CALIFORNIA STATE UNIVERSITY, NORTHRIDGE

DIGITAL GRAPHIC AUDIO EQUALIZER

A graduate project submitted in partial satisfaction of the requirements for the degree of Master of Science in

Engineering

by

Alberto Luis Andres

January, 1983

The Graduate Project of Alberto Luis Andres is approved:

Jack Gaston

Roger M. Di Julio

Robert Y. Wong, Chairma

California State University, Northridge

TABLE OF CONTENTS

Approval page ii

Table of contents iii

Abstract vi

Chapter 1. Basis for the design

1. Introduction 1
2. Description and application of graphic equalizers 2
3. Design requirements and approaches 3
4. Selection of the processor 8
5. Basic block diagram 9

Chapter 2. Design of the filters

1. Digital bandpass filters 12
2. Digital lowpass filters 29
3. Analog lowpass filters 35

Chapter 3. Software design

1. Architecture and program features of the AMI S2811 41
2. Memory mapping 44
3. The program 48

Chapter 4. Hardware design

1. General architecture 62
2. Analog to digital converter 68
3. Digital to analog converter 75
4. Base number ROM, gain controls and gain coefficient ROM 80
   Base number ROM (ROM1) 80
   Gain controls 82
   Gain coefficient ROM (ROM2) and multiplexer 86
5. Control logic and timing 88
   Hardware implementation 92
   ROM1 counter control 96

Summary and conclusions 99

References 101

Appendix 103

TABLES

1.1 Sequence of filters processing 7
2.1 Combined output of BPF's 23
4.1 Addresses for decoder 74154 85
4.2 ROM2 addresses and gain coefficients 87

TABLE OF CONTENTS

FIGURES

1.1	Basic block diagram	10
2.1	Processing sequence of BPF and LPF	14
2.2	Amplitude response of a BPF	14
2.3	Response of the first 6 BPF's	21
2.4	Response of all the BPF's in log scales	21
2.5	Poles and zeros of H(z) for BPF	25
2.6	Implementation of BPF of eq. (2.12,2.13)	25
2.7	Implementation of the 4th order LPF of eq. (2.19, 2.20)	33
2.8	Schematic of 4th order biquad elliptic LPF	39
3.1	Flow chart of the signal processing	45
3.2	Memory mapping of SPP processor	47
4.1	Block diagram of the complete equalizer	63
4.2	Program events timing	67
4.3	Schematic of the A/D converter	69
4.4	Timing diagram, for the A/D conversion	71
4.5	Schematic of the D/A converter	76
4.6	Timing diagram of the D/A conversion	78
4.7	Schematic of gain controls, ROM1, ROM2 and multiplexer interface	81
4.8	Timing diagram of ROM's, Mux and Gain ctr.	83
4.9	Schematic of the control logic section	89
4.10	Timing diagram of the control logic	90

ABSTRACT

DIGITAL GRAPHIC AUDIO EQUALIZER

by

Alberto Luis Andres

Master of Science in Engineering

Digital means are becoming common in the processing of audio signals but they have not been as yet directly utilized in home entertainment audio components due, in part, to the fast sampling rates and large dynamic ranges required for good audio reproduction. This project presents the design of an audio graphic equalizer using digital filters and signal processing. Graphic equalizers are traditionally designed with a bank of bandpass filters spaced in frequency by octaves or fractions of octaves. The design in this project uses 10 bandpass filters with frequencies related by octaves to cover the entire audio range. The system designed emphasizes the application of state of the art digital signal processing hardware to audio equipment. The

signal processor AMI S2811 is used as the heart of the system due to its fast processing speed and flexible instruction set. The speed, however, is not enough to process ten filters in a sampling period. To solve this problem, a scheme is used whereby a basic digital bandpass filter is sampled at successively halved rates to generate all the lower octave filters. Digital lowpass filters are interposed to prevent aliasing. With this scheme, only two bandpass and two lowpass filters need be processed per each sampling period. The program and hardware are designed to perform this processing and to input and output the data. The design, which has been done for 12 bit resolution, incorporates gain controls to adjust the gain of each filter so as to shape the frequency response as desired.

CHAPTER 1

BASIS FOR THE DESIGN

1. INTRODUCTION

The objective of this project is to design an audio graphic equalizer using digital techniques. The motivating idea is to show the feasibility of designing audio equipment traditionally produced in analog form, with the presently available digital hardware. It is recognized that the necessary processing speeds have long been achieved in large computers, but only recently signal processors specially intended for audio signals have been introduced in the market at reasonable prices. Besides the relatively high speeds required for the processing of audio signals, there exists the need for large dynamic ranges.

Present day audio components can have a dynamic range of up to 80 or 90 dB. However, the dynamic range of the media used (tapes and records) is typically no more than 50 or 60 dB, or somewhat better when companding schemes are introduced. The media of the near future, instead, will offer excellent dynamic ranges. For example, digitally recorded tapes are common place in recording studios and are now being introduced in home equipment with adaptors for video recorders; the digital audio disc is now in the marketplace.

All these media have resolutions of 14 to 16 bits, what gives dynamic ranges of 84 to 96 dB using the rule of thumb that digital processing yields about 6 dB per bit. Therefore, audio processing requires large number of bits at high speeds, a combination not easy to achieve. Nevertheless, the trend is toward digital processing of audio signals and in the future, it may be expected that most components, such as preamplifiers, amplifiers, equalizers, etc., will be digital, with probably simpler and easier to interface designs.

It should be obvious that our design would be extremely costly when compared with its analog counterpart, but the purpose is to show that audio components processing signals digitally can be made with readily available hardware. The present trend in integration and cost reduction of digital parts may, in the near future, make a project like this economically competitive with today's analog designs.

2. DESCRIPTION AND APPLICATIONS OF GRAPHIC EQUALIZERS

Audio graphic equalizers are normally used in conjunction with other audio processing equipment, in order to compensate for deficiencies in the acoustic environment by shaping the overall frequency response of the audio system.

Typically, the audio graphic equalizer consists of a bank of bandpass filters whose frequencies are spaced to cover the entire audio range. The audio signal is processed through all these filters and the outputs of all of them are then summed to reconstruct the original input. The level

of the signals at the output of the filters, or equivalently, the gain of the filters, can be adjusted with front panel controls so that the user can shape the frequency response of the system where the equalizer is installed to suit his requirements. A typical range of adjustment is ±10 dB. The gain controls are usually of the slide type and placed side by side, whereby the approximate frequency response is displayed by the position of the control knobs; hence, the name of "graphic".

The frequency spacing of the bandpass filters depends on the sophistication of the equalizer but the ratio between adjacent frequencies is always an octave or a fraction of octave. For high quality, non-professional equalizers the separation between frequencies is usually an octave with 10 to 11 bandpass filters used for each channel.

3. DESIGN REQUIREMENTS AND APPROACHES

For a digital design, the idea is to use digital filters instead of their analog counterparts. There are many advantages to this approach. Once a design has been made, it can be easily modified for different frequencies by just changing the coefficients of the filters. Multipole, higher order filters can be easily implemented by repeated passes of the signals through second order filters or by introducing minor program changes. Furthermore, these features can be made software controllable by the user giving great flexibility in professional applications, such as in parametric equalizers.

In designing real time signal processors, such as graphic equalizers, one of the major difficulties encountered is the need for very fast arithmetic processing. A second order digital filter implementation may require 5 multiplications and 4 additions, plus possibly the multiplication of the input sample by a gain factor. In the 20 Hz-20 KHz audio range, a sampling rate higher than 40 KHz is mandatory if we are to avoid aliasing. Even though the processed audio spectrum does not normally extends up to 20 KHz, this sampling frequency may seem too low if we try to avoid highly sophisticated hardware and antialiasing filters. In any event, with only 20 to 25 usec (ideally) between samples and having to process the signal through 10 filters in that period of time, there is no conventional microprocessor in today's market capable of performing 50 multiplications, 40 additions and overhead instructions in those 20 to 25 usec.

In the case of a graphic equalizer with octave spaced filters, however, the frequencies and bandwidths of all the filters are related by powers of 2. This means that the equalizer filters are not of the constant bandwidth type but rather of the constant percentage bandwidth.

Now, the transfer function of a digital filter is defined only with respect to the sampling frequency through $z = e^{sT}$, where T is the sampling period, $\underline{z}$ is the discrete complex variable and $\underline{s}$ is the continuos complex variable. Therefore, by sampling at half the rate ($\frac{1}{2}f_s = 1/2T$), the

center frequency, as well as the bandwidth of the filter is halved. By processing the input to the same filter at successively halved sampling rates, we obtain precisely what we wanted: filters separated by octaves and with constant percentage bandwidth. Of course, to prevent aliasing, every time we halve the sampling rate, the samples have to be band limited to one half their previous bandwidth. This is again done with one digital lowpass filter whose cut-off frequency is halved every time the input is sampled at successively halved rates. The cut-off frequency of course, is chosen below the $\frac{1}{2}f_s$, where f_s is the sampling frequency of the corresponding bandpass filter.

For our design we decided to use 10 filters spaced in frequency by octaves and a sampling rate of 64 KHz. The processing is arranged so that the highest frequency filter, the 16 KHz, processes every sample; the 8 KHz filter processes every second sample; the 4 KHz filter, every fourth sample and so on. The 31 Hz filter processes every 512th sample. Similarly, the lowpass filter of the highest frequency (12 KHz in our case) processes every sample; the 6 KHz, every second sample and so on. (We will refer to 16K, 8K Hz filters,etc., but the filter is the same; only the sampling rate changes)

The procedure is then to pass the original signal, which has been band limited to a frequency below $\frac{1}{2}64$ KHz (we chose 21 KHz) and which has been sampled at 64 KHz, through the 16 KHz bandpass and through the 12 KHz lowpass filters. Although the input to the 12 KHz lowpass filter is still

sampled at 64 KHz, its output can be sampled at 32 KHz by discarding every second sample, since the bandwidth has been limited to 12 KHz, i.e., below $\frac{1}{2}f_s = \frac{1}{2}32$ KHz = 16 KHz. If this is done, that output can again be processed through the bandpass and lowpass filters at $\frac{1}{2}f_s$, giving outputs corresponding to 8 KHz bandpass and 6 KHz lowpass filters respectively. By continuing this process we can obtain all the other lower frequency filters.

Since every second sample is discarded, we can interleave the processing of all the filters in a sort of time division multiplexing fashion. In so doing, we realize that only two filters need to be processed per sample. For example, according to the procedure explained before, the 16 KHz filter will be processed 16 times in 16 samples; the 8 KHz, 8 times; the 4 KHz, 4 times; the 2 KHz, 2 times and the 1 KHz, once. If we were using only these 5 bands, there would be a total of 31 filters processed in 16 samples or 2 filters per sample. The same holds true for any number of bands. Therefore, all we need to do is to process two bandpass filters and two lowpass filters per sample, thus reducing our processing needs from the original 10 filters to only four.

Table 1.1 shows the sequence in which the bandpass (BPF) and lowpass (LPF) filters must be processed to interleave properly. In every sample, the first BPF is always the 16 KHz and the first LPF, the 12 KHz. Under 2nd BPF and 2nd LPF input and output are the sample numbers, with a subindex, the LPF through which the sample was processed. For example, 3_{12}

TABLE 1.1

Samp. No.	BPF Processed		2nd BPF Input	Samp. freq.	LPF Processed		2nd LPF Input	2nd LPF Output
1	16K	8K	1_{12}	32K	12K	6K	1_{12}	1_{6}
2	16K	4K	1_{6}	16K	12K	3K	1_{6}	1_{3}
3	16K	8K	3_{12}	32K	12K	6K	3_{12}	3_{6}
4	16K	2K	1_{3}	8K	12K	1.5K	1_{3}	$1_{1.5}$
5	16K	8K	5_{12}	32K	12K	6K	5_{12}	5_{6}
6	16K	4K	5_{6}	16K	12K	3K	5_{6}	5_{3}
7	16K	8K	7_{12}	32K	12K	6K	7_{12}	7_{6}
8	16K	1K	$1_{1.5}$	4K	12K	750	$1_{1.5}$	1_{750}
9	16K	8K	9_{12}	32K	12K	6K	9_{12}	9_{6}
...	...	...	...	...	...	...	...	...
16	16K	500	1_{750}	2K	12K	375	1_{750}	1_{375}
...	...	...	...	...	...	...	...	...
32	16K	250	1_{375}	1K	12K	187	1_{375}	1_{187}
...	...	...	...	...	...	...	...	...
64	16K	125	1_{187}	500	12K	94	1_{187}	1_{94}
...	...	...	...	...	...	...	...	...
128	16K	62	1_{94}	250	12K	47	1_{94}	1_{47}
...	...	...	...	...	...	...	...	...
256	16K	31	1_{47}	125	12K	--	1_{47}	--
...	...	...	...	...	...	...	...	...
512	16K	--	--	62	12K	--	--	--
...	...	...	...	...	...	...	...	...
768	16K	31	1_{47}	125	12K	--	1_{47}	--

designates the sample No. 3 after it has been processed by the 12 KHz LPF; 1_3 corresponds to the sample No. 1 processed by the 3 KHz LPF. All the frequencies in the table are in Hz. A few things should be noted in this table. The input to the 16 KHz BPF and 12 KHz LPF has not been indicated because it is the same for both and it is always the input sample to the processor. The input to the second BPF and LPF processed in each sample is also the same but it has been filtered by a LPF as indicated by the subindices. Note that the sequence from samples1 to 7 repeats again from samples 9 to 15; the sequence from 1 to 15 repeats from 17 to 31 and so forth. Therefore, we have indicated only the samples where a new filter is processed for the first time in the first 512 samples. The whole sequence repeats every 512 samples. By looking at the subindices of the input sample to the second BPF and LPF, we see that the 12 appears every 2 samples, the 6 every 4,the 3 every 8, etc. At the output of the LPF, on the other hand, the 6 appears every 2 samples but every other is discarded, as discussed before; the same pattern can be observed in the other samples. As the basic sampling frequency is 64 KHz or 4 times 16 KHz, the sampling frequency for every second BPF is 4 times its frequency.

4. SELECTION OF THE PROCESSOR

Even with only 4 filters required, the processing of all of them in real time is not feasible with the speed of presently available conventional microprocessors. One

approach could be the use of high speed bit slice sections to form a dedicated computer with adequate microprogramming. Another possibility is to use the recently introduced processors that perform the operations of parallel multiplication and addition in a single execution cycle. Examples of such processors are the NEC uPD7720 and the TRW TDC1010J, which can multiply 16 x 16 bits and accumulate the result in 250 nsec and 150 nsec, respectively.

For our project, however, we have chosen the Signal Processing Peripheral (SPP) AMI S2811, from American Microsystems Inc., which has a 12-bit multiplier, but can process 16-bit input words. We chose this processor because we intended to use a 12-bit analog to digital converter which can yield an acceptable 72 dB dynamic range at a moderate price. The AMI processor can perform the multiplication and accumulation of two 12-bit words in 300 nsec. We found this speed and all the conveniences offered by the architecture and instructions set of this processor quite suitable to our requirements (see Appendix for the specifications of the S2811)

5. BASIC BLOCK DIAGRAM

Figure 1.1 shows a basic block diagram of the equalizer. At the heart of the system is the SPP processor AMI S2811 preceded by a 12-bit Analog to Digital converter system and followed by a corresponding Digital to Analog converter system. The SPP processor has a data memory in the form of a matrix with 32 bases or rows of 8 locations each. Half of

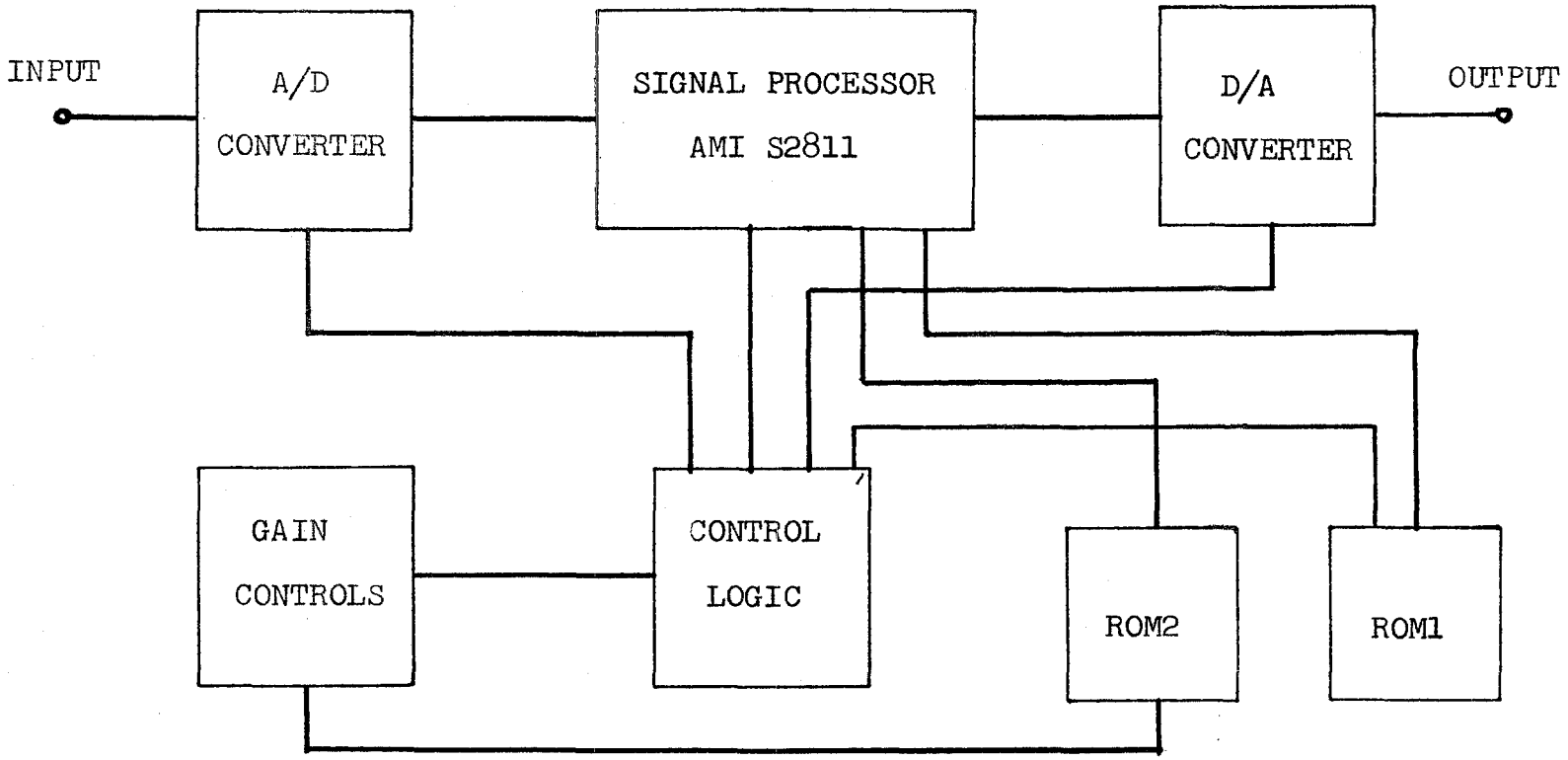


Figure 1.1. Basic Block Diagram

the 8 locations are RAM and half are ROM. We will store the data for our bandpass and lowpass filters in these bases. The ROM1 will keep the sequence of the filter bases in the order they should be processed according to the pattern of Table 1.1. The gain controls will set the gain for each bandpass filter by addressing one of the gain coefficients stored in ROM2 at the time the corresponding bandpass filter is processed. The control logic section will keep all the portions of the equalizer system properly synchronized and will supply the necessary control signals.

CHAPTER 2

DESIGN OF THE FILTERS

1. DIGITAL BANDPASS FILTERS

The bandpass filters to be implemented digitally in the SPP processor AMI S2811 are of the constant percentage bandwidth type, with their center frequencies spaced by octaves from 16 KHz down to 31 Hz. As it was discussed before, a single digital filter presented with input signals sampled at successively halved rates will produce responses with successively halved center frequencies and bandwidths. When the samples are time multiplexed in the sequence explained in Chapter 1, a bandpass filter designed for a 16 KHz center frequency and appropriate bandwidth at a 64 KHz sampling rate will generate an 8 KHz, half bandwidth filter when its input is sampled at 32 KHz; a 4 KHz, quarter bandwidth when sampled at 16 KHz, and so on.

These are exactly the desired results; octave frequency spacing and constant percentage bandwidth. Therefore, only one bandpass filter (BPF) needs to be implemented for the 10 bands of the equalizer. To avoid aliasing, however, the signals are band limited with suitable digital lowpass filters every time the sampling rate is halved. Again, only one digital lowpass filter (LPF) need be implemented since its

cut-off frequency will be halved each time the sampling rate is halved.

The processing of BPF and LPF is organized as shown in Figure 2.1, where the Gain coefficients, determined by the settings of the front panel band-gain controls, are also indicated.

The cut-off frequencies of the LPF have been chosen to permit the signals of interest to pass relatively unaltered while still allowing a reasonable roll-off toward the folding frequency (one half the sampling frequency). The first LPF has a cut-off frequency of 12 KHz and delivers its output to the 8 KHz BPF which processes signals sampled at 32 KHz. Therefore, the cut-off frequency of the LPF is well below the folding frequency of 16 KHz. Similarly, the rest of the LPF's have their cut-off frequencies related by octaves to the 12 KHz LPF by virtue of the successively halved sampling frequency.

Digital filters can be designed in the form of non-recursive, finite impulse response (FIR) or recursive, infinite impulse response (IIR) types. The latter are easier to implement and it is the type for which the SPP S2811 is particularly suited by reason of its architecture and programming capabilities. This is important due to the time limitations of our processing scheme. The S2811 can process a second order IIR filter in 1.5 usec, while a comparable performance FIR filter would take much longer.

Most of the digital filter design methods use a trans-

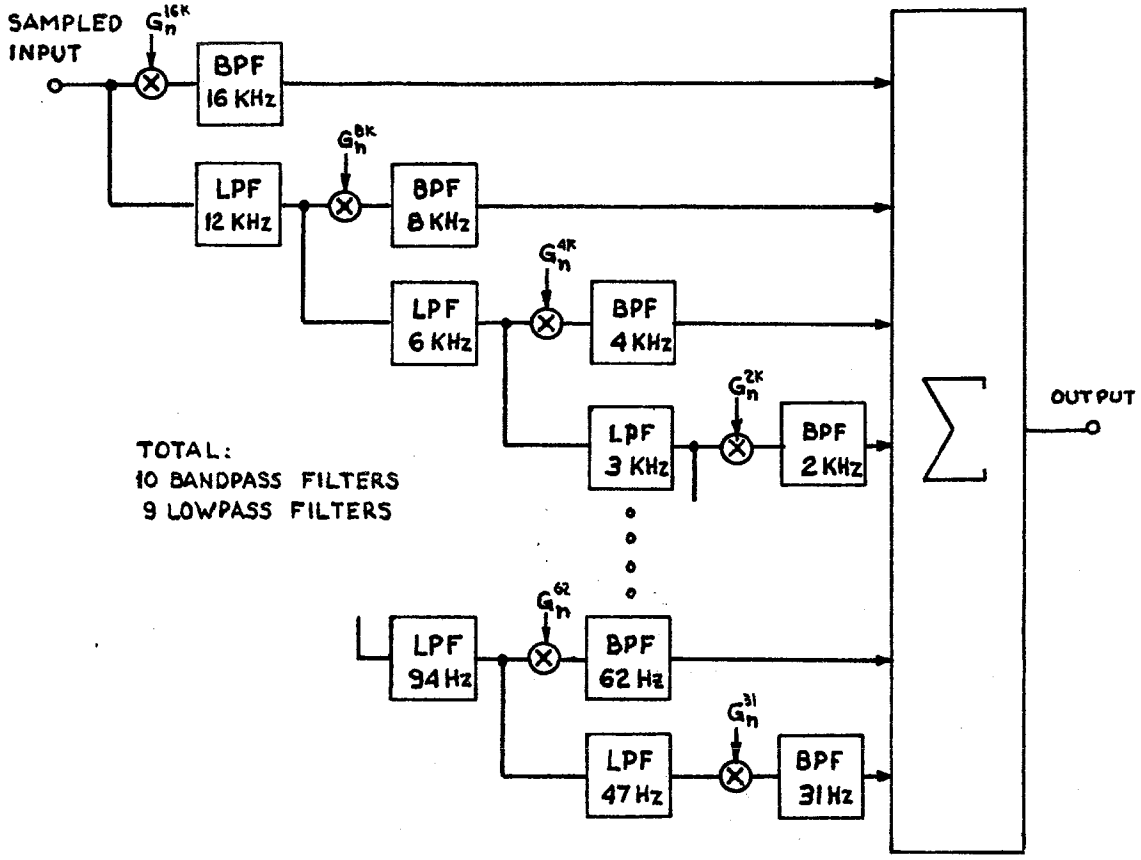


Figure 2.1 Processing sequence of BPF and LPF.

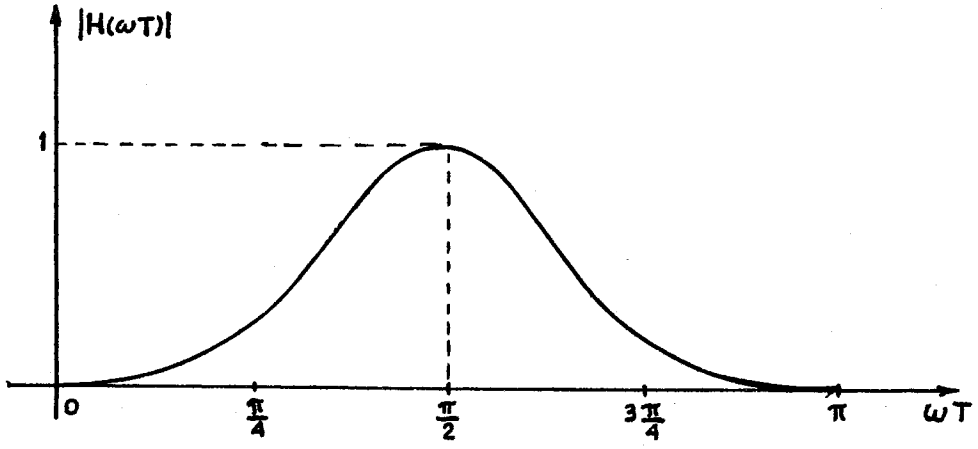


Figure 2.2 Amplitude response of a BPF.

formation from an analog, conventional type that has a performance related in some way to the desired digital filter. The transformation results in the change of a transfer function in the Laplace complex variable s to another function in the discrete complex variable z, with both variables being related by

$$z = e^{sT}$$

where T is the sampling period.

The impulse invariance method uses the z transform of the impulse response of the analog filter sampled at 1/T rate. The analog filter transfer function is of course, the Laplace transform of the same impulse response. With this method, therefore, the analog and digital filters have the same impulse response.

Another method is the so called matched z transformation where the poles and zeros in the s plane are mapped directly to poles and zeros in the z plane. Thus, for a pole or zero at $s = -a$ there is a pole or zero at $z = e^{-aT}$.

Both of these methods have the disadvantage that the response of the resultant digital filter can be significant at the folding frequency if the analog filter has zeros at frequencies higher than the folding frequency. Severe aliasing may result in those cases due to the periodicity of the digital filter response. This problem is circumvented with the bilinear transformation that maps the s plane into the z plane with the relation

$$s = C\frac{1 - z^{-1}}{1 + z^{-1}} \qquad (2.1)$$

If steady state conditions are considered then $s = j\Omega$ and $z = e^{j\omega T}$, that is, a unit circle in the z plane, and (2.1) takes the form

$$j\Omega = C\,\frac{1 - e^{-j\omega T}}{1 + e^{-j\omega T}} \qquad (2.2)$$

where Ω and ω are the analog and digital circular frequencies respectively, and C is a positive constant.

The above expression can be written as

$$j\Omega = C\,\frac{e^{-j\omega T/2}\,(e^{j\omega T/2} - e^{-j\omega T/2})}{e^{-j\omega T/2}\,(e^{j\omega T/2} + e^{-j\omega T/2})} = C\tanh j\frac{\omega T}{2}$$

or, since $\tanh jx = j \tan x$

$$\Omega = C\tan\frac{\omega T}{2} \qquad (2.3)$$

It is seen that the analog frequency Ω is mapped into the digital frequency ω through (2.3). Thus, for $\omega = 0$, $\Omega = 0$ and for $\omega = \pi/T$ (folding frequency) $\Omega \rightarrow \infty$. With LPF and BPF that have zeros at infinity, the response of the corresponding digital filter is then guaranteed to be 0 at the folding frequency and no aliasing will occur even though the response will be periodic with period $2\pi/T$. We choose this method of design for our digital bandpass and lowpass filters.

From the bilinear transformation of equation (2.1) we obtain with $s = \sigma + j\omega$

$$z = \frac{C + s}{C - s} = \frac{C + \sigma + j\omega}{C - \sigma - j\omega} \qquad (2.4)$$

It is seen that for $\sigma < 0$, $|z| < 1$ and viceversa. Then, the negative half of the s plane maps inside the unit circle on the z plane. For $\sigma = 0$, $|z| = 1$ for all values of ω and the entire $j\omega$ axis maps into the unit circle.

Due to the non-linear nature of the bilinear transformation, the typical method of design is to establish the critical frequencies of the desired digital filter, calculate the corresponding analog frequencies with (2.3) (prewarping of frequencies) and then find the transfer function of an analog filter that meets the response requirements at the prewarped frequencies. By applying the bilinear transformation to that transfer function, the transfer function of the matching digital filter results.

For our design, however, we shall use a different procedure starting off, somewhat arbitrarily, with a standard first order Butterworth bandpass filter and transforming it into a digital filter with the bilinear transformation. The reasoning behind this approach is that the final purpose of the audio equalizer is not to strictly separate the ten frequency bands but rather to shape the overall frequency response. Therefore, the responses of the filters are allowed to overlap and the Q of the filter is used as a parameter whose value is chosen to render a flat overall response when the magnitudes of all the filters are added together.

The transfer function of a first order Butterworth

bandpass filter in terms of the Q and the center frequency Ω_c is

$$H(s) = \frac{\frac{1}{Q}\Omega_c s}{s^2 + \frac{1}{Q}\Omega_c s + \Omega_c^2} \qquad (2.5)$$

To apply the bilinear transformation we must determine the constant C in (2.1). One way to do this is to select a value such that a particular frequency of the digital filter matches exactly a corresponding frequency of the analog filter, with the other frequencies being related by the transformation of equation (2.3).

In our case, the sampling frequency is chosen to be 64 KHz. The highest frequency filter has a center frequency of 16 KHz and we choose to match this frequency in the analog and digital responses. Therefore, from (2.3)

$$\Omega_c = 2\pi(16 \times 10^3) = C \tan \frac{2\pi(16 \times 10^3)}{2(64 \times 10^3)}$$

from where

$$C = \Omega_c \cot \frac{\pi}{4} = \Omega_c$$

Hence, the transformation relation is

$$s = \Omega_c \frac{1 - z^{-1}}{1 + z^{-1}}$$

Substituting in (2.5) and simplifying it results

$$H(z) = \frac{1 - z^{-2}}{2Q + 1 + (2Q - 1)z^{-2}} \qquad (2.6)$$

This is the transfer function of our digital bandpass filter. For steady state response we evaluate (2.6) on the unit circle by setting $z = e^{j\omega T}$. Then

$$H(\omega T) = \frac{1 - e^{-j2\omega T}}{2Q + 1 + (2Q - 1)e^{-j2\omega T}} \qquad (2.7)$$

The magnitude response can be found by setting

$$|H(\omega T)|^2 = H(\omega T)\ H^*(\omega T)$$

After some algebraic manipulations it results

$$|H(\omega T)| = \frac{\sin \omega T}{\sqrt{4Q^2 \cos^2 \omega T + \sin^2 \omega T}} \qquad (2.8)$$

The amplitude response of the filter for a given Q is sketched in Figure 2.2 as a function of the normalized digital frequency $\omega T = \omega / f_s$, where f_s is the sampling frequency. The response for all the other 9 filters will be identical to this one, as explained before, since for every octave the sampling period will be multiplied by a factor $N = 2^k$ and the frequencies will be correspondingly divided by the same factor to yield a constant ωT.

From equation (2.8) it is recognized that $|H(\omega T)| \leq 1$. The maximum value occurs for the center frequency ω_c, where $\omega_c T = \pi/2$ and $|H(\omega T)| = 1$. To display all the fil-

ters on the same (normalized) frequency axis we assume that the sampling period T is fixed and then we multiply ωT by $N = 2^k$ ($k = 0,1,...9$), or successive powers of 2 to obtain the response of the lower frequency filters. For example, with $k = 1$ ($N = 2$) the center frequency of the filter occurs for $\omega_c NT = \pi/2$ or $\omega_c T = \pi/4$, which corresponds to a denormalized frequency

$$\omega_c = 2\pi f_c = \frac{\pi}{4T} = 2\pi \frac{(64 \times 10^3)}{8}$$

or $f_c = 8$ KHz. Similarly, with $k = 2$ ($N = 4$) $\omega_c T = \pi/8$ and the center frequency is $f_c = 4$ KHz, and so on. Finally, with $k = 9$ ($N = 512$) the center frequency is $f_c = 31.25$ Hz. When this N factor is included, equation (2.8) takes now the general form

$$|H(\omega T)| = \frac{\sin \omega NT}{\sqrt{4Q \cos^2 \omega NT + \sin^2 \omega NT}} \qquad (2.9)$$

A few of the filters generated by varying the value of N in equation (2.9) are sketched in Figure 2.3 in a (linear) normalized frequency axis. The filters are numbered from right to left with k (0,1,2...) since the basic filter corresponds to the 16 KHz frequency. It will be noted that at any normalized frequency below $\pi/2$, the total output is contributed by the response of all the filters to the right and the one to the left of the frequency considered. Therefore, the total response at a frequency $\omega_1 T$ between the kth and the $(k+1)$th filter is given by (see Figure 2.3)

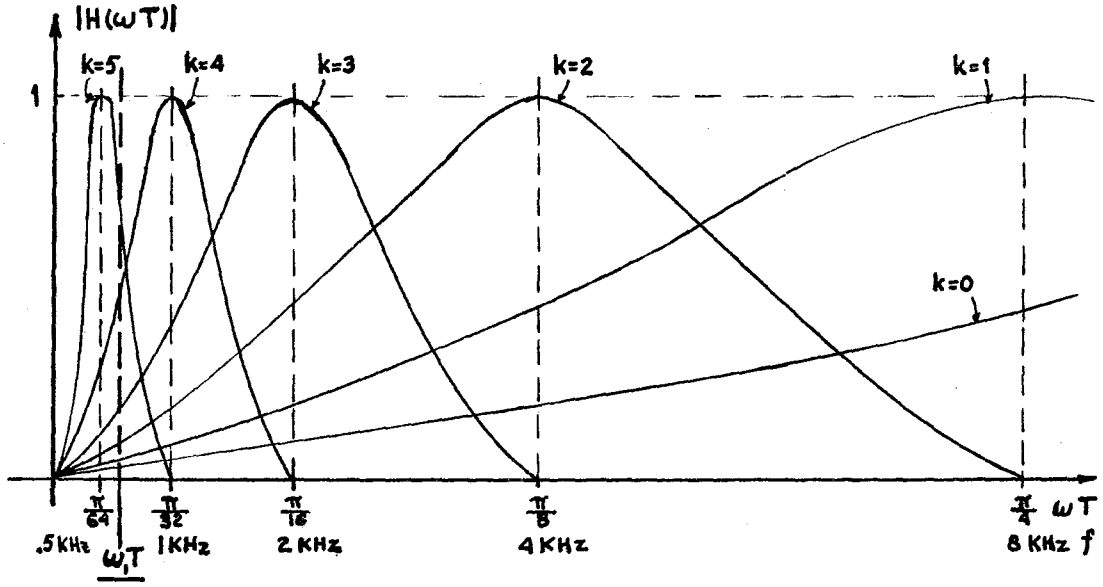


Figure 2.3 Response of the first 6 BPF's.

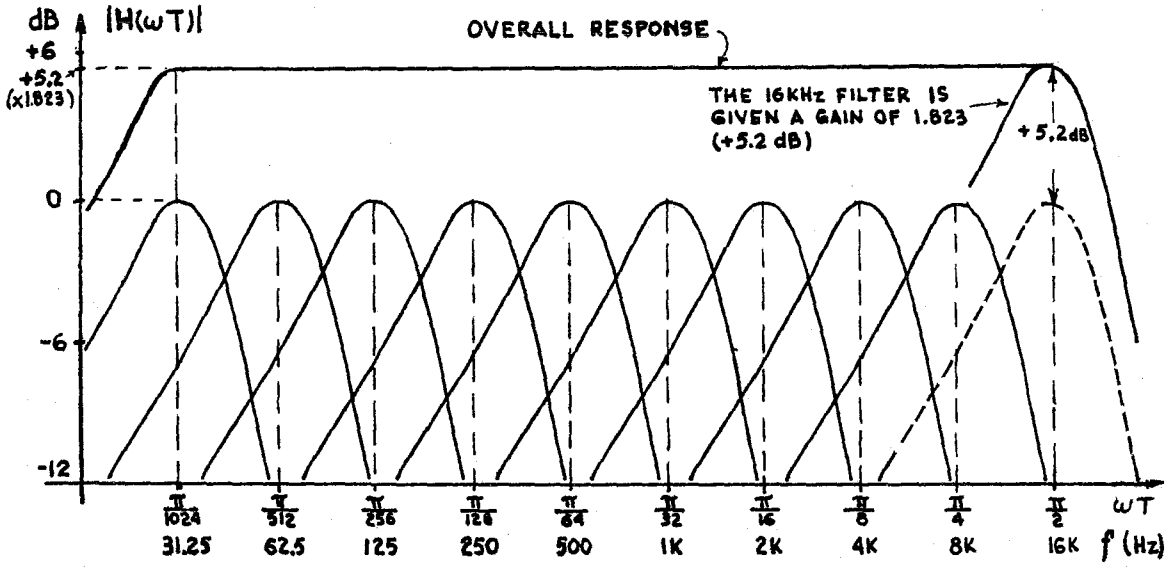


Figure 2.4 Response of all the BPF's in log scales.

$$\left|H(\omega_1 T)\right| = \sum_{n=0}^{k+1} \frac{\sin 2^n \omega_1 T}{\sqrt{4Q^2 \cos^2 2^n \omega_1 T + \sin^2 2^n \omega_1 T}} \qquad (2.10)$$

For low Q the response tends to have a sinusoidal shape and for practical purposes only the contribution of 4 or 5 filters to the right and one to the left of the given frequency need be considered.

Now we must determine the value of Q that will make the overall response as flat as possible. Due to the continuity of the responses we can do this numerically by trying different values of Q and calculating the response at several frequencies between the kth and the $(k+1)$th filter. We concluded that with a value of $Q = 1$ the response between two consecutive filters based in the contribution of 5 filters to the right and one to the left is within ± 0.25 dB, a quite acceptable result that is applicable to any pair of filters except the first four. Table 2.1 below shows the resultant ouput for 10 intermediate frequencies normalized to the frequency of the $(k+1)$th filter, that is, varying between 1 and 2. It can be seen that the average output is about 1.823 and that the maximum and minimum deviations are within the range of ± 0.25 dB.

For frequencies of $\pi/2$ (16 KHz) or larger only the first filter response contributes to the output. To compensate for the lower resultant output, the magnitude of this filter is multiplied by 1.823 to bring it up to the level of the other filters. In so doing, the total response at the

TABLE 2.1

Normalized Freq.	Total output between the k and k+1 filters*				
	k = 5 N = 32	k = 4 N = 16	k = 3 N = 8	k = 2 N = 4	k = 1 N = 2
1.0	1.84294	1.83865	1.83044	1.81692	1.81527
1.1	1.89557	1.89087	1.88194	1.86800	1.87432
1.2	1.88429	1.87918	1.86957	1.85548	1.87199
1.3	1.85386	1.84834	1.83809	1.82415	1.85289
1.4	1.83160	1.82568	1.81483	1.80138	1.84375
1.5	1.82528	1.81896	1.80754	1.79495	1.85106
1.6	1.83237	1.82566	1.81372	1.80239	1.87003
1.7	1.84600	1.83890	1.82649	1.81675	1.89025
1.8	1.85740	1.84993	1.83709	1.82962	1.89825
1.9	1.85750	1.84965	1.83645	1.83162	1.87977
2.0	1.83865	1.83044	1.81692	1.81527	1.82300

* The magnitude of the 16K BPF (N=2) is multiplied by 1.823

frequencies below $\pi/2$, where less number of filters are contributing to the output, is leveled with the response at all the other frequencies. The net effect is a uniform output accross the bands, as shown in Table 2.1.

Figure 2.4 shows the frequency response of all the filters plotted in a log frequency scale and with the output in dB's. The desired overall average gain is 1, but due to the

overlapping of the filters, it results about 1.823. Therefore, in the final implementation of the filters the gain coefficients of all of them, except the 16 KHz, are divided by 1.823.

With $Q = 1$, the final transfer function of our digital filter becomes

$$H(z) = \frac{1}{3} \frac{1 - z^{-2}}{1 + \frac{1}{3} z^{-2}} \qquad (2.11)$$

which has poles at $z_1 = (1/\sqrt{3})e^{j\,\pi/2}$ and $z_2 = (1/\sqrt{3})e^{-j\,\pi/2}$, and zeros at $z_3 = 1$ and $z_4 = -1$, and it is therefore stable (see Figure 2.5).

The final form of the filter is shown in Figure 2.6. The adjustable gain factors, which can be controlled from the front panel are integrated with the 1/3 factor of equation (2.11) to form the gain coefficient G_n.

The difference equations for the nth sample are obtained by partitioning the transfer function as follows

$$H(z) = \frac{Y(z)}{X(z)} = \frac{Y(z)}{W(z)} \frac{W(z)}{X(z)} = G_n \frac{1 - z^{-2}}{1 + \frac{1}{3} z^{-2}}$$

$$\frac{W(z)}{X(z)} = G_n \frac{1}{1 + \frac{1}{3} z^{-2}} \qquad \frac{Y(z)}{W(z)} = 1 - z^{-2}$$

from where the difference equations are

$$w_n = G_n . x_n - \frac{1}{3} w_{n-2} \qquad (2.12)$$

$$y_n = w_n - w_{n-2} \qquad (2.13)$$

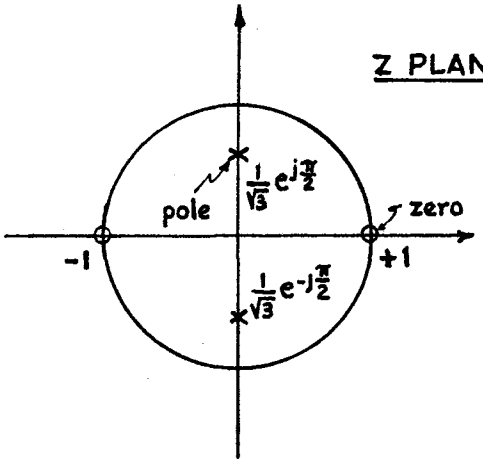


Figure 2.5 Poles and zeros of H(z) for BPF.

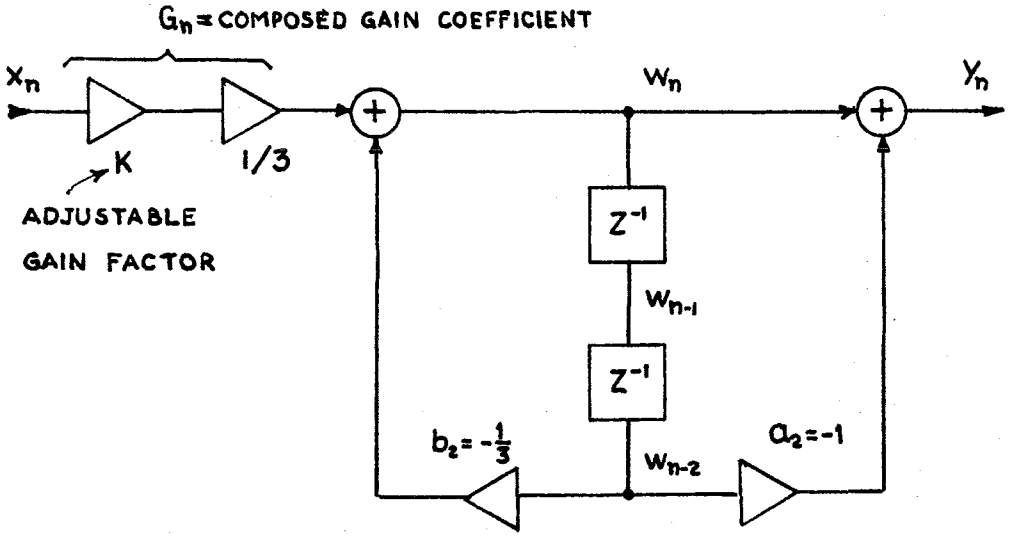


Figure 2.6 Implementation of BPF of eq. (2.12, 2.13)

The intermediate results w_{n-1} and w_{n-2} are indicated in Figure 2.6. This is the form of the difference equation that is implemented in the program for the processor AMI S2811.

Gain coefficients

For the calculation of the gain coefficients G_n we observe that 10 dB is equivalent to a factor of $\sqrt{10}$. However, we must take into account the fact that the output at a given frequency is contributed by the response of a few filters, mostly by the first to the right, that is, the next higher in frequency. We must emphasize at this point that the purpose of the equalizer is to shape the frequency response rather than to give an accurate output at each frequency. In practice, several adjacent BPF controls are usually adjusted to shape the response, so that the result depends on the combined effect of all of them.

To be more specific, we observe from Figure 2.3 that a given filter, say the 2 KHz, does not contribute any output at the center frequency of the higher order filter (4 KHz). The 4 KHz BPF, on the other hand, contributes about 45% of its output to the response at 2 KHz. Therefore, the gain control of the 2 KHz BPF affects only partially the overall response.

One way to improve the effect of the gain controls is to multiply the output of the next higher order filter (4 KHz in our example) by the same gain factor used for the BPF being processed. If this factor is 1, as in the 0 dB

gain setting, nothing will change. If the gain factor is not 1, the corresponding boosting or attenuation will be reinforced making more effective the action of the gain control. This multiplication is implemented in the program (see Chapter 3).

In practice, when only one gain control is adjusted, the resultant change will be 1 to 2 dB from the nominal setting. When two or more adjacent controls are adjusted, instead, the gains will be very close to the nominal values. To illustrate this point if, for example, we use a setting of -6 dB, that is, a factor of 0.5 in only one filter the overall attenuation at the filter frequency will be

$$20 \log \frac{0.5 + (0.823).0.5}{1.823} = -4.5 \text{ dB}$$

where 1.823 is the reference output, as we know, and the values within parenthesis are the contribution of the BPF to the right of the one considered. If the adjacent BPF is also set to -6 dB, then the attenuation at the filter frequency will be

$$20 \log \frac{0.5 + (0.45)0.5^2 + (0.2)0.5 + (0.173)}{1.823} = -6.2 \text{ dB}$$

where the values within parenthesis are again the contribution of the BPF to the right, that is, those of higher frequencies.

The gain coefficients are then given by

$$G_n = \frac{1}{3} \log^{-1} \frac{A}{20}$$

where A is the gain setting in dB's and the factor 1/3 is the coefficient of equation (2.11) which, as we said, was to be integrated with G_n.

There is another fact that must be taken into consideration. The SPP AMI S2811 only accepts numbers in the range -1 to +1. But for the highest setting of +10 dB, the factor is $\sqrt{10}$ which is larger than 1. If this factor is used, the output of the respective BPF will be larger than 1. To avoid this possibility, we divide all the coefficients by $\sqrt{10}$ and to make up for the difference we give a gain of $\sqrt{10}$ or +10 dB to the output analog lowpass filter. Now the gain coefficient becomes

$$G_n = \frac{1}{3\sqrt{10}} \log^{-1} \frac{A}{20}$$

For example, with A = +10 dB, $G_n = 1/3$. These numbers have to be stored in an 8-bit wide ROM in 2's complement form. We find that the 2's complement representation giving the minimum error (less than $\frac{1}{2}$LSB) is 00101011 equivalent to decimal 0.3359. All the other coefficients are calculated in like manner and their values are included in Table 4.2 of Chapter 4 (page 87)along with the ROM addresses where they are stored.

As we said before, the 16 KHz BPF gain must be 1.823 times higher than the others to obtain a uniform response. This is implemented in the programming of the SPP by multiplying the gain of all the other filters by 1/1.823, while keeping the gain of the 16 KHz BPF at 1.

2. DIGITAL LOWPASS FILTERS

Every digital bandpass filter, except the one for the 16 KHz band, is preceded by a digital lowpass filter that limits the bandwidth of the sampled signal to below the folding frequency. For example, the 8 KHz BPF that receives the audio signal sampled at 32 KHz is preceded by a LPF with a cut-off frequency of 12 KHz and sufficient attenuation at the folding frequency of 16 KHz. The 4 KHz BPF nas a 6 KHz LPF infront of it, and so forth. The 16 KHz BPF, on the other hand, receives the input signal sampled at 64 KHz and band limited to 21 KHz by an analog LPF (antialiasing filter) placed at the input of the equalizer, before the analog to digital conversion.

For the design of the digital lowpass filters we will use again the bilinear transformation, following this time the conventional method of design. The design is carried out for the first LPF with a cut-off frequency of 12 KHz and all the others are obtained by successively halving the sampling frequency, as we did with the bandpass filters (see Figure 2.1).

The specifications we set for our basic digital LPF are as follows:

Cut-off frequency, 12 KHz

Edge of stop band, 16 KHz

Attenuation in the stop band, > 30 dB

Ripple in the passband, < 0.5 dB

First, we normalize the digital frequencies to the sampling

frequency $f_s = 64$ KHz. Then

$$\text{Digital cut-off freq.} \quad \omega_c T = \frac{2\pi(12 \times 10^3)}{(64 \times 10^3)} = \frac{3}{8}\pi$$

$$\text{Edge of stop band} \quad \omega_s T = \frac{2\pi(16 \times 10^3)}{(64 \times 10^3)} = \frac{\pi}{2}$$

By applying the bilinear transformation of equation (2.3) we obtain the corresponding prewarped analog frequencies

$$\Omega_c = C \tan \frac{3}{8}\frac{\pi}{2} \tag{2.14}$$

$$\Omega_s = C \tan \frac{\pi}{4} \tag{2.15}$$

The resultant (prewarped) analog transition ratio is

$$r = \frac{\Omega_s}{\Omega_c} = \frac{\tan \frac{\pi}{4}}{\tan \frac{3}{16}\pi} = 1.4966$$

Hence, we must find an analog filter with a transition ratio of 1.4966, an attenuation in the stop band of more than 30 dB and a maximum ripple in the passband of 0.5 dB. From a handbook of filter design tables (Reference 8, p.66) we find that our specifications are met by a fourth order elliptic filter with a transition ratio of 1.4945, stopband attenuation of 33.5 dB and passband ripple of 0.28 dB. This filter has the standard factored form

$$H(s) = K \frac{s^2 + A_0}{s^2 + B_1 s + B_0} \cdot \frac{s^2 + C_0}{s^2 + D_1 s + D_0} \qquad (2.16)$$

where the values of the coefficients for the normalized ($\Omega_c = 1$ rad/sec) transfer function are

$K = 0.02124958$ $\qquad$ $C_0 = 11.98287105$

$A_0 = 2.51603207$ $\qquad$ $D_0 = 0.58946535$

$B_0 = 1.12249240$ $\qquad$ $D_1 = 1.08608904$

$B_1 = 0.28990329$

We see that this transfer function represents two cascaded second order filters. The corresponding digital filter has the general form

$$H(z) = K \frac{a_0 + a_1 z^{-1} + a_2 z^{-2}}{1 + b_1 z^{-1} + b_2 z^{-2}} \cdot \frac{c_0 + c_1 z^{-1} + c_2 z^{-2}}{1 + d_1 z^{-1} + d_2 z^{-2}} \qquad (2.17)$$

To apply the bilinear transformation from s to z of equation (2.1) we need to select the coefficient C. We do this by choosing to match the normalized digital cut-off frequency $\omega_c T = (3/8)\pi$ to the normalized analog cut-off frequency $\Omega_c = 1$ rad/sec. Then

$$\Omega_c = 1 \text{ rad/sec} = C \tan \frac{\omega_c T}{2} = C \tan \frac{3\pi}{16}$$

or $$C = 1.4966$$

Using this value of C, we can insert in equation (2.16) the expression for s in terms of z of equation (2.1), that

is, the bilinear transformation, to obtain the coefficients for H(z) in (2.17). This is facilitated by using the following relations taken from Reference 4 , p. 174.

For the first second order section

$$A = B_0 + B_1 C + B_2 C^2$$
$$a_0 = (A_0 + A_1 C + A_2 C^2)/A$$
$$a_1 = (2A_0 - 2A_2 C^2)/A$$
$$a_2 = (A_0 - A_1 C + A_2 C^2)/A$$
$$b_1 = (2B_0 - 2B_2 C^2)/A$$
$$b_2 = (B_0 - B_1 C + B_2 C^2)/A$$

In our case $A_1 = 0$ and $A_2 = B_2 = 1$. Equivalent relations can be applied to the 2nd second order section. When th ulations are carried out and the gain coefficient K is integrated with the z^{-1} coefficients, the following transfer function results

$$H(z) = \frac{1 + a_1 z^{-1} + z^{-2}}{1 + b_1 z^{-1} + b_2 z^{-2}} \cdot \frac{c_0 + c_1 z^{-1} + c_0 z^{-2}}{1 + d_1 z^{-1} + d_2 z^{-2}} \qquad (2.18)$$

where the coefficients have the following values:

$a_0 = a_2 = 1$ $\qquad$ $c_0 = c_2 = 0.084994$

$a_1 = 0.116153$ $\qquad$ $c_1 = 0.116448$

$b_1 = -0.588662$ $\qquad$ $d_1 = -0.740947$

$b_2 = 0.771418$ $\qquad$ $d_2 = 0.270239$

Figure 2.7 shows the diagramatic representation of the two second order sections. The intermediate values as well

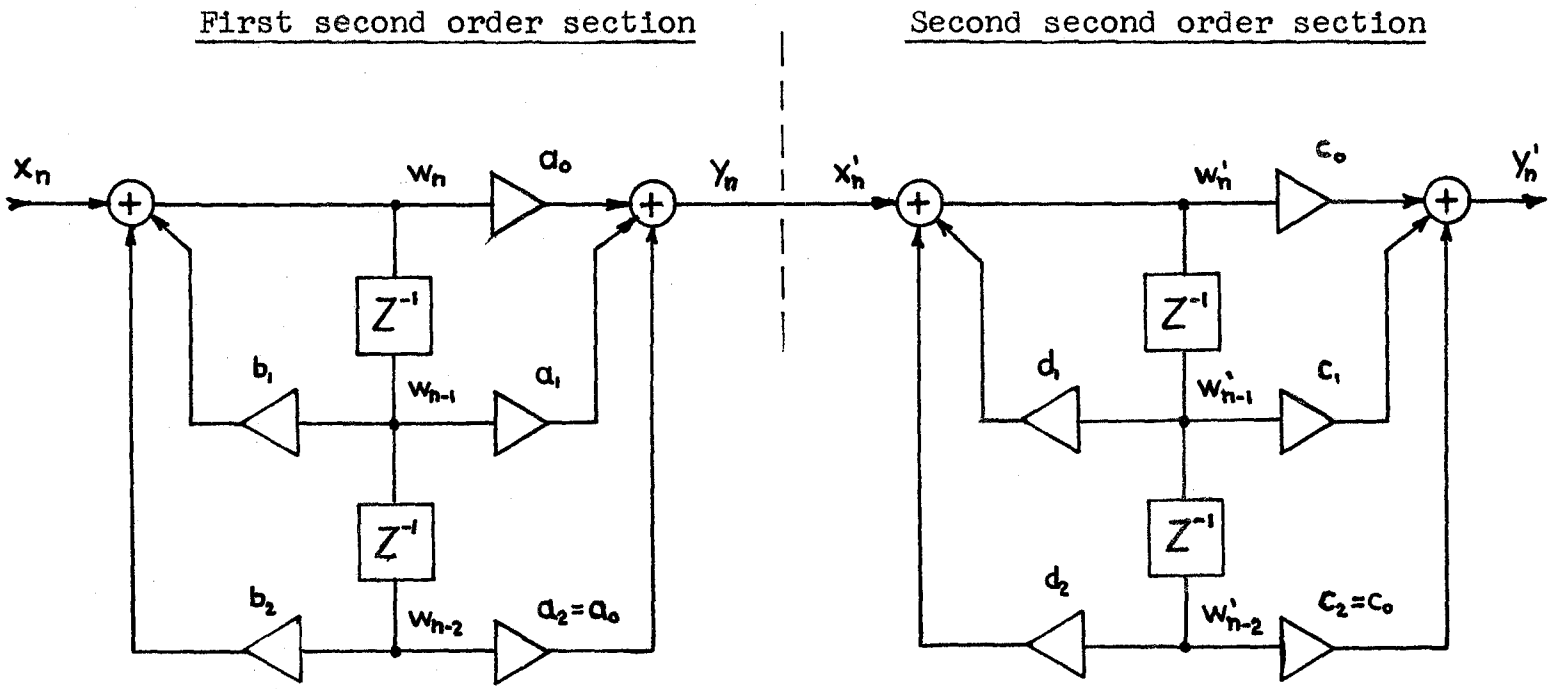


Figure 2.7 Implementation of the 4th order LPF of eq. (2.19, 2.20)

as the input and output are primed for the second section.

From equation (2.18) the poles of the first section are found to be $z_{1,2} = 0.88 \angle \pm 70.4^{\circ}$, and those of the second section $z_{3,4} = 0.52 \angle \pm 44.5^{\circ}$. Since both pair of poles are inside the unit circle the filter is stable.

The difference equations for the nth. sample, derived from (2.18) by the same partitioning of the transfer function used in the BPF are

First section

$$\begin{aligned} w_n &= x_n - b_1 w_{n-1} - b_2 w_{n-2} \\ y_n &= w_n + a_1 w_{n-1} + w_{n-2} \end{aligned} \qquad (2.19)$$

Second section

$$\begin{aligned} w'_n &= x'_n - d_1 w'_{n-1} - d_2 w'_{n-2} \\ y'_n &= c_0 w'_n + c_1 w'_{n-1} + c_0 w'_{n-2} \end{aligned} \qquad (2.20)$$

These are the forms of the equations to be implemented as two cascaded second order sections in the programming of the SPP processor S2811. All the coefficients are between -1 and +1 and therefore no scaling is required.

3. ANALOG LOWPASS FILTERS

For the analog input (antialiasing) and output (smoothing) lowpass filters our specifications can be set as follows. In order to have a reasonable roll-off we will choose a cut-off frequency of about 20 KHz and an attenuation of about 30 dB at the folding frequency of 32 KHz. The rationale for these choices is that our input signal is bandlimited below 20 KHz first, because the spectral content of the musical material has inherently very little energy beyond 15 KHz and second, because the transmission media, FM broadcasting, records and tapes have bandwidths no wider than 20 KHz at best. For these reasons, the signal energy at 32 KHz is practically nil and 30 dB of attenuation at that frequency and beyond is more than sufficient to prevent aliasing.

For the output lowpass filter the same considerations are valid and besides the bandwidth of the output signal is limited to well below 32 KHz due to the response of the bilinear transformed bandpass filters. We therefore use the same filter design for the analog input and output lowpass filters. To complete our specifications we define a maximum ripple of 0.1 dB in the passband.

The transition width, normalized to the cut-off frequency is $TW = (32 - 20)/20 = 0.6$. From a Handbook of Active Filters (Reference 9) we find that an elliptic, 4-pole lowpass filter with 0.1 dB ripple and 30 dB attenuation in the stopband has a transition width of 0.53. Then, we can use

a cut-off frequency of 32 KHz/1.53 = 21 KHz. From the same handbook we obtain the transfer function of the filter

$$H(s) = K1(c_1/a_1)\frac{s^2 + a_1\omega_c}{s^2 + b_1\omega_c s + c_1\omega_c^2}\ K_2(c_2/a_2)\frac{s^2 + a_2\omega_c}{s^2 + b_2\omega_c s + c_2\omega_c^2}$$

where K_1 and K_2 are the gains of the second order sections and the coefficients have the following values:

$a_1 = 2.644233$ $\qquad$ $a_2 = 12.746572$

$b_1 = 1.392323$ $\qquad$ $b_2 = 0.355395$

$c_1 = 0.854559$ $\qquad$ $c_2 = 1.261686$

The cut-off radian frequency is $\omega_c = 2\pi(21 \times 10^3)$. Gains are $K_1 = K_2 = 1$ for the input filter and $10^{\frac{1}{4}}$ for the output filter.

For easy of tuning we choose a biquad topology for each of the second order sections. Figure 2.8 is the schematic of the complete filter whose component values are calculated as follows, using the procedure of Reference 9, p. 65.

$K_1 = K_2 = 1$ for the input filter. $K_1 = K_2 = 10^{\frac{1}{4}}$ for the output filter. All the values are the same for the input and output filters, except R_{11}, R_{14} and R_{15}. All values are in ohms. The closest standard 1% values are indicated.

A) First second order section

Value of $C_{11} = C_{12}$ selected as 470 pF

Using the values of ω_c and coefficients indicated above.

$$R_{11} = \frac{a_1}{K.b_1.c_1.\omega_c.C_{11}} = 35.8K \text{ (use 36.5K) input}$$

$$20.1K \text{ (use 20.5K) output}$$

$$R_{12} = \frac{1}{b_1 \cdot \omega_c \cdot C_{11}} = 11.6K \text{ (use 11.5K)}$$

$$R_{13} = \frac{1}{\sqrt{c_1} \cdot \omega_c \cdot C_{11}} = 17.4K \text{ (use 17.8K)}$$

$$R_{17} = \frac{1}{\omega_c \cdot C_{11}} = 16.1K \text{ (use 16.2K)}$$

$$R_{14} = \frac{a_1 \cdot R_{17}}{K \cdot c_1} = 50.1K \text{ (use 51.1K)} \quad \text{input}$$

$$28.2K \text{ (use 28.7K)} \quad \text{output}$$

$$R_{15} = \frac{1}{K \cdot \sqrt{c_1} \cdot \omega_c \cdot C_{12}} = \frac{R_{13}}{K} = 17.8K \quad \text{input}$$

$$10K \quad \text{output}$$

$$R_{16} = \frac{C_{11}}{C_{12}} R_{13} = R_{13} = 17.8K$$

Resistors R_{10}, R_{18} and R_{19} are calculated for lowest DC offset as the approximate parallel value of the input and feedback resistors, resulting

$R_{10} = 6.9K$ (use 6.8K, 5%) input

$6K$ (use 6.2K, 5%) output

$R_{18} = 8.9K$ (use 9.1K, 5%) input

$6.4K$ (use 6.2K, 5%) output

$R_{19} = 8.1K$ (use 8.2K, 5%)

B) Second second order section

Same values of K. The component values for the second section are calculated as the first, using the appropriate coefficients. The results are:

$C_{21} = C_{22} = 470$ pF

$R_{21} = 458K$ (use 464K) input
$257K$ (use 261K) output

$R_{22} = 45.3K$ (use 44.2K)

$R_{23} = 14.3K$ (use 14.0K)

$R_{27} = 16.1K$ (use 16.2K)

$R_{24} = 163K$ (use 162K) input
$92K$ (use 90.9K) output

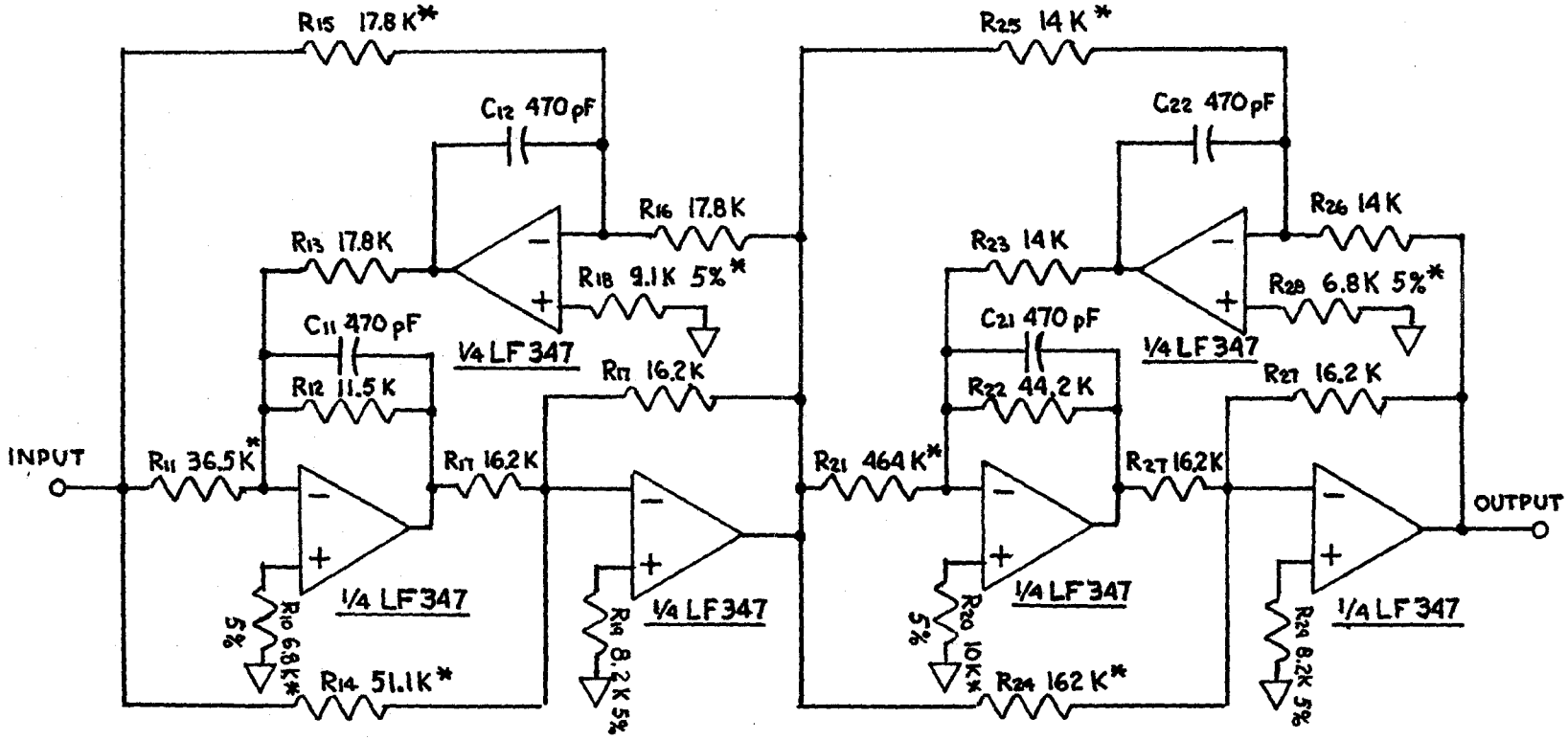
$R_{25} = 14.0K$ input
$7.87K$ output

$R_{26} = R_{23} = 14.0K$

$R_{20} = 10.6K$ (use 10K, 5%) input
$10.4K$ (use 10K, 5%) output

$R_{28} = 7K$ (use 6.8K, 5%) input
$5K$ (use 5.1K, 5%) output

$R_{29} = 8.1K$ (use 8.2K, 5%)



NOTES

1. ALL RESISTOR VALUES IN OHMS, 1% UNLESS OTHERWISE INDICATED
2. RESISTOR VALUES ARE FOR INPUT LPF. VALUES MARKED * ARE DIFFERENT IN THE OUTPUT LPF

Figure 2.8 Schematic of 4th order biquad elliptic LPF (LPF1, LPF2)

The requirements for the operational amplifiers are moderate. The input impedance should be large, at least 10 times the largest input resistor, or about 5 MΩ. The open loop gain at the cut-off frequency f_c, should be at least 50 times the gain of the filter, that is, 100 times or 40 dB in our case and its slew rate should be at least $\frac{1}{2}V_o \omega_c \times 10^{-6}$ V/usec, where V_o is the p-p output voltage, in our case 5V, giving a slew rate of 0.33 V/usec.

From the many types of op amps available we chose the National Semiconductor LF347, a quad JFET op amp with very high input impedance (10^{12} ohms), high slew rate of 13V/usec and open loop gain at 21 KHz in excess of 40 dB for $\pm$15V supply. The distortion is very small, less than 0.02%, a good feature for our application. Since we need 2 filters or 12 op amps, 3 DIP would be sufficient for our implementation.

CHAPTER 3

SOFTWARE DESIGN

1. ARCHITECTURE AND PROGRAM FEATURES OF THE AMI S2811

The architecture and instruction set of the AMI S2811 SPP (Signal Processing Peripheral) is specially suited for the efficient processing of digital filters at high speed. At the clock frequency of 20 MHz, each instruction takes 300 nsec and due to the pipelined structure, only 5 instructions or 1.5 usec are required to process a second order recursive digital filter. In addition, two buffered serial ports clocked independently from the main processor, permit the input and output of data simultaneously with the program run. There is also an 8-bit bus to allow the input and output of data in parallel form. In all cases, the input data is held in the Input Register (IR) and the output data is placed in the Output Register(OR).

From the S2811 specifications sheet, it can be seen that the memory for the data is organized in a matrix of 32 Bases or rows, with each base having 8 locations of 16 bits each called displacements and numbered from 0 to 7 as illustrated in Figure 3.2. One half of this memory, comprising the displacements 0 to 3 of all the bases is RAM and the other half with locations 4 to 7 is mask programmable ROM. There is

also an 8-byte scratch pad RAM an of course, an instruction ROM capable of holding 256 programming steps.

The main memory can be accessed from two ports, labeled U and V and what makes this arrangement useful is that the two ports can be addressed simultaneously in one instruction, as long as the respective operands are stored in the same base. Alternatively, the scratch pad (S) can substitute for the port V in any instruction.

The instruction set has two type of operations that can be specified simutaneously in a single program step, except for a few cases of incompatibility. The OP1 type of instruction includes all the arithmetic operations such as APA (Add Product and Accumulator), that adds the contents of the accumulator to the product of two multiplicands placed at the input of the multiplier in the previous instruction. The OP2 type of instruction refer to operations of loading, such as TIRV (Transfer Input Register to V location in RAM) or LACO (Load Accumulator in the Output Register); branching as in JMIF (Jump if I.F. is not set) and some miscellaneous such as DECB and INCB (Decrement and Increment the Base respectively).

The instruction word format contains two fields for the operations OP1 and OP2 and a field for the operand address. There are four addressing modes. One is the UV or US type, where two operands of the same Base or one operand from the Base (U) and one from the Scratch pad (S) are accessed at the same time. The Base Register that points to the Base No.

has to be set previously by inserting the desired Base No. through the instruction LIBL (Load the Input Register to the Base Register and Loop Counter) or by incrementing or decrementing the current Base No.

Other mode of addressing is the Direct Addressing D(XX.X) where the first two digits indicate the Base and the third, the Displacement. Only one operand can be accessed with this mode. The third mode is the Direct Transfer DT where a branching to a given address is specified, as in a Jump instruction. The fourth mode is called Literal an it just includes an OP1 operation and a data word in the instruction.

A four line bus F can be used to control with external hardware (typically a microprocessor) several different modes of operation, such as setting the Serial Input or the Serial Output, addressing the upper byte of a two bytes word etc. When the SPP is not used in conjunction with a microprocessor, as in our case, those modes can be set from the program with the OP2 intruction MODE.

One useful feature of the SPP S2811 is the VP register that stores the operand from the V port specified in one instruction and makes it available in the following instruction with the command TVPV (Transfer contents of the VP register to V location in RAM) or TVIB (the same but incrementing the Base at the same time). This feature implements in software the z^{-1} delays of the digital filters.

It should be noted that the specified U and V (or S)

memory locations are placed at the input of the Multiplier during the same instruction, but the product is available in the following instruction. This product can be added to or substracted from the contents of the accumulator.

Besides the parallel bus and the serial input and output ports, the S2811 has an active LOW $\overline{RST}$ input that clears all the registers, including the Base (not the RAM) and starts the program execution at the first instruction; an Interrupt Request ($\overline{IRQ}$) output that goes LOW upon the execution of the instruction JMIF, that is, jump if the Input Flag (IF) is not set, indicating that no input of data has taken place, and an Interface Enable ($\overline{IE}$) input to enable the parallel bus. The $\overline{IRQ}$ output can also be set with the instruction JMOF, i.e., jump if the Output Flag (OF) is set.

2. MEMORY MAPPING

The flow chart of Figure 3.1 indicates the basic steps followed in the program. Two bandpass filters (BPF) and two lowpass filters (LPF) are processed in every program run, or equivalently, in every sampling period.

The memory mapping of Figure 3.2 shows how each BPF is stored in one base with the coefficients in ROM and the input sample and intermediate values in RAM. The LPF's, being of fourth order, require two bases for storage. A few points should be noted in the arrangement of the memory locations.

The highest frequency BPF (16 KHz) occupies the Base No. 0, followed by the highest frequency LPF (12 KHz) which

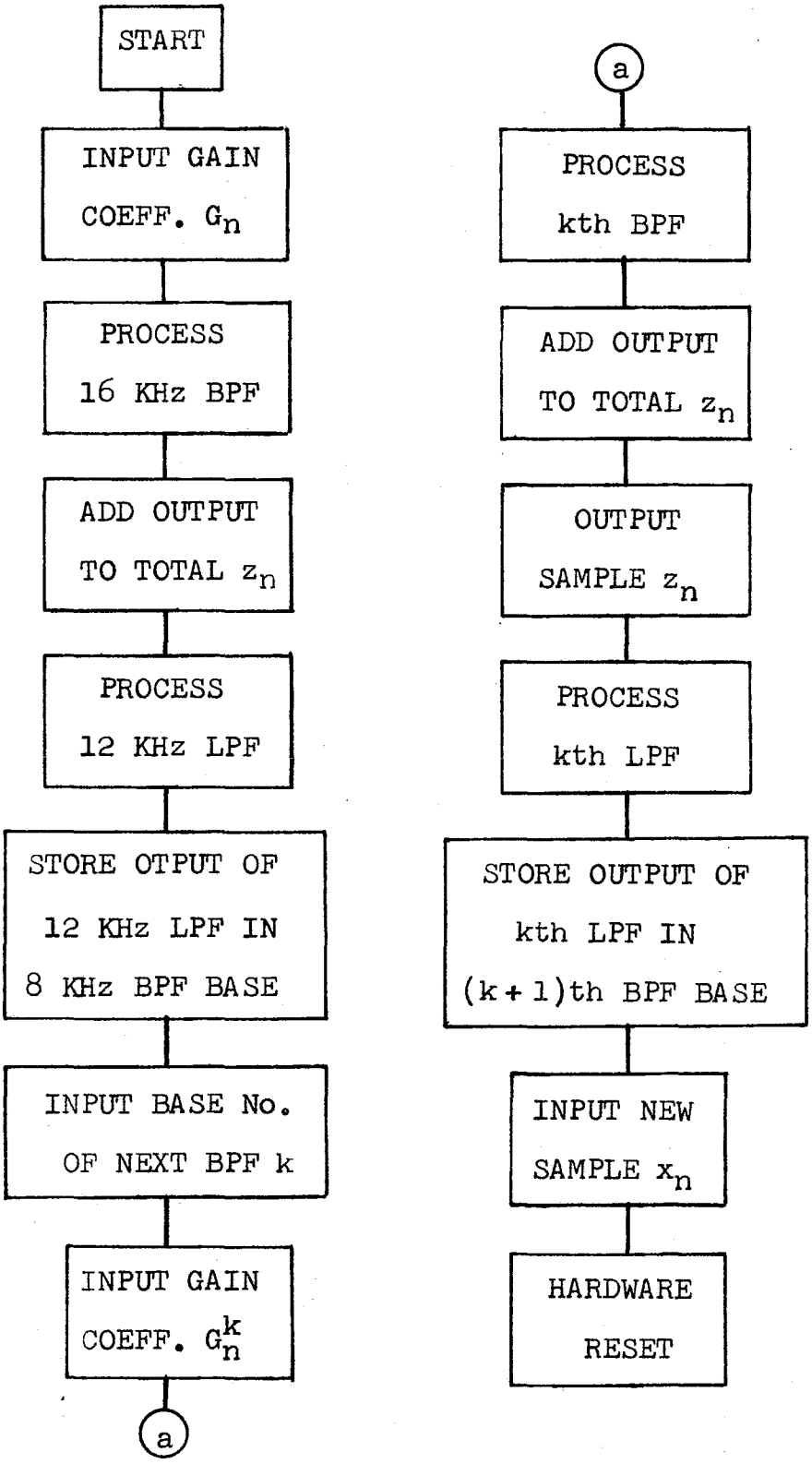


Figure 3.1 Flow chart of the signal processing

is taking Bases Nos. 1 and 2. The rest of the filters' data are stored in decreasing order of frequency; first one BPF followed by the LPF which processes the same sample according with the sequence depicted in Figure 2.1. The output of the LPF is placed in the next Base, to be used as the input "sample" for the next lower order BPF and LPF. This way, only the Base No. for the BPF needs to be inputted; the bases for the LPF are accessed by incrementing the Base No. with the instruction INCB.

Refering to the memory mapping of Figure 3.2 and the comments in the program, the notation x_n indicates the sample received from the Serial Input port which is always stored in location (00.0). The notation x_n^k denotes the sample after being filtered by the kth. LPF, that is, the output of that LPF. Other notations correspond to the nomenclature used in the respective difference equations of Chapter 2.

Due to limitations in the RAM space, the output of each BPF, such as y_n^{16K} is stored in the next lower Base and since it has to be saved for the next program run it becomes y_{n-1}^{16K}, as noted. The sum of all the BPF outputs, z_n, is stored in location 0 of the scratch pad memory (not shown in Figure 3.2). This sum, which constitutes the output of the audio equalizer after the D/A conversion, is updated after the processing of each BPF by adding to it the current output of the BPF and substracting the previous one. That is why those outputs must be saved in each program run.

Base No	Filter	Displacement 0	1	2	3	4	5	6	7
0	16 KHz BPF	x_n	G_{n-1}^{16K}	w_{n-1}	w_{n-2}	-1/3	1		
1	12 KHz LPF		y_{n-1}^{16K}	w_{n-1}	w_{n-2}	1	a_1	$-b_1$	$-b_2$
2	12 KHz LPF			w'_{n-1}	w'_{n-2}	c_o	c_1	$-d_1$	$-d_2$
3	8 KHz BPF	x_n^{12K}	G_{n-1}^{8K}	w_{n-1}	w_{n-2}	-1/3	1	K	
4	6 KHz LPF	G_{n-1}^{4K}	y_{n-1}^{8K}	w_{n-1}	w_{n-2}	1	a_1	$-b_1$	$-b_2$
5	6 KHz LPF			w'_{n-1}	w'_{n-2}	c_o	c_1	$-d_1$	$-d_2$
6	4 KHz BPF	x_n^{6K}	G_{n-1}^{4K}	w_{n-1}	w_{n-2}	-1/3	1	K	
24	62.5Hz BPF	x_n^{94}	$G_{n-1}^{62.5}$	w_{n-1}	w_{n-2}	-1/3	1	K	
25	47 Hz LPF	G_{n-1}^{31}	$y_{n-1}^{62.5}$	w_{n-1}	w_{n-2}	1	a_1	$-b_1$	$-b_2$
26	47 Hz LPF			w'_{n-1}	w'_{n-2}	c_o	c_1	$-d_1$	$-d_2$
27	31 Hz BPF	x_n^{47}	G_{n-1}^{31}	w_{n-1}	w_{n-2}	-1/3	1	K	

NOTES: x_n^{12K} denotes the sample filtered by the 12K LPF; etc.
y_{n-1}^{8K} and G_{n-1}^{8K} denotes the output and gain factor respectively for the 8K BPF; etc. K= 1/1.823

Figure 3.2. Memory mapping of SPP processor

3. THE PROGRAM

The following pages list the program steps with some brief comments related to the software implementation of equations (2.12), (2.13), (2.19) and (2.20). The SPP processor instruction set allows a lot of flexibility in the development of the program steps required to process the digital filters.

This portion of the project was one of major difficulties since the program had to be written in the most efficient way to compress in a short period of time (15.6 usec) the processing of two 2nd order BPF and two 4th order LPF plus the accumulation of the BPF outputs and overhead I/O instructions. The ability of the SPP to execute in one instruction the multiplication of two operands and the accumulation of the product, while the serial input and output are operating independently, permitted to achieve the necessary processing speed.

The flow chart of Figure 3.1 shows in block form the basic steps followed in the program. The interrelation between the program steps and the hardware operations is clarified in the Program Events timing of Figure 4.2, Chapter 4. In fact, both, the software and hardware operations had to be correlated so that the proper data is available at the proper time in the proper place. Figure 4.2 helps to understand the timing requirements of the relevant signals that are later implemented in the peripheral hardware, as explained in Chapter 4.

PROGRAM

STEP	LABEL	OP1	OP2	OPERAND	COMMENTS
1		DUH	MODE	UV(3,4)	8-bit Mode set. Base=0 w_{n-2}, $a_2(-1/3) \rightarrow$ MULT.
2	AGAIN	NOP	JMIF	DT(AGAIN)	Waits for G_n coeff. input. Sets $\overline{IRQ}$.
3		APZ	TIRV	UV(0,1)	$w_{n-2}.a_2 \rightarrow$ACC. $G_n \rightarrow$V(1).(Clears I.F.) x_n, $G_{n-1} \rightarrow$ MULT.
4		APA	TACV	UV(-,2)	$w_n = x_n.G_{n-1}+a_2.w_{n-2} \rightarrow$ACC ACC($w_n$)$\rightarrow$ V(2). $w_{n-1} \rightarrow$VP.
5		SVA	TVIB	UV(-,3)	$w_{n-2} - w_n = -y_n \rightarrow$ ACC. $w_{n-1} \rightarrow$V(3).Base incr.
6		NEG	TACV	UV(4,1)	$-y_n \rightarrow y_n$; $y_n \rightarrow$ V(1),Base=1 1, $y_{n-1} \rightarrow$ MULT.
7		SPA	DECB		$1.y_{n-1} - y_n \rightarrow$ ACC. Base decremented.
8		SVA	TACV	US(-,0)	$z_n=z_{n-1}-(y_{n-1}- y_n) \rightarrow$ACC. $z_n \rightarrow$S(0) (Update total) Base= 0

STEP	LABEL	OP1	OP2	OPERAND	COMMENTS
9		AVZ	INCB	UV(-,0)	$x_n \rightarrow$ ACC. (From Base 0) Base incremented. Starts LPF (12 KHz).
10		NOP	NOOP	UV(6,2)	$w_{n-1}; -b_1 \rightarrow$ MULT. Base = 1
11		APA	NOOP	UV(7,3)	$x_n - b_1 . w_{n-1} \rightarrow$ ACC. $w_{n-2}; -b_2 \rightarrow$ MULT.
12		APA	TACV	UV(5,2)	$w_n = x_n - b_1 . w_{n-1} - b_2 . w_{n-2}$ $w_n \rightarrow$ ACC. $\rightarrow$ V(2). Replaces w_{n-1}. $w_{n-1} \rightarrow$ VP. w_{n-1}; $a_1 \rightarrow$ MULT.
13		APA	TVIB	UV(4,3)	$w_n + a_1 . w_{n-1} \rightarrow$ ACC. $w_{n-1} \rightarrow$ V(3) Replaces w_{n-2} 1; $w_{n-2} \rightarrow$ MULT. Base incremented.
14		APA	NOOP	UV(6,2)	$x_n' = y_n = w_n + a_1 . w_{n-1} + w_{n-2}$ $y_n \rightarrow$ ACC. Base = 2. $w'_{n-1}; -d_1 \rightarrow$ MULT.
15		APA	NOOP	UV(7,3)	$x'_n - d_1 . w'_{n-1} \rightarrow$ ACC. $w'_{n-2}; -d_2 \rightarrow$ MULT.
16		APA	TACV	UV(5,2)	$w'_n = x'_n - d_1 . w'_{n-1} - d_2 . w'_{n-2}$ $w'_n \rightarrow$ ACC. $\rightarrow$ V(2). Replaces w'_{n-1}. $w'_{n-1} \rightarrow$ VP

STEP	LABEL	OP1	OP2	OPERAND	COMMENTS
					w'_{n-1}; $c_1 \rightarrow$MULT.
17		APZ	TVPV	UV(4,3)	$c_1 . w'_{n-1} \rightarrow$ACC.
					$w'_{n-1} \rightarrow$V(3) Replaces w'_{n-2}
					w'_{n-2}; $c_0 \rightarrow$MULT.
18		APA	INCB	UV(4,2)	$c_1 . w'_{n-1} + c_0 . w'_{n-2} \rightarrow$ACC.
					w'_n; $c_0 \rightarrow$MULT.
					Base incremented.
19		APA	TACV	UV(-,0)	$y'_n = c_0 . w'_n + c_1 . w'_{n-1} + c_0 . w'_{n-2}$
					$y'_n \rightarrow$ ACC. $\rightarrow$V(0). Base = 3
20	NEXT	NOP	JMIF	DT(NEXT)	Waits for next Base No.
21		NOP	LIBL	---	Base No. of next BPF to Base Reg. Clears I.F.
22	SAME	NOP	JMIF	DT(SAME)	Waits for G_n^k coeff. input. Sets $\overline{\text{IRQ}}$. Base = B
23		SRI	MODE	UV(3,4)	Second BPF starts.
					a_2; $w_{n-2} \rightarrow$MULT
					Serial Input Mode set.
24		APZ	TIRV	UV(0,1)	$w_{n-2} . a_2 \rightarrow$ACC.
					$G_n^k \rightarrow$V(1). Clears I.F.
					x_n^k; $G_{n-1}^k \rightarrow$MULT.

STEP	LABEL	OP1	OP2	OPERAND	COMMENTS
25		APA	TACV	UV(-,2)	$w_n = x_n^k . G_{n-1}^k - a_2 . w_{n-2} \rightarrow$ ACC ACC(w_n) $\rightarrow$ V(2); $w_{n-1} \rightarrow$ VP
26		SVA	TVIB	UV(-,3)	$w_{n-2} - w_n = -y_n \longrightarrow$ ACC $w_{n-1} \rightarrow$ V(3). Base incr.
27		NEG	TACV	UV(0,1)	$-y_n \rightarrow y_n \rightarrow$ V(1). Base = B+1 G_{n-1}^{k+1}; $y_{n-1} \rightarrow$ MULT.
28		APZ	NOOP	UV(0,1)	$G_{n-1}^{k+1} . y_{n-1} \rightarrow$ ACC. G_{n-1}^{k+1}; $y_n \rightarrow$ MULT.
29		SPA	DECB	---	$y_n . G_{n-1}^{k+1} - y_{n-1} . G_{n-1}^{k+1} \rightarrow$ ACC Base decr.
30		AVA	TACV	US(-,0)	$z_n = z_{n-1} + y_n G_{n-1}^{k+1} - y_{n-1} G_{n-1}^{k+1}$ $z_n \longrightarrow$ ACC $\longrightarrow$ S(0). Base = B
31		NOP	LACO	UV(1,6)	$z_n \rightarrow$ OR G_n^k; K $\rightarrow$ MULT.
32		APZ	TACV	UV(-,1)	$G_n^k . K \rightarrow$ ACC. $\rightarrow$ V(1)
33		AVZ	INCB	UV(-,1)	$x_n^k \rightarrow$ ACC, Base incr.
34		SRO	MODE	UV(6,2)	w_{n-1}; $b_1 \rightarrow$ MULT. Base = B+1. SRO Mode set
35		APA	NOOP	UV(7,3)	$x_n^k - b_1 . w_{n-1} \longrightarrow$ ACC. w_{n-2}; $-b_2 \rightarrow$ MULT.

STEP	LABEL	OP1	OP2	OPERAND	COMMENTS
36		APA	TACV	UV(5,2)	$w_n = x_n^k - b_1 \cdot w_{n-1} - b_2 \cdot w_{n-2}$ $w_n \rightarrow$ ACC. $\rightarrow$ V(2). Replaces w_{n-1}. $w_{n-1} \rightarrow$ VP. $w_{n-1}; a_1 \rightarrow$ MULT.
37		APA	TVIB	UV(4,3)	$w_n + a_1 \cdot w_{n-1} \rightarrow$ ACC. $w_{n-1} \rightarrow$ V(3) 1; $w_{n-2} \rightarrow$ MULT. Base incremented.
38		APA	NOOP	UV(6,2)	$x_n' = y_n = w_n + a_1 \cdot w_{n-1} + w_{n-2}$ $y_n' \rightarrow$ ACC. Base = B + 2. $w_{n-1}'; -d_1 \rightarrow$ MULT.
39		APA	NOOP	UV(7,3)	$x_n' - d_1 \cdot w_{n-1}' \rightarrow$ ACC. $w_{n-2}'; -d_2 \rightarrow$ MULT.
40		APA	TACV	UV(5,2)	$w_n' = x_n' - d_1 \cdot w_{n-1}' - d_2 \cdot w_{n-2}'$ $w_n' \rightarrow$ ACC. $\rightarrow$ V(2) $w_{n-1}' \rightarrow$ VP. $w_{n-1}'; c_1 \rightarrow$ MULT.
41		APZ	TVPV	UV(4,3)	$c_1 \cdot w_{n-1}' \rightarrow$ ACC. $w_{n-1}' \rightarrow$ V(3) w_{n-2}'; $c_0 \rightarrow$ MULT.
42		APA	INCB	UV(4,2)	$c_1 \cdot w_{n-1}' + c_0 \cdot w_{n-2}' \rightarrow$ ACC. w_n'; $c_0 \rightarrow$ MULT. Base incremented.

STEP	LABEL	OP1	OP2	OPERAND	COMMENTS
43		APA	TACV	UV(-,0)	$y'_n = c_0 \cdot w'_n + c_1 \cdot w'_{n-1} + c_0 \cdot w'_{n-2}$ $y'_n \rightarrow$ ACC $\rightarrow$ V(0). Base = B + 3
44		AVZ	DECB	UV(-,1)	$G_{n-1}^{k+1} \rightarrow$ ACC. Base decr.
45		NOP	DECB	---	Base = B+2. Base decr.
46		NOP	TACV	UV(-,0)	$G_{n-1}^{k+1} \rightarrow$ V(0). Base = B + 1
47	WAIT	NOP	JMIF	DT(WAIT)	Waits for next x_n sample from SRI. $\overline{IRQ}$ not set
48		NOP	TIRV	D(00.0)	x_n (next) to Base 00, displacement 0.
49		NOP	NOOP	---	Dummy instruction to be repeated until $\overline{RST}$ pulse restarts the program

We will now explain briefly the program events refering to the same step numbers in the program list. Besides the program, refer also to the memory mapping of Figure 3.2 and the difference equations (2.12), (2.13), (2.19) and (2.20).

1. The Base No. has been set to 0 at $\overline{RST}$, pointing to the 16 KHz BPF that is always the first BPF to be processed. The sample from the serial input buffer has been placed in location (00.0) at the end of the previous program run. The DUH MODE is set so that only the upper half byte (8bits) of a 16 bits word is inputted through the parallel bus . Also the multiplier (MULT) input is set up with the address UV(3,4) that calls the values of w_{n-2} and a_2 (-1/3) thus starting the processing of the 16 KHz BPF.
2. The instruction JMIF sets a waiting loop to input the gain coefficient G_n for the 16 KHz BPF. Since the Input Flag (IF) has not been set, the instruction is executed. This sets LOW the $\overline{IRQ}$ what in turn pulses $\overline{IE}$. The effect of these signals is to switch the parallel bus to input G_n at the rise of $\overline{IE}$, what sets the IF and clears $\overline{IRQ}$. G_n is then transferred to the Input Register (IR).
3. The product of step 1 remained in the MULT during step 2 because address mode DT was used. With APZ that product is added to zero and placed in the accumulator (ACC). This is the same as loading the ACC

with the product. TIRV transfers G_n to location V(1) and the IF is cleared. The MULT, however, is set up before with the current values of UV(0,1), which are the sample x_n and the previous value G_{n-1} of G_n. This should cause no problem since the change of G_n, when it occurs, is much slower than the program speed.

4. APA adds the product to the contents of the ACC and places the sum back in the ACC thus producing w_n. w_n is tranferred to V(2) to replace the previous value w_{n-1} and this, in turn, is held in the VP register
5. SVA substracts V(3), that is w_{n-2} and the ACC content to give $-y_n$. TVIB transfers w_{n-1} that was in the VP register to V(3) to replace the previous w_{n-1} and at the same time increments the Base No. at the end of the instruction to place y_n in Base 1 in the next instruction.Now the equation (2.13) is completed, except for the sign, and the intermediate values have been updated.
6. The new sample output y_n must be saved for the next run of this BPF. The previous output sample must be brought to the ACC to be substracted from y_n and the difference must be added to the total accumulated output of all the filters, z_n, to update it. Then, we change sign of y_n (NEG) and move it to V(1) of Base 1 with TACV. At the same time y_{n-1} is picked up by placing it with a 1 at the input of the MULT.
7. SPA substracts the product and y_n, while the Base No.

is decremented to prepare for the initiation of the LPF processing that requires to access again x_n.

8. The previous total output, z_{n-1}, is kept in location 0 of the scratch pad,i.e. S(0). SVA substracts this value and the accumulator what amounts to add the new y_n and substract the previous one y_{n-1}, as desired. The result is placed back in S(0).
9. Here the sample x_n is moved to the ACC with AVZ and the Base is incremented again to access the first section of the LPF.
10. Refering to equations (2.19) the MULT is set up for the second term of w_n.
11. The product is added to x_n in the ACC with APA. The MULT is set up for the third term of w_n.
12. APA adds the product to the ACC thus completing the expression for w_n. TACV places w_n in V(2) to replace w_{n-1} which is, in turn, transferred to the VP register. The ACC now holds w_n. The MULT is set up for the second term of y_n in equations (2.19).
13. APA adds the product to w_n. w_{n-1} (from VP) is transferred to V(3) with TVIB to replace w_{n-2}. w_{n-2} is placed at the input of the MULT with a 1. The Base No. is incremented at the end of the instruction to access the second section of the LPF.
14. APA adds the product to the ACC. Now the ACC holds y_n, the output of the first section which is the input x'_n to the second section. The Base is now 2.

Referring to equations (2.20) the steps are now repeated. The MULT is set up for the second term of w'_n.

15. APA adds the product to x'_n and the MULT is set up for the third term of w'_n.
16. APA adds the product to the ACC completing w'_n. w'_n is then transferred to V(2) to replace w'_{n-1} which is, in turn, moved to VP. To save one step, we will now set up the MULT first for the second term of y'_n.
17. APZ places the product in the ACC. w'_n is erased but it has been saved in V(2). With TVPV, w'_{n-1} is transferred from VP to V(3) to replace w'_{n-2}. The MULT is set up for the third term of y'_n.
18. APA adds the product to the ACC. The MULT is set up for the first term of y'_n, picking up from V(2) the w'_n just obtained in step 16. The Base No. is incremented once more to access the next BPF base where the output of the LPF, y'_n, will be placed as the "sample" for the lower octave BPF and LPF.
19. APA adds the product to the ACC thus completing y'_n and TACV places it in V(0) of Base 3.
20. This is a waiting loop, such as in step 2, to input the Base No. of the next BPF. This Base No. will be placed in bus in the first loop (see Figure 4.2). The Base No. is latched in the IR and the IF is set.
21. LIBL transfers the contents of the IR to the Base register and the IF is cleared. Base No. is now some number B according to the sequence in Base No. ROM.

22. This is another waiting loop to input the gain coefficient G_n^k for the kth BPF to be processed.
23. The SRI (serial input) MODE is set. This enables the Serial Input port to accept the sample from the ADC. The MULT is set up for the values of w_{n-2} and a_2 thus starting the processing of a second BPF, as it was done for the first BPF in step 1.
24. Same as step 3 for the first BPF . Now x_n^k represents the "sample" for the kth BPF. G_n^k is loaded to V(1).
25. Same as step 4.
26. Same as step 5.
27. Same as step 6 but the MULT is not set up for y_{n-1} and 1 as in step 6, but for y_{n-1} and G_{n-1}^{k+1} which is the gain coefficient of the next lower octave BPF. This coefficient is brought up to this base in instructions 44-46. This instruction and the two following implement the correction of the response mentioned under Gain Coefficients.
28. APZ places the product in the ACC and the MULT is set up to make the same multiplication as in step 27, this time with the current BPF output y_n.
29. The two products are substracted with SPA and the Base is decremented to access again the sample x_n^k for the processing of the LPF.
30. Here AVA adds the previous total output z_{n-1} to the difference between the current and the previous output samples to give z_n. z_n, updated once more and fi-

nal for this program run, is placed back in S(0).

31. The updated z_n is also placed in the Output Register (OR) to be transferred to the Serial Output port. The MULT is set up to multiply G_n^k by K=1/1.823, what is done in all the BPF except the 16 KHz, as explained before.

32. APZ places the product in the ACC and TACV transfers it back to V(1).

33. AVZ loads the ACC with the sample x_n^k and increments the Base to B+1 to access the LPF base.

34. SRO (Serial Output) MODE is set to allow the output of z_n. The MULT is set up with the second term of w_n in equation (2.19) thus starting the processing of the 2nd LPF in the program.

35-43. These instructions are identical to the 11-19 involving the processing of the LPF except for the difference in Base Nos. At the end, the output y_n' of the LPF is placed in V(0) of the following Base B+3. This will be the "sample" for the lower octave BPF and LPF.

44,45. The Base No. is now that of the (k+1)th BPF. AVZ loads the ACC with the gain coefficient of that BPF and we will now move it two bases down. Then, Base is decremented twice.

46. Now TACV places $G_{n-\frac{1}{2}}^{k+1}$ in V(0) for the step 27.

47. This is a waiting loop for the input of the next sample from the Serial Input buffer. $\overline{IRQ}$ is not set be-

cause we are in the serial mode. This instruction is normally not executed because SIEN fall has occurred before and the IF is set.

48. TIRV transfer the sample to Base 0 in preparation for the next program run.
49. This instruction can be repeated until the $\overline{\text{RST}}$ pulse restarts the program.

CHAPTER 4

HARDWARE DESIGN

1. GENERAL ARCHITECTURE

Figure 4.1 shows a block diagram of the complete graphic equalizer. The audio signal input is routed to an anti-aliasing lowpass filter LPF1. This is a 4-pole, elliptic type filter with a cut-off frequency of about 21 Khz, a ripple of 0.1 dB and a unity gain. The filter delivers its output to a Track and Hold amplifier (T/H) Analog Devices type AD583 where the analog audio signal is sampled at a rate of 64K samples per second. The sampling rate is determined by a signal generated by the control logic, which is used as Start of Conversion (SOC) as well as Reset ($\overline{RST}$) pulse to synchronize all the hardware sections to the beginning of the program (see Program Events Timing of Figure 4.2).

The sampled audio signal is converted to digital form in the 12-bit A/D Converter, Data Device Corp. type ADH-8585 and the resultant digitized signal is serially clocked into the Serial Input (SRI) of the SPP processor AMI S2811. All the A/D conversion operations are initiated by the SOC (same as $\overline{RST}$) pulse but thereafter the conversion is entirely controlled by the ADC module itself, which

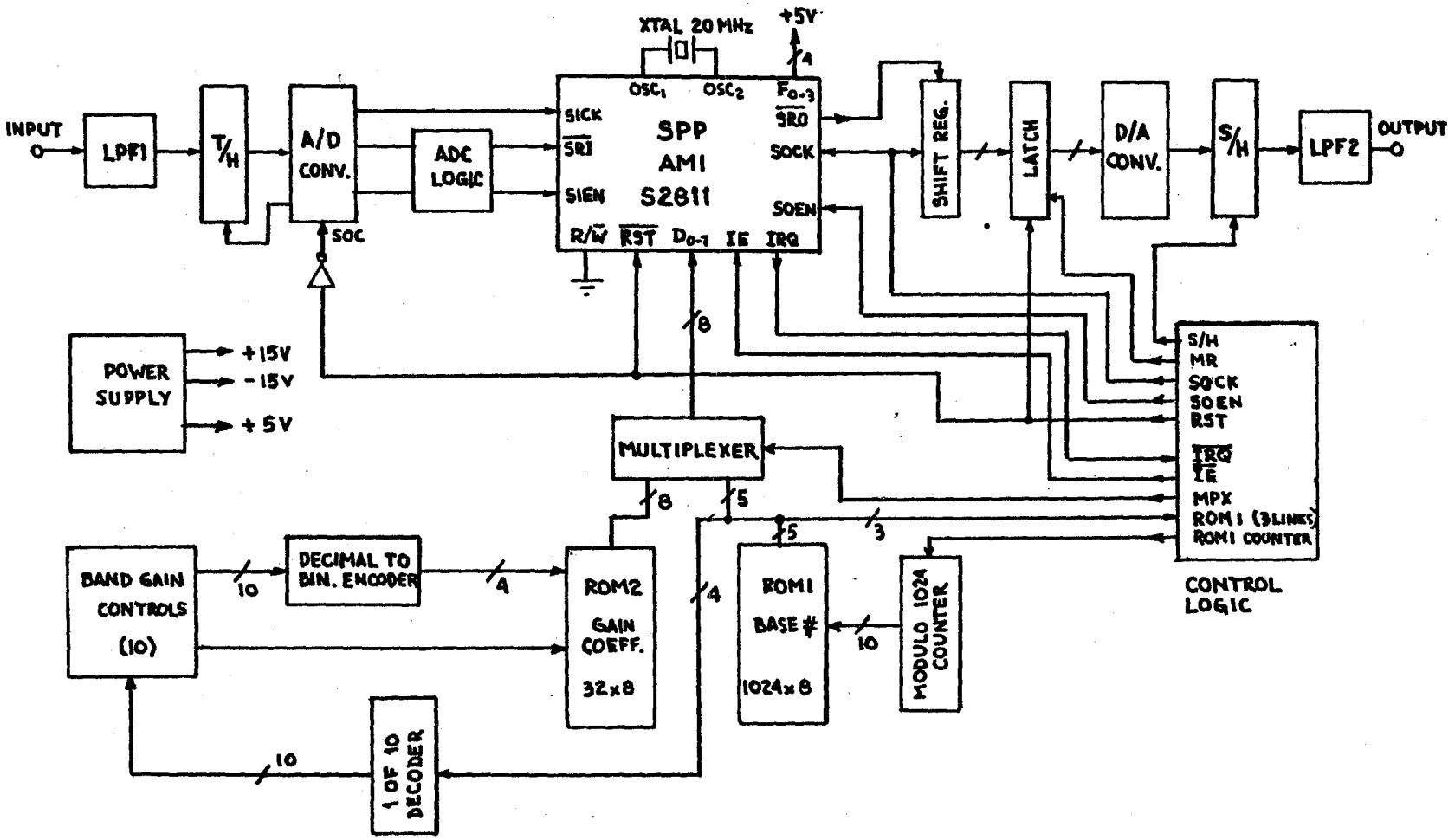


Figure 4.1 Block diagram of the complete equalizer (one channel only).

generates the Clock (SICK), the Track and Hold control signal (T/H) and the Serial Input Enable (SIEN) pulses. Some control logic is interposed between the ADC module and the SPP processor in order to place the input data in the proper format.

The SPP processor, of course, is the heart of the system since, as explained elsewhere, all the digital filtering operations and output totalizing, that is, the summing of the outputs of all the filters, takes place there. The SPP's serial input and output ports handle the input and output of the samples simultaneously and independently of the program, albeit in synchronism with it through the $\overline{RST}$ pulse. The clock rate of the SPP is set by a 20 MHz crystal.

Once the input has been processed in the SPP, a new output sample (updated sum of the outputs of all the filters) appears at the Serial Output port (SRO) of the SPP. At a time determined by the control logic the Serial Output is enabled with the Serial Output Enable (SOEN) pulse. The Serial Output Clock (SOCK), which is the same as Master Clock (4.096 MHz), will then clock the output into a Shift Register. This is required to convert the serial data into parallel form in order to be accepted by D/A converter. A 12-bit latch is used to hold the output sample for further processing. Note that the latch signal is the same $\overline{RST}$ pulse so that the D/A converter actually processes the previous sample.

The D/A converter, a Burr Brown type DAC-800, in com-

bination with the Sample and Hold (S/H) amplifier, another Analog Devices type AD583, restore the digitized output sample to the analog form which is then smoothed out by the lowpass filter LPF2, a 4-pole, elliptic type filter of characteristics similar to the LPF1 filter.

As explained before, the (RAM and ROM) of the SPP processor is divided in 32 bases of 8 bytes each, with the coefficients and the intermediate results of the 10 bandpass filters occupying 10 of those bases, respectively. As mentioned in Chapter 1, the processing of the bandpass filters must be interleaved in a given sequence, in a sort of time division multiplexing fashion. The proper sequence of base numbers is stored in a 1024 x 8 ROM, the ROM1. Actually, only 5-bit words are required to address the 32 bases but an 8-bit memory is used for convenience and standardization. This ROM1 is, in turn, addressed by a modulo 1024 counter which is advanced twice per sample, so as to address the two bases corresponding to that sample.

To set the gain of each of the 10 filters there are 10 front panel gain controls accessible to the user. Note that for a stereo arrangement, all the hardware should be duplicated and two sets of gain controls should be provided. The gain controls are slide switches that can be set for the nominal gains of -10 dB to +10 dB in steps of 1 dB. The corresponding 21 coefficients for each of the 21 gain settings, including 0, are stored in a 32 x 8 ROM (ROM2). The various settings of the gain switches are encoded with a decimal to

binary encoder and some additional logic to a 5-bit word that addresses the ROM2 to the proper coefficient.

At the time of processing a given BPF, its base number appears at the output of ROM1. This number may then be used to enable the gain control corresponding to that filter. This is done by connecting in parallel all the gain control switches and returning their wipers to the output of a 1 of 10 decoder which is, in turn, addressed by the output of ROM1. Thus, for example, if the 4 KHz BPF is being processed the Base No. 6 (see Figure 3.2) will be present at the output of ROM1 and at the input of the decoder. One output of the decoder will then go LOW and since that output is connected to the 4 KHz gain control, that control only will be enabled. The setting of this control will determine what coefficient in ROM2 will be outputted to the processor.

To input the filter base and gain coefficient data to the SPP processor an 8-bit parallel bus (D_0-D_7) is used. In order to input this data at the proper time in the program, a Multiplexer switches the parallel bus between the outputs of ROM1 and ROM2.

The switching command for the Multiplexer and the Enable pulses for the parallel bus ($\overline{IE}$) are originated in the Control Logic section, using a software generated Interrupt Request ($\overline{IRQ}$) pulse to keep proper synchronism with the program.

Other control signals are also produced in the Control Logic section. Here a 4.096 MHz Master Clock and a counter

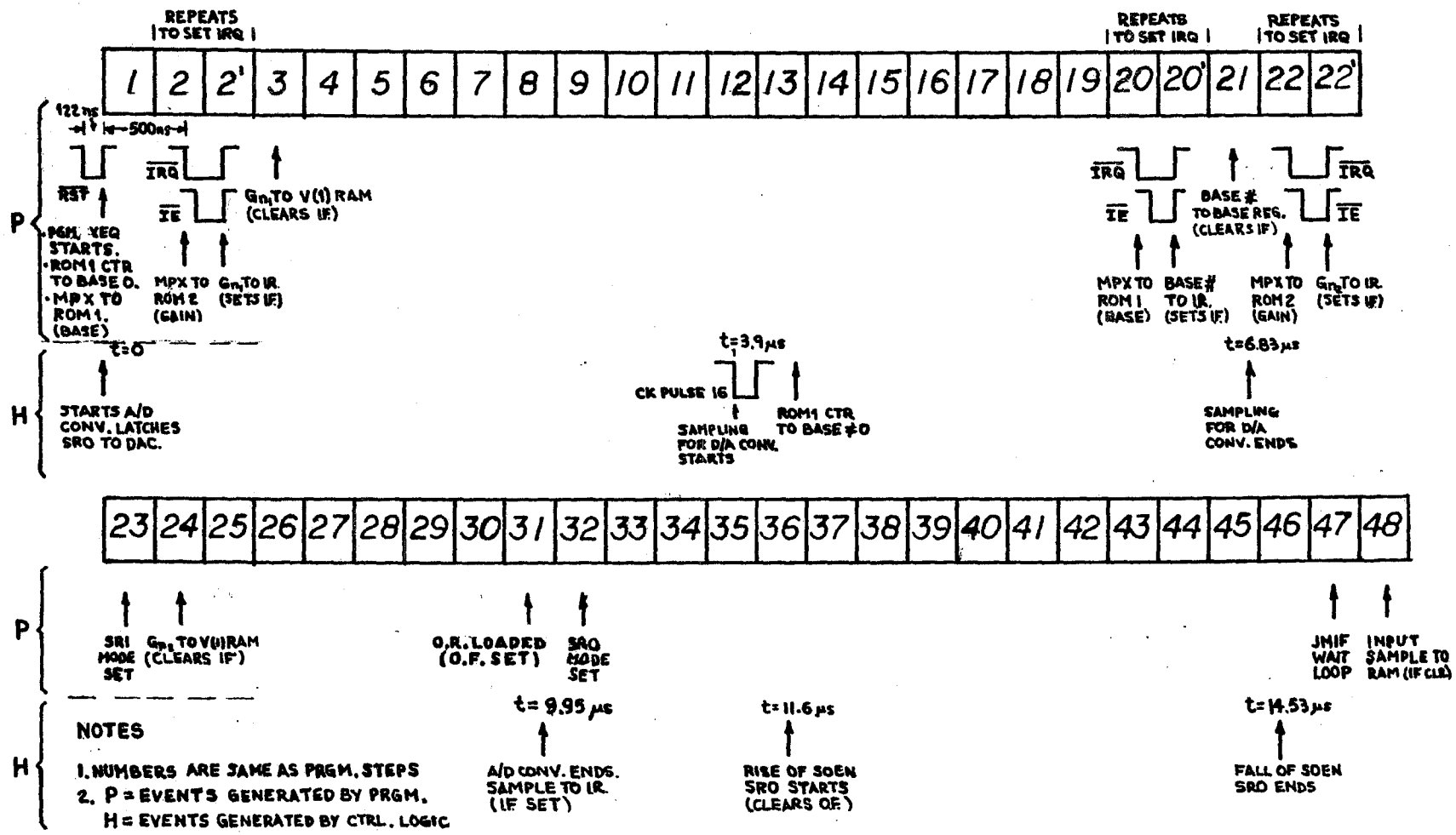


Figure 4.2 Program Events Timing

divider are combined to generate the necessary control pulses that activate the various sections of the equalizer in the appropriate timing relationship. A power up Master Reset ($\overline{MR}$) pulse is generated at turn on time to place all the sections in their initial status.

The block diagram also shows the required power supply that delivers the +5V DC and the ±15V DC voltages necessary to power the various sections of the circuit. The power supply is assumed to be acquired as a separate module according with the specifications given below and is not made a part of this project.

It must be noted that the basic block diagram just described can process only one audio channel. For a stereo equalizer, all this hardware must be duplicated.

2. ANALOG TO DIGITAL CONVERTER

The Analog to Digital converter section (see Figure 4.3) comprises a lowpass filter LPF1, a Track and Hold amplifier, the actual A/D Converter module and the interface logic.

For the A/D Converter module we chose a resolution of 12 bits. This resolution was considered a good compromise between cost and performance. Although the SPP processor can handle 16-bit serial input and output data, A/D converters with this resolution and the required speed are extremly expensive. We chose a Data Devices Corp. type ADH-8585 (see Appendix for specifications). This is a hybrid type with

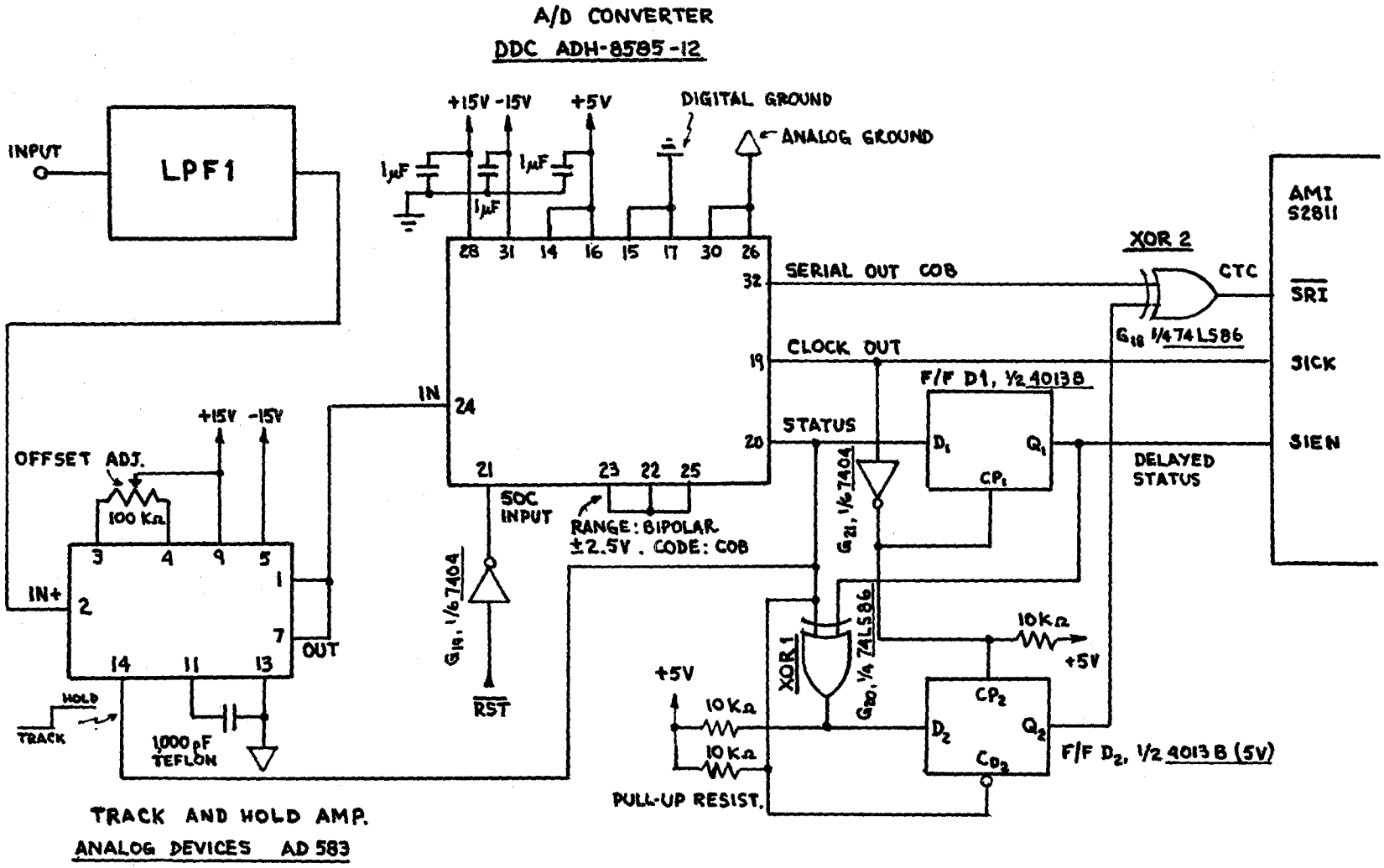


Figure 4.3 Schematic of the A/D Converter

a self-contained clock and reference voltage, offering adjustable input voltage ranges and serial output. The conversion time is 10 usec, more than sufficient for our sampling period of 15.6 usec and the linearity error is 0.012 % of FS, what in our case translates into -72 dB since we are using bipolar ranges. These characteristics make this ADC quite suitable for our application and eliminates the need for additional hardware.The ADC is connected for $\pm$2.5V bipolar range to allow enough headroom with typical inputs of 0.5 to 1 V P-P.

In order to start the conversion in synchronism with the program, the $\overline{\text{RST}}$ pulse from the Control Logic section is used as SOC pulse (refer to timing diagram of Figure 4.4).

The clock output of the ADC (about 1.3 MHz) is used as SICK and the STATUS is used as SIEN for the serial input of the SPP. The timing specifications of the SPP require that the rising and falling edges of the SIEN pulse follow the falling edge of the clock SICK, but the timing of the ADC is such that the edge of the STATUS signal follows the rising edge of the ADC clock, as shown in Figure 4.4. One possible solution is to delay the STATUS signal one half clock cycle.

As depicted in the timing diagram of Figure 4.4, the ADC produces 13 clock pulses in each conversion period. The first clock cycle tries the Bit 1 but it does not produce valid data; the Bit 1 is valid in the second cycle of the clock. The serial input register of the SPP, on the other

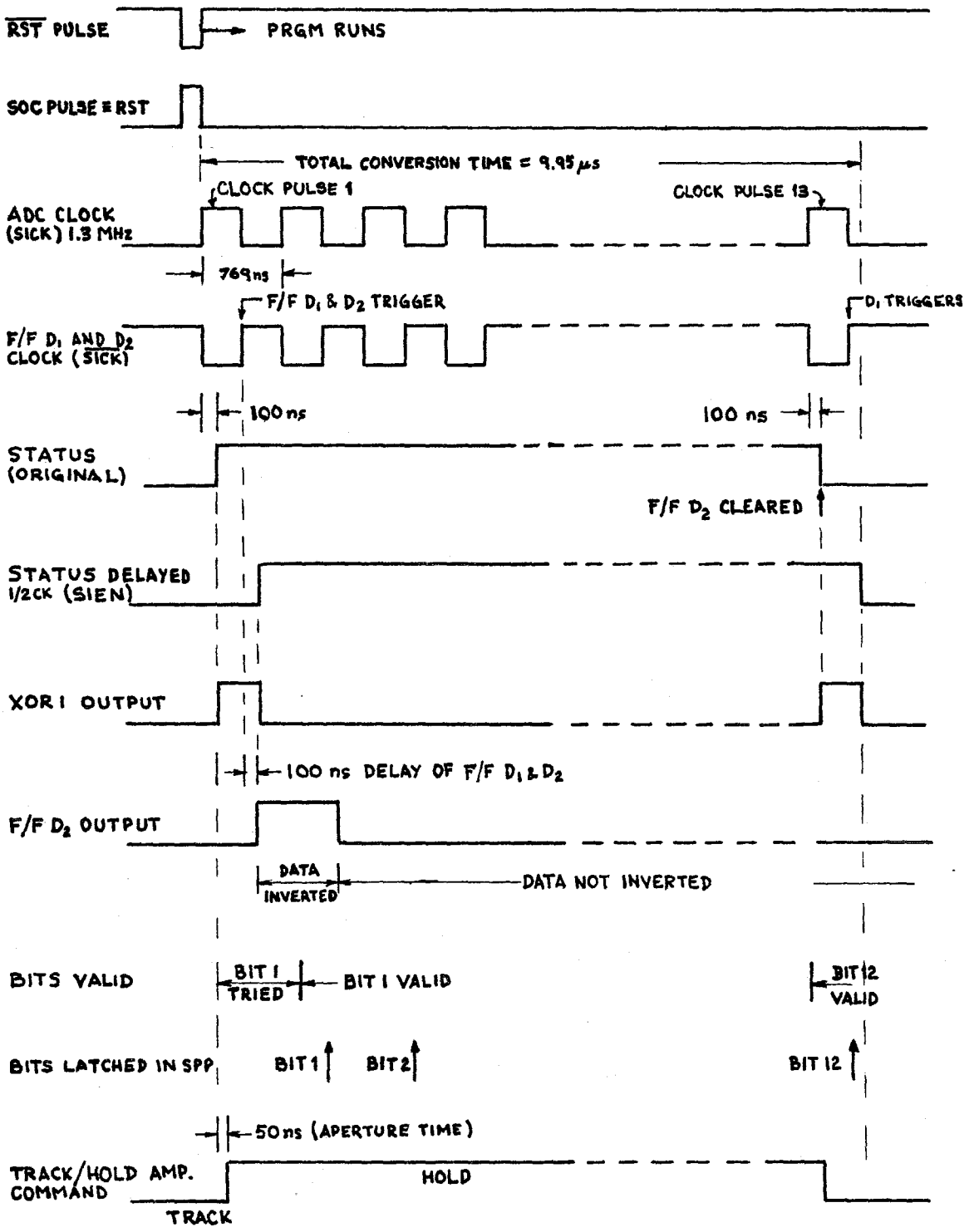


Figure 4.4 Timing diagram for the A/D conversion.

hand, latches each bit on the falling edge of the clock SICK. If the Serial Input were allowed to be enabled during the first clock cycle, invalid data would be fed into the Serial buffer register. By delaying STATUS one half clock cycle, instead, before applying it to the SIEN input not only the timing requirements of the SPP processor are satisfied, but the serial input is not enabled during the first clock cycle (first falling edge of the clock). Thus, the latching of the first, invalid bit is avoided.

The schematics of Figure 4.3 shows how the STATUS signal is delayed by means of an edge triggered D flip-flop D1. This F/F is triggered by the rising edge of its clock input so that, by clocking it with the inverted ADC clock, the first triggering occurs one half clock cycle after SOC (see timing diagram of Figure 4.4). At the time of D1 triggering, the STATUS signal at its input is already HIGH and the respective '1' is transferred to the Q output of D1, which is connected to the SIEN input of the SPP. Since the STATUS signal remains HIGH during the conversion, so does the SIEN pulse. The last triggering of D1 occurs after the clock 13 at which time the Bit 12 is already valid and it is latched in the SRI. The 100 to 200 nsec delay of D1 allows enough time for the latching of Bit 12.

Another interface problem originates from the fact that the SPP processor accepts data only in the form of complimentary sign and magnitude or else complimentary two's complement (CTC), while the ADC serial output in the bipolar

mode only supplies complimentary offset binary (COB) data. However, the COB code can be easily turned into CTC by reversing the first (MSB) bit. To accomplish this result we observe from the timing diagram of Figure 4.4 that the STATUS and SIEN (STATUS delayed) signals have different value during the first clock cycle. Using this fact and applying both signals to an exclusive OR gate (XOR1) a pulse is generated during the first clock cycle. If this pulse is delayed through a second D F/F D2, triggered with the same inverted clock used for D1, we see that a pulse is generated only during the time that Bit 1 is valid, since previous and subsequent D2 triggerings will find the STATUS and SIEN signals in the same state and therefore will generate '0' output from the F/F D2.

The pulse thus obtained from F/F D2 while Bit 1 is valid, is applied to a second exclusive OR gate (XOR2) together with the serial data, as illustrated in Figure 4.3. The first data bit (MSB) finds the other side of XOR2 at logic '1' and it is therefore reversed. For the rest of the data bits stream, however, the second side of XOR2 is at logic '0' and no inversion takes place. Thus, the COB is converted into a CTC code as desired. At the end of the conversion cycle, while the Bit 12 is valid, another pulse is generated at the output of XOR1 since STATUS and SIEN have once again different value at that point. To avoid inverting the Bit 12 we use the STATUS signal which falls LOW before the Bit 12 is latched, to reset the F/F D2 through the Clear input C_D.

Therefore, no output is produced from F/F D2 during the time the Bit 12 is valid. Note that the STATUS signal goes HIGH at the start of the conversion before the F/F D2 is triggered, thus enabling that F/F on time for the reversal of the first bit process just explained.

To sample the audio analog input signal we use a Track and Hold amplifier Analog Devices AD583. With the chosen 1,000 pF holding capacitor the acquisition time for 0.1 % of the output is about 4usec. Since the A/D conversion time is less than 10 usec, the total time is less than the 15.6 usec sampling period. The capacitor is a Teflon type for minimum leakage, giving a droop of 11 uV/msec. The aperture time of 50 nsec is sufficiently short to yield a steady voltage at the time the conversion of the first bit starts. Therefore, the STATUS signal can be conveniently used as the Track/Hold (T/H) command.

The process of A/D conversion is initiated, as mentioned before, by the SOC pulse which is just the $\overline{\text{RST}}$ pulse inverted. After 150 nsec the STATUS goes HIGH and some 50 nsec later the T/H amplifier is holding the analog input within 0.1 % of its final value. The Bit 1 is tried at this time and becomes valid one clock (769 nsec) later. By then SIEN is HIGH and Bit 1 is latched at the falling edge of clock pulse No. 2; this particular Bit had been inverted by the XOR2 gate. The delay of the XOR gates is neglegible. The conversion continues until Bit 12 is latched and about 100 nsec later (delay of F/F D1) SIEN falls and the contents of

the serial buffer register in the SPP processor is transferred to the Input Register (IR) setting the Input Flag (IF) in the SPP processor. The entire conversion process takes slightly less than 10 usec. The T/H has nowmore than 5 usec to settle in tracking the input signal (refer to Program Events Timing of Figure 4.2 for the foregoing sequence of events). The sample is held in the IR until is transferred to the RAM portion of the memory, at Base location 00 at the end of the program.

The lowpass filter LPF1 is constructed with conventional operational amplifiers according to the circuit of Figure 2.8. We have chosen quad op amps National Semiconductors LF347 for the reasons explained in Chapter 2.

The ADC, the T/H amplifier and the SPP are all TTL compatible. For the interface we have used, however, 4013B, CMOS D flip-flops operated at 5V to provide some extra delays and insure smooth operation. The attendant delays have been indicated in the timing diagram of Figure 4.4. The XOR gates are 74LS86 to suit the driving capabilities of the CMOS outputs. Pull-up resistors were added where necessary to guarantee the '1' state input voltage levels demanded by the CMOS devices.

3. DIGITAL TO ANALOG CONVERTER

The Digital to Analog Converter section comprises a serial to parallel shift register, a 12-bit latch, the D/A converter proper, a Sample and Hold amplifier and a lowpass filter LPF2. The schematics is shown in Figure 4.5.

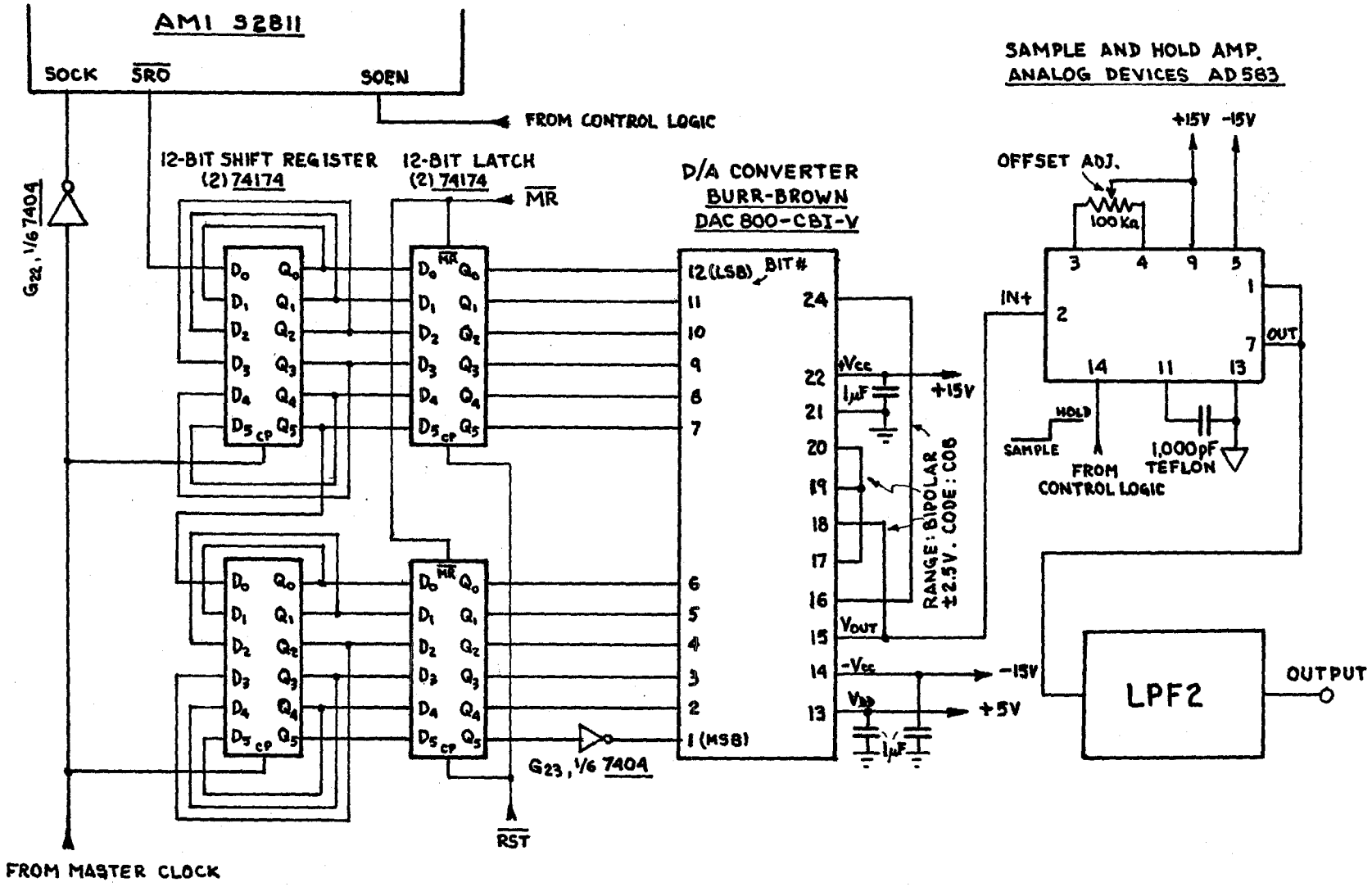


Figure 4.5 Schematic of the D/A Converter

The choice of the D/A converter module (DAC) was based on the requirement of short settling time,,good linearity and low cost. We selected the Burr Brown type DAC-800 (see specifications in the Appendix). It offers settling time of less than 2.5 usec for 0.1 % of FS range, good linearity of 1/2 LSB maximum error and it has a built-in reference voltage and output amplifier. For compatibility with the A/D converter, the DAC is wired for a $\pm$2.5V bipolar range and consequently, accepts a complimentary offset binary code.

Since this DAC accepts only parallel inputs and the outputs from the SPP processor is in serial form, a shift register and latch is used to output the data from the SPP and deliver it to the DAC. For the required 12-bit shift register we found convenient to use two hex D F/F type 74174 and interconnect the individuals F/F in series in order to form the desired register.

To hold the output sample until the next sampling period, a 12-bit latch is connected between the shift register and the DAC. For the latch we use again two 74174's, this time connected as a parallel-in, parallel-out register.

The serial output (SRO) port of the SPP is similar to the serial input port. As seen in the schematics of Figure 4.5 and in the timing diagram of Figure 4.6, the Serial Output Clock (SOCK) that shifts the data out of the SPP is just the 4.096 MHz Master Clock inverted. The Serial Output Enable (SOEN) pulse is formed in the Control Logic section (refer to the Control Logic section description). This pulse

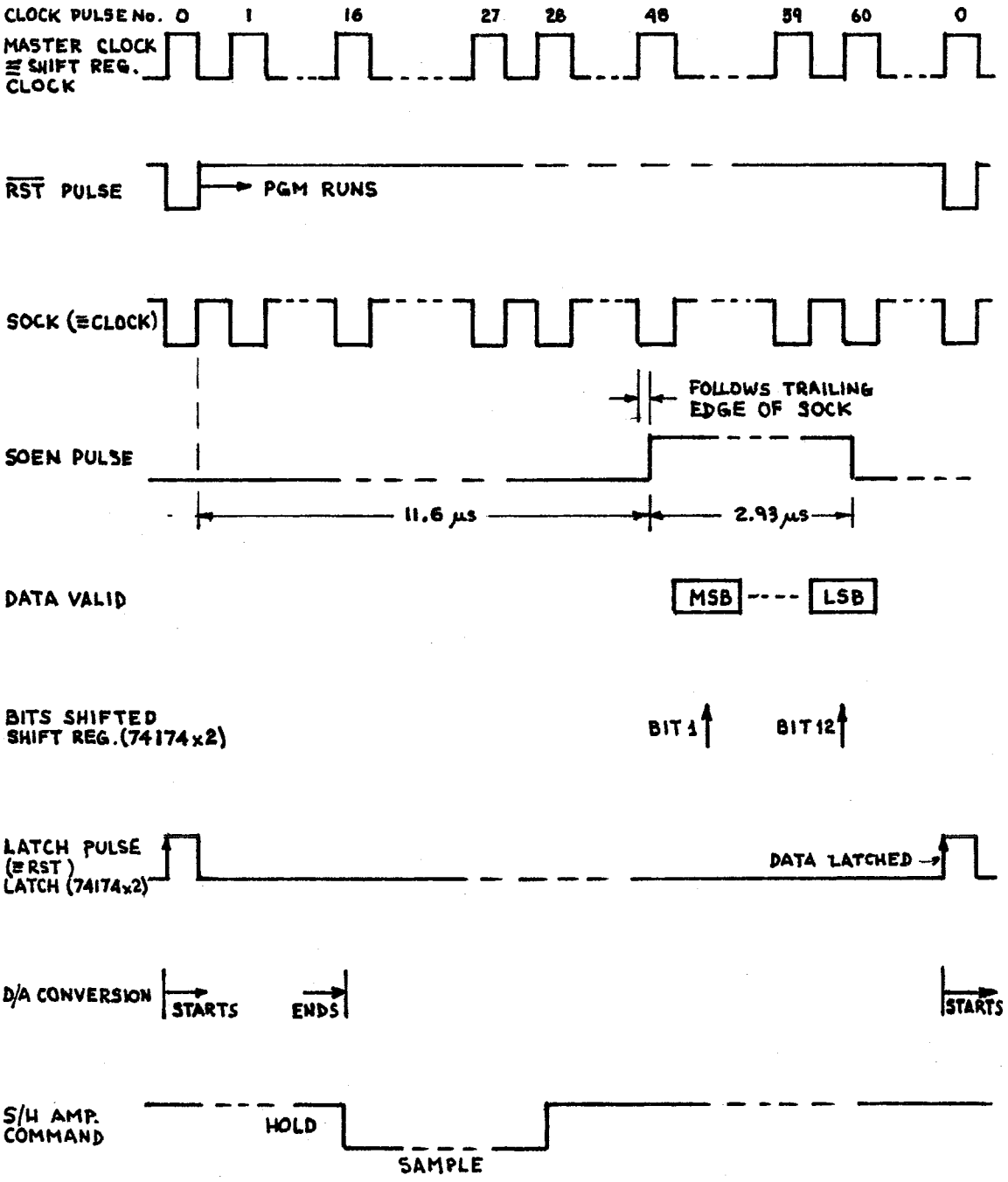


Figure 4.6 Timing diagram of the D/A conversion.

starts 11.6 usec after the program run starts with the $\overline{RST}$ pulse. This allows sufficient time for the processing of one sample before the resultant output is ready for the D/A conversion. The shifting of the 12 bits from the SRO output is completed in the 12 clock cycles that SOEN is HIGH, that is in 2.93 usec. The transfer of data from the Serial Output is therefore completed within 14.53 usec from the start of the program and before the end of a sampling period of 15.6 usec (refer to the chart of Figure 4.2).

The Master Clock is inverted before being used as SOCK in order to satisfy the SPP requirement that the rising edge of SOEN follows the falling edge of the clock, as shown in Figure 4.6. Since the data is valid between rising edges of SOCK and the Shift Register is clocked at the rising edges of the uninverted clock (or falling edges of SOCK), the data is stable at the time of shifting and no set-up or hold time problems may develop.

The data at the input of the latch is latched at the rise of $\overline{RST}$. After that, stable data is available for the D/A conversion which actually takes place during the next run of the program. An inverter is interposed in the line of the MSB between the latch and the DAC because the output code of the SPP is CTC and the DAC is wired for COB code. Inverting the MSB accomplish the code change from CTC to COB.

The S/H amplifier is of the same type used for the ADC and a 1,000 pF holding capacitor is also used. The sample

command is derived from the Control Logic section and spans the time between clock pulses 16 to 27; therefore, the D/A conversion runs for 3.9 usec before it is sampled.

The DAC and the S/H amplifier are both TTL compatible so that TTL devices are use throughout.

The output of the S/H amplifier is delivered to the equalizer output through a lowpass filter LPF2 to smooth out the analog signal. This analog LPF2 is similar to the input filter LPF1, the only difference being the gain as explained in Chapter 2.

4. BASE NUMBER ROM, GAIN CONTROLS AND GAIN COEFFICIENT ROM

Figure 4.7 is the schematic of the section comprising the ROM1 that stores the base numbers, the Gain Controls that adjust the gain of each bandpass filter, the ROM2 that stores the coefficients for every gain setting and a Multiplexer that switches the parallel bus between the ROM's.

Base Number ROM (ROM1)

The ROM1 is a 1024 x 8 bipolar ROM Signetics type 82S181 but only 5 bits are used to address 10 of the 32 bases of the SPP memory. A fast ROM was selected because at the beginning of the program there are only 700 nsec available to output a new base number, to address the ROM2 and to output the respective gain coefficient to the SPP processor (see timing diagram of Figure 4.8). Slower memories, such as MOS types would not have been suitable for this purpose.

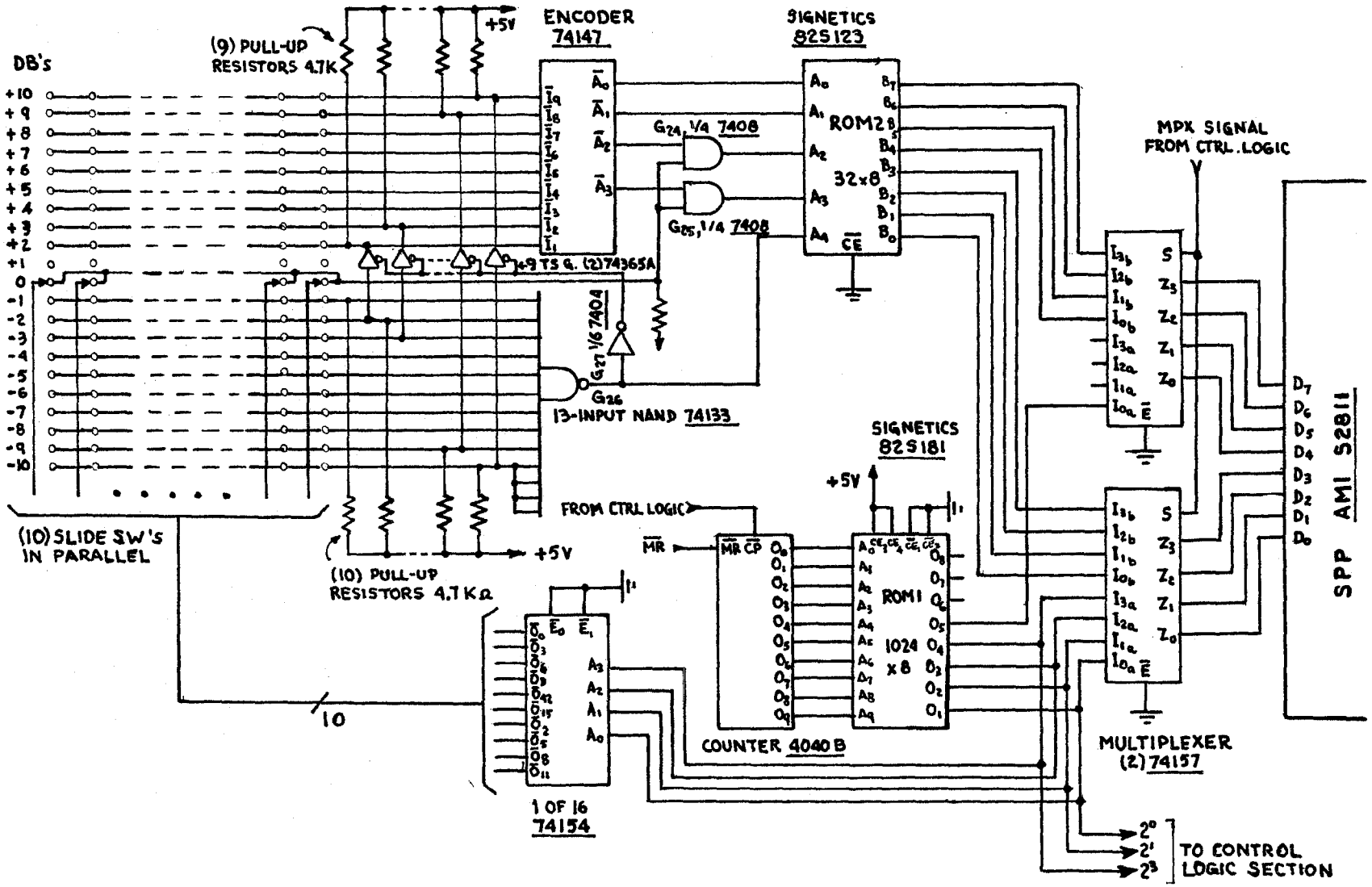


Figure 4.7 Schematic of Gain Controls, ROM1, ROM2 and Multiplexer interface

The ROM1 is addressed by a modulo 1024 counter type 4040B; this is actually a 12 stage counter but only 10 are used. The counter addresses the ROM1 locations that contain the SPP memory Base numbers in the sequence they should be processed according with the pattern of Table 1.1 of Chapter 1. The counter, in turn, is clocked by pulses from the Control Logic section and it is advanced twice during a program length; once at the reset time by the $\overline{RST}$ pulse and once in the pulse 16 of the Master Clock. Since the base sequence repeats every 512 runs of the program, 1024 storage locations are required. As shown in Table 4.1, the 10 possible outputs of ROM1 or, equivalently the 10 base numbers addressed by ROM1 are the Base Nos. 0, 3, 6, 9, 12, 15, 18, 21, 24 and 27. The two intermediate bases, used for the processing of the digital lowpass filters are accessed through instructions in the program.

We have used a CMOS counter for the convenience of having the counter in a single "chip". No interface problems are created, however, because the 82S181 ROM needs only 150 uA maximum for the LOW input current, well within the driving capabilities of the counter. To insure proper HIGH levels at the CP input which is driven from TTL logic, a pull-up resistor is used at that point.

Gain Controls

To control the gain of the 10 octave bands, ten 21-position slide switches are used (see schematic of Figure 4.7).

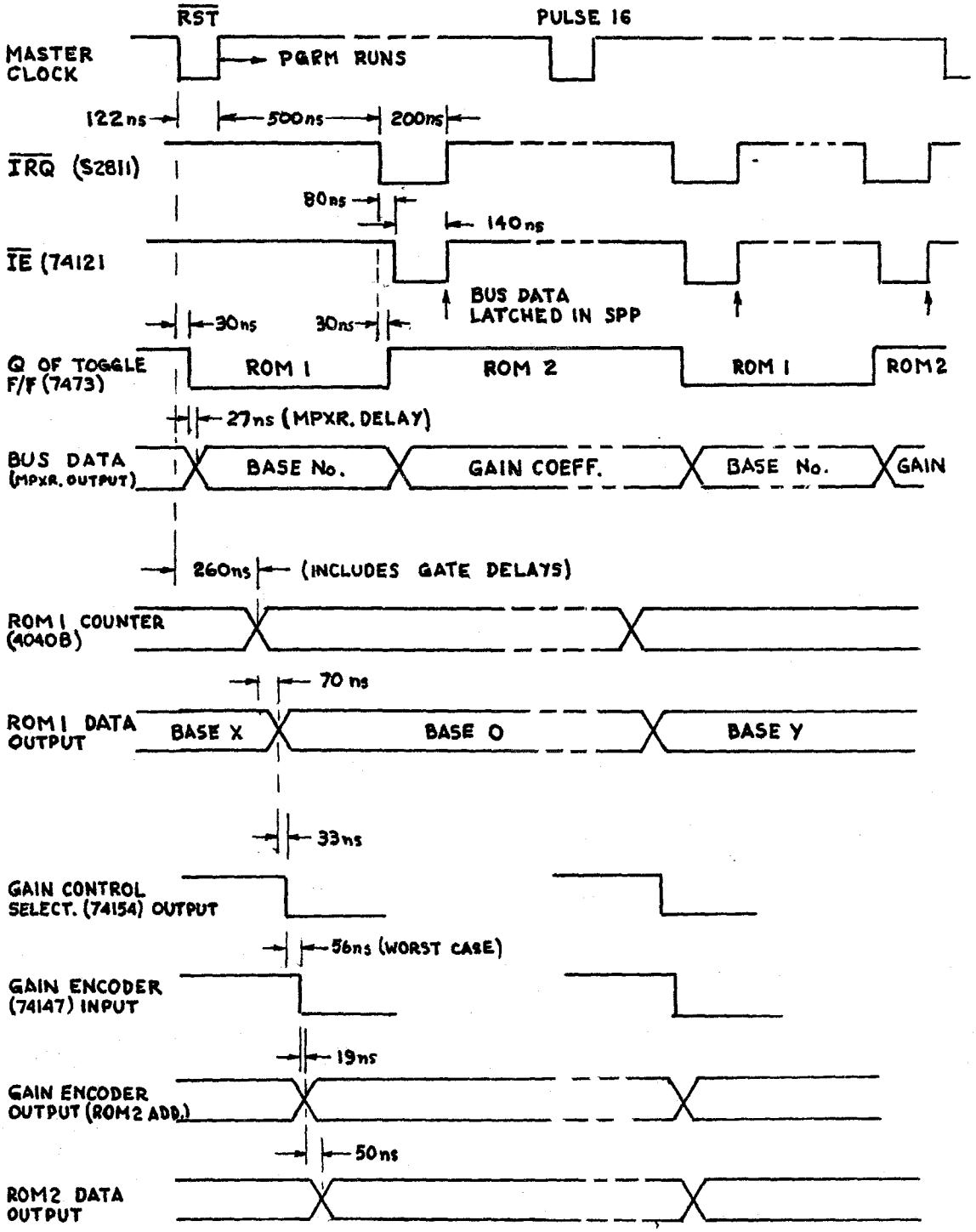


Figure 4.8 Timing diagram of ROM's, Mux. & Gain Ctrl.

This provides for a center (flat) setting plus nominal 10 dB boost and 10 dB cut in steps of 1 dB for each band.

The 10 gain switches are connected in parallel in a 20 line bus with all the lines held HIGH with as many pull-up resistors to the +5V supply. Notice that one of the switches terminals is not connected. To select one of the switches, its wiper is grounded what in turn grounds the bus line to which the wiper is set. The selective grounding of the switch wipers is accomplished by means of a 1 of 16 decoder type 74154, whose input is connected to the ROM1 output and it is therefore addressed with the base number of the BPF being processed. According to the input code of the decoder, one of its outputs goes LOW. Thus, by appropriately connecting 10 of these outputs to the gain control wipers, the control corresponding to the BPF under process can be selected. Only the 4 LSB of the 5 output bits of ROM1 are needed to address the decoder, as shown in Table 4.1. This table makes clear the reason for choosing a 1 of 16 decoder rather than a 1 of 10; the ten input codes span the decimals 0 to 15.

For each filter processed, the 20 line bus is set at a state dictated by the position of the switch selected. Therefore, the base number or equivalently, the BPF being processed is tied to the gain coefficient selected by its corresponding gain control.

The 21 coefficients for the 21 possible filter gains are stored in the ROM2. In order to address this ROM a decimal to binary encoder type 74147 is used. This encoder

TABLE 4.1

BPF Frequency	ROM1 Output	Decimal equiv. of 4 LSB	Output of the decoder
16 KHz	00000	0	0_0
8 KHz	00011	3	0_3
4 KHz	00110	6	0_6
2 KHz	01001	9	0_9
1 KHz	01100	12	0_{12}
500 Hz	01111	15	0_{15}
250 Hz	10010	2	0_2
125 Hz	10101	5	0_5
62 Hz	11000	8	0_8
31 Hz	11011	11	0_{11}

has 9 active LOW inputs and 4 complimentary binary outputs. When one of the I_1 to I_9 inputs is taken LOW the respective code is outputted. If none of the inputs is LOW, the output is all 1's (complimentary 0). It can be seen in Figure 4.7 that the +2 dB to +10 dB lines of the gain switches bus are connected to the 9 inputs of the encoder. The +1 dB line is left unconnected and it thus corresponds to the complimentary 0 output of the encoder. When the wiper of the selected switch is set to one of the upper +1 to +10 dB positions, a 0 to 9 complimentary BCD code is obtained from the 74147.

For the ten lower settings from -1 to -10 dB the res-

pective lines of the bus are connected to a 10-input NAND gate 74133 (this has 13 inputs but 3 are not used). Since the bus lines are pulled HIGH, the output of the NAND gate is normally 0. When the control wiper is grounding one of the ten lower lines, however, the NAND output becomes 1. The output of the NAND gate is used in two ways. One is to enable 9 tri-state gates that interconnect the lower lines to the upper ones when the wiper is in one of the lower settings. The other is to differentiate between the boost and cut positions of the wipers; to this effect, the NAND output is appended to the encoder output as a fifth bit (0 for boost, 1 for cut).

Table 4.2 shows the resultant output codes for the various gain settings, as well as the corresponding coefficients stored in ROM2. These coefficients were calculated as explained in Chapter 2 under Bandpass Filters-Gain Coefficients. For the center position we chose one of the unused codes, 00011, which is obtained by disabling two AND gates placed in the A_2 and A_3 outputs of the encoder.

Gain Coefficient ROM (ROM2) and Multiplexer

The ROM2 is a 32 x 8 bipolar ROM Signetics 82S123 used to store the coefficients for the gain controls. The 8-bit Multiplexer is formed with two 4-bit data selectors 74157 and is used to switch the SPP parallel bus between ROM1 and ROM2 under the control of the program generated $\overline{IRQ}$ signal.

Figure 4.8 shows the timing relationship between the control signals of ROM1 and ROM2 and the gain controls.

TABLE 4.2

Nominal gain setting	Output of 74147 (ROM2 addr.)	Dec. equiv.	Gain coeff.	2's complement coded coeff.
+10	0 0 1 1 0	6	0.3359	0 0 1 0 1 0 1 1
+9	0 0 1 1 1	7	0.2969	0 0 1 0 0 1 1 0
+8	0 1 0 0 0	8	0.2656	0 0 0 0 1 1 1 0
+7	0 1 0 0 1	9	0.2344	0 0 0 1 1 1 1 0
+6	0 1 0 1 0	10	0.2109	0 0 0 1 1 0 1 1
+5	0 1 0 1 1	11	0.1875	0 0 0 1 1 0 0 0
+4	0 1 1 0 0	12	0.1640	0 0 0 1 0 1 0 1
+3	0 1 1 0 1	13	0.1484	0 0 0 1 0 0 1 1
+2	0 1 1 1 0	14	0.1328	0 0 0 1 0 0 0 1
+1	0 1 1 1 1	15	0.1172	0 0 0 0 1 1 1 1
0	0 0 0 1 1	3	0.1016	0 0 0 0 1 1 0 1
-1	1 1 1 1 1	31	0.0938	0 0 0 0 1 1 0 0
-2	1 1 1 1 0	30	0.0859	0 0 0 0 1 0 1 1
-3	1 1 1 0 1	29	0.0781	0 0 0 0 1 0 1 0
-4	1 1 1 0 0	28	0.0703	0 0 0 0 1 0 0 1
-5	1 1 0 1 1	27	0.0625	0 0 0 0 1 0 0 0
-6	1 1 0 1 0	26	0.0547	0 0 0 0 0 1 1 1
-7	1 1 0 0 1	25	0.0469	0 0 0 0 0 1 1 0
-8	1 1 0 0 0	24	0.0391	0 0 0 0 0 1 0 1
-9	1 0 1 1 1	23	0.0313	0 0 0 0 0 1 0 0
-10	1 0 1 1 0	22	0.0234	0 0 0 0 0 0 1 1

5. CONTROL LOGIC AND TIMING

The Control Logic is required to synchronize the operation of the various sections of the equalizer circuit so that the input and output of the proper data from the SPP processor takes place at the proper time during the program.

The Control Logic section generates the following signals (refer to schematic of Figure 4.9 and timing diagram of Figure 4.10):

a) A Master Clock pulse train at a frequency of 4.096 MHz. Since the sampling frequency is 64 KHz, there are 64 clock pulses generated per sample. This clock is used directly to clock out the Serial Output data (SOCK signal) and, in combination with a divider chain, to generate a number of other control signals necessary for the operation of the equalizer.

b) A $\overline{\text{RST}}$ pulse, which is the first of the 64 clock pulses per sample. The $\overline{\text{RST}}$ signal is used to synchronize the enabling of several hardware portions to the beginning of the program so that proper timing is maintained. The $\overline{\text{RST}}$ pulse must do the following:

- Clear all the registers of the SPP and start the execution of the program at instruction No. 1.
- Start the A/D conversion (SOC pulse).
- Latch the output data from the Serial Output port.
- Advance the count of ROM1 (Base No. ROM) counter.

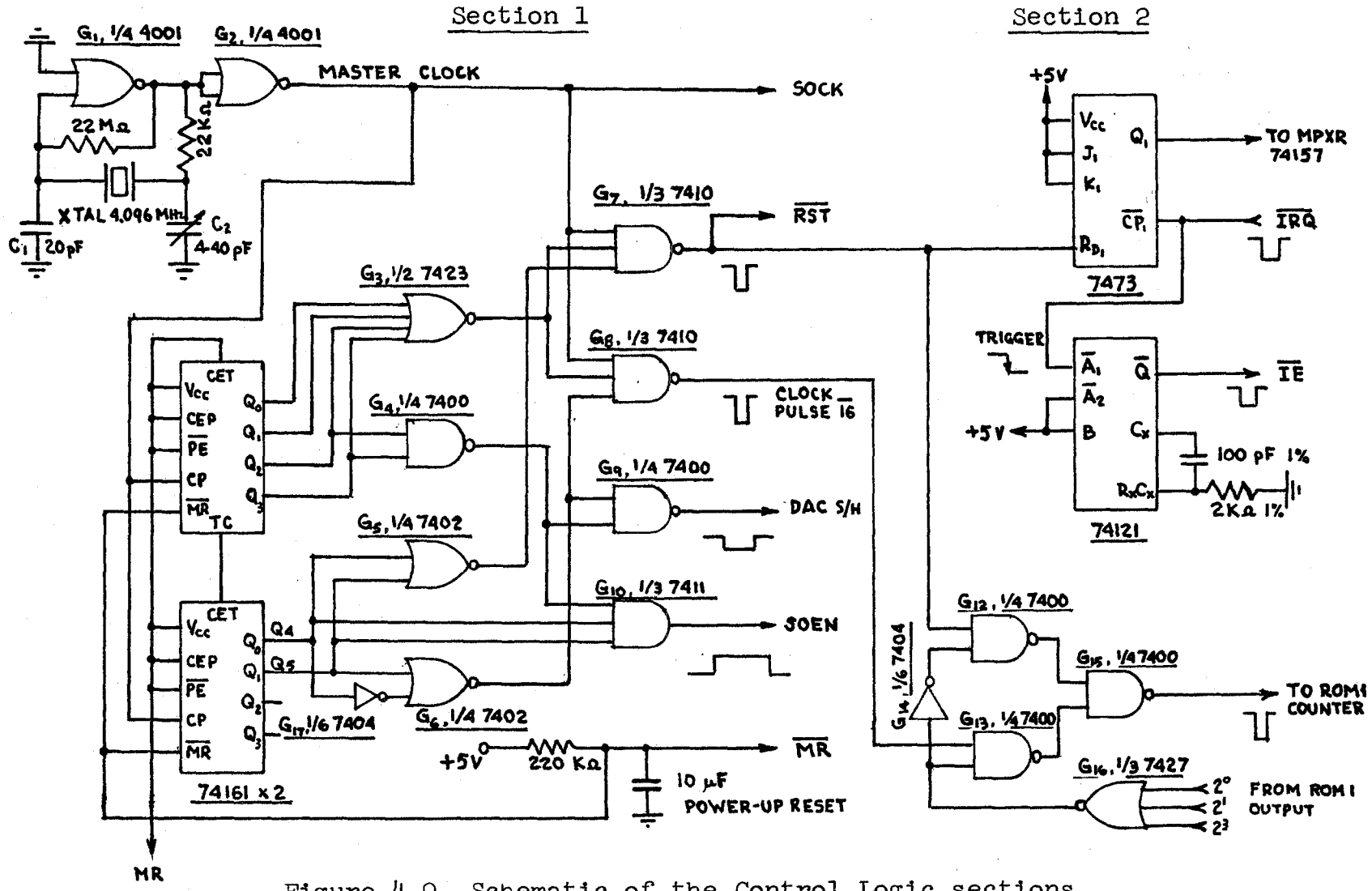


Figure 4.9 Schematic of the Control Logic sections

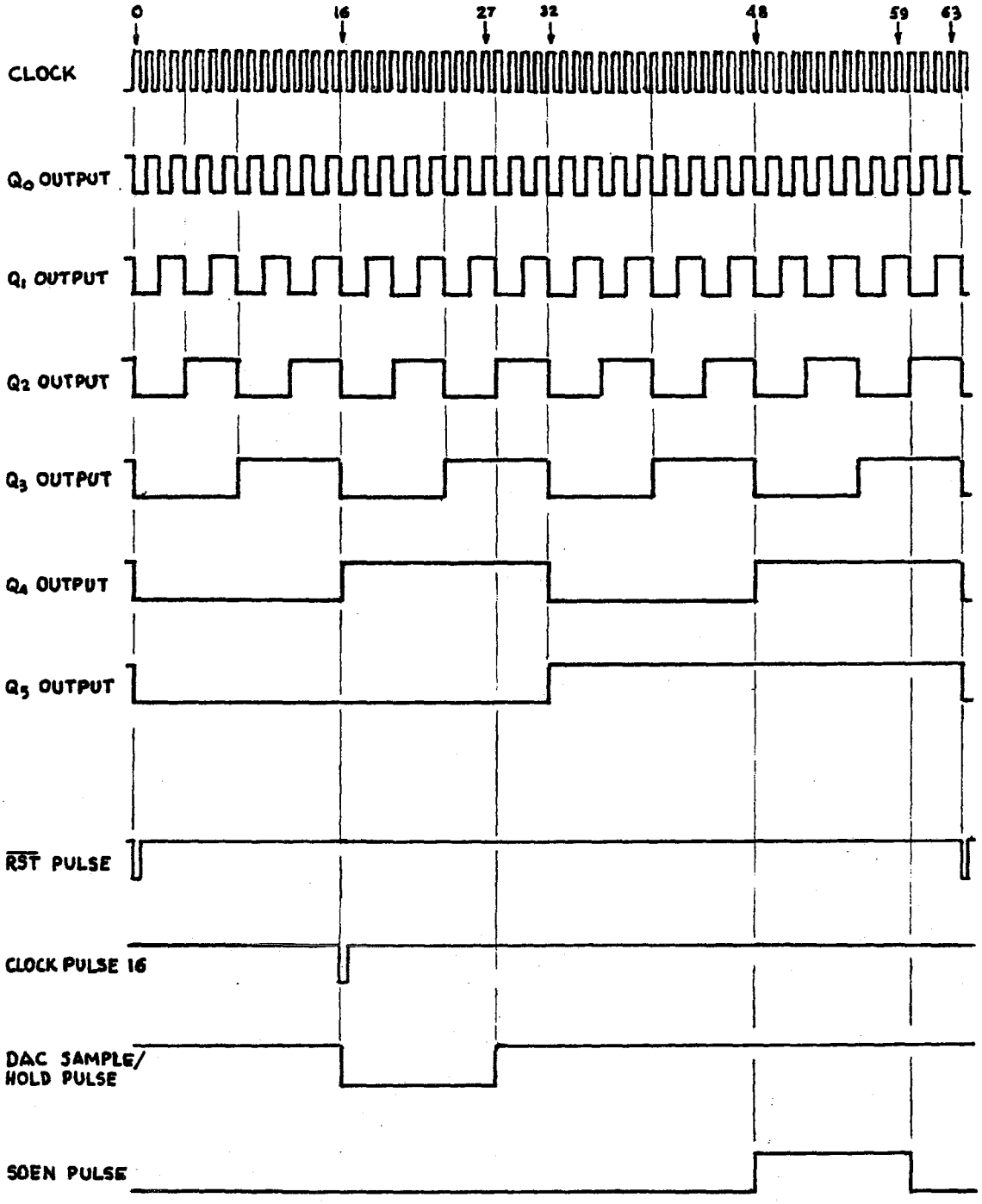


Figure 4.10 Timing diagram of the Control Logic.

-Reset the toggle Flip Flop that switches ROM1 and ROM2.

c) A pulse to advance the ROM1 counter a second time in a program run. This is the clock pulse No. 16. Recall that two BPF's are processed in each sample period; first, the 16 KHz BPF stored in Base No. 0 which is addressed automatically by the $\overline{RST}$ pulse, and second, one BPF that varies from sample to sample and it must therefore be determined by the output of ROM1.

d) The SOEN (Serial Output Enable) pulse that enables the serial output port of the SPP.

e) The "Sample" pulse for the Sample and Hold amplifier of the D/A converter.

f) A toggling signal to toggle the multiplexer that switches either the ROM1 or ROM2 into the parallel Bus of the SPP. This signal is triggered by the $\overline{RST}$ pulse and by the $\overline{IRQ}$ pulse generated by the program (see timing diagram of Figure 4.8)

g) The $\overline{IE}$ (Interface Enable) pulse, synchronized to the $\overline{IRQ}$ pulse, and used to enable the SPP's parallel input port.

h) A power-up Master Reset pulse to place all hardware in initial status at turn on.

Note that all the signals to control the SRI (Serial

Input port) and the A/D conversion are generated by the same A/D converter independently from the Master Clock, as explained under "Analog to Digital Converter" above. Only the $\overline{RST}$ pulse used as SOC (Start Of Conversion) signal maintain the necessary synchronization of the ADC operation with the program.

Hardware Implementation

Most of the logic is combinatorial. Only two synchronizations are needed: one for the sampling time, another for the Base No. and Gain data input to the SPP. There are two sections of control logic.

Section 1 is synchronous with the sampling time but asynchronous with the program, except for the $\overline{RST}$ pulse which restarts the program at each sample time . This section generates, besides the $\overline{RST}$, the SOEN signal and SOCK clock for the control of the D/A converter and Serial Ouput port, the Sample signal for the DAC Sample and Hold amplifier and a pulse (clock pulse No. 16) to advance the ROM1 counter.

Section 2 is synchronous with the program but asynchronous with the sampling, except at Reset. The synchronization with the program is accomplished via the software initiated $\overline{IRQ}$ (Interrupt Request) pulse, which in turn generates the $\overline{IE}$ pulse that enables the parallel bus of the SPP to transfer data from ROM1 or ROM2. This section also controls the Multiplexer to feed the appropriate data at the proper time.

Since the two sections are asynchronous they can run on different clocks; 4.096 MHz and 20 MHz (in SPP) respectively.

Section 1. As shown in the schematic diagram of Figure 4.9, this section is composed of the Master Clock, divider and combinatorial logic to generate the appropriate pulses.

The Master Clock is simply made of (2) NOR gates G_1 and G_2, connected as inverters, with a 4.096 MHz crystal to insure stable clock frequency. Capacitor C_2 is used to adjust slightly the frequency.

The clock output is divided by a modulo 64 counter, formed with (2) 74161 synchronous counters (two outputs are not used). A synchronous counter was chosen to avoid or minimize the decoding spikes that would have been produced if a ripple counter had been used. From the timing diagram of Figure 4.10 it can be seen how the outputs of the divider are combined to generate the required pulses. The propagation delays have been ignored in the diagram because they are about the same for all the signals due to the synchronous nature of the counter. The delays are also small (less than 20 nsec typically) compared with the clock cycle of 244 nsec.

The combination logic is almost self-explanatory. For example, the clock pulse No. 16 for the ROM1 counter is separated with the combination $\overline{Q_5}.Q_4.\overline{Q_3}.\overline{Q_2}.\overline{Q_1}.\overline{Q_0}.Clk$. The SOEN pulse is HIGH during the time between clock cycles 48 to 59. The proper combination is, therefore, $Q_5.Q_4.(\overline{Q_2.Q_3})$. Similarly, the S/H pulse for the D/A conversion is LOW during the time between clock cycles 16 to 27. Hence, it is formed with the combination $\overline{\overline{Q_5}.Q_4.(\overline{Q_2.Q_3})}$. The various output signals are distributed to the appropriate points in the cir-

cuit (refer to block diagram of Figure 4.1). The application of these signals is explained under the respective sections of hardware (see previous sections of this Chapter).

Section 2. The signals from this section are used to control the parallel data input to the SPP processor S2811. This data consist of the Base Nos. of the SPP's memory where the data for the BPF's is stored and the Gain coefficients corresponding to the BPF being processed. As explained before, the Base Nos. are stored in ROM1 in the proper sequence (see Figure 2.1) and the Gain coefficients are stored in ROM2. The data from these two ROM's must be inserted at precise points in the program, as shown in the Program Event timing of Figure 4.2.

In the typical use of the SPP S2811, a host microprocessor would be used to synchronize the exchange of data via the parallel bus. In our stand-alone application, however, we can use the $\overline{IRQ}$ pulse to insured the required synchronism due to the fact that the parallel data goes only one way; it is always written into the SPP. The $\overline{IRQ}$ output of the SPP is set LOW upon the execution of the instruction JMIF during the program run. As shown in Figure 4.9, this $\overline{IRQ}$ transition is used for two purposes. One is to toggle the JK Flip Flop 7473 whose output switches the Multiplexer 74157 x 2 between ROM1 and ROM2. The other is to trigger a monostable multivibrator 74121 that generates the $\overline{IE}$ (Interface Enable) pulse. The $\overline{IE}$ input to the SPP must be set LOW to enable the parallel bus input; the data on this bus is

latched on the LOW to HIGH transition of $\overline{IE}$ which also resets the $\overline{IRQ}$ to the HIGH state.

With reference to the timing diagram of Figure 4.8 and the schematic diagram of Figure 4.9, the $\overline{RST}$ pulse first forces LOW the Q output of toggle 7473, thus switching the Multiplexer to ROM1. The Base No. 0 must be in bus at this time, but is not necessary to input it since the $\overline{RST}$ pulse automatically sets the Base register to 0. The toggling through $\overline{RST}$ is, however, necessary because there are three $\overline{IRQ}$'s generated during the program and the $\overline{RST}$ must complete the required even number of toggles. Thereafter, every time the $\overline{IRQ}$ goes LOW the F/F 7473 toggles, the Multiplexer switches ROM's placing the appropriate data in the parallel bus and a pulse of about 140 nsec (the $\overline{IE}$ pulse) is generated by the monostable 74121. At the rising edge of this $\overline{IE}$ pulse, the data in bus is latched into the SPP buffer and the $\overline{IRQ}$ goes HIGH again.

This whole sequence is repeated three times during a program run. The first time, the bus is multiplexed to ROM2 and the Gain coefficient for the first filter to be processed (which is always the 16 KHz filter) is inputted. The second time, the Multiplexer is switched to ROM1 and a Base No. other than 0 is latched at the rise of $\overline{IE}$ to address the second BPF to be processed. Note that the ROM1 address, that is the Counter 4040B, had been advanced to another Base No. with the clock pulse No. 16 some time between the first and second $\overline{IRQ}$. The third $\overline{IRQ}$ brings to the SPP the Gain coeffi-

cient for the BPF corresponding to the Base previously inputted.

The output of ROM1 addresses the ROM2 via the Band Gain switches and gain level encoder, as explained in Section 4 of this Chapter. The respective timing diagram is shown in Figure 4. where the propagation delays for all the above operations have been indicated. In all cases, stable data is available in bus at the rising edge of $\overline{IE}$, when that data is latched into the SPP.

ROM1 Counter Control

We have seen that the ROM1 contains the SPP's memory Base Nos., or equivalently the BPF's, in the sequence they should be processed for proper time interleaving. The synchronism of the ROM1 address counter with the program is accomplished through the use of the $\overline{RST}$ pulse. Since two BPF's are processed per sampling period, the counter is advanced twice per sample. The first time, the $\overline{RST}$ pulse advances the counter to an address containing the Base No. 0. The second time, the clock pulse No. 16 sets the counter to address a Base other than 0. In both cases, the output of ROM1 is used to address the gain coefficient for the setting of the respective BPF gain control, that is, for the BPF whose Base No. is in the bus.

From the foregoing, it is seen that there is a one to one correspondence between the Base No. of one BPF and the coefficient for the setting of its gain control and that all the synchronism with the program is in the ROM1 counter. In

other words, if the counter operates in the proper sequence the proper Base will be addressed in ROM1 and fed to the SPP at the proper time. Since the Base No. (output of ROM1) selects the Gain control, the right Gain control will be enabled and since the Gain control interfaces with the gain coefficient ROM2 the appropriate coefficient will be outputted. Now, the $\overline{RST}$ pulse that advances once the counter, also starts the program run, starts the A/D conversion and enables the output of ROM2 through the $\overline{IRQ}$ toggle at the beginning of the program. Therefore, if the counter keeps the proper sequence, anything else going wrong, such as $\overline{IRQ}$ not toggling or other errors will affect only one sample; everything will be reset at the end of the program length.

If, however, the counter jumps one count and starts addressing the Base No. 0 (16 KHz BPF) when it should have been addressing some other Base (8 KHz, 4 KHz, etc. BPF), the synchronism between the program and the Base No. will be lost. Thus, means should be provided to restore the proper sequence of Base Nos., for otherwise the wrong sequence would continue indefinitely. This is easily accomplished because the program always starts with Base No. 0 and therefore that Base is addressed every other count of the counter. Note, incidentally, that if the counter jumps two counts the proper sequence will not be lost. The $\overline{RST}$ pulse always advances the counter to Base No. 0 and the clock pulse No. 16, to a Base other than 0. Therefore, in order to keep the proper sequence, the $\overline{RST}$ pulse should advance the counter only if

the ROM1 output (Base No.) is not 0 and the clock pulse No. 16 should advance the counter only if the ROM1 output is 0. This feature has been implemented with gates G_{12} through G_{16} (see Figure 4.9). The reference for the 0 output from ROM1 is taken from three outputs which are all 0 only when the whole output is 0 (there are many choices). The three lines used correspond to the binary weights 2^0, 2^1 and 2^3. It can be seen that gate G_{13} that receives the clock pulse No. 16, is enabled only when those three lines are at binary 0. Similarly, gate G_{12} that gets the $\overline{RST}$ pulse, is enabled only when at least one of the three output lines is not at 0. This arrangement also helps for initialization, since at power turn-on the $\overline{RST}$ pulse should start the program but should not advance the counter which will be then at 0.

Power supply requirements

Power supplies of +5V and ±15V are required for the equalizer. The currents for one channel are as follows:

SPP Processor	240 mA @ +5V	
LF347 op amps (3)		22 mA @ ±15V
ADC and DAC	57 mA @ +5V	45 mA @ ±15V
S/H amplifiers (2)		10 mA @ ±15V
ROM's 82S123 and 82S181	202 mA @ +5V	
All TTL IC's	720 mA @ +5V	
All CMOS IC's	0.16 mA @ +5V	
Totals	1,220 mA @ +5V,	77 mA @ ±15V

Then, +5V @ 2.4A and ±15V @ .15A are required for 2 channels.

SUMMARY AND CONCLUSIONS

It has been shown that digital signal processing can be applied to home entertainment audio equipment using presently available hardware. It is conceivable that the same techniques could be applied to other, non-professional audio components. The cost and complexity of the digital version is obviously much more than that of the analog one, but both the price and complexity of the digital approach may be expected to decrease substantially with the present trend in integration and improved production of digital hardware.

There are of course, improvements and alternatives that can be introduced in our design. The dynamic range we could achieve is comparable with equivalent analog systems. A natural attempt would be to try to improve the dynamic range by using more bits of resolution. With 16 bits, for example, an excellent range of 96 dB may be realized. At the required speed, however, 16 bit processing would imply a major increase in cost of the processor and the associated analog to digital converter, with the present state of the art.

Another possibility is the addition of a display to show the gain levels and some form of momentary switch to increase or decrease the gain by addressing the coefficient ROM with an up-down counter. This is mostly cosmetics.

A more interesting variation would be to integrate a real time analyzer with the equalizer. In this case an appropriate acoustic source, such as pink noise through an amplifier and speaker, can be used to equalize a listening room. The pink noise would be received by a microphone and fed to the equalizer after the required amplification. The output of the bandpass filters would then be compared with the signal fed to the speaker amplifier and the gain controls would be adjusted to match the equalized signal with the original one. The difference between the two signals could be displayed on the front panel to facilitate the adjustments or perhaps the display can show the characteristics of the listening environment. This application will require sharply selective bandpass filters, a suitable processor capable of handling the signals at the proper speed to allow time for the additional operations, and the necessary display and interfaces. It would probably be a major undertaking but from the results of our project it seems possible to accomplish it within a reasonable cost.

REFERENCES

1. Alan V. Oppenheim, "Applications of Digital Signal Processing"; Prentice Hall, 1978.

2. A. Peled and B. Liu, "Digital Signal Processing"; John Wiley & Sons, 1976.

3. R. B. Randall, "Frequency Analysis"; Application Notes from Bruel & Kjaer, 1977.

4. William D. Stanley, "Digital Signal Processing"; Reston Publishing Co., 1975.

5. Richard J. Karwosky, Computer implementation of Digital Bandpass Filters; Electronic Design, Sept. 1, 79 and Dec. 20, 1979.

6. L. R. Rabiner and B. Gold, "Theory and Application of Digital Signal Processing"; Prentice-Hall, 1975.

7. Technical Articles from American Microsystems on the AMI S2811. Reprints from Electronics, August 30, 1979, Electronic Design, February 15, 1980.

8. E. Christian and E. Eisenman, "Filter Design Tables and Graphs"; John Wiley & Sons, 1966.

9. D. E. Johnson, et al., "A Handbook of Active Filters"; Prentice-Hall, 1980

10. Eugene L. Zuch, Editor, "Data Acquisition and Conversion Handbook"; Datel Intersil, 1979.

11. American Microsystems Inc.; "S2811 User's Manual"

APPENDIX

The following pages contain specifications of the signal processor AMI S2811, the A/D converter DDC ADH-8585 and the D/A converter Burr Brown DAC-800. For the sake of brevity, only the relevant parts of those specifications have been included.

AMERICAN MICROSYSTEMS, INC.

ADVANCED PRODUCT DESCRIPTION

S2811

SIGNAL PROCESSING PERIPHERAL

September 1981

Features

- ☐ Programmable for Digital Processing of Signals in Voice-Grade Communications Systems and Other Applications with Signals in the Audio Frequency Range
- ☐ On-Chip 12-Bit Parallel Multiplier (One Cycle Multiplication Time)
- ☐ Built-in Program ROM (256x17)*, 3-Port Data Memory (256x16) and Add/Subtract Unit (ASU)
- ☐ Pipeline Structure for High Speed Instruction Execution (300ns Cycle Time)
- ☐ Bus-Oriented Parallel I/O for Easy Microprocessor Interface
- ☐ Additional Double Buffered I/O for Ease of Asynchronous Serial Interface
- ☐ On-Chip Crystal Oscillator (20MHz) Circuit
- ☐ Pre-Programmed Standard Parts Available

General Description

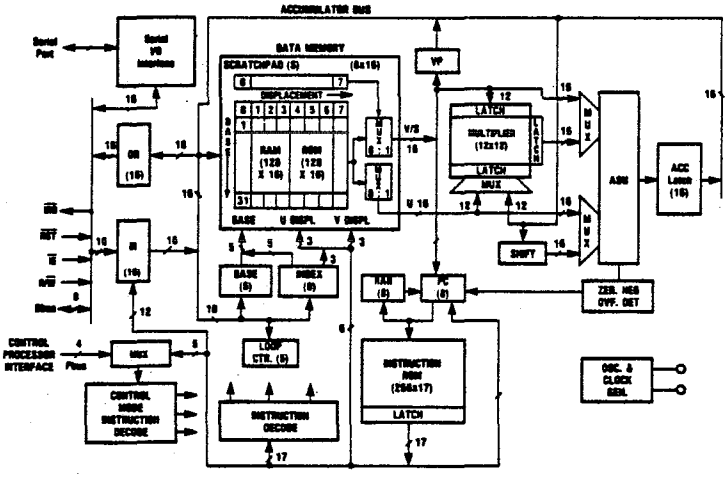
The S2811 Signal Processing Peripheral (SPP) is a high speed special purpose arithmetic processor with on-chip ROM, RAM, multiplier, adder/subtractor, accumulator and I/O organized in a pipeline structure to achieve an effective operation of one multiply, add and store of up to 12-bit numbers in 300 nanoseconds.

User Support

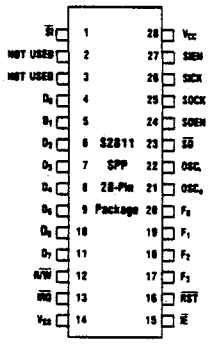
A real time in circuit emulator, the RTDS2811 is available. This is a fully compatible hardware emulator with software assembler/disassembler and editor for rapid program development and debugging. An S2811 assembler and software simulator program package SSPP2811 is also available.

*Out of the 256 instruction locations of the ROM, 250 are usable by the user program. Six instruction locations are reserved for in-house testing.

Block Diagram



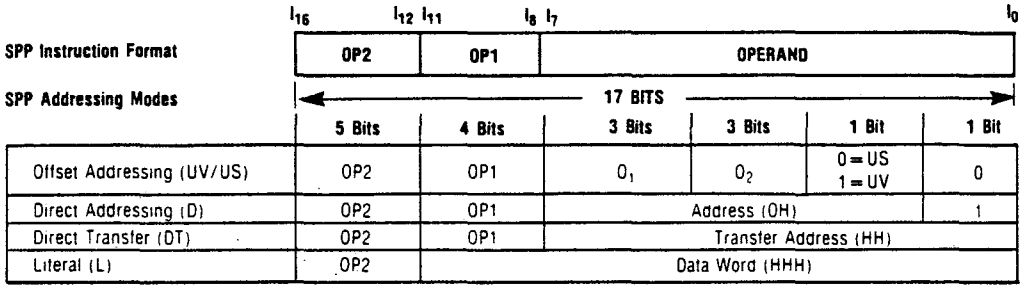
Pin Configuration



The SPP is a memory-mapped peripheral, occupying 16 locations of the microprocessor memory space. Providing the proper SPP address will activate the corresponding control mode.

The control modes and the LIBL command enable real-time modification of the SPP programs. This permits a single SPP program to be used in several different applications. For example, an SPP might be programmed as a "universal" digital filter, with cutoff frequency, filter order, and data source (serial or parallel port) selected at execution time by the control microprocessor.

Figure 1. S2811 Object Code Instruction Formats.



NOTE: O indicates an octal digit (3 bits) and H indicates a hexadecimal digit (4 bits)

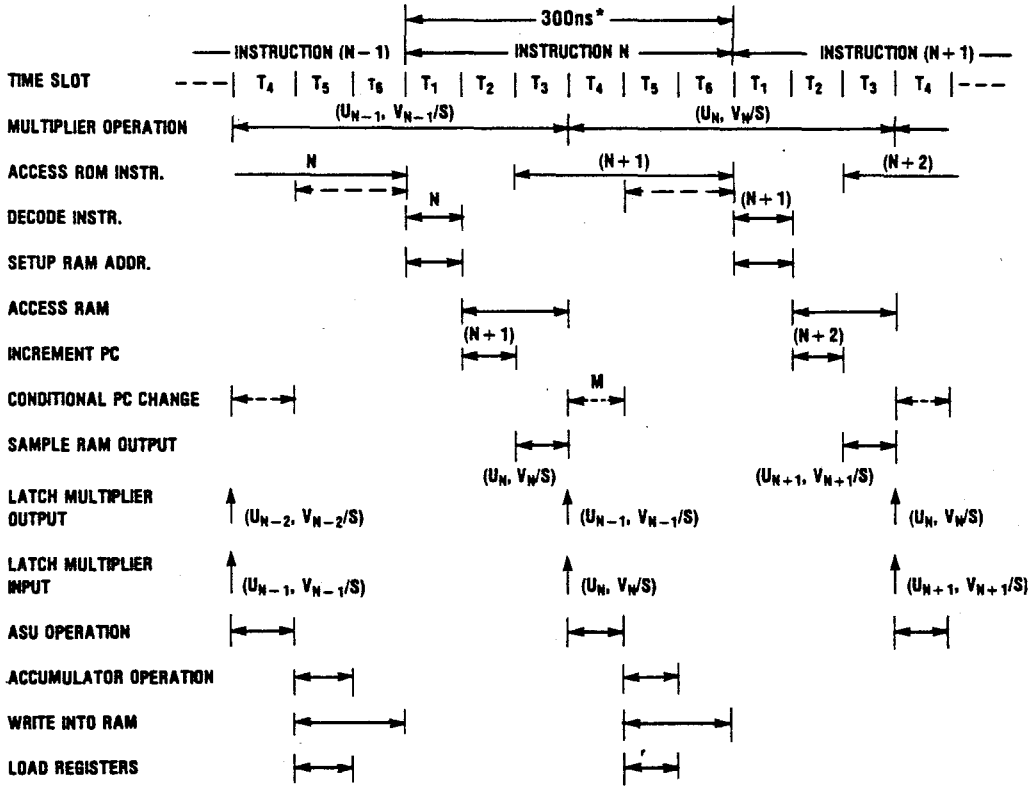
Addressing Mode	Effective Address U	Effective Address V/S	Multiplier Operands
UV	(BAS) + O_1	V = (BAS) + O_2	P = U • V
US	(BAS) + O_1	S = O_2	P = U • S
D	—	OH	P = A • V

Table 1. SPP Control Modes and Operations

Input leads F_0-F_3 define several control modes and operations to facilitate the interface between the SPP and a control processor. In general, these inputs are derived from the control processor address leads. The SPP will therefore occupy 16 memory locations, being a memory mapped peripheral.

F-Bus (F_3-F_0) Hex Value	Mnemonic	Control Modes and Operations: Operation/Function
0	CLR (Clear)	Resets control modes to normal operation.
1	RST (Reset)	Software master reset. Clears all SPP registers and starts execution at location 00. Also resets control modes to normal operation.
2	DUH (Data U/H)	Specifies MSByte of data word. DUH terminates data word transfer.
3	DLH (Data L/H)	Specifies LSBs of data word.
4	XEQ (Execute)	Starts execution at location specified on data lines (HH).
5	SRI (Ser. Inp.)	Enables serial input port.
6	SRO (Ser. Out.)	Enables serial output port.
7	SMI (S/M Inp.)	Converts sign-magnitude serial input data to 2's complement form.
8	SMO (S/M Out.)	Converts 2's complement internal data to sign-magnitude serial output.
9	BLK (Block)	Enables block data transfer.
A	XRM (Ext. ROM)	Permits control of SPP using external instruction ROM. A special mode used primarily for testing.
B	SOP	Set Overflow Protect (Normal mode of operation).
C	COP	Clear Overflow Protect.
D,E,F		Do not use.

Figure 2. SPP Instruction Timing Diagram



*ASSUMES 20MHz CLOCK FREQUENCY

EACH TIME SLOT = 50ns*

Figure 3. SPP Interface Environment

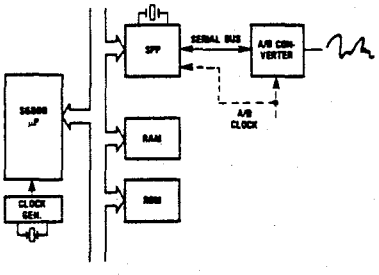


Figure 4. Loading BASE and INDEX Registers and LOOP and PROGRAM Counters

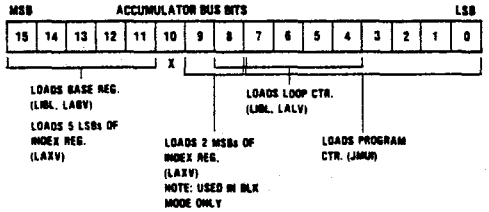
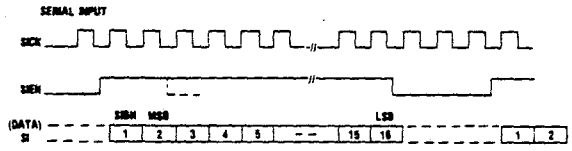
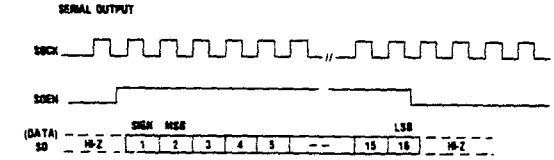


Figure 5D. SPP Serial Interface Timing

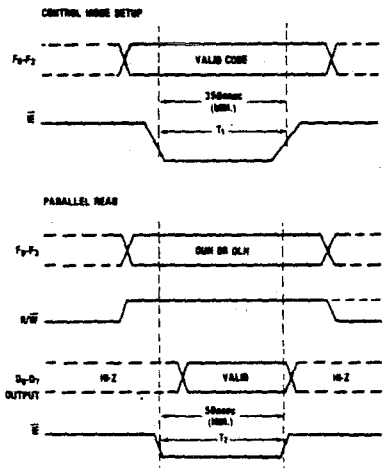


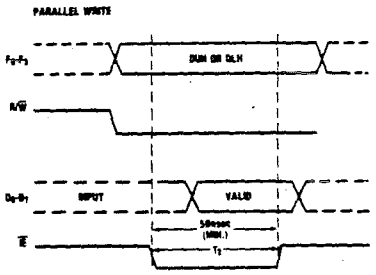
1. SIEN MUST BE SYNCHRONIZED TO THE FALLING EDGE OF SICK SUCH THAT THE RISE AND FALL OF SIEN FOLLOW FALLING EDGE OF SICK.
2. DATA MAY CONTAIN 1 TO 16 BITS DEFINED BY WIDTH OF SIEN. SPP WILL LEFT JUSTIFY DATA WORDS 16 BITS.
3. DATA ARE SAMPLED ON THE TRAILING EDGE OF SICK.
4. MINIMUM 16 SICK PULSES + 64 CYCLES OF S2815 OSCILLATOR ARE REQUIRED BETWEEN SIEN RISING EDGES.
5. IF SERIAL INPUT BUFFER IS FULL, SPP WILL IGNORE NEW INPUT SAMPLES.
6. THE SERIAL DATA IS INVERTED AND MAY BE EITHER IN SIGN + MAGNITUDE OR TWO'S COMPLEMENT CODE.



1. RISE AND FALL OF SOEN MUST FOLLOW FALLING EDGE OF SOCK.
2. OUTPUT DATA WILL BE 1 TO 16 BITS DEFINED BY WIDTH OF SOEN.
3. DATA ARE VALID FROM RISING EDGE TO RISING EDGE OF SOCK SO THAT THE RECEIVING SYSTEM CAN SAMPLE DATA ON TRAILING EDGE.
4. MINIMUM 16 SOCK PULSES + 64 CYCLES OF S2815 OSCILLATOR ARE REQUIRED BETWEEN SOEN RISING EDGES.
5. IF THE SERIAL OUTPUT BUFFER IS EMPTY, ALL ONES WILL BE OUTPUT.
6. SO WILL BE IN A HIGH IMPEDANCE STATE WHEN NOT ENABLED BY SERIAL OUTPUT SEQUENCE.
7. THE SERIAL DATA IS INVERTED AND MAYBE EITHER IN SIGN + MAGNITUDE OR TWO'S COMPLEMENT CODE.

Figure 5E. SPP Parallel Interface Timing





NOTE: ALL TIMES SHOWN ARE MINIMUM AND MUST BE INCREASED PROPORTIONALLY WHEN USING REDUCED SPECIFICATION PARTS AS FOLLOWS:

$$T_{1\,min} = \left[\frac{5000}{f_{CLK}}\right] + 50 \quad \text{nsec}$$

$$T_{2\,min} = \frac{1000}{f_{CLK}} \quad \text{nsec}$$

WHERE f_{CLK} IS IN MHz

Table 2. SPP Instruction Set

OP1 Instructions

Type	Mnemonic	Hex Code I11-18	Address Modes	Operations	Description
No Operation	**NOP**	0	- - -	None	**NO OP**eration
Accumulator Operations	**ASB**	C	- - -	ABS (A)→A	**ASB**olute value of accumulator is placed in accumulator.
	NEG	D	- - -	—(A)→A	**NEG**ate accumulator contents (two's complement) and replace in accumulator.
	SHR	E	- - -	(A)/2→A	**SH**ift **R**ight accumulator contents 1-bit position. Equivalent to dividing contents by two.
	SGV	F	UV/US, D	(A)→A, if sign (A) = sign V/S −(A)→A, if sign (A)≠sign V/S	**S**ign of RAM output **V** is the sign of accumulator contents. Accumulator contents are negated (two's complement) if different sign from V. Useful in implementing hard limiter function.
Addition Operations	**AUZ**	2	UV/US	(U) + 0→A	**A**dd **U** and **Z**ero. Loads RAM output U into the accumulator.
	AVZ	1	UV/US, D	(V/S) + 0→A	**A**dd **V**/S and **Z**ero. Loads RAM output V/S into the accumulator.
	AVA	8	UV/US, D	(V/S) + (A)→A	**A**dd **V**/S and **A**ccumulator contents. Sum is placed back into accumulator.
	AUV	4	UV/US	(U) + (V/S)→A	**A**dd RAM outputs **U** and **V**/S and place sum in accumulator.
Subtraction	**SVA**	9	UV/US, D	(V/S)—(A)→A	**S**ubtract **V**/S and **A**ccumulator contents. The difference (V—A) is placed in the accumulator.
	SVU	5	UV/US	(V/S)—(U)→A	**S**ubtract RAM outputs **V** and **U** and place difference (V—U) in the accumulator.
Multiply/ Add Operations	**APZ**	3	- - - (current inst.) UV/US, D (prec. instr)	(P) + 0→A	**A**dd **P**roduct and **Z**ero. Loads multiplier product into the accumulator. The multiplier inputs were set up in the preceding instruction by addressing mode.
	APA	A	- - - (current inst.) UV/US, D (prec. instr)	(P) + (A)→A	**A**dd **P**roduct and **A**ccumulator contents. Result is placed in the accumulator. The multiplier inputs were set up in the preceding instructions by addressing mode.
	APU	6	UV (current instr) UV/US, D (prec. instr)	(P) + (U)→A	**A**dd **P**roduct and RAM output **U**. Sum is placed in accumulator. The multiplier inputs were set up in preceding instruction by addressing mode.
Multiply/ Subtract Operations	**SPA**	B	- - - (current instr) UV/US, D (prec. instr)	(P)—(A)→A	**S**ubtract **P**roduct and **A**ccumulator contents. Difference (P—A) is placed in accumulator. The multiplier inputs were set up in preceding instruction by addressing mode.
	SPU	7	UV/US (current instr) UV/US, D (prec. instr)	(P)—(U)→A	**S**ubtract **P**roduct and RAM output **U**. Difference (P—U) is placed in accumulator. The multiplier inputs were set up in preceding instruction by addressing mode.

Table 3. SPP Instruction Set
OP2 Instructions

Type	Mnemonic	Hex Code I16-I12	Address Modes	Operations	Description
Load Instructions	**LLTI**	1E	Literal	HHH→IR	Load LiTeral in Input register. A 12-bit (3 hex digits) literal is transferred to the input register. This instruction cannot be used with an OP1 instruction or with a specified addressing mode. Literal is left justified to occupy bits 4-15 in register.
	LBL	07	- - -	(IR)→BAS (IR)→LC	Load Input contents to Base register and Loop counter. See Figure 4. Clears input flag (LOW).
	LACO	02	- - -	(A)→OR	Load ACcumulator contents into the Output Register. This is the basic data output instruction. Sets output flag (HIGH). The $\overline{IRQ}$ line will be set low if the SRO mode is not set.
	LAXV	05	UV/US, D	(A)→IX, V/S (A)→A	Load Accumulator contents into index register and RAM location V/S. Accumulator is truncated to 5 most significant bits after the operation. See Figure 4.
	LALV	04	UV/US, D	(A)→LC, V/S	Load Accumulator to Loop counter and RAM location V/S. See Figure 4.
	LABV	03	UV/US, D	(A)→BAS, V/S (A)→A	Load Accumulator to Base and RAM location V/S. Truncate accumulator contents to most significant 5 bits after the operation. See Figure 4.
Data Transfer Instructions	**TACU**	0B	UV/US	(A)→U	Transfer Accumulator Contents into RAM location U.
	TACV	0C	UV/US, D	(A)→V/S	Transfer Accumulator Contents into RAM location V/S.
	TIRV	08	UV/US, D	(IR)→V/S	Transfer Input Register Contents to RAM location V/S. This is the basic data input instruction. Clears input flag (LOW).
	TVPV	09	UV/US, D	VP→V/S	Transfer contents of VP register (equals previous value of output V) to RAM location V/S.
	TAU	10	UV/US	(A)→U	Transfer Accumulator contents into RAM location U using Index register as base.
Accumulator Operations	**CLAC**	01	- - -	0→A	CLear the ACcumulator. Forces SWAP mode to normal operation and clears overflow flag.
Register Manipulation Instruction	**INX**	0D	- - -	(IX) + 1→IX	INcrement the IndeX register.
	DECB	0E	- - -	(BAS)—1→BAS	DECrement the Base register.
	INCB	0F	- - -	(BAS) + 1→BAS	INCrement the Base register.
	SWAP	06	- - -	BAS↔IX	SWAP the roles of Base and Index registers.
Unconditional Branch Instruction	**JMUD**	15	DT	HH→PC	JuMp Unconditionally Direct to location indicated by 8-bit (two hex digits) literal HH. Cannot be used with an OP1 instruction requiring specific addr. mode.
	JMUI	11	UV/US, D	[(IX)]→PC	JuMp Unconditionally Indirect to location indicated by contents of RAM address pointed to by index and displacement indicated by V/S. $[V/S)_{0-7}]$→PC.
Conditional Branch Instructions	**JMCD**	16	DT	HH→PC, if LC≠0 (LC)—→LC	JuMp Conditionally Direct to location indicated by 8-bit (two hex digits) literal HH, if loop counter is not zero. Loop Counter is decremented after the test.
	JMPZ	19	DT	HH→PC if (A) = 0	JuMP to location specified if accumulator contents are Zero as a result of previous instruction.
	JMPN	1A	DT	HH→PC if (A) :	JuMP to location specified if accumulator contents are Negative as a result of previous instruction.
	JMPO	1B	DT	HH→PC if (A) Overflows	JuMP to location specified if accumulator Overflows as a result of previous instruction. Clears overflow flag.
	JMIF	1C	DT	HH→PC if IF = 0	JuMp if Input Flag is low to location specified (Note 4). $\overline{IRQ}$ line will be set low if the SRI mode is not set.
	JMOF	1D	DT	HH→PC if OF = 1	JuMp if Output Flag is high to location specified (Note 4).
Subroutine Instruction	**JMSR**	14	DT	(PC) + 1→RAR, HH→PC	JuMp to SubRoutine. Execution jumps unconditionally to location indicated by 8-bit (two hex digits) literal HH. Return address is stored in RAR. Cannot be used with an OP1 instruction requiring specified address mode.
	RETN	13	- - -	(RAR)→PC	RETurN from subroutine. Execution continues at instruction following the JMSR instruction.

Table 3. SPP Instruction Set
OP2 Instructions (Continued)

Type	Mnemonic	Hex Code I16-I12	Address Modes	Operations	Description
Complex Instructions	**JCDT**	18	DT	HH→PC if LC≠0, (LC)—1→LC (BAS) + 1 →BAS, (IX) + 1→IX	Jump Conditionally Direct Dual Tracking. Increment base and Index registers. Loop Counter is decremented after test.
	JCDI	17	DT	HH→PC if LC ≠0, (BAS) + 1→BAS (LC)—1→LC	Jump Conditionally Direct and Increment base register. Loop Counter is decremented after test.
	TVIB	0A	UV/US	(VP)→V/S, (BAS) + 1→BAS	Transfer contents of VP register to RAM location V/S and Increment Base register.
	MODE	1F	- - -	Control mode replaces OP1	OP1 code in this instruction can select any one of the several control **MODE**s/operations specified in Table 1.
	REPT	12	- - -	PC inhibited if LC≠0 (next instruction) (LC)—1→LC (each iteration of next instruction.)	**REP**eat next instruction until LC = 0. Increment PC to access next instruction, then suppresses increment of PC if LC≠0. Loop Counter is decremented when REPT is executed, so that number of repeats is equal to original value of LC.

NOTES:
1. Whenever the Index register is selected by an instruction OP2 it controls the entire line of code.
2. Loop Counter cannot underflow.
3. S refers to scratchpad.
4. Input flag is low if SPP has not received a new input word.
5. (A) represents truncation of the accumulator to 5 most significant bits (sign and 4 MSB).
6. Multiplier input latches and the VP register are not updated when either the DT or L addressing modes are used in conjunction with an OP2 instruction.
7. - - - indicates don't care address mode.
8. When D address mode is used, accumulator contents as a result of previous instruction replace U input to multiplier.

SPP Addressing Modes

The SPP provides four methods of data access (see Figure 1). In the direct mode, the full address of the data is specified. Due to limitations in the instruction word size, only one data word at a time may be accessed in this manner, and only even displacement addresses.

In the relative (to base) mode the base register is set up using a LLTI/LIBL sequence, or LABV, and two data words are accessed simultaneously by specifying U and V displacements in the instruction word.

Data may be stored/retrieved from the scratchpad memory by specifying the scratchpad mode and providing scratchpad and U port displacements. The U port data is accessed relative to the base register. The scratchpad data is treated exactly the same as data accessed via the U and V RAM ports, except the 8-word scratchpad block is substituted for the V data block.

The fourth addressing mode is dual-tracking base addressing. This mode greatly increases throughput in matrix operations.

The JMIF and JMOF instructions provide the capability to synchronize the SPP when operating in synchronous sampled data systems. When executed these commands cause the SPP to set the $\overline{IRQ}$ output low, thus requesting service from the microprocessor. The SPP can be put in a wait loop until a new data sample is available at the IR or has been read from the OR, as appropriate. The TIRV and LIBL commands facilitate transfer of input data from the IR to data memory or the base register and loop counter respectively. LACO command provides for data transfer to the OR.

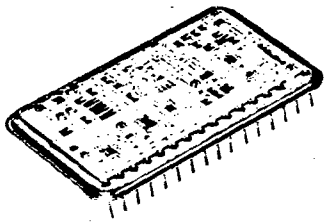
Block Data Transfer (BLK Mode)

The contents of the RAM portion of the data memory may be loaded or dumped via the parallel interface by use of the Block Transfer mode. This mode is ideally suited for transfer using a DMA Controller. The sequence and timing are shown in Figure 6. Eight bit words may be transferred using the DUH mode only. The memory is addressed by the index register in this mode, and the register is automatically incremented after each word transfer. The displacement is addressed by the 2 most significant bits of this register (see Figure 4) so that the addressing is done base-by-base, then next displacement, i.e., columnwise. The starting address is selected by pre-

ILC DATA DEVICE CORPORATION

ADH-8585/ADH-8586

12 BIT HYBRID A/D CONVERTER
5μs Conversion Time; ±0.012% F.S. Range Linearity Error



DESCRIPTION

The ADH-8585 and ADH-8586 are complete in a 32 pin triple DIP hermetically sealed metal package and are pin compatible with other generic ADC-85 12 bit A/D converters. An advanced design using thin-film and MSI technologies results in conversion times of 10μsec for the ADH-8585 and 5μsec for the ADH-8586, corresponding to word rates of nearly 100 KHz and 200 KHz, respectively. Conversion times may be reduced to less than this by short cycling and increasing the internal clock rate if less resolution and accuracy are acceptable. Gain and offset errors can be trimmed to zero using two external potentiometers, making accuracy equal to the ±1/2 LSB linearity. The internal reference and internal clock are externally accessible, and an external clock may be used.

APPLICATIONS

Because of their high reliability, hermetically sealed metal cases, and wide operating temperature ranges, the ADH-8585 and ADH-8586 will meet the most demanding military and industrial requirements. Typical applications include data acquisition systems, automatic test equipment and electronic countermeasures systems. Standard processing at no added cost is based on MIL-STD-883, except for burn-in which is an option. These converters are excellent for remotely located and hard to access equipment where small size and high MTBF are critical.

FEATURES

- *PIN COMPATIBLE ADC-85 TYPE WITH WIDE TEMPERATURE RANGE AND LOWER POWER CONSUMPTION*
- *PIN SELECTABLE CODING: Complementary Binary, Complementary Offset Binary, Complementary Two's Complement*
- *VOLTAGE RANGES: ±2.5V, ±5V, ±10V, 0 to +5V, 0 to +10V*
- *BOTH SERIAL AND PARALLEL OUTPUT*
- *INTERNAL OR EXTERNAL CLOCK*
- *POWER CONSUMPTION 1.2W TYP*
- *SUPPLIES REQUIRED: ±15V AND +5V DC*

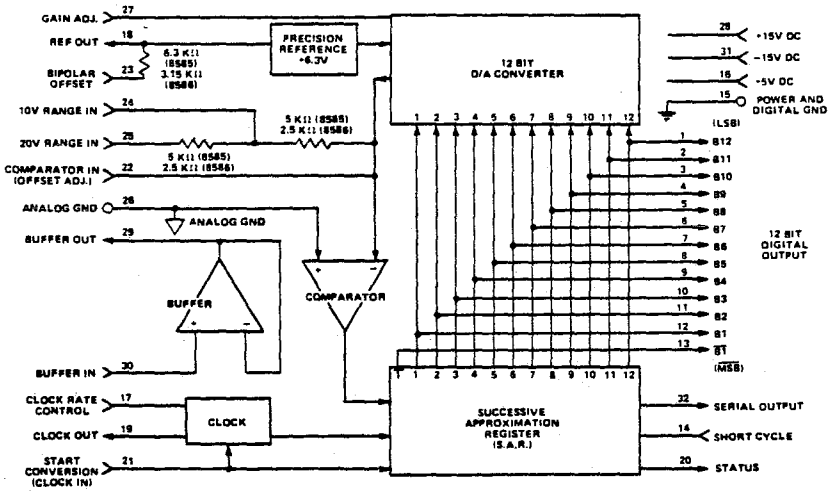


FIGURE 1. ADH-8585 AND ADH-8586 BLOCK DIAGRAM

PARAMETER	UNITS	VALUE		
POWER SUPPLIES				
Supply Voltages	V	+15 ±5%	−15 ±5%	+5 ±5%
Max Voltage Without Damage	V	+18	−18	+7
Current				
8585	mA	35 typ 45 max	25 typ 35 max	50 typ 75 max
8586	mA	40 typ 50 max	30 typ 45 max	110 typ 150 max
Power Supply Sensitivity	%FSR/%PS	±0.002	±0.002	±0.001
THERMAL CHARACTERISTICS				
Temperature Ranges (Case)				
Operating				
−1 Option	°C	−55 to +125		
−3 Option	°C	−25 to +85		
Storage	°C	−55 to +135		
Thermal Impedances				
Case to Air	°C/Watt	θ_{CA} = 15		
Junction to Case	°C/Watt	θ_{JC} = 2		
PHYSICAL CHARACTERISTICS				
Type of Package		Metal case, hermetically sealed, 32 pin triple DIP		
Size	inches	1.75 x 1.05 x 0.22 (4.45 x 2.67 x 0.56 cm)		
Weight	oz	0.67 typ (19 g)		

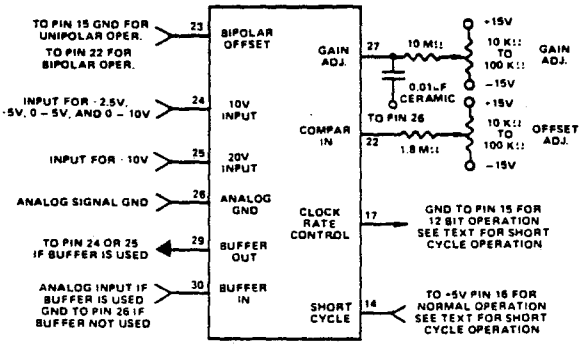


FIGURE 2. CONNECTIONS FOR NORMAL OPERATION

TIMING DIAGRAM AND EXTERNAL CLOCK

The timing for normal 12 bit operation of the ADH-8585 and ADH-8586 is shown in Figure 4. The trailing edge of the START CONVERSION pulse starts the conversion by initiating the first of the 13 equal positive pulses of the internal clock. The leading edge of the first clock pulse then causes the STATUS to go to logic "1" and also causes the first bit to be tried. The leading edge of the second clock pulse causes Bit 2 to be tried, and Bit 1 then becomes valid. The twelfth clock pulse causes Bit 12 to be tried. Bit 12 becomes valid after the leading edge of the clock pulse 13, and the STATUS then drops to logic "0" to indicate that the conversion has been completed.

The SERIAL DATA OUTPUT is a non return to zero type (NRZ). Data for each bit becomes valid at the same time as the corresponding parallel data for the same bit, and remains valid for one clock cycle. A recommended way to clock data from the SERIAL OUTPUT is to use the trailing edge of each clock pulse to shift the data into a twelve bit shift register.

An EXTERNAL CLOCK can also be used as indicated in Figure 4. If the START CONVERSION input is at logic "1" at a time when any internal clock would normally be initiated, the clock is turned off and neither that pulse nor subsequent pulses will exist. Because of this feature an external clock with negative pulses can be applied to the Start Conversion input, pin 21. The leading edge of the first negative external clock pulse turns on the internal clock and Bit 1 is tried. If the external clock pulse is shorter than 200 ns, the Start Conversion input will be at logic "1" at the time when the second internal clock would normally be initiated, so the internal clock will turn off. The external clock is then free to initiate the trial of Bit 2 at any time. The only limitation on the rate of the external clock is that it must be no faster than twice the rate of the internal clock. The clock rate control could be used to adjust the internal clock rate to meet this criterion.

SHORT CYCLING AND CLOCK RATE CONTROL

The minimum conversion time and minimum cycle time depend on the number of bits tried and on the clock period. The clock period must be long enough to allow the comparator to settle out. The ADH-8586 is twice as fast as the ADH-8585 because its comparator settles faster at the expense of lower input impedance.

The number of bits tried can be reduced for both units changing the SHORT CYCLE pin connection. Pin 14 is connected to the next higher bit than the number of bits to be tried. For instance, for 10 bits, connect pin 14 to pin 2 (bit 11). For 8 bits, connect pin 14 to pin 4 (bit 9).

When fewer bits are tried, it is possible to increase the clock rate because less accuracy is required. For full 12 bit operation when no particular or optimum conversion time is required, the Clock Rate Control, pin 17, is usually grounded as shown in Figure 2. In the same way, when rates are not critical, pin 17 can be connected to +5V for 10 bit operation and +15V for 8 bit operation. Optimum clock rates will generally require fine adjustment of the clock voltage. The approximate clock rates with the three standard voltage levels are:

Pin 17 Voltage	Approximate Clock Rate	
	ADH-8585	ADH-8586
0V	1.3 MHz	2.6 MHz
+5V	2 MHz	4 MHz
+15V	2.5 MHz	5 MHz

If both short cycling and pin-programmed clock rate control are used to increase the conversion rate, the following nominal conversion times are obtained for 10 and 8 bits:

Resolution	Pin 17 Voltage	Conversion Time	
		ADH-8585	ADH-8586
12 Bits	0V	10μs	5μs
10 Bits	+5V	6μs	3μs
8 Bits	+15V	4μs	2μs

TRANSITION VALUE		DIGITAL BIT OUTPUTS											
UNIPOLAR	BIPOLAR	MSB											LSB
COMPLEMENTARY BINARY	COMPLEMENTARY OFFSET BINARY	B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12
+F.S. – 3/2 LSB	+F.S. – 3/2 LSB	0 0	0 0	0 0	0 0	0 0	0 0	0 0	0 0	0 0	0 0	0 0	0 0
+3/4 F.S. – 1/2 LSB	+1/2 F.S. – 1/2 LSB	0 0	0 1	1 0	1 0	1 0	1 0	1 0	1 0	1 0	1 0	1 0	1 0
+1/2 F.S. + 1/2 LSB	+1/2 LSB	0 0	1 1	1 1	1 1	1 1	1 1	1 1	1 1	1 1	1 1	1 1	0 1
+1/2 F.S. – 1/2 LSB	–1/2 LSB	0 1	1 0	1 0	1 0	1 0	1 0	1 0	1 0	1 0	1 0	1 0	1 0
+1/4 F.S. + 1/2 LSB	–1/2 F.S. + 1/2 LSB	1 1	0 0	1 1	1 1	1 1	1 1	1 1	1 1	1 1	1 1	1 1	0 1
+3/2 LSB	–F.S. + 3/2 LSB	1 1	1 1	1 1	1 1	1 1	1 1	1 1	1 1	1 1	1 1	0 1	1 0
+1/2 LSB	–F.S. + 1/2 LSB	1 1	1 1	1 1	1 1	1 1	1 1	1 1	1 1	1 1	1 1	1 1	0 1

3A. Theoretical transition values for Complementary Binary and Complementary Offset Binary coding. For Complementary Two's Complement coding, all values are the same as for Complementary Offset Binary, except that the MSB is reversed (MSB bits "0" become "1" and "1" become "0").

VOLTAGE RANGE	FULL SCALE (VOLTS)	1/2 LSB (VOLTS)
±2.5V	2.50000	0.00061
±5V	5.00000	0.00122
±10V	10.00000	0.00244
0 – 5V	5.00000	0.00061
0 – 10V	10.00000	0.00122

3B. Full Scale (F.S.) and 1/2 LSB for 12 bit accuracy.

FIGURE 3. THEORETICAL TRANSITION VALUES

Resolution	+V	R	Conversion Time 8585	Conversion Time 8586
12 Bits	+5V	2 KΩ	6.8 – 10μs	3.5 – 5μs
10 Bits	+15V	5 KΩ	4.0 – 6μs	2.0 – 3.0μs
8 Bits	+15V	5 KΩ	3.5 – 6μs	1.75 – 3.0μs

Note that if the clock rate is increased to a value greater than that specified for the number of bits required, the linearity error will be substantially increased.

The CLOCK RATE CONTROL can also be connected to negative voltages as large as –15V, and this will decrease the clock rate.

The clock rates and conversion times listed are nominal values. To adjust the clock rate accurately, pin 17 may be connected to a voltage divider as shown in Figure 5. R is a multi-turn trim potentiometer with a tempco of ±100 ppm/°C or less. The range of adjustment of the clock rate will be nominally as follows:

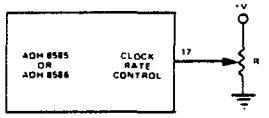


FIGURE 5. OPTIONAL CLOCK RATE FINE ADJUSTMENT

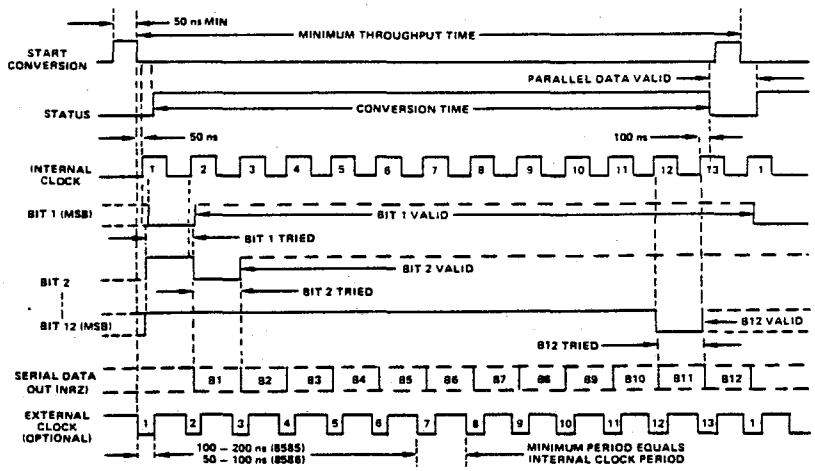
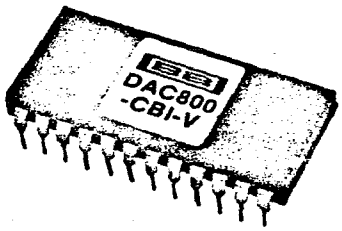


FIGURE 4. ADH-8585 AND ADH-8586 TIMING DIAGRAM

BURR-BROWN®



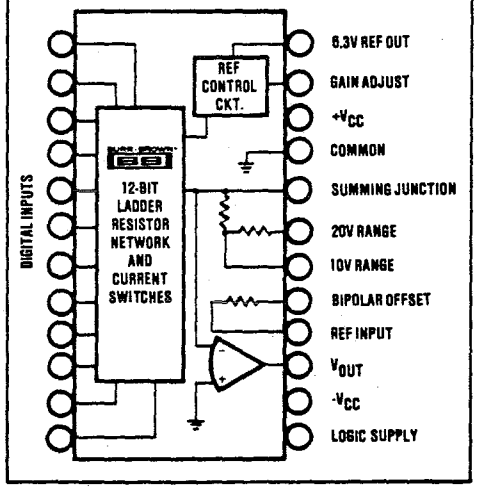
DAC800

Integrated Circuit
DIGITAL-TO-ANALOG CONVERTER

FEATURES

- LOW COST HIGH RELIABILITY SINGLE-CHIP REPLACEMENT FOR INDUSTRY-STANDARD DAC80
- 12-BIT RESOLUTION
- ±1/2LSB MAXIMUM NONLINEARITY, 0°C TO +70°C
- GUARANTEED MONOTONICITY, 0°C TO +70°C
- DUAL-IN-LINE PACKAGE WITH INDUSTRY-STANDARD (DAC80) PINOUT

VOLTAGE MODEL



DESCRIPTION

The DAC800 is a third-generation monolithic Integrated Circuit that is a pin-for-pin equivalent to the industry-standard DAC80 first introduced by Burr-Brown. It has all of the functions of its predecessor plus faster settling time and enhanced reliability because of its monolithic construction.

The current output model of the DAC800 is a single-chip integrated circuit containing a subsurface zener reference diode, high speed current switches, and laser-trimmed thin-film resistors. The DAC800 provides output voltage ranges of ±2.5V, ±5V, ±10V, 0 to +5V, 0 to +10V (V models) or output current ranges of ±1mA or 0 to -2mA (I models).

This high accuracy converter offers a maximum nonlinearity error of ±1/2LSB, ±30ppm/°C maximum gain drift and guaranteed monotonicity, all over 0°C to 70°C. In the bipolar configuration total drift is guaranteed to be less than 25ppm of FSR/°C.

The DAC800 is packaged in a 24-pin dual-in-line package with the popular DAC80 pinout.

For designs that require a wide temperature range and a hermetically sealed package see Burr-Brown models DAC850 and DAC851.

International Airport Industrial Park - P.O. Box 11400 - Tucson, Arizona 85734 - Tel. (602) 746-1111 - Twx: 910-952-1111 - Cable: BBRCORP - Telex: 66-6491

PDS-440

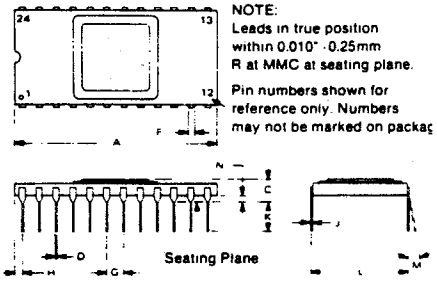
SPECIFICATIONS

ELECTRICAL

Typical at 25°C and rated power supplies unless otherwise noted.

MODEL	DAC800-CBI MIN	TYP	MAX	UNITS
DIGITAL INPUT				
Resolution			12	
Logic Levels (over spec. temp range)(1)				
V_{IH} (Logic "1")	+2		16.5	VDC
V_{IL} (Logic "0")	0		+0.8	VDC
I_{IH} (V_{IN} = +2.4V)			+10	µA
I_{IL} (V_{IN} = +0.4V)			-0.36	mA
ACCURACY				
Linearity Error at 25°C		±1/4	±1/2	LSB
Differential Linearity Error		±1/2	+1, -3/4	LSB
Gain Error(2)		±0.1	±0.2	%
Offset Error(2)		±0.05	±0.15	% of FSR(3)
Monotonicity Temp. Range, min	0		+70	°C
DRIFT(4) (0°C to +70°C)				
Bipolar Drift, (±full scale drift for the bipolar connection)		±10	±25	ppm of FSR/°C
Total error over 0°C to +70°C(5)				
Unipolar		±0.06	±0.15	% of FSR
Bipolar		±0.05	±0.12	% of FSR
Gain		±10	±30	ppm/°C
Unipolar Offset		±1	±3	ppm of FSR/°C
Bipolar Offset		±7	±15	ppm of FSR/°C
Differential Linearity 0°C to +70°C		±1/2	+1, -7/8	LSB
Linearity Error 0°C to +70°C			±1/2	LSB
CONVERSION SPEED/V models				
Settling Time to ±0.01% of FSR For FSR Change				
20 volt range, 2kΩ load		3	5	µsec
10 volt range, 2kΩ load		2.5	4	µsec
For 1LSB Change, Major Carry, 2kΩ load		1.5		µsec
Slew Rate, 2kΩ load	10	15		V/µsec
CONVERSION SPEED/I models				
Settling Time to ±0.01% of FSR For FSR Change				
10Ω to 100Ω load		300		nsec
1kΩ load		1		µsec
ANALOG OUTPUT/V models				
Ranges	±2.5, ±5, ±10, 0 to +5, 0 to +10			V
Output Current	±5			mA
Output Impedance DC		0.05		Ω
Short Circuit to Common, Duration		Indefinite		
ANALOG OUTPUT/I models				
Ranges - Bipolar	±0.88	±1.175,	±1.47	mA
Unipolar	0 to -1.76	0 to -2.35	0 to -2.94	mA
Output Impedance - Bipolar		3.1		kΩ
Output Impedance - Unipolar		7.2		kΩ
Compliance	-2.5		-2.5	V
REFERENCE VOLTAGE OUTPUT	+6.23	+6.30	+6.37	V
Current - for external loads , Source	1.5	2.5		mA
Tempco of Drift		±10	±30	ppm/°C
POWER SUPPLY SENSITIVITY				
+15V and +5V Supplies		±0.0001	±0.001	% of FSR/% Vcc
-15V Supply		±0.003	±0.005	% of FSR/% Vcc
POWER SUPPLY REQUIREMENTS				
±Vcc	±13.5	±15	±16.5	VDC
V_{DD}(6)	+4.5	+5.0	+16.5	VDC
Supply Drain				
+15V, -15V - no load		+8, -20	+12, -25	mA
+5V logic supply		+7	+10	mA
TEMPERATURE RANGE				
Specification	0		+70	°C
Operating	-25		+85	°C
Storage	-60		+100	°C

MECHANICAL



NOTE:
Leads in true position within 0.010" (0.25mm) R at MMC at seating plane.

Pin numbers shown for reference only. Numbers may not be marked on package.

DIM	INCHES MIN	INCHES MAX	MILLIMETERS MIN	MILLIMETERS MAX
A	1.185	1.215	30.10	30.86
C	.105	.170	2.67	4.32
D	.015	.021	0.38	0.53
F	.035	.060	0.89	1.52
G	.100 BASIC		2.54 BASIC	
H	.030	.070	0.76	1.78
J	.008	.012	0.20	0.30
K	.120	.240	3.05	6.10
L	.600 BASIC		15.24 BASIC	
M	–	10°	–	10°
N	.025	.060	0.64	1.52

NOTE:
Metal Lid connected to -Vcc internally.

CASE: Ceramic
MATING CONNECTOR: 0245MC
WEIGHT: 8.4 grams (0.3oz.

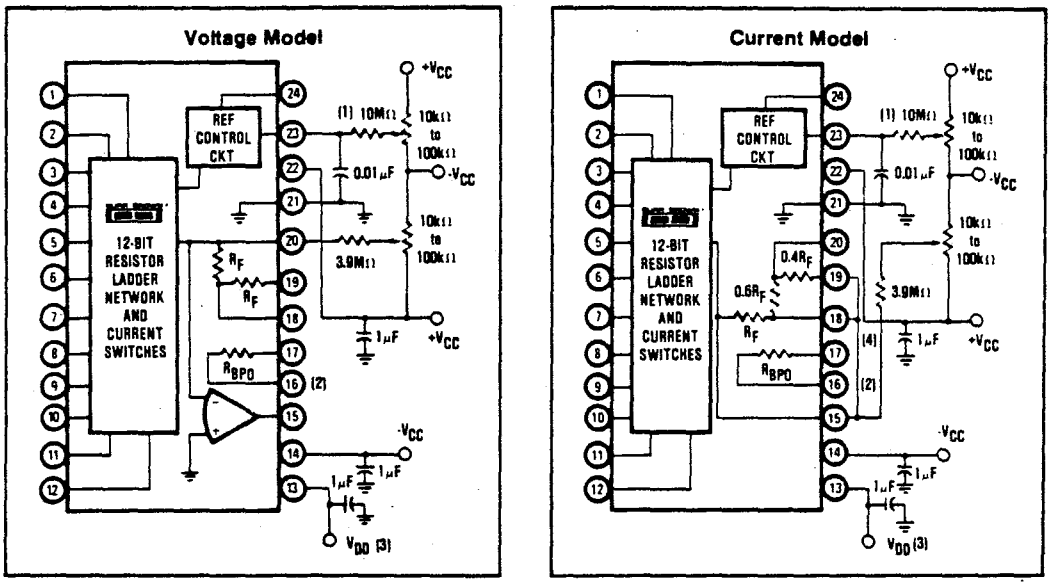
PIN ASSIGNMENTS

I MODELS	PIN NO.	V MODELS
MSB BIT 1	1	BIT 1 MSB
BIT 2	2	BIT 2
BIT 3	3	BIT 3
BIT 4	4	BIT 4
BIT 5	5	BIT 5
BIT 6	6	BIT 6
BIT 7	7	BIT 7
BIT 8	8	BIT 8
BIT 9	9	BIT 9
BIT 10	10	BIT 10
BIT 11	11	BIT 11
LSB BIT 12	12	BIT 12 LSB
LOGIC SUPPLY, V_{DD}	13	LOGIC SUPPLY, V_{DD}
-Vcc	14	-Vcc
I_{OUT}	15	V_{OUT}
REF. INPUT	16	REF. INPUT
BIPOLAR OFFSET	17	BIPOLAR OFFSET
SCALING NETWORK	18	10V RANGE
SCALING NETWORK	19	20V RANGE
SCALING NETWORK	20	SUMMING JUNCTION
COMMON	21	COMMON
+Vcc	22	+Vcc
GAIN ADJUST	23	GAIN ADJUST
6.3V REF. OUT	24	6.3V REF. OUT

NOTES:

1. Refer to Logic Input Compatibility section.
2. Adjustable to zero with external trim potentiometer.
3. FSR means "Full Scale Range" and is 20V for ±10V range, 10V for ±5V range, etc.
4. To maintain drift spec internal feedback resistors must be used - current output models.
5. Includes the effects of gain, offset and linearity drift. Gain and offset errors are adjusted to zero at +25°C.
6. Power dissipation is an additional 100mW, max. when V_{DD} is operated at +15V.

CONNECTION DIAGRAMS



NOTES:

1. DAC80 which may be replaced by DAC800 requires a 33MΩ resistor. DAC800 requires a 10MΩ resistor. DAC80's may also be operated with a 10MΩ resistor resulting in increased trim range.
2. Pin 16 of DAC800 is used only to connect the bipolar offset resistor. An external reference voltage may not be used with DAC800 as is possible with DAC80.
3. If connected to +VCC, which is permissible, power dissipation increases 75mW typ, 100mW max.
4. For fastest settling time connect pins 19, 18, and 15 together.

DISCUSSION OF SPECIFICATIONS

DIGITAL INPUT CODES

The DAC800 accepts complementary binary digital input codes. The CBI model may be connected by the user for any one of three complementary codes; CSB, CTC, or COB.

TABLE I. Digital Input Codes.

DIGITAL INPUT	ANALOG OUTPUT		
MSB LSB	CSB Compl. Straight Binary	COB Compl. Offset Binary	CTC* Compl. Two's Compl.
000000000000	+Full Scale	+Full Scale	-1LSB
011111111111	+1/2 Full Scale	Zero	-Full Scale
100000000000	1/2 Full Scale -1LSB	-1LSB	+Full Scale
111111111111	Zero	-Full Scale	Zero

*Invert the MSB of the COB code with an external inverter to obtain CTC code.

ACCURACY

Linearity of a D/A converter is the true measure of its performance. The linearity error of the DAC800 is specified over its entire temperature range. This means that the analog output will not vary by more than ±1/2LSB, maximum, from an ideal straight line drawn between the end points (inputs all "1"s and all "0"s) over the specified temperature range of 0°C to +70°C.

Differential linearity error of a D/A converter is the deviation from an ideal 1LSB voltage change from one adjacent output state to the next. A differential linearity error specification of ±1/2LSB means that the output voltage step sizes can range from 1/2LSB to 3/2LSB when the input changes from one adjacent input state to the next.

Monotonicity over a 0°C to +70°C range is guaranteed in the DAC800 to insure that the analog output will increase or remain the same for increasing input digital codes.

DRIFT

Gain Drift is a measure of the change in the full scale range output over temperature expressed in parts per million per °C (ppm/°C). Gain Drift is established by: 1) testing the end point differences for each DAC800 model at 0°C, +25°C and +70°C; 2) calculating the gain change with respect to the +25°C value and; 3) dividing by the temperature change. This figure is expressed in ppm/°C.

Offset Drift is a measure of the change in output with all "1"s on the inputs over the specified temperature range. The Offset is measured at 0°C, +25°C and +70°C. The maximum change in Offset is referenced to the Offset at +25°C and is divided by the temperature change. This drift is expressed in parts per million of full scale range per °C (ppm of FSR/°C).