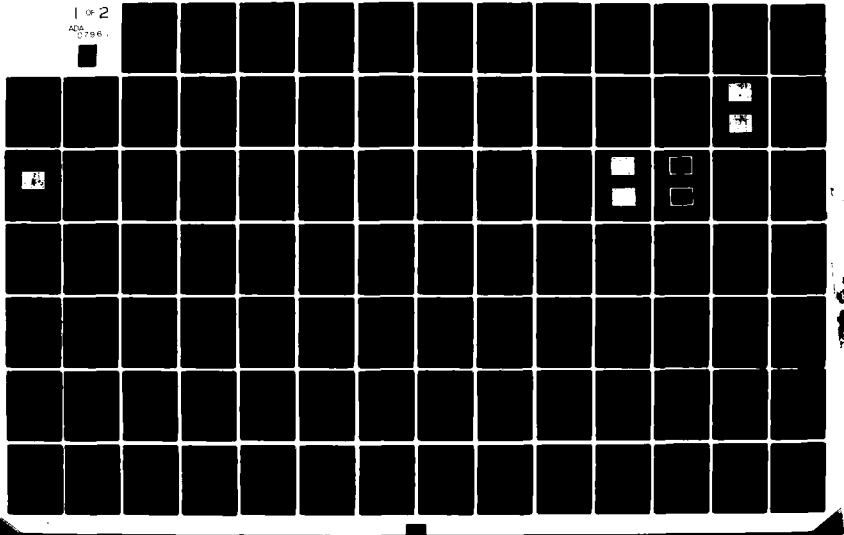
AD-A079 611 CALIFORNIA UNIV BERKELEY DEPT OF ELECTRICAL ENGINEER--ETC F/G 17/2
CONTINUATION OF RESEARCH IN THE APPLICATION OF ANALOG SAMPLED D--ETC(U)
JUN 78 R W BRODERSEN, P R GRAY N00173-78-C-0230

UNCLASSIFIED

1 OF 2
AD A079611



R & D FINAL STATUS REPORT.

CONTINUATION OF RESEARCH IN THE APPLICATION OF ANALOG SAMPLED DATA SIGNAL PROCESSING TO LOW-COST SPEECH BANDWIDTH COMPRESSION.

NRL Contract N00173-78-C-0230

Final status rpt. : Jul 77-30 Jun 78,

R. W. Brodersen and P. R. Gray

Department of Electrical Engineering and Computer Sciences
and the Electronics Research Laboratory
University of California, Berkeley, California 94720

This final report covers the research on low cost implementation of analysis and synthesis of speech using linear predictive coding (LPC). The report is in two completely separate parts which were originally the reports for Masters of Science projects by Paul Hurst (Part I) and Eugene Evancoe (Part II).

Part I describes an approach to adaptive autocorrelation analysis which could be implemented using analog sampled data techniques for multiplication and filtering.

Part II is an investigation of the use of high order switched-capacitor ladder filters for use as an LPC synthesis filter.

80 1 16 014

TABLE OF CONTENTS

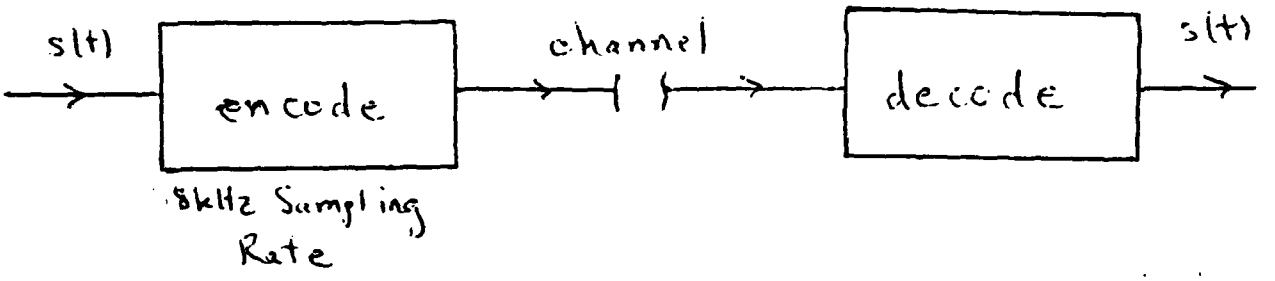
1. Introduction

Low bit-rate transmission of speech has been an important speech research goal. Many systems have been realized through the use of high speed digital computers. Only recently has the application of MOS-LSI made possible sophisticated single I.C. speech processors [1]. At this time, the most visible attempt at implementation of digital transmission rate reduction are the "codec" (coder-decoder) chips. Now available from a number of companies, the codecs use a logarithmic transfer curve to achieve a digital transmission rate of 64kbits/sec. This is a large bandwidth compared to the bandwidth of speech (4kHz). Therefore, it is desirable to find a means of reducing the transmission rate even further.

The most attractive speech bandwidth compression schemes suffer from the need of fairly complex digital processing which requires expensive and complex hardware. In this project, a possible solution is explored which combines analog and digital circuitry into a system which can operate in real time and be implemented on a few MOS-LSI chips. An approach which seems promising in this respect is an analysis-synthesis system based on linear prediction (fig. 1).* Before discussing the system, a short review of linear prediction will be presented.

* A similar approach is applied to the pitch detection problem by Dalrymple [2].

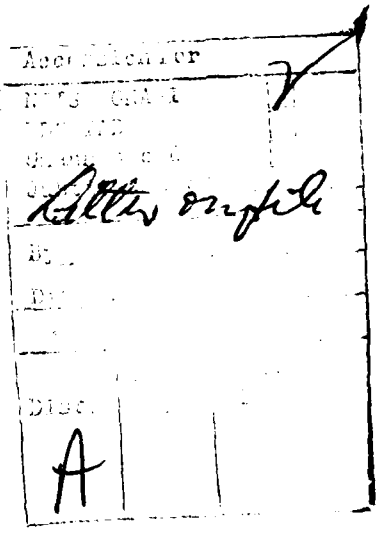
Comparison



8kHz Sampling Rate

ENCODING	TRANSMISSION BIT RATE
12 bit linear code	96 kbits/sec
8 bit log code (codec)	64 kbits/sec
Linear Predictive Coding	2.4 kbits/sec

Figure 1



2. Linear Prediction with Infinite Impulse Response Windows

Linear prediction is applied to speech to extract the minimum information to describe the vocal tract frequency response.* This information is transmitted digitally at a very low rate to a receiver where the speech is reconstructed by a compatible speech synthesizer. Such schemes are capable of transmitting reasonable quality speech at rates as low as 2.4kbits/second.

Linear predictive coding (L.P.C.) of speech is based on a very simple model of the human vocal tract (fig. 2 & 3). Sounds are produced by the following process:

[1] Air is forced out of the lungs.

[2] As the air passes through the glottis (vocal cords), it is either allowed to flow through unrestricted (unvoiced sounds) or it is transformed into short bursts of air by the periodic opening and closing of the glottis (voiced sounds). The unrestricted flow of air is modelled as white noise. The short bursts of air are modelled as impulses.

[3] The air then passes through the vocal tract. The shape of the vocal tract and the position of the lips determine the resonant frequencies associated with the sound. (In some cases, the air also passes through the nasal passage.)

It is the resonances of the vocal tract that we attempt to model using linear prediction. The shape of the vocal tract varies as

* An excellent introduction to linear predictive coding is given by Makhoul [3].

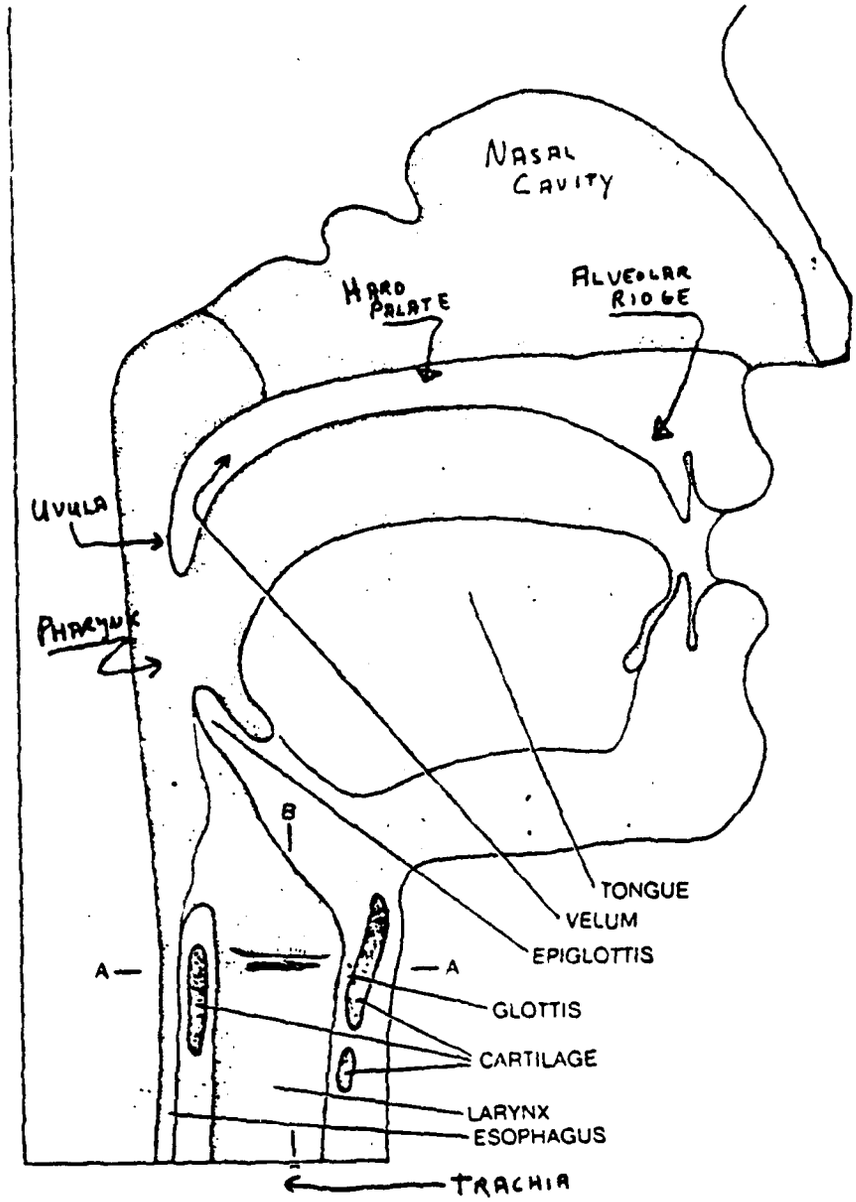


Figure 2 Cross Section of the Vocal Tract

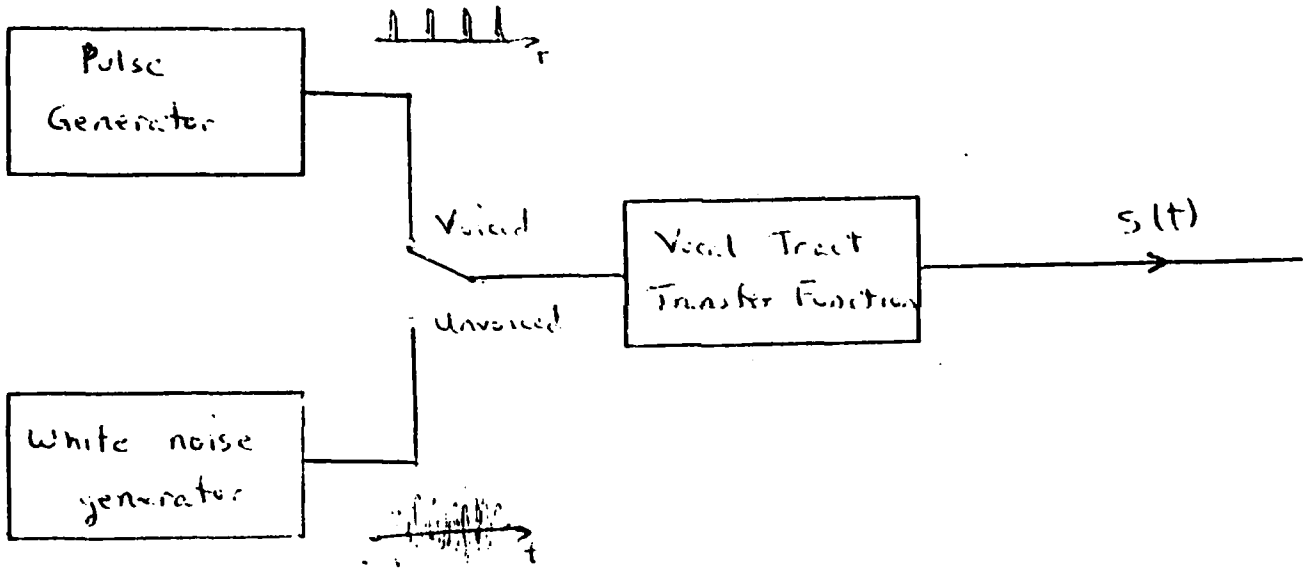


Figure 3. A simple speech production model

different sounds are produced, but it varies slowly (every 10 to 20 milliseconds). So linear prediction can be applied to segments of speech of this length.

L.P.C. is used to model the vocal tract resonances by fitting an all-pole transfer function to the vocal tract transfer function (fig. 4). The model has the form (in z-transform notation):

$$\frac{S_o(z)}{S_i(z)} = H(z) = \frac{1}{1 - \sum_{i=1}^{p} a_i z^{-i}} \qquad p = \text{number of poles in the model.}$$

The model parameters are the a_i's. The number of poles in the model can be determined by the following: The resonances of the vocal tract are generally separated by about 1kHz. Each resonance requires a pair of complex conjugate poles in the model. The glottal waveform requires 1 to 3 poles for modelling. So if the sampling rate is f_s, the model should contain at least p poles, where p is given by:

$$p \geq \frac{f_s}{1\text{kHz}} + 1.$$

More poles improve the model accuracy. But a minimum number of poles is desired to minimize the transmission rate.

One approach to the calculation of the L.P.C. coefficients (the a_i's in the vocal tract transfer function) is the autocorrelation approach [3]. This approach requires the computation of p+1 autocorrelation values of the speech waveform computed over a period during which the speech is not changing (i.e. only one sound is being made). The autocorrelation values can be directly transformed into the L.P.C. coefficients (see Appendix A). To accomplish this, the speech is bandlimited to half the sampling frequency and then sampled. A string of

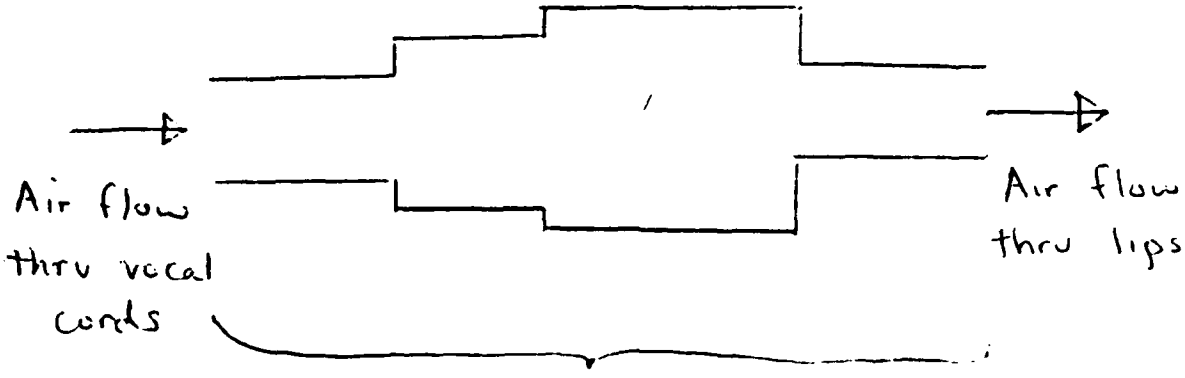
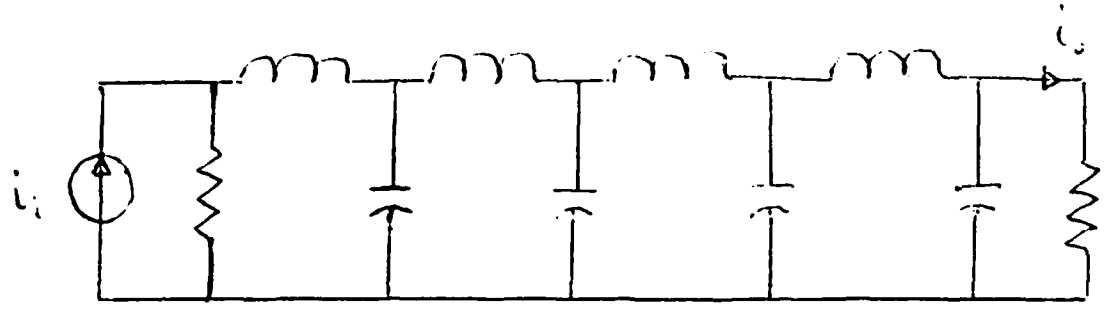


Figure 4a) vocal tract model



$$\frac{i_o}{i_i} = \frac{1}{D(s)}$$

Figure 4b) Electrical Equivalent of 4a).

approximately 240 consecutive speech samples are then used in the auto-correlation calculation:

$$R(k)=\sum_{i=-\infty}^{\infty} s_w(i)s_w(i+k) \quad k=0,1,\ldots p$$

where s_w refers to the "windowed" speech samples (see below). The model parameters (the a_i's) can be calculated directly from the R(k)'s (see appendix A).

The usual approach is to multiply the samples of the incoming speech waveform by a finite impulse response (F.I.R.) time window. Typical windows that are applied to speech are Hanning and Hamming windows [4]. A much simpler method for calculating the autocorrelation values was pointed out by Barnwell [5]. Rather than use a finite impulse response time window on the speech, Barnwell considered using an infinite impulse response (I.I.R.) time window. As long as the I.I.R. window is very small outside a 30 msec wide region; the results should be very similar to those obtained from the application of a F.I.R. window. The advantage of the I.I.R. approach is that the I.I.R. window can be realized by a simple recursive filter. A brief summary of Barnwell's work follows:

> The speech is sampled. Let's call the sequence of sampled speech s(n). This input sequence is divided into "frames" consisting of 240 consecutive samples * (for an 8kHz sampling rate). Each frame is multiplied by a time window. Let the index "j" refer to the largest index of a sample in any frame (i.e. frame j ends with sam-

* Only for a F.I.R. window. An I.I.R. window multiplies an infinite number of samples.

ple s(j) and begins with sample s(j-240)). For convenience, the time window w(i) is defined such that:

$$w(i)\neq 0 \quad \text{for} \quad i\leq 0$$

$$w(i)=0 \quad \text{otherwise}$$

For a finite impulse response window, $w(i)\neq 0$ only for $-240\leq i\leq 0$ (fig. 5). After applying the time window to frame j, a new sequence is created:

$$x(i,j)=s(i)w(j-i). \qquad \{1\}$$

The autocorrelation values for frame j can then be calculated as

$$R(k,j)=\sum_{i=-\infty}^{\infty} x(i,j)x(i+k,j) \qquad k=0,1,2,\ldots p. \qquad \{2\}$$

where $R(k,j)$ is the k^{th} autocorrelation value calculated from the samples in frame j.

Usually a finite length Hanning window of 30 msec width is used. If a time window which is infinite in length but very small outside a 30 msec period is used instead, the L.P.C. system is greatly simplified.

The time window which will be used is the impulse response of a second order filter having two coincident, real poles:

$$W(z)=\frac{1}{(1-\alpha z^{-1})^2}.$$

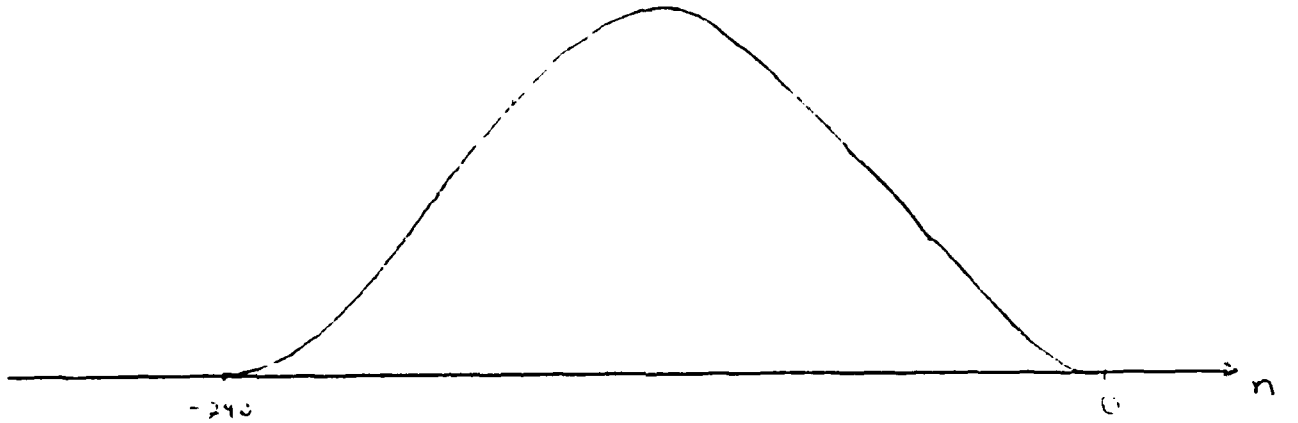
Its time-reversed impulse response is (fig. 5):

$$w(n)=(1-n)\alpha^{-n} \qquad n\leq 0$$

$$w(n)=0 \qquad n>0$$

Using equations {1} and {2}, we get

Hanning Window (F.I.R.)



Barnwell's Window (I.I.R.)

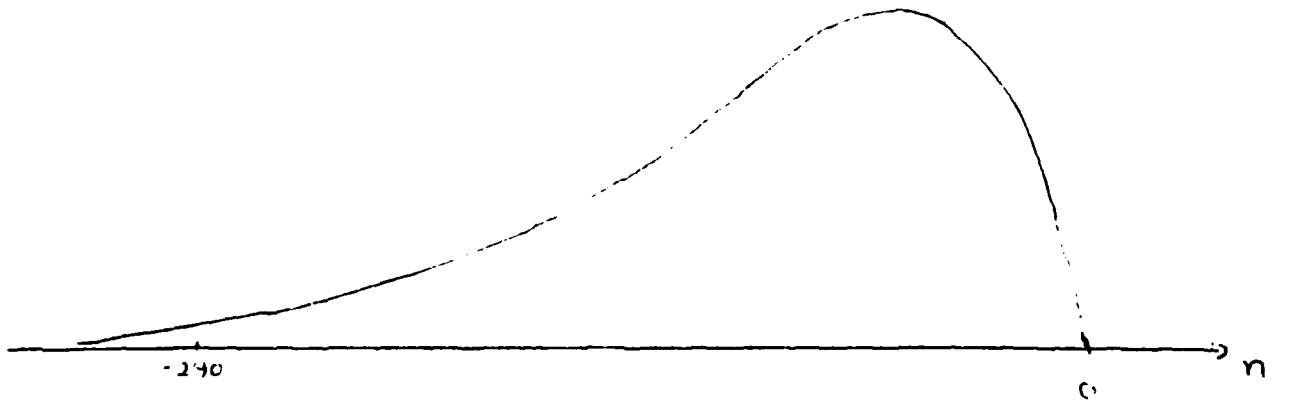


Figure 5

$$R(k,j)=\sum_{i=-\infty}^{\infty} s(i)w(j-i)s(i+k)w(j-i-k). \qquad (3)$$

Now define

$$s'(i,k)=s(i)s(i+k)$$

$$w'(i,k)=w(i)w(i-k).$$

Then we can write equation (3) as

$$R(k,j)=\sum_{i=-\infty}^{\infty} s'(i,k)w'(j-i,k).$$

From the above equation it can be seen that R(k,j) is the convolution of s'(i,k) with w'(i,k). So if we can produce the sequence s'(i,k) and pass it through the linear, time invariant filter with impulse response w'(i,k), we will have calculated the autocorrelation function for lag k (using the data from frame j).

Producing the sequence s'(i,k) requires only multiplication and delay. We need to find the filter corresponding to w'(i,k). Recall that

$$w'(i,k)=w(i)w(i+k). \qquad (4)$$

The corresponding z-transforms are defined as:

$$w(i) \rightarrow W(z)$$

$$w(i-k) \rightarrow z^{-k}W(z)$$

$$w'(i,k) \rightarrow W'_k(z)$$

Since multiplication in the time domain corresponds to convolution in the frequency domain, we can get $W'_k(z)$ from equation (4):

$$W'_k(z) = w(z)*z^{-k}W(z) = \frac{1}{2\pi j}\oint W(v)\left(\frac{z}{v}\right)^{-k}W\left(\frac{z}{v}\right)v^{-1}dv.$$

This integral can be evaluated to give

$$W'_k(z) = \frac{(k+1)d^k + (1-k)d^{k+2}z^{-1}}{(1-d^2z^{-1})^3}.$$

Note that each value of k corresponds to a different filter. All the filters have 3 poles at d^2 and one real zero. Barnwell has found experimentally that d=0.98 is the best choice for a 9 pole L.P.C. model with an 8kHz sampling rate.

Notice that the I.I.R. system does not require any lengthy memory storage. In fact, the only elements of the system are a sampler, a 10-element delay line, multipliers, and filters. Rather than implement the system on a digital computer, it is possible to realize the entire L.P.C. system using MOS switched-capacitor circuit techniques to handle the autocorrelation computation with a microprocessor handling Durbin's Recursion (Appendix A). This might lead to a 3 or 4 chip L.P.C. speech analysis system (fig. 6).

Before designing the integrated circuits, some important questions have to be answered. For example: How good must the multipliers be? How important are the characteristics of the 10 integrating filters? What is the best approach to the system which will perform all the desired computations and take as little silicon area as possible? To answer these and other questions, the system was constructed on a breadboard using only standard MOS building blocks (op amps, switches, and capacitors).

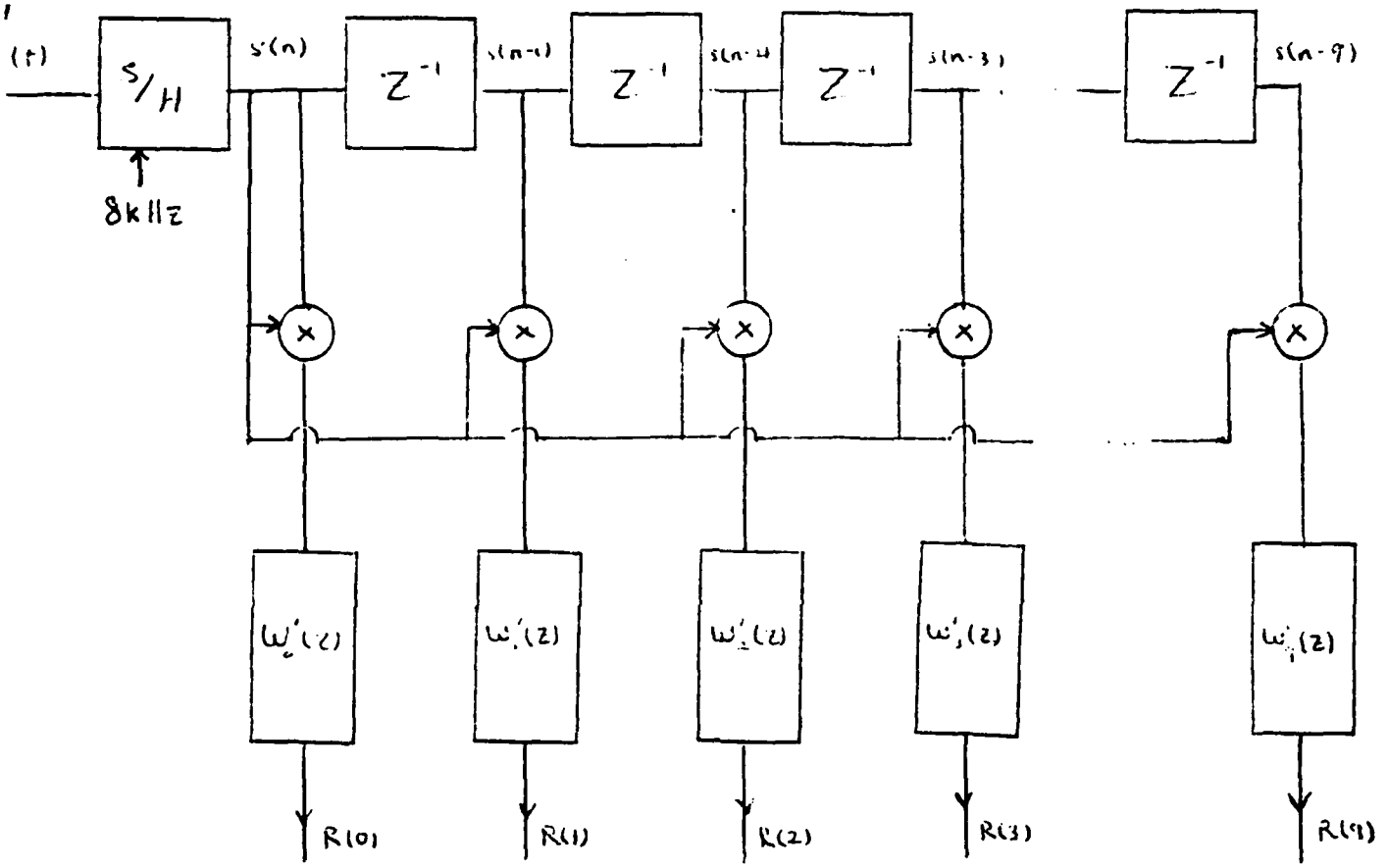


Figure (6.a) Block Diagram of System

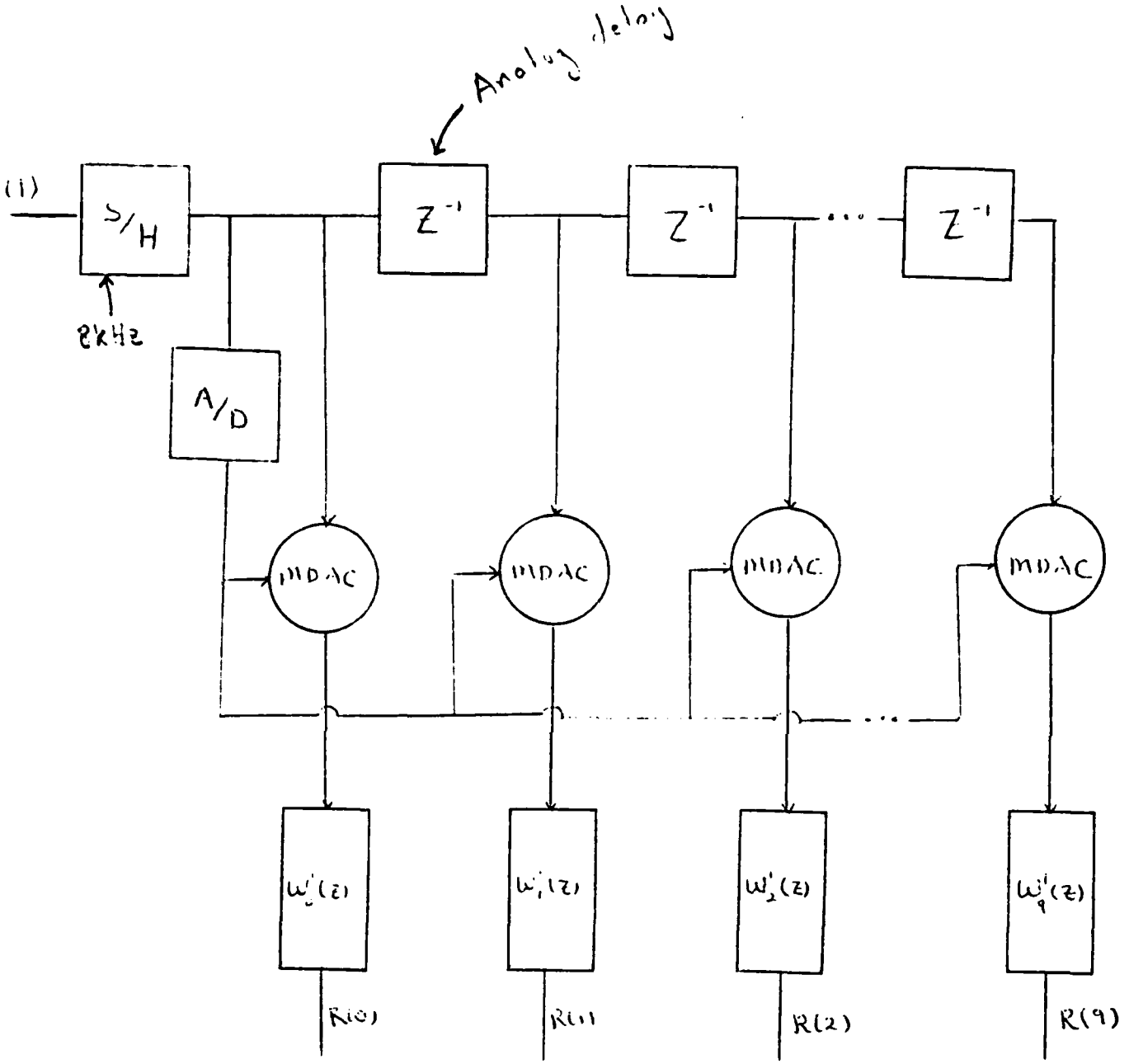


Figure (6.b) One possible implementation of 6.a).

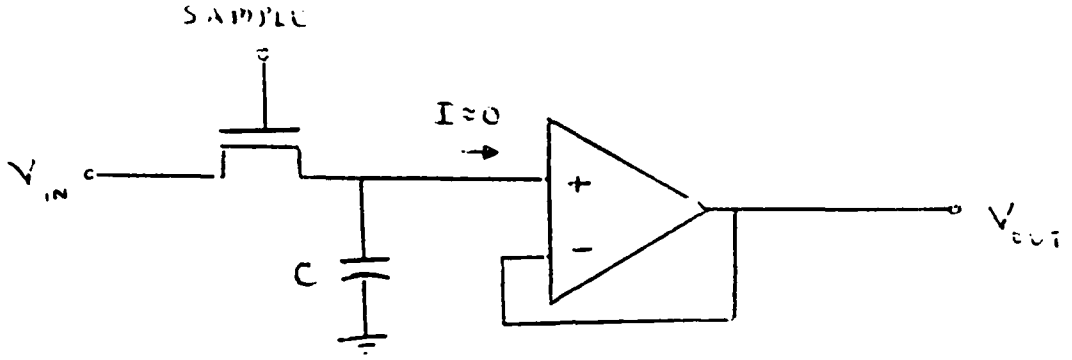


Figure 7a) Sample & Hold

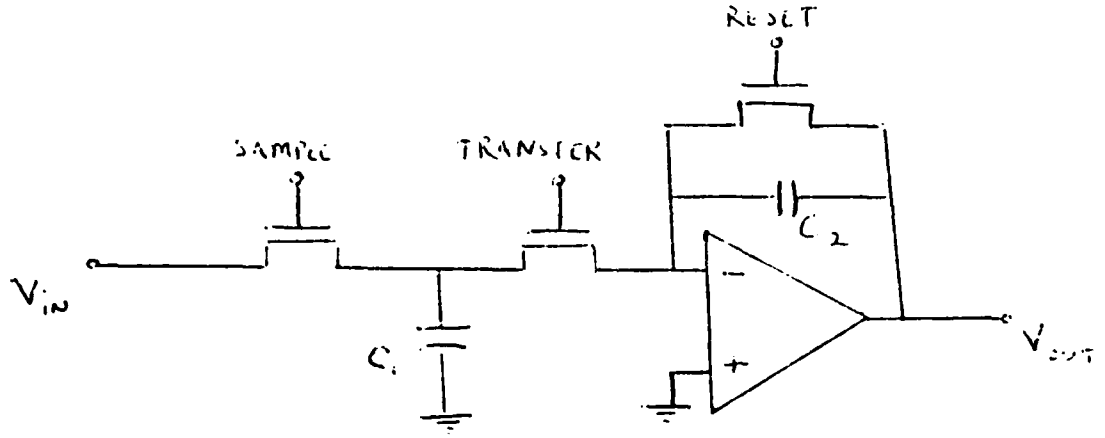
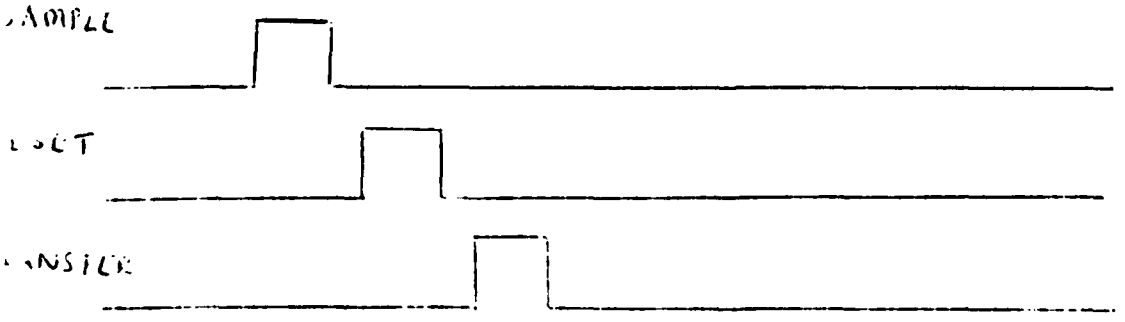


Figure 7b). Analog Delay Element



3. The Breadboard

A breadboard version of the I.I.R. system was built using the 10 filters suggested by Barnwell. Following his results, the system operates at an 8kHz sampling rate and uses a 9 pole model.

As mentioned earlier, the elements of the system are 1) a sample and hold, 2) a delay line, 3) multipliers, and 4) filters. Since a breadboard was being constructed, circuits employing a minimum number of precision ratioed capacitors were favored. This was an important consideration in selecting a multiplier scheme.

All the individual parts of the system will be presented separately. The following discussion assumes that all circuit elements are ideal and that all capacitor ratios are exactly as desired. Non-idealities will be considered in a later section.

All MOS transistors drawn in the figures are n-channel devices. Therefore they act like a closed switch when their gate voltage is high. Also, to prevent adjacent switches from being on simultaneously, all clock signals are non-overlapping (i.e. ϕ and $\overline{\phi}$ are non-overlapping complements).

3.1. Sample and Hold

A sample and hold consists of a switch for sampling, a capacitor for storing the sampled voltage, and an op amp for buffering (fig. 7). The sample and hold circuit acquires a new speech sample every 125 microseconds.

3.2. Delay Line

This particular delay line uses a 3-phase clock (fig.7). On SAMPLE=1, the input voltage is stored on C_1. On RESET=1, C_2 discharges completely. On TRANSFER=1, the charge on C_1 is forced onto C_2. This gives an output voltage

$$V_{out}(n+1) = -\frac{C_1}{C_2} \cdot V_{in}(n).$$

This voltage remains at the output during the next SAMPLE pulse, allowing similar stages to be cascaded to form a delay line. (Ideally, $C_1=C_2$ which gives $V_{out}(n+1)=-V_{in}(n)$.)

3.3. The Multiplier

A high-accuracy multiplier was desired for use in this system. Analog multipliers typically have accuracies on the order of 1%. Rather than build a breadboard with such a limited multiplier accuracy, it was decided to construct the multiplier from an analog-to-digital converter (A/D) and a multiplying digital-to-analog converter (MDAC). This approach can provide sufficient accuracy and allows flexibility for experimentation (i.e a simple wiring change allows us to see how the system will perform with an 8-bit multiplier).*

* Since the output of each multiplier is passed through a low pass filter (bandwidth of about 30Hz), it is not clear that a high-accuracy multiplier is required. But a large dynamic range is necessary.

3.3.1. The A/D Converter

The A/D converter used was first presented by R. McCharles [6]. It requires only 5 precision ratioed capacitors, 2 op amps, switches, and some associated control logic (fig. 8). The digital word is output serially. The digital output is always monotonic. The number of bits per conversion is set by the control logic, but there are many circuit limitations which set a practical upper limit on the number of bits.

An 11-bit, bipolar converter was constructed (10 bits + sign bit). It accepts inputs between +10V and -10V. The digital word is an offset binary representation of the analog voltage.

The McCharles' A/D Converter implements a non-restoring divide algorithm. If we consider each op amp and its integrating capacitor as an analog register, the operation of the converter can be written compactly:

1. Clear (Op Amp 1) and (Op Amp 2) {discharge all capacitors}
2. (Op Amp 1) $\leftarrow V_{in1}$; S=sign(Op Amp 1) {load Op Amp 1, set sign bit}
3. (Op Amp 2) $\leftarrow$ (Op Amp 1) {shift}
4. (Op Amp 1) $\leftarrow 2*(\text{Op Amp 2}) - S*V_{ref}$; S=sign(Op Amp 1) {set MSB}
5. for i=1 to 9
6. (Op Amp 2) $\leftarrow$ (Op Amp 1) {shift}
7. (Op Amp 1) $\leftarrow 2*(\text{Op Amp 2}) - S*V_{ref}$; S=sign(Op Amp 1) {set next bit}
8. next i
9. end {done after LSB}

The function "sign(Op amp 1)" is defined to be the logical "0" when

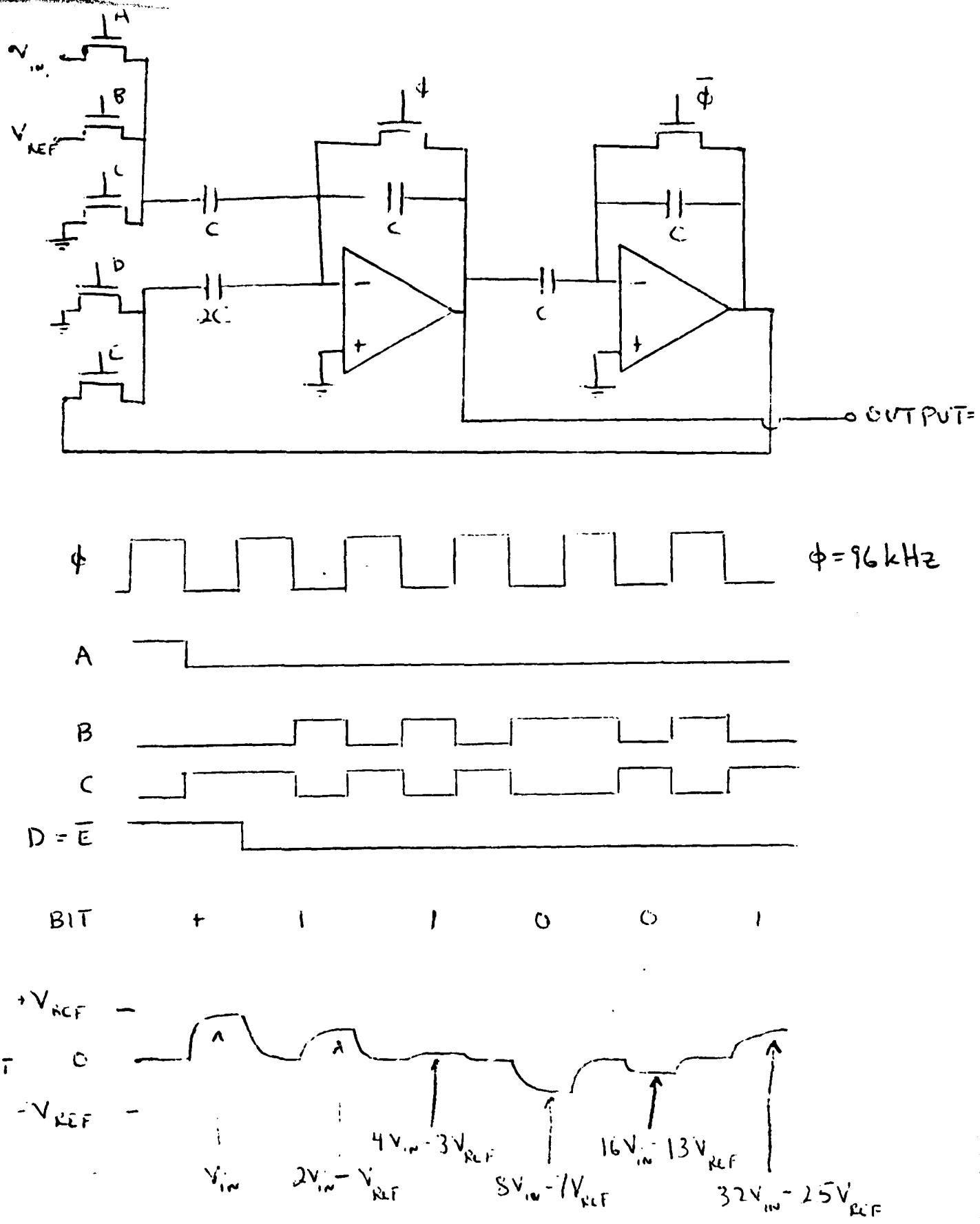


Figure 8 A/D Converter and Logic Signals for $V_{IN} = \frac{25}{32} V_{REF}$

the op amp output voltage is negative and the logical "1" when the op amp output voltage is non-negative. The variable "S" is the serial output of the A/D converter. One conversion requires 12 clock cycles. To complete one conversion every 125 microseconds (8kHz sampling rate), the A/D converter uses a 96kHz clock.

3.3.2. The Multiplying D/A Converter

A simple D/A converter was designed by R. Fellman. It requires only 2 precision ratioed capacitors, making it an ideal candidate for the breadboard. It is capable of doing four quadrant multiplication and is compatible with the McCharles' A/D (fig. 9).

The two equal capacitors ($C_1 = C_2$) are used to divide the voltage V_{in2} by successive factors of two. These fractions of V_{in2} are then selectively integrated by C_3 depending on the bit of the incoming digital word (see fig. 9 for an example). After 12 clock cycles, C_3 is charged to a voltage equal to

$$V_{out}(n) = \frac{V_{in1}(n) \cdot V_{in2}(n)}{10.0V} .$$

3.4. Filters

The system requires the use of filters at both the input and the output. The incoming speech must be bandlimited to 4kHz and preemphasized before it is processed. The output of the multipliers must be integrated by the 10 low pass filters to give the autocorrelation values.

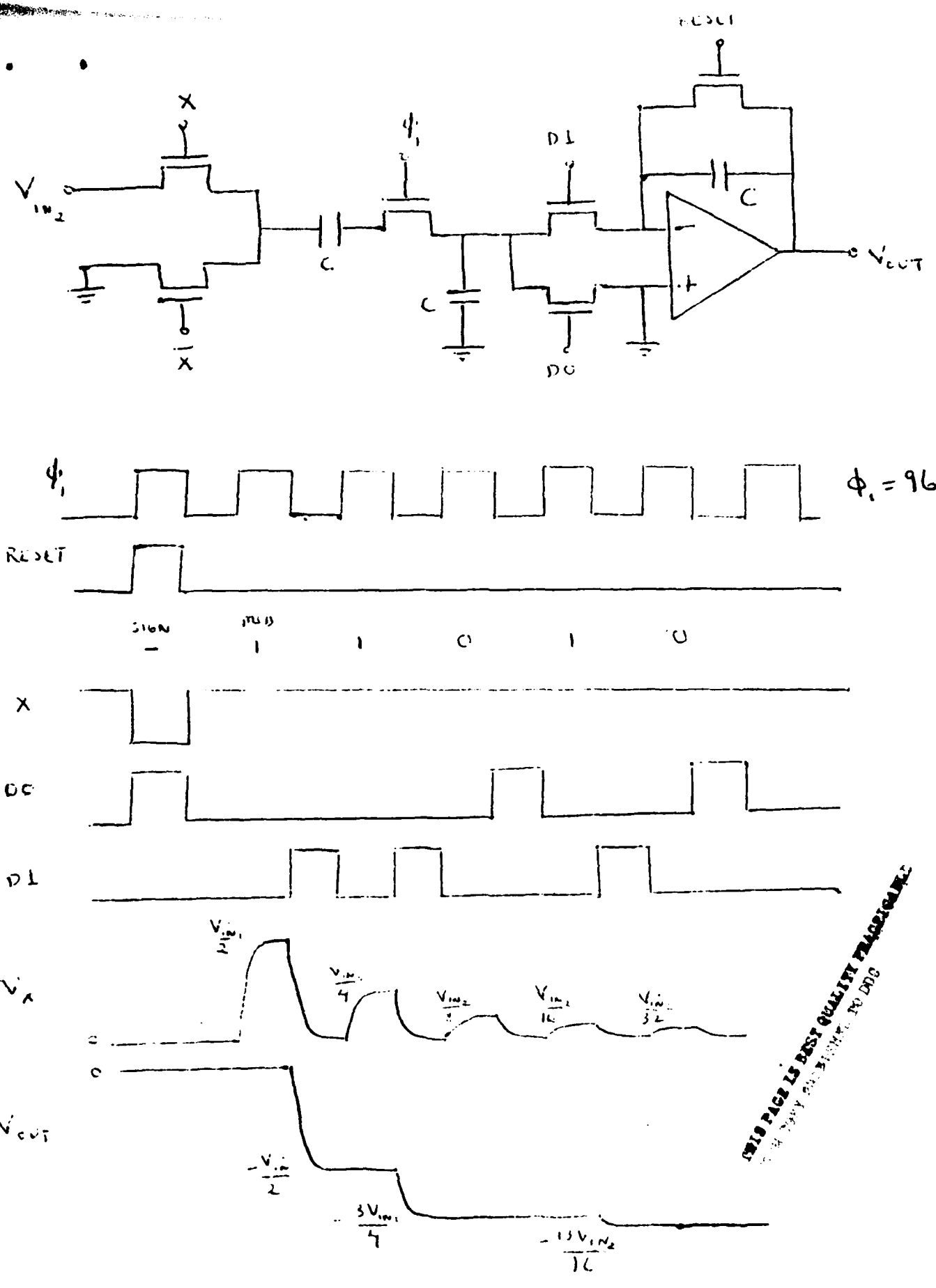


Figure 4 D/A Converter and Logic Signals

3.4.1. Input filters

To bandlimit the incoming speech to 4kHz, a fifth order Chebyshev filter was used. It has 1db of ripple in the passband (0 to 3kHz). At 4kHz, the response is down 24db (fig. 10). A complete discussion of the filter can be found in [7].

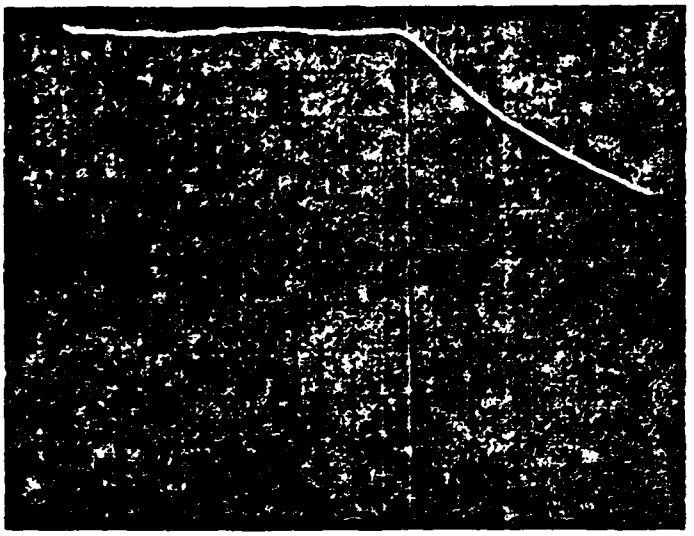
The speech was pre-emphasized to whiten the spectrum. This results in a better L.P.C. model [3]. The pre-emphasis filter has one zero at 500Hz and a pole at 4kHz (fig. 11).

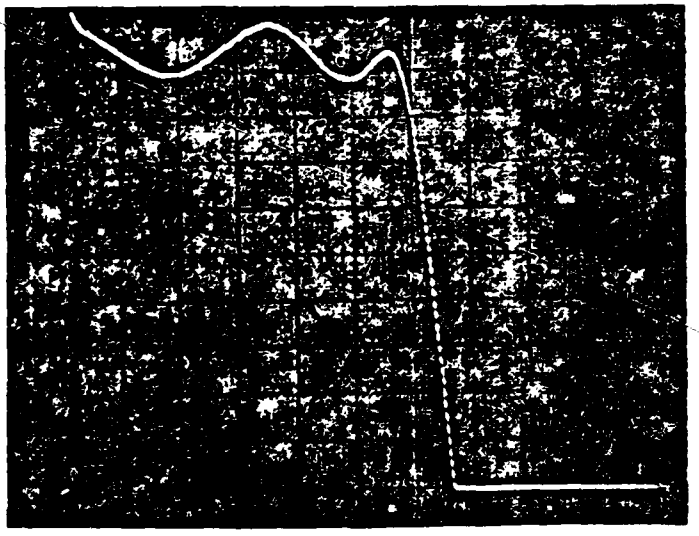
3.4.2. Output filters

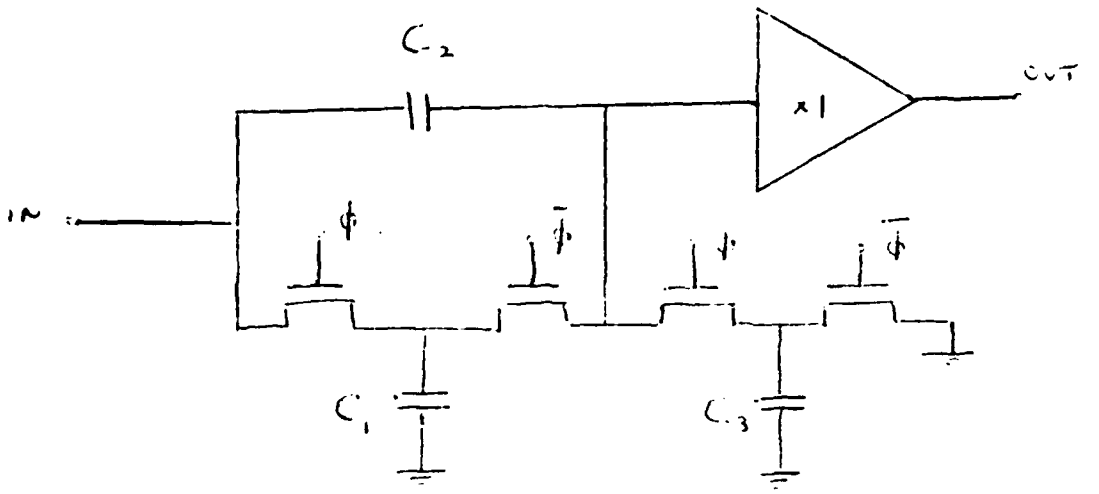
The 10 filters at the output of the multipliers have 3 coincident, real poles and one real zero. These filters can be realized simply (fig. 12). The first two sections of the filter each realize a pole at $z=d^2=0.9604$. The third section realizes a pole and a zero. The location of the zero depends on the autocorrelation lag k. Table 1 lists the pole-zero locations for all 10 filters. As with all switched-capacitor filters, the locations of the poles and zeros are determined by capacitor ratios.

The switched-capacitor filters all have a gain of 1.00 at D.C. Table 1 shows that all the filters should have a slightly different gain at D.C. This D.C. gain factor can be included in the first step of the Durbin's Recursion Algorithm (see appendix A).

The low-pass action of the output filters reduces the contribution of quantization noise and circuit noise at the outputs.







$\phi = 8\,kHz$

Figure 11a). A Pre-emphasis Filter

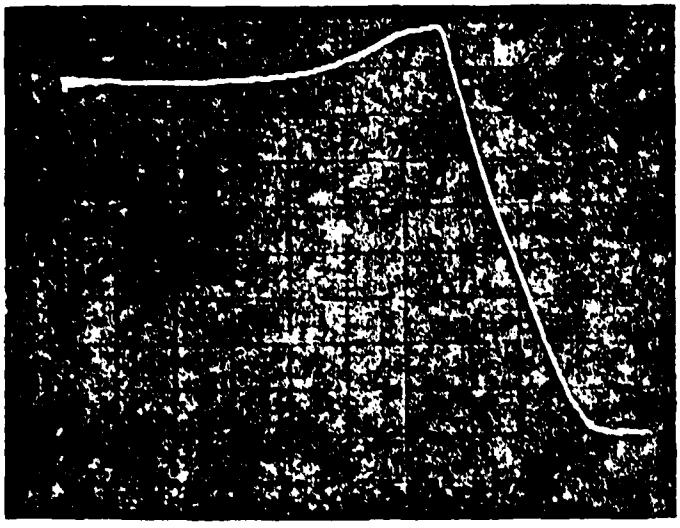
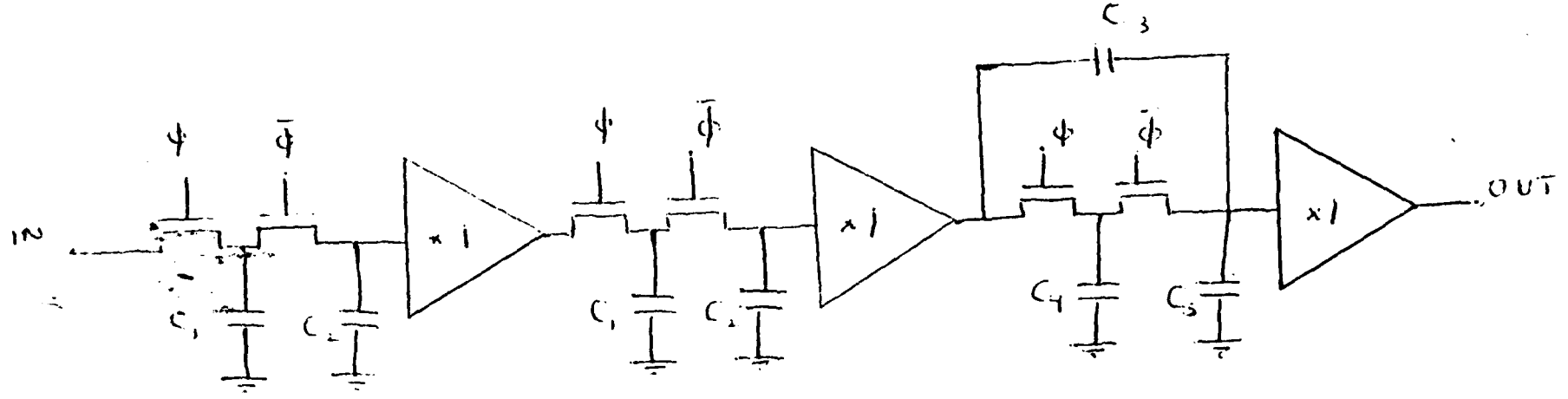


Figure 11.b). Frequency Response of the synthesis filter in series with Low Pass Filter



$$W_K'(z) = \frac{V_{OUT}}{V_{IN}}(z) = \left[\frac{\frac{C_1}{C_1+C_2}}{z - \frac{C_2}{C_1+C_2}}\right]^2 \left[\frac{C_3}{C_3+C_4+C_5} \cdot \left(\frac{z - \left(1 - \frac{C_4}{C_3}\right)}{z - \frac{C_3+C_5}{C_3+C_4+C_5}}\right)\right]$$

$\phi = 8\text{kHz}$

Figure 12. Circuit for the Output Filters

Table 1

filter number	DC gain	zeros (z =)	poles (z =)
k = 0	1.9604	-0.9604	0.9604
k = 1	1.9600	0.0000	0.9604
k = 2	1.9588	0.3201	0.9604
k = 3	1.9569	0.4802	0.9604
k = 4	1.9543	0.5762	0.9604
k = 5	1.9510	0.6403	0.9604
k = 6	1.9471	0.6860	0.9604
k = 7	1.9425	0.7203	0.9604
k = 8	1.9374	0.7470	0.9604
k = 9	1.9316	0.7683	0.9604

DC gain and pole-zero locations for the 10 output filters

Table 2

filter number	$\frac{C_2}{C_1}$	$\frac{C_3}{C_4}$	$\frac{C_5}{C_4}$
k=0	24.25	1.960	46.54
k=1	24.25	1.000	23.25
k=2	24.25	0.680	15.49
k=3	24.25	0.520	11.61
k=4	24.25	0.424	9.28
k=5	24.25	0.360	7.72
k=6	24.25	0.314	6.61
k=7	24.25	0.280	5.78
k=8	24.25	0.253	5.14
k=9	24.25	0.232	4.62

Capacitor ratios required for figure 12 to achieve pole-zero locations of Table 1.

4. Circuit Limitations

On the breadboard, the parts used were:

silver-dipped mica capacitors
glass capacitors
National LF356 op amps
Siliconix DG201 CMOS switches
Siliconix DG190 JFET switches
Siliconix DG303 CMOS switches
Siliconix D130 switch drivers
Signetics SD5000 DMOS switches
National LF398 sample and holds
74LS00 series logic gates

The LF356 op amps have a JFET input stage, so the input bias currents are very small (< 8 nanoamps). This is necessary for switched capacitor circuits since the input bias current must flow through the capacitors. These op amps are unconditionally stable when driving capacitive loads of 10 nanofarads, which allows large capacitors to be used. Other excellent properties of the 356 are high slew rate, low noise, and fast settling.

The commercial switches all have very low on resistance (< 200 ohms). Very large MOS transistors are used. Unfortunately, the large devices have large parasitic capacitances. The gate-to-drain and gate-to-source overlap capacitances inject a portion of the clock signals into the signal path. The junction capacitances (source-to-bulk and drain-to-bulk) are non-linear functions of voltage. Since they shunt the precision ratioed capacitors, they can introduce distortion into the system. The use of large capacitors can minimize both problems. An upper limit on the size of the capacitors is set by the clock period and the on resistance of the switches.

For all the circuits except the multiplier, the feedthrough of the clock onto the signal path results in a D.C. offset because the same amount of feedthrough occurs on every clock transition. The multiplier circuit is more complicated (see below).

4.1. Delay Line Limitations

From section 3.2, we have

$$V_{out}(n+1) = -\frac{C_1}{C_2} \cdot V_{in}(n).$$

If $C_1 \neq C_2$, the delay element has introduced a multiplicative error in the system which must be corrected. This is most conveniently done by the microprocessor which is implementing Durbin's Recursion (Appendix A).

Each delay element introduces a D.C. offset voltage into the delay line. This is undesirable because the D.C. offset of each delay element multiplies the signal. If the offset voltage is kept small (< 10 millivolts), the contribution of this undesired signal feedthrough will be negligible due to the effect of the low pass filters. (This problem is eliminated if a digital delay line is employed.)

4.2. Multiplier Limitations

4.2.1. A/D Converter Limitations

The A/D converter has 3 major error sources: 1)loop gain error, 2)loop offset error, and 3)slow settling error.

Ideally, the circuit has a loop gain of exactly two. It is this loop gain which sets the radix of the conversion. The loop gain is set by the capacitor ratios and the open loop gain of the op amps. On the

breadboard, an adjustable capacitor was connected in parallel with capacitor "2C" to allow the loop gain to be trimmed. For the 10-bit converter, the error in the loop gain must be less than 0.1%.

Loop offset error is caused by op amp input offset voltage, comparator offset voltage, and clock feedthrough. To insure less than $\pm \frac{1}{2}$ LSB error at the origin, we require

$$2^{10}(V_{\text{loop offset}}) < 10.0V$$
$$\text{or } V_{\text{loop offset}} < 10mV.$$

Slow settling is caused by the op amp outputs not settling to within 0.1% of their final value in the time allowed. The time required to settle has two components:RC time constants and op amp settling time. For this system, the clocks ran slow enough so that adequate time was allowed for settling.

4.2.2. D/A Converter Limitations

The sources of error in the D/A are 1)capacitor ratio error, 2)op amp offset voltage, 3)finite op amp gain, 4)clock feed through, and 5)slow settling error.

As in the A/D converter, the radix of the conversion is set by capacitor ratios:

$$\text{radix} = 1 + \frac{C_1}{C_2} .$$

To guarantee $\pm \frac{1}{2}$ LSB linearity, the capacitors must match to better than 0.1%.

Capacitor C_2 is always discharged on $\bar{\phi}$. If the bit of the digital

word is a "1", the capacitor is discharged to the virtual ground at the op amp's inverting input. If the bit is a "0", the capacitor is discharged to ground. The op amp virtual ground is at a potential V_{offset} above ground. To assure proper division of V_{in2} by factors of 2, C_2 must be discharged into the same potential on a "1" and on a "0". The op amp's offset voltage was externally nulled to ground potential to eliminate this error.

Due to the finite gain of the op amp, there is a potential difference between the inverting and non-inverting terminals:

$$V_+ - V_- = \frac{V_{out}}{\text{open loop gain}}.$$

This difference gets larger as V_{out} increases. When the D/A converter is processing the LSB, this difference (worst case) is

$$V_+ - V_- \simeq \frac{10.0V}{\text{open loop gain}}.$$

To maintain $\pm \frac{1}{2}$ LSB accuracy, the op amp must have an open loop gain of at least 2000. For the LF356, the open loop gain is about 200,000. This gives a difference voltage of only 50 microvolts, which is negligible.

The dominant limitation in the D/A performance is the clock feedthrough. The logic signals inject charge onto capacitor C_2. Unfortunately, two of the logic signals are the digital word and its complement. Since the digital word is always changing, the amount of feedthrough onto C_2 depends on the digital word. This input-dependent feedthrough is added to the D/A analog output voltage. This error was larger than all other errors, setting the lower limit on the analog output voltage at approximately 5 millivolts.

To make the D/A converter compatible with the McCharles' A/D converter, it ran at the same clock frequency (96 kHz). Therefore, the D/A converter had adequate time for settling.

4.3. Filters

The simplicity of the filters results in few problems. The op amp input offset voltage and clock feedthrough result in a D.C. offset voltage at the output. The effect of non-linearities due to junction capacitances in parallel with the switched capacitors were minimized by using large capacitors.

5. Measured Results

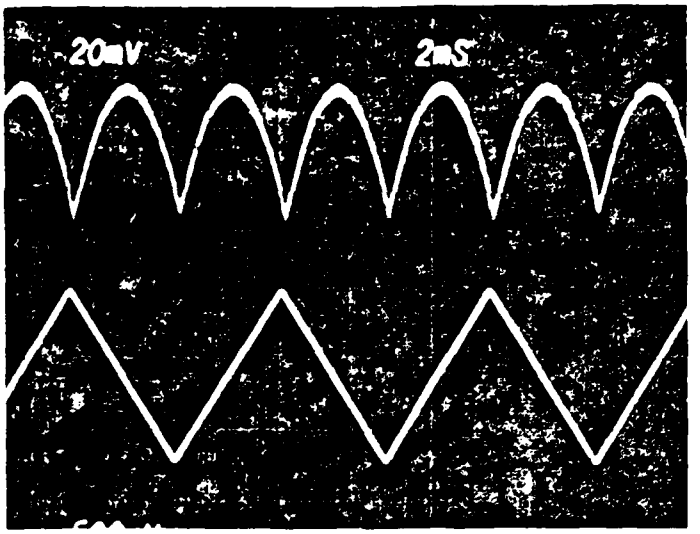
The multiplier was tested as a unit. First the A/D and D/A converters were checked for proper operation, and then the multiplier was tested. Figure 13 shows the multiplier operating in the squaring mode.

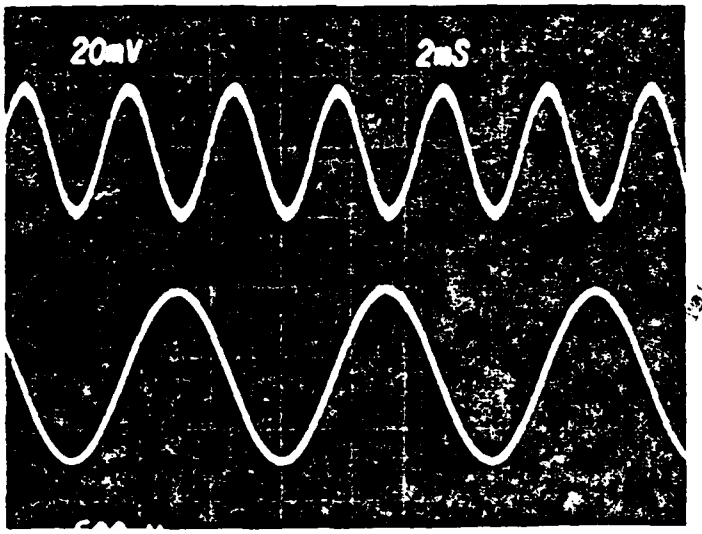
The multiplier output voltage has a range from 10.0V down to 5mV (see section 4.2.2). This is a 66db range.

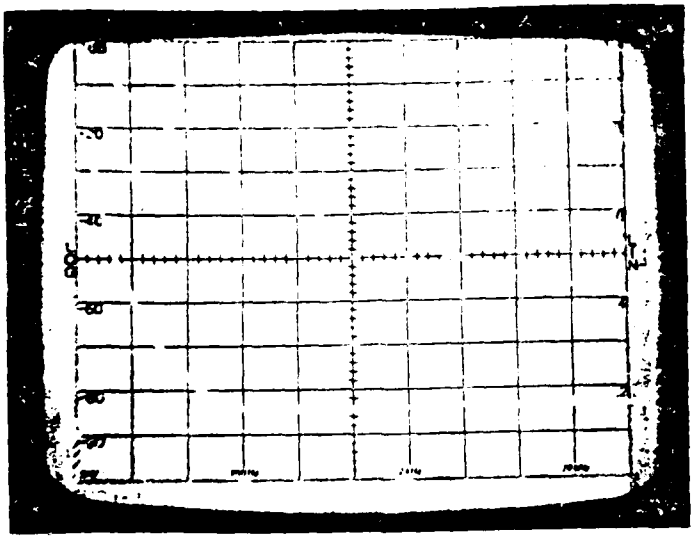
The output filters were checked for dynamic range and frequency response. The maximum input voltage to the filters is 10.0V. The noise level in the filters is below 1 millivolt. This corresponds to a dynamic range of better than 80db.

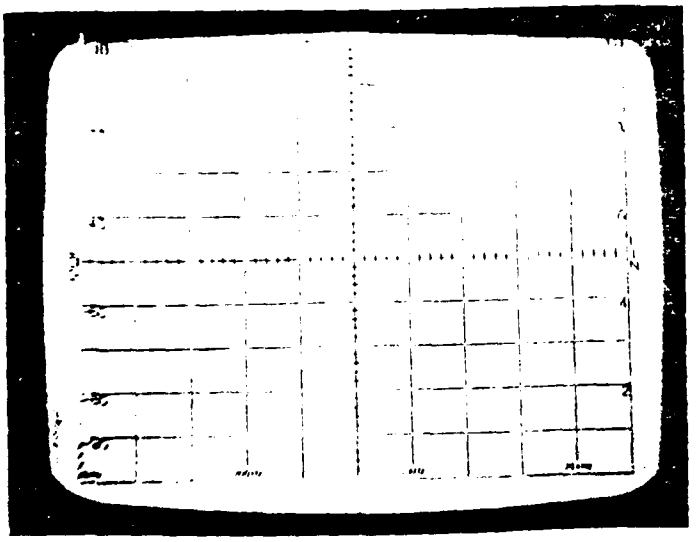
The frequency response for output filters k=0 and k=9 are shown in figure 14. Notice the similarity in the characteristics. The zero in the transfer function seems to have little effect.

A breadboard system similar to the one described here has been used to process speech [9]. Figure 15 shows the result of fitting a 9-pole L.P.C. transfer function to the sound /i/ (the 'e' sound in 'beet'). The fit is very good.









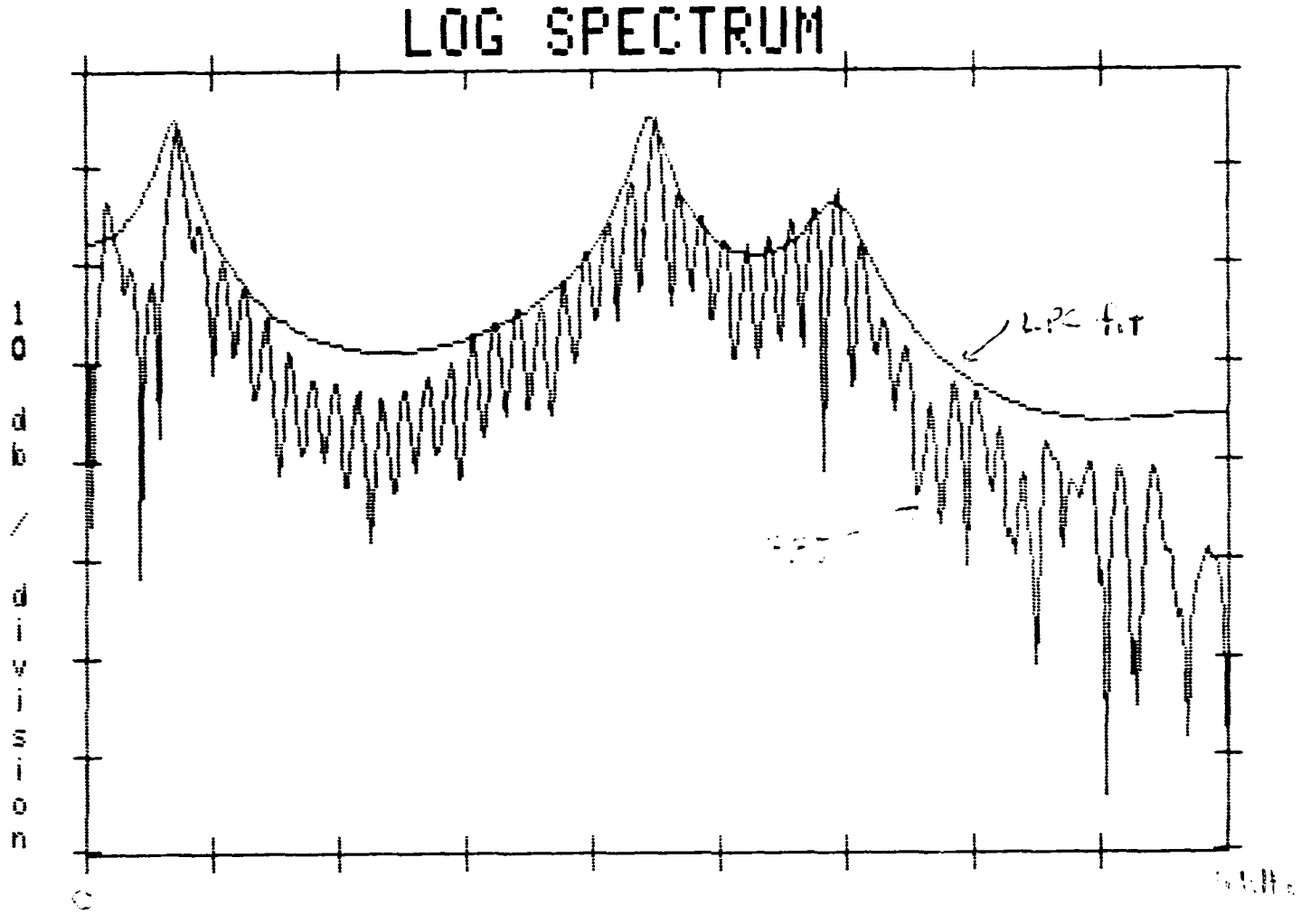


Figure 15. LPC fit to sound /i/.

5. Future Considerations and Suggested Improvements

The system proposed here is feasible, but is not the best approach to be taken for an integrated version. An integrated version could benefit from a number of changes:

[1] A digital delay line should be used rather than an analog delay line. This is more accurate and requires less silicon area.

[2] One MDAC can do the work of ten. The one MDAC could be time multiplexed to handle all 10 multiplications in the 125 microsecond clock period (fig. 16). This is not a severe requirement, and it would take less area. If only one MDAC is used, then all 10 autocorrelation values will have the same scale factor error due to the gain error of the A/D and MDAC. But since the solution of Durbin's Recursion depends only on the relative values rather than the absolute values of the autocorrelation function, this error need not be corrected.

[3] The buffer amplifiers in the output filters can be time multiplexed. Each filter needs buffers between stages. To save area, three buffer amplifiers can be shared among all 10 filters if the filters are driven by 10 skewed clocks (all at 8kHz).

[4] An A/D converter is required at the output of the system to convert the analog output voltages into digital words for use by the microprocessor. If all the autocorrelation output voltages are free of D.C. offsets, then we can use R(0) as the reference voltage of the A/D converter. (Recall that $R(0) \geq R(\tau)$ for all τ.) This results in an automatic scaling and normalization of the

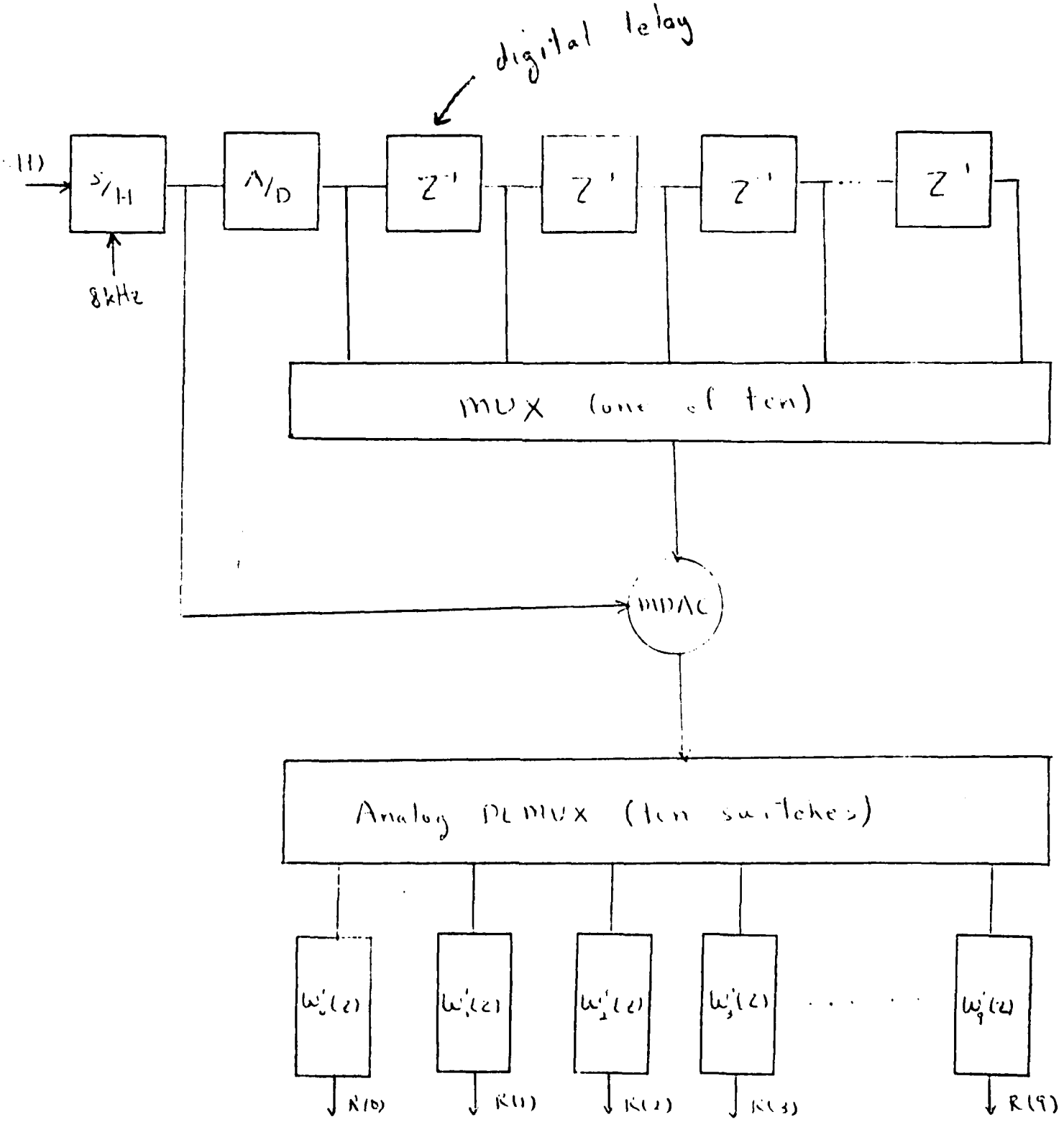


Figure 16 An Implementation of Figure 6.6
which is suitable for interpolation

autocorrelation values. This should ease the work of the microprocessor.

[5] To allow the system to handle a wide dynamic range of speech signals, the L.P.C. system should be preceded by some type of automatic gain control which would take full advantage of the A/D converter. The gain factor would have to be transmitted with the other model parameters to assure accurate synthesis.

APPENDIX A

An autocorrelation L.P.C. is based on minimization of the square of the predictor error over all speech samples. If we call e(i) the predictor error for sample "i" and $\hat{s}(i)$ the predicted speech sample, then the error can be written as

$$e(i) = s(i) - \hat{s}(i).$$

For an all-pole model of the vocal tract, the predicted speech sample is given as a linear weighted sum of the past p speech samples (p is the order of the predictor). In fact, the model parameters are the weighting coefficients:

$$\hat{s}(i) = \sum_{k=1}^{p} a_k s(i-k).$$

The a_k's are the model parameters discussed in section 2 (also called the L.P.C. coefficients).

We want to minimize the total mean squared error E_t:

$$E_t = \sum_{i=-\infty}^{\infty} e(i)^2 = \sum_{i=-\infty}^{\infty} \left[s(i) - \sum_{k=1}^{p} a_k s(i-k) \right]^2 .$$

The parameters (the a_k's) should be chosen to minimize this total error. The condition for minimum error can be found from:

$$\frac{\delta E_t}{\delta a_k} = 0 \qquad k=1,2,\ldots p$$

This results in a set of p linear equations which can be solved for the a_k's:

$$\begin{bmatrix} R(0) & R(1) & \cdots & R(p-1) \\ R(1) & R(0) & \cdots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ R(p-1) & R(p-2) & \cdots & R(0) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} R(1) \\ R(2) \\ \vdots \\ R(p) \end{bmatrix}$$

where R(k) is the k^{th} autocorrelation lag of the sampled speech (see section 2, equation (3)). If we have the p+1 values of the autocorrelation function, the model parameters can be obtained by solving this matrix equation.

Fortunately, the square matrix has a special form (it is Toeplitz). Therefore the a_k's can be solved for recursively. The most efficient method is Durbin's Recursion [3]:

1. $a_1^1 = \frac{R(1)}{R(0)}$
2. $E(1) = [1 - (a_1^1)^2]R(0)$
3. for i=2 to p
4. $a_i^i = \frac{R(i) - \sum_{j=1}^{i-1} a_j^{i-1} R(i-j)}{E(i-1)}$
5. $a_j^i = a_j^{i-1} - a_i^i a_{i-j}^{i-1} \quad 1 \leq j \leq i-1$
6. $E(i) = [1 - (a_i^i)^2]E(i-1)$
7. next i
8. end

where a_j^i refers to the j^{th} model parameter of an i-pole model. When i=p, we have calculated the model parameters for the p-pole model.

Durbin's Recursion requires about 100 multiplications (or divisions) and about 400 additions (or subtractions) for a 9 pole model. With the faster microprocessors (i.e. Texas Instruments 9900), the recursion can be completed with the transmission interval (= 10 milliseconds). Also, Durbin's Recursion uses only 18 storage locations.

Appendix B

Capacitor Values for the Figures

figure 7a): $C = 1000\,pF$

figure 7b): $C_1 = 560\,pF$

$C_2 = 560\,pF$

figure 8: $C = 560\,pF$

$2C = 1120\,pF$

figure 9: $C = 2200\,pF$

figure 11: $C_1 = 4740\,pF$

$C_2 = 10000\,pF$

$C_3 = 29500\,pF$

Capacitor Values for Figure 12 (in pF)

Iter number	C_1	C_2	C_3	C_4	C_5
=0	620	15000	484	950	22500
-=1	620	15000	0	620	15000
-=2	620	15000	950	646	14700
=3	620	15000	950	495	11000
-=4	620	15000	950	402	8820
=5	620	15000	950	342	7330
-6	620	15000	950	298	6290
-7	620	15000	950	266	5490
=8	620	15000	950	240	4880
=9	620	15000	950	220	4390

REFERENCES

[1] C.R. Hewes, W.L. Eversole, D.J. Mayer, T-K Hui, H.K. Hester, R.C. Pettengill, "A CCD/NMOS Channel Vocoder," Proc. Int'l Conf. Appl. Charge-Coupled Devices, 1978, pp. 3A-17 - 3A-24.

[2] M. Dalrymple, "Pitch Extraction Using MOS-LSI Circuitry," Dept. of EECS, Univ. of Calif., Berkeley, Masters Project Report.

[3] J. Makhoul, "Linear Prediction: A Tutorial Review," Proc. of the IEEE, April 1975, pp. 561-580.

[4] A.V. Oppenheim, R.W. Schafer, Digital Signal Processing, Englewood Cliffs, New Jersey: Prentice-Hall, 1975.

[5] T.P. Barnwell, "Recursive Autocorrelation Computation for LPC Analysis," Proc. Int'l Conf. on Acoustics, Speech, and Sig. Proc., 1977.

[6] R.H. McCharles, D.A. Hodges, "Charge Circuits for Analog LSI," IEEE Trans. on Circuits and Systems, July 1978, pp. 490-497.

[7] D.J. Allstot, R.W. Brodersen, P.R. Gray, "MOS Switched Capacitor Ladder Filters," IEEE Jour. of Solid State Circuits, Dec. 1978, pp. 806-814.

[8] G.S. Kang, "Application of Linear Prediction Encoding to a Narrowband Voice Digitizing," Naval Research Laboratory, Washington, D.C., N.R.L. Report 7774, Oct. 31, 1974, pp. 65-66.

[9] S. Lum, "An Analog-Digital Correlator," Dept. of EECS, Univ. of Calif., Berkeley, Masters Project Report.

[10] J. Markel, A. Gray, Linear Prediction of Speech, Springer-Verlag: New York, 1976.

[11] J.L. Flanagan, Speech Analysis Synthesis and Perception, second edition, New York: Springer-Verlag, 1972.

[12] L.R. Rabiner, R.W. Schafer, Digital Processing of Speech Signals, Englewood Cliffs, New Jersey: Prentice-Hall, 1978.

Introduction

The design of integrated monolithic frequency filters has made rapid advances in the last few years. Analog sampled data responses, which are continuous in amplitude and discrete in time, have been successfully implemented in NMOS switched capacitor technology, and numerous fixed coefficient filters have been fabricated [1]-[4].

A simplified diagram of a speech synthesis system is shown in Figure 1. The focus of this paper is on the time varying filter which models the vocal tract, the response of which must change to match speech characteristics. The switched capacitor technique lends itself to programmable filters, since the capacitors which control the vital characteristics of the response can be made variable arrays by switching in different amounts of capacitance rather than being of fixed size. Using this technique an electrically programmable NMOS second order switched capacitor filter has recently been developed by D. J. Allstot [5]. The vocal tract can be characterized by a ten pole response [6], [7], and a speech synthesizer can be implemented by cascading five of these second order sections. An alternative is a single ten pole filter. Several approaches for synthesizing the tenth order transfer function with a single switched capacitor circuit are discussed in this paper.

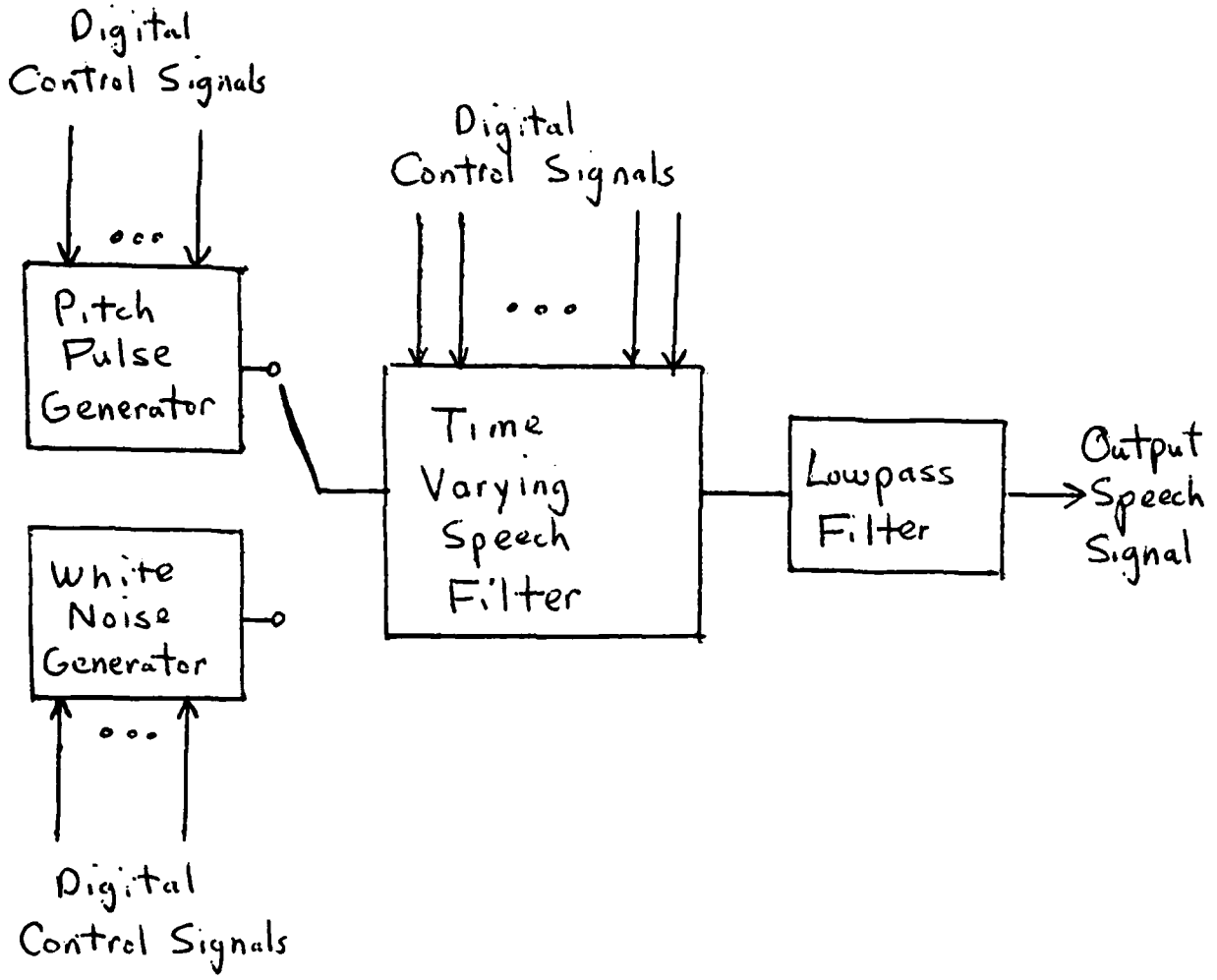


Figure 1: the total speech synthesis system

Chapter 1 - The General Problem

Before proceeding to specific synthesis techniques, a general overview of the problem and possible approaches to solution will be reviewed.

The z-domain sampled data transfer function of the vocal tract can be given by $T(z) = 1/(1+\sum_{i=1}^{10} h_i z^{-i})$. The h_i's are generally called the predictor coefficients and will be assumed to have been obtained from a previous linear prediction analysis of the speech [6], [7]. A typical sample of speech is used in this paper to test the techniques developed. Several basic filter structures will be investigated and compared with respect to sensitivity of the response to the number of bits accuracy used in realizing the filter coefficients. Bits are used for comparison because in an IC implementation the filter parameters are implemented as capacitor ratios which are varied by connecting or disconnecting elements in capacitor arrays. Binary weighting of the arrays is one of the easiest schemes to implement. A different weighting may be more efficient in certain cases, but a binary code gives general sensitivity properties and also a direct indication of chip area required, since increased accuracy implies larger capacitance area. The upper limit on accuracies achieved is about 10 bits [8].

The original processing of the speech to obtain the predictor coefficients was done at an 8 KHz sampling rate, so the sampled data filter simulations will also be done at the same rate, with the exception of a few cases in Chapter 4. To obtain the frequency response to be duplicated, consider the given transfer function $T(z) = 1/(1+\sum_{i=1}^{10} h_i z^{-i})$. In this paper only magnitude response is considered since the ear is not particularly sensitive to phase. To determine the frequency response from the

z-transform, $T(z)|z = \exp(j\omega T_s)$ is used, with $T_s = 1/8$ KHz.

$$T(\exp(j\omega T_s)) = 1/(1+ \sum_{i=1}^{10} h_i \exp(-ji\omega T_s)) \quad (1)$$

$$|T(\exp(j\omega T_s))| = 1/\sqrt{T_{REAL}^2 + T_{IMAG}^2} \quad (2)$$

$$T_{REAL} = 1 + h_1 \cos \omega T_s + h_2 \cos 2\omega T_s + \ldots + h_{10} \cos 10\omega T_s \quad (3)$$

$$T_{IMAG} = h_1 \sin \omega T_s + h_2 \sin 2\omega T_s + \ldots + h_{10} \sin 10\omega T_s \quad (4)$$

$$|T(\exp(j\omega T_s))| = -20 \log (\sqrt{T_{REAL}^2 + T_{IMAG}^2}) \text{ in dB} \quad (5)$$

The program for calculating and plotting the magnitude frequency response at 100 points between 100 Hz and 4 KHz is given in Figure 2. The predictor coefficients are read into the array H(10), and Y(100) is the output magnitude plotted vs. the frequency array WW(100) in subroutine PLOT. The user can specify the vertical range of the plot by supplying YLIT and YBIG and setting IPLT = 0, but if IPLT = 1 as in the example at hand, the subroutine calculates YLIT and YBIG from the array to be plotted, Y. The computer output, which is the magnitude response to be the standard for all filters in this paper, is given in Figure 3. From the plot four distinct resonant peaks are identified, and the center frequency f_0 and Q of each are as follows:

$F_1 = 607$ Hz $\qquad Q_1 = 5.2$

$F_2 = 1543$ Hz $\qquad Q_2 = 29.7$

$F_3 = 2401$ Hz $\qquad Q_3 = 8.4$

F_4 = 3454 Hz $\qquad$ Q_4 = 19.0

As will be seen in a later section, the thrid resonance is created by two adjacent complex pole pairs, but only one center frequency and Q can be identified from the plot. The above numbers will be the standards of comparison in sensitivity studies of the various filters to be investigated. On the plot it is also important to note the heights of all the peaks, as several filters in chapter 4 realize nearly correct center frequencies and Q's but have incorrect amplitude levels.

Sampled data filter circuit elements (delays, multipliers, and adders) will be used for synthesis and simulation. Typical examples of switched capacitor versions of these circuit elements are given in Figure 4. Further discussions can be found in [1]-[4], [9].

```
      DIMENSION W(500),Y(500),WW(500)
      NPTS=100
      IPLT=1
      XLIT=0
      W=0
      DO 1 I=1,10
    1 READ(3,2) H(I)
    2 FORMAT(F5.0)
      FC=8000.0
      PI=3.14159265
      W=629
      DO 2 I=1,100
      TREAL=1+H(1)*COS(W/FC)+H(2)*COS(2*W/FC)+H(3)*COS(3*W/FC)+H(4)*
     1COS(4*W/FC)+H(5)*COS(5*W/FC)+H(6)*COS(6*W/FC)+H(7)*COS(7*W/FC)
     1+H(8)*COS(8*W/FC)+H(9)*COS(9*W/FC)+H(10)*COS(10*W/FC)
      TIMAG=H(1)*SIN(W/FC)+H(2)*SIN(2*W/FC)+H(3)*SIN(3*W/FC)+H(4)*SIN
     1(4*W/FC)+H(5)*SIN(5*W/FC)+H(6)*SIN(6*W/FC)+H(7)*SIN(7*W/FC)+
     1H(8)*SIN(8*W/FC)+H(9)*SIN(9*W/FC)+H(10)*SIN(10*W/FC)
      TMAG=20*ALOG10(SQRT(TREAL**2+TIMAG**2))
      Y(I)=-TMAG
      WW(I)=W/(2*PI)
    2 W=W+245
      CALL PLOT(WW,Y,NPTS,IPLT,XLIT,BIG)
      END

      SUBROUTINE PLOT(X,Y,NPTS,IPLT,XLIT,BIG)
      DIMENSION X(500),Y(500),NA(26),KSTP(5),YY(11)
      DATA KSTP  4H*   ,4H *  ,4H  *,4H   *,4H    /
      IF(IPLT.EQ.0) GO TO 300
      XLIT=Y(1)
      DO 100 I=2,NPTS
  100 XLIT=AMIN1(XLIT,Y(I))
      BIG=Y(1)
      DO 200 I=2,NPTS
  200 BIG=AMAX1(BIG,Y(I))
  300 RANGE=BIG-XLIT

      DO 101 I=1,11
  101 YY(I)=XLIT+(I-1)*RANGE*.1
  440 FORMAT(1H8.14X,11F10.3)
      PRINT 440,YY
      PRINT 480
  480 FORMAT(20X,3H-1.10(10H----------),1H-)
      DO 600 I=1,26
  600 NA(I)=KSTP(5)
      DO 700 K=1,NPTS
      J=K
      IW=(Y(J)-XLIT)*RANGE*100. +1.5
      IF(IW.GT.101) IW=101
      IF(IW.LT.1) IW=1
      ITEM=(IW-1)/4
      NA(ITEM+1)=KSTP(IW-4*ITEM)
      PRINT 490
  490 FORMAT(' ',20X,1H.,10(10H         ),1H)
      PRINT 640,(J,(NA(I),I=1,26)
  640 FORMAT(' ',I4,F12.4,3X,1HI,25A4,A1,1HI)
  700 NA(ITEM+1)=KSTP(5)
      PRINT 480
  450 FORMAT(1HC,14X,11F10.3)
      PRINT 450,YY
      RETURN
      END
```

Figure 2 : Frequency Response Program, T(exp(jωTs))

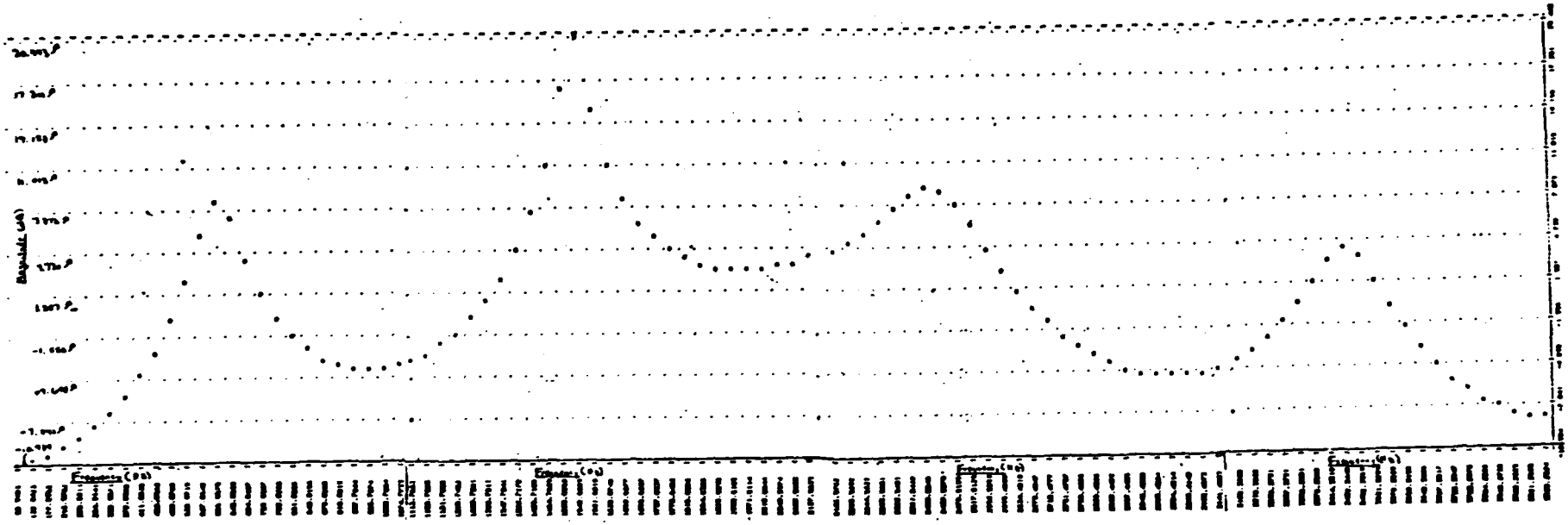


Figure 3:
Ideal Frequency Response

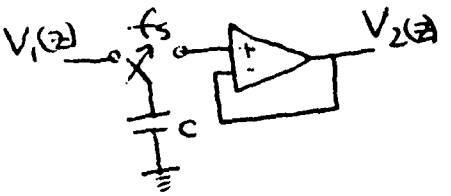
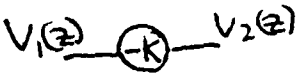
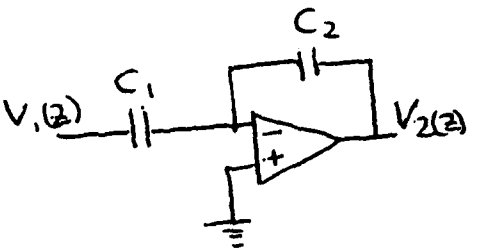
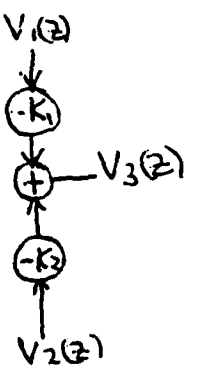
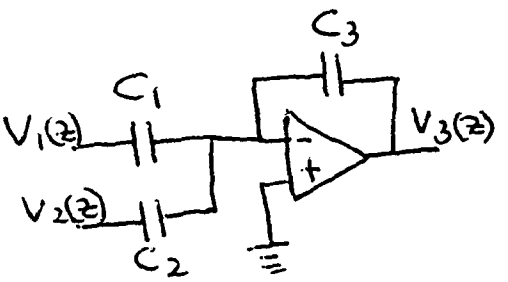
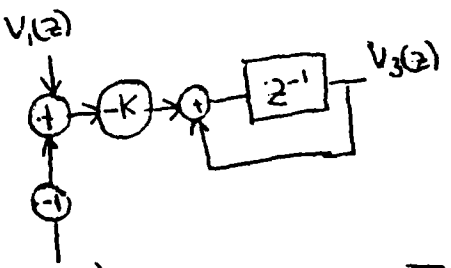
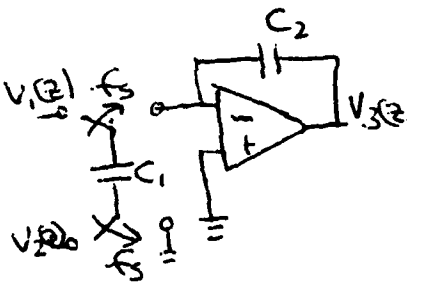
Circuit Function	Z-Domain Representation	Switched Capacitor Implementation
(a) Delay	 $V_2(z) = z^{-1}V_1(z)$	 $V_2(z) = z^{-1}V_1(z)$
(b) Multiply	 $V_2(z) = -KV_1(z)$	 $V_2(z) = -\frac{C_1}{C_2}V_1(z)$
(c) Add and Multiply	 $V_3(z) = -K_1V_1(z) - K_2V_2(z)$	 $V_3(z) = -\frac{C_1}{C_3}V_1(z) - \frac{C_2}{C_3}V_2(z)$
(d) PDI Section	 $V_3(z) = \frac{-Kz^{-1}}{1-z^{-1}}[V_1(z) - V_2(z)]$	 $V_3(z) = \frac{-\frac{C_1}{C_2}z^{-1}}{1-z^{-1}}[V_1(z) - V_2(z)]$

Figure 4: Circuit Elements

Chapter 2 - The Singly Terminated Ladder

One of the simplest structures from a synthesis standpoint is the singly terminated ladder described by Mitra and Sherwood in "Digital Ladder Networks," [10]. In this paper a transfer function in positive powers of z is required, which is obtained by multiplying the numerator and denominator of T(z) by z^{10} so that

$$T(z) = z^{10} \times (1/(z^{10} + \sum_{i=1}^{10} h_i z^{-i+10})) \tag{6}$$

since $|z^{10}| = 1$, the problem is reduced to synthesizing the equivalent transfer function (with respect to magnitude response) of

$$T(z) = 1/(z^{10}+h_1 z^9+ \ldots +h_9 z+h_{10}) \tag{7}$$

The method of Mitra and Sherwood is directly applicable to the case at hand, and through a proper change of variables the pure delays of their procedure can be replaced by Direct Discrete Integrator (DDI) sections shown in Figure 4(d).

As described in the Mitra and Sherwood paper, the general form of the filter without terminations is that of Figure 5. If the network is considered a 2-port, then the terminal characteristics can be related by a "C" matrix as follows:

$$\begin{bmatrix} B_1 \\ B_2 \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} \tag{8}$$

If the ladder is terminated at the output by letting $A_2 = - B_2$, then the "C" matrix takes the following modified form:

$$\begin{bmatrix} B_1 \\ B_2 \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix} \begin{bmatrix} A_1 \\ -B_2 \end{bmatrix} \tag{9}$$

Expanding the second equation,

$$B_2 = C_{21}A_1 - C_{22}B_2, \quad \text{or} \quad B_2/A_1 = C_{21}/(1+C_{22}) \tag{10}$$

For a continuous time LC ladder network, the terminal voltages and currents can be related by the small signal short circuit admittance parameters as follows [11]:

$$\begin{bmatrix} i_1(s) \\ i_2(s) \end{bmatrix} = \begin{bmatrix} y_{11}(s) & y_{12}(s) \\ y_{21}(s) & y_{22}(s) \end{bmatrix} \begin{bmatrix} v_1(s) \\ v_2(s) \end{bmatrix} \tag{11a}$$

If the network is singly terminated with one ohm at the output, then the small signal voltage gain is given by

$$A_v(s) = - y_{12}(s)/(1+y_{22}(s)) \tag{11b}$$

In the continuous time case the Cauer synthesis technique is as follows: Given $A_v(s) = 1/D(s)$:

(1) for the even order case,

$$y_{22}(s) = \text{Even}(D(s))/\text{Odd}(D(s))$$

$$\text{or} \quad 1/y_{22}(s) = 1/(\text{Even}(D(s))/\text{Odd}(D(s))) \tag{12}$$

(2) Perform a Cauer I expansion (for the even order case) of $1/y_{22}(s)$ to obtain the element values of the LC ladder. The resulting Cauer I expansion is of the form

$$1/y_{22}(s) = \cfrac{1}{C_{10}s + \cfrac{1}{L_9 s + \cfrac{1}{C_8 s + \ldots + \cfrac{1}{C_2 s + \cfrac{1}{L_1 s}}}}} \tag{13}$$

(3) The desired $A_v(s)$ is realized up to a multiplicative constant which can be determined by comparing the original $A_v(s)$ and the transfer function $A_v'(s)$ actually realized at any convenient frequency. DC is often the easiest. Then a multiplier $G = A_v(0)/A_v'(0)$ is attached at the output to complete the realization.

Equation (10) suggests that a similar procedure can be used for calculating the coefficients of the sampled-data ladder if B_2 is the output voltage and A_1 is the input voltage. A Cauer I expansion of $C_{22} = (B_2/A_2)|A_1 = 0$ can be put in the following form:

$$C_{22} = \cfrac{1}{\alpha_1 z + \cfrac{1}{\alpha_2 z + \ldots + \cfrac{1}{\alpha_9 z + \cfrac{1}{\alpha_{10} z}}}} \tag{14}$$

By longhand analysis of the filter in Figure 5,

$$C_{22} = \cfrac{1}{T_1 + \cfrac{1}{T_2 + \ldots + \cfrac{1}{T_9 + \cfrac{1}{T_{10}}}}} \tag{15}$$

Comparing equations (14) and (15), the sampled data ladder requires $1/T_i = z^{-1}/\alpha_i$, $i = 1$ through 10. Then the singly terminated ladder

takes the form in Figure 6.

The filter coefficients are given by $P_1 = -1/\alpha_{10}$, $P_2 = 1/\alpha_9$, ..., $P_9 = -1/\alpha_2$, $P_{10} = 1/\alpha_1$, where α_1 through α_{10} are the results of the Cauer I expansion of $C_{22} = \text{Even}(D(z))/\text{Odd}(D(z))$. $D(z) = z^{10} + h_1 z^9 + \ldots + h_9 z + h_{10}$ as given previously. The gains of the desired T(z) and the sampled data ladder at DC(z=1). $G = A_{v_{desired}}(z=1)/A_{v_{realized}}(z=1)$, where $A_{v_{desired}}(z=1) = T(z)\big|z = 1 = 1/(1+\sum_{i=1}^{10} h_i)$ and $A_{v_{realized}}(z=1)$ is obtained by a longhand analysis of the ladder at z = 1. It is found that

$$A_{v_{realized}}(z=1) = F_1F_2F_3F_4F_5F_6F_7F_8F_9F_{10} \tag{16}$$

with

$$F_1 = 1/(1+\alpha_1), \quad F_2 = 1/(F_1+\alpha_2), \ldots, \quad F_{10} = 1/(F_9+\alpha_{10}) \tag{17}$$

The program used to calculate G and P_1 through P_{10} is given in Figure 7. In the main program, the h_i's are read in to the array H(10) and arrays A and C are filled with even and odd elements of H, respectively, plus A(6) = 1 to account for the unity coefficient of z^{10} in D(z). Then the subroutine COEFFS is used to calculate the Cauer expansion of Even(D(z))/Odd(D(z)), and returns the coefficients in array ALPH. Finally the filter parameters P_1 through P_{10} are computed as discussed earlier and put in the array PARAM(10), and the gain factor GAIN is calculated in the subroutine GAINP. The algorithm used in COEFFS is similar to the Routh tabulation in stability analysis of control systems [12] and is illustrated in Figure 8, with equations in the subroutine. The first two rows of the chart are the coefficients of the even and odd

powers of z in D(z), respectively, in descending order.

The output of the program for the specific case given in Chapter 1 is shown in Figure 9. To test the synthesis algorithm, the Purdue digital simulation program DINAP was used [13]. The frequency response from 100 Hz to 4KHz is shown in Figure 10. Excellent agreement with the desired response is observed. The results of the sensitivity study made by changing the number of bits accuracy in the filter coefficients are shown in Figure 11. For the general sensitivity analysis all parameters were changed simultaneously; no attempt was made to vary only selected multipliers. To round a number p to n bits, $p \times 2^n$ was rounded to the nearest integer and divided by 2^n.

$$p|n \text{ bits} = \text{Integer}(p \times 2^n)/2^n \tag{18}$$

The number of bits required depends on the accuracy desired in the response. For a requirement of no more than 10% error on all center frequencies and Q's, 11 bits of accuracy were needed. Below 11 bits the response varied randomly and for other speech responses than the one simulated here, the trends would be different but unpredictable.

The high bit requirement severely limits the usefulness of the singly terminated ladder due to the large chip area required in fabrication plus the excessive amount of control circuitry needed to program the filter coefficients. However, the concepts developed will be useful in other algorithms.

Despite the sensitivity limitations of this circuit, it is of general interest to synthesize the filter with the DDI sections in Figure 4(d). An alternative would be a structure using Lossless

Discrete Integrator (*LDI*) sections, which has a transfer function $T(z) = z^{-1/2}/(1-z^{-1})$ and differs only in switch phasing from the DDI section in switched capacitor implementation [4]. However, the synthesis procedure for the LDI transformation is theoretically much more difficult and has not yet been developed.

The general DDI section without subtraction and multiplication included is given in Figure 12. The transfer function $V_{out}(z)/V_{in}(z) = z^{-1}/(1-z^{-1})$ is defined as a new variable ω^{-1}, so that the previous synthesis technique for ladders with pure delays can be used.

$$z^{-1}/(1-z^{-1}) = \omega^{-1}, \quad \text{or} \quad z = \omega + 1 \tag{19}$$

If z is replaced by $\omega + 1$ in T(z), all expressions are multiplied out, and the coeffieicnts of each power of ω are collected, then the original synthesis procedure can be used for $T(\omega) = T(z)\,|\,z = \omega + 1$, the transfer function of the DDI filter. No simple relation like Equation (19) can be derived for the LDI section. In that case $V_{out}(z)/V_{in}(z) = z^{-1/2}/(1-z^{-1})$ and $\omega = z^{1/2} - z^{-1/2}$, which greatly complicates the above substitution and expansion process. The general form of the DDI structure is shown in Figure 13. A different DC gain factor needs to be calculated also.

To find the coefficients of each power of the new variable ω, the following procedure is used:

$$T(\omega) = \frac{1}{(\omega+1)^{10}+h_1(\omega+1)^{9}+ \ldots + h_9(\omega+1)+h_{10}} \tag{20}$$

$$(\omega) = \frac{1}{\omega^{10}+g_1\omega^{9}+ \ldots + g_9\omega+g_{10}} \tag{21}$$

By long multiplication and collection of powers of ω,

$$
\begin{aligned}
g_1 &= 10 + h_1 \\
g_2 &= 45 + 9h_1 + h_2 \\
g_3 &= 120 + 36h_1 + 8h_2 + h_3 \\
g_4 &= 210 + 84h_1 + 28h_2 + 7h_3 + h_4 \\
g_5 &= 252 + 126h_1 + 56h_2 + 21h_3 + 6h_4 + h_5 \\
g_6 &= 210 + 126h_1 + 70h_2 + 35h_3 + 15h_4 + 5h_5 + h_6 \\
g_7 &= 120 + 84h_1 + 56h_2 + 35h_3 + 20h_4 + 10h_5 + 4h_6 + h_7 \\
g_8 &= 45 + 36h_1 + 28h_2 + 21h_3 + 15h_4 + 10h_5 + 6h_6 + 3h_7 + h_8 \\
g_9 &= 10 + 9h_1 + 8h_2 + 7h_3 + 6h_4 + 5h_5 + 4h_6 + 3h_7 + 2h_8 + h_9 \\
g_{10} &= 1 + h_1 + h_2 + h_3 + h_4 + h_5 + h_6 + h_7 + h_8 + h_9 + h_{10}
\end{aligned}
\qquad (22)
$$

The computer program used to calculate the DDI filter coefficients and gain multiplier is given in Figure 14. The same subroutine COEFFS is used to calculate the Cauer I expansion and was given in Figure 7. The arrays A and C are modified with the relations of Equation (22), however. The gain factor is calculated in subroutine GAIND. The DC gain desired is $T(z)\big|z = 1 = 1/(1+\sum_{i=1}^{10} h_i)$ and the DC gain of the DDI filter in Figure 13 is seen to be one, so the gain factor is $G = 1/(1+\sum_{i=1}^{10} h_i)$. The output of the DDI calculation program is in Figure 15.

The DINAP simulation with no rounding of coefficients is shown in Figure 16. Again the agreement with the ideal response is nearly perfect. The results of the sensitivity study are in Figure 17. Overall the results are worse than those of the filter with simple delays. For 10% accuracy on all center frequencies and Q's, 13 bits of accuracy are required. Below 13 bits the variations are again random, and in several

instances one or more of the resonances even disappear. Clearly the DDI singly terminated ladder is not a feasible solution to the problem either.

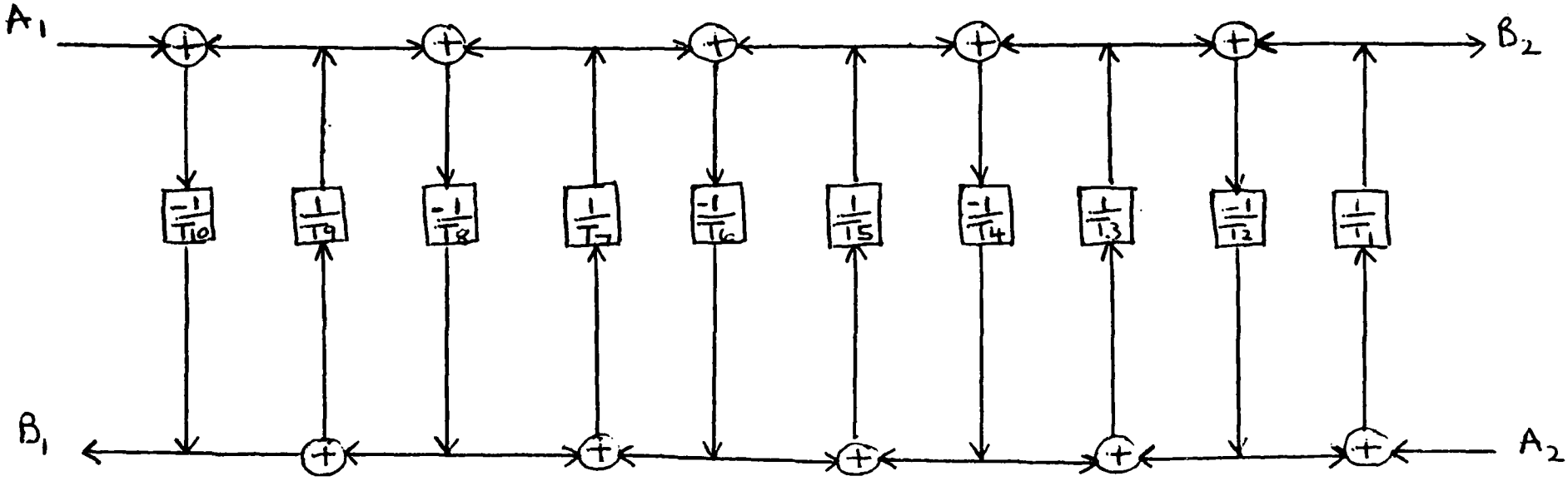


Figure 5: General form of the unterminated ladder

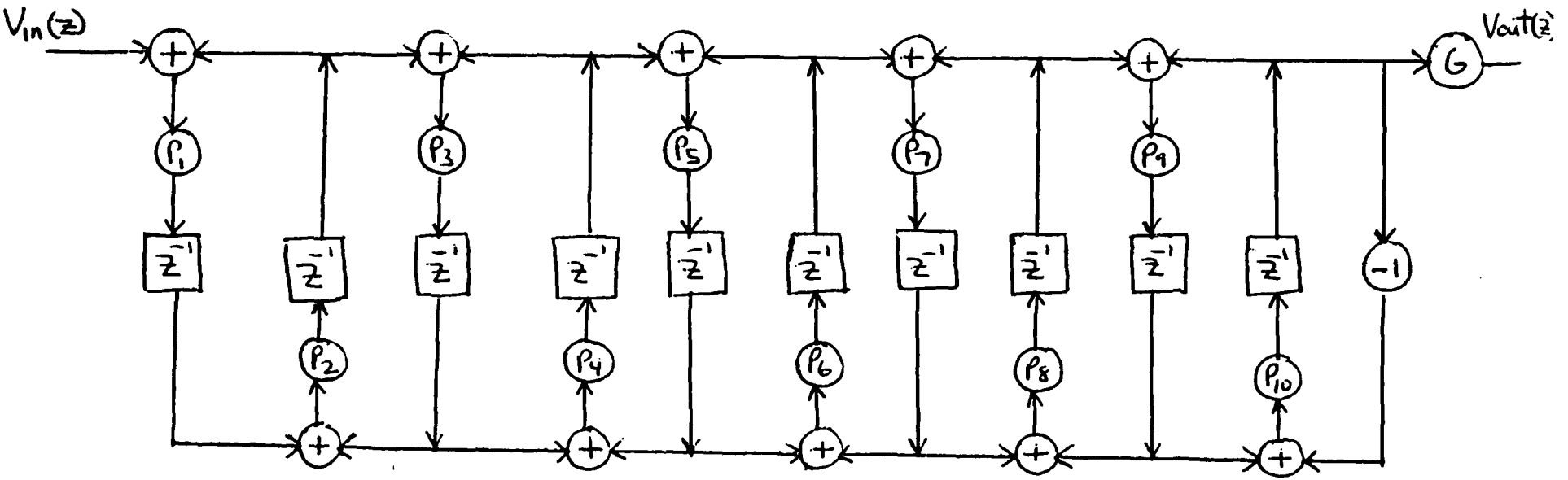


Figure 6: Singly terminated digital ladder network

```
      DIMENSION A(6),C(5),ALPH(10),PARAM(10),H(10)
      DO 1 I=1,10
    1 READ 2,H(I)
    2 FORMAT(F5.0)
      A(6)=1
      A(5)=H(2)
      A(4)=H(4)
      A(3)=H(6)
      A(2)=H(8)
      A(1)=H(10)
      C(5)=H(1)
      C(4)=H(3)
      C(3)=H(5)
      C(2)=H(7)
      C(1)=H(9)
      CALL COEFFS(A,C,ALPH)
      PARAM(1)=-1/ALPH(10)
      PARAM(2)=1/ALPH(9)
      PARAM(3)=-1/ALPH(8)
      PARAM(4)=1/ALPH(7)
      PARAM(5)=-1/ALPH(6)
      PARAM(6)=1/ALPH(5)
      PARAM(7)=-1/ALPH(4)
      PARAM(8)=1/ALPH(3)
      PARAM(9)=-1/ALPH(2)
      PARAM(10)=1/ALPH(1)
      CALL GAINP(ALPH,H,GAIN)
      PRINT5
    5 FORMAT('0','LADDER PARAMETERS 1 THRU 10')
      DO 3 I=1,10
    3 PRINT 4,PARAM(I)
    4 FORMAT(' ',E17.10)
      PRINT 6,GAIN
    6 FORMAT('0','GAIN FACTOR=',E17.10)
      END
```

Figure 7: Calculation program for singly terminated ladder

```
SUBROUTINE [illegible]
DIMENSION Q(5),P(4),SS(4),TT(3),U(3),VV(3),W(2),AA(6),CC(5),ALPHA
1(10)
Q(1)=((CC(5)+AA(5))-(AA(6)*R(4)))/CC(5)
Q(2)=((CC(5)+AA(4))-(AA(6)*CC(4)))/CC(5)
Q(3)=((CC(5)+AA(3))-(AA(6)*CC(3)))/CC(5)
Q(4)=((CC(5)+AA(2))-(AA(6)*CC(2)))/CC(5)
Q(5)=AA(1)
R(1)=((Q(1)*CC(4))-(CC(5)*Q(2)))/Q(1)
P(2)=((Q(1)*CC(3))-(CC(5)*Q(3)))/Q(1)
P(3)=((Q(1)*CC(2))-(CC(5)*Q(4)))/Q(1)
R(4)=((Q(1)*CC(1))-(CC(5)*AA(5)))/Q(1)
SS(1)=((P(1)*Q(2))-Q(1)*P(2))/P(1)
SS(2)=((R(1)*Q(3))-Q(1)*P(3))/P(1)
SS(3)=((R(1)*Q(4))-Q(1)*P(4))/P(1)
SS(4)=Q(5)
TT(1)=((SS(1)*P(2))-P(1)*SS(2))/SS(1)
TT(2)=((SS(1)*P(3))-(P(1)*SS(3)))/SS(1)
TT(3)=((SS(1)*R(4))-(P(1)*SS(4)))/SS(1)
U(1)=((TT(1)*SS(2))-(SS(1)*TT(2)))/TT(1)
U(2)=((TT(1)*SS(3))-(SS(1)*TT(3)))/TT(1)
U(3)=SS(4)
VV(1)=((U(1)*TT(2))-(TT(1)*U(2)))/U(1)
VV(2)=((U(1)*TT(3))-(TT(1)*U(3)))/U(1)
W(1)=((VV(1)*U(2))-(U(1)*VV(2)))/VV(1)
W(2)=U(3)
X=((W(1)*VV(2))-(VV(1)*W(2)))/W(1)
Y=W(2)
ALPHA(1)=AA(6)/CC(5)
ALPHA(2)=CC(5)/Q(1)
ALPHA(3)=Q(1)/P(1)
ALPHA(4)=R(1)/SS(1)
ALPHA(5)=SS(1)/TT(1)
ALPHA(6)=TT(1)/U(1)
ALPHA(7)=U(1)/VV(1)
ALPHA(8)=VV(1)/W(1)
ALPHA(9)=W(1)/X
ALPHA(10)=X/Y
RETURN
END
SUBROUTINE GAINP(ALP,HH,GAINN)
DIMENSION ALP(10),HH(10)
F1=1./(1+ALP(1))
F2=1./(F1+ALP(2))
F3=1./(F2+ALP(3))
F4=1./(F3+ALP(4))
F5=1./(F4+ALP(5))
F6=1./(F5+ALP(6))
F7=1./(F6+ALP(7))
F8=1./(F7+ALP(8))
F9=1./(F8+ALP(9))
F10=1./(F9+ALP(10))
CKTGN=F1*F2*F3*F4*F5*F6*F7*F8*F9*F10
CCTGN=1./(1+HH(1)+HH(2)+HH(3)+HH(4)+HH(5)+HH(6)+HH(7)+HH(8)+HH(9)+
1HH(10))
GAINN=CCTGN/CKTGN
RETURN
END
```

Figure 7: (cont)

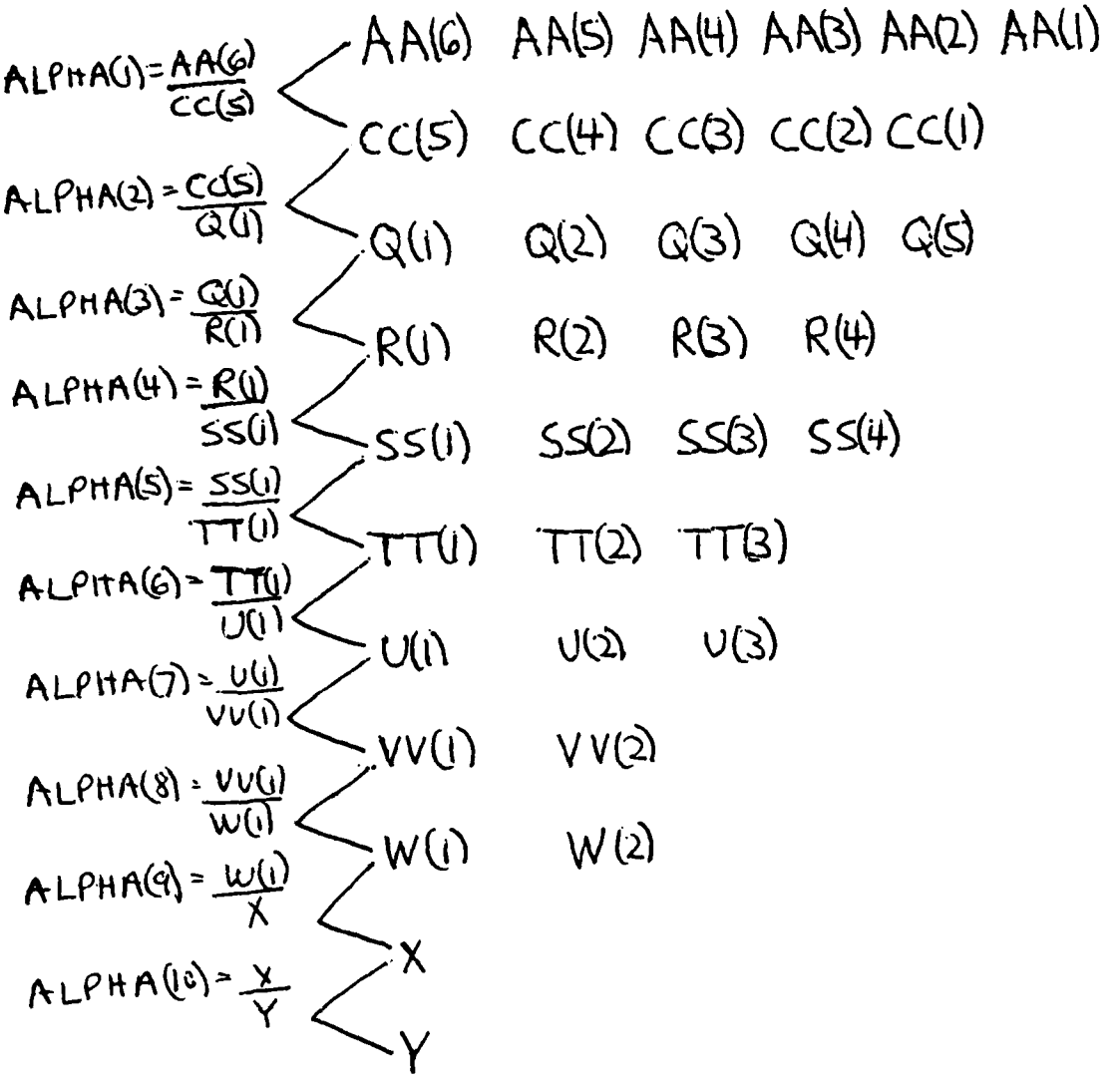


Figure 8 : Calculation of Cauer expansion coefficients

```
LADDER PARAMETERS 1 THRU 10
 -.6366356513E+01
  .5931981029E-01
 -.1641201614E+01
  .7402732652E-01
 -.2468311652E+03
 -.7353814231E-01
  .2011142906E+01
  .2841623314E-01
  .2217341837E+03
 -.2800000000E-01

GAIN FACTOR=  .3304094755E+01
```

Figure 9: filter coefficients for singly terminated ladder

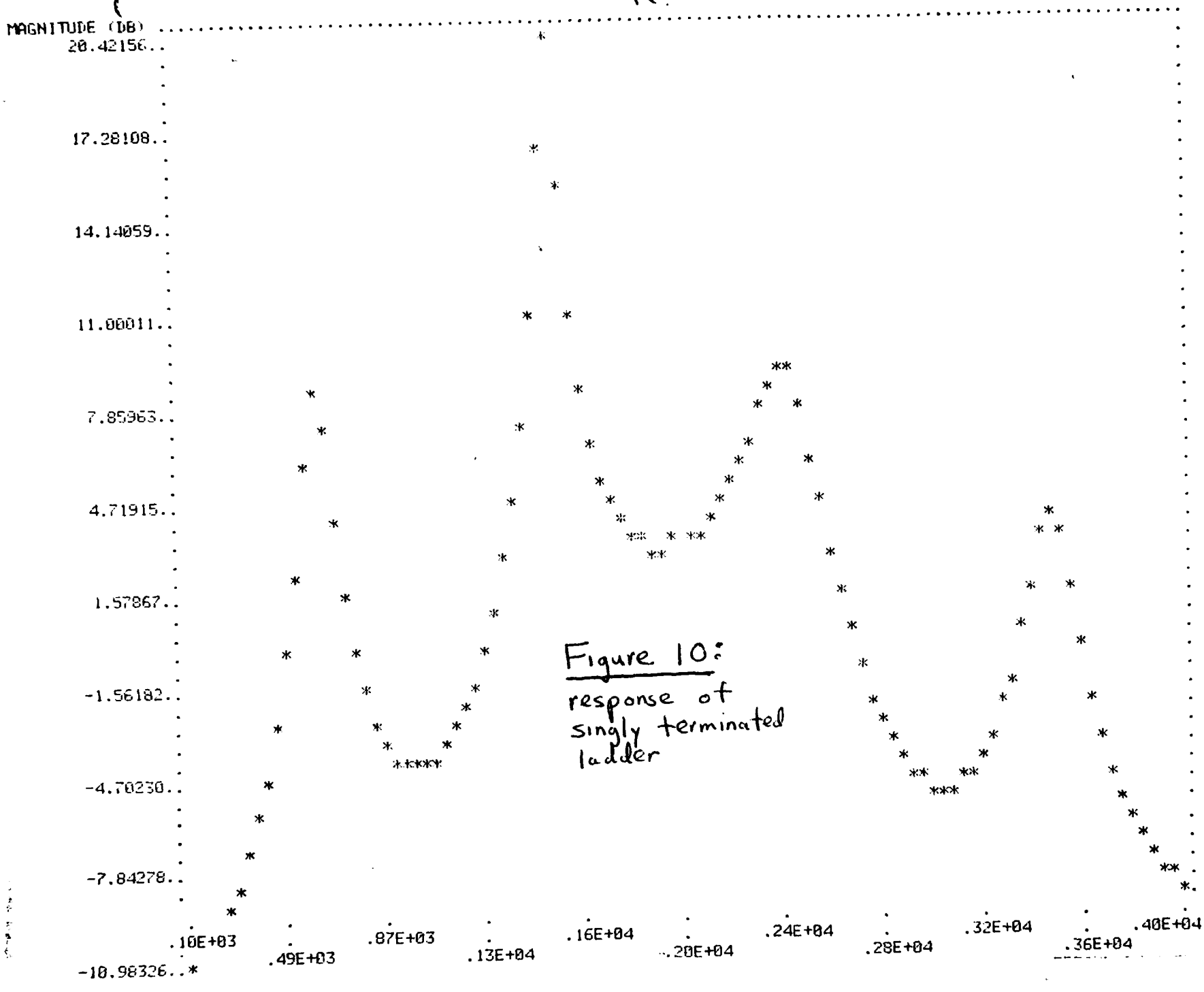


Figure 10:
response of singly terminated ladder

Precision on Coeffs	F_1	Q_1	F_2	Q_2	F_3	Q_3	F_4	Q_4
Full Precision	607	5.2	1543	29.7	2401	8.4	3454	19.0
12 bits	607	5.2	1543	31.0	2401	8.2	3454	18.6
11 bits	607	5.0	1543	27.8	2401	8.6	3454	17.2
10 bits	646	4.6	1582	15.4	2401	14.2	3415	10.3
9 bits	529	16.8	1465	7.9	2479	3.1	3571	28.0
8 bits	646	8.7	1543	16.1	2401	5.2	3415	20.7
7 bits	607	17.7	1543	4.5	2362	6.8	3493	34.2
6 bits	607	13.3	1543	34.2	2401	15.5	3454	44.0
5 bits	568	4.9	1582	20.1	2323	5.6	3493	16.5

Figure 11.: Results of sensitivity study of singly terminated ladder

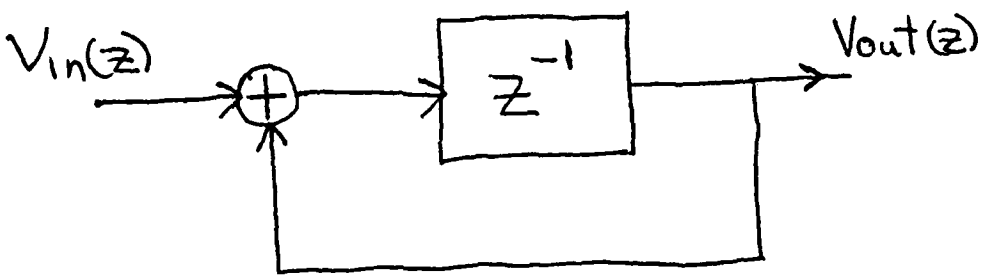


Figure 12: the general DDI section

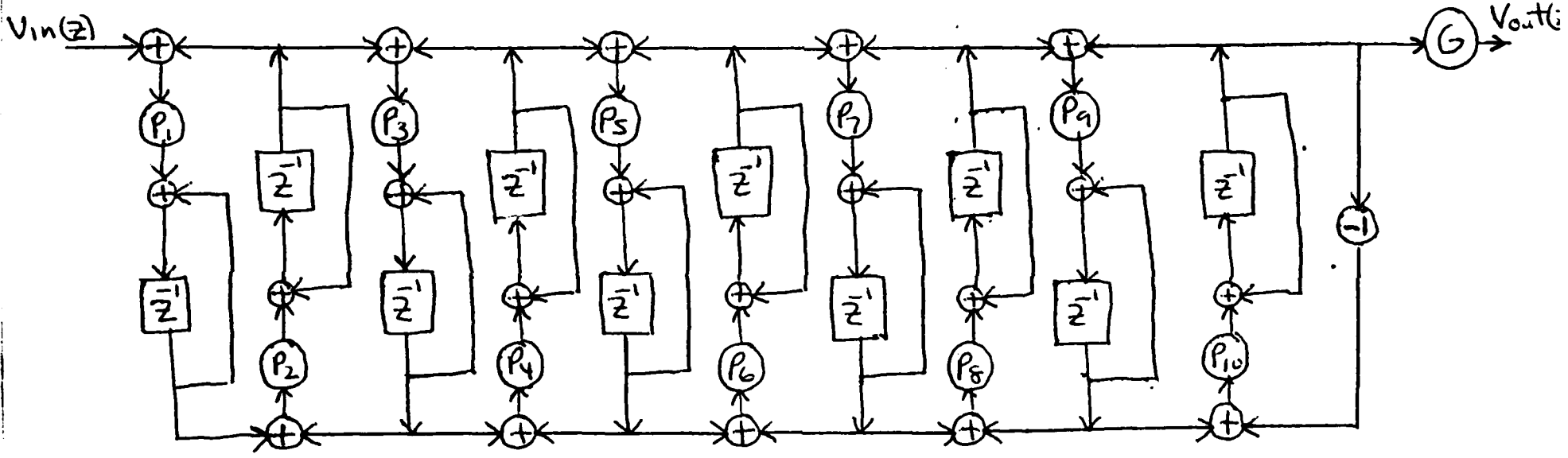


Figure 13: DDI singly terminated ladder structure

```
      DIMENSION A(6),C(5),ALPH(10),H(10),PARAM(10)
      DO 1 I=1,10
    1 READ2,H(I)
    2 FORMAT(F5.0)
      A(6)=1.0
      A(5)=45+9*H(1)+H(2)
      A(4)=210+84*H(1)+28*H(2)+7*H(3)+H(4)
      A(3)=210+126*H(1)+70*H(2)+35*H(3)+15*H(4)+5*H(5)+H(6)
      A(2)=45+36*H(1)+28*H(2)+21*H(3)+15*H(4)+10*H(5)+6*H(6)+3*H(7)++
      A(1)=1+H(1)+H(2)+H(3)+H(4)+H(5)+H(6)+H(7)+H(8)+H(9)+H(10)
      C(5)=10+H(1)
      C(4)=120+36*H(1)+8*H(2)+H(3)
      C(3)=252+126*H(1)+56*H(2)+21*H(3)+6*H(4)+H(5)
      C(2)=120+84*H(1)+56*H(2)+35*H(3)+20*H(4)+10*H(5)+4*H(6)+H(7)
      C(1)=10+9*H(1)+8*H(2)+7*H(3)+6*H(4)+5*H(5)+4*H(6)+3*H(7)+2*H(8)
     1H(9)
      CALL COEFFS(A,C,ALPH)
      PARAM(1)=-1/ALPH(10)
      PARAM(2)=1/ALPH(9)
      PARAM(3)=-1/ALPH(8)
      PARAM(4)=1/ALPH(7)
      PARAM(5)=-1/ALPH(6)
      PARAM(6)=1/ALPH(5)
      PARAM(7)=-1/ALPH(4)
      PARAM(8)=1/ALPH(3)
      PARAM(9)=-1/ALPH(2)
      PARAM(10)=1/ALPH(1)
      CALL GAIND(H,GAIN)
      PRINT5
    5 FORMAT('0','LADDER PARAMETERS 1 THRU 10')
      DO 3 I=1,10
    3 PRINT 4,PARAM(I)
    4 FORMAT(' ',E17.10)
      PRINT 6,GAIN
    6 FORMAT('0','GAIN FACTOR=',E17.10)
      END

      SUBROUTINE COEFFS(AA,CC,ALPHA) Given in figure 7

      SUBROUTINE GAIND(HH,GAINN)
      DIMENSION HH(10)
      CKTGN=1.0
      ACTGN=1/(1+HH(1)+HH(2)+HH(3)+HH(4)+HH(5)+HH(6)+HH(7)+HH(8)+HH(9)
     1HH(10))
      GAINN=ACTGN/CKTGN
      RETURN
      END
```

Figure 14: calculation program for DDI singly terminated ladder

```
LADDER PARAMETERS 1 THRU 10
 -.6191760035E+00
  .2431757530E+00
 -.5145078412E+00
  .6427863050E+00
 -.8106862333E+00
  .1032806464E+01
 -.1369362044E+01
  .1956687387E+01
 -.3303399301E+01
  .9972000000E+01

GAIN FACTOR=  .2717391304E+00
```

Figure 15: filter coefficients for DDI singly terminated ladder

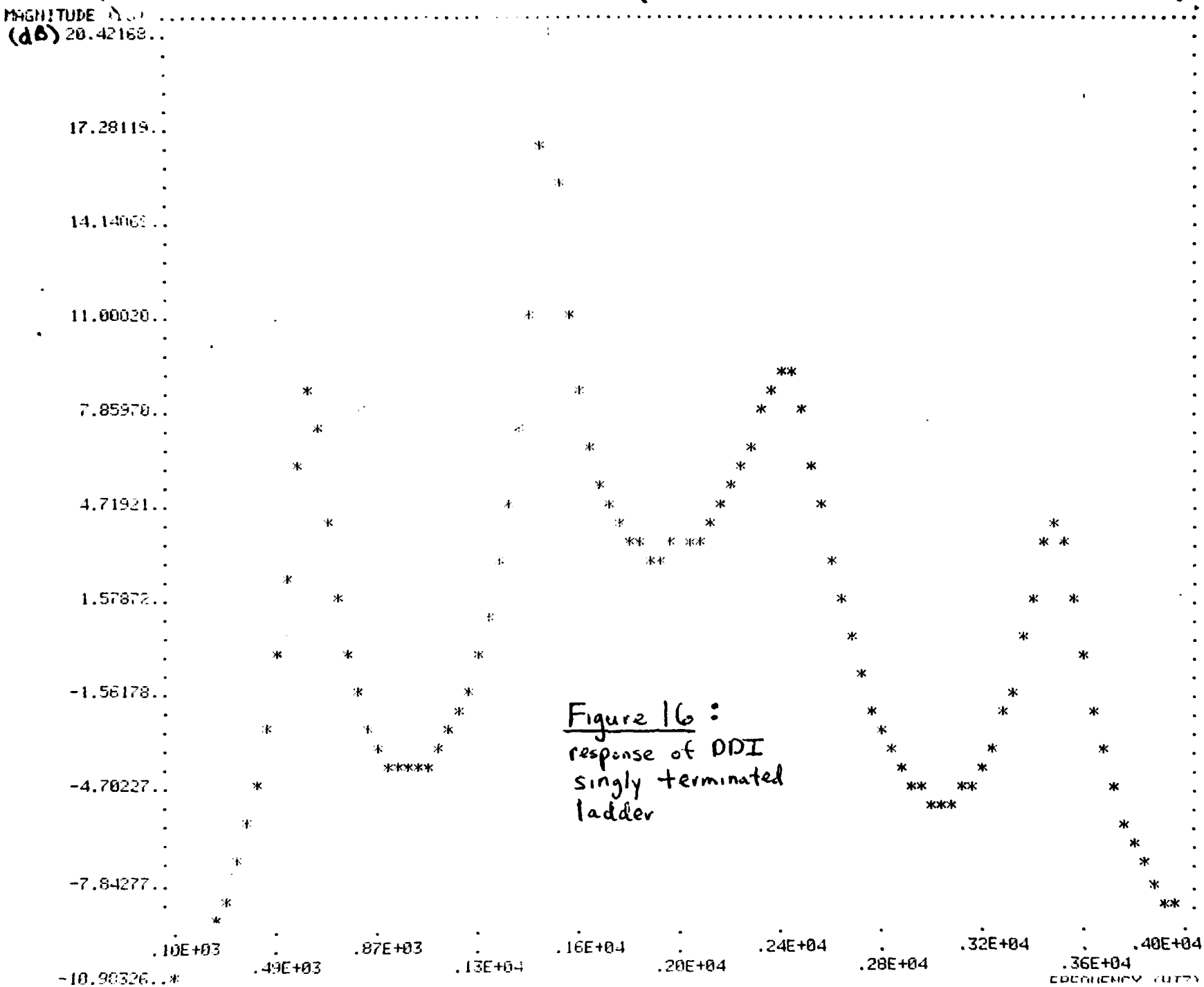


Figure 16 :
response of DDI
singly terminated
ladder

Precision on Coeffs	F_1	Q_1	F_2	Q_2	F_3	Q_3	F_4	Q_4
Full Precision	607	5.2	1543	29.7	2401	8.4	3454	19.0
14 bits	607	5.2	1543	29.7	2401	8.4	3454	19.0
13 bits	607	5.1	1543	28.7	2401	7.7	3454	**18.0**
12 bits	607	5.0	1543	36.6	2440	10.3	3454	17.4
11 bits	607	5.1	1543	24.1	2479	6.1	3532	8.2
10 bits	607	5.1	1543	15.5	none		2713	5.5
9 bits	607	3.6	1543	9.3	none		2635	10.0
8 bits	607	2.6	1543	15.0	2284	3.4	3259	.87
7 bits	607	4.9	1543	36.1	2089	4.8	none	
6 bits	646	4.7	1504	19.9	none		3571	1.3
5 bits	646	7.3	1543	5.2	none		none	
4 bits	607	6.3	1231	3.0	none		none	

Figure 17: Results of sensitivity study of ODI singly terminated ladder

Chapter 3 - The Lattice Filter

Another filter structure with a simple synthesis procedure is the lattice filter with reflection coefficients as its multipliers, classically known for its good sensitivity properties. Using a simple recursive formula, ten reflection coefficients r_1 through r_{10} can be derived from the predictor coefficients h_1 through h_{10}. There are many forms of lattice filters, but one of the simplest, requiring only ten multipliers to realize ten poles, is discussed here [6], [7].

Physically the reflection coefficients characterize the behavior of the acoustic traveling wave in the human vocal tract, but with proper interpretation, the synthesis algorithm to be described can be applied to any tenth-order all pole z-domain transfer function P(z). If the coefficients of each power of z are collected so that P(z) has the form

$$P(z) = \frac{1}{z^{10}+d_1 z^9 + \ldots + d_9 z + d_{10}} \tag{23}$$

then d_1 through d_{10} can be interpreted as predictor coefficients and the corresponding lattice filter realization can be derived.

The reflection coefficients r_i can be calculated from the predictor coefficients h_i by the following procedure:

$$r_i = h_i^i$$

$$h_j^{i-1} = (h_j^i - h_i^i h_{i-j}^i)/(1-r_i^2) \tag{24}$$

for j = 1 to i - 1 while i = 10 decreasing to 1, where h_i^{10} is the predictor coefficient h_i. The computer program for the calculation is given in Figure 18. A(10,10) is used for reading in the predictor coefficients rather than H, and A(I,10), i = 1 through 10, are the spaces used. The gain factor of the filter is also calculated as will be discussed later.

The total ten pole lattice filter is realized by a cascade of ten sections, each characterized by one reflection coefficient. One of the simplest sections, requiring only one multiplier, is represented in the z-domain as shown in Figure 19. For the first and last sections the structure must be modified. The input section, characterized by r_{10}, is shown in Figure 20. The bottom left connection is left open since there are no stages to the left of the first stage. At the last stage the proper connection is as given in Figure 21. Physical arguments govern such an arrangement and are discussed in [6], but the same termination applies to any transfer function with the denominator coefficients interpreted as predictor coefficients.

The total filter is shown in Figure 22. By hand analysis it is seen that the DC(z=1) gain is one (without the gain multiplier at output), and the desired gain is $1/(1+\sum_{i=1}^{10} h_i)$. Thus $G = 1/(1+\sum_{i=1}^{10} h_i)$ and is calculated in the program of Figure 18. The computer output giving the reflection coefficients and gain factor is shown in Figure 23. The DINAP simulation with no parameter errors is in Figure 24, and the results of the sensitivity study are given in Figure 25. With no coefficient errors, the agreement of the lattice filter response with the desired response is excellent.

The sensitivity properties of the lattice filter are much better than those of the singly terminated ladder, as only 7 bits of accuracy are required to assure center frequencies and Q's within 10% of their ideal values. Another important observation is that the response does not degrade as severely as that of the singly terminated ladder for lower number of bits, and even for 4 bits and less the response is still reasonable. The center frequencies are especially insensitive, staying constant down to 4 bits accuracy.

If a DDI implementation were desired, one might attempt to use the same transformation $z = \omega + 1$ as for the singly terminated ladder. The new predictor coefficients used to find the modified reflection coefficients would be calculated by collecting the coefficients of each power of ω as done for the DDI ladder. In Figure 26 the array elements B(I,10), i = 1 through 10, contain the modified predictor coefficients, and Figure 27 shows the new reflection coefficients. To implement the DDI lattice filter, each delay in the original lattice is replaced by a DDI section.

The DINAP simulation with no coeffient errors is shown in Figure 28. The response is not as expected, and although the four resonances are still present, they are not sharp and the Q's are severely degraded. It appears that the use of DDI sections causes a multiple zero at DC, and the DDI lattice filter is not acceptable.

```
      DIMENSION A(10,10),R(10)
      DO 1 I=1,10
    1 READ2,A(I,10)
    2 FORMAT(F5.0)
      DO 3 I=1,10
      K=11-I
      R(K)=A(K,K)
      IF(K.EQ.1)GO TO 5
      DO 4 J=1,K-1
    4 A(J,K-1)=(A(J,K)-A(K,K)*A(K-J,K))/(1-R(K)**2)
    3 CONTINUE
    5 CONTINUE
      PRINT8
    8 FORMAT('0','FILTER PARAMETERS 1 THRU 10')
      DO 6 I=1,10
    6 PRINT7,R(11-I)
    7 FORMAT(' ',E17.10)
      GAIN=1/(1+A(1,10)+A(2,10)+A(3,10)+A(4,10)+A(5,10)+A(6,10)+A(7,10)
     1+A(8,10)+A(9,10)+A(10,10))
      PRINT9,GAIN
    9 FORMAT('0','GAIN FACTOR=',E17.10)
      END
```

Figure 18: Reflection coefficient calculation

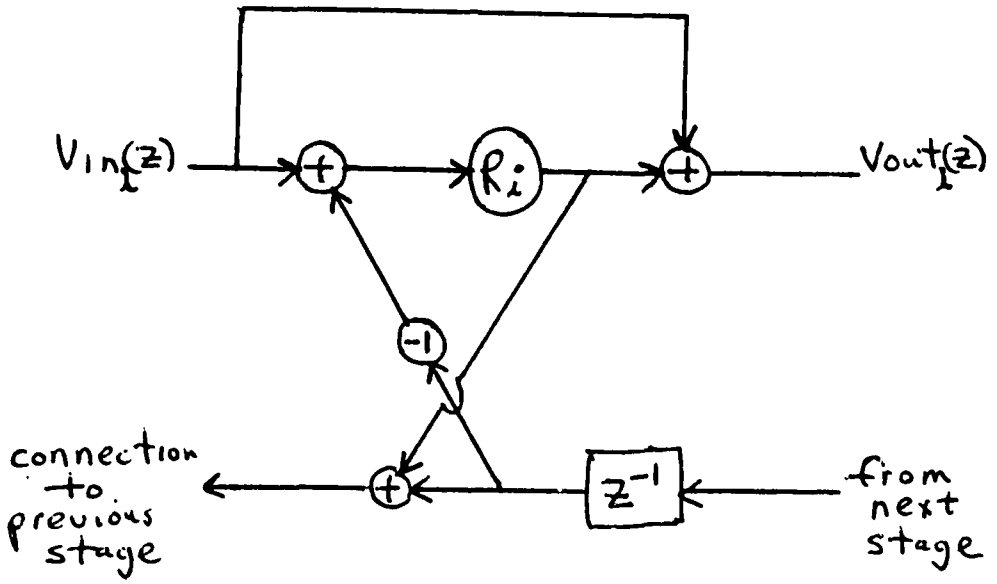


Figure 19 : lattice filter section

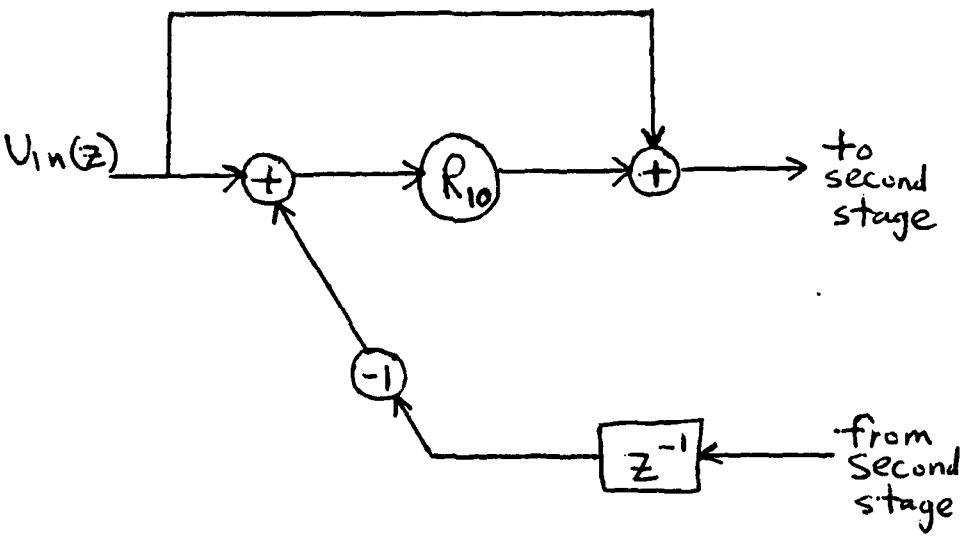


Figure 20: first stage of lattice filter

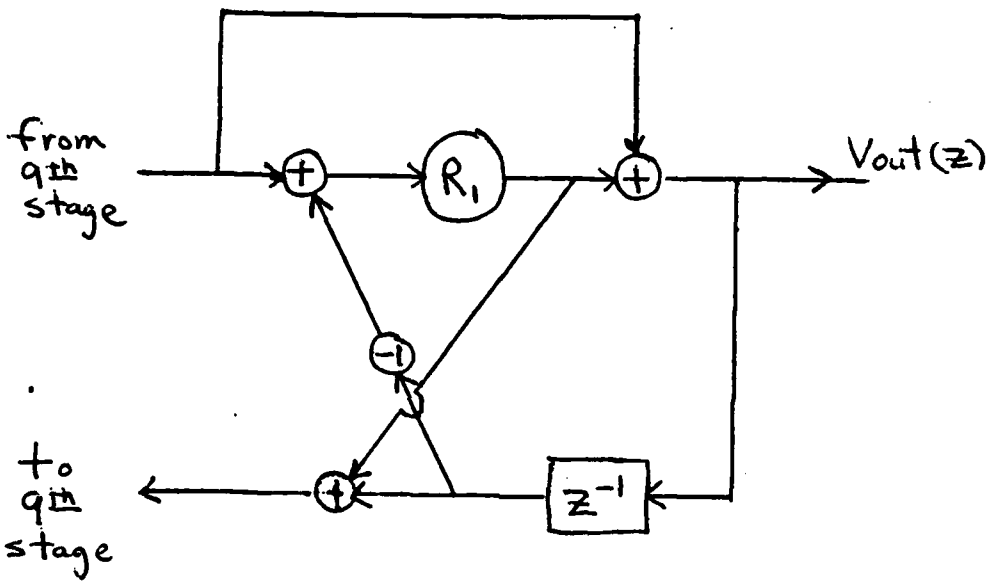


Figure 21: last stage of lattice filter

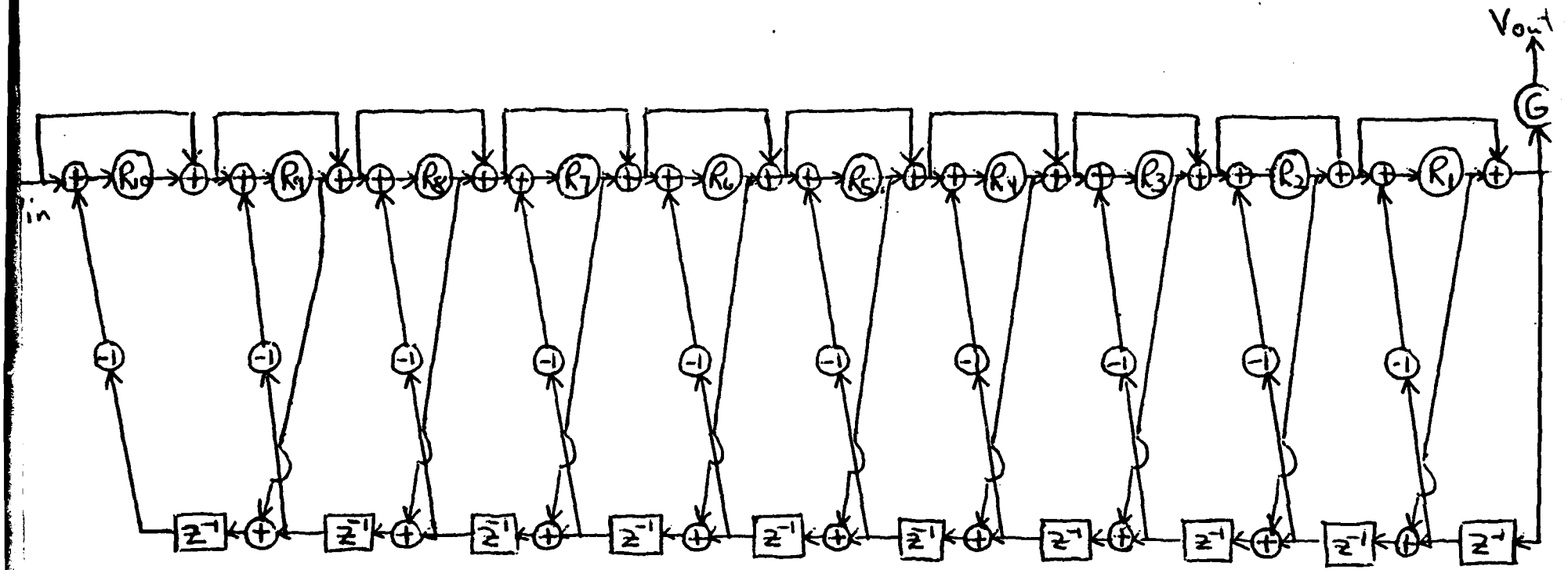


Figure 22: complete lattice filter

```
FILTER PARAMETERS 1 THRU 10
  .2955000000E+00
  .9606217296E-01
  .6023220283E+00
  .1295542772E+00
 -.1669097966E+00
 -.8307427828E-01
  .1781659174E+00
  .1968960618E+00
  .6650700451E+00
 -.2016437681E+00

GAIN FACTOR=  .2717391304E+00
```

Figure 23: lattice filter coefficients

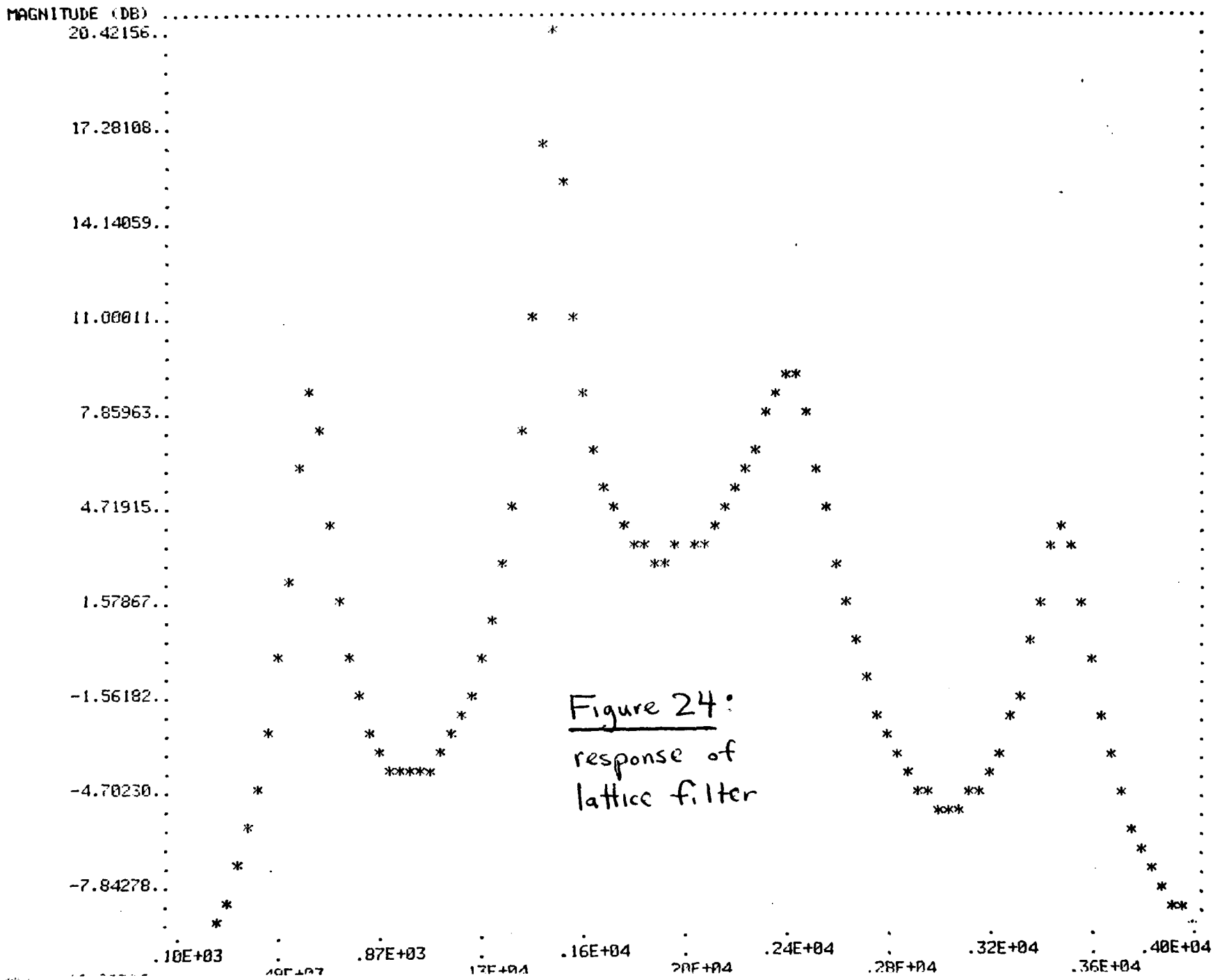


Figure 24:
response of
lattice filter

Precision on Coeffs	F_1	Q_1	F_2	Q_2	F_3	Q_3	F_4	Q_4
Full Precision	607	5.2	1543	29.7	2401	8.4	3454	19.0
7 bits	607	5.2	1543	29.7	2401	8.4	3454	19.0
6 bits	607	5.9	1543	29.1	2401	7.3	3454	18.2
5 bits	607	6.0	1543	28.9	2401	6.3	3454	15.1
4 bits	607	5.0	1543	28.1	2401	6.3	3454	17.8
3 bits	607	5.9	1504	34.7	2401	5.3	3454	16.5
2 bits	685	3.5	1543	14.3	2401	11.3	3532	8.8

Figure 25: Results of sensitivity study of lattice filter

```
      DIMENSION A(10),B(10,10),P(10)
      DO 1 I=1,10
    1 READ2,A(I)
    2 FORMAT(F5.0)
      B(1,10)=10+A(1)
      B(2,10)=45+9*A(1)+A(2)
      B(3,10)=120+36*A(1)+8*A(2)+A(3)
      B(4,10)=210+84*A(1)+28*A(2)+7*A(3)+A(4)
      B(5,10)=252+126*A(1)+56*A(2)+21*A(3)+6*A(4)+A(5)
      B(6,10)=210+126*A(1)+70*A(2)+35*A(3)+15*A(4)+5*A(5)+A(6)
      B(7,10)=120+84*A(1)+56*A(2)+35*A(3)+20*A(4)+10*A(5)+4*A(6)+A(7)
      B(8,10)=45+36*A(1)+28*A(2)+21*A(3)+15*A(4)+10*A(5)+6*A(6)+3*A(7)+
     1A(8)
      B(9,10)=10+9*A(1)+8*A(2)+7*A(3)+6*A(4)+5*A(5)+4*A(6)+3*A(7)+2*
     1A(8)+A(9)
      B(10,10)=1+A(1)+A(2)+A(3)+A(4)+A(5)+A(6)+A(7)+A(8)+A(9)+A(10)
      DO 3 I=1,10
      K=11-I
      P(K)=B(K,K)
      IF(K.EQ.1)GO TO 5
      DO 4 J=1,K-1
    4 B(J,K-1)=(B(J,K)-B(K,K)*B(K-J,K))/(1-P(K)**2)
    3 CONTINUE
    5 CONTINUE
      PRINT8
    8 FORMAT('0','FILTER PARAMETERS 1 THRU 10')
      DO 6 I=1,10
    6 PRINT7,P(11-I)
    7 FORMAT(' ',E17.10)
      END
```

Figure 26: DDI reflection coefficient calculation

FILTER PARAMETERS 1 THRU 10
.3680000000E+01
.1216701748E+01
-.1409209059E+01
-.7231364899E+01
.1061986966E+01
.5900134237E+00
.8099637357E+00
.8378027005E+00
.9030130348E+00
.8498641734E+00

Figure 27: DDI
reflection coefficients

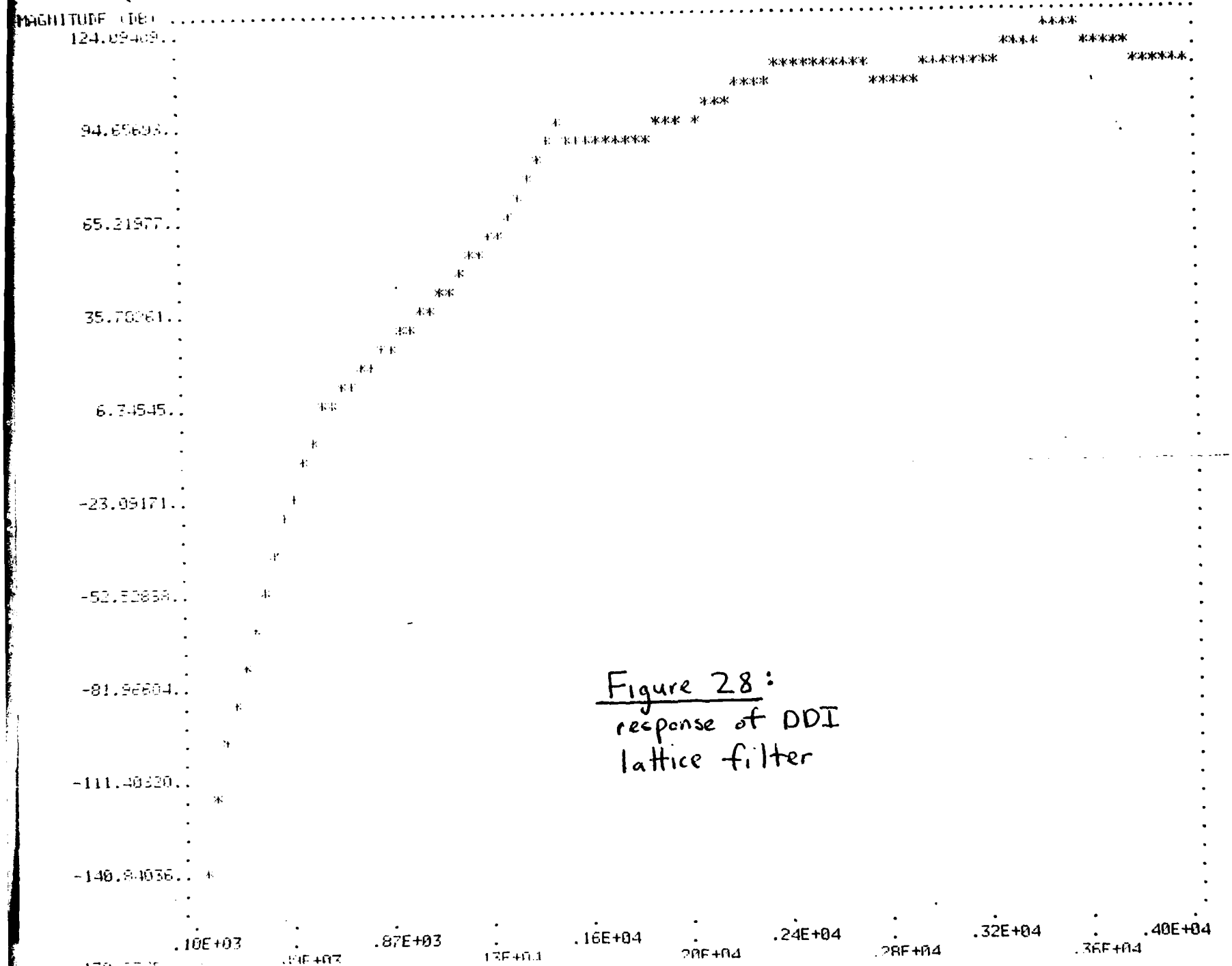


Figure 28:
response of DDI
lattice filter

Chapter 4 - Pole Transformation for DDI and LDI Implementation

The last alternative to be discussed for synthesizing T(z) is transforming the z-plane poles of T(z) to s-plane poles and constructing an equivalent continuous time filter which in turn is implemented using sampled data filter techniques. The procedure and limitation of the method will be discussed.

To find the poles of T(z), the equation

$$z^{10}+h_1z^9+\ldots+h_9z+h_{10} = 0 \tag{25}$$

must be solved for its ten complex roots. Using equations derived in Hostica's paper [3], the five corresponding continuous time center frequencies and Q's are calculated as follows, where R and θ are the radius and angle from the origin of each z-plane pole. Since the z-plane poles come in complex conjugate pairs, only one of each pair is needed for calculation:

$$f_0 = (f_s/2\pi)\sqrt{\theta^2+(\ell nR)^2} \tag{26}$$

$$Q = (-\pi f_0)/(f_s \ell nR) \tag{27}$$

The equivalent continuous time transfer function T(s), valid up to a gain constant, is given by

$$T(s) = \prod_{i=1}^{5} \frac{1}{s^2+(\frac{2\pi f_{0_i}}{Q_i})s+(2\pi f_{0_i})^2} \tag{28}$$

After carrying out the multiplication

$$T(s) = \frac{1}{s_{11}s^{10}+s_{10}s^9+\ldots+s_2s+s_1} \tag{29}$$

Standard s-domain procedures can be used to synthesize the continuous time filter. A singly terminated LC ladder, shown in Figure 29, is

chosen here because of its simple synthesis algorithm and the presence of design techniques for switched capacitor implementation of sampled data ladder filters. The basic algorithm for finding the L and C values has already been introduced in chapter 2 and is summarized as follows, with the basic ladder network now characterized by its open circuit impedance parameters [11]:

$$\begin{bmatrix} v_1(s) \\ v_2(s) \end{bmatrix} = \begin{bmatrix} z_{11}(s) & z_{12}(s) \\ z_{21}(s) & z_{22}(s) \end{bmatrix} \begin{bmatrix} i_1(s) \\ i_2(s) \end{bmatrix} \tag{30}$$

Now the small signal voltage gain is given by

$$A_v(s) = z_{12}(s)/(1+z_{11}(s)) \tag{31}$$

(1) Divide the denominator of T(s) into even and odd parts

$$T(s) = 1/(D_{even}(s)+D_{odd}(s)) \tag{32}$$

(2) Put T(s) in the form

$$T(s) = z_{12}(s)/(1+z_{11}(s)) = (1/D_{odd(s)})/(1+D_{even}(s)/D_{odd}(s)) \tag{33}$$

(3) With a one ohm source resistance, $z_{11}(s)$ is the impedance looking to the right of the resistor, and the L and C values are found by a Cauer I expansion of $z_{11}(s) = D_{even}(s)/D_{odd}(s)$. The form of the expansion is

$$z_{11}(s) = D_{even}(s)/D_{odd}(s) = L_1 s + \cfrac{1}{C_2 s + \cfrac{1}{L_3 s + \ldots + \cfrac{1}{L_9 s + \cfrac{1}{C_{10} s}}}} \tag{35}$$

(4) By inspection of Figure 29, the DC gain of the continuous time ladder is one, so to make the DC level the same as T(s), a voltage divider of value $G = 1/(1+\sum_{i=1}^{10} h_i)$ is needed at the output, as shown in Figure 30. It is important to note that the gain adjusting network must not load the LC ladder or else the basic frequency response will be changed. This requirement is satisfied in a switched capacitor

implementation if the op amps are nearly ideal.

The computer program used to calculate the LC ladder parameters is given in Figure 31(a), with subroutines CALC and COEFFS in Figure 31(b) and the complex root finder PROOT in Figure 31(c). The constant 1.0 (with multiplies the z^{10} term in equation 25) followed by the predictor coefficients h_i are read into the array T (since H is used in PROOT) and reversed in order in array A for input to PROOT. A(1) is the coefficient of the constant term of the polynomial to be solved, while A(11) multiplies the z^{10} term. The other parameters of PROOT are as follows: N is the degree of the polynomial to be solved. NPLUS2 = N+2

U and V are arrays containing the real and imaginary parts of the solution. Dimensions of U and V are N+2, but only elements N and lower contain useful information.

H, B, and C are arrays of dimension N+2 used for temporary storage.

CONV is the difference required to each root in successive iterations for the root to be considered final.

Next the relations of equations 26 and 27 are used to calculate the five continuous time center frequencies and Q's. Subroutine CALC multiples out the denominator of T(s) and returns the coefficients of each power of s in the array S, with S(11) being the coefficient of s^{10} and S(1) that of the constant term. Arrays AI and CI collect the coefficients of the even and odd powers of s, and subroutine COEFFS is again used to do the Cauer I expansion. The L and C values are returned in the array ALPH, with L_1 = ALPH(1) and C_{10} = ALPH(10). In Figure 31(c), the gain adjusting resistor $R = 1/(1+\sum_{i=1}^{10} h_i)$ is calculated. In the program T(2) through T(11) are h_i through h_{10}.

The computer output of the program is in Figure 32, and the SPICE simulation is given in Figure 33. From Figure 32 it is seen that two adjacent complex pole pairs form the third visible resonance. As is

characteristic property of continuous time filters, the correct resonances are realized, but the amplitude of the response falls off much faster than that of the discrete time response to be duplicated. It is clear that a discrete time implementation of the continuous time filter must be used to raise the levels of the higher frequency poles.

The first steps in the synthesis of the DDI or LDI implementation of the continuous time LC ladder are writing the describing equations of the circuit and constructing the all-voltage signal flow graph with inverting integrators that represents the equations. These steps are shown in Figures 34 and 35(a), respectively. In 35(a) the currents at the lower nodes are converted to voltages by multiplication by a scaling resistor R, chosen to be one so that all other gains in the circuit stay the same. The DDI and LDI sampled-data filters have exactly the same structure as the signal flow graph, but s-domain inverting integrators are replaced by their discrete time approximations, as shown in Figure 35(b). To simulate half delays in DINAP, full delays are used with the sampling rate doubled. The multipliers are given by

$$P_i = 1/(f_s(L_i \text{ or } C_i)) \qquad (35)$$

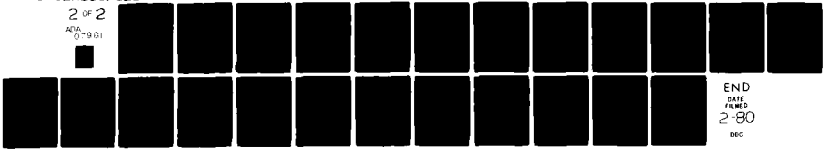
In Figure 31(c) the calculation is done, with FCLOCKD being the sampling rate of the discrete time filter. It will be seen that changing this parameter has a large effect on the frequency response. For the discrete time filter, the gain multiplier is simply $G = 1/(1+\sum_{i=1}^{10} h_i)$ and is attached after the V_{out} node in Figure 35(a).

The output of the program for f_s = 8kHz is shown in Figure 32. Similar runs were done for f_s = 10.88 kHz, 1skHz, and 120kHz. The computer outputs are not included here, but the DINAP responses will be investigated in detail.

AD-A079 611 CALIFORNIA UNIV BERKELEY DEPT OF ELECTRICAL ENGINEER--ETC F/G 17/2
CONTINUATION OF RESEARCH IN THE APPLICATION OF ANALOG SAMPLED D--ETC(U)
JUN 78 R W BRODERSEN, P R GRAY N00173-78-C-0230

UNCLASSIFIED

2 OF 2
AD A079 611



END
DATE FILMED
2-80
DDC

First the implementation using DDI sections as replacements for integrators is considered. The transformation is

$$s = (z-1)/T_s, \text{ or } s = sT_s + 1 \tag{36}$$

and the block diagram is shown in figure 36(a).

If any pole z_k has a magnitude greater than one (lies outside the unit circle), the circuit will be unstable.

$$|z_k| = \sqrt{(\mathrm{Re}(s_k T_s)+1)^2+(\mathrm{Im}(s_k T_s))^2} \tag{37}$$

For a complex pole pair with center frequency f_0 and Q given, the s-plane poles are

$$s = -\pi f_0/Q \pm j2\pi f_0\sqrt{1-1/(4Q^2)} \tag{38}$$

For $f_0 = 3462$ and $Q = 19.6$ as given in Figure 32 (difference between 3462 and 19.6 and observed $f_0 = 3454$ and $Q = 19.0$ in earlier plots is due to plotting at only 39Hz intervals), $s_k = -554.9 \pm j21745.3$, and for $T_s = 1/8\text{kHz}$, $s_k T_s = -.06936 \pm j2.718$. $|z_k| = \sqrt{(1-.06936^2)+(2.718)^2} = 2.87 > 1$, so the DDI filter will be unstable. Thus in general the DDI transformation is limited in usefulness and not a possibility here.

The same filter can be implemented using the LSI transformation, for which the basic equation is

$$s = (z^{1/2}-z^{-1/2})/T_s \tag{39}$$

with the block diagram shown in Figure 36(b). The effects on the frequency response are different than in the DDI case and are derived as follows [14]: For frequency response, $s = j\omega$ and $z = \exp(j\lambda)$, where λ is the discrete time frequency of the filter

$$j\omega = (1/T_s)(\exp(j\lambda/2)-\exp(-j\lambda/2)) \tag{40}$$

$$\omega = (2/T_s)\sin(\lambda/2) \tag{41}$$

$$f = (f_s/\pi)\sin(\lambda/2) \tag{42}$$

When implementing the continuous time filter with LDI sections continuous time frequencies f are transformed to discrete time frequencies λ. A plot of equation 42 is given in Figure 37. It is evident that a frequency expansion occurs, with the distortion increasing with frequency. In discrete time filter theory $\lambda = \pi$ corresponds to $f = f_s/2$, and the baseband signal exists for $|\lambda| < \pi$. The basic limitation of the LDI transformation is that since $|\sin \lambda/2| < 1$, only continuous time frequencies less than f_s/π can be seen in the sampled data frequency response. For the case at hand, to put the 3462 Hz center frequency within the baseband region (neglecting putting it correctly at 3462 Hz), the LDI filter must run at a minimum sampling rate of $f_s = \pi \times 3462 \cong 10.88$ kHz. For rates less than that value, the last resonance will be lost, and the higher the sampling rate, the closer it will be transformed to the desired discrete time frequency, $\lambda = (3462/8000) \times 2\pi = 2.72$.

Unfortunately as the sampling rate of the filter is increased, the response takes on more of the qualities of the continuous time response, with excessive amplitude rolloff with frequency. DINAP responses of the LDI filter for sampling rates of 8, 10.88, 15 and 120 kHz are shown in Figures 38, 39, 40, and 41. With $f_s = 120$ kHz the fourth resonance is visible, but it is so severely reduced in amplitude that is is not effective. For $f_s = 10.88$ and 15 kHz, the fourth resonance is present but not clearly identifiable, and the second and third peaks are already reduced from the desired levels.

In actual switched capacitor implementation, the path representing the source resistance will contain a half delay; $R = z^{-1/2}$ instead of 1.

Simulations were made of this effect with the same four sampling rates, and no difference in the response was seen. That result is somewhat surprising since for frequency response $z^{-1/2} = \cos(\omega T_s/2) - j\sin(\omega T_s/2)$, and the source impedance becomes complex for low sampling rates. For example, with $\omega = (2\pi)(2\text{kHz})$ and $T_s = 1/8$ kHz, $z^{-1/2} = \cos(\pi/4) - j\sin(\pi/4) = \frac{\sqrt{2}}{2} - j\frac{\sqrt{2}}{2}$.

In a different application with lower frequencies of interest, a singly terminated LDI ladder could be useful, and prewarping of the s-plane poles could be performed by the relation in equation 42 to obtain a more correct sampled data frequency response.

To see the general properties of the structure, a sensitivity study was made of the three center frequencies and Q's realized (although incorrect) by the LDI ladder for $f_s = 8$ kHz. The results of the study are shown in Figure 42. With full precision on coefficients, the discrete time center frequencies are in error as predicted by the relation

$$\lambda = 2\sin^{-1}(\pi f/f_s) \qquad (43)$$

For example, the 1537 Hz peak is expected to be seen in the sampled data response at $\lambda = 2\sin^{-1}(\pi \cdot 1537/8000) = 1.296$, which for an 8 kHz sampling rate corresponds in frequency to $f = (1.296/2\pi) \times 1650$ Hz, close to the 1660 Hz observed. From Figure 42, 8 bits of accuracy are required to keep all center frequencies and Q's within 10% of the values with no errors in the coefficients, better than the singly terminated ladder of chapter 2, but worse than the lattice filter.

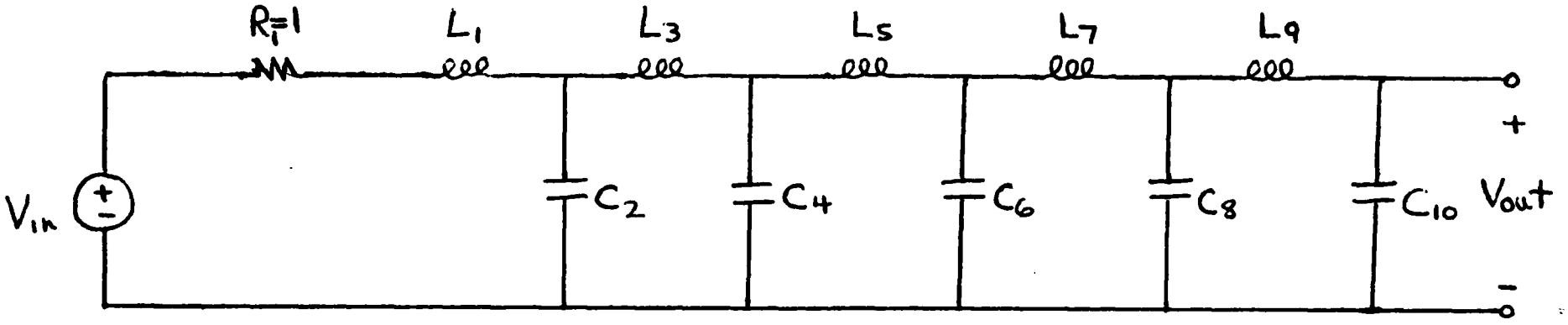


Figure 29: Singly terminated LC ladder

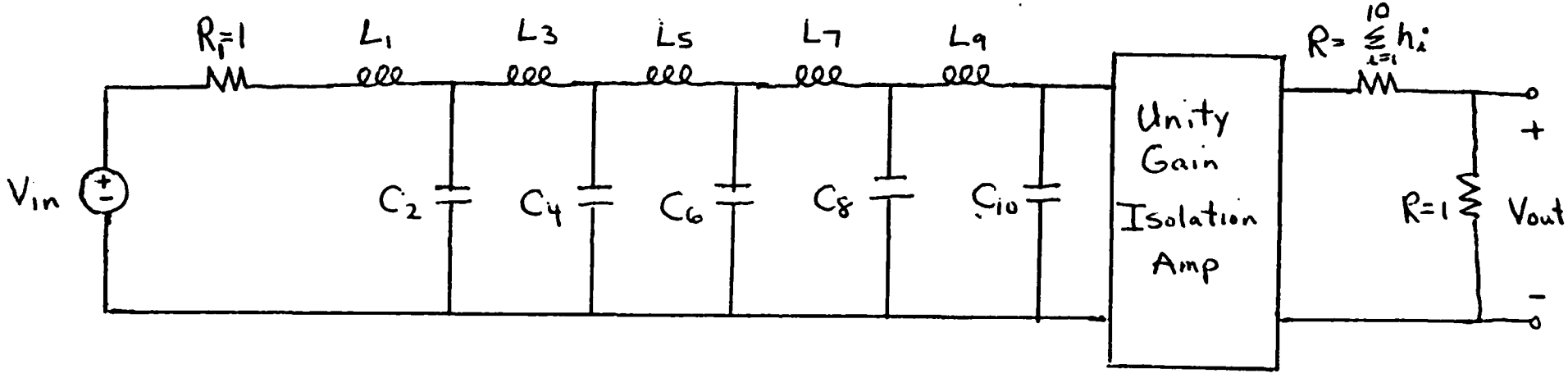


Figure 30: Singly terminated LC ladder with gain adjustment

```
      DIMENSION A(12),V(12),H(12),B(12),C(12),T(11),UU(5),VV(5),
     1FO(5),Q(5),U(12),WO(5),S(11),AI(6),CI(5),PARAM(10),ALPH(10
      PI=3.14159264
      FCLOCK=8000.0
      A(12)=0.
      N=10
      NPLUS2=12
      DO 1 I=1,11
    1 READ2,T(I)
    2 FORMAT(F5.0)
      DO 5 I=1,11
    5 A(I)=T(12-I)
C FIND Z PLANE POLES
      CALL PROOT(N,A,U,V,H,B,C,CONV,NPLUS2)
      DO 10 I=1,5
      UU(I)=U(2*I-1)
   10 VV(I)=V(2*I-1)
      PRINT 30
   30 FORMAT('0','Z PLANE POLES')
      PRINT 33
   33 FORMAT(' ','REAL PART            IMAG PART')
      DO 31 I=1,10
   31 PRINT 32,U(I),V(I)
   32 FORMAT(' ',E17.10,3X,E17.10)
C FIND S PLANE POLES
      DO 11 I=1,5
      DIV=UU(I)/SQRT(UU(I)**2+VV(I)**2)
      THETA=ACOS(DIV)
      R=SQRT(UU(I)**2+VV(I)**2)
      FO(I)=(FCLOCK/(2*PI))*(SQRT(THETA**2+ALOG(R)**2))
      WO(I)=2*PI*FO(I)
   11 Q(I)=(-PI*FO(I))/(FCLOCK*ALOG(R))
      PRINT 60
   60 FORMAT('0',' ')
      DO 40 I=1,5
   40 PRINT41,I,FO(I),I,Q(I)
   41 FORMAT(' ','FO(',I1,')=',E17.10,'     Q(',I1,')=',E17.10)
C FIND COEFFS OF EACH POWER OF S
      CALL CALC(WO,Q,S)
C COMPUTE COEFFS FOR ELEMENT FINDING SUBROUTINE
      AI(6)=S(11)
      AI(5)=S(9)
      AI(4)=S(7)
      AI(3)=S(5)
      AI(2)=S(3)
      AI(1)=S(1)
      CI(5)=S(10)
      CI(4)=S(8)
      CI(3)=S(6)
      CI(2)=S(4)
      CI(1)=S(2)
C COMPUTE ELEMENT VALUES
      CALL COEFFS(AI,CI,ALPH)
   RUNW.2 COMPILER. MAR 73 VERSION

      PRINT 13
   13 FORMAT('0','ANALOG L AND C VALUES L1 THRU C10')
      DO 20 I=1,10
   20 PRINT 14,ALPH(I)
   14 FORMAT(' ',E17.10)
```


Figure 31(a): Element-finding program for analog LC ladder

```
      DIMENSION WOO(5),OO(5),SS(11)
      ST3=WOO(2)*OO(2)+WOO(1)*OO(1)
      ST2=WOO(1)**3+WOO(2)**2+(WOO(1)*WOO(2))/OO(1)*OO(2))
      ST1=((WOO(1)*WOO(2)**2)/OO(1)+((WOO(1)**2)*WOO(2))/OO(2)
          ST0=WOO(1)**2*WOO(2)**2
      ST8=WOO(4)*OO(4)+WOO(3)*OO(3)
      ST7=WOO(3)**2+WOO(4)**2+(WOO(3)*WOO(4))/OO(3)*OO(4))
      ST6=((WOO(3)*WOO(3)**2)*OO(3)+((WOO(3)**2)*WOO(4))/OO(4)
      ST5=WOO(3)**2*WOO(4)**2
      T1=WOO(5)*OO(5)
      T0=WOO(5)**2
      X7=ST3+ST8
      X6=ST2+ST7+ST3*ST8
      X5=ST1+ST6+ST3*ST7+ST2*ST8
      X4=ST0+ST5+ST3*ST6+ST2*ST7+ST1*ST8
      X3=ST3*ST5+ST2*ST6+ST1*ST7+ST0*ST8
      X2=ST2*ST5+ST1*ST6+ST0*ST7
      X1=ST1*ST5+ST0*ST6
      X0=ST0*ST5
      SS(11)=1
      SS(10)=X7+T1
      SS(9)=X6+T1*X7+T0
      SS(8)=X5+T1*X6+T0*X7
      SS(7)=X4+T1*X5+T0*X6
      SS(6)=X3+T1*X4+T0*X5
      SS(5)=X2+T1*X3+T0*X4
      SS(4)=X1+T1*X2+T0*X3
      SS(3)=X0+T1*X1+T0*X2
      SS(2)=T1*X0+T0*X1
      SS(1)=T0*X0
      RETURN
      END
      SUBROUTINE COEFFS(AA,CC,ALPHA)
      DIMENSION Q(5),R(4),SS(4),TT(3),U(3),VV(2),W(2),AA(6),CC(5),ALPH
     1(10)
      Q(1)=((CC(5)*AA(5))-(AA(6)*CC(4)))/CC(5)
      Q(2)=((CC(5)*AA(4))-(AA(6)*CC(3)))/CC(5)
      Q(3)=((CC(5)*AA(3))-(AA(6)*CC(2)))/CC(5)
      Q(4)=((CC(5)*AA(2))-(AA(6)*CC(1)))/CC(5)
      Q(5)=AA(1)
      R(1)=((Q(1)*CC(4))-(CC(5)*Q(2)))/Q(1)
      R(2)=((Q(1)*CC(3))-(CC(5)*Q(3)))/Q(1)
      R(3)=((Q(1)*CC(2))-(CC(5)*Q(4)))/Q(1)
      R(4)=((Q(1)*CC(1))-(CC(5)*Q(5)))/Q(1)
      SS(1)=((R(1)*Q(2))-(Q(1)*P(2)))/R(1)
      SS(2)=((R(1)*Q(3))-(Q(1)*R(3)))/R(1)
      SS(3)=((P(1)*Q(4))-(Q(1)*R(4)))-R(1)
      SS(4)=Q(5)
      TT(1)=((SS(1)*R(2))-(P(1)*SS(2)))/SS(1)
      TT(2)=((SS(1)*R(3))-(P(1)*SS(3)))/SS(1)
      TT(3)=((SS(1)*R(4))-(P(1)*SS(4)))/SS(1)
      U(1)=((TT(1)*SS(2))-(SS(1)*TT(2)))/TT(1)
      U(2)=((TT(1)*SS(3))-(SS(1)*TT(3)))/TT(1)
      U(3)=SS(4)
      VV(1)=((U(1)*TT(2))-(TT(1)*U(2)))-U(1)
      VV(2)=((U(1)*TT(3))-(TT(1)*U(3)))/U(1)
      W(1)=((VV(1)*U(2))-(U(1)*VV(2)))/VV(1)
      W(2)=U(3)
      X=((W(1)*VV(2))-(VV(1)*W(2)))/W(1)
      Y=W(2)
      ALPHA(1)=AA(6)/CC(5)
      ALPHA(2)=CC(5)/Q(1)
      ALPHA(3)=Q(1)/R(1)
      ALPHA(4)=R(1)/SS(1)
      ALPHA(5)=SS(1)/TT(1)
      ALPHA(6)=TT(1)/U(1)
      ALPHA(7)=U(1)/VV(1)
      ALPHA(8)=VV(1)/W(1)
      ALPHA(9)=W(1)/X
      ALPHA(10)=X/Y
      RETURN
```


Figure 31(b):
Subroutines for LC ladder calculation

```
C FIND MULTIPLIER COEFFS FOR DIGITAL FILTER
      FCLOCK=1.00000.0
      DO 12 I=1,10
   12 PARAM(I)=1.*FCLOCK*ALPH(I)
      PRINT 15
   15 FORMAT('0','DIGITAL LADDER PARAMETERS 1 THRU 10')
      DO 16 I=1,10
   16 PRINT 17,PARAM(I)
   17 FORMAT(' ',E17.10)
C FIND ANALOG RESISTOR AND DIGITAL MULTIPLIER TO ADJUST DC GAIN
      SUM=T(2)+T(3)+T(4)+T(5)+T(6)+T(7)+T(8)+T(9)+T(10)+T(11)
      P=SUM
      FACTOR=1./(1+SUM)
      PRINT18,P
   18 FORMAT('0','RESISTOR VALUE=',E17.10)
      PRINT19,FACTOR
   19 FORMAT('0','DIGITAL GAIN FACTOR=',E17.10)
      END
      SUBROUTINE PROOT(N,A,U,V,H,B,C,CONV,NPLUS2)
      DIMENSION A(NPLUS2),U(NPLUS2),V(NPLUS2),H(NPLUS2),B(NPLUS2)
     1C(NPLUS2)
      CONV=1.E-35
      NC=N+1
      DO 1 I=1,NC
    1 H(I)=A(I)
      P=0.
      Q=0.
      R=0.
      IREV=1
    3 IF(H(1))4,2,4
    2 NC=NC-1
      V(NC)=0.
      U(NC)=0.
      DO 1002 I=1,NC
 1002 H(I)=H(I+1)
      GO TO 3
    4 IF(NC-1)5,100,5
    5 IF(NC-2)7,6,7
    6 P=-H(1)/H(2)
      GO TO 50
    7 IF(NC-3)9,8,9
    8 P=H(2)/H(3)
      Q=H(1)/H(3)
      GO TO 70
    9 IF(ABS(H(NC-1)/H(NC))-ABS(H(2)/H(1)))10,19,19
   10 IREV=-IREV
      M=NC/2
      DO 11 I=1,M
      NL=NC+1-I
      F=H(NL)
      H(NL)=H(I)
   11 H(I)=F
      IF(Q)13,12,13
   12 P=0.
      GO TO 15
   13 P=P/Q
      Q=1./Q
   15 IF(R)16,19,16
   16 R=1./R
   19 E=5.E-10
      B(NC)=H(NC)
      C(NC)=H(NC)
      B(NC+1)=0.
      C(NC+1)=0.
      NP=NC-1
   20 DO 49 J=1,1000
      DO 21 II=1,NP
      I=NC-II
      B(I)=H(I)+P*B(I+1)
   21 C(I)=B(I)+P*C(I+1)
      IF(ABS(B(1)/H(1))-E)50,50,24
```


Figure 31(c):
Continuation of basic program

```
   IF(C(2))21,22,23
22 R=R+1
   GO TO 30
23 R=R-B(1)/C(2)
30 DO 37 II=1,NR
   I=NC-II
   B(I)=H(I)-R*B(I+1)-Q*B(I+2)
37 C(I)=B(I)-R*C(I+1)-Q*C(I+2)
   IF(H(2))30,31,32
31 IF(ABS(B(2)/H(1))-E)33,33,34
32 IF(ABS(B(2)/H(2))-E)33,33,34
33 IF(ABS(B(1)/H(1))-E)70,70,34
34 CBAR=C(2)-B(3)
   D=C(3)**2-CBAR*C(4)
   IF(D)36,35,36
35 P=P-2
   Q=Q*(Q+1.)
   GO TO 49
36 P=P+(B(2)*C(3)-B(1)*C(4))/D
   Q=Q+(-B(2)*CBAR+B(1)*C(3))/D
49 CONTINUE
   E=E*10
   IF(E-CONV)20,20,40
40 CONV=E
   GO TO 20
50 NC=NC-1
   V(NC)=0.
   IF(IREV)51,52,52
51 U(NC)=1./R
   GO TO 53
52 U(NC)=R
53 DO 54 I=1,NC
54 H(I)=B(I+1)
   GO TO 4
70 NC=NC-2
   IF(IREV)71,72,72
71 QP=1./Q
   PP=P/(Q+2.0)
   GO TO 73
72 QP=Q
   PP=P/2.0
73 F=(PP)**2-QP
   IF(F)74,75,75
74 U(NC+1)=-PP
   U(NC)=-PP
   V(NC+1)=SQRT(-F)
   V(NC)=-V(NC+1)
   GO TO 76
75 U(NC+1)=-SIGN(ABS(PP)+SQRT(F),PP)
   V(NC+1)=0.
   U(NC)=QP/U(NC+1)
   V(NC)=0.
76 DO 77 I=1,NC
77 H(I)=B(I+2)
   GO TO 4
100 RETURN
   END
```

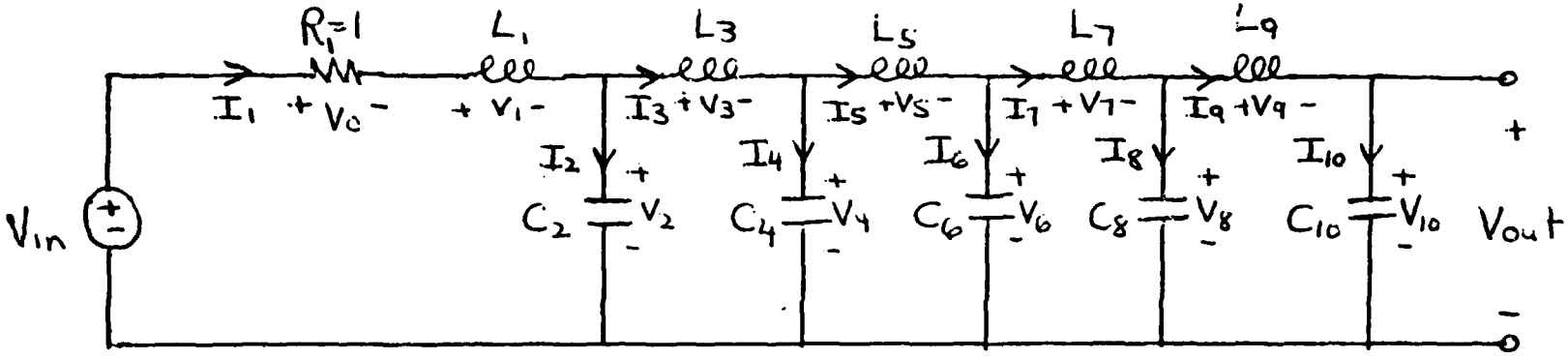

Figure 31(c) (cont)

Continuation of root finding subroutine

```
FD(1)=  .2[illegible]E+04
FD(2)=  .[illegible]E+04
FD(3)=  .21[illegible]E+04
FD(4)=  .15[illegible]E+04
FD(5)=  .61[illegible]E+03

ANALOG L AND C VALUES L1 THRU C10
 .10[illegible]E-03
 .42[illegible]E-01
 .32[illegible]E-02
 .11[illegible]E-04
 .10[illegible]E-02
 .10[illegible]E-04
 .[illegible]
 .17[illegible]
 .33[illegible]
 .97[illegible]E-05

DIGITAL LADDER PARAMETERS 1 THRU 10
 .1219086442E+01
 .2955157669E+01
 .3888720899E+00
 .1079773710E+02
 .1235113376E+00
 .1170040395E+02
 .1416488959E+00
 .7031565116E+01
 .5379639019E-01
 .2169258672E+02

RESISTOR VALUE=  .2680000000E+01
DIGITAL GAIN FACTOR=  .2717391304E+00

L(1)=  .1[illegible]E+02
L(2)=  .21[illegible]E+01
C(3)=  .10[illegible]E+01
C(4)=  .2[illegible]E+02
C(5)=  .6[illegible]E+01
```

Figure 32 : Computer output for analog ladder and digital implementation (f_s = 8KHz)

Figure 33 : Analog LC ladder response



$I_2 = I_1 - I_3$ $\quad V_2 = \frac{1}{sC_2} I_2$ $\quad V_1 = V_{in} - V_o - V_2$ $\quad I_1 = \frac{1}{sL_1} V_1$ $\quad V_{out} = V_{10}$

$I_4 = I_3 - I_5$ $\quad V_4 = \frac{1}{sC_4} I_4$ $\quad V_3 = V_2 - V_4$ $\quad I_3 = \frac{1}{sL_3} V_3$ $\quad V_o = I_1 \ (R=1)$

$I_6 = I_5 - I_7$ $\quad V_6 = \frac{1}{sC_6} I_6$ $\quad V_5 = V_4 - V_6$ $\quad I_5 = \frac{1}{sL_5} V_5$

$I_8 = I_7 - I_9$ $\quad V_8 = \frac{1}{sC_8} I_8$ $\quad V_7 = V_6 - V_8$ $\quad I_7 = \frac{1}{sL_7} V_7$

$I_{10} = I_9$ $\quad V_{10} = \frac{1}{sC_{10}} I_{10}$ $\quad V_9 = V_8 - V_{10}$ $\quad I_9 = \frac{1}{sL_9} V_9$

Figure 34 : Basic equations of LC ladder

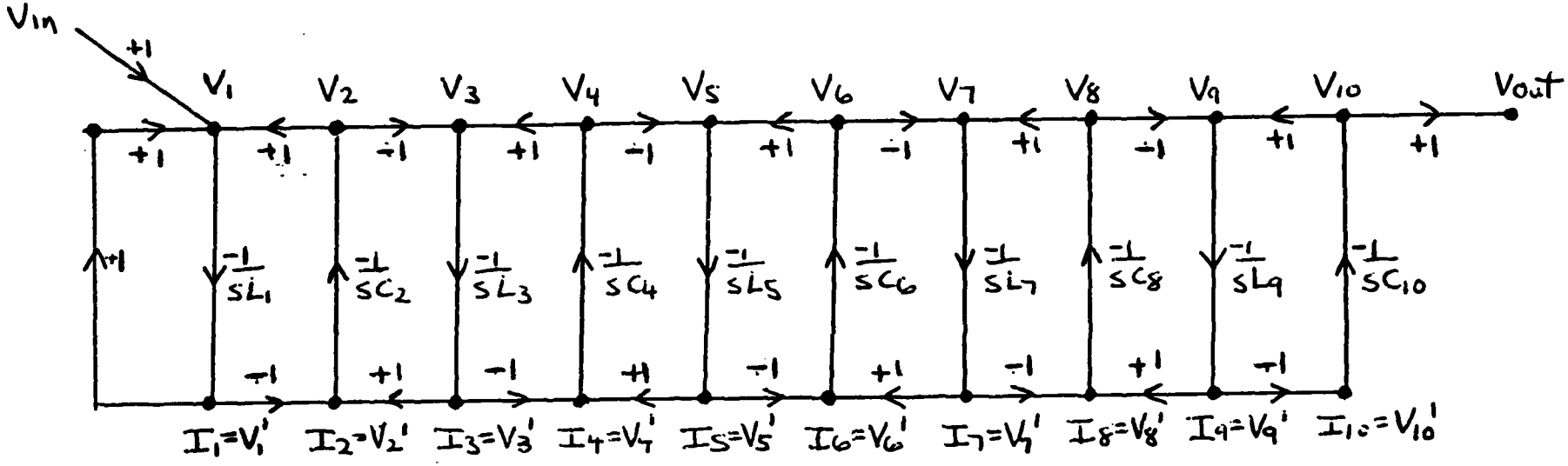
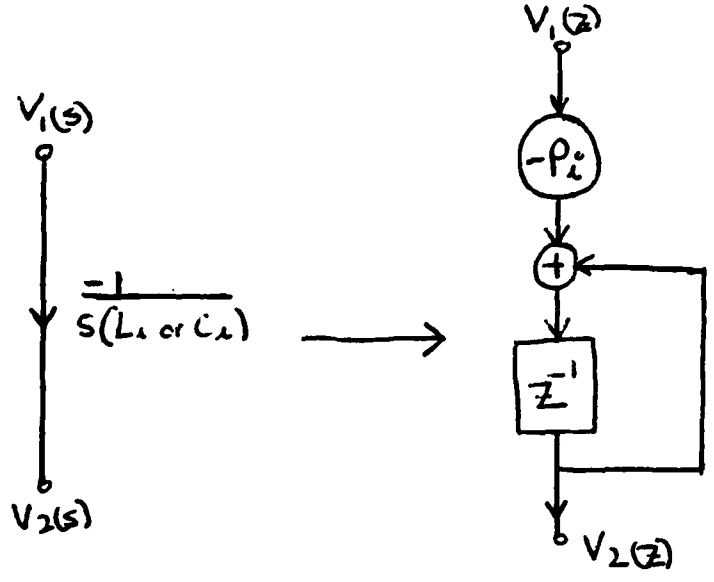
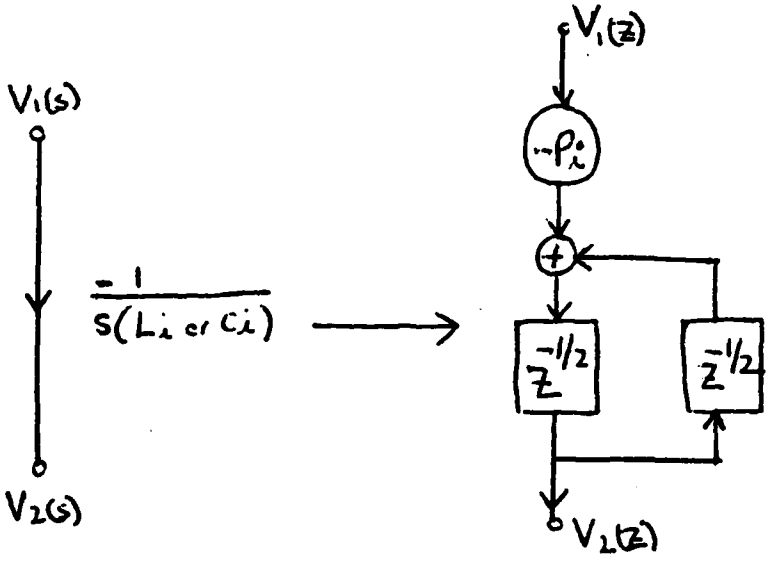


Figure 35(a) Signal flow diagram for singly terminated LC ladder

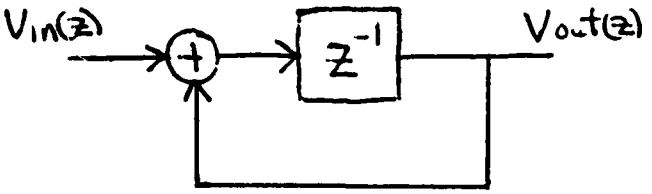


(a) DDI



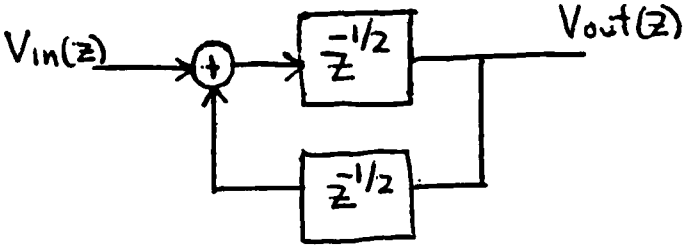
(b) LDI

Figure 35(b): Replacement of analog integrations by digital approximations



$$s \rightarrow (z-1)/T_s$$

(a) DDI



$$s \rightarrow (z^{1/2} - z^{-1/2})/T_s$$

(b) LDI

Figure 36: Transformations for digital implementation of analog singly terminated LC ladder

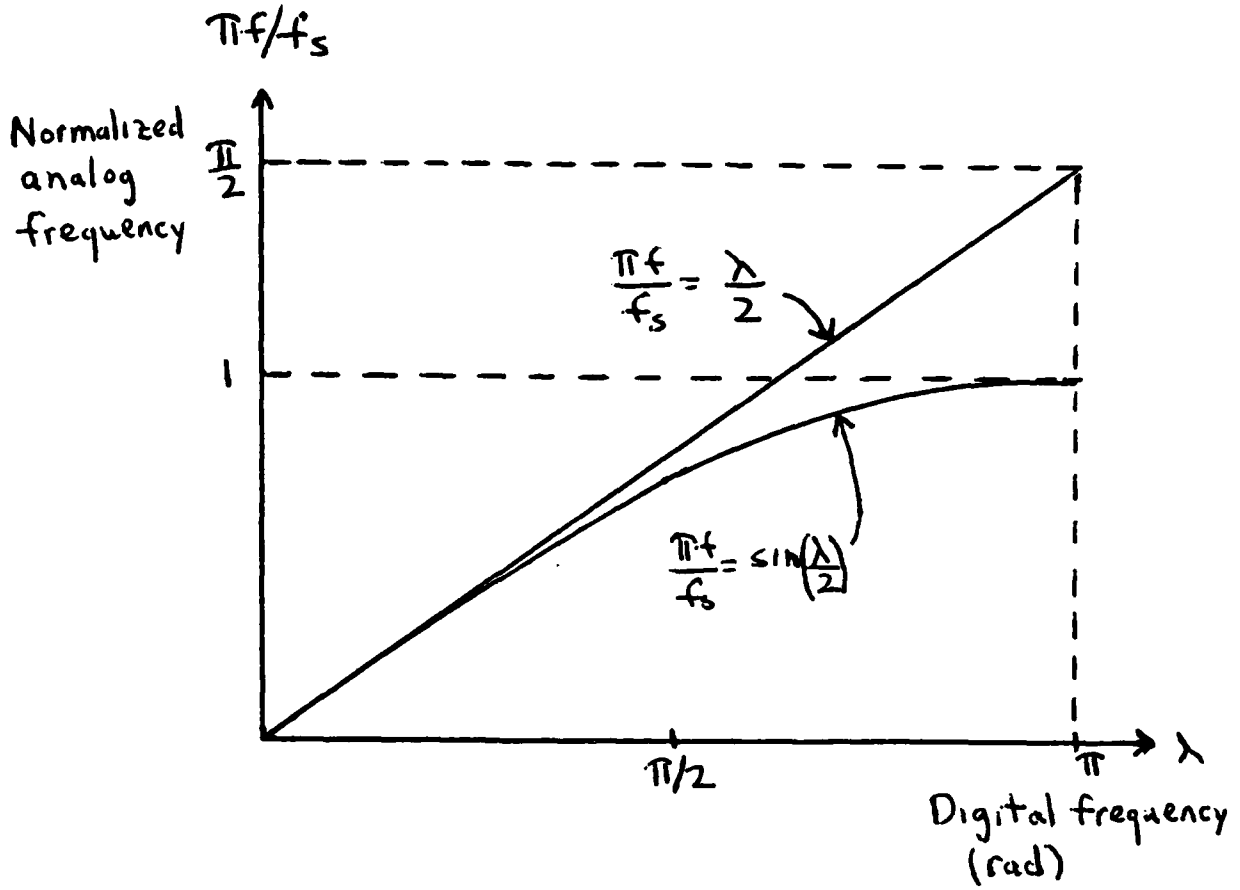


Figure 37 : Frequency warping characteristic of the LDI transformation

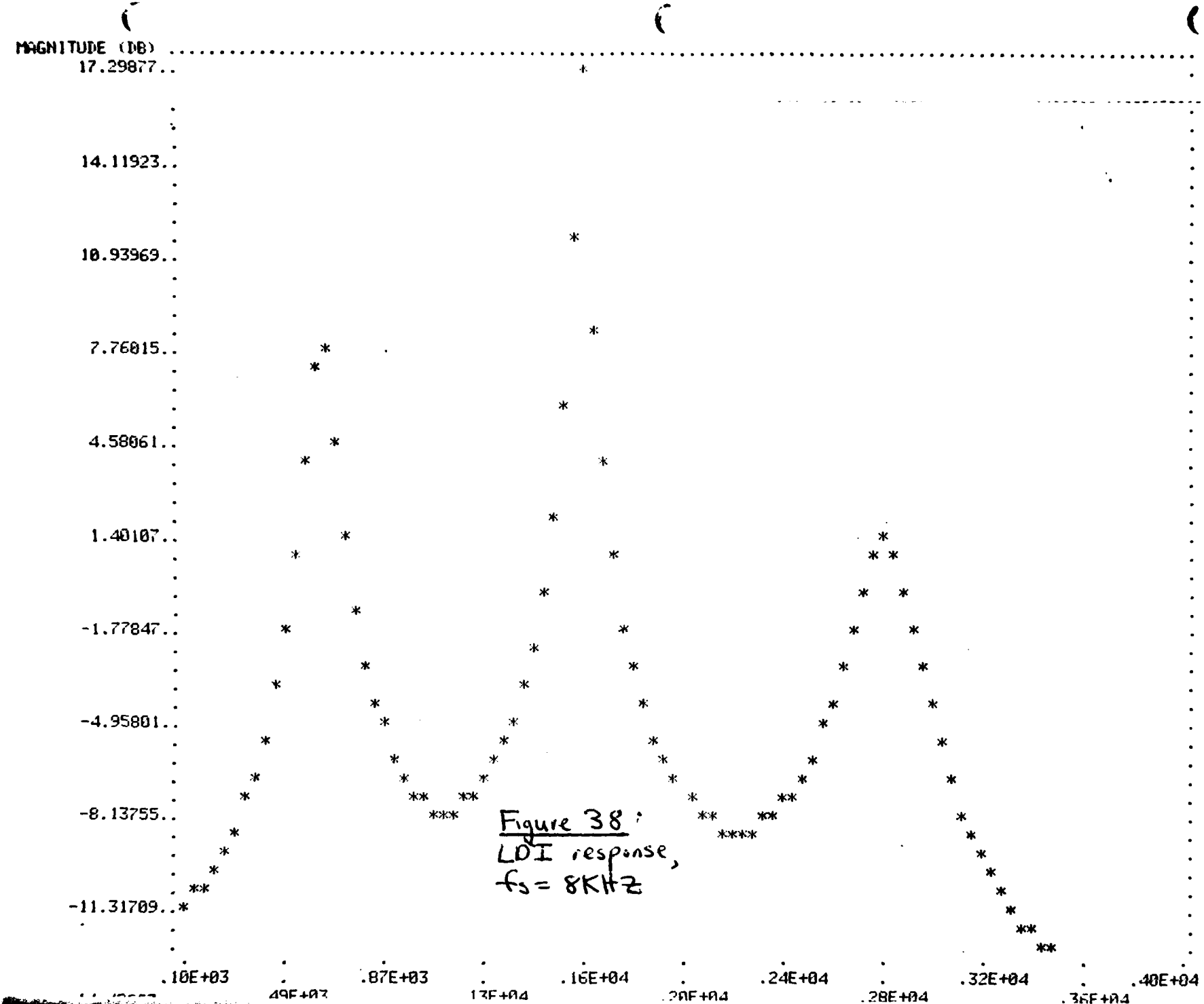


Figure 38 :
LDI response,
f_s = 8KHz

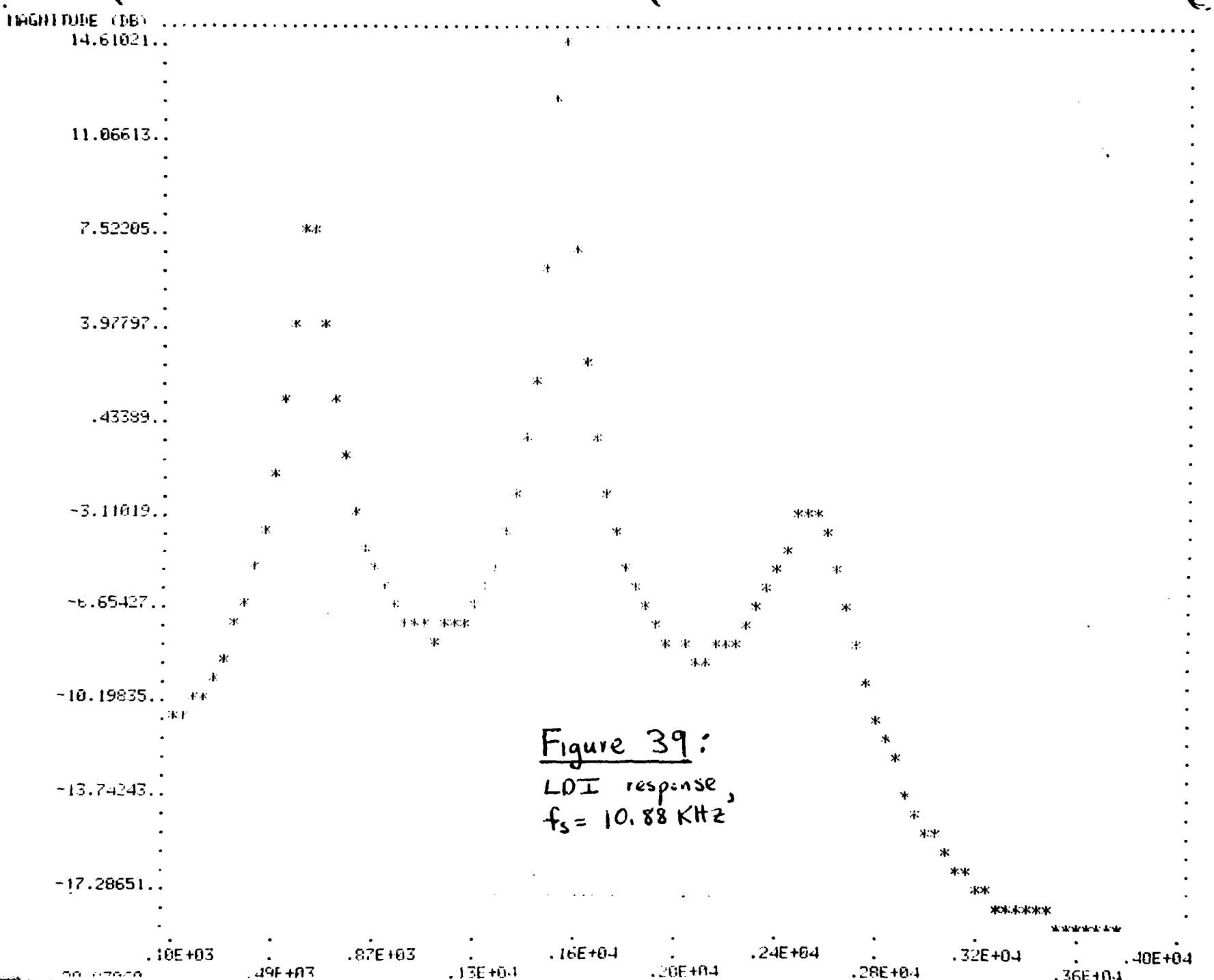


Figure 39:
LDI response,
f_s = 10.88 KHz

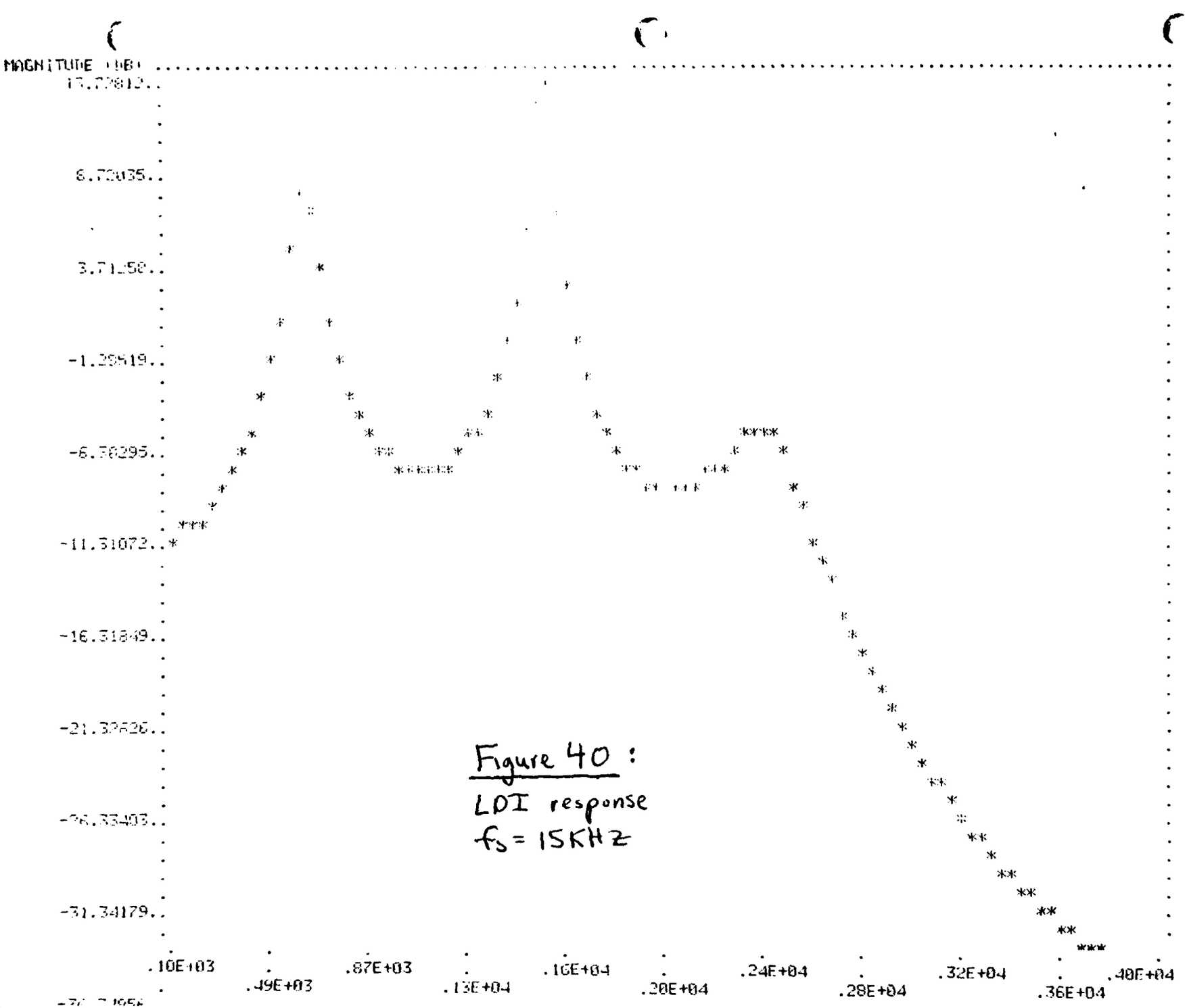


Figure 40 :
LDI response
f_s = 15KHz

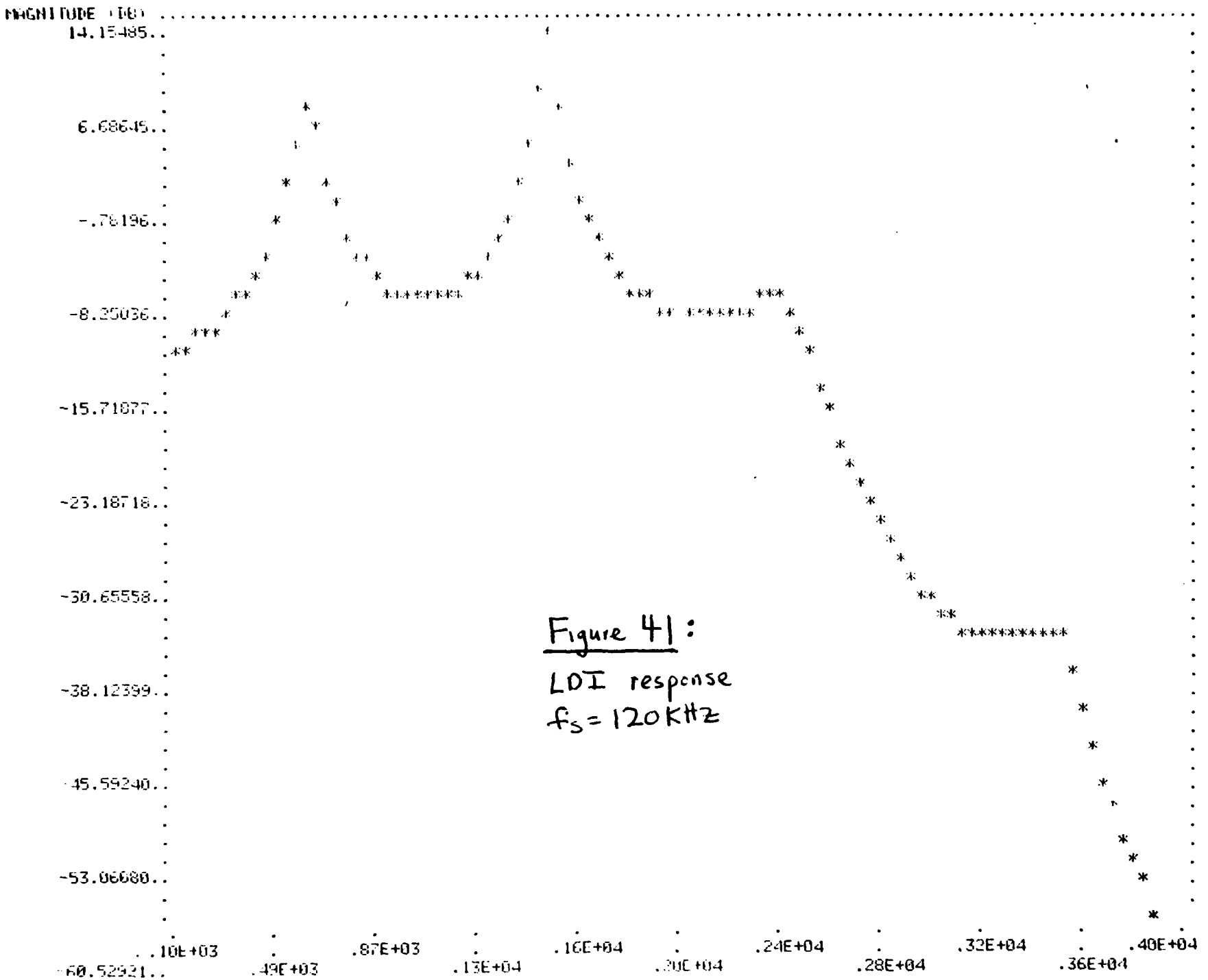


Figure 41:
LDI response
$f_s = 120\,KHz$

Precision en Coeffs	F_1	Q_1	F_2	Q_2	F_3	Q_3
Full Precision	646	7.0	1660	49.9	2791	4.1
8 bits	646	7.0	1660	49.9	2791	4.1
7 bits	646	7.0	1660	44.8	2791	4.7
6 bits	646	6.7	1543	19.3	2752	9.1
5 bits	646	7.5	1777	51.1	2947	7.7
4 bits	607	3.8	1777	45.4	2635	11.8

Figure 42: Results of sensitivity study for singly terminated LDI ladder

Conclusion

Three basic filter structures have been investigated for implementing a ten pole transfer function of the human vocal tract. From a practical standpoint only the lattice filter has sensitivity properties good enough to warrant serious consideration.

In the process of developing the synthesis procedures for the specific digital filters, several more general computer program segments were developed, such as the plotter, the complex root finder, and the algorithm to do the Cauer expansion. It is hoped that the material in this paper will be of general use to those investigating analog sampled data filters in the future.

References

[1] R. W. Brodersen, P. R. Gray, and D. H. Hodges, "MOS Switched-Capacitor Filters," Proceedings of the IEEE, Vol. 67, No. 1, January 1979, pp. 61-74.

[2] G. M. Jacobs, D. J. Allstot, R. W. Brodersen, and P. R. Gray, "Design Techniques for MOS Switched Capacitor Ladder Filters," IEEE Journal of Solid State Circuits, Vol. CAS-25, No. 12, December 1978, pp. 1014-1021.

[3] B. J. Hosticka, R. W. Brodersen, and P. R. Gray, "MOS Sampled-data Recursive Filters Using Switched Capacitor Integrators," IEEE Journal of Solid State Circuits, Vol. SC-12, December 1977, pp. 600-608.

[4] G. M. Jacobs, "Practical Design Considerations for Switched Capacitor Ladder Filters," Memorandum No. UCB/ERL-M77/69, University of California, Berkeley, 1977.

[5] D. J. Allstot, R. W. Brodersen, and P. R. Gray, "An Electrically Programmable Analog NMOS Second-Order Filter," ISSCC Digest of Technical Papers, 1979, pp. 76-77.

[6] J. D. Markel and A. H. Gray, Jr., Linear Prediction of Speech, Springer-Verlag, 1976.

[7] L. R. Rabiner and R. W. Schafer, Digital Processing Speech Signals, Prentice-Hall, 1978.

[8] J. L. McCreary and P. R. Gray, "All-MOS Charge Redistribution Analog-to-Digital Conversion Techniques - Part I," IEEE Journal of Solid State Circuits, Vol. SC-10, December 1975, pp. 371-379.

[9] R. H. McCharles and D. H. Hodges, "Charge Circuits for Analog LSI," IEEE Journal of Solid-State Circuits, Vol. CAS-25, July 1978, pp. 490-497.

[10] S. K. Mitra and R. J. Sherwood, "Digital Ladder Networks," IEEE Trans. on Solid State Circuits, Vol. AU-21, February 1973, pp. 30-36.

[11] G. C. Temes and J. W. LaPatra, Circuit Synthesis and Design, McGraw-Hill, 1977, pp. 154-174.

[12] B. C. Kuo, Automatic Control Systems, Prentice-Hall, 1975, pp. 322-324.

[13] S. E. Belter, "DINAP-Digital Network Analysis Program," Ph.D. Thesis, Purdue University.

[14] Private communication, Tat Choi.