28} + \bar{r}_{28} \bar{r}_{24} \bar{r}_{22} + \bar{r}_{28} \bar{r}_{24} \bar{r}_{21} + \bar{r}_{28} \bar{r}_{24} r_{22} r_{21} \Big\{ r_{18} + \bar{r}_{18} \bar{r}_{14} \bar{r}_{12} + \bar{r}_{18} \bar{r}_{14} r_{12} + \bar{r}_{18} r_{14} \bar{r}_{12} \bar{r}_{11} (z_2 z_1 + \bar{z}_t) \Big\} \Big] \Big\}$$

Appendix C

LOGIC DESIGN OF THE DIGITAL SUMMING ELEMENT

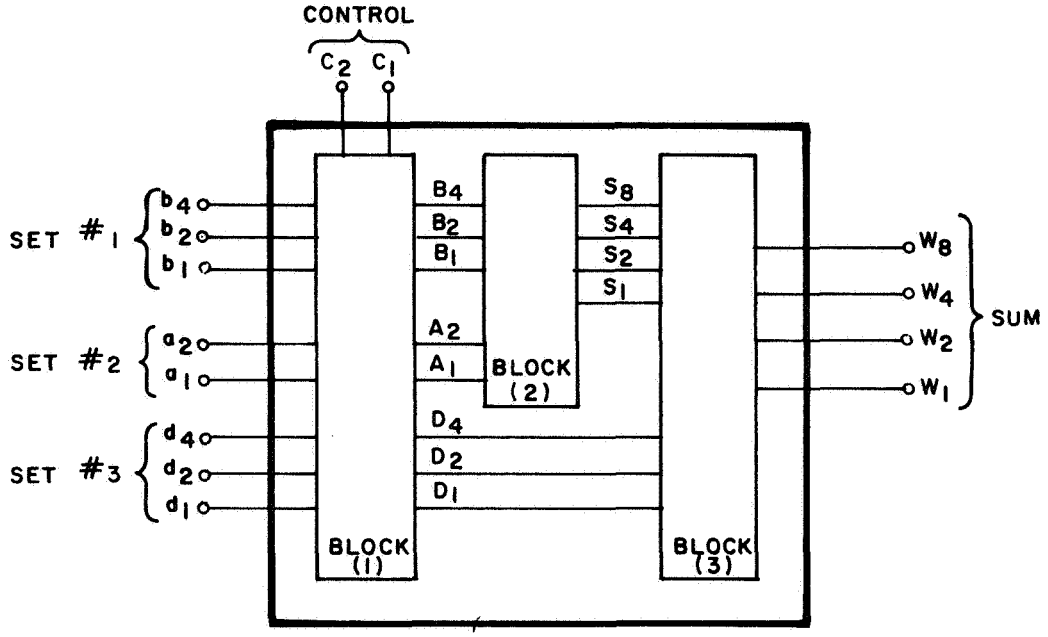


Fig. 69. INTERNAL BLOCK STRUCTURE OF DIGITAL SUMMING ELEMENT.

TRUTH TABLE C-1 FOR BLOCK (1)

C_2C_1 = 11,10,01						C_2C_1 = 11,01				C_2C_1 = 11,10					
b_4	b_2	b_1	B_4	B_2	B_1	a_2	a_1	A_2	A_1	d_4	d_2	d_1	D_4	D_2	D_1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	1	0	1	0	1	0	0	1	0	0	1
0	1	0	0	1	0	1	1	1	1	0	1	0	0	1	0
0	1	1	0	1	1					0	1	1	0	1	1
1	1	1	1	1	1					1	1	1	1	1	1
1	1	0	1	1	0					1	1	0	1	1	0
1	0	1	1	0	1					1	0	1	1	0	1

$C_2C_1 = 00$						$C_2C_1 = 00,10$				$C_2C_1 = 00,01$					
b_4	b_2	b_1	B_4	B_2	B_1	a_2	a_1	A_2	A_1	d_4	d_2	d_1	D_4	D_2	D_1
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	0	1	1	1	0	0	1	1	1	1
0	1	0	1	1	0	1	1	0	1	0	1	0	1	1	0
0	1	1	1	0	1					0	1	1	1	0	1
1	1	1	0	0	1					1	1	1	0	0	1
1	1	0	0	1	0					1	1	0	0	1	0
1	0	1	0	1	1					1	0	1	0	1	1

TRUTH TABLE C-2 FOR BLOCK (2)

A_2	A_1	B_4	B_2	B_1	S_8	S_4	S_2	S_1
0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	1
0	0	0	1	0	0	0	1	0
0	0	0	1	1	0	0	1	1
0	0	1	1	1	1	1	1	1
0	0	1	1	0	1	1	1	0
0	0	1	0	1	1	1	0	1
0	1	0	0	0	0	0	0	1
0	1	0	0	1	0	0	1	0
0	1	0	1	0	0	0	1	1
0	1	0	1	1	0	1	0	0
0	1	1	1	1	0	0	0	0
0	1	1	1	0	1	1	1	1
0	1	1	0	1	1	1	1	0
1	1	0	0	0	1	1	1	1
1	1	0	0	1	0	0	0	0
1	1	0	1	0	0	0	0	1
1	1	0	1	1	0	0	1	0
1	1	1	1	1	1	1	1	0
1	1	1	1	0	1	1	0	1
1	1	1	0	1	1	1	0	0

Boolean Equations:

$$B_4 = (C_2 + C_1)\, b_4 + \overline{C}_2\overline{C}_1\overline{b}_4(b_2 + b_1)$$

$$B_2 = (C_2 + C_1)\, b_2 + \overline{C}_2\overline{C}_1(\overline{b}_1 b_2 + \overline{b}_2 b_1)$$

$$B_1 = (C_2 + C_1 + \overline{C}_2\overline{C}_1)\, b_1 = b_1$$

$$A_2 = C_1 a_2 + \overline{C}_1\overline{a}_2 a_1$$

$$A_1 = C_1 a_1 + \overline{C}_1 a_1 = a_1$$

$$D_4 = C_2 d_4 + \overline{C}_2\overline{d}_4(d_1 + d_2)$$

$$D_2 = C_2 d_2 + \overline{C}_2(\overline{d}_2 d_1 + d_2\overline{d}_1)$$

$$D_1 = d_1$$

$$S_8 = C_2C_1[\overline{A}_2\overline{A}_1B_4 + \overline{A}_2A_1B_4B_2\overline{B}_1 + \overline{A}_2A_1B_4\overline{B}_2B_1 + A_2A_1\overline{B}_4\overline{B}_2\overline{B}_1 + A_2A_1B_4B_2]$$

$$S_4 = S_8 + [\overline{A}_2A_1 + \overline{B}_4B_2B_1]\, C_2C_1$$

$$S_2 = C_2C_1[\overline{A}_2\overline{A}_1B_2 + \overline{A}_2A_1B_2\overline{B}_1 + \overline{A}_2A_1\overline{B}_2B_1 + A_2A_1\overline{B}_4\overline{B}_2\overline{B}_1 + A_2A_1B_2B_1]$$

$$S_1 = C_2C_1[A_1\overline{B}_1 + \overline{A}_1B_1]$$

TRUTH TABLE C-3 FOR BUILDING BLOCK (3)

D_4	D_2	D_1	S_8	S_4	S_2	S_1	w_8	w_4	w_2	w_1
0	0	0	0	0	0	0	0	0	0	0
			0	0	0	1	0	0	0	1
			0	0	1	0	0	0	1	0
			0	0	1	1	0	0	1	1
			0	1	0	0	0	1	0	0
			1	1	1	1	1	1	1	1
			1	1	1	0	1	1	1	0
			1	1	0	1	1	1	0	1
			1	1	0	0	1	1	0	0
0	0	1	0	0	0	0	0	0	0	1
			0	0	0	1	0	0	1	0
			0	0	1	0	0	0	1	1
			0	0	1	1	0	1	0	0
			0	1	0	0	0	1	0	1
			1	1	1	1	0	0	0	0
			1	1	1	0	1	1	1	1
			1	1	0	1	1	1	1	0
			1	1	0	0	1	1	0	1
0	1	0	0	0	0	0	0	0	1	0
			0	0	0	1	0	0	1	1
			0	0	1	0	0	1	0	0
			0	0	1	1	0	1	0	1
			0	1	0	0	0	1	1	0
			1	1	1	1	0	0	0	1
			1	1	1	0	0	0	0	0
			1	1	0	1	1	1	1	1
			1	1	0	0	1	1	1	0
0	1	1	0	0	0	0	0	0	1	1
			0	0	0	1	0	1	0	0
			0	0	1	0	0	1	0	1
			0	0	1	1	0	1	1	0
			0	1	0	0	0	1	1	1

D_4	D_2	D_1	S_8	S_4	S_2	S_1	w_8	w_4	w_2	w_1
0	1	1	1	1	1	1	0	0	1	0
			1	1	1	0	0	0	0	1
			1	1	0	1	0	0	0	0
			1	1	0	0	1	1	1	1
1	1	1	0	0	0	0	1	1	1	1
			0	0	0	1	0	0	0	0
			0	0	1	0	0	0	0	1
			0	0	1	1	0	0	1	0
			0	1	0	0	0	0	1	1
			1	1	1	1	1	1	1	0
			1	1	1	0	1	1	0	1
			1	1	0	1	1	1	0	0
			1	1	0	0	1	0	1	1
1	1	0	0	0	0	0	1	1	1	0
			0	0	0	1	1	1	1	1
			0	0	1	0	0	0	0	0
			0	0	1	1	0	0	0	1
			0	1	0	0	0	0	1	0
			1	1	1	1	1	1	0	1
			1	1	1	0	1	1	0	0
			1	1	0	1	1	0	1	1
			1	1	0	0	1	0	1	0
1	0	1	0	0	0	0	1	1	0	1
			0	0	0	1	1	1	1	0
			0	0	1	0	1	1	1	1
			0	0	1	1	0	0	0	0
			0	1	0	0	0	0	0	1
			1	1	1	1	1	1	0	0
			1	1	1	0	1	0	1	1
			1	1	0	1	1	0	1	0
			1	1	0	0	1	0	0	1

Boolean Equations:

$$w_8 = \bar{D}_4\bar{D}_2\bar{D}_1S_8 + \bar{D}_4\bar{D}_2D_1S_8(\bar{S}_2 + \bar{S}_1) + \bar{D}_4D_2\bar{D}_1S_8\bar{S}_2 + \bar{D}_4D_2D_1S_8S_4\bar{S}_2\bar{S}_1$$
$$+ D_4D_2D_1(\bar{S}_4\bar{S}_2\bar{S}_1 + S_8) + D_4D_2\bar{D}_1(S_8 + \bar{S}_4\bar{S}_2) + D_4\bar{D}_2D_1(S_8 + \bar{S}_4\bar{S}_2 + \bar{S}_4\bar{S}_1)$$

$$w_4 = \bar{D}_4\bar{D}_2\bar{D}_1(S_8 + S_4\bar{S}_2\bar{S}_1) + \bar{D}_4\bar{D}_2D_1(\bar{S}_8\bar{S}_4S_2S_1 + S_4\bar{S}_2\bar{S}_1 + S_8S_4\bar{S}_1 + S_8S_4\bar{S}_2)$$
$$+ \bar{D}_4D_2\bar{D}_1(\bar{S}_8\bar{S}_4S_2 + S_4\bar{S}_2\bar{S}_1 + S_8S_4\bar{S}_2) + \bar{D}_4D_2D_1[\bar{S}_8(S_4 + S_2 + S_1) + S_4\bar{S}_2\bar{S}_1]$$
$$+ D_4D_2D_1(\bar{S}_4\bar{S}_2\bar{S}_1 + S_8) + D_4D_2\bar{D}_1(\bar{S}_8\bar{S}_4\bar{S}_2 + S_8S_4S_2)$$
$$+ D_4\bar{D}_2D_1(\bar{S}_8\bar{S}_4\bar{S}_2 + \bar{S}_8\bar{S}_4\bar{S}_1 + S_8S_4S_2S_1)$$

$$w_2 = \bar{D}_4\bar{D}_2\bar{D}_1(\bar{S}_8\bar{S}_4S_2 + S_8S_4S_2) + \bar{D}_4\bar{D}_2D_1(\bar{S}_8\bar{S}_4\bar{S}_2S_1 + \bar{S}_8\bar{S}_4S_2\bar{S}_1 + S_8S_4S_2\bar{S}_1$$
$$+ S_8S_4\bar{S}_2S_1) + \bar{D}_4D_2\bar{D}_1(\bar{S}_8\bar{S}_4\bar{S}_2 + \bar{S}_8\bar{S}_2\bar{S}_1 + S_8S_4\bar{S}_2)$$
$$+ \bar{D}_4D_2D_1(\bar{S}_8\bar{S}_2\bar{S}_1 + \bar{S}_8\bar{S}_4S_2S_1 + S_8S_4S_2S_1 + S_8S_4\bar{S}_2\bar{S}_1)$$
$$+ D_4D_2D_1(\bar{S}_8\bar{S}_2\bar{S}_1 + \bar{S}_8\bar{S}_4S_2S_1 + S_8S_4S_2S_1 + S_8S_4\bar{S}_2\bar{S}_1)$$
$$+ D_4D_2\bar{D}_1(\bar{S}_8\bar{S}_4\bar{S}_2 + \bar{S}_8\bar{S}_2\bar{S}_1 + S_8S_4\bar{S}_2)$$
$$+ D_4\bar{D}_2D_1(\bar{S}_8\bar{S}_4\bar{S}_2S_1 + \bar{S}_8\bar{S}_4S_2\bar{S}_1 + S_8S_4S_2\bar{S}_1 + S_8S_4\bar{S}_2S_1)$$

$$w_1 = D_1\bar{S}_1 + \bar{D}_1S_1$$

Appendix D

SIMULATION OF SINUSOIDAL RESPONSE OF THE TRANSFER FUNCTION G(s) = 1/(s+1)

The program has been written in extended ALGOL. It has been run on B5500 machine at Stanford University.

```
BEGIN
     REAL  Z,YTRUE,X,XX,XXX,YAPPROX;
     INTEGER  M,N,P,XREG, Y1REG,DX,DY,DYY,YREG,RREG,T,DT,TFINAL;
     READ  (M,N,TFINAL);
           P←2*N;
           YREG←Y1REG←XREG←RREG←0;
     FOR  T←1 STEP 1 UNTIL TFINAL DO
     BEGIN
       Z←T/P;
       X←(P-1)×SIN(Z);
       XX←X-XREG; XXX←ENTIER(ABS(XX));
     IF  XX<0 THEN DX←-XXX ELSE DX←XXX;
         XREG←XREG+DX;
     IF  ABS (RREG)≥M AND RREG>0 THEN
       BEGIN  DY←1; RREG←RREG-M
       END ELSE IF ABS (RREG) ≥M AND RREG<0 THEN
     BEGIN DY←-1; RREG←RREG+M
     END ELSE DY←0;
       DYY←DX-DY;
       Y1REG←Y1REG+DYY;
       RREG←RREG+Y1REG;
     YREG←YREG+DY;
     YTRUE←P×0.5×(EXP(-Z)+SIN(Z)-COS(Z));
       WRITE (T,XREG,Y1REG,RREG,YREG,YTRUE);
       END
       END
     DATA CARD
     256.0      8.0      1500
```

Appendix E

UNIT STEP RESPONSE OF G(s) = 1/(s+1) SIMULATION ON B5500 MACHINE

```
BEGIN

     REAL  M,N,P,Z1,Z2,YYREG,YREG,DY,DYY,X1,X2,DX,DELX,RREG,T,TFINAL;
     READ  (M,N,TFINAL);
           P←2*N; YREG←YYREG←RREG←0;
     FOR   T←1 STEP 1 UNTIL TFINAL DO
     BEGIN
       Z1←T/P; X1←(P-1);
     IF T←1 THEN
     BEGIN DX←(P-1)
     END ELSE DX←0;
     IF RREG≥M  THEN
     BEGIN DY←1; RREG←RREG-M
     END ELSE DY←0;
     DYY←DX - DY; YYREG←YYREG+DYY;
     RREG←RREG+YYREG; YREG←YREG+DY;
     WRITE (T,YREG)
     END
END.
```

Appendix F

APPROXIMATION OF THE IDEAL LOW-PASS FILTER

In the frequency domain approximation, the principal problem is to find a rational function G(s) whose magnitude $|G(j\omega)|$ approximates the ideal low-pass characteristic according to a predetermined error criterion. Two approximations are discussed below.

1. The Maximally Flat Low-Pass Filter Approximation

The equation,

$$|G(j\omega)| = \frac{1}{\sqrt{1 + \omega^{2n}}}$$

is known as the n^{th} order Butterworth or maximally flat low-pass filter response and is an approximation of Fig. 70.

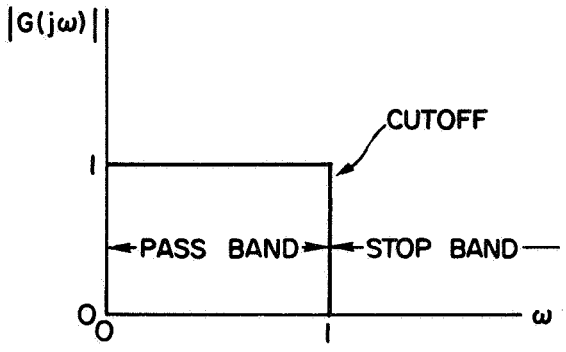


Fig. 70. IDEAL LOW-PASS FILTER CHARACTERISTIC.

The poles of this function are defined by

$$1 + (-s^2)^n = 0$$

and their locations are

$$S_k = \exp\left(j \frac{2k - 1}{n} \frac{\pi}{2}\right) \qquad n \text{ even}$$

$$S_k = \exp\left(j \frac{2k}{n} \frac{\pi}{2}\right) \qquad n \text{ odd}$$

or

$$S_k = \exp\left(j \frac{2k + n - 1}{n} \frac{\pi}{2}\right) \qquad k = 1, 2, 3, \ldots, 2n$$

These poles, thus defined, are located on a unit circle in the s-plane and are symmetrical with respect to both the real and imaginary axes. To form the function G(s) from the given $|G(j\omega)|^2$, the right-half plane poles are rejected, and the left-half plane poles form the all-pole function,

$$G(s) = \frac{1}{1 + a_1 s + a_2 s^2 + \ldots + a_n s^n}$$

The coefficients of the denominator polynomials of G(s), sometimes called Butterworth polynomials, are tabulated in Table 4.

Table 4

COEFFICIENTS OF THE DENOMINATOR POLYNOMIALS OF G(s)

n	a_1	a_2	a_3	a_4	a_5	a_6
1	1.0000					
2	1.4142	1.0000				
3	2.0000	2.0000	1.0000			
4	2.6131	3.4142	2.6131	1.0000		
5	3.2361	5.2361	5.2361	3.2361	1.0000	
6	3.8637	7.4641	9.1416	7.4641	3.8637	1.0000

2. Chebyshev or Equal-Ripple Approximation

The squared magnitude form

$$|G(\omega)|^2 = \frac{1}{1 + \epsilon^2 \, Cn^2(\omega)}$$

is an equal-ripple approximation of Fig. 70, where $Cn(\omega)$ is the n^{th} order Chebyshev polynomial and $\epsilon < 1$ is a real constant. These polynomials are defined in terms of the real variable z by

$$C_n(z) = \cos (n \cos^{-1} z)$$

Let $z = \cos \omega$, then

$$C_n(\omega) = \cos n\omega$$

and a recursion formula can be found as

$$C_{n+1}(z) = 2zC_n(z) - C_{n-1}(z)$$

with

$$C_o(z) = 1 \qquad C_1(z) = z$$

The poles of this equal-ripple form of response can be found as

$$S_k = \sigma_k + j\omega_k$$

where $\sigma_k = \pm \sinh a \sin \frac{2k - 1}{n} \frac{\pi}{2}$

$\omega_k = \cosh a \cos \frac{2k - 1}{n} \frac{\pi}{2}$ $\qquad k = 1,2,3, \ldots, 2n$

$a = \frac{1}{n} \sinh^{-1} \frac{1}{\epsilon}$

Again, the right-half plane poles are rejected.

Appendix G

ERROR ANALYSIS OF THE MODIFIED TRAPEZOIDAL INTEGRATION

Using the error-analysis methods given by Nelson [Ref.28], a comparison of error results between the classical trapezoidal integration and the proposed modified trapezoidal integration has been carried out in the following manner.

1. Trapezoidal-Integration Error

For trapezoidal integration of poles not at the origin, say $s = -\alpha$, the function to be integrated is $Y(s) = 1/(s + \alpha)$, and in z-transform, $Y(z) = Y^*(s) = 1/(1 - az^{-1})$. The real solution after integration should be $R(s) = Y(s)/s = (1/\alpha)[(1/s) - 1/(s + \alpha)]$, or in z-transform, $R(z) = R^*(s) = (1/\alpha)[1/(1 - z^{-1}) - 1/(1 - az^{-1})]$. Because of the imperfect integration of the digital computer, the solution yields

$$R(z)' = \frac{T(1 + z^{-1})}{2(1 - z^{-1})(1 - az^{-1})} - \frac{T}{2(1 - z^{-1})}$$

$$= \frac{[T(1 + a)]/[2(1 - a)]}{(1 - z^{-1})} - \frac{[T(1 + a)]/[2(1 - a)]}{(1 - az^{-1})}$$

where $a = \exp(-\alpha T)$.

The error in the trapezoidal integration is

$$U(z) = R(z) - R(z)' = \left[\frac{1}{\alpha} - \frac{T(1 + a)}{2(1 - a)}\right]\left[\frac{1}{1 - z^{-1}} - \frac{1}{1 - az^{-1}}\right]$$

In the time domain,

$$u(t) = b_1[1 - \exp(-\alpha t)]$$

where

$$b_1 = (1/\alpha) - T[(1 + a)/2(1 - a)] = (1/\alpha) - T[1 + \exp(-\alpha T)]/2[1 - \exp(-\alpha T)]$$

Here, the expression b_1 can be expanded to

$$b_1 = \frac{1}{\alpha} - T\left[\frac{1}{\alpha T} + 0 + \frac{1}{2\cdot 3!}\alpha T + 0 + \frac{(\alpha T)^3}{3!\ 5!} + \ldots.\right]$$

$$= -\frac{\alpha T^2}{2\cdot 3!} + \frac{\alpha^3 T^4}{3!\ 5!} + \ldots. \approx \frac{\alpha T^2}{2\cdot 3!}$$

As $t \to \infty$, error tends to go to $\alpha T^2/2\cdot 3!$ in magnitude.

2. Modified Trapezoidal-Integration Error

For modified trapezoidal integration, the computer yields

$$R(z)'' = \frac{T(3z^{-1} - z^{-2})}{2(1 - z^{-1})}\left[\frac{1}{(1 - az^{-1})}\right] = \frac{Tz^{-1}}{2}\left[\frac{2/(1-a)}{1 - z^{-1}} - \frac{(3a-1)/(1-a)}{1 - az^{-1}}\right]$$

Here, the error is $R(z) - R(z)'' = U(z)'$:

$$U(z)' = U^*(s)' = \left[\frac{1}{\alpha} - \frac{Tz^{-1}}{1-a}\right]\left[\frac{1}{1 - z^{-1}}\right] - \left[\frac{1}{\alpha} - \frac{T(3a-1)\ z^{-1}}{2(1-a)}\right]\left[\frac{1}{1 - az^{-1}}\right]$$

In the time domain,

$$u(t)' = \frac{1}{\alpha} - \frac{T}{1-a} + Z^{-1}\left[\frac{T}{1-a}\right] - \left[\frac{1}{\alpha} - \frac{T(3a-1)}{2(1-a)}\right]$$

$$\cdot \exp(-\alpha T) - Z^{-1}\left[\frac{T(3a-1)}{2(1-a)}\right]\exp(\alpha T)$$

$$= \frac{1}{\alpha} - \frac{T}{1-a} - \frac{T}{1-a}g(t) - \left[\frac{1}{\alpha} - \frac{T(3a-1)}{2(1-a)}\right]$$

$$\cdot \exp(-\alpha T) - \frac{T(3a-1)}{2(1-a)}\exp(-\alpha T)g(t)$$

$$= \frac{1}{\alpha} - \frac{T}{1 - \exp(-\alpha T)} - \frac{T}{1 - \exp(-\alpha T)} g(t)$$

$$- \frac{T[3 \exp(-\alpha T)-1]}{2[1-\exp(-\alpha T)]} g(t)$$

$$- \left[\frac{1}{\alpha} - \frac{T[3 \exp(-\alpha T)-1]}{2[1-\exp(-\alpha T)]}\right] \exp(-\alpha t)$$

where

$$g(t) = 0 \quad \text{if} \quad t \neq 0$$
$$= 1 \quad \text{if} \quad t = 0$$

3. Error Comparison

$$u(t) - u(t)' = \frac{T}{1 - \exp(-\alpha T)} - \frac{T[1+\exp(-\alpha T)]}{2[1-\exp(-\alpha T)]}$$

$$- \left[\frac{T}{1 - \exp(-\alpha T)} - \frac{T[3 \exp(-\alpha T)-1]}{2[1-\exp(-\alpha T)]}\right] g(t)$$

$$+ \left[\frac{T[1+\exp(-\alpha T)]}{2[1-\exp(-\alpha T)]} - \frac{T[3 \exp(-\alpha T)-1]}{2[1-\exp(-\alpha T)]}\right] \exp(-\alpha t)$$

$$= \left[\frac{T[1-\exp(-\alpha T)]}{2[1-\exp(-\alpha T)]} - \frac{3T[1-\exp(-\alpha T)]}{2[1-\exp(-\alpha T)]}\right] g(t)$$

$$- \left[\frac{T[1-\exp(-\alpha T)]}{[1-\exp(-\alpha T)]}\right] \exp(-\alpha t)$$

$$= \left[\frac{T}{2} - \frac{3T}{2}\right] g(t) - T \exp(-\alpha t)$$

At $t = 0$, $u(0) - u(0)' = -T + T = 0$

For $t > 0$, $u(t) - u(t)' = T \exp(-\alpha t)$

As $t \to \infty$, $[u(t) - u(t)']\big|_{t \to \infty} = 0$

The greatest difference occurs at $t = T$, the first sample time.

These results state that the modified scheme is almost as good as the trapezoidal integration. The maximum-error bound is $T \exp(-\alpha T)$. For very large t, the integration differs only by a negligibly small quantity $T \exp(-\alpha n T)$, where $n \gg 1$ is a very large integer.

Appendix H

APPLICATION OF INCREMENTALS TO COMPUTER REALIZATION OF TRANSFER FUNCTIONS

The existing realization method [Refs. 3 and 14] for an integrator of trapezoidal rule, for example, can be expressed, in difference-equation form, as

$$y(nT) = y[(n-1)T] + \frac{T}{2}\{u(nT) - u[(n-1)T]\} \qquad (H.1)$$

where $y(nT)$ is the integration at $t = nT$ from $t = 0$, $u(nT)$ is the coordinate of the curve to be integrated at $t = nT$, and T is the sampling interval.

In terms of the z-transform, Eq. (H.1) becomes

$$Y(z) = z^{-1} Y(z) + \frac{T}{2}[U(z) + z^{-1} U(z)] \qquad (H.2a)$$

or

$$\frac{Y(z)}{U(z)} = \frac{T}{2}\frac{1 + z^{-1}}{1 - z^{-1}} \qquad (H.2b)$$

The implementation of Eq. (H.2b) by using delay and summing elements is shown in Fig. 71. On the other hand, to apply the incremental idea, Eq. (H.1) can be rewritten by substituting $n - 1$ for n, as

$$y[(n-1)T] = y[(n-2)T] + \frac{T}{2}\{u[(n-1)T] - u[(n-2)T]\} \qquad (H.3)$$

Because the only data available will be incremental data, subtracting Eq. (H.3) from Eq. (H.2a) will obtain

$$y(nT) - y[(n-1)T] = \{y[(n-1)T] - y[(n-2)T]\} + \frac{T}{2}\{u(nT) - u[(n-1)T]\} + \{u[(n-1)T] - u[(n-2)T]\} \qquad (H.4)$$

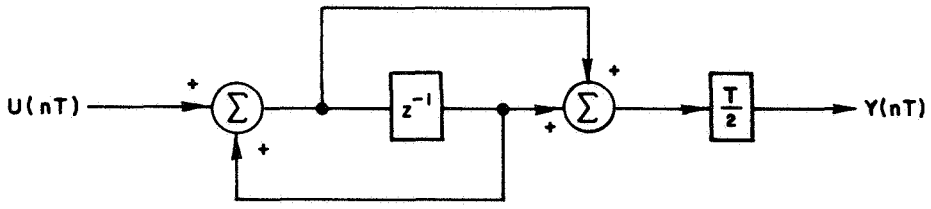


Fig. 71. IMPLEMENTATION OF EQ. (H.2) BY USING DELAY AND SUMMING ELEMENTS.

Incremental data can be defined as

$$\Delta y(iT) = y(iT) - y[(i-1)T] \qquad i \text{ being integers} \tag{H.5a}$$

$$\Delta u(iT) = u(iT) - u[(i-1)T] \tag{H.5b}$$

thus, Eq. (H.4) can be rewritten as

$$\Delta y(nT) = \Delta y[(n-1)T] + \frac{T}{2}\{\Delta u(nT) - \Delta u[(n-1)T]\} \tag{H.6}$$

In z-transform,

$$\Delta Y(z) = z^{-1}\,\Delta Y(z) + \frac{T}{2}[\Delta U(z) + z^{-1}\,\Delta U(z)] \tag{H.7a}$$

or

$$\frac{\Delta Y(z)}{\Delta U(z)} = \frac{T}{2}\,\frac{1 + z^{-1}}{1 - z^{-1}} \tag{H.7b}$$

The implementation of Eq. (H.7b) is found to be exactly as that of Fig. 71, with $Y(z)$, $U(z)$ changed to $\Delta Y(z)$, $\Delta U(z)$.

Comparing Eqs. (H.1) with (H.6) and (H.2) with (H.7), it can be seen that, with the same configuration of realization, the two methods differed only in dealing with either the whole or incremental signal. It is well known that the precision of the analog-to-digital converter

is dependent on the number of quantization levels; that is, to represent the same analog quantity, it depends on the number of bits of the digital elements. Therefore, with the same number of digital bits available, a small signal can be represented more accurately than can a large signal. For example, with five binary bits available, a small signal of 31 mV and a large signal of 315 mV can be represented as 11111 (10^{-3} V) and 11111 (10^{-2} V), respectively, whereby the last digit of the 315 mV is rounded off. Also, the analog-to-digital conversion time is usually less for small signals.

REFERENCES

1. F. F. Kuo and J. F. Kaiser, System Analysis by Digital Computer, John Wiley and Sons, Inc., New York, 1966, pp. 218-285.

2. P. E. Mantey, "Digital-Computer Implementation of Linear Systems," Rept. No. SU-SEL-66-063, Stanford Electronics Laboratories, Stanford University, Stanford, Calif., Oct 1966.

3. C. M. Rader and B. Gold, "Digital Filter Design Techniques," Technical Note 1965-63, M. I. T. Lincoln Laboratory, Cambridge, Mass.

4. R. M. Golden, "Digital Computer Simulation of Communication Systems Using the Block Diagram Compiler: BLODIB," Third Annual Allerton Conference on Circuit and System Theory, Monticello, Illinois, Oct 1965, pp. 690-707.

5. R. M. Golden and J. F. Kaiser, "Design of Wideband Sampled-Data Filters," Bell Systems Technique J., XLIII, 4, Jul 1964.

6. C. P. Yu, "Digital Filter Design Techniques and the Realization of Transfer and Driving-Point Immittance Functions by Using Digital Elements," Scientific Report No. 22, Stanford Electronics Laboratories, Stanford University, Stanford, Calif., Nov 1967.

7. J. S. Prince and G. T. Sendzuk, "MOS LSI Distributed Arithmetic Computation," presented at WESCON, San Francisco, Calif, Aug 1967, or "Build Filters with MOS Chips," Electronic Design, 18, 1 Sep 1967, pp. 55-59.

8. R. L. Petritz, "Current Status of Large Scale Integration Techniques," Fall Joint Computer Conference, 1967, pp. 65-85.

9. A. Avizienis, "Signed-Digit Number Representations for Fast Parallel Arithmetic," IRE Trans. on Electronic Computers, Sep 1961, pp. 389-400.

10. A. Avizienis, "Arithmetic Microsystems for the Synthesis of Function Generators," Proc. IEEE, Dec 1966, pp. 1910-1919.

11. R. W. Newcomb, Active Integrated Network Synthesis, Prentice-Hall, Englewood Cliffs, N.J., Sep 1968.

12. J. B. Knowles and R. Edwards, "Effects of a Finite-Word-Length Computer in a Sampled-Data Feedback System," Proc. IEE (London), 112, Jun 1965, pp. 1197-1207.

13. J. B. Knowles and E. M. Olcayto, "Coefficient Accuracy and Digital Filter Response," IEEE Trans. on Circuit Theory, Mar 1968, pp. 31-41.

14. J. Tou, Digital and Sampled-Data Control System, McGraw-Hill Book Co., N.Y., 1959, pp. 200-215.

15. F. V. Mayorov, Electronic Digital Integrating Computers--Digital Differential Analyzers (ed. by Dr. Y. Chu), American Elsevier Publishing Co., New York, 1964, Chapter 1.

16. Y. N. Patt, "Synthesis of Switching Functions Using a Minimum Number of Integrated-Circuit Modules," TR No. 6901-1, SU-SEL-66-055, Stanford Electronics Laboratories, Stanford University, Stanford, Calif., Dec 1966.

17. E. L. Braun, Digital Computer Design, Academic Press, New York, 1963, pp, 448-520,

18. G. E. Moore, "Cramming More Components into Integrated Circuits," Electronics, Apr 1965.

19. H. Wolf and K. Greenough," The Trend toward Monolithic Subsystems," Second International Symposium on Microelectronics, Munich, Oct 1966.

20. W. A. Notz, E. Schischa, J. L. Smith, and M. G. Smith, "Large Scale Integration Benefiting the System Designer," Electronics, Feb 1967, pp. 130-141.

21. R. N. Clark, Introduction to Automatic Control System, John Wiley and Sons, Inc., New York, 1962.

22. J. F. Kaiser, "Some Practical Considerations in the Realization of Linear Digital Filters," Proc. 3rd. Allerton Conf. on Circuit and System Theory, Monticello, Ill., Oct 1965, pp. 621-633.

23. F. F. Kuo, Network Analysis and Synthesis (2nd ed.), John Wiley and Sons, Inc., New York, 1966.

24. M. E. Van Valkenburg, Introduction to Modern Network Synthesis, John Wiley and Sons, Inc., New York, 1960, pp. 373-392.

25. L. Weinburg. Network Analysis and Synthesis, McGraw-Hill Book Co., New York, 1962, pp. 485-553.

26. E. S. Kuh and D. O. Pederson, Principles of Circuit Synthesis, McGraw-Hill Book Co., N.Y., 1959, pp. 41-43.

27. H. R. Beelitz, S. Y. Levy, R. J. Linhardt, and H. S. Miller, "System Architecture for LSI," Fall Joint Computer Conference, 1967, p. 185.

28. D. J. Nelson, "A Foundation for the Analysis of Analog-Oriented Combined Computer System," TR No. 1002-1, Stanford Electronics Laboratories, Stanford University, Stanford, Calif., Apr 1962.

ADDITIONAL SOURCE MATERIAL

1. R. E. Bradley, and J. F. Gena, "Design of a One Megacycle Iteration Rate DDA."

2. M. W. Goldman, "Design of a High Speed DDA," Fall Joint Computer Conference, 1965, pp. 929-949.

3. J. M. Mitchell and S. Ruhman, "The TRICE- A High Speed Incremental Computer," IRE National Convention Record, Pt 4, 1958, pp. 206-216.

4. E. Thomas and J. D. Callan "Slow, but Small May Win the Race," Electronics, 20 Feb 1967, pp. 179-182.

5. C. S. Weaver, P. E. Mantey, R. W. Lawrence, and C. A. Cole, "Digital Spectrum Analyzers," TR No. 1809/1810-1, Stanford Electronics Laboratories, Stanford University, Stanford, Calif., Jun 1966.