LINEARISATION OF ANALOGUE TO DIGITAL AND DIGITAL TO ANALOGUE CONVERTERS

ALAN CHRISTOPHER DENT, B.Sc, AMIEE, MIEEE

A thesis submitted for the degree of

Doctor of Philosophy, to the Faculty of Science,

at the University of Edinburgh, May 1990.

DECLARATION OF ORIGINALITY

I hereby declare that this thesis and the work reported herein was composed and originated entirely by myself in the Department of Electrical Engineering, at the University of Edinburgh, between October 1986 and May 1990.

ABSTRACT

Monolithic high resolution Analogue to Digital and Digital to Analogue Converters (ADC's and DAC's), cannot currently be manufactured with as much accuracy as is desirable, due to the limitations of the various fabrication technologies. The tolerance errors introduced in this way cause the converter transfer functions to be nonlinear. This nonlinearity can be quantified in static tests measuring integral nonlinearity (INL) and differential nonlinearity (DNL). In the dynamic testing of the converters, the transfer function nonlinearity is manifested as harmonic distortion, and intermodulation products. In general, regardless of the conversion technique being used, the effects of the transfer function nonlinearities get worse as converter resolution and speed increase. The result is that these nonlinearities can cause severe problems for the system designer who needs to accurately convert an input signal.

A review is made of the performance of modern converters, and the existing methods of eliminating the nonlinearity, and of some of the schemes which have been proposed more recently. A new method is presented whereby code density testing techniques are exploited so that a sufficiently detailed characterisation of the converter can be made. This enables a digital mapping of the converter transfer function to be derived, which improves the overall linearity of the device.

This is particularly true for the INL error, which is generally reduced, at the expense of the DNL error which is slightly increased. However, the dynamic effects of the nonlinearity are almost always completely erradicated, as can be seen from harmonic analysis of the mapped converter.

The number of signal samples which have to be taken in order to form the distribution histograms required by the code density tests, and used by the mapping algorithms, is much reduced from that needed for accurate characterisation of the converter's transfer function, using standard code density testing.

In any hardware implementation of the system, the digital mapping would be stored in memory, as a look-up table, and techniques are developed which greatly reduce the total amount of storage needed. This involves segmenting the device's transfer function into several smaller pieces, which are approximately linear.

To validate the linearisation technique it was necessary to build a hardware implementation of the system and the mapping algorithm. This allowed real ADC nonlinearities to be used to test the algorithm, rather than just those simulated.

The linearisation technique was also modified so that it could be applied to digital to analogue converters as well. This entailed modification of the linearisation hardware structure and the mapping algorithm.

ACKNOWLEDGEMENTS

I would like to thank all the members of the Signal Processing Group, past and present, for all of their stimulating questions and ideas, which have helped immensely over the years. Special thanks go to my close friends for not letting me get too involved in my work, and to my Father for making sure that I did. Extra special thanks to Tennent Caledonian Breweries, the Bristo Bar and all its barstaff for sustaining me in my hours of need.

LIST OF ABBREVIATIONS

ADC - Analogue to Digital Converter

ADCL - Analogue to Digital Converter Linearisation

CMOS - Complementary Metal Oxide Semiconductor

DAC - Digital to Analogue Converter

DDS - Digital subscriber loop Development System

DFT - Discrete Fourier Transform

DNL - Differential Nonlinearity

DSP - Digital Signal Processing

EPROM - Eraseable, Programmable, Read Only Memory

FFT - Fast Fourier Transform

HF - High Frequency

IC - Integrated Circuit

INL - Integral Nonlinearity

ISDN - Integrated Services Digital Network

LECS - all digital Linearity Error Correction by Code Shift

LSB - Least Significant Bit

NCC - Network Communications Controller

PCB - Printed Circuit Board

pdf - Probability Distribution Function

RAM - Random Access Memory

ROM - Read Only Memory

SAR - Successive Approximation Register

TTM - Threshold Tracking Mapping

VDU - Video Display Unit

Contents

Chapter 1

INTRODUCTION

The growing importance of digital signal processing techniques, and the ready availability of high speed digital computers has led to a need for higher resolution analogue to digital and digital to analogue converters (ADC's and DAC's). One of the major difficulties this has posed for the semiconductor manufacturers is that, as processing and lithographic accuracies are approaching their physical limits, it is very difficult to increase the resolution of the converters, whilst maintaining acceptable linearity. For this reason, several techniques have emerged in recent years which allow the converter's transfer function to be linearised, after its manufacture, rather than at the fabrication, or design stage [1-22].

The purpose of this work was to investigate a new technique for the linearisation of these converters, which was based upon the mapping of the device's original transfer function, to another which exhibits superior linearity. This mapping was to be performed in the digital domain, so that the input digital codes could be transformed into another set of more linear digital codes. Obviously this entails performing the on-line processing of the mapping of the transfer function after conversion from the analogue domain has taken place.

In order that such a mapping technique could be performed, it would be necessary to have some method by which the original transfer function of the converter could be characterised. Several techniques exist to perform this function, one being code density testing [23-26], where an input signal is sampled a large number of times

and a histogram constructed from the digital output of the converter. Analysis of this histogram then gives a characterisation of the device's transfer function.

After the converter transfer function has been determined, some algorithmic method is required to calculate the necessary mapping. This algorithm must use the description of the converter's nonlinearity as its input.

In order that such a system for converter linearisation could be investigated, several requirements have to be met. Initially at least, it would be most convenient to use ADC simulations and to experiment with the linearisation system in that way. This implies that ADC and DAC simulation software is needed, in addition to a method of introducing specific nonlinearities into the system. In order to maintain the generality of the technique and the results, it was decided to adopt a black box approach to simulating the converters and their transfer function nonlinearity. This was structured in such a way that it was possible to simulate any desired resolution of converter, with any one of a large number of different nonlinearities.

The simulations catered for the code density testing of the converters as well as performing the operation of the mapping algorithm. It was decided at an early stage that the conventional measures of converter transfer function linearity (integral and differential nonlinearity) were inappropriate to measure the performance changes initiated by the application of the mapping function. A new measurement was thus developed to assess the difference between the actual transfer function and the straight line ideal, over the full scale range of the converter. This differs from conventional linearity measures in that it is an average rather than simply a maximum value of the transfer function distortion.

A more conventional method for assessing converter performance is to examine the spectrum of a sampled signal, and in particular the harmonic responses. Linear transfer functions produce little or no harmonic distortion, whilst nonlinearity leads to large harmonic and intermodulation components.

Once a functioning system had been constructed, and the basic technique had been proved, there remained the question of whether the system could be implemented in hardware. In a real system, the digital mapping would be stored in some form of look-up table, which requires relatively simple hardware. The only practical difficulty is that the size of the look-up table required for high resolution converters becomes prohibitively large.

To reduce the amount of memory needed to implement the system, requires that the quality of the transfer function mapping is reduced, as less information will be stored. By dividing the transfer function up into smaller segments, and only storing the mapping for those points, a large reduction in the size of the look-up table can be envisaged. The problem then is how to obtain optimum performance from this reduced resolution system.

The linearisation system has been designed and tested purely by simulation. To ensure that the linearisation techniques and the simulation results were valid for real converters and their transfer functions, it would be necessary to use real converters and associated hardware. This could be used to duplicate the results obtained from the simulations, proving that the technique was feasible and that the assumptions made in the simulations were realistic.

The hardware that this required included an ADC with adjustable nonlinearity, so that the linearisation techniques could be tested with a wide range of converter nonlinearities, rather than the handful which would be feasible using separate devices for each test. Some means for implementing the mapping algorithm was also needed, as well as for collecting and analysing the data. It would also be necessary to implement the mapping look-up table in hardware. As the whole system needed flexible control, the easiest solution would seem to be to interface the system to a dedicated computer, which would also be able to facilitate analysis of the results.

After the study of ADC's was complete it would be advantageous to attempt to apply similar techniques to the related nonlinearity problems encountered in digital

to analogue converters. This time however, the transfer function mapping would have to occur before conversion to the analogue domain.

Again this implies that the linearisation structure of the system would have to be modified, but perhaps the largest difference in approach would be in the characterisation of the DAC's nonlinearity. Code density testing could not be applied in a conventional sense as the finite number of input signals which can be generated (digital codes) do not allow a normal probability distribution function to be constructed. As before, it would be necessary to alter the form of the mapping algorithm so that it could produce the required output, from the information available. Also of importance for DAC's is the problem of implementability i.e. is it possible to reduce the memory requirement of the system, by similar techniques to those used for ADC's.

1.1. Thesis Outline

This thesis is divided up into several segments which cover the areas of work outlined above, in the following way. Chapter 2 discusses the background to the problem of converter linearity, and surveys the state of the art in ADC's and in the means to linearise them. Chapter 3 introduces and analyses the use of transfer function mapping, and the threshold tracking algorithm in particular, to improve the linearity of an ADC. Chapter 4 is concerned with reducing the amount of memory required to implement the mapping look-up table, as this is a serious restriction of the applicability of the technique. Chapter 5 describes the hardware implementation which was used to validate the simulations which had been the only experimental technique used until that point. Chapter 6 shows how the basic mapping technique can be adapted so that it can be applied to D/A converters as well. The final chapter draws some conclusions from the work, and discusses its likely applicability, before briefly introducing some areas of further work.

Chapter 2

STATE OF THE ART ADC LINEARITY

There are currently a great many different types of analogue to digital and digital to analogue converters both on the market and at various stages of development. These can be broadly classified by their resolution, speed, conversion technique and linearity.

In the area of telecommunications and ISDN, 12 bits resolution is the current accepted norm [27,28], whereas for radar applications, between 8 and 10 bits resolution are used [29], but 12 or more would be preferred for improved performance [30]. Many applications require much higher resolution than 12 bits, especially in the fields of instrumentation [31] and HF radio [32,33] where up to 22 bits resolution and linearity may be used or required. As yet of course, very few devices posses this level of performance. Digital audio and medical imaging are areas where upwards of 16 bits are required to obtain the maximum performance of the systems [1,34].

These and other application areas, as well as needing differing resolutions, also have different conversion speed requirements. Instrumentation generally does not place a great restriction on device bandwidth, whereas ISDN ADC applications need to be able to convert at higher rates. The areas where conversion speed is of the essence are radar, high data rate communication, and other occasions where the signal of interest has a very large bandwidth. It is the degree of linearity exhibited by

conversion devices in the various classes of interest, which this review will concentrate on.

Many methods of improving device linearity have been proposed and implemented. These can range from the more classical techniques to advanced autocalibration schemes, and these will be discussed and appraised in some detail.

2.1. Common ADC Types

Although there are many different types of ADC the most popular conversion methods are currently successive approximation, oversampling (sigma-delta), and flash or half-flash [35,36]. Multislope integrating converters remain predominant in digital voltmeter and other low bandwidth measurement devices [37-39], but as these systems are of no real relevance to this study, they will not be discussed further.

2.1.1. Successive Approximation ADC's

Successive approximation converters work in an algorithmic fashion and as their name suggests make an iterative estimate of the final output of the device. As can be seen from figure 2.1 which shows a block diagram of this type of converter, in which the main analogue components are the comparator and the reference DAC. It is the performance and more specifically the linearity, slew rate and stability of these components which determine the accuracy, speed and most importantly, the overall linearity of the converter. By assuming that the converters are well designed, and hence stable, and that they are being operated within their maximum rated sampling rates, then the major problem remaining is the converter's linearity. This is mainly limited by the performance of the reference structures, and specifically the linearity of the DAC in the feedback loop of the converter.

In successive approximation it is the Successive Approximation Register (SAR) which holds the current estimate of the digital value to be output. The SAR is

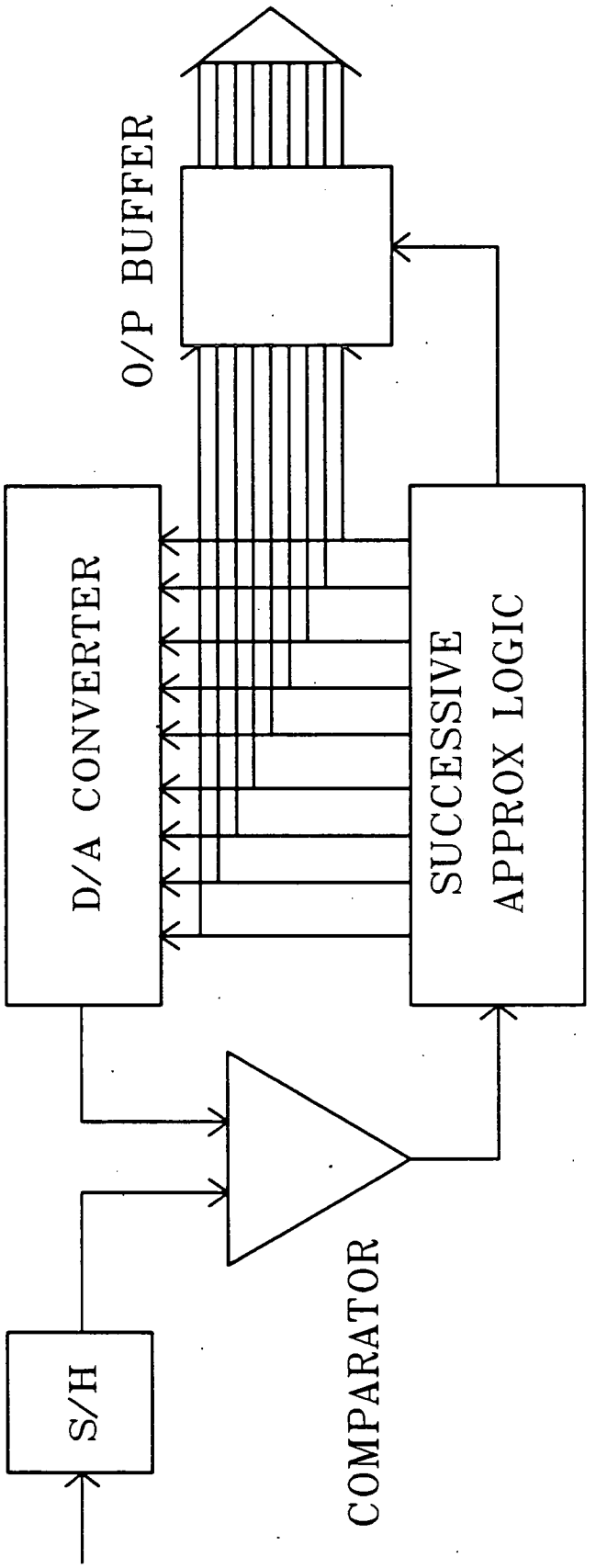

Figure 2.1: Block diagram of a Successive Approximation converter.

initialised to half full scale, i.e. the most significant bit of the register is set whilst all of the other bits are held to zero. The contents of the SAR are converted to an analogue signal by the reference DAC and the output signal compared with the analogue input. If the sampled signal is greater than the present output of the reference DAC then the last bit to be set in the SAR is retained, otherwise it is cleared. This process is then repeated for each successive bit in the SAR, until the least significant bit has been decided upon.

Current successive approximation designs cover a wide spectrum of speed, resolution and power requirements. Twelve bit resolution converters are available with near 12 bit linearity and a wide range of speeds and powers, with many of the designs implemented on single CMOS chips along with their own clock signal generators, sample and holds, input multiplexers, microprocessor control lines, variable data output formats and circuitry, as well as input signal multiplexers [40-45].

The larger, more complex designs of 16 bit resolution successive approximation converters are more difficult to manufacture to maximum linearity, as the physical limitations imposed by the fabrication processes become more troublesome. For example, CMOS converters using ratioed capacitors in their reference DAC's have the problem that it is currently not possible to fabricate the ratios to an accuracy much better than 0.1% [46].

For this and other reasons, such as larger circuit area and circuit complexity it is less common to find 16 bit (or greater) successive approximation converters with full 16 bit linearity. Certainly there are many 16 bit devices on the market which only approach full linearity, with perhaps only 14 or 15 bits linearity [47,48]. These larger resolution devices also tend to have fewer extra features such as the on chip sample and holds and clock generators now common on twelve bit successive approximation ADC's, as well as generally reduced maximum throughput rates, and of course a higher price [49,50].

2.1.2. Oversampling Converters

There is a lot of misunderstanding and confusion surrounding oversampling converters. Put in the simplest way possible, single stage oversampling or sigma-delta type Analogue to Digital Converters work in a way analogous to Delta Modulators [51,52]. That is, they compare the current level of the input to a quantised version of the integrated previous inputs. This allows a binary decision to be made as to the output of the device at that particular sampling instant. The sequential data from the quantiser is fed to a decimation filter which produces a parallel output from a whole sequence of serial data. It is the rate at which data exits the decimation filter which determines the bandwidth of the sigma-delta converter, not the primary sampling rate of the quantiser. With reference to figure 2.2 it is easy to see that the linearity of such a converter is determined by the accuracy of the integrator in being able to add or subtract exactly one least significant bit (LSB) from the running total. In more complex designs, where several integrators are used, this problem is exacerbated.

It would seem to be desirable to use a multibit quantiser, to speed up the conversion process [53]. This however, puts a constraint upon the loop DAC that its linearity must be equal to the final resolution of the converter [54]. For this reason, single bit DAC's which can generally be made to have an extremely high degree of linearity are used, although problems may still occur if the DAC is unbalanced [55].

However, even though oversampling converters are thought to be virtually linear does not mean that they are the perfect choice for all applications. Due to their oversampling nature they are not able to provide converters with the very highest rates of sampled data throughput and they are usually not of the highest resolution. Oversampling converters, however, are compact, simple and hence a good design for fabrication on silicon. Problems do exist pertaining to the stability of the more complex, multistage oversampling converters where the feedback paths can become difficult to analyse [56,57].

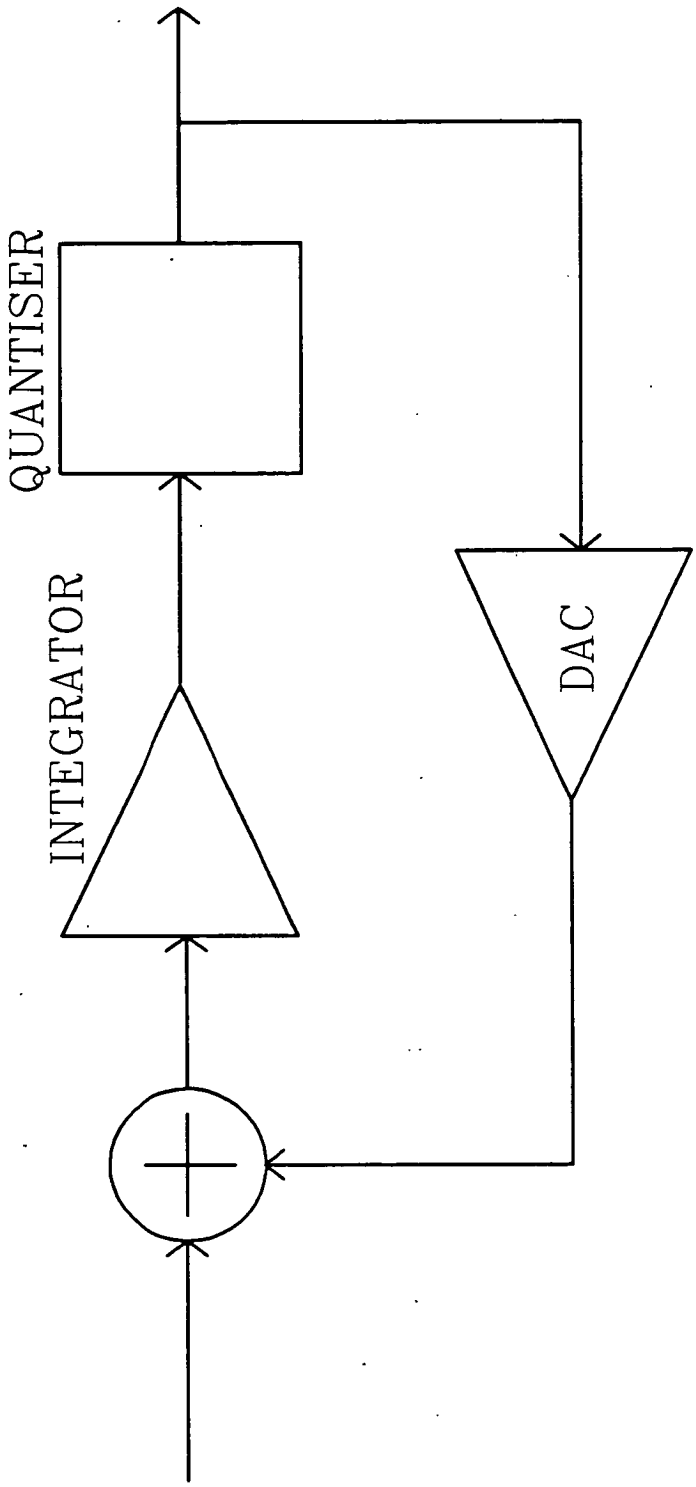

Figure 2.2: Block diagram of an oversampling Sigma-Delta converter.

Many good designs exist today for sigma-delta type converters which are able to meet the specification required for applications in digital telecommunications equipment [58,59]. The current state of the art for oversampling converters can be thought to be the multistage (3 or greater) sigma-delta devices which can give a resolution of greater than 15 bits at bandwidths approaching 1 MHz [60-62].

Clearly the sampling rates of these devices can not be pushed much further with silicon technology, and as yet Gallium Arsenide integrated circuitry is not at a stage where it is able to implement the necessary decimation filtering for converters which sample the signal in the Giga-hertz region [63,64].

2.1.3. Flash and Half-flash Converters

Flash ADC's are so called because conversion is performed very quickly i.e. in only a single clock cycle, for a full flash converter. This is achieved by a high degree of parallelism in the design of the converter. In most other designs of ADC, the input signal is compared to a single reference signal internal to the device. In a full flash converter the input is compared simultaneously with all of the possible quantised analogue values of the signal. Hence for the 3 bit, 8 level full flash converter shown in figure 2.3, 7 comparisons of the input signal have to be made at the same time. Clearly for an 8 bit ADC, 255 comparators will be needed (2^8-1), and 4095 for a 12 bit ADC, etc. This is obviously becoming a more difficult design and fabrication problem as the resolution of the full flash converter is increased. Not only do more comparators have to be fitted onto the die but a larger output decoder as well. Also the timing problems associated with presenting the input signal to all of the comparators simultaneously, and the increased capacitance of the longer signal distribution lines, both go to make the converter design harder.

For these reasons a compromise converter type has been developed, where the parallelism of the full flash device is used twice, in a circulatory manner. This type of converter is termed a half flash converter and a block diagram of it is given in

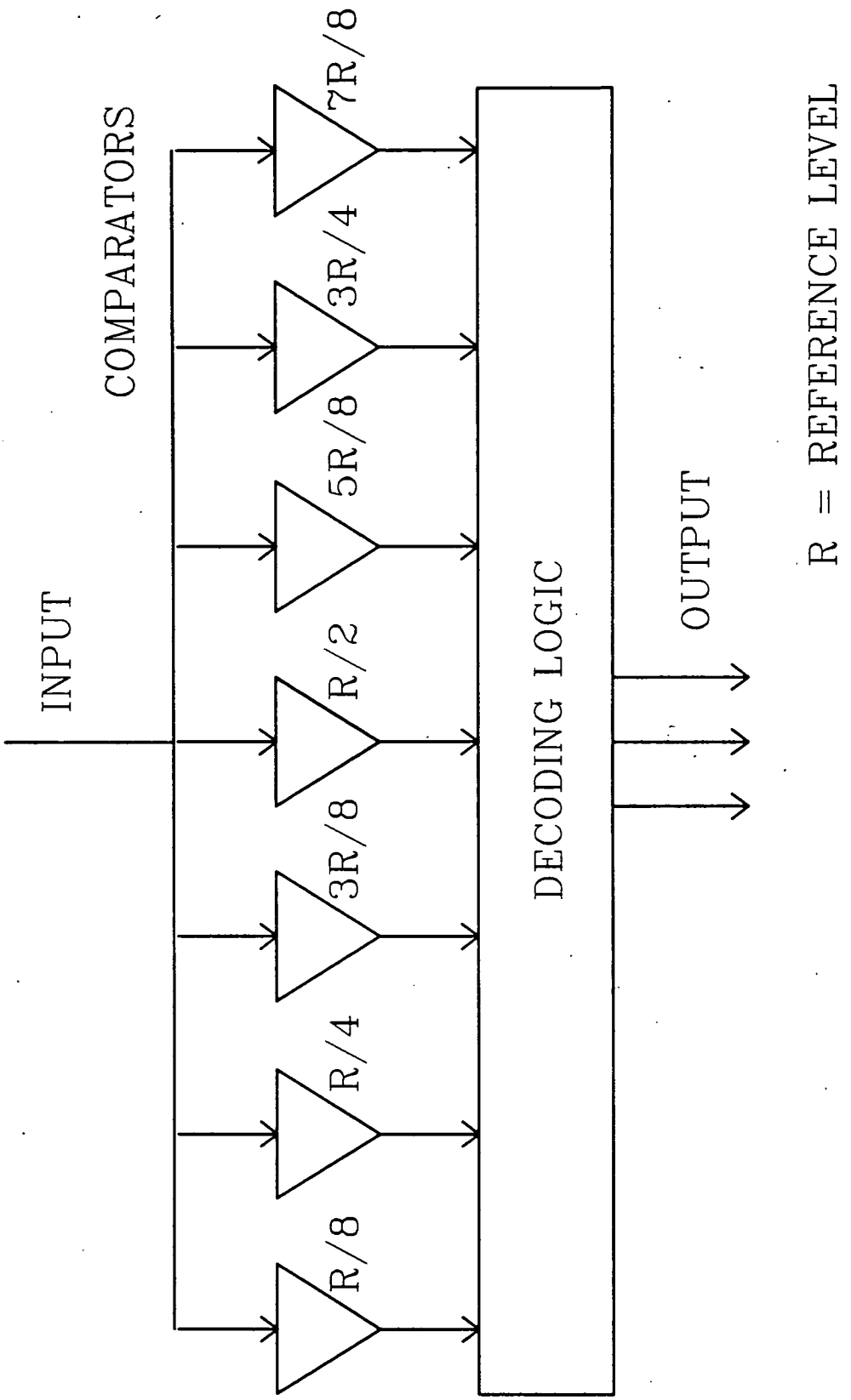

Figure 2.3: Block diagram of a Flash converter.

figure 2.4. The input signal is quantised into the appropriate number of bits by the array of comparators. The analogue equivalent of the digital output of the first stage is then subtracted from the input signal and the remainder is multiplied (amplified) by a factor determined by the resolution of the comparator array. The first digital output is stored temporarily as it represents the more significant bits of the final output of the converter. The amplified analogue remainder of the first pass is then quantised again by the comparators, to generate the less significant bits of the ADC's output.

The drawbacks of the half flash converter are that it now takes two or more clock cycles to generate an output signal, and that additional analogue and digital control circuitry is needed. The advantages are that the number of comparators required for a certain resolution of conversion is greatly reduced (i.e. for a 16 bit conversion, with two passes of 8 bits each, only 255 comparators are needed instead of 65535) or, conversely that a resolution of twice the original conversion can be obtained from the same number of devices.

Linearity of Flash Converters

Whilst both types of flash converters are very fast, allowing high data bandwidths, their linearity is generally not as good as slower converter types. This is caused by the fact that a flash device, because of the large number of comparators needed in its design, requires an equal number of precision reference sources, as is clear from figure 2.3. Intuitively, the easiest way to solve this problem would be to use an impedance chain from the positive reference voltage to the corresponding negative reference, with taps for each comparator. Immediately the problem with this approach, in terms of converter linearity, is obvious. The resistor or other impedance string can not be made accurately enough to provide the linearity required, especially over such a long span as is needed for the large number of comparators involved.

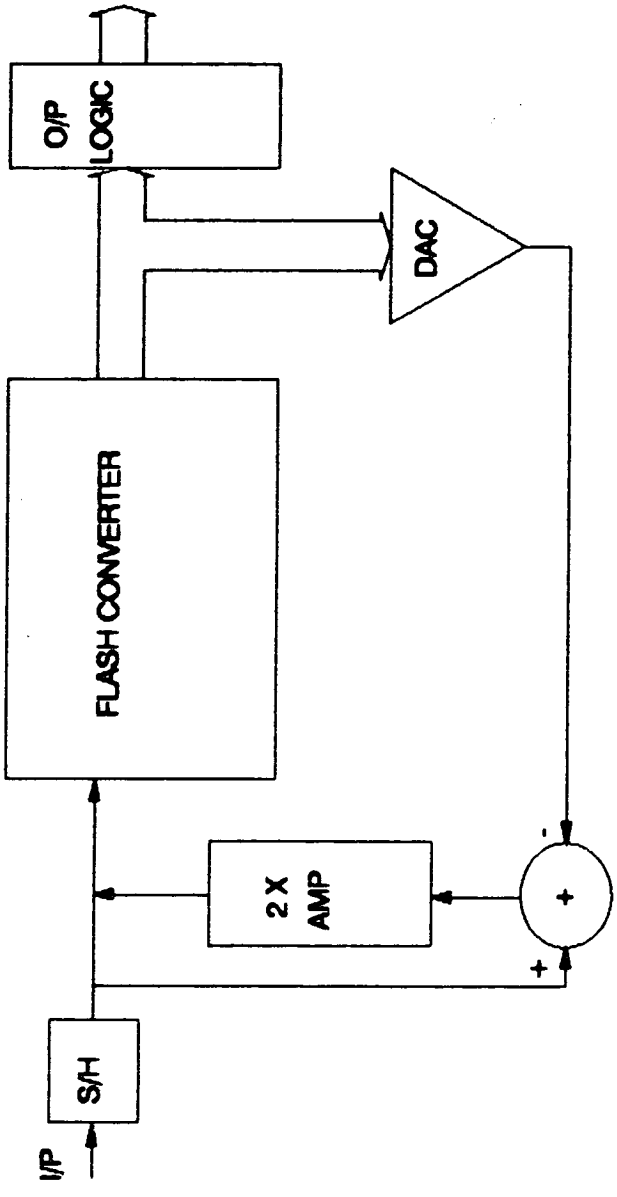

Figure 2.4: Block diagram of a half-flash converter

This problem also applies to half flash architectures, with the added disadvantage that any nonlinearity in the referencing of the comparators used for the first pass of conversion will distort the residual signal. This in turn implies that the effect of the nonlinearity will be magnified by the amplification involved in scaling the remnant signal for the second conversion pass.

This brings to light another source of nonlinearity in half flash ADC's, which is the accuracy of the scaling of the first pass residual. Any slight error here will cause distortion of the final output of the converter which could be of the order of several LSB, effectively reducing the linear resolution of the device.

Half flash devices suffer an extra source of nonlinearity, namely the DAC used in the feedback loop which calculates the residual signal. Any nonlinearity in this area will, of course, cause further distortion of the final output of the ADC, in much the same way as it did for successive approximation devices.

Due to the extremely high speed at which the flash conversion takes place, dynamic errors are very important. These may be caused by differences in the analogue characteristics of the fast comparators, which may be spread over a wide area of silicon, due to their sheer number. Another possible cause of dynamic nonlinearity is timing errors or glitches, due to the difficulties of trying to distribute timing and control signals as well as data, keeping them unaffected by parasitics and time delays from physically long signal paths.

Realisable Flash Converters

Silicon flash converters of 6 to 8 bits resolution are becoming more and more common, with most manufacturers offering products [65-68]. It is even possible to find 10 bit devices [69,70] but it is the speed of conversion upon which the major emphasis is put as the linearity of these devices is often very poor, regardless of their resolution. Here too, as with oversampling ADC's if they are to operate much faster than is currently possible, then a move to fabrication in Gallium Arsenide is

called for.

There are already several designs either in production or at the research stage [71-75]. The resolution of these devices is of course comparatively low when compared to their silicon cousins, with 6 bits being the maximum achieved to date. The linearity of these converters, or converter building blocks, is very poor, as can be expected for such an immature technology. The conversion speeds attainable however, are extremely high with claims for 3 GHz now being made. Clearly half flash architectures will be utilised to extend the resolution of these devices, potentially on the same substrate, when the circuit complexities possible become high enough [76].

2.1.4. Algorithmic Converters

Algorithmic or recirculating ADC's, like successive approximation devices, use a highly iterative technique to achieve conversion. They could in fact, be considered to having been derived from both the half flash and the successive approximation converters. They are analogous to the half flash technique in that after each pass at the input signal, a scaled residual is formed which is re-presented to the converter. However, in this case the decisions on whether each bit of the output should be set or not is decided upon one bit at a time, as in the successive approximation technique.

The main components of an algorithmic converter scheme are shown in figure 2.5. The analogue input signal, captured by the sample and hold and compared to a value representing the quantisation level of the most significant bit of the device's output. If the input is greater than the reference (half full scale) then the appropriate bit in the output register is set and the reference is subtracted from the original input signal. This residual signal is then multiplied by two before being re-presented for comparison with the same reference voltage as before. The reference is now equivalent to the quantisation level of the next most significant bit

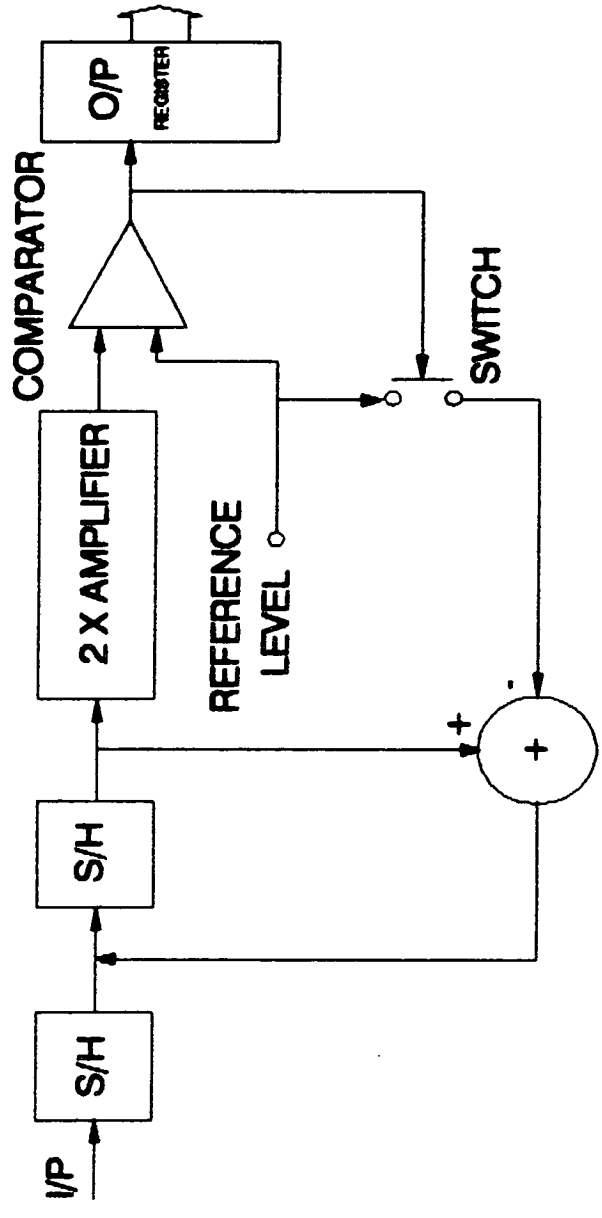

Figure 2.5: Block diagram of an algorithmic converter

of the digital output, due to the multiplication of the residual by a factor of two. This whole process is repeated as many times as is required to obtain the desired resolution of conversion.

This system of analogue to digital conversion seems fairly complex in terms of its control, and indeed, this is why it has not seen wide use until the recent advent of mixed bipolar and CMOS technologies which has allowed the integration of high quality analogue and digital logic functions on the same substrate [77,78]. The advantage which the system enjoys is that it enables conversion with only a single analogue reference voltage, rather than a whole set, whether they be produced by a DAC or an impedance string. Also there is very little analogue circuitry involved in the converter as the digital circuitry is used to great advantage.

Obviously the major source of nonlinearity in such a scheme is bound to be the multiplication by exactly two. Any small distortion of this figure will introduce an error, though initially small in magnitude, which will be multiplied by approximately two, every time round the loop. This amounts to a magnification of the original error by a factor of roughly 2^{n-1}, where n is the bit resolution of the converter.

Manufacturers are now starting to produce devices using the algorithmic conversion technique, and research is being conducted into their behaviour. The resolution and linearity of the devices are tied closely together, as they are both decided by the accuracy of the two times analogue multiplier. Already devices with greater than 12 bits resolution, with linearity of ± 0.5 LSB have been produced [79,80] and slower circuits which are able to have their resolution altered, to suit each individual data conversion are also appearing [81,82].

2.2. Common Techniques of Converter Linearisation

By far the most common technique currently in use for the linearisation of ADC's is laser trimming, where the analogue components, generally constructed using thin

film technology, are altered in size and shape by the use of a laser. Hence any imperfections in the manufacturing process which have led to the introduction of device nonlinearity, can be eradicated. The obvious way in which this technique could be most easily implemented, i.e. obviating the need for test and characterisation of the converter, would be to ensure that all the referencing components were too large. They could then be trimmed to their original intended sizes. So successful has the established technique of laser trimming proved to be that devices are often advertised as being factory trimmed to a certain linearity.

Another technique which linearises the analogue reference components in a converter by a once only process which alters the values of the components is known as ROM trimming. It does not rely on the highly accurate use of lasers to trim components, but similarly the linearisation process is carried out at the same stage of the packaging process of the manufactured device. The reference components are fabricated, not as a single area of silicon, but as several interconnected pieces. The electrical connections between the circuit segments are made by a series of switches of the type used for programmable ROM's. The reference component values are tested and then the appropriate connections/switches are blown so the distributed components are connected in the appropriate manner. The linearity resolution which this technique can effect is determined by the number and size of the alternative pieces of component which can be switched into or out of the circuit. Usually a binary weighting is applied to the spare components, to give a large number of easily calculated trimming values which can be used to alter the reference components [2].

Both of the above techniques could be applied to any reference components and hence to any type of converter. This can not generally be said to be true of the other more modern, signal processing techniques, which are about to be described.

2.3. Modern Linearisation Techniques

The advent of advanced DSP applications, which has prompted the requirement for high resolution analogue to digital converters with good linearity, has also provided some of the solutions. These techniques, many of which had existed only as rough ideas for many years, have been made possible by the availability of silicon fabrication technology which allows the mixing of high density digital logic with accurate analogue components. Although the accuracy limits of the popular fabrication technologies is only sufficient for 10 bit devices, many circuit design techniques have been applied to enable higher resolutions than this to be achieved. Some of these solutions involve totally new conversion methods, or innovative circuit designs, all of which are of no general applicability to other converters. In this review, therefore, circuit design solutions, or novel conversion techniques will not be considered. Instead those designs which use some form of linearity correction will be surveyed.

Up until a few years ago, there was little published work in the area of ADC's with adjustable linearity correction, whilst now there are many designs which can be purchased at low prices, compared to the same specification of component linearised by conventional means. This is in part caused by the high cost and low reliability of laser trimmed devices.

2.3.1. Smart Successive Approximation

Most of the linearisation techniques, on which papers have been published, use the popular successive approximation conversion method. The major cause of nonlinearity in successive approximation converters is the DAC which is used in the feedback part of the system. Nonlinearity in the DAC is directly mirrored by the ADC, and hence most of the linearisation schemes seek to ensure that the DAC nonlinearity is either eliminated or compensated for. Also, successive approximation converters are already digitally controlled and hence lend themselves well to further

digital processing needs.

In the smart successive approximation scheme [3], a normal converter is allowed to operate far faster than it normally would. This means that the comparator and the reference DAC are not allowed enough time to settle to the accuracy required for the resolution of conversion being made. This inevitably leads to errors in the bit decisions made by the converter, which are then corrected.

The first few (most significant) bits of the conversion are allowed to settle in the time required for that number of bits accuracy, i.e. far more quickly than the settling time required for accuracy. After the initial values of these first few bits have been derived, then the correction routine is started. The converter is allowed to settle to the full resolution of the device, at which point the value of the DAC output is compared to that of the input voltage. If the DAC output is too high then the short binary word (made up of the first few bits) is decremented by one. If the output is lower than the input signal, the short conversion word is incremented. This correction cycle is repeated in a slightly modified form, at the end of which the least significant bits are successively approximated in the normal manner.

The use of this technique enabled a speed up of the converter by a factor of two, whilst the error correction ensured that its linearity was unimpaired.

2.3.2. Self Calibration

Another linearity correction technique which is employed in a successive approximation scheme is self calibration [4-7] in which a capacitor weighted DAC is used for the reference. As the device is fabricated by CMOS technology the accuracy of the binary ratioed capacitors is only sufficient for 8-10 bits linearity, whereas the converter is of 15 bits resolution. The converter goes through a calibration sequence where each of the capacitors is evaluated and a correction term (if necessary) is found for it. This is achieved by first charging up the capacitor being tested to the reference voltage. Due to the binary weighting scheme, the sum

of the capacitance of all of the smaller capacitors, plus one LSB, must be equal to that of the capacitor under test. These capacitors are hence also charged up to the reference voltage, and any charge difference between the two sets must represent the error in the value of the bit capacitor being calibrated. This charge error is digitised and the digital correction value is stored in RAM.

When this process has been repeated for all of the weighted reference capacitors then calibration is complete. When the ADC is used to sample real data, every time that a bit is selected by the successive approximation algorithm, the charge error associated with it is reconstructed from the memory by a low resolution subsidiary DAC and is added onto a correction capacitor associated with the main DAC. At the end of a conversion cycle the correction capacitor will have a charge on it which will be the sum of all the corrections for the main DAC bits which have been selected by the successive approximation algorithm.

This technique has been extended by others [8] to give even better performance, hence ensuring that the converter is linear to at least 15 bits, without any need for external trimming.

2.3.3. Smart Analogue

Another method of converter linearisation has been developed which corrects the capacitor weighting errors in a charge redistribution ADC as described above [9]. In this scheme however, the digital memory is used to set up a matrix of switches which are used to adjust the capacitance of the reference DAC capacitors. These are made up of several smaller capacitors, which are switched in and out of the circuit as determined by the requirement to match the total capacitance of the less significant capacitors in the reference DAC, plus the LSB dummy capacitor.

This calibration scheme has enabled the production of 16 bit converters, with full linearity, operating at audio frequencies and above. A full calibration cycle will last over fifty thousand clock periods, but can also be performed in an interleaved mode

with normal conversions. The ADC can therefore be continuously relinearising itself, hence compensating for any drift processes.

2.3.4. Redundant Coding

One solution to the problem of manufacturing a suitably linear reference DAC for a successive approximation ADC is to use code redundancy [10]. This means using a reference DAC which has more bits of resolution than the ADC itself. Hence most of the DAC codes will not be utilised. If a 17 bit DAC were used for a 14 bit converter, then there would be 8 codes (on average) within one converter LSB of each ideal quantisation level. Hence any nonlinearity in the reference DAC can be calibrated out by selecting the appropriate 17 bit DAC code to represent the ideal 14 bit code.

There is no necessity for the 17 bit DAC to be linear and in fact it does not have to use a normal reference structure at all. Hence instead of a R-2R ladder, a R-1.85R ladder is used, which results in the redundant DAC codes covering the analogue output range without missing codes.

The calibration process is quite simple in that an accurately generated reference voltage, corresponding to the ideal level of a linear 14 bit DAC, with only one bit set, is sampled by the ADC. The device successively approximates until a 17 bit value is found. This is then stored in an on-chip EPROM, so that during normal conversion, when a bit is being tested by the successive approximation algorithm, the nonlinear 17 bit equivalent can be read from the memory and fed to the 17 bit DAC. The device also uses the error correcting smart successive approximation technique described earlier. The 17 bit result is finally rounded to the accurate 14 bits and is output.

2.3.5. Combined Self Calibration and Redundancy

Recently a synthesis of the techniques of self calibration for charge redistribution

type successive approximation converters and redundant DAC coding, has been developed [11]. The device uses an 18 bit internal DAC with a radix of 1.85, to provide an eventual resolution and linearity of 16 bits. A problem with non binary radix redundancy coding is that it is difficult to fabricate the reference ladder with much accuracy.

For binary coded charge redistribution converters the sum of the charge on all the smaller capacitors was equal to that on the bit under calibration, plus any error. This is not the case with a radix 1.85 ladder, but there is however a constant ratio between the two sets of charges. Clearly if this ratio can be found then the linearisation problem can be solved. Unfortunately due to the ladder inaccuracies the ratio is neither constant, nor is its value known for any particular bit. The values of the charge ratio can be measured under calibration with accurately known test inputs, but the values obtained will not be very precise. Using an iterative technique the ratio values can be more accurately evaluated and the converter can hence be reliably linearised.

The iteration technique has been shown in theory and in practice to be stable and converge rapidly. An important feature of this linearisation method is that it is insensitive to component matching tolerances up to a few percent, and therefore there is no tradeoff between matching and effort as there is for other self calibrating converters.

2.3.6. Algorithmic Converter Self Calibration

A method for the linearisation of algorithmic ADC's has been developed [12-15], in which the nonlinearity introduced by the converter comparator and inaccuracies in the gain of the loop amplifier are reduced so that a 13 bit converter can be realised without the need to trim components. In a similar way to that employed in other schemes, the capacitors which determine the exact value of the gain of the multiply by two amplifier, are actually comprised of many smaller capacitors which can be

switched in or out of the circuit by the self calibration algorithm.

The calibration method is fairly straightforward, and is based on the fact that if the reference voltage is presented to the converter then the output should be full scale i.e. the all ones code. The trim capacitors are adjusted until a change in the trim by the smallest capacitor element causes the all ones code to drop by one LSB. Due to the large number of capacitor circuit elements (64 K) the calibration process takes over eight thousand clock cycles to perform, and even then it is not guaranteed to have linearised the converter. Another reason for this long calibration time is that to eliminate the effects of noise on the input signal, each successive bit decision is made by a majority voting scheme from a poll of seven.

2.3.7. Self Compensation

Self compensation is the name given to a technique developed for the linearisation of DAC's [1], and although it has not been applied directly to ADC's, it is obvious that the methods could readily be employed in a successive approximation converter scheme. The DAC is a 14 bit device, fabricated such that the untrimmed accuracy of the converter is more than 9 bits. Hence only the five most significant bits need to be adjusted for the whole converter transfer function to be linear. Calibration is by reference to an internally generated ramp signal which is linear to around 17 bits. This is produced by a Miller integrator using an external capacitor with very low dielectric absorption and a high gain (100 dB) amplifier.

A counter is used to determine the linearity error at each of the five most significant bit switching points (32 in total), as the ideal value of the counter can readily be compared to its actual value. This counter error can then be stored in RAM and compensated for by a subsidiary DAC which adds to the output of the main DAC.

2.3.8. Correction Using Walsh Analysis

Analysis of ADC transfer characteristics by Walsh functions [16-19] has proved to be a useful technique due to their binary nature and ability to represent complex nonlinearities with only a relatively small number of terms. A correction technique which has been suggested as a cheaper solution than purchasing a better resolution/linearity converter has been developed [20].

The converter first has to be characterised before the Walsh analysis can be performed. This results in estimates of the bit weighting errors in the converter, and some other deviations of the Walsh functions which are not related to single bit switching effects. The number of interacting bits is generally no greater than two, which means that the correction process is considerably simplified.

The linearisation technique is to add a lot of extra analogue circuitry to the reference DAC in the successive approximation converter. Every time that a bit with a calculated bit error is switched on an additional voltage, dropped across an accurately known resistor is added to the DAC output. For the cases in which more than one bit is interfering to cause the nonlinearity, then a more complex decoding scheme is required.

The performance of this system appears to be good although the extra analogue and digital circuitry is different from device to device, as are the accurate compensation resistors.

2.3.9. Correction by Code Shift

A technique for ADC linearisation known as Linearity Error Correction by Code Shift (LECS) [21,22] uses a deliberately nonlinear reference ladder in the DAC in its successive approximation converter. A nominally 16 bit converter, aimed at the digital audio market, uses two R-2R ladder structures in combination, each of which can be more or less guaranteed to be linear to eight bits, when used in isolation. A specific form of nonlinearity is introduced into the 16 bit reference

structure by the addition of an extra resistor between the two smaller ladders. This causes the DAC transfer function to contain what are referred to as negative jumps i.e. at the switching points of the eight more significant bits there are several digital codes which represent the same analogue value, but there can never be a situation where an analogue value cannot be represented by the appropriate digital one.

Accurate calibration values related to the switching points of the more significant bits are converted so that a digital mapping of the DAC can be performed and the results stored in the on-chip memory. An adder is also used to reduce the storage capacity needed by the system. The results obtainable are quite good although there is some loss of dynamic range due to loss of a large number of the digital codes which can be output by the system. These have to be recoded so that no codes are missing. Some of the results given are presented in a very favourable light i.e. LSB's measured relative to a 15.5 bit converter, and the claimed improvement in performance includes the nonlinearity deliberately introduced by the system.

Chapter 3

ADC TRANSFER FUNCTION MAPPING

Whilst the methods outlined in the previous chapter enable correction at source of the causes of transfer function nonlinearity, the techniques presented here try to eliminate its effects. Using digital post conversion processing of the original, unaltered ADC output code, it is possible to have a linearisation system which can work independently of the type of converter to which it is applied.

This excludes sigma-delta type converters, which have quite different transfer function qualities altogether to the majority of converter designs. For analogue to digital conversion techniques where the device's output is unrelated to its previous state, post conversion processing of the digital signal allows a very neat way of linearising already existing ADC devices, as well as incorporation of the technique into the development of new designs.

3.1. Mapping Functions

At the heart of the linearisation system presented here is the concept of transfer function mapping. This involves changing the digital code which has been output by the nonlinear ADC to a different code which represents a more linearly quantised equivalent of the original analogue input signal. By examination of figure 3.1, this concept can be more readily understood.

The figure shows two 7 bit transfer functions, one of an ideal converter which is a

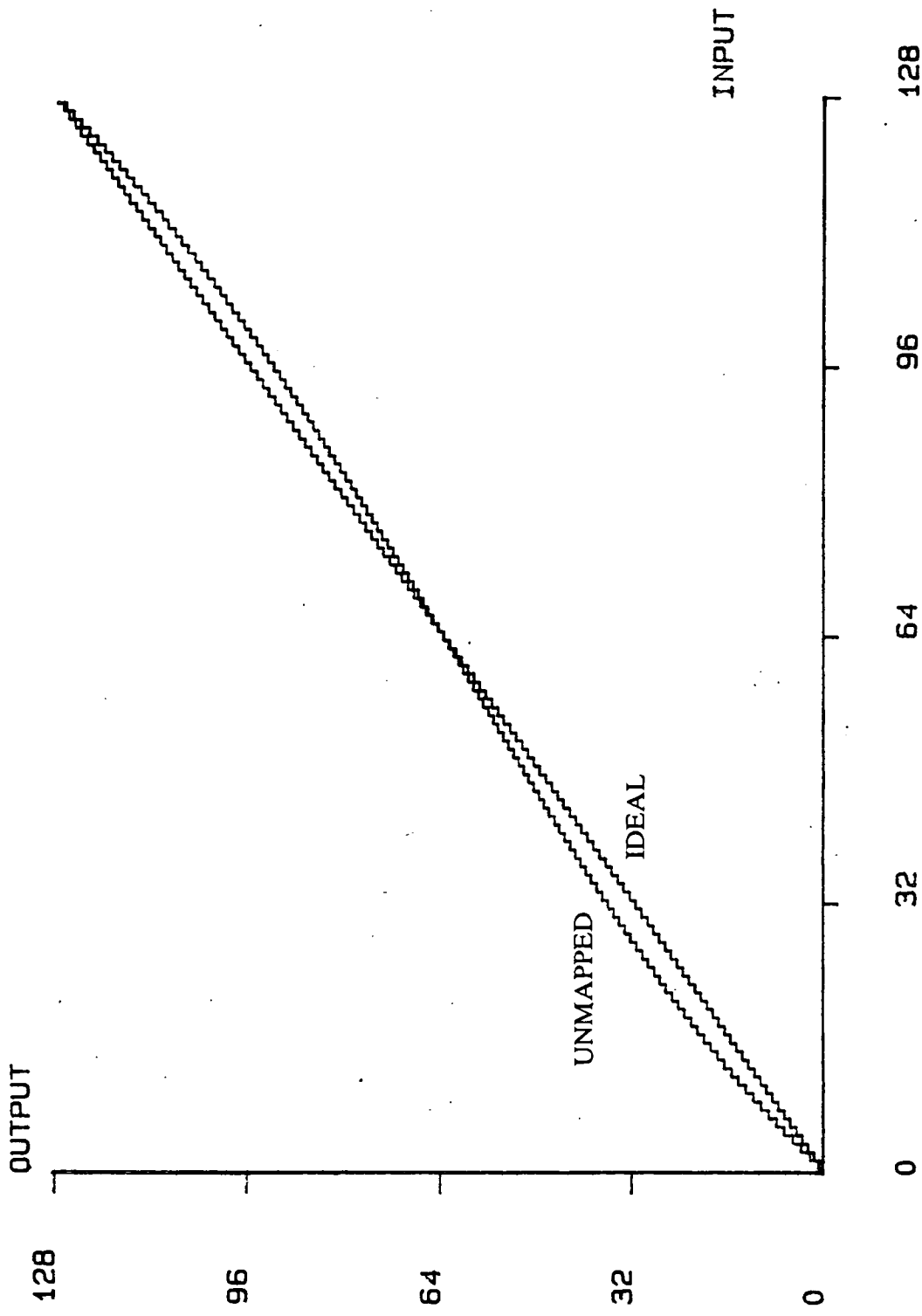

Figure 3.1: Ideal and nonlinear 7 bit ADC staircase transfer functions.

uniform staircase, and the second of a nonlinear converter which is represented by a more irregular staircase function. At the point on the two characteristics where the analogue input has an ideally quantised value of 32, the nonlinear converter will output the code value for round about 37. This, of course, represents an integral nonlinearity of 5 LSB. The mapping therefore, for code 37 would be 32. Similarly for an analogue input equivalent to an ideal ADC output code of 96, the distorted ADC would produce the value 91, hence causing the transfer function mapping for code 91 to be 96. The rest of the digital mapping function for the complete ADC characteristic could be determined in a similar fashion, with the net result looking somewhat like the plot shown in figure 3.2. The mapped transfer function is now very much closer to the ideal staircase (not shown) than the original nonlinear characteristic was.

Differential nonlinearity is not removed by this method as the device's quantisation levels remain at their original levels, and in fact, as can be seen from the figure, DNL is sometimes increased by the mapping process. This is due, in part, to the introduction of missing code errors into the output of the linearisation system, caused by the mapping of consecutive ADC codes to nonconsecutive values.

The next step in the linearisation procedure is the implementation of the digital mapping. The simplest way to do this is to use a look-up table of the contents of the mapping function, stored in memory. This has to be positioned after the ADC in the system architecture, so that the converter's output code can address the mapping directly, with its contents being read out by the rest of the digital signal processing system. This type of layout is shown in figure 3.3, where the block which controls the overall operation of the system, and its associated data paths will be explained later.

So far the mapping procedure has been examined only as a technique for improving the integral nonlinearity of any given ADC. No mention has been made of the fact that only nonlinearities which are well characterised can be mapped. That is, if a

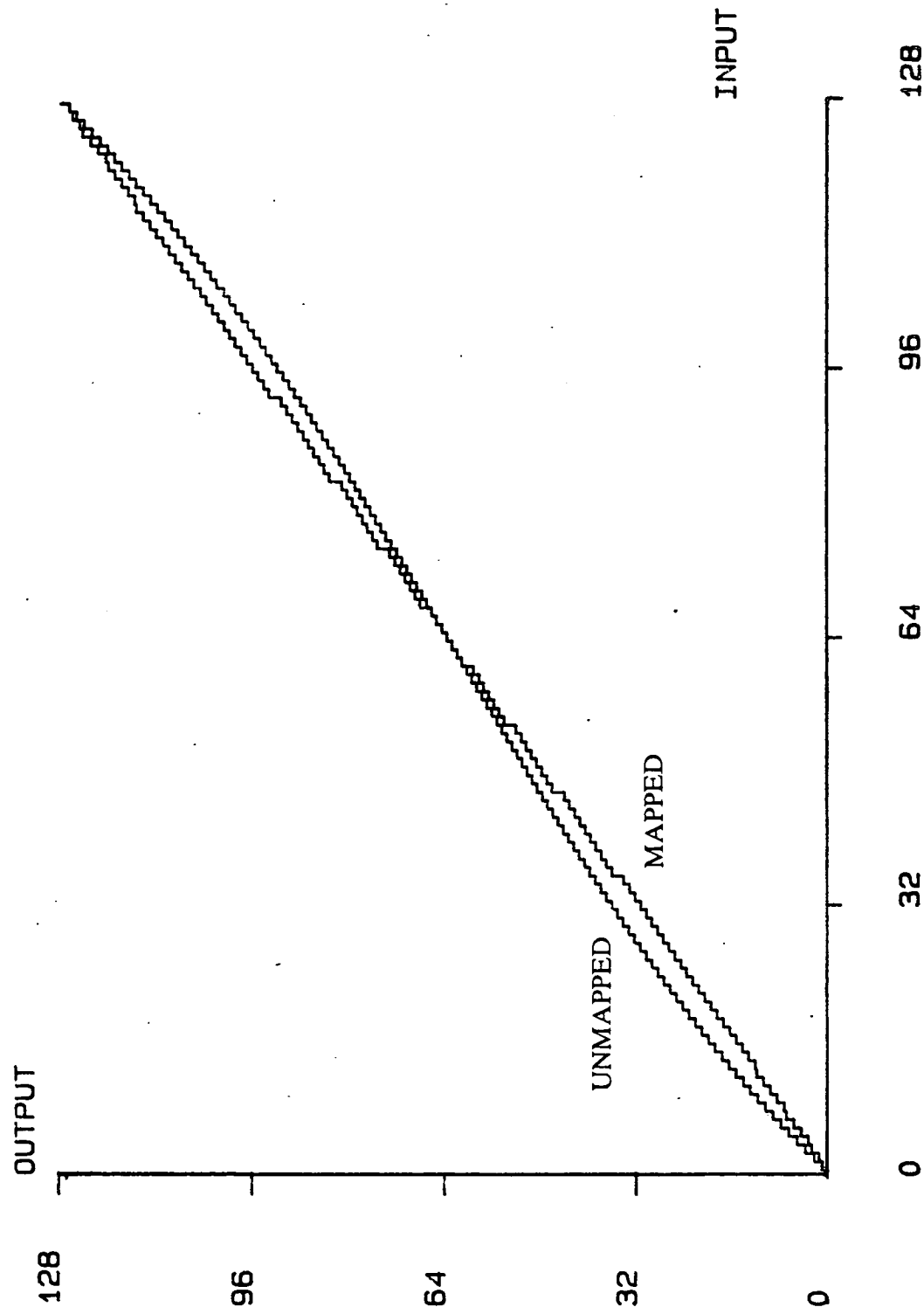

Figure 3.2: Nonlinear and mapped 7 bit ADC transfer functions.

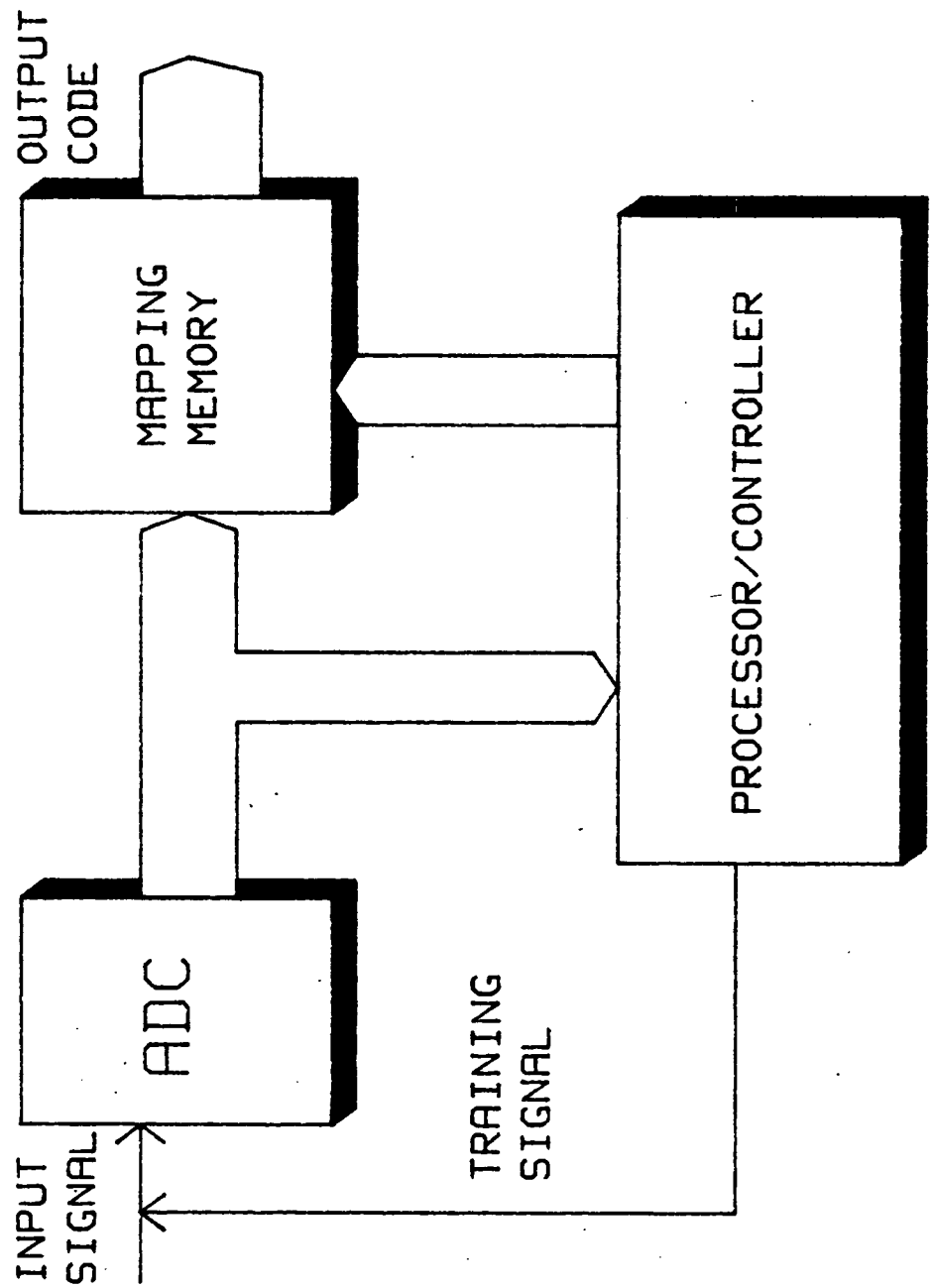

Figure 3.3: Linearisation and mapping system architecture.

particular device's distorted transfer function is unknown, then it is impossible to determine a mapping function for it.

Obviously then, the linearisation technique must incorporate some method of finding out just what the ADC's distorted transfer function is. One way to do this in practice, is to use code density or histogram testing [23-26].

3.2. Code Density Tests

A code density test is a way of dynamically characterising an ADC's transfer function. It is performed by measuring the frequency of occurrence of each and every output code of the ADC when it is sampling an analogue input signal. The converter nonlinearity will cause slightly different codes to be output by the ADC, depending on the form which the transfer function error takes. Histograms can be formed of the measured frequencies and figures 3.4 and 3.5 show two such measured histograms, corresponding to an idealy and nonlinearly sampled random uniformly distributed signal respectively. The change in the shape of the normalised histogram is entirely due to the converter transfer function nonlinearity. These distorted histograms can be used to determine the integral and differential nonlinearities of the converter. This information can only be uncovered if it is known what the contents of the histogram would be if the analogue signal had been sampled by an ideal ADC. In the absence of such a device, it is necessary to know the probability distribution function (pdf) of the analogue signal so that an idealised histogram can be calculated.

The simplest case is for a uniformly distributed analogue signal, such as a linear ramp, or a perfectly filtered pseudo random sequence, where the histogram frequency of an idealy sampled signal is equal for all codes. When such a signal is used, the differential nonlinearity of code i, $DNL(i)$, is given by equation 3.1, where n is the resolution of the converter under test and differential nonlinearity is defined as the error in the size of a quantisation interval, measured relative to the

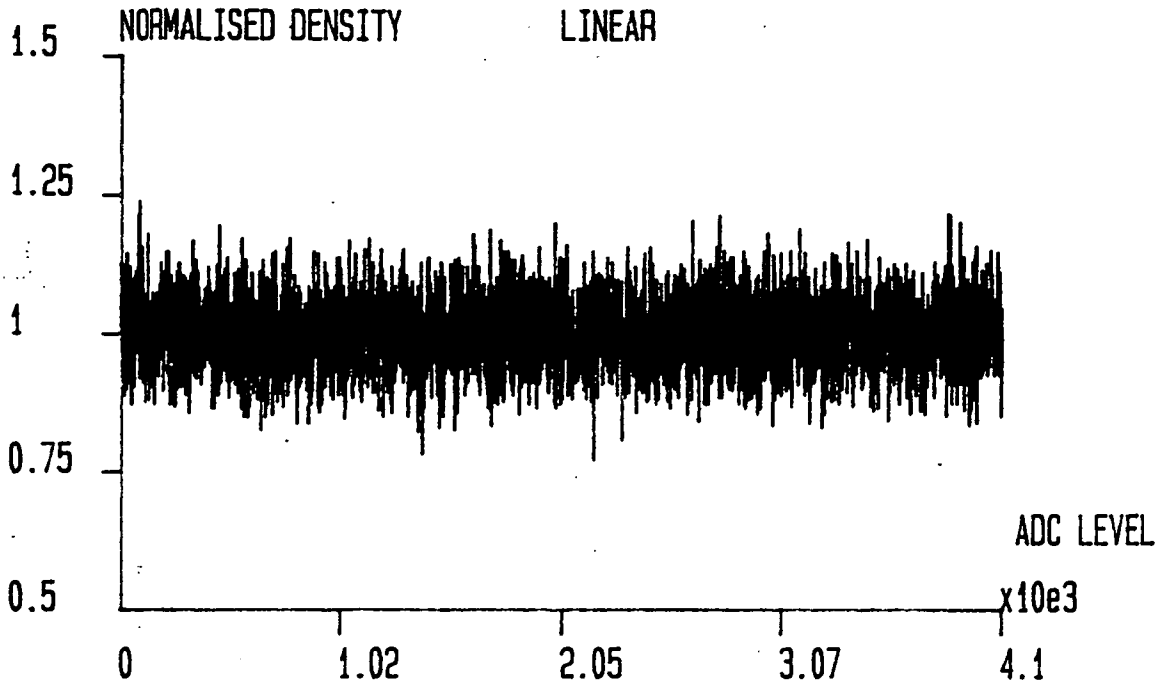

Figure 3.4: The sampled pdf of a random uniformly distributed signal.

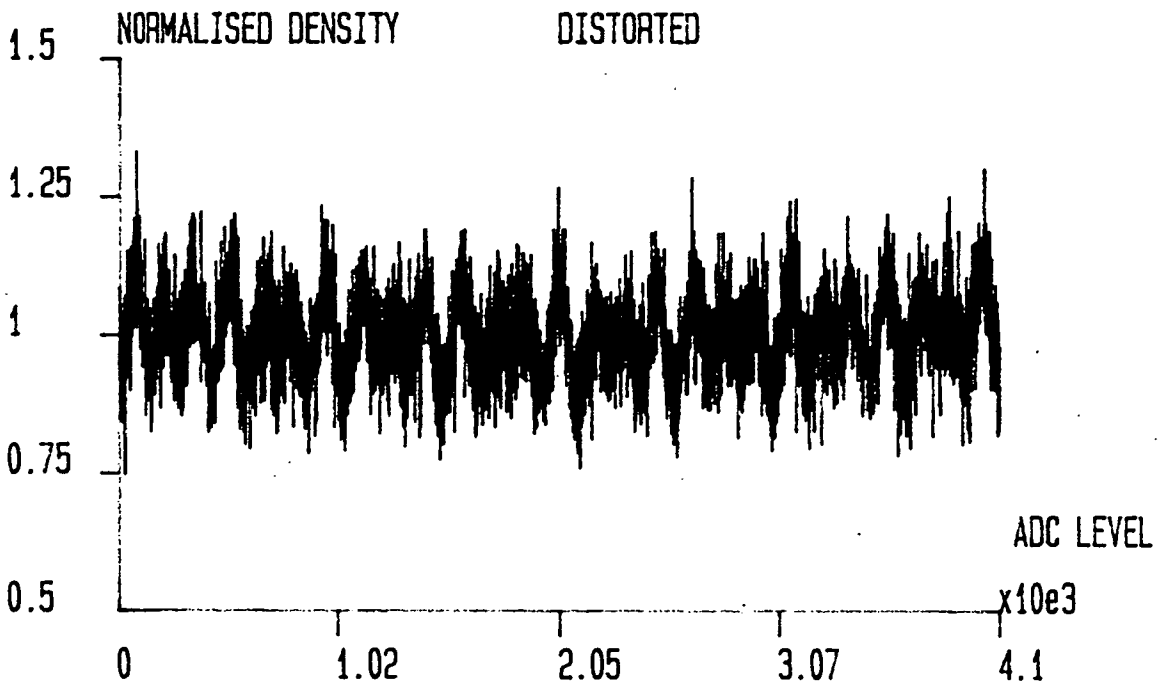

Figure 3.5: The pdf of a uniform signal sampled by a nonlinear ADC.

ideal interval of 1 LSB. $H(i)$ is the contents of the histogram bin i and S is the total number of signal samples.

$$DNL(i) = \frac{H(i)}{S/2^n} - 1 \qquad 3.1$$

Integral nonlinearity, the error in the actual quantisation level of code i compared to its ideal level, measured in multiples of an LSB, can be determined by equation 3.2, assuming that the ADC codes range from 0 to 2^n.

$$INL(i) = \frac{2^n}{S}\sum_{0}^{i} H(i) - (i+1) \qquad 3.2$$

3.2.1. Sinusoidal Test Signals

Uniform pdf test signals are all very well in theory, giving very simple formulae for INL and DNL, but the generation of such a signal, to the necessary degree of linearity, is a fundamental drawback. Even a 1% linearity error on a ramp signal would cause the affected histogram bins to contain 1% too many, or too few, code samples, which would quickly accumulate in the INL calculations, especially considering that high resolution converters have several thousand different codes.

The solution to this problem is to use a sinusoidal analogue test signal. It is well defined mathematically, and it can readily be generated by commercial signal generators with very low harmonic distortion levels. Unfortunately, the pdf of a sine function is not as simple as that for a uniform pdf signal, with the probability of occurrence of a voltage level V, $p(V)$ being defined by equation 3.3, where A is the amplitude of the sinewave.

$$p(V) = \frac{1}{\pi\sqrt{(A^2 - V^2)}} \qquad 3.3$$

When the amplitude of the signal is such that the sinusoid exactly fits the input range of a linear ADC then, the probability that a sample will be quantised in histogram bin i is given by the equation 3.4.

$$p(i) = \frac{1}{\pi}\left[\sin^{-1}\left(\frac{2i - 2^n - 1}{2^n}\right) - \sin^{-1}\left(\frac{2i - 2^n - 3}{2^n}\right)\right] \qquad 3.4$$

It is possible to formulate general expressions for INL and DNL, for any signal pdf, in terms of $H(i)$ and $p(i)$, and these are presented in equations 3.6 and 3.7.

$$DNL(i) = \frac{H(i)}{Sp(i)} - 1 \qquad 3.6$$

$$INL(i) = 2^n \left(\sum_0^i \frac{H(i)}{S} - \sum_0^i p(i) \right) \qquad 3.7$$

Now that we have the ability to determine the integral nonlinearity of an ADC at any point on its transfer function, we have effectively characterised the device.

3.2.2. Quantisation Levels and Pdf Sums

Other items of information can be extracted from pdf histograms such as missing codes (a 0 in the appropriate pdf bin) and systematic bit errors (periodicity in the plotted histogram). But the most interesting quality of pdf histograms, as far as the construction of a transfer function mapping is concerned, is that the cumulative sum of all the histogram bin contents up to and including a particular code, is closely related to the quantisation level of that code. This means that to map that code, what we have to do is to find the code for which the equivalent sum of ideally sampled signals is equal.

If this process were to be repeated for all the codes of an ADC then a complete mapping function would be revealed. This task is trivial for uniformly distributed analogue test signals, and not terribly difficult for sinusoidal signals. Problems arise because it is generally not possible to obtain an exact match between the two pdf sums, which causes the mappings to be inexact. This process of matching up the quantisation thresholds of distorted and linear ADC's, by tracking their respective code density histograms, is the basis for the mapping algorithm referred to as the threshold tracking mapping algorithm, or TTM for short.

3.3. The Threshold Tracking Algorithm

Assuming that the test signal pdf is known prior to the mapping process being performed, then the idealy sampled signal pdf histogram can be constructed, before the actual ADC is tested. This means that the TTM algorithm does not need to concern itself with the physical quantisation levels of either the real device being linearised or the ideal converter, which is assumed to have produced the ideal histogram. The only information required is the contents of the two sampled pdf code density histograms.

The problem of the two pdf sums not matching exactly can be overcome by allowing some leniency in this requirement by the use of a threshold margin, into which the difference between the two pdf sums must fall. This then gives the formulation of the mapping algorithm that is given in equation 3.8, where $L(q)$ is the contents of bin q of the linearly sampled pdf histogram, T is the mapping threshold value and $\Phi(i)$ is the mapping of code i. Both p and q can be thought of as integer variables which in a software implementation of the algorithm would be the indices of the two pdf histogram arrays.

$$i \rightarrow \Phi(i) \text{ when},$$

$$\sum_{q=0}^{q=\Phi(i)} L(q) - TL(\Phi(i)) \leq \sum_{p=0}^{p=i} H(p) \leq \sum_{q=0}^{q=\Phi(i)} L(q) + TL(\Phi(i)) \qquad 3.8$$

From 3.8 it is clear that the effect of the threshold value T is to limit the amount of integral nonlinearity that is to be allowed between the ideal staircase function and the mapped nonlinear function. Clearly the value of T is directly related to the maximum amount of INL, and can be measured in multiples of the LSB.

3.3.1. Nonmonotonicity

For the actual realisation of the threshold tracking algorithm in the simulation software, steps are taken to ensure that no transfer function nonmonotonicity can occur i.e. that code $i+1$ can never map to a value lower than that mapped to by code i. This holds for all of the codes in the ADC transfer function.

Nonmonotonicity is a serious problem for any ADC, as it implies that the quantisation levels of say, code $i+1$ is lower than that of code i. As the TTM algorithm is coded up so that it will not allow this type of error to occur in the mapped function it produces, it can be used as a method of eliminating nonmonotonicity. This sort of transfer function distortion is an impossibility in several types of ADC, especially those that use some form of weighted resistor or capacitor chain as a DAC reference structure.

However, the restriction which the elimination of nonmonotonicity places on the operation of the algorithm means that it can allow large amounts of integral nonlinearity, represented by the difference between pdf sums, to exist in the mapped transfer function. This is probably preferable to any nonmonotonicity.

The implementation of the algorithm also contains some other elements which ensure its correct operation, particularly at the two extreme ends of the transfer function range. These extra elements can be seen in figure 3.6, which shows a block diagram of the computational realisation of the threshold tracking mapping algorithm. The 'C' code equivalent to this flow diagram could easily be written in less than 30 lines, containing only 2 major loops and 3 conditional statements. The routine would have virtually no multiplication and could therefore be programmed easily on even the simplest of DSP systems, with very low computational complexity. A listing of the necessary 'C' code can be found in Appendix A.

3.4. Operation of the System

The linearisation system as a whole has to be considered as having two modes of operation, the training mode and the working mode. The working mode is the final system configuration, such that the mapping memory look-up table and the ADC are the only hardware needed. The analogue input signal is sampled by the ADC and the digital signal now accesses the look-up table to output the mapped code to the rest of the DSP system.

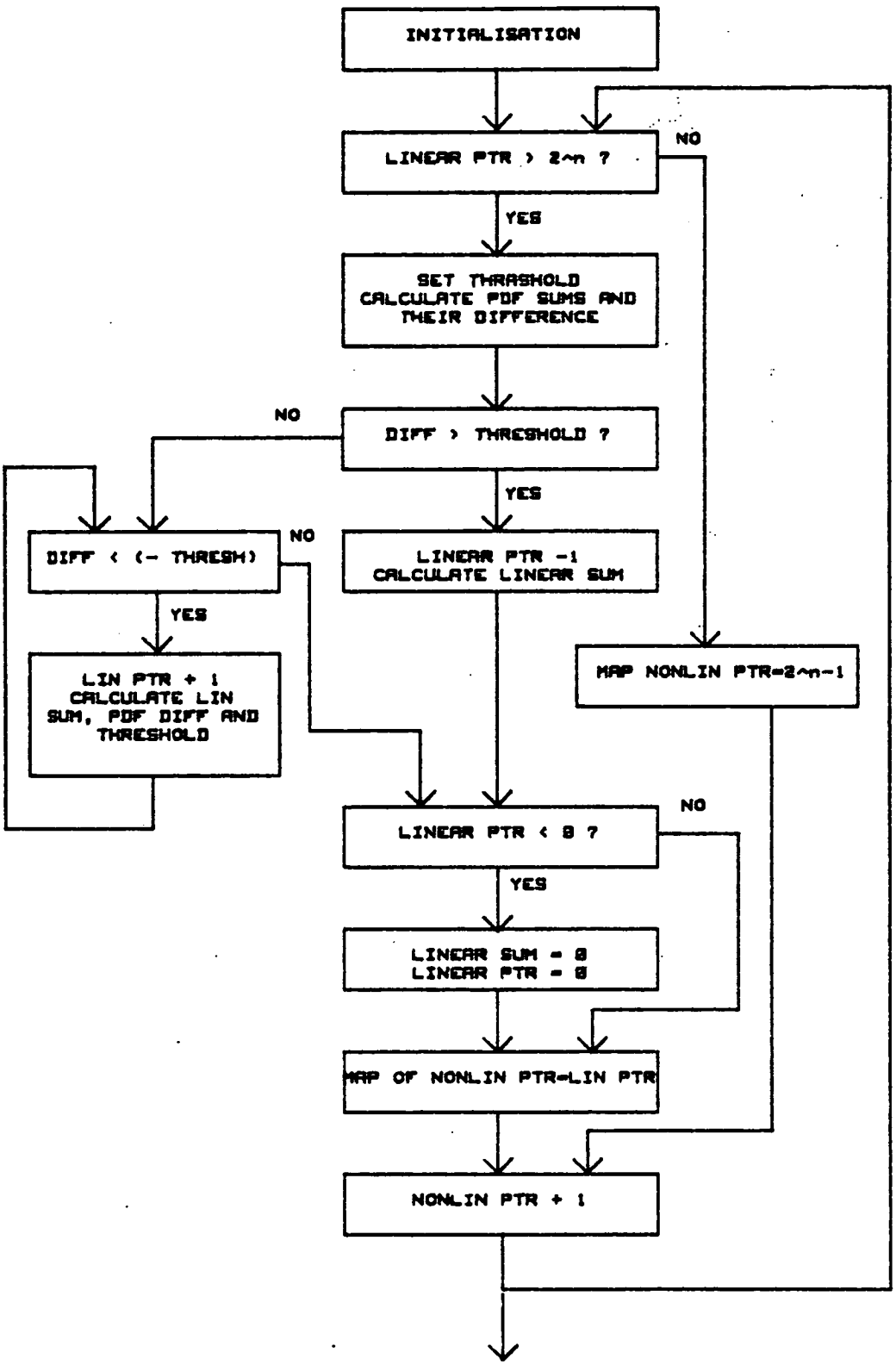

Figure 3.6: Threshold Tracking Mapping Algorithm flow diagram.

The training mode, however, requires all of the previously unexplained components and data paths shown on figure 3.3. The most complicated process involved in the training mode of the system is the generation of a suitable analogue test signal, by the processor/controller block of the system architecture. This signal must have an accurately known pdf or else the assumptions which are made by the TTM algorithm during the mapping process will be in error. Generation of the simplest pdf (uniform) to a good degree of accuracy is not very easy, so the test signal is more likely to be a spectrally pure sinusoid.

Next, the code density histogram of the nonlinearly sampled signal is built up, by sampling the test signal and collating the data in an array of memory in the processor. The total number of samples required is influenced by a number of factors, the principal one being the resolution of the converter. After the TTM algorithm has produced the mapping function, it is written into the look-up table memory, ready to be accessed by the converter in its working mode.

These two modes of operation are implemented by linearisation system simulations, which were able to model converters, nonlinearities, training signals and look-up tables in software.

3.4.1. Simulations

To enable extensive testing of prototype algorithms, ADC simulations were designed to allow a wide range of device distortions to be experimented with, and a number of different performance measures to be taken.

The basic simulation program configuration followed the routine of generating a large number of analogue signal samples, which were scaled to the input range of the simulated ADC. These would then be distorted by predetermined functions, to be discussed presently, which were used to simulate converter transfer function nonlinearity. The original and distorted analogue signals were then sampled by an idealised quantiser which enabled two separate pdf histograms to be built up, and

these were then used by the TTM algorithm to generate a mapping.

The process of events from this point onwards would depend entirely upon the type of measurement that was being made, of which there were several. A graphical representation of the mapping function and the original nonlinearity staircases could be produced, but this was only feasible for low resolution converters (up to 9 bits) due to the sheer number of points to be plotted and their size on the output devices. The integral nonlinearity of the transfer function at every code could be displayed graphically. Periodic signals could be sampled by the nonlinear and mapped converters, so that when the digital codes had been reconstructed (using a perfect DAC), they could be discrete Fourier transformed, so that an analysis of harmonic distortion could be performed. The last type of measurement that was taken was the mean deviation error of the converter.

3.4.2. Mean Deviation

This was devised to get a better feel for any improvement in linearity gained by the mapping process, as the traditional measures, INL and DNL, are only maximum measures taken over the whole transfer function of the device, and INL plots, like transfer function staircases before them, could not be calculated or plotted for anything above 12 bit resolution. Mean deviation is the average, taken over the number of output codes of the converter, of the area between the transfer function in question and the ideal. It is measured in dB, relative to a deviation of 1 LSB at every code. In using mean deviation error, it is not so important what the absolute value for a mapped device is, but rather how much this has been improved by the application of the linearisation process. What tends to happen with mapped converters is that the degree of mean deviation improvement is dependent upon the amount of nonlinearity present in the original transfer function. So that the final mean deviation value, all other variables being constant, for converters of a particular resolution is approximately the same, no matter what distortion function

was used, assuming that good performance has been achieved.

3.4.3. Transfer Function Distortions

In trying to mathematically model transfer function nonlinearity accurately, many problems are encountered. The biggest of these is a lack of knowledge as to the general shape and extent of excursions from the ideal, as well as what the expected levels of INL/DNL might be for real converters. The only guide to this is to be obtained from manufacturers' databooks, which only relate the information pertinent to the specification ranges of their devices, and not the characteristics of those lower grade, less linear converters. This 'problem' is now so acute that it is very difficult to get hold of a 12 - 16 bit ADC with linearity much worse than ± 5 LSB, at least for low conversion rates.

The only other solution, short of modelling several types of converter complete with errors in all of the appropriate analogue components and hence, losing the generality of the mapping technique, was to pick some arbitrary distortion functions which would generate reasonable results. To this end, after examining closely all representations of converter nonlinearities in published papers [83,84] and manufacturers data sheets, distortions based upon cubic functions were chosen. Several basic third order shapes were developed, which were intended to have maximum integral nonlinearities of 16 LSB. Later these were supplemented by the superposition of a few periodic functions on top of the cubics.

When simulations were run to measure the harmonic distortion of mapped and nonlinear converters, the spectra that were obtained were very similar to those which are generally reproduced in manufacturers' databooks, where dynamic testing of their devices has been performed [9]. This seemed to indicate that the choice of simple third order distortions was good enough to make the simulation results credible.

Even so, the threshold tracking algorithm was severely tested by a further set of extremely large and unrealistic nonlinearities, for which it performed admirably, demonstrating the power of the technique, and the relative irrelevance of the choice of transfer function distortion.

3.5. Simulation Results

The results of the simulations are divided up into several sections, depending upon the type of measurements they are showing, rather than for the resolutions of converters, or form of transfer function nonlinearity, used in the simulation. A listing of the software necessary to generate all of these results is given in Appendix A.

3.5.1. Mean Deviation

As discussed previously, the actual value mean deviation error is unimportant, rather it is useful as a measure for determining how a certain set of circumstances has affected it. An example would be to discover whether the use of twice as many training signals is worth the extra sampling time needed to acquire them. The experimental results therefore, which were measured by the amount of mean deviation they allowed from an ideal transfer function, are presented in separate sections, depending upon what the variable quantity was.

Optimal Threshold Value

It was important to know what the best choice for the threshold value was, considering its effects upon the INL mapping boundaries. To keep integral nonlinearity to a minimum, it could be thought that a choice of 0.5 LSB as the mapping threshold would be appropriate, but this may not be correct, especially in cases where large positive differential nonlinearity was present. This would make the INL of the mapped code even larger than in the original, unprocessed transfer function. In this sort of situation the best mapping threshold would be exactly half

of the of the DNL plus 0.5 LSB for the size of the ideal quantisation interval. This would make the INL equal, before and after the mapping. The opposite effects would cause the optimum threshold value to fall in cases of excessive negative differential nonlinearity.

It seems that these two sets of effects tend to cancel each other out, over the full length of the ADC transfer function, as can be seen from figure 3.7. Here the turning points of the plots of mean deviation against the threshold value, are centered around the 0.5 LSB value, for the wide range of converter nonlinearities simulated. The threshold value in the TTM algorithm was then set to 0.5, for all further experiments.

Training Sample Size

It is possible to determine statistical confidence limits, which can be used to calculate the number of training signal samples which must be taken to ensure that the data provided by a code density test is reliable [23]. These numbers tend to be very large indeed, e.g. 4.2 million samples will give a 99% confidence limit on the accuracy of quantisation intervals to 0.1 LSB, for a 12 bit converter. Clearly the required sample size for this level of accuracy will tend to increase rather rapidly with resolution, such that for very high resolutions the necessary sample size would be prohibitively large.

This problem is not so serious for the threshold tracking algorithm, as any inaccuracy in the contents of the pdf histograms will be swamped by the thresholding process, and any errors consistent over a large segment of the transfer function, will be averaged out by the summing process inherent in the algorithm. It is therefore necessary to work out the minimum required sample sizes for different resolutions of converters by experiment, using measurement of the the remnant mean deviation after mapping, with a range of sample sizes.

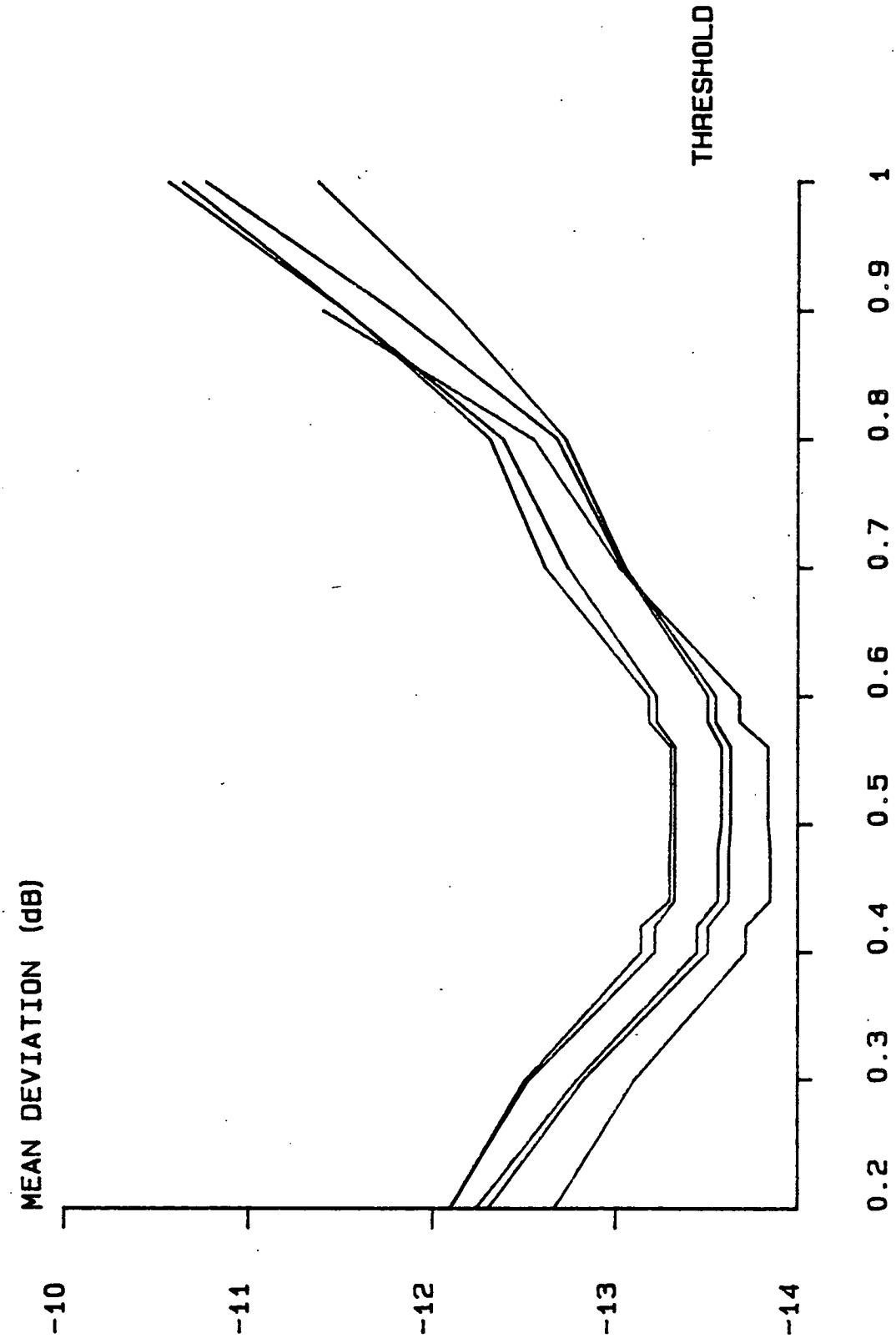

Figure 3.7: The effect of the mapping decision threshold value on performance with several nonlinearities.

Figure 3.8 shows the result of such a simulation with a range of sample sizes (2^6 to 2^{20}) and a selection of different 12 bit converter nonlinearities. The figure shows that very low average code densities of less than 1 sample/code, 2^{12} samples, give poor mean deviation, whereas the highest densities, around 256 samples/code or 2^{20} samples, give the best performance. When the raw mean deviation values are studied it can be seen that the difference between 256 samples/code and 16 samples/code is only a few hundredths of a dB mean deviation error, and there are still only fractions of a dB lost when the density is 4 samples/code.

For the 16 bit converter simulations shown in figure 3.9, the same sort of comments can be made, except that near optimum mean deviation is now achieved above 2^{18} signal samples. This equates to a sample density of 4 samples/code, and as for the 12 bit case we can deduce that the minimum number of samples required to get a reasonable mapping is that amount which will result in a average code density of between 4 and 16 samples/code. This result holds true for all simulated converter resolutions, implying that proportionately more training signal samples are needed to linearise higher resolution ADC's.

Noise Effects

For a hardware implementation of a linearisation system, working with ADC's in a mainly digital environment, noise can cause problems, particularly for analogue signals. To determine what effect, an additive noise signal on the sampled training signal, would have on the linearisation process, considering that the linear pdf histogram is calculated, and not measured inclusive of any noise, uniformly distributed random signals were added to simulated test signals.

Several observations were made from the simulation results, the most basic of these being that as the maximum level of the noise signal was increased, measured in LSB, the mean deviation performance of the linearised ADC went down. This is what would generally be expected but when the number of signal samples in the pdf

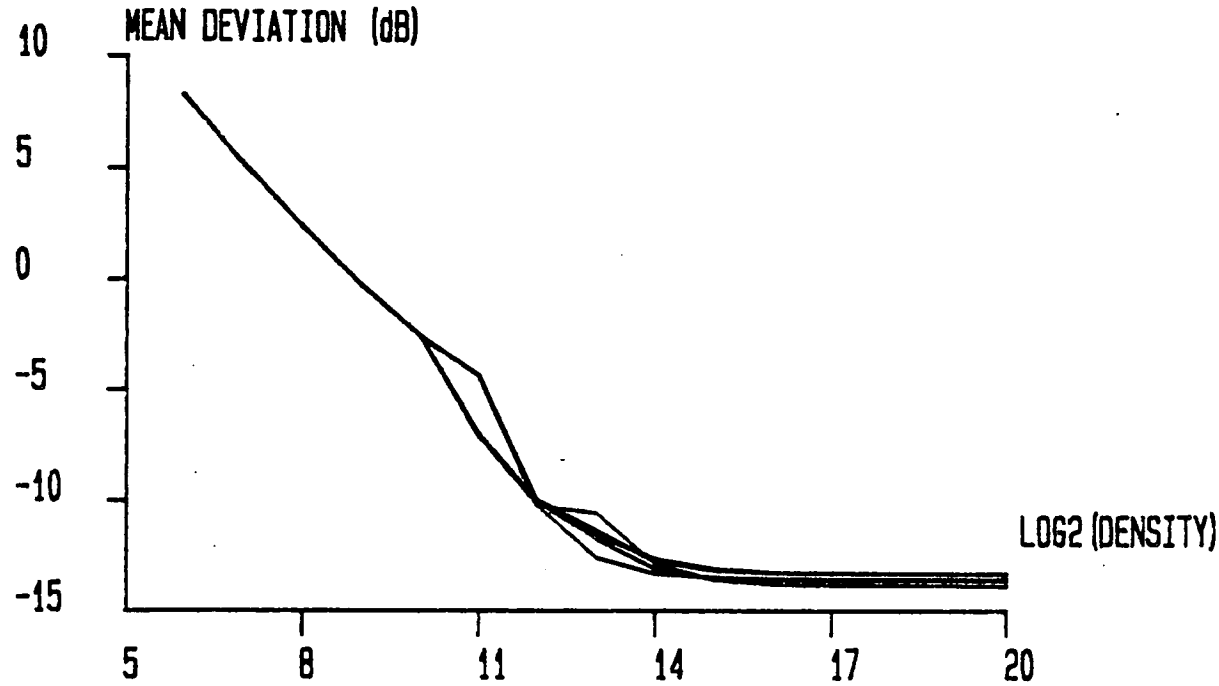

Figure 3.8: Linearisation performance of several distorted 12 bit ADC's with increasing average pdf densities.

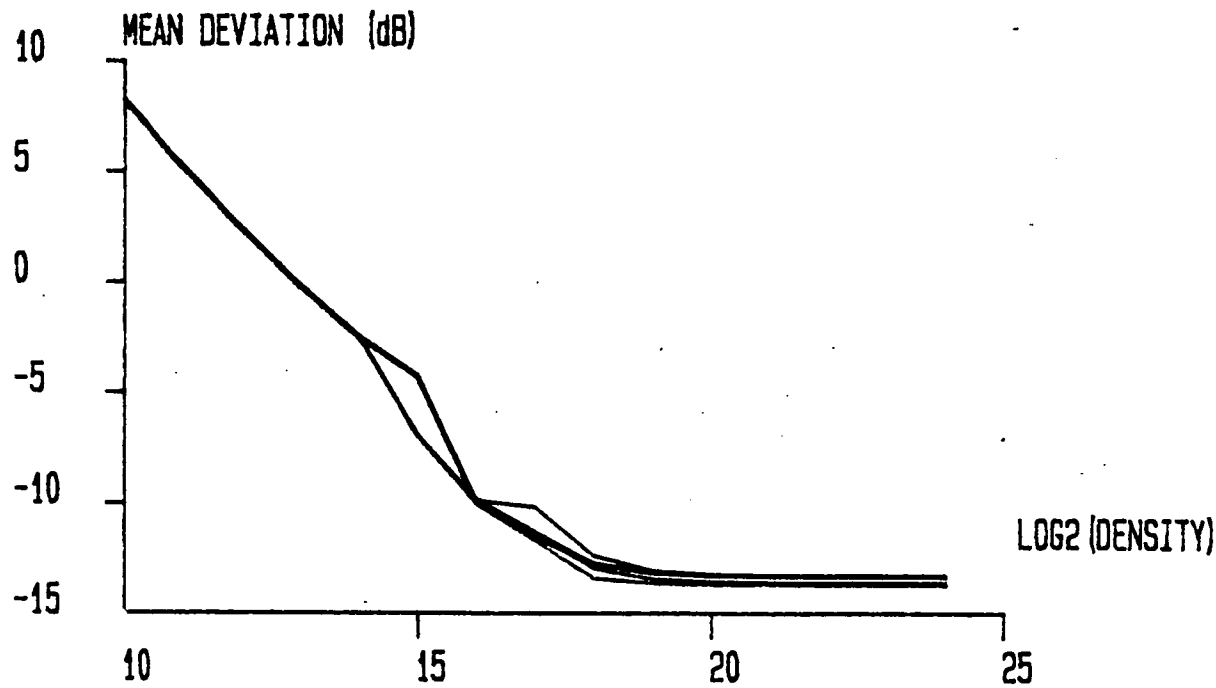

Figure 3.9: Performance and average sample density for 16 bit devices.

histogram was increased, the degradation of the converter's mean deviation did not occur until higher noise levels. Figure 3.10 shows this situation for a 12 bit converter with noise levels of between 1 and 4096 i.e. 2^0 and 2^{12}, LSB's.

When only 8 thousand training signal samples are taken to form the code density histograms, the addition of even small levels of noise is enough to increase the transfer function's mean deviation. When 262 thousand samples are taken however, the noise level has to reach around 16 LSB before any loss of performance is suffered, and for very high densities of 8 million samples, 64 LSB of noise is needed to have any effect at all. For the linearity of the ADC to be made worse that it was before mapping (the 0 dB deviation point), noise levels of around the 256 LSB mark have to be present, which represents 1/16th of the full scale of the 12 bit converter.

The corollary of this is that to enable the linearisation training signal to have higher noise immunity, more samples of it have to be taken. This is true for most reasonable levels of noise, but for very high levels, it is not, as is clear from figure 3.10. The increase in noise immunity gained by raising the sample size by a factor of 2^5 is much less on the second occasion (from 262 thousand to 8 million samples). This implies that some sort of hard limit for noise immunity is reached beyond which further increases in the sample size will have no effect.

This is probably caused by the very large noise signals, representing significant proportions of the converter's transfer function, causing a large percentage of all the signal samples to be quantised at the two extremities of the ADC's input range. This represents a gross distortion of the sampled pdf, with which the threshold tracking algorithm is not able to cope.

3.5.2. Transfer Functions

For low resolution converters, plots of the mapped ADC transfer function staircases are convenient and useful to enable the visualisation of what the mapping procedure

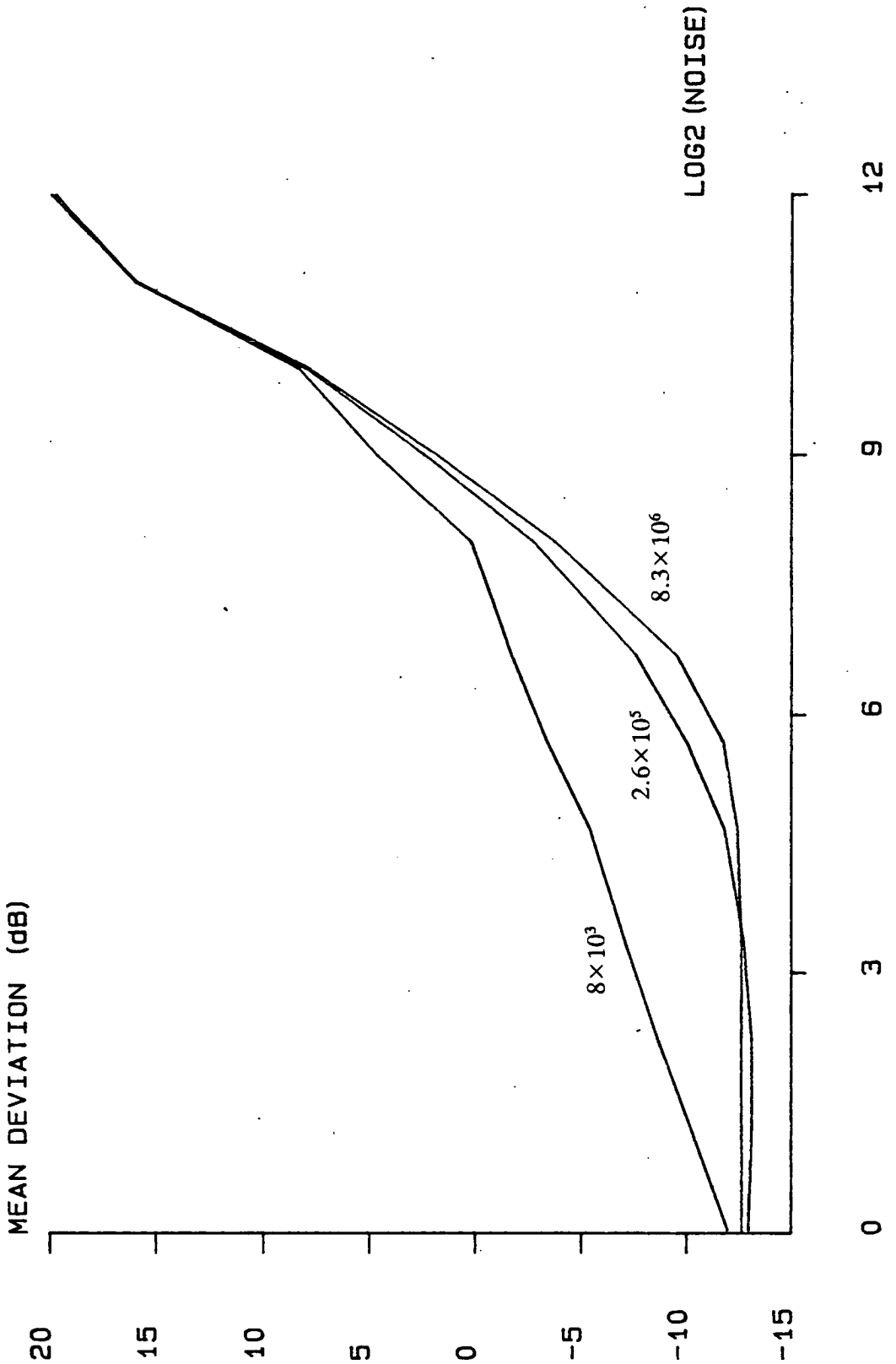

Figure 3.10: The effect of additive training signal noise of between 0 and 2^{12} lsb maximum magnitude on the mapping of a 12 bit ADC with different sample sizes.

means in terms of the converter and its physical quantisation levels. Figures 3.11 and 3.12 show a further two nonlinearities and their mappings, still for 7 bit ADC's, where again, the ideal staircase has been omitted for clarity.

For higher resolution converters however, it becomes impractical to use this form of result presentation as the scale of the nonlinearities becomes so small that it is impossible to distinguish even the nonlinearity from the ideal. This is obvious when it is considered that a four bit transfer function nonlinearity, equivalent to 16 LSB's INL, represents only approximately 1/4000th of the signal range of a 16 bit converter. To overcome this problem an expansion of scale is needed, and this is provided by the use of integral nonlinearity plots, where only the differences between actual and ideal transfer functions need to be accommodated on the axes.

3.5.3. Integral Nonlinearity

Although INL plots can easily be generated for lower resolution ADC's, it is for medium resolution devices that they are very useful in determining how well a nonlinearity has been equalised by the mapping process. Therefore, 12 bit converters have been simulated to generate integral nonlinearity plots, with the wide range of transfer function distortions available. Higher resolutions were not possible as the amount of data to be plotted became too great for the software used.

Figure 3.13 shows a simple nonlinearity, before and after mapping, where the maximum INL of the distorted transfer function is approximately 8 LSB. The mapped transfer function, however, only displays a maximum of around ± 0.5 LSB. Whenever the INL of the mapping strays such that it reaches the threshold value of 0.5 LSB, a mapping is performed so that the converter's linearity can be maintained to within the limits set by the threshold. The figure shows that as the gradient of the INL increases, so does the frequency of these mapping discontinuities. The steeper gradient relates to a high differential nonlinearity in the quantisation intervals of the ADC codes at these points on the transfer function.

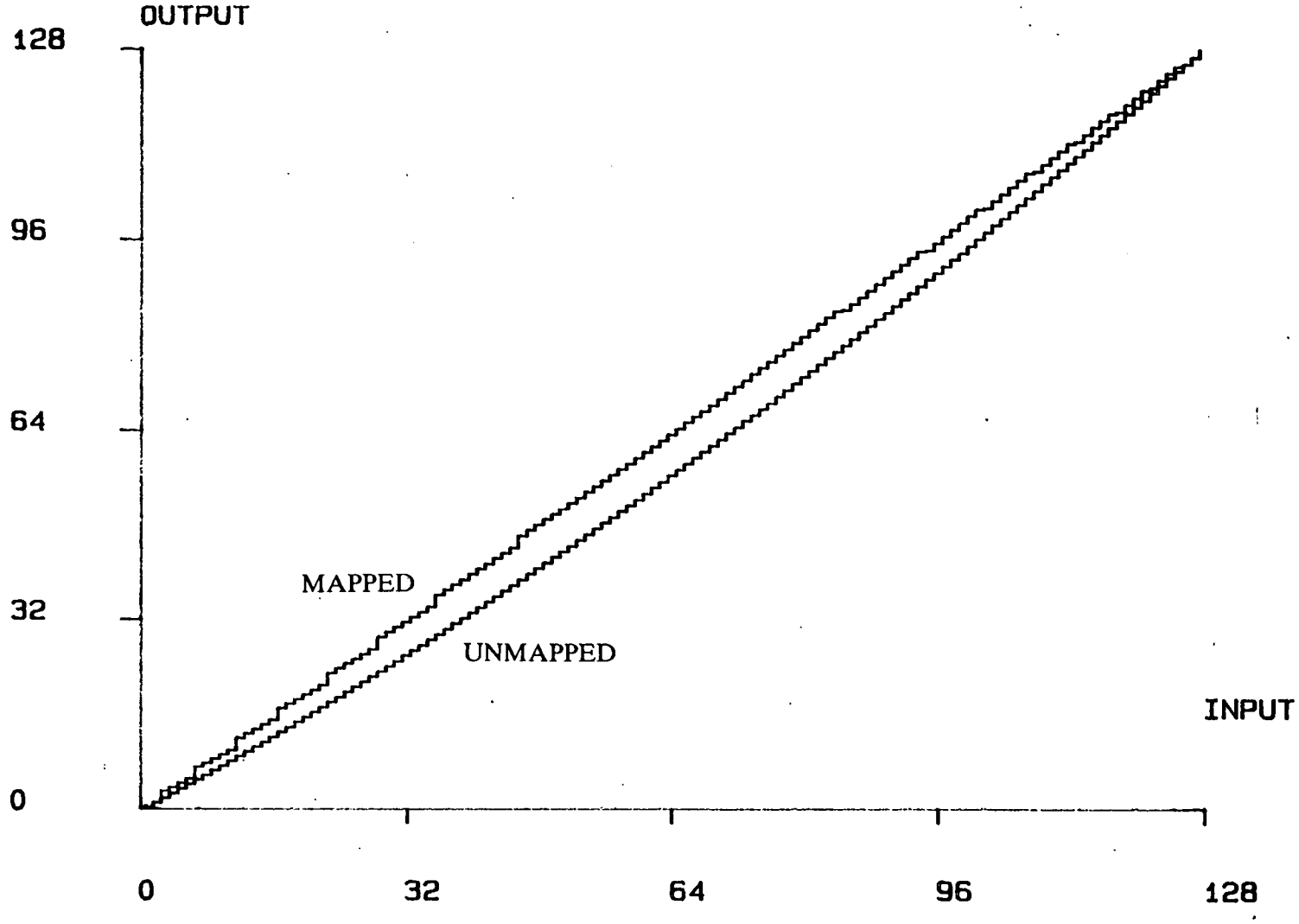

Figure 3.11: A 7 bit ADC nonlinear staircase function and its' mapping.

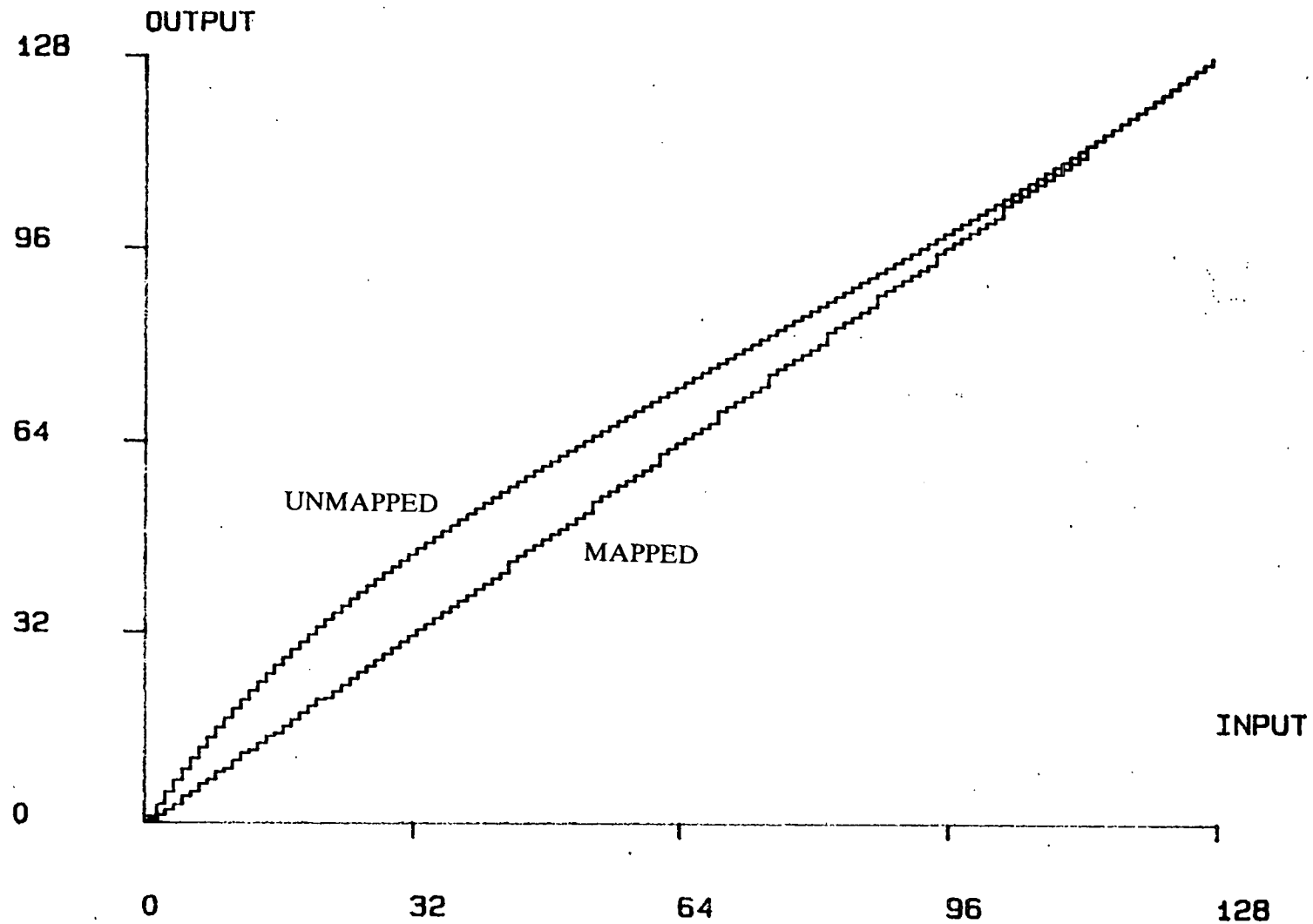

Figure 3.12: A different 7 bit ADC and its' mapping.

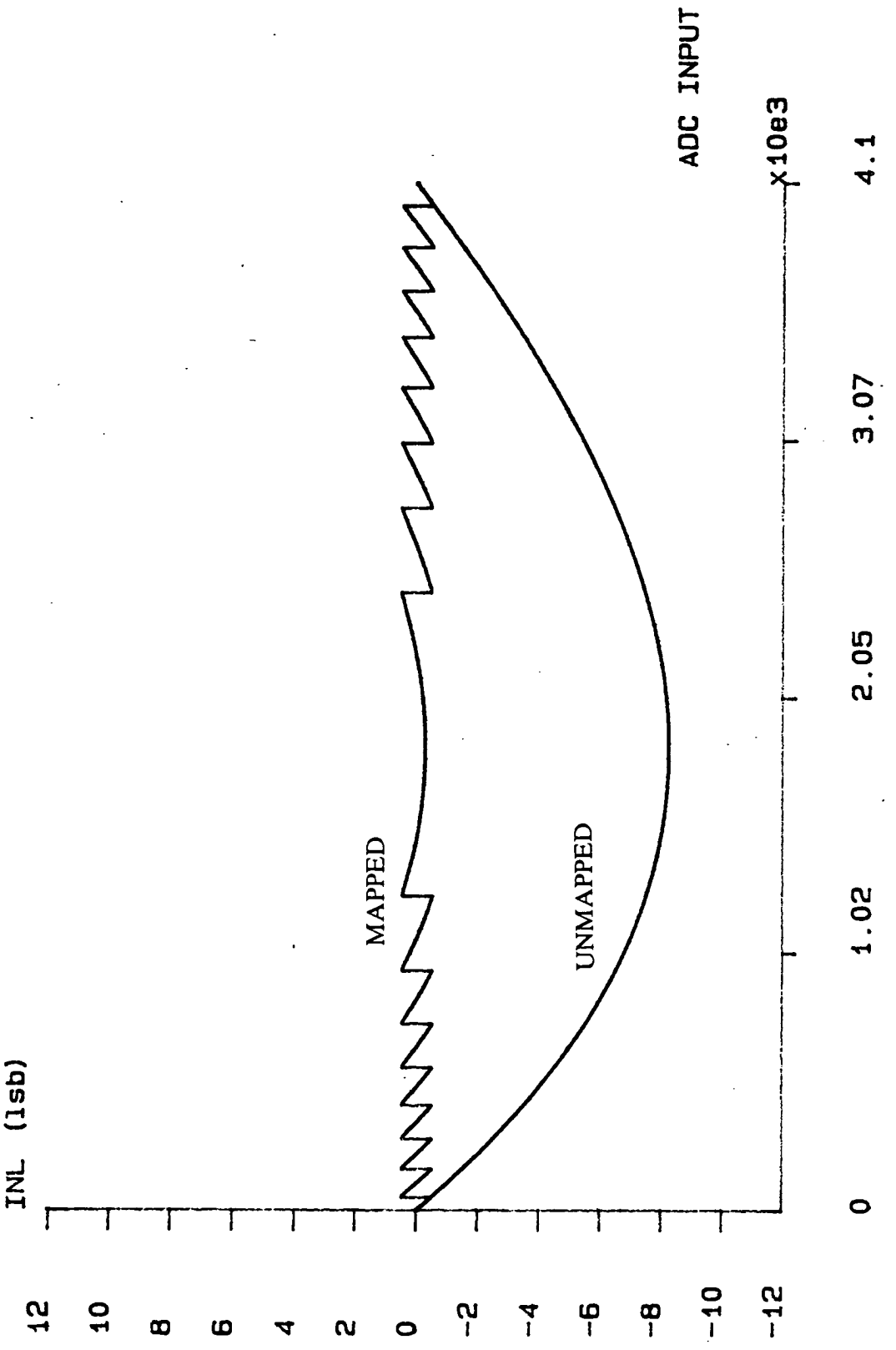

Figure 3.13: The original and mapped integral nonlinearity of a 12 bit ADC.

Figure 3.14 shows the INL of the mapping of another of the simpler, third order nonlinearities, which has a slightly larger peak value of over 10 LSB. The overall mean deviation of the unmapped distortion is approximately the same however. Due to the higher levels of differential nonlinearity in this distorted transfer function, compared to the previous figure, 25% more mapping discontinuities occur. These are concentrated in the lower third of the transfer function where the DNL problem is most severe. However, the mapped INL is still kept to within ± 0.5 LSB.

When distortions with much higher levels of DNL are simulated, the frequency of mapping discontinuities becomes even greater, as can be seen from figure 3.15. This figure shows the mapping of one of the more extreme converter nonlinearities, which involves superposition of periodic distortions of high DNL on top of the third order characteristic. This particular nonlinearity crosses the ideal, 0 LSB line, several times during its traverse of the converter's range. All of these potential problems however, are easily dealt with by the mapping process and the threshold tracking algorithm, resulting in the well bounded INL shown in figure 3.15.

3.5.4. Spectral Distortions

The results referred to in this section are representations of Fourier transforms performed on the data produced by sampling the reconstructed analogue signals generated by the digitising of two tone sinusoidal signals. The tonal signals have a maximum amplitude equal to 75% of the full scale range of the converter, with frequencies chosen so that an integer number of full cycles will fall within the time window of the implemented FFT. The signal reconstruction was achieved by the simulation of an idealised DAC.

Figure 3.16 shows results from a 12 bit converter which has one of the complex simulated nonlinearities described earlier, producing the intermodulation peaks and distorted harmonics present in the spectrum. The quantisation noise floor of the device is also very high and spikey, primarily due to the periodic nature of the

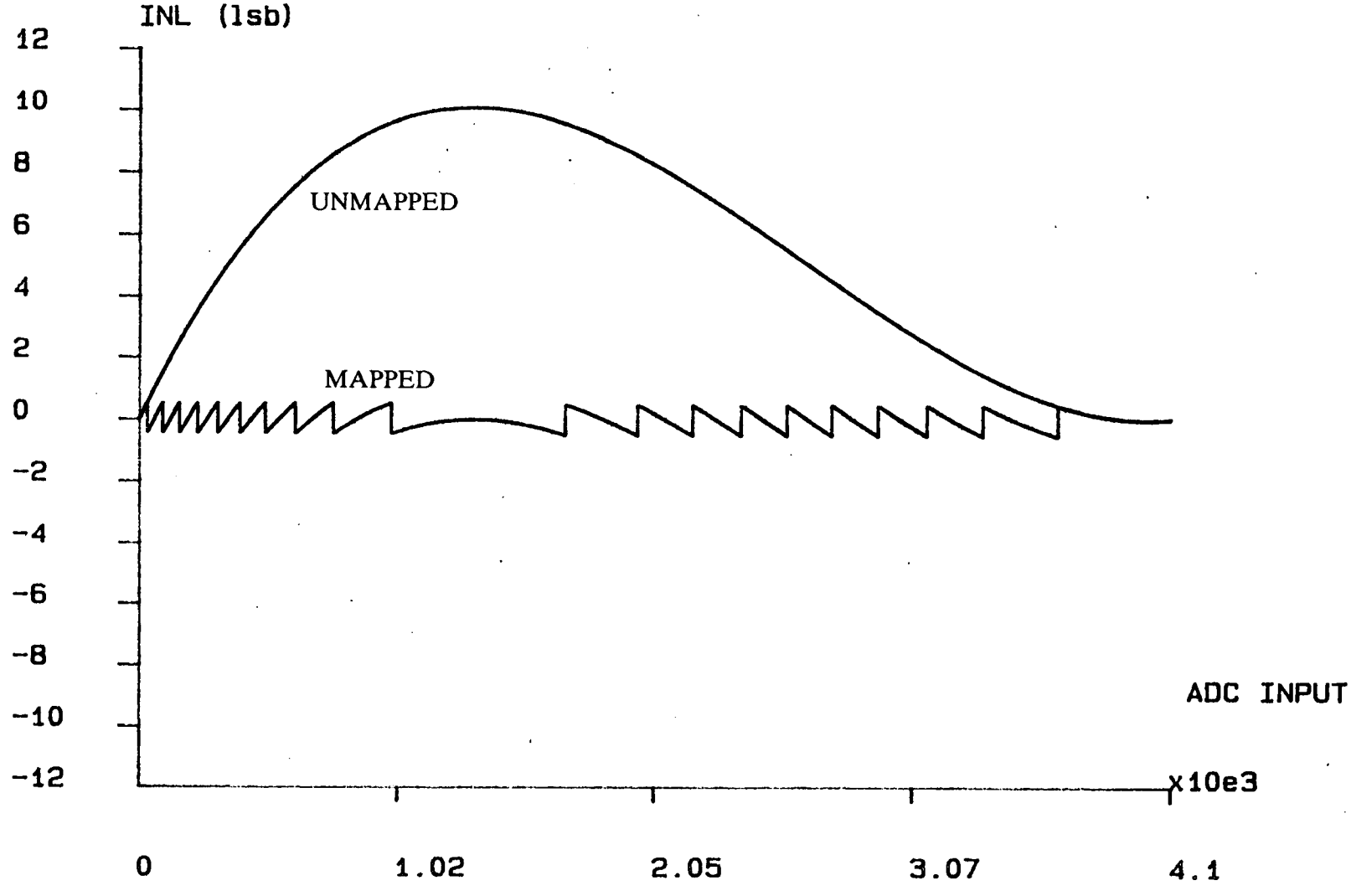

Figure 3.14: The INL of another 12 bit converter before and after mapping.

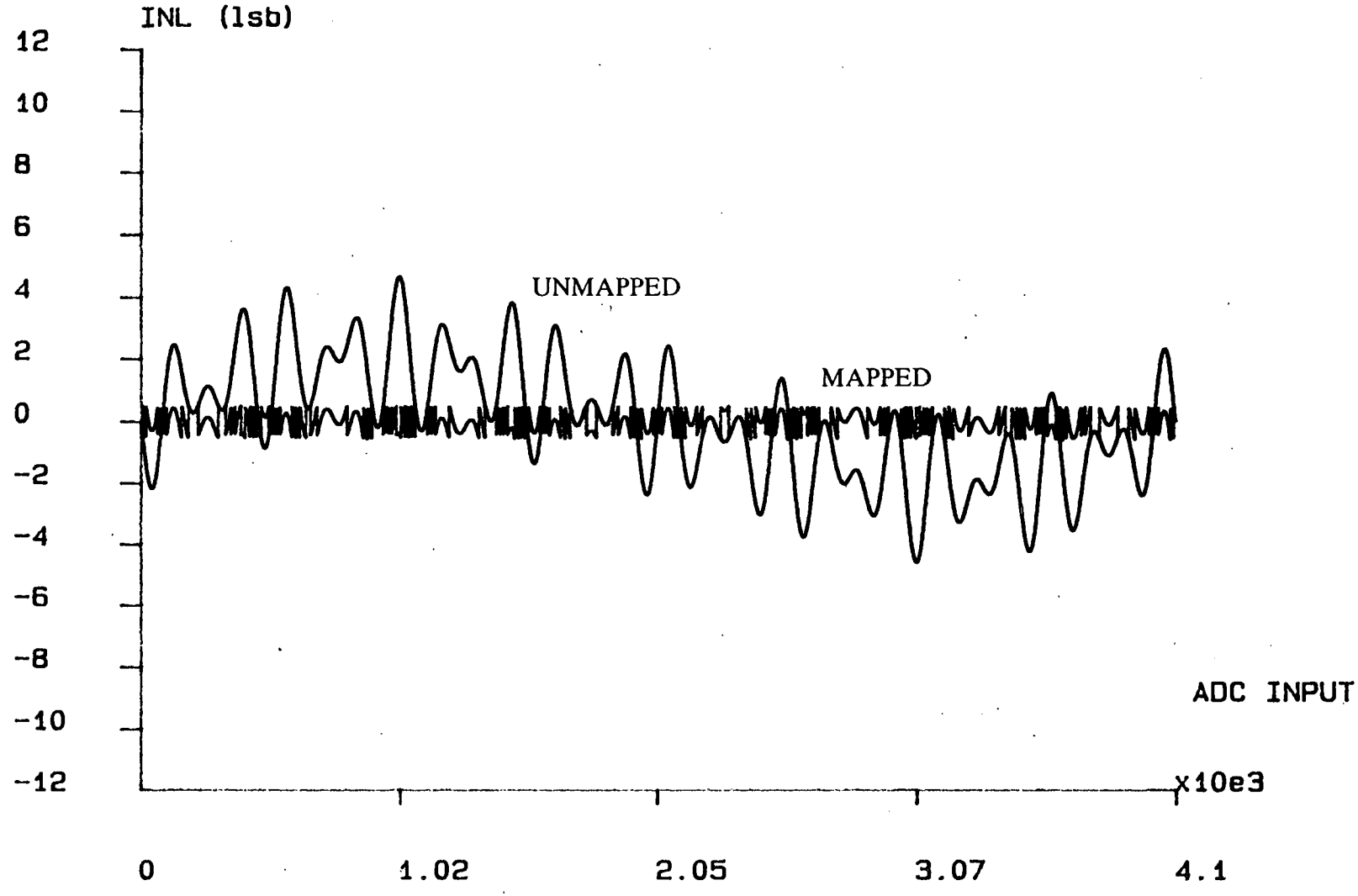

Figure 3.15: The INL of a more complex 12 bit nonlinearity and its' mapping.

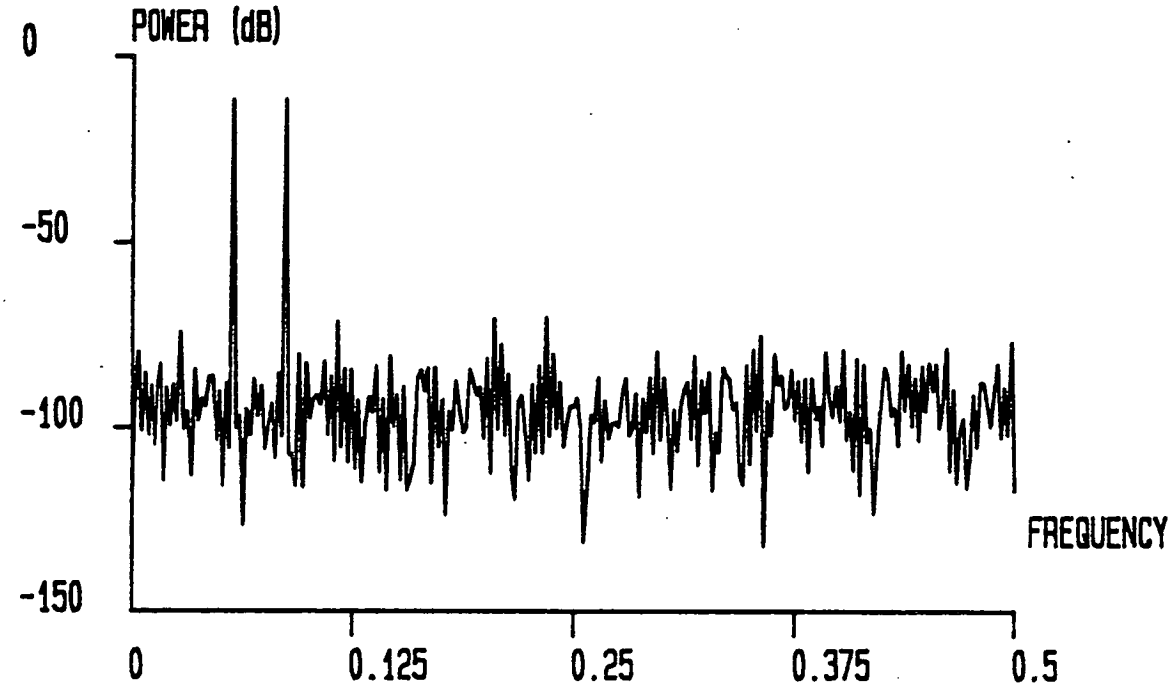

Figure 3.16: An FFT of a two tone sinusoid of 75% magnitude sampled by a non-linear 12 bit ADC.

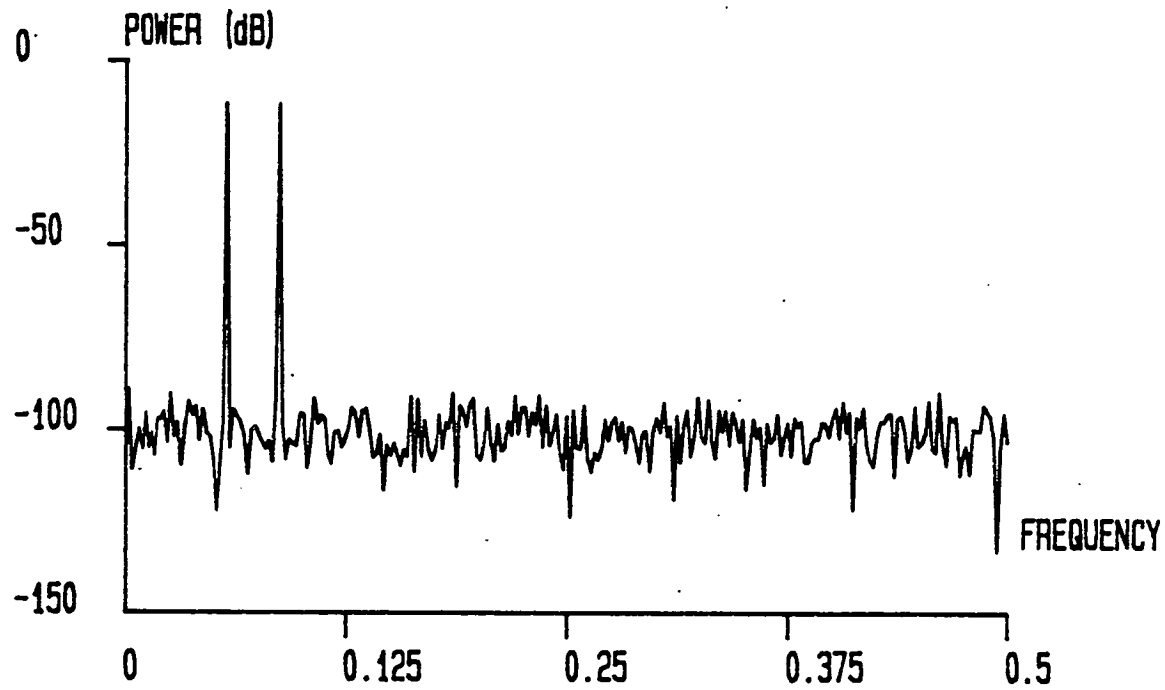

Figure 3.17: The same signal sampled by the same converter after mapping.

transfer function nonlinearity. The mapped converter of figure 3.17, on the other hand, shows none of the above characteristics, with the peak distortions having been reduced by around 20 dB in magnitude.

Unlike INL tests, harmonic distortion can be examined for any resolution of converter, and so gives a second measure (other than mean deviation) of the performance of the mapping procedure for high resolution converters, such as the 16 bit device shown in figure 3.18. Here the intermodulation products and distorted signal harmonics are much more prominent than for figure 3.16, i.e. the difference between the harmonic distortion peaks and the rest of the raised quantisation noise is higher. This is because the simulated nonlinearity is a simple third order function, rather than the more complex nonlinearity used in the previous example. When this converter is mapped, in figure 3.19, again the spurious peaks have completely disappeared into the noise floor, from a previous level of greater than -95 dB. As before, the mapping process has led to a reduction in the size of the distorted harmonics of over 20 dB.

This sort of linearisation continues over the whole range and resolution of converters and their nonlinearities, as can be seen from figures 3.20 and 3.21, which are the unmapped and mapped Fourier transforms respectively, of a 16 bit converter with one of the more complex distortions. Not only are the dominant peaks of the spectrum removed, but the quantisation noise floor is tidied up as well, especially with regard to the area around the 0.13 relative frequency point.

3.6. Sinusoidal Training Signals

All of the simulation results presented so far have assumed that a good quality uniformly distributed signal could be generated for the purpose of obtaining pdf histograms for the linearisation mapping. In practice, however, this is not so easy, and it is far more likely that a sinusoidal signal would have to be adopted, as these can be readily generated with a high spectral purity. This makes it necessary to find

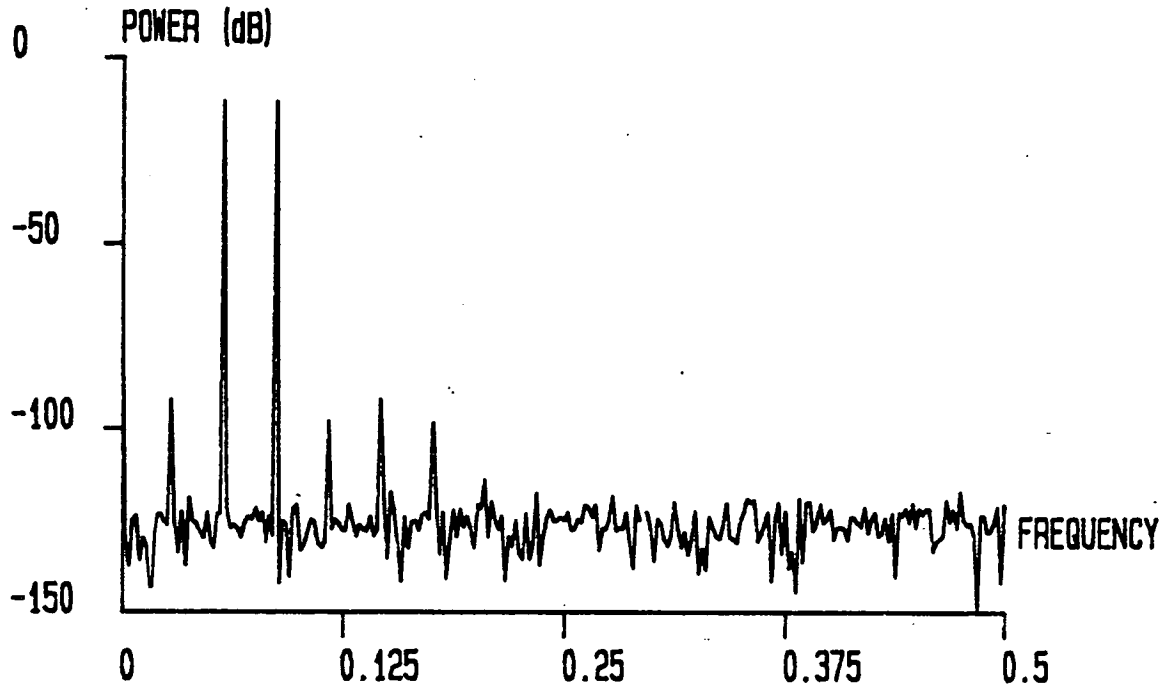

Figure 3.18: The FFT of a simpler 16 bit unmapped nonlinearity.

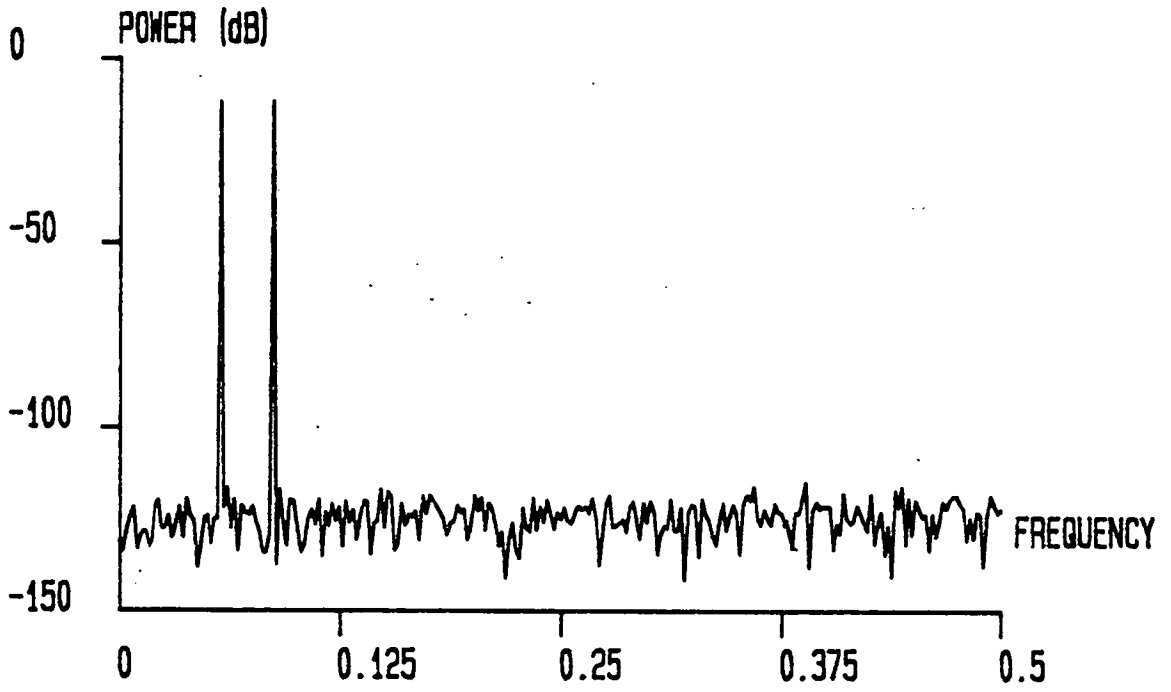

Figure 3.19: The same sampled signal after mapping.

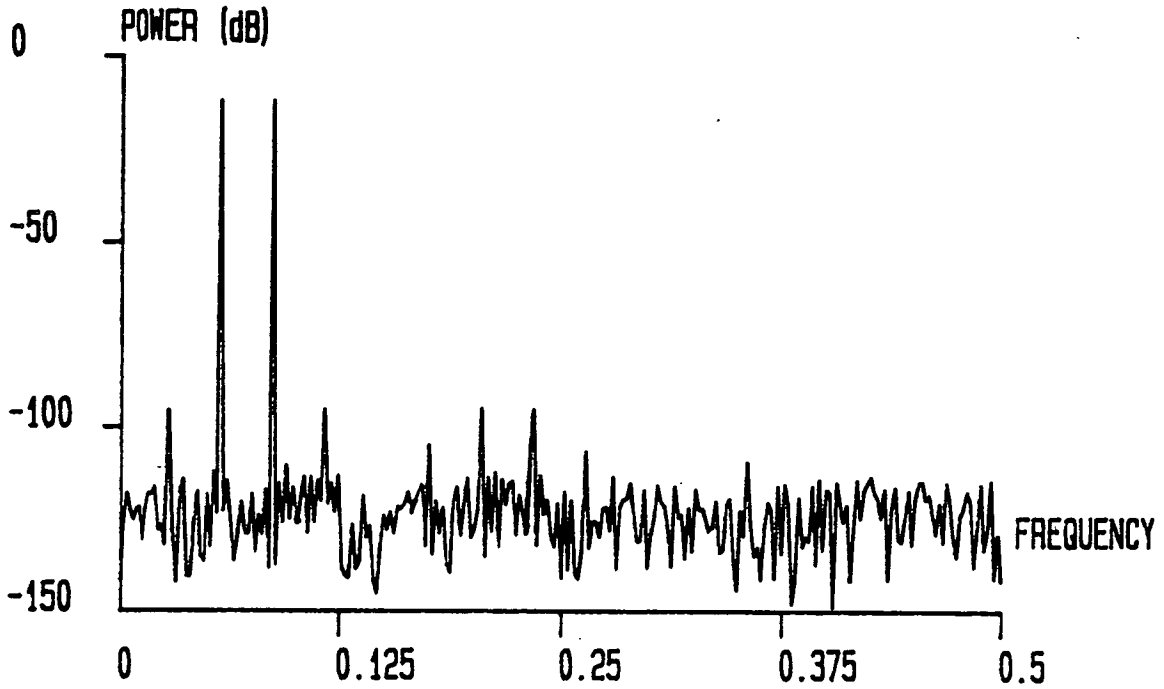

Figure 3.20: A more complex unmapped 16 bit nonlinearity.

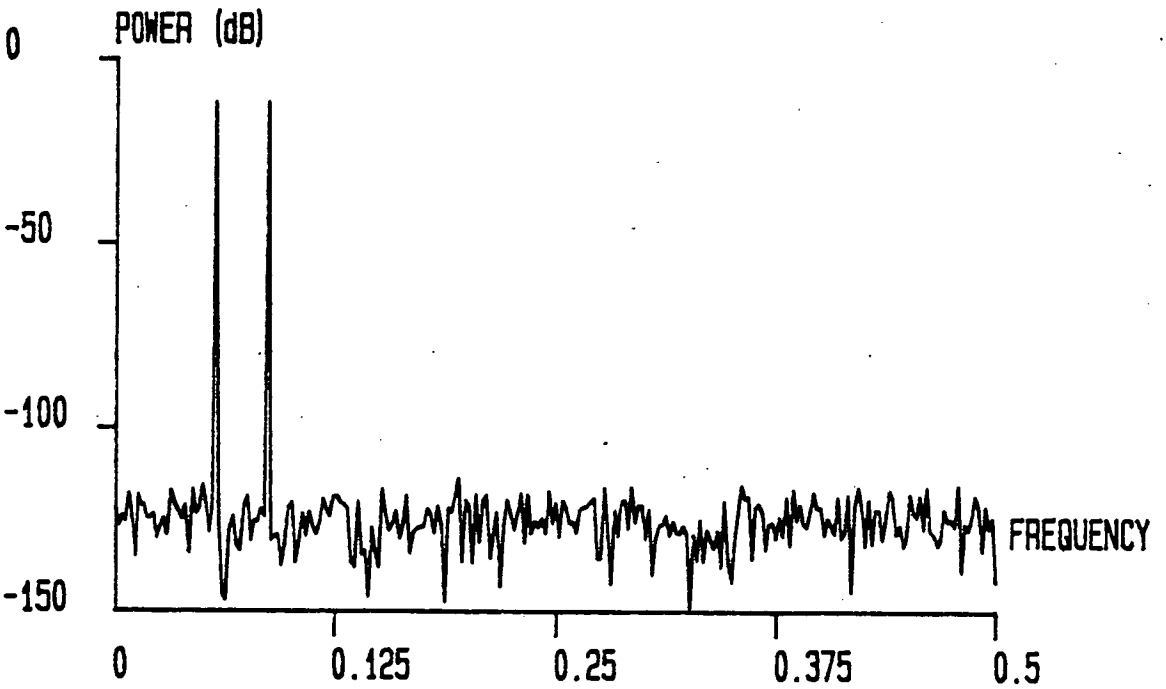

Figure 3.21: The corresponding mapped spectrum.

out what adjustments have to be made to the linearisation process to accommodate this change of training signal.

The simulation software was rewritten so that sinusoidal pdf's could be used in the linearisation process. It then became necessary to discover what effect, if any, this would have on the linearisation performance of the system. As can be seen from figures 3.22 and 3.23, which show mean deviation versus the training signal sample density, using both forms of training signal, for 12 and 16 bit converters respectively, there is no loss of performance, at the higher training signal densities.

The major effect seems to be that a few more training signal samples are needed to achieve an equal level of mean deviation performance as that previously obtained. This is due, of course, to the lower number of samples of the sinusoidal training signal at some points in the code density histogram, i.e. the middle codes. Similarly the codes at the two ends of the pdf histogram have increased values over the uniformly distributed signal.

Indeed, simulations have shown that an increase in the number of signal samples, by a certain amount, eliminates the observed loss of linearisation performance. The numerical value of this correction factor is determined from the ratio of the average number of samples of a uniformly distributed signal, at the centre of the code density histogram, and that of the corresponding sinusoidal distribution, and is found to be approximately 1.6, from measurements. Hence, the minimum number of code samples, on average, which are added to the sums in the threshold tracking algorithm is constant, for both pdf's.

3.7. Summary

This chapter has introduced and examined the idea and effectiveness of using a digital mapping to eliminate ADC transfer function nonlinearity. The mapping, which is stored in a look-up table, situated after the ADC in the overall system architecture, is generated from the characterisation of the converter nonlinearity

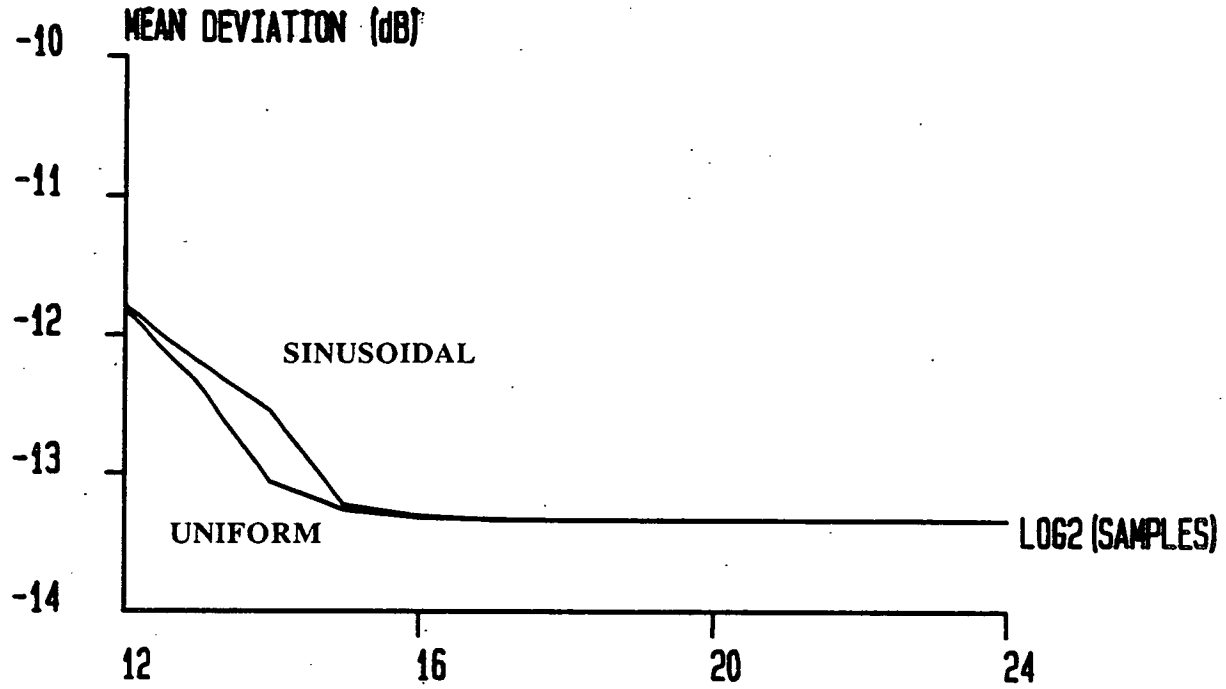

Figure 3.22: Comparison between uniform and sinusoidal pdf training signals, for a 12 bit nonlinearity.

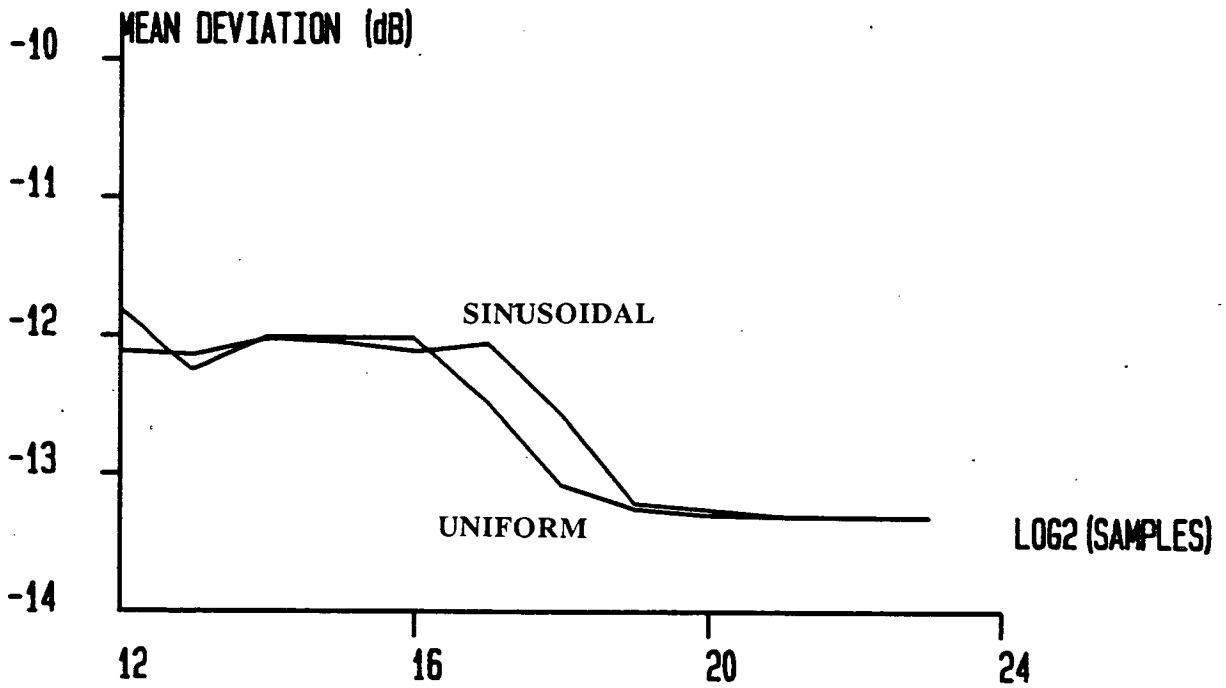

Figure 3.23: Comparison between uniform and sinusoidal pdf training signals, for a 16 bit nonlinearity.

provided by a form of code density testing. The threshold tracking algorithm utilises the information contained in the idealised and sampled training signal pdf histograms.

Thorough testing of the linearisation technique, with a large selection of ADC nonlinearities, has shown that the optimum value of the algorithm's threshold is in the region of 0.5 LSB, and that at an average code density of at least 4 samples/code is needed for satisfactory operation, with a uniformly distributed training signal.

When a more easily generated accurate sinusoid is used as the training signal, a larger amount of signal samples are needed. This does not apply when the undistorted training signal is also sampled, as both pdf's are accurately known and not calculated. This is equally true for uniform signals, although it is not so important.

The improvements in device linearity which are possible with threshold tracking mapping are very great indeed. Transfer functions become so close to the ideal that their graphical representation is difficult. Integral nonlinearity plots show that the mapped INL is bounded between approximately $\pm$ 0.5 LSB, for all simulated nonlinearities. Harmonic distortions and intermodulation products almost completely vanished in sampled signals, and noise floors returned to the levels that are expected from purely quantisation effects.

Overall, the technique can be said to work extremely well, for what amounts to little extra work on the part of the manufacturer, who is already, at least for high resolution devices, conducting code density testing.

Chapter 4

REDUCED RESOLUTION MAPPING

The solution to the problem of ADC transfer function nonlinearity provided by the mapping technique and the threshold tracking algorithm, offers large improvements in performance. In operation, the only hardware required by the system is the block of memory needed to implement the mapping look-up table.

4.1. Realisation of the Look-up Table

The size of this memory block is dependent upon the resolution of the converter. For instance, an 8 bit ADC would require only $2^8 \times 8 = 2$ kbits of memory, whilst a 16 bit device would need $2^{16} \times 16 = 1024$ kbits. This quantity escalates in an approximately quadratic manner such that a 20 bit ADC would need 20 Mbits of data storage. Clearly this causes problems for the higher resolution converters for which the linearisation process is most useful.

4.1.1. External Memory

If the look-up table is to be held in memory external to the main ADC circuitry, then there are two main choices for its type. It can be either volatile (RAM) or nonvolatile (ROM).

Volatile RAM chips have ever increasing densities, with 1 Mbit devices now becoming common place. Densities will increase slightly as fabrication feature sizes are reduced still further, but the technology is nearing its physical limits. These high

densities will allow the storage of a converter's mapping on only a handful of integrated circuits, which keeps the linearisation circuitry chip count low. Any reduction in the amount of memory needed would, of course, lower the overall number of required RAM chips, or at least the capacity, and hence price, of the single memory chip needed.

If a volatile storage medium is used, its contents will be lost when the power supply is removed from the ADC system. This is problematical for the construction, operation and shipping of converters calibrated at their place of manufacture, and not in-situ. The simplest solution would be to have the mapping stored on a nonvolatile medium, such as a magnetic disc, from where it could be downloaded into the RAM chips. This would require some additional interface circuitry and depends upon the existence of some sort of computer within the overall system architecture, from which to download the mapping. Clearly the use of volatile RAM storage of the ADC transfer function mapping is not very convenient.

Nonvolatile storage of the mapping, in some form of ROM, has many advantages over volatile storage. There are no longer any problems involved in shipment or operation, and no need for extraneous interface circuitry. Unfortunately, several other factors become important when some form of ROM storage is used.

Firstly, programmable ROM's have longer access times than RAM's and hence, will reduce the throughput rate of the analogue to digital conversion system. Secondly, programmed ROM devices do not have the same levels of packing densities as RAM chips do, so a larger number of devices would be required to implement the look-up table compared to the use of volatile memories. Again, therefore, as with volatile storage media, the system chip count would be lowered if the overall memory requirement for the look-up table could be reduced.

4.1.2. Internal Memory

With the high packing densities available using current technology it is clearly possible to fabricate enough transistors and other devices to realise an ADC on a single silicon chip, complete with a sample and hold function and an internal clock generator. It should also be possible, therefore, to incorporate the mapping look-up table on the same silicon chip as the converter. Indeed, many recently developed devices include blocks of programmable ROM [3]and RAM [10,11] (generally for the purposes of automatic linearisation).

The use of on-chip RAM would allow the highest packing densities, and hence would use the least silicon area to implement, although it would still not be likely that the total amount of memory required for a high resolution converter mapping could be put on the same chip as the ADC. As already examined, the use of random access memories would incur the penalties associated with all volatile storage media.

Standard ROM, although offering a space efficient, fast and convenient solution to the problem of on-chip mapping storage, would have to be mask programmed. This means that as each mask would probably only be useful for a single ADC, the realisation of the look-up table by conventional ROM would be prohibitively expensive.

Using some form of user programmable ROM is the best answer in terms of operation of the device, ease of calibration and cost. Unfortunately, field programmable logic is not very dense and would require a large silicon circuit and i/o pad area to implement. Also, the fabrication of programmable nonvolatile memory is not as simple as RAM, and therefore, the use of such circuitry is limited to the chip foundries that can accommodate this type of device construction. If the amount of memory bits required could be reduced, not only would less silicon area be used, but the restrictions on the type of fabrication process would be relaxed.

Overall it would seem that the easiest solution to realisation of the look-up table is

to implement it using externally programmable ROM chips. The only problem with this is that it makes for a high device count for the linearisation system. If the amount of memory needed by the system could be reduced, then the chip count could be much smaller.

4.2. Offset Storage

One simple and easily implemented method of reducing the required memory size is to only store mapping offsets. A mapping offset is the difference between an ADC code value and its mapping. To use such a mapping look-up table, it is necessary to include a digital adder in the linearisation architecture, between the mapping memory and the output of the ADC system, as shown in figure 4.1.

The introduction of this adder, which could easily be fabricated on the same silicon as the ADC (when an on-chip memory was being implemented) would introduce an extra delay into the overall cycle time of the converter. For an on-chip device this would be very small, but it may well be significant for an external adder and memory, where it represents an increase in the system chip count as well.

The advantage gained by the use of offset storage is determined by the size of the maximum allowable offset. The size of the maximum offset, in turn determines the maximum converter integral nonlinearity that can be accurately mapped. For example, a 5 bit offset would allow ± 16 LSB's of INL to be completely mapped whereas, a 10 bit offset would cater for ± 512 LSB's INL. The size of the allowable offset also influences, to some extent, the size of the adder required, but this is of little importance in silicon terms, considering the resolution of converters involved. The size of the offset used is determined entirely by the amount of INL which can reasonably be expected to occur in a particular ADC transfer function. This is variable between converter types, resolutions and speeds.

For the simulations presented here, an offset of 5 bits magnitude was assumed to be sufficient for the scale of nonlinearities being used.

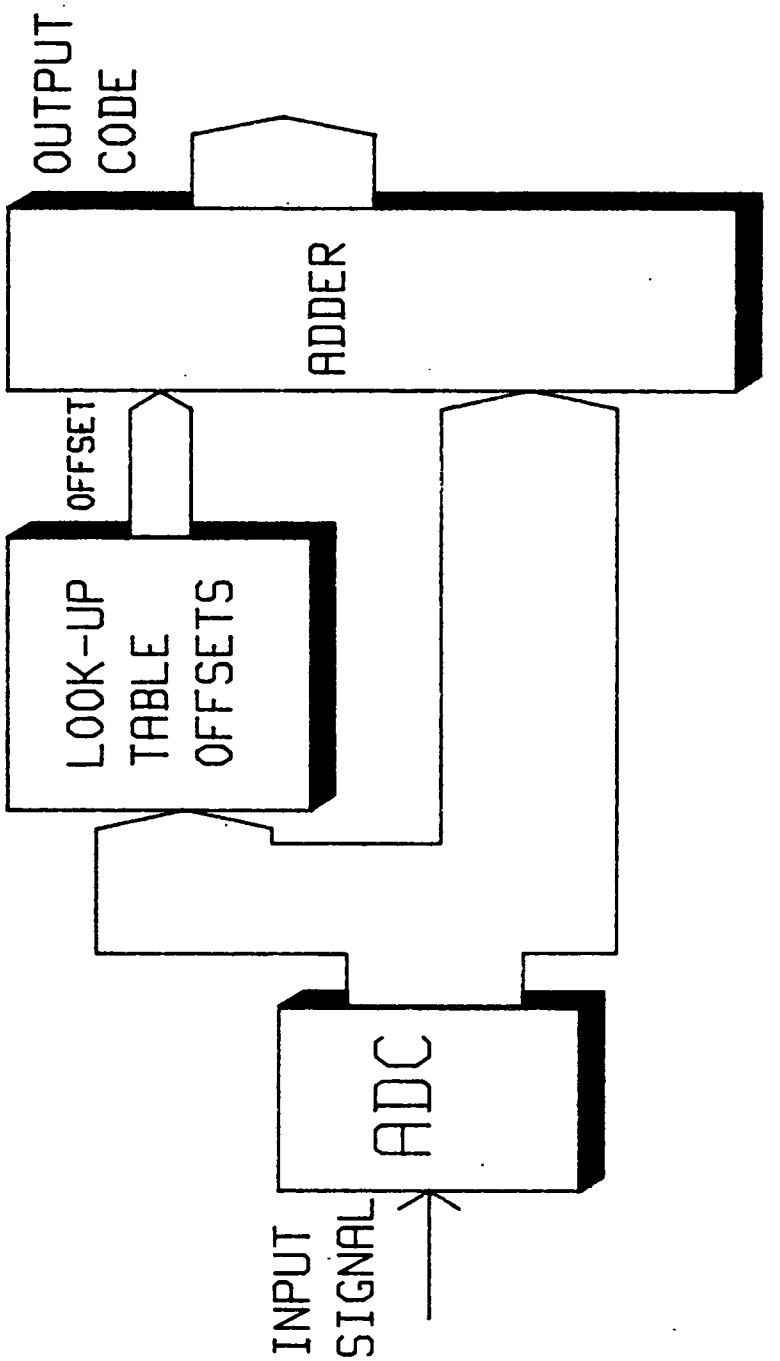

Figure 4.1: The architecture for a linearisation system which only utilises mapping offsets.

For a 16 bit converter then, the reduction in memory requirement is from $2^{16} \times 16 = 1024$ kbits to $2^{16} \times 5 = 320$ kbits, representing a reduction by a ratio of over three to one. Clearly this will improve as the resolution of the converter increases, such that for a 20 bit device the memory requirement goes down from 20 Mbits to 5 Mbits, which represents a four to one reduction.

4.3. Segmentation and Interpolation

Although offset storage has significantly reduced the mapping table memory requirement, it is still fairly large, particularly for higher resolution converters. The only way to reduce the size of the memory blocks further is to reduce the number of locations addressed. This implies that not every mapping is stored, with a consequent reduction in the total amount of memory needed. Such a storage technique would require some method of reconstructing the now missing mapping codes when they were accessed by the outputs from the system ADC.

4.3.1. Segmentation of the Transfer Function

When an actual transfer function mapping is examined in some detail it can be seen that generally, the mapping can be said to be continuous with only very infrequent discontinuities. A continuous mapping, in this context, is taken to mean that each consecutive code maps to the value of the previous mapping incremented by one, with a mapping discontinuity being an occasion when this does not occur.

If solely the mapping discontinuities were stored in the look-up table, then the continuous segments of the transfer function could be deduced by simply adding the distance between the last mapped discontinuity and the code in question. This would mean that perhaps only a few hundred individual mapping points, those at the mapping discontinuities, would have to be stored in memory, giving a reduction in the total memory requirement of a few orders of magnitude.

The added difficulty of this sort of mapping storage strategy is that when the input

code to the mapping table does not correspond directly to a mapped code, some combinational logic scheme must be used to determine which mapped codes it lies between, so that its correct mapping may be reconstructed. Also the storage of several unknown addresses and their associated mapping contents is considerably more complex than using a fully addressed system. A level of address indirection is inserted into the scheme with its attendant extra accessing delay, which is to be added to that of the customised windowing control logic required. This sort of system becomes either highly complex or highly individual to each ADC, and hence, uneconomical.

A lot of these problems can be overcome by the technique of always storing the same mapped codes in the look-up table memories i.e. by regularly segmenting the ADC transfer function. These addresses, constant for every mapping, and no longer necessarily those associated with the mapping discontinuities, will require no complex addressing scheme, and no indirection, only simple address comparison. Unfortunately, the reconstruction of the code mappings becomes more complex and less than perfect, as the segments of transfer function between each stored mapping point are unlikely to be continuous. Hence, the straightforward addition of the distance from the code of interest to the last mapping point is likely to introduce errors. These can be reduced if the number of stored mapping points is increased so that there are likely to be less mapping discontinuities in each segment.

4.3.2. Reconstruction and Interpolation

If the mapping offset at the lower of the two mapping points bounding a segment is taken as the offset for the whole segment, then, if there are no discontinuities, perfect reconstruction of the fully addressed mapping will be achieved.

If however, as is likely, discontinuities exist within a transfer function segment, then they will accumulate so that at the upper end of the segment the total size of all of these errors will be the mapped integral nonlinearity at that point. This is the sort

of situation is shown in figure 4.2. The INL errors when viewed over the whole segment, would initially be zero, then one, then two LSB's etc. until they reached their maximum. This is true for both increasing and decreasing INL over the ADC transfer function segment.

For the more complex cases involving both increasing and decreasing INL within the segment, many things can happen that are too varied to go into here. Any integral nonlinearity which occurs within a segment however, remains until it is either cancelled out by further INL of the opposite polarity, or the end of the segment is reached. Overall, this method of mapping reconstruction is adequate (if a sufficiently large number of segments are used) but by no means perfect.

Some form of interpolation of the offset for the transfer function segments between mapped points would, to some extent, remove some of the errors in the reconstructed mapping caused by the discontinuities. Use of a simple average of the mapping offsets at the mapping points at either end of the transfer function segment under consideration, for the offset of the whole segment, should at least give some improvement in performance.

Where there is increasing INL across the segment the result of averaging is that the lower end of the segment has negative integral nonlinearity whilst the upper end has positive. The centre of the segment, on the other hand, has zero INL over a relatively large section of the segment. Taken over the whole section of the transfer function, the mean deviation from the ideal is approximately half of what it was for the simpler method of reconstruction suggested earlier. This situation is shown in figure 4.3. Similar results pertain to the case of decreasing INL across the segment.

For more complex combinations of mapping discontinuities the situation is much less clear. The averaging method generally gives an improvement in the mean deviation of the segment over the simpler method, but it is easy to find situations where it will degrade the linearity of the segment. These tend to be rather strange

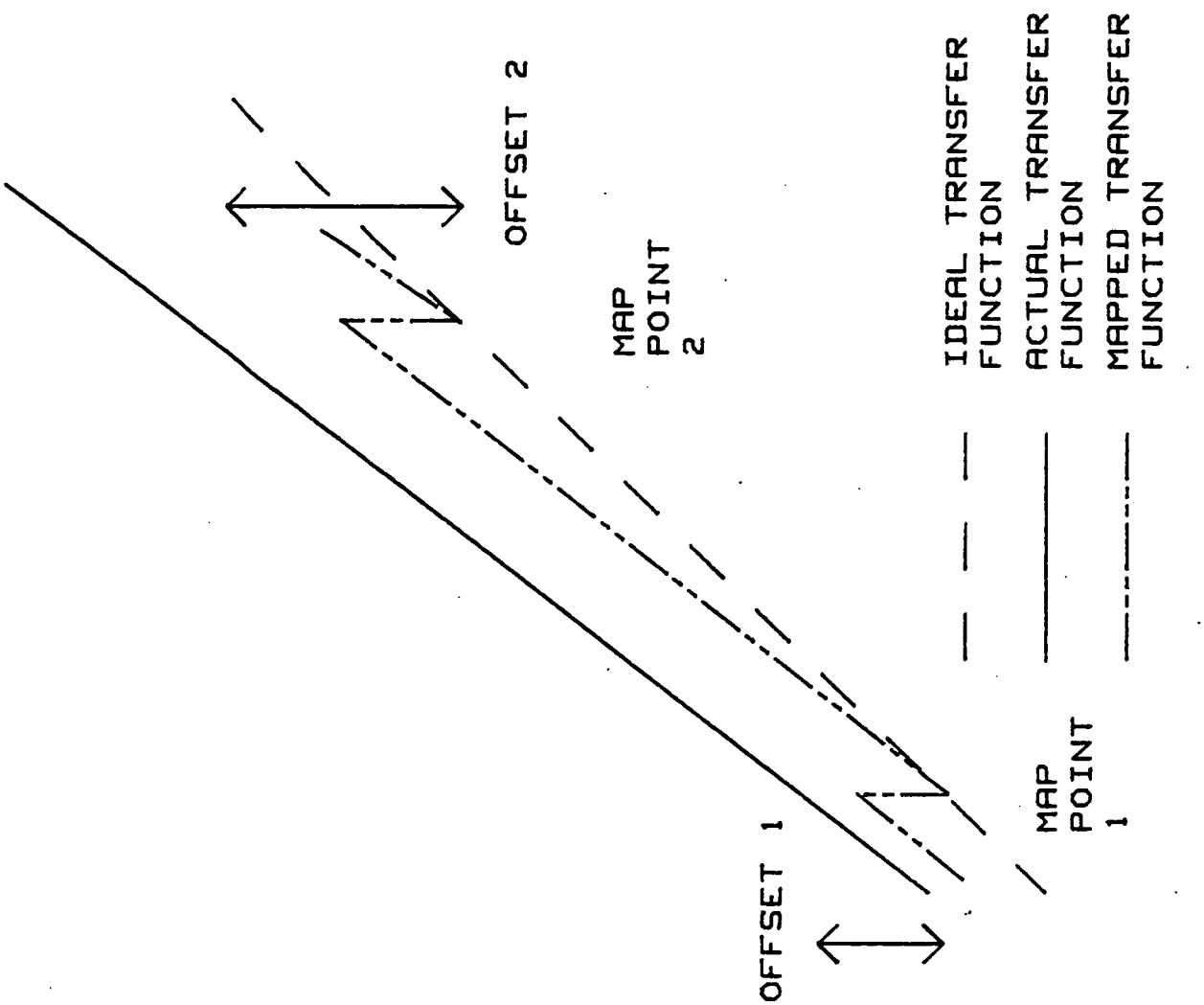

Figure 4.2: The resultant INL of a mapping when segmentation is used.

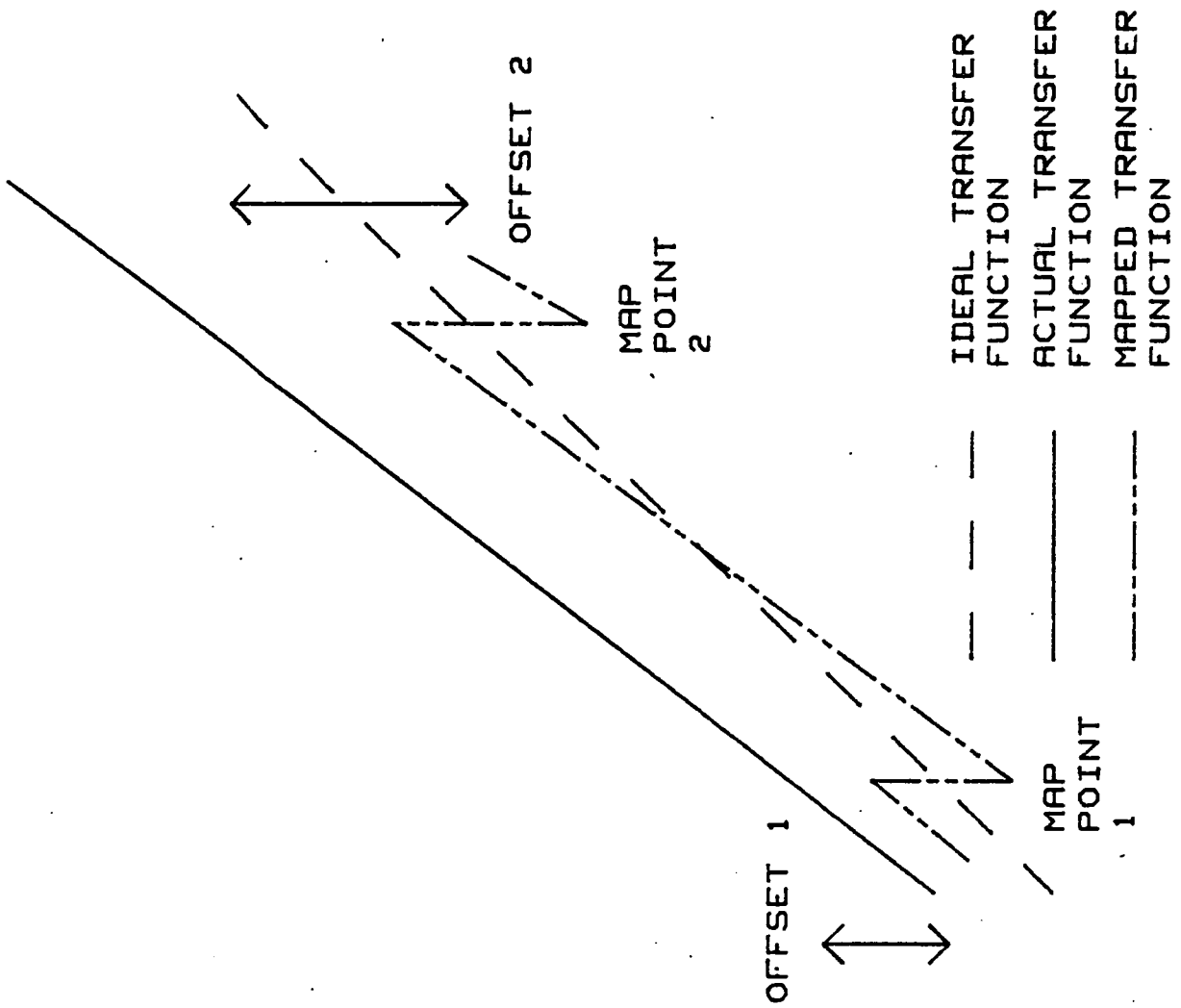

Figure 4.3: A segmented transfer function where the mapping offsets have been averaged.

and unusual transfer function shapes, which are likely to have a low probability of occurrence. One way round this new source of error is to make the segments smaller, thus forcing them to more closely approximate either of the two simple scenarios, for which it is known that the averaging strategy works well.

It should be emphasised that the average offsets are calculated during the mapping process, before they are stored, and not during their reconstruction in the operation of the ADC system.

4.4. Optimal Segmentation

There are many ways to regularly segment an ADC transfer function and the best number of segments to use will depend on several factors. Firstly, the number and position of discontinuities in the original fully addressed mapping. Secondly, the maximum differential nonlinearity of the transfer function, and lastly, the degree to which the nonlinearity can be said to be smoothly or abruptly changing. These factors are all determined by the actual distorted transfer function being mapped, and do not depend on any system requirement.

An aspect of the addressing system which does have some bearing on the choice of the number of segments, is the ease by which they can be accessed by the ADC output code. That is, it is advantageous to make the selection of the appropriate stored offset as simple as possible. The best way to ensure a trivial solution to this, is to use 2^n segments, where the whole number n lies between one and the resolution of the converter, inclusively. Therefore, by this criteria, maximum segmentation gives the fully addressed, original mapping, which allows no storage saving, other than that provided by the use of offsets.

The reason for this limitation to the number of segments is that it allows the addressing of the memory look-up table to be done solely by the more significant bits of the ADC output. For example, if $2^8 = 256$ regularly spaced segments are chosen then the mapping points, for which an offset has to be stored, will be

coincident with the points on the transfer function when the lower order bits (all of the bits except for the most significant 8) are zero.

If 2^{10} = 1024 segments are made, with a corresponding number of offsets stored, then the most significant 10 bits output by the ADC will be used to address the look-up table. The architecture of this type of system, which is similar to that for a fully addressed offset storage mapping, is shown in figure 4.4.

The use of 2^n segmentation completely removes any need to have even simple address comparison logic to determine which stored offset to add to the ADC output code. This implies that the overall linearisation system will operate as fast as the fully addressed offset mapping storage set-up, with no additional hardware.

4.4.1. Required Addressing Lines

As 2^n segmentation has been decided upon as the method of compartmentalising the ADC transfer function, all that now remains is to select how many of the most significant bits must be used to address the memory look-up table. To determine this, simulations of 12 and 16 bit ADC's were run over a wide ranging selection of transfer function nonlinearities, with a suitably large number of training signal samples i.e. 2^{16} and 2^{20} samples for the 12 and 16 bit converters respectively. These sampled pdf densities allowed the ADC simulations to perform at, or around, their minimum mean deviation level. This meant that the effects of the reduced resolution addressing would be viewed in a situation which was as close to that of a real world linearisation scenario as possible.

All the simulation results showed that as the number of more significant bits addressing the look-up table was increased from 5 towards the maximum, that the mean deviation performance of the reduced resolution mapping improved. This was true for all the nonlinear distortions simulated, as can be seen from figures 4.5 and 4.6, which show mean deviation against the number of lines addressing the mapping memory. Each separate trace on the figures is generated by a different nonlinearity,

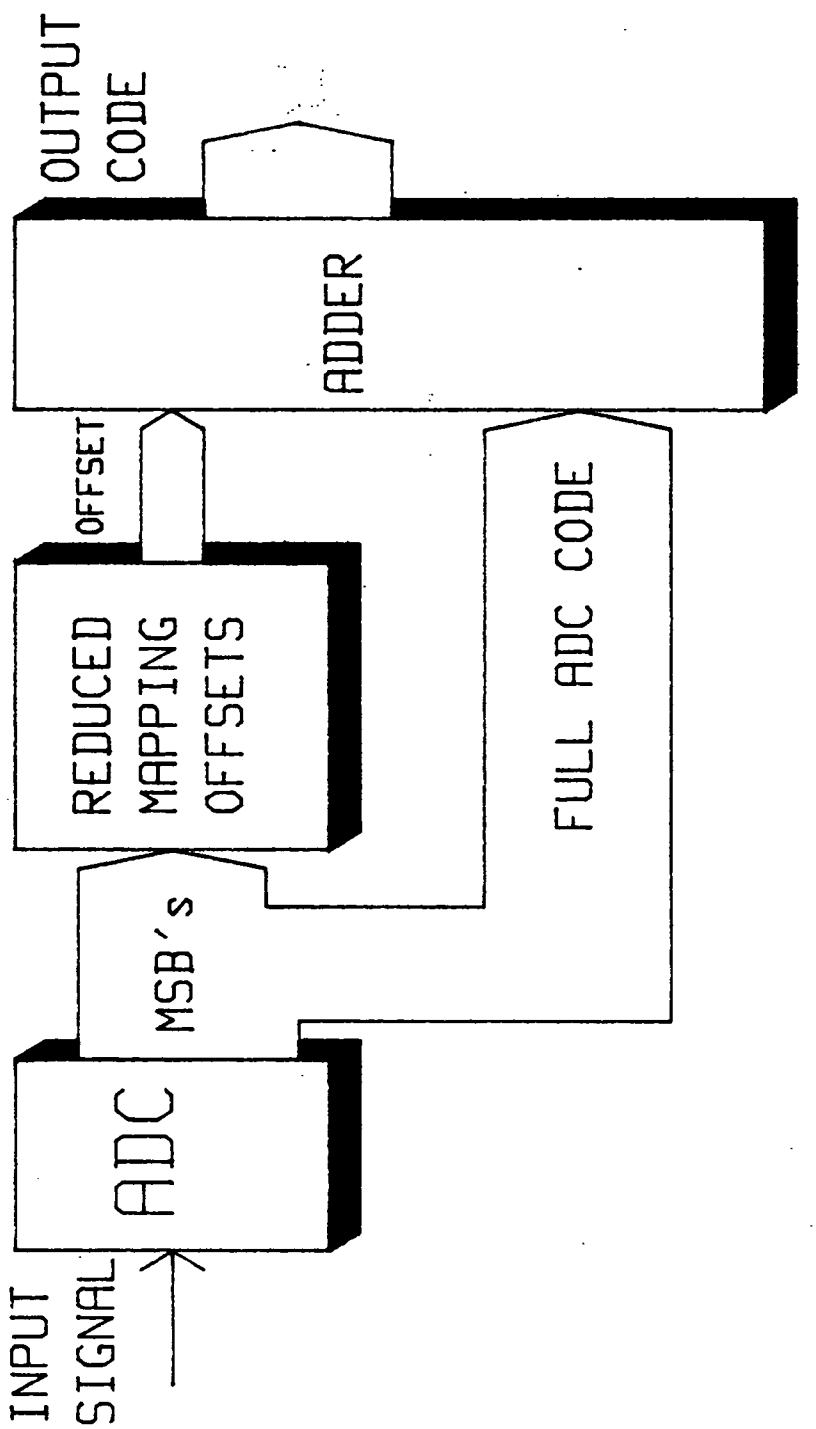

Figure 4.4: The linearisation system for reduced resolution mapping.

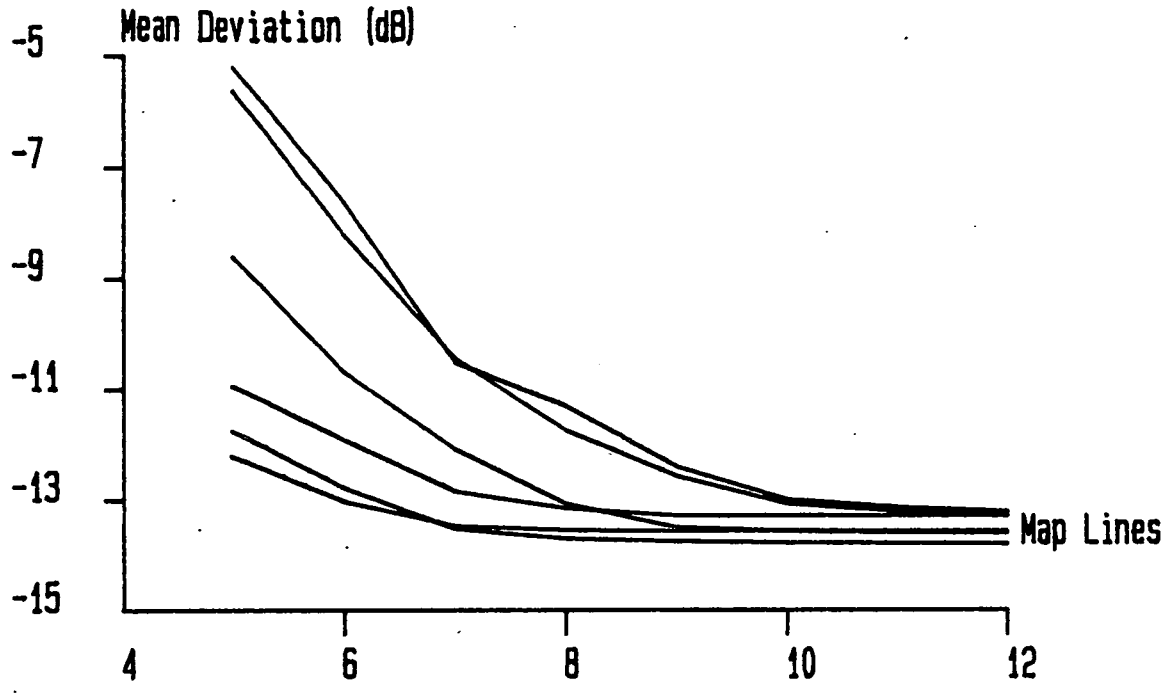

Figure 4.5: The mean deviation performance of 12 bit distortions with a range of mapping resolutions.

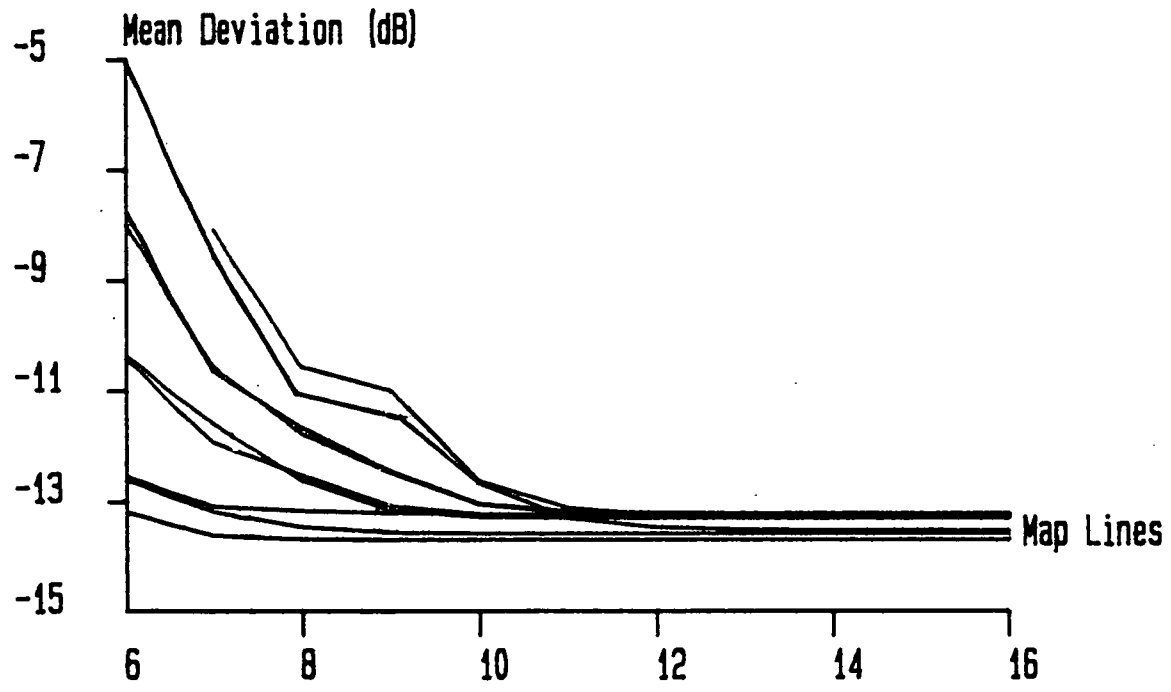

Figure 4.6: 16 bit distortions with varying mapping resolutions.

and figures 4.5 and 4.6 show the results for 12 and 16 bit conversion simulations respectively.

No improvement in mean deviation can be achieved by the use of further addressing lines on the horizontal sections of the plots. The ideal situation would be when all the simulated distortions were linearised to their maximum level, by the use of a particular, small number of address lines. In the 12 bit simulations of figure 4.5, four of the transfer function nonlinearities are linearised to their optimum levels by the use of only 9 address bits, whilst the other two distortions require more than 11 of the more significant bits to be employed. In the 16 bit simulations of figure 4.6, the situation is very similar, in that the mean deviation plots of five nonlinearities have become horizontal with the use of 9 addressing lines whilst the rest, of what turns out to be the more severe distortions, require upwards of 11 bits.

4.4.2. Acceptable Losses

For both resolutions of converters, of which the simulation results are shown, 11 address bits would seem to be the minimum level needed to ensure the best realisable performance of the system. This is also true of other simulated transfer functions, which have not been reproduced here. These results have been determined by using a limited, though varied, range of nine device distortions. Obviously, some margin of safety would have to be built into any actual system, and this would mean that 12 or even 13 address bits would have to be laid down as the minimum number of look-up table address lines allowable.

However. in all the simulations, the amount of mean deviation improvement yet to be gained by the point at which 10 address lines have been used, is very small, of the order of 0.5 dB. This relates to the severest of ADC transfer function distortions which include periodic perturbations, and not the more realistic third order nonlinearities of the distortions which can be linearised with only 8 or 9 bits. Therefore, it can reasonably be assumed that the use of 10 bits to address the

mapping look-up table will give optimum, or near optimum, mean deviation performance for any realistic ADC nonlinearity, with a good margin of safety. The extent to which this assumption causes a reduction in linearisation, particularly for the more severe distortions, is examined in some detail in the next section. An increase in the integral nonlinearity of mapped transfer functions and some distortion of the harmonics of reconstructed periodic signals, can be expected.

4.5. Results with Reduced Resolution

Simulations were written to perform any given resolution of mapping, with any number of training signal samples, and to output the data necessary to generate mean deviation plots, integral nonlinearity plots and reconstructed two tone periodic signal samples, for Fourier transformation so that analysis of distorted harmonics and intermodulation products could be studied. The software used to do all this is given in Appendix A. Most of the results of interest that were gathered from the mean deviation plots have been presented already, which leaves only the other two measures of the reduced resolution mapping technique to be reported.

4.5.1. Reduced Resolution and INL

A plot of the integral nonlinearity of a 12 bit ADC transfer function, mapped with only 5 address lines is shown in figure 4.7. The reduced resolution mapping has performed well, with the maximum INL reduced from around 9 LSB to less than 2 LSB, compared to the ± 0.5 LSB results which can generally be obtained with a fully addressed mapping. The discontinuities caused by the mapping process can be seen to be confined to regularly spaced points, whereas in a full resolution mapping they would occur as and when the mapping crossed over the ± 0.5 LSB levels. This is why the reduced resolution mapping process gives less than perfect results.

The mapped transfer function represents an increase in mean deviation error of 2.4 dB over the full resolution mapping. Even though the distortion chosen for linearisation was not too difficult to map, in terms of its previously determined

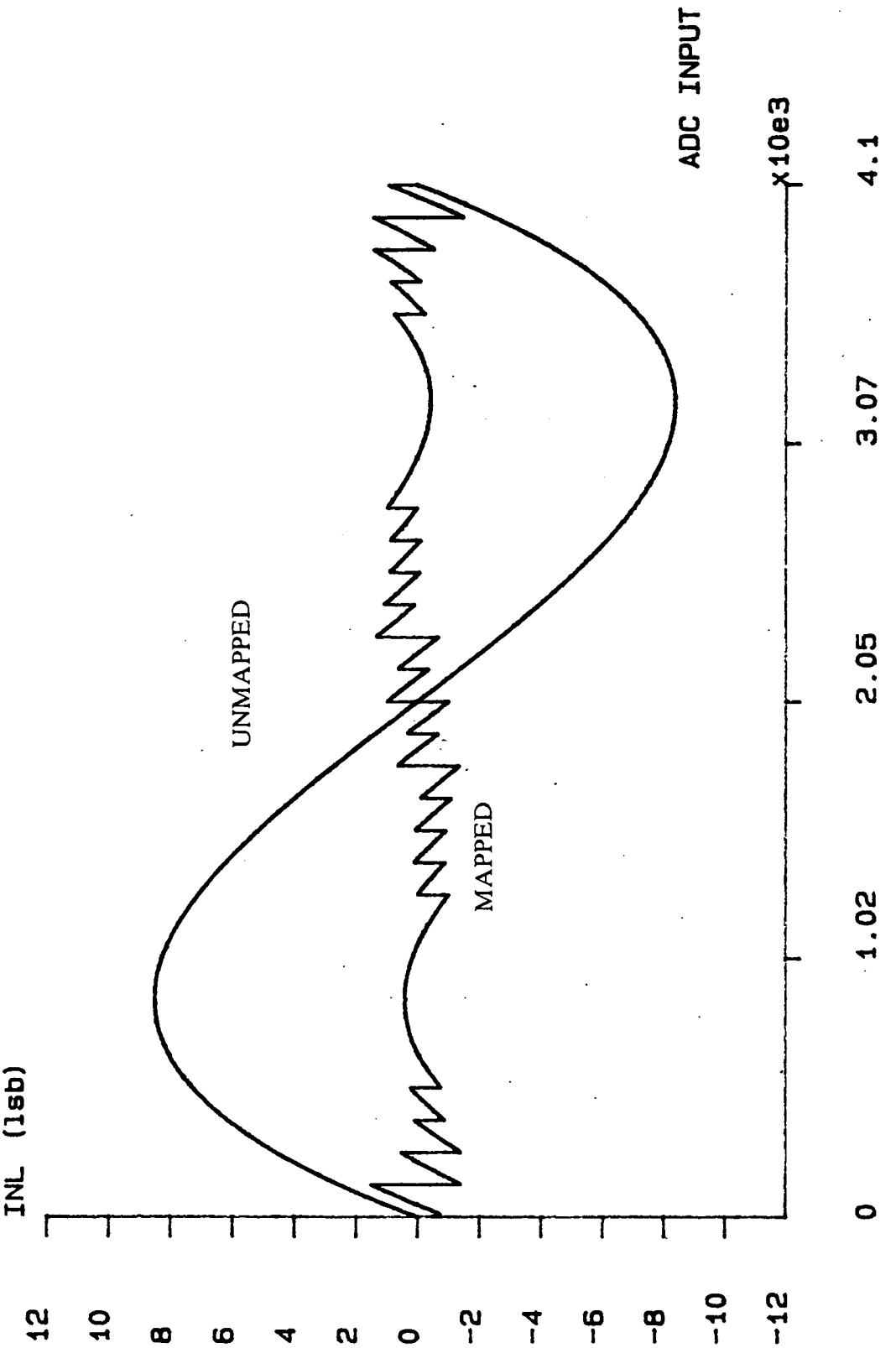

Figure 4.7: The INL of a 12 bit ADC with a simple nonlinearity and a 5 bit mapping.

optimum number of addressing bits, the mapping has clearly given very good results, considering that only 5 instead of the indicated 8 to 10 addressing lines have been used.

When the 9 most significant address bits are used, then the INL performance improves further, such that in mean deviation terms it is no longer suboptimum, and its maximum levels are now more or less ± 0.5 LSB, as in a full resolution mapping. This situation is represented in figure 4.8 and indeed, it would be extremely difficult to discern any difference between this and a fully addressed INL plot by eye alone.

Certainly the use of 10 bits (or even 9 as in the plotted result) gives very good INL performance, especially for the easier distortions. The more complex nonlinearities, though, may cause more problems, especially at points of high differential nonlinearity.

When these more difficult distortions are mapped the results are generally good, although lower resolutions of mapping tend to give much poorer quality results than with simple nonlinearities. When only 5 bits are used to address the look-up table, as in figure 4.9, the overall maximum INL level is only reduced from 11 LSB to perhaps 5 LSB. This is nowhere near as good as in the simpler nonlinearity of figure 4.7, and represents an increase of 7.7 dB in the mapped mean deviation, compared to full resolution. The sharp vertical lines in the mapped transfer function indicate where mapping discontinuities have occurred.

When the mapping is performed with 9 address lines the mean deviation increases by only 0.7 dB from that of a full resolution mapping. As can be seen from figure 4.10 the INL is now bounded between approximately ± 1 LSB, and is usually smaller than these limits. Clearly an increase to 10 addressing bits will allow improved mean deviation as well as better integral nonlinearity.

As predicted, it can be demonstrated that 10 bit reduced resolution addressing gives

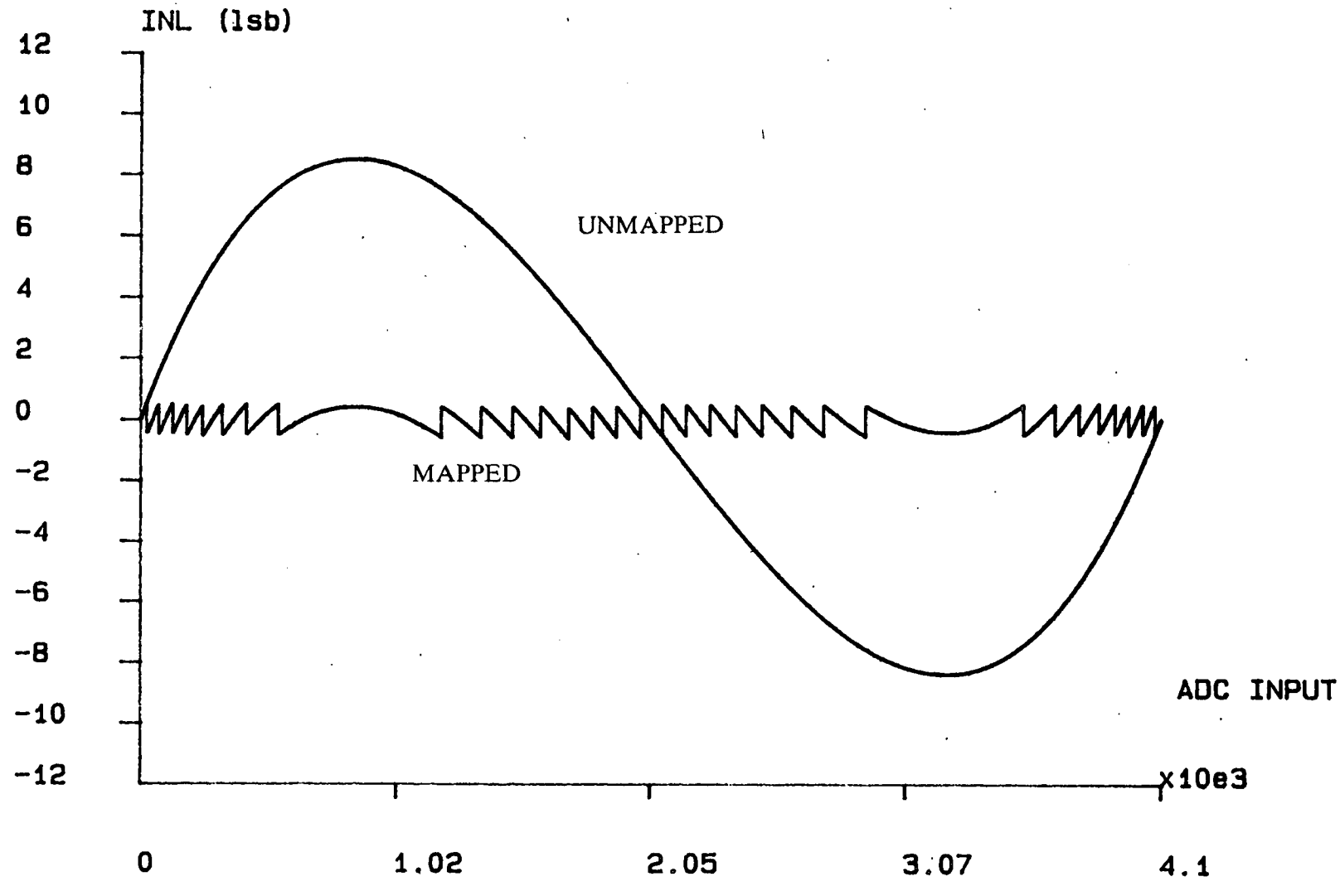

Figure 4.8: The 9 bit mapping of the ADC distortion of figure 4.7.

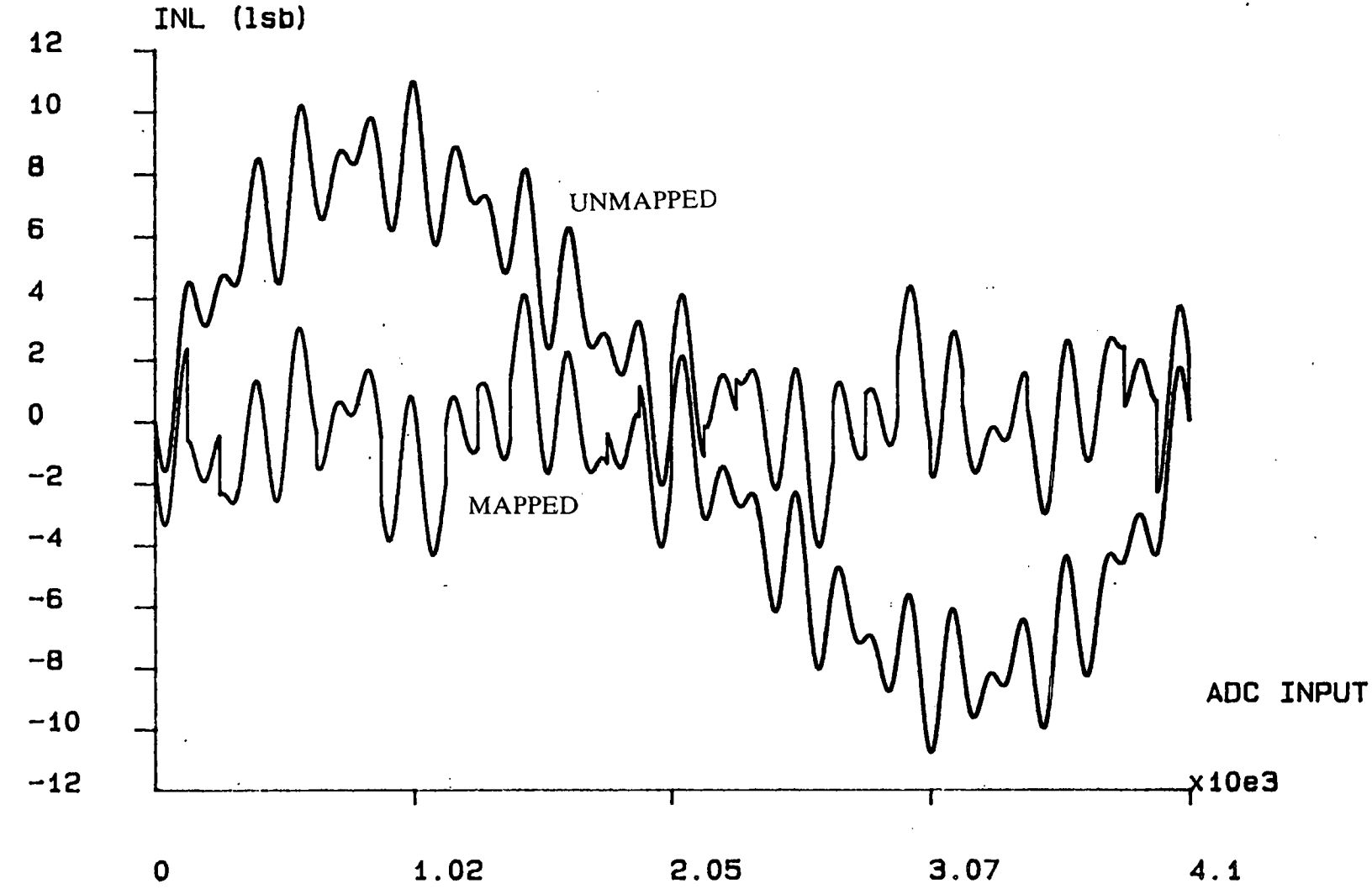

Figure 4.9: A complex nonlinearity mapped with 5 bit resolution.

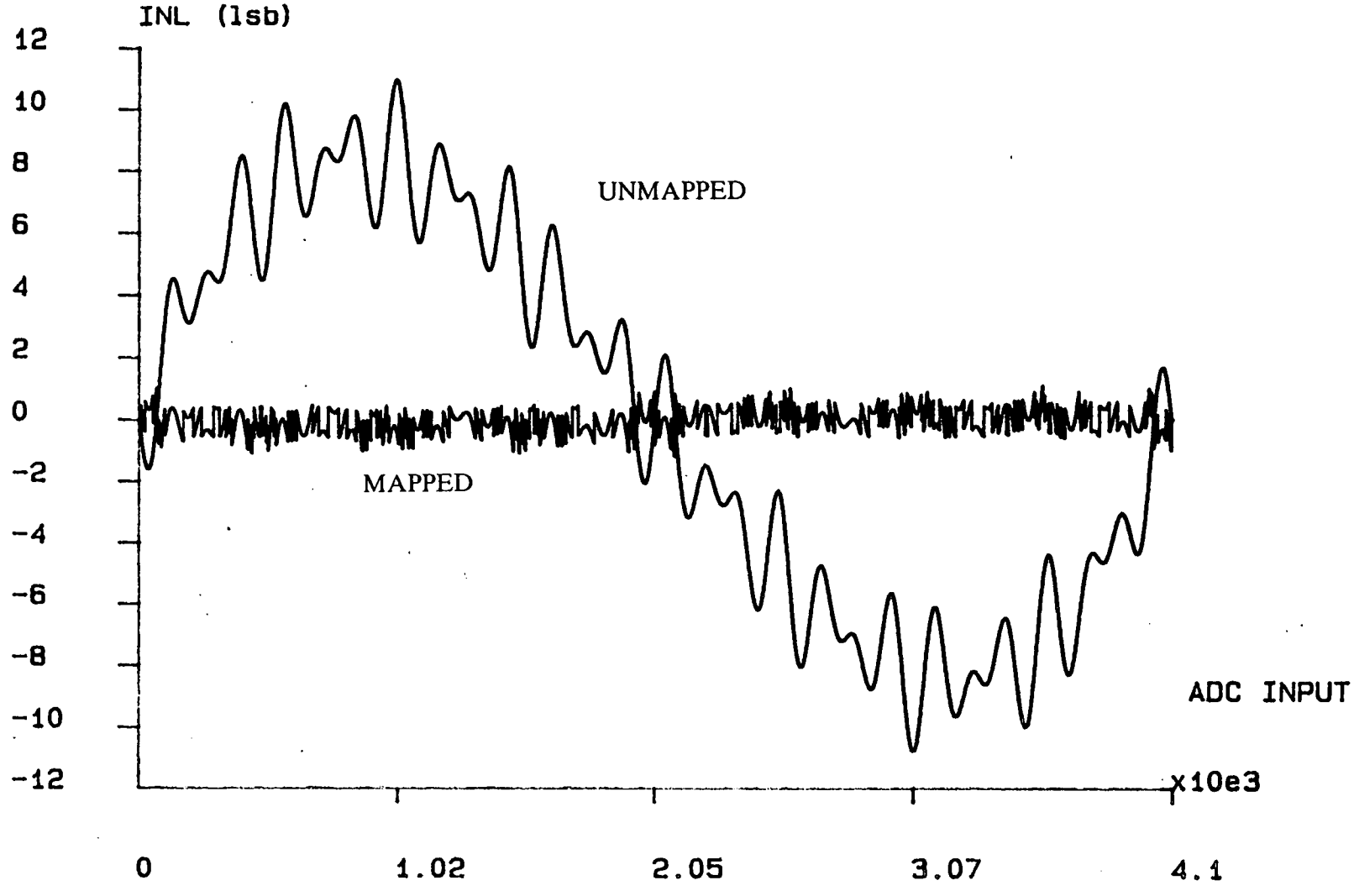

Figure 4.10: The previous complex nonlinearity mapped with 9 bit resolution.

a very good quality transfer function mapping, with a decent margin of safety to enable the overall system to handle any extreme distortions.

4.5.2. Reduced Resolution and Spectral Distortion

The Fourier transforms of reconstructed signals of data from 16 bit ADC's generated the normal type of spectra, with poor mappings leaving some harmonic distortion and intermodulation products, as well as raised noise floors. Figure 4.11 shows an unmapped coverter which has one of the easier simulated transfer function distortions. Of most importance in the figure are the large size and number of harmonically related peaks. In figure 4.12, the full resolution mapping of the nonlinearity of figure 4.11, can be seen. This has been included to give an indication of the mapping procedure's maximum possible performance level, for use when comparing the results of lower resolutions. The figure shows little remnants of the previously dominant distortions, except perhaps for the small feature to the right of the higher frequency fundamental.

When the converter is mapped with 5 address bits, which is equivalent to a reduction in mean deviation performance of 1.4 dB, the remaining spectral distortion is very small, as can be seen from figure 4.13. Here only some tiny vestiges of the original peaks remain, with the spike adjacent to the high frequency fundamental being the most important of these. Otherwise, hardly anything is left of the initial nonlinearity. As with the INL tests, the use of only 5 address lines has given a satisfactory solution to the problem.

When 9 bits are used, the outcome of the Fourier transform becomes almost indistinguishable from that of the fully addressed mapping. This plot is given in figure 4.14, and confirms what was found from the earlier INL tests, that approximately 8 bits are necessary to map the more simple transfer function distortions to a satisfactory level.

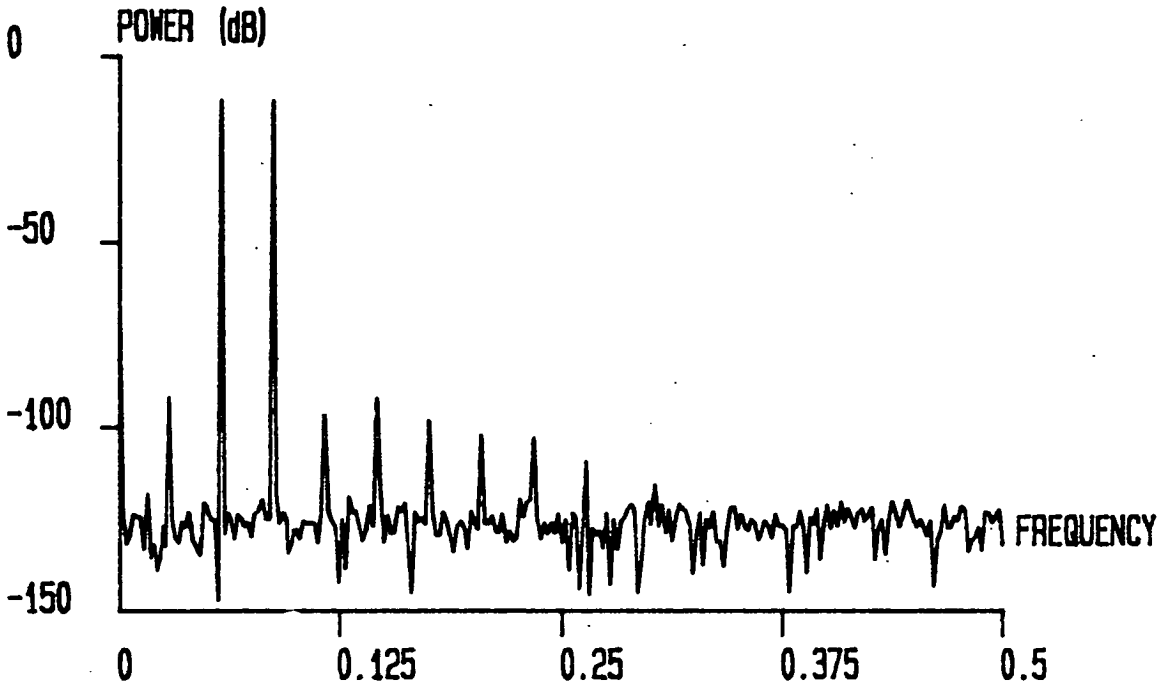

Figure 4.11: The Fourier transform of an ideally reconstructed two tone sinusoid sampled by a 16 ADC with a simple nonlinearity.

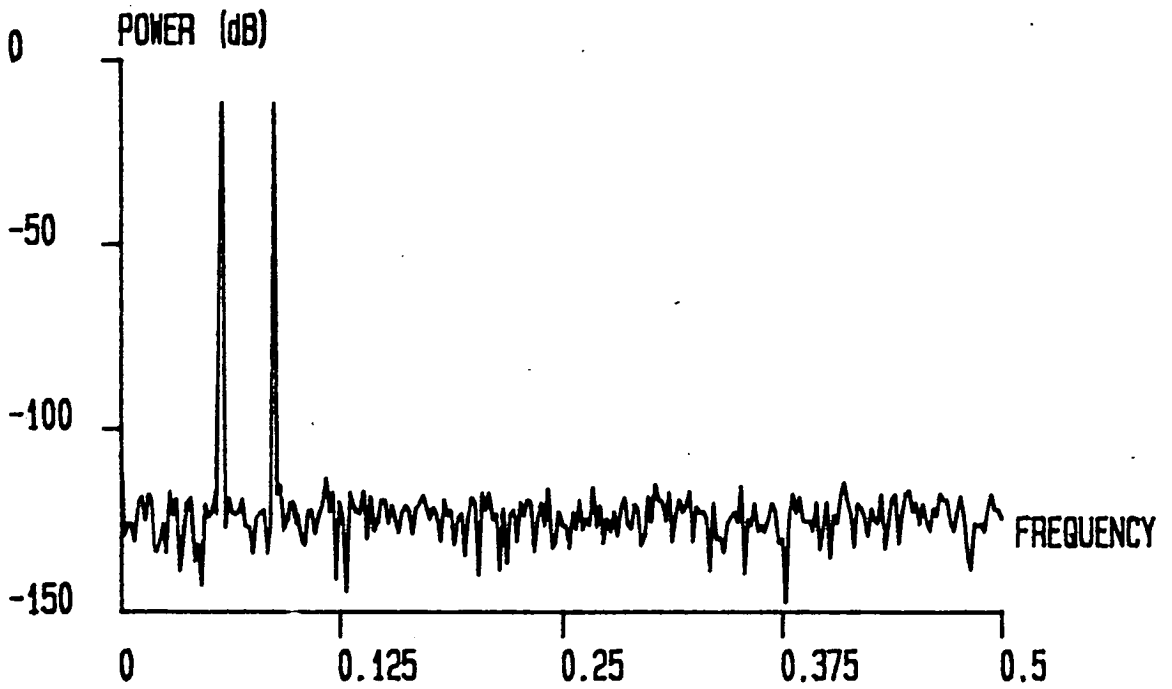

Figure 4.12: The full resolution mapping of the distorted spectrum above.

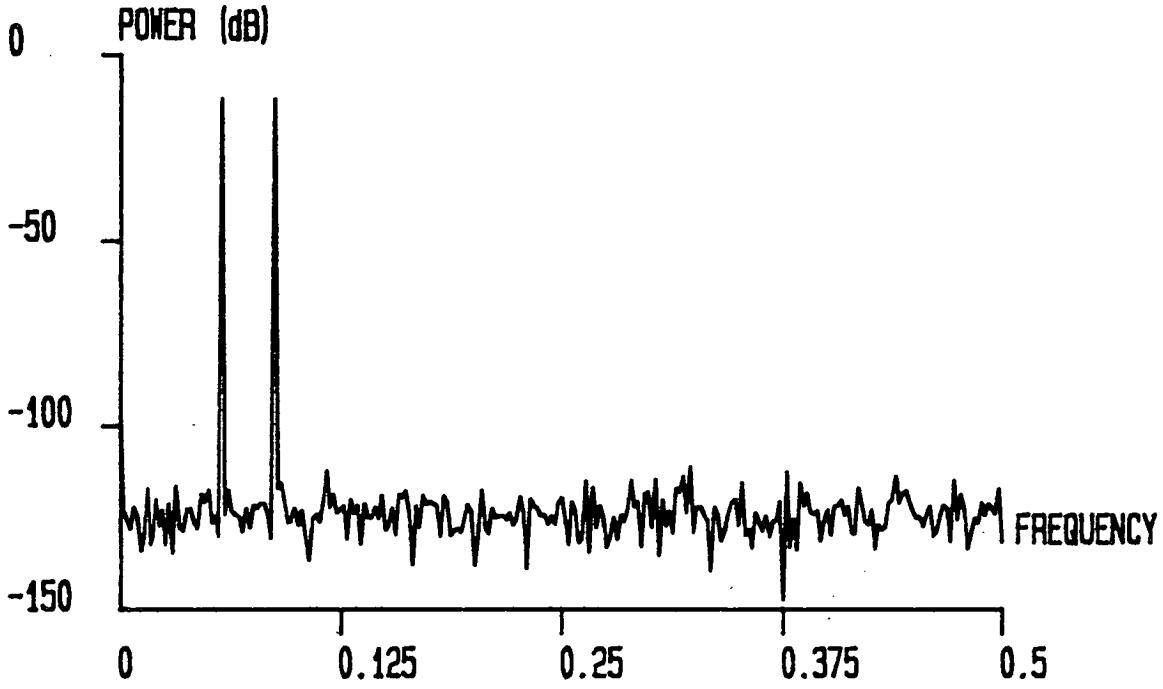

Figure 4.13: The 5 bit resolution mapping of the nonlinear ADC of figure 4.11.

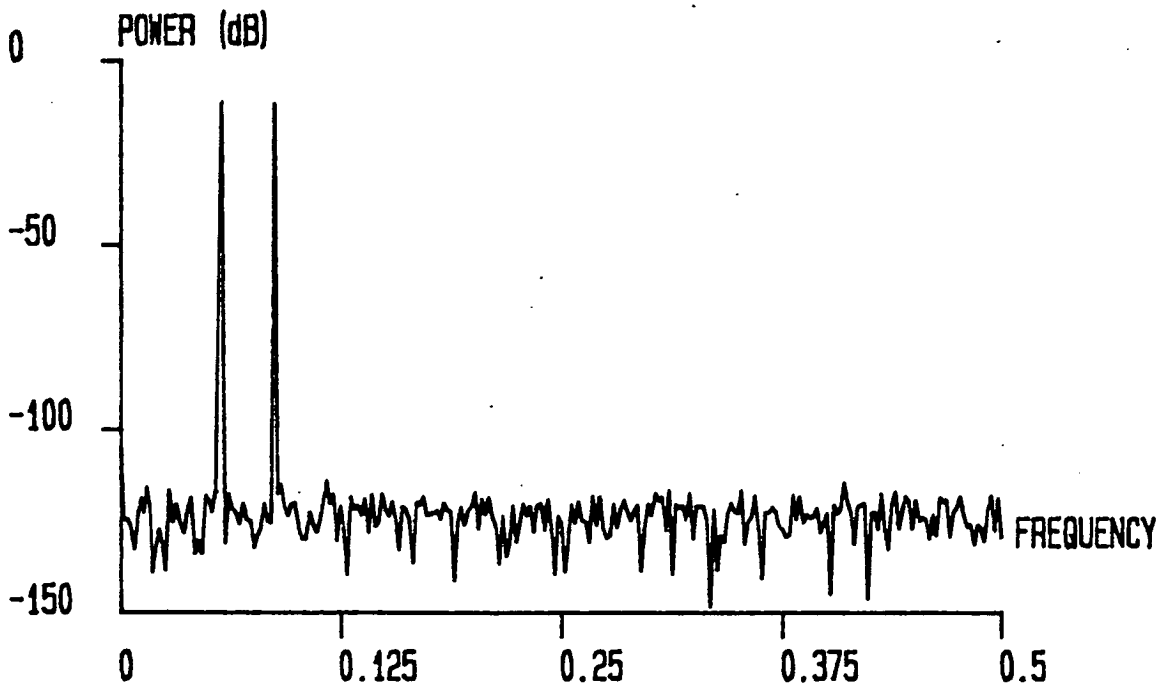

Figure 4.14: The 9 bit resolution mapping of the simple distortion.

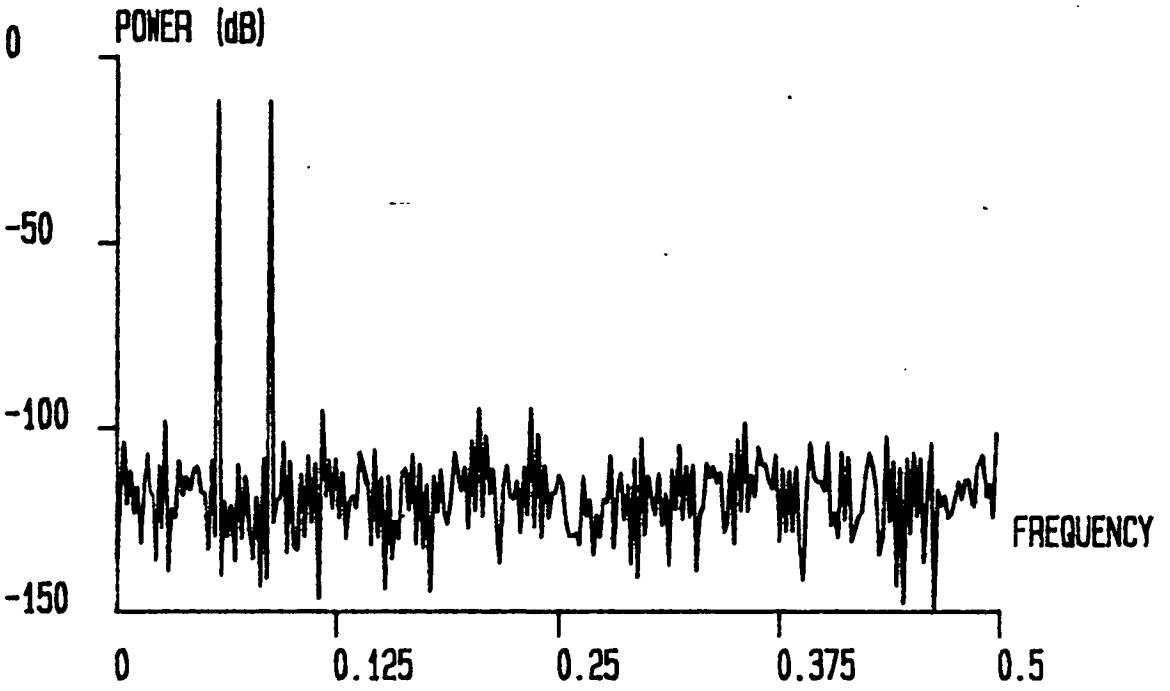

Figure 4.15: A sampled spectrum which has been distorted by being sampled with a 16 bit ADC which has a complex distortion.

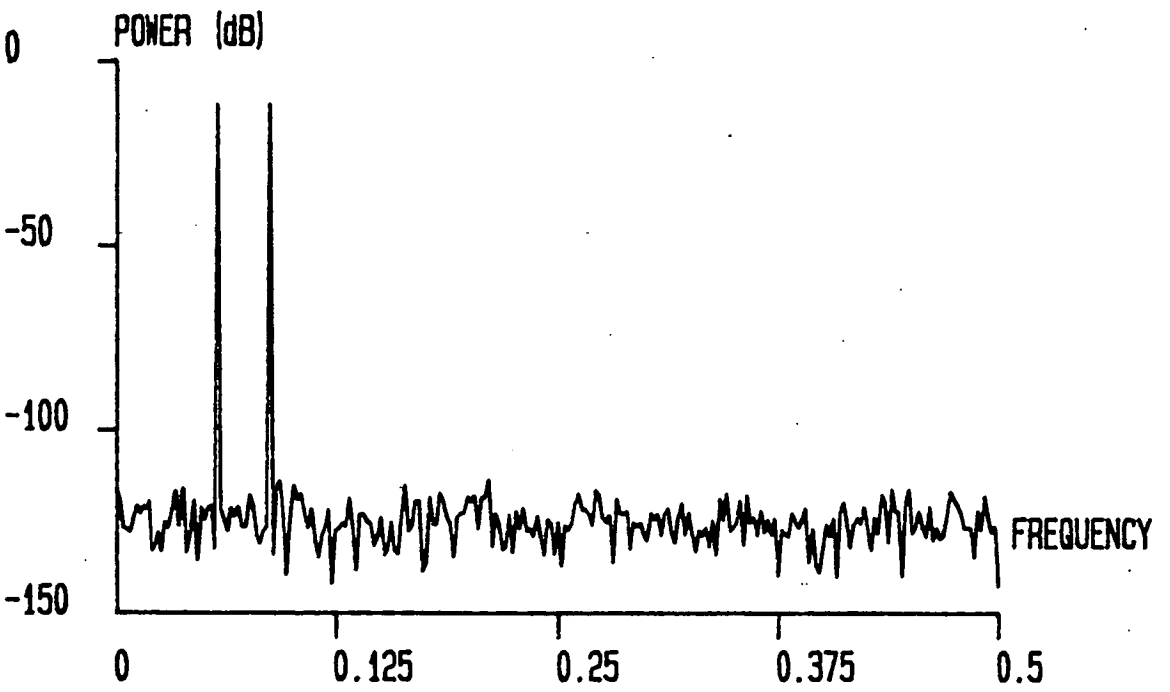

Figure 4.16: The full resolution mapping of the above spectrum.

A much more difficult nonlinearity to map has been used in the simulation results given in figure 4.15. Here the addition of periodic components into the distortion has brought up the level of the general converter noise floor, such that it almost dominates the harmonically related peaks. A full resolution mapping, as shown for reference in figure 4.16, completely gets rid of both the peaks and the increase in the level of the quantisation noise floor.

When only 5, out of the possible 16, output bits of the ADC address the look-up table, then the slight improvement over unmapped performance shown in figure 4.17 results. With the mapped transfer function's mean deviation being 7.6 dB short of its maximum, no really significant gains can be expected. However, most of the dominant peaks have been eliminated and it is only really the raised noise floor which is still giving problems.

The use of 9 bits gives the plot of figure 4.18, where only a single spike at the quarter sampling rate point remains of the initial unmapped distortion. Not only have all the other harmonics been removed, but the general level of the noise floor has been lowered, as well as its general appearance being smoothed out. Although with a mean deviation of 0.8 dB more than full resolution would give, the 9 bit mapping is by no means perfect, it does give a very useful level of linearisation.

This all implies that the 10 bit addressing level suggested earlier, is probably sufficient to deal with all kinds of transfer function distortion likely to be encountered.

4.6. Conclusions

It has been shown that the basic technique of mapping the transfer function of a nonlinear ADC to a more linear shape by the use of a look-up table, can not easily be implemented in hardware. This is due to the large amount of memory required, especially for high resolution converters, no matter what kind of mapping storage medium is used. Two methods of reducing the amount of storage necessary have

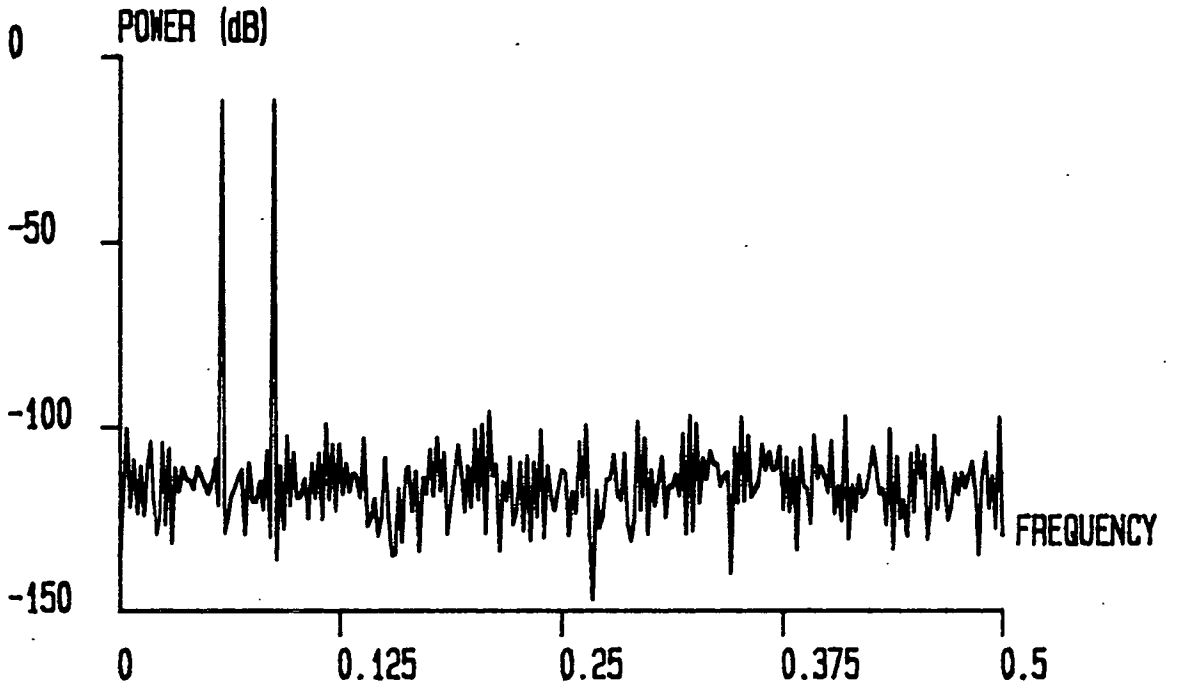

Figure 4.17: The 5 bit mapping of the distortion of figure 4.15.

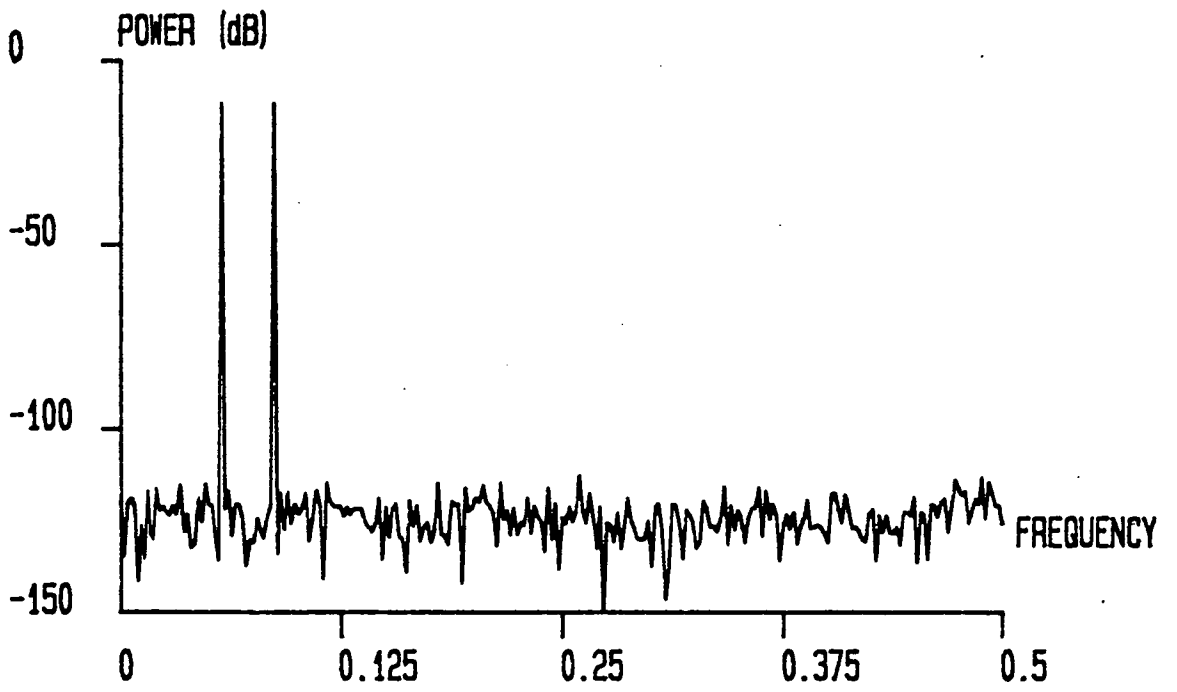

Figure 4.18: The 9 bit mapping of the same complex nonlinearity.

been discussed.

Firstly, the storage of the mapping offsets only, and the introduction of a digital adder into the structure of the linearisation system. This allows a reduction of the memory requirement by a factor of three or more, for high resolution converters.

Secondly, the idea of partitioning the ADC transfer function into many small segments, for which the mapping is assumed to be continuous. This uses only the hardware needed for offset storage, providing that there is 2^n segmentation of the converter transfer function. This type of segmentation allows the most significant bits of the ADC output code to address the look-up table directly, without the need for any intermediate logic to determine which offset should be accessed.

The results obtained for mean deviation, integral nonlinearity and spectral distortion compared well with those for converters mapped with a fully addressed look-up table, assuming that 10 address bits were used. This allowed a good margin of safety, even for the most severe of mapped distortions, whereas with simpler nonlinearities, even far fewer than 10 bits gave near optimum performance.

4.6.1. Overall Storage Saving

The savings which these results reflect can be demonstrated by an example. A full resolution 16 bit ADC transfer function mapping requires $2^{16} \times 16 = 1024$ kbits, which is reduced to 320 kbits if only 5 bit offsets are allowed. When only 10 bits are used to address the mapping look-up table, then only $2^{10} \times 5 = 5$kbits are needed, which represents an overall saving of almost a Megabit of storage, and a new memory requirement of less than 0.5 % of the original. Obviously, as the resolution of the ADC is increased, the saving will get bigger, as the total storage needed remains constant at 5 kbits. This is assuming that 5 bit offsets and 10 address bits are sufficient for the overall linearisation system requirements at higher resolutions.

This small amount of memory is easily realisable on a single external integrated circuit, and would also not be too difficult to integrate onto the same piece of silicon as the ADC, if processing allows.

Chapter 5

A HARDWARE IMPLEMENTATION

It was seen as highly beneficial to the overall project if a hardware demonstration of the work previously completed could be made to show that the system was easily implemented, would work in real time, and most particularly that it would work with real ADC nonlinearities.

5.1. Aims and Objectives

To these ends it was necessary to develop some hardware which would implement the mapping look-up table. This would have to be addressed by an ADC with a nonlinear characteristic which would preferably be adjustable so that a wide range of ADC transfer function nonlinearities could be examined. A pdf of a well defined shape would also have to be generated by some method or else an unknown pdf accurately measured so that linearisation could be performed. In view of the known difficulties of generating such a well defined signal it was decided at an early stage that measurement of the pdf of a reasonable signal was the better solution.

Some facility to run the threshold tracking mapping algorithm was also required either in terms of an independent processor under programmable machine code control which could be used to control the mapping, the ADC's and ancillary circuitry, or to interface a stand alone computer to a hardware system and to run the mapping algorithm and other routines under software control, using a high level language.

This second method was clearly advantageous as it allowed easier debugging of software, simpler circuitry, no necessity to learn how to program the processor directly and also gave access to simple methods of visually displaying results. i.e. the monitor and line printer/plotter.

The type of results which it was desirous to have were of several forms. It would be very useful to have some representation of the shape of real life nonlinearities as simulated nonlinearities have previously been of a geometric form. The shape of these real nonlinear characteristics could be generated fairly easily from the information contained in the mapping function or from the sampled nonlinear pdf. Some analysis of the harmonic distortion of sampled signals before and after mapping is essential to confirm the earlier simulated results. Further to this, the signals could be reconstructed using a good quality DAC before being fed to a spectrum analyser, where an extremely convincing real time demonstration could be provided of the linearisation system's capability to improve signal quality and purity.

Lastly an impression of what particular distortions would do to the sampled pdf's would be interesting and would allow visual analysis of the distorted pdf's to be carried out in the future.

What the above stated aims would then require in terms of hardware was as follows.

1) An ADC of adjustable nonlinearity. This clearly could not be bought off the shelf and would have to be designed and made from scratch.

2) A very good quality ADC to accurately measure the undistorted pdf of the training signal.

3) Circuitry to implement the mapping.

4) Some form of interface of the rest of the circuitry to a computer, to be used to control the linearisation process.

5) A good quality DAC, of equal resolution to the ADC's, to be connected to the digital output of the mapping memory.

5.2. Implementation

5.2.1. An ADC with Adjustable Nonlinearity.

It was quickly realised that the simplest way to develop this circuit would be to devise a successive approximation type ADC using a Successive Approximation Register (SAR) integrated circuit to perform all the decision logic and supply the digital output signal. A 12 bit TTL device, the Am2504, was found to be suitable for this purpose. Whilst a higher resolution device would have been preferable, for demonstration of the power of the technique, using 12 bit hardware made design of the ADC interface and linearisation circuitry easier, and cheaper to build.

The use of an R/2R ladder made up from variable resistors of nominal values of 50 kΩ and 100 kΩ respectively was found to be a suitable reference for the DAC part of the Successive Approximation converter structure. A high speed comparator and a good quality Sample and Hold (S/H) were also needed to complete the circuit configuration.

A custom designed PCB was fabricated (with a ground plane) on which to mount the ADC circuitry and provision was made for the attachment of power supplies and data/control lines. The converter was found to work well with a 200 kHz clock and a 10 kHz sampling rate.

A circuit diagram of the ADC circuitry, excluding power supply decoupling, is presented in Appendix B.

5.2.2. Reference ADC

As the requirement for the linearity of this ADC was very high it was thought sensible to utilise a higher resolution, high specification device. To this end a 14 bit self calibrating ADC was purchased; the Crystal Semiconductor CS5014. This circuit

has an on chip Sample and Hold and hence requires only one external control line and a clock signal. A Schmidt trigger inverter was used in conjunction with an R-C network to initiate reset and autocalibration of the ADC on power up, as recommended in the manufacturer's databook [9]. After this process the converter was specified to have a linearity of approximately 15 bits, which was 3 bits more than the resolution of the poor quality ADC. This ensured that the pdf of the training signal was very accurately measured.

A high precision 5V reference was also needed for accurate functioning of both the self calibration sequence and the normal operation of the ADC. This was originally specified as a single IC, an AD586, but was later replaced with a pair of 2.5V reference diodes of the ZN458 type. This was due to a failure of the original integrated circuit which required the alteration of the reference source as there was no immediately available replacement.

The external clock rate for this ADC was 1.25 MHz and a circuit diagram of the CS5014 and support circuitry, excluding power supply decoupling, is given in Appendix B. This ADC was also mounted on a custom designed PCB with a ground plane, again with provision for the necessary I/O lines and power supply connections.

5.2.3. Interfaces and System Configuration

One of the most convenient ways available of interfacing a computer to a hardware system was to use a computer controlled 19" rack system, developed within British Telecom Research Laboratories by group RT4231. This system consisted of a Hewlett Packard 200 series computer, a rack for double sized Eurocards, an extensive backplane, self contained power supplies and cooling fans. There existed within RT4231 a large number of circuit cards designed to operate in this environment with associated software to drive them, as well as to perform other general purpose signal processing functions. This collection of equipment was

known as the Digital Subscriber Loop Development System or DDS. The timing of the backplane, control signals and the circuitry for general purpose card interface was already well defined and hence, the DDS was seen to offer large advantages in terms of development time and ease of operation. Amongst the large amount of ancillary equipment already in existence was a high quality, fast settling (400 nS) 12 bit DAC, which was well suited to the linearisation system's measurement requirements.

Using the DDS as an interfacing and control system would also allow the mapping memory, ADC control logic and clock signals to be generated on the same card. The overall system layout is summarised in figure 5.1, where :

[1] is the Network Communications Controller Card (NCC) which handles backplane signal timing and general I/O to the controlling 200 series computer.

[2] is the PLL card which generates or buffers fundamental clocks and supplies some other logic signals.

[3] is the threshold tracking mapping card (TTM) which performs all the linearisation system interfacing requirements as well as containing and controlling the RAM based look-up table for mapping storage.

5.2.4. The TTM Card

This was where the bulk of the design effort was to be centered as this card was the largest and most complicated circuit element in the complete system. The card design could be divided into several parts, the interface to the DDS backplane, the memory look-up table and associated control, the ADC port and clock/control signal generation and lastly the DAC port and control circuitry. A block diagram showing the high level interconnection of the TTM card circuit elements is shown in figure 5.2, where the routing of clock and timing signals to most of the blocks has been omitted for clarity. A circuit diagram of the overall interconnection of the TTM card components, excluding the DDS interface, is given in Appendix B.

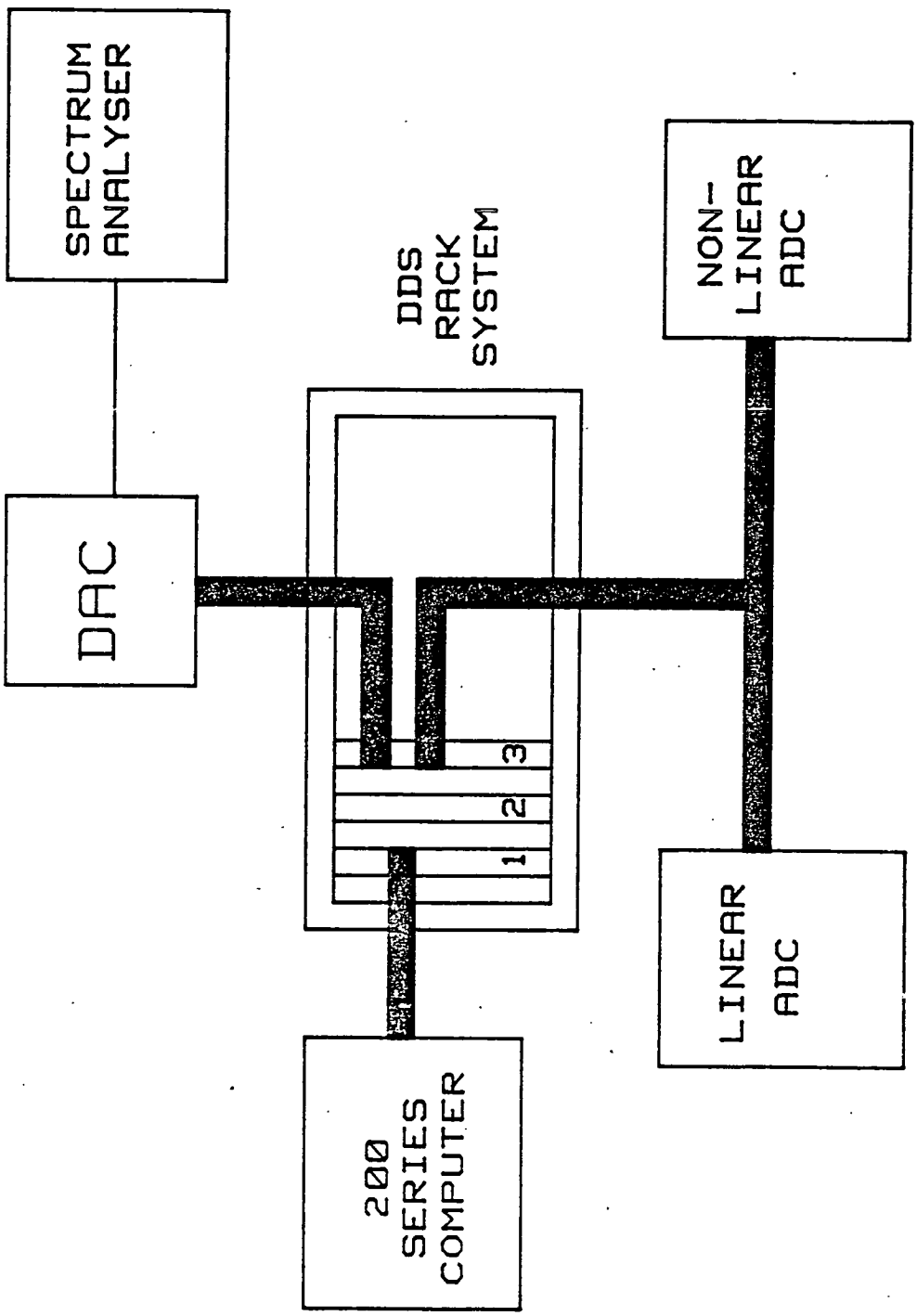

Figure 5.1: The layout of the combined hardware systems.

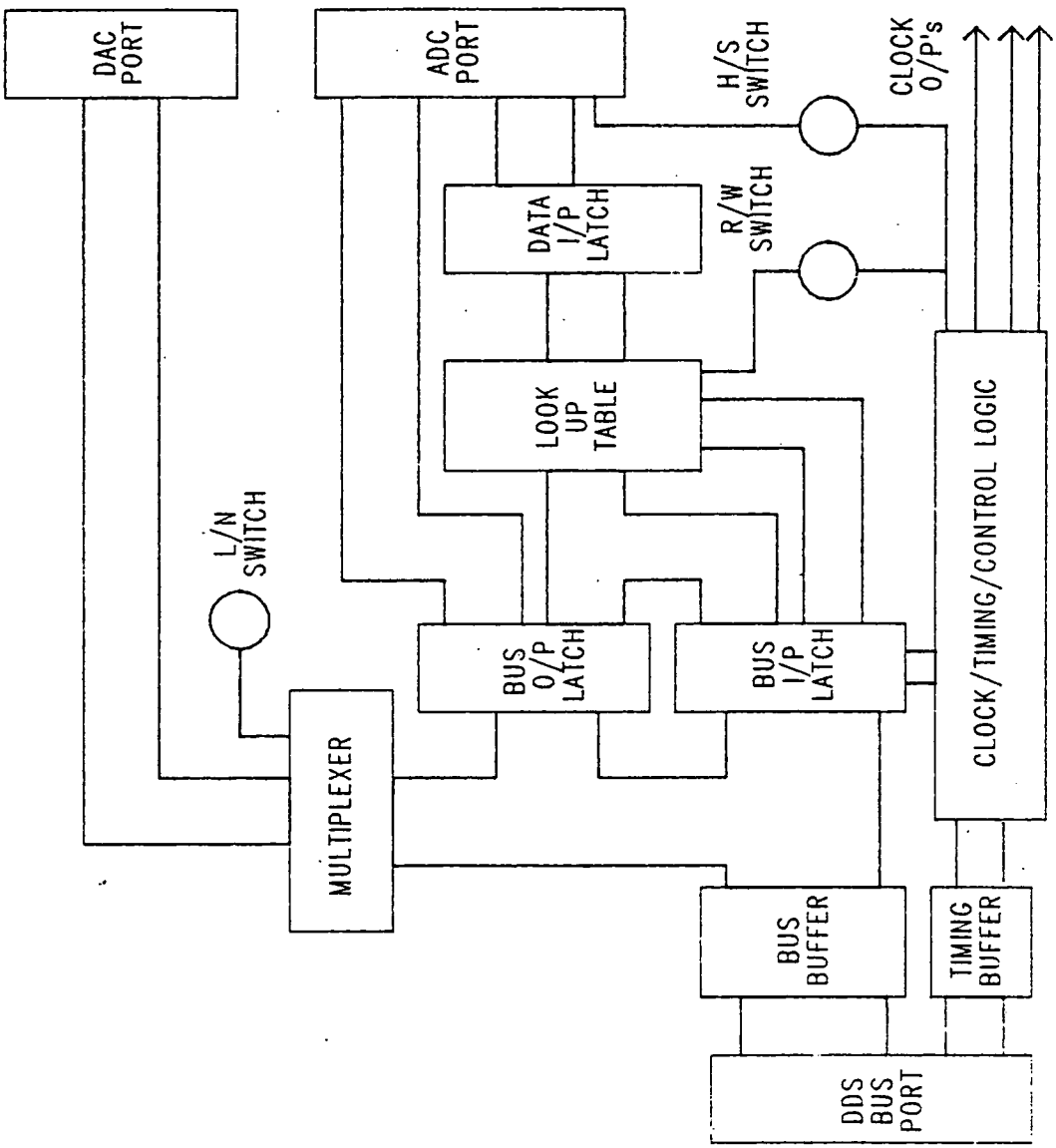

Figure 5.2: The TTM card high level interconnect.

DDS Interface

The TTM card circuitry can only be described if the timing scheme of events on the DDS backplane is understood in some detail. A full description of these protocols is given in [85] but a sufficiently detailed account of their use in the implementation of the TTM card is given in Appendix B, which also contains a full description of the card's interface circuitry.

Look-up Table Hardware and Control

The look-up table was implemented as a 4096 × 12 bit store formed by three, 4096 × 4 bit static RAM's addressed in parallel. The address inputs came from the input data latches, so that the memories could be loaded with the required mappings, and from the ADC input data latches, so that the mapped code could be read out and transferred to the DDS backplane.

As has been suggested above, the data lines of the static RAM's were connected to both the output data bus driver latches and the bus data input latches, as data had to be both read from, and written to, the memories. This use of the same connections for data input and output meant that care had to be taken to ensure that there was no contention on the memory "data bus" and in particular that both sets of latches addressing the the memories were not output enabled at the same moment in time. A more detailed examination of this problem is given in Appendix B.

ADC Port and Control

Timing control of the ADC's, their associated sample and hold commands and logic resets was quite complicated due to several operational difficulties.

Firstly, due to hardware limitations imposed by the NCC card in use, only a limited amount of data could be gathered at any one time. This meant that there were long periods of inactivity whilst this data was being uploaded to the controlling

computer.

Secondly, for the spectrum analyser to be used the ADC's would have to be able to sample the test signal continuously over a long period of time. This meant that the ADC control signals would have to be duplicated by the card's hardware so that they could be generated independently of the computer.

These and other problems were overcome and a detailed explanation of their solutions and implementations is given in Appendix B.

DAC Port and Control

Similarly to the ADC port, the DAC port also required timing signals and connection to the digital data from the linear ADC and the mapping look-up table. This circuitry is described in Appendix B.

5.3. Software and System Control

The software needed to linearise the particular nonlinearity with which the adjustable ADC had been set was contained in the program file "ADCL_FINAL", so called because it was the FINAL version of the ADC Linearisation programs. It was written in Hewlett Packard BASIC version 2.0, and a listing of the program is given in Appendix C, and an overview of the software and an intimation as to how it is used is given in Appendix B.

5.3.1. General Purpose Subroutines

The program had to be able to control the signals being sent down the DDS backplane on each relevant timeslot and, likewise, be able to read those transmitted by the TTM card. This control was achieved by the use of a number of subroutines to operate the NCC card in the modes required. These subroutines were collectively known as "NCC_SUB_8K" and they consisted of routines to download an 8K long vector of 32 bit data to the NCC card, upload an equivalent 8K vector of data logged from the DDS backplane, and to set up the NCC card, in terms of which

transactions were to be used and how often to log/sequence bus data.

As one of the main measures of the linearisation system performance was to be spectral analysis of sampled sinusoidal signals, it would be necessary to have Fourier analysis performed by the software. To this end three utility subroutines were used which were capable of performing forward and inverse Discrete Fourier Transforms (DFT's) on arrays of floating point numbers. One main routine accomplished the DFT whilst the other two converted the data format of the input and output arrays to a form that was usable/understandable by the DFT routine and the rest of the program respectively.

5.3.2. Special Purpose Subroutines

In addition to the routines described above, other subroutines which would only be of use in the ADC linearisation programs, also had to be written. These included subroutines to load the 8K vectors of sequencing data into the correct arrays (this data was principally the addressing and mapping data for the TTM card look-up table and that to generate the ADC control signals), to extract every fourth data element from the uploaded 8K array and to form them into a pdf of 12 bit samples, to prepare the 12 bit data in the uploaded vector into 16 bit numbers to be passed to the DFT routines, and lastly, the routine to generate the threshold tracking mapping.

Also required were several pieces of code to produce graphical representations of data and program results on the computer's VDU, from where the screen could be dumped to a printer, but these were not written as subroutines for the sake of simplicity.

5.3.3. Use with the Spectrum Analyser

When the spectrum analyser was being used to demonstrate the relative performance facilitated by the linearisation system, the time taken by the analyser to complete a

full frequency scan accurately, was found to be several orders of magnitude greater than the time taken to sequence all 8K control signals. It was therefore necessary to use the hardware generated timing signals to obtain the continuous, real time ADC sampling that the spectrum analyser needed for the quality of measurement required. This could only be done when nothing was being written to or read from the TTM card i.e. when the program was sitting in one of the many wait states. Obviously however, the output of the analyser was only useful at certain points in the program, but these were simply related to what was going on in the software at the same time.

Obtaining spectral analysis of the test signal being sampled by the linear ADC was simply a matter of switching the system multiplexers so that the alternative input was directed to the DAC.

One program then was able to control the operation of the TTM card in conjunction with the three manual switches, on the circuit board, whilst providing all the data output representations required.

5.4. Computed Results

In this section results will be presented which have been generated from the data gathered by the system's interface to the TTM card, and hence the transfer functions shown are not measured but calculated, as are the Fourier transforms.

5.4.1. Measured Pdf's

One of the most important aspects of developing and testing a real-time hardware implementation of the linearisation system was that the technique's dependence on good quality training signals and hence, pdf's, could be evaluated. The training signal used during the system test was usually a low frequency (less than 1 kHz) ramp, triangle or sawtooth waveform. The choice of waveform was found to have a negligible effect on the pdf's sampled from it. Figure 5.3 shows a typical normalised

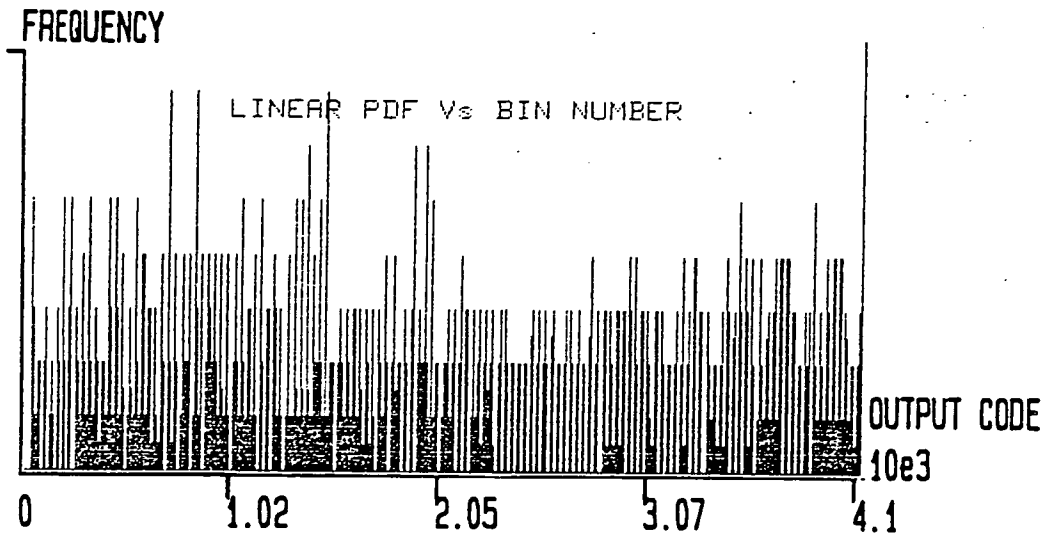

Figure 5.3: A typical normalised linearly sampled pdf of 2K samples.

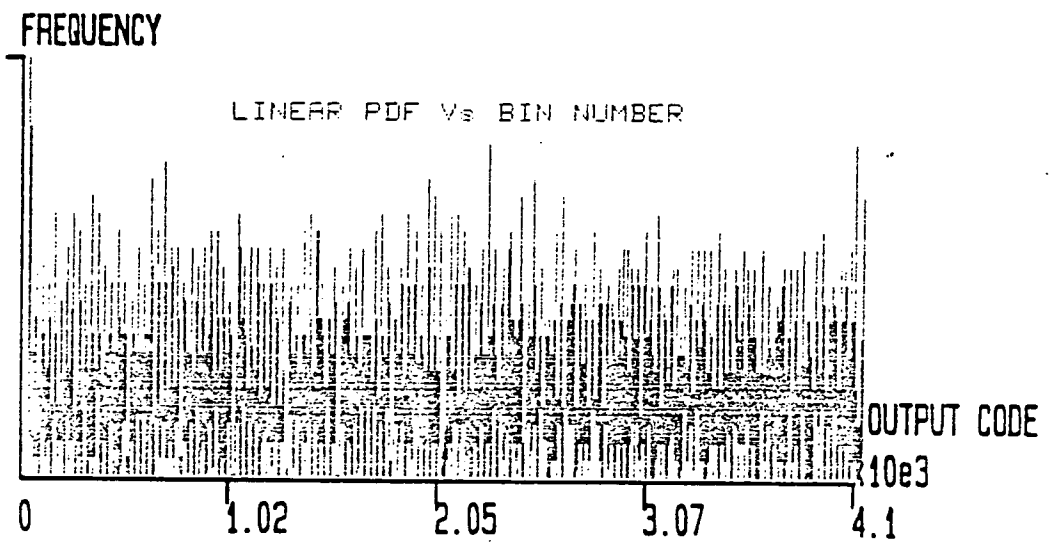

Figure 5.4: A normalised linear pdf of 16K samples.

pdf generated by taking two thousand samples from the linear ADC whilst using a ramp test signal.

Even with the coarse resolution of the plot it is possible to see that a large proportion of the pdf histogram bins are empty and that those that are not empty often contain more than one sample. This can be regarded as a purely statistical process as the mean code density for two thousand samples is only 0.5. When a larger sample size is taken these effects are smoothed out as can be seen in figure 5.4 where sixteen thousand samples have been used to form the pdf. Similar effects can be observed in the signal pdf's sampled by the nonlinear ADC. The net outcome of using sample sizes which previous simulation results would tend to categorise as sub-optimum seemed to have little or no visible effect on the shape of the mapping produced. This was partly due to the lack of resolution offered by the computer graphics and the extremely small sizes of the actual differences, but was also likely to have been affected by the use of the precisely known linear pdf, rather than just assuming that it had a precise mathematical form i.e. that all the histogram bin frequencies are equal for a uniform pdf. These slight differences between what the simulations assumed, and were actually using, for the linear pdf may account for any perceived improvement in the system's performance.

5.4.2. Actual Transfer Functions

After the nonlinear ADC had been calibrated by setting all the resistors in the R/2R network as close to their correct values as was worthwhile with a multimeter, and initial tests had been run to ensure the correct operation of all the system components, the job of presenting the test signal with specific nonlinearities began. This involved changing the values of the components of the reference ladder structure relative to each other. This procedure generated a large number of nonlinearities of differing shapes and sizes, some of which are included here. Of more interest were the effects produced by more systematic changes in the resistor

values, and it is these that will now be examined.

5.4.3. Pdf Gaps

By reducing the relative values of the 50 kΩ, R resistors, noticeable changes occurred in the sampled nonlinear pdf. Small gaps started to appear in the basically uniform code densities. These were regularly spaced and seemed to correspond to the more significant bit transition positions, with an additive effect when one or more transition overlapped i.e. the gaps became bigger, at the mid and quarter scale points, etc. As the size of the nonlinearity was increased so did the size and number of the gaps. This would seem to suggest that the accuracy of the more significant bits had more effect on the overall linearity of the converter than the lesser bits, which agrees with common sense. As the size of the nonlinearity was increased still further some of the pdf histogram gaps associated with the more significant bit transitions became so large that they overlapped with their neighbours, giving rise to extremely large gaps. This situation is shown in figure 5.5 which shows the nonlinear sampled pdf, when the R resistors are set to around 50-70% of their correct value. If this process is continued further, the gaps start to dominate the pdf such that only a few relatively large peaked regions remain.

As the mapping produced by the threshold tracking algorithm is directly dependent upon the measured nonlinear pdf, these pdf gaps must affect the shape of the ADC nonlinearity in a dramatic way. This can be seen in figure 5.6 which is the transfer function mapping generated for the pdf of figure 5.5. It is readily seen that where there are gaps in the pdf there are corresponding horizontal segments in the transfer function of the mapped device. This can easily be traced back to the cumulative distributions of the sampled signals and their relationship in the mapping algorithm. The maximum deviation of the transfer function shown in figure 5.6 from the ideal is very large as would be expected for the magnitude of the relative errors in the R resistors. This deviation would seem to correspond to about only 4 to 5 bits of

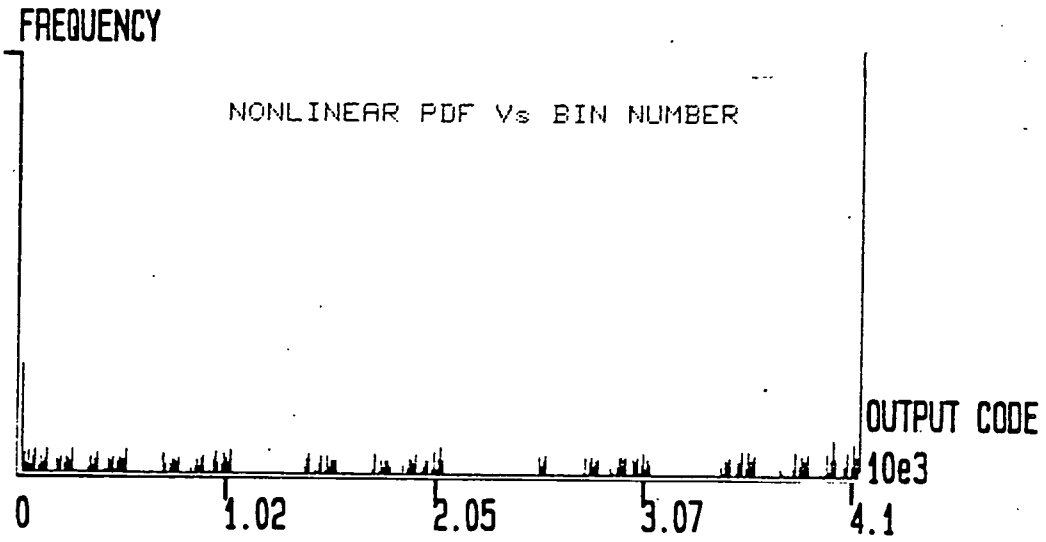

Figure 5.5: A pdf sampled by the nonlinear ADC when the R resistors are up to 50% out of adjustment.

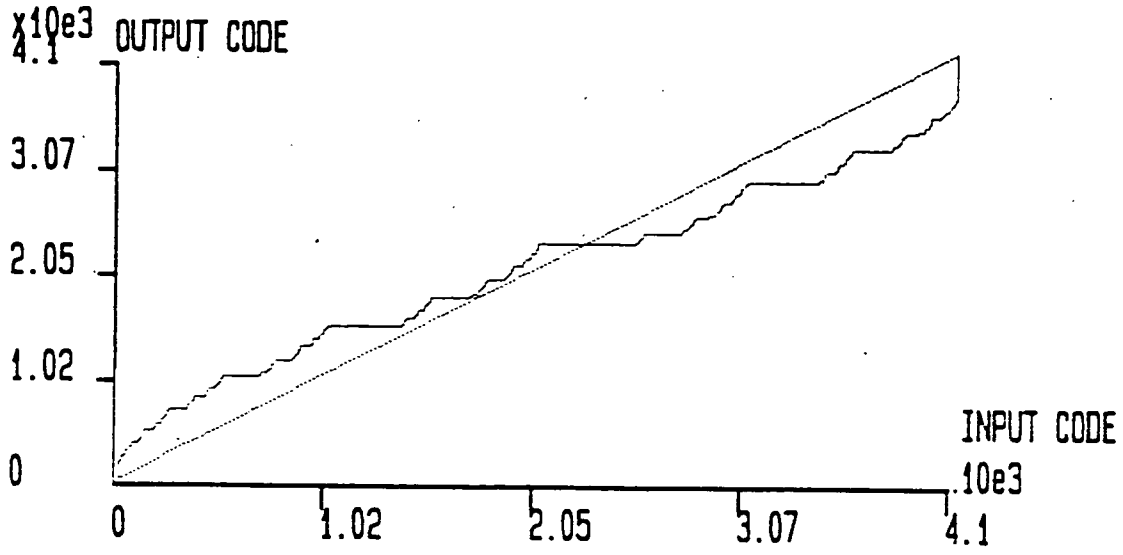

Figure 5.6: The transfer function mapping obtained from the pdf of figure 5.5.

linearity in the converter as the maximum INL looks to be around a few hundred LSB.

The shape of the nonlinearity is very interesting as it is very like that used in some of the ADC simulations and this would seem to justify the choice of a third order polynomial to model the converter transfer function. Figure 5.7 shows the same type of nonlinearity but of significantly less severity, and it can be seen that the horizontal segments in the characteristic are smaller, as would be the pdf gaps, and that the maximum deviation from the ideal transfer function is much reduced.

5.4.4. Pdf Spikes

If the values of the R resistors are reset to their correct values and those of the 2R resistors are now reduced, then a quite different set of distortions is produced. Now the more significant bit transitions have spikes in the sampled pdf, as can be seen in figure 5.8, which is a rather extreme example. These spikes in the nonlinear pdf produce the opposite effect to gaps on the mapped transfer function i.e. they cause vertical segments, the size of which is determined by the degree of 2R resistor misadjustment. The mapping generated from the pdf of figure 5.8 is in turn shown in figure 5.9, where it is obvious that it is now a very roughly quantised transfer function, with excessive differential nonlinearity at the vertical segments.

In general, nonlinearities caused by misadjustment of the 2R resistors are not as successfully mapped as those with other resistor errors. It should be remembered, however, that the scales of nonlinearities being generated are extremely large e.g. it requires a misadjustment of the 2R resistors to around 40% of their correct value to produce the transfer function shown in figure 5.9. When smaller, slightly more realistic, errors are used the nonlinearity is easily mapped to a satisfactory standard.

5.4.5. Combined Misadjustment

When the R/2R ladder is not so systematically altered more complex effects occur,

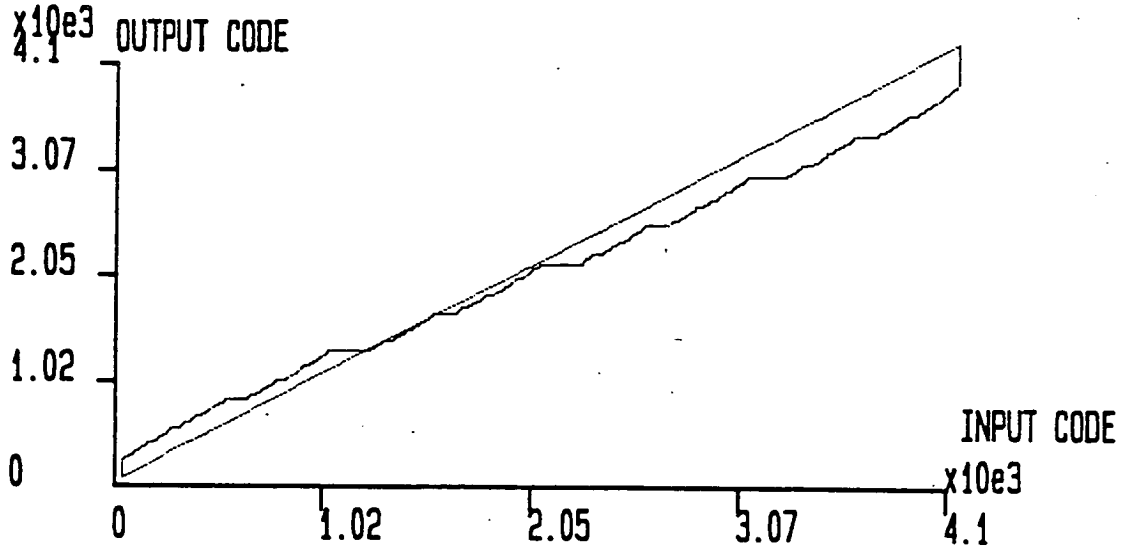

Figure 5.7: A similar transfer function mapping for a less severe misadjustment.

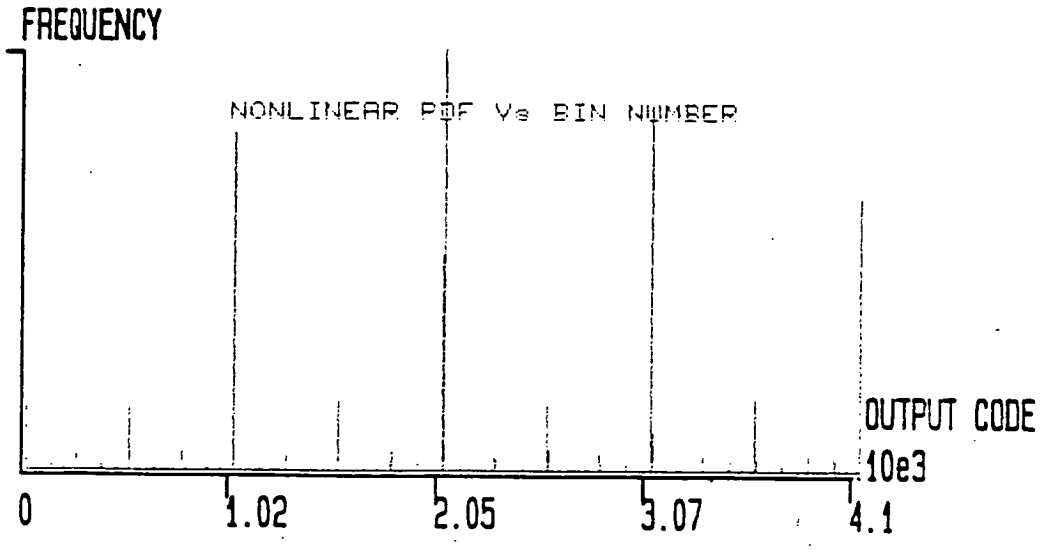

Figure 5.8: A nonlinear pdf caused by large misadjustment of the converters' 2R resistors.

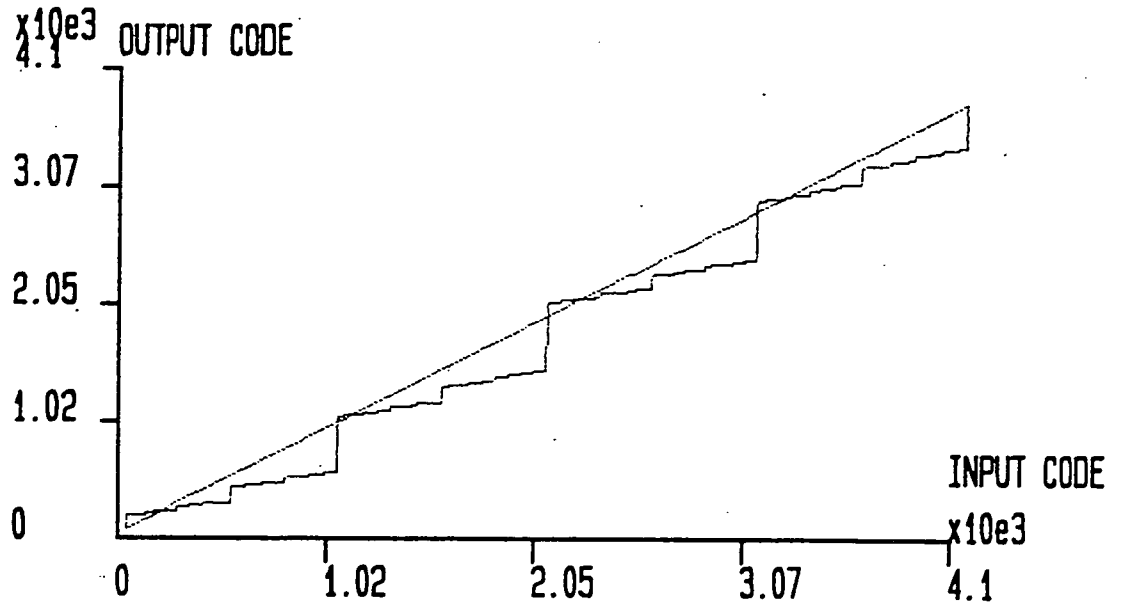

Figure 5.9: The mapping associated with the sampled pdf of figure 5.8.

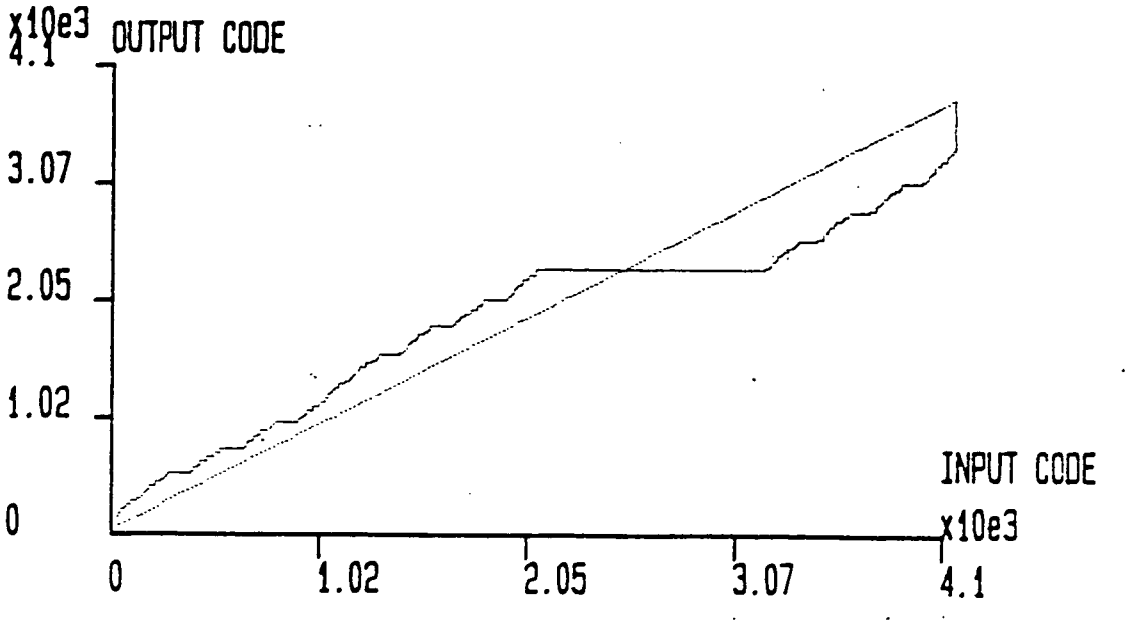

Figure 5.10: A mapping of a nonlinear pdf sampled by a converter with combined misadjustment.

the relative values of adjacent resistors must have a larger role to play in determining the shape of the nonlinearities. This can be seen from figure 5.10 where there are both horizontal and vertical dislocations in the mapping and also sections where there are several different gradients and deviations. This extremely large amount of misadjustment was mapped very well and gave excellent results when a sampled signal was Fourier transformed.

5.4.6. Fourier Transforms of Mapped Signals

After a mapping had been produced from the sampled pdf's it was possible to generate a calculated Fourier Transform of any sampled signal. This allowed spectral analysis of the performance of the linearisation process of the same form as simulations had previously used. Comparison of figures 5.11 and 5.12, showing the unadjusted and the mapped converter outputs respectively, indicates how well the system has worked to eliminate the signal harmonics in figure 5.11 by the time the converter has been mapped in the second figure. The transfer function nonlinearity and its mapping, as shown in figure 5.6 are those used to produce the spectra of figures 5.11 and 5.12, and they correspond to the same nonlinearity setting on the adjustable ADC. The same suppression of the signal's erroneous harmonics can be seen in figures 5.13 and 5.14, where the specific nonlinearity being considered corresponds to that shown in figure 5.10.

It can also be seen that the patterns of distorted harmonics of the test signal fundamentals are of the same forms that were observed in the pure simulations performed on simulated ADC's with perfect test signals i.e. real nonlinearities produce the same kind of effects that the simply simulated nonlinearities did. This seems to suggest that the real nonlinearities experienced by the hardware test signal were of a similar type to those used in the previously performed computer simulations. This then suggests that the simulated geometric nonlinearities were fairly realistic and adequately modelled the physical situation.

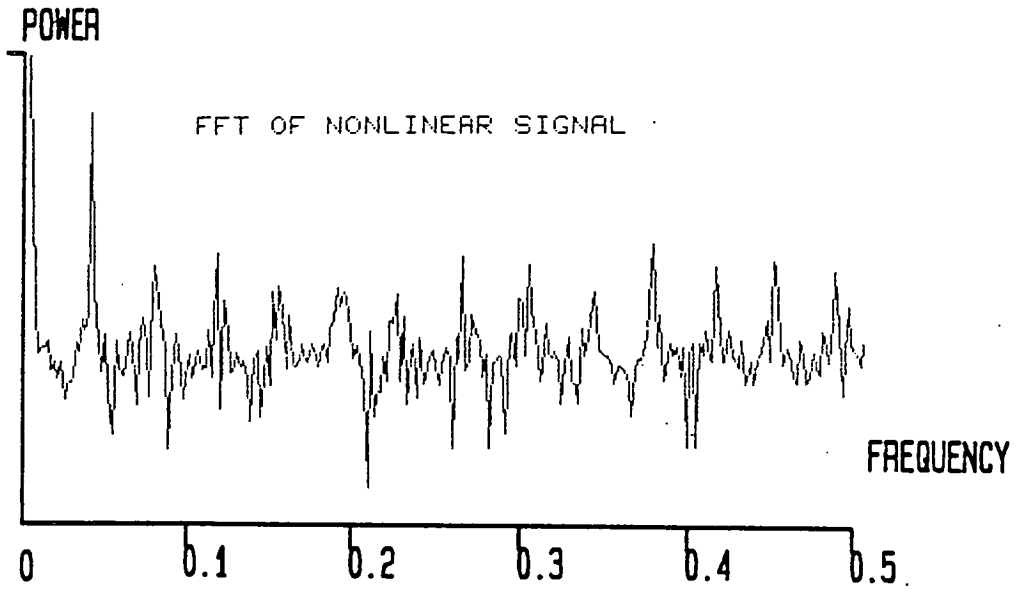

Figure 5.11: A calculated Fourier transform of a signal sampled by the nonlinear converter of figure 5.6.

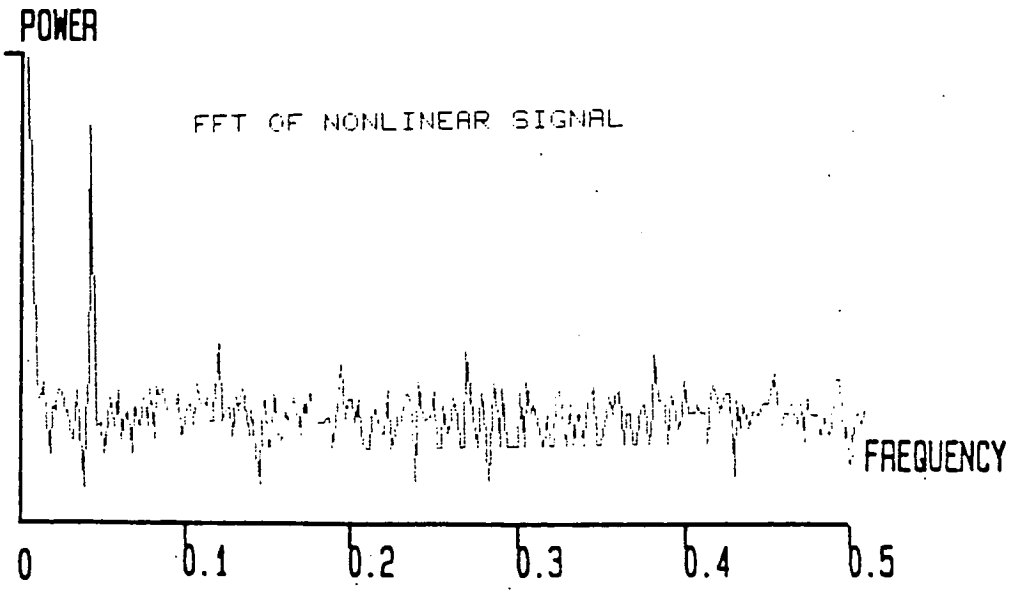

Figure 5.12: The same signal sampled by the now mapped nonlinear converter of figure 5.6.

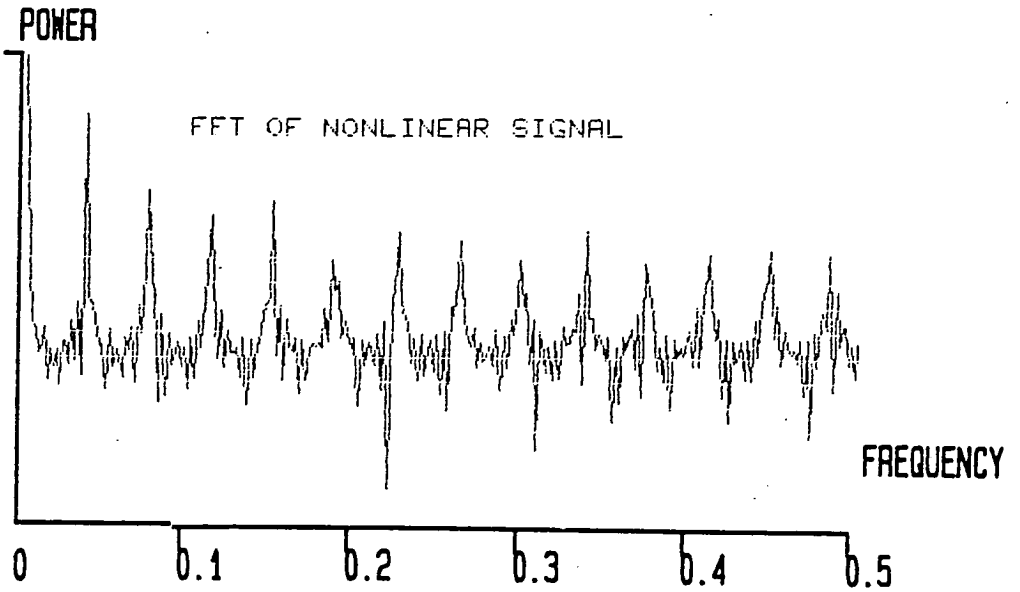

Figure 5.13: A Fourier transform of a signal sampled by the nonlinear converter of figure 5.10.

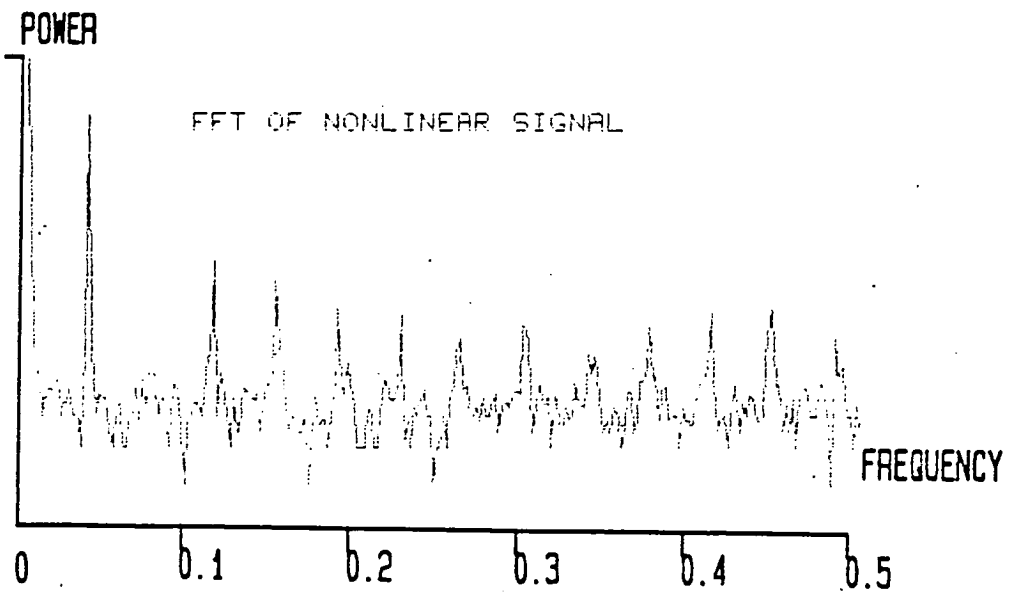

Figure 5.14: Another, very different, spectrum sampled by a very nonlinear converter.

5.5. Real Time Results

The previous sets of spectral data were calculated by a computer in much the same way as in the simulations presented in previous chapters. It was useful, therefore, to generate spectral data in real time, by a method other than Fourier analysis. This was achieved by use of a hardware spectrum analyser and a high quality DAC to generate analogue signals in real time, from the digital data produced by the ADC's. The following photographs were taken from the CRT of the spectrum analyser.

Figure 5.15 is the output produced by the linear ADC when sampling the sinusoidal test signal, and hence acts as a reference for comparison with the other results. Note the smooth noise floor and the single fundamental peak. Figures 5.16 and 5.17 are the original and mapped outputs from the nonlinear ADC when adjusted to the nonlinearity shown in figure 5.10. When comparing the calculated and actual distorted harmonics, (figures 5.13 and 5.16) the same relative peak heights and pattern can be observed, lending credence to the proposition that both signals have been sampled by an ADC with the same nonlinearity, and hence that the simulation results are correct. The distorted harmonics are as sharp as the fundamental but the noise floor has increased in level and is also very spikey. The mapping, as shown in figure 5.17, has much lower harmonic levels and noise floor, showing the increased signal to noise ratio possible by the application of the mapping process.

Figure 5.18 and 5.19 show the spectral data for a test signal sampled by the original and the mapped transfer functions of the nonlinear ADC, with a different nonlinearity. This nonlinearity was the same as that previously examined in figure 5.14 and again the harmonic peak patterns are the same, taking into account the different scaling of the axes and the noise averaging properties of the spectrum analyser. Again the converter noise floor has increased but by figure 5.19 we can see quite a large reduction in this and in the level of the harmonics.

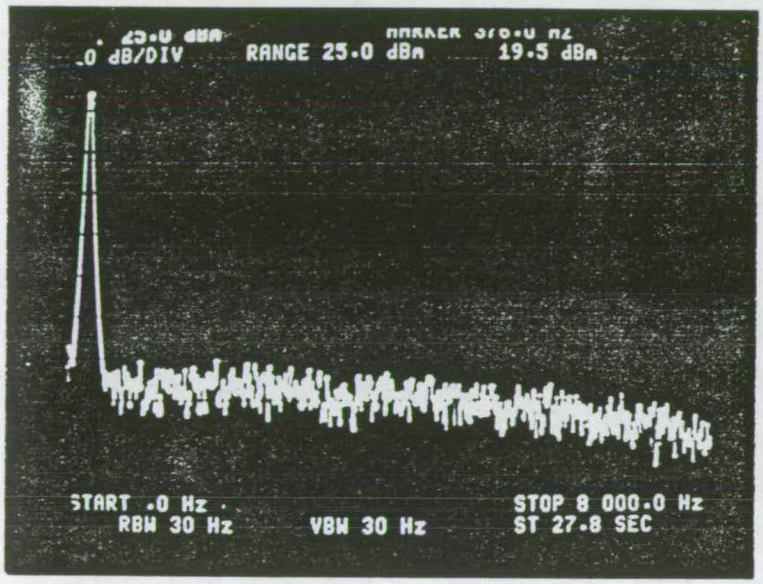

Figure 5.15: A photograph of the spectrum analyser output when the linear converter is sampling the test signal.

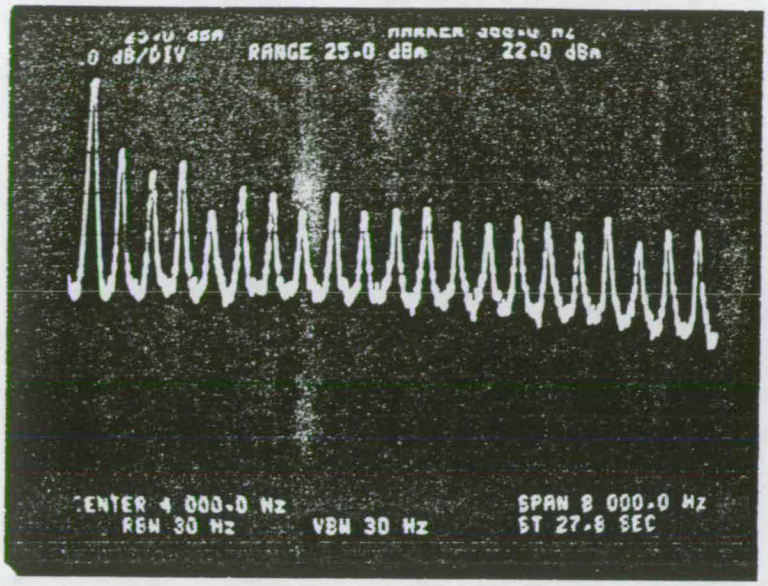

Figure 5.16: A photograph of the analyser display when sampling with the nonlinear converter of figure 5.10 which previously gave the spectrum in figure 5.13.

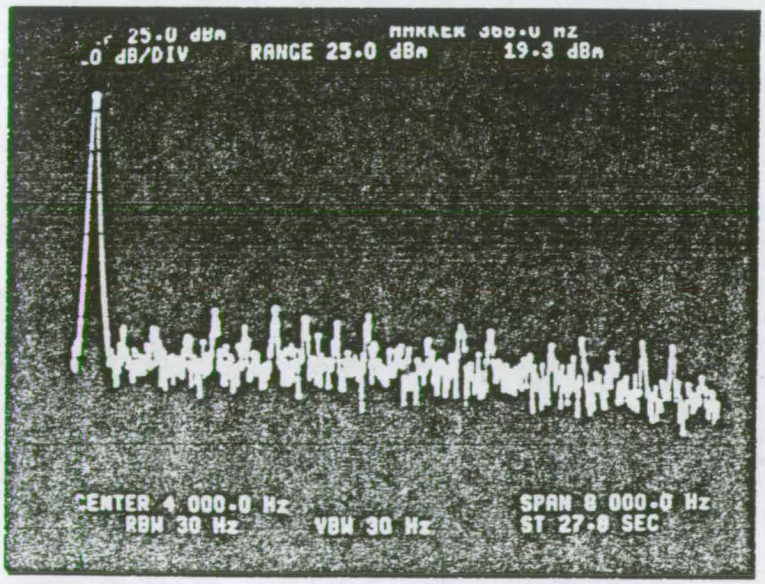

Figure 5.17: A photograph showing the spectrum after the converter used to generate figure 5.16 has been mapped.

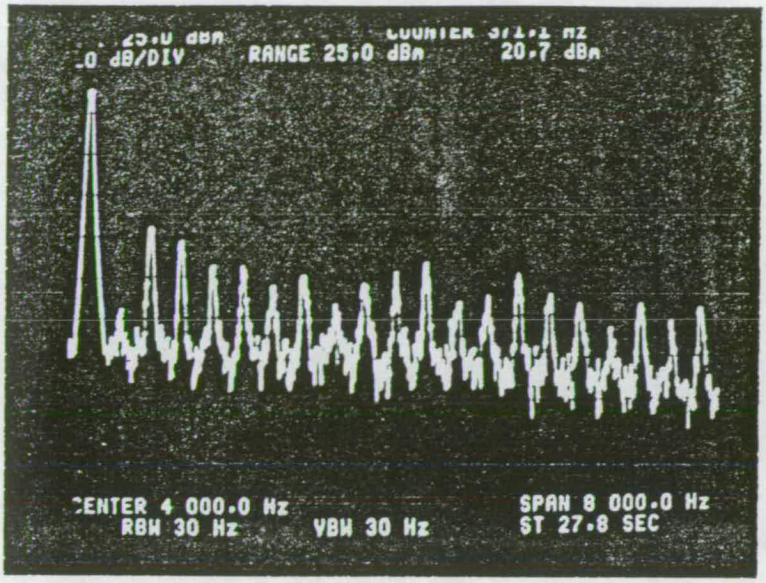

Figure 5.18: The unmapped measured spectrum which is equivalent to that in figure 5.14.

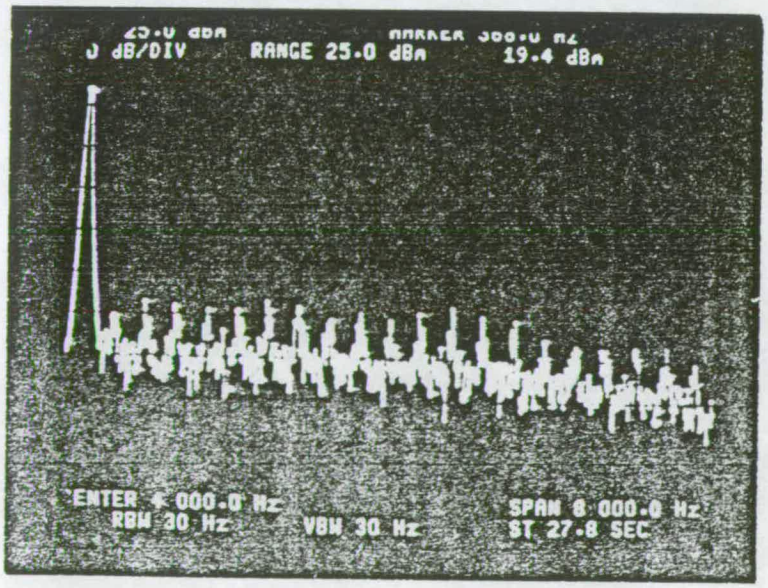

Figure 5.19: The spectrum of figure 5.18 after mapping.

5.6. Summary

This chapter has shown that a hardware implementation of the threshold tracking algorithm applied to linearising ADC's is not a task involving a large amount of leading edge technology, and that it can be done fairly cheaply. The results justify all aspects of the computer simulations that they tested, with the similarity in the spectral harmonic peak patterns verifying the simulation results from chapter 3. The fact that there were only a very few, rather extreme, nonlinearities which the system could not improve is again a testament to the ability of the threshold tracking algorithm to linearise any realistic transfer function.

Chapter 6

APPLICATION TO DAC's

Of the many measures of DAC performance, perhaps resolution, speed and linearity are the most important. Assuming that the resolution and basic speed of the converter are selected during its design, then the linearity of the device transfer function becomes the biggest factor affecting the quality of signal reproduction. Nonlinearity will in part be determined by the tradeoffs made at the design stage, especially any concessions made to gain speed, but it is mainly due to inaccuracies in weighted reference structures. It would seem feasible, therefore, to attempt to produce a mapping of the device's transfer function, in a similar manner to that employed for ADC's, which would result in higher converter linearity.

6.1. System Layout and Training

The notional hardware system layout required to implement a mapping of a DAC transfer function is in fact very similar to that used for an ADC. This time however, the memory look-up table must be situated before the converter, on the digital side, as shown in figure 6.1. It is also clear from the figure that the training signal required to determine the DAC's transfer function has to be digital in format and generated by the system processor/controller block.

As with ADC's the linearisation system operates in two modes; conversion and training. In conversion mode the digital input signal is mapped in the look-up table before being fed to the DAC which then determines the system output. The training

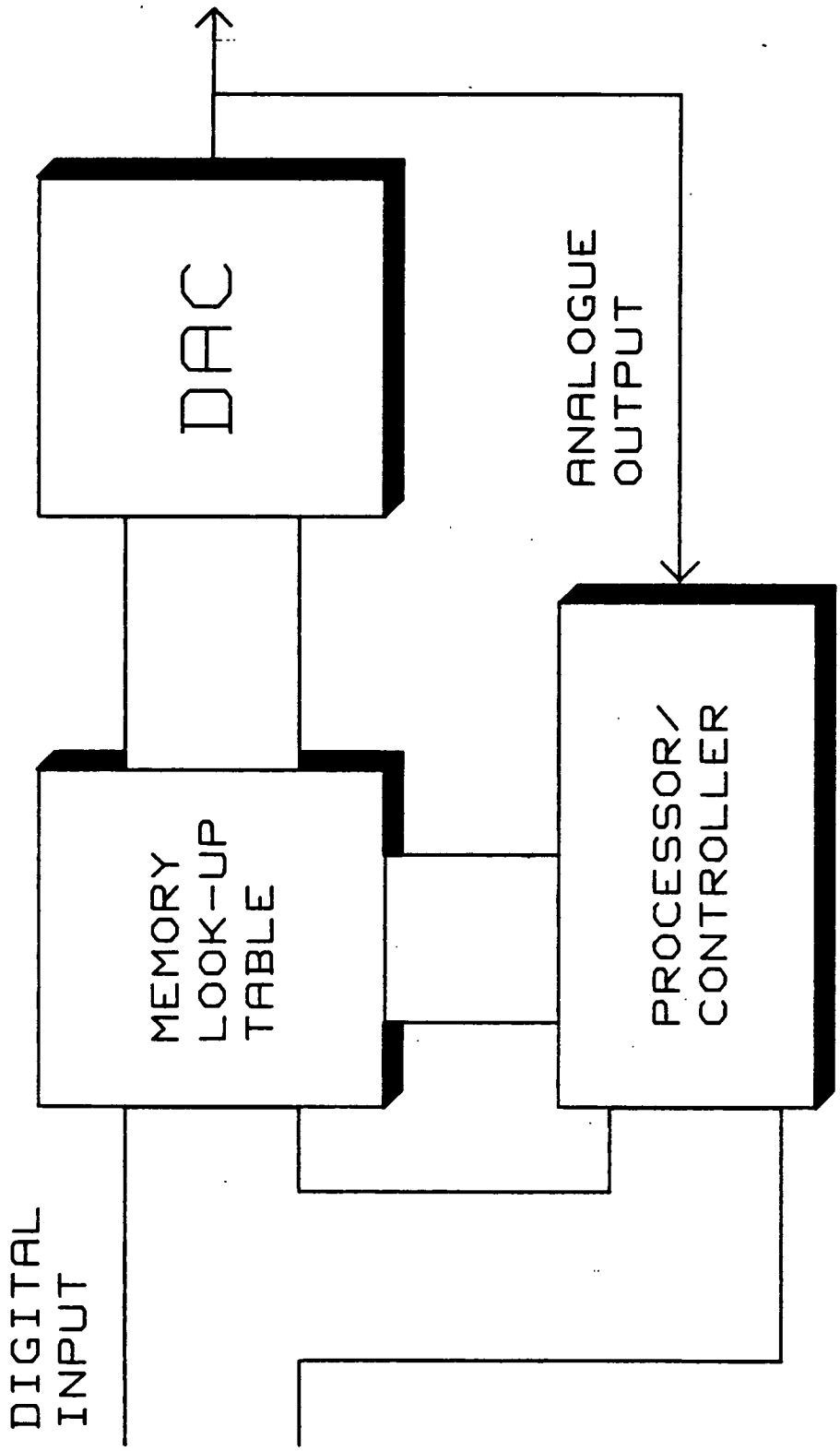

Figure 6.1: The hardware architecture for a DAC linearisation system.

mode is concerned with characterising the converter's transfer function nonlinearity and then applying a mapping algorithm to calculate the contents of the mapping memory.

6.1.1. Training Hardware and Operation

In the earlier case of ADC's this characterisation was obtained by forming sampled pdf's from analogue training signals of known statistics, or in the hardware case by direct measurement of the training signal pdf. Forming pdf's of analogue signals is a lot more complex than forming those of digital signals.

The easiest way round this is to convert the DAC's analogue output back to the digital domain with a high quality (linearity) ADC. This now allows simple collation and generation of digital pdf's of both the distorted and the undistorted signals. Also as the DAC has only a finite number of digital input signals and a corresponding number of analogue output voltages (assuming the device is allowed to settle properly) it becomes necessary to exercise each DAC input code on only a single occasion to obtain a full characterisation of the device's transfer function, to the required resolution.

The hardware needed to train a DAC can then become a simple counter, connected to the DAC, a linear ADC reconverting the analogue signal to form a crude distorted pdf. This can then be used by a modified threshold tracking algorithm to calculate the transfer function mapping.

The algorithm has been modified to implement the pdf array pointers in the opposite way to that employed in the standard algorithm. This means that the linear "pdf" sum is incremented and decremented to match the current nonlinear "pdf" total, which is adjusted for each DAC input code. As each bin in the linear "pdf" contains only a one then an exact equivalence between the two totals can always be found, and hence no threshold is required to be tested in the mapping process. A listing of the software used in the simulations for testing converter mean deviation

and integral nonlinearity is given in Appendix D.

The use of the modified tracking algorithm however, can be conceptually simplified as the output from the ADC can be used directly as the current address in the look-up table and the present value of the counter used as the contents of that address. This comes about as the distorted output from the DAC, when digitised, is the actual value output by the DAC for the current input from the counter. In turn, this means that to obtain the analogue version of the ADC output, the counter value must be fed to the DAC. Hence, the ADC output represents the look-up table address and the counter its contents. An overview of the complete training setup can be seen in figure 6.2.

6.1.2. Address Gaps

Unfortunately, due to the effects upon the uniform "pdf" caused by the converter nonlinearity not all output codes of the ADC are exercised. This is to be expected as distortion causes quantisation intervals to be unequal and hence it is likely that two or more DAC outputs would be converted to the same digital code. This then means that there will be gaps in the mapping written to the look-up table, with spaces where no address has been output by the ADC. If these codes were accessed by the input to the linearised DAC system, their output would be undefined and would probably cause a large discontinuity in the converter's transfer function. The memory contents at these addresses would have to be forced to have sensible values i.e. those which give the minimum distortion of the DAC transfer function.

The simplest way to determine what values should be placed at these missing addresses would be to interpolate between the closest accessed addresses, above and below the gap in the look-up table. However, when this simplified, non-algorithmic technique was experimented with, problems were discovered. These related to converter nonlinearities with large discontinuities, large amounts of differential nonlinearity, or nonmonotonicity. These were the sort of effects which caused large

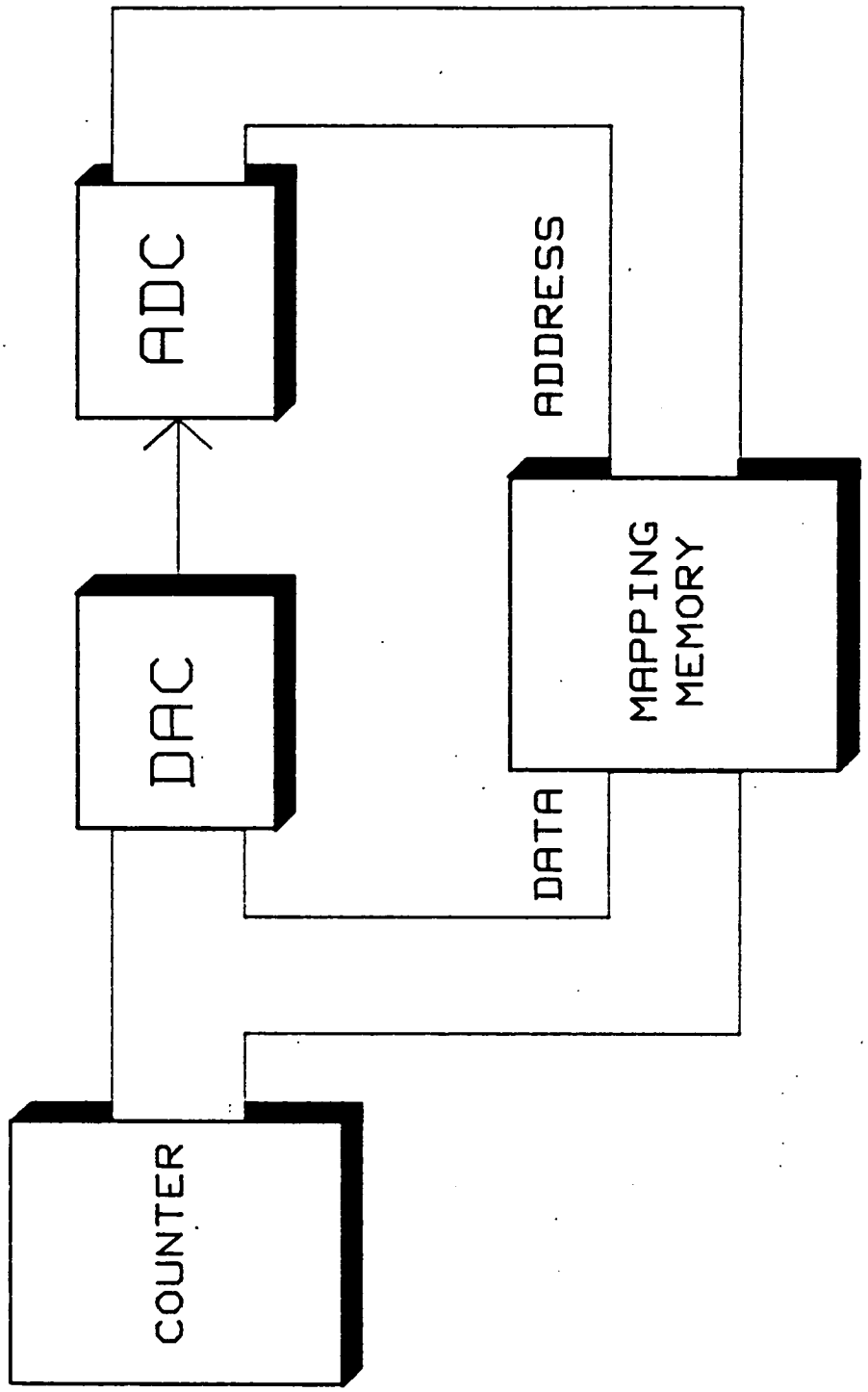

Figure 6.2: DAC linearisation system in its training set-up.

numbers of consecutive addressing gaps to appear in the mapped transfer function. As the method of implementation of the simplified technique allowed old mapping data to be overwritten by the current values being presented by the linearisation system, a lot of correctly mapped data was lost and then distorted by simple interpolation. Interpolation seemed to cause other indeterminate problems which could not be solved by simple means. The only feasible solution seemed to be to keep a record of all the data generated during the linearisation process and then to calculate the correct mappings from this information. The overall mapping process now became so complex as to no longer be simpler than the use of the modified threshold tracking algorithm and it was therefore discarded.

6.2. Simulated Results

Simulations were run on various types of DAC nonlinearity, generally with the same distortions that were used for the ADC tests. This time, however, the way in which the mappings and distortions were applied was very different to that used in the ADC simulations. First the digital code was converted to an analogue level by an idealised DAC. Then the analogue distortion function was applied to give the final analogue output value of the DAC. The mapping table and ADC portion of the overall linearisation system were applied in the positions indicated by the hardware block diagrams, figures 6.1 and 6.2.

Results were quantified by three major measurements: a mean deviation measurement of the mapped transfer function, integral nonlinearity plots showing exactly how the device transfer function had been mapped, and by the reduction of harmonic distortion in reconstructed sampled periodic signals.

6.2.1. Mean Deviation

There was nothing at all surprising about the results of the mean deviation measurements. They closely resembled the data generated by the ADC simulations, even in that particular distortions still had their own characteristics, in terms of their

deviation value.

There was nothing to plot mean deviation against as the mapping algorithm as such has no variable thresholds, and there could only be one "pdf" sample size.

6.2.2. Integral Nonlinearity

Several sets of simulations were run to produce plots of integral nonlinearity for the original and mapped DAC transfer functions. In general the results of these were very good and agreed well with those generated by the ADC simulations. Figure 6.3 shows the INL of a converter with a simple nonlinearity which has no large differential nonlinearities and a smooth transfer function throughout. The mapped characteristic has its integral nonlinearity bounded by approximately ± 0.5 LSB, over the full range of the DAC. Some DNL has been introduced by the mapping process, but this is to be expected and is similar to the effect experienced in ADC's.

The distortion mapped in figure 6.4 shows the same characteristics and again compares well with the ADC simulations performed for the same nonlinearity.

Figure 6.5, on the other hand, shows a much more complex situation where the nonlinearity involved has quite large amounts of differential nonlinearity and an unmapped transfer function which crosses the ideal (zero INL) line several times. The nonlinearity also contains several periodic components which considerably complicate the DAC transfer function. The mapped function is again, well confined between ± 0.5 LSB INL, for all parts of the converter's range.

From these results it can be inferred that when a modified threshold tracking mapping algorithm is applied to DAC's the results, as far as integral nonlinearity is concerned, are as good as those obtained from simulations of ADC's. This suggests that the technique is as robust and capable as that for ADC's and therefore, should be able to improve the linearity of any given DAC transfer function.

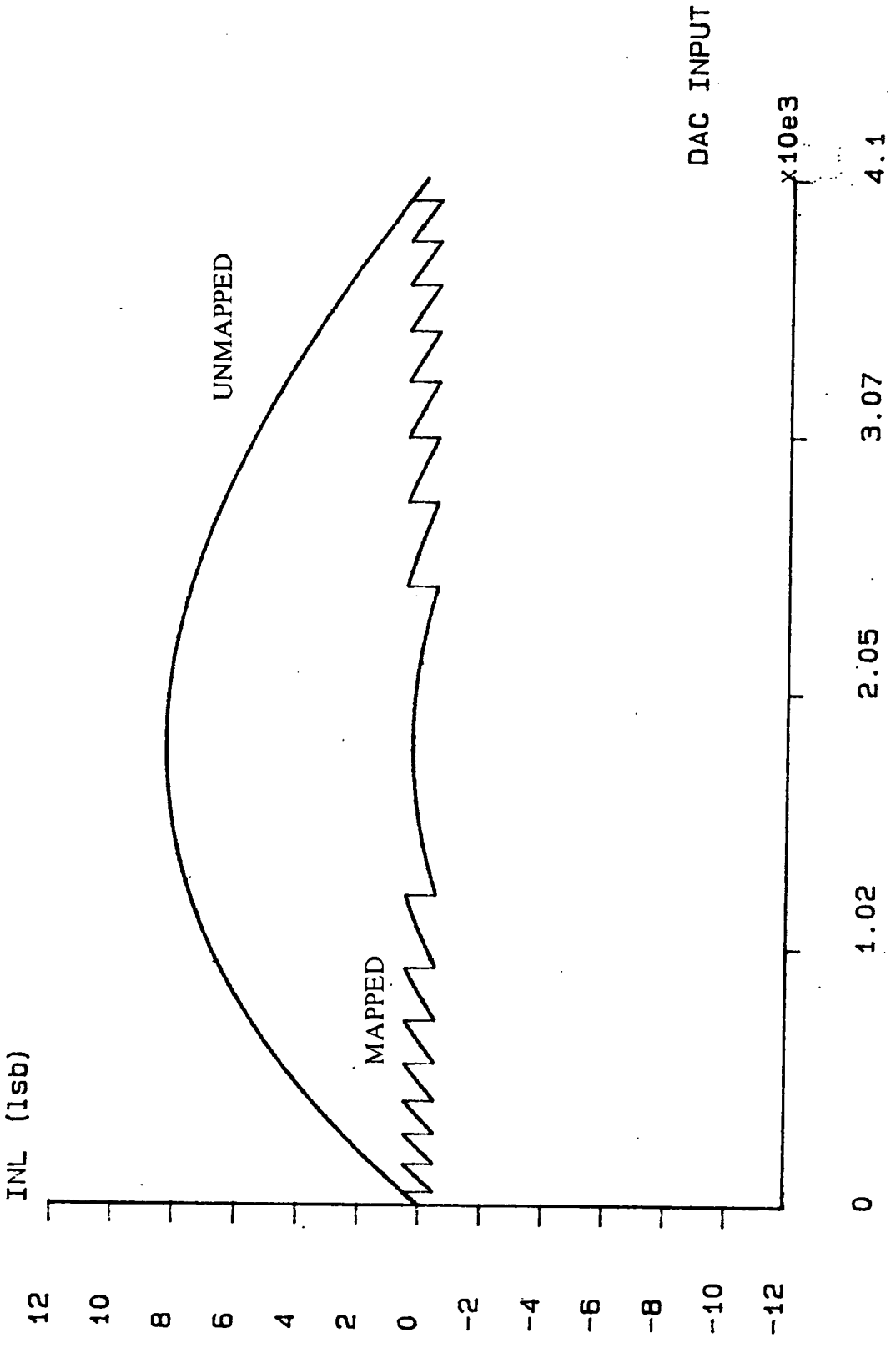

Figure 6.3: The integral nonlinearity plot of a DAC with a fairly simple simulated nonlinearity.

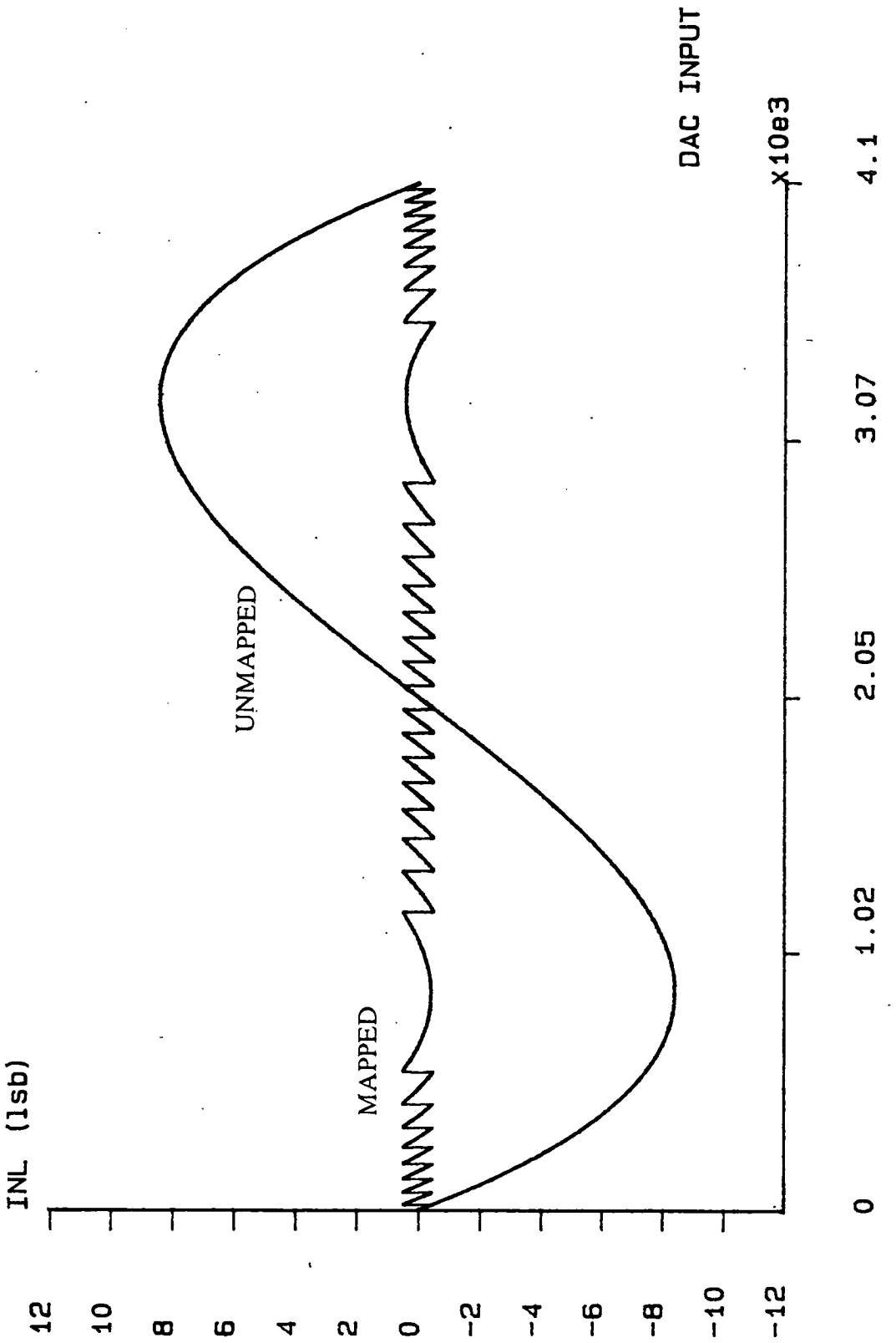

Figure 6.4: Another integral nonlinearity plot of a simulated DAC before and after mapping.

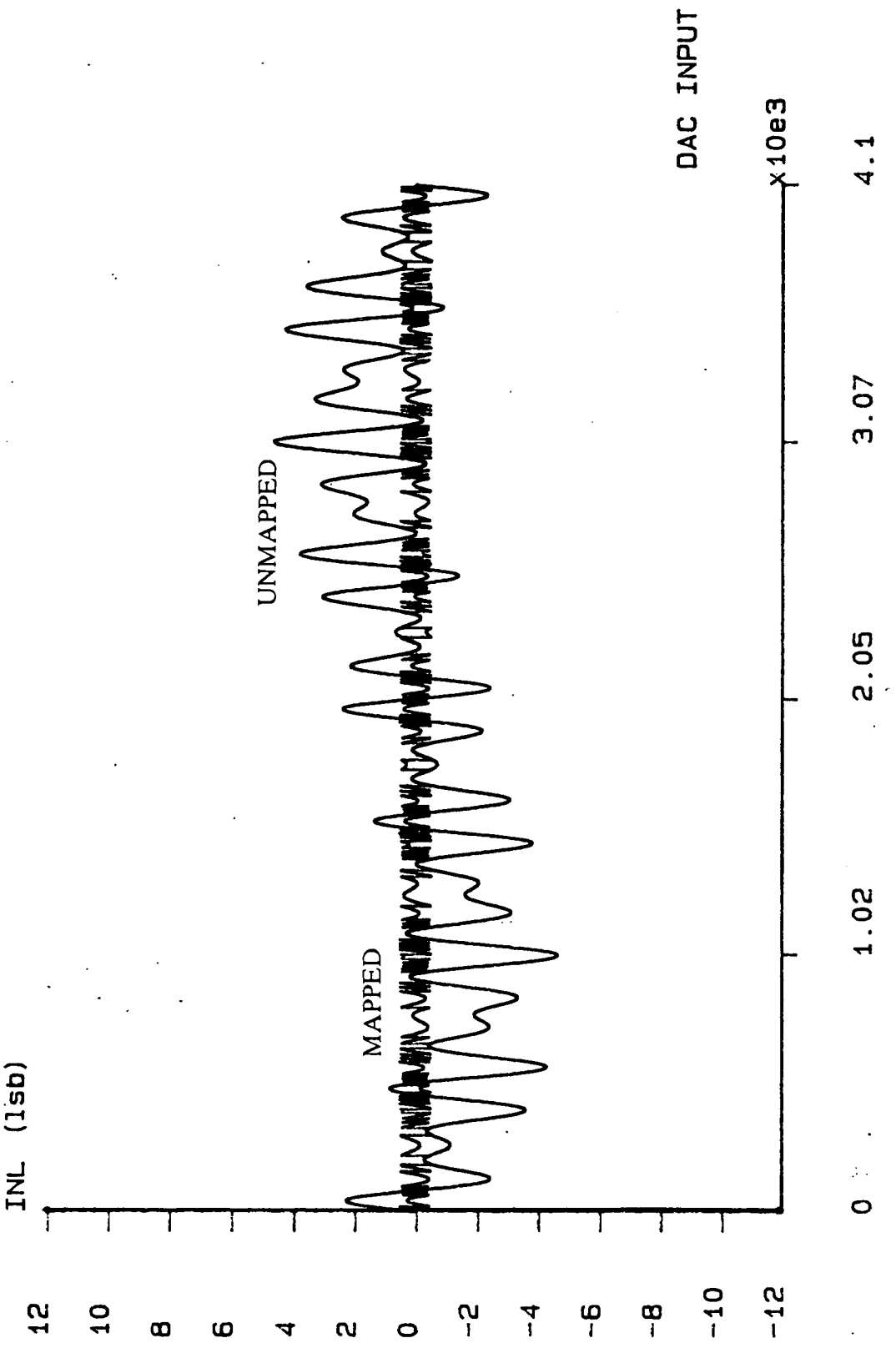

Figure 6.5: The original and mapped INL of a more complex transfer function distortion.

6.2.3. Harmonic Distortion

To determine how the DAC nonlinearity affected reconstructed digitally sampled periodic signals and to what respect this could be countered by threshold tracking, simulations to generate, sample, reconstruct and distort two tone periodic signals were written. These allowed the behaviour of both distorted harmonics and intermodulation products to be observed when the program's output had been Fourier transformed. A listing of the piece of code required to generate the reconstructed signals after the mapping has been performed is given in Appendix D.

Figure 6.6 shows the distortion produced by the same shape of nonlinearity used in the INL plot of figure 6.3, except that the size of the nonlinearity has been scaled for a 16 bit converter. The spurious peaks are very noticeable as they are around 30-40 dB above the noise floor and hence, represent a serious distortion of the original signal. After the converter has been mapped these peaks have virtually disappeared and it is almost impossible to distinguish them from the noise floor, as can be seen from figure 6.7.

Figure 6.8 shows the situation for a different nonlinearity which has been scaled from that used in generating the INL plot of figure 6.4. This time even higher order harmonics and modulation products are prominent and of the same level as those of figure 6.6. When this nonlinear DAC is mapped, in figure 6.9, the spurious peaks are seen to be again largely diminished, although they may be slightly easier to distinguish than those in figure 6.7.

When the more complex nonlinearity used in simulating the INL pattern of figure 6.5 is scaled to a 16 bit converter, not only do distorted harmonics and intermodulation products appear as expected, but the level of the noise floor increases by 15-20 dB as well. This obviously represents a severe problem to the linearisation process as can be seen in figure 6.10. After mapping however, the transfer function must be reasonably linear as figure 6.11, which shows the mapped

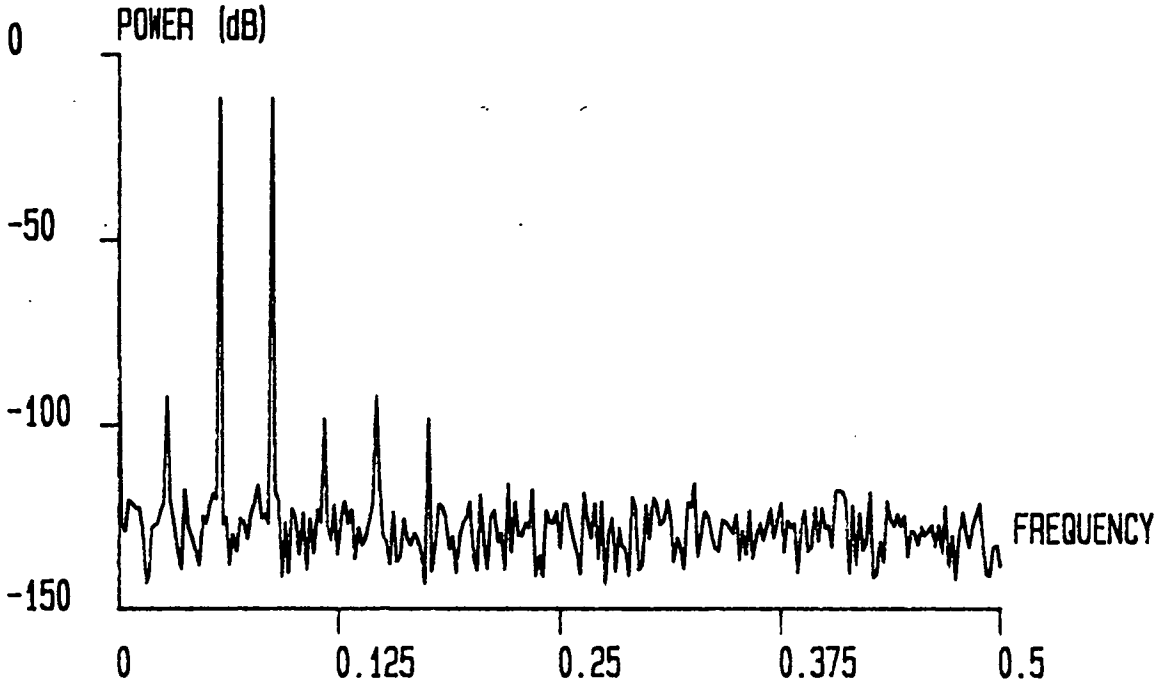

Figure 6.6: The Fourier transform of an ideally sampled two tone sinusoidal signal which has been reconstructed by a 16 bit DAC with a simulated nonlinearity of the same shape as that shown in figure 6.3.

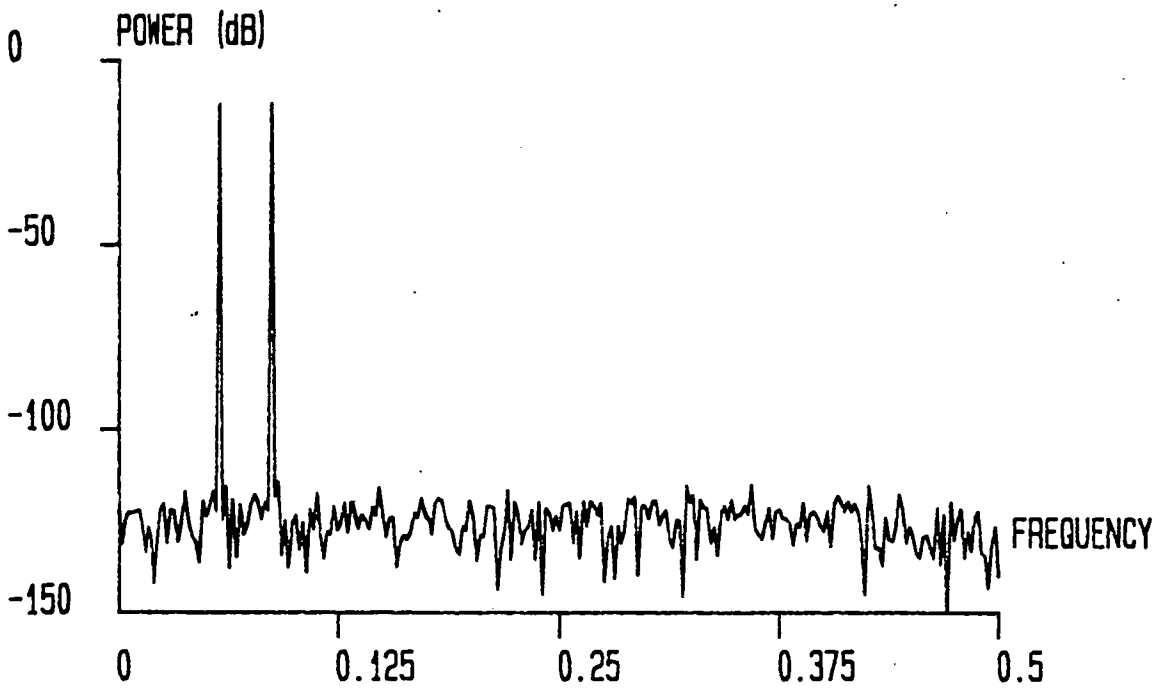

Figure 6.7: The reconstructed spectrum of the same signal as above after the DAC has been mapped.

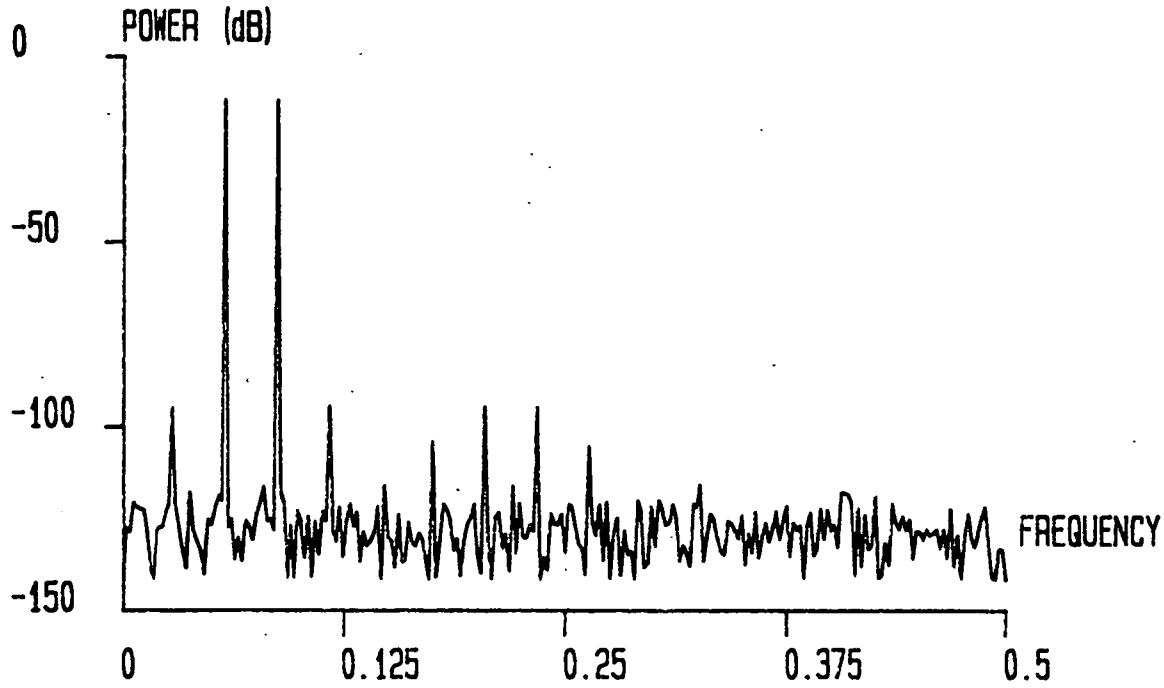

Figure 6.8: The FFT of a two tone sinusoid after reconstruction by a 16 bit DAC with a transfer function distortion of the same form as that of figure 6.4.

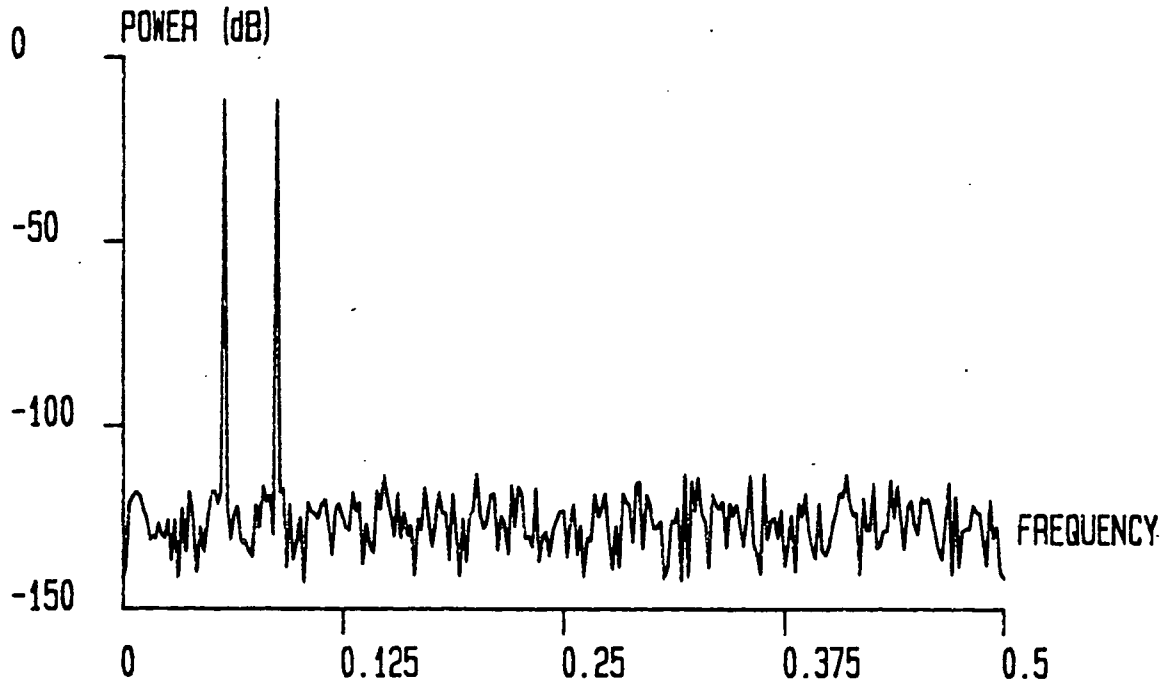

Figure 6.9: The equivalent spectrum to the above after the reconstructing DAC has been mapped.

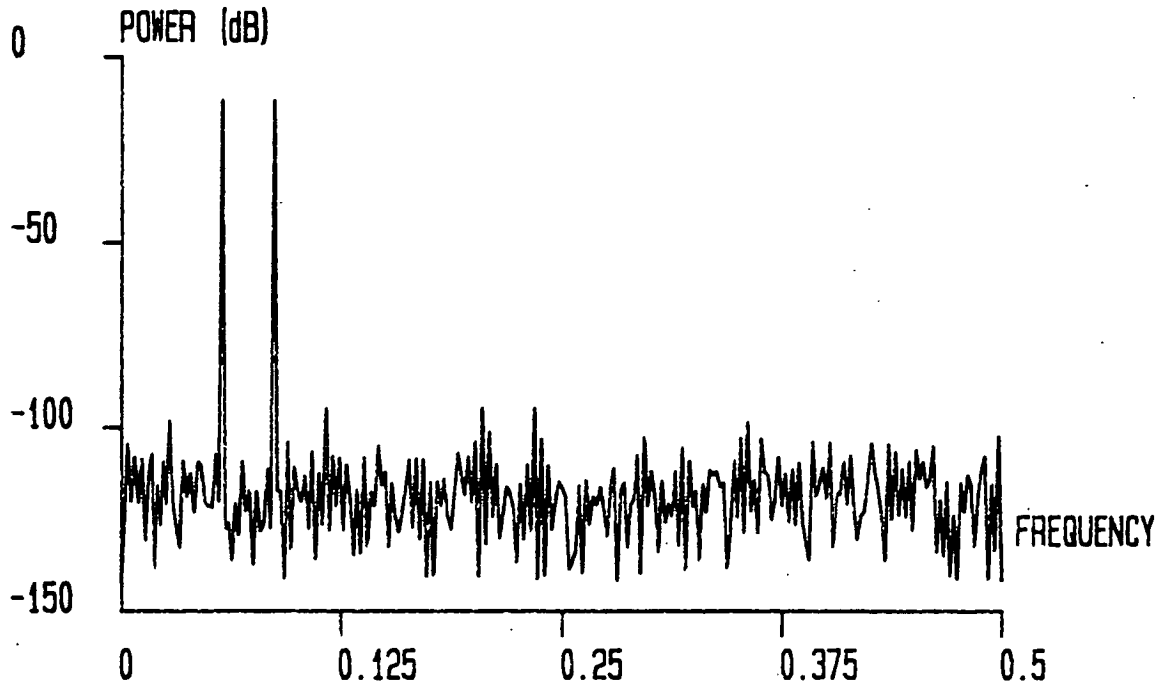

Figure 6.10: The unmapped spectrum for a 16 bit DAC with a nonlinearity of approximately 4 bits magnitude and similar in shape to that of figure 6.5.

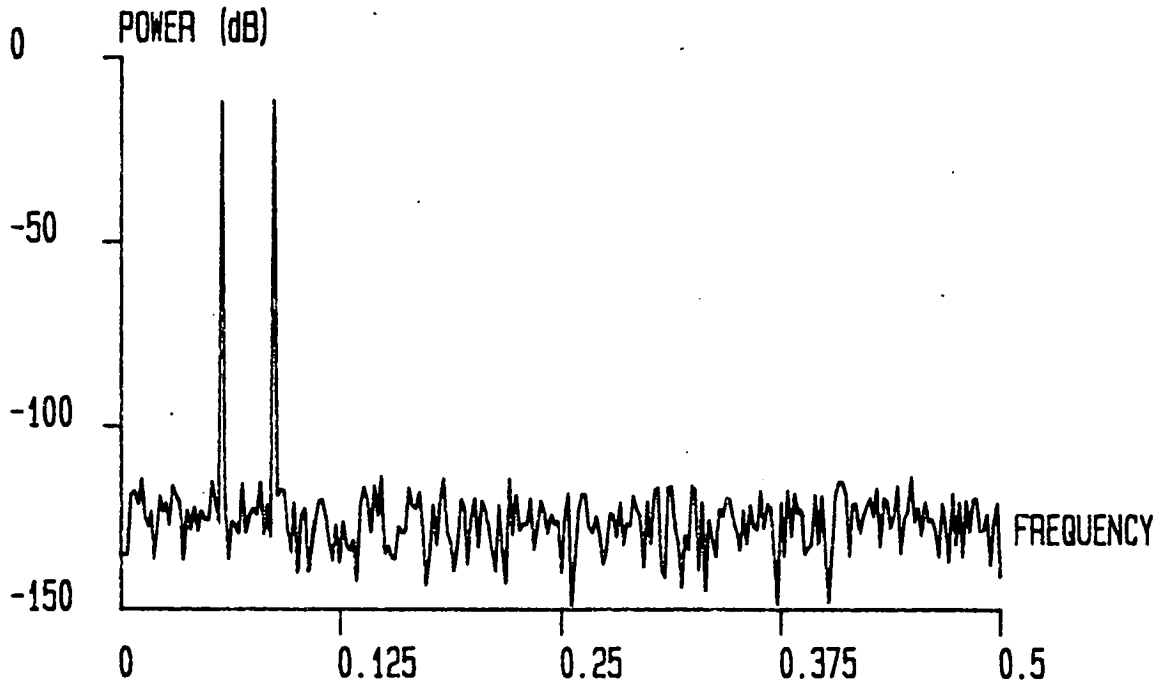

Figure 6.11: The mapped spectrum corresponding to the above distorted version.

spectrum, has no spurious peaks worth noting and a noise floor comparable to that of the previous mappings, which is well below that of the unmapped converter.

These three sets of figures and their associated simulations have shown that the digital to analogue converter nonlinearity distorts harmonics and creates intermodulation products in much the same manner as ADC nonlinearity does. Use of the modified threshold tracking mapping algorithm reduces both harmonically related peaks in the spectrum of signals reconstructed by mapped DAC's, and reduces any distortion or perturbation of the device's noise floor as well.

6.3. Implementation of Reduced Resolution

As with ADC's, although the mapping process has worked well, the linearisation system is left requiring a large amount of memory to implement the mapping, especially for higher resolution converters. Although memory is relatively inexpensive it would not be simple to fabricate large segments of ROM or RAM onto the same integrated circuit as a digital to analogue converter. Any reduction in the size of the memory block needed would therefore be of great benefit. Adapting the reduced resolution mapping process, developed for ADC linearisation, to work in the significantly different DAC application, would therefore be very useful indeed.

This task is not as difficult as it might seem as the process of forming a mapping based around only a few points, with straight line segments of transfer function mapping assumed between them, depends only upon the number of stored mapping points and is not influenced by the use or location of the mapping within the overall system. This means that the same program segments can be used to generate the reduced resolution mapping as were used in the ADC linearisation application. Similarly then, the same sort of tests can be made to determine the efficiency and usefulness of this technique when it is applied to DAC linearisation. The hardware layout of such a system is outlined in figure 6.12, which shows not merely the use of

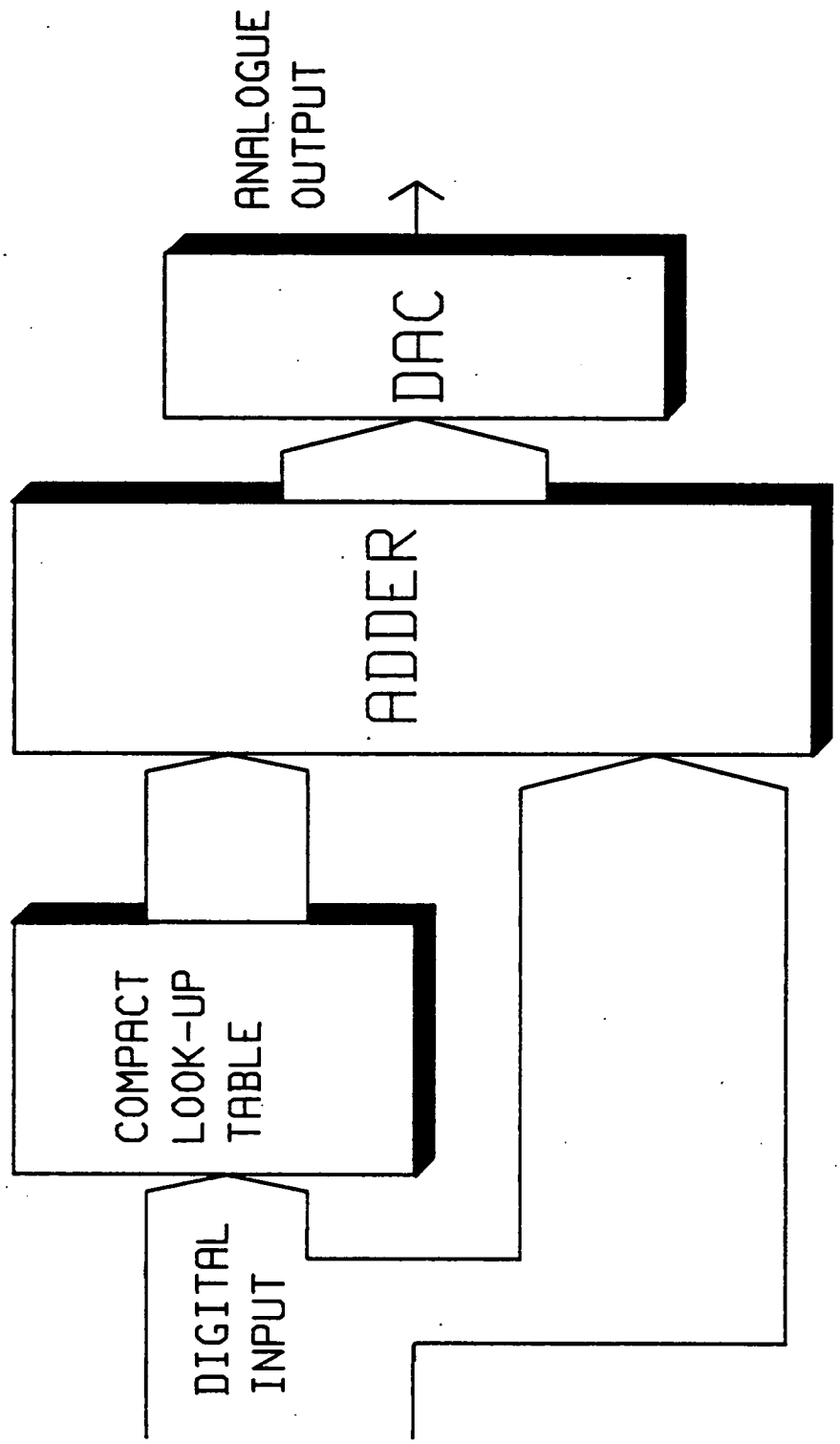

Figure 6.12: The architecture and modification required for the implementation of a reduced resolution DAC mapping system.

mapping offsets, but the full memory reduction system, where only the more significant bits of the digital input address the look-up table.

6.3.1. Required Addressing

The first simulations to be tested were used to determine a suitable number of addressing lines for the reduced resolution mapping. This could be measured by comparing the mean deviation of the range of nonlinearities mapped with different reduced resolutions. When there was little or no loss of performance caused by the mapping procedure, for a certain number of look-up table address lines, across a wide spectrum of converter nonlinearities, then it could be assumed to be safe to map any distortion to that size of compact look-up table.

For ADC's this limit was taken to be 10 address lines, although for most distortions 8 bits were generally sufficient. The results of the DAC simulations are represented in figures 6.13 and 6.14 which were generated by data from 12 and 16 bit linearised converters respectively.

Figure 6.13 shows that all but one of the mappings has converged to near its optimum level (a horizontal line) by the time that 9 address lines have been used. Only one of the more complex nonlinearities, which involves periodic components, has yet to stabilise, which it finally achieves by 12 bits which represents no reduction in the resolution of the mapping. However, by 10 bits even this distortion is within 0.3 dB mean deviation of its final value, which, as can be seen later, is probably sufficiently close.

A similar situation exists for figure 6.14 with most of the nonlinearities having converged to a very good approximation to the full resolution mapping by around 10 address bits. Certainly all the distortions simulated have reached their optimum mapping level by the time that 12 address bits have been used. Again though, it may be sufficiently accurate to use only 10 lines as all the nonlinearities are extremely close to their final mapped mean deviations by this point.

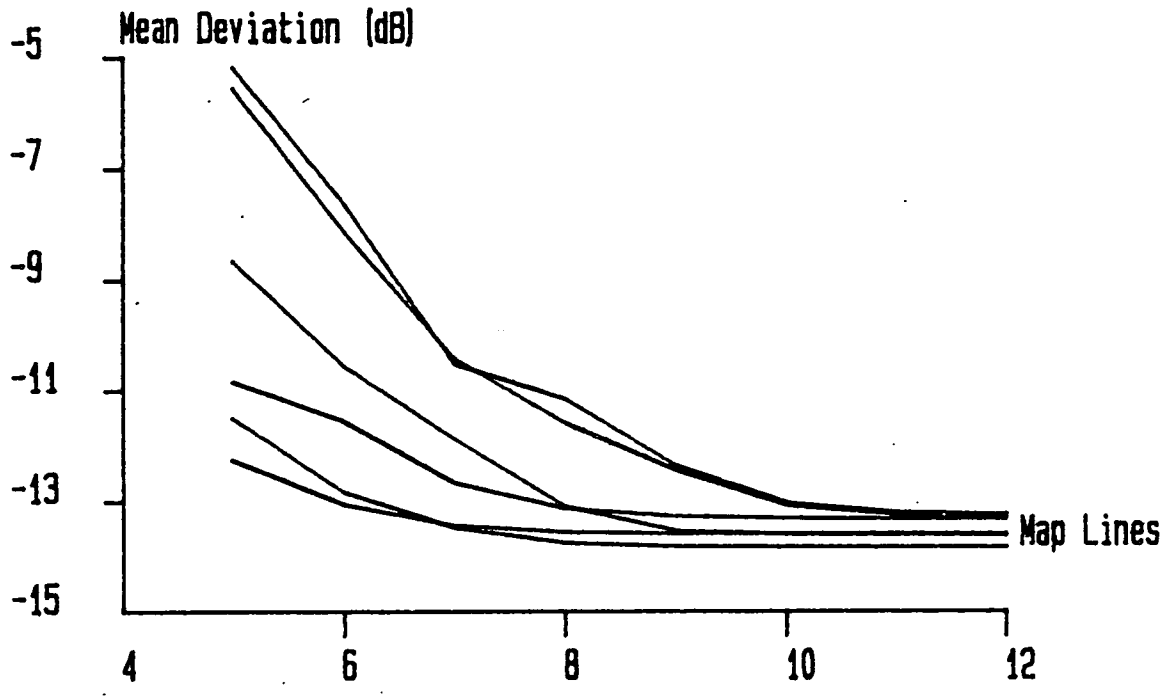

Figure 6.13: The mean deviation error performance of a selection of different 12 bit converter nonlinearities over a range of look-up table address lines.

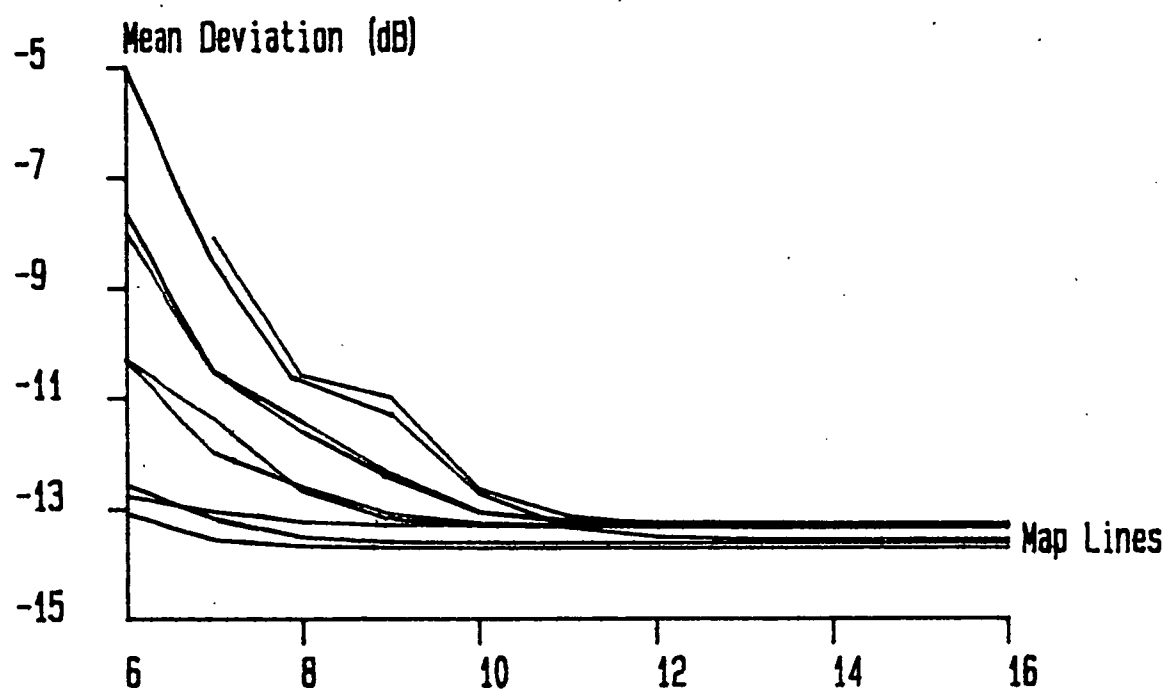

Figure 6.14: Mean deviation against the number of address lines used to access the mapping table for a range of different 16 bit nonlinearities.

6.4. Results with Reduced Resolution Mappings

Similarly to the full resolution DAC simulations, the results are divided into two categories: integral nonlinearity/transfer function plots, and those showing distortion of harmonics and intermodulation products of reconstructed periodic signals.

6.4.1. Integral Nonlinearity

As was to be expected the size of any INL on the plots was increased by the introduction of a reduced resolution mapping system. Whereas previously, INL was bounded between ± 0.5 LSB it is now less restrained and able to wander further from the ideal zero line. Figure 6.15 is the INL plot of the nonlinearity used in figure 6.4, with a mapping resolution of only 5 bits, or 32 mapping points. The integral nonlinearity now varies between ± 2 LSB, and interestingly the end points of the converters input range are not zeroes of INL. Discontinuities of the mapping, where they occur, are now regularly spaced (at the mapping points) whereas in the full resolution plot (figure 6.4) they were spaced as frequently as was needed to maintain ± 0.5 LSB linearity. This demonstrates the manner of the loss of performance of INL introduced by the reduced resolution of the mapping.

Figure 6.16 shows a case where there is a much larger loss of linearity due to reduced addressing. The vertical lines on the mapping show where the severe nonlinearity has been mapped. This is the result when only 5 bits are used to address a complex nonlinearity, which has segments of large differential errors. Even so, there has still been a noticeable improvement in the overall integral linearity of the converter. Whilst in figure 6.15 using only 5 address bits still gave a reasonable solution, clearly something closer to the suggested 10 bit minimum is going to be needed for a decent level of performance this time.

In fact, when 10 address lines are used, as in figure 6.17, very good performance in terms of INL is achieved. Here we are almost back to the optimum situation with INL bounded very approximately between ± 0.5 LSB with no major excursions.

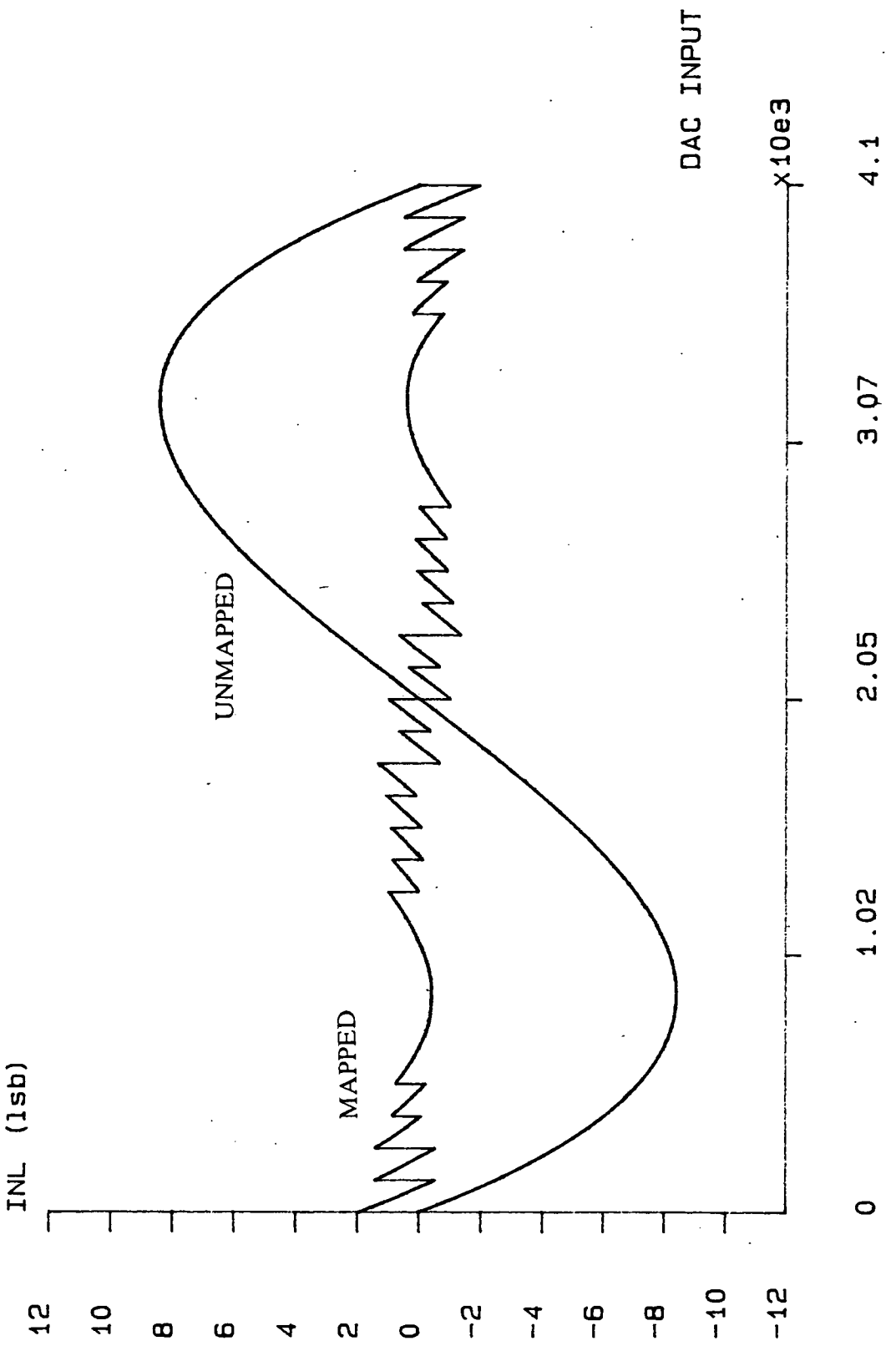

Figure 6.15: An integral nonlinearity plot of the 12 bit DAC nonlinearity used in figures 6.4 and 6.8, showing linearisation performance when only 5 address lines are used.

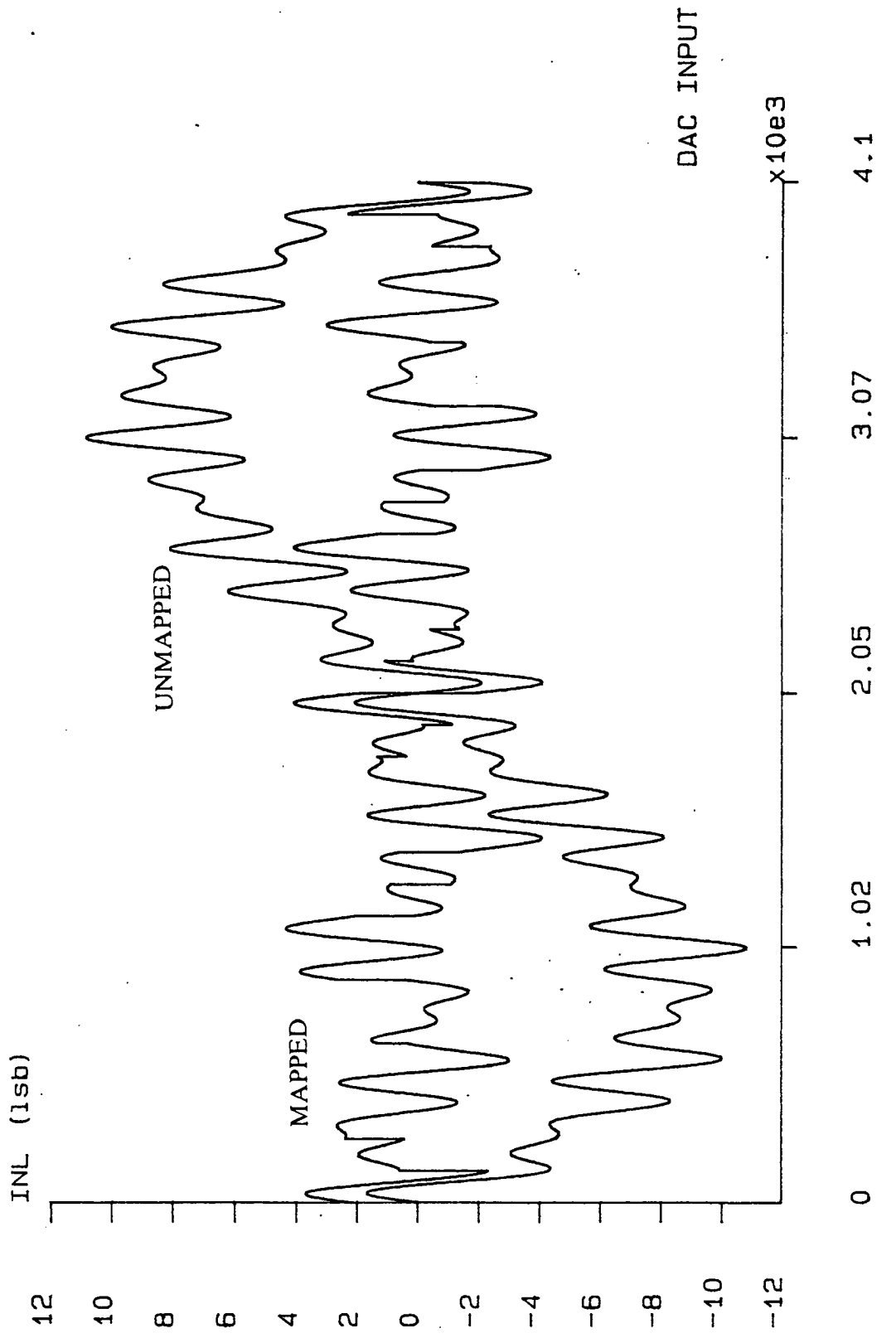

Figure 6.16: An INL plot for a converter which has only 5 bits addressing the LUT for a complex nonlinearity.

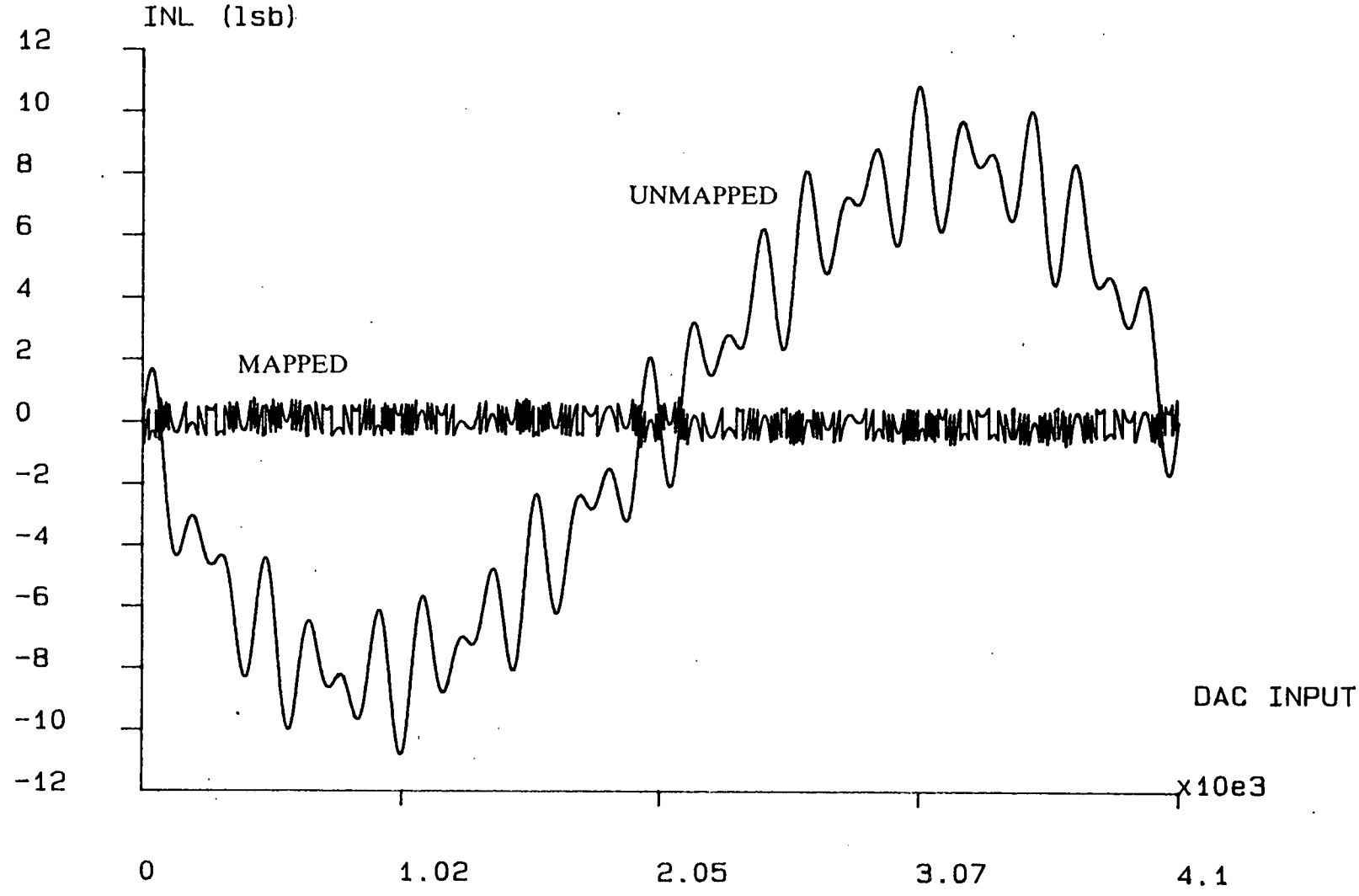

Figure 6.17: An equivalent INL plot to that of figure 6.16 except that now 10 address lines are used in the mapping.

This implies that although using only 10 address lines lost 0.3 dB of mean deviation performance, that small an amount is negligible as far as integral nonlinearity in DAC's is concerned. This in turn suggests that 10 address bits is overly sufficient to implement a reduced resolution mapping of even complex nonlinearities.

6.4.2. Spectral Distortions

As an alternative measure of system performance and to confirm the conclusions made from the INL plots, Fourier transforms were performed on digitally sampled two tone periodic signals, reconstructed with a range of mapped nonlinear converters. On the whole these were as was expected bearing in mind the integral nonlinearity simulations and the earlier reduced resolution ADC data. Figure 6.18 shows the unmapped distortion caused by the nonlinearity of figure 6.15, scaled to 16 bits, which has prominent distorted peaks, while figure 6.19 shows the output when the DAC has been mapped with 5 address bits, far below the recommended amount. The mapped spectrum is much improved as only a few remnant peaks remain, the larger of which are the two either side of the fundamentals. Apart from these two features the mapped spectrum compares very favourably with that of figure 6.9 which shows a full resolution mapping of this nonlinearity.

When more complex nonlinearities are used, such as the one used in the previous INL plots of figures 6.16 and 6.17, 5 address bits are no longer anywhere near good enough. Figures 6.20 and 6.21 show the unmapped and full resolution mapped spectra for the nonlinearity in question. The initially high noise floor should be noted, above which distorted peaks can be seen to rise. In the mapped converter, both of these features have been eliminated and the spectrum looks very linear.

When figure 6.21 is compared with the mapped spectra of figures 6.22 and 6.23, where 5 and 10 address lines out of the original 16 have been utilised respectively, several observations can be made.

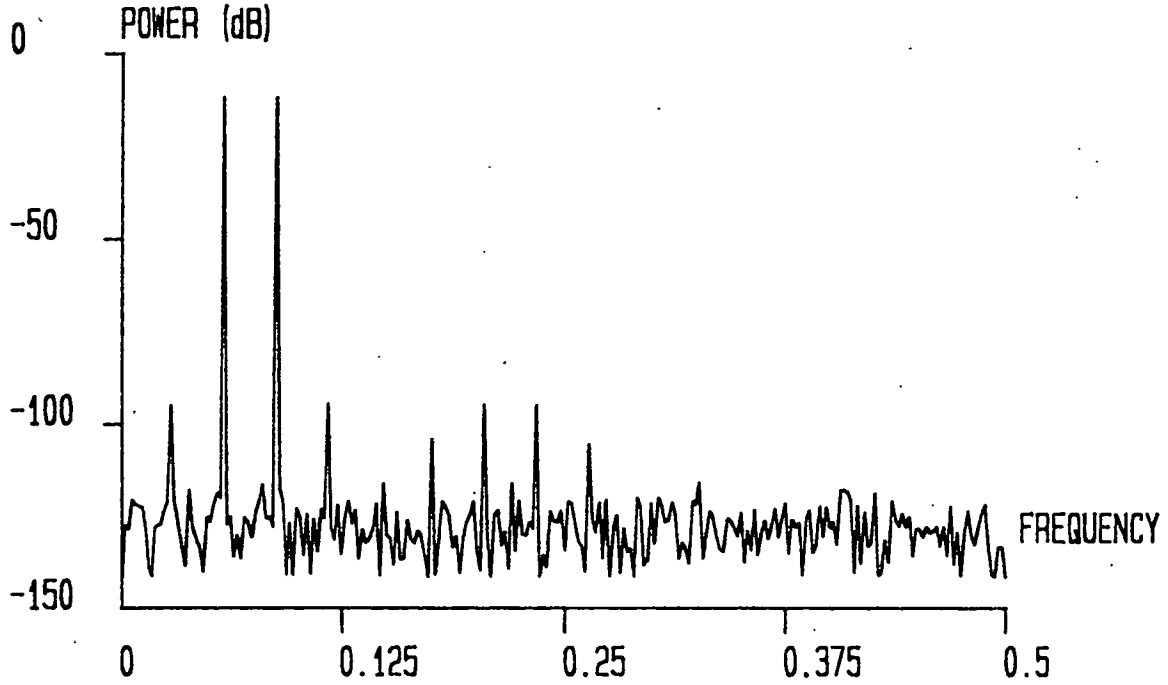

Figure 6.18: An FFT of a two tone sinusoid reconstructed by a 16 bit DAC with a nonlinearity of the type shown in figures 6.5 and 6.10.

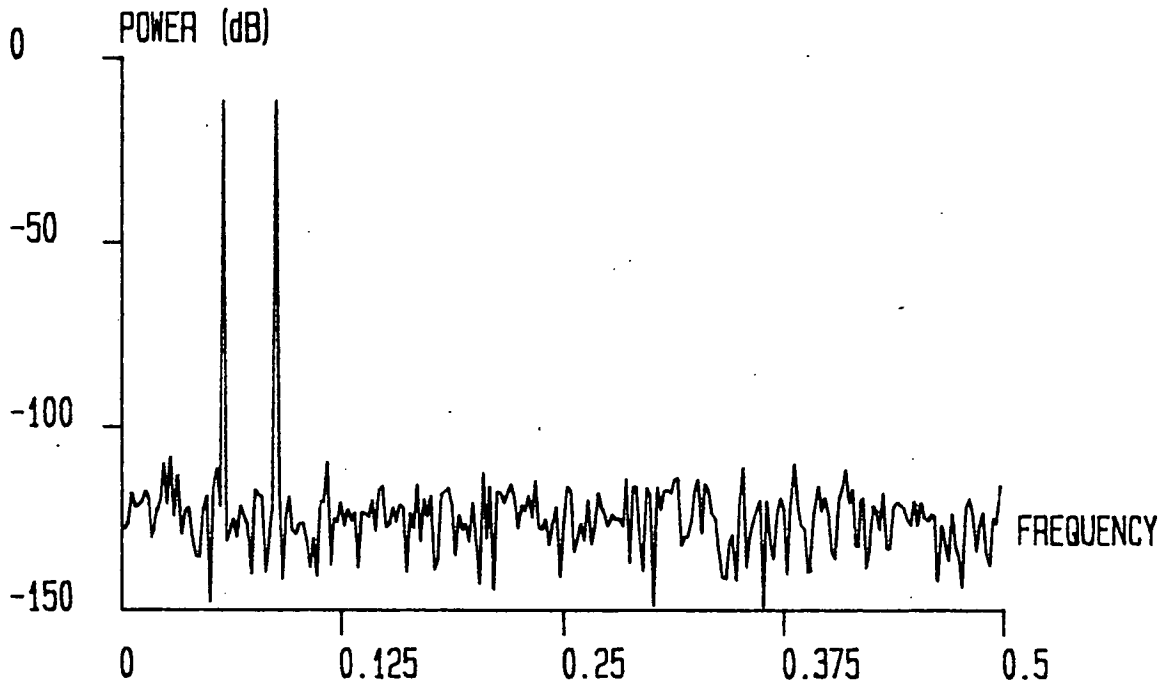

Figure 6.19: The mapped spectrum of the corresponding to the situation shown above, where the mapping has been implemented with only 5 bits.

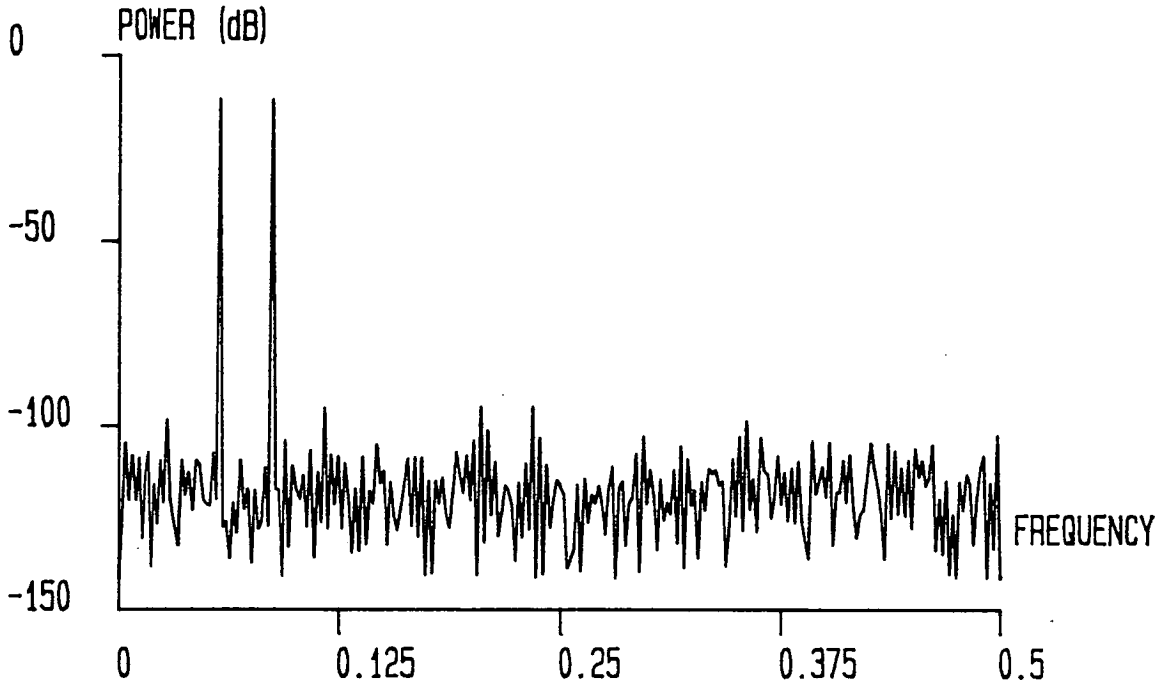

Figure 6.20: The unmapped spectrum of a 16 bit DAC with the complex nonlinearity of figure 6.16 and 6.17.

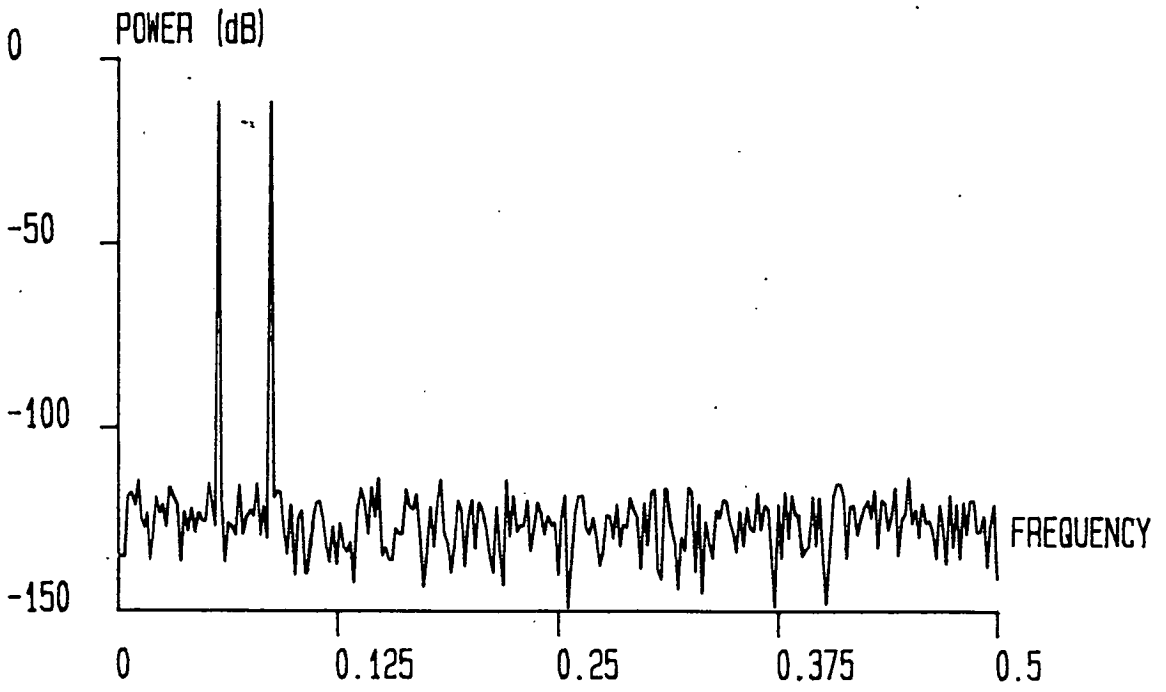

Figure 6.21: A full resolution mapping of the above converter shows the maximum performance that is available.

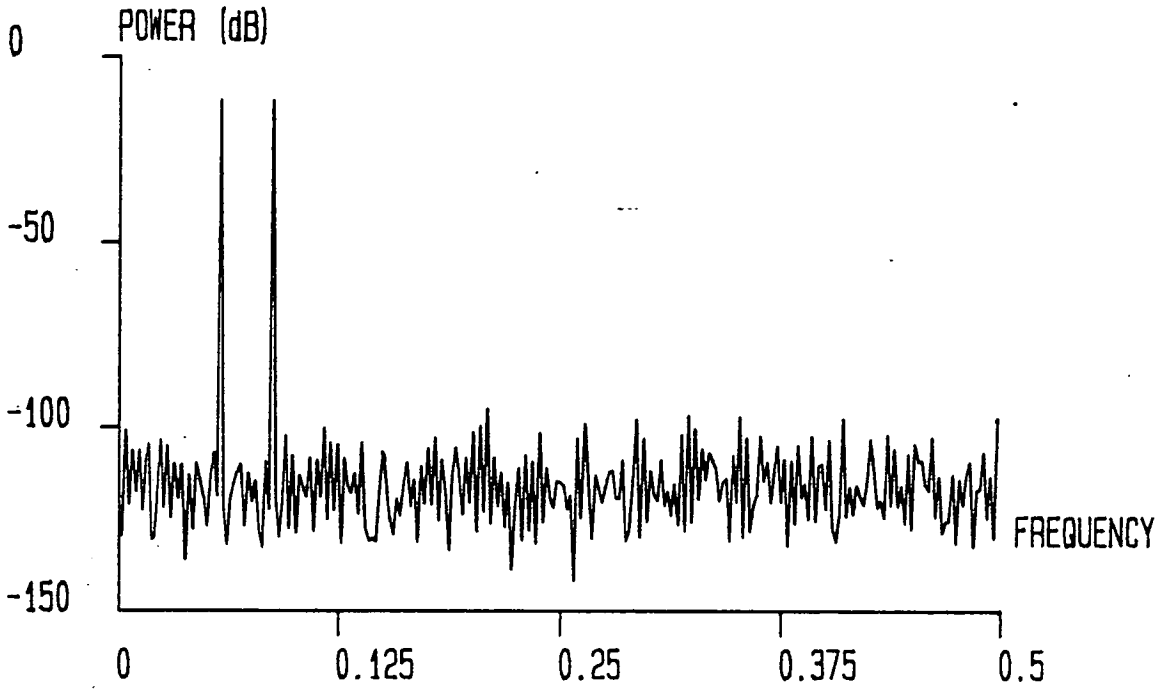

Figure 6.22: The previously distorted spectrum when it has been mapped with 5 address bits for the look-up table.

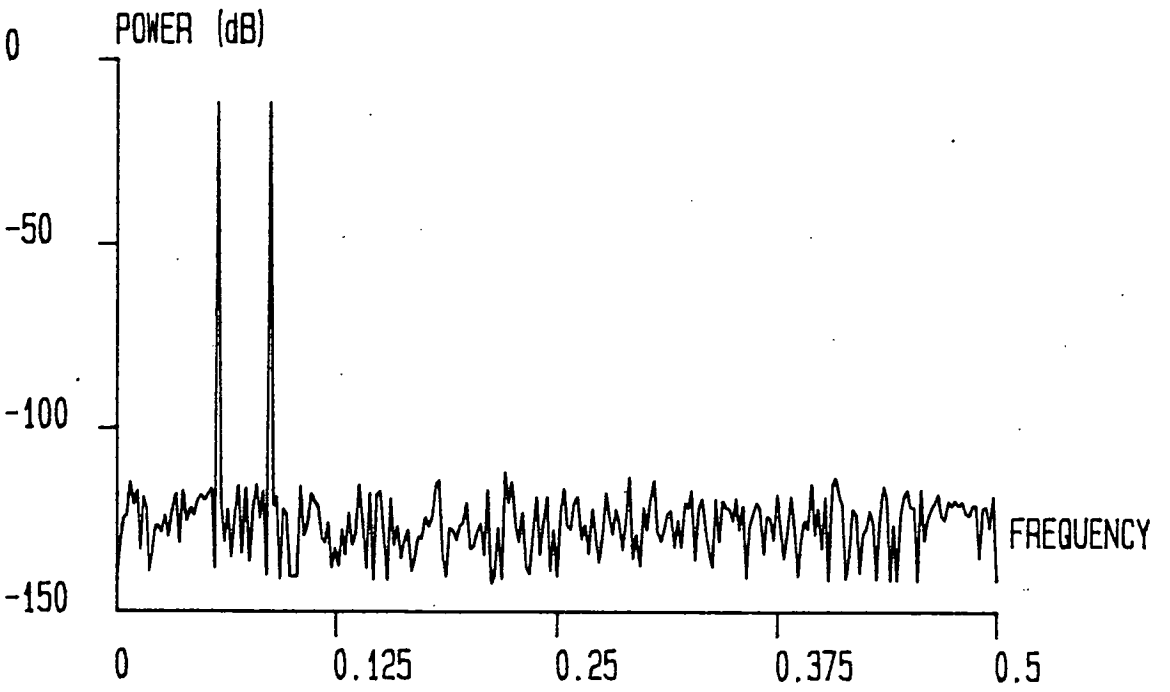

Figure 6.23: The mapped spectrum of the nonlinear converter of figure 6.20 when 10 addressing bits have been implemented.

Firstly, for 5 address bits, there has been some reduction in the magnitude of the harmonic peaks, particularly the the lower order distortions. Some of the higher order peaks, however, are actually emphasised by the linearisation process, whereas the noise floor has remained at approximately its unmapped level. Overall though, the mapped spectrum is probably at least qualitatively better than the unmapped.

Secondly, for the 10 address line mapping the distorted harmonics and intermodulation products have been almost totally eliminated and the noise floor has been lowered to almost the full resolution mapped level. Only a couple of noise like peaks remain to be linearised by the addition of the 6 least significant address lines.

As with the simpler, less severe nonlinearities, 10 address lines seems to give a very good level of reduced distortion, and there appears to be little, if any, point in using further address bits.

6.5. Conclusions

It has been shown that the threshold tracking mapping technique can be easily modified to enable its application to DAC linearisation with results of a comparable quality to those achieved for ADC systems. The training process for DAC linearisation is far simpler than that required for ADC's, needing only a counter and a good linearity ADC.

The application of reduced resolution mapping techniques to DAC's is also straightforward and again the results compare favourably with those for ADC's. Generally the use of 10 addressing bits, with 5 bit offsets, for the mapping look-up table gives virtually optimum performance, in terms of mean deviation, with a negligible increase in INL bounding and spectral distortion.

Chapter 7

CONCLUSIONS

The aim of this chapter is to draw conclusions from the work which has been done, and to show how the technique for ADC and DAC linearisation compares with other methods. A basic summary of the results achieved is followed by a discussion on their relevance and importance. Finally some suggestions for future work on similar topics and extensions to the current techniques are given.

7.1. Summary of Results

Chapter three introduced and examined the idea and effectiveness of using a digital mapping to eliminate ADC transfer function nonlinearity. The mapping, which is stored in a look-up table situated after the ADC in the overall system architecture, is generated from the characterisation of the converter nonlinearity provided by a form of code density testing. An algorithm was derived which could utilise the information contained in the idealised and sampled training signal pdf histograms.

Thorough testing of the linearisation technique, with a large selection of simulated ADC nonlinearities, has shown that the basic technique is valid. Extensive investigation was made into the size of the signal pdf's needed to obtain optimum performance from the mapping algorithm. It was determined that a minimum code density of 4 samples per histogram bin (assuming a uniformly distributed training signal) is required to get a level of mean deviation which is close to the maximum achievable. For a real system however, it would not be so easy to generate an

accurate uniformly distributed signal, as it was in simulation.

When a more easily generated sinusoid is used as the training signal, a larger number of signal samples is needed, to achieve the same level of mean deviation performance for the mapped converter. This, however, does not affect the maximum achievable levels of linearisation.

The improvements in device linearity which are possible with threshold tracking mapping are so good as to bound the mapped transfer function INL error between approximately ± 0.5 LSB, for all simulated nonlinearities. Harmonic distortions and intermodulation products are almost completely eliminated in Fourier transformed sampled signals, and quantisation noise floors are returned to their normal levels.

For high resolution converters i.e. over 12 bits, the proposed look-up table architecture was found to require too much storage. Two methods of reducing the system's memory requirements were developed. The first of these was to only store the mapping offsets, which required the introduction of a digital adder into the structure of the linearisation system. This allows a reduction of the memory requirement by a factor of three or more, for high resolution converters.

The second technique was to partition the ADC transfer function into many small segments, for which the mapping is assumed to be linear. This uses only the hardware needed for offset storage, providing that there is 2^n segmentation of the converter transfer function. This type of segmentation allows the most significant bits of the ADC output code to address the look-up table directly, without the need for any intermediate logic to determine which offset should be accessed.

The results obtained for mean deviation, integral nonlinearity and spectral distortion compared well with those for converters mapped with a fully addressed look-up table. The use of only the 10 most significant bits to address the look-up table was found to be the maximum number required for reasonable performance. This

enabled the memory requirement for a 16 bit converter to be reduced to less than 0.5% of that originally needed. This effect would be greater for higher resolution converters.

The small amount of memory now required is easily realisable on a single external integrated circuit, and would also not be too difficult to integrate onto the same piece of silicon as the ADC itself.

The hardware implementation of the threshold tracking algorithm proved to be very successful. The results validate all aspects of the computer simulations, with the measured harmonic patterns being very similar to those calculated from the raw sampled digital data. The mappings were found to work adequately for a wide range of real converter nonlinearities. This shows that the technique will work in real situations, and confirmed that the simulated nonlinearities exhibited sufficient realism to adequately model the expected distortions.

Modification of the threshold tracking mapping technique to enable its application to DAC linearisation was shown to be relatively easy, generating results of a comparable quality to those achieved for ADC systems. The training process for DAC linearisation is far simpler than that required for ADC's, needing only a counter and an ADC of good linearity.

The application of reduced resolution mapping techniques to DAC's is also straightforward and again the results compare favourably with those for ADC's. Generally the use of 10 addressing bits, with 5 bit offsets, for the mapping look-up table gives more or less optimum performance. There is however, a small increase in the limits bounding the mapped INL, and in the magnitude of the remaining spectral distortion.

7.2. Discussion

Overall, the work has shown that a practical system for the linearisation of analogue to digital and digital to analogue converters can be achieved by threshold tracking,

using a readily generated training signal and small amounts of simple additional hardware.

The limitations of the proposed system are that it always requires an adequately accurate training signal, and indeed needs to be able to operate in a training mode. If the converter transfer function is to be maintained as linear throughout the lifetime of the device, then it will require to be re-linearised to compensate for the drift of the device's analogue components with ageing. This is also true of temperature drift, although the problem can be circumvented, especially for monolithic converters, by means of training the device at the expected operating temperature and employing other measures to ensure that the actual temperature operated at is as near to predicted as possible.

This secondary linearisation procedure implies that it would be necessary to be able to generate the required training signals, internally to the converter. Other linearisation systems allow the ADC to be re-linearised at any time, some even during the normal operation of the device, and hence do not suffer from the drift problems outlined above.

The DAC linearisation scheme, on the other hand, has no such problems generating the appropriate training signal, as it is purely digital. It does however, require a very linear ADC, with at least as much resolution and linearity as the DAC being linearised, and hence it is impractical to attempt this form of converter re-linearisation in-situ.

All this means that the threshold tracking linearisation schemes would be most applicable to erradicating large scale nonlinearities, in that the loss of performance due to drift processes would be negligible when compared to the initial nonlinearity. This implies that the linearisation and storage of the mapping could be implemented as one of the latter stages in the production process. In this scenario, where full speed testing and characterisation is often already carried out, it would be of

relatively minor extra cost to add a linearisation step to the production process. This form of converter linearisation would eliminate the requirement for laser trimming in a large number of cases, and would also allow higher resolution/linearity devices to be packaged from common chip sources, than is currently possible.

7.3. Future Work

Possible future developments might centre around reducing the dependency of the system on a large number of accurately known training signal samples. This might allow the the training system to be contained within the converter. This then means that the converter could be re-linearised as often as necessary for the application to which it was being applied.

One method by which this could be achieved might be to replace the current look-up table mapping with an equalisation filter, which would adapt to have a transfer function which was a direct inverse of that of the converter. To achieve this the adaptive nonlinear filter would have to be trained by a known sequence, as is the case for other forms of adaptive equalisers. The length of this sequence might extend to only a few hundred samples. This is a significant reduction on the tens or hundreds of thousands of samples required in the threshold tracking method (for ADC linearisation).

That such a small number of samples could be sufficient to characterise the transfer function of the converter, can be deduced from the small number of mapping points used in the reduced resolution mapping simulations. These showed that even quite large nonlinearities can be considered to be linear over small segments of their transfer functions, and hence, any deviation of the inverted transfer function between the training samples would probably have relatively minor effects.

As already stated, the form of the required adaptive equaliser is nonlinear. The exact form which this nonlinearity would take, would be determined by the actual converter nonlinearity. Therefore it would be necessary to develop a general form

for the filter nonlinearity (eg. Chebychev polynomials and exponentials) which would be a good enough generalisation to model all realistic transfer function inverses. An initial study of this technique showed that the choice of the nonlinear filter, and its compatability with the modelled transfer function is crucial to the performance of the equaliser.

Of course, there may be no generally useful form that the nonlinear equaliser can take, which suggests that implementation might only be possible using neural networks.

REFERENCES

[1] K. Maio, M. Hotta, N. Yokozawa, M. Nagata, K. Kaneko and T. Iwasaki, "An Untrimmed D/A Converter with 14-bit Resolution", IEEE J-SC vol. 16, no. 6, pp. 616-621, Dec. 1981.

[2] J.J. Connolly et al, "A Monolithic 12b + Sign Successive Approximation Converter" Conference Proceedings IEEE ISSCC'80, pp.12-3, Feb. 1980.

[3] K. Bacrania, "A 12-bit Successive-Approximation-Type ADC with Digital Error Correction", IEEE J-SC, vol. 21, no. 6, pp. 1016-1025, Dec. 1986.

[4] H-S. Lee and D.A. Hodges, "Self-Calibration Technique for A/D Converters", IEEE Trans. CAS, vol. 30, pp. 188-190, Mar. 1983.

[5] H-S. Lee, D.A. Hodges and P.R. Gray, "A Self-Calibrating 12 bit 12 μs CMOS ADC", Conference Proceedings IEEE ISSCC'84, pp. 64-5, Feb. 1984.

[6] H-S. Lee, D.A. Hodges and P.R. Gray, "A Self-Calibrating 15 bit CMOS A/D Converter", IEEE J-SC, vol. 19, pp. 813-9, Dec. 1984.

[7] H-S. Lee and D.A. Hodges, "Accuracy Considerations in Self-Calibrating A/D Converters", IEEE Trans. CAS, vol. 32, no. 6, pp. 590-7, June 1985.

[8] T. Tsukada, K. Takagi, Y. Kita and M. Nagata, "An Automatic Error Cancellation Technique for Higher Accuracy A/D Converters", IEEE J-SC, vol. 19, no. 2, pp. 266-8, Apr. 1984.

[9] "Smart Analog", Crystal Semiconductor Corporation Databook, pp. 6-31,6-57, Jan. 1988.

[10] Z.G. Boyacigiller, B. Weir and P.D. Bradshaw, "An Error Correcting 14b/20μs CMOS A/D Converter", Conference Proceedings IEEE ISSCC'81, pp. 62-3, 1981.

[11] D. Draxelmayr, "A Self Calibration Technique for Redundant A/D Converters Providing 16b Accuracy", Conference Proceedings IEEE ISSCC'88, pp. 204-5,375, Feb. 1988.

[12] "ML2200, ML2208 Data Acquisition Peripheral", Micro Linear Data Sheets, Oct. 1988.

[13] "μP Compatible 12-Bit Plus Sign A/D Converter with Sample and Hold", Micro Linear Data Sheets, Dec. 1987.

[14] M. Armstrong, H. Ohara, H Ngo, C Rahim, A. Grossman and P.R. Gray, "A CMOS Programmable Self-Calibrating 13b Eight-Channel Analog Interface Processor", Conference Proceedings IEEE ISSCC'87, pp. 44-45,333-334, Feb. 1987.

[15] H. Ohara, H.X. Ngo, M.J. Armstrong, C.F. Rahim, and P.R. Gray, "A CMOS Programmable Self-Calibrating 13-bit Eight-Channel Data Acquisition Peripheral", IEEE J-SC, vol. 22, no. 6, pp. 930-938, Dec. 1987.

[16] N.J. Fine, "The Walsh Functions", Encyclopedic Dictionary of Physics, Pergamon Press, Oxford, 1969.

[17] E.A. Sloane, "Application of Walsh Functions to Converter Testing", IEEE Int. Test Conf. 1980, pp. 81-86, Nov. 1980.

[18] E.A. Sloane, "Applying Walsh Functions to Converter", Electronic Test. vol. 4, no. 4, Apr. 1981.

[19] E.A. Sloane, "Dynamic Converter Testing Using Walsh Functions", Conference Proceedings Semicon Europa'81, Mar. 1981.

[20] E.A. Sloane and P.W. Dodd, "A General Method for Increasing Converter Accuracy and Resolution", IEEE Int. Test Conf. 1983, pp. 598-605, 1983.

[21] Y. Matsuya, Y. Akazawa and A. Iwata, "High-Linearity and High-Speed CMOS 1-Chip A/D, D/A Converters. All Digital Linearity Error Correction (LECS)", Electronics and Communications in Japan Part 2, vol. 70, no. 2, pp. 73-84, 1987.

[22] Y. Akazawa, Y. Matsuya and A. Iwata, "New Linearity Error Correction Technology for A/D and D/A Converter LSI", Japanese Journal of Applied Physics, vol. 22, supplement 22-1, pp. 115-119, 1983.

[23] J. Doernberg, H-S. Lee and D.A. Hodges, "Full-Speed Testing of A/D Converters", IEEE Trans. Solid-State Circs. vol. SC-19, no. 6, pp. 820-827, Dec. 1984.

[24] M. Vanden Bossche, J. Schoukens and J. Renneboog, "Dynamic Testing and Diagnostics of A/D Converters", IEEE Trans. Circuits Syst. vol CAS-33, no. 8, pp. 775-785, Aug. 1986.

[25] G. Pretzl, "Dynamic Testing of High Speed A/D Converters", IEEE J. Solid-State Circs. vol. SC-13, pp. 368-371, 1978.

[26] O.J. Downing and P.T. Johnson, "A Method for the Assessment of the Performance of High Speed Analogue/Digital Converters", Electron. Lett. vol. 14, no. 8, pp. 238-240, Apr. 1978.

[27] R. Koch, B. Heise, F. Eckbauer, E. Engelhardt, J.A. Fisher and F. Parzefall, "A 12 bit Sigma-Delta Analog-to-Digital Converter with a 15-MHz Clock Rate", IEEE J-SC vol. 21, no. 6, pp. 1003-1010, Dec. 1986.

[28] P.F. Adams and R.P. Colbeck, "Considerations in the VLSI Circuit Implementation of 2B1Q Transceivers", Conference Proceedings IEEE ISCAS 88, vol. 1, pp. 861-4, Helsinki, June 1988.

[29] P. Bradsell, "Analogue-Digital Converter Requirements for Radar Systems", IEE Colloquium on 'Systems Aspects and Applications of ADC's for Radar, Sonar and Communications' Dig. No. 1987/92, pp. 8/1-8/3, London, April

1987.

[30] J.F. Roulston, "Digitiser Specification for Airborne Pulse Doppler Radar", IEE Colloquium on 'Systems Aspects and Applications of ADC's for Radar, Sonar and Communications' Dig. No. 1987/92, pp. 6/1-6/13, London, April 1987.

[31] B. Schweber and W. Labs, "A/D Converters: 16 bit Resolution isn't Always Enough" ,I & C S (USA), vol. 61, part 2, pp. 51-2, Feb. 1988.

[32] T.H. Pearce and A.C. Baker, "Analogue to Digital Conversion Requirements for H.F. Radio Receivers", IEE Colloquium on 'Systems Aspects and Applications of ADC's for Radar, Sonar and Communications' Dig. No. 1987/92, pp. 4/1-4/5, London, Nov. 1987.

[33] B.M. Sosin, "HF Communication Receiver Performance Requirements and Realisation", IERE Journal, vol. 41, no. 7, pp. 321-9, July 1971.

[34] E.C. Dijkmans and P.J.A. Naus, "An Experimental 16 bit A/D Converter for Digital Audio Applications", IEE Colloquium on 'Advanced A/D Conversion Techniques', Coll. Dig. No. 1987/48, pp. 5/1-5/4, London, April 1987.

[35] P.G.A. Jespers, "Integrated D/A and A/D Converters", Microelectronics Journal, vol. 19, no. 4, 1988.

[36] D. Shear, "Monolithic High-Resolution ADC's" EDN, pp. 116-130, May 12, 1988.

[37] H. Chenhall, "Seeking the ultimate in A/D precision", Electronic Product Design (USA) pp. 29-42, Dec. 1987.

[38] B.J. Rodgers and C.J. Thurber, "A Monolithic ±5½-Digit BiMOS A/D Converter", IEEE J-SC, vol. 24, no. 3, pp. 617-626, June 1989.

[39] A. Kayanuma et al, "An Integrated 16b A/D Converter for PCM Audio Systems", Conference Proceedings IEEE ISSCC'81, pp. 56-7, Feb. 1981.

[40] "AD368/AD369 - Complete 12-bit A/D Converters with Programmable Gain", Analog Devices Data Book, pp. 3/13-3/22, 1988.

[41] "AD1332 - Complete 12-bit Sampling A/D Converter for Digital Signal Processing", Analog Devices Data Book, pp. 3/175-3/178, 1988.

[42] "AD7870 - Fast Complete 12-bit A/D Converter with ac Performance up to 50kHz", Analog Devices Data Book, pp. 3/317-3/322, 1988.

[43] "MP574 - 12 bit Microprocessor Compatible A/D Converter", Micro Power Systems Data Sheets, 1989.

[44] "ICL7115 - Fast 14-bit Microprocessor Compatible ADC", Harris Semiconductor Corporation Data Sheets, 1989.

[45] "ADC1225 - 12-bit Plus Sign μP Compatible A/D Converter", National Semiconductor Corporation Data Sheets, 1989.

[46] F. Ueno, T. Inoue, K. Sugitani and S. Araki, "Reduction Methods of Capacitor Mismatch Errors in Switched-Capacitor A/D, D/A Converters", Conference Proceedings IEEE ISCAS'88, vol. 3, pp. 2813-6, 1988.

[47] "AD1380 - Low Cost 16-bit Sampling ADC", Analog Devices Data Book, pp. 3/187-3/188, 1988.

[48] "PCM78P - 16 bit High Performance Audio A/D Converter", Burr Brown Corporation Data Sheets, 1989.

[49] "AD374 - Low Power True 16-bit A/D Converter", Analog Devices Data Book, pp. 3/23-3/30, 1988.

[50] "AD376 - Complete High Speed 16-bit A/D Converter", Analog Devices Data Book, pp. 3/31-3/38, 1988.

[51] F. de Jager, "Delta Modulation, A method of PCM Transmission Using the 1-Unit Code", Phillips Res. Refs. 7, pp. 442-466, Nov. 1952.

[52] H. Inose, J. Yasuda and J. Murakami, "A Telemetering System by Code

Modulation - Sigma-Delta Modulation", IRE Trans. PGSET 8, pp. 204, Sep. 1962.

[53] D.F. Simpson and R.A. Belcher, "Feedback Coding Techniques - Review and Implementation", IEE Coll. Dig. no. 1989/72, pp. 3/1-3/8, May 1989.

[54] D. Cyganski and N.J. Krikelis, "Optimum Design, Performance Evaluation, and Inherent Limitations of DPCM Encoders", IEEE Trans. CAS, vol. 25, no. 7, pp. 448-460, Jul. 1978.

[55] J.E. Iwersen, "Calculated Quantizing Noise of Single Integration Delta Modulation Coders", Bell System Technical Journal, pp. 2359-2389, Sep. 1969.

[56] M.I. Mathew and C.P. Lewis, "A Review of Sigma-Delta Modulation Structures", IEE Coll. Dig. no. 1989/72, pp. 4/1-4/8, May 1989.

[57] M.J. Hawksford, "N'th Order Recursive Sigma-ADC Machinery at the Analogue-Digital Gateway", 78th AES Convention, May 1985.

[58] O. Agazzi and A.A. Adan, "An Analog Front End for Full Duplex Digital Transceivers Working on Twisted Pairs", IEEE J-SC, vol. 24, no. 2, pp. 229-240, Apr. 1989.

[59] B. Roessler and E. Wolter, "CMOS Analog Front End of a Transceiver with Digital Echo Cancellation for ISDN", IEEE J-SC, vol. 23, no. 2, pp. 311-7, Apr. 1988.

[60] Y. Matsuya et al, "A 16-bit Oversampling A-to-D Conversion Technology Using Triple Integration Noise Shaping", IEEE J-SC, vol. 22, no. 6, pp. 921-9, 1987.

[61] R.W. Adams, "Design and Implementation of an Audio 18 bit Analog to Digital Converter Using Oversampling Techniques", Journal of Audio Engineering Society, vol. 34, pp. 153-166, Mar. 1986.

[62] S.R. Norsworthy, I.G. Port and H.S. Fetterman, "A 14-bit 80-kHz Sigma-Delta A/D Converter: Modeling, Design and Performance Evaluation", IEEE J-SC, vol. 24, no. 2, pp. 256-266, Apr. 1989.

[63] K. de Graff et al, "Ga As Technology for Analog to Digital Conversion", IEEE Ga As Symposium, pp. 205-8, Oct. 1986.

[64] M.W. Hauser and R.W. Brodersen, "Circuit and Technology Considerations for Delta-Sigma A/D Converters", Conference Proceedings IEEE ISCAS'86, pp. 1310-5, 1986.

[65] "AD770 - 8-bit 200 MSPS ADC", Analog Devices Data Sheets, pp. 3/147-154, 1988.

[66] "MC10319P - 8-bit High Speed Flash Converter", Motorola Data Sheets, 1989.

[67] "ADC302 - 8-bit Video Flash Converter", Datel Corporation Data Sheets, 1989.

[68] "AD9002 - Ultrahigh Speed 8-bit Monolithic ADC", Analog Devices Data Sheets, pp. 3/339-344, 1988.

[69] "MP7685KD - 11-bit Low Power Flash A/D Converter", Micro Power Systems Data Sheets, 1989.

[70] "AD310 - 10-bit 20-MHz Flash A/D Converter" Datel Corporation Data Sheets, 1989.

[71] K.C. Wang, P.M. Asbeck, M.F. Chang, G.J. Sullivan and D.L. Miller, "A 4-bit Quantizer Implemented with Al Ga As/Ga As Heterojunction Bipolar Transistors", IEEE Ga As IC Symposium, pp. 83-6, 1987.

[72] T. Ducourant, M. Binet, J.C. Baelde, C. Rocher and J.M. Gibereau, "3 GHz, 150 mW, 4 bit Ga As Analogue to Digital Converter", IEEE Ga As IC Symposium, pp. 209-212, 1986.

[73] K. Poulton, J.J. Corcoran and T. Hornak, "A 1-GHz 6-bit ADC system",

IEEE J-SC, vol 22, no. 6, pp. 962-970, Dec. 1987.

[74] J. Kleks et al, "A 4-bit Single Chip Analog-to-Digital Converter with a 1.0 GHz Analog Input Bandwidth", IEEE Ga As IC Symposium, pp. 79-82, 1987.

[75] T. Wakimoto, Y. Akazawa and S. Konaka, "Si Bipolar 2-GHz 6-bit Flash A/D Conversion LSI", IEEE J-SC, vol. 23, no. 6, pp. 1345-1350, Dec. 1988.

[76] L.E. Larson, "High-Speed Analog-to-Digital Conversion with Ga As Technology: Prospects, Trends and Obstacles", Conference Proceedings IEEE ISCAS'88, pp. 2871-8, June 1988.

[77] T. Matsuda et al. "A New Bi-CMOS Structure for Analog/Digital Systems" European Solid State Device Research Conf. pp. 174, 1981.

[78] Y. Kowase, Y. Yanagawa, T. Inaba, J. Mameda, N. Horie, S. Ueda and M. Nagata, "A Bi-CMOS Analog/Digital LSI with the Programmable 280 bit SRAM", Conference Proceedings IEEE CICC'85, pp. 170-3, 1985.

[79] P.W. Li, M.J. Chin, P.R. Gray and R. Castello, "A Ratio Independent Algorithmic Conversion Technique", IEEE J-SC, vol. 19, pp. 828-836, Dec. 1984.

[80] C-C. Shih and P.R. Gray, "Reference Refreshing Cyclic Analog-to-Digital and Digital-to-Analog Converters", IEEE J-SC, vol. 21, pp. 544-554, Aug. 1986.

[81] T.A.C.M. Claasen, W.F.G. Mecklenbrauker, J.B.H. Peek and N. Van Hurck, "Signal Processing Method for Improving the Dynamic Range of A/D and D/A Converters", IEEE Trans. ASSP, vol. 28, no. 5, pp. 529-538, Oct. 1980.

[82] W. Freeman, "Flexible TSC500 ADC Simplifies Design Tradeoffs", Teledyne Semiconductor Data Sheets, 1985.

[83] M.T. Abuelma'atti, "Effect of Nonmonotonicity on the Intermodulation Performance of A/D Converters", IEEE Trans. COM, vol. 33, no. 8, pp. 839-843, Aug. 1985.

[84] O.B. Semenov, "ADC and DAC Nonlinearity Specifications in Audio Signal Processing and Transmission Systems", Telecommunications and Radio Engineering in the Soviet Union, vol. 40, no. 12, pp. 95-8, 1987. translated from Radiotekhnika, no. 11, pp. 28-31, 1985.

[85] J.W. Cook, 'The Digital Subscriber Loop Development System: A Description of the Backplane, Card, Common Circuitry, Inter-card Communication and Timing", British Telecom Research and Technology Internal Memorandum RT42 88/038, June 1988.

Appendix A

THRESHOLD TRACKING SOFTWARE

```
# include <stdio.h>
# include <math.h>

# define begin {
# define end }
# define boolean int
# define false 0
# define true 1
# define NULL 0

# define levels 65536   /* quantisation levels of main DAC */
# define PI 3.1415926

int mapping[levels],    /* the mappings produced */
    offset[levels],     /* stores mapping point offsets */
    ref_map[levels],    /* the reference mapping for current sample size */
    non_lin[levels],    /* distorted pdf */
    temp,
    new,
    last,
    map_points,
    points=1024,
    i;

double threshold=0.5,   /* the fraction of linear pdf used for d.t. */
    step=2.0/levels,    /* ideal quantisation step */
    base_error=0.0,             /* the unmapped error */
    samples,
    inc,
    temp2,
    temp3,
    xx,yy,
    distortion();

char option,*errfile=NULL,*samfile=NULL,*mapfile=NULL,
       *msfile=NULL,*sigfile=NULL;

FILE *err_fp,*fopen(),*sam_fp,*map_fp,*ms_fp,*sig_fp;

boolean not_mapped,start_of_loop;

/********** SUBROUTINES ***********/

rounded(a) double a;
{
int result;
if(a<0.0) return((int)(a-0.5));
else return((int)(a+0.5));
}

find_error()            /* adds up the total error in a mapping */
begin
double ni,error=0.0,increment;  /* increment is a single error square */
err_fp=fopen(errfile,"a");
for(i=0;i<levels;++i)
    begin
    increment=((i+1)*step/(distortion((mapping[i]+1)*step-1.0)+1.0)-1.0);
    increment*=(i+1)*step;
    increment*=step;
```

```
    error+=fabs(increment);
    end;
if(error!=0.0)
    {
    error/=levels;              /* to give the mean square error */
    error=10*log10(error);
    }
if(base_error!=0.0)
    error-=base_error;
    else  base_error=error;
fprintf(err_fp,"%6d MP %6.1f S %6.3f dB.\n",map_points,samples,error);
increment=0.0;
fclose(err_fp);
end

thresh_track_map()              /* Algorithm 3,ver. 2 mapping subroutine */
begin
int a,b=0,                      /* the array pointer variables */
    non_sum=0;
double diff,                    /* the difference between the sums */
    lin_sum=0.0,
    limit=threshold*samples/levels;     /* decision level for mapping */
boolean done_loop;              /* ensures only one while loop entered */

for(a=0;a<levels;++a)
    begin
    if(b<levels)
        begin
        lin_sum+=(samples/levels);      /* update the linear sum */
        non_sum+=non_lin[a];    /* update the non-linear sum */
        done_loop=false;
        diff=lin_sum-(double)non_sum;   /* calculate the difference */
        if(diff>limit)  /* a map needs to be adjusted */
                begin           /* adjust pointer as requuired */
                lin_sum-=(samples/levels);/* sub the last pdf height and */
                b--;            /* decrement the pointer */
                done_loop=true; /* this ensures only one loop entered */
                end;
        while((diff<(0-limit))&&(done_loop==false))
                begin           /* non_sum - lin_sum > decision threshold */
                lin_sum+=(samples/levels);      /* add the next bin frequency */
                b++;
                diff=lin_sum-(double)non_sum;
                                /* recalculate the difference */
                end;
        if(b<0)                         /* test for negative maps */
                begin
                lin_sum=0;      /* catches errors at start of procedure */
                b=0;
                end;
        mapping[a]=b++;         /* mapping is recorded and linear pointer
                                                is incremented */
        end             /* non-linear pointer incremented by for loop */
    else  mapping[a]=levels-1;
    end
end

main(argc,argv) int argc; char *argv[];
begin

while(--argc>0 && (*++argv)[0]=='-')
    begin
    option = *(argv[0]+1);
    switch(option)
        begin
        case 'm':               /* contains Map Point list (last 0) */
            mapfile =  *++argv;
            argc--;
            break;
        case 'e':               /* error storage file */
            errfile =  *++argv;
            argc--;
            break;
```

```
        case 'o':               /* output signal (unmapped file) */
            sigfile =  *++argv;
            argc--;
            break;
        case 'd':               /* output harmonics (mapped) */
            msfile =  *++argv;
            argc--;
            break;
        case 's':               /* sample sizes (last 0.0) */
            samfile =  *++argv;
            argc--;
            break;
        default:
            fprintf(stderr,"inlplot: illegal option %c\n",option);
            exit(1);
            break;
        end;
    end;
if((samfile==NULL)||(mapfile==NULL)||(errfile==NULL))
    begin
    fprintf(stderr,"inlplot: not enough files specified\n");
    exit(1);
    end;
if(!(ms_fp=fopen(msfile,"w")))
    begin
    fprintf(stderr,"inlplot: can't open \"%s\"",msfile);
    exit(1);
    end;
if(!(sig_fp=fopen(sigfile,"w")))
    begin
    fprintf(stderr,"inlplot: can't open \"%s\"",sigfile);
    exit(1);
    end;
if(!(err_fp=fopen(errfile,"w")))
    begin
    fprintf(stderr,"inlplot: can't open \"%s\"",errfile);
    exit(1);
    end;
fclose(err_fp);
if(!(map_fp=fopen(mapfile,"r")))
    begin
    fprintf(stderr,"inlplot: can't open \"%s\"",mapfile);
    exit(1);
    end;
if(!(sam_fp=fopen(samfile,"r")))
    begin
    fprintf(stderr,"inlplot: can't open \"%s\"",samfile);
    exit(1);
    end;
for(i=0;i<levels;i++)  mapping[i]=i;
find_error();                           /* unmapped error */
fscanf(sam_fp,"%lf",&samples);
while(samples!=0.0)
    begin
    inc=2.0/(samples+1.0);
    temp2=inc/2.0-1.0;                  /* setting up sampling */
    for(i=0;i<levels;i++)  non_lin[i]=0;/* zeroing pdf */
    while(temp2<1.0)
        (
        temp3=distortion(temp2);
        temp3=(temp3+1.0)*levels/2.0;
        temp=(int)temp3;
        ++non_lin[temp];                /* getting pdf histogram */
        temp2+=inc;
        )
    thresh_track_map();
    map_points=0;
    find_error();                       /* full mapped performance */
        xx=0.0;
        yy=0.0;
        for(i=1;i<(points+1);i++)
            begin
            temp3=sin(xx)+sin(yy);
            temp3*=0.375;
            temp2=distortion(temp3);
            temp3=(temp2+1.0)*levels/2.0;
```

```
                temp=(int)temp3;
                temp2=(temp+0.5)*step-1.0;
                fprintf(sig_fp,"%f\n",temp2);
                temp2=(mapping[temp]+0.5)*step-1.0;
                fprintf(ms_fp,"%f\n",temp2);
                xx+=180.0*PI/points;
                yy+=120.0*PI/points;
                if(xx>(2.0*PI))  xx-=(2.0*PI);
                if(yy>(2.0*PI))  yy-=(2.0*PI);
                end
        for(i=0;i<levels;i++)  ref_map[i]=mapping[i];
        fscanf(map_fp,"%d",&map_points);
        while(map_points!=0)
            {
            last=0;
            for(i=1;i<=map_points;i++)
                {                               /* array index of Map Points */
                temp=((int)(i*levels/map_points))-1;
                new=ref_map[temp]-temp;     /* offset at the Map Point */
                offset[i-1]=rounded((double)((new+last)/2.0));
                last=new;                       /* average between two MP's */
                }
            for(i=0;i<levels;i++)
                {                   /* convert "mapping" for "find_error" */
                mapping[i]=i+offset[(int)(i*map_points/levels)];
                if(mapping[i]>(levels-1)) mapping[i]=levels-1;
                    else if(mapping[i]<0) mapping[i]=0;
                }
            find_error();
            xx=0.0;
            yy=0.0;
            for(i=1;i<(points+1);i++)
                begin
                temp3=sin(xx)+sin(yy);
                temp3*=0.375;
                temp2=distortion(temp3);
                temp3=(temp2+1.0)*levels/2.0;
                temp=(int)temp3;
                temp2=(temp+0.5)*step-1.0;
                fprintf(sig_fp,"%f\n",temp2);
                temp2=(mapping[temp]+0.5)*step-1.0;
                fprintf(ms_fp,"%f\n",temp2);
                xx+=180.0*PI/points;
                yy+=120.0*PI/points;
                if(xx>(2.0*PI))  xx-=(2.0*PI);
                if(yy>(2.0*PI))  yy-=(2.0*PI);
                end
            fscanf(map_fp,"%d",&map_points);
            }
        fclose(map_fp);
        map_fp=fopen(mapfile,"r");          /* reset file pointer */
        fscanf(sam_fp,"%lf",&samples);
        end
    fclose(sam_fp);
    fclose(map_fp);
    fclose(ms_fp);
    fclose(sig_fp);
end
```

Appendix B

HARDWARE IMPLEMENTATION DETAILS

This Appendix contains detailed information on how the linearisation hardware was implemented. This includes several circuit diagrams, and functional structure diagrams. The software required to operate the system is contained in Appendix C.

1. Circuit Diagram of Nonlinear ADC

Figure B.1 shows the main circuit diagram for the ADC with adjustable linearity. This was achieved by the use of variable resistors in the construction of the R-2R reference ladder.

2. Circuit Diagram of Linear ADC

Figure B.2 is the circuit diagram for the CS5014 ADC, excluding the power supply decoupling components, for the sake of clarity. This 14 bit resolution device was linear to 14 bits and would, therefore, be very accurate when only 12 of the output bits were being utilised, as in this case.

3. High Level Interconnect of TTM Card

The block level interconnect of the circuit components for the TTM card is shown in figure B.3, where the data/control flow of the basic structure can be easily seen.

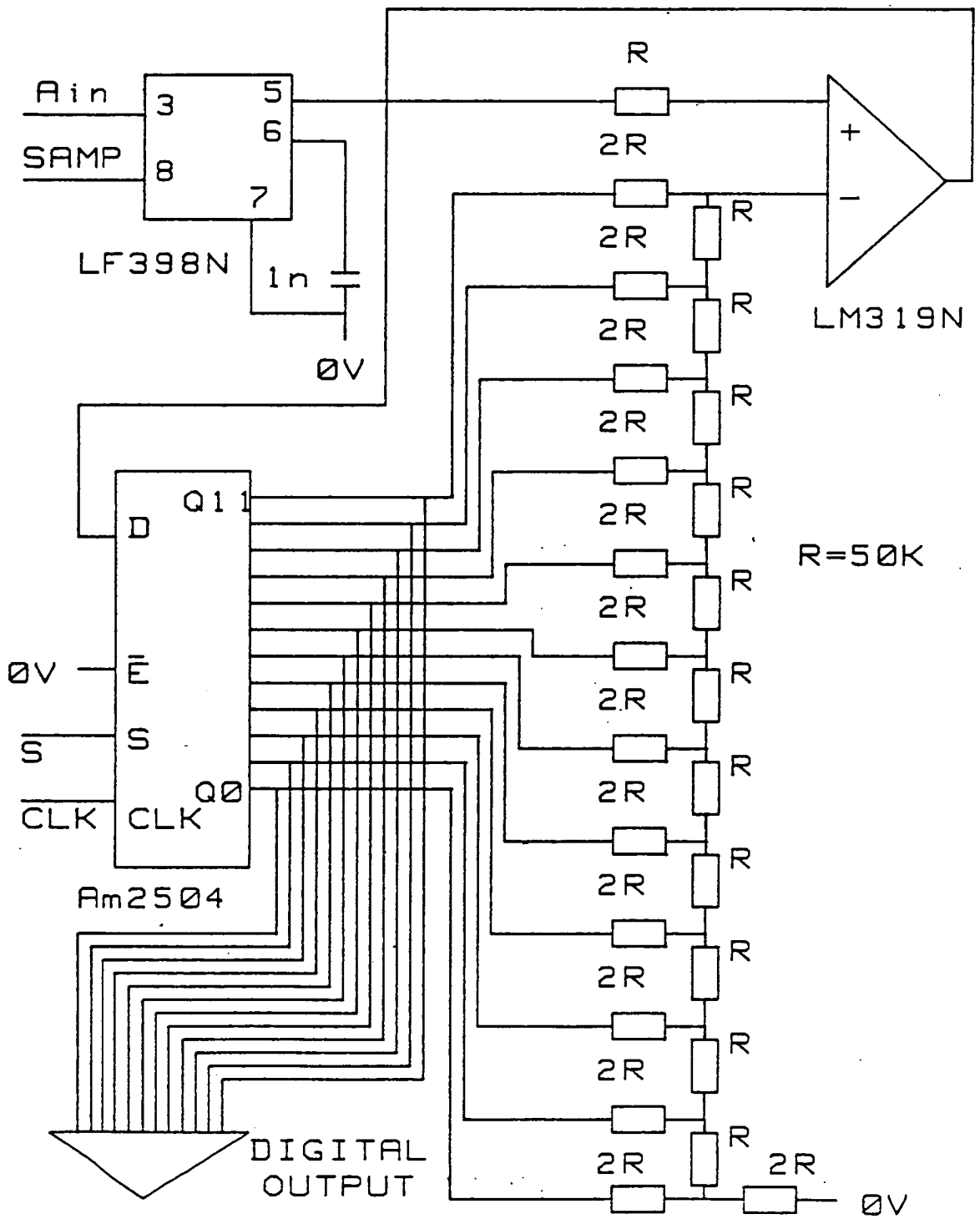

Figure B.1: Circuit diagram of the nonlinear successive approximation ADC.

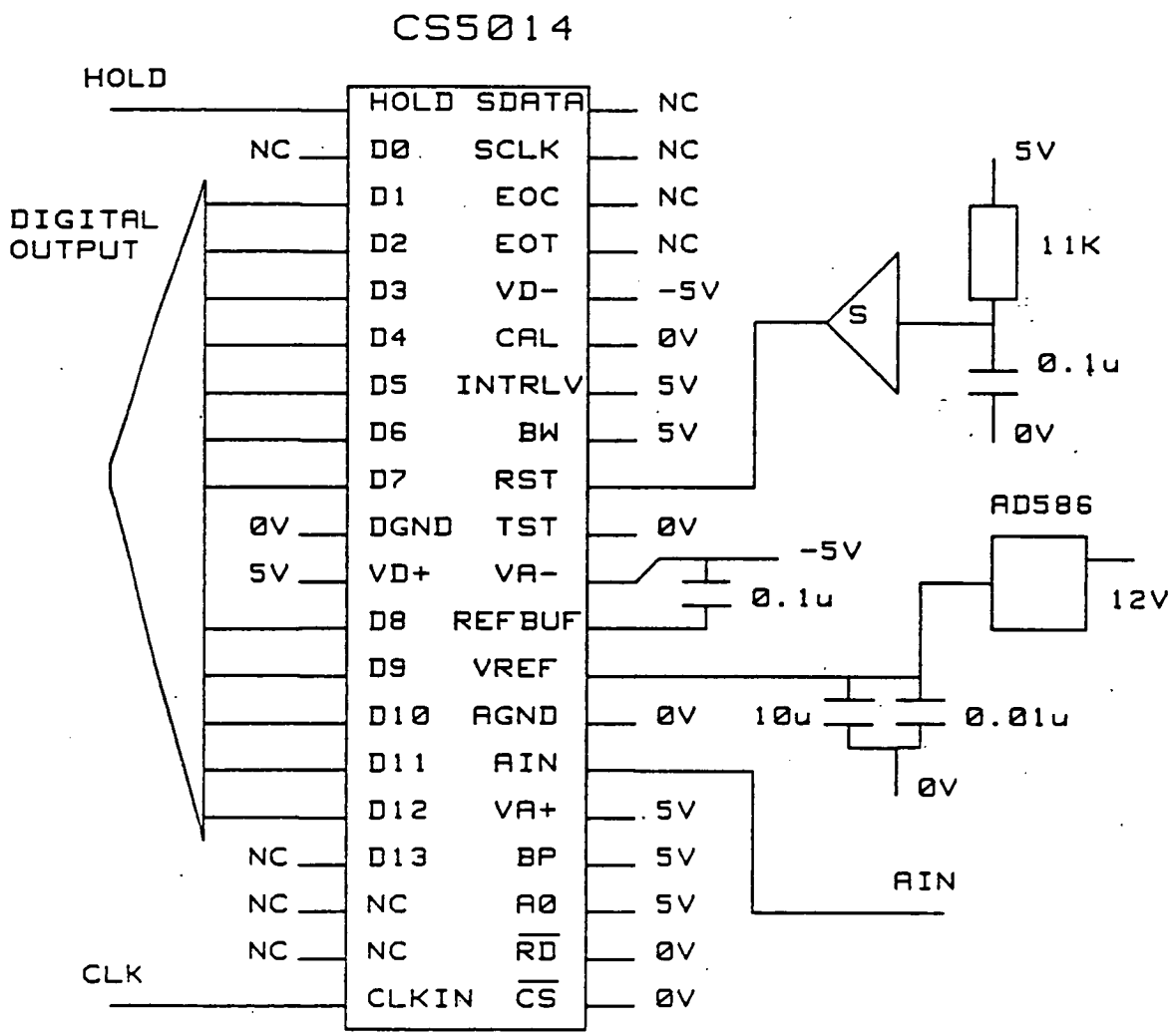

Figure B.2: Circuit diagram of the CS5014, highly linear converter.

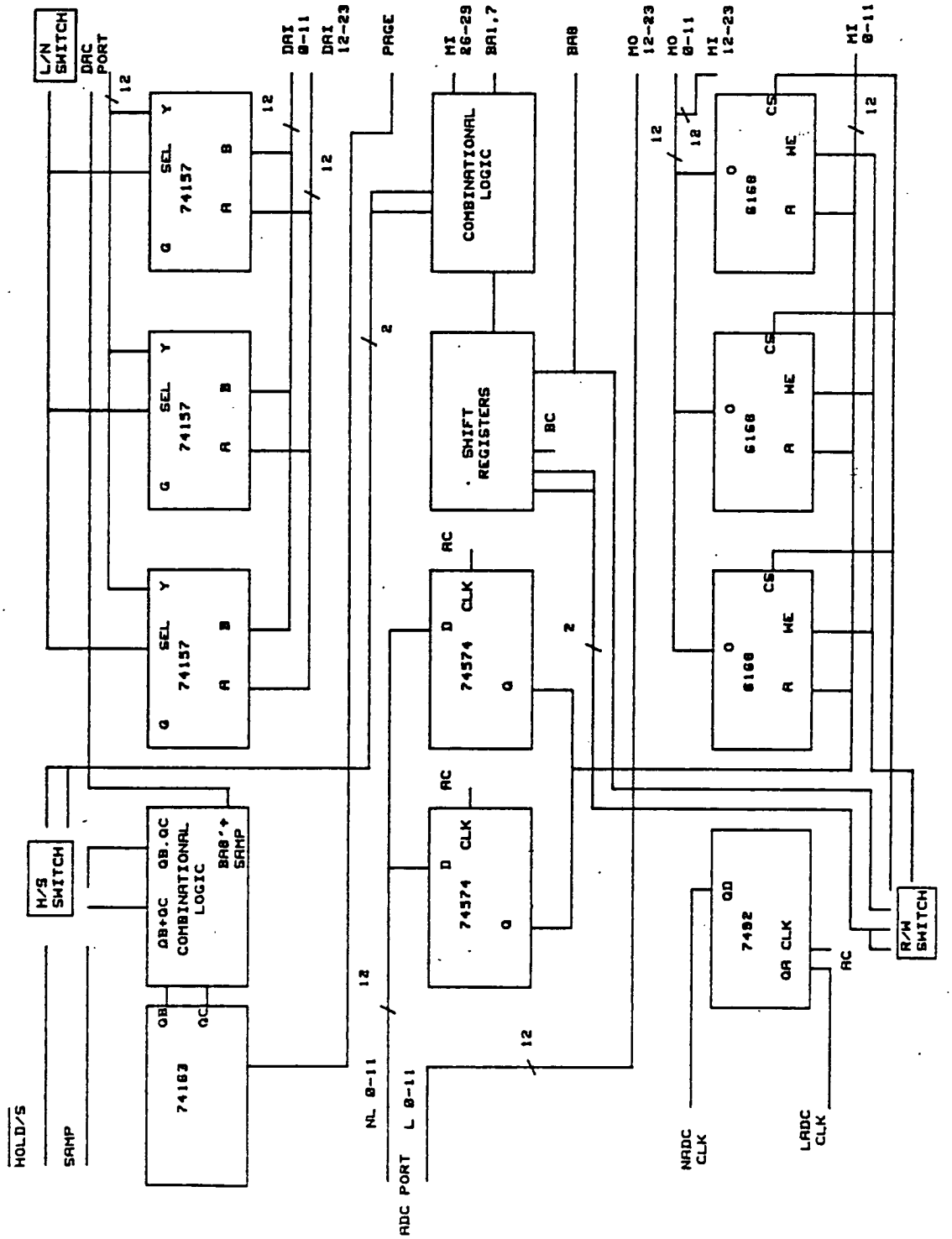

Figure B.3: Block diagram of the TTM card circuitry.

4. The TTM Card System Interface

In order that the operation of the TTM card can be more easily understood, it is necessary to know something of the operation of the system in which it operates. At the heart of the DDS system is its backplane, and the protocols used in its operation, and it is these signals, in as much as they affect the TTM card, which are described here.

Each cycle of the bus signals was divided into 32 transactions in which data could be sent or received to/from the data bus. The transactions were further divided into two sets of sixteen by the use a line called PAGE which was low for the first set and high for the second. For ease of decoding individual transactions there were 16 signals labelled BA0 to BA15 which went low during the appropriate bus transaction slot. A more detailed view of transaction timing can be gained from figure B.4.

Two clock signals were also supplied by the backplane. These were the bus clock (BC) and the address clock (AC). BC was the fundamental DDS clock and had a frequency of 10.560 MHz and AC was a quarter of that. AC was used, by all DDS cards, primarily to latch incoming data from the bus. This means that all output registers and latches had to be loaded well in advance of AC going high. Figure B.5 shows the relationship of these two clocks in more detail and also shows the times that input/output data on the card data bus (DA) was valid.

Now that the timing of bus transactions has been explained it is possible to understand the DDS interface circuitry. As can be seen in figure B.6 the interface to the system backplane involved several sets of latches and buffers. The timing signal buffers were 74-244 unidirectional buffers, whereas the DDS bus buffers were 74-245 circuits which were bidirectional and controlled by one signal line. As the 74-245 defaulted to the card input condition (DIR pin high) care had to be taken to ensure that the 29823 bus driver circuits were only output enabled when the

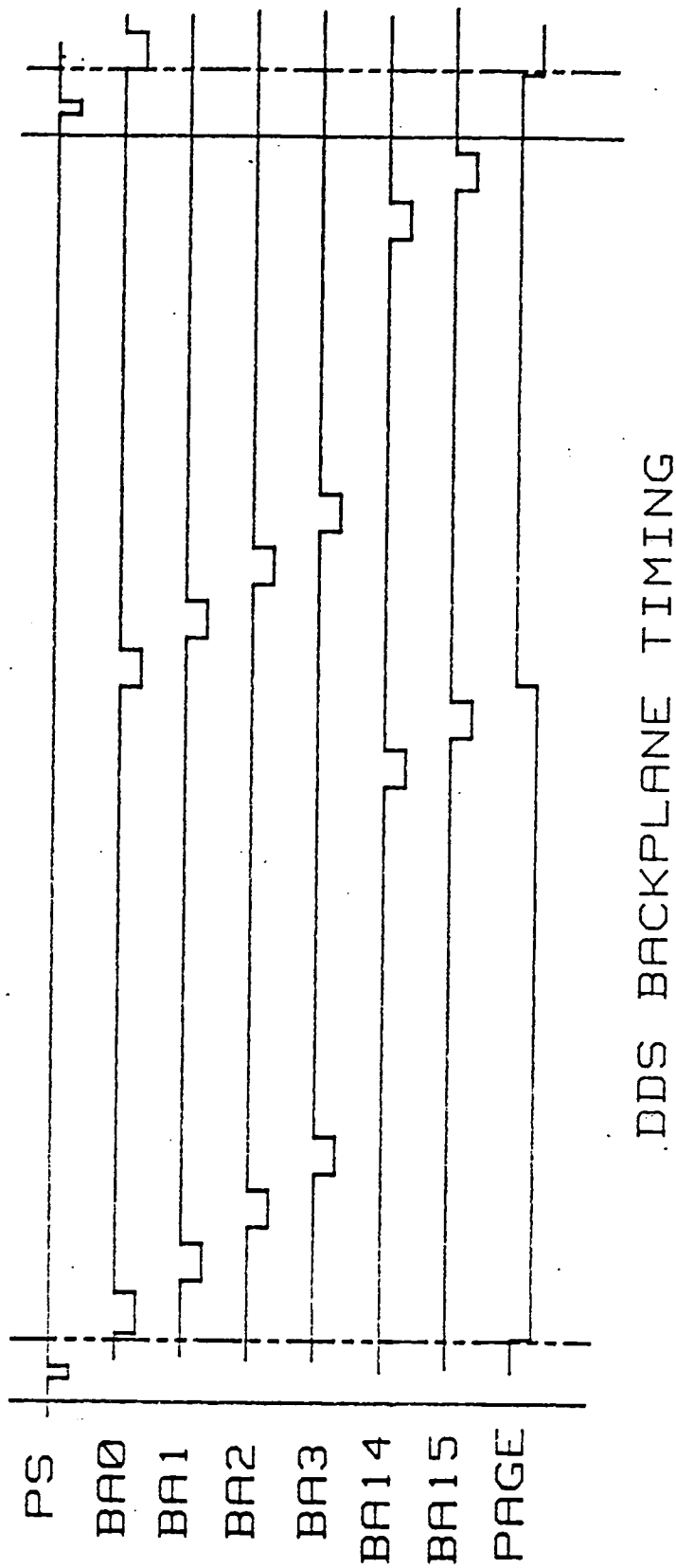

Figure B.4: DDS backplane transaction timing.

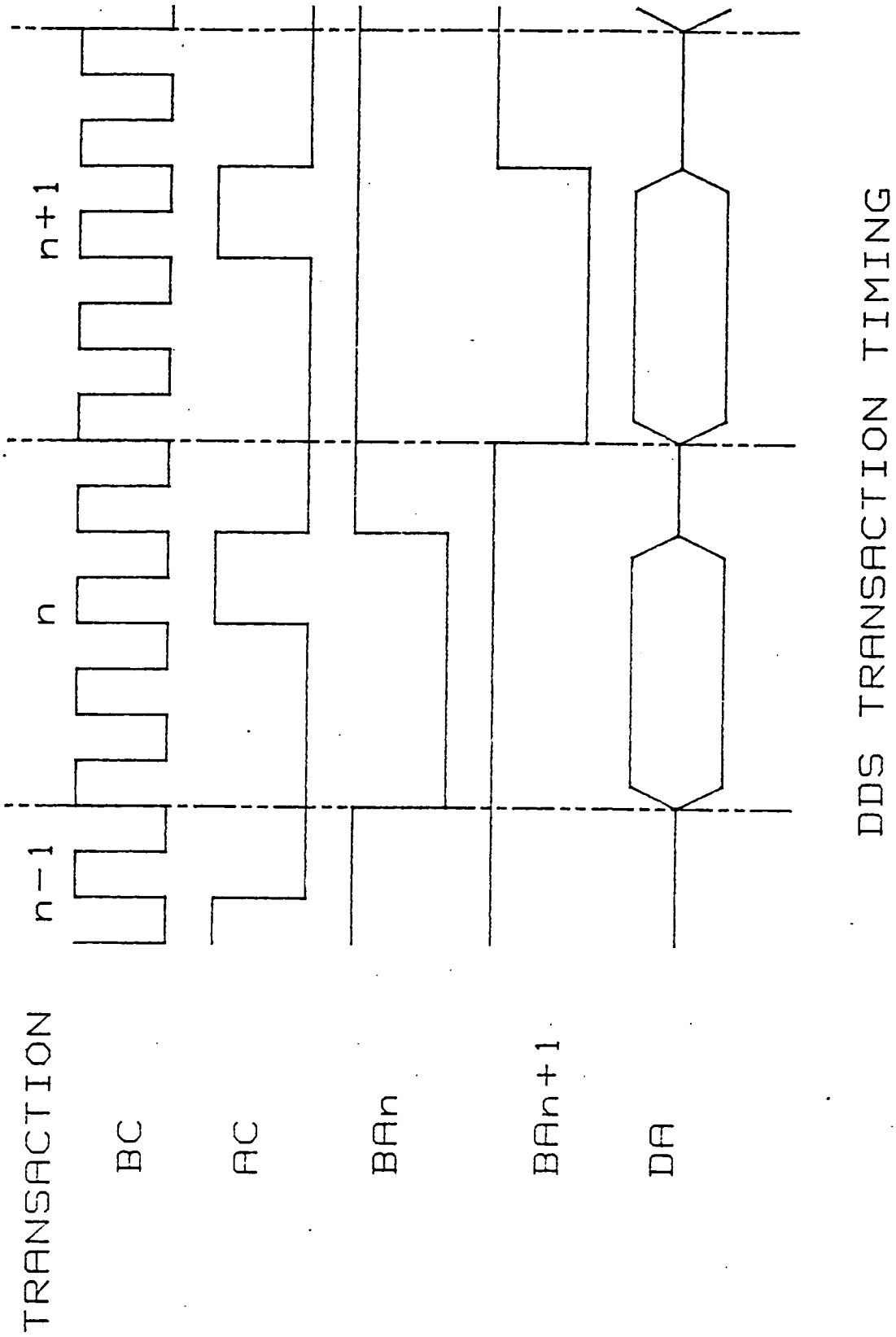

Figure B.5: Timing relationship of the some of the system signals.

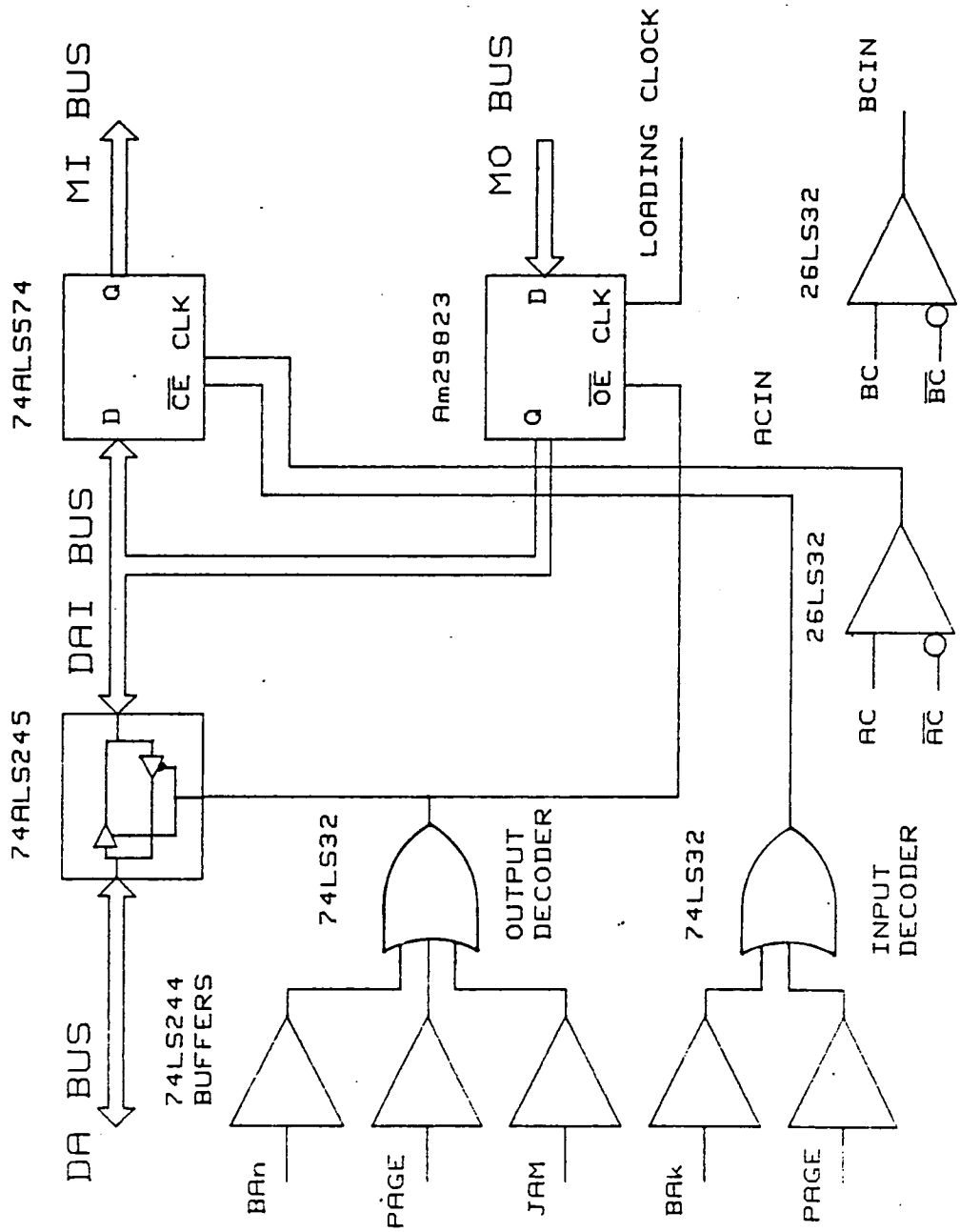

Figure B.6: The on-card DDS interface circuitry.

bidirectional buffers were configured in the output mode. This implied that data had to be loaded onto the bus driver's inputs in the period between the last time data was read from the card and a point sufficiently before the driver was output enabled to comply with the required set up times for the device.

This was most easily achieved by enabling the 29823 clock during the previous transaction to that used for card output, so that data could be clocked onto the bus driver latch at that point.

Operation of the input latch was much simpler in that it was only necessary to enable the data during the appropriate transaction. Due to the tristate output capabilities of all latches used in the DDS interface it was necessary to take into consideration the potentially transitory nature of all data. i.e. data was only valid when latches were output enabled, and this was generally only during one transaction timeslot every bus cycle.

In the case of the TTM card, transaction 1 had been designated the data input timeslot and transaction 8 the output. Although these choices were fairly arbitrary, they did allow ample time for setting up data and make data timing restraints less stringent. This resulted in JAM, PAGE and the BA8 pulse being OR'ed together to provide the directional signal to the 74-245 buffers and to output enable the 29823 bus driver latches. The BA1 pulse was OR'ed with the PAGE signal and the result was then shifted by two BC pulses before being connected to the $\overline{OC}$ input of the 74-574 input data latches.

The reason for the bus input latch enable being shifted through two D-type flip flops was that it allowed simpler implementation of the R/W switch and the general timing of the look-up table operations.

5. Look-up Table Hardware and Control

Due to the way that the look-up table had been constructed, the data lines for input and output to the rest TTM card circuitry, were connected to both the input side of

a set of latches and the output side of a different set of latches. This meant that care was needed to ensure that both sets of latches, and the memories, were not being driven simultaneously.

The tight timing requirements of the Hitachi HM6168-70 4096 × 4 bit Static RAM's meant that a foolproof circuit scheme was needed to ensure that the control of the two independent signals, $\overline{CS}$ and $\overline{WE}$ was correct. To this end a manual two pole switch was fitted to the board so that the contents of the memories could not be overwritten accidentally and that the very different signal timings in the read and write modes could be kept under strict control. This switch was labelled the R/W switch, and figure B.7 shows a schematic of the logic in which its use facilitated control of the memories in the way described. A number of delay D-types were needed in the two circuits, this was make sure that the data was input enabled at the correct point in the read/write cycle of the memories and that there was no overlap or incorrect accessing of the memories.

6. ADC Port and Control

In order for the correct operation of the spectrum analyser, it was necessary to have the ADC's sampling the input signal over a long period of time (several minutes). To this end the ADC's had to be controlled by hardware, as the software system could not provide continuous operation.

The ADC port consisted of two pattern 17 connecters for the two ADC clocks and a 26 way 3-M male connecter. Of the 26 pins, 24 were used for the two sets of 12 bit parallel data from the ADC's, with the other two pins being used for the $\overline{HOLD}/\overline{S}$ and SAMP signals. Figure B.8 shows exactly how the connecter was configured, with the least significant data bits being kept away from the control signals in an attempt to reduce any crosstalk problems.

Figure B.9 shows the detailed timing and sequence of the two control signals. The rates and shapes of these signals were chosen so that only one conversion was made

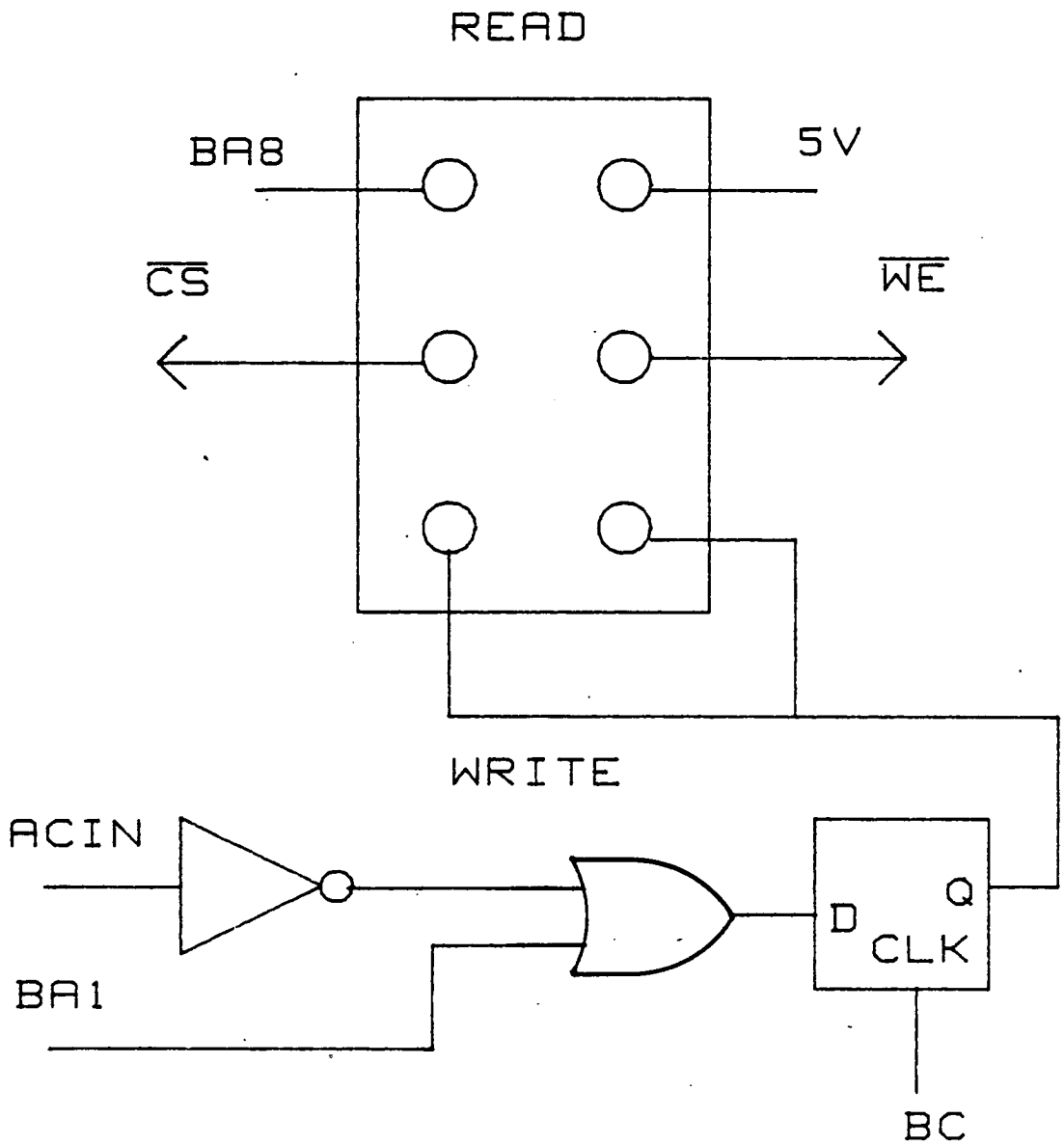

Figure B.7: The R/W switch control logic.

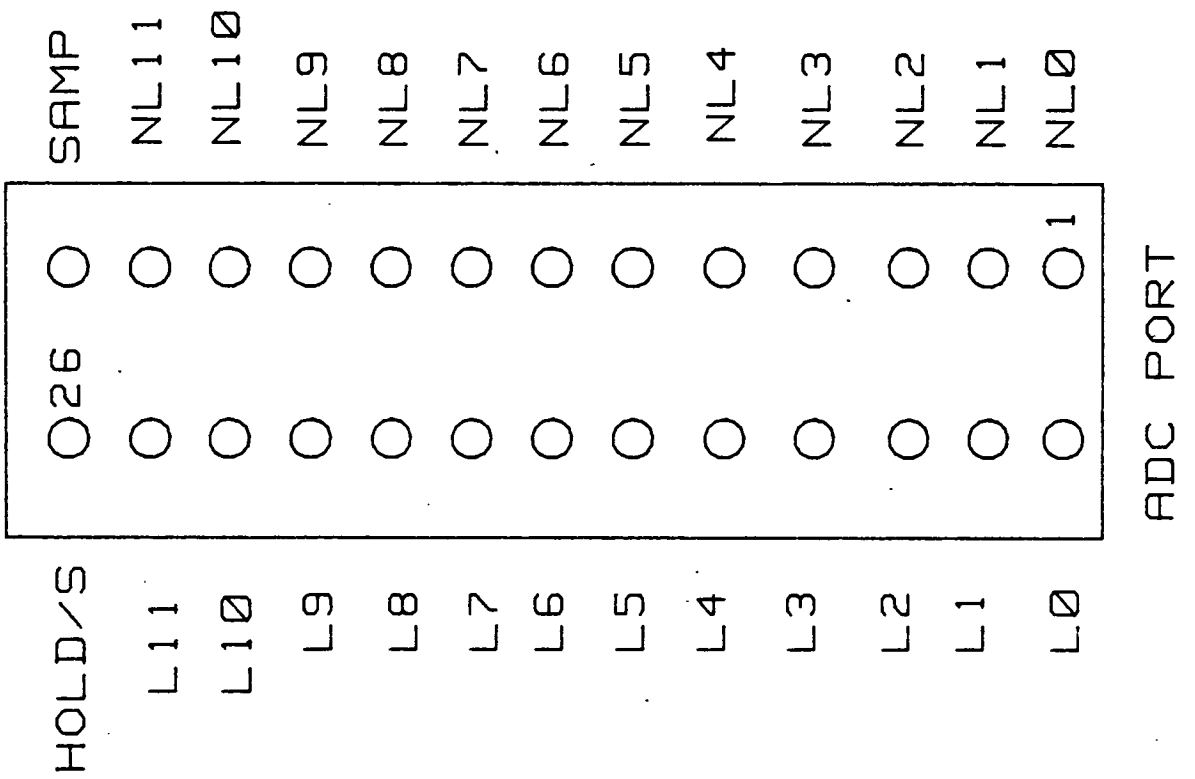

Figure B.8: Connection diagram for the interface to the two ADC's.

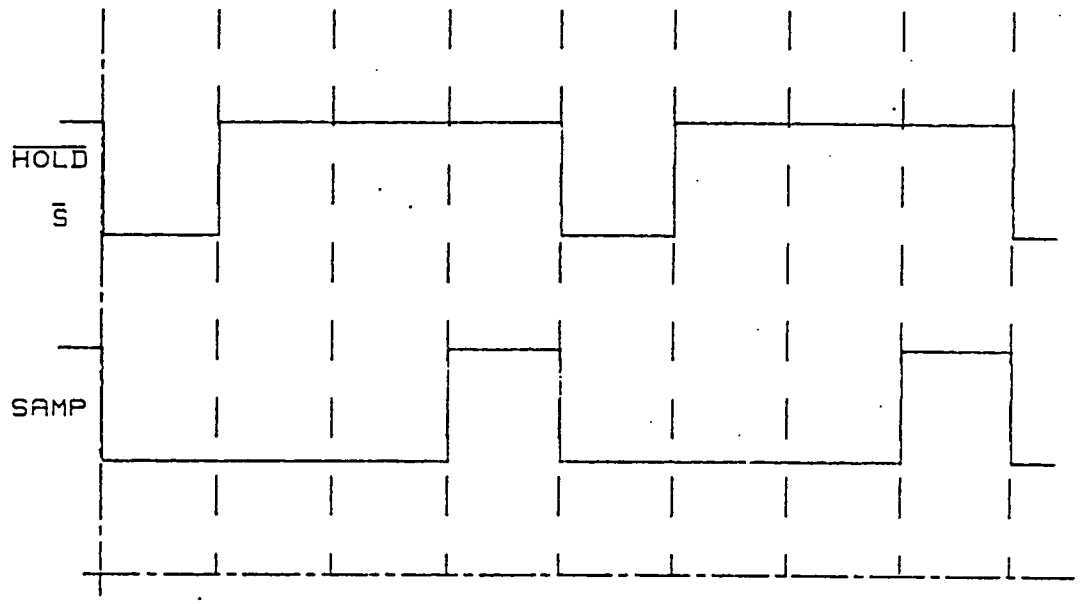

Figure B.9: Timing control for the ADC's.

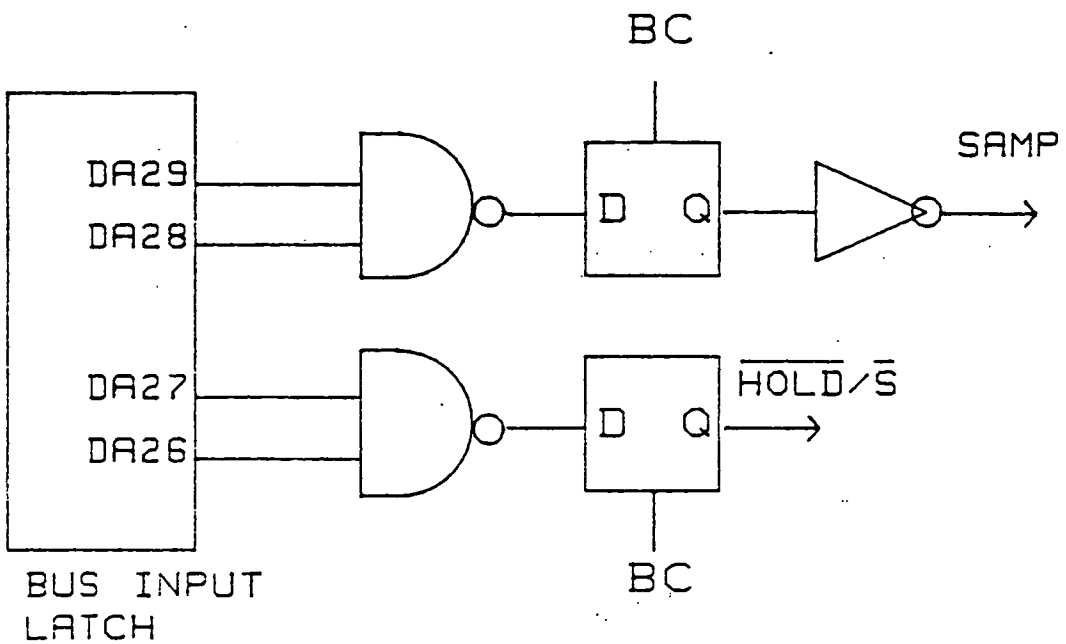

Figure B.10: Logic to generate the ADC control signals.

by the linear ADC in the sample time allotted, that there was ample time for the nonlinear ADC to complete a conversion whilst the sampled analogue signal was held and that sufficient time was allowed between conversions for the S/H mechanisms to track the analogue signal accurately after the hold cycle.

Figure B.10 shows the logic schematic used to generate $\overline{HOLD}/\overline{S}$ and SAMP. It will be noted that the same signals are derived from two different circuits. This is because the design and operation of the NCC card data sequencer/logger hardware meant that only 8K data samples could be sequenced/logged at any one time, before the whole block had to be uploaded to the 200 series computer. The 8K samples took only 0.2 seconds to log and this was not enough time for the spectrum analyser to produce an accurate representation of the spectral information contained in the sampled signal.

To get round this problem it was necessary for the ADC's to be able to sample signals continuously, independently of software controlled signals. Consequently a counter was used to generate the two control signals from the hardware originated PAGE signal. To avoid the problem of having to ensure that these signals were synchronised with the control software's sampling requirements, a switch was provided (H/S) which allowed the origin of the signals to be changed with only a transitory disturbance in ADC operation.

The two clock signals of approximately 1.25 MHz and 200 kHz were generated by dividing the frequency of the BC and AC clock pulses using counters with division factors of 8 and 12 respectively. It was not necessary for the phase of the clocks to be in step with the ADC control signals.

The 24 bits of input data from the ADC port were processed differently depending upon which ADC they originated from. The linear ADC was connected directly to the input side of the 29823 bus driver latches, whilst the nonlinear data was latched by the ADC data input 74-574's. This allowed control of when the Static RAM

address lines were tri-stated and when they were driven by the nonlinear ADC output, and hence avoided any bus contentions. The output enable line to the ADC data latches was provided by the BA7 signal shifted in time by two BC clock pulses and OR'ed with BA8, thus giving an elongated enable pulse, which allowed enough time for the memories to be accessed by the ADC address data before the RAM output was latched by the bus drivers.

7. DAC Port and Control

In order to interface the TTM card to the external ADC, it was necessary to generate an appropriate timing signal, and some method of switching between the two converters. These functions were provided by the DAC port circuitry.

The DAC port consisted of a 3-M 26 way male connecter and one pattern 17 connecter. The DAC clock signal was fed to the external pod via the pattern 17 lead. The signal for this implemented by OR gating the output enable signal to the 29823's with an inverted version of the ADC SAMP signal, so that at the moment the ADC S/H circuits were re-tracking the analogue signal, their output data was stable and complete and the 29823's were driving the DDS, the DAC was triggered.

Of the 26 pins on the male connecter, one whole side was earthed and one pin was unconnected. This was the configuration specified for use with the DAC pod and can be seen in more detail in figure B.11. The data to be fed to the DAC came from the outputs of three quad 2 to 1 74-157 multiplexers. The inputs to the multiplexers in turn came from the bus driver latch outputs, with the two sets of data being the linear and mapped nonlinear ADC data respectively. Linear or Nonlinear ADC data was selected by the position of the L/N switch which connected the multiplexer select signal to logic high and low, as required.

As has been explained previously, there was no easy way of controlling the DAC port by software and consequently its output only made sense when the system software was in a wait state and the ADC's, and the rest of the TTM card, were

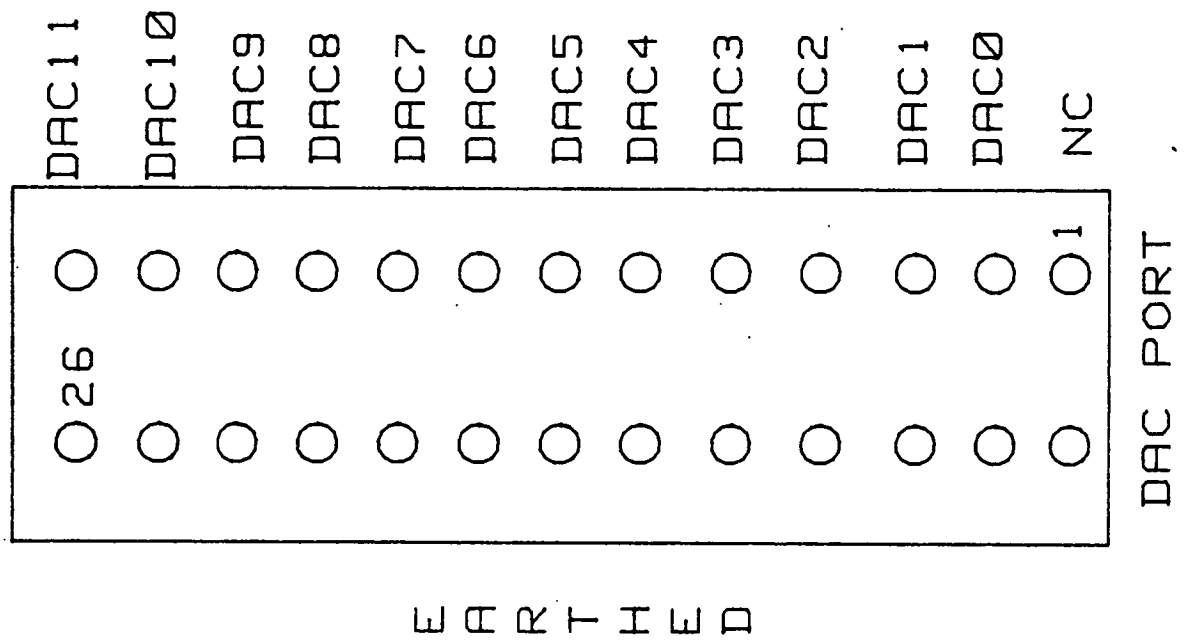

Figure B.11: The DAC port connector wiring diagram.

being controlled by the hardware.

8. Program Overview

This section gives a brief overview of the control program for the ADC linearisation hardware, and how to use it. A block structured flow diagram of program control is shown in figure B.12 and is described here.

The first thing the program had to do was to initialise all arrays and variables. This included setting the bins of the pdf histogram arrays to zero and required setting up the array which stored the ADC mapping with an X=X mapping which would enable the actual nonlinear ADC output data to be transferred directly onto the DDS.

After this mapping had been set up, it had to downloaded to the NCC card from where it could be written to the TTM card's look-up table, after the program user had first ensured that the R/W switch had been set to the write position. Once this process was complete and the sample size of the pdf to be taken had been decided, the sequencer array had to be set up with the correct bit patterns to generate the ADC sampling control signals. This array would then be sequenced to the TTM card whilst the output of the ADC's was logged by the NCC card. It was necessary to have the H/S switch set to the Software position during this stage of the program and the analogue input signal would have to be correctly scaled and of the correct type to generate the required linearly sampled pdf e.g. a ramp signal varying with time between 0 and 5V (the input range of the ADC's).

Next the pdf array would be updated by the 2K samples grabbed by the 8K of sequence data ADC control signals. This data acquisition loop would be repeated until the complete pdf sample size had been logged in blocks of 2K. The two accumulated pdf's could then be displayed on the monitor, with their y-axes scaled to the maximum bin value in the respective histograms.

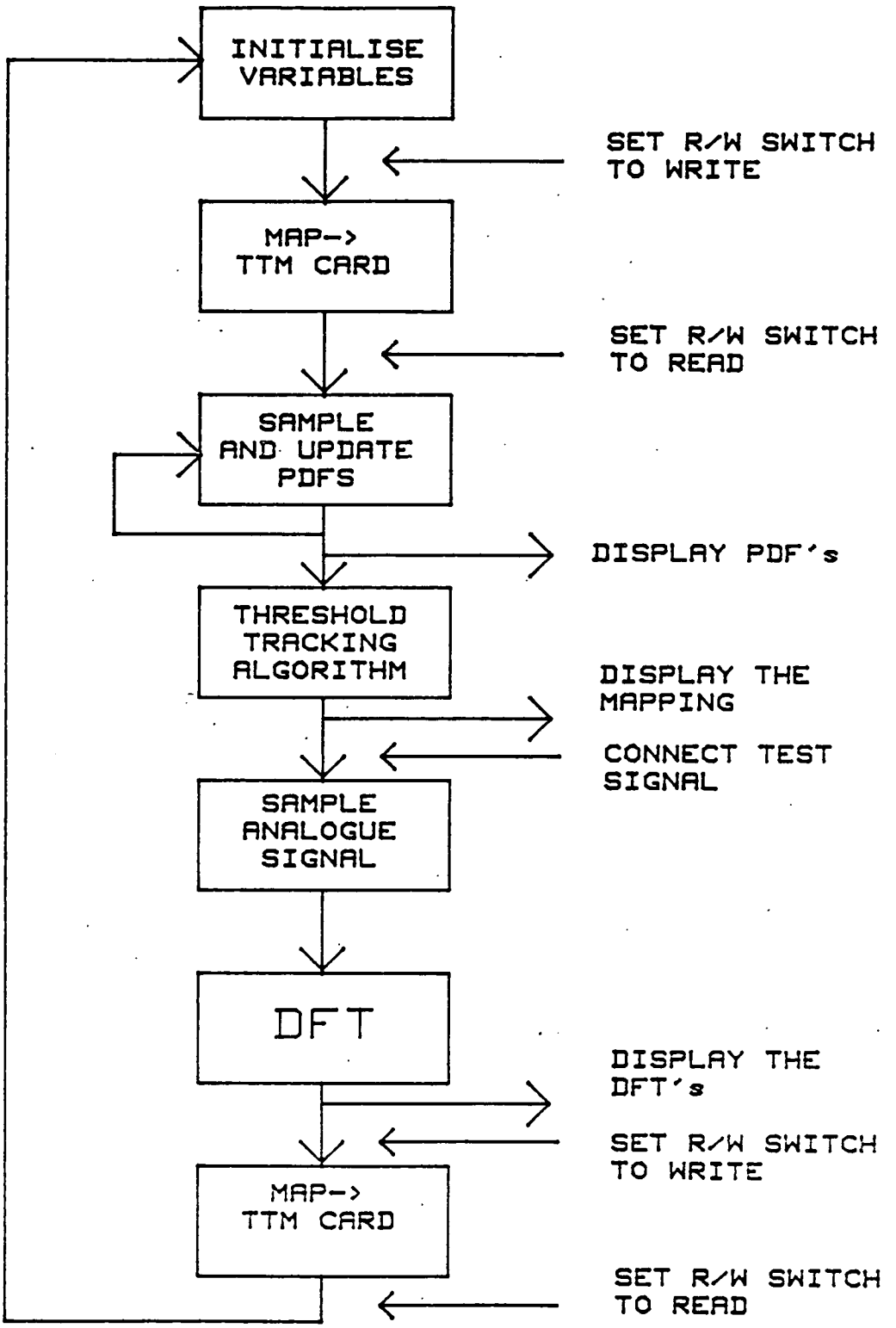

Figure B.12: Flow of control and event sequence for the system.

The next step in the program was the calculation of the nonlinear transfer function mapping from the information contained in the two pdf's. After the threshold tracking algorithm had generated the mapping it was displayed on the monitor. Although it was not possible, strictly speaking, to display the transfer function mapping without explicit knowledge of the the ADC nonlinearity, an approximate idea of the dimensions and shape of the inverse of the nonlinearity could be gained by plotting the values of the mapping as a set of coordinates. It was not therefore possible to show changes in the transfer function relating to the use of the Mapping Algorithm, as neither the original nonlinearity nor the real, mapped transfer function could be shown.

The analogue signal would then be changed to the test signal, usually a sinusoid of a magnitude of approximately two thirds of the ADC input range. This signal had to be of a frequency which would give a certain number of full cycles in the DFT time window, in order to avoid any data windowing problems. In order to exercise the largest number of ADC output codes, the number of full cycles should be relatively prime to the order of the DFT. This meant that due to the sampling rate of the software controlled ADC's, a suitable test signal frequency was calculated to be 371.09 Hz. This corresponded to nineteen full cycles of the waveform in the DFT time window and made the first harmonic of the sampled signal appear at a point suitably near the y-axis of the DFT plot, hence leaving lots of space for a large number of higher order harmonics to be displayed.

The sequencer array was then triggered once more so that samples of the sinusoidal test signal could be logged. This logged data was converted back to the 12 bit ADC samples and then to the ideally DAC reconstructed floating point format required before the DFT software could be used. After the DFT's had been performed the results could be displayed on the monitor.

The mapping produced from the two sampled pdf's could then be loaded into the TTM card look-up table by the same method as the original X=X mapping was.

As before, the R/W switch would have to be set to write so that the original mapping could be overwritten.

Program control then returns to the initialisation stage to ensure that the DFT software will perform a second time as this was found to be problematical area. The user then cycles through all of the above procedures, omitting any unnecessary pieces of code when given the option to do so. This time the sampled signal has been mapped before its DFT has been taken and hence any changes in the signal's spectral content after the mapping process could be clearly seen by comparison of the monitor outputs.

Appendix C

ADCL_FINAL SOFTWARE

```
10    ! ********** M A I N  P R O G R A M **********
20    !
30    !          A D C L _ F I N A L
40    !
50    !         11th NOVEMBER 1988.
60    !
70    ! THIS PROGRAM RUNS THE ADC LINEARISATION ROUTINES FOR USE
80    ! WITH THE TTA MAPPING CARD AND TWO TWELVE BIT ADC's, WHICH
90    ! ARE CONTROLLED BY THE TWO LINES HOLD(bar) AND SAMP, WHICH
100   ! IN TURN ARE CONTROLLED BY DATA LINES DA26,27 AND DA28,29
110   ! RESPECTIVELY. FIRSTLY A X=X MAP IS STORED IN THE MEMORIES
120   ! SO THAT THE INPUT FROM THE NONLINEAR ADC CAN BE DIRECTLY
130   ! ACCESSED BY THE REST OF THE PROGRAM. THE TWO TWELVE BIT
140   ! INPUT SIGNALS ARE THEN COLLATED INTO A PAIR OF PDFS WHICH
150   ! THE THRESHOLD TRACKING ALGORITHM THEN TURNS INTO A MAPPING.
160   ! (IT MAY BE NECESSARY TO GO ROUND THE ABOVE TWO STEPS IN A
170   ! LOOP TO GENERATE A PDF OF A LARGE ENOUGH SIZE FOR THE MAPPING
180   ! TO BE EFFECTIVE.
190   ! THIS MAPPING IS THEN STORED IN THE MEMORIES AND A TEST SIGNAL
200   ! SHOULD THEN BE CONNECTED TO THE ADC's INSTEAD OF THE TRAINING
210   ! SIGNAL. THIS SIGNAL IS THEN SAMPLED AS BEFORE BUT THIS TIME THE INPUT
220   ! HAS BEEN FED THROUGH THE MAPPING. THE TWO RECEIVED SIGNALS ARE THEN
230   ! CONVERTED TO A FORM WHICH CAN BE READILY USED IN AN FFT ALGORITHM,
240   ! BEFORE DISPLAY. IT IS NECESSARY TO LOAD THE SUBPROGRAMS INCLUDED
250   ! AT THE END OF THIS LISTING SEPARATELY FROM THE APPROPRIATE DISK.
260   ! ( PROTEK FFT UTILITY SN: 2/233001 1984 )
270   !
280   ! THIS IS THE VERSION OF THE PROGRAM WHICH USES AN ACTUAL LINEAR PDF
290   ! AS SAMPLED BY THE LINEAR ADC. BUT ALSO TRIES TO DODGE
300   ! REPETITOUS PARTS OF THE PROGRAM.
310   ! ********************************************
320   ! **                                        **
330   ! **            ONE OF THE                  **
340   ! **        A D C _ L I N E A R             **
350   ! **             PROGRAMS                   **
360   ! **        CHRIS. DENT                     **
370   ! **        C/O JOHN COOK                   **
380   ! **        OCTOBER/NOVEMBER 1988           **
390   ! **                                        **
400   ! ********************************************
410   !
420   OPTION BASE 0
430   DEG
440   INTEGER Mapping(4095)
450   REAL Rec_pdf(2047)
```

```
460   REAL Rec_lin(2047)
470   INTEGER Pdfs(4095,1)
480   INTEGER Rlog(8191,1)
490   INTEGER Dat(8191,1),Samp_arr(8191,1)
500   INTEGER Timeslots(4)
510   INTEGER Big(1,1),No_map,First_time
520   REAL Calib,Samples
530   INTEGER Buff(-1:511)
540   !
550   PRINT CHR$(12)
560   GCLEAR
570   DISP "ADC LINEARISATION BY TRANSFER FUNCTION MAPPING"
580   FOR A=0 TO 4095
590   Mapping(A)=A ! NO MAPPING, DIRECT I/P FROM ADC
600   Pdfs(A,0)=0
610   Pdfs(A,1)=0
620   NEXT A
630   Big(0,0)=0
640   Big(0,1)=0
650   Big(1,0)=0
660   Big(1,1)=0
670   INPUT "DO YOU WANT TO WRITE A MAPPING? (N)",C$
680   IF C$="N" THEN
690   GOTO 790
700   END IF
710   CALL Write_map(Timeslots(*),Mapping(*),Dat(*))
720   CALL Tslots_mode_8(Timeslots(*))
730   CALL Seq_load_8(Dat(*))
740   DISP "PLEASE SET SWITCH TO 'WRITE' IF YOU WANT TO SET 'NO-MAP'"
750   PAUSE
760   CALL Trig
770   DISP "NOW SET SWITCH TO 'READ'"
780   PAUSE
790   INPUT "HOW MANY SAMPLES IN THE PDF's?",Samples
800   Samples=(Samples DIV 2048)*2048
810   DISP "PDF OF",Samples,"SAMPLES WILL TAKE",((Samples/2048)+1)*.3,"MINUT
820   WAIT 2
830   A$=" "
840   INPUT "OK ? (N)",A$
850   IF A$="N" THEN
860   A$=" "
870   GOTO 790
880   END IF
890   CALL Sample(Timeslots(*),Dat(*),Samp_arr(*))
900   BEEP
910   FOR B=1.0 TO (Samples DIV 2048) STEP 1.0
920   CALL Tslots_mode_8(Timeslots(*))
930   CALL Seq_load_8(Dat(*))
940   CALL Trig
950   CALL Log_dump_8(Rlog(*))
960   CALL Form_pdfs(Rlog(*),Pdfs(*),Big(*))
970   NEXT B
980   BEEP
990   INPUT "DO YOU WANT A PLOT OF THE PDFS? (N)",D$
1000  IF D$="N" THEN
1010  GOTO 1280
1020  END IF
```

```
1030  ALPHA OFF
1040  GRAPHICS ON
1050  VIEWPORT 0,150,30,100
1060  WINDOW 0,4096,0,Big(0,0)
1070  MOVE 1000,.8*Big(0,0)
1080  LABEL "NONLINEAR PDF Vs BIN NUMBER"
1090  MOVE 0,0
1100  FOR A=0 TO 4095
1110  DRAW A,Pdfs(A,0)
1120  NEXT A
1130  ALPHA ON
1140  DISP "PDF PEAK IS ",Big(0,0),"AT BIN ",Big(1,0)
1150  PAUSE
1160  ALPHA OFF
1170  GCLEAR
1180  VIEWPORT 0,150,30,100
1190  WINDOW 0,4096,0,Big(0,1)
1200  MOVE 1000,.8*Big(0,1)
1210  LABEL "LINEAR PDF Vs BIN NUMBER"
1220  MOVE 0,0
1230  FOR A=0 TO 4095
1240  DRAW A,Pdfs(A,1)
1250  NEXT A
1260  ALPHA ON
1270  DISP "PDF PEAK(LINEAR) IS ",Big(0,1),"AT BIN ",Big(1,1)
1280  PAUSE
1290  CALL Tslots_mode_8(Timeslots(*))
1300  CALL Seq_load_8(Dat(*))
1310  DISP "ENSURE A SINUSOIDAL TEST SIGNAL NOW"
1320  PAUSE
1330  CALL Trig
1340  CALL Log_dump_8(Rlog(*))
1350  CALL Pres_data(Rec_lin(*),Rec_non(*),Rlog(*))
1360  CALL A_to_b(0,512,Rec_non(0),Calib,Buff(-1))
1370  CALL Fft(Calib,Buff(-1))
1380  CALL B_to_a(1,Calib,Buff(-1),Rec_non(0))
1390  CALL A_to_b(0,512,Rec_lin(0),Calib,Buff(-1))
1400  CALL Fft(Calib,Buff(-1))
1410  CALL B_to_a(1,Calib,Buff(-1),Rec_lin(0))
1420  FOR A=2 TO 512 STEP 2 ! TO DO WITH FFT O/P DATA FORMAT
1430  Rec_non(A DIV 2)=Rec_non(A)^2+Rec_non(A+1)^2
1440  IF Rec_non(A DIV 2)<>0 THEN
1450  Rec_non(A DIV 2)=10*LGT(Rec_non(A DIV 2))
1460  ELSE
1470  Rec_non(A DIV 2)=-100
1480  END IF
1490  Rec_lin(A DIV 2)=Rec_lin(A)^2+Rec_lin(A+1)^2
1500  IF Rec_lin(A DIV 2)<>0 THEN
1510  Rec_lin(A DIV 2)=10*LGT(Rec_lin(A DIV 2))
1520  ELSE
1530  Rec_lin(A DIV 2)=-100
1540  END IF
1550  NEXT A
1560  ALPHA OFF
1570  GRAPHICS ON
1580  GCLEAR
1590  VIEWPORT 0,150,30,100
1600  WINDOW 0,255,-100,0
1610  MOVE 50,-20
```

```
1620  LABEL "FFT OF NONLINEAR SIGNAL"
1630  MOVE 0,Rec_non(0)
1640  FOR A=1 TO 255
1650  DRAW A,Rec_non(A)
1660  NEXT A
1670  PAUSE
1680  GCLEAR
1690  MOVE 50,-20
1700  LABEL "FFT OF LINEAR SIGNAL"
1710  MOVE 0,Rec_lin(0)
1720  FOR A=1 TO 255
1730  DRAW A,Rec_lin(A)
1740  NEXT A
1750  PAUSE
1760  GCLEAR
1770  CALL Tta_map(Pdfs(*),Mapping(*),No_map,Samples)
1780  IF No_map=1 THEN
1790  DISP "NO MAPPING IS NECESSARY"
1800  PAUSE
1810  GOTO 2360
1820  END IF
1830  INPUT "DO YOU WANT A PICTURE? (Y)",A$
1840  IF A$="Y" THEN
1850  INPUT "FULL OR PARTIAL? (F/P)",A$
1860  IF A$="P" THEN
1870  GOTO 1940
1880  ELSE
1890  GOTO 2090
1900  END IF
1910  ELSE
1920  GOTO 2160
1930  END IF
1940  GCLEAR
1950  VIEWPORT 0,150,30,100
1960  WINDOW 0,512,0,512
1970  MOVE 0,0
1980  DRAW 512,512
1990  MOVE 100,400
2000  LABEL "IDEAL AND MAPPED TRANSFER FUNCTIONS"
2010  MOVE 0,0
2020  FOR B=0 TO 7
2030  FOR A=0 TO 511
2040  DRAW A,Mapping(B*512+A)-B*512
2050  NEXT A
2060  NEXT B
2070  PAUSE
2080  GOTO 2160
2090  GCLEAR
2100  VIEWPORT 0,150,30,100
2110  WINDOW 0,4095,0,4095
2120  MOVE 0,0
2130  DRAW 4095,4095
2140  MOVE 0,0
2150  FOR A=1 TO 4095
2160  DRAW A,Mapping(A)
2170  NEXT A
2180  PAUSE
```

```
2190  !  First_time=First_time+1
2200  !  IF First_time=2 THEN
2210  !  GOTO 620
2220  !  END IF
2230  !  END
2240    CALL Write_map(Timeslots(*),Mapping(*),Dat(*))
2250    CALL Tslots_mode_8(Timeslots(*))
2260    CALL Seq_load_8(Dat(*))
2270    DISP "SWITCH TO' WRITE' TO ENTER MAPPED TRANSFER FUNCTION"
2280    PAUSE
2290    CALL Trig
2300    DISP "SET SWITCH TO 'READ'"
2310    PAUSE
2320    GOTO 10
2330    END
2340  !
2350  SUB Seq_load_8(INTEGER Dat(*))
2360  !
2370  ! THIS SUB PROGRAM LOADS THE 8K SEQUENCER RAM WITH THE CONTENTS
2380  ! OF THE ARRAY Dat(*).
2390  !
2400  DISP "LOADING SEQUENCER PLEASE WAIT"
2410  CONTROL 12,2;2
2420  CONTROL 12,0;1
2430  INTEGER Seqbuff(8191,1) BUFFER
2440  ASSIGN @Path TO 12;WORD,FORMAT OFF
2450  ASSIGN @Path1 TO BUFFER Seqbuff(*);FORMAT OFF
2460  OUTPUT @Path1;Dat(*)
2470  TRANSFER @Path1 TO @Path;WAIT
2480  DISP " "
2490  SUBEND
2500  !
2510  SUB Trig
2520  !
2530  ! THIS SUB PROGRAM PUTS THE NCC INTO BUS I/O MODE
2540  ! ITS EXACT MODE OF OPERATION DEPENDS ON THE OPTIONS
2550  ! / TIMESLOTS SELECTED IN SUB T_slots_mode.
2560  ! NB AN ACTIVE Sti0 INDICATES THE LOGGER RAM IS FULL.
2570  !    AN ACTIVE Sti1 INDICATES THE SEQUENCER RAM IS FULL.
2580  !
2590  CONTROL 12,0;1
2600  CONTROL 12,2;0
2610  DISP "GRABBING DATA FROM THE BUS PLEASE WAIT"
2620  STATUS 12,5;Value       ! CHECK STATUS OF Sti0,IF HIGH
2630  Sti0=BIT(Value,0)       ! (LOW IN PROGRAM) THEN LOGGER
2640  IF Sti0=0 THEN          ! IS NOT FULL AND WE MUST WAIT.
2650  GOTO 2620               ! TO CHECK IF SEQUENCER IS FINISHED
2660  END IF                  ! USE Sti1 IE BIT(Value,1)
2670  DISP " "
2680  SUBEND
2690
2700  SUB Tslots_mode_8(INTEGER Timeslots(*))
2710    THIS SUB PROGRAM LOADS THE TIMESLOT AND MODE CONTROL
2720    REGISTERS. A '1' IN A BIT LOCATION INDICATES AN ENABLED
2730    TIMESLOT.
2740
```

```
2750  DISP "ERROR: CLOCK OR POWER SUPPLY NON-EXISTANT"
2760  CONTROL 12,2;3
2770  CONTROL 12,0;1
2780  INTEGER Srbuff(4) BUFFER
2790  ASSIGN @Route TO 12;WORD,FORMAT OFF
2800  ASSIGN @Buff0 TO BUFFER Srbuff(*);FORMAT OFF
2810  OUTPUT @Buff0;Timeslots(*)
2820  TRANSFER @Buff0 TO @Route;WAIT
2830  DISP " "
2840  SUBEND
2850  !
2860  SUB Log_dump_8(INTEGER Rlog(*))
2870  !
2880  ! THIS SUB PROGRAM LOADS THE CONTENTS OF THE LOGGER
2890  ! RAM BACK INTO THE COMPUTER. THE DATA ENDS UP IN AN
2900  ! ARRAY Rlog(*) IN THE SAME FORMAT AS THAT OF Dat(*)
2910  ! IN SUB SEQ_LOAD_8. THIS PROGRAM IS FOR USE IN THE SINGLE
2920  ! SHOT LOGGING MODE.
2930  ! NOTE THAT IN SINGLE SHOT MODE THE LAST 32 BIT VECTOR IN
2940  ! THE ARRAY Rlog(*) WILL BE INVALID WHILST, IN WRAPAROUND
2950  ! MODE THE LAST VECTOR WILL BE VALID.
2960  !
2970  DISP "UPLOADING DATA FROM LOGGER PLEASE WAIT"
2980  CONTROL 12,2;1
2990  CONTROL 12,0;1                          ! *** SEE NOTE ABOVE ***
3000  INTEGER Resultsbuff(8191,1) BUFFER
3010  ASSIGN @Route TO 12;WORD,FORMAT OFF
3020  ASSIGN @Buff1 TO BUFFER Resultsbuff(*);FORMAT OFF
3030  TRANSFER @Route TO @Buff1;WAIT
3040  ENTER @Buff1;Rlog(*)
3050  DISP " "
3060  SUBEND
3070  !
3080  SUB Tta_map(INTEGER Pdfs(*),Mapping(*),No_map,REAL Samples)
3090  ! THIS SUB PROGRAM DOES A FULL RESOLUTION THRESHOLD
3100  ! TRACKING ALGORITHM MAPPING OF A NONLINEAR ADC, THE
3110  ! SAMPLED PDF OF WHICH IS STORED IN PDFS(0,*). THE LINEAR
3120  ! PDF IS STORED IN PDFS(1,*).
3130  !
3140  INTEGER B,Done_loop
3150  REAL Non_sum,Lin_sum,Diff,Limit
3160  DISP "THRESHOLD TRACKING ALGORITHM ACTIVATED"
3170  B=0
3180  Non_sum=0
3190  Lin_sum=0
3200  No_map=1
3210  Limit=Samples*.5/4095
3220  FOR A=0 TO 4095
3230  IF B<4096 THEN
3240  Lin_sum=Lin_sum+Pdfs(B,1)
3250  Non_sum=Non_sum+Pdfs(A,0)
3260  Done_loop=0
3270  Diff=Lin_sum-Non_sum
3280  IF Diff<Limit THEN
3290  Lin_sum=Lin_sum-Pdfs(B,1)
3300  B=B+1
3310  Done_loop=1
3320  END IF
```

```
3330  WHILE (Diff<0-Limit) AND (Done_loop=False)
3340  B=B+1
3350  Lin_sum=Lin_sum+Pdfs(B,1)
3360  Diff=Lin_sum-Non_sum
3370  END WHILE
3380  IF B<0 THEN
3390  Lin_sum=0
3400  B=0
3410  END IF
3420  Mapping(A)=B
3430  IF A<>B THEN
3440  No_map=0
3450  END IF
3460  B=B+1
3470  ELSE
3480  Mapping(A)=4095
3490  END IF
3500  NEXT A
3510  SUBEND
3520  !
3530  SUB Sample(INTEGER Timeslots(*),Dat(*),Samp_arr(*))
3540  ! THIS SUB SETS UP THE ARRAY TIMESLOTS FOR THE
3550  ! PURPOSES OF SAMPLING THE DIGITAL SIGNAL SUPPLIED TO THE
3560  ! TTM CARD FROM THE TWO ADC's. IT ALSO LOADS THE DAT ARRAY
3570  ! WITH THE RELEVANT INFORMATION TO CONTROL THE HOLD BAR AND
3580  ! SAMP LINES TO THE ADC's.
3590  !
3600  DISP "SETTING UP THE TIMING SIGNALS"
3610  Timeslots(0)=2    ! TRANSACTION 1
3620  Timeslots(1)=0
3630  Timeslots(2)=256  ! TRANSACTION 8
3640  Timeslots(3)=0
3650  Timeslots(4)=16   ! SUBRATE LOGGING/SEQUENCING (2)
3660  FOR A=0 TO 8191
3670  SELECT (A MOD 4)
3680  CASE =0
3690  Dat(A,1)=3072     ! 2^(26-16)+2^(27-16)
3700  CASE =3
3710  Dat(A,1)=12288    ! 2^(28-16)+2^(29-16)
3720  CASE =1,=2
3730  Dat(A,1)=0
3740  END SELECT
3750  Dat(A,0)=0
3760  Samp_arr(A,0)=0
3770  Samp_arr(A,1)=Dat(A,1)
3780  NEXT A
3790  SUBEND
3800  !
3810  SUB Form_pdfs(INTEGER Rlog(*),Pdfs(*),Big(*))
3820  ! THIS SUB WILL PICK OUT THE RELEVANT DATA FROM THE
3830  ! ARRAY RLOG WHICH IS FILLED BY LOG_DUMP_8
3840  ! IN THIS SET-UP ONLY EVERY FOURTH DATA OBJECT
3850  ! IS VALID.
3860  !
3870  INTEGER Temp
3880  DISP "FORMING THE PDF's FROM THE LOGGED DATA"
3890  FOR A=0 TO 8190
3900  IF ((A MOD 4)=3) THEN
3910  Temp=SHIFT(SHIFT(Rlog(A,3),-4),4)
```

```
3920  Pdfs(Temp,0)=Pdfs(Temp,0)+1
3930  IF Pdfs(Temp,0)>Big(0,0) THEN
3940  Big(0,0)=Pdfs(Temp,0)
3950  Big(1,0)=Temp
3960  END IF
3970  Temp=SHIFT(SHIFT(Rlog(A,1),-8),4)+SHIFT(Rlog(A,0),12)
3980  Pdfs(Temp,1)=Pdfs(Temp,1)+1
3990  IF Pdfs(Temp,1)>Big(0,1) THEN
4000  Big(0,1)=Pdfs(Temp,1)
4010  Big(1,1)=Temp
4020  END IF
4030  END IF
4040  NEXT A
4050  SUBEND
4060  !
4070  SUB Write_map(INTEGER Timeslots(*),Mapping(*),Dat(*))
4080  ! THIS SUB SETS UP THE SEQUENCER ARRAY DAT SO THAT THE
4090  ! REQUIRED MAPPING IN STORED IN DA12-DA23 AND HENCE IS FED
4100  ! TO THE MEMORIES ON THE TTM BOARD
4110  !
4120  INTEGER Temp
4130  DISP "SETTING UP SEQUENCER ARRAY WITH A MAPPING."
4140  Timeslots(0)=2
4150  Timeslots(1)=0
4160  Timeslots(2)=256
4170  Timeslots(3)=0
4180  Timeslots(4)=0 ! SET UP FOR CORRECT I/O AND FULL RATE
4190  FOR A=0 TO 8191
4200  IF (A<=4095) THEN
4210  Dat(A,0)=A+SHIFT(Mapping(A),-12)
4220  Dat(A,1)=SHIFT(Mapping(A),4)
4230  ELSE
4240  Dat(A,0)=0
4250  Dat(A,1)=0
4260  END IF
4270  NEXT A
4280  SUBEND
4290  !
4300  SUB Prep_data(REAL Rec_lin(*),Rec_non(*),INTEGER Rlog(*))
4310  ! PUTS DIGITAL DATA FORM RLOG INTO THE ARRAY RECORD
4320  ! IN A FORMAT (REAL NUMBERS) WHICH CAN BE USED BY AN FFT PROGRAM
4330  !
4340  INTEGER Temp
4350  DISP "PREPARING LOGGED DATA FOR FFT"
4360  FOR A=0 TO 8191
4370  IF ((A MOD 4)=3) THEN
4380  Temp=SHIFT(SHIFT(Rlog(A,0),-4),4)
4390  Rec_non(A DIV 4)=(Temp+.5)/4096
4400  !  Rec_non(A DIV 4)=Rec_non(A DIV 4)*.5*COS((2*(A DIV 4)-512)*180/512)+.5
4410  Temp=SHIFT(SHIFT(SHIFT(Rlog(A,1),-8),8),-4)+SHIFT(Rlog(A,0),12)
4420  Rec_lin(A DIV 4)=(Temp+.5)/4096
4430  !  Rec_lin(A DIV 4)=Rec_lin(A DIV 4)*.5*COS((2*(A DIV 4)-512)*180/512)+.5
4440  END IF
4450  NEXT A
4460  SUBEND
4470  !
4480  OSUB Fft(Calib,INTEGER Buff)
4490  OSUB A_to_b(INTEGER Ins,Num,REAL Arry,Calib,INTEGER Buff)
4500  OSUB B_to_a(INTEGER Ins,REAL Calib,INTEGER Buff,REAL Arry)
```

Appendix D

DAC LINEARISATION SOFTWARE

```
# include <stdio.h>
# include <math.h>

# define begin {
# define end }
# define boolean int
# define false 0
# define true 1
# define NULL 0

# define levels 65536   /* quantisation levels of main DAC */
# define PI 3.1415926
# define points 1024

int mapping[levels],    /* the mappings produced */
    non_lin[levels],    /* the psuedo NL pdf */
    offset[levels],     /* stores mapping point offsets */
    ref_map[levels],    /* the reference mapping for current sample size */
    temp,
    new,
    last,
    map_points,
    i;

double threshold=0.5,   /* the fraction of linear pdf used for d.t. */
    step=2.0/levels,    /* ideal quantisation step */
    base_error=0.0,             /* the unmapped error */
    inc,
    temp2,
    temp3,
    xx,yy,
    distortion();

char option,*errfile=NULL,*mapfile=NULL,*sigfl=NULL,*mapedfl=NULL;

FILE *err_fp,*fopen(),*map_fp,*mpd_fp,*sig_fp;

boolean not_mapped,start_of_loop;

/********** SUBROUTINES **********/

rounded(a) double a;
{
int result;
if(a<0.0) return((int)(a-0.5));
else return((int)(a+0.5));
}

find_error()            /* adds up the total error in a mapping */
begin
double ni,error=0.0,increment;  /* increment is a single error square */
err_fp=fopen(errfile,"a");
for(i=0;i<levels;++i)
    begin
    increment=distortion((mapping[i]+0.5)*step-1.0)-((i+0.5)*step-1.0);
    increment*=step;
                                        /* area of the square */
    error+=fabs(increment);
    end;
```

```
if(error!=0.0)
    {
    error/=levels;              /* to give the mean square error */
    error=10*log10(error);
    }
if(base_error!=0.0)
    error-=base_error;
    else  base_error=error;
fprintf(err_fp,"%6d MP %6.3f dB.\n",map_points,error);
increment=0.0;
fclose(err_fp);
end

thresh_track_map()              /* Modified algorithm for DAC t.f. mapping */
begin
int a,b=0;                      /* the array pointer variables */
double diff,                    /* the difference between the sums */
    non_sum=0.0,lin_sum=0.0,
    limit=threshold;            /* decision level for mapping */
boolean done_loop;              /* ensures only one while loop entered */

for(a=0;a<levels;++a)
    begin
    if(b<levels)
        begin
        ++lin_sum;      /* update the linear sum */
        non_sum+=non_lin[a];    /* update the non-linear sum */
        done_loop=false;
        diff=lin_sum-non_sum;   /* calculate the difference */
        if(diff>limit)  /* a map needs to be adjusted */
                begin           /* adjust pointer as requuired */
                --lin_sum;
                                /* subtract the last pdf height and */
                b--;            /* decrement the pointer */
                done_loop=true; /* this ensures only one loop entered */
                end;
        while((diff<(0-limit))&&(done_loop==false))
                begin           /* non_sum - lin_sum > decision threshold */
                ++lin_sum;      /* add the next bin frequency */
                b++;
                diff=lin_sum-non_sum;
                                /* recalculate the difference */
                end;
        if(b<0)                         /* test for negative maps */
                begin
                lin_sum=0.0;    /* catches errors at start of procedure */
                b=0;
                end;
        mapping[a]=b++;         /* mapping is recorded and linear pointer
                                                is incremented */
        end             /* non-linear pointer incremented by for loop */
    else  mapping[a]=levels-1;
    end
end

main(argc,argv) int argc; char *argv[];
begin

while(--argc>0 && (*++argv)[0]=='-')
    begin
    option = *(argv[0]+1);
    switch(option)
        begin
        case 'm':               /* contains Map Point list (last 0) */
            mapfile =  *++argv;
            argc--;
            break;
        case 'e':               /* error storage file */
            errfile =  *++argv;
            argc--;
            break;
        case 's':               /* INL plot data file */
            sigfl =  *++argv;
            argc--;
            break;
        case 'd':               /* INL plot data file */
            mapedfl =  *++argv;
            argc--;
            break;
```

```
            fprintf(stderr,"DACInew: illegal option %c\n",option);
            exit(1);
            break;
        end;
    end;
if((mapfile==NULL)||(errfile==NULL))
    begin
    fprintf(stderr,"DACInew: not enough files specified\n");
    exit(1);
    end;
if(!(sig_fp=fopen(sigfl,"w")))
    begin
    fprintf(stderr,"DACInew: can't open \"%s\"",sigfl);
    exit(1);
    end;
if(!(mpd_fp=fopen(mapedfl,"w")))
    begin
    fprintf(stderr,"DACInew: can't open \"%s\"",mapedfl);
    exit(1);
    end;
if(!(err_fp=fopen(errfile,"w")))
    begin
    fprintf(stderr,"DACInew: can't open \"%s\"",errfile);
    exit(1);
    end;
fclose(err_fp);
if(!(map_fp=fopen(mapfile,"r")))
    begin
    fprintf(stderr,"DACInew: can't open \"%s\"",mapfile);
    exit(1);
    end;
for(i=0;i<levels;i++)  mapping[i]=i;
find_error();                           /* unmapped error */
for(i=0;i<levels;i++)  non_lin[i]=0;
for(i=0;i<levels;i++)
    {
    temp3=(i+0.5)*step-1.0;
    temp3=distortion(temp3);
    temp3=(temp3+1.0)*levels/2.0;
    temp=(int)temp3;
    ++non_lin[temp];            /* getting pdf histogram */
    }
thresh_track_map();
map_points=0;
find_error();                   /* full mapped performance */
for(i=0;i<levels;i++)  ref_map[i]=mapping[i];
fscanf(map_fp,"%d",&map_points);
while(map_points!=0)
    {
    last=0;
    for(i=1;i<=map_points;i++)
        {                               /* array index of Map Points */
        temp=((int)(i*levels/map_points))-1;
        new=ref_map[temp]-temp; /* offset at the Map Point */
        offset[i-1]=rounded((double)((new+last)/2.0));
        last=new;                       /* average between two MP's */
        }
    for(i=0;i<levels;i++)
        {                       /* convert "mapping" for "find_error" */
        mapping[i]=i+offset[(int)(i*map_points/levels)];
        if(mapping[i]>(levels-1)) mapping[i]=levels-1;
            else if(mapping[i]<0) mapping[i]=0;
        }
    find_error();
    fscanf(map_fp,"%d",&map_points);
    }
fclose(map_fp);
xx=0.0;
yy=0.0;
for(i=1;i<(points+1);i++)
    begin
    temp3=sin(xx)+sin(yy);
    temp3*=0.375;
    temp=(int)((temp3+1)*levels/2.0);
    temp2=distortion((temp+0.5)*step-1.0);
    fprintf(sig_fp,"%f\n",temp2);
    temp2=distortion((mapping[temp]+0.5)*step-1.0);
    fprintf(mpd_fp,"%f\n",temp2);
    xx+=180.0*PI/points;
    yy+=120.0*PI/points;
    if(xx>(2.0*PI))  xx-=(2.0*PI);
    if(yy>(2.0*PI))  yy-=(2.0*PI);
    end
fclose(mpd_fp);
fclose(sig_fp);
end
```

Appendix E

INDEX OF PUBLICATIONS

[1] A.C. Dent and C.F.N. Cowan, "High Linearity, High Resolution Analogue to Digital Conversion for Telecommunications", IEE Coll. Dig. No. 1987/92, pp. 2/1-2/5, Nov. 1987.

[2] A.C. Dent, M.J. Smith and C.F.N. Cowan, "Equalisation of Analogue to Digital Converter Nonlinearities", Conference Proceedings, IEEE ISCAS'88, pp. 2201-4, Helsinki, June 1988.

[3] A.C. Dent and C.F.N. Cowan, "Improving DAC Linearity by Threshold Tracking", IEE Coll. Dig. No. 1989/72, pp. 1/1-1/4, May 1989.

[4] A.C. Dent and C.F.N. Cowan,"Linearisation of Analogue to Digital Converter Nonlinearities", accepted for published in IEEE Trans. CAS, vol. 37, no 5 or 6, 1990

Equalisation of Analogue to Digital Converter Nonlinearities

A.C. Dent, M.J. Smith† and C.F.N. Cowan.

Department of Electrical Engineering, University of Edinburgh, Mayfield Road, Edinburgh. EH9 3JL Scotland.
† now with Ferranti Radar Systems, Crewe Toll, Edinburgh.

ABSTRACT

A signal processing technique called Threshold Tracking is presented which can improve the linearity of any A/D conversion circuit. The equalisation system is described and the convergence and equalisation performance in the presence of training signal noise is shown.

1. Introduction

Analogue to Digital Converters (ADC) are a vital component in many digital signal processing systems but it is in the fields of digital communication, digital audio/video, radar signal processing and nuclear spectroscopy that their performance is most critical. In these applications a large degree of linearity in the transfer function of the ADC is desirable as well as good resolution and speed, as even very small non linearities can cause severe harmonic distortion of a sampled signal.

The cumulative effect of the mechanisms causing nonlinearity is to alter the quantisation levels of the device and hence the quantisation intervals are of unequal size.

Currently the only ways to linearise an ADC, after it's initial manufacture involve altering the analogue component values by laser trimming, zener zapping, ROM trimming and other such methods. These expensive techniques provide only a temporary linearisation as component drift will still be caused by device ageing and operating temperature changes.

Several methods for compensating for ADC non linearity have been proposed [1],[2],[3],[4] but all of these are device dependent in that they are specific solutions for a certain design of ADC. Only one more general method for device linearisation has been proposed [5] but it is limited to successive approximation converters.

This paper presents a signal processing technique which generates a more linear set of quantisation levels by producing a mapping from the original device levels. This process can be performed at any point in the lifetime of the device and can hence overcome the effects of ageing.

2. Proposed System

The basic idea is to use a mapping of the original output codes and their associated non ideal quantisation levels, to a set of codes whose corresponding quantisation levels more closely match those of an ideal ADC. When the mapped codes are then assumed to be associated with ideal quantisation levels, by the rest of the signal processing system, the error between the actual and the ideal levels is reduced. This improvement can be seen in Figure 1, where the maximum deviation of the typical nonlinearity shown is about two bits.

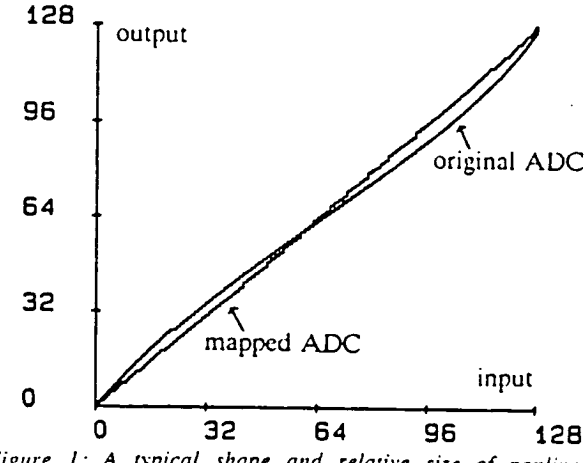

Figure 1: A typical shape and relative size of nonlinearity modelled with a maximum deviation from the ideal of two bits.

Information about the quantisation levels is extracted from a probability distribution histogram which is generated by sampling a signal of a known probability distribution function (pdf). This technique has been used extensively in the test and characterisation of ADC's [6].

The architecture of the proposed system is shown in Figure 2 and it has two modes of operation. The first of these is the conversion mode, where the digital codes produced by the non linear ADC are used to directly address the memory look-up table where a linearisation mapping is stored. The second is the equalisation mode where the training signal is sampled by the ADC to

CH2458-8/88/0000-2201$1.00 © 1988 IEEE

generate the distribution histogram. The ADC nonlinearity will distort the shape of the pdf of the training signal and information contained in the pdf about the nonlinearity can be exploited by the Threshold Tracking algorithm to perform a linearisation mapping [7].

As the algorithm deals only with ideal and distorted pdf's and never quantisation levels or actual signals, it doesn't need explicit knowledge of the ADC non linearity.

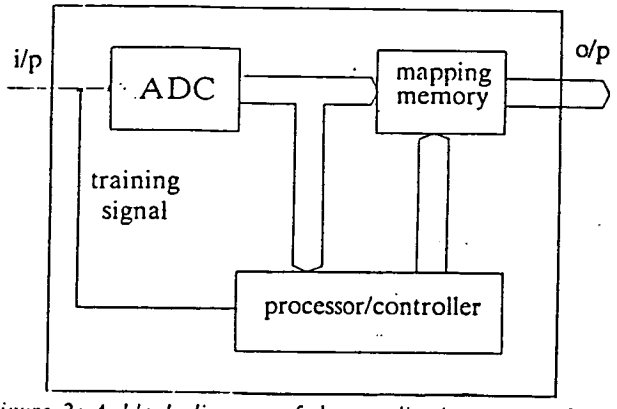

Figure 2: A block diagram of the equalisation system showing the ADC as a single element and hence of any type or specification.

3. Results

Various non linearities were modelled on various sizes of converters (numbers of bits) all with a maximum deviation of four bits from the ideal. The traditional ADC linearity measures, Differential and Integral Non Linearity (DNL and INL) were found to be inadequate for measuring the performance of the system simulations, so the alternate method of a mean deviation measurement of the device non linearity from an ideal ADC was adopted. This was formed by calculating the area between the two transfer function staircases, at each quantisation step, and taking the mean.

3.1. Harmonic Distortion

Fourier transforms have been used to analyse the reduction of harmonic distortion introduced by the ADC non linearity. Figures 3 and 4 show the frequency content of a two tone signal sampled by a non ideal and a mapped 16 bit ADC respectively. The frequencies of the sine waves were selected so as to not fit exactly into the time window of the FFT. As can be seen the harmonic content of the sampled signal was greatly distorted by the four bit nonlinearity of the ADC. Figure 4 shows that the distortion of the harmonics has been reduced by about 12 dB by the linearisation process so that they are almost buried in the quantisation noise.

3.2. Sample Density

As the number of signal samples required to give a sufficiently dense distribution was directly related to the time needed to train the system in the equalisation mode, it was important to know how performance was affected by the number of signal samples taken. Figure 5

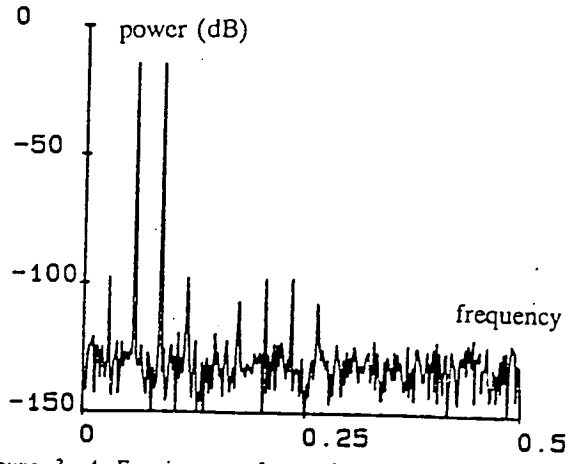

Figure 3: A Fourier transform of a two tone signal with a maximum amplitude of 75% of full range sampled by a 16 bit converter with a four bit nonlinearity shows distorted harmonics.

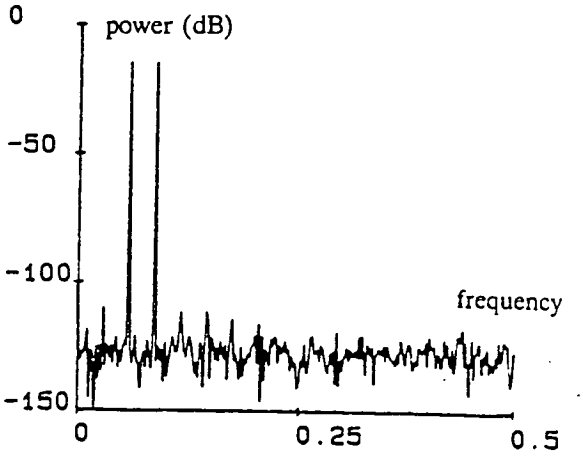

Figure 4: The mapped ADC produces 12 dB less distortion in the harmonics of the sampled sinewave.

shows the performance of 12 and 16 bit simulations trained with a wide range of training signal samples. Four bit distortions were used to emphasise the algorithm's performance as two bits in sixteen is an extremely small distortion. It can be seen that as the number of signal samples was increased and the actual input pdf tended towards the desired uniform pdf the performance of the system, which assumed a perfectly flat training pdf, improved to the point where there was no significant gain in performance for an increase in sample density. This density was approximately one sample per ADC level, implying that the higher the resolution of the device, the longer the training period needed.

4. Noise Effects on Sampled PDF

A small bipolar random signal, of a uniform pdf, was added to the sampled input signal. The result of this process was to distort the pdf of the training signal from that of the ideally flat original, although the linearisation algorithm assumed an ideal input signal. This additional distortion allowed the effects of training signal noise to

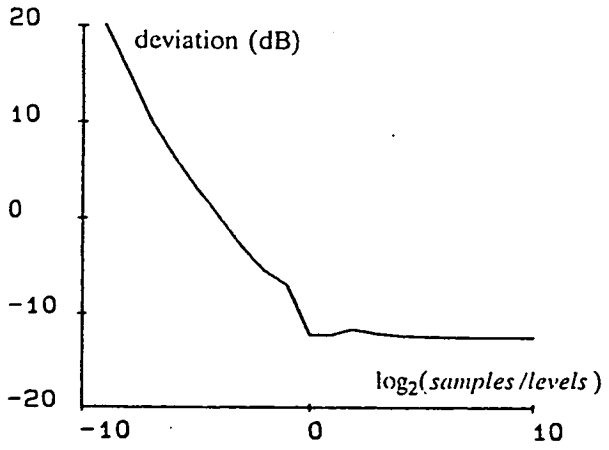

Figure 5: The performance of ADC systems after they have been equalised with varying training signal densities.

be studied. Noise levels were measured as a ratio of the maximum noise level to the maximum signal level.

As the mean deviation error of the transfer function of a simulated ADC can readily be calculated for mappings produced by training signals with and without noise, the difference between the two figures could be attributed to the performance degradation caused by the effect of noise. As the mapping algorithm's performance is dependent upon the number of samples taken to form the training signal pdf's, the degradation figures produced by the ADC simulations refer to identical situations, except for the presence of noise.

4.1. Increasing Noise Power

The behaviour of a 12 bit ADC system in the presence of noise is shown in Figure 6, with 0 dB deviation being equivalent to the original performance of the device. As might be expected, noise on the training signal does have a detrimental effect on the performance of the system. Noise levels up to around -50 dB maximum relative to the maximum input signal to the ADC have no noticeable effect on the system.

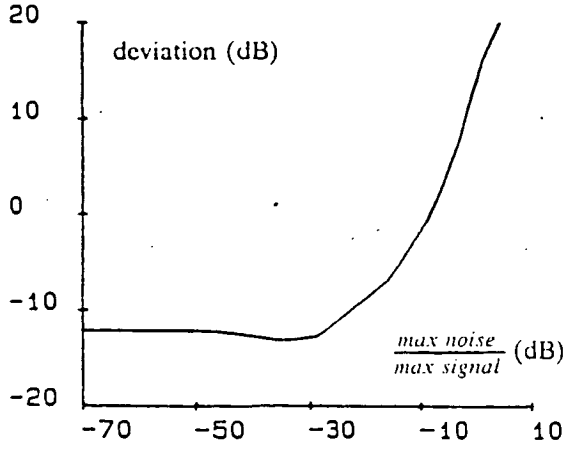

Figure 6: The effect of training signal noise on the equalised performance of a 12 bit ADC.

Noise levels of -15 dB degrade the performance of the mapping only slightly, an increase in mean deviation error of a few dB. It is not until noise signals of the same order of magnitude as the input range of the sampling ADC are present that the system fails to improve the performance of the device and actually degrades its linearity.

4.2. Effects On Training Times

Figure 7 shows how the convergence of the system to the minimum nonlinearity possible with the technique is affected by training signal noise of -46, -23 and -3 dB. It is fairly obvious that although small noise levels can and do cause training signal samples to be quantised at adjacent distribution histogram bins the cumulative effect of this error is smoothed, or averaged out, over the whole distribution so that there is no noticeable degradation in the equalisation performance.

Medium noise levels, however, do slow down convergence ie a larger signal sample is required to achieve maximum performance. This may be explained by the coarser effects of the larger noise levels taking more signal samples to be averaged out of the distribution.

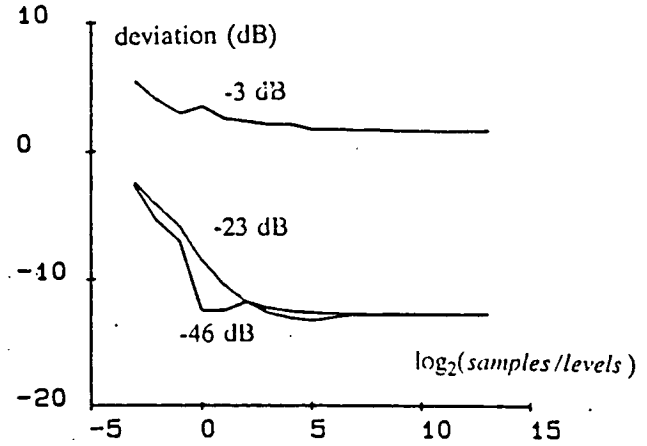

Figure 7: The training performance of a 12 bit device in noise levels of -46, -23 and -3 dB. There is no discernible degradation at the -46 dB level.

Very large noise levels, causing large distortions of the training signal distribution irrevocably defeat the equalisation system due to the large number of signals which are quantised in the two bins at the extreme ends of the distribution.

From these results it is obvious that the system can be made noise tolerant merely by increasing the number of training signal samples taken to form the distribution histogram and it can also be seen, from Figure 8, that by increasing the size of the sampled distribution it is possible to increase the point before which noise starts to affect performance. It is clear, however, that there is a law of diminishing returns operating here and that the noise tolerance can not be improved indefinitely as there seems to be some asymptotic behaviour in Figure 8 around the -10 dB point.

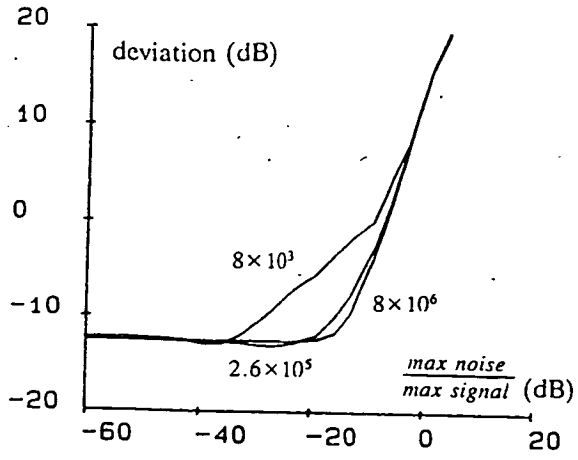

Figure 8: A 12 bit ADC trained with 8×10^3, 2.6×10^5 and 8×10^6 samples in constant noise, showing improved noise immunity by increasing the training period.

4.3. Relative Noise and Samples

Simulations were run over a large range of ADC resolutions to see how performance in noise and system tolerance were affected by the size of the least significant bit of the device. When noise levels were kept constant and simulations were run with equal sample densities it was found that the performance degraded as the size of the device lsb decreased. This implies that the noise level relative to the size of the device lsb was the important factor rather than the absolute size of the noise. Hence, higher resolution ADC equalisation systems can tolerate less noise. Figure 9 shows four almost level traces over a wide bit resolution range for equal relative noise levels for each line. The figure shows that over wide bit ranges and relative noise levels, tolerance is affected by the noise levels relative to the size of the least significant bit of the device.

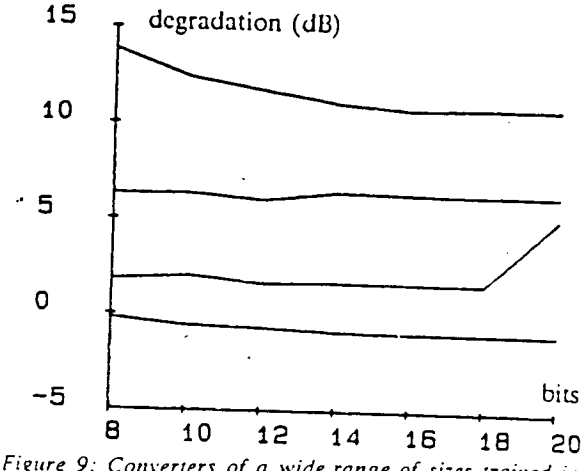

Figure 9: Converters of a wide range of sizes trained in noise. The four lines each represent an equal level of noise measured relative to 1 lsb of the device.

5. Conclusion

It is envisaged that the current system could be used to calibrate an ADC at the manufacturing stage, programming a look-up table specific to each converter to improve overall device linearity. This would not however, resolve ageing and drift problems, unless the device was recalibrated at a later stage. Future work will be aimed at achieving a reduction in the amount of memory required to implement the look-up tables.

Acknowledgement

This work has been supported by SERC and a CASE award from British Telecom Research Laboratories, Ipswich.

References

[1] K. Maio, M. Hotta, N. Yokozawa, M. Nagata, K. Kaneko and T. Iwasaki,"An Untrimmed D/A Converter with 14 Bit Resolution," IEEE J-SC vol. 16, no. 6, pp. 616-620,Dec. 1981.

[2] R.J. Van de Plassche and H.J. Schouwenaars,"A Monolithic 14 Bit A/D Converter," IEEE J-SC vol. 17, no. 6,pp. 1112-1117, Dec. 1982.

[3] H-S. Lee, D.A. Hodges and P.R. Gray,"A Self-Calibrating 15 Bit CMOS A/D Converter," IEEE J-SC vol. 19, no. 6, pp. 813-819, Dec. 1984.

[4] J. Croteau, D. Kearth and D. Welland, "Autocalibration Cements 16 bit Performance," Electronic Design, 4 Sep. 1986.

[5] Z.G. Boyacigiller, B. Weir and P.D. Bradshaw,"An Error Correcting 14b/20μs CMOS A/D Converter," Tech. Dig. IEEE ISSCC 1981, pp. 62-63.

[6] M. Vanden Bossche, J. Schoukens and J. Renneboog,"Dynamic Testing and Diagnostics of A/D Converters," IEEE Trans. CAS vol. 33, no. 8, pp. 775-785, Aug. 1986.

[7] A.C. Dent and C.F.N. Cowan,"High Linearity, High Resolution Analogue to Digital Conversion for Telecommunications," IEE Coll. Dig. No. 1987/92, pp. 2/1-2/5, Nov. 1987.