ON THE DESIGN AND REALIZATION OF TWO DIMENSIONAL DIGITAL FILTERS

by

Bu-Chin Wang

March 1979

A dissertation submitted to the Graduate Faculty in Engineering in Partial fulfillment of the requirements for the degree of Doctor of Philosophy, The City University of New York.

This manuscript has been read and accepted for the Graduate Faculty in Engineering in satisfaction of the dissertation requirement for the degree of Doctor Philosophy.

May 10, 79 — Gerald E Subak-Sharpe
date — Chairman of Examining Committee

May 11, 1979 — Frederick E. Thau
date — Executive Officer

Professor Se-Jeung Oh

Professor Frederick E. Thau

Professor Louis Weinberg

Professor Gerald Subak-Sharpe, Chairman
Supervisory Committee

The City University of New York

ABSTRACT

In this thesis, a new two-dimensional rectangular digital filter class was introduced. Design methods applicable to this class of filter were presented and illustrated by several examples. A new real-time parallel processing hardware realization method for 2-D finite-impulse-response and infinite-impulse-response filters was proposed and illustrated by examples. The realization of filters of high order was made possible by hardware saving techniques, such as the replacement of multipliers by ROM's. In the case of linear phase FIR filters, the additional techniques of system function factorization and of code conversion leads to further hardware saving. Two high-speed processing methods, one using section-by-section data processing and the other using microprocessor replacement of single hardware channels, were discussed. Finally, some future applications of 2-D parallel processors were described.

ACKNOWLEDGMENT

The author thanks his thesis advisor, Prof. G. E. Subak-Sharpe, for guidance and encouragement. He also thanks Prof. S. J. Oh and Prof. L. Weinberg who served on the thesis revision committee.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

CHAPTER ONE

Introduction

1.1 History and Brief Survey of One-dimensional Digital Processing.

Digital signal processing is concerned with manipulating a signal $x(t)$ in terms of its sample values $x(t_1) = x_1$, $x(t_2) = x_2$, ... $x(t_n) = x_n$. For the purpose of this thesis, only equally spaced samples will be considered. Unlike sampled data systems, the sample values in a digital system are quantized, that is, their values are expressed by finite length numbers. This in turn produces round-off errors in the final number computed.

A one-dimensional (1-D) digital processor operates on input samples x_n and produces output samples y_n according to the relation

$$y_n = \sum_{k=0}^{K} a_k x_{n-k} - \sum_{k=1}^{M} b_k y_{n-k} \qquad (1.1a)$$

When all coefficients, b_k, of equ.(1.1a) are zero, the processor or filter is called non-recursive, otherwise it

is said to be a recursive filter. Clearly, for the non-recursive case, output values y_n depend only on input values x_{n-k} and not on previous output values y_{n-k}. Equation (1.1) linearly combines input values x_{n-k} and output values y_{n-k}. When coefficients a_k and b_k are constants, the processor is said to be a linear, time-invariant (LTI) processor or filter. A filter is said to be causal if the output value, $y_{n=n_0}$, is only dependent on input values $x_{n<n_0}$. Relation (1.1a) may also be written in the form

$$\sum_{k=0}^{M} b_k y_{n-k} = \sum_{k=0}^{K} a_k x_{n-k} \qquad (b_0=1) \quad (1.1b)$$

If M = 0, this relation reduces to

$$y_n = \sum_{k=0}^{K} h_k x_{n-k} \qquad (h_k=a_k) \quad (1.1c)$$

where h_k is called the impulse response function. For finite K , the processor is now said to be a finite impulse response (FIR) system of order K and clearly a FIR system is non-recursive. On the other hand, with K infinite, a non-recursive processor would result, which is not, however, FIR. An infinite impulse response (IIR) system results for M>0, in which case an equation similar to (1.1c) can be written, with K infinite. Clearly an IIR system is always recursive and vice versa.

Digital processors may also be characterized by their systems function, H(z), that is

$$Y(z) = H(z).X(z) \text{ or } H(z) = \frac{Y(z)}{X(z)} \qquad (1.2)$$

where Y(z), X(z) are the Z-transforms of y_n, x_n respectively and $z = \exp(j\omega)$. Accordingly, from equ.(1.1)

$$H(z)\Big|_{z=e^{j\omega}} = \frac{\sum_{k=0}^{K} a_k \exp(-j\omega k)}{\sum_{k=0}^{M} b_k \exp(-j\omega k)} = \sum_{k=0}^{\infty} h_k \exp(-j\omega k) \qquad (1.3)$$

A digital filter is said to be stable, if $H(\omega)^*$ does not become infinite for any value of ω. Clearly, by this definition, all FIR filters are stable, because $b_n = 0$ for $n>0$ and $b_0 = 1$. On the other hand, IIR filters must be tested for stability. The first step in the design of a K-th order FIR filter consists in establishing the value of K and the values of coefficients $\{h_i\}$, which satisfy given performance specifications. On the other hand, the design of an IIR filter requires establishing the values of K and M and values of coefficients $\{a_i\}$ and $\{b_i\}$, which satisfy given performance specifications. In addition, the values of coefficients $\{b_i\}$ obtained must lead to a stable system function $H(\omega)$. Subsequent steps in filter design ensure that the obtained filter coefficients are quantized to a fixed

* Instead of H (exp (jw)) we shall write H(w).

word length. Next, a specific hardware structure is chosen in which the filter is to be realized, or a software program is utilized to perform the equivalent convolution computation. The Fast Fourier Transform used to carry out this convolution is most often used, and will now be briefly mentioned below.

It may be shown that the transfer function $H(\omega)$ of an LTI system is ideally represented by sine and cosine functions. Hence the Fourier series and the Fourier transform play an important role in much of data processing. The Discrete Fourier Transform (DFT) is widely used to represent sequences of samples and system functions. Computation of the DFT by means of the Fast Fourier Transform (FFT) algorithm, rediscovered by Cooley and Tukey [5], enables computation to be done in $N \log_2 N$ arithmetic operations, rather than in N^2 operations, where N is the number of sampling points. This difference in number of operations is of fundamental importance, because it leads to a tremendous reduction in machine time. For instance, for $N = 1024$, the new computation time is only about 1% of the original time.

Digital processing has proceeded in two directions, along the software and along the hardware route. When digital processing is carried out on a general-purpose digital computer by means of a special computer program, one speaks of a software realization. When a special-purpose

digital computer is realized by means of integrated circuits [6], one speaks of a hardware realization. This latter development has received a tremendous impetus due to the growing availability of cheap, large-scale integrated circuits (LSI). The integrated circuit electronic revolution is presently in full swing with no abatement in sight, and hardware realizations should become available in increasing numbers.

We now briefly survey the literature on 1-D digital filter design.

Linear phase FIR filters may, among other methods, be designed by the window method, by the frequency sampling method, and by optimal filter design methods.

With respect to the window method [7, 8] let us suppose it is desired to design a FIR filter having the system function $H(\omega)$. This function corresponds to the set of impulse coefficients $\{h(k);\ 0<k<\infty\}$. Simple truncation of this set to the finite length N causes the well known Gibbs phenomenon, that is, it creates fixed values of overshoot at points of discontinuity of $H(\omega)$ as well as values of ripple. To avoid this, coefficients h(k) are multiplied by weighting or smoothing coefficients w(k) (window coefficients) and the final finite impulse response coefficients, $\hat{h}(k)$, of the FIR filter are given by the set $\{\hat{h}(k) = h(k)\cdot w(k);\ 0\leq k\leq N\}$. This

set corresponds to a new system function $\hat{H}(\omega)$, which is an adequate approximation to the desired system function $H(\omega)$.

With respect to frequency sampling [9, 10], a desired continuous system function, $H(\omega)$, may be approximated by frequency samples at N equi-spaced points $\{\exp(j2\pi k/N); 0 \leq k \leq N-1\}$ around the unit circle. At these points, the sampled values are made exactly equal to the continuous values. At other points, the continuous values may be considered an interpolation of the sampled frequency response. The approximation can be greatly improved if certain sample values can be left unconstrained.

Lastly, the optimal filter design method [11, 12, 13, 14, 15] ensures that the peak approximation error to the systems function, taken over an entire interval, is minimized and is equiripple.

IIR filters are often designed by digitizing continuous filters, whose design techniques are well known [16]. One frequently used method is that of the bilinear transformation [17] which consists of mapping the left half of the s-plane into the interior of the unit circle in the Z-plane. This transformation causes the digital frequency to warp with respect to the analog frequency. Nevertheless, by means of a prewarping technique, the above method is often usefully applied. The other frequently used method of digitizing an

analog filter is called the impulse-invariant transformation [18] and ensures that the impulse response of the digital filter remains a sampled version of the impulse response of the analog filter.

Besides the above methods of digitizing an analog filter design, there also exist methods of direct digital filter design, either in the frequency domain or in the time domain [19,20,21].

Finally, there exist optimization methods for designing IIR filters, one method being based on a minimum mean-squared-error design [21].

IIR filters are generally more efficient than FIR filters in achieving given specifications on the magnitude response. On the other hand, FIR filters may have exactly linear phase characteristics, while IIR filters never do.

Many hardware structures exist for realizing digital filters, the most common of which are the direct form, the parallel form and the cascade form. Actual choice of form will depend on considerations of hardware economy and also on maintaining good accuracy in filter response when coefficient quantization and round-off noise are taken into account [22].

Apart from hardware realizations, many software techniques, notably the FFT, may be used for performing the filter function. One then speaks of software realization.

One-dimensional digital signal processing has been successfully applied to the areas of acoustics, bio-medicine, geology, radar and to other disciplines. These applications will probably increase for some time, as computing techniques and algorithms become more widely understood and while the cost of digital hardware, based on integrated circuits, continues to decrease.

1.2 Survey of Two-dimensional (2-D) Filtering

There are many inherently two-dimensional (2-D) signals, such as photographic data, for instance. These will include weather photographs taken by satellite cameras, space photographs, such as the terrain of the moon or that of Mars, taken by spacecraft cameras, bio-medical photographs, such as X-rays, taken by scanners and many others. Such photographic data may be expressed in terms of 2-D sample values, $x(n_1,n_2)$, where n_1 and n_2 are integer variables representing the two spatial dimensions and where the value $0 \leq |x(n_1 n_2)| \leq 1$ could, for instance, represent the shade of a black and white photograph, with black being assigned the value 1 and white the value zero for instance. The enhancement or

restoration of such photographic data by means of 2-D digital filters constitutes a useful and important digital signal processing problem.

Two-dimensional signals may be processed by applying the 1-D processing techniques, briefly discussed in the previous section. However, many important aspects of 1-D processing cannot readily be extended to two-dimensions. For instance, consider the n-th order polynomial, $H(z)$, which naturally arises as the Z-transform of the impulse response function $h(n)$. This polynomial may be factored

$$H(z) = (z-z_1)\cdot(z-z_2) \ldots\ldots (z-z_n)$$

where $z_1, z_2, \ldots\ldots z_n$ are the n-roots of $H(z) = 0$. No similar factorization property is known with respect to two-dimensional polynomials, $H(z_1,z_2)$, for instance, which would represent the 2-D impulse response function $h(n_1,n_2)$.

A 2-D digital filter is characterized by the difference equations

$$y(n_1,n_2) = \sum_{i=0}^{p}\sum_{j=0}^{q} a_{ij}x(n_1-i,\ n_2-j) - \sum_{\substack{i=0\\ i\neq j=0}}^{p}\sum_{j=0}^{q} b_{ij}y(n_1-i,n_2-j)$$

$$(0\leq n_1,n_2\leq\infty) \qquad (1.4)$$

Here, $y(n_1,n_2)$ is the output sample, $x(n_1,n_2)$ is the input sample and coefficients a_{ij}, b_{ij} characterize the filter. When $b_{ij}=0$ for all i,j except $b_{00}=1$, then the filter is said to be non-recursive or FIR. Otherwise the filter is recursive, or IIR, with recursion in the $+n_1,+n_2$ direction. Because $n_1,n_2 \geq 0$, the filter is causal. Moreover, for solution, a set of initial conditions $\{y(m,n);\ -p \leq m;\ -q \leq n;\ mn \neq 0\}$ is necessary. For stability, it is necessary that the impulse response is bounded.

$$\sum_{n_1=0}^{\infty} \sum_{n_2=0}^{\infty} |h(n_1,n_2)| < \infty \qquad (1.5)$$

where

$$H(z_1,z_2) = \sum_{n_1=0}^{\infty} \sum_{n_2=0}^{\infty} h(n_1,n_2) z_1^{-n_1} z_2^{-n_2} = \frac{A(z_1,z_2)}{B(z_1,z_2)} \qquad (1.6)$$

and

$$A(z_1,z_2) = \sum_{i=0}^{p} \sum_{j=0}^{q} a_{ij} z_1^{-i} z_2^{-j}$$

$$B(z_1,z_2) = \sum_{i=0}^{p} \sum_{j=0}^{q} b_{ij} z_1^{-i} z_2^{-j}$$

The stability determination of IIR filters is often difficult and time consuming, although modified stability theorems due to Shanks [23] and due to Huang [24] have greatly reduced computation time. The design of 2-D FIR filters completely avoids the stability problem. Design

methods based on windowing [25], on frequency sampling, and on optimization, similar to 1-D methods described in section 1.1, but now extended to 2-D, have been successfully used.

Another very useful design method for 2-D FIR filters is based on a frequency transformation from one to two dimensions, which is often referred to as the McClellan transformation [26]. An optimally designed FIR filter, with a N-point impulse response, has a frequency response

$$H[\exp(j2\pi f)] = \sum_{n=0}^{(N-1)/2} b(n)C(2\pi fn)$$

$$= \sum_{n=0}^{(N-1)/2} \hat{b}(n)C^n(2\pi f) \qquad (1.7)$$

where $C(x)=\cos x$. By substituting the McClellan transformation

$$2C(2\pi f) = C(2\pi f_1) + C(2\pi f_2) + C(2\pi f_1)C(2\pi f_2) - 1 \qquad (1.8)$$

into equ. (1.7) one obtains, after some algebra

$$H[(\exp[j2\pi f_1),\exp(j2\pi f_2)] = \sum_{n_1=0}^{(N_1-1)/2} \sum_{n_2=0}^{(N_2-1)/2} a(n_1,n_2) \cdot$$

$$C(2\pi f_1 n_1)C(2\pi f_2 n_2) \qquad (1.9)$$

which is the desired form for a 2-D linear-phase FIR filter.

The resulting 2-D approximation given by equ. (1.9) frequently preserves the optimality criterion used in the design of the 1-D filter. There are, however, certain questions that are raised by such a transformation, which we briefly mention, without examination. First, the McClellan transformation is only one of a possible large number of suitable transformations. How does one systematically find the best transformation? Secondly, although it is always possible to project a 2-D complex into one dimension, creating a 2-D complex from one dimension is more restrictive. How restrictive is a transformation like the McClellan transformation and what design flexibility is lost by its use?

A second 2-D FIR filter design method using 1-D filter design knowledge is based on the assumption of system function separability. A system function $H(z_1, z_2)$ is said to be separable if it can be expressed in the form

$$H(z_1, z_2) = H(z_1)H(z_2) \tag{1.10}$$

The 2-D system function $H(z_1, z_2)$ may now be considered the tandem connection of two 1-D system functions. An extended separability criterion will be discussed in chapter 2, in connection with the design of a new class of 2-D rectangular digital filters.

The software realization of 2-D FIR filters may be either by direct convolution or by the FFT technique. Direct convolution consists of carrying out the operation

$$y(n_1,n_2) = \sum_{m_1=0}^{M_h-1} \sum_{m_2=0}^{N_h-1} h(m_1,m_2)x(n_1-m_1,n_2-m_2) \tag{1.11}$$

where $h(m_1,m_2)$ is a M_h x N_h array. When the input array $x(n_1,n_2)$ is M_xN_x, then the total number of multiplication and additions, and therefore the total machine time is proportional to $M_xM_hN_xN_h$.

The FFT technique first computes the DFT's $X(k,\ell)$ of $x(n_1,n_2)$ and $H(k,\ell)$ of $h(n_1,n_2)$ and so

$$Y(k,\ell) = H(k,\ell)X(k,\ell) \tag{1.12}$$

By inverse DFT, one now obtains $y(n_1,n_2)$. The computation time needed may be shown to be proportional to [28] $(M_x+N_x)(M_h+N_h)\text{Log}_2[(M_x+N_x)(M_h+N_h)]$.

A 2-D digital filter is used to reshape the frequency spectrum of 2-D digital signals. Two kinds of such filters have found recent use. The first type utilizes circular symmetry in the frequency domain. That is, a desired system function H is approximated by $\hat{H}$ such that

$$H[\exp(j\omega_1),\exp(j\omega_2)] \approx \hat{H}[\exp(j\sqrt{\omega_1^2+\omega_2^2})] \tag{1.13}$$

A typical circular symmetric system function is shown in Fig. 1-1a. This type of filter is mathematically convenient and has been used for picture processing [29] .

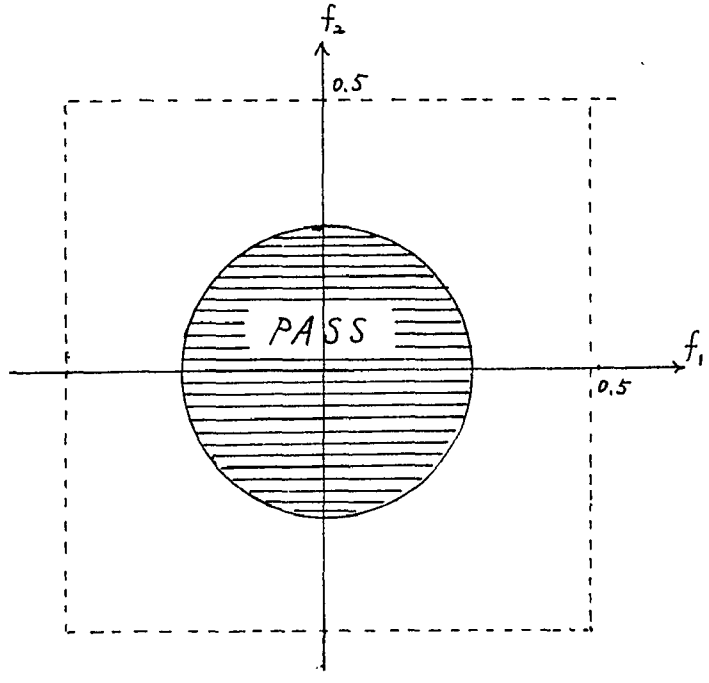
Another filter that has found practical application in geology is the fan filter, a typical characteristic of which is shown in Fig. 1-1b. This type of filter is used in seismic record processing and effectively rejects high-velocity disturbances, while passing low-velocity ones.

A new rectangular filter class, mentioned below, forms the subject of Chapter 2.

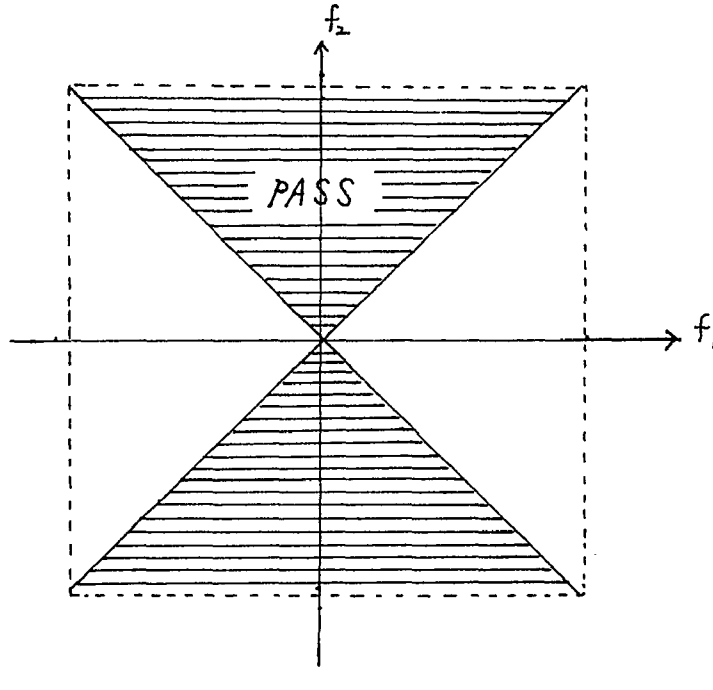
Now folows an outline of the thesis problems and the author's contributions.

1.3 Statement of Thesis Problems.

The first problem investigated in this thesis is the proposal of a new rectangular filter class. It is shown that if these filters are separable and of low degree - as described in Chapter 2 - then they can be easily designed and realized. In addition, these filters can be designed to meet special frequency requirements, for instance, those of a star-shaped filter which is used for some 2-D picture processing purposes.



(a)



(b)

Fig. 1-1 2-D circular symmetric filter and fan filter

The second problem investigated is a 2-D filter realization, which enables 2-D data to be processed in parallel form. As a consequence of this new realization, real-time processing is possible. Low-order FIR and IIR filters are realized by using unit delays, adders and multipliers. Some timing problems, due to the inherent feedback of IIR filters, are also discussed. Chapter 4 generalizes the realization method to higher order filters, by replacing all multipliers with ROMs and thereby reducing the hardware requirements. Further hardware saving in the McClellan-transformed FIR filter case is made possible by the author's introduction of a code converter before ROM and the factorization of a 1-D system function before it was transformed into a 2-D function.

The third problem investigated is a cost saving high-speed processing method which, however, is not as fast as real-time parallel processing. The first method is a 2-D section-by-section with overlap-data processing technique. A second method utilizes commercially available microprocessors to replace single-channels in the hardware structure. The processing speed with microprocessors is high speed in the low order filter case, but becomes low speed when the filter order is high. Future applications of parallel 2-D processing are also mentioned. The author's contributions are the 2-D section-by-section with data-overlap processing technique and the proposed microprocessor replacement in the 2-D filter structure.

CHAPTER TWO

Design of Two Dimensional Rectangular Digital Filters

2.1 Introduction

The word "rectangular" in the above filter class refers to the geometric shape of the frequency spectrum. A typical low-pass characteristic in both the ω_1 and ω_2 frequency axes is given by Fig. 2-1, for which

$$H[\exp(j\omega_1),\ \exp(j\omega_2)] = \begin{cases} 1, & |\omega_1| \leq \omega_{10};\ |\omega_2| \leq \omega_{20} \\ 0, & \text{elsewhere} \end{cases}$$

Note that all system functions are doubly periodic in frequency, that is

$$H[\exp(j\omega_1),\ \exp(j\omega_2)] = H[\exp(j\omega_1+j2\pi L),\exp(j\omega_2+j2\pi M)] \tag{2.1}$$

where L,M are integers, and so all frequency plots need only display the range $-\pi \leq \omega_1 \leq \pi$, $-\pi \leq \omega_2 \leq \pi$. A system function is said to be "separable to degree N" if it can be expressed in the form

$$H[\exp(j\omega_1, j\omega_2)] = \sum_{r=1}^{N} H_{1r}[\exp(j\omega_1)] \cdot H_{2r}[\exp(j\omega_2)] \tag{2.2}$$

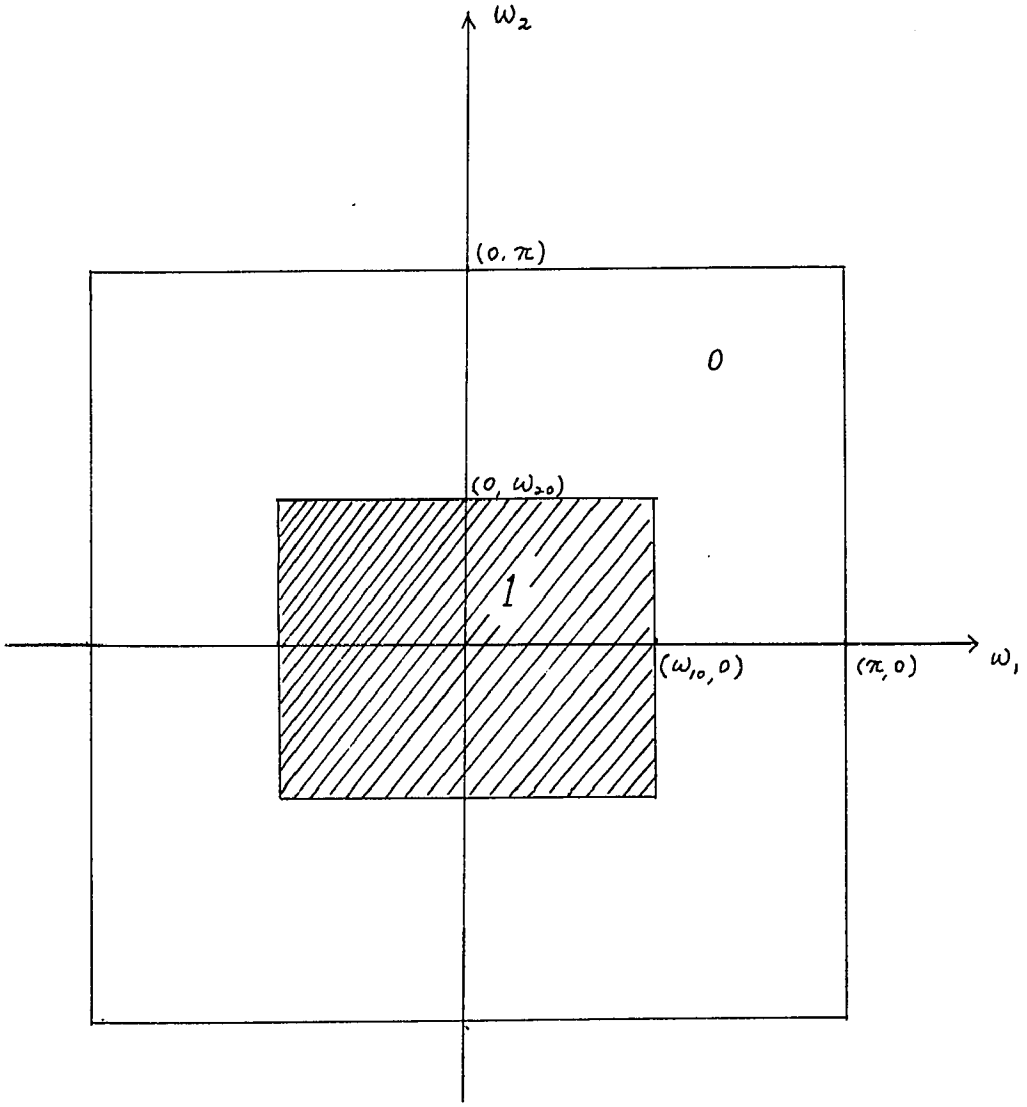


Fig. 2-1 A 2-D rectangular symmetric lowpass, lowpass filter

If the system function is separable to degree N, so also is the impulse response function $h(n_1,n_2)$. This follows on substituting equ. (2.2) into the relation

$$h(n_1,n_2) = \frac{1}{4\pi^2} \int_{-\pi}^{\pi} \int_{-\pi}^{\pi} H[\exp(j\omega_1), \exp(j\omega_2)] \exp(j\omega_1 n_1)$$

$$\exp(j\omega_2 n_2)\, d\omega_1 d\omega_2 =$$

$$\sum_{r=1}^{N} \{\frac{1}{2\pi} \int_{-\pi}^{\pi} H_{1r}[\exp(j\omega_1)] \exp(j\omega_1 n_1)\, d\omega_1\} \{\frac{1}{2\pi} \int_{-\pi}^{\pi} H_{2r}[\exp(j\omega_2)]$$

$$\exp(j\omega_2 r_2)\, d\omega_2\} =$$

$$\sum_{r=1}^{N} h_{1r}(n_1) h_{2r}(n_2) \tag{2.3}$$

where

$$h_{1r}(n_1) = \frac{1}{2\pi} \int_{-\pi}^{\pi} H_{1r}[\exp(j\omega_1)] \exp(j\omega_1 n_1)\, d\omega_1$$

$$h_{2r}(n_2) = \frac{1}{2\pi} \int_{-\pi}^{\pi} H_{2r}[\exp(j\omega_2)] \exp(j\omega_2 n_2)\, d\omega_2 \tag{2.4}$$

The converse relationship also holds and follows from the definition

$$H[\exp(j\omega_1), \exp(j\omega_2)] = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} h(n_1, n_2) \exp(-j\omega_1 n_1) \exp(-j\omega_2 n_2) \tag{2.5}$$

For the case of the lowpass filter depicted in Fig. 2-1, one has

$$H[\exp(j\omega_1),\exp(j\omega_2)]=\begin{cases}1 & -\omega_{10}\leq\omega_1\leq\omega_{10} \\ & -\omega_{20}\leq\omega_2\leq\omega_{20} \\ 0 & \text{elsewhere}\end{cases}$$

Application of equ.(2.3) now yields

$$h_{11}(n_1)=\frac{\sin(\omega_{10}n_1)}{\pi n_1}\ ;\ h_{21}(n_2)=\frac{\sin(\omega_{20}n_2)}{\pi n_2} \tag{2.6}$$

and so the impulse response function is separable to degree one.

For rectangular 2-D filters, of which Fig. 2-1 is characteristic, the system function displays both vertical and horizontal symmetry. That is

$$H[\exp(j\omega_1),\ \exp(j\omega_2)]=H[\exp(-j\omega_1),\ \exp(j\omega_2)] \quad \text{(Vertical symmetry)}$$

$$H[\exp(j\omega_1),\exp(j\omega_2)]=H[\exp(j\omega_1),\exp(-j\omega_2)] \quad \text{(Horizontal symmetry)}$$

For a filter separable to degree N, this implies

$$H_{1r}[\exp(j\omega_1)]=H_{1r}[\exp(-j\omega_1)] \qquad \forall r$$

$$H_{2r}[\exp(j\omega_2)]=H_{2r}(\exp(-j\omega_2)] \qquad \forall r \tag{2.7}$$

that is both H_{1r} and H_{2r} are even functions of ω_1 and ω_2. In addition, all system functions corresponding to real impulse-response functions $h(n_1,n_2)$ display the symmetry

$$H[\exp(j\omega_1),\exp(j\omega_2)]$$

$$= \sum_{m_1=-\infty}^{\infty} \sum_{m_2=-\infty}^{\infty} h(m_1,m_2)\exp(-j\omega_1 m_1)\exp(-j\omega_2 m_2)$$

$$= \sum_{m_1=-\infty}^{\infty} \sum_{m_2=-\infty}^{\infty} h^*(m_1,m_2)\exp^*(j\omega_1 m_1)\exp^*(j\omega_2 m_2)$$

$$= H^*[\exp(-j\omega_1),\exp(-j\omega_2)] \qquad (2.8)$$

That is, knowledge of H in the first quadrant implies knowledge of H in the third quadrant and similar remarks apply to H in the second and fourth quadrants. Now follow some simple definitions of 2-D rectangular filters.

2.2 Definitions of Some Simple 2-D Rectangular Filters

Filters will be divided into filter classes, specified in the form$\{\omega_1,\omega_2\}$. Thus, a lowpass characteristic in both the ω_1 and ω_2 coordinate axes would be denoted by{lowpass, lowpass}. Moreover, the ideal filter characteristics about to be described are unrealistic in the sense that no transition

band between passband and stopband is specified, and so "brickwall type" filters, which are known to be noncausal, are assumed.

Lowpass filter:

An ideal {lowpass,lowpass} rectangular filter obeys the frequency response:

$$H[\exp(j\omega_1),\exp(j\omega_2)] = \begin{cases} 1, |\omega_1| \leq \omega_{10}, |\omega_2| \leq \omega_{20}; \text{ passband.} \\ 0, \text{ elsewhere; stopband.} \end{cases}$$

This filter is shown in Fig. 2-1, with the passband shaded. Note that the impulse response function is separable to degree N = 1, and was derived by (2.6).

Bandpass filter:

An ideal {bandpass,bandpass} rectangular filter obeys the frequency response

$$H[\exp(j\omega_1),\exp(j\omega_2)] = \begin{cases} 1, \ \omega_{10} \leq |\omega_1| \leq \omega_{11}, \omega_{20} \leq |\omega_2| \leq \omega_{21}; \text{passband} \\ 0, \text{ elsewhere; stopband} \end{cases}$$

This type of filter is shown in Fig. 2-2, with the passband shaded. The impulse-response function is separable to degree

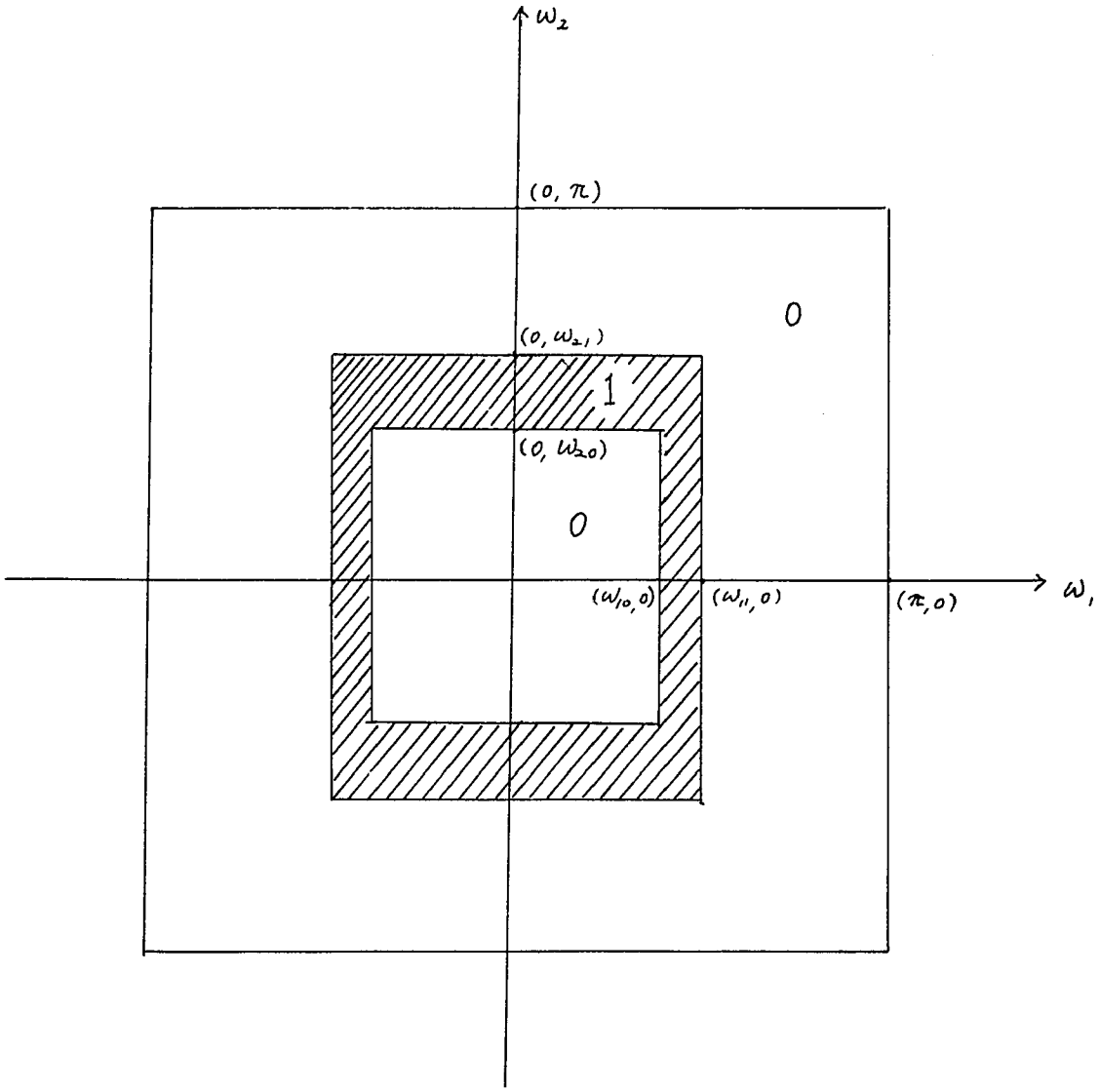


Fig. 2-2 A 2-D rectangular symmetric bandpass,bandpass filter

N = 2 and is given by

$$h(n_1,n_2) = \frac{1}{\pi^2}\{\frac{\sin b_1 n_1}{n_1}\ \frac{\sin b_2 n_2}{n_2} - \frac{\sin a_1 n_1}{n_1}\ \frac{\sin a_2 n_2}{n_2}\} \tag{2.9}$$

Note that the {lowpass,lowpass} filter is derivable from the {bandpass,bandpass} filter by putting $a_1=a_2=0$, $\omega_{10}=b_1$, $\omega_{20}=b_2$, when (2.5) is reobtained. Similarly a {highpass, highpass} filter results, when $b_1=b_2=\pi$, $a_1=\omega_{10}, a_2=\omega_{20}$. The frequency spectrum of the {highpass,highpass} filter is shown in Fig. 2-3. The three filter types, lowpass,bandpass and highpass are basic and may be combined in various ways, to be shown subsequently, in order to obtain a wide variety of filters. The structure of such filters in certain simple cases will now be described.

2.3 Simple 2-D Rectangular Filter Structures

The separability of the system function immediately suggests a filter structure. Since the output of a 2-D filter, $y(n_1,n_2)$, is related to the input, $x(n_1,n_2)$ by the system function

$$Y[\exp(j\omega_1),\exp(j\omega_2)]=X[\exp(j\omega_1),\exp(j\omega_2)]H[\exp(j\omega_1),\exp(j\omega_2)] \tag{2.10}$$

therefore in the case of a system function separable to degree one,

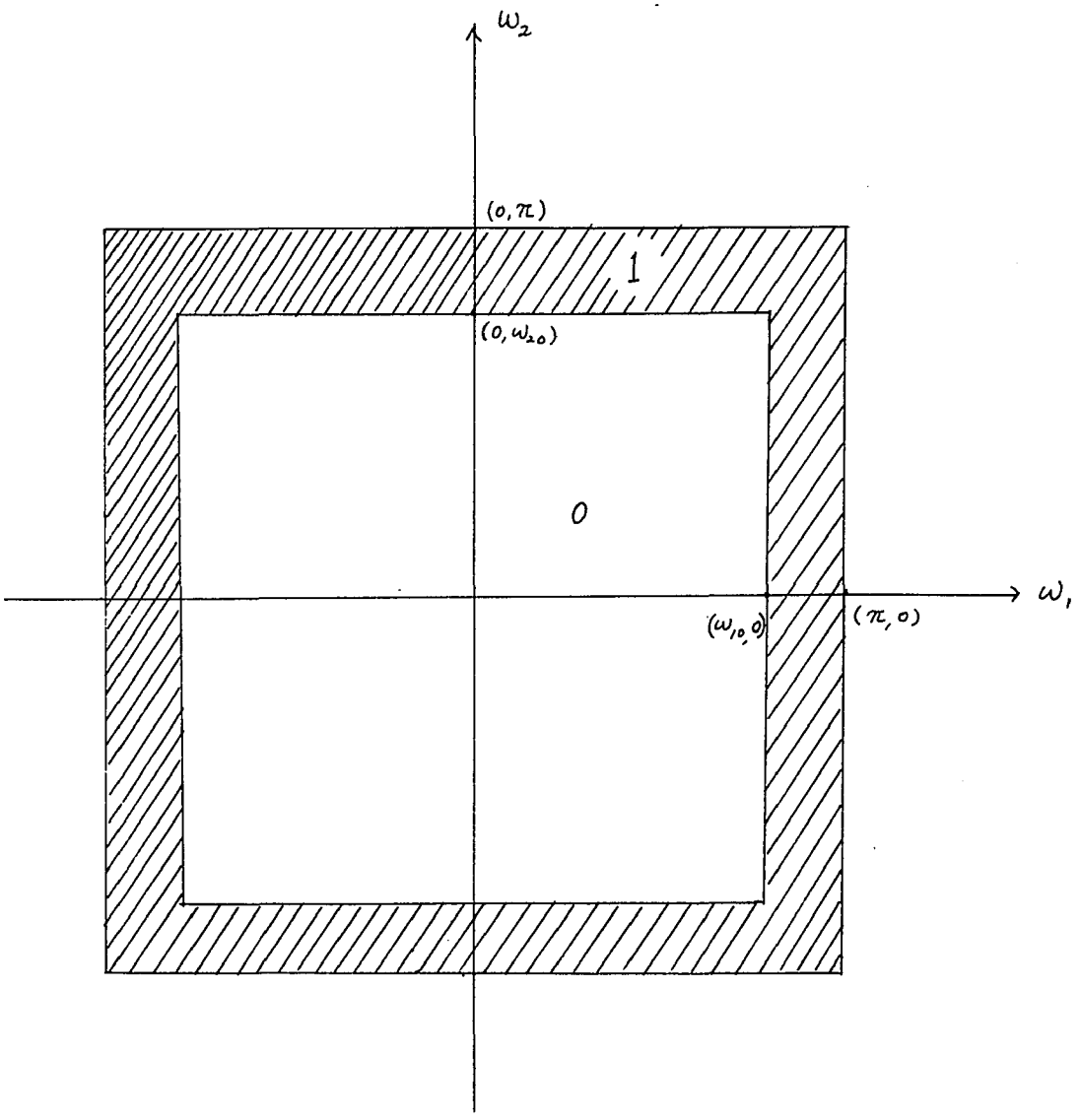


Fig. 2-3 A 2-D rectangular symmetric highpass,highpass filter

$$Y[\exp(j\omega_1),\exp(j\omega_2)]$$

$$=X[\exp(j\omega_1),\exp(j\omega_2)]\cdot H_1(\exp(j\omega_1)]\cdot H_2[\exp(j\omega_2)] \qquad (2.11)$$

One can therefore consider H_1 or H_2 as a tandem connection shown in Fig. 2-4a, with $H_1[\exp(j\omega_1)]$ only acting on variable n_1 and $H_2[\exp(j\omega_2)]$ only acting on variable n_2. Such a tandem connection can realize a number of interesting filter shapes; some of which are shown in Fig. 2-5. Fig.2-5a shows a {lowpass,lowpass} filter, Fig.2-5b shows a {bandpass,lowpass} filter and Fig.2-5c shows a {highpass,lowpass} filter. The individual 1-D filter functions, H_1 and H_2, may be chosen as FIR filters or as IIR filters. Recollect that a 1-D, N-point FIR filter will possess linear phase,

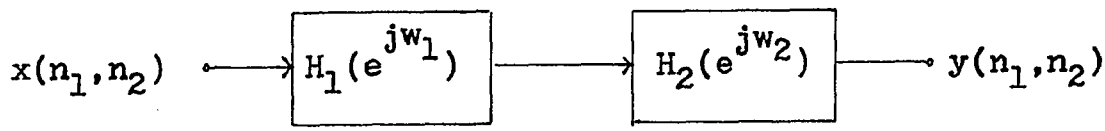
$$H_1[\exp(j\omega_1)] = \pm|H_1[\exp(j\omega_1)]|\exp(j\alpha\omega_1)$$

provided the phase constant α and impulse response h(n) satisfy the relations

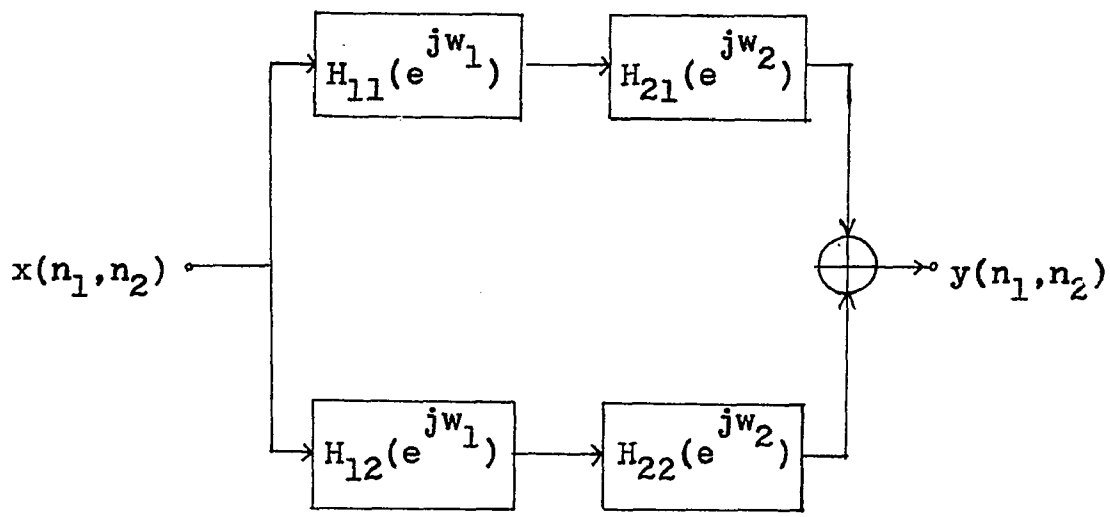
$$\alpha = (N-1)/2;\ h(n)=h(N-1-n), \quad 0\leq n\leq N-1 \qquad (2.12)$$

whereas IIR filters always have non-linear phase. The magnitude of the system function in the tandem case of (2.11) depends only on the magnitudes of the individual 1-D component magnitudes $|H_1|$ and $|H_2|$, regardless whether these are FIR or IIR, because

$$|H_1[\exp(j\omega_1),\exp(j\omega_2)]|=|H_1[\exp(j\omega_1)]||H_2[\exp(j\omega_2)]|$$



(a)



(b)

Fig. 2-4 Block diagram for designing 2-D rectangular symmetric filters by serial and parallel combination of 1-D filters

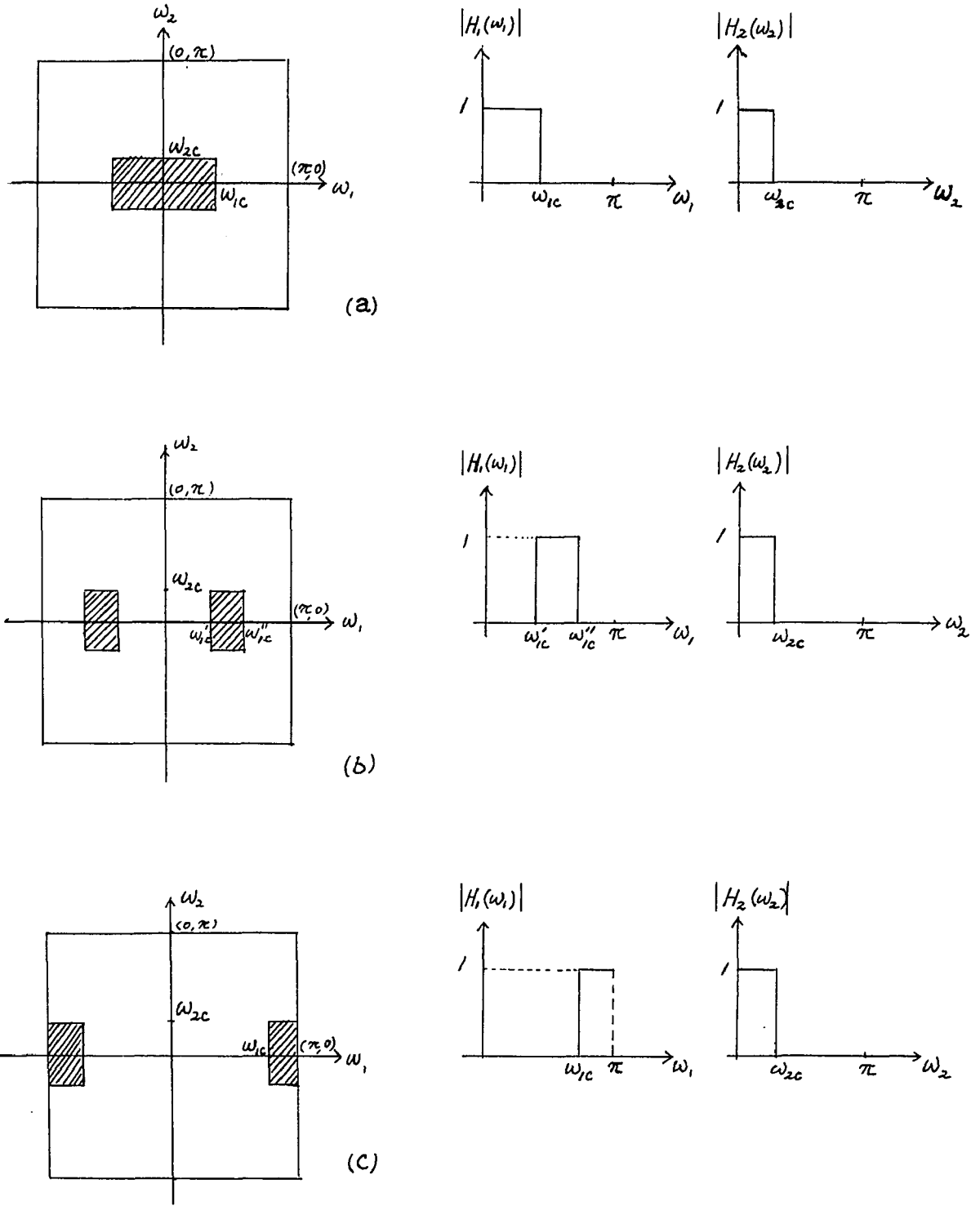


Fig. 2-5 Frequency spectra of 2-D rectangular symmetric filters obtained by cascading 1-D filters

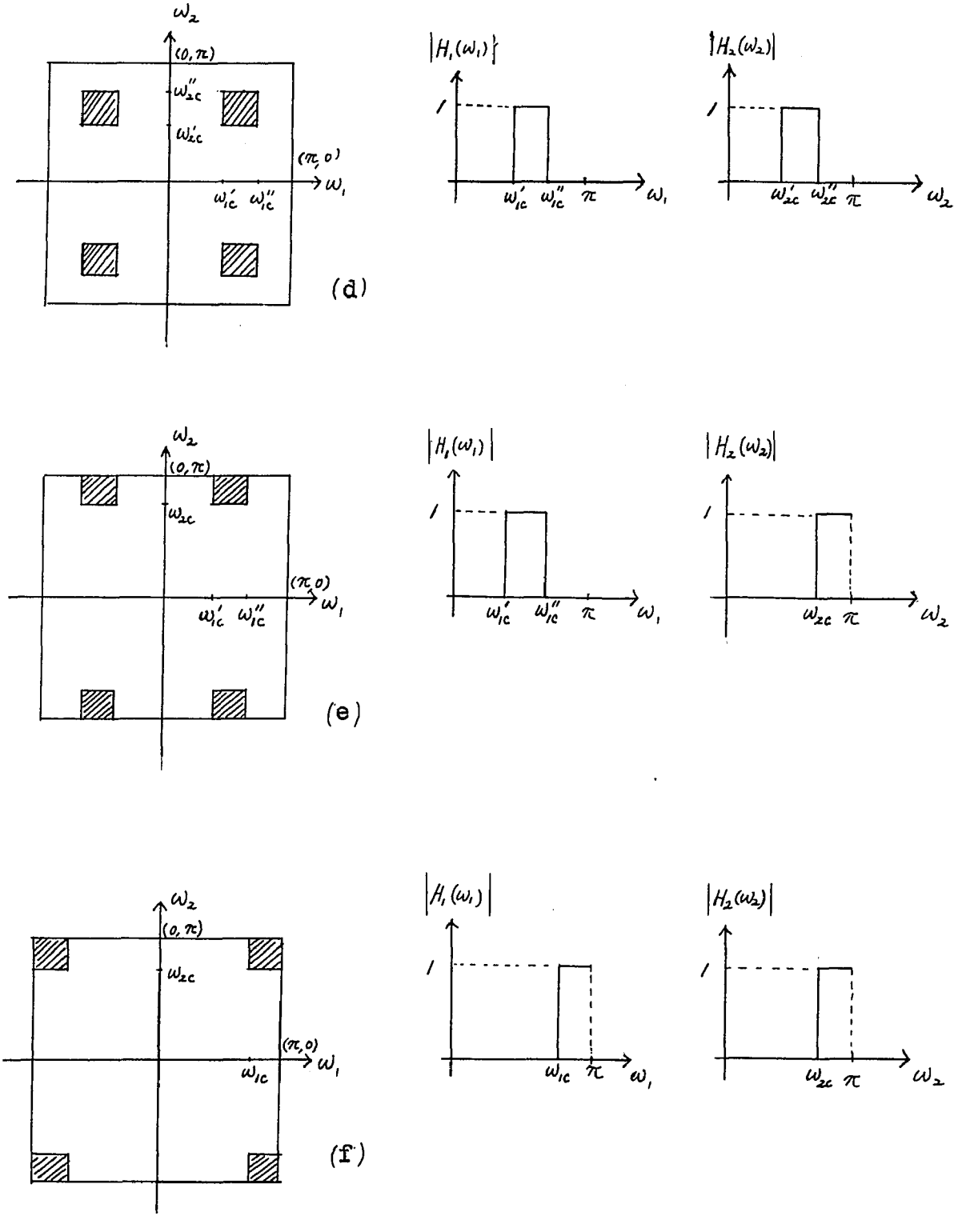


Fig. 2-5 Continue

However, in the case of a system function separable to degree N = 2, one has

$$H[\exp(j\omega_1),\exp(j\omega_2)] = H_{11}[\exp(j\omega_1)]H_{21}[\exp(j\omega_2)]+H_{12}[\exp(j\omega_1)]H_{22}[\exp(j\omega_2)] \tag{2.13}$$

where $H_{11}[\exp(j\omega_1)]=|H_{11}[\exp(j\omega_1)]|\cdot\exp[j\theta_{11}(\omega_1)]$ and similarly for H_{21}, H_{12} and H_{22}. The corresponding filter structure is shown in Fig. 2-4b. The magnitude of H is now given by

$$|H|=||H_{11}H_{21}|\exp(j\theta_{11}+j\theta_{21})+|H_{12}H_{22}|\exp(j\theta_{12}+j\theta_{22})| \tag{2.14}$$

Hence the magnitude of H depends not only on the 1-D magnitude components H_{11}, H_{21}, H_{12} and H_{22} but also on the phase components θ_{11}, θ_{21}, θ_{12} and θ_{22}. Dependence upon phase components may be removed by making all 1-D filters FIR linear phase filters, and choosing $\theta_{11}=-\alpha_{11}\omega_1$, $\theta_{21}=-\alpha_{21}\omega_2$, etc., here

$$\alpha_{11} = \alpha_{12} = (N_1-1)/2;\ \alpha_{21} = \alpha_{22} = (N_2-1)/2 \tag{2.15}$$

In that case

$$|H[\exp(j\omega_1),\exp(j\omega_2)]| = |H_{11}H_{21}|+|H_{12}H_{22}| \tag{2.16}$$

and the 1-D magnitudes may be considered to add in the parallel-series manner of Fig. 2-6a through f, which shows a

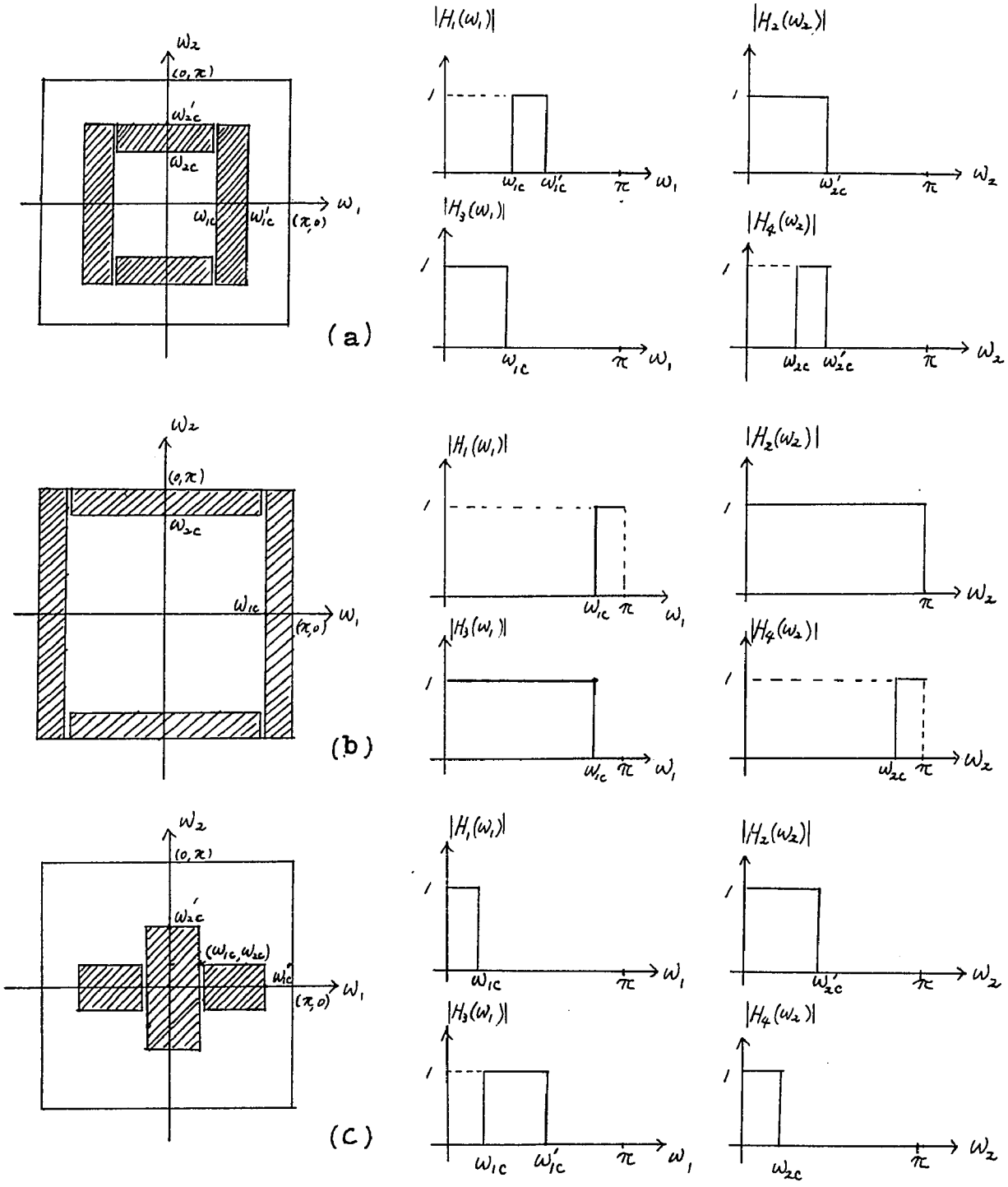


Fig. 2-6 Frequency spectra of 2-D rectangular symmetric filters obtained by a parallel combination of 1-D filters

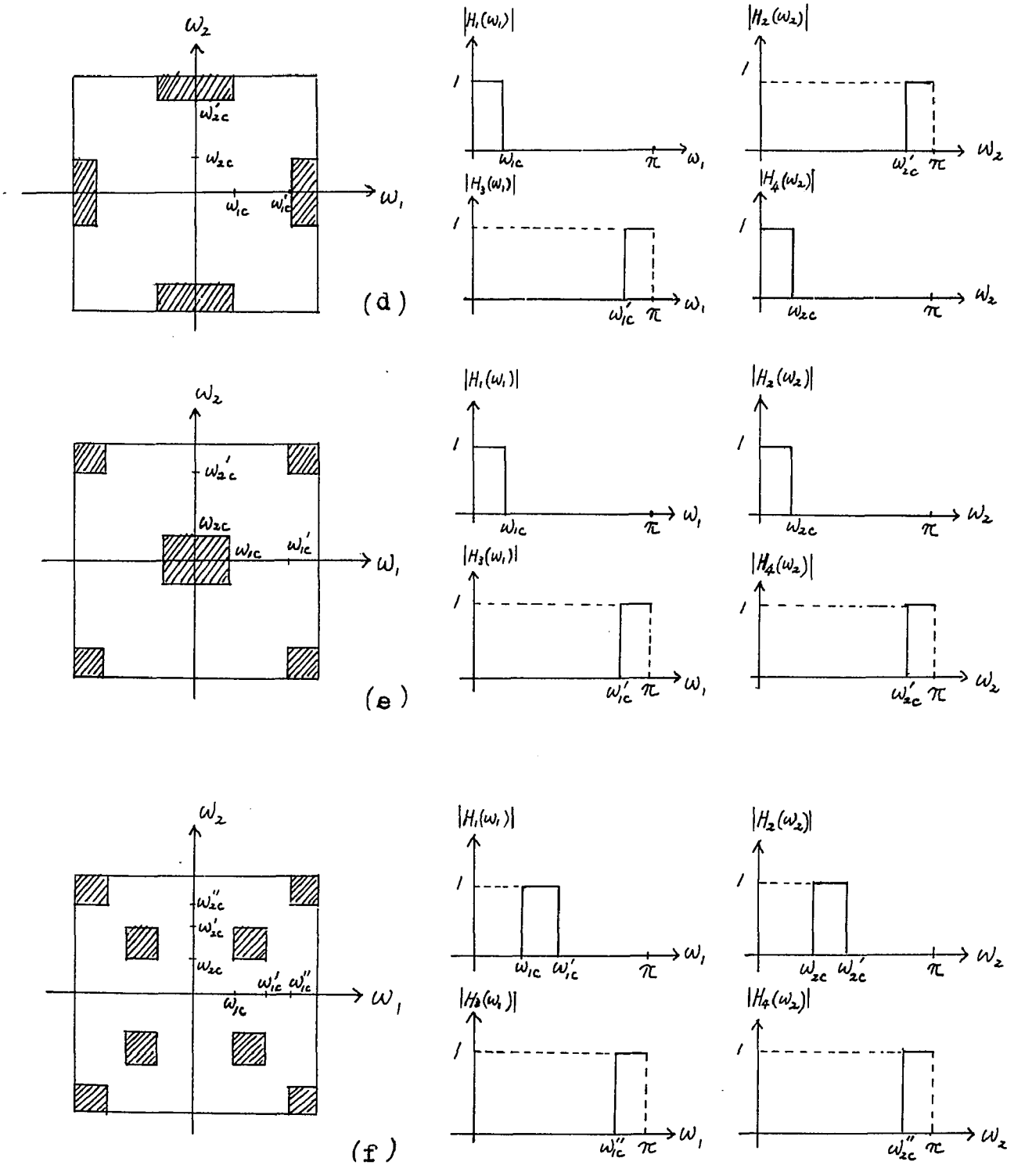


Fig. 2-6 Continue

variety of possible filter characteristics. If $H_{11}=H_{12}=H_1$, $H_{21}=H_2$, $H_{22}=H_3$, the filter structure shown in Fig. 2-7 is obtained. Fig. 2-8a through c shows those possible magnitude characteristics based on this filter structure. Practical filters will now be discussed by some examples.

2.4 Design Examples of Simple 2-D Rectangular Filters

Real 2-D filters, which will now be designed, differ from the ideal filter types, by the existence of a transition band in which the absolute value of the system function is smaller than unity but greater than zero, that is $0<|H|<1$. Moreover, the system function in the stopband cannot be made zero, but usually differs from zero by the existence of a small amount of ripple, say $20\text{Log}_{10}\varepsilon_s$ db. Similarly, the system function in the passband is not equal to unit (or prescribed value), but differs from the required value by $20\text{Log}_{10}(1+\varepsilon_p)$ db of ripple. The edges of the transition band are specified by the cut off frequencies f_{1p}, f_{1s}, f_{2p}, f_{2s} etc., where p stands for passband and s stands for stopband. This is illustrated for the case of a 1-D filter in Fig. 2-9a and in case of a 2-D {lowpass,lowpass} filter by Fig. 2-9b. For all other filter types similar remarks apply. The actual filter specification is supplied to the designer, who then has to come up with a suitable design. This he may do by varying the

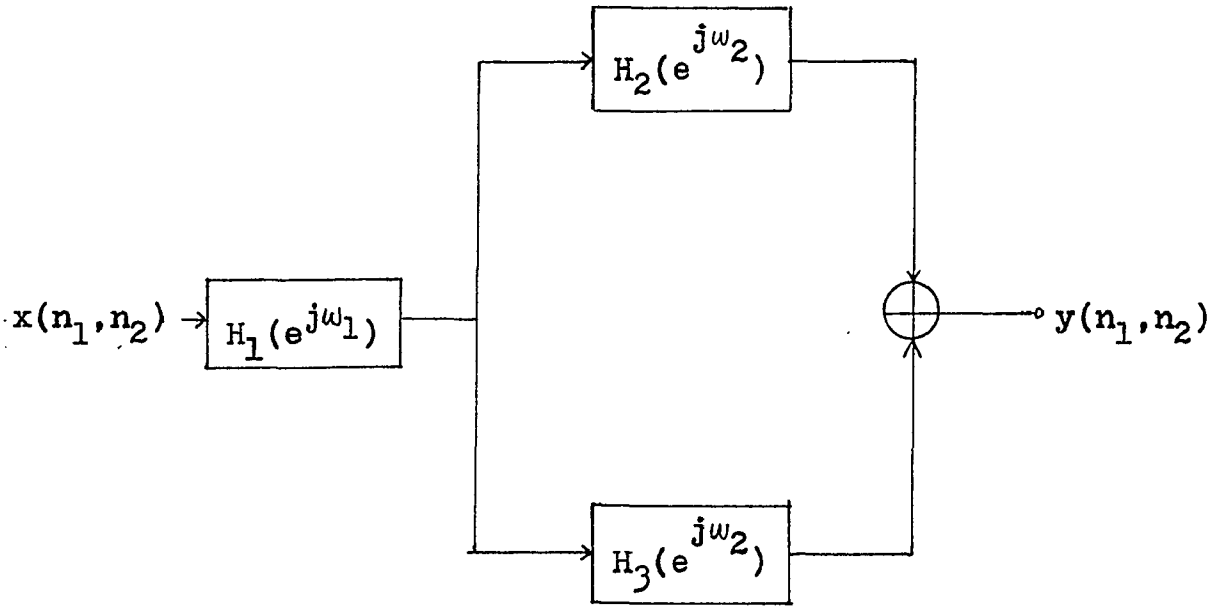


Fig. 2-7 Block diagram for designing 2-D rectangular symmetric filters by both cascading and paralleling 1-D filters

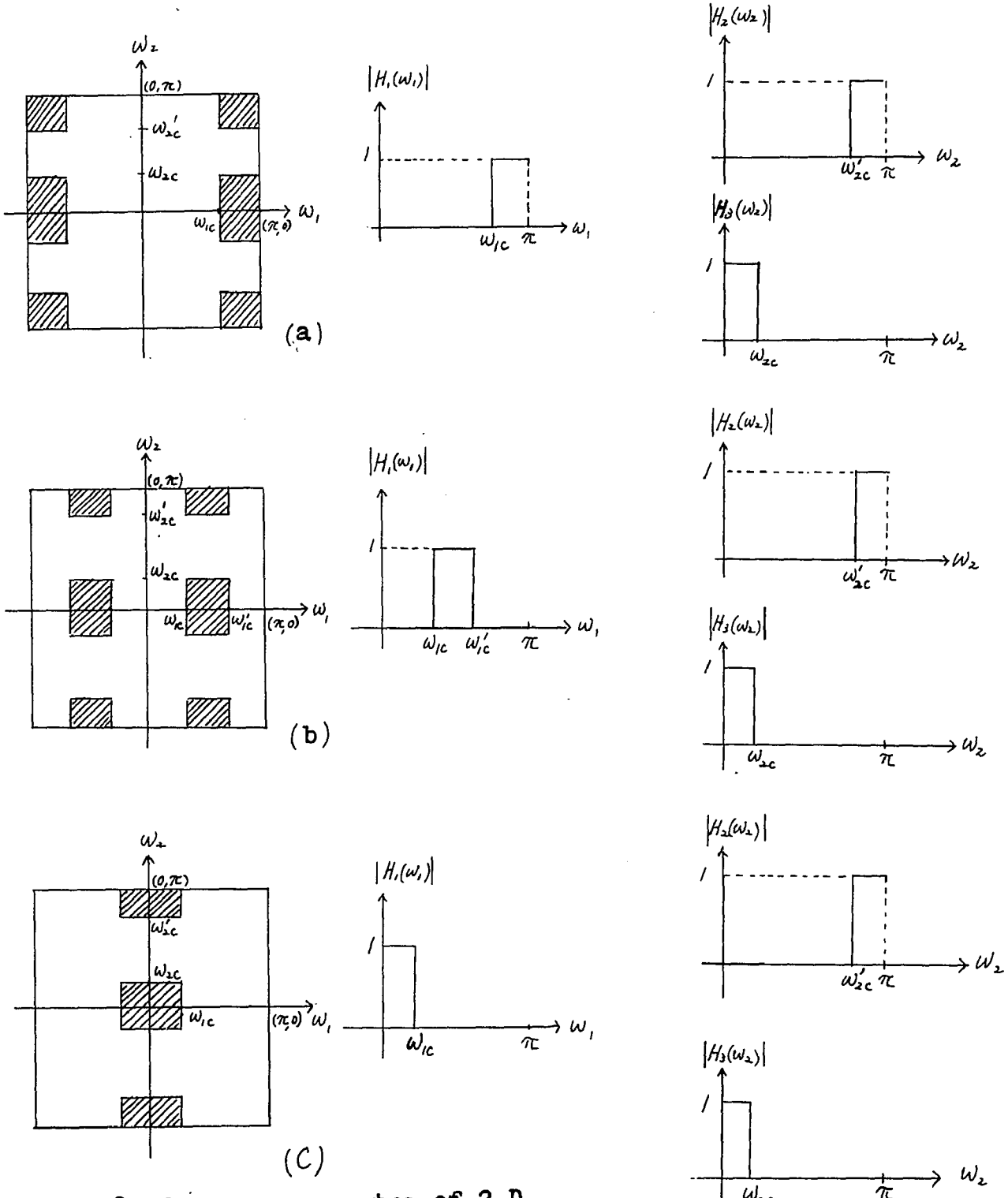
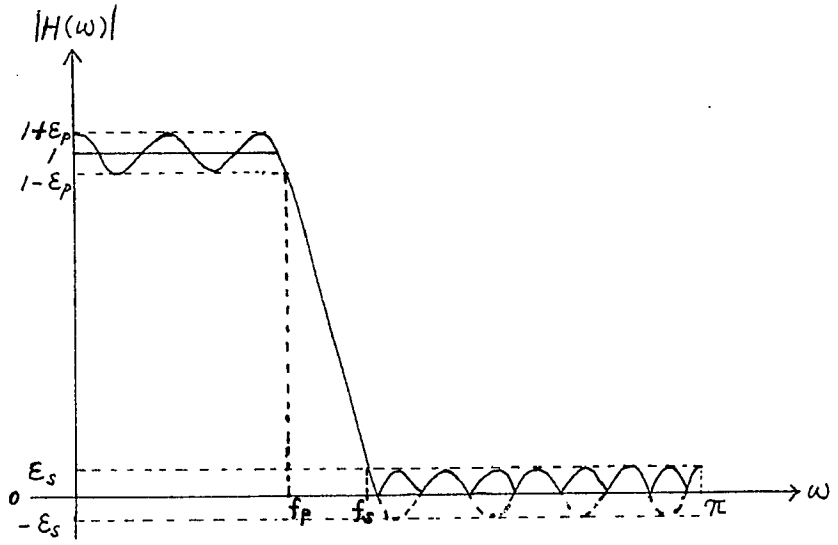
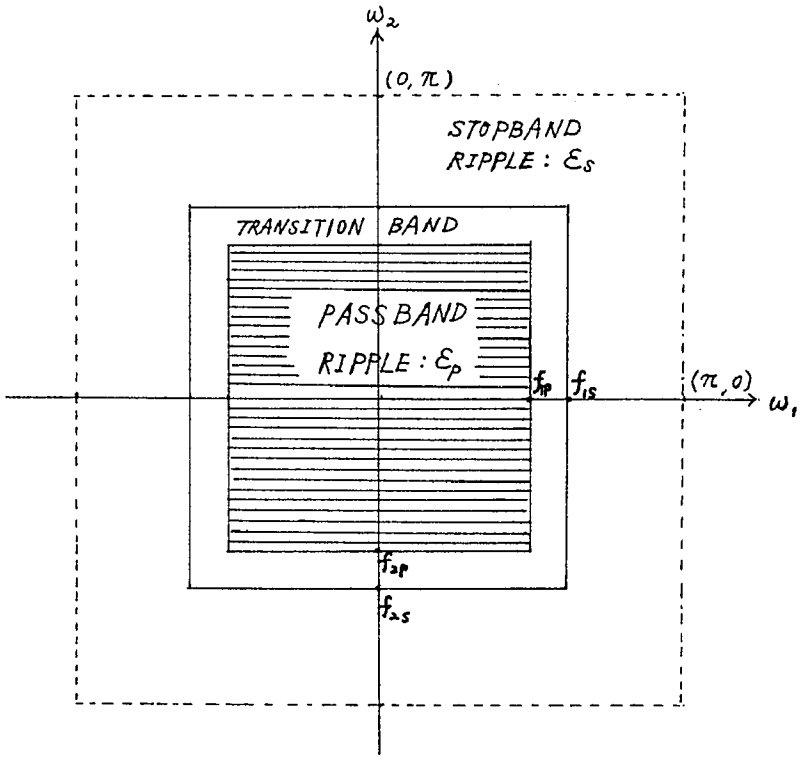


Fig. 2-8 Frequency spectra of 2-D rectangular symmetric filters obtained by both cascading and paralleling 1-D filters



(a)



(b)

Fig. 2-9 Characteristics of practical 1-D filters and 2-D rectangular symmetric filters

order N_1, N_2 of the filter, and also by selecting a filter structure most suited to meet the required specification. When linear phase is required in both axes, as in the following design examples, FIR filters must be used and N_1 is usually chosen equal to N_2, although this choice is not essential. Normally, for other filter design, the use of IIR filters allows the choice of Chebyshev and Elliptic filters, and this choice tends to reduce the order of both N_1 and N_2.

In all plots previously presented, the frequency was already normalized with respect to the sampling period T, because $\omega'=\omega T$. Furthermore $T = 1/f_s$ where f_s is the sampling frequency and $-\pi \leq \omega' \leq \pi$. Substituting $\omega' = 2\pi f'$, one obtains $-0.5 \leq f' \leq 0.5$, where f' is the normalized frequency, that is $f' = f/f_s$. Because of vertical and horizontal symmetry, only the first quadrant needs to be plotted in subsequent examples. Now follow some simple examples.

2.4.1 Example 1

Design of a 2-D {lowpass,lowpass} filter with the following specifications:

Cut off frequencies: $f_{1p} = 0.1, f_{1s} = 0.15$; $f_{2p} = 0.2, f_{2s} = 0.25$

The maximum deviation in the passband is 0.6 db (passband ripple). The stopband is to be at least 29 db down.

The given specification can probably be met by the simplest possible structure, namely the tandem section

illustrated in Fig. 2-4a. Recollect that for this structure $|H(\exp j\omega_1, \exp j\omega_2)| = |H_1(\exp j\omega_1)| \cdot |H_2(\exp j\omega_2)|$. The worst deviation of $|H|$ in the passband, where nominally $|H| = 1$ is by the amount ε_p such that $1-\varepsilon_p \leq |H| \leq 1+\varepsilon_p$. When this is the case $|H_1|$ will differ from unity by ε_{1p} and $|H_2|$ will differ from unity by ε_{2p}, such that in the worst case

$$|1+\varepsilon_p| = |1+\varepsilon_{1p}||1+\varepsilon_{2p}|$$

that is

$$\varepsilon_p = \varepsilon_{1p} + \varepsilon_{2p} + \varepsilon_{1p}\varepsilon_{2p}$$

$$\simeq \varepsilon_{1p} + \varepsilon_{2p} \qquad (\varepsilon_{1p}, \varepsilon_{2p} \ll 1)$$

(2.17)

Since $20\text{Log}_{10}(1+\varepsilon_p) = 0.6$, therefore $\varepsilon_p = 0.071519$. Allowing each dimension to contribute half this value then gives $\varepsilon_{1p} = \varepsilon_{2p} = 0.03576$.

The stopband deviation, ε_s, is related to the specification "x db down" by the relation $20\text{Log}_{10}\varepsilon_s = -x$, and so for this example $20\text{Log}_{10}\varepsilon_s = -29$. Hence, $\varepsilon_s = 0.03458$

For the stopband ripple there exist three possible regions, namely:

1). f_1 in stopband
f_2 in stopband
$\}$ $f_{1s} \leq f_1 \leq 0.5$, $0 \leq f_2 \leq f_{2p}$

$|\varepsilon_s| = |\varepsilon_{1s}||1+\varepsilon_{2p}|$ or $\varepsilon_s \simeq \varepsilon_{1s}$

2). f_1 in passband $(0 \leq f_1 \leq f_{1p})$
f_2 in stopband $(f_{2s} \leq f_2 \leq 0.5)$
$|\varepsilon_s| = |1+\varepsilon_{1p}||\varepsilon_{2s}|$ or $\varepsilon_s \simeq \varepsilon_{2s}$

3). f_1 in stopband $(f_{1s} \leq f_1 \leq 0.5)$
f_2 in stopband $(f_{2s} \leq f_2 \leq 0.5)$
$|\varepsilon_s| = |\varepsilon_{1s}||\varepsilon_{2s}|$ or $\varepsilon_s = \varepsilon_{1s}\varepsilon_{2s} \simeq 0$

The worst case therefore, occurs in regions one or two. For simplicity, let $\varepsilon_{1s} = \varepsilon_{2s}$ and so $\varepsilon_s = 0.03548 = \varepsilon_{1s} = \varepsilon_{2s}$. The 2-D filter can now be designed by using the well known 1-D filter design [11], for each dimension ω_1 and ω_2 with minor modifications. By trial and error, it was found that a filter length of $N_1 = N_2 = 28$ could satisfy the filter specification. The complete printout of impulse-response coefficients h(n), extremal frequencies and other data are shown in appendix 2.A for frequency axis ω_1, and appendix 2.B for frequency axis ω_2. As may be seen, values $\varepsilon_{1p} = 0.031904$ and $\varepsilon_{2p} = 0.033773$ were used, giving $\varepsilon_{1p} + \varepsilon_{2p} = 0.065677 < \varepsilon_p = 0.071519$ (specification); further, those for the stopband $\varepsilon_{1s} = 0.031904$ and $\varepsilon_{2s} = 0.033773$ lead to $\varepsilon_{1s}, \varepsilon_{2s} < \varepsilon_s = 0.03548$ (specification). Because this tabular information

is difficult to visualtize, the magnitude responses for each frequency axis are also shown in Figs. 2-10 for f_1 and in Fig. 2-11 for f_2, and very clearly display cutoff frequencies, ripple and extremal frequencies. From these latter two figures the 2-D Fig. 2-12 could be prepared using $|H|=|H_1H_2|$. Notice that the passband ripple is approximately equiripple, whereas the stopband is not.

2.4.2 Example 2

Design of a 2-D {bandpass,lowpass} filter with the following specifications:

passband cutoff frequencies: $f_{1p}^{(1)} = 0.15$, $f_{1p}^{(2)} = 0.35$, $f_{2p} = 0.10$

stopband cutoff frequencies: $f_{1s}^{(1)} = 0.1$, $f_{1s}^{(2)} = 0.4$, $f_{2s} = 0.15$.

For greater clarity, these cutoff frequencies are displayed on the 2-D magnitude response plot, ultimately to be obtained in Fig. 2-13. The maximum passband deviation is to be 0.66 db, and the stopband is to be at least 27 db down.

For this filter, a similar computation of worst case extrema in both passband and stopband, as in the previous example, is undertaken.

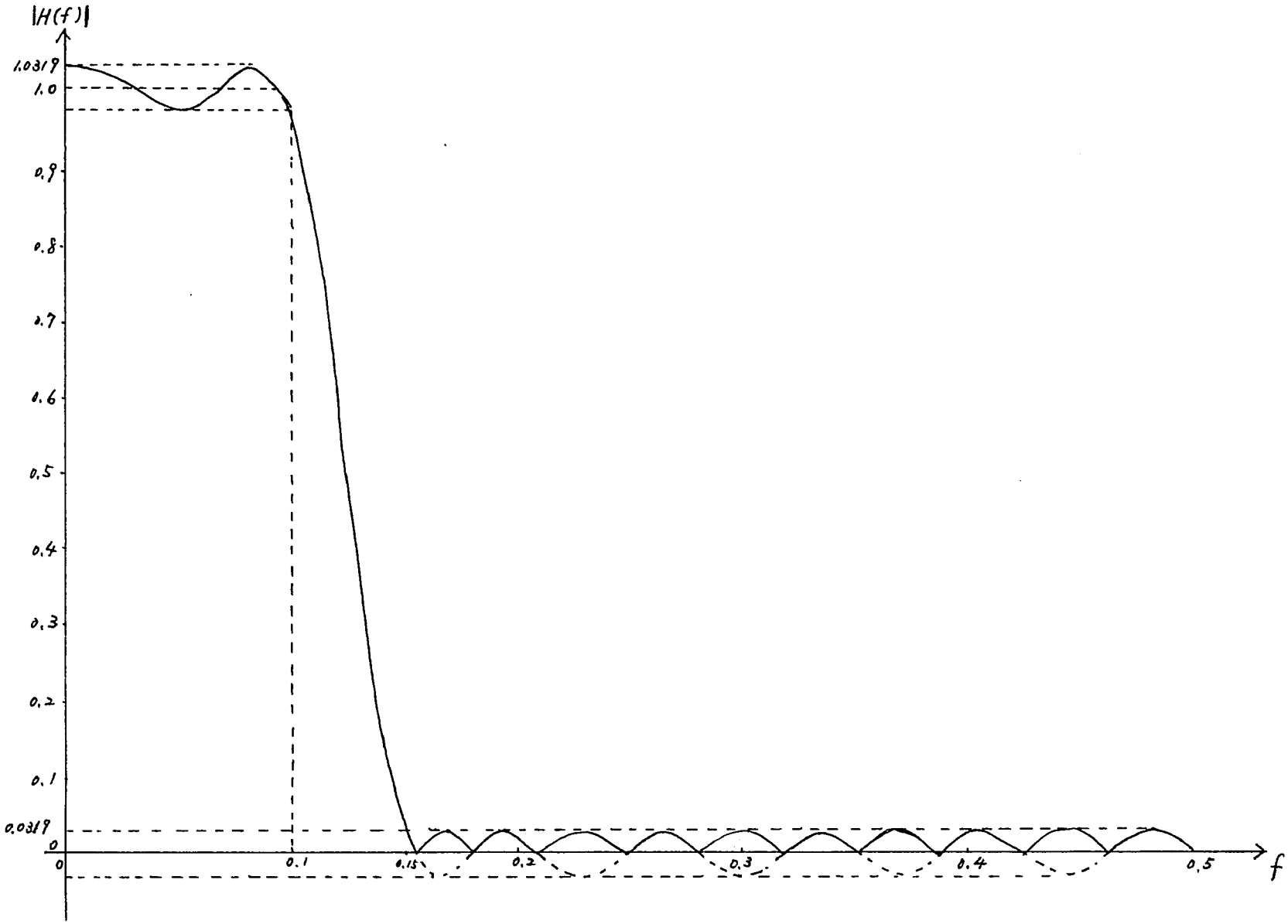


Fig. 2-10 A 27th order 1-D lowpass FIR filter characteristic corresponding to Appendix 2A

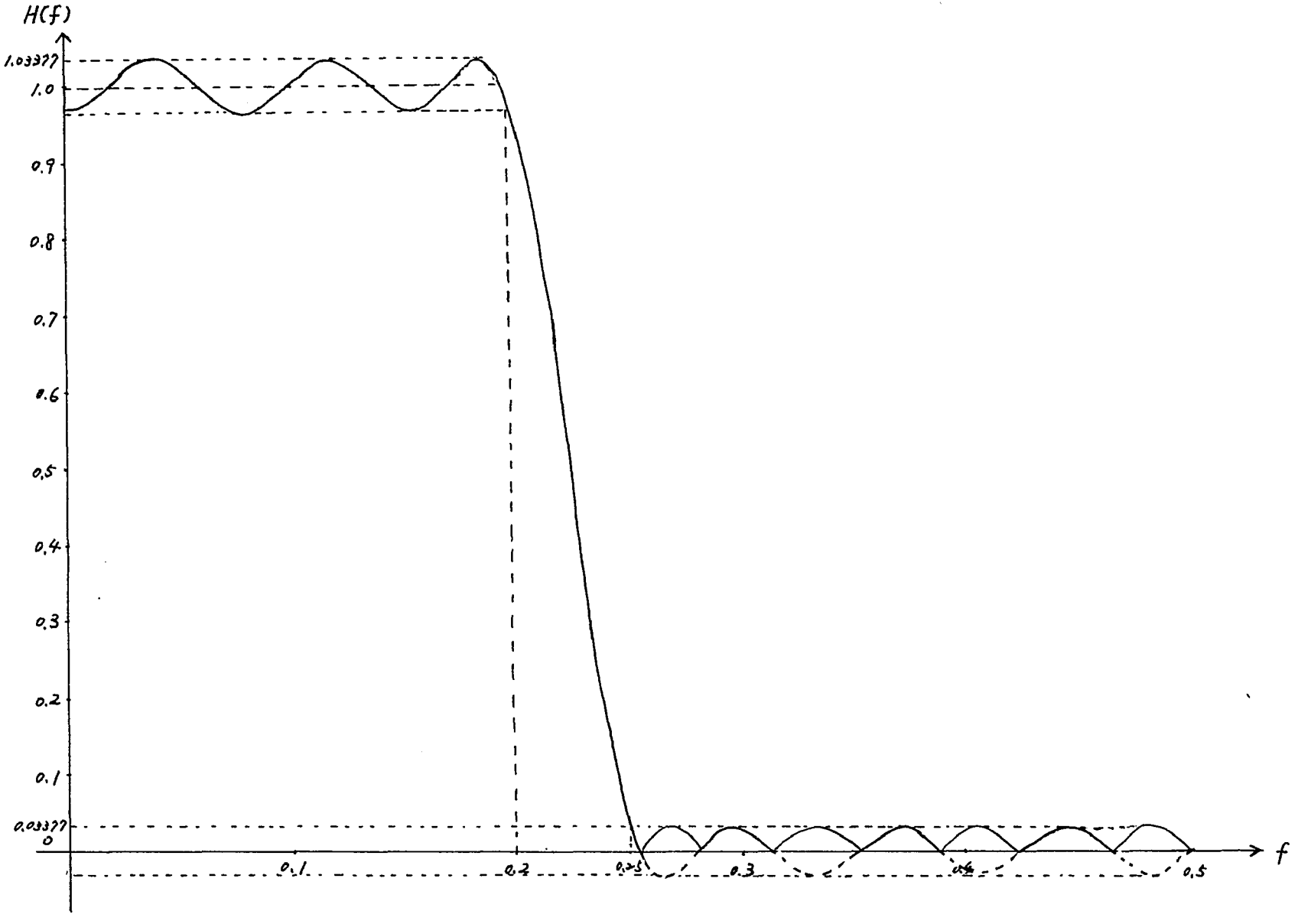


Fig. 2-11 A 27th order 1-D lowpass FIR filter characteristic corresponding to Appendix 2B

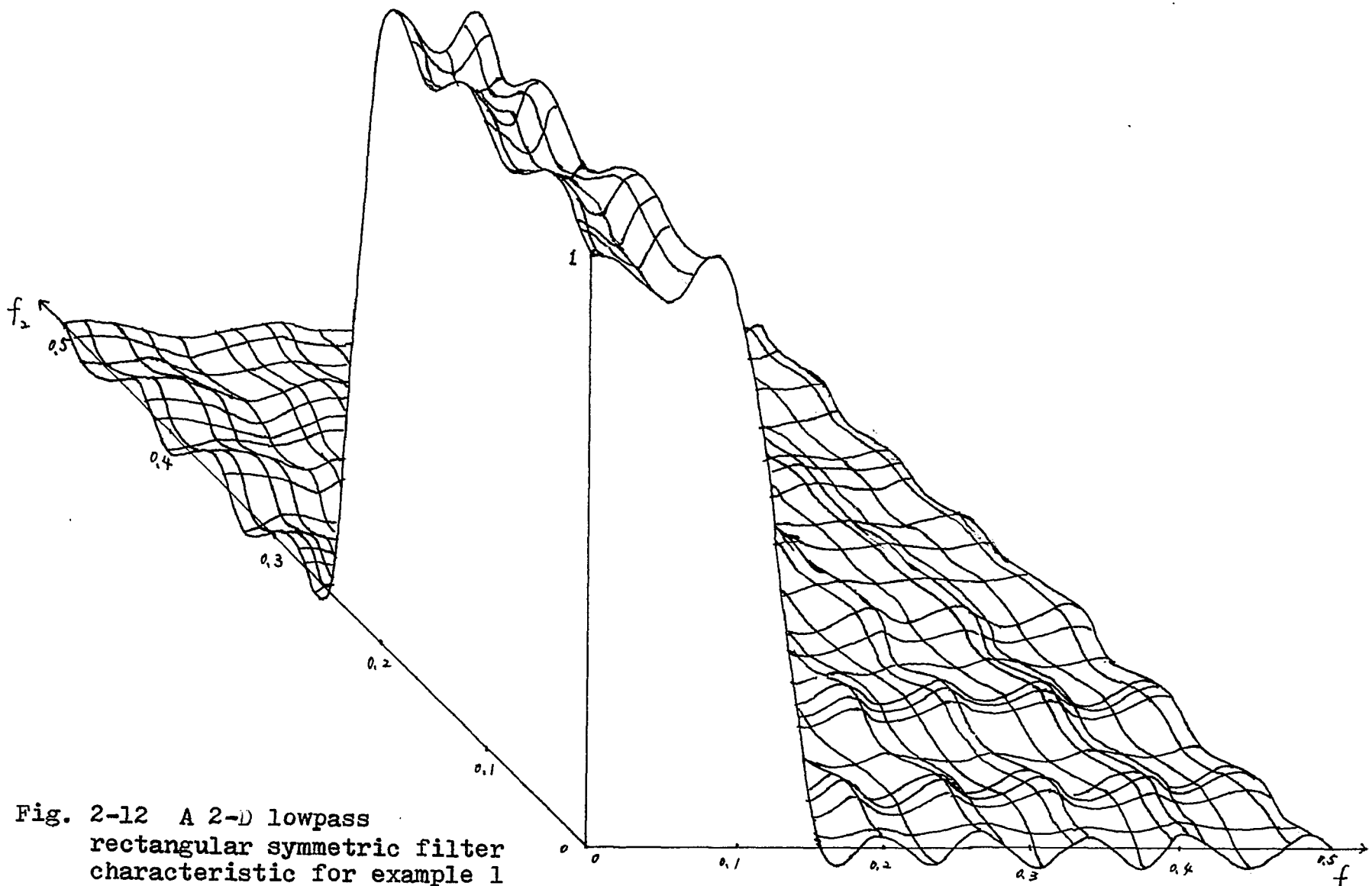


Fig. 2-12 A 2-D lowpass rectangular symmetric filter characteristic for example 1

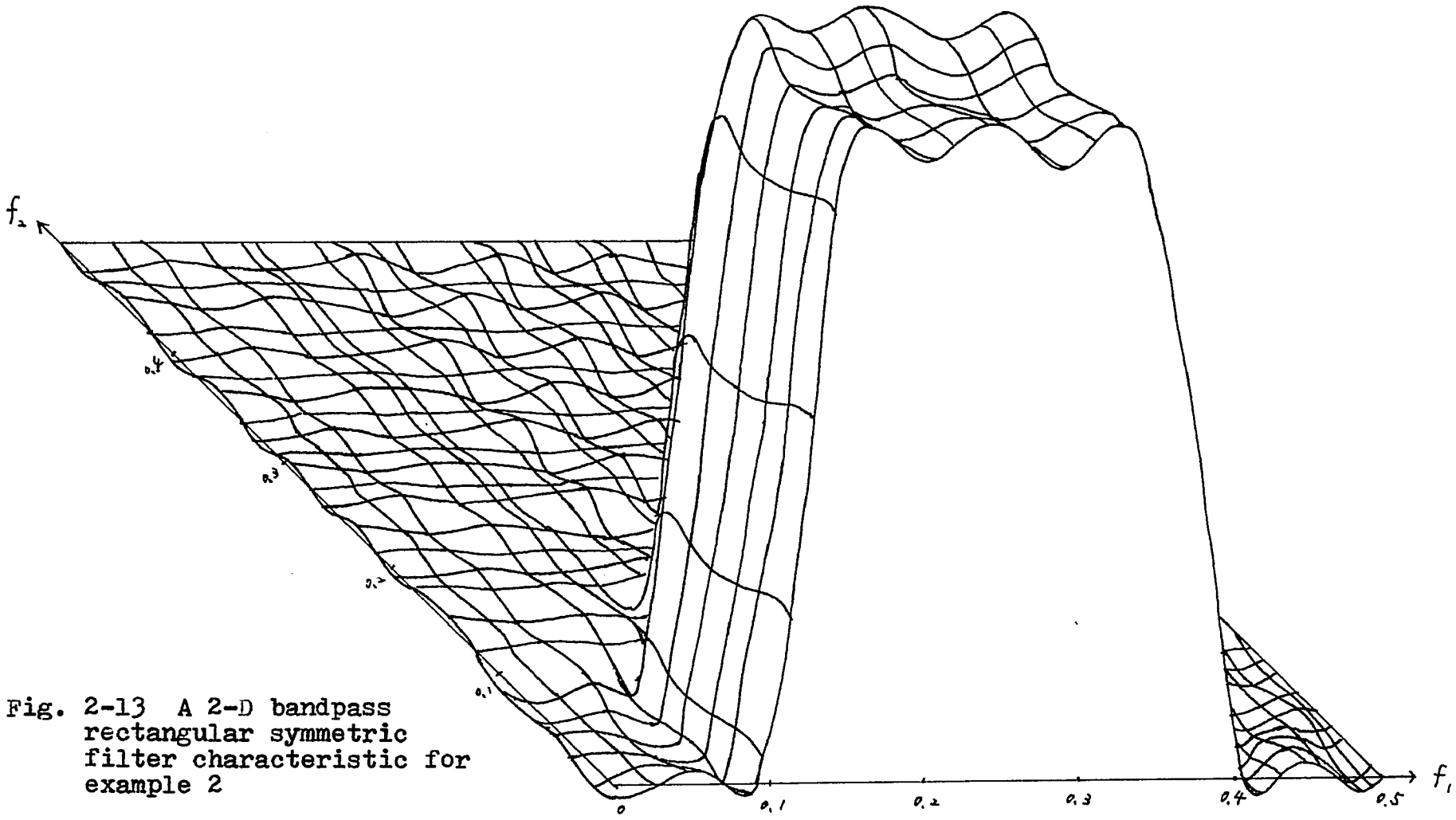


Fig. 2-13 A 2-D bandpass rectangular symmetric filter characteristic for example 2

For the passband, the deviation is exactly as in (2.17) except for the frequency limits

$$\varepsilon_p=\varepsilon_{1p}\varepsilon_{2p}+\varepsilon_{1p}+\varepsilon_{2p}\simeq\varepsilon_{1p}+\varepsilon_{2p},\ (f_{1p}^{(1)}\leq f_1\leq f_{1p}^{(2)};\ 0\leq f_2\leq f_{2p})$$

For the stopband, worst case ripple analysis, five regions are pertinent as indicated on Fig. 2-14 and listed below.

1). $\varepsilon_s = \varepsilon_{1s}(1+\varepsilon_{2p})$ for $0\leq f_1\leq f_{1s}^{(1)};\ 0\leq f_2\leq f_{2p}$

$\simeq \varepsilon_{1s}$

2). Same as above for $f_{1s}^{(2)}\leq f_1\leq 0.5;\ 0\leq f_2\leq f_{2p}$

3). $\varepsilon_s = \varepsilon_{2s}(1+\varepsilon_{1p})$ for $f_{1p}^{(1)}\leq f_1\leq f_{1p}^{(2)};\ f_{2s}\leq f_2\leq 0.5$

$\simeq \varepsilon_{2s}$

4). $\varepsilon_s = \varepsilon_{1s}\varepsilon_{2s}$ for $0\leq f_1\leq f_{1s}^{(1)};\ f_{2s}\leq f_2\leq 0.5$

5). Same as above for $f_{1s}^{(2)}\leq f_1\leq 0.5;\ f_{2s}\leq f_2\leq 0.5$

Now $20\text{Log}_{10}(1+\varepsilon_p) = 0.66$, therefore, $\varepsilon_p = 0.078947 \simeq \varepsilon_{1p}+\varepsilon_{2p}$

Assuming an ε_{2p} value as in the previous example, that is $\varepsilon_{2p} = 0.03576$, one then has $\varepsilon_{1p} = 0.04318$. For the stopband

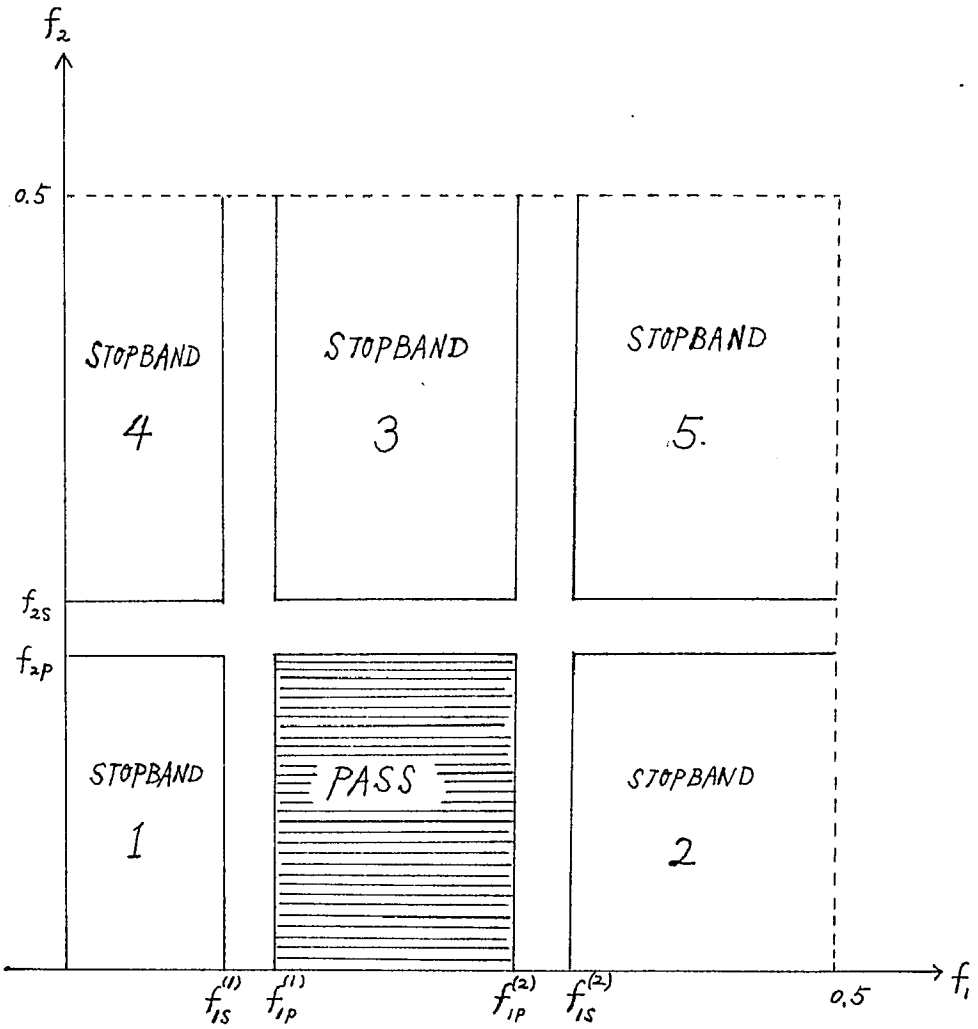


Fig. 2-14 Stopband-ripple analysis for example 2

deviation, $20\text{Log}_{10}\varepsilon_s = -27$ and so $\varepsilon_s = 0.044668$. For region 1, $\varepsilon_s \simeq \varepsilon_{1s}$ and for region 3, $\varepsilon_s \simeq \varepsilon_{2s}$ and these represent the worst cases. Therefore $\varepsilon_{1s} = \varepsilon_{2s} = 0.044668$ is taken, and a 1-D lowpass and a 1-D bandpass filter are designed. By trial and error, it was found that $N_1 = N_2 = 28$ would satisfy the requirement, and so the 1-D lowpass filter design is exactly as in example 1, appendix 2A and Fig. 2-10. The bandpass filter design follows the computer program given in [11], and the print-out is shown in appendix 2C. A magnitude-frequency plot is presented in Fig. 2-15. The magnitude plots of Figs. 2-10 and 2-15 were then combined to give the 2-D magnitude plot of Fig. 2-13 for the {bandpass,lowpass} filter.

As a result

$$\begin{aligned}\varepsilon_p &= \varepsilon_{1p} + \varepsilon_{2p} \\ &= 0.039797 + 0.031904 \\ &= 0.071701 < 0.0789467 \quad \text{(specification)}\end{aligned}$$

and

$$\begin{aligned}\varepsilon_s &= \max\{\varepsilon_{1s}, \varepsilon_{2s}\} \\ &= \max\{0.039797,\ 0.031904\} \\ &= 0.039797 < 0.044668 \quad \text{(specification)}\end{aligned}$$

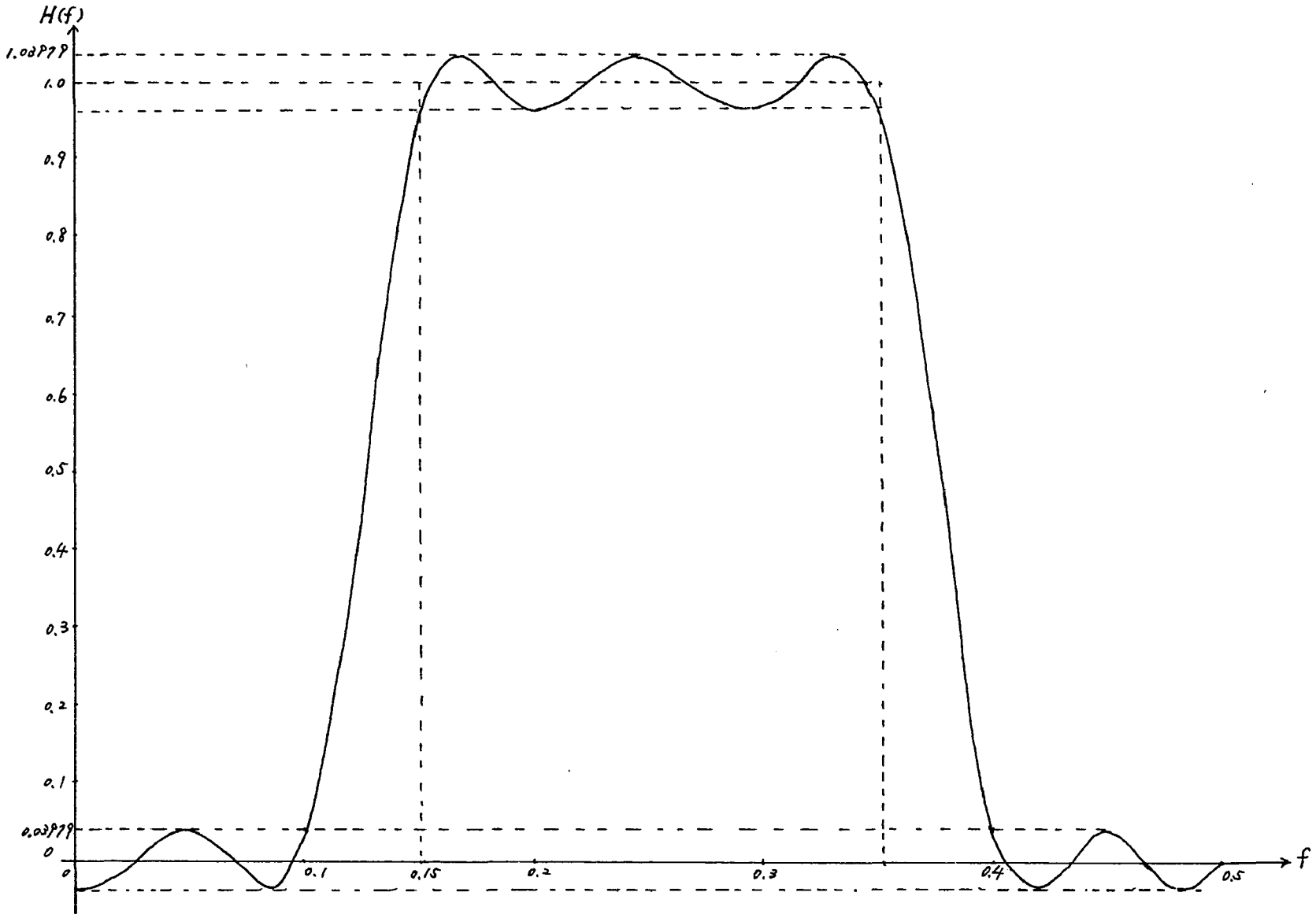


Fig. 2-15 A 27th order 1-D bandpass FIR filter characteristic corresponding to Appendix 2C

2.4.3 Example 3

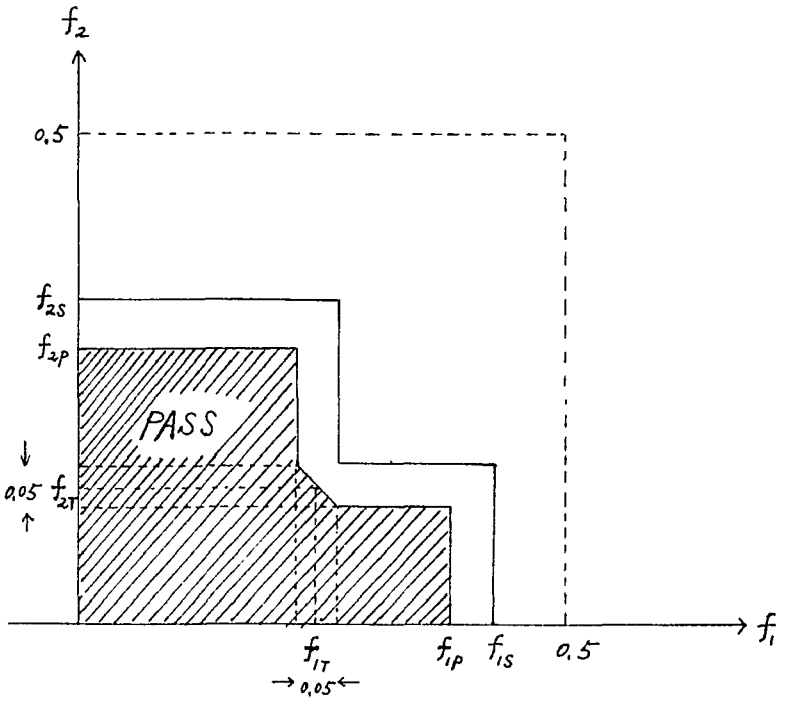
Design of a 2-D star-shaped rectangular filter.

An ideal star-shaped filter characteristic is shown in Fig. 2-6c. This type of filter can be used for shaping certain 2-D signals. A practical version of this filter including the transition region is shown in Fig. 2-16a. As demonstrated in Fig. 2-6c, this type of filter can be constructed from a 2-D lowpass filter and a 2-D bandpass filter. Care is taken that the transition bands of these two filters, at which $|H_1|=|H_2|= 0.5$ coincide at the points f_{1T} and f_{2T}, as the sum there will then be $|H|\simeq|H_1|+|H_2|=1$. To achieve this, the transition bandwidths of the two filters are chosen to be equal. The actual filter specification is given below and corresponds to Fig. 2-16a.

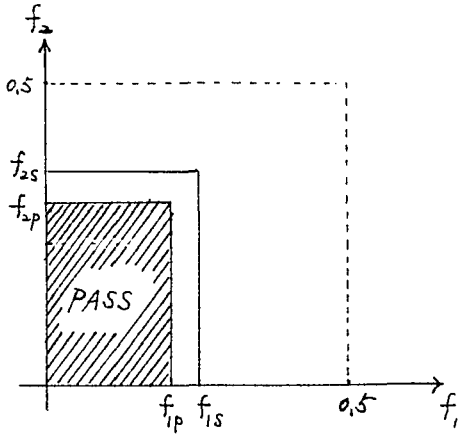
$f_{1p}=0.35, f_{2p}=0.2, f_{1s}=0.4, f_{2s}= 0.25, f1_T = 0.125, f_{2T}= 0.125.$

The maximum ripple in the passband is to be 1 db, and the stopband is to be at least 22 db down.

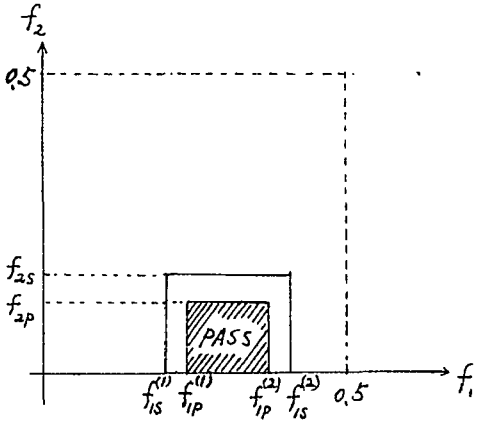
Refer to Fig. 2-16a. Measuring off equal distance from point f_{1T}, f_{2T}, one can create a transition band of width 0.05. It therefore appears that one can combine a lowpass filter (shown in Fig. 2-16b), together with a bandpass filter, shown in Fig. 2-16c, to yield the star-shaped filter.



(a)



(b)



(c)

Fig. 2-16 Stopband-ripple analysis for example 3

The {lowpass,lowpass} filter is to have the following parameters:

$f_{1p} = 0.1$, $f_{1s} = 0.15$; $f_{2p} = 0.2$, $f_{2s} = 0.25$

passband ripple ε_{Lp}, stopband ripple ε_{Ls}.

The {bandpass,lowpass} filter is to have the following parameters:

$f_{1p}^{(1)} = 0.15$, $f_{1p}^{(2)} = 0.35$; $f_{1s}^{(1)} = 0.1$, $f_{1s}^{(2)} = 0.4$; $f_{2p}=0.1$, $f_{2s} = 0.15$

passband ripple ε_{Bp}, stopband ripple ε_{Bs}

The computation of worst-case ripple parameters is similar to the calculations given in examples 1 and 2, and is presented below without further explanation:

In the passband

1). $\varepsilon_p=\varepsilon_{Lp}+\varepsilon_{Bs}$ for $0 \le f_1 \le f_{1T}$, $0 \le f_2 \le f_{2T}$

and $0 \le f_1 \le f_{1T}-0.025$, $f_{2T} \le f_2 \le f_{2p}$

2). $\varepsilon_p=\varepsilon_{Bp}+\varepsilon_{Ls}$ for $f_{1T} \le f_1 \le f_{1p}$, $0 \le f_2 \le f_{2T}-0.025$

In the stopband

$\varepsilon_s = \varepsilon_{Bs}+\varepsilon_{Ls}$ for $f_{1s} \le f_1 \le 0.5$, $0 \le f_2 \le f_{2T}+0.025$

and $f_{1p} \le f_1 \le 0.5$, $f_{2T}+0.025 \le f_2 \le 0.5$

and $0 \le f_1 \le f_{1T}+0.025$, $f_{2s} \le f_2 \le 0.5$

From the specification, $20\text{Log}_{10}(1+\varepsilon_p) = 1$, and so $\varepsilon_p = 0.122$ and this is the maximum value of the ripple, therefore

$$\varepsilon_{Lp}+\varepsilon_{Bs} \leq 0.122; \quad \varepsilon_{Bp}+\varepsilon_{Ls} \leq 0.122$$

From the stopband specification, $20\text{Log}_{10}\varepsilon_s = -22$, and so $\varepsilon_s = 0.07943$, therefore

$$\varepsilon_{Bs} + \varepsilon_{Ls} \leq 0.07943$$

Any combination of {lowpass,lowpass} filter and {bandpass,lowpass} filter satisfying these requirements will satisfactorily meet the passband and stopband specifications. Recollect that in example 1, the values $\varepsilon_{1p}+\varepsilon_{2p} = \varepsilon_p = 0.065677$ and $\max(\varepsilon_{1s},\varepsilon_{2s}) = 0.03373 = \varepsilon_s$ were used in the sense of ε_{Lp} and ε_{Ls}, corresponding to the {lowpass,lowpass} filter. Further in example 2, the values $\varepsilon_p=\varepsilon_{1p}+\varepsilon_{2p} = 0.071701$ and $\varepsilon_s = \max(\varepsilon_{1s}, \varepsilon_{2s}) = 0.039787$ were used in the sense of ε_{Bp} and ε_{Ls}, corresponding to the {bandpass,lowpass} filter. As a result

$$\varepsilon_p = \max(\varepsilon_{Lp}+\varepsilon_{Bs}, \varepsilon_{Bp}+\varepsilon_{Ls})$$

$$= \max(0.065677+0.039796 = 0.105473;$$

$$0.071701+0.033730 = 0.105431)$$

$$= 0.105473<0.1220 \text{ (specification)}$$

Similarly

$$\varepsilon_s = \varepsilon_{Bs} + \varepsilon_{Ls} = 0.039796 + 0.03373$$
$$= 0.073526$$
$$< 0.07943 \text{ (specification)}$$

Hence the {lowpass,lowpass} filter of example 1 and the {bandpass,lowpass} filter of example 2 satisfy the requirements of example 3, and their designs can be used. Therefore, Fig. 2-12 and Fig. 2-13 may be added to yield the required Fig. 2-17, which represents the star shaped filter characteristic.

Notice that keeping N = 28 in all filter designs ensures that the phase angles of all filter are identical and the series-parallel structure of Fig. 2-4b can be used to create the star-shaped filter. Note also that the characteristic near f_{1T} and f_{2T} is difficult to control, and further research may be needed here.

In conclusion, this chapter showed that simple 2-D rectangular filters may be constructed from low degree separable 1-D filters. These can be implemented by means of simple structures such as the tandem and series-parallel combination. Where equal-phase requirements exist, linear-phase FIR filter must be used.

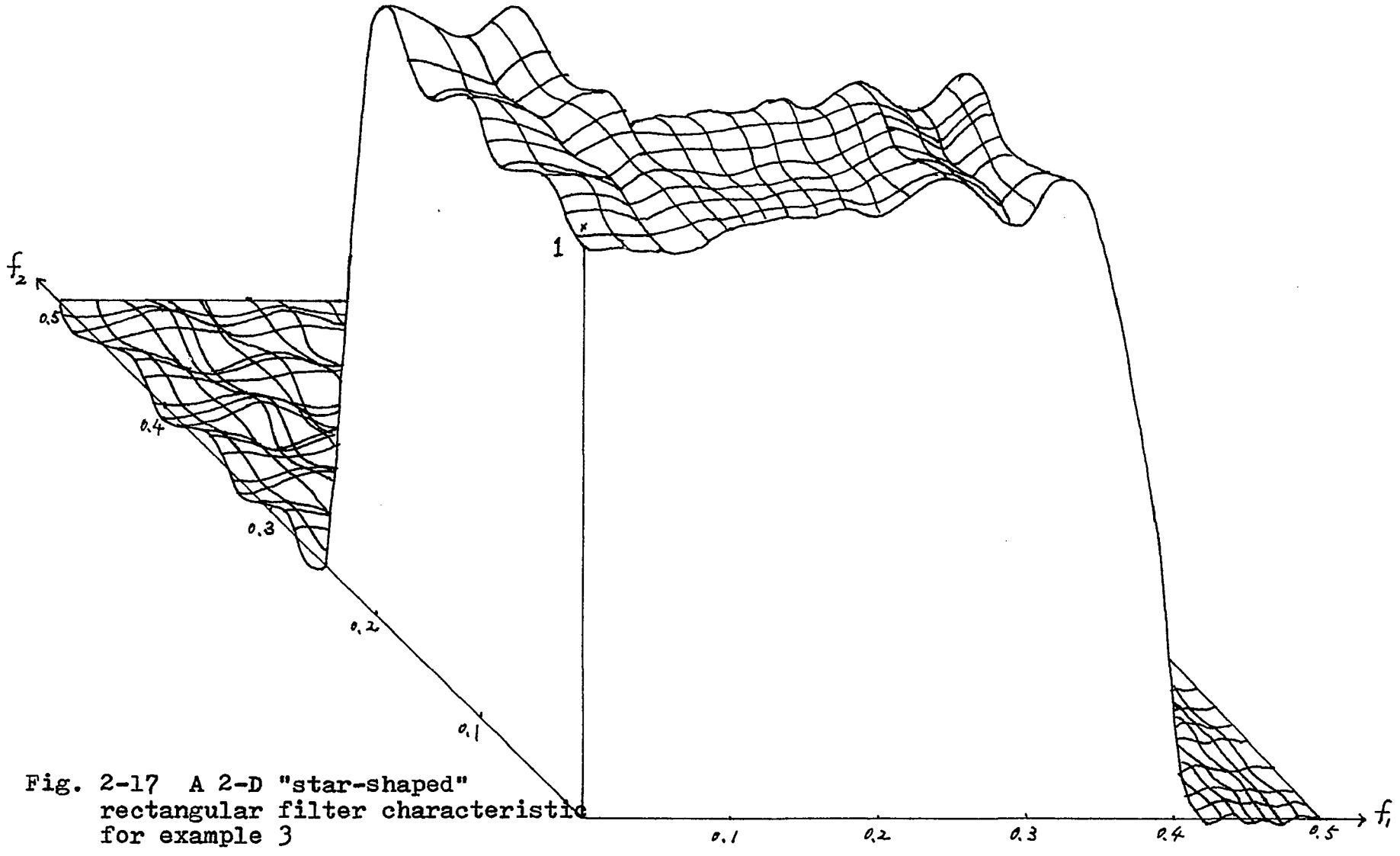


Fig. 2-17 A 2-D "star-shaped" rectangular filter characteristic for example 3

APPENDIX 2-A

FILTER LENGTH = 28

********** IMPULSE RESPONSE **********

```
H( 1) = -0.18281320E-01 = H(28)
H( 2) = -0.67690010E-03 = H(27)
H( 3) =  0.96972950E-02 = H(26)
H( 4) =  0.20911820E-01 = H(25)
H( 5) =  0.24112620E-01 = H(24)
H( 6) =  0.12849200E-01 = H(23)
H( 7) = -0.11495500E-01 = H(22)
H( 8) = -0.37400850E-01 = H(21)
H( 9) = -0.47009900E-01 = H(20)
H(10) = -0.24690520E-01 = H(19)
H(11) =  0.33526100E-01 = H(18)
H(12) =  0.11532330E 00 = H(17)
H(13) =  0.19499360E 00 = H(16)
H(14) =  0.24409300E 00 = H(15)
```

	BAND 1	BAND 2
LOWER BAND EDGE	0.000000000	0.149999900
UPPER BAND EDGE	0.100000000	0.500000000
DESIRED VALUE	1.000000000	0.000000000
WEIGHTING	1.000000000	1.000000000
DEVIATION	0.031904330	0.031904330
DEVIATION IN DB	-29.922980000	-29.922980000

EXTREMAL FREQUENCIES

0.0000000	0.0535714	0.0848213	0.1000000	0.1500000
0.1633928	0.1924105	0.2258925	0.2616067	0.2995530
0.3352671	0.3709813	0.4089276	0.4446417	0.4825880

APPENDIX 2-B

FILTER LENGTH = 28

********** IMPULSE RESPONSE **********

```
H( 1) =  0.44931760E-02 = H(28)
H( 2) = -0.23540370E-01 = H(27)
H( 3) = -0.10537370E-01 = H(26)
H( 4) =  0.14434440E-01 = H(25)
H( 5) =  0.17852350E-01 = H(24)
H( 6) = -0.14709040E-01 = H(23)
H( 7) = -0.31608810E-01 = H(22)
H( 8) =  0.96595510E-02 = H(21)
H( 9) =  0.51468530E-01 = H(20)
H(10) =  0.52189970E-02 = H(19)
H(11) = -0.84444280E-01 = H(18)
H(12) = -0.47670720E-01 = H(17)
H(13) =  0.17937830E 00 = H(16)
H(14) =  0.41311840E 00 = H(15)
```

	BAND 1	BAND 2
LOWER BAND EDGE	0.000000000	0.250000000
UPPER BAND EDGE	0.199999900	0.500000000
DESIRED VALUE	1.000000000	0.000000000
WEIGHTING	1.000000000	1.000000000
DEVIATION	0.033773210	0.033773210
DEVIATION IN DB	-29.428520000	-29.428520000

EXTREMAL FREQUENCIES

0.0000000	0.0401786	0.0803570	0.1183033	0.1540174
0.1852673	0.2000000	0.2500000	0.2633928	0.2946427
0.3303568	0.3683031	0.4040173	0.4419636	0.4799098

APPENDIX 2-C

FILTER LENGTH = 28

********** IMPULSE RESPONSE **********

```
H( 1) =  0.11325150E-01 = H(28)
H( 2) = -0.70066670E-02 = H(27)
H( 3) =  0.99606960E-02 = H(26)
H( 4) = -0.34321930E-01 = H(25)
H( 5) = -0.34926980E-01 = H(24)
H( 6) =  0.16841540E-01 = H(23)
H( 7) = -0.19292510E-01 = H(22)
H( 8) =  0.51589620E-01 = H(21)
H( 9) =  0.68021410E-01 = H(20)
H(10) = -0.35309790E-01 = H(19)
H(11) =  0.46379230E-01 = H(18)
H(12) = -0.16174810E 00 = H(17)
H(13) = -0.27561330E 00 = H(16)
H(14) =  0.34420370E 00 = H(15)
```

	BAND 1	BAND 2	BAND 3
LOWER BAND EDGE	0.000000000	0.149999900	0.399999900
UPPER BAND EDGE	0.100000000	0.350000000	0.500000000
DESIRED VALUE	0.000000000	1.000000000	0.000000000
WEIGHTING	1.000000000	1.000000000	1.000000000
DEVIATION	0.039796960	0.039796960	0.039796960
DEVIATION IN DB	-28.002970000	-28.002970000	-28.002970000

EXTREMAL FREQUENCIES

0.0000000	0.0468750	0.0825891	0.1000000	0.1500000
0.1656249	0.1991069	0.2415175	0.2906244	0.3330350
0.3500000	0.4000000	0.4156249	0.4468748	0.4825889

CHAPTER THREE

Hardware Realization of Low-Order Two-Dimensional Digital Filters.

3.1 Introduction

It was pointed out in Chapter One that a 2-D digital filter can be realized either by software methods or hardware methods. The hardware realization method, available at this time (1979) processes the 2-D data point-by-point. In this chapter, a new parallel-processing hardware implementation of a 2-D filter is proposed. A block diagram of a 2-D system which employs the proposed parallel-processing filter is shown in Figure 3-1. As can be seen in this figure, the 2-D data may be first sensed by a scanner, and then multiplexed and forms the input to the 2-D filter. After filtering, the output data, which is generated column-by-column, is demultiplexed for display or other purposes. The 2-D filter is implemented by means of unit delays, adders and multipliers. This is discussed in more detail below.

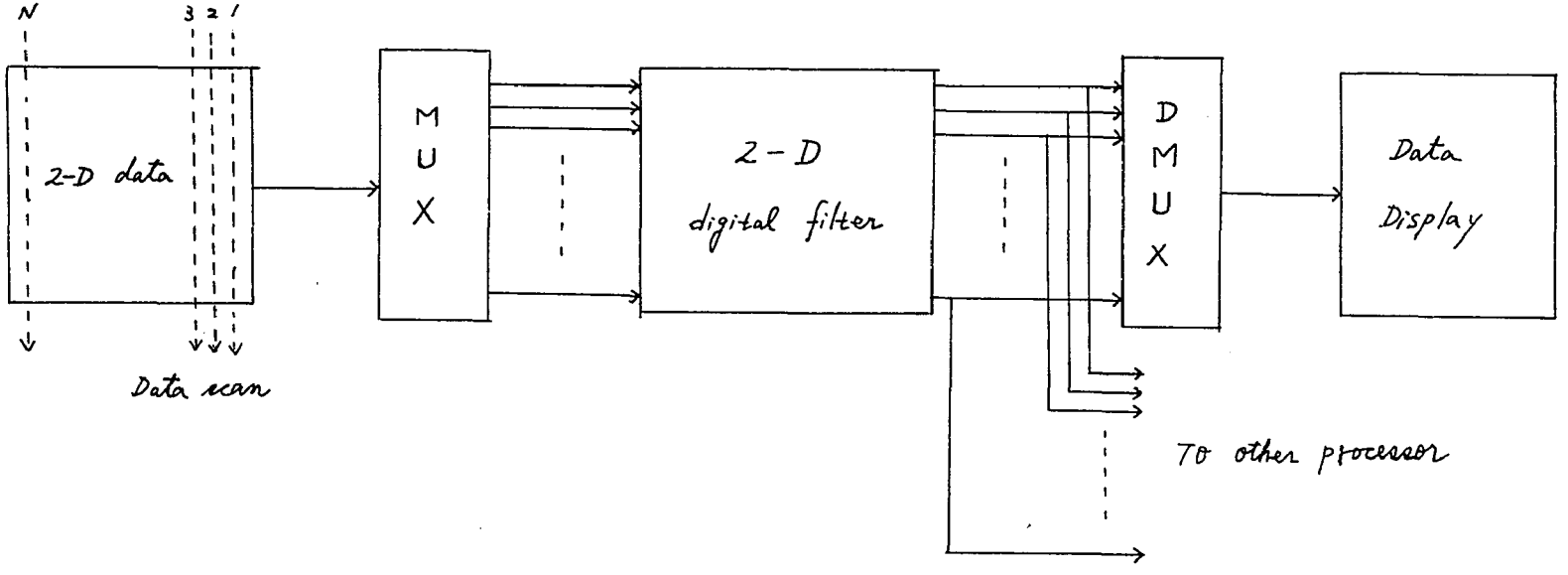


Fig. 3-1 Block diagram of a parallel processing 2-D system

3.2 Principles of Hardware Realization of 2-D FIR Filters

A two-dimensional array of input signals $[x(m,n);\ 0\leq m\leq M,\ 0\leq n\leq N]$ may be processed either by a 2-D FIR filter or IIR filter. The FIR filter case will be treated first.

Let the 2-D FIR filter be represented by the impulse response array $[h(n_1,n_2);\ 0\leq n_1\leq K,\ 0\leq n_2\leq L]$. The filter is now said to be of order $(K+1)(L+1)$. In many cases it is convenient to have K=L, although this is not a necessary condition. The output point $y(n_1,n_2)$ is then given by the convolution sum

$$y(n_1,n_2) = \sum_{m=0}^{M}\sum_{n=0}^{N} x(m,n)\ h(n_1-m,n_2-n) \qquad (0\leq n_1\leq K+M+1;\ 0\leq n_2\leq L+N+1) \qquad (3.1a)$$

An alternative formulation of this convolution sum is

$$y(n_1,n_2) = \sum_{m=0}^{K}\sum_{n=0}^{L} h(m,n)\ x(n_1-m,n_2-n) \qquad (0\leq n_1\leq K+M+1;\ 0\leq n_2\leq L+N+1) \qquad (3.1b)$$

This latter formulation is most useful for a specified low-order filter.

With respect to the input signal x(m,n), it is convenient to consider the second variable n as a time variable, whereas the first variable m is a spatial variable. As a result, x(m,n-1) may be obtained from x(m,n) through a unit time delay element as shown in Fig. 3-2. On the other hand, the variable

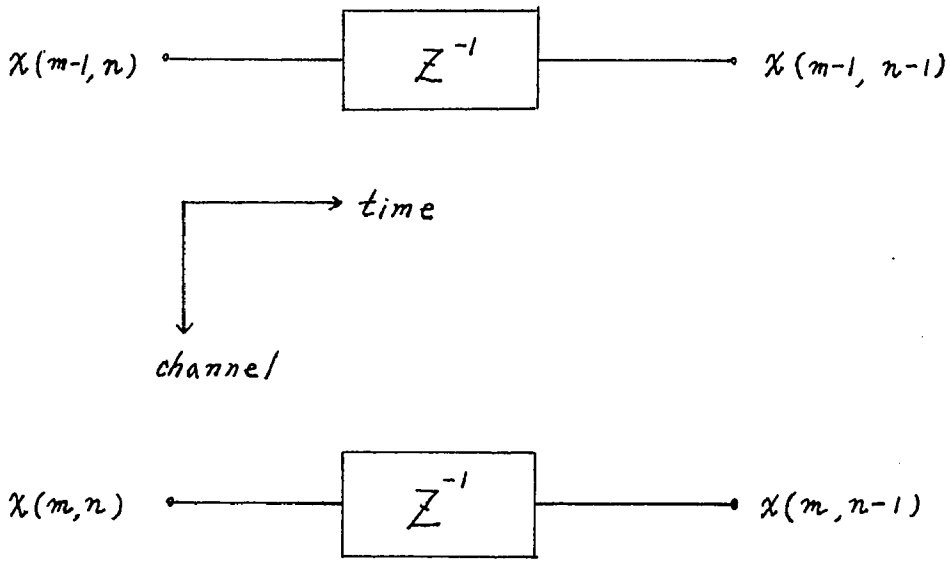


Fig. 3-2 Block diagram of 2-D delay units

m refers to the spatial channel, and so x(m-1,n) is related to x(m,n) by shifting one channel as is shown in Fig. 3-2. Similar remarks apply to the output signal $y(n_1,n_2)$ where n_2 is considered as a time variable and n_1 is treated as a spatial variable.

For convenience, it is useful to put equations (3.1a) and (3.1b) into the form of sum of vector products. One may therefore define

$$\underset{\sim}{X}^T_{(m,n;i)} = [x(m,i), x(m-1,i), \ldots x(n,i)] \qquad (m>n) \tag{3.2a}$$

and

$$\underset{\sim}{H}^T_{(k,\ell;i)} = [h(k,i), h(k-1,i), \ldots h(\ell,i)] \quad (k>\ell) \tag{3.2b}$$

Therefore (3.1a) may be written

$$y(n_1,n_2) = \sum_{i=0}^{N} \underset{\sim}{H}^T_{(n_1,n_1-M;n_2-i)} \underset{\sim}{X}_i \tag{3.3a}$$

where $\underset{\sim}{X}_i$ is the i-th column of $\underset{\sim}{X} = [x(m,n)]$. Similarly

$$y(n_1,n_2) = \sum_{i=0}^{L} \underset{\sim}{X}^T_{(n_1,n_1-K;n_2-i)} \underset{\sim}{H}_i \tag{3.3b}$$

where $\underset{\sim}{H}_i$ is the i-th column of $\underset{\sim}{H} = [h(u,v)]$. Equation (3.3a) can be directly associated with a filter hardware structure as shown in Fig. 3-3. Here the input vector $\underset{\sim}{X}_N$ moves progressively

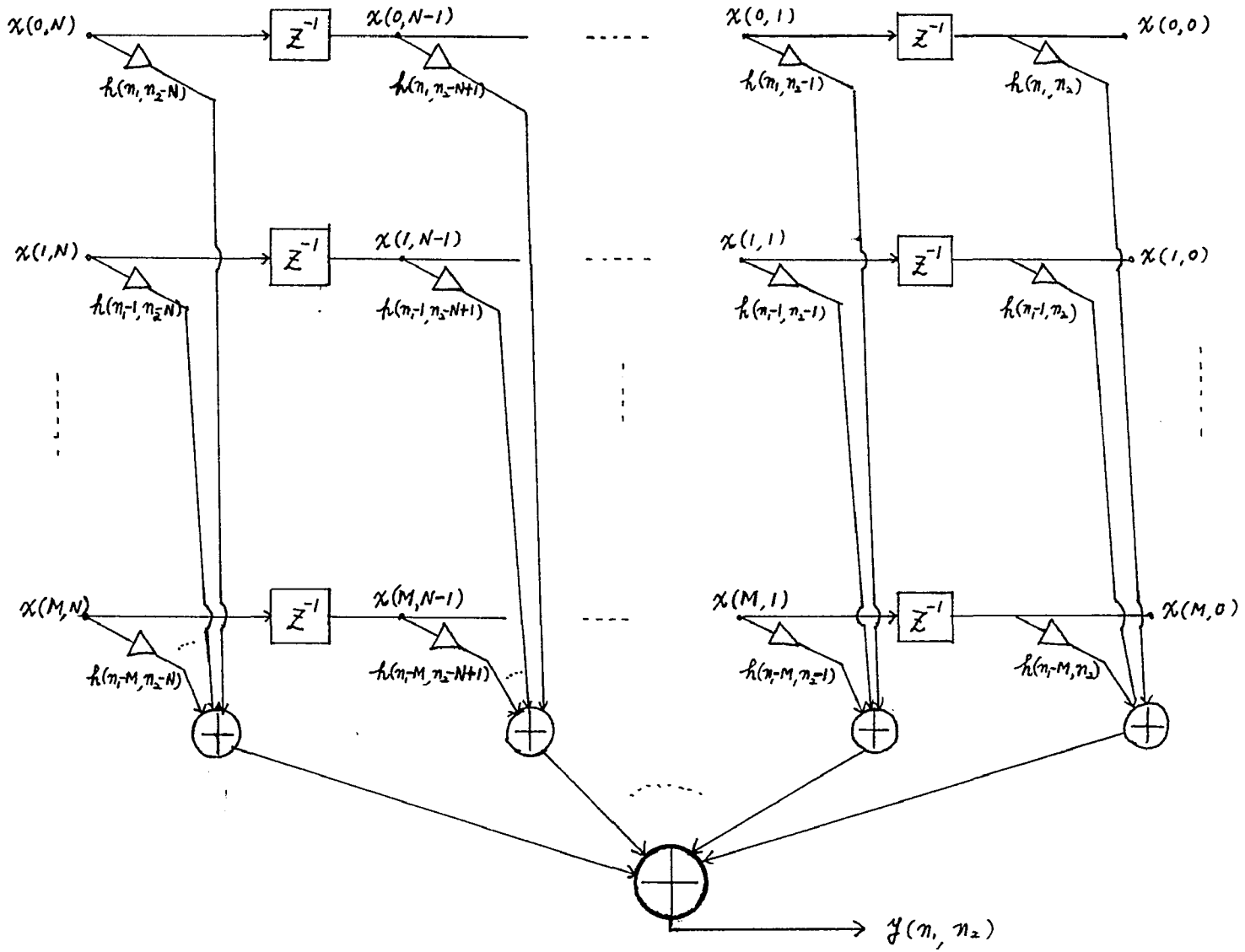


Fig. 3-3 Hardware realization of a vector from representation for a 2-D FIR filter

to the right, thereby experiencing unit delays and becoming $\underset{\sim}{X}_{N-1}$, $\underset{\sim}{X}_{N-2}$, ..., $\underset{\sim}{X}_{0}$. The impulse response vector $\underset{\sim}{H}^{T}(n_1,n_1-M;n_2-i)$ has entries $h(n_1,n_2-i),h(n_1-1,n_2-i)$, ... $,h(n_1-M,n_2-i)$, which are the values of multiplier coefficients between channels of the hardware structure. Summation of the terms of (3.3a) is represented in the hardware structure by the presence of one or more summing devices.

As an example, consider a 2x2 array FIR filter, represented by

$$y(n_1,n_2) = \sum_{m=0}^{1} \sum_{n=0}^{1} h(m,n)\ x(n_1-m,n_2-n) \tag{3.4}$$

Let the input array $\underset{\sim}{X}$ be (M+1)(N+1). In this case, (3.4) becomes

$$y(n_1,n_2) = \sum_{n=0}^{1} \underset{\sim}{H}_n^{T}\ \underset{\sim}{X}(n_1,n_1-1;n_2-n) \tag{3.5}$$

and corresponds directly with the hardware structure of Fig. 3-4. Since only two column vectors of $\underset{\sim}{X}$ are involved, there is only one delay element present in each channel. The same four multiplier coefficients $h(0,0),h(0,1),h(1,0)$ and $h(1,1)$ are present in each of the (M+1) channels. The summation implied by (3.5) requires one summing element in each channel.

As a second example, consider a 3x3 FIR filter specified by

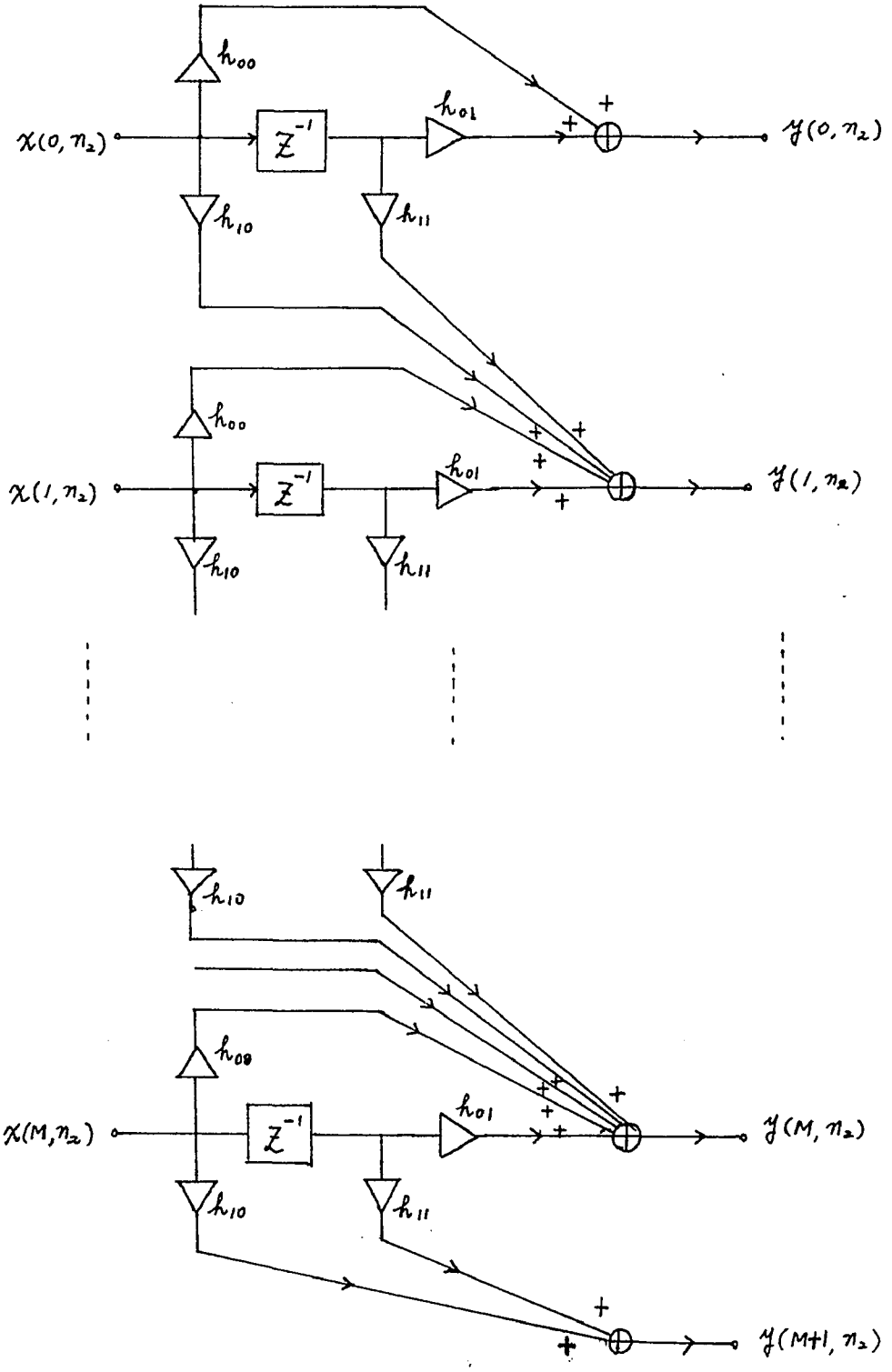


Fig. 3-4 Hardware realization of a general first order 2-D FIR filter

$$y(n_1,n_2) = \sum_{n=0}^{2} \underset{\sim}{H}_n^T \underset{\sim}{X}_{(n_1,n_1-2;n_2-n)} \tag{3.6}$$

Assuming an input array of order (M+1)·(N+1), and following a similar interpretation as given in the previous example, yeilds the filter hardware structure of Fig. 3-5. Note that there are two delay units, nine multipliers and one summing element present in each channel. Moreover, there are two more output channels than input channels. This interpretation may be extended to higher-order FIR filters. For instance, if the filter impulse response coefficients, h(m,n), are of array size (K+1)·(L+1), then L delay elements are needed in each channel. If the input array is of size (M+1)·(N+1), then there will be (M+1) input channels and (M+1)+(K+1)-1=M+K+1 output channels, each of which contains one summing device. The total number of multipliers required in each input channel is (K+1)·(L+1).

3.3 Principles of Hardware Realization of 2-D IIR Filters

Two-dimensional IIR filters perform similar functions to 2-D FIR filters. One advantage of IIR filters over FIR filters is that the same performance specification can generally be satisfied with a lower order IIR filter, thereby leading to a saving in components. On the other hand, an IIR filter has a stability problem, which must be satisfied.

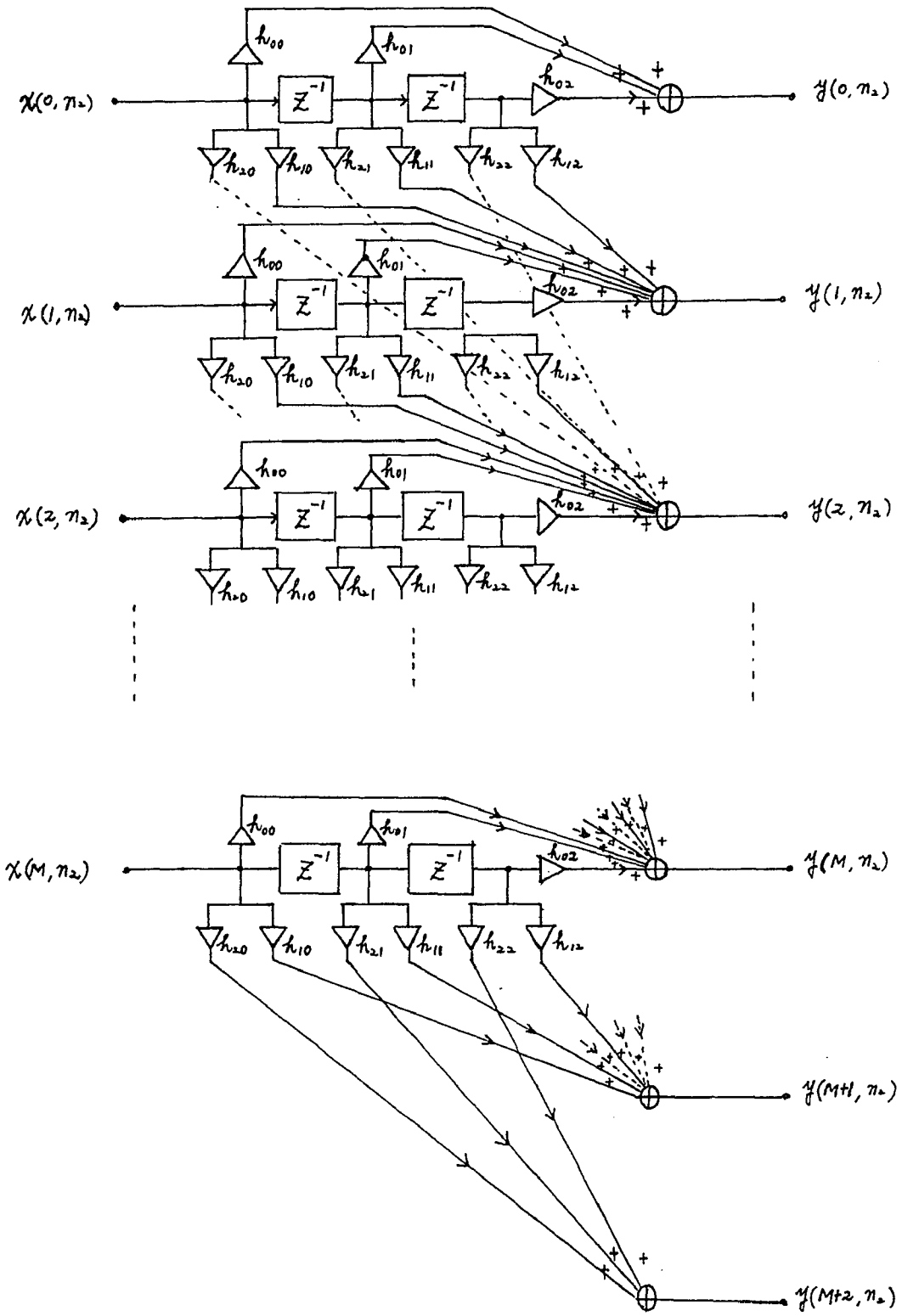


Fig. 3-5 Hardware realization of a general second order 2-D FIR filter

Furthermore, the phase characteristic of such filters is inherently nonlinear with respect to the ω_1 and ω_2 variables. This is a disadvantage when linear phase is required. The input-output relationship of such filters was given in (1.4), and is repeated here in modified form for convenience:

$$y(n_1,n_2) = \sum_{i=0}^{p}\sum_{j=0}^{q} a(i,j)x(n_1-i,n_2-j) - \sum_{\substack{i=0 \\ i\neq j=0}}^{p}\sum_{j=0}^{q} b(i,j)y(n_1-i,n_2-j) \qquad (0\leq n_1,n_2\leq\infty) \tag{3.7}$$

Note that in theory the number of output samples $y(n_1,n_2)$ is infinite due to the recursive relation. In practice, one processes with data having finite (B bits) word length, and so when the output sample value is less than 2^{-B}, one truncates. That is, if

$$|y(n_1>K,n_2>L)|<2^{-B} \quad \text{then } y(n_1,n_2) = 0 \tag{3.8}$$

Equation (3.7) may be put into the form of vector sums as was done previously in the case of FIR filters. This yields:

$$y(n_1,n_2) = \sum_{j=0}^{q} \{\underline{A}_j^T \underline{X}(n_1,n_1-p;n_2-j) - \underline{B}_j^T \underline{Y}(n_1,n_1-p;n_2-j)\} + b_{00}y(n_1,n_2) \tag{3.9}$$

Note that the b_{00} term on the right hand side of the equation cancels out. Note further, that the summation term, which includes the vector $\underline{A}_j^T$, is identical to the term of the FIR structure discussed in the previous section. The summation term which includes vector $\underline{B}_j^T$ is similar, but now represents a feedback term from the output variables y. The actual filter structure for p = q = 1 displays this indicated symmetry and is shown in Fig. 3-6. This figure has been drawn on the further assumption that $|y(n_1>M+K,n_2)|<2^{-B}$, and can be truncated as explained in (3.8) Although there are only M+1 input channels, the output channels become M+K+1 in number, due to the convolution inherent in filtering. Each individual channel has two unit delays, seven multipliers and one summing device.

A slightly more complicated version of the above filter, taking p = q = 2 in (3.7), is shown in Fig. 3-7. Similar remarks as discussed in the case of p = q = 1 apply to this filter. The individual channel now has four unit delays, seventeen multipliers and one summing device. In general, an IIR filter with p = M, q = N, will need (M+N) unit delays, 2 x(M+1) x(N+1)-1 multipliers (which correspond to the number of filter coefficients) and one summing device.

A filter structure which reduces the number of unit delay elements to one half will now be discussed.

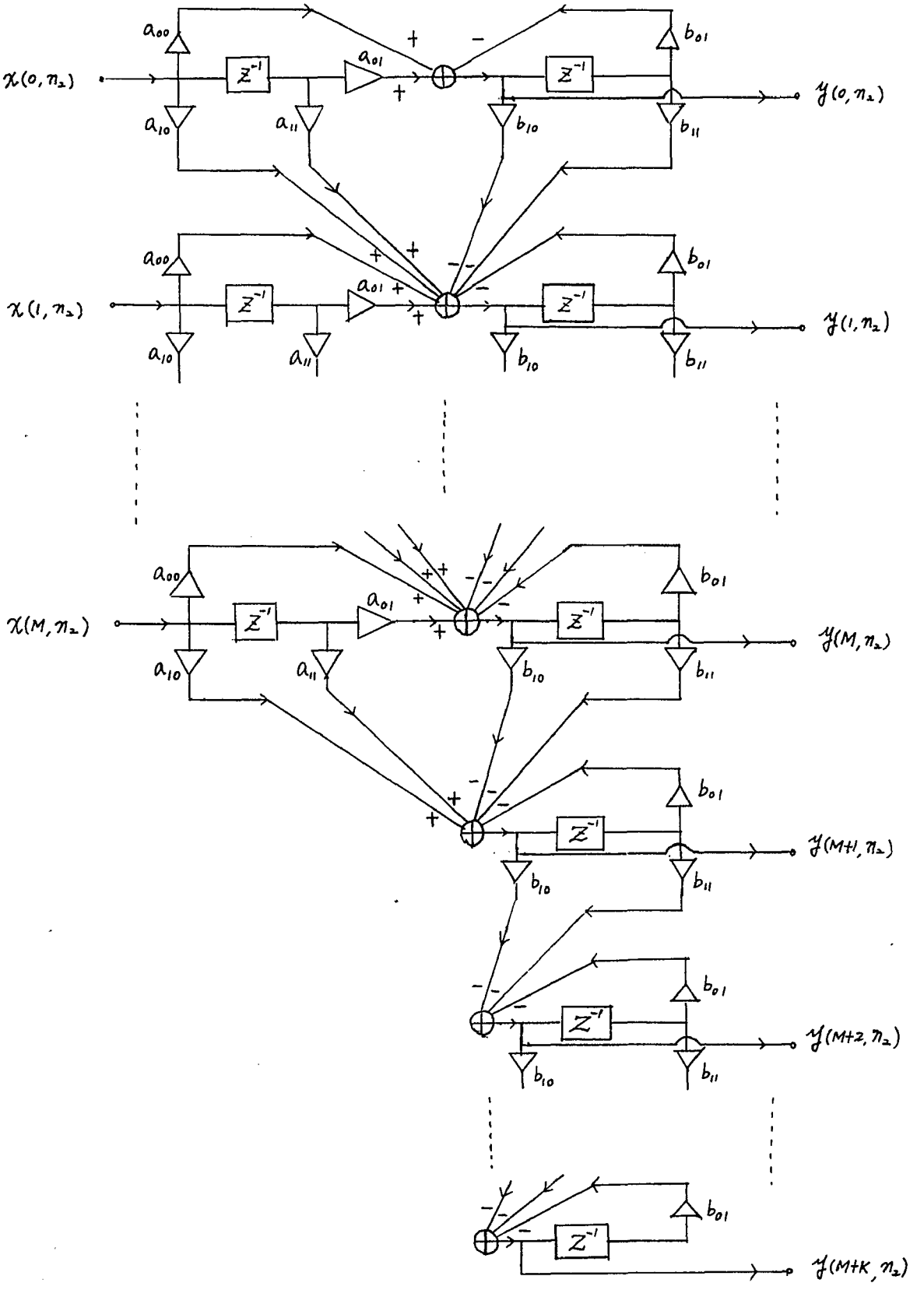


Fig. 3-6 Hardware realization of a general first order 2-D IIR filter (Direct form 1)

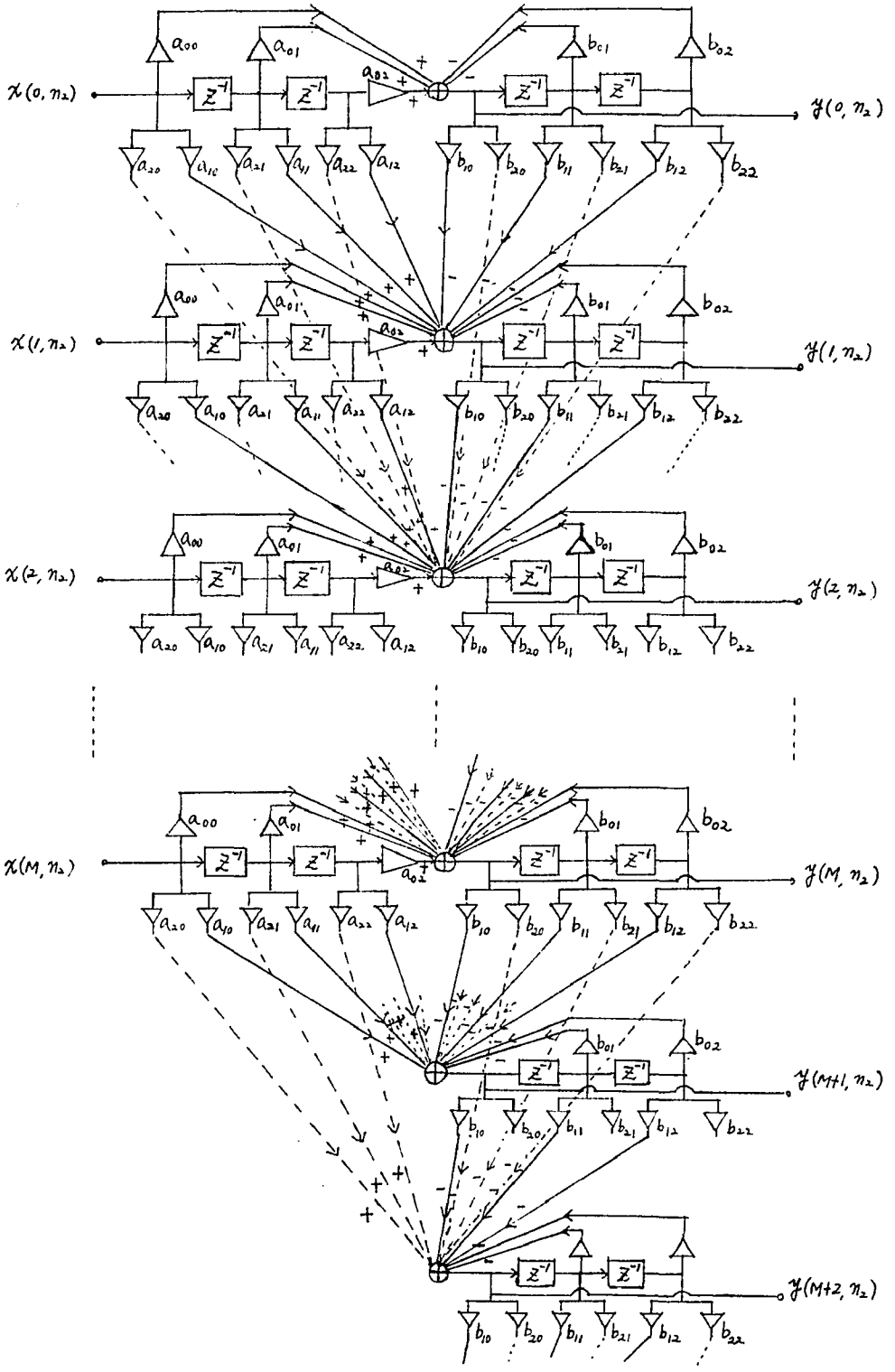


Fig. 3-7 Hardware realization of a general second order 2-D IIR filter (Direct form 1)

Consider the system function of a 2-D IIR filter, which was mentioned in Chapter One and is repeated here in modified form:

$$H(z_1,z_2) = \frac{Y(z_1,z_2)}{X(z_1,z_2)} = \frac{A(z_1,z_2)}{B(z_1,z_2)} = \frac{\sum_{i=0}^{p}\sum_{j=0}^{q} a(i,j) z_1^{-i} z_2^{-j}}{\sum_{i=0}^{p}\sum_{j=0}^{q} b(i,j) z_1^{-i} z_2^{-j}} \tag{3.10}$$

Let $W(z_1,z_2)$ be an auxiliary function defined as follows:

$$H(z_1,z_2) = \frac{W(z_1,z_2)}{X(z_1,z_2)} \cdot \frac{Y(z_1,z_2)}{W(z_1,z_2)} \tag{3.11}$$

For the case $p = q = 1$, one has

$$\frac{W(z_1,z_2)}{X(z_1,z_2)} = \frac{1}{\sum_{i=0}^{1}\sum_{j=0}^{1} b(i,j) z_1^{-i} z_2^{-j}} \tag{3.12}$$

and

$$\frac{Y(z_1,z_2)}{W(z_1,z_2)} = \sum_{i=0}^{1}\sum_{j=0}^{1} a(i,j) z_1^{-i} z_2^{-j} \tag{3.13}$$

Equations (3.12) and (3.13) lead to the following difference equations:

$$w(n_1,n_2) = x(n_1,n_2) - \sum_{\substack{i=0 \\ i\neq j=0}}^{1} \sum_{j=0}^{1} b(i,j)w(n_1-i,n_2-j)$$

$$= x(n_1,n_2) - \sum_{j=0}^{1} \underset{\sim}{B}_j^T \underset{\sim}{W}_{(n_1,n_2-1;n_2-j)} + w(n_1,n_2)$$

(3.14)

where $b_{00} = 1$, and

$$y(n_1,n_2) = \sum_{i=0}^{1} \sum_{j=0}^{1} a(i,j)\, w(n_1-i,\ n_2-j) = \sum_{j=0}^{1} \underset{\sim}{A}_j^T \underset{\sim}{W}_{(n_1,n_1-1;n_2-j)}$$

(3.15)

The above difference equations lead to the hardware structure of Fig. 3-8, where only one unit delay is required. By comparison, it is clear that the new scheme uses half of the unit delays required in Fig. 3-6, by the old realization. This new form is called the "direct form 2" while the previous forms given in Figures (3-6) and (3-7) are referred to as "direct form 1". Since the new structure consists of the least number of delay elements, it is also called a "canonic form".

For the case of $p = q = 2$ in (3.7), the canonical structure of the Fig. 3-9 results, where only two delay elements are required as expected.

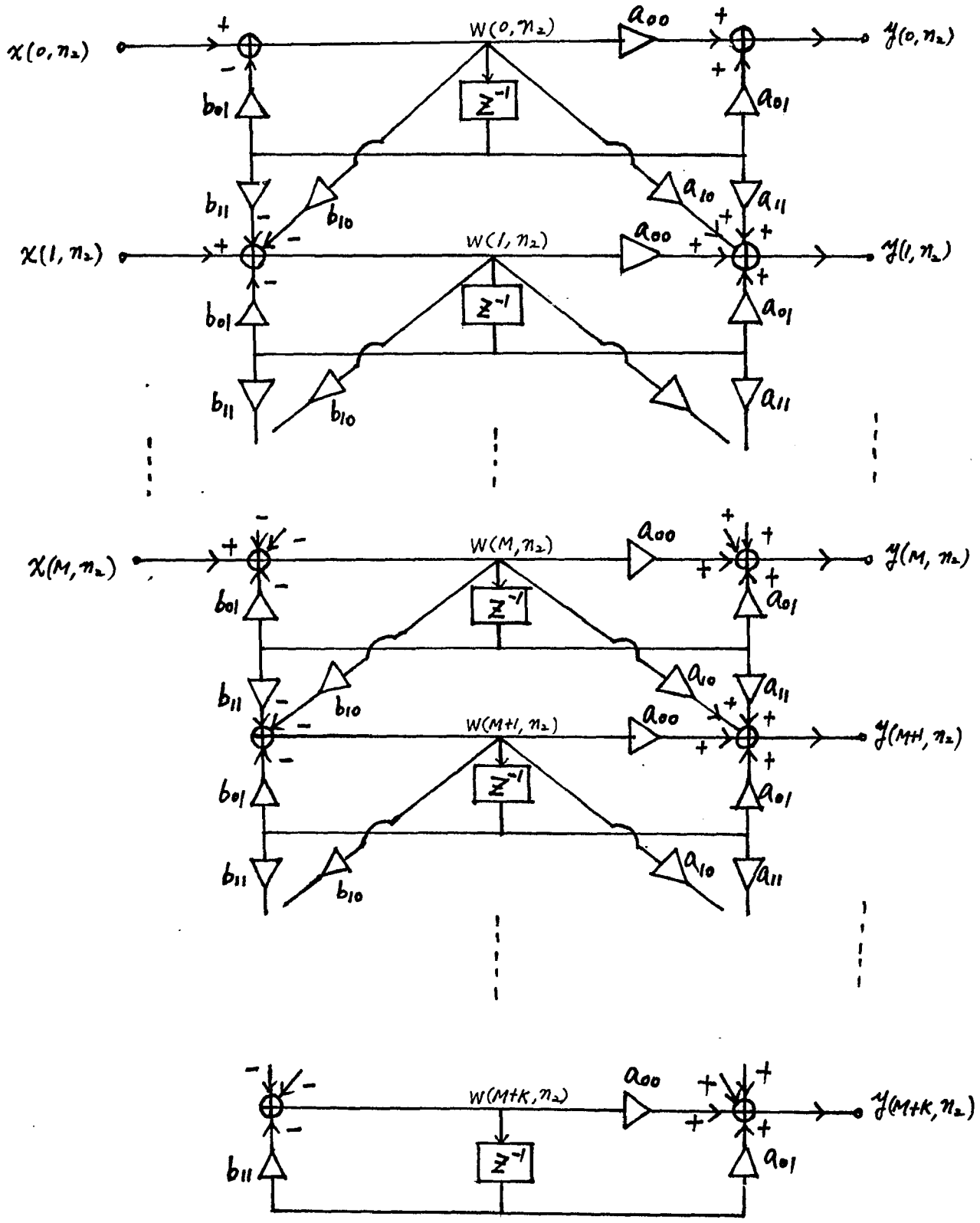


Fig. 3-8 Hardware realization of a general first order 2-D IIR filter (Direct form 2)

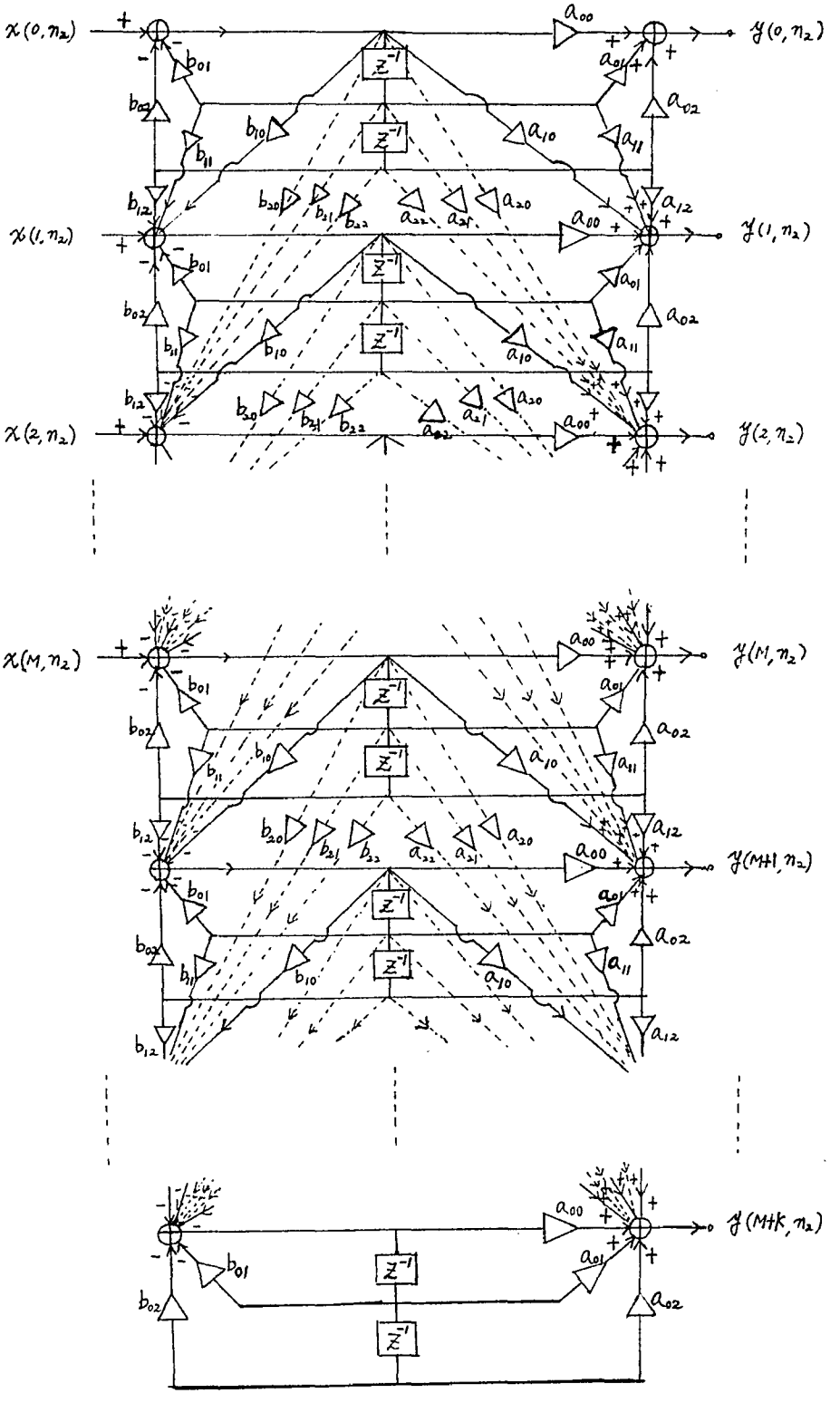


Fig. 3-9 Hardware realization of a general second order 2-D IIR filter (Direct form 2)

3.4 Timing Problem and Data Rearrangement of 2-D IIR Filtering

From Fig. 3-4 depicting a first-order FIR filter, it is seen that each output value $y(k,n_2)$ is independent of any other output value $y(m,n_2)$ and so all such values are available at time n_2.

A different situation occurs, however, in the case of an IIR filter. Consider for example, the general first-order IIR filter shown by the direct form 1 in Fig. 3-6. As may be seen, the output $y(r,n_2)$ depends also on the output $y(r-1,n_2)$, which is multiplied by coefficient b_{10} and then summed, before it can contribute to $y(r,n_2)$. These operations take some finite time, say t_m+t_s, where t_m corresponds to the time needed for multiplying and t_s corresponds to the time needed for summing. (A similar situation occurs in the direct form 2 as shown in Fig. 3-8, where $w(r,n_2)$ depends on $w(r-1,n_2)$ which is multiplied by b_{10} and then summed). For (M+1) input channels, there will be needed a time $M(t_m+t_s)$ to achieve the output $y(M+1,n_2)$. Hence, if $M(t_m+t_s)$ is smaller than the sampling time T_s, then no problem exists. However, if $M(t_m+t_s) \geq T_s$, then the inputting of the input array $x(n_1,n_2)$ must be rearranged in time. For example, taking a 6x7 input array and the special condition $t_m+t_s=T_s$, the input array should be applied to the input of the filter at times which are constant along diagonal lines as shown in Fig. 3-10. Thus sample x(0,0) should be applied at time t_0, samples x(0,1) and x(1,0) should be applied at time

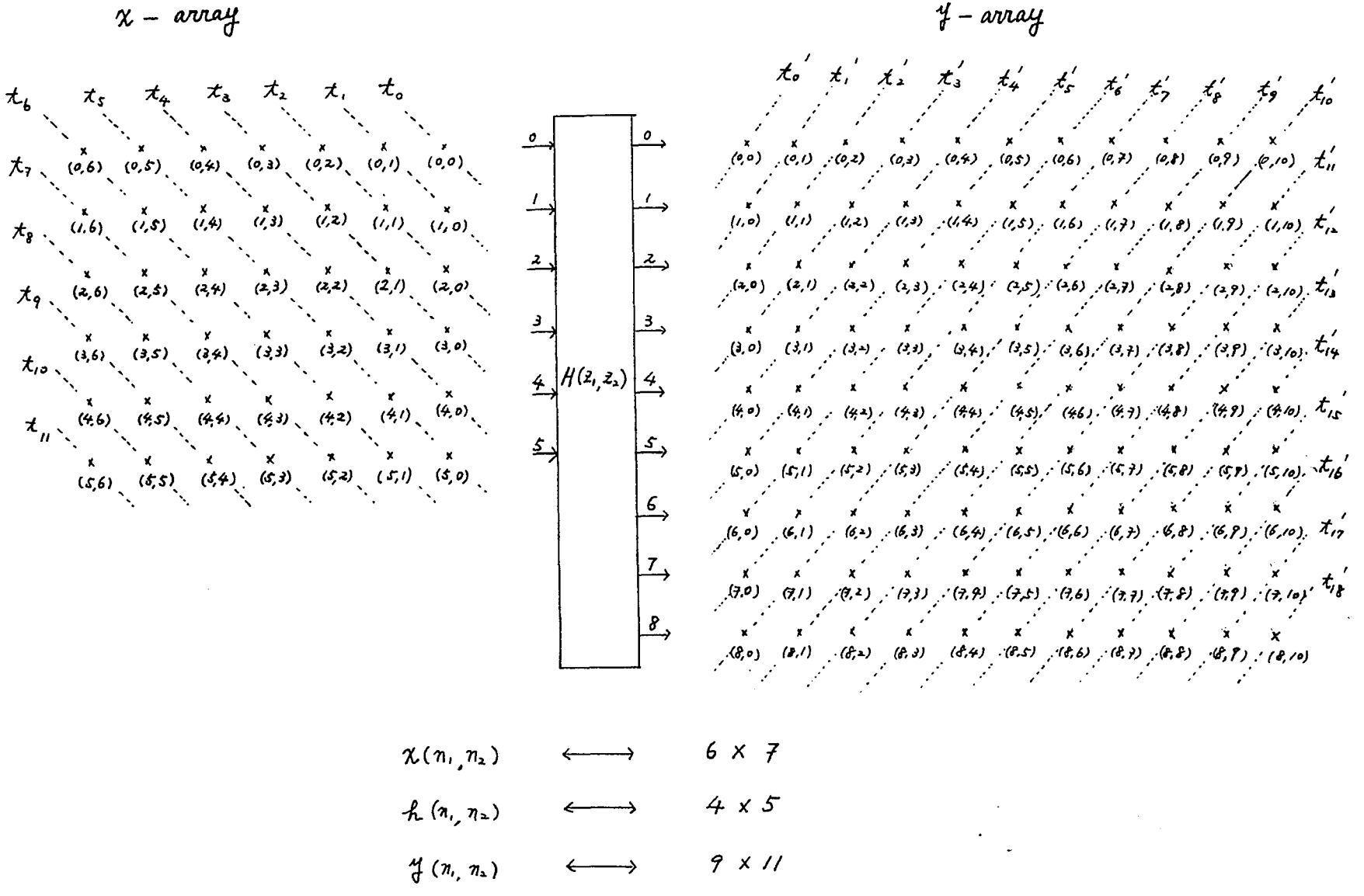


Fig. 3-10 Data arrangement of a 2-D IIR filter (first example)

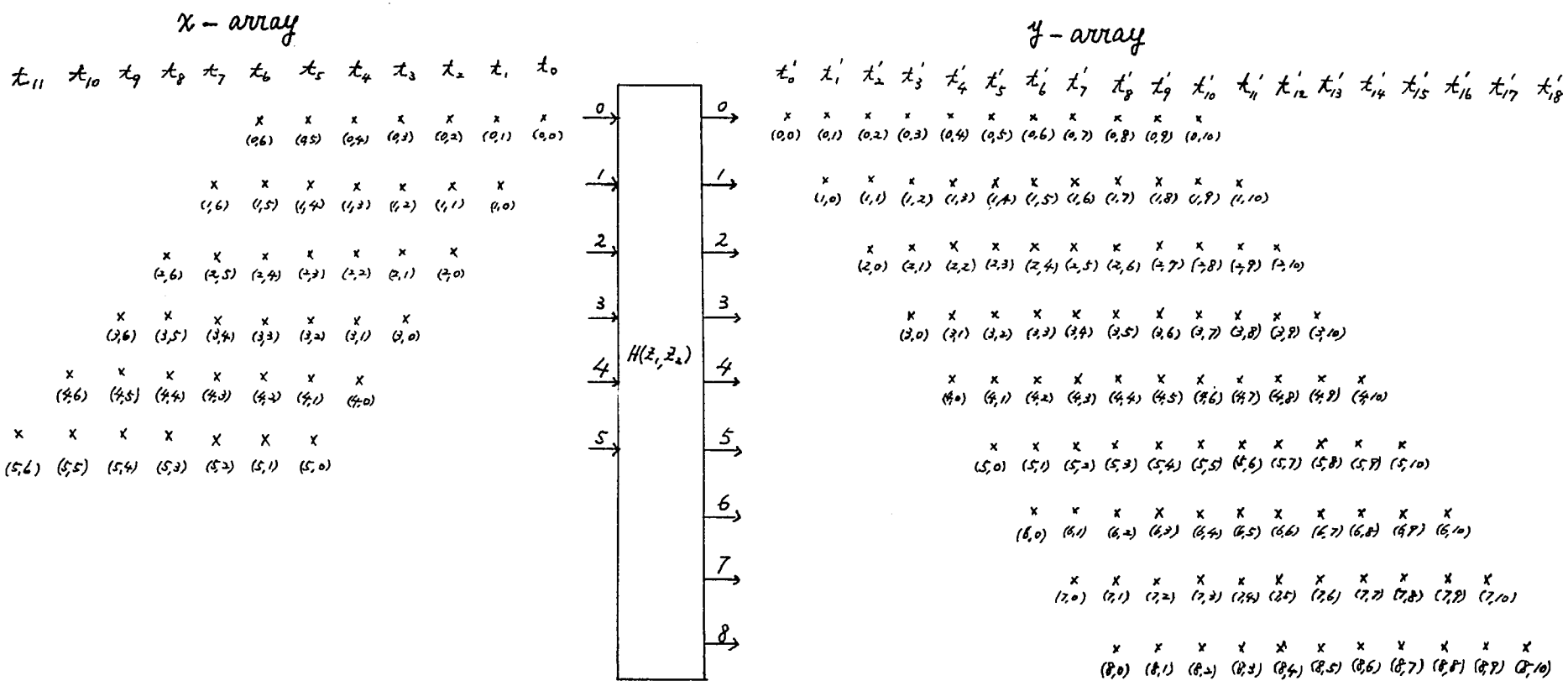
$t_1=t_0+T_s$, samples x(0,2), x(1,1) and x(2,0) should be applied at time $t_2=t_0+2T_s$ and so on as illustrated more clearly in the left-hand sides of Figs. 3-10 and 3-11.

Similarly, the outputs will be available as follows: y(0,0) at $T_0=t_o+\delta_t$, where $\delta_t=t_m+t_s$; y(o,1) and y(1,0) at time T_0+T_s where in this special case $T_s=\delta_t$; y(0,2), y(1,1) and y(2,0) at time T_0+2T_s and so on as also shown in Fig. 3-11. This corresponds to a general first-order IIR filter $H(z_1,z_2)$.

As a second example, consider an input data array $x(n_1,n_2)$ of dimension 50x50, and the same first order IIR filter function as used previously. Assume $t_m+t_s=T_s/10$, then the input data must be rearranged with the first ten rows applied to the input terminals at $t=t_0$, the next ten rows at $t=t_0+T_s$ and so on. The output data $y(n_1,n_2)$ will appear as follows: the first ten rows of output appear at $t=t_0+t_s+t_m=T_0$, the next ten rows of output appear at T_0+T_s and so on. This is clearly shown in Fig. 3-12.

For other conditions, the same principles can be applied and similar diagrams may be prepared.

In conclusion, this chapter sketches a parallel-processing hardware implementation of either FIR or IIR low-order two-dimensional digital filters from a given system function $H(z_1,z_2)$. A vector representation form of a 2-D convolution which easily relates to the hardware realization of 2-D filters was derived.

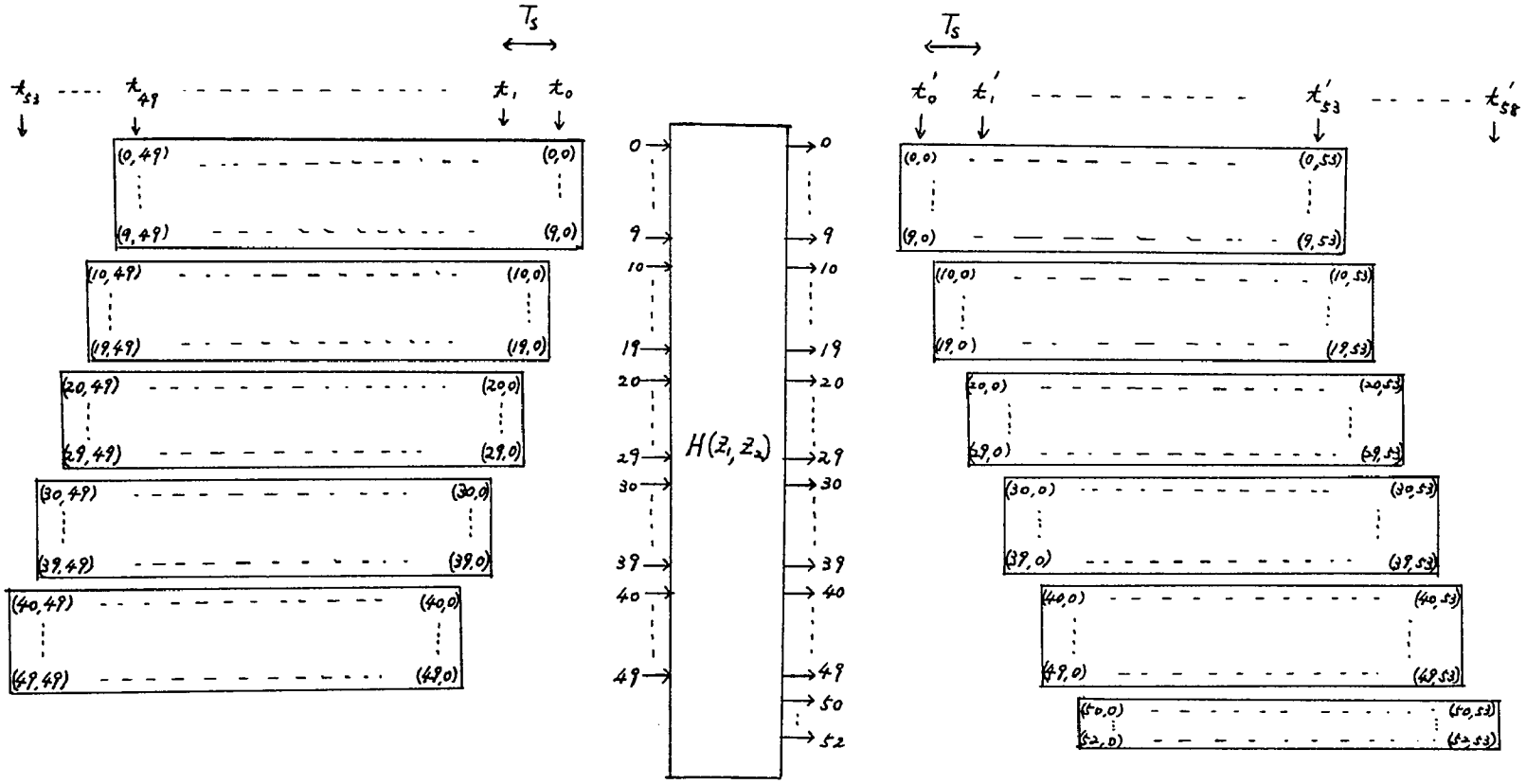


$x(n_1, n_2) \longleftrightarrow 6 \times 7$

$h(n_1, n_2) \longleftrightarrow 4 \times 5$

$y(n_1, n_2) \longleftrightarrow 9 \times 11$

Fig. 3-11 Data arrangement of a 2-D IIR filter (first example, second version)



$x(n_1, n_2) \longleftrightarrow 50 \times 50$

$h(n_1, n_2) \longleftrightarrow 4 \times 5$

$y(n_1, n_2) \longleftrightarrow 53 \times 54$

Fig. 3-12 Data arrangement of a 2-D IIR filter (second example)

Two different forms, Direct Form 1 and Direct Form 2, were proposed for both the general first-order and the general second-order system functions. Some rearrangements of input data pertinent to possible timing problems in the IIR filter case were demonstrated.

CHAPTER FOUR

Hardware Realization of General Order 2-D Digital Filters

4.1 Introduction

It was pointed out in Chapter 3 that a 2-D digital filter can be realized as a parallel-processing structure which consists of multipliers, adders and unit delay elements. However, the proposed hardware realization requires a massive number of multipliers which are proportional to the order of the 2-D filter. As the order of the filter gets larger, the proposed realization method becomes impractical and prohibitively expensive.

Recently, it was pointed out by Croisier et al [30] and also by Peled and Liu [31] that multiplication in a 1-D digital processor with fixed coefficients invariably involved a constant number, namely the fixed filter coefficient or multiplier and a variable number, namely the input signal, called the multiplicand. Hence the most general form of multiplier, that is one involving two arbitrary numbers, is not needed. Multiplication could in fact be implemented by means of "Read Only Memories" (ROM's) together with adders and registers. This new arrangement can be shown to perform at higher speed, to generate lower noise and most importantly to be more economic than the arrangement using multipliers [31].

In the 2-D digital filter case a similar situation occurs and multiplication can also be implemented by means of ROM's together with adders and registers.

By extending the 1-D ROM technique to general order 2-D digital filters, and by including some additional hardware-saving techniques, practical 2-D filter realizations become feasible. The ROM multiplication technique will be discussed first.

4.2 Review of the ROM Technique for Replacing Multipliers Used in 1-D Digital Filtering.

Consider a second-order 1-D IIR filter whose system function is given by

$$H(z) = \frac{a_0+a_1z^{-1}+a_2z^{-2}}{1+b_1z^{-1}+b_2z^{-2}} = \frac{Y(z)}{X(z)} \quad . \tag{4.1}$$

The filter output y(n) may be written in different equation form, as

$$y(n)=a_0x(n)+a_1x(n-1)+a_2x(n-2)-b_1y(n-1)-b_2y(n-2) \tag{4.2}$$

Assume that all signals are bounded by ± 1 and that B binary bits, including the sign bit, in 2's complement form are used to represent data. That is

$$x(n) = -x_0(n) + \sum_{j=1}^{B-1} x_j(n) 2^{-j} \qquad x_j(n) = 0 \text{ or } 1 \tag{4.3}$$

Equation (4.2) can, on substitution of (4.3), be rewritten as

$$y(n) = \sum_{j=1}^{B-1} \{a_0 x_j(n) + a_1 x_j(n-1) + a_2 x_j(n-2) - b_1 y_j(n-1) - b_2 y_j(n-2)\}$$

$$-\{a_0 x_0(n) + a_1 x_0(n-1) + a_2 x_0(n-2) - b_1 y_0(n-1) - b_2 y_0(n-2)\} \tag{4.4}$$

Let an auxiliary function, ψ_j be introduced and defined as

$$\psi_j(a,b,c,d,e) = a_0 a_j + a_1 b_j + a_2 c_j - b_1 d_j - b_2 e_j \tag{4.5}$$

Then (4.4) may be written

$$y_n = \sum_{j=1}^{B-1} \psi_j\{x(n), x(n-1), x(n-2), y(n-1), y(n-2)\}$$

$$-\psi_0\{x(n), x(n-1), x(n-2), y(n-1), y(n-2)\} \tag{4.6}$$

The function ψ_j defined in (4.5) can take on only $2^5 = 32$ distinct values depending on the binary vector that forms its arguments. It can be realized by a ROM having a

capacity of 2^5xB bits, addressed by five address lines, as shown in Fig. 4-1. At the end of B+1 clock periods [31, 32], the desired value of y(n) is in register R_2. The circuit is once again ready to compute a new array of multiplication. Note that all hardware multipliers have been eliminated. The speed of operation, noise properties and reduction of hardware cost of this ROM arrangement are also discussed in [31, 32].

A partition of the auxiliary function ψ_j, or equivalently the ROM, is sometimes required when large bit capacities are needed and when commercially available ROM is insufficient. This is now explained in more detail.

Let ψ_j be a function of N arguments, that is

$$\psi_j(x_1,x_2, \ldots x_N)=\sum_{i=1}^{N} a_i x_{ij}, \quad (x_{ij} = 0 \text{ or } 1) \tag{4.7}$$

The ROM corresponding to ψ_j above needs N addressing lines with each addressing line capable of 2 states, namely 0 or 1. For a word length of B bits, the capacity of the corresponding ROM is $Bx2^N$ bits. If N is a large number, say N = 21 and B = 8, then $8x2^{21} \simeq 2x10^7$ bits and this number exceeds the bit capacity of ROM's commercially

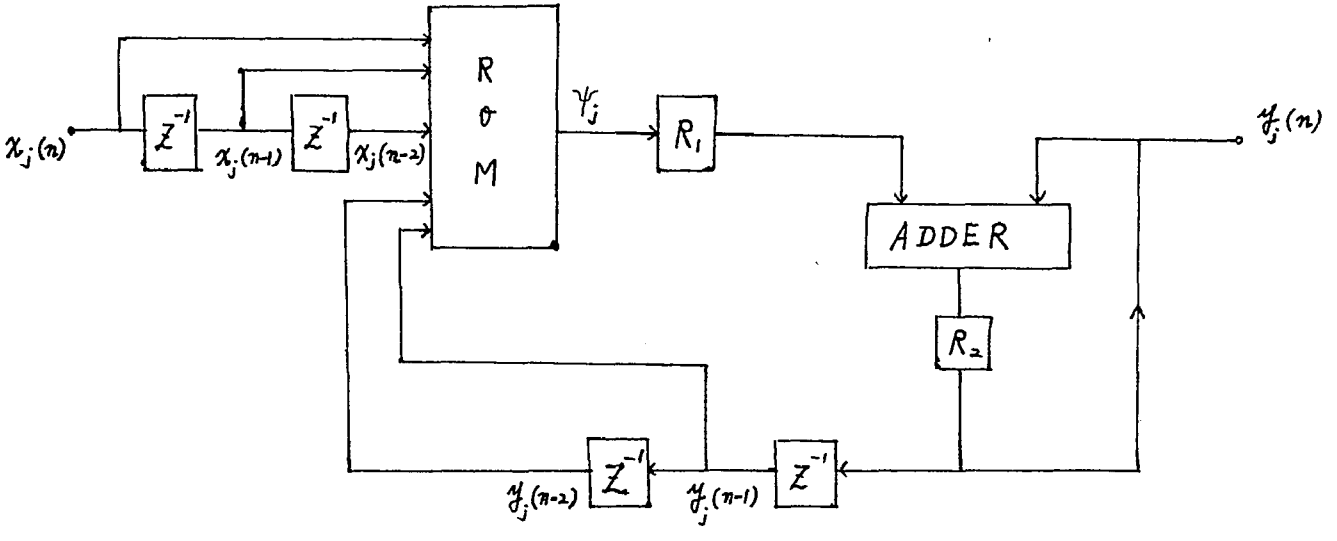


Fig. 4-1 Hardware realization of a 1-D IIR filter with multipliers replaced by ROM,s

available at the time of writing, 1979. Therefore, more ROM's or some method of ROM partitioning is required when N is large.

Let ψ_j be partitioned into L equal subdivisions, that is N=Lxw, and so

$$\psi_j = \sum_{s=1}^{L} \psi_{js} = \sum_{s=1}^{L} \left(\sum_{i=(s-1)w+1}^{s.w} a_i x_{ij} \right) \qquad (4.8)$$

This implies that instead of using one ROM having a capacity of $Bx2^N$ bits, L ROM's with a capacity each of $Bx2^w$ bits are to be used. The total required bit capacity of the partitioned ROM model is then equal to $LxBx2^w$, which is much less than $Bx2^N$ if N is large and w is small. In fact, the ratio of $Bx2^N/LxBx2^w$ equals $2^{w(L-1)}/L$ which may be a large number. More specifically, let N=21, w=3 and B=8. Hence L=N/w=21/3=7 and so capacity has been reduced by a factor $2^{18}/7 \simeq 3x10^4$.

Although the bit capacity of a ROM is greatly reduced by using the partition method, more hardware components must be used and a slowdown of operation speed results. This will be discussed in detail in Section 4.4.

4.3 Hardware Realization of 2-D IIR Filters with ROM Replacement of Multipliers

A 2-D IIR filter can, in general, be represented in one of the following three forms:

(a) Parallel Form

$$H(z_1,z_2)=\sum_{i=1}^{N} H_i(z_1,z_2) \tag{4.9a}$$

(b) Cascade Form

$$H(z_1,z_2)=\prod_{i=1}^{N} H_i(z_1,z_2) \tag{4.9b}$$

(c) General Form

$$H(z_1,z_2)=\frac{\sum_{i=0}^{M_a}\sum_{j=0}^{N_a} a_{ij} z_1^{-i} z_2^{-j}}{1+\sum_{\substack{i=0\\ i\neq j=0}}^{M_b}\sum_{j=0}^{N_b} b_{ij} z_1^{-i} z_2^{-j}} \tag{4.9c}$$

The realization method to be discussed can be applied to any one of the above three forms. For convenience, form (c) will now be used for illustration.

The input-output relation, in difference equation form, of the system function (4.9c) can be expressed as

$$y(n_1,n_2)=\sum_{i=0}^{M_a}\sum_{j=0}^{N_a} a_{ij} x(n_1-i,\, n_2-j)-\sum_{\substack{i=0\\ i\neq j=0}}^{M_b}\sum_{j=0}^{N_b} b_{ij} y(n_1-i,n_2-j) \tag{4.10}$$

Let the input signal $x(n_1,n_2)$ be represented as a B-bit word in two's complement form, similar to the 1-D form given in (4.3)

$$x(n_1,n_2)=-x_0(n_1,n_2)+\sum_{k=1}^{B-1} x_k(n_1,n_2)\cdot 2^{-k} \tag{4.11}$$

where $x_k(n_1,n_2)= 0$ or 1 for k = 0, 1, ... ,B-1.
Substituting (4.10) into (4.11) yields

$$y(n_1,n_2)=\sum_{k=1}^{B-1}\{\sum_{i=0}^{M_a}\sum_{j=0}^{N_a} a_{ij}x_k(n_1-i,n_2-j)-\sum_{\substack{i=0\\ i\neq j=0}}^{M_b}\sum_{j=0}^{N_b} b_{ij}y_k(n_1-i,n_2-j)\}2^{-k}$$

$$-\{\sum_{i=0}^{M_a}\sum_{j=0}^{N_a} a_{ij}x_0(n_1-i,n_2-j)-\sum_{\substack{i=0\\ i\neq j=0}}^{M_b}\sum_{j=0}^{N_b} b_{ij}y_0(n_1-i,n_2-j)\}$$

$$=\sum_{k=1}^{B-1}\psi_k 2^{-k}-\psi_0 \tag{4.12a}$$

where

$$\psi_k=\sum_{i=0}^{M_a}\sum_{j=0}^{N_a} a_{ij}x_k(n_1-i,n_2-j)-\sum_{\substack{i=0\\ i\neq j=0}}^{M_b}\sum_{j=0}^{N_b} b_{ij}y_k(n_1-i,n_2-j) \tag{4.12b}$$

the function ψ_k is a sum of N products where

$$N=(M_a+1)(N_a+1)+(M_b+1)(N_b+1)-1. \tag{4.12c}$$

Each product can take only the values zero, a_{ij} or b_{ij}, depending on whether the coefficients x_k or y_k is equal to zero or one. Therefore, ψ_k can take 2^N distinct values and can be realized by a ROM addressed by the N arguments (x_k's and y_k's). The output $y(n_1,n_2)$ is then obtained by successively shifting and adding the ψ_k's as can be seen from (4.12a).

Assume the filter coefficients and output data are limited to ± 1, and are represented by B-bit words. Further, assume that the bit capacity of the high speed ROM is $Bx2^W$. Therefore, if $Bx2^N$ is greater than $Bx2^W$, a partition of the ROM is required. The partition implies that

$$\psi_k = \sum_{i=1}^{L} \psi_{ki} \tag{4.13}$$

Where ψ_{ki} will have a bit capacity equal to or less than the maximum bit capacity $(B \cdot 2^W)$. The number of sub-ROM's, L, is chosen to be

$$L = \left[\frac{N}{W}\right] = \left[\frac{(M_a+1)(N_a+1)+(M_b+1)(N_b+1)-1}{W}\right] \tag{4.14}$$

where the square bracket means "the greatest integer of." The system function of any IIR filter can now be realized by using the technique discussed in Chapter 3, together with the above mentioned multiplier-ROM replacement. This will

now be illustrated by an example.

4.3.1 Example 1. Realization of a General Form IIR Filter

Consider a general form IIR filter which has the difference equation shown in (4.10) and $M_a=N_a=M_b=N_b=3$. By replacing $x(n_1,n_2)$ with its two's complement form, and assuming the word length is 8 bits, one obtains, from (4.12a),

$$y(n_1,n_2)=\sum_{k=1}^{7}\psi_k 2^{-k}-\psi_0 \qquad (4.15)$$

where ψ_k is defined in (4.12b). Because $M_a=N_a=M_b=N_b=3$, therefore, by virtue of (4.12c), N=31 and so the required ROM capacity of $Bx2^N = 8x2^{31}$ is excessive. Assume next that commercially available high speed ROM's having a bit capacity of $8x2^{10}$ (w=10) may be used. Then a partition, according to (4.14), is required and so

$$L=\left\lceil\frac{4x4+4x4-1}{10}\right\rceil=\lceil 3.1\rceil = 4$$

and four sub-ROM's must be employed.

In general, any four numbers N_1, N_2, N_3 and N_4, which correspond to the bit capacities of four ROM's ($8x2^{N1}$, $8x2^{N2}$, $8x2^{N3}$, and $8x2^{N4}$) respectively and satisfy the relation $N_1+N_2+N_3+N_4=(M_a+1)x(N_a+1)x(M_b+1)x(N_b+1)-1=31$ can be chosen. For convenience, the four sub-ROM's are chosen to

have bit capacities $8x2^8$, $8x2^8$, $8x2^7$ and $8x2^8$. That is $N_1=N_2=N_4=8$ and $N_3=7$. Therefore, from (4.15) one has

$$y(n_1,n_2)=\sum_{k=1}^{7}(\psi_{k1}+\psi_{k2}+\psi_{k3}+\psi_{k4})2^{-k}-(\psi_{01}+\psi_{02}+\psi_{03}+\psi_{04}) \qquad (4.16)$$

The auxiliary sub-ROM functions ψ_{k1}, ψ_{k2}, ψ_{k3} and ψ_{k4} (k=0,1 ... 7) mentioned in (4.16) can, by virtue of (4.12b) be expressed as

$$\psi_{k1}=\sum_{i=0}^{1}\sum_{j=0}^{3}a_{ij}x_k(n_1-i,n_2-j) \qquad \text{(8 terms)}$$

$$\psi_{k2}=\sum_{i=2}^{3}\sum_{j=0}^{3}a_{ij}x_k(n_1-i,n_2-j) \qquad \text{(8 terms)}$$

$$\psi_{k3}=-\sum_{\substack{i=0\\ i\neq j=0}}^{1}\sum_{j=0}^{3}b_{ij}y_k(n_1-i,n_2-j) \qquad \text{(7 terms)}$$

$$\psi_{k4}=-\sum_{i=2}^{3}\sum_{j=0}^{3}b_{ij}y_k(n_1-i,n_2-j) \qquad \text{(8 terms)}$$

A hardware realization of the 2-D filter corresponding to the above arrangement is shown in Fig. 4-2. Note that for simplicity only one channel, namely the r-th channel, is shown whereas in practice (M+K+1) channels, as explained in Chapter 3, Section 3.3, are required. The feedback connections are clearly shown, feedback signals being labeled by square brackets [a,b], whereas ordinary signals are represented by round brackets (a,b). Some actual connections have been

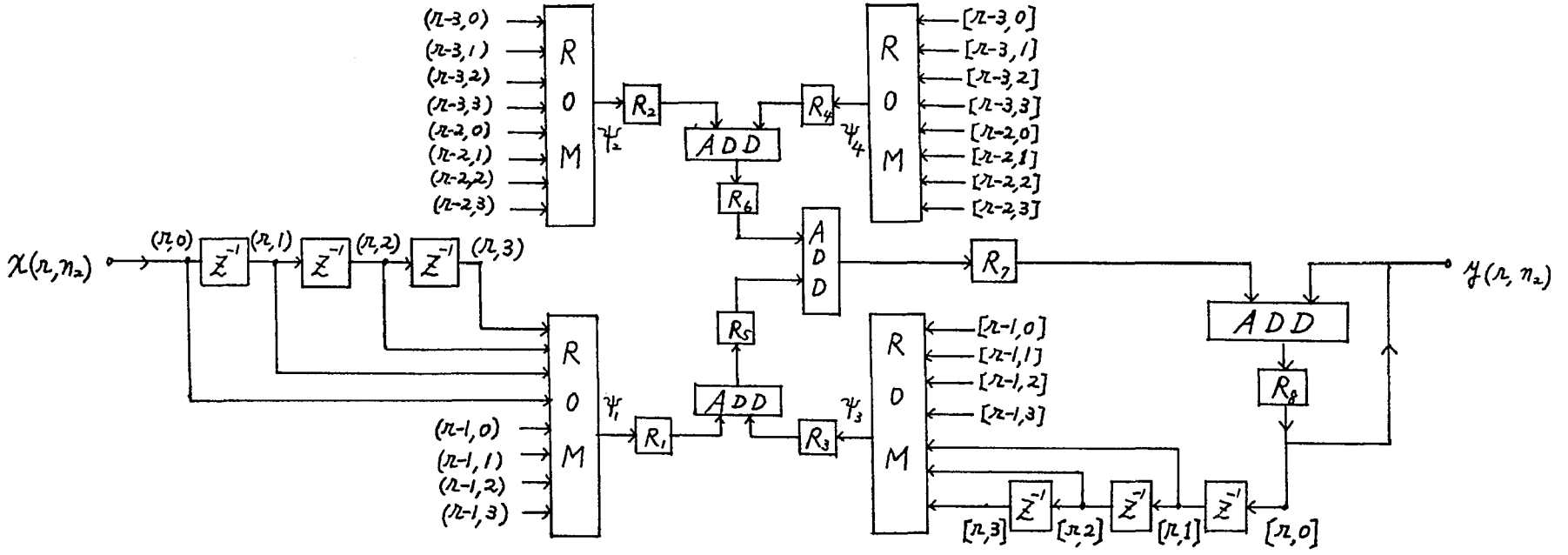


Fig. 4-2 Hardware realization of a general third order 2-D IIR filter (only r-th channel shown)

omitted for clarity. The total hardware count for the one channel represented by Fig. 4-2 consists of four ROM's total bit capacity of $2^3x(2^8+2^8+2^8+2^7)$, six shift registers, four eight-bit adders and eight registers. The operation bit rate depends on the access time of the ROM's,t_ψ, or the addition time of the adders, t_a, whichever is greater. Assuming TTL IC's are used and t_ψ=50ns, t_a=40ns, the operation bit-rate will be $1/t_\psi$= 20MH$_z$. The word rate, (B=8 bit word length), however, is equal to 20MH$_z$/(B+3)=1.82MH$_z$, where the 3 is due to the additional three clock pulses required in the circuit. Fig. 4-3 shows the operation diagram of the r-th channel as shown in Fig. 4-2. As can be seen, the superscript (k) refers to the k-th bit and the subscripts (i,j) refer to the i-th channel at the j-th node and this notation has been adopted solely in this case for additional clarity. Inputs to the ROM's are listed separately and the outputs of ROM's are stored in corresponding registers represented by R_i, i=1 to 4. The results of first level addition, which corresponds to two additions (R_1+R_3 and R_2+R_4), are stored in registers R_5 and R_6 respectively. Since the operation of addition takes one clock pulse, the results of R_5 and R_6 are obtained one clock pulse after the ROM's are addressed. The result of second level addition, corresponding to R_5+R_6, is stored in

CLK	INPUTS OF ROM$_1$								R_1	INPUTS OF ROM$_2$								R_2	INPUTS OF ROM$_3$							R_3
0	$x^7_{r,0}$	$x^7_{r,-1}$	$x^7_{r,-2}$	$x^7_{r,-3}$	$x^7_{r-1,0}$	$x^7_{r-1,-1}$	$x^7_{r-1,-2}$	$x^7_{r-1,-3}$	φ_1^7	$x^7_{r-2,0}$	$x^7_{r-2,-1}$	$x^7_{r-2,-2}$	$x^7_{r-2,-3}$	$x^7_{r-3,0}$	$x^7_{r-3,-1}$	$x^7_{r-3,-2}$	$x^7_{r-3,-3}$	φ_2^7	$y^7_{r,-1}$	$y^7_{r,-2}$	$y^7_{r,-3}$	$y^7_{r-1,0}$	$y^7_{r-1,-1}$	$y^7_{r-1,-2}$	$y_{r-1,-3}$	φ_3^7
1	$x^6_{r,0}$	$x^6_{r,-1}$	$x^6_{r,-2}$	$x^6_{r,-3}$	$x^6_{r-1,0}$	$x^6_{r-1,-1}$	$x^6_{r-1,-2}$	$x^6_{r-1,-3}$	φ_1^6	$x^6_{r-2,0}$	$x^6_{r-2,-1}$	$x^6_{r-2,-2}$	$x^6_{r-2,-3}$	$x^6_{r-3,0}$	$x^6_{r-3,-1}$	$x^6_{r-3,-2}$	$x^6_{r-3,-3}$	φ_2^6	$y^6_{r,-1}$	$y^6_{r,-2}$	$y^6_{r,-3}$	$y^6_{r-1,0}$	$y^6_{r-1,-1}$	$y^6_{r-1,-2}$	$y_{r-1,-3}$	φ_3^6
2	$x^5_{r,0}$	$x^5_{r,-1}$	$x^5_{r,-2}$	$x^5_{r,-3}$	$x^5_{r-1,0}$	$x^5_{r-1,-1}$	$x^5_{r-1,-2}$	$x^5_{r-1,-3}$	φ_1^5	$x^5_{r-2,0}$	$x^5_{r-2,-1}$	$x^5_{r-2,-2}$	$x^5_{r-2,-3}$	$x^5_{r-3,0}$	$x^5_{r-3,-1}$	$x^5_{r-3,-2}$	$x^5_{r-3,-3}$	φ_2^5	$y^5_{r,-1}$	$y^5_{r,-2}$	$y^5_{r,-3}$	$y^5_{r-1,0}$	$y^5_{r-1,-1}$	$y^5_{r-1,-2}$	$y_{r-1,-3}$	φ_3^5
3	$x^4_{r,0}$	$x^4_{r,-1}$	$x^4_{r,-2}$	$x^4_{r,-3}$	$x^4_{r-1,0}$	$x^4_{r-1,-1}$	$x^4_{r-1,-2}$	$x^4_{r-1,-3}$	φ_1^4	$x^4_{r-2,0}$	$x^4_{r-2,-1}$	$x^4_{r-2,-2}$	$x^4_{r-2,-3}$	$x^4_{r-3,0}$	$x^4_{r-3,-1}$	$x^4_{r-3,-2}$	$x^4_{r-3,-3}$	φ_2^4	$y^4_{r,-1}$	$y^4_{r,-2}$	$y^4_{r,-3}$	$y^4_{r-1,0}$	$y^4_{r-1,-1}$	$y^4_{r-1,-2}$	$y_{r-1,-3}$	φ_3^4
4	$x^3_{r,0}$	$x^3_{r,-1}$	$x^3_{r,-2}$	$x^3_{r,-3}$	$x^3_{r-1,0}$	$x^3_{r-1,-1}$	$x^3_{r-1,-2}$	$x^3_{r-1,-3}$	φ_1^3	$x^3_{r-2,0}$	$x^3_{r-2,-1}$	$x^3_{r-2,-2}$	$x^3_{r-2,-3}$	$x^3_{r-3,0}$	$x^3_{r-3,-1}$	$x^3_{r-3,-2}$	$x^3_{r-3,-3}$	φ_2^3	$y^3_{r,-1}$	$y^3_{r,-2}$	$y^3_{r,-3}$	$y^3_{r-1,0}$	$y^3_{r-1,-1}$	$y^3_{r-1,-2}$	$y_{r-1,-3}$	φ_3^3
5	$x^2_{r,0}$	$x^2_{r,-1}$	$x^2_{r,-2}$	$x^2_{r,-3}$	$x^2_{r-1,0}$	$x^2_{r-1,-1}$	$x^2_{r-1,-2}$	$x^2_{r-1,-3}$	φ_1^2	$x^2_{r-2,0}$	$x^2_{r-2,-1}$	$x^2_{r-2,-2}$	$x^2_{r-2,-3}$	$x^2_{r-3,0}$	$x^2_{r-3,-1}$	$x^2_{r-3,-2}$	$x^2_{r-3,-3}$	φ_2^2	$y^2_{r,-1}$	$y^2_{r,-2}$	$y^2_{r,-3}$	$y^2_{r-1,0}$	$y^2_{r-1,-1}$	$y^2_{r-1,-2}$	$y_{r-1,-3}$	φ_3^2
6	$x'_{r,0}$	$x'_{r,-1}$	$x'_{r,-2}$	$x'_{r,-3}$	$x'_{r-1,0}$	$x'_{r-1,-1}$	$x'_{r-1,-2}$	$x'_{r-1,-3}$	φ_1'	$x'_{r-2,0}$	$x'_{r-2,-1}$	$x'_{r-2,-2}$	$x'_{r-2,-3}$	$x'_{r-3,0}$	$x'_{r-3,-1}$	$x'_{r-3,-2}$	$x'_{r-3,-3}$	φ_2'	$y'_{r,-1}$	$y'_{r,-2}$	$y'_{r,-3}$	$y'_{r-1,0}$	$y'_{r-1,-1}$	$y'_{r-1,-2}$	$y_{r-1,-3}$	φ_3'
7	$x^0_{r,0}$	$x^0_{r,-1}$	$x^0_{r,-2}$	$x^0_{r,-3}$	$x^0_{r-1,0}$	$x^0_{r-1,-1}$	$x^0_{r-1,-2}$	$x^0_{r-1,-3}$	φ_1^0	$x^0_{r-2,0}$	$x^0_{r-2,-1}$	$x^0_{r-2,-2}$	$x^0_{r-2,-3}$	$x^0_{r-3,0}$	$x^0_{r-3,-1}$	$x^0_{r-3,-2}$	$x^0_{r-3,-3}$	φ_2^0	$y^0_{r,-1}$	$y^0_{r,-2}$	$y^0_{r,-3}$	$y^0_{r-1,0}$	$y^0_{r-1,-1}$	$y^0_{r-1,-2}$	$y_{r-1,-3}$	φ_3^0
8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	$x^7_{r,1}$	$x^7_{r,0}$	$x^7_{r,-1}$	$x^7_{r,-2}$	$x^7_{r-1,1}$	$x^7_{r-1,0}$	$x^7_{r-1,-1}$	$x^7_{r-1,-2}$	φ_1^7	$x^7_{r-2,1}$	$x^7_{r-2,0}$	$x^7_{r-2,-1}$	$x^7_{r-2,-2}$	$x^7_{r-3,1}$	$x^7_{r-3,0}$	$x^7_{r-3,-1}$	$x^7_{r-3,-2}$	φ_2^7	$y^7_{r,0}$	$y^7_{r,-1}$	$y^7_{r,-2}$	$y^7_{r-1,0}$	$y^7_{r-1,0}$	$y^7_{r-1,-1}$	$y^7_{r-1,-2}$	φ_3^7
12	$x^6_{r,1}$	$x^6_{r,0}$	$x^6_{r,-1}$	$x^6_{r,-2}$	$x^6_{r-1,1}$	$x^6_{r-1,0}$	$x^6_{r-1,-1}$	$x^6_{r-1,-2}$	φ_1^6	$x^6_{r-2,1}$	$x^6_{r-2,0}$	$x^6_{r-2,-1}$	$x^6_{r-2,-2}$	$x^6_{r-3,1}$	$x^6_{r-3,0}$	$x^6_{r-3,-1}$	$x^6_{r-3,-2}$	φ_2^6	$y^6_{r,0}$	$y^6_{r,-1}$	$y^6_{r,-2}$	$y^6_{r-1,0}$	$y^6_{r-1,0}$	$y^6_{r-1,-1}$	$y^6_{r-1,-2}$	φ_3^6

CLK	INPUTS OF ROM$_4$								R_4	R_5	R_6	R_7	R_8
0	$y^7_{r-2,0}$	$y^7_{r-2,-1}$	$y^7_{r-2,-2}$	$y^7_{r-2,-3}$	$y^7_{r-3,0}$	$y^7_{r-3,-1}$	$y^7_{r-3,-2}$	$y^7_{r-3,-3}$	φ_4^7	0	0	0	0
1	$y^6_{r-2,0}$	$y^6_{r-2,-1}$	$y^6_{r-2,-2}$	$y^6_{r-2,-3}$	$y^6_{r-3,0}$	$y^6_{r-3,-1}$	$y^6_{r-3,-2}$	$y^6_{r-3,-3}$	φ_4^6	$\varphi_1^7+\varphi_3^7$	$\varphi_2^7+\varphi_4^7$	0	0
2	$y^5_{r-2,0}$	$y^5_{r-2,-1}$	$y^5_{r-2,-2}$	$y^5_{r-2,-3}$	$y^5_{r-3,0}$	$y^5_{r-3,-1}$	$y^5_{r-3,-2}$	$y^5_{r-3,-3}$	φ_4^5	$\varphi_1^6+\varphi_3^6$	$\varphi_2^6+\varphi_4^6$	$\varphi_1^7+\varphi_2^7+\varphi_3^7+\varphi_4^7$	0
3	$y^4_{r-2,0}$	$y^4_{r-2,-1}$	$y^4_{r-2,-2}$	$y^4_{r-2,-3}$	$y^4_{r-3,0}$	$y^4_{r-3,-1}$	$y^4_{r-3,-2}$	$y^4_{r-3,-3}$	φ_4^4	$\varphi_1^5+\varphi_3^5$	$\varphi_2^5+\varphi_4^5$	$\varphi_1^6+\varphi_2^6+\varphi_3^6+\varphi_4^6$	$(\varphi_1^7+\varphi_2^7+\varphi_3^7+\varphi_4^7)\cdot 2^{-1}$
4	$y^3_{r-2,0}$	$y^3_{r-2,-1}$	$y^3_{r-2,-2}$	$y^3_{r-2,-3}$	$y^3_{r-3,0}$	$y^3_{r-3,-1}$	$y^3_{r-3,-2}$	$y^3_{r-3,-3}$	φ_4^3	$\varphi_1^4+\varphi_3^4$	$\varphi_2^4+\varphi_4^4$	$\varphi_1^5+\varphi_2^5+\varphi_3^5+\varphi_4^5$	$\sum_{j=6}^{7}(\varphi_1^j+\varphi_2^j+\varphi_3^j+\varphi_4^j)\cdot 2^{-(j-5)}$
5	$y^2_{r-2,0}$	$y^2_{r-2,-1}$	$y^2_{r-2,-2}$	$y^2_{r-2,-3}$	$y^2_{r-3,0}$	$y^2_{r-3,-1}$	$y^2_{r-3,-2}$	$y^2_{r-3,-3}$	φ_4^2	$\varphi_1^3+\varphi_3^3$	$\varphi_2^3+\varphi_4^3$	$\varphi_1^4+\varphi_2^4+\varphi_3^4+\varphi_4^4$	$\sum_{j=5}^{7}(\varphi_1^j+\varphi_2^j+\varphi_3^j+\varphi_4^j)\cdot 2^{-(j-4)}$
6	$y'_{r-2,0}$	$y'_{r-2,-1}$	$y'_{r-2,-2}$	$y'_{r-2,-3}$	$y'_{r-3,0}$	$y'_{r-3,-1}$	$y'_{r-3,-2}$	$y'_{r-3,-3}$	φ_4'	$\varphi_1^2+\varphi_3^2$	$\varphi_2^2+\varphi_4^2$	$\varphi_1^3+\varphi_2^3+\varphi_3^3+\varphi_4^3$	$\sum_{j=4}^{7}(\varphi_1^j+\varphi_2^j+\varphi_3^j+\varphi_4^j)\cdot 2^{-(j-3)}$
7	$y^0_{r-2,0}$	$y^0_{r-2,-1}$	$y^0_{r-2,-2}$	$y^0_{r-2,-3}$	$y^0_{r-3,0}$	$y^0_{r-3,-1}$	$y^0_{r-3,-2}$	$y^0_{r-3,-3}$	φ_4^0	$\varphi_1'+\varphi_3'$	$\varphi_2'+\varphi_4'$	$\varphi_1^2+\varphi_2^2+\varphi_3^2+\varphi_4^2$	$\sum_{j=3}^{7}(\varphi_1^j+\varphi_2^j+\varphi_3^j+\varphi_4^j)\cdot 2^{-(j-2)}$
8	0	0	0	0	0	0	0	0	0	$\varphi_1^0+\varphi_3^0$	$\varphi_2^0+\varphi_4^0$	$\varphi_1'+\varphi_2'+\varphi_3'+\varphi_4'$	$\sum_{j=2}^{7}(\varphi_1^j+\varphi_2^j+\varphi_3^j+\varphi_4^j)\cdot 2^{-(j-1)}$
9	0	0	0	0	0	0	0	0	0	0	0	$\varphi_1^0+\varphi_2^0+\varphi_3^0+\varphi_4^0$	$\sum_{j=1}^{7}(\varphi_1^j+\varphi_2^j+\varphi_3^j+\varphi_4^j)\cdot 2^{-j}$
10	0	0	0	0	0	0	0	0	0	0	0	0	$\sum_{j=1}^{7}(\varphi_1^j+\varphi_2^j+\varphi_3^j+\varphi_4^j)\cdot 2^{-j}-(\varphi_1^0+\varphi_2^0+\varphi_3^0+\varphi_4^0)$
11	$y^7_{r-3,1}$	$y^7_{r-3,0}$	$y^7_{r-3,-1}$	$y^7_{r-3,-2}$	$y^7_{r-3,0}$	$y^7_{r-3,0}$	$y^7_{r-3,-1}$	$y^7_{r-3,-2}$	φ_4^7	0	0	0	0
12	$y^6_{r-3,1}$	$y^6_{r-3,0}$	$y^6_{r-3,-1}$	$y^6_{r-3,-2}$	$y^6_{r-3,1}$	$y^6_{r-3,0}$	$y^6_{r-3,-1}$	$y^6_{r-3,-2}$	φ_4^6	$\varphi_1^7+\varphi_3^7$	$\varphi_2^7+\varphi_4^7$	0	0

Fig. 4-3 Timing and operation diagram associated with Fig. 4-2

R_7 and occurs two clock pulses after the ROMs are addressed. The third level addition of R_7 and R_8 must be undertaken with proper scaling factors and the results are stored in R_8. At the end of the 11th clock pulse, the content of R_8 becomes the r-th channel output. Since the circuit shown in Fig. 4-2 needs 3 levels of addition and uses 8-bit word length, the timing diagram of Fig. 4-3 implies that the word rate of Fig. 4-2 is equal to bit-rate/(8+3)=20MH_z/11 =1.82MH_z, as mentioned before. In general, the word rate depends not only on the word length, but also on the number of levels of addition required. If this number is N and if the word length is B, then the word rate is (bit-rate)/(B+N).

4.4 Hardware Realization of 2-D FIR Filters.

4.4.1 Introduction

As mentioned in Chapter One, FIR filters can be designed to have nonlinear phase or linear phase characteristics. For nonlinear phase FIR filters, the realization closely follows that given in Section 4.3. On the other hand, the linear phase filters have the valuable property that signals are processed without distortion. Hence, the following sections will confine themselves to the realization of linear phase FIR filters.

In the 1-D case, the zeroes of linear phase FIR filters have quadrantal symmetry, therefore the 1-D system function can be factorized into sub-system functions, each of which still retains symmetry. As a consequence, a 2-D linear phase FIR filter may be designed by starting with a 1-D linear phase FIR filter and using a cosine transformation from 1-D to 2-D (known as McClellan Transformation).

The hardware saving realization of 2-D linear phase FIR filters will be achieved first by reducing the number of filter coefficients through the factorization of 1-D system functions and then by applying the 1-D to 2-D McClellan transformation. Further saving on realization will be achieved through the introduction of code converters which reduce the number of ROM addressing lines and therefore the required bit capacity of the ROM.

The following is a detailed discussion of the above topics.

4.4.2 Factorization of the System Function of 1-D FIR Linear Phase Filters Due to Quadrantal Symmetry.

It is well known that a linear phase FIR filter must have an impulse response which satisfies the symmetry relation:

$$h(n) = \pm h(N-1-n) \qquad \text{for } 0 \leq n \leq N-1 \tag{4.17}$$

where (N-1) is the order of the system function H(z). The plus sign is associated with bandlimited filters, and the minus sign is associated with differentiators and Hilbert Transformers.

It is also well known [10] that the zeros of the linear phase FIR filter system function H(z) are identical to the zeros of $H(z^{-1})$. As a result, if H(z) has a complex zero at $z_i = r_i e^{j\theta i}$ with $r_i \neq 1$, $\theta_i \neq 0$, then H(z) must also have a mirror image zero at $z^{-1} = (1/r_i)e^{-j\theta i}$. Since the impulse response of the filter is real, every complex zero of H(z) has a complex conjugate zero and so H(z) has quadrantal symmetry. In view of the above remarks, H(z) consists of four elementary factors:

$$H_i(z) = (1-z^{-1}r_i e^{j\theta i}) x (1-z^{-1}r_i e^{-j\theta i}) x (1-z^{-1}\frac{1}{r_i} e^{j\theta i}) x$$

$$(1-z^{-1}\frac{1}{r_i} e^{-j\theta i})$$

which may be written

$$H_i(z) = \sum_{k=0}^{4} h_{ki} z^{-k} \qquad (1 \leq i \leq N_1)$$

where

$$h_{0i}=h_{4i}=1 \quad ; \quad h_{1i}=h_{3i}=-2(r_i+\frac{1}{r_i})\cos\theta_i$$

$$h_{2i}=r_i^2+\frac{1}{r_i^2}+4\cos\theta_i$$

If the zeros lie on the unit circle and $\theta_i \neq 0,\pi$, then the elementary factor $H_i(z)$ is modified to the form

$$H_i'(z)=(1-z^{-1}e^{j\theta_i})(1-z^{-1}e^{-j\theta_i})=\sum_{k=0}^{2} h'_{ki}z^{-k}, \quad (1\leq i\leq N_2)$$

If $r_i \neq 1$, but $\theta_i=0$ or π, then the modification

$$H_i''(z)=(1\pm r_i z^{-1})(1\pm\frac{1}{r_i}z^{-1})=\sum_{k=0}^{2} h''_{ki}z^{-k}, \quad (1\leq i\leq 2)$$

results, where the plus sign corresponds to $\theta_i=\pi$ and the minus sign corresponds to $\theta_i=0$.

Finally, if $r_i=1$ and $\theta_i=0, \pi$, then

$$H_i'''(z)=1\pm z^{-1}=\sum_{k=0}^{1} h'''_{ki}z^{-k} \qquad (1\leq i\leq 2)$$

From the above analysis one can conclude that for any linear phase FIR filter, the system function H(z) can be written

$$H(z)=H_0 \times \prod_{j=1}^{N_1} \left(\sum_{i=0}^{4} h_{ij} z^{-i} \right) \times \prod_{j=1}^{N_2} \left(\sum_{i=0}^{2} h'_{ij} z^{-i} \right) \times \left(\sum_{i=0}^{2} h''_{i} z^{-i} \right) \times \left(\sum_{i=0}^{1} h'''_{i} z^{-i} \right) \tag{4.18}$$

where $4N_1+2N_2+2+1=N-1$, and H_0 is a real constant. One can now go from 1-D to 2-D by applying McClellan's transformation to each system function factor in (4.18). This transformation will be reviewed in the next section, but the end result obtained is a 2-D system function as given below (assume H(z) is a bandlimited function):

$$H(z_1,z_2)=H_0 \times \prod_{k=1}^{N_1} \left(\sum_{i=0}^{4} \sum_{j=0}^{4} h_{ijk} z_1^{-i} z_2^{-j} \right) \times \prod_{k=1}^{N_2} \left(\sum_{i=0}^{2} \sum_{j=0}^{2} h'_{ijk} z_1^{-i} z_2^{-j} \right) \times \left(\sum_{i=0}^{2} \sum_{j=0}^{2} h''_{ij} z_1^{-i} z_2^{-j} \right) \tag{4.19}$$

4.4.3 Review of The McClellan Transformation.

Consider a N-th order (N odd) 1-D linear phase FIR filter with system function

$$H(z)\Big|_{z=e^{j\omega}} = \sum_{n=0}^{N-1} h(n) z^{-n} = e^{-j\frac{N-1}{2}\omega} \times \sum_{n=0}^{\frac{N-1}{2}} a_n \cos n\omega \tag{4.20}$$

Note that $h(n)=h(N-1-n)$, a condition of the linear phase constraints, and also that $a_n=2h(\frac{N-1}{2}-n)$. The McClellan transformation states that one may relate the 1-D frequency

variable ω with the 2-D frequency variables ω_1 and ω_2 by means of the equation

$$\cos\omega = A\cos\omega_1 + B\cos\omega_2 + C\cos\omega_1 \cos\omega_2 + D \tag{4.21}$$

When the above formula is substituted into the 1-D systems function (4.20), one obtains a 2-D linear phase FIR filter function,

$$H(z_1,z_2)\Big|_{\substack{z_1=e^{j\omega_1}\\ z_2=e^{j\omega_2}}} = e^{-j\frac{N-1}{2}(\omega_1+\omega_2)} \cdot \sum_{n_1=0}^{\frac{N-1}{2}} \sum_{n_2=0}^{\frac{N-1}{2}} b_{n_1,n_2} \cos n_1\omega_1 \cos n_2\omega_2$$

$$= \sum_{n_1=0}^{N-1} \sum_{n_2=0}^{N-1} h(n_1,n_2) z_1^{-n_1} z_2^{-n_2} \tag{4.22}$$

Note that the values of A, B, C and D in (4.21) determine the frequency spectra of the transformed 2-D filter. For instance, the choice A=B=C=-D=0.5 implies a resultant circular symmetric filter, whereas by letting A=-B=0.5 and C=D=0 one obtains a 2-D fan-shaped filter, as mentioned in Chapter One.

By comparing (4.20) with (4.22) one finds that the $\frac{N+1}{2}$ distinct filter coefficients of (4.20) must now be reconciled with $\frac{N+1}{2}x\frac{N+1}{2}$ distinct filter coefficients in (4.22). Therefore, the number of multipliers required to realize (4.22), based on the parallel processing realization discussed in previous sections, will be the large number $\frac{N+1}{2}x\frac{N+1}{2}$ for each

channel. Even if the ROM multiplier replacement with partitioning technique were employed, this large number would imply that a great deal of heardware is required to implement it. A technique which reduces the number of filter coefficients h(i,j) of the transformed 2-D filter, and thereby reduces the required total number of multipliers for realization, was the factorization method mentioned in Section 4.4.2. An example of the combined factorization and transformation methods now follows.

Consider a 1-D linear phase FIR filter with N=11, and assume it can be factored according to (4.18)

$$H(z)=\sum_{i=0}^{10} h_i z^{-i}$$

$$=H_0\cdot\{\sum_{n=0}^{4} h^{(1)}(n)z^{-n}\}\cdot\{\sum_{n=0}^{2} h^{(2)}(n)z^{-n}\}\cdot\{\sum_{n=0}^{2} h^{(3)}(n)z^{-n}\}\cdot\{\sum_{n=0}^{2} h^{(4)}(n)z^{-n}\}$$

Apply the McClellan transformation (4.21) to the above equation, taking H_0=1. The result is

$$H(z_1,z_2)=\{\sum_{i=0}^{4}\sum_{j=0}^{4} d_{ij}^{(1)} z_1^{-i} z_2^{-j}\}\cdot\prod_{k=2}^{4}\{\sum_{i=0}^{2}\sum_{j=0}^{2} d_{ij}^{(k)} z_1^{-i} z_2^{-j}\} \tag{4.23}$$

From the above equation, it is clear that the total distinct 2-D filter coefficients are $\frac{5+1}{2}x\frac{5+1}{2}+3x\frac{3+1}{2}x\frac{3+1}{2}=9+3x4=21$, which is less than $\frac{N+1}{2}x\frac{N+1}{2}=6x6=36$, if the direct transformation without factorization is used. As may be seen, a

considerable saving of coefficients or multipliers has resulted.

Although the factored transformation uses fewer filter coefficients than the direct transformation, more bits are required for the same accuracy of data representation as a result of the multiplication of factors, if the same filter specification is to be met. The trade-off between saving filter coefficients and impaired filter coefficients will be left for future research.

As a result of filter coefficient symmetry, fewer ROM addressing lines can be achieved. This leads to saving of ROM capacity and will now be illustrated.

4.4.4 Saving of ROM Address Lines by Means of Code Converters.

The impact of filter coefficient symmetry on the input-output relation of a 2-D FIR linear phase filter system,

$$y(n_1,n_2)=\sum_{k_1=0}^{N-1}\sum_{k_2=0}^{N-1} h(k_1,k_2)x(n_1-k_1,n_2-k_2) \qquad (4.24)$$

will now be discussed.

If the filter has N=2M, the following symmetry conditions prevail

$$h(n_1,n_2)=h(N-1-n_1,n_2)=h(n_1,N-1-n_2)=h(N-1-n_1,N-1-n_2)$$

With these substitutions, (4.24) becomes

$$y(n_1,n_2)=\sum_{k_1=0}^{M-1}\sum_{k_2=0}^{M-1} h(k_1,k_2)\{x(n_1-k_1,n_2-k_2)+x(n_1-N+1+k_1,n_2-k_2)+$$

$$x(n_1-k_1,n_2-N+1+k_2) + x(n_1-N+1+k_1,n_2-N+1+k_2)\} \tag{4.25}$$

Once again we assume all signals are bounded by ± 1 and B-bit words in two's complement form are used for data representation as in (4.11) with the substitution

$$A_j(k_1,k_2)=x_j(n_1-k_1,n_2-k_2)+x_j(n_1-N+1+k_1,n_2-k_2)$$

$$+x_j(n_1-k_1,n_2-N+1+k_2)+x_j(n_1-N+1+k_2,n_2-N+1+k_2) \tag{4.26}$$

equation (4.25) becomes

$$y(n_1,n_2)=\sum_{k_1=0}^{M-1}\sum_{k_2=0}^{M-1} h(k_1,k_2)\{\sum_{j=1}^{B-1} A_j(k_1,k_2)-A_0(k_1,k_2)\} \tag{4.27}$$

By defining the auxiliary function, ψ_j,

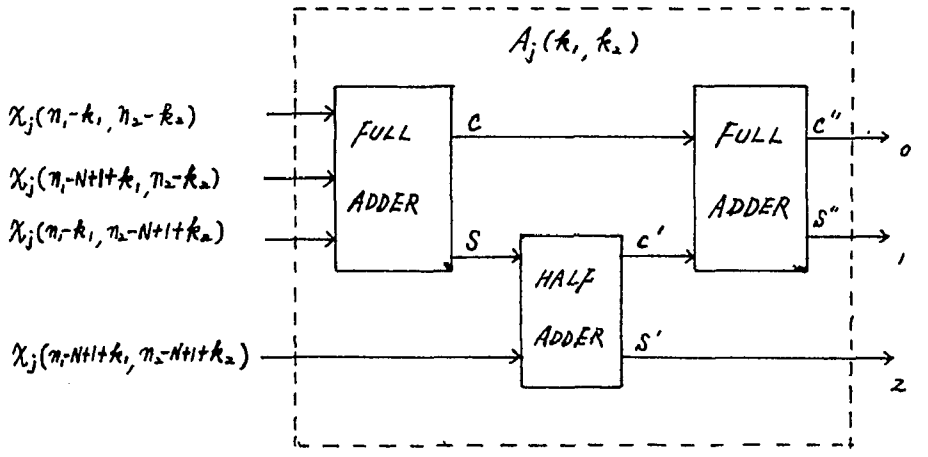
$$\psi_j = \sum_{k_1=0}^{M-1}\sum_{k_2=0}^{M-1} h(k_1,k_2)A_j(k_1,k_2) \tag{4.28}$$

equation (4.27) may be put into its final form

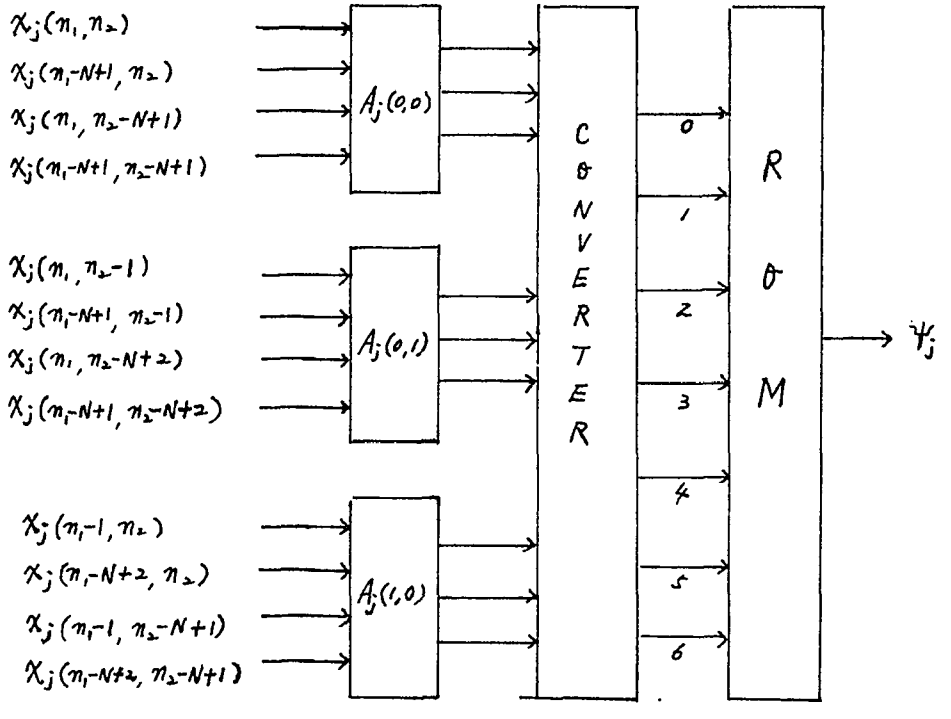
$$y(n_1,n_2)=\sum_{j=1}^{B-1}\psi_j 2^{-j}-\psi_0 \tag{4.29}$$

Notice the similarity of this representation with (4.15). Because variable $x_j(n_1,n_2)$ can have only two values, namely zero or one, therefore variable $A_j(k_1,k_2)$ can only have the five values 0, 1, 2, 3 or 4. In binary form these five states can be represented by means of three bits ($2^3=8$), leaving three states 5, 6, 7, as "don't care" states. This is equivalent to the statement that function $A_j(k_1,k_2)$ is a code converter, having four input lines and three output lines. A possible realization of such a code converter is shown in Fig. 4-4a. From (4.27) there are M^2 distinct terms $A_j(k_1,k_2)$ each of which requires $3M^2$ output lines. Various schemes of saving address lines to the ROM are possible, one such scheme which combines 3 $A_j(k_1,k_2)$'s is shown in Fig. 4-4b. Here 12 address lines producing a possible $5^3=125$ distinct states are converted into 7 address lines, corresponding to $2^7=128$ distinct states.

The code converters shown in Fig. 4-4a and Fig. 4-4b are used to save the memory address lines which in turn will save the bit capacity of the ROM. However, due to the hardware cost of the additional converter and the slowdown



(a)



(b)

Fig. 4-4 (a). Code converter designed for saving ROM bit capacity. (b). An example for realizing the auxiliary function ψ_j.

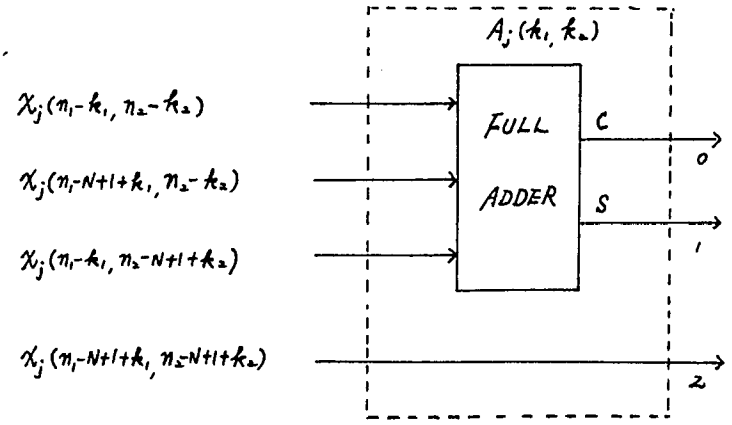
of operation speed, a trade-off between ROM capacity saved and converter complexity required must be reached in practice. One of these schemes, which uses only one full adder in the code converter for $A_j(k_1,k_2)$ is shown in Fig. 4-5a. This converter is then used M^2 times at the input of the ROM as shown in Fig. 4-5b.

A similar discussion is pertinent when the number of data points, N, is odd; that is when N=2M+1. This discussion is presented in Appendix 4.

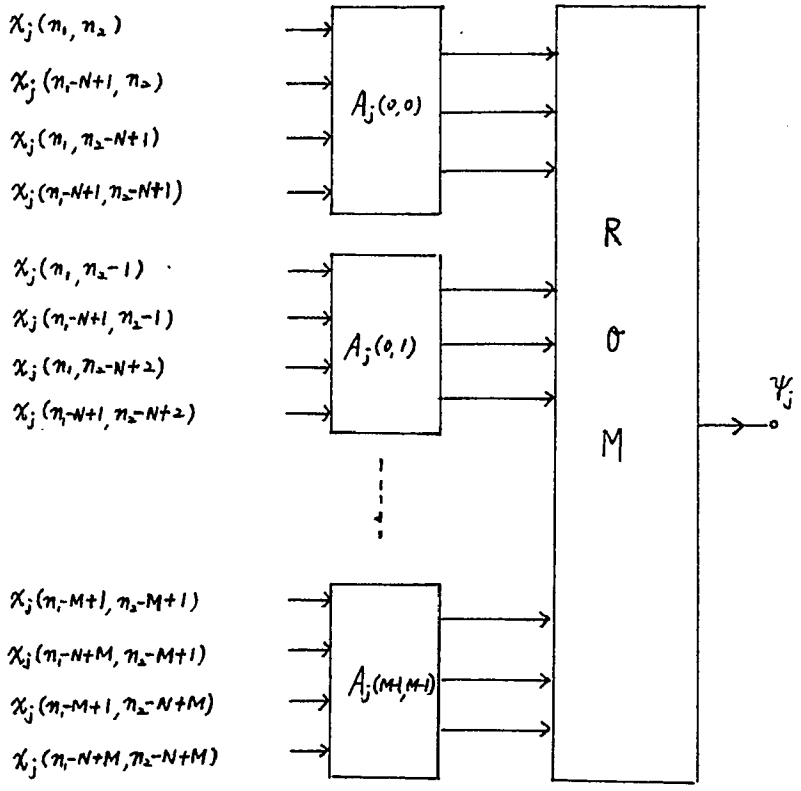
We now continue with an example which illustrates the hardware saving methods based on the preceeding three sub-sections, namely 4.4.2 through 4.4.4.

4.4.5 Realization Example of a Linear Phase 2-D FIR Filter.

The system function given in (4.23) is the factored form in 2-D, obtained by transformation of a 1-D linear phase FIR filter, as outlined in Section 4.4.2. This factored form will now be used in a hardware realization. The first step in our procedure identifies the required four factors as



(a)



(b)

Fig. 4-5 (a). Simpler code converter for saving ROM bit-capacity. (b). A second example for realizing the auxiliary function ψ_j.

$$H(z_1,z_2)=Y(z_1,z_2)/X(z_1,z_2)$$

$$= \frac{G(z_1,z_2)}{X(z_1,z_2)} \cdot \frac{E(z_1,z_2)}{G(z_1,z_2)} \cdot \frac{F(z_1,z_2)}{E(z_1,z_2)} \cdot \frac{Y(z_1,z_2)}{F(z_1,z_2)}$$

$$= H_1(z_1,z_2) \cdot H_2(z_1,z_2) \cdot H_3(z_1,z_2) \cdot H_4(z_1,z_2) \qquad (4.30)$$

where $G(z_1,z_2)$, $E(z_1,z_2)$ and $F(z_1,z_2)$ are auxiliary polynomials in z_1 and z_2 and where N=11. From (4.23) and (4.30) now follows

$$g(n_1,n_2)= \sum_{i=0}^{4} \sum_{j=0}^{4} d_{i,j}^{(1)} x(n_1-i,n_2-j)$$

$$e(n_1,n_2)= \sum_{i=0}^{2} \sum_{j=0}^{2} d_{i,j}^{(2)} g(n_1-i,\ n_2-j)$$

$$f(n_1,n_2)= \sum_{i=0}^{2} \sum_{j=0}^{2} d_{i,j}^{(3)} e(n_1-i,n_2-j)$$

$$y(n_1,n_2)= \sum_{i=0}^{2} \sum_{j=0}^{2} d_{i,j}^{(4)} f(n_1-i,n_2-j) \qquad (4.31)$$

With the aid of Appendix 4 which refers to code converters (for N=2M+1) and in our case for N=11, one obtains with the use of 2's complement notation

$$g(n_1,n_2) = \sum_{k=1}^{B-1} \psi_{k1} 2^{-k} - \psi_{01} \qquad (4.32)$$

The auxiliary code converter functions are defined

$$A_k(i,j)=x_k(n_1-i,n_2-j)+x_k(n_1-N+1+i,n_2-j)$$
$$+x_k(n_1-i,n_2-N+1+j)+x_k(n_1-N+1+i,n_2-N+1+j)$$
$$B_k(i,2)=x_k(n_1-i,n_2-2)+x_k(n_1-4+i,n_2-2)$$
$$C_k(2,j)=x_k(n_1-2,n_2-j)+x_k(n_1-2,n_2-4+j) \qquad (4.33)$$

and are related to the auxiliary ψ functions in the form

$$\psi_{kl}=\sum_{i=0}^{1}\sum_{j=0}^{1} d_{ij}^{(1)} A_k(i,j)+\sum_{i=0}^{1} d_{i,2}^{(1)} B_k(i,2)+\sum_{j=0}^{1} d_{2j}^{(1)} C_k(2,j)$$
$$+d_{22}^{(1)} x_k(n_1-2,n_2-2) \qquad (4.34)$$

From (4.33) and Figures 4-4a or 4-5a it is clear that converter $A_k(i,j)$ has four input lines and three output lines. On the other hand, converters $B_k(i,2)$ and $C_k(2,j)$ have two input lines and two output lines and so no conversion or saving is obtained here. Therefore, the auxiliary function ψ_{kl} of (4.34) corresponds to 21 addressing lines. In fact, the first term of (4.34) represents 4x3=12 lines; the second and third term each contribute 2x2=4 lines; and the last single term represents an additional line. Therefore, without ROM partitioning we would require a ROM capacity of $Bx2^{21}$ bits, which exceeds present commercially available high speed ROM capacities. Therefore a partition of the ROM is necessary. The word length B is assumed to be 8.

Assume the maximum bit capacity of a high speed ROM is $8x2^{10}=8192$ bits. It is clear that ψ_{k1} needs 3 ROMs to implement, therefore ψ_{k1} is divided into ψ_{k11}, ψ_{k12} and ψ_{k13}, corresponding to the convenient sizes $8x2^8$, $8x2^8$ and $8x2^5$ bits respectively. That is

$$\psi_{k11}=d_{00}^{(1)}A_k(0,0)+d_{01}^{(1)}A_k(0,1)+d_{02}^{(1)}B_k(0,2)$$

$$\psi_{k12}=d_{11}^{(1)}A_k(1,1)+d_{10}^{(1)}A_k(1,0)+d_{12}^{(1)}B_k(1,2)$$

$$\psi_{k13}=d_{20}^{(1)}C_k(2,0)+d_{21}^{(1)}C_k(2,1)+d_{22}^{(1)}x_k(n_1-2,n_2-2) \qquad (4.35)$$

As a consequence, (4.32) becomes

$$g(n_1,n_2)=\sum_{k=1}^{7}(\psi_{k11}+\psi_{k12}+\psi_{k13})2^{-k}-(\psi_{011}+\psi_{012}+\psi_{013}) \qquad (4.36)$$

This equation represents the output of the first stage of the hardware channel shown in Fig. 4-6. Similarly, one may develop the second, third and fourth stages of Fig. 4-6 by means of the functions

$$e(n_1,n_2)=\sum_{k=1}^{7}\psi_{k2}2^{-k}-\psi_{02}$$

$$f(n_1,n_2)=\sum_{k=1}^{7}\psi_{k3}2^{-k}-\psi_{03}$$

$$y(n_1,n_2)=\sum_{k=1}^{7}\psi_{k4}2^{-k}-\psi_{04} \qquad (4.37)$$

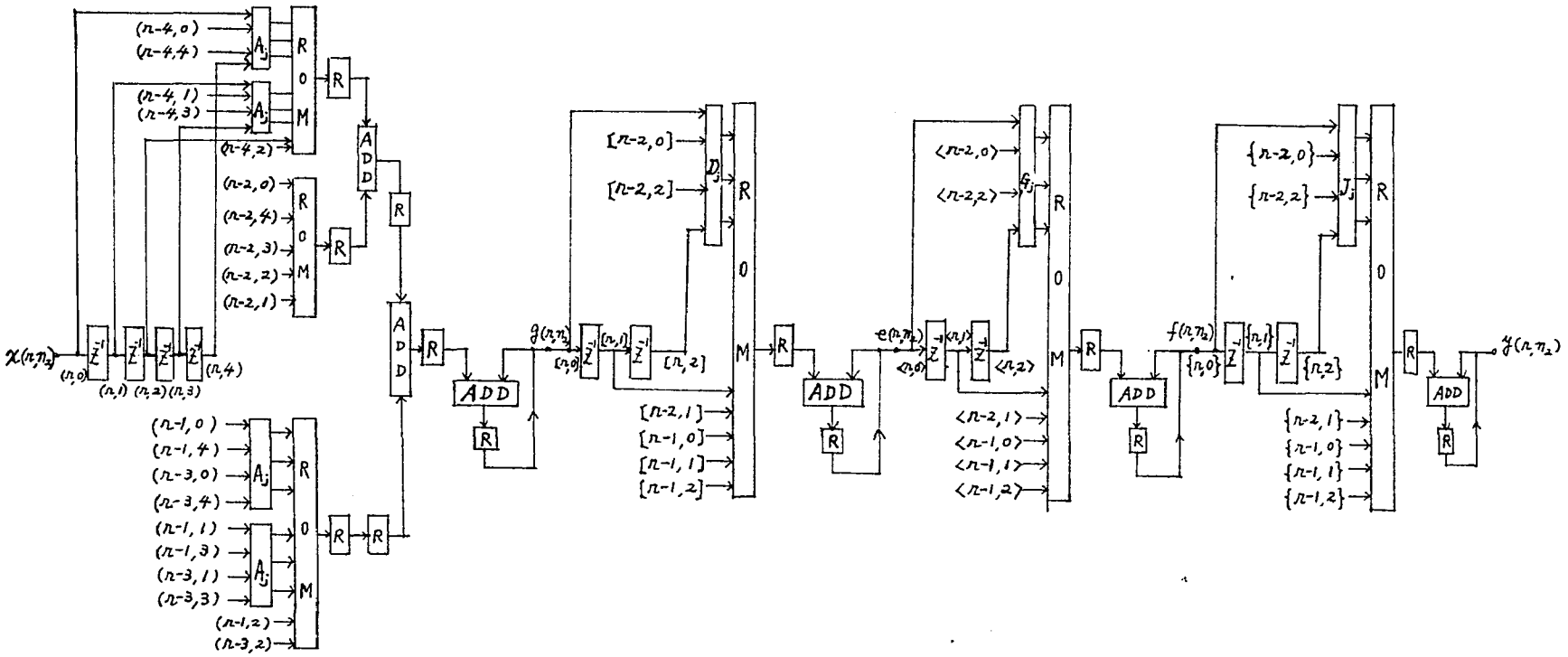


Fig. 4-6 Hardware realization of a 11 x 11 2-D linear phase FIR filter (only r-th channel shown)

The ψ functions are further expressed in terms of auxiliary code converter functions D, E, F ... L.

$$\psi_{k2}=d_{00}^{(2)}D_k(0,0)+d_{01}^{(2)}E_k(0,1)+d_{10}^{(2)}F_k(1,0)+d_{11}^{(2)}g_k(n_1-1,n_2-1)$$

$$\psi_{k3}=d_{00}^{(3)}G_k(0,0)+d_{01}^{(3)}H_k(0,1)+d_{10}^{(3)}I_k(1,0)+d_{11}^{(3)}e_k(n_1-1,n_2-1)$$

$$\psi_{k4}=d_{00}^{(4)}J_k(0,0)+d_{01}^{(4)}K_k(0,1)+d_{10}^{(4)}L_k(1,0)+d_{11}^{(4)}f_k(n_1-1,n_2-1) \qquad (4.38)$$

These equations represent the second, third and fourth stage of the channel shown in Fig. 4-6. Actually, only the r-th spatial channel is shown in the figure, the complete realization form can be drawn as described in Chapter 3. For each single channel there are six ROMs which correspond to a bit capacity of $2^3x(2^8+2^8+2^5+2^8+2^8+2^8)\approx 5 \times 2^{11}\approx 10^4$ bits. There are seven code converters which convert four addressing lines to three addressing lines, six eight-bit adders, ten shift registers and thirteen registers. The operational bit rate depends on t_ψ, the access time of the ROMs, t_d, the delay time of the code converters and t_a, the adding time of adders. If a register is provided before each adder, simultaneous operation of ROM access and adder's addition is possible. For a standard TTL ROM (or PROM), $t_a\approx 40ns$, $t_\psi\approx 50ns$ and $t_d\approx 15ns$. Therefore, the 2-D filter shown in Fig. 4-6 can operate at a bit rate of approximately $1/(t_\psi+t_dC)\approx 15MH_z$. Assuming an 8-bit word, a word rate equal

to (bit rate)/8=15MH$_z$/8=1.875MH$_z$ is possible. The operation diagram of Fig. 4-6 can be prepared in a similar manner as discussed in Fig. 4-3, except in the present case no feedback loops exist. Notice that the adders, provided after the sub-ROMs, will not affect the word rate, but will slow down the throughput of the filter. Notice also that nodes in Fig. 4-6 are labeled (a,b), <a,b>, [a,b] or {a,b}, where "a" refers to the spatial channel and "b" to the time variable. Actual connections between nodes have often been omitted for clarity, but two nodes which are in fact connected together have the same bracket and entries a, b. The size of the complete realization, particularly the number of channels, depends on the order of the matrix of input samples $x(n_1,n_2)$, which has been discussed in Chapter 3.

4.5 Hardware Saving Realization of Rectangular Filters.

The rectangular filters, discussed in Chapter 2, are separable filters and can be designed by using 1-D filters. The required 1-D filters can be either in FIR form or IIR form, and the system function can be either in cascade form or parallel form or both. Therefore, the rectangular filter computes the data either recursively or non-recursively.

The realization procedure of rectangular filters is similar to that of IIR filters and will not be repeated here. However, an example will now be given of the hardware realization of a rectangular filter to illustrate the relatively simple hardware realization involved.

4.5.1 Example of a 2-D Rectangular Filter Realization.

Consider a 2-D rectangular filter which has the system function

$$H(z_1,z_2)=H_1(z_1,z_2)\cdot H_2(z_1,z_2)$$
$$=\left(\sum_{i=0}^{10} h_1(i)z_1^{-i}\right)\cdot\left(\sum_{j=0}^{10} h_2(j)z_2^{-j}\right) \qquad (4.39)$$

The above equation is a 2-D separable system function, and can be rewritten in terms of the input function $X(z_1,z_2)$ and the output function $Y(z_1,z_2)$

$$Y(z_1,z_2)=H_1(z_1)\cdot H_2(z_2)\cdot X(z_1,z_2)$$
$$=A(z_1,z_2)\cdot H_2(z_2) \qquad (4.40)$$

Where $A(z_1,z_2)=H_1(z_1)\cdot X(z_1,z_2)$

the corresponding difference equations become

$$a(n_1,n_2)=\sum_{i=0}^{10} h_1(i)x(n_1-i,n_2)$$

$$y(n_1,n_2)=\sum_{j=0}^{10} h_2(j)a(n_1,n_2-j) \qquad (4.41)$$

Assume that $\{x(n_1,n_2)\}$, $\{h_1(n_1)\}$ and $\{h_2(n_2)\}$ are all dynamically limited to the range ± 1, a situation which can be obtained by scaling the data before processing. Applying the 2's complement form to the signals and assuming a word length B=8, (4.41) becomes:

$$a(n_1,n_2)=\sum_{i=1}^{7}\left\{\sum_{j=0}^{10} h_1(j)x_i(n_1-j,n_2)\right\}2^{-i}-\sum_{j=0}^{10} h_1(j)x_0(n_1-j,n_2)$$

$$y(n_1,n_2)=\sum_{i=1}^{7}\left\{\sum_{k=0}^{10} h_2(k)a_i(n_1,n_2-k)\right\}2^{-i}-\sum_{k=0}^{10} h_2(k)a_0(n_1,n_2-k)$$

(4.42)

From (4.42) it is found that there are 11 terms in the first bracket of $a(n_1,n_2)$ which corresponds to 11 address lines for the auxiliary function ψ_i. Hence a total bit capacity of $8 \times 2^{11}=2^{14}$ bits is required, which exceeds the previously assumed available high-speed ROM bit capacity, namely 8×2^{10} bits. Hence, partition of ψ_i into two parts is required, and various partitions are possible. In this

example ψ_{i1} and ψ_{i2} are chosen to be the sub-auxiliary ROM functions of $a(n_1,n_2)$, and correspond to a ROM with 8×2^8 and 8×2^3 bits respectively. Similarly, ψ_{i3} and ψ_{i4} are chosen as the sub-auxiliary functions of $y(n_1,n_2)$, and have the corresponding ROM bit capacity of 8×2^8 and 8×2^3 bits respectively. As a result

$$a(n_1,n_2)=\sum_{i=1}^{7}(\psi_{i1}+\psi_{i2})2^{-i}-(\psi_{01}+\psi_{02})$$

$$y(n_1,n_2)=\sum_{i=1}^{7}(\psi_{i3}+\psi_{i4})2^{-i}-(\psi_{03}+\psi_{04})$$

where

$$\psi_{i1}=\sum_{j=0}^{7}h_1(j)x_i(n_1-j,n_2)\ ;\ \psi_{i2}=\sum_{j=8}^{10}h_1(j)x_i(n_1-j,n_2)$$

$$\psi_{i3}=\sum_{k=0}^{7}h_2(k)a_i(n_1,n_2-k)\ ;\ \psi_{i4}=\sum_{k=8}^{10}h_2(k)a_i(n_1,n_2-k) \qquad (4.43)$$

It is clear from both (4.43) and Fig. 4-7 that for each channel one needs 10 shift registers, 8 registers, 4 ROMs and 4 adders; and furthermore, a bit capacity of $8 \cdot (2^8+2^8+2^3+2^3) = 4,224$ bits is required for the ROMs. As may be seen, the hardware realization is relatively simple and uncomplicated.

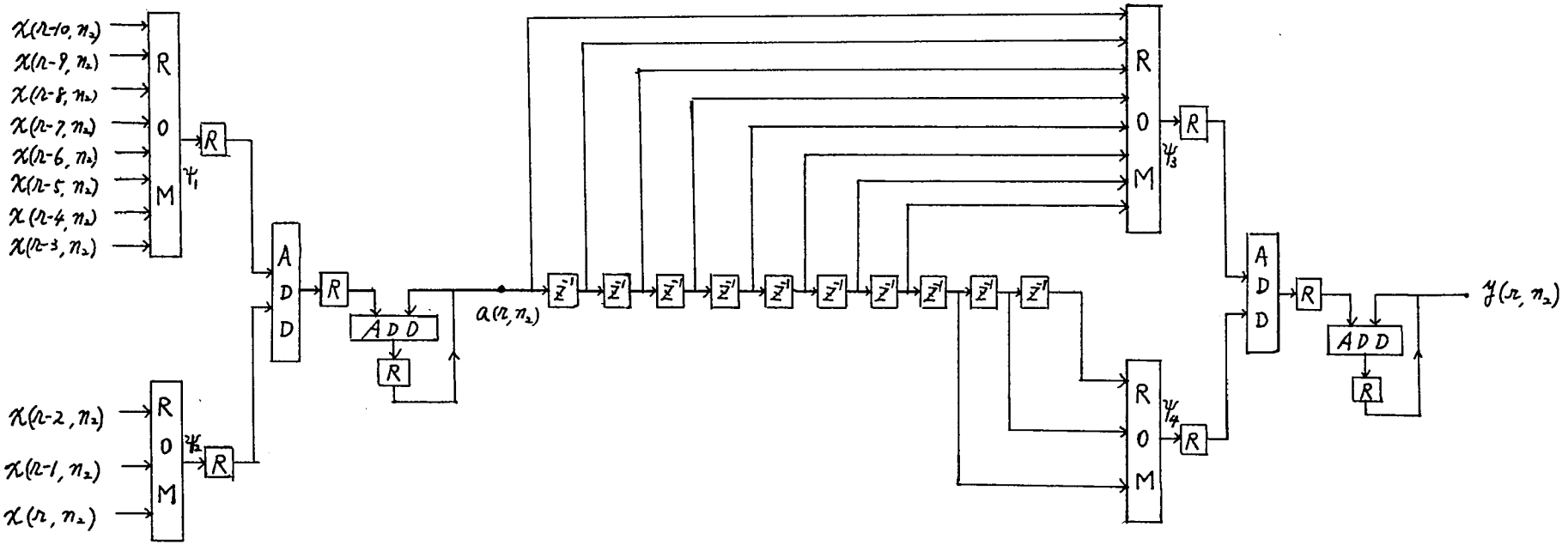


Fig. 4-7 Hardware realization of a 11 x 11 2-D rectangular filter

The operation bit rate is similar to that of the general form IIR filters discussed in Section 4.3.1, except that there is no feedback loop. If TTL IC's are used, the bit rate is about 20 MH_z. For an 8-bit word, this implies a word rate of 20 $MH_z/8=2.5MH_z$ per channel.

The complete realization form consisting of more channels like the one shown in Fig. 4-7 can be drawn by referring to Chapter 3.

In conclusion, the general order 2-D digital filter was realized by means of hardware saving techniques. Section 4.2 reviewed some 1-D processing techniques. In Section 4.3, a general form of IIR filter was demonstrated together with a discussion of its operation and timing diagram. Section 4.4 then discussed the realization of 2-D FIR filters with Section 4.4.2 dealing with factorization and Section 4.4.3 describing the McClellan transformation. Section 4.4.4 discussed the code converter and a realization example of a linear phase FIR filter was given in Section 4.4.5. Finally, a simple example of the realization of a separable rectangular FIR filter was given in Section 4.5.

As can be seen from Figures 4-2, 4-6 and 4-7, the hardware realization of 2-D digital filters appears to be highly symmetric and repetitive. This characteristic suggests that it would be possible to have the complete hardware components of one channel fabricated on a single IC chip. Consider as a simple example the 11 x 11 order cascaded rectangular filter shown in Fig. 4-7. Assuming TTL bipolar transistors are used, there are 4 ROMs (4224 bits) which require $2 \times 4224 \simeq 10^4$ transistors, 10 shift registers (8 bits each) which require $2 \times 8 \times 10 \simeq 2 \times 10^2$ transistors, 8 registers (8 bits each) which require $2 \times 8 \times 8 \simeq 2 \times 10^2$ transistors and four 8-bit adders which require 4×10^3 transistors. In addition to the above transistors some gating circuits, clock circuits and resistors are also required and are implemented by transistors. Therefore, a total of about 2×10^4 transistors is required for a single channel as shown in Fig. 4-7. This is in the realm of the LSI state of the art at present (1979), and so the hardware realization of 2-D filters becomes practical.

Appendix 4

Partitioning of ROMs for 2-D Linear Phase FIR Filters when N=2M+1.

The 2-D linear phase FIR filter obtained from the McClellan's transformation can be written as

$$H(z_1,z_2)=\sum_{n_1=0}^{N-1}\sum_{n_2=0}^{N-1} h(n_1,n_2)z_1^{-n_1}z_2^{-n_2} \tag{4A.1}$$

and the input-output relation is

$$Y(z_1,z_2)=H(z_1,z_2)\cdot X(z_1,z_2)$$

or

$$y(n_1,n_2)=\sum_{k_1=0}^{N-1}\sum_{k_2=0}^{N-1} h(k_1,k_2)\cdot x(n_1-k_1,n_2-k_2) \tag{4A.2}$$

Due to the symmetry characteristics of $h(k_1,k_2)$ that is,

$$h_r(n_1,n_2)=h_r(N-1-n_1,n_2)=h_r(n_1,N-1-n_2)=h_r(N-1-n_1,n_2) \tag{4A.3}$$

(4A.2) becomes

$$y(n_1,n_2)=\sum_{k_1=0}^{M-1}\sum_{k_2=0}^{M-1}h(k_1,k_2)\{x(n_1-k_1,n_2-k_2)+x(n_1-2M+k_1,n_2-k_2)+$$

$$x(n_1-k_1,n_2-2M+k_2)+x(n_1-2M+k_1,n_2-2M+k_2)\}+$$

$$\sum_{k_1=0}^{M-1}h(k_1,M)\{x(n_1-k_1,n_2-M)+x(n_1-2M+k_1,n_2-M)\}+$$

$$\sum_{k_2=0}^{M-1}h(M,k_2)\{x(n_1-M,n_2-k_2)+x(n_1-M,n_2-2M+k_2)\}+$$

$$h(M,M)x(n_1-M,n_2-M) \qquad (4A.4)$$

By a procedure similar to that discussed in Section 4.4.4, one obtains

$$y(n_1,n_2)=\sum_{k_1=0}^{M-1}\sum_{k_2=0}^{M-1}h(k_1,k_2)\{\sum_{j=1}^{B-1}A_j(k_1,k_2)2^{-j}-A_0(k_1,k_2)\}+$$

$$\sum_{k_1=0}^{M-1}h(k_1,M)\{\sum_{j=1}^{B-1}B_j(K_1,M)2^{-j}-B_0(k_1,M)\}+$$

$$\sum_{k_2=0}^{M-1}h(M,k_2)\{\sum_{j=1}^{B-1}C_j(M,k_2)2^{-j}-C_0(M,k_2)\}+$$

$$h(M,M)\{\sum_{j=1}^{B}x_j(n_1-M,n_2-M)2^{-j}-x_0(n_1-M,n_2-M)\} \qquad (4A.5)$$

where

$$B_j(k_1,M)=x_j(n_1-k_1,n_2-M)+x_j(n_1-2M+k_1,n_2-M)$$

$$C_j(M,k_2)=x_j(n_1-M,n_2-k_2)+x_j(n_1-M,n_2-2M+k_2)$$

and $B_j(k_1,M)$, $C_j(M,k_2)$ can assume three distinct states, either 0, 1 or 2; a code converter might be used here, however, since only one extra state can be saved (from 4 states to three) no such scheme was undertaken.

Equation (4A.5) can then be written in terms of the auxiliary function

$$y(n_1,n_2)=\sum_{j=1}^{B-1}\psi_j 2^{-j} - \psi_0$$

where

$$\psi_j=\sum_{k_1=0}^{M-1}\sum_{k_2=0}^{M-1}h(k_1,k_2)A_j(k_1,k_2)+\sum_{k_1=0}^{M-1}h(k_1,M)B_j(k_1,M)+$$

$$\sum_{k_2=0}^{M-1}h(M,k_2)C_j(M,k_2)+h(M,M)x_j(n_1-M,n_2-M)$$

and an arrangement realizing ψ_j is shown in Fig. 4-8.

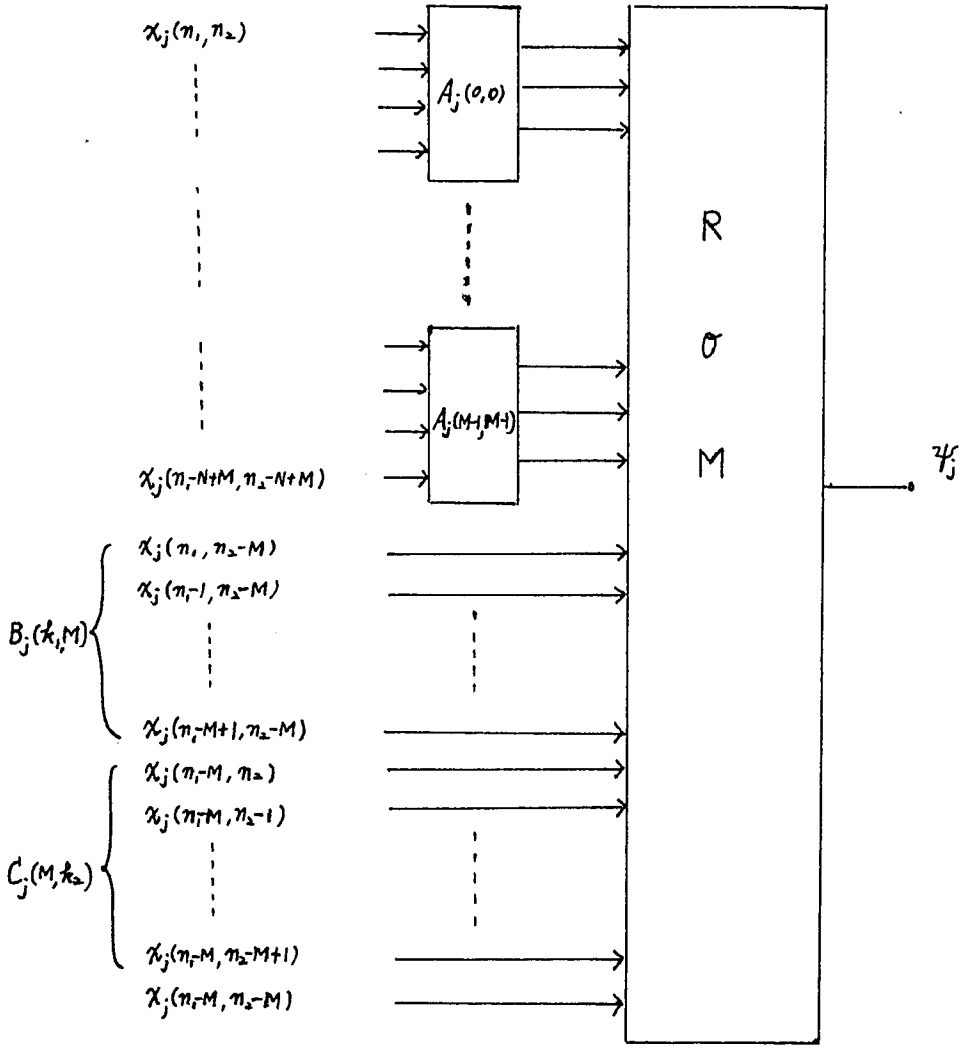


Fig. 4-8 A third example for realizing the auxiliary function ψ_j.

CHAPTER FIVE

High Speed 2-D Digital Filtering and some Possible Applications.

5.1 Introduction

The hardware structures discussed in Chapters 3 and 4 have been shown to be real-time processors for 2-D data. The data processing speed of such structures is much higher than that of single-input single-output software processing methods [28, 34]. However, the hardware needed for the real-time processor is more complicated and more expensive to implement; moreover, its complexity depends on the size of the input 2-D data.

For a small 2-D data array, for instance seismic data which usually uses about 20 rows, the real-time processor is still applicable. For a large 2-D data array, for instance image data which usually has more than 120 rows, the required structure becomes too complicated and expensive to implement.

A compromise "overlap" method is therefore proposed which uses a finite but smaller number of hardware channels, depending on the required processing speed of the 2-D data. This will be discussed in Section 5.2. A microprocessor-based realization of 2-D low-order digital filters will

be discussed in section 5.3. Possible applications of hardware processing techniques will then be given in Section 5.4.

5.2 High Speed Section by Section Processing of 2-D Data.

Consider a 2-D input array $i(n_1,n_2)$ of dimension 250 x 250 and a 11 x 11 array 2-D linear phase FIR filter, which can be factored into three factors as discussed in Chapter 4, that is

$$H(z_1,z_2)=H_1(z_1,z_2)\cdot H_2(z_1,z_2)\cdot H_3(z_1, z_2)$$

$$=\frac{O(z_1,z_2)}{I_1(z_1,z_2)}\cdot\frac{I_1(z_1,z_2)}{I_2(z_1,z_2)}\cdot\frac{I_2(z_1,z_2)}{I(z_1,z_2)}$$

where

$$\frac{O(z_1,z_2)}{I_1(z_1,z_2)}=H_1(z_1,z_2)=\sum_{i=0}^{4}\sum_{j=0}^{4} h^{(1)}z_1^{-i}z_2^{-j} \text{ with } N_1^{(1)}=N_2^{(1)}=5$$

$$\frac{I_1(z_1,z_2)}{I_2(z_1,z_2)}=H_2(z_1,z_2)=\sum_{i=0}^{2}\sum_{j=0}^{2} h^{(2)}z_1^{-i}z_2^{-j} \text{ with } N_1^{(2)}=N_2^{(2)}=3$$

$$\frac{I_2(z_1,z_2)}{I(z_1,z_2)}=H_3(z_1,z_2)=\sum_{i=0}^{4}\sum_{j=0}^{4} h^{(3)}z_1^{-i}z_2^{-j} \text{ with } N_1^{(3)}=N_2^{(3)}=5$$

Because $H(z_1,z_2)$ has the order $N_1=N_1^{(1)}+N_1^{(2)}+N_1^{(3)}-2=11=N_2$, therefore, if $i(n_1,n_2)$ is directly convolved with $h(n_1,n_2)$, either by general purpose computer computation or by 250 channels of hardware processor, the output data $o(n_1,n_2)$ will be a 260 x 260 array, as shown in Fig. 5-1.

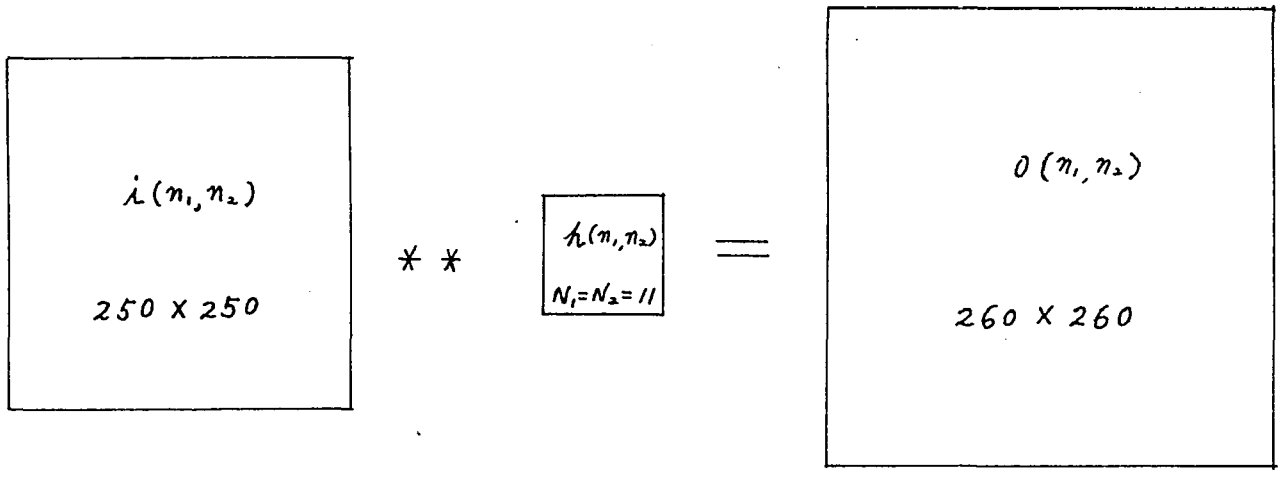


Fig. 5-1 Block diagram of a 2-D discrete convolution

Now, suppose we are using 50 channels of hardware (1/5 of the total number of channels), then we can first process the first 50 rows of $i(n_1,n_2)$ and generate an enlarged 60 rows of the output $o(n_1,n_2)$. However, the last 10 rows of the output must be thrown away, because their formation did not depend on the presence of the 51st to the 60th rows of the input. This is true for all the sections that will be processed, except the last section. Thus to form the output rows 51 to 90, one must take rows 41 to 90 of the previous input and similarly, to form output rows 91 to 130 one must take input rows 81 to 130 as illustrated in Fig. 5-2, which clearly shows the overlap of ten rows, occurring at the input. Altogether one needs 7 output sections to process the 250 x 250 input data $i(n_1,n_2)$. Note, the final output section requires the 241st row up to the 250th row of the input data $i(n_1,n_2)$, and generates the 251st row up to the 260th row of the output data $o(n_1,n_2)$.

In general, if the 2-D input data array is MxN, and the array size of the filter's impulse response is N_1xN_2, then the number of rows in the overlapped section, L_{op}, must be equal to $N_1 - 1$. Assuming $1/T_p$ is the operation speed (or word rate) of the complete parallel structure, the operational speed of the section-by-section method, with L_s sections used, is

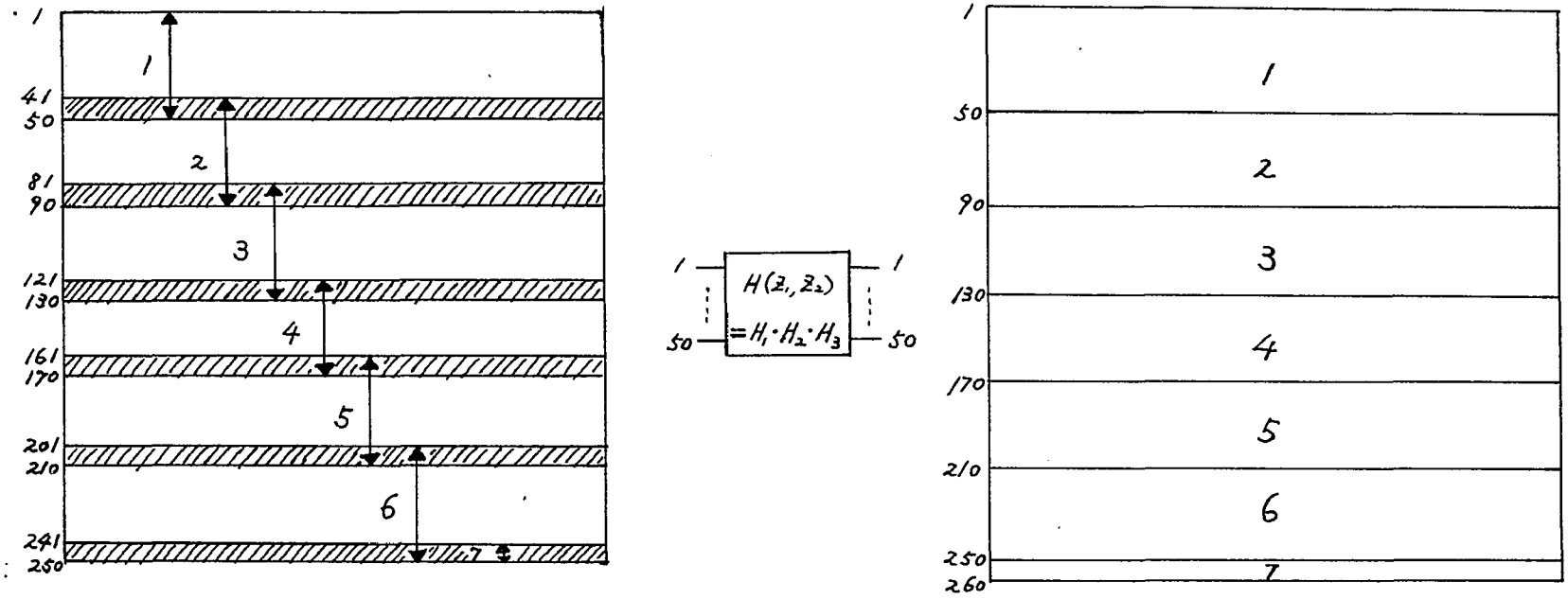


Fig. 5-2 High speed, section by section with overlap, processing of 2-D signals

$$\frac{1}{T_p \times L_s} \tag{5.1}$$

The number of hardware channels required in the section-by-section method is

$$N_s \geq \left[\frac{M - (N_1-1)}{L_s} + (N_1-1)\right] \tag{5.2}$$

where the square bracket represents "the smallest integar of."

In the example discussed previously where 250 x 250 input data was processed by 11 x 11 filters, the minimum number of hardware channels required is therefore $\frac{250-(11-1)}{7} + (11-1) = 45$. This implies that by using $\frac{45}{250} = 18\%$ of the complete hardware channels, the operation speed is slowed down to $\frac{1}{7} = 14.3\%$ of that of the complete parallel structure.

A further alternative software realization of low order 2-D filters is possible by using commercially available microprocessors and this will now be discussed.

Careful examination of the low order filter realizations, as discussed in Chapter 3, indicates that each channel of the realization involves a limited number of multipliers. For instance, the first order IIR filter, shown in Fig. 3-5, needs 6 multipliers per channel. Therefore, each channel

can be replaced by a program controlled hardware or microprocessor-based structure, as will now be explained.

5.3 Microprocessor-based Realization of 2-D Low-order Filters.

From equation (3.7) of Chapter 3, it is clear that a 2-D digital filtering process is an operation of "sum of products." In other words (3.7) can be considered to be of the form

$$y = \sum_{i=1}^{N} a_i x_i \qquad (5.3)$$

Therefore, an integrated circuit which performs the above operation at high speed, known as an arithmetic logic unit (ALU), will be adopted for realization. A block diagram of a commercially available ALU [40] is shown in Fig. 5-3. This ALU chip operates on two numbers in two's complement form. The multiplication is done through internal shift and add. The sum of products is made possible by inserting a holding register between the accumulator and the A multiplexer. The holding register stores the past sum of products which is then added with the current product to produce the current sum of products.

A general purpose microprocessing system can now be employed to realize a 2-D low-order filter. Fig. 5-4 shows the block diagram of the structure. The ALU performs the

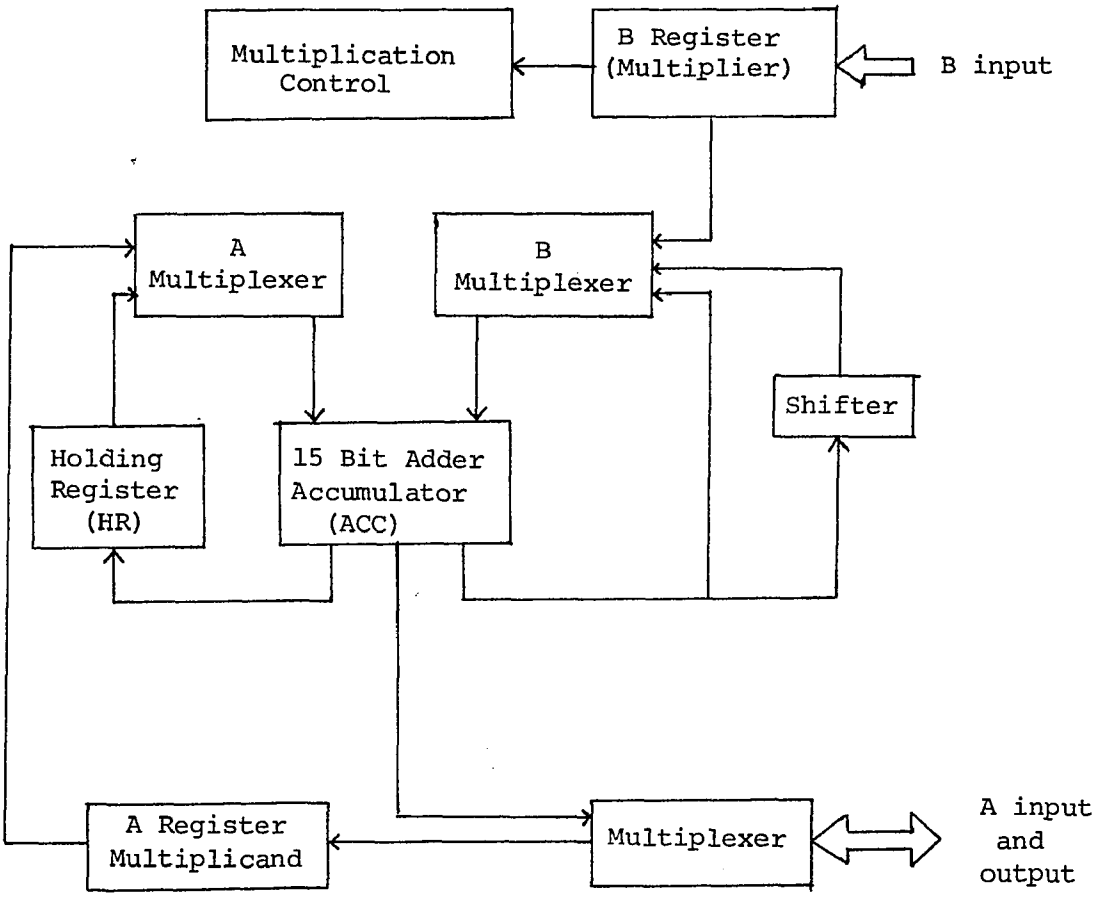


Figure 5.3 MOS/LSI Arithmetic Logic Unit Organization

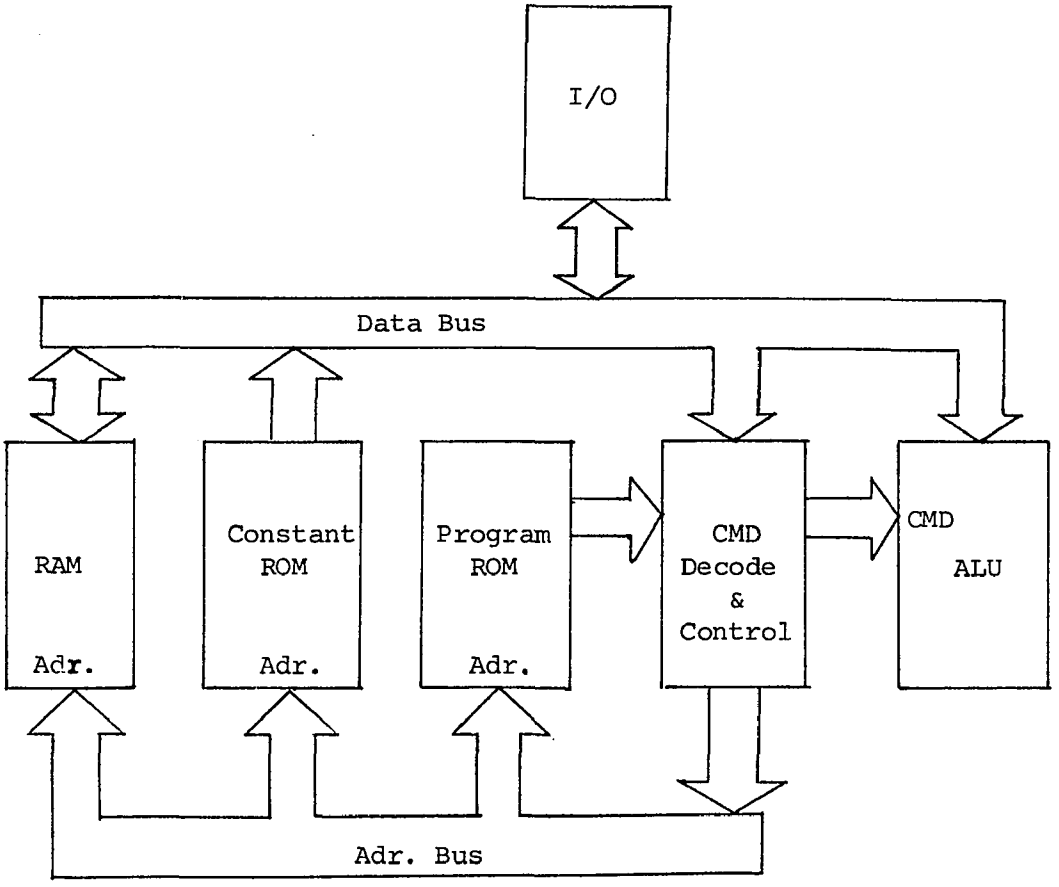


Fig. 5-4 Block Diagram of Microprocessor System

mathematical computation of sum of products under the control of CMD-decoder. The program ROM controls the data flow and the ALU operation. The constant ROM supplies the filter coefficients and other constants. The RAM is used for temporary data storage and shifting. The I/O port is used for input and output data storage.

A program which implements a first order 2-D IIR filter will now be described in a flow chart form.

Consider (3.7) with $p=q=1=A_{00}$, that is

$$y(n_1,n_2)=x(n_1,n_2)+A_{10}x(n_1-1,n_2)+A_{01}x(n_1,n_2-1)+A_{11}x(n_1-1,n_2-1)-B_{10}y(n_1-1,n_2)-B_{01}y(n_1,n_2-1)-B_{11}y(n_1-1,n_2-1) \tag{5.4}$$

To realize the above equation, six multiplications and six additions are required. The six filter coefficients, A_{01}, A_{10}, A_{11}, B_{01}, B_{10} and B_{11} affect the multiplications and therefore are stored in the constant ROM. The values of $x(n_1,n_2-1)$, $x(n_1-1,n_2-1)$, $y(n_1,n_2-1)$ and $y(n_1-1,n_2-1)$ are stored in the RAM, and are obtained by shifting $x(n_1,n_2)$, $x(n_1-1,n_2)$, $y(n_1,n_2)$ and $y(n_1-1,n_2)$. The new values of $x(n_1,n_2)$, $x(n_1-1,n_2)$ and $y(n_1-1,n_2)$ are inputted through the I/O port and then stored in the RAM. The flow chart, which corresponds to the command instructions stored in

program ROM, is given in Fig. 5-5. The required RAM map and constant ROM map are also listed for reference. The instructions used in the flow chart are described below.

READ: The ALU transfers the data stored in a specified RAM or ROM location into the accumulator.

WRITE: The ALU transfers the data in the accumulator into a specified RAM location.

INPUT: The ALU transfers the data from the I/O port into the accumulator.

OUTPUT: The ALU transfers the data from the accumulator into I/O port.

SHIFT: The ALU transfers the data in a specified RAM location into the accumulator, and then transfers it to another specified RAM location.

The operational speed of the microprocessor-based realization depends on the speed of the ALU and also on the total number of instructions stored in the program ROM. For higher order filter realizations, more filter coefficients are required and therefore more multiplications and additions are needed. As more instructions are needed the operational speed becomes slower. Hence, the microprocessor-based

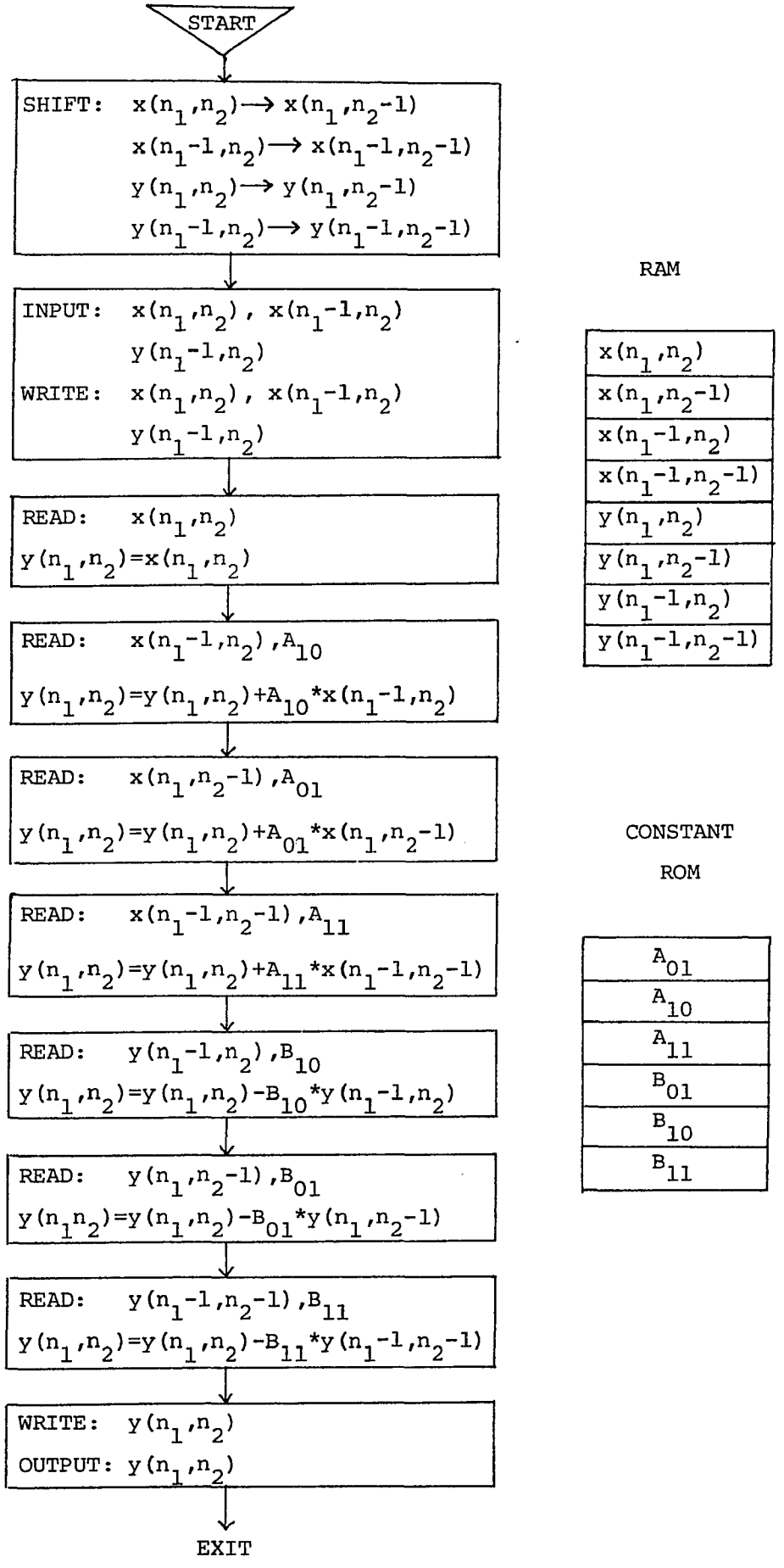


Fig. 5-5 Flow Chart of Microprocessor-based Single Channel 2-D Filter Realization

structure can only be used for low-order filter realizations or non-real time filtering processes. Such a software method forms an alternative to the realization of low-order 2-D filters and is mentioned here for completeness.

5.4 Possible Applications of Parallel Processing Techniques

5.4.1 Seismic Data Processing

The direct and simplest application of the parallel processing technique is to seismic data, which is most commonly obtained through oil exploration or by geophysicists recording earthquakes. The procedure usually involves the spatial placing of sensors underground (each sensor corresponds to an array row or channel n_1) and their recorded values in time (corresponding to time variable n_2). The fan filtering technique, which is widely used, allows one to pass mechanical vibrational velocities which fall within the fan-shaped passband region in the frequency (f_1,f_2) domain as shown in Fig. 5-6, while rejecting velocities which fall in the stop-band. This filter therefore separates slow velocities from fast velocities or vice versa. There are several approximation methods to design a fan filter [35]. One method is the McClellan transformation method [27] which converts a 1-D linear phase FIR filter into a

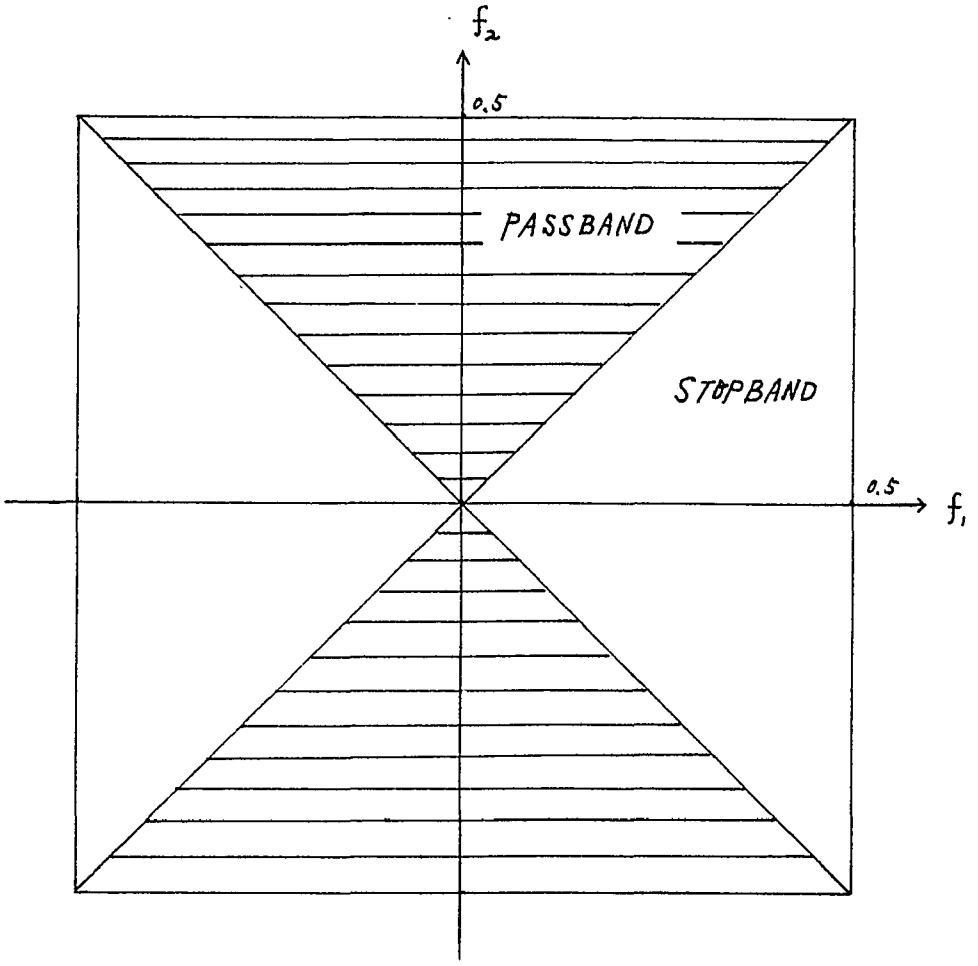


Fig. 5-6 Frequency characteristic of a fan filter

2-D linear phase FIR filter. Hence a 2-D fan filter with system function $H(z_1,z_2)$ can be obtained and realized by hardware as discussed in Chapters 3 and 4. Since processing of seismic data always uses a small number of input traces (about 20 input channels), therefore, the parallel processing structures described in Chapters 3 and 4 are well able to process seismic data in real time.

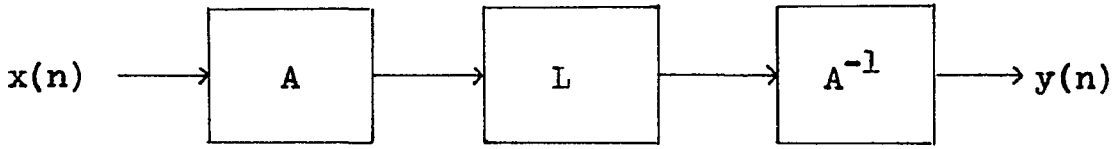
5.4.2 Digital Image Processing

The techniques used in digital image processing mainly deal with two subjects: image coding and image restoration or enhancement [39]. Only the latter topic is of interest here and the parallel structure of 2-D digital filters can be used with advantage either to restore or to enhance an image as will be seen below.

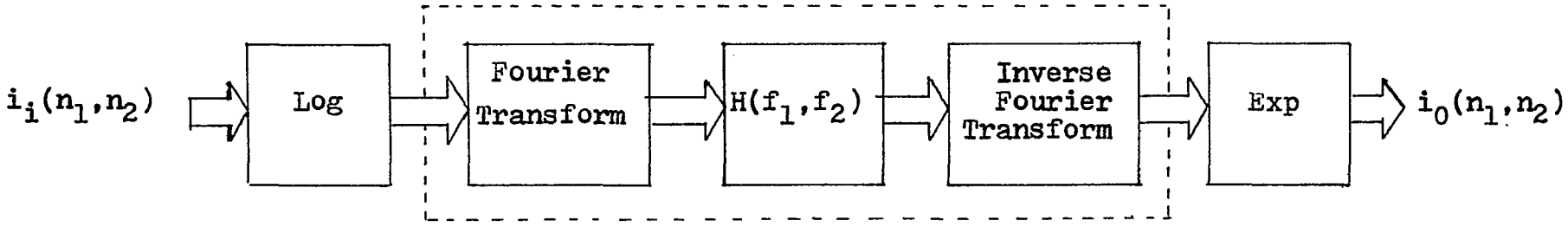
Consider the simplest application of a 2-D digital filter used in image processing which consists of passing the image through either a lowpass or highpass filter [29]. The lowpass filtered image retains most of the image information but lacks the sharp detail where the original image changes intensity sharply. The highpass filtered image preserves sharp constraints, but loses the information within

the constant intensity regions. Both the lowpass and highpass filters are designed in terms of the 2-D Z-transform, therefore they can be hardware realized in parallel form which implies real time processing. By using the section-by-section technique, hardware cost is reduced and high speed processing is still applicable.

As a second example of the use of a 2-D hardware processor, the field of homomorphic filtering on images [38] may be cited. Homomorphic filtering is nonlinear filtering which applies to a class of systems that obey a generalized superposition principle. Characteristic of this processing scheme is, for example, the cascade operation of 3 system blocks as shown in Fig. 5-7a, where blocks A and A^{-1} are mutually inverse non-linear processors such as logarithm and exponential, and block L represents a linear time invariant filter. A block diagram of an image processor based on this principle is shown in Fig. 5-7b. Note that the central block represents a 2-D digital filter which is linear and time invariant. A software realization of this scheme in 2-D has proved successful [38]. Clearly this scheme may also be implemented by using a hardware method.



(a)



(b)

Fig. 5-7 Block diagram for a 2-D homomorphic image processor

5.4.3 A Possible Application of a 2-D Digital Communication System.

One-dimensional digital filters have been widely used in communication systems. One practical example is given by Freeny, et al, in their Time Division Multiplex (TDM) to Frequency Division Multiplex (FDM) translator, where a series of 12 TDM speech inputs (or a series of 12 audio signals with each of the 12 channels assigned to a different portion of the spectrum) were added together for FDM transmission [37]. The digital filters used include both IIR and FIR lowpass filters. The IIR filter was a ninth-order design that was realized with 8-bit coefficients; the FIR filter was a 20th order design and was realized with 10-bit filter coefficients.

In this section, the 2-D digital filters will be used for a possible TDM to FDM transmission.

Consider the modulation of 2-D signals. Assume that the 2-D data $i(n_1,n_2)$ has the array size MxN. The frequency spectrum of $i(n_1,n_2)$ is then given by a finite 2-D Fourier series

$$I(\omega_1,\omega_2)= \sum_{n_1=0,n_2=0}^{M \quad N} i(n_1,n_2)e^{-jn_1\omega_1} e^{-jn_2\omega_2} \tag{5.5}$$

Assume $I(\omega_1,\omega_2)$ is bandlimited in $0\leq\omega_1\leq\omega_{1m}$ and $0\leq\omega_2\leq\omega_{2m}$ where $\omega_{1m},\omega_{2m}\leq\pi$. The corresponding input signal $i(n_1,n_2)$ and frequency spectrum $I(\omega_1,\omega_2)$ are shown in Fig.5-8 where only a portion of $I(\omega_1,\omega_2)$ is displayed.

Consider now the modulating functions and let

$$\cos\omega_{10}n_1=\cos\omega_{10}t_1\Big|_{t_1=n_1T_1=n_1} \text{ and } \cos\omega_{20}n_2=\cos\omega_{20}t_2\Big|_{t_2=n_2T_2=n_2} \tag{5.6}$$

where ω_{10} and ω_{20} are the modulating frequencies normalized with respect to the sampling frequency. Assume further that the sampling frequency used in the modulating signal is the same as that used in the input signal $i(n_1,n_2)$. Now consider the 2-D modulation which is defined to be the multiplication of $i(n_1,n_2)$ with $\cos n_1\omega_{10}\cdot\cos n_2\omega_{20}$, i.e.

$$i_m(n_1,n_2)=i(n_1,n_2)\cdot\{\cos(n_1\omega_{10})\cos(n_2\omega_{20})\} \tag{5.7}$$

The frequency spectrum of $i_m(n_1,n_2)$ is then

$$I_m(\omega_1,\omega_2)=\sum_{n_1=0}^{M}\sum_{n_2=0}^{N} i(n_1,n_2)\cos(n_1\omega_{10})\cos(n_2\omega_{20})e^{-jn_1\omega_1}e^{-jn_2\omega_2}$$

$$=\sum_{n_1=0}^{M}\sum_{n_2=0}^{N} i(n_1,n_2)\cdot\left(\frac{e^{jn_1\omega_{10}}+e^{-jn_1\omega_{10}}}{2}\right)\cdot\left(\frac{e^{jn_2\omega_{20}}+e^{-jn_2\omega_{20}}}{2}\right)\cdot e^{-jn_1\omega_1}e^{-jn_2\omega_2}$$

$$=\tfrac{1}{4}\sum_{n_1=0}^{M}\sum_{n_2=0}^{N} i(n_1,n_2)\{e^{-jn_1(\omega_1-\omega_{10})}e^{-jn_2(\omega_2-\omega_{20})}+$$

$$e^{-jn_1(\omega_1-\omega_{10})}e^{-jn_2(\omega_2+\omega_{20})}+e^{-jn_1(\omega_1+\omega_{10})}e^{-jn_2(\omega_2-\omega_{20})}+$$

$$e^{-jn_1(\omega_1+\omega_{10})}e^{-jn_2(\omega_2+\omega_{20})}\}$$

$$=\tfrac{1}{4}\{I(\omega_1-\omega_{10},\omega_2-\omega_{20})+I(\omega_1-\omega_{10},\omega_2+\omega_{20})+I(\omega_1+\omega_{10},\omega_2-\omega_{20})+$$

$$I(\omega_1+\omega_{10},\omega_2+\omega_{20})\} \tag{5.8}$$

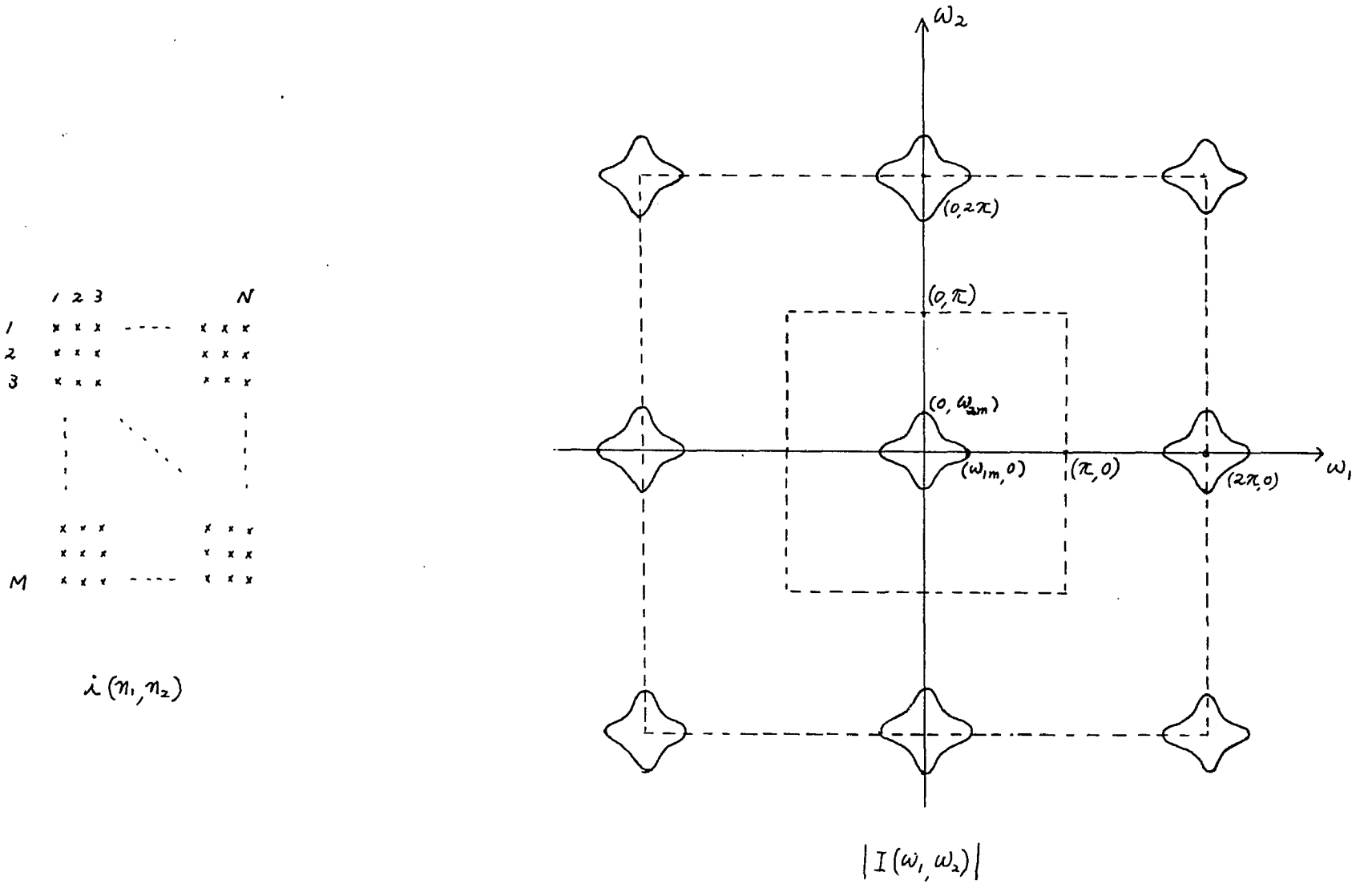


Fig. 5-8 2-D digital signal and its frequency spectrum

From (5.8) it is clear that after 2-D modulation, the frequency spectrum of $i(n_1,n_2)$ is shifted from the center point (0,0) (see Fig. 5-8) to new points, centered about $(\pm\omega_{10},\pm\omega_{20})$ as shown in Fig. 5-9a. If $\omega_{1m},\omega_{2m},\omega_{10},\omega_{20}$ are chosen such that $\omega_{1m}\leq\omega_{10},\omega_{2m}\leq\omega_{20}$, there will be no overlap of components. This is also shown in Fig. 5-9a.

Consider the frequency spectrum shown in Fig. 5-9b, where only one quarter of the original spectrum is shown. Let $i_q(n_1,n_2)$ represent the time domain function of Fig. 5-9b. If $i_q(n_1,n_2)$ is multiplied by $\cos n_1\omega_{10}\cdot\cos n_2\omega_{20}$, which is defined as the 2-D demodulation, the original signal $i(n_1,n_2)$ can be recovered by passing the resultant signal through a 2-D lowpass filter. The discussion of demodulation is similar to that of modulation and will not be repeated here.

From the above discussion it is clear that a quarter-band signal can be used to represent a 2-D signal just as the single sideband signal can be used to recover the original signal in a 1-D system. The generation of quarter-band signals can be directly obtained by filtering the modulated signal $i_m(n_1,n_2)$, as shown in Fig. 5-9a, with a 2-D rectangular {bandpass,bandpass} filter as shown by dashed lines in Fig. 5-9a.

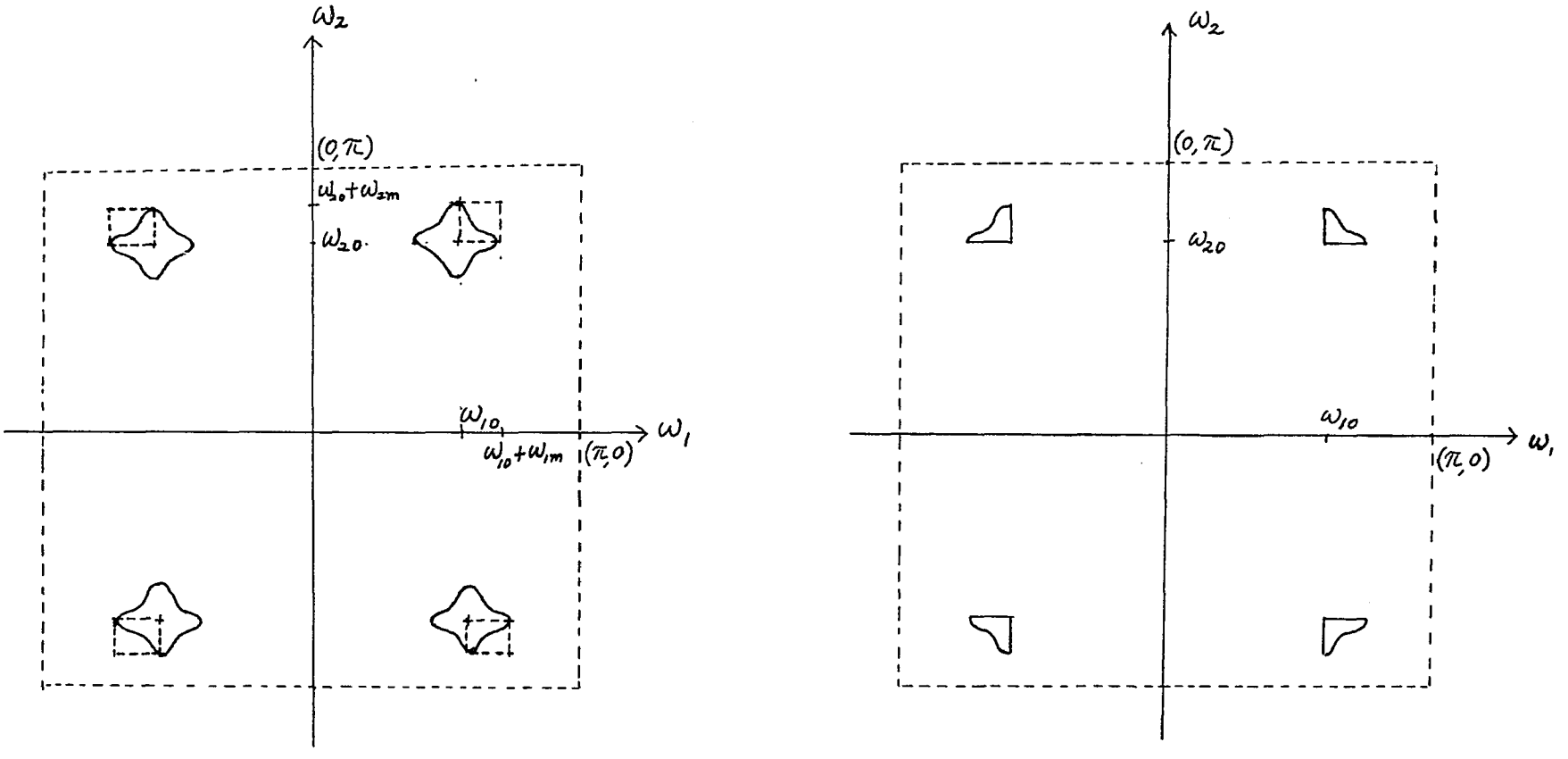


Fig. 5-9 A relocated 2-D frequency spectrum and its quarter band spectrum

Another approach to generate the quarter-band signal follows from the method used by Weaver, where a single side band of a 1-D signal was generated [37]. The Weaver method splits the 1-D baseband signal into two paths, and various steps of modulation and filtering are performed in each path such that when two paths are rejoined together, the unwanted sideband cancels out. This procedure is shown in Fig. 5-10.

In the 2-D signal case refer to Fig. 5-11. The baseband 2-D input signal must be split into four paths, with each path 2-D modulated in the time domain. The modulating frequencies are chosen following Weaver to be equal to one half of the related maximum frequencies ω_{1m}, ω_{2m}. A 2-D rectangular {lowpass, lowpass} filter is then applied in each path and followed by another 2-D modulation with modulation frequencies set at $\omega_{1c} - \frac{\omega_{1m}}{2}$ and $\omega_{2c} - \frac{\omega_{2m}}{2}$, where ω_{1c}, ω_{2c} are the carrier frequencies along the ω_1, ω_2 axes respectively. Finally the four paths are added together and a quarter-band 2-D signal is obtained. The double line shown in this figure implies that the data is parallel processed. The frequency spectra corresponding to each node, labeled as (a), (b), ... (j), in Fig. 5-11 are shown in Fig. 5-12 with the shaded areas representing negative magnitudes. The operation principle for quarter-band 2-D

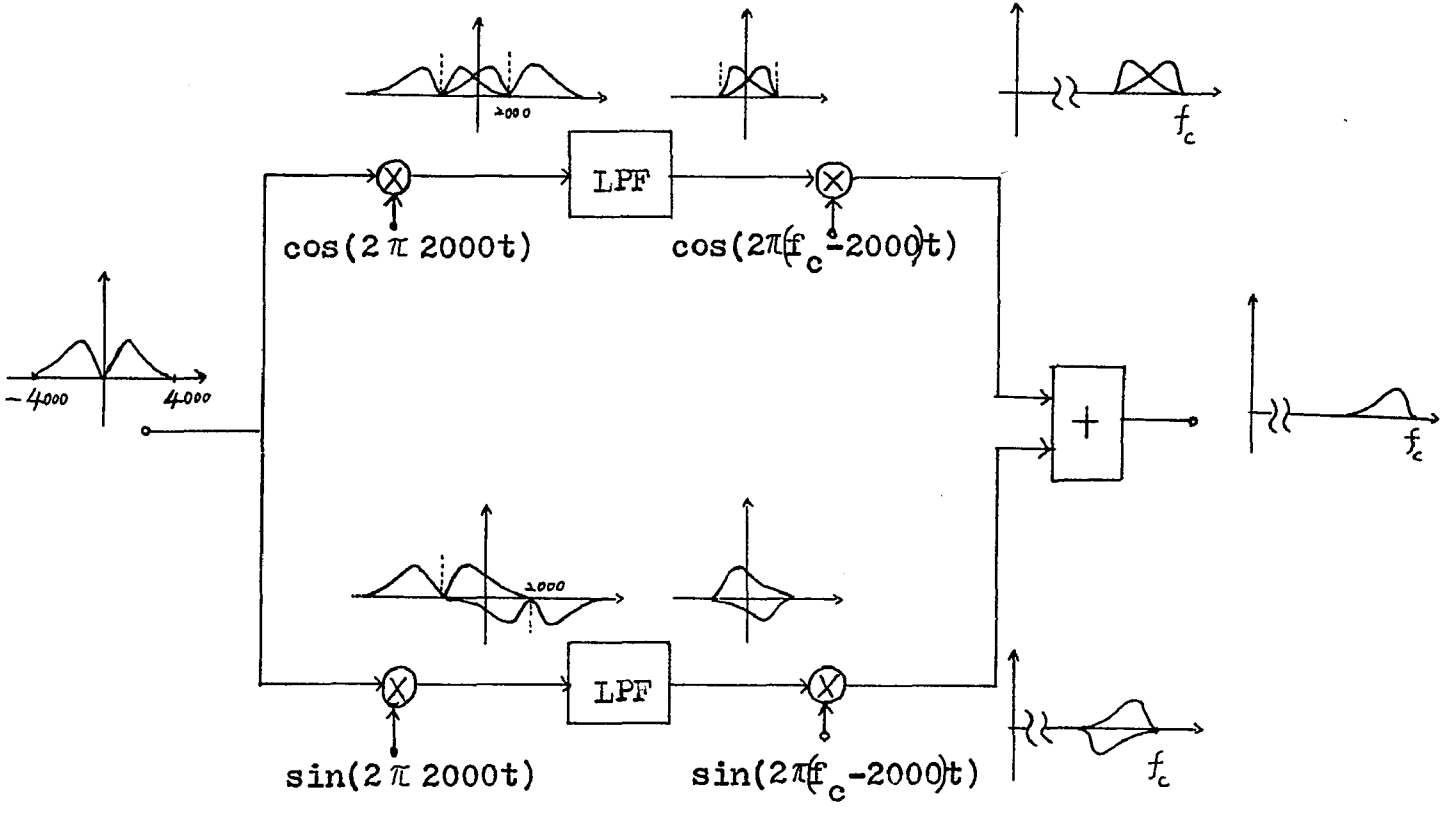


Fig. 5-10 The 1-D Weaver system

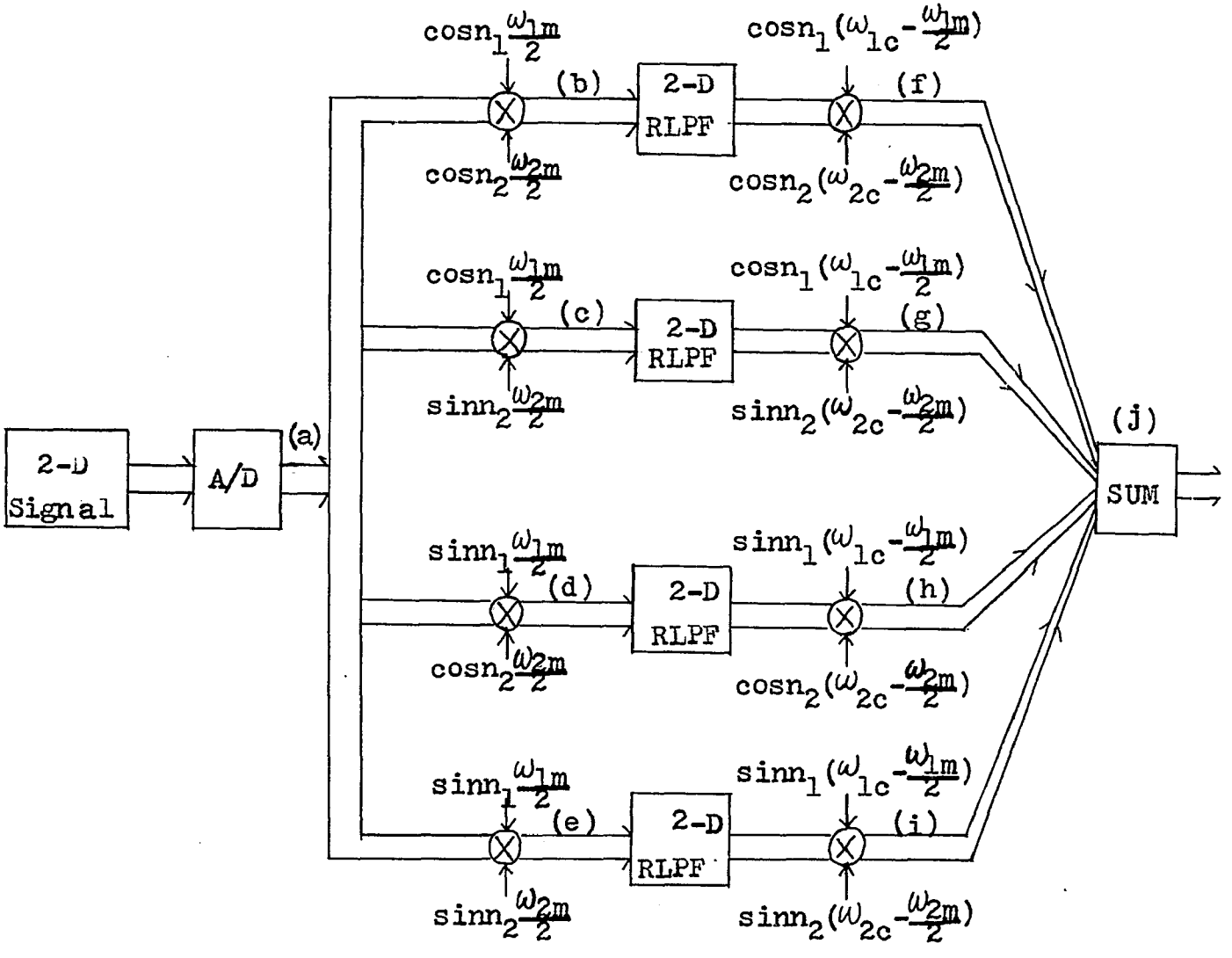


Fig. 5-11 Generalized 2-D Weaver system

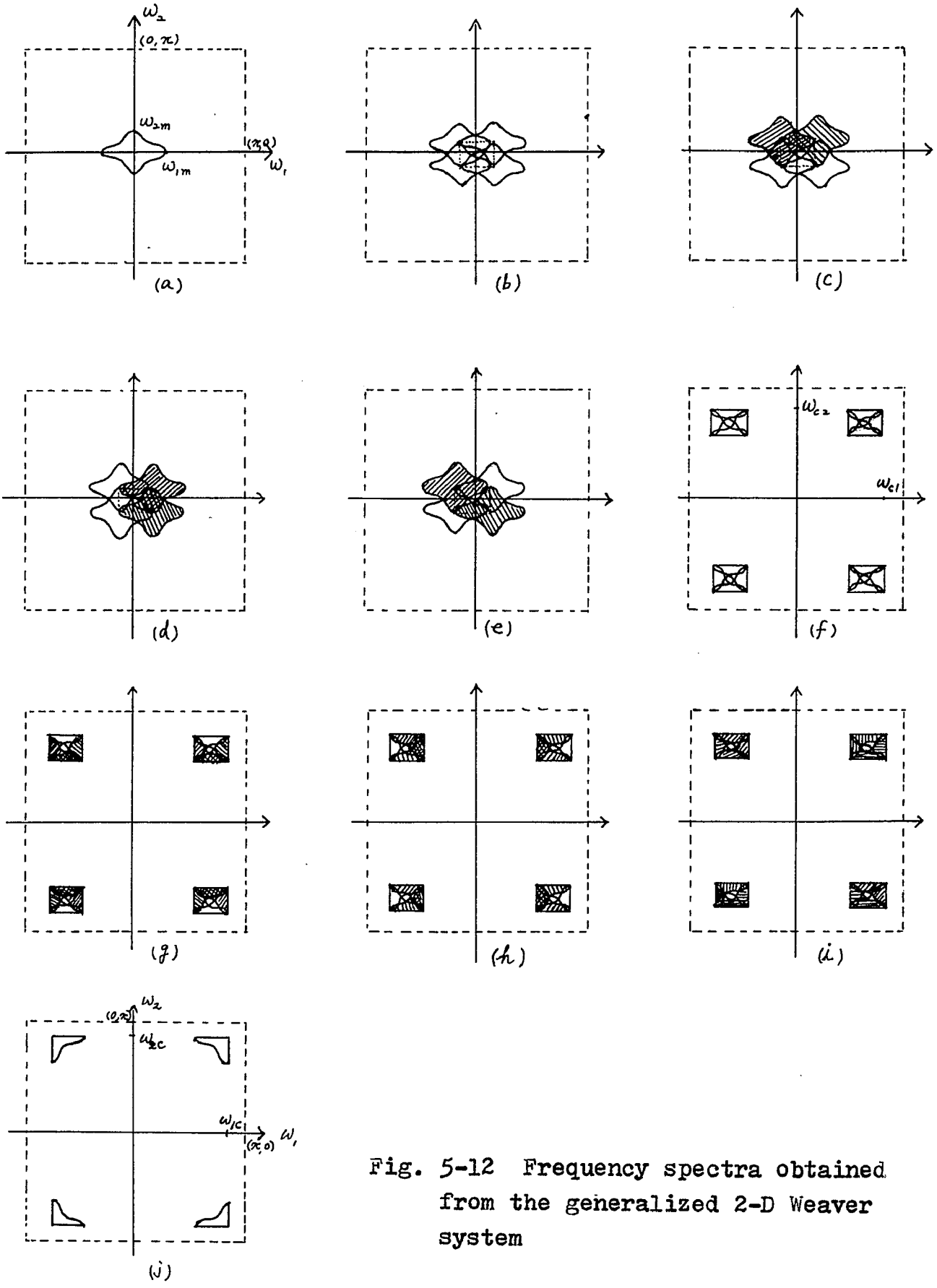


Fig. 5-12 Frequency spectra obtained from the generalized 2-D Weaver system

signal generation is the same as that of single sideband signals discussed by Weaver and will be illustrated next. Consider the first path of Fig. 5-11. The 2-D sampled data was first multiplied by $\cos(n_1\frac{\omega_{1m}}{2})$ and $\cos(n_2\frac{\omega_{2m}}{2})$; this step will translate the original spectrum as shown in Fig. 5-12(a) into 4 parts centered around 4 new points $(\pm\omega_{10},\pm\omega_{20})$ as shown in Fig. 5-12(b). A 2-D rectangular lowpass filter is then used and only the frequency components within the small rectangular dotted box of Fig. 5-12(b) remain. Then a second 2-D modulation is applied which shifts the filtered spectrum into 4 parts around 4 new points $(\pm\omega_{1c} - \omega_{10};\ \pm\omega_{2c} - \omega_{20})$ shown in Fig. 5-12(f). The same operation but different modulation signals were used in the other three paths, and finally the four paths are added together and a 2-D quarter-band signal is then obtained, as shown in Fig. 5-12(j). The remaining parts of Fig. 5-12 correspond to the three remaining paths and will not be further referred to.

Based on the above discussion, a simple block diagram of a 2-D digital Frequency Division Multiplex-Time Division Multiplex translator using quarter-band signals is shown in Figs. 5-13 and 5-14. Note, it is assumed that all the 2-D signals are band limited to $\omega_{1m}^{(i)}$, $\omega_{2m}^{(i)}$ with respect to the ω_1,ω_2 axes and the sampling frequencies ω_{1s}, ω_{2s} are much larger than $\omega_{1m}^{(i)}$, $\omega_{2m}^{(i)}$. Fig. 5-13 shows a possible

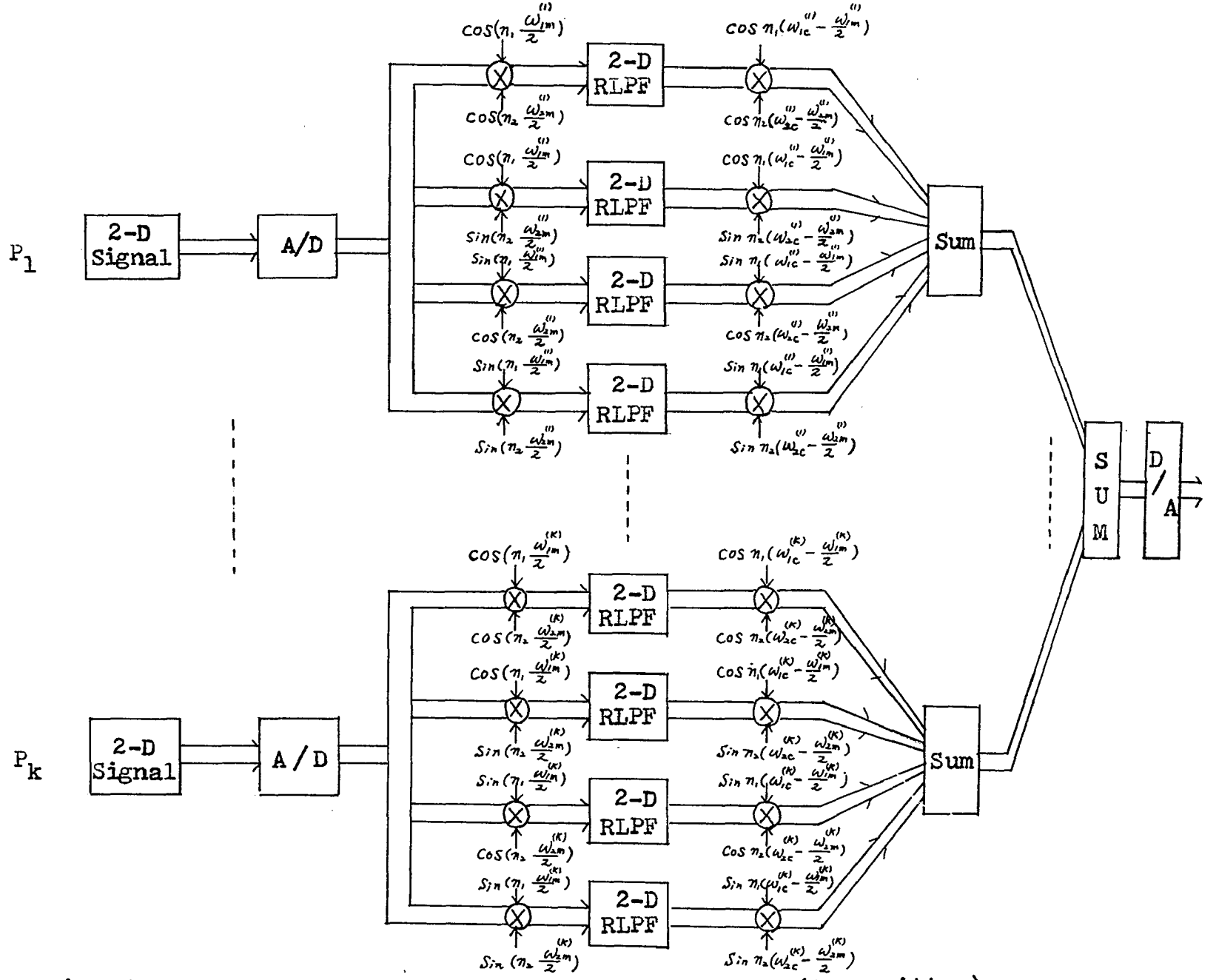


Fig. 5-13 An example of a 2-D communication system (transmitter)

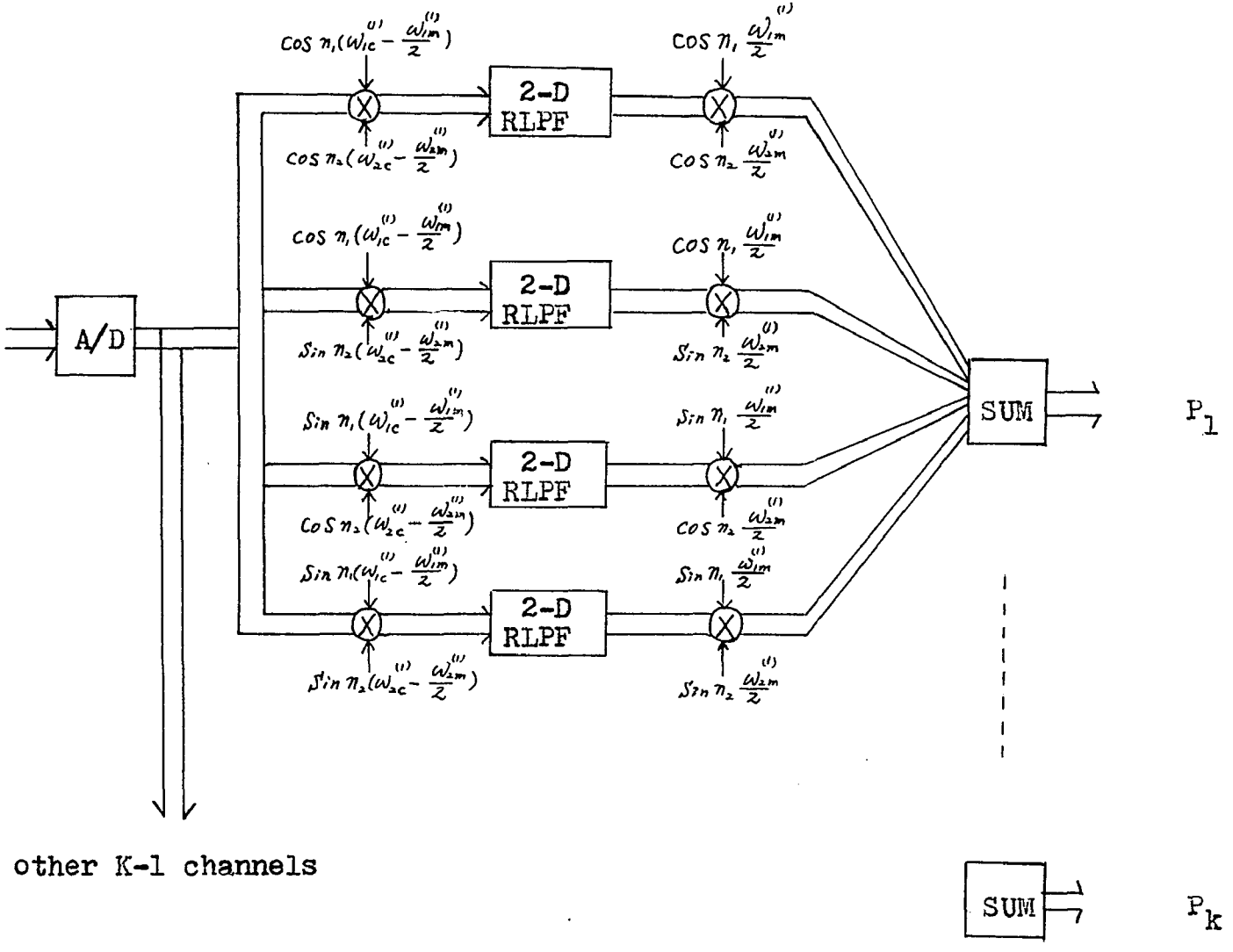


Fig. 5-14 An example of a 2-D communication system (receiver)

modulation system where a total of K, 2-D quarter-band signals are to be transmitted. Fig. 5-14 shows a possible demodulation system where modulated signals are filtered out to produce K, 2-D individual signals.

In conclusion, a high speed 2-D signal processing technique using the section-by-section (with some overlap) method was discussed. This procedure needs less hardware for 2-D filtering but has slower operation speed. It operated with a word rate equal to $\frac{1}{N}$ of the original word rate, where N is the number of sections used. Replacement of low order filter hardware by means of a program-controlled microprocessor based structure was next proposed. Finally, possible applications of 2-D digital filters which can process the 2-D signals, whether in real time or at high speed, were discussed. A possible 2-D Frequency Division Multiplex-Time Division Multiplex translator, which used quarter-band 2-D signals was also described.

CHAPTER SIX

Conclusions

6.1 Summary of Contributions and Results.

The main contributions of this research were:

1. The design of 2-D separable rectangular symmetric filters. The chief use of these filters is for special purpose spectrum shaping.
2. The hardware realization of 2-D parallel-processing digital filters which can process data in real time was proposed. Both the FIR and the IIR filters were described. Low order filter realization was made of unit delays, adders and multipliers. High order filter realization was made possible by using multiplier-ROM replacement and other hardware saving techniques.
3. Special high-speed 2-D digital filtering based on section-by-section data processing was presented. Microprocessor replacement of each channel was also discussed. Possible future applications to 2-D filtering were suggested.

A special class of 2-D separable rectangular symmetric filters was defined in Chapter 2. Several design methods were given and a variety of frequency responses was displayed. Three examples of filters which achieve special purpose frequency

responses were discussed and the relevant computer outputs and graphical representation of the system functions in the (f_1, f_2) domain were given.

Since this filter class is new, much further research remains to be done. The future research should include design methods of special frequency responses which minimize both the passband and stopband ripples, and also the control of transition bandwidth.

A parallel processing technique to hardware realize 2-D digital filters was proposed in Chapter 3. Only low order filters that could be realized with adders, multipliers and delays were discussed. Several realization forms for both FIR and IIR filters were shown. Input data rearrangement which is required for IIR filters was discussed and illustrated with examples.

The hardware low-order realization techniques of Chapter 3 were extended to general order filters in Chapter 4. The techniques used to save hardware cost include the replacement of all multipliers by ROMs and also the partition of ROMs into smaller sub-ROMs. A procedure to hardware realize a 2-D FIR filter and IIR filter was proposed and illustrated by examples. The operation speed and timing diagram of the proposed structures were also examined.

A cost saving high-speed section-by-section method was proposed in Chapter 5. This method used a few hardware channels, but is still much faster than the 1-D processing technique. Also proposed was a possible microprocessor replacement for each channel. This method was shown to be suitable for low-order filter realization. For high-order filters the operational speed of the microprocessor-based realization becomes slow, due to the increased instruction sets.

For future research, input and output devices must be designed to process 2-D signals column by column. The effect of finite word length on the factorization of the system function, as discussed in Section 4.2.3, is worthy of further research. Applications of 2-D parallel processing for seismic and image data were also discussed. The 1-D Weaver's single sideband signal generating technique was extended to the two dimensional case, where a 2-D quarter-band signal was obtained. More applications in these areas are expected in the near future.

REFERENCES

1. L. R. Rabiner and B. Gold, "Theory and Application of Digital Processing", Prentice Hall Inc., Englewood Cliffs, N. J., 1975.
2. A. V. Oppenheim and R. W. Schafer, "Digital Signal Processing", Prentice Hall Inc., Englewood Cliffs, N. J., 1975.
3. R. W. Hamming, "Digital Filters", Prentice Hall Inc., Englewood Cliffs, N. J., 1977.
4. A Peled and B. Liu, "Digital Signal Processing", John Wiley & Sons, Inc., N. Y., 1976.
5. J. W. Cooley and J. W. Tukey, "An Algorithm for The Machine Computation of Complex Fourier Series", Math. Comp. Vol. 19, April 1965, pp. 297-301.
6. A. B. Glaser and G. E. Subak-Sharpe, "Integrated Circuit Engineering", Addison Wesley Inc., Reading, Mass. 1977.
7. J. F. Kaiser, "Digital Filters", Chapter 7, in "System Analysis by Digital Computer", (F. F. Kuo and J. F. Kaiser, editors), John Wiley and Sons Inc., N. Y. 1966.
8. R. W. Hamming, ibid, Chapter 9.
9. B. Gold and C. M. Rader, "Digital Processing of Signals", McGraw-Hill Book Co., N. Y. 1969.
10. L. R. Rabiner and B. Gold, ibid, Chapter 3.

11. J. W. Parks and J. H. McClellan, "A Program for The Design of Linear Phase, Finite Impulse Response, Digital Filters", IEEE Trans. on Audio and Electroacoustics, AU-20, No. 3, Aug. 1972, pp. 195-199.

12. O. Herrmann, "Design of Non-recursive Digital Filters with Linear Phase", Electronics letters, Vol. 6, No. 11, 1970, pp. 328-329.

13. O. Herrmann and H. W. Schuessler, "Design of Non-recursive Digital Filters with Minimum Phase", Electronic letters, Vol. 6, No. 11, 1970, pp. 329-330.

14. E. Hofstetter, A. Oppenheim and J. Siegel, "A New Technique for The Design of Non-recursive Digital Filters", Proc. fifth annual Princeton conferency on infermation sciences and systems, 1971, pp. 64-72.

15. L. R. Rabiner and B. Gold, ibid, Chapter 3.

16. L. Weinberg, "Network Analysis and Synthesis", McGraw-Hill Book Co., N. Y., 1962.

17. J. F. Kaiser, ibid.

18. L. R. Rabiner and B. Gold, ibid, Section 4.6.

19. C. S. Burrns and T. W. Parks, "Time Domain Design of Recursive Digital Filters", IEEE Trans. Audio, Vol. 18, pp. 137-141.

20. J. L. Shanks, "Recursion Filters for Digital Processing" Geophysics, Vol. 32, Feb. 1967, pp. 33-51.

21. F. Brophy and A. C. Sabazar, "Recursive Digital Filter Synthesis In The Time Domain", IEEE Trans. on Acoustics Speech and Signal Processing, Vol. ASSP 22, No. 1, Feb. 1974, pp. 45-55.

22. A. G. Deczky, "Synthesis of Recursive Digital Filters Using The Minimum P-error Criterion," IEEE Trans. on Audio and Electroacoustics, AU 20, No. 4, Oct. 1972, pp.257-263.

23. L. R. Rabiner and B. Gold, ibid, Chapter 5.

24. J. L. Shanks, S. Treitel and J. H. Justice, "Stability and Synthesis of Two-Dimensional Recursive Filters", IEEE Trans. on Audio and Electroacoustics, AU 20, No. 2, June 1972, pp. 115-128.

25. T. S. Huang, "Stability of Two-Dimensional Recursive Filters", IEEE Trans. on Audio and Electroacoustics, AU 20, No. 2, June 1972, pp. 158-163.

26. T. S. Huang, "Two-Dimensional Windows", IEEE Trans. on Audio and Electroacoustics, AU 20, No. 1, March 1972, pp. 88-89.

27. J. H. McClellan, "On The Design of One-Dimensional and Two-Dimensional FIR Filters", Ph.D. thesis, Rice Institute, Houston, Texas, April 1973.

28. R. M. Mersereau and D. E. Dudgeon, 'Two-Dimensional Digital Filtering". Proc. IEEE, Vol. 63, pp. 610-623, April 1975.

29. L. R. Rabiner and B. Gold, ibid, Chapter 7.

30. A. Croisier, D. J. Esteban, M. E. Levilion and V. riso, "Digital Filter for PCM Encoded Signals", U. S. Patent 3777130, Dec. 4,1973.

31. A. Peled and B. Liu, "A New Hardware Realization of Ditital Filters", IEEE Trans. on Acoustic, Speech and Signal Processing, Vol. ASSP-22, pp. 456-4-2, 1974.

32. Bu-Chin Wang, "Comments on : A New Hardware Realization of Digital Filters", IEEE Trans. on Acoustic, Speech and Signal Processing, Aug. 1977, Vol. Assp-25.

33. A. Peled and B. Lir, "A New Approach to the Realization of Nonrecursive Digital Filters". IEEE Trans. on Audio Electroacoustic, Vol. AU-21, pp. 477-484, Dec. 1973.

34. S. K. Mitra, A. D. Sagar and N. A. Pendergrass, "Realization of 2-D Recursive Digital Filters", IEEE Trans. on Circuits and Systems, Vol. CAS-22, No. 3, March 1975, pp. 177-184.

35. Treitel S. Shanks, J., and Frasier, C. W., "Some Aspects of Fan Filtering", Geophysics, Vol. 32, pp. 789-800.

36. L. B. Jackson, J. F. Kaiser and H. S. McDonald, "An Approach to The Implementation of Digital Filters". IEEE Trans., Audio and Electroacoustics, Vol. AU-16, pp. 413-421, Sep. 1968.

37. S. L. Freeny, R. B Kieburtz, K. V. Mina and S. K. Tewksbury, "Design of Digital Filters for An All Digit Frequency Division Multiplex - Time Division Multiplex Translator". IEEE Trans. on Circuit Theory, Vol. CT-18, No. 6, Nov.1971, pp. 702-711.

38. A. V. Oppenheim, R. W. Schafer and T. G. Stockham, Jr., "Nonlinear Filtering of Multiplied and Convolved Signals." Proc. IEEE Vol. 56, pp. 1264-1291, Aug. 1968.

39. Harry C. Andrews, "Digital Image Processing", IEEE Computer, May 1974, pp. 17-19.

40. "MOS/LSI Arithmetic Logic Unit." Hycom, Inc., 16841 Armstrong Ave., Irvine, California