DEVELOPMENT AND EVALUATION OF A PROGRAMMABLE RADIO FREQUENCY SIGNAL GENERATOR USING DIRECT DIGITAL SYNTHESIS

by

Bernard van der Walt

DISSERTATION

Submitted in fulfilment of the requirements for the

MAGISTER TECHNOLOGIAE: ENGINEERING: ELECTRICAL

in the

FACULTY OF ENGINEERING

(Department Electrical Engineering)

at the

TECHNIKON FREE STATE

Supervisor: Mr D. Brynard M. Eng.

and

Co-supervisor: Mr G.D. Jordaan M. Dip. Tech.

Bloemfontein, 1 June 1997

DECLARATION

I, Bernard van der Walt, do hereby declare that this research project, submitted for the degree MAGISTER TECHNOLOGIAE: ENGINEERING: ELECTRICAL, is my own independent work that has not been submitted before to any institution by me or anyone else as part of any qualification.

B van der Walt

1 June 1997

ACKNOWLEDGEMENTS

All glory to God Almighty, for guidance during this research.

Appreciation to Mr D Brynard for acting as supervisor and Mr GD Jordaan as co-supervisor.

Special thanks to Ms V Hanekom for philological assistance.

Appreciation to my wife Ronél and sons, Luan and Elandré for their patience and support.

SUMMARY

Most commercially available signal generators make use of a phase-locked loop in combination with analogue frequency synthesis to generate the desired frequency range. Advances in the development of components being used in digital frequency synthesis have made the use of direct digital synthesis (DDS) a viable option in radio frequency (RF) signal generation.

The project consists of designing the interfacing between a DDS unit and a microcontroller to provide a versatile frequency generator in the lower high frequency (HF) spectrum.

The research was aimed at testing the following hypothesis:

A programmable Radio Frequency signal generator can be developed, using a DDS-based system with a microcontroller providing the required intelligence. A continuously variable frequency range in 1 Hz steps over a spectrum of 0 - 10 MHz can be achieved.

The following features were included in the design of the signal generator:

- Setting the generator to a specific frequency;
- Displaying the frequency and prompts from the microcontroller on a liquid crystal display;
- Interfacing with a keypad;
- Interfacing with a personal computer for remote RS232 operation;
- Interfacing with a rotary optical encoder for up-and-down frequency control;
- Sweeping of a range of frequencies;

- Setting the step size of frequency increments;
- Frequency shift keying (FSK) capability.

The above features allowed ample demonstration of the software control over the associated hardware and enabled easy evaluation of the product.

To evaluate the product, it was decided to concentrate on the following measurable aspects of a typical radio frequency (RF) signal generator:

- The accuracy of the output frequency;
- Evaluating the frequency range limits of the generator;
- Making a spectral analysis of the output signal.

During the execution of the project, insight was gained with respect to the following:

- DDS theory;
- DDS hardware interfacing;
- C-programming as well as using the versatile DS5000 microcontroller;
- The importance of sound design principles in a hybrid digital and analogue radio frequency project.

- Setting the step size of frequency increments;
- Frequency shift keying (FSK) capability.

The above features allowed ample demonstration of the software control over the associated hardware and enabled easy evaluation of the product.

To evaluate the product, it was decided to concentrate on the following measurable aspects of a typical radio frequency (RF) signal generator:

- The accuracy of the output frequency;
- Evaluating the frequency range limits of the generator;
- Making a spectral analysis of the output signal.

During the execution of the project, insight was gained with respect to the following:

- DDS theory;
- DDS hardware interfacing;
- C-programming as well as using the versatile DS5000 microcontroller;
- The importance of sound design principles in a hybrid digital and analogue radio frequency project.

OORSIG

Die oorgrote meerderheid van seingenerators maak gebruik van 'n fasesluitlus gekombineerd met analoog frekwensiesintese om die gewenste frekwensiestrek te kan genereer. Vooruitgang in die ontwikkeling van komponente vir digitale frekwensiesintese het die doeltreffende gebruik van direkte digitale frekwensiesintese (DDS) in die radiofrekwensiespektrum vergemaklik.

Die projek behels die interkoppeling tussen 'n DDS-eenheid en 'n mikrobeheerder om 'n veelsydige seingenerator in die laer hoëfrekwensiespektrum te vorm.

Die doel van die ontwikkelingsprojek was om die volgende hipotese te toets:

'n Programmeerbare radiofrekwensie seingenerator kan ontwikkel word deur gebruik te maak van 'n DDS-gebaseerde stelsel met 'n mikrobeheerder as eenheid van intelligensie. 'n Deurlopende frekwensiestrek van 0 - 10 MHz in stappe van 1 Hz kan met so 'n stelsel bereik word.

Die volgende eienskappe is in die ontwerp van die seingenerator ingesluit:

- Die stel van die eenheid op 'n spesifieke frekwensie;
- Die vertoon van die frekwensie en data-insetversoeke op 'n vloeibare-kristal vertooneenheid;
- Interkoppeling met 'n sleutelbord;
- Interkoppeling met 'n rekenaar vir afstandbeheerde RS232-werking;
- Interkoppeling met 'n roteerbare optiese enkodeerder (ROE) vir op-en-af frekwensiebeheer;
- Die veeg van 'n frekwensiestrek;

- Die stel van ‘n stapgrootte vir frekwensieveranderings;
- Frekwensieskuifsleutel (FSK) fasiliteit.

Die suksesvolle implementering van bogenoemde eienskappe het voldoende beheer van die sagteware oor die hardeware gedemonstreer en die maklike evaluering van die stelsel moontlik gemaak..

Die evaluering van die finale produk is aan die hand van die volgende meetbare aspekte van ‘n tipiese radiofrekwensie-seingenerator gedoen:

- Akkuraatheid van die uitsetfrekwensie;
- Evaluasie van die maksimum frekwensiereikwydte van die generator;
- Uitvoer van ‘n spektrale analise van die uitsetsein.

Kundigheid t.o.v. die volgende aspekte is gedurende die ontwikkeling van die projek opgedoen:

- DDS-teorie;
- DDS-hardeware-interkoppeling;
- C-programmering, sowel as die gebruik van die DS5000-mikrobeheerder;
- Die belangrikheid van gesonde ontwerpsbeginsels in ‘n projek wat van hibriede digitale en radiofrekwensie-analoog tegnieke gebruik maak.

CONTENTS

1. Introduction

2. Literature Overview

3. DDS Theory

4. Hardware Development

Appendices

LIST OF FIGURES

Chapter 3

Chapter 4

Chapter 5

LIST OF ACRONYMS

ADC	Analogue-to-digital converter
AM	Amplitude modulation
ASIC	Application-specific integrated circuit
BCD	Binary-coded decimal
CMOS	Complimentary metal oxide semiconductor
DAC	Digital-to-analogue converter
dB	Decibel
DDS	Direct digital synthesis
DIP	Dual inline package.
FM	Frequency modulation
FSK	Frequency shift keying
FSW	Frequency setting word
GaAs	Gallium Arsenide
GPS	Global positioning system
HF	High frequency
LCD	Liquid crystal display
LPF	Low pass filter
LSB	Least significant bit
LSI	Large scale integration
MSB	Most significant bit
NCO	Numerically controlled oscillator
NRC	Noise reduction circuit
PC	Personal Computer
PLL	Phase locked loop
PM	Phase modulation
RADAR	Radio Detection and Ranging
RAM	Random access memory
RF	Radio frequency
ROE	Rotary optical encoder
ROM	Random access memory
SSB	Single sideband
TTL	Transistor-transistor-logic
UART	Universal asynchronous receiver-transmitter
VCO	Voltage-controlled oscillator

CHAPTER 1- INTRODUCTION

1.1 Definition of Investigation

Most commercially available signal generators make use of a phase-locked loop in combination with analogue frequency synthesis to generate the desired frequencies. Advances in the development of components being used in digital frequency synthesis have made the use of DDS a viable option in RF signal generation.

This research project incorporates Direct Digital Synthesis (DDS) into a micro-processor controlled frequency generator.

The project consists of the interfacing between a DDS unit and a microcontroller to provide a versatile frequency generator in the lower High Frequency (HF) spectrum.

Software for the microcontroller to control specific functions in the generator was also developed.

1.2 Purpose of Study

The purpose of this study was to develop a radio frequency signal generator variable from 0 - 10 MHz, utilising direct digital frequency synthesis. The result of this project is a low-cost, very agile frequency generator.

CHAPTER 1- INTRODUCTION

1.1 Definition of Investigation

Most commercially available signal generators make use of a phase-locked loop in combination with analogue frequency synthesis to generate the desired frequencies. Advances in the development of components being used in digital frequency synthesis have made the use of DDS a viable option in RF signal generation.

This research project incorporates Direct Digital Synthesis (DDS) into a micro-processor controlled frequency generator.

The project consists of the interfacing between a DDS unit and a microcontroller to provide a versatile frequency generator in the lower High Frequency (HF) spectrum.

Software for the microcontroller to control specific functions in the generator was also developed.

1.2 Purpose of Study

The purpose of this study was to develop a radio frequency signal generator variable from 0 - 10 MHz, utilising direct digital frequency synthesis. The result of this project is a low-cost, very agile frequency generator.

1.3 Hypothesis

The research was aimed at testing the following hypothesis:

A programmable Radio Frequency signal generator can be developed, using a DDS-based system with a microcontroller providing the required intelligence. A continuously variable frequency range in 1 Hz steps over a spectrum of 0 - 10 MHz can be achieved.

1.4 Scope of Project

To determine the final scope of the project, it was decided to develop the signal generator to include the following features:

- Setting the generator to a specific frequency;
- Displaying the frequency and prompts from the microcontroller on a Liquid crystal display;
- Interfacing with a keypad;
- Interfacing with a personal computer for remote RS232 operation;
- Interfacing with a rotary optical encoder for up and down frequency control;
- Sweeping of a range of frequencies;
- Setting the step size of frequency increments;
- Frequency shift keying (FSK) capability.

The above features allowed ample demonstration of the software control over the associated hardware and enabled easy evaluation of the product.

To evaluate the product, it was decided to concentrate on the following measurable aspects of a typical radio frequency (RF) signal generator.

- Measuring the accuracy of the output frequency;
- Evaluating the frequency range limits of the generator;
- Testing the accuracy of the step size;
- A spectral analysis of the output signal measuring the following:

  - spurious signals

  - harmonics & sub-harmonics

  - phase noise.

The aim of the evaluation was to determine the performance of this specific DDS generator, in order to be able to compare it with existing commercially available equipment, as well as its compliance with the hypothesis.

1.5 Development

The signal generator consists of different modules that can be tested and evaluated on their own, therefore the development was done in stages as outlined below:

- Microcontroller section and software;
- The DDS Application-specific Integrated Circuit (ASIC) module;
- Digital-to-analogue converter stage;
- Low-pass filter design.

Evaluation of the complete project was done when all the modules had been interfaced and tested.

CHAPTER 2 - LITERATURE OVERVIEW

2.1 Frequency Synthesis

Frequency synthesis is the generation of a new frequency from some reference source, where the characteristics of the output correlate with the characteristics of the reference signal. The conventional methods of frequency synthesis include:

- Direct frequency synthesis
- Phase-locked loop (PLL)
- Direct digital synthesis (DDS)
- Hybrids of the above

Each method has certain advantages and disadvantages when compared to other methods. Each application requires selection, based upon the most acceptable combination of compromises.

Since the early 1980s there has been a steadily increasing degree of interest in the digital method of generating frequencies, especially in the radio frequency (RF) spectrum. Direct digital synthesis (DDS) only became a viable option to use when constraints placed on it by component limitations were overcome. Advancement in the development of high-speed DDS application-specific integrated circuits (ASICs), as well as digital-to-analogue converters, has benefited the field of DDS applications. TriQuint has developed digital-to-analogue converters (DACs) that can operate at 1000 million (1G) samples per second. This type of advancement has helped to open the field for wide-band DDS applications (Gallant, 1989: 102).

2.2 DDS Applications

Typical DDS applications include spread-spectrum communications, arbitrary wave-form generation, test signal generators, receiver local oscillators, quadrature synthesis, FM stereo broadcast generators, frequency-hopping radios, video wave-form generators and wide application possibilities in high-speed data communications (Gallant, 1989: 95). Similar diverse applications exist for audio applications, where slower clock frequencies allow for 16 or more bit DACs. A typical application in this field would be a music synthesiser or a high performance audio test sweep oscillator (Zavrel, 1988: 32).

Most of the performance limitations of synthesisers can be traced to modulation and frequency agility. The great power of a numerically controlled oscillator (NCO) DDS system is that both functions are performed simultaneously. Furthermore, these two functions are accomplished with precision and linearity that is impossible with analogue techniques.

Amplitude modulation (AM), frequency modulation (FM) and phase modulation (PM) can all be simultaneously implemented with an NCO. Therefore, nearly any conceivable complex or simple modulated wave form is possible within the system bandwidth limitations. Typically one microsecond or less, depending on the clock speed of the NCO, is needed to execute a change in frequency on an NCO. Also, no current spike is added to the signal when the transition from one frequency to another other takes place (Hickman, 1992: 631).

2.3 Comparison with PLL Synthesisers

PLL synthesisers can offer fine frequency resolution and very low levels of spurious outputs at comparatively low cost. A main drawback of PLL is slow switching times - due to the loop filter settling time (Hickman, 1992: 632). PLL synthesiser performance has approached theoretical operating limitations given loop bandwidths, settling times and phase noise trade-offs. In contrast, DDS techniques are limited mainly by DAC performance (Zavrel & Mc Cune, 1988: 3). Paragraphs 2.3.1 to 2.3.3 deal with some of the advantages of DDS over PLL.

2.3.1 Settling Time

In a DDS system there are virtually no loop filter time constants that will slow down the frequency settling time. In a well-designed DDS system it takes only two clock pulses to load the new frequency data. The NCO simply begins sequencing the new information, limiting the delay only to the time it takes to load the new frequency into a register. As mentioned earlier, this typically takes less than one microsecond. The limiting factors, however, are in the settling time of the DAC and the low pass filter.

2.3.2 Phase Continuity

Phase continuous frequency hops are a great advantage of DDS. This means that undesirable glitches from a change in frequency are completely eliminated. The heart of a DDS system is a phase accumulator loaded with a frequency word. Sequencing this word in the phase accumulator, the NCO generates a certain frequency. By changing the frequency word, the output frequency will change accordingly without any phase distortion or delay.

Phase delay in a PLL synthesiser is related to the settling time. In a DDS system the signal consists of discrete amplitude samples addressed at a certain rate to produce a sine wave. When a new frequency word is loaded, the new frequency sine wave begins from the last discrete amplitude value of the old frequency. Consequently a DDS system is truly phase-continuous.

Phase continuity is extremely useful in high-speed data applications where wide bandwidths are required.

2.3.3 Modulation Properties

In an NCO, the three most commonly used modulation techniques, AM, FM and PM, are defined by digital numbers.

The phase accumulator containing the frequency word, typically consists of a 32 or 48 bit register. By digitally adding or subtracting from this register, frequency modulation can be effected. Thus FM over the entire bandwidth with 32 or 48 bit linearity is possible. No PLL system can compete with this feature over multiple octaves of frequency.

Frequency and phase deviation in a DDS system are independent of the centre frequency. In an FM modulated PLL system the deviation will not stay constant when the carrier frequency is varied. In a DDS system the deviation stays constant over the entire operating bandwidth. Modulation in the NCO is an additive process, not a scaling multiplicative process (Zavrel, 1988: 29)

2.4 DDS Application-specific Integrated Circuits (ASIC) Manufacturer Data

The purpose of including manufacturer information in this chapter is to provide the reader with a background of currently available DDS technology. Information given in the following paragraphs are taken directly from new frequency sine wave begins from the last discrete amplitude value of the old frequency. Consequently a DDS system is truly phase-continuous.

Phase continuity is extremely useful in high-speed data applications where wide bandwidths are required.

2.3.3 Modulation Properties

In an NCO, the three most commonly used modulation techniques, AM, FM and PM, are defined by digital numbers.

The phase accumulator containing the frequency word, typically consists of a 32 or 48 bit register. By digitally adding or subtracting from this register, frequency modulation can be effected. Thus FM over the entire bandwidth with 32 or 48 bit linearity is possible. No PLL system can compete with this feature over multiple octaves of frequency.

Frequency and phase deviation in a DDS system are independent of the centre frequency. In an FM modulated PLL system the deviation will not stay constant when the carrier frequency is varied. In a DDS system the deviation stays constant over the entire operating bandwidth. Modulation in the NCO is an additive process, not a scaling multiplicative process (Zavrel, 1988: 29)

2.4 DDS Application-specific Integrated Circuits (ASIC) Manufacturer Data

The purpose of including manufacturer information in this chapter is to provide the reader with a background of currently available DDS technology. Information given in the following paragraphs are taken directly from specification sheets and application notes provided by the suppliers. Only the top range products of each manufacturer and his specific or unique applications will be discussed.

2.4.1 Qualcomm

The Qualcomm Q2234 family of Dual Direct Digital Synthesisers generates high resolution digitised sine wave signals using phase accumulation techniques combined with a patented on-chip sine lookup table and Noise Reduction Circuit (NRC). The Q2334 contains two independent DDS functions controlled from a single microprocessor interface. The interface controls both the phase and the frequency of the generated sine waves, as well as the device's operating mode. Synchronous inputs are also provided to allow for phase and frequency modulation (Qualcomm, 1993:5).

Three pin-compatible versions of the Q2334 DDS are available, with 20, 30 and 50 MHz maximum clock speeds respectively. The two independent on-chip DDS functions provide an efficient technique for implementation of full-duplex systems, quadrature oscillators, and spread spectrum systems. This product is widely used in military frequency hopping systems (Qualcomm, 1993:5).

2.4.2 Stanford Telecom

In the DDS market, Stanford Telecom is one of the world leaders with a comprehensive selection of chip, board and chassis-level DDS systems.

Stanford's wide range of DDS products vary from the STEL 1174, a low cost 16 bit 50 MHz DDS, to the STEL 2173, a 32 bit GaAs based chip, capable of output frequencies of up to 500 MHz. The product range covers ASIC units with clock rates from 25 MHz up to 1000 MHz and 16 to 48 bit phase resolution. They also offer a wide range of dedicated NCO units - for example a 60 MHz Quadrature Amplitude Modulator, a 60 MHz chirp NCO and several Modulated NCOs. Dual DDS packages and custom-made DDS sub-systems are also available from Stanford Telecom.

Stanford products are applicable to a wide range of applications, including radio, RADAR, GPS, and spread spectrum (Stanford Telecom AN 102 Edition 2, 1990 and Edition 7, 1992).

2.4.3 Analog Devices

Analog Devices offer the AD7008 CMOS DDS Modulator. The AD7008 is a numerically-controlled oscillator with 32 bit phase resolution, supporting clock rates from 20 MHz to 50 MHz (Analog Devices, 1993).

This unit has a sine and cosine lookup table as well as a 10 bit digital-to-analogue converter integrated on a single CMOS chip. Modulation capabilities are provided for phase modulation, frequency modulation, and both in-phase and quadrature amplitude modulation suitable for Single Sideband (SSB) generation. Special modulation registers operating independently and asynchronously from the DDS clock have been incorporated into the unit. The registers can be loaded in serial or parallel, which simplifies implementing different modulation techniques on this product (Analog Devices, 1993).

2.4.4 Sqiteq Electronics Inc.

Sqiteq products and technologies are available as ASICs, at board level, in modules or as complete subsystems depending on customer requirements. Their latest DDS products cover a bandwidth of 1200 MHz with spurious signals lower than -60 dBc over the full spectrum. Their developments are concentrated on DDS+PLL systems as well as DDS+Mixer designs. The latter are developed especially for the cellular industry (Sqiteq electronics Inc.,1988).

2.5 Conclusion

A lack of information about DDS systems in textbook format resulted in that most of the information on DDS theory, products and systems products was taken from specification sheets and application notes provided by the different suppliers.

DDS has outstanding advantages over more conventional methods of frequency synthesis. It is, however, widely used in combination with PLL circuits to achieve a more versatile method of frequency synthesis.

CHAPTER 3 - DDS Theory

3.1 Introduction

Understanding the concept of direct digital synthesis (DDS) is subject to a knowledge of digital sampling theory.

To be able to store an analogue signal digitally, it first needs to be sampled at the proper rate. After sampling, the instantaneous amplitude value is quantified to produce a digital amplitude value which is stored in a digital memory. Quantization is done using an analogue-to-digital converter (ADC). The ADC converts the analogue sample into a digital value. The sample values can be processed or stored consecutively in a memory bank.

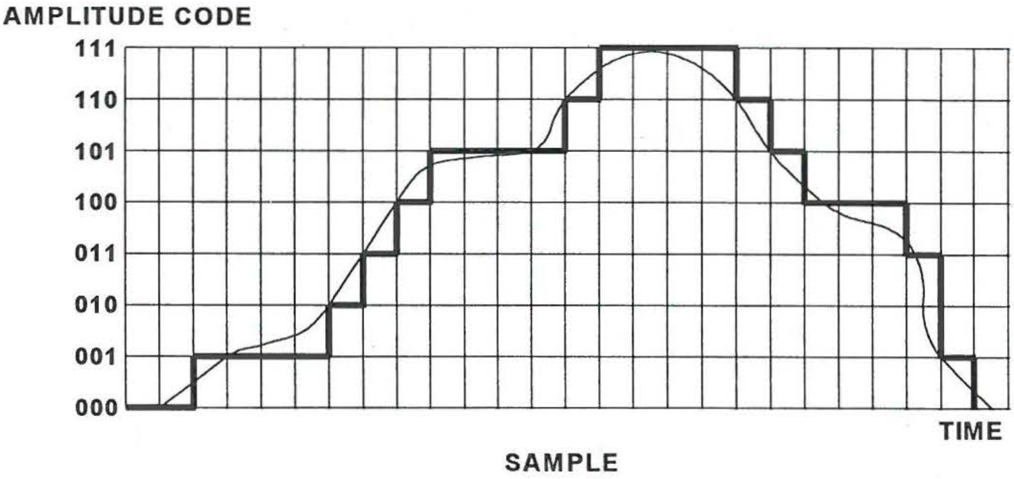
To regenerate the analogue signal, it is necessary to address the memory locations consecutively at the same rate that data was written into the memory. Each memory location contains the digital amplitude representation of that specific phase sample of the analogue signal. The digital amplitude value is now converted to an analogue value by means of a digital-to-analogue converter (DAC). The consecutive digital-to-analogue converted values, after filtering, reconstruct the initial analogue signal.

Direct digital synthesis relies on the same basic principles as mentioned above. The memory bank of a DDS system, however, already contains a given function. This data is fixed and the output wave form can be changed by changing the rate at which the samples are being converted back to analogue values.

Before starting with DDS theory in detail, it may be necessary to explain some of the factors involved in the digital representation of analogue signals. This includes an explanation of Nyquist rate, digital storage of samples and aliasing distortion. These are all factors that influence the representation of a digitally stored analogue signal.

3.2 Analogue-to-digital Conversion

Sampling is the acquisition of amplitude values from a continuous signal at discrete time intervals. After sampling, the analogue signal is only represented at discrete times, with the value of each sample equal to that of the original analogue signal, at the instant of sampling. The analogue sample is therefore represented by a numeric value at a specific position in the time-domain.



ADC	C	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
PARALLEL	B	0	0	0	0	0	0	1	1	0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	1	0	0	0
OUTPUTS	A	0	0	1	1	1	1	0	1	0	1	1	1	1	0	1	1	1	1	0	1	0	0	0	1	1	0	0

Fig 3.1: Conversion of an analogue signal into 3 bit digital format.

Figure 3.1 shows a schematic representation of the digital conversion of an analogue wave form. At each sample time, the digital code corresponding to the voltage level appears on the output of the ADC. An address can be given to each sample in a memory bank. The time-based values are read into the memory consecutively to eventually capture the whole signal in digital format.

3.3 Sampling Rate

The sampling rate is the number of samples taken in a specific time frame. Proper sampling of a signal must conform to the Nyquist theorem to be able to represent it correctly. According to this theorem, to have an accurate digital representation, the sampling frequency must be at least twice the highest frequency component of the signal being sampled (Ifeachor & Jervis, 1993:15).

The specific phase amplitude sample of the analogue signal is represented by a binary number. This number is updated at the sampling rate. If the sampling rate is too low, the digital amplitude values will represent a low frequency alias, as well as the original analogue signal. This condition is called under-sampling.

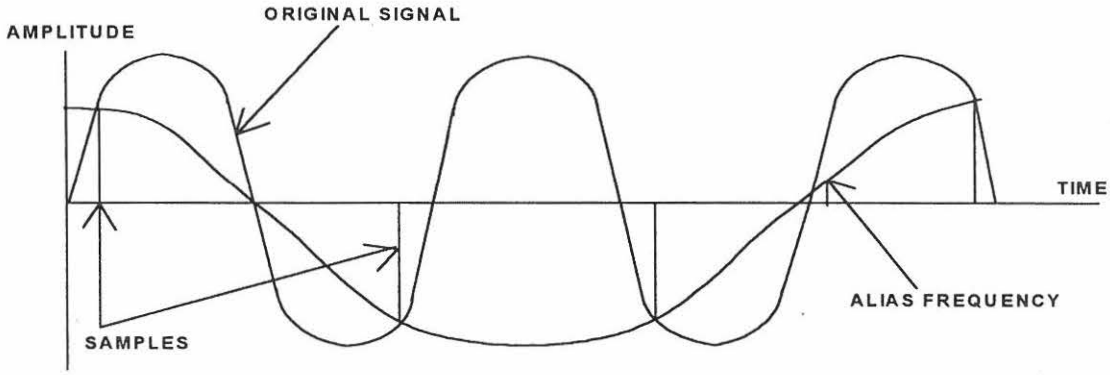


Fig 3.2: Sampling at a rate less than the Nyquist rate.

Figure 3.1 shows a schematic representation of the digital conversion of an analogue wave form. At each sample time, the digital code corresponding to the voltage level appears on the output of the ADC. An address can be given to each sample in a memory bank. The time-based values are read into the memory consecutively to eventually capture the whole signal in digital format.

3.3 Sampling Rate

The sampling rate is the number of samples taken in a specific time frame. Proper sampling of a signal must conform to the Nyquist theorem to be able to represent it correctly. According to this theorem, to have an accurate digital representation, the sampling frequency must be at least twice the highest frequency component of the signal being sampled (Ifeachor & Jervis, 1993:15).

The specific phase amplitude sample of the analogue signal is represented by a binary number. This number is updated at the sampling rate. If the sampling rate is too low, the digital amplitude values will represent a low frequency alias, as well as the original analogue signal. This condition is called under-sampling.

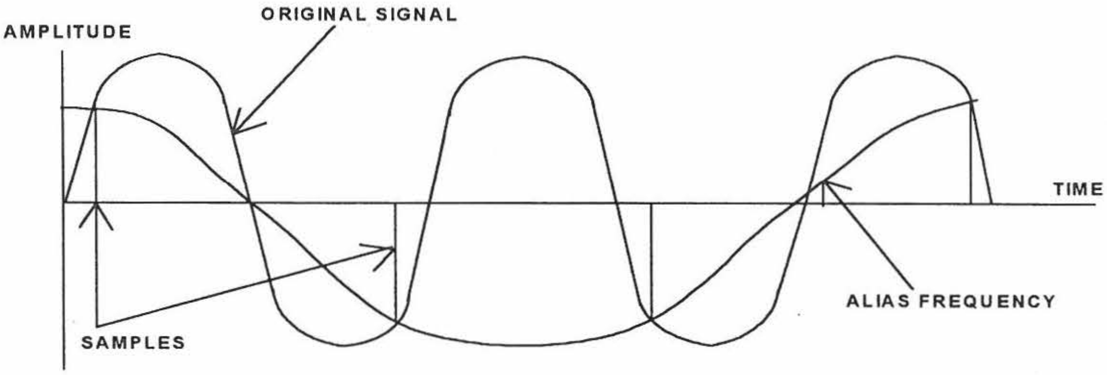


Fig 3.2: Sampling at a rate less than the Nyquist rate.

The samples in figure 3.2 represent a low frequency alias as well as the original signal. Trying to reproduce the original signal from the digital samples, will result in only the alias signal being produced. The original signal is lost as a result of major portions of it not being sampled.

Sampling is done by multiplying the original analogue signal by a periodic pulse signal. The Fourier transform of this signal yields a $\frac{\sin x}{x}$ wave form (SINC) (Zavrel, 1993:10-20).

In the sampling process, because multiplication of two signals is involved, double sideband images of the original signal, centred at multiples of the sampling frequency, are formed about the sampled signal. The shaded portions of Figure 3.3 show these harmonic signals. For clarity only the first image spectrum around the sampling frequency is shown. The sampling frequency (FS) is exactly twice the highest frequency (FA). To correctly represent the analogue signal, the image must be filtered out. In this case only an ideal filter response would succeed in filtering out the unwanted signals. It is impossible to implement such a filter, therefore the desired pass band cannot be properly filtered from the composite signal containing the alias signals.

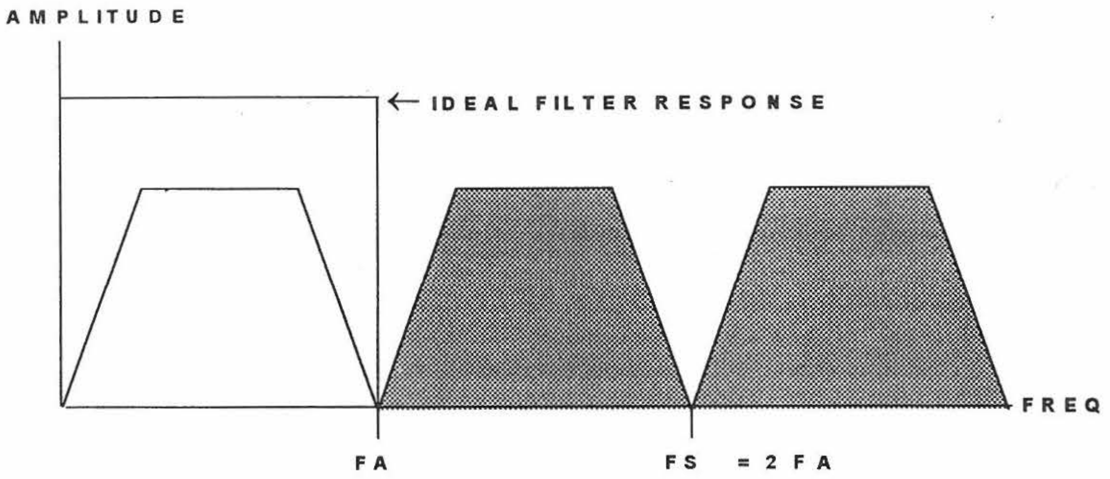


Fig 3.3: Sampling at exactly the Nyquist rate.

Figure 3.4 shows that when the sampling frequency (FS) is not equal to, or higher than, the Nyquist rate, there is an overlap between the higher frequencies of the desired signal and the first image frequency. Because of this overlap, the image cannot be eliminated without eliminating some of the desired signal. This is called aliasing or foldover and is the direct result of under-sampling. Aliasing is a major cause of noise in the reconstruction of an analogue signal.

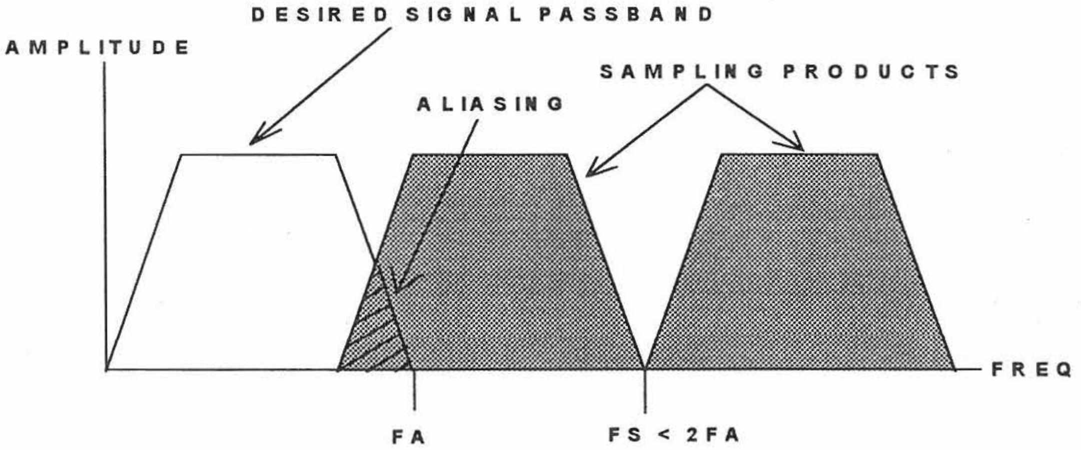


Fig 3.4: Sampling at a rate less than the Nyquist rate.

In Figure 3.5 it is shown that, when the sampling rate is above the Nyquist rate (FS>2FA), there is some separation between the desired signal and the image spectrum. With this gap between the two spectra, practical filters can filter out the image and leave the desired signal unattenuated. The higher the sampling-rate with respect to the highest frequency in the composite analogue signal, the further the image is from the desired signal.

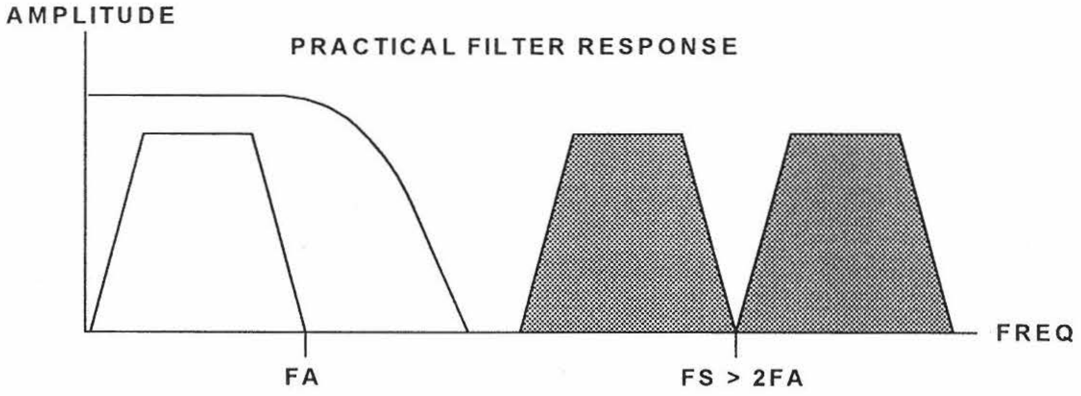


Fig 3.5: Sampling at a rate well above the Nyquist rate.

3.4 Sampling Resolution

Analogue signals consist of an infinite number of amplitudes. An ADC that contains N output bits, only represent 2^N amplitudes. Therefore an ADC cannot perfectly represent the continuous wave form of an analogue signal. Sampling resolution therefore is a measure of the accuracy with which the digital sample represents the analogue voltage level (Hickman, 1992: 630).

The number of bits in an ADC that represents the voltage value determines the resolution of the converter. An analogue signal can be shown as a continuous line while the digital representation thereof will have a stair-step characteristic. Figure 3.6 shows a linear analogue transfer function and a digital representation of the same signal.

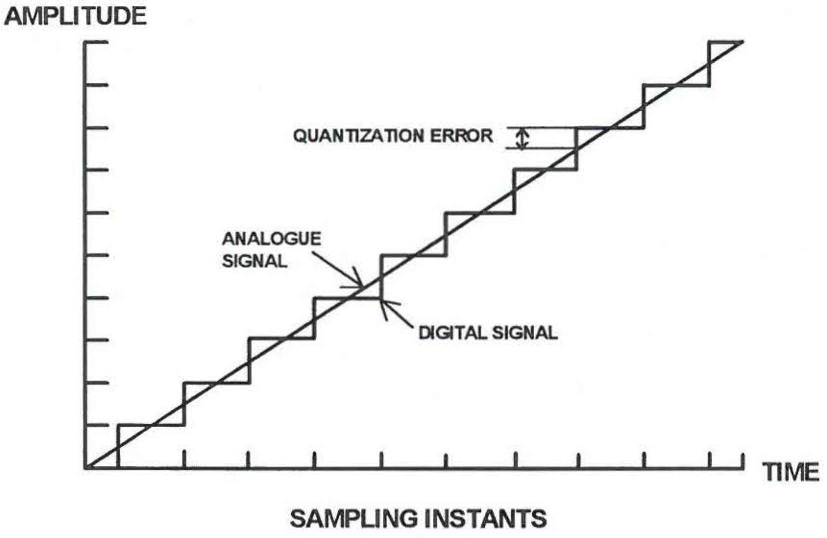


Fig 3.6: Digitising of an analogue transfer function showing the quantization errors.

At each sampling instant shown in Figure 3.6, there is a difference between the analogue signal and the closest available digital value. This difference is called the quantization error, which introduces noise into the sampled signal, called quantization noise. The higher the resolution of the ADC, the lower the quantization error.

It is clear that the best resolution is obtained with frequent and long wordlength samples. There is a trade-off, however. Long wordlength samples require longer sampling times and it takes time to process these samples.

3.5 Direct Digital Synthesis

Direct digital synthesis basically means that the three most important parameters of a synthesised wave form - frequency, phase and amplitude - are defined or calculated digitally. Any frequency can be defined as a change in phase per unit time. Mathematically this can be represented as:

$$\omega = 2\pi f = \frac{\Delta\phi}{\Delta t} \qquad \textbf{3.1}$$

This formula clearly indicates that with Δt a fixed value, the only way to vary the frequency is to alter the change in phase. The faster the change in phase, the higher the frequency.

A simple block diagram of a basic DDS system is shown in Figure 3.7. A basic system consists of a phase accumulator, lookup table, digital-to-analogue converter and an analogue filter.

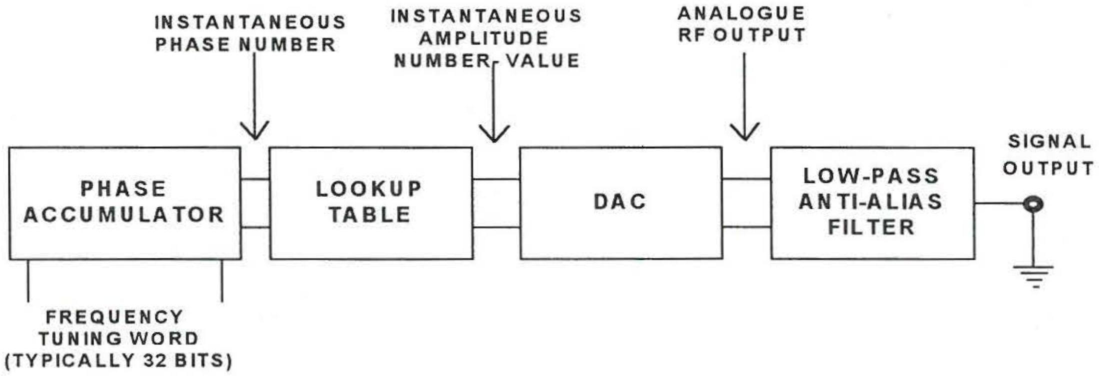

INSTANTANEOUS PHASE NUMBER
INSTANTANEOUS AMPLITUDE NUMBER, VALUE
ANALOGUE RF OUTPUT
SIGNAL OUTPUT
PHASE ACCUMULATOR
LOOKUP TABLE
DAC
LOW-PASS ANTI-ALIAS FILTER
FREQUENCY TUNING WORD (TYPICALLY 32 BITS)

Fig 3.7: Block diagram of a basic DDS system.

The function of each block will be explained separately. Before going into detail, it may be helpful to relate some basic oscillator fundamentals to digital techniques.

Zavrel (1993:10-17) compares the phase accumulator to a point on a rotating wheel. Each complete rotation of the wheel represents one cycle of the oscillator wave form. The rotation speed corresponds to the frequency, and if the wheel spins at a constant speed the oscillator produces a sine wave. Therefore each point on the wheel corresponds to a particular phase point on the oscillator wave form. Figure 3.8 shows a schematic representation of this relation.

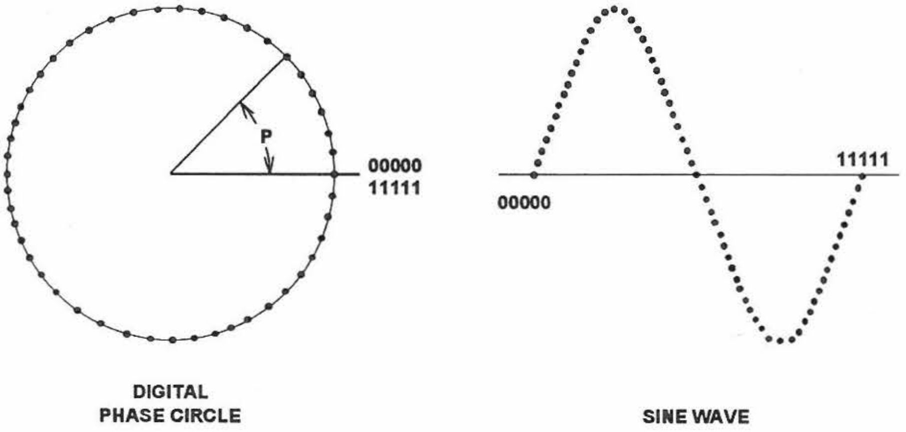


Fig 3.8: Phase to amplitude conversion.

Each phase point around the circle can be given a digital number or "address". The address represents the value in amplitude of that specific phase point. A complete rotation of the wheel represents one cycle of the oscillator wave form.

The rotation speed corresponds to the frequency. The output wave form depends on the amplitude values contained in the phase addresses. In Figure 3.8 a sine wave is shown made up of discrete phase points starting at 00000 and ending at 11111.

Each binary number, or address, points to the amplitude value of the phase point being represented. If more bits are used to indicate the address points, more points can be represented around the circle. If **N** bits are used, then 2^N points can be realised. If a larger **N** value is used, there will be smaller amplitude steps between samples and the resolution of the converted digital-to-analogue signal will increase.

If each rotation of the wheel in Figure 3.8 represents one cycle of the oscillator wave form, there are two ways to alter the frequency. The amount of rotations per time period can be changed or, if the rotation speed is fixed, the only way to alter the frequency is to skip phase points on the circle which will result in lower resolution of the signal.

The angle P in Figure 3.8 indicates a jump between phase points. If the jump between phase points is increased, a higher frequency wave form will be represented. If the amount of jump between phase points is held constant during a rotation, a sine wave is represented.

In a DDS system the amplitude samples are contained in a lookup table. The samples are loaded in the lookup table in such a way that, when properly addressed, it will provide a DAC with the necessary data to produce a wave form.

Figure 3.9. shows a wave form represented by digital amplitude values at discrete phase points. The binary DAC value represents the amplitude value at each phase point and is the amplitude value which is stored as a digital number in the lookup table.

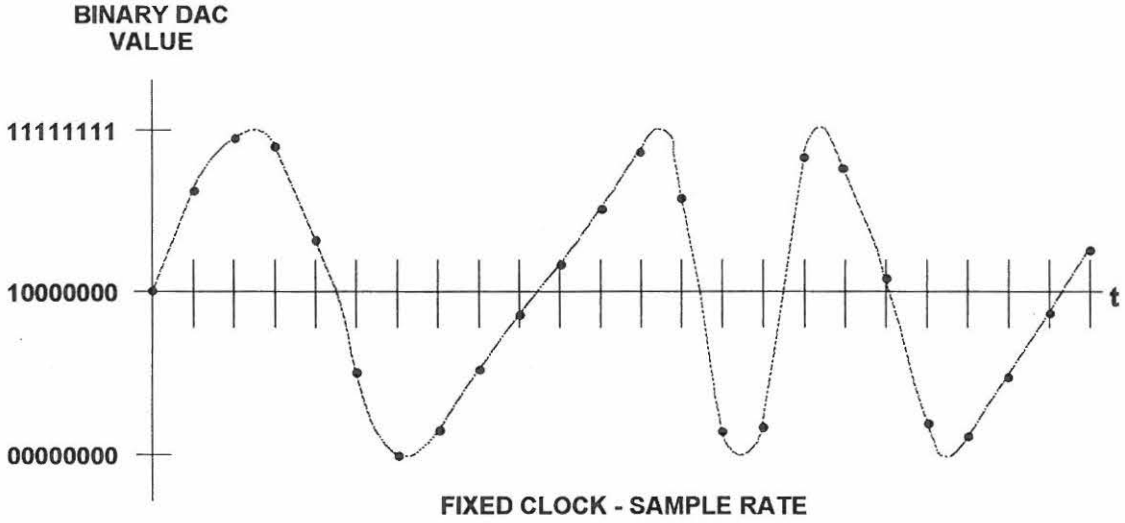


Fig 3.9: A typical wave form represented by binary amplitude values at discrete phase points.

In the above figure it is shown that where the phase changes little between samples, a low frequency wave form is generated, but when the change in phase between samples is large, a higher frequency wave form is generated.

In a DDS system a frequency word is programmed into the phase accumulator. The phase accumulator then generates the appropriate change in phase per time unit sequence. This establishes how far through the cycle each amplitude sample occurs.

As these amplitude values are addressed by the phase progression, the digital amplitude values are converted to analogue values by the DAC and the analogue signal is "constructed" from this.

3.6 The Phase Accumulator

A phase accumulator, as utilised in a DDS, is a variable modulus counter designed to add a single number to the value already in the accumulator every time that the circuit receives a clock pulse. The accumulator defines the instantaneous phase point on the phase circle shown in Figure 3.8. If the counter overflows, another cycle around the digital phase circle is started.

Figure 3.10 shows a more detailed block diagram of a phase accumulator. The basic accumulator consists of a frequency setting register, a digital adder and a latch. The frequency setting word (FSW) is an input to the digital adder. Usually this is a 32 bit word that can be loaded either in serial or parallel, depending on the FSW register of the DDS ASIC being used. The clock input to the latch is a fixed-frequency clock.

The operation of the phase accumulator is as follows: At the start of the sequence the FSW is loaded into the adder at A. The B input from the latch is zero. At the first clock pulse, the output of the adder is written into the latch. The output of the latch is immediately available at the B input of the adder and a number of the most significant bits, depending on the type of read-only memory (ROM), are applied to the address lines of the ROM containing the sine lookup table. At the next clock pulse, the Σ value of the adder, which now contains two times the FSW value (A+B), is again written into the latch. The new output value of the latch is again fed back to the adder. The address information to the ROM is also updated. This feedback loop will continue until the value in the latch overflows the amount that can be stored in it. At an overflow condition, the latch will contain the overflow value from where adding of the FSW will continue.

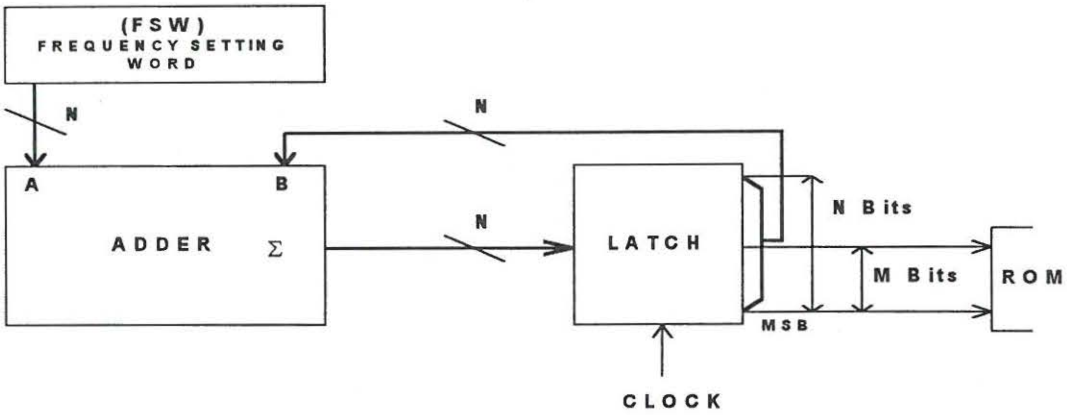


Fig 3.10: Block diagram of a typical phase accumulator.

N indicates all the buses that typically contain a 32 bit word. M identifies the number of address bits used by the ROM. If the ROM has 16 address lines, it would use the 16 most significant bits of the 32 available on the latch output.

The phase accumulator in its simplest form can be compared to a binary variable-modulus counter. A 32 bit counter can have 2^{32} possible states.

The FSW sets the counter modulus and represents the step size. The step size tells the accumulator how many points on the phase circle should be skipped before it addresses the next value. The output of the counter represents the instantaneous phase value of the desired output signal.

Note that the FSW output represents a phase increment, not a frequency value. The required FSW is calculated from the required frequency with regard to the clock frequency and the accumulator length. The output frequency is given by the rate of change in phase per time unit (expression 3.1).

To calculate the value of the FSW in a 32 bit system, the following formula is used:

$$FSW(N) = \frac{f_{out} * 2^{32}}{f_{clk}} \qquad 3.2$$

The $FSW(N)$ is a 32 bit binary word. f_{out} Is the desired frequency value and f_{clk} is the fixed clock value.

The frequency resolution, or tuning step size, of a DDS system is given by:

$$f(step) = \frac{f(clock)}{2^N} \qquad 3.3$$

N is the number of bits in the phase-accumulator counter. If a clock frequency of 20 MHz is taken with an accumulator length of 24 bits, a frequency resolution of 1.2 Hz can be obtained. If the accumulator length is increased to 32 bits, the resolution improves to less than 5 nanohertz.

3.7 The Lookup table

Phase is a linear time function for a given frequency, but the output wave form amplitude is a sinusoid function. There is a constant relationship between phase and amplitude values, and digital sine wave synthesisers make use of this fact. The lookup table is a digital-phase-to-amplitude converter usually configured from a read-only memory or ROM (Zavrel, 1993: 10-19).

The output of the phase accumulator is applied to the ROM address bus. The ROM data bus provides a digital-to-analogue converter with the binary equivalent of the corresponding amplitude value. The ROM can contain the full 360 degrees of the sine wave, but ROM size is reduced by including only 180 or 90 degrees of sine or cosine data. The phase accumulator then runs in both directions through the ROM and the output sign is set accordingly. Figure 3.11 shows the relationship between the address bus and the data bus values in the ROM, containing a table of 90° of cosine data. The phase accumulator skips up and down the cosine values to provide the proper amplitude value to the DAC for each discrete time value.

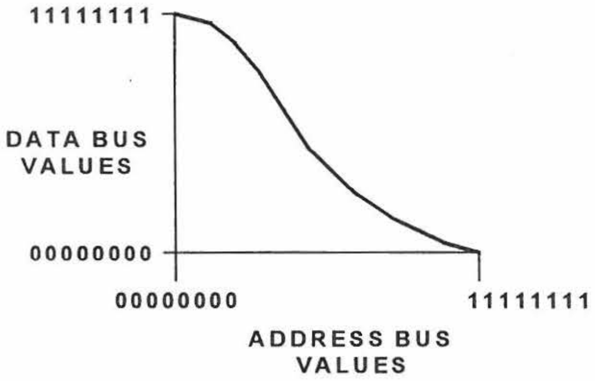


Fig 3.11: Graphic display of the contents of a cosine lookup table.

3.8 The Digital-to-analogue Converter (DAC)

The DAC accepts parallel digital data from the lookup table and produces an analogue output signal which is related to the digital code at its input. The DAC is an essential part of the DDS system, and it is also the major contributor to in-band spurious signals. To minimise these spurious signals, a DAC with good integral and differential non-linearity specifications must be used (Gallant, 1989: 100).

The spectral purity of a DDS system is largely dependent upon the input and output of the lookup table. The input to the lookup table consists of a number of the most significant bits of the phase accumulator. The limit in addressing capability of the lookup table causes phase truncation. In most commercially available DDS ASICs this is internal to the chip and cannot be externally influenced. The spurs caused by phase truncation from 32 bits to 16 bits are usually below -76 dBc (Qualcomm Q2334, 1993: 10).

The output of the lookup table usually consists of an 8 or 12 bit wide data bus. On most DDS ASICs, this bus is available to the user and the selection of a suitable DAC has a large influence on the quality of the converted sine wave. An ideal sine or cosine representation would obviously require an infinite number of bits for each amplitude value in the lookup table. Amplitude truncation, better known as quantization error (Fig. 3.6), occurs as a result of the limitation in the number of bits that can be applied to the DAC.

3.8.1 Integral Non-linearity

Integral non-linearity is the overall variation of the DAC transfer function from that of an ideal DAC which would represent a straight line as shown in Figure 3.12 (a). Since integral non-linearity tends to produce a smooth curve, the primary spur contribution is output signal harmonics (Zavrel & Mc Cune, 1988: 3).

3.8.2 Differential Non-linearity

Differential non-linearity is a small-scale measurement of the transfer function variations from a straight line as shown in Figure 3.12(b). It is in essence a bit-by-bit output linearity measurement. Differential non-linearity's by nature are very fast moving with respect to the primary output signal.

These rapid variations are high-order non-linear terms, and therefore manifest themselves as cross modulations and high order harmonics. Cross modulations may have sign relationships with the primary output signal, and may include "cross-over" signals. A cross-over is where a spurious signal moves in the opposite direction than the primary output and at some point crosses over, or shares, the same frequency (Zavrel & Mc Cune, 1988: 3).

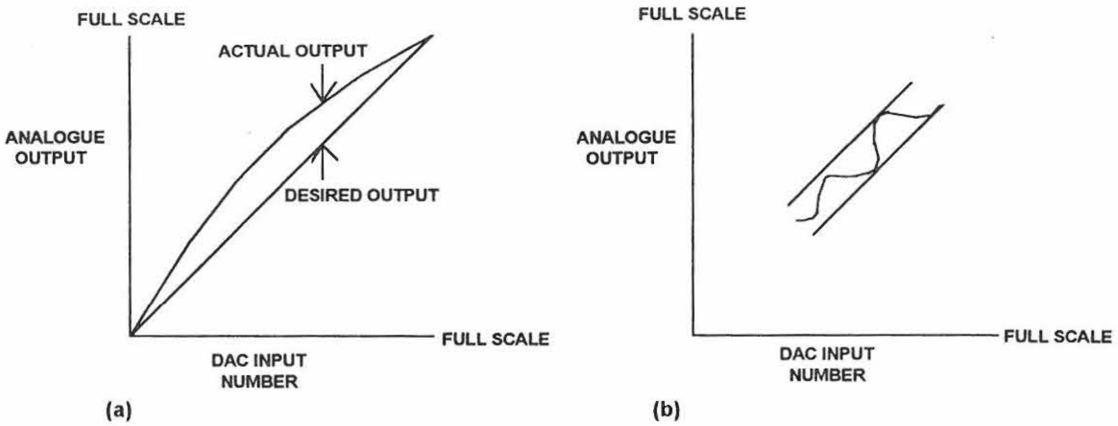


Fig 3.12: A graphic display of (a) Integral non-linearity and (b) Differential non-linearity of a typical DAC transfer function.

3.8.3 Time Skew

Gallant (1989: 100) describes time skew as another contributor to in-band spurious signals related to the digital-to-analogue conversion process. Time skew results when the arrival times of the input bits to the DAC differ by a few nanoseconds. If this skew is substantial compared with the settling time of the DAC, the output of the DAC will respond to the changing input data, while it is varying until it stabilises. This extra generated energy in the output signal will occur as non-harmonic spurs.

A cure for this problem is latching the data before applying it to the DAC. Different latching methods are in use. Some DACs use a level-activated latch where a voltage level, applied to the latching pin, controls the latch register. The DAC then operates in transparent mode while this signal is valid. A different concept uses edge triggering on the latch control. This signal is usually derived from the system clock or from an external signal depending on the application.

3.8.4 Glitch Energy

Glitch energy is not an energy measurement as the name implies. This term refers to the "spike" observed on the output of the DAC output while a transition settles. The unit of measure is volt-seconds. It is a measure of impulse area rather than energy (Gallant, 1989: 102).

Two factors contribute to the forming of the "glitch pulse". Time skew is the first one. The varying output amplitude due to time skew will have a similar characteristic to the glitch pulse. Another factor well-known in phase-locked loop designs is the analogue settling time of a linear system. At this point, on the output of the DAC, the DDS system is no longer digital, but analogue and linear. The reaction of this linear system to the step input provided by the DAC input, is some type of damped sinusoid. A wider bandwidth, as well as a DAC with no time skew, will result in a shorter settling time (Zavrel & Mc Cune, 1988: 4).

3.8.5 Output Stage Balance

Output stage balance relates to the symmetry of all equally sized and oppositely signed transitions. For a transition from one output level to another, the return transition should be its mirror image. An imbalance will produce even order harmonics of the primary output signal with amplitudes as high as -25 dBc (Zavrel & Mc Cune, 1988: 4).

3.9 Low-pass Anti-alias Filter

Much of the development work done in DDS design involves the reducing of spurious levels, which represents the most important limiting specification in DDS systems. Many of the limitations are due to component restraints and can only be cured by improving on the design of the individual components, e.g. the DAC.

Alias signals are produced as a natural consequence of the sampling process. The exact frequency and amplitude of each alias signal is predictable. Figure 3.13 shows the frequency spectrum of a typical DDS generated signal, including the associated alias signals that will appear on the output. This example was taken from the Stanford Telecom Application Note 102, (page vii).

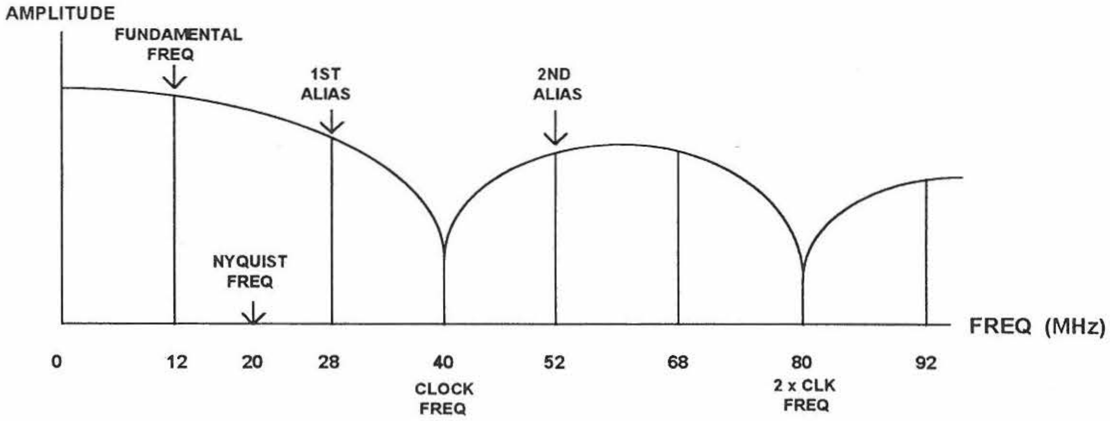


Fig 3.13: Spectrum of a DDS output with the fundamental frequency at 12 MHz and a 40 MHz clock frequency.

Pairs of alias signals appear equally spaced from the clock and clock harmonic frequencies. In the example shown in Fig 3.13, a 12 MHz signal is being synthesised from a 40 MHz clock. The first pair of alias signals appear 12 MHz above and below the clock frequency. As the operating frequency is increased towards the Nyquist frequency, the first alias drops towards the Nyquist frequency. As explained in paragraph 3.3, the alias signals need to be eliminated from the output signal to ensure a clean spectrum with only the desired fundamental frequency present. It is obvious that if the fundamental frequency approaches the Nyquist limit, it will become extremely difficult to separate the alias signals from the fundamental.

Passive or active filters can be used as alias filters. In narrow-band DDS applications the alias signals can also be effectively eliminated, using notch filters or a band-pass filter around the fundamental. Depending on the relevant frequencies, a three-pole passive low-pass filter is usually sufficient to suppress alias signals to an acceptable level (Stanford AN 102, 1990: 44).

3.10 Numeric Modulation in DDS Systems

Inherent to the technique of direct digital synthesis is the possibility of numeric modulation of all three wave form parameters, namely phase, frequency and amplitude. The output wave form can be defined as A Sin(ωt+ϕ). Figure 3.14 shows a block diagram of a basic DDS system with all three possible modulators in place. All known modulation techniques use one, two or all three basic modulation types simultaneously. With DDS, any known wave form can be synthesised within the Nyquist bandwidth limitations.

Pairs of alias signals appear equally spaced from the clock and clock harmonic frequencies. In the example shown in Fig 3.13, a 12 MHz signal is being synthesised from a 40 MHz clock. The first pair of alias signals appear 12 MHz above and below the clock frequency. As the operating frequency is increased towards the Nyquist frequency, the first alias drops towards the Nyquist frequency. As explained in paragraph 3.3, the alias signals need to be eliminated from the output signal to ensure a clean spectrum with only the desired fundamental frequency present. It is obvious that if the fundamental frequency approaches the Nyquist limit, it will become extremely difficult to separate the alias signals from the fundamental.

Passive or active filters can be used as alias filters. In narrow-band DDS applications the alias signals can also be effectively eliminated, using notch filters or a band-pass filter around the fundamental. Depending on the relevant frequencies, a three-pole passive low-pass filter is usually sufficient to suppress alias signals to an acceptable level (Stanford AN 102, 1990: 44).

3.10 Numeric Modulation in DDS Systems

Inherent to the technique of direct digital synthesis is the possibility of numeric modulation of all three wave form parameters, namely phase, frequency and amplitude. The output wave form can be defined as A Sin(ωt+ϕ). Figure 3.14 shows a block diagram of a basic DDS system with all three possible modulators in place. All known modulation techniques use one, two or all three basic modulation types simultaneously. With DDS, any known wave form can be synthesised within the Nyquist bandwidth limitations.

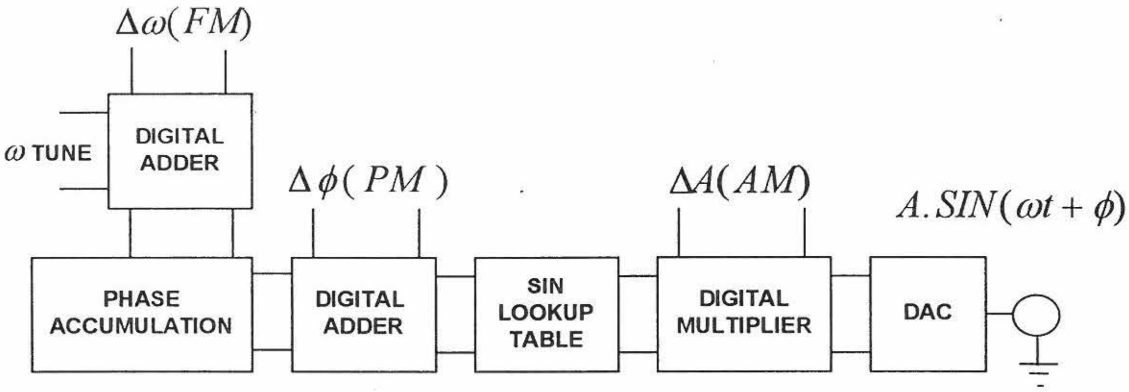


Fig 3.14: Comprehensive modulation capabilities of a DDS synthesiser.

3.10.1 Numeric Frequency Modulation

In analogue circuits, FM is accomplished by using reactance modulation. Typically the modulating analogue signal is applied to a varicap diode which forms part of the resonant oscillator circuit. Unfortunately the relationship between modulating voltage and oscillator frequency is non-linear. Great effort is undertaken in analogue FM modulation to linearize this voltage versus frequency curve. It is also difficult to prevent over-modulation and to maintain constant deviations over a wide frequency range.

All the above problems are solved with DDS numeric FM. FM can be realised by changing the modulus of the phase accumulator in accordance with a digitised modulating signal. An FM digital port is easily implemented by placing an adder with the frequency tuning port. The instantaneous amplitude value of the modulating frequency is added to the ωTUNE value, which contains the fundamental or desired carrier frequency tuning word.

With numeric modulation, discrete units of frequency are added linearly rather than modulating a reactance. Therefore the linearity in a DDS system is much better than in an analogue reactance modulator. Over-modulation is eliminated because you cannot overshoot a DAC in the frequency domain (Zavrel & Edwards, 1990:11). Consistent deviation over the entire operating range is realised because of the algebraic nature of the modulator.

The trade-off in numeric FM is the generation of alias signals in the modulating signal. The modulation signal must also be transformed to a digital signal if it is in analogue form. This transformation, done with an analogue-to-digital converter (ADC), generates alias signals. If the sampling rate of the modulation input is close to the DDS system clock, unwanted signals will appear close to the main signal. The alias signals are eliminated in practice by employing low-pass or band-pass filters at the modulation input. Careful analysis of the nature of these alias signals is necessary when designing digital FM systems.

3.10.2 Frequency Shift Keying

In a simple analogue FSK system, the frequency change is implemented by switching between two pre-set oscillators that modulate a carrier. In a DDS system, FSK is realised by simply keying the appropriate tuning word bits with the FSK binary signal and returning to the first frequency. Thus $\omega 1$ and $\omega 2$ are keyed with a logic 0 and 1. Another method is to sequence two discrete frequency values. This is a bit more complex, because a new frequency must be loaded every time that the logic state changes. This is not a problem for DDS, due to fast switching and settling times. Most DDS ASICs have this method incorporated into their design. Two registers are loaded with their respective frequencies and switching between them is done by changing the logic state of one of the pins on the IC.

3.10.3 Numeric Phase Modulation

A simple analogue phase modulation system would make use of delay lines switched in and out of the modulating signal or carrier path. In DDS, phase modulation can be accomplished by placing a digital adder at the output of the phase accumulator to advance the phase word. A typical system would allow for 12 bits phase modulation. If the DDS is followed by analogue circuits that produce undesirable phase shifts, phase error correction can be accomplished digitally before the signal passes through them.

3.10.4 Numeric Amplitude Modulation

In an analogue system, AM is accomplished by multiplying two frequencies, the modulating signal and the carrier. This produces a composite signal with the amplitude of the carrier varying in accordance with the modulating signal. Numeric AM can be realised by placing a digital multiplier between the lookup table and the DAC. A single quadrant multiplier will produce a full-carrier double sideband signal. A four-quadrant multiplier will produce a suppressed-carrier double sideband signal (Zavrel & Edwards, 1990: 13).

Amplitude modulation as well as PM and FM can also be implemented on the output of the DDS system using conventional techniques.

respective frequencies and switching between them is done by changing the logic state of one of the pins on the IC.

3.10.3 Numeric Phase Modulation

A simple analogue phase modulation system would make use of delay lines switched in and out of the modulating signal or carrier path. In DDS, phase modulation can be accomplished by placing a digital adder at the output of the phase accumulator to advance the phase word. A typical system would allow for 12 bits phase modulation. If the DDS is followed by analogue circuits that produce undesirable phase shifts, phase error correction can be accomplished digitally before the signal passes through them.

3.10.4 Numeric Amplitude Modulation

In an analogue system, AM is accomplished by multiplying two frequencies, the modulating signal and the carrier. This produces a composite signal with the amplitude of the carrier varying in accordance with the modulating signal. Numeric AM can be realised by placing a digital multiplier between the lookup table and the DAC. A single quadrant multiplier will produce a full-carrier double sideband signal. A four-quadrant multiplier will produce a suppressed-carrier double sideband signal (Zavrel & Edwards, 1990: 13).

Amplitude modulation as well as PM and FM can also be implemented on the output of the DDS system using conventional techniques.

3.11 Hybrid DDS/PLL Synthesisers

High performance in general coverage High Frequency (HF) receivers requires a 30 MHz receive range, fine resolution and outstanding spectral purity in the local oscillator. The combination of fine resolution and fast settling time of DDS and the broadband characteristics of PLLs make a DDS/PLL hybrid synthesiser a viable alternative to conventional HF synthesiser design.

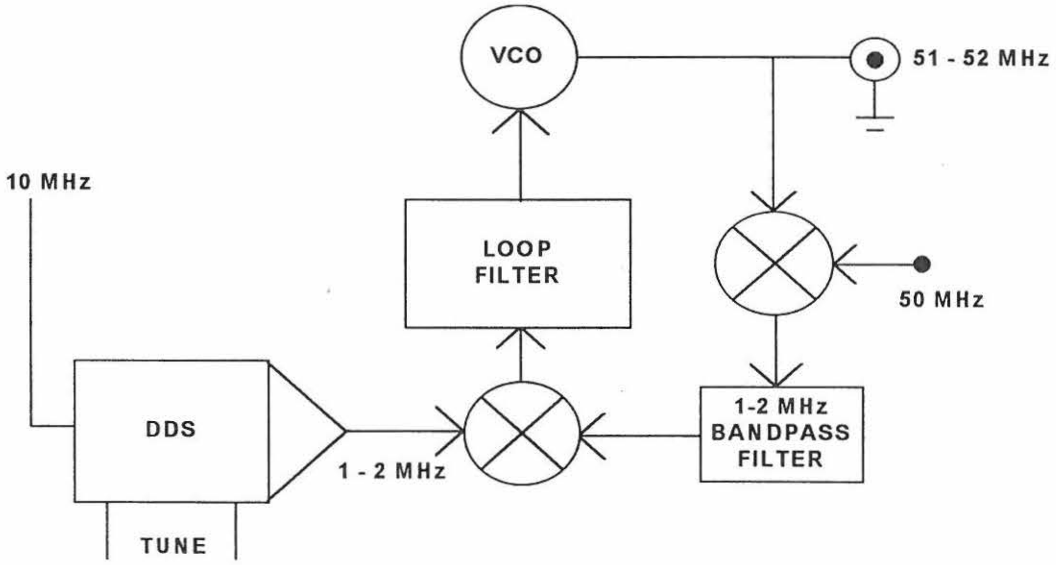


Fig 3.15: Simplified block diagram of a typical DDS/PLL combination synthesiser.

Figure 3.15 shows a block diagram of a simple up-conversion DDS/PLL Synthesiser. Typical values have been chosen to assist in the explanation. The operation of the loop is straightforward. A relatively low frequency signal (1-2 MHz) is applied to the PLL phase detector. A clock frequency of 10 MHz will allow low-power low-cost components to be used in the DDS portion of the design. The low DDS output frequency, with the 10 MHz clock, permits excellent spurious response. The output of the VCO represents a signal from 51-52 MHz adjustable in steps determined by the DDS resolution. In the feedback path, to the second input of the phase detector, a mixer is utilised in place of a conventional divider. A divider in this case is not necessary, because the reference frequency of the PLL, namely the DDS, provides the variation in phase needed to effect a change in the output frequency of the loop.

A combination system like that in Figure 3.15 would allow the use of DDS technology, providing excellent resolution, at frequencies well above the limits of a single DDS system. The PLL is used to cover coarse steps and the DDS is used in place of the divider to select finer frequency steps. Another advantage of such a system is the reducing of phase noise being caused by the large division ratios in the feedback loops of PLLs and multi-loop PLLs. The 10 MHz clock to the DDS and the 50 MHz input to the mixer in the PLL feedback path must however be from the same source to limit phase noise and to ensure phase coherency.

3.12 Summary - DDS Theory

This chapter covered some of the theory related to the digital sampling and storage of analogue signals and representing it again. This, along with the theory of direct digital synthesis was included to provide the reader with sufficient background to this method of generating frequencies.

The project under discussion makes use of the direct digital synthesis technique in order to generate frequencies within a certain spectrum. Much of the theory covered, e.g. modulating a DDS system, was not practically implemented in this specific project. It was only included to show the vast capabilities of this method of frequency synthesis.

CHAPTER 4 - HARDWARE DEVELOPMENT

4.1 Introduction

The hardware of this system can be divided into five separate modules. Although the interaction between the modules is vital for the performance of the final product, they were developed and evaluated individually.

The DDS signal generator is made up of a controlling module, a direct digital synthesiser (DDS), a digital-to-analogue converter (DAC), an anti-alias filter and a power supply.

Each module will be discussed separately, highlighting the design specifications, choice of components, interface signals and evaluation of the individual modules. Although the hardware and software are covered in different chapters, reference to the software will be made to assist in describing the hardware development process.

4.2 Modularity

A modular design approach assisted the project development in several ways. Firstly, each module could be evaluated separately before going on to the next module or integrating it into the rest of the system. Secondly, evaluating progress of the complete project was made easier. Thirdly, it allowed for the changing and upgrading of modules without disrupting the complete project.

Due to the nature of this project it was not necessary to develop any additional test- or simulation equipment to evaluate the separate modules. Interfacing between the controlling module, the DDS chip and the DAC consisted mainly of digital signals under software control. Where the speed of data transfer was too high to accurately measure transitions with a voltmeter, an adjustment to the software was made to slow down the transfer process for evaluation purposes.

4.3 Basic Design Specifications

Before describing the development of the modules, it is necessary to highlight the basic requirements that had to be met in the design of this signal generator.

- The signal generator had to make use of direct digital synthesis technology.
- Microprocessor control over input and output functions were required. This included keypad entry of frequency data and display on a Liquid Crystal Display (LCD) module.
- Apart from the keypad/LCD user interface, a series connection to a personal computer was also required to monitor and control certain functions.
- A continuously variable frequency range from 1 Hz to 10 MHz with a frequency resolution of 1 Hz and a fixed amplitude had to be covered.
- Up-and-down frequency control by means of a rotary optical encoder (ROE) interfaced with the microcontroller.

On completion of the hardware and software development, a thorough evaluation of the end product had to take place in order to compile a detailed specification sheet.

During the hardware development phase, evaluation of the modules took place ensuring that the basic design specifications were met and to compare certain components in order to achieve the best results possible.

The signal generator unit was built on a 233 mm by 160 mm square pad vero board. Wire wrapping was used to do the interconnecting between components and modules.

4.4 Development

Partial development of the intelligent part of the unit, the Controlling Module, was done first to provide a user-interface to the rest of the hardware of the signal generator. Development of this module consisted of interfacing a microcontroller with a Liquid Crystal Display and a 4 x 4 matrix keypad.

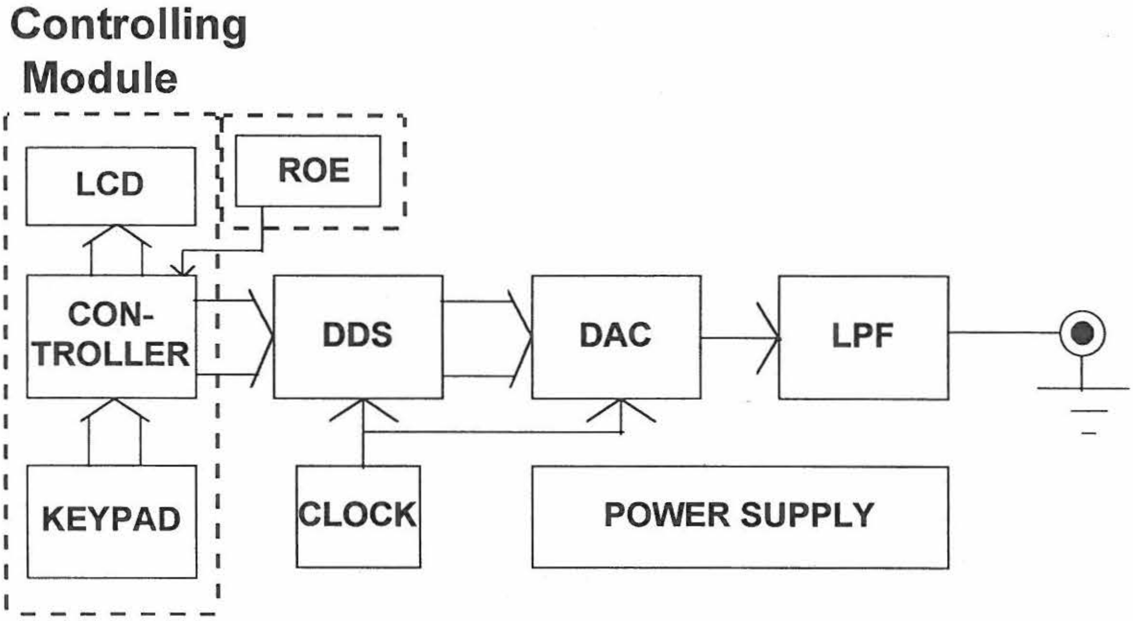


Fig 4.1. Block diagram of the DDS-based signal generator.

The Controlling Module, consisting of the LCD, microcontroller, keypad and the rotary optical encoder (ROE), can be seen on the left of Figure 4.1.

Following the Controlling Module, a DDS chip was added as the second module. A 30 MHz crystal clock with a transistor-transistor-logic (TTL) output, forms part of this module. With the interfacing to the DDS chip completed, evaluation of two different 12 bit DACs took place to decide which one best suited the purpose of the project.

The final module incorporated into the system was an anti-alias filter. The basic design specification of this filter was to attenuate the unwanted alias signals above the maximum operating frequency and to provide as little attenuation as possible to the pass band frequencies.

The power supply module design changed throughout the project as different voltages were required by the different DACs that were evaluated. The final design could therefore only be done after completion of the DAC evaluation.

4.5 Controlling Module

The controlling module had to be able to accept an input from a matrix keypad, process the information, display it on an LCD and provide the necessary control signals for the rest of the circuitry.

Another important requirement was the ability to quickly incorporate software updates from a PC to the microcontroller for diagnostic capabilities, e.g. displaying control signals on the PC screen without affecting the circuit operation.

A description of each of the main components making up this module will be given before describing the interfacing between them.

4.5.1 Microcontroller

The Soft Microcontroller ™ range built by Dallas provided the best options for this specific application. The product chosen for this project is the DS5000T 32-16 Soft Microcontroller. It is a 40 pin DIP device containing an 8051 8-bit processor, 32K non-volatile RAM and 4 configurable IO ports. The RAM can be configured in multiples of 2K to function as programme resident memory or normal RAM. The device runs on a clock speed of up to 16 MHz. The IO ports are also pin-addressable by software which simplified control signals required for the rest of the circuitry.

Loading of the software takes place through the RS232 serial connection. The non-volatile RAM facilitates easy programme changes and loading. An internal lithium battery makes the device capable of retaining programme data for more than ten years without any external power connection. Other features of the device not used in this design is a real time clock (RTC) and two independent timers, T0 and T1 (Dallas Semiconductor, 1992: 2).

4.5.2 Liquid Crystal Display (LCD)

A main component of the user interface is an LCD display that gives visual indication of the signal generator status. The most important requirements for this unit was simplicity (a single unit), 8 bit micro-compatibility, small size and it had to accept ASCII characters.

The LCD display chosen for this project was the OPTREX 4-line, 20-character display. This unit makes use of the HD44780 LSI controller, driving the segment driver and LCD, to simplify programming operation from an external 8-bit bus interface. It has 14 connecting pins of which 8 are used as the bus interface, 3 are used as external control lines, 2 are used as the supply inputs and the last one is connected to a voltage divider circuit to adjust the display angle.

The unit accepts the standard ASCII character set which simplified programming from the DS5000T microcontroller.

4.5.3 Matrix Keypad

The 16-key matrix keypad forms part of the user interface. The keypad, combined with a 74922 4x4 matrix-to-BCD encoder, is connected to one of the ports on the DS5000T. The output of the 74922 is in Binary-coded decimal (BCD) format. Two pins on the 74922 allow the chip to be used in a tri-state configuration. As soon as any key is pressed, a Data Available pin is activated which indicates to the processor that a key has been pressed. A Data Enable pin, controlled from the DS5000T, allows the BCD value of the key pressed out on the data bus.

Another option of keypad interfacing is connecting the keypad directly to one of the DS5000T ports. This would however complicate the software and take up a lot of processor time. The 74922 allows for debouncing of the keypad as well.

4.5.4 Rotary Optical Encoder

The Rotary optical encoder (ROE) is a Bourns 128 step-per-rotation encoder used for rotary up-and-down frequency control of the signal generator. Decoding of the outputs is done with a 74LS74 D-latch for up-and-down frequency control. A monostable multivibrator (555 circuit) with a 1 second time constant is used to indicate if the ROE is active. Both digital signals are fed into the DS5000 controller from where they are accessed by the software routines.

4.5.5 Controlling Module - Hardware interfacing

The schematic diagram of the DS5000T Control Module is included in Appendix A1. All the digital signals are TTL levels. The relevant interface signals are covered in the software chapter.

The diagram shows the 8-bit bus of the LCD connected to Port 0 of the DS5000T. The 100K (R8P9) resistor network connected to this bus is due to Port 0 not having any internal pull-up resistors. The LCD control lines are connected to Port 3. The Read/Write select pin is connected to Port 3 pin 5, the Register Select line to pin 6 and the Enable signal to pin 4. The VO pin on the LCD adjusts the contrast of the display. This pin is connected to a 10K variable resistor between Vcc and ground.

The data lines of the 74922 keypad encoder are connected to Port 1 of the DS5000T. The control lines, Data Available and Output Enable, are connected to pins 6 and 7 of Port 2. Only a 4-digit binary code is needed to identify the 16 different keystrokes. Therefore only 4 bits of Port 1 is used for the 74922 data lines.

The ROE circuit is connected to pin 2 and 3 of port 3. Interface signals leaving this module are one RS 232 serial port connection and four control signals to the DDS module.

4.6 The DDS Module

The task of the DDS Module is to convert the frequency word from the control module into digital amplitude signals which are then fed to the digital-to-analogue converter. The circuit diagram of the DDS Module is shown in Appendix A2.

4.6.1 Harris HSP45102 DDS

The HSP45102 DDS from Harris was chosen for its relatively low cost, availability and the serial loading capability of the frequency data. The chip consists of a 32-bit phase accumulator, a phase offset adder and a Sine ROM. The 12-bit output from the Sine ROM will, in theoretical terms, achieve a worse case spur level of less than -69 dBc.

4.6.2 Functional description of the HSP45102 DDS

This chip produces a 12-bit sinusoid whose frequency and phase are digitally controlled. The frequency of the sine wave is determined by selecting one of two 32-bit words contained in a 64-bit shift register. These two words are called the LSB and the MSB words. Selection of the active word can be achieved by controlling the SEL_L/M signal. The phase of the output is controlled by the two-bit input P0 - 1, which is used to select a phase offset of 0º, 90º, 180º or 270º (Harris, HSP45102: 3).

The HSP45102 consists of a frequency control section, a phase accumulator, a phase offset adder and a sine ROM. The frequency control section serially loads the frequency control word into the frequency register. The phase accumulator and phase offset adder compute the phase angle using the frequency control word and the two phase modulation inputs. The sine ROM generates the sine of the computed phase angle (Harris, HSP45102: 1).

4.6.2.1 Frequency Control Section

The frequency control section serially loads the frequency data into a 64-bit, bi-directional shift register. The shift direction is selected with the MSB/LSB input. This is a software decision and does not affect the operation of the device. The register shifts on the rising edge of SCLK when SFTEN is low.

The 64 bits of the frequency register are sent to the Phase Accumulator Section where 32 bits are selected to control the frequency of the sinusoidal output.

4.6.2.2 Phase Accumulator Section

The phase accumulator and phase offset adder compute the phase of the sine wave from the frequency control word and the phase modulation bits P0-1. The most significant 13 bits of the 32-bit phase accumulator are summed with the two-bit phase offset to generate the 13-bit phase input to the sine ROM.

The phase accumulator advances the phase by the amount programmed into the frequency control register. The output frequency is equal to:

$$f_o = \mathrm{N^*Fclk}/2^{32} \qquad 4.1$$

where N is the value in the selected 32 bits of the frequency control word and Fclk is the clock frequency.

The TXFR control line clocks the selected 32 bits into the phase accumulator's input register. At each clock, the content of this register is summed with the content of the accumulator to step to the new phase. The phase accumulator stepping may be inhibited by holding ENPHAC high. The LOAD signal zeroes the feedback to the phase accumulator.

4.6.3 Clock

The DDS chip allows clocking speeds of up to 30 MHz. In this design a Jauch 30 MHz TTL oscillator is used. It has a frequency accuracy of 100 parts per million (ppm) and a 10 TTL fanout (Jauch Datasheet).

The clock signal is applied to the DDS chip as well as the DAC. The clock signal can be applied to the DAC in any of several ways to obtain the best performance. This will be covered in the following description of the DAC.

4.7 The Digital-to-Analogue Converter Module

Two different DACs were used in the development of this module. Refer to Appendix A3 for the schematic diagram of this module incorporating the Harris DAC. Appendix A4 shows the module with the Analog Devices DAC.

As explained in paragraph 3.8, this module plays a fundamental part in the performance of the system regarding spectral purity. It was therefore necessary to evaluate different DACs to choose the one best suited for this project. In this project two DACs, namely the HI565A from Harris and the Analogue Devices AD9713B, were evaluated.

Both DACs tested are 12-bit devices. The AD9713B, however, has a latch on the data input which means that the DAC is transparent only when the latch enable signal is low (Analog Devices, AD9713B: 3). This is a useful feature in a system where time skew can be a problem. (See paragraph 3.8.3)

4.7.1 DAC Evaluation

To be able to make a comparison between the two DACs, tests were done at specific frequencies. First a plot was made with the DDS set at 25% of the clock frequency. At integer divisions of the clock frequency many of the spurs fall directly on the main tuning frequency. Any of these integer divisions should theoretically give the best obtainable performance for the DAC under test. In this case, 25% of 30 MHz is equal to 7.5 MHz.

Another plot at 7.6 MHz was done to show the spurs close to the fundamental signal. A third plot to examine the generation of harmonic signals was done by setting the DDS to 1 MHz. With the spectrum analyser set to a span of 5 MHz, the presence of any harmonic signals are clearly visible.

The initial DAC evaluation was done without the anti-alias filter on the output. The stop band of the filter does not produce a flat response and would have made a proper comparison between the components impossible.

4.7.1.1 Evaluating the HI565A DAC

The first DAC incorporated into the system was the HI565A. This device operates permanently in the transparent mode (Harris, HI565A: 8-42). Clocking the DDS at 30 MHz, several frequency spectrum plots were made to assess the performance.

Frequency plot 1.1 in Appendix B shows the 7.5 MHz fundamental signal at

-47.7 dBm and the first harmonic, 15 MHz, at almost the same level. The worst case spur over the operating bandwidth was measured to be -30 dBc. The noise floor at its lowest point was measured at -55 dBc. Comparing this plot to previous research plots showed that this output signal has many undesirable properties regarding spectral purity.

Shifting the output frequency to 7.6 MHz in order to examine the spurs, showed a marked increase in the average noise floor and the worst case spurs were measured to be in the order of -10 to 20 dBc (Appendix B1.2).

To evaluate the DAC with regard to the generation of harmonic signals, a frequency of 1 MHz was generated and viewed with a span of 5 MHz. Appendix B1.3 shows that the first few even and odd harmonics are all within 10 dBm of the fundamental signal. According to Zavrel & Mc Cune (1988: 3), this type of harmonic distortion is mainly caused by output stage balance and integral non-linearity. Using a unipolar or bipolar voltage approach on this DAC did not solve the problem.

As a result of the bad test results, it was decided to reduce the clocking speed of the DAC. Clocking the DDS at 10 MHz, with an output frequency of 2.5 MHz, a marked improvement in the performance of the system regarding the average noise floor and the spurs in the operating bandwidth was measured (Appendix B2.1). The first harmonic in this case was 15 dB down, a definite improvement.

Appendix B2.2 at 2.6 MHz shows that the worst case spurs are 20 dB down from the fundamental. There is also a 10 dB improvement on the first plot done with the 30 MHz clock. Appendix B2.3, the 1 MHz evaluation, shows almost no difference to the previous plots.

This is a clear indication that, with a clock frequency of 30 MHz, the DAC would be used beyond its limitations and that it was not suitable for the project. The settling time of 250 ns, and the switching time of 30 ns, as indicated in the specification sheet, proved to be insufficient to be used at a clocking rate of 30 MHz for this type of application (Harris, HI565A: 8-44).

4.7.1.2 Evaluating the AD9713B DAC

With a switching rate of 3 ns and a settling time of 30 ns, this device was expected to be the answer to the problems experienced with the slower DAC.

Using the AD9713B in the transparent mode at a clock speed of 30 MHz shows that, within the operating bandwidth, a worst case spur measurement of -60 dBc is achieved, with the first harmonic being at -55 dBc (Appendix B3.1). To be able to compare the Analog Devices DAC with the Harris DAC, similar plots were also done with the carrier set at 7.6 MHz and 1 MHz. The results are shown in Appendices B3.2 and B3.3. From these plots it is clear that the AD9713B circuit is much more suitable for the intended purpose. With the carrier set at 7.6 MHz, a multitude of alias signals are visible on both sides of the main signal, but the strongest alias is still 62 dB down.

For the above tests on the AD9713B the latch enable signal was kept in the transparent mode to be able to compare it to the HI565As operating mode. To optimise the output of the AD9713B DAC, the clock signal can be applied to the latch enable signal. Manipulation of the clock signal, regarding polarity, phase and duration, as applied to the latch enable signal, could further enhance the output signal purity.

4.7.2 Latch Enable Signal

As mentioned previously, the latch enable signal allows transparent operation of the AD9713B when kept at ground potential. Manipulating this signal with a variation of the clock signal applied to it, should lessen the effects of time skew and bit transitions on the DAC operation.

Except for the transparent mode, the clock signal can be applied directly, or invertedly, to the latch enable signal. Another method would be to narrow the clock pulse. The latch enable operates as a level-triggered input. Whilst ensuring that the data on the input is valid before the DAC changes state, a narrow clock pulse applied to the latch input might improve the overall performance.

4.7.3 Manipulating The Clock Signal

To simplify comparative testing it was necessary to implement clock manipulation circuits on the main board. Being able to switch between them allowed easy comparisons under virtually the same external conditions.

The circuits used to manipulate the clock signal to be inverted and narrowed, are shown in Appendix A5. To invert the clock signal, a NAND gate (74HCT00) was used with the inputs tied together. The output of the NAND gate was then applied to the latch enable signal of the DAC.

Narrowing the clock pulse was done by applying the clock pulse to both legs of an AND gate (74HCT08). One of the inputs was sent through a NAND gate, inverting this input and delaying it by approximately 10 to 15 ns (propagation delay of NAND gate). The output of the AND gate is a short duration logic 1 pulse. The latch enable signal operates on a logic 0 level. Therefore the AND gate output is inverted by another NAND gate. The result is a short duration negative-going clock pulse.

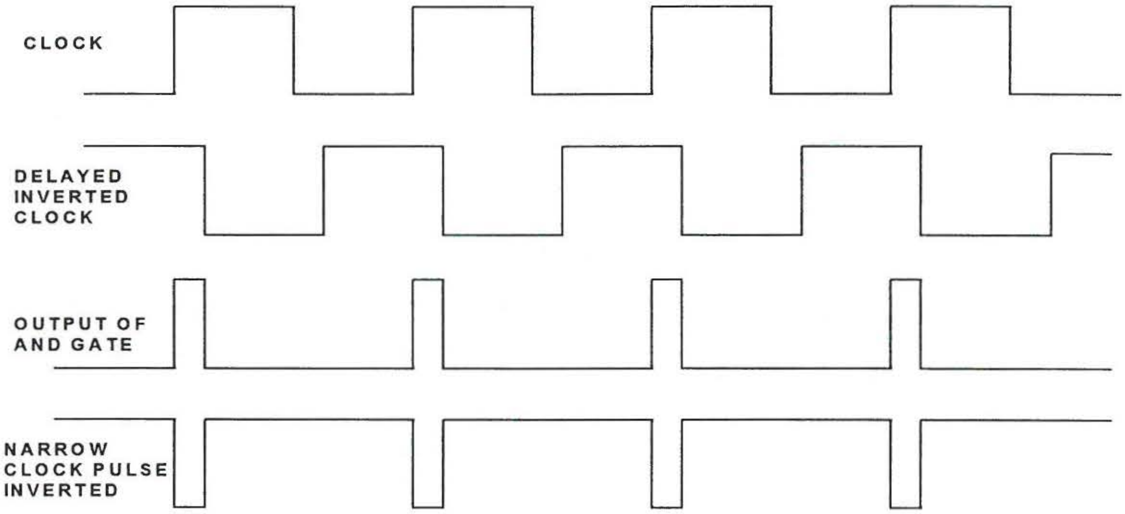


Fig 4.2: Manipulation of the clock pulse

Figure 4.2 shows the clock signal at the top with the delayed inverted clock signal just below. The AND gate produces a high output when both inputs to it are high, producing a narrow positive-going pulse. This output is taken through a NAND gate, inverting the signal to a narrow negative-going pulse which can be applied to the latch enable of the DAC. Although this concept relies on the propagation delay of a certain gate, it worked well, even with different chip packages.

These pulses were applied in turn to the latch enable signal to determine which signal produced the best results.

4.7.4 Evaluating Different Latch Enable Pulses

Having a latched data input on a DAC reduces the problems encountered with time skew. Time skew on the data bus results in non-harmonic spurs and a general increase in the noise floor.

A low noise floor close to the fundamental signal was important for the primary application of this project. It was therefore decided to evaluate the different latch applications with a narrow span on the spectrum analyser (100 kHz) at 7.5 MHz.

Appendix B4.1 shows the output with the DAC in a transparent mode with the latch enable signal grounded. The result was an average noise floor of -62 dBc and a worst case spur of -57 dBc.

Appendix B4.2 shows the output with the clock pulse applied directly to the latch enable signal. A marked decrease in the noise floor was experienced and virtually no recognisable spurs were present in the viewing bandwidth. The noise floor was down to almost -70 dBc at plus and minus 10 kHz from the fundamental signal. At plus and minus 50 kHz the noise floor was down to -80 dBc.

Appendix B4.3 shows the output with the inverted clock signal applied to the DAC. Not much difference to the previous plot was experienced within 10 kHz of the fundamental. However, definite random spurs, at -65 dBc in the worst case, were experienced around that. The average noise floor was also slightly higher than in the previous plot.

Appendix B4.4, showing the output with the narrow latch pulse, was somewhat disappointing. The signal compares well with the plot of the direct clock pulse within 10 kHz of the fundamental, but beyond that the average noise floor is about 10 dB higher.

4.7.5 Summary of Clock Manipulation

The plots in Appendices B4.1-4 clearly indicates that the best output was obtained with the clock pulse applied directly to the latch enable signal.

The reason for this becomes apparent when the latching registers of the DDS chip are examined. The output register of the DDS chip is operated by the leading edge of the clock pulse. This means that the data on the output bus of the DDS chip stays fixed until another leading edge of the clock pulse is applied.

The AD9713B DAC, however, requires a negative signal (level activated) on its latch enable pin to operate in the transparent mode. The data on the bus between the DDS and the DAC therefore has almost half a clock pulse duration to settle before the latch of the DAC is operated by the negative-going portion of the clock pulse. The positive-going portion of the clock pulse closes the latch of the DAC and the new DDS data becomes available for the next clock cycle.

With the slightly delayed and inverted clock pulse it is possible that the data on the bus has not settled yet, before the latch on the DAC is operated.

The spectral purity of the signal achieved with the direct clock pulse application, when compared to previous research, was sufficient not to require further experimentation with the interfacing between the DDS and the DAC.

4.8 Anti-alias Filter Module

The design of this filter took place after the DAC Module development had been finalised. The ideal characteristics of the anti-alias filter for the DDS signal generator, are as follows:

- Maximum flatness in the pass band
- High attenuation in the stop band
- Sharp roll-off above the cut-off frequency.

Since this is a sampled data system, spectral components will be generated at all the frequencies $nf_{CLK} \pm f_{OUT}$, where n is an integer. The fundamental frequency is at n=0 and the frequencies given by all other values of n are above the Nyquist frequency.

An ideal response for the anti-alias filter would not allow any signal through above the pass band, and within the pass band there would be no attenuation at all.

4.8.1 Filter Parameters

The output spectrum of the AD9713B DAC, as shown in Appendix B5.1, was used to determine the required filter parameters for this design. This plot shows the spectrum over a span of 35 MHz with a fundamental frequency of 7.5 MHz. The harmonic signal at 15 MHz was measured to be 50 dB down, the first alias signal at 22.5 MHz only 15 dB down, and the clock signal at 30 MHz was measured to be 32 dB below the fundamental.

This observation showed that the most attenuation is needed in the spectrum where the first alias signal would appear. In this case it is 20 MHz to 30 MHz.

It was therefore necessary to design a filter with a cut-off frequency of 10 MHz with a definite stop band from 20 MHz to 30 MHz.

4.8.2 Elliptic Filter Design

An elliptic filter is ideally suited for this design, because it provides a sharp cut-off with deep notches in the stop band (ARRL Handbook, 1993: 2-41).

A filter design with 5 stop band poles were chosen for this design with a cut-off frequency of 12 MHz and a stop band of 20 MHz. A ripple factor of 0.1 dB in the pass band was allowed in the theoretical calculations.

A schematic diagram of the filter, showing the theoretical component values, is included in Appendix A6. For the construction of the filter, values for the capacitors were chosen as closely as possible to these values. The inductors were wound with S.W.G. 22 on toroidal core T30-17.

4.8.3 Filter Response

The spectral response of the filter is shown in Appendix B5.2. The marker is set to 12.49 MHz, just above the cut-off frequency of 12 MHz. From the 12 MHz point to 18 MHz there is 35 dB attenuation. A deep null is shown at 24 MHz and at 30 MHz, the clock frequency, the attenuation is also 35 dB down from the average pass band level.

The plot differs somewhat from the required response in the stop band. This is as a result of the actual capacitor values in the filter not being equal to the calculated values. However, the notch at 24 MHz proved to be effective being in the middle of the stop band for the first alias signals.

Appendix B5.3 shows the same spectrum as the plot in Appendix B5.1, this time with the anti-alias filter included. The harmonic at 15 MHz was measured to be 67 dB down from the fundamental, the first alias 66 dB down and the 30 MHz clock only 40 dB down. The average noise floor had also improved with the filter in the output.

The response of the filter, as shown in Appendix B5.2, closely corresponds to the amount of attenuation experienced in the stop band of the plot in Appendix B5.3, except for the clock signal at 30 MHz. Only a 7 dB drop in level occurs while the filter is supposed to have 35 dB attenuation at that frequency.

At this stage of the design it was possible that clock feed through on the earth of the system could have been causing this problem. It was decided to do another measurement after the unit had been installed in a case with additional ground decoupling. If the problem persisted, a notch filter tuned to 30 MHz could be used to provide the required attenuation.

4.8.4 Summary - Anti-alias Filter

The spectral purity of the output achieved with the 5-pole elliptic filter proved to be acceptable for this project. Except for a possible notch filter at the clock frequency, no additional filtering was required at this stage.

CHAPTER 5 - SOFTWARE DEVELOPMENT

5.1 Introduction

The software provides an operating system for the signal generator, controlling the input, output and processing of data.

The source code was written in C and compiled to DS5000 (8051) machine code. Loading the software into the DS5000 is done with the RS232 serial loader available on the chip. A printout of the source code is shown in Appendix C.

5.2 Menu-driven System

The liquid crystal display (LCD) makes a menu-driven system possible, providing the user with a visual indication of options to select. This simplifies the operation of the system, because the user acts on prompts displayed on the LCD.

5.3 Subroutines

The software consists of a series of subroutines that interact to provide the required operating system. The software consists of input, output and processing routines. A short discussion of the relevant subroutines with flowcharts in menu context is provided to highlight their respective functions.

5.3.1 Main Routine - main()

In the main subroutine the initialisation of the UART in the DS5000 is done, setting it to 8 bits and the baud rate to 9600. The UART provides a serial link from the DS5000 to a personal computer. The LCD initialisation routine is also called from here.

After all the initialisation routines (as shown in Figure 5.1) are completed, the software enters into a loop calling the main_menu() routine.

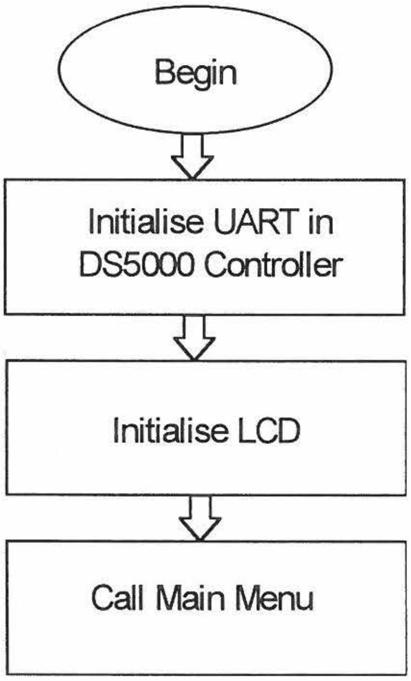


Fig 5.1. Flowchart of Main routine - main()

5.3.2 Main Menu - void main_menu(void)

The main menu offers a selection of 3 options displayed on the LCD, namely:

1. **SIG GEN:** In this mode the unit operates as a standard signal generator. The unit prompts for a required frequency to be typed in. The frequency entered is displayed on the LCD.

2. **FSK:** Two different frequencies can be loaded and selected at random by placing a high or a low signal on an externally provided pin.

3. **SWEEP:** A start-and-stop frequency can be entered to do a continuous sweep between the two frequencies in steps of 1 kHz.

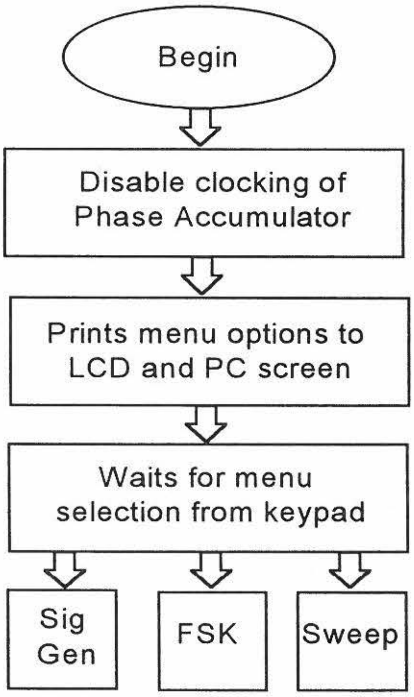

Begin
Disable clocking of Phase Accumulator
Prints menu options to LCD and PC screen
Waits for menu selection from keypad
Sig Gen
FSK
Sweep

Fig 5.2. Flowchart of Main menu - void main_menu(void)

When the main menu routine is entered into, it immediately disables the clocking of the phase accumulator. This inhibits the output of the DDS chip and prevents any unwanted frequencies from being generated while in the menu-select mode.

***5.3.3 Menu Option 1: Signal Generator** - void menu_opt1(void)*

When entering this menu, the user is prompted to enter a frequency. Any frequency between 0.001 and 10 MHz may be entered.

The Rotary Optical Encoder (ROE) is active in this menu for up/down frequency control. A decimal step size selection of 1Hz per step, up to 100 kHz per step, for the up-and-down control, can be selected on the keypad.

The flowchart in Figure 5.3 shows the sequence of events in this menu option.

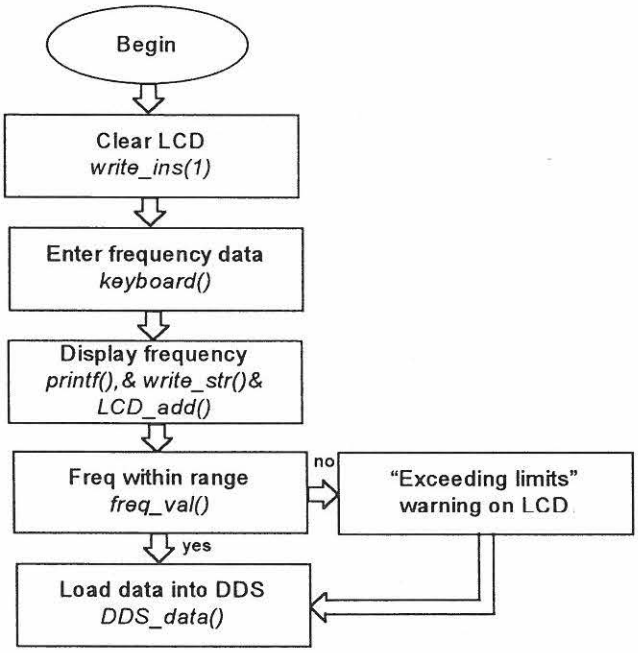


Fig 5.3. Flowchart of Menu Option 1

To clear the LCD and to return the cursor to the first position, an instruction routine, *write_ins(1),* is executed by the software.

The *printf()* routine is similar to the *write_str()* routine, the only difference being that the first prints out to a personal computer screen connected to the RS232 interface, and the latter to the LCD. This allows remote and dual control over the unit. *LCD_add()* sets the position of the cursor on the 4x20 line LCD.

The *freq_val()* routine gets the frequency data as entered by the keypad or terminal connected to the RS232 port, which is then displayed on the LCD. If the frequency value is above the upper cut-off limit of 10 MHz, a warning message is displayed. In the case of out-of-limit data, for the purpose of evaluation, a load attempt with the *DDS_data()* routine is still carried out towards the DDS.

5.3.4 Menu Option 2: FSK Mode - void menu_opt2(void)

In this mode, the user is prompted to enter two frequencies, namely Frequency A and Frequency B. Both frequencies are displayed on the LCD, but only one frequency is loaded into the DDS chip at a time.

Initially frequency A will be generated. To change the DDS output to frequency B, pin 22 of the DS5000 controller must be taken to a logic low level. This pin is identified as (A_B) in the software and is connected to a DB15 connector on the backplane of the signal generator unit (See Appendix A1).

Figure 5.4 shows the sequence of events taking place in the FSK software routine.

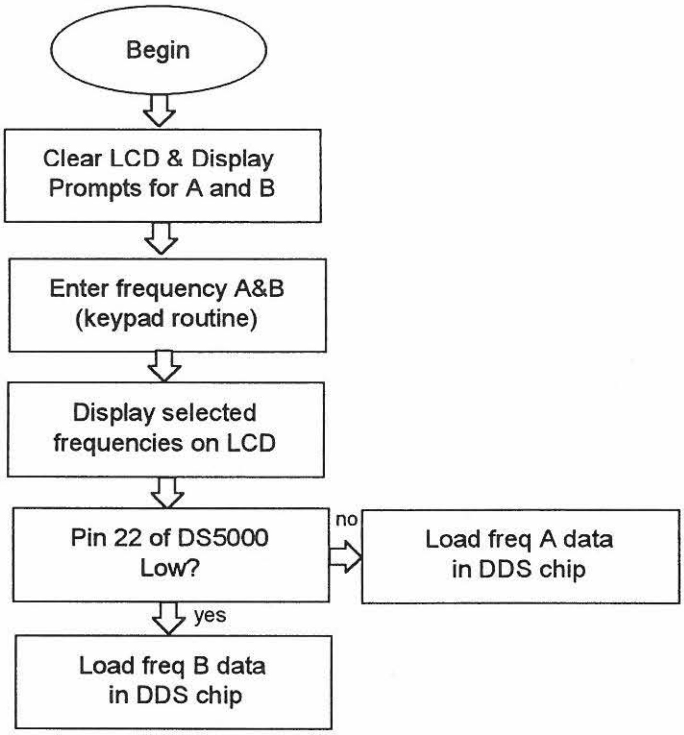

Begin
Clear LCD & Display Prompts for A and B
Enter frequency A&B (keypad routine)
Display selected frequencies on LCD
Pin 22 of DS5000 Low?
no
Load freq A data in DDS chip
yes
Load freq B data in DDS chip

Fig 5.4. Flowchart of Menu Option 2

5.3.5 Menu Option 3: Sweep Mode - *void menu_opt3(void)*

In this mode, the user is again prompted to enter two frequencies, namely the start - and stop frequency. Both frequencies are displayed on the LCD, but only the start frequency is initially loaded into the DDS chip.

After the frequency has been loaded into the DDS chip, it is incremented by one kHz and re-loaded. After every new loading sequence another increment takes place until the value of the stop frequency is reached.

At this point the initial start frequency is again loaded into the DDS chip and the whole sequence repeats itself until any key on the keypad is pressed.

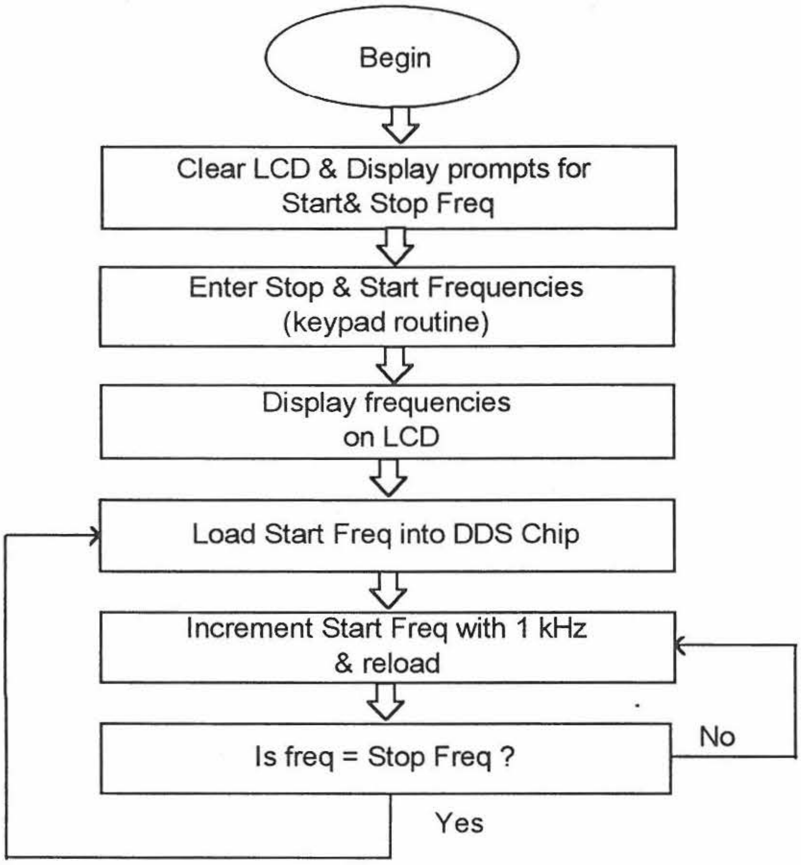

Begin
Clear LCD & Display prompts for Start& Stop Freq
Enter Stop & Start Frequencies (keypad routine)
Display frequencies on LCD
Load Start Freq into DDS Chip
Increment Start Freq with 1 kHz & reload
Is freq = Stop Freq ?
No
Yes

Fig 5.5. Flowchart of Menu Option 3

5.4 Special Routines

The following two routines were developed to simplify the integration of the hardware and need to be dealt with separately.

To explain the specific function of these routines, it will be stated from where these routines are called, what takes place in the routine and if any value is returned to the caller.

5.4.1 float freq_val()

The freq_val() routine is called by all three the menu options and is used to form the actual frequency value that is entered.

The routine starts with *write_ins(6)* which instructs the LCD to shift the cursor to the right with every new digit that is entered with the *keyboard()* routine.

As each new digit is entered, an array consisting of a maximum of digits, namely *freq_word[n]*, is formed. This array, when fully entered, contains the required frequency with the decimal point included.

To use this data in an equation, however, it must be converted to a floating point number. The line, *freq=atof(freq_word),* converts the array to a floating point number. This floating point number (freq) is then returned to the menu option that called this routine.

5.4.2 int DDS_data (float freq_data)

The *DDS_data()* routine calculates the phase accumulator word and controls the up/down frequency manipulation. Setting of the step size by means of the keypad, as well as controlling the ROE, is done in this routine. This routine is only called by menu option 1.

The floating point frequency value *(freq_data)* sent to this routine is converted into a long integer (PA) which is in the form of a 32-bit word.

This 32-bit word is sent to *DDS_ctrl(PA)* where the necessary control signals to the DDS chip are generated.

In the *DDS_ctrl(PA)* routine, the 32-bit word is clocked bit-by-bit into the phase accumulator.

After the frequency data has been loaded into the chip, the DDS-data() routine goes into a loop checking for any activity on the up/down controls, the ROE and the loop break command.

A problem was encountered to display the new frequency on the LCD if the ROE or up/down controls were activated. The frequency value had to be converted back to an array to enable byte-by-byte character loading to the LCD.

To re-display the actual frequency data on the LCD, a routine was developed, namely *float_to_string(float freq_LCD_in).* Unlike converting an array or string to a floating point number, no fixed command in C converts a floating point to an array or string where it can be printed out character-by-character on an LCD.

The *DDS-data()* routine does not return any data and can be terminated by pressing the # key on the keypad. If this routine is terminated, the software returns to the main menu.

5.5 Summary - Software

During the development of the software, each new routine was tested individually by inserting several printf() commands at hardware address points in the software. This simplified the debugging and testing of the code.

In cases where a data stream was involved, delays were inserted into the code to enable evaluation of the actual data with an oscilloscope or voltmeter.

The requirements set for the software regarding the operation and functioning of the system are met in full with the source code as included in Appendix C.

CHAPTER 6 - EVALUATION

6.1 Introduction

To evaluate the product it was decided to concentrate on the following measurable aspects of a typical radio frequency (RF) signal generator.

- Measuring the accuracy of the output frequency.
- Evaluating the frequency range of the generator.
- Spectral analysis of the output signal.

6.2 Frequency Accuracy

The aim of this test was to determine whether the frequency being displayed on the LCD corresponded to the actual output frequency of the generator.

6.2.1 Factors Influencing Frequency Accuracy

A look was taken at the factors which will have an influence on the output frequency of a DDS system. With reference to equation 3.2, the output frequency is determined by the following equation:

$$f_{out} = \frac{FSW(N)f_{clk}}{2^{32}} \qquad 6.1$$

From the above equation it is clear that the output frequency is determined by the FSW (frequency setting word), as well as the value of the clocking frequency.

With the FSW being a digitally fixed value, the only other factor influencing the accuracy of a DDS system is therefore the clocking frequency.

6.2.2 Clocking Frequency Drift

In this specific DDS design, a 30 MHz Jauch TTL-type oscillator with a 100 PPM frequency stability was used as a clocking source (Jauch Datasheet).

At a fixed room temperature of 25 degrees Celsius, the TTL oscillator took plus minus 30 minutes to stabilise on a frequency within 3 Hz of 30 MHz. No further drift was experienced even with slight changes in room temperature.

When a maximum frequency drift of 3 Hz in the clocking frequency is substituted in equation 6.1, it becomes clear that, with the output frequency set to 10 MHz, a drift of only 1 Hz can be expected on the actual output of the DDS. At a frequency output of 1 MHz, a drift of 0.1 Hertz can be expected with the same amount of drift in the clocking frequency.

6.2.3 Frequency Accuracy Evaluation

The frequency accuracy of the DDS signal generator was determined simply by setting the output to a number of different frequencies in the 0 to 10 MHz range and measuring the output with a HP5350B Hewlett Packard frequency counter. An initial warm-up period of 30 minutes was allowed before any final measurements were taken.

To determine if the calculations shown in 6.2.2 are valid, measurements were taken during the warm-up period as well. Throughout all these measurements, the frequency error on 10 MHz were exactly one third of the clock error as calculated in 6.2.2. With the output set to 1 MHz, the effect of clock drift on the output was exactly one tenth of that experienced on 10 MHz.

6.2.4 Summary - Frequency Accuracy

From the above calculations and tests it is clear that, after a warm-up period of 30 minutes, the DDS signal generator is performing within the set specifications. Above tests have also emphasised the fact that if the clocking frequency is much higher than the output frequency, less frequency error will occur on the output.

The provision of a stable and accurate clock source is therefore one of the most important design considerations in a DDS system.

6.3 Evaluating the Frequency Range

The requirement set in the design specification was for the frequency generator to be adjustable from DC to 10 MHz in 1 Hz steps with the minimum amount of attenuation over the filter passband.

This evaluation determined if the required 0-10 MHz can be achieved and to measure the flatness of the output amplitude in the passband.

6.3.1 Limitations

The limitations in the frequency range of a DDS system are the amount and level of unwanted spurs close to the fundamental output frequency that can be tolerated for a certain application.

Practical DDS designs rarely utilises frequencies higher than one third of the clocking frequency (Gallant, 1989: 97). With that in mind, a filter with a 3 dB cut-off point of 12 MHz was chosen. The measured response of this filter is shown in Appendix B5.2.

6.3.2 Range and Output Level

Table 6.1 shows the output frequency (Fout in MHz) with the associated output level (Pout in -dBm) in 1 MHz steps from 1 to 15 Mhz.

Table 6.1 Output frequency and level.

Fout	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Pout	2.66	2.76	2.81	2.94	2.98	3.00	3.02	3.42	3.33	4.11	5.43	6.28	12.7	28.5	38.4

The contents of the above table closely follows the response of the anti-alias filter curve as shown in Appendix B5.2. The maximum difference in output level over the entire design range up to 10 MHz does not exceed 1.45 dBm.

6.3.3 Summary - Frequency Range

The required 0 to 10 MHz tuneable range was achieved with a 1.45 dBm variation in output level over the entire range. If a flatter passband is required, an external attenuator/amplifier module can be incorporated to increase or decrease the level as required.

For the purpose of this design, above results were satisfactory and provided the full range tuneable in 1 Hz steps with an acceptable amount of attenuation.

6.4 Spectral Analysis of the Output Signal

Spectral purity is a fundamental objective in any synthesiser design. In DDS systems the two undesirable signal responses are the alias and spurious signals.

6.4.1 Expected Performance

Gallant (1989: 104) states that empirical testing has shown that for every quantisation bit that is added to the DAC, non-harmonic spurious signals will be suppressed 6 dB below the carrier frequency.

The above implies that in this case, with a 12 bit DAC, a non-harmonic spurious response of -72 dB should be achieved.

6.4.2 Evaluation Method

Zavrel has worked out a simplified procedure for evaluating DDS spurious responses as described in the Stanford Telecom Application Note 102,(page viii).

His method of evaluation is based on the fact that at integer divisions of the clock frequency, many of the spurs fall directly on the main tuning frequency. For example, all the odd-order spurs fall on the tuned frequency when it is at 1/4 clock. Similarly, many even-order spurs and also many odd-order spurs fall on the tuned frequency when it is at1/3 clock.

By setting the tune frequency slightly offset from the integer division of the clock, the spurs will spread out nicely within a relatively narrow bandwidth, facilitating sensitive accurate measurements on a spectrum analyser.

6.4.3 Harmonic and Non-harmonic Spurious Response

Appendix B6.1 shows a plot with the carrier set at 970 kHz. This frequency enables viewing of the harmonic as well as the non-harmonic spurs present over a large spectrum.

The 2nd and 3rd harmonic spurs are both 64 dB down from the carrier while the worst non-harmonic spurs are measured to be -73 dBc.

Appendix B6.2 shows the carrier output at 9.99925 MHz which is just below 1/3 of the clock frequency. The highest spur within this 50 kHz spectrum shown is -63 dBc. The rest of the spurs further than 10 kHz from the carrier are all better than-75 dBc.

Appendix B6.3 shows the carrier frequency set at 7.525 MHz which is close to a quarter of the clock frequency. The non-harmonic spurs are all better than -73 dBc. The worst case spur within the 1 MHz span is -62 dBc.

6.4.4 Summary - Spectral Analysis

The plots done at very close to 1/3 and 1/4 of the clock frequency has shown the worst case spurs to be in the order of -62 dBm with the non-harmonic spurs down by better than -73 dBc.

6.5 Summary - Evaluation

The results achieved in the above tests correspond closely to what was expected from the hardware utilised.

However, the main concern during the development and the evaluation, was the high level of the 30 MHz clocking frequency present on the output even after the filter was added.

The high clock level on the output is caused by clock feedthrough on the earth of the system. When the generator was fully assembled, no improvement in the clock feedthrough was obtained as expected. While a notch filter on 30 MHz may cure the problem, the best engineering practice would be to provide further isolation between the digital and analogue grounds. This however, was beyond the scope of the project and it was not attempted.

The proposed signal generator was therefore developed and tested successfully.

CHAPTER 7 - SUMMARY

The purpose of this study, namely to develop a radio frequency signal generator variable from 0 - 10 MHz, utilising direct digital frequency synthesis, was achieved. The result of this project is a low-cost but very agile frequency generator.

Chapter 3 dealt with the theory of digital sampling, as well as the theory of direct digital frequency synthesis. A background of sampling theory is highly relevant in understanding the theory of digital frequency synthesis.

Problems associated with DDS systems and the main causes of spectral impurities are also covered in this chapter.

The hardware development is covered in Chapter 4. The design specifications are given with a description of the development of each individual module. An evaluation of two different DACs, in order to determine which one is best suited for the application, is also included in this chapter.

Chapter 5 covers the software development with reference to the interfacing of software routines to efficiently control the hardware.

The evaluation of the final product is covered in Chapter 6. The results achieved with this signal generator is explained with reference to several frequency spectrum plots done during the development and final evaluation. A short introduction into the testing of DDS systems is also included in this chapter.

During the execution of the project, insight was gained with respect to the following:

- DDS theory;
- DDS hardware interfacing;
- C programming as well as using the versatile DS5000 microcontroller;
- The importance of sound design principles in a hybrid digital and radio frequency project.

APPENDIX A

Circuit Diagrams

1. DS5000 Control Module

2. HSP45102 DDS Module

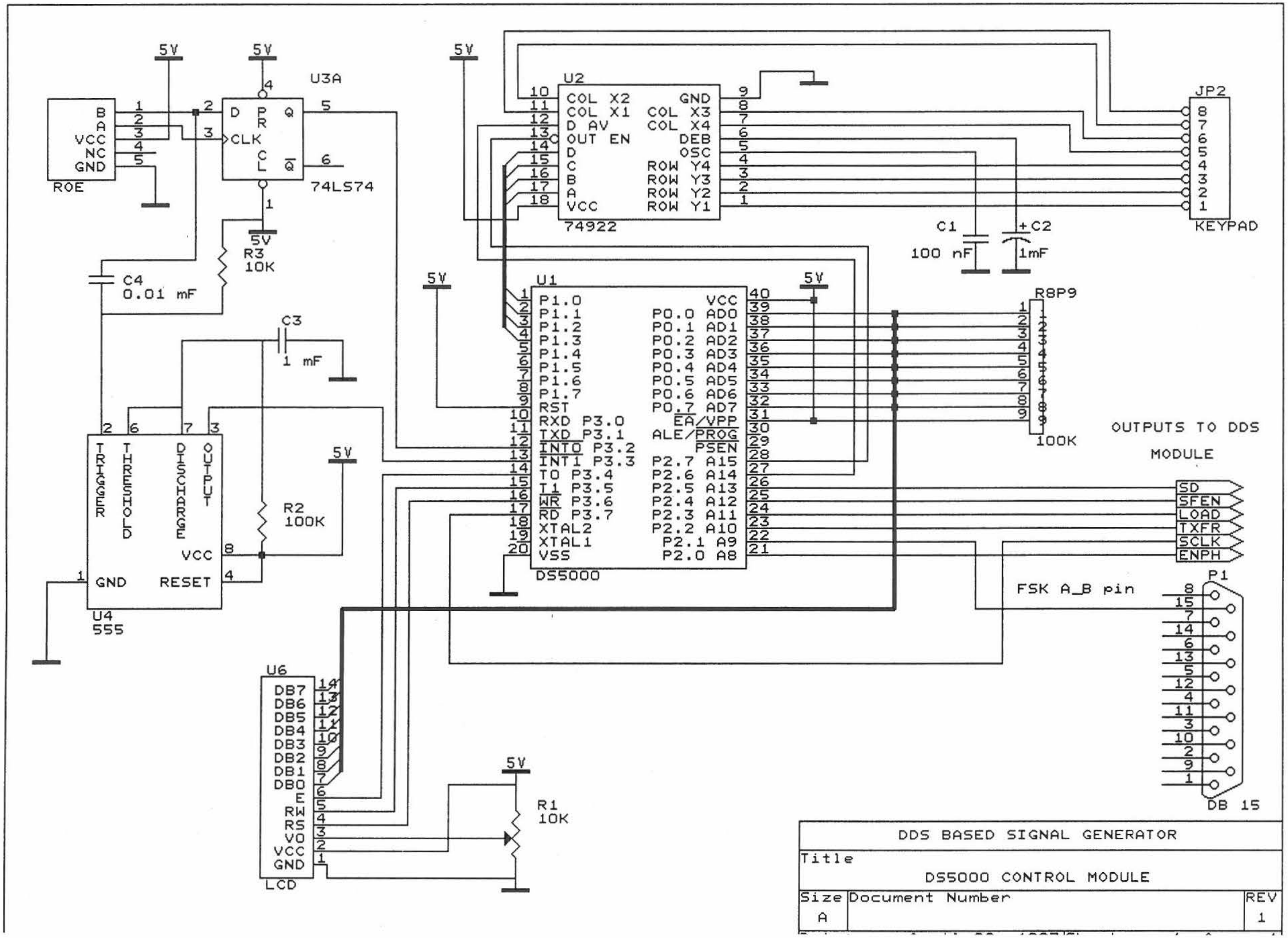
3. HI565 Harris DAC Module

4. AD9713 Analog Devices DAC Module

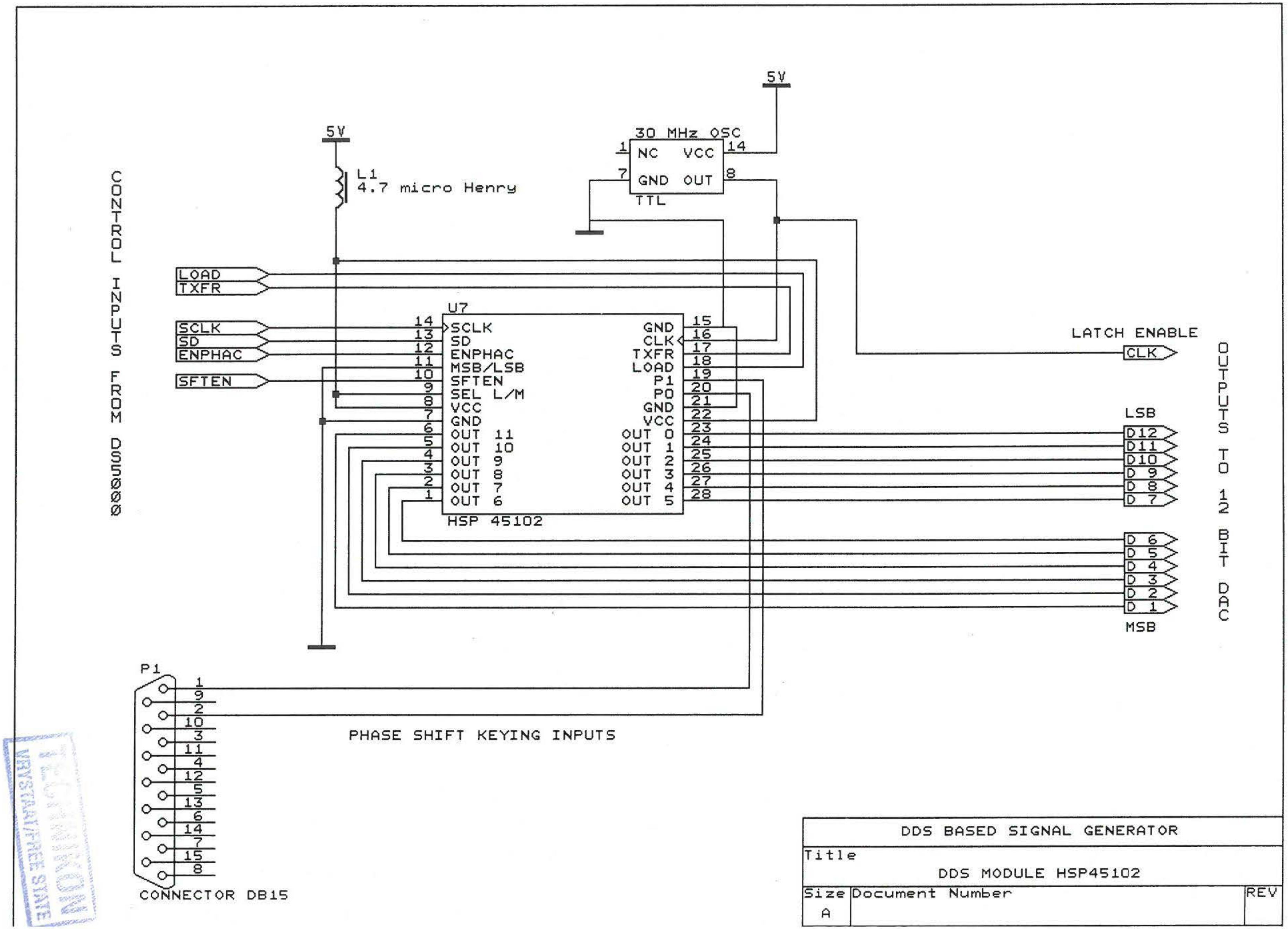
5. Clock Pulse Manipulation

6. Anti-Alias Filter

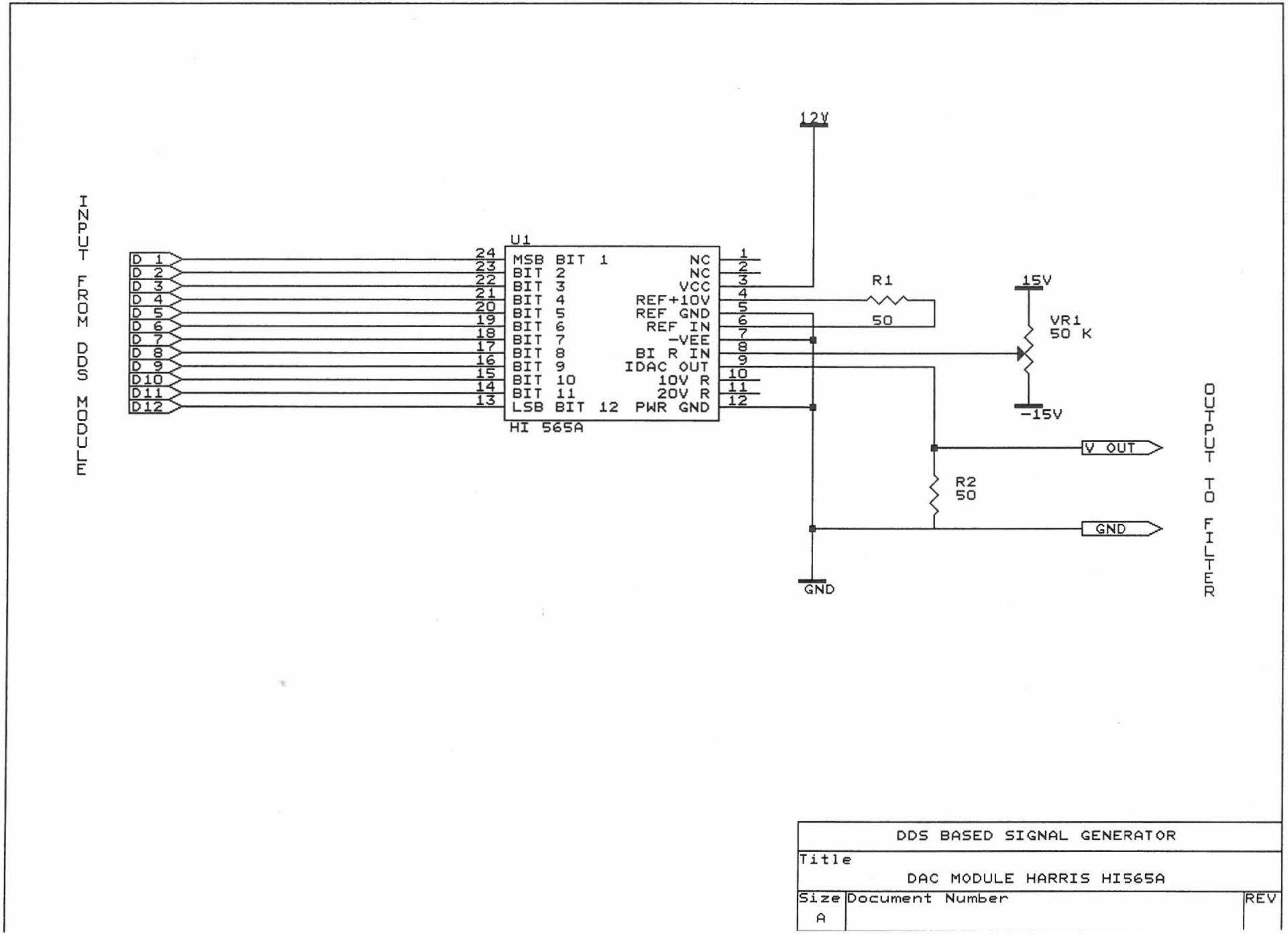
Appendix A1: DS5000 Control Module



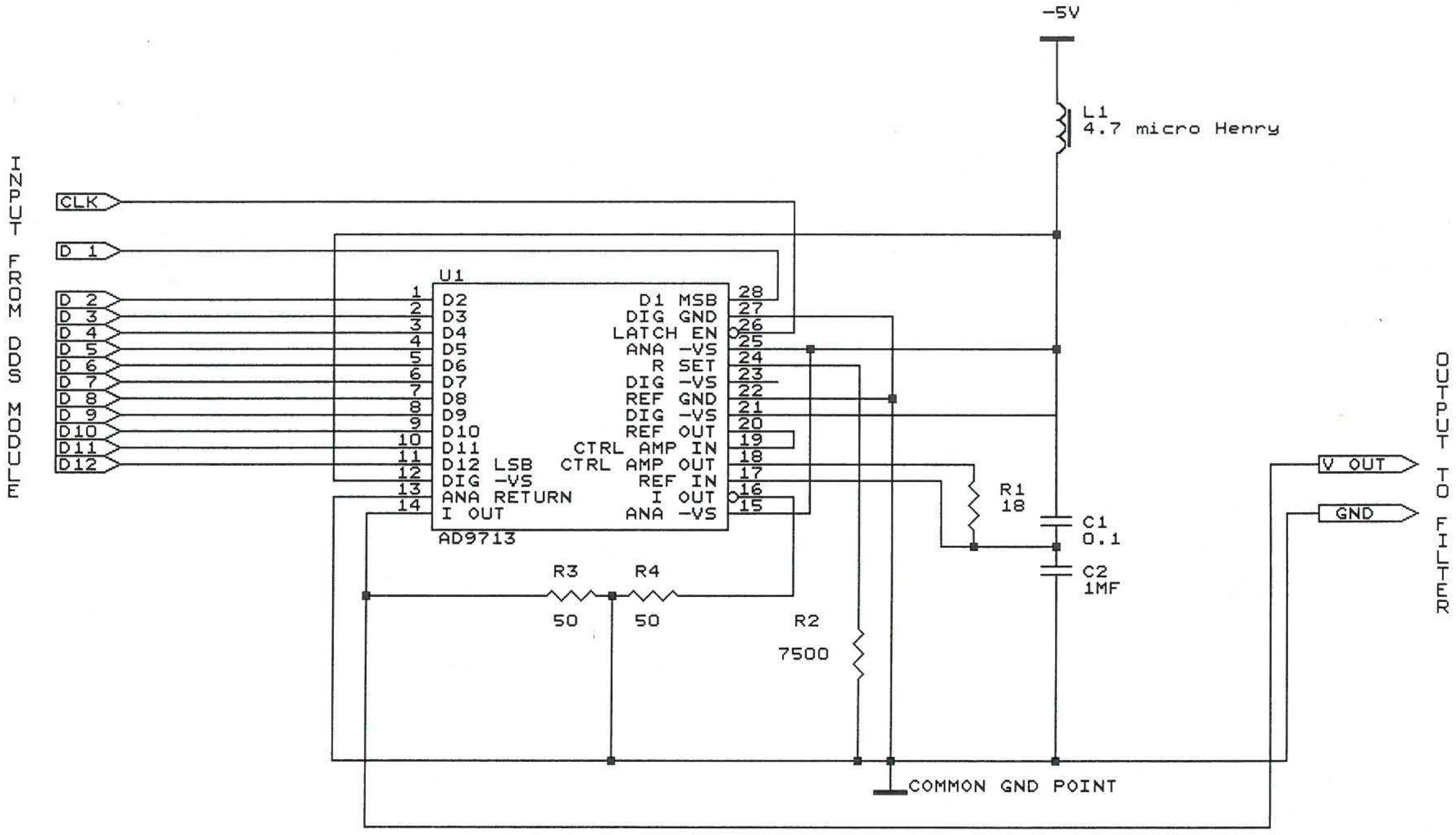
Appendix A2: HSP45102 DDS Module



Appendix A3: HI565 Harris DAC Module



Appendix A4: AD9713 Analog Devices DAC Module



DDS BASED SIGNAL GENERATOR

Title

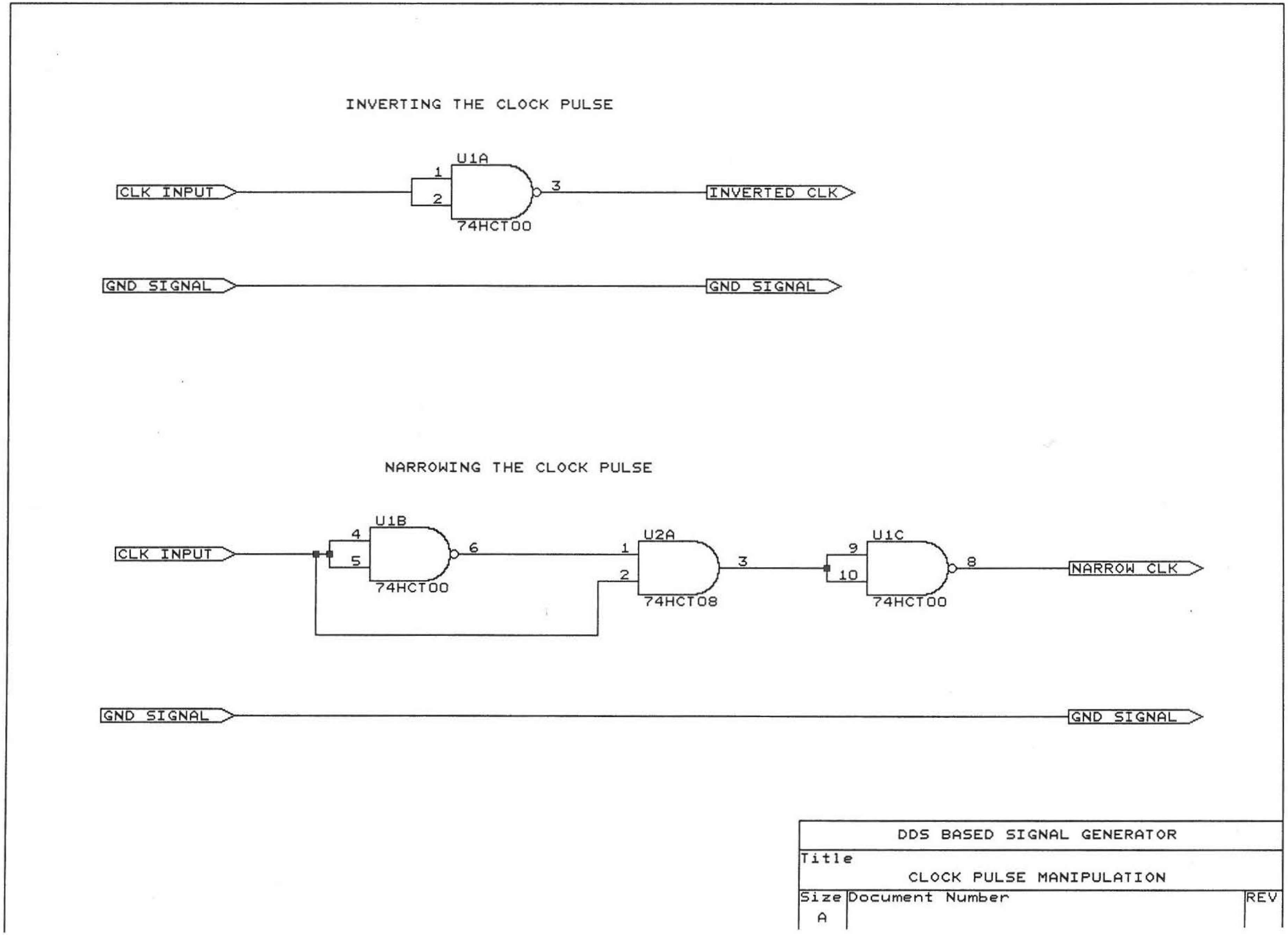
DAC MODULE ANALOG DEVICES AD9713

Size A

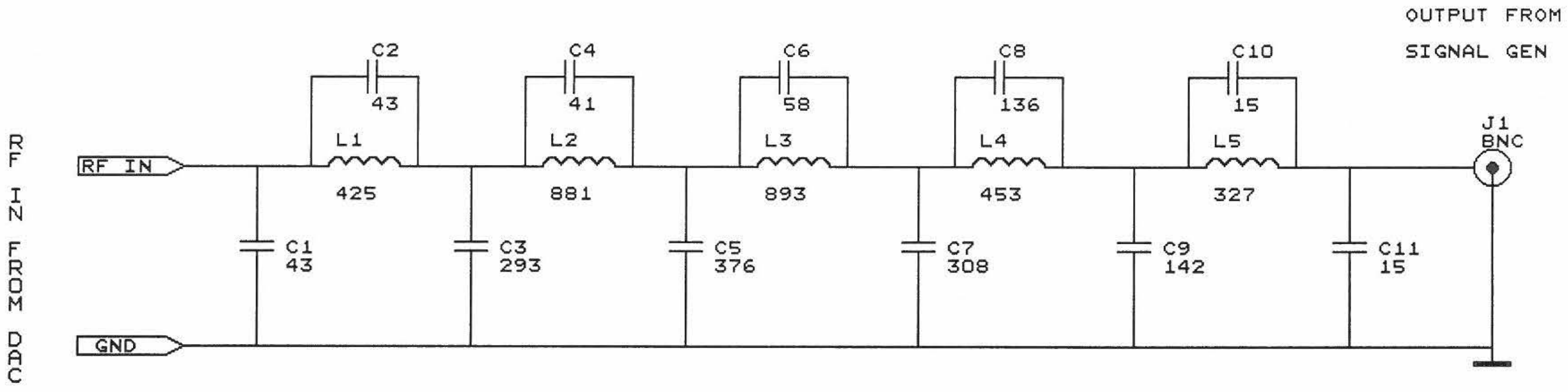
Document Number

REV

Appendix A5: Clock Pulse Manipulation



ANTI-ALIAS FILTER MODULE



ALL INDUCTORS WOUND WITH SWG 22

ON TOROIDAL CORE T30-17

DDS BASED SIGNAL GENERATOR		
Title ANTI-ALIAS FILTER		
Size A	Document Number	REV

APPENDIX B

Spectrum Analyser Plots

1. HI 565 Dac with Fundamental at	1) 7.5 Mhz	(30 MHz clock)
	2) 7.6 MHz	
	3) 1 MHz	
2. HI 565 DAC with Fundamental at	1) 2.5 Mhz	(10 MHz clock)
	2) 2.6 MHz	
	3) 1 MHz	
3. AD 9713 DAC - Fundamental at	1) 7.5 Mhz	(30 MHz clock)
	2) 7.6 MHz	
	3) 1 MHz	
4. AD 9713 DAC - 7.5 MHz	1) Transparent Latch	
	2) Direct Clock Pulse	
	3) Inverted Clock Pulse	
	4) Narrow Clock Pulse	

5.1) AD 9713 DAC output with no Anti-Alias filter.

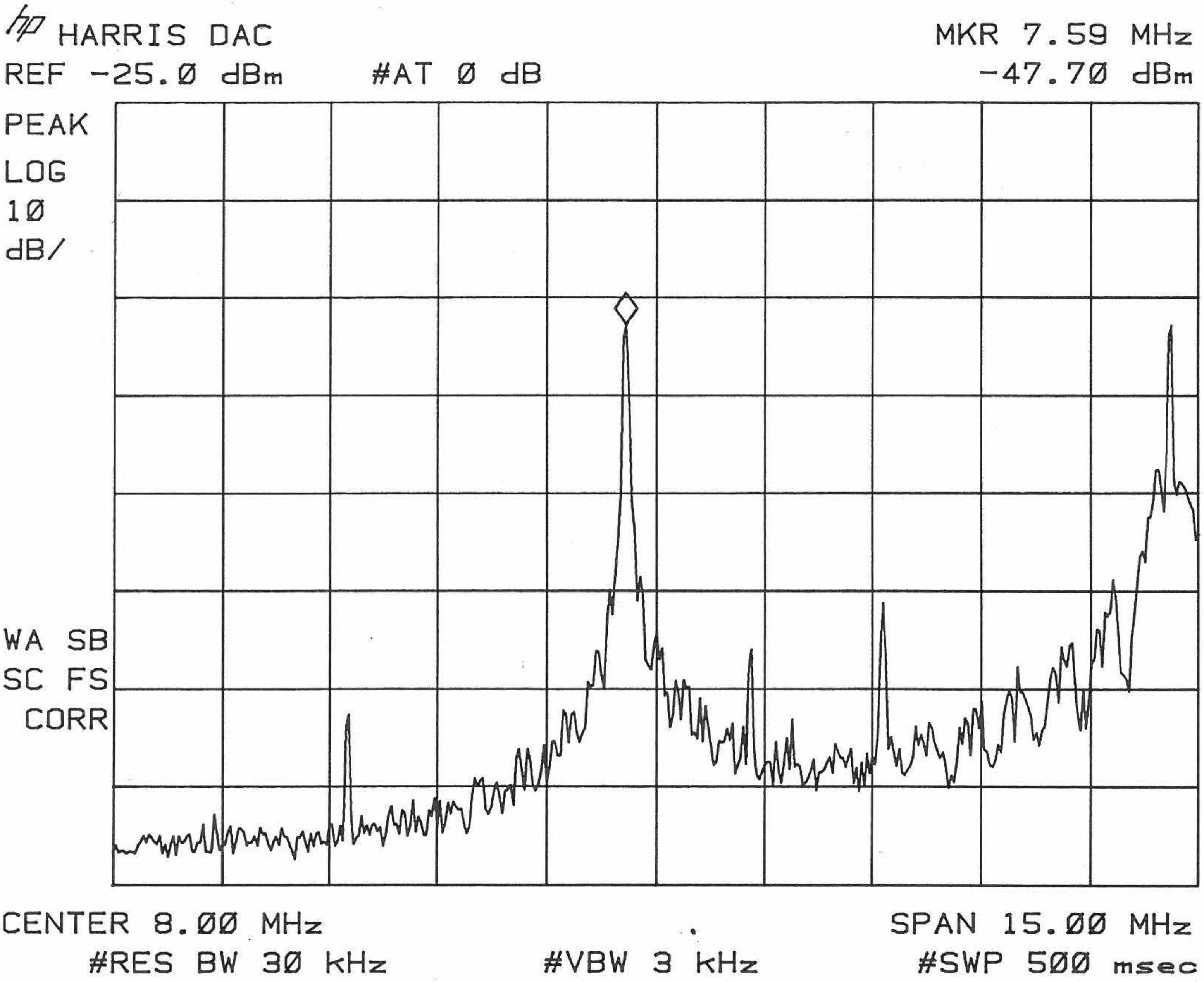
2) Anti-Alias Filter Response.

3) AD 9713 DAC output with Anti-Alias Filter.

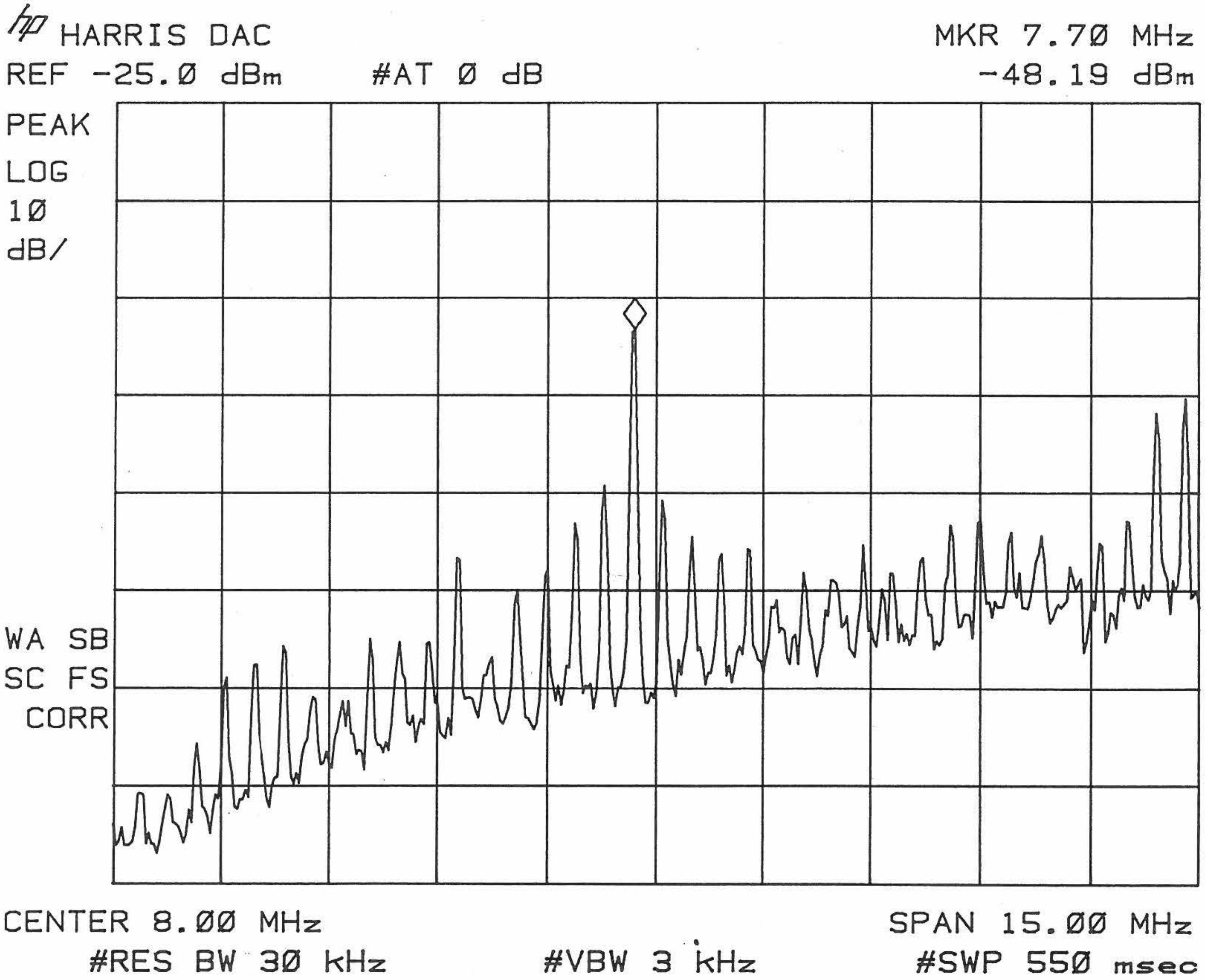
6.1) Harmonic spur evaluation

2) Carrier at 1/3 clock evaluation

3) Carrier at 1/4 clock evaluation

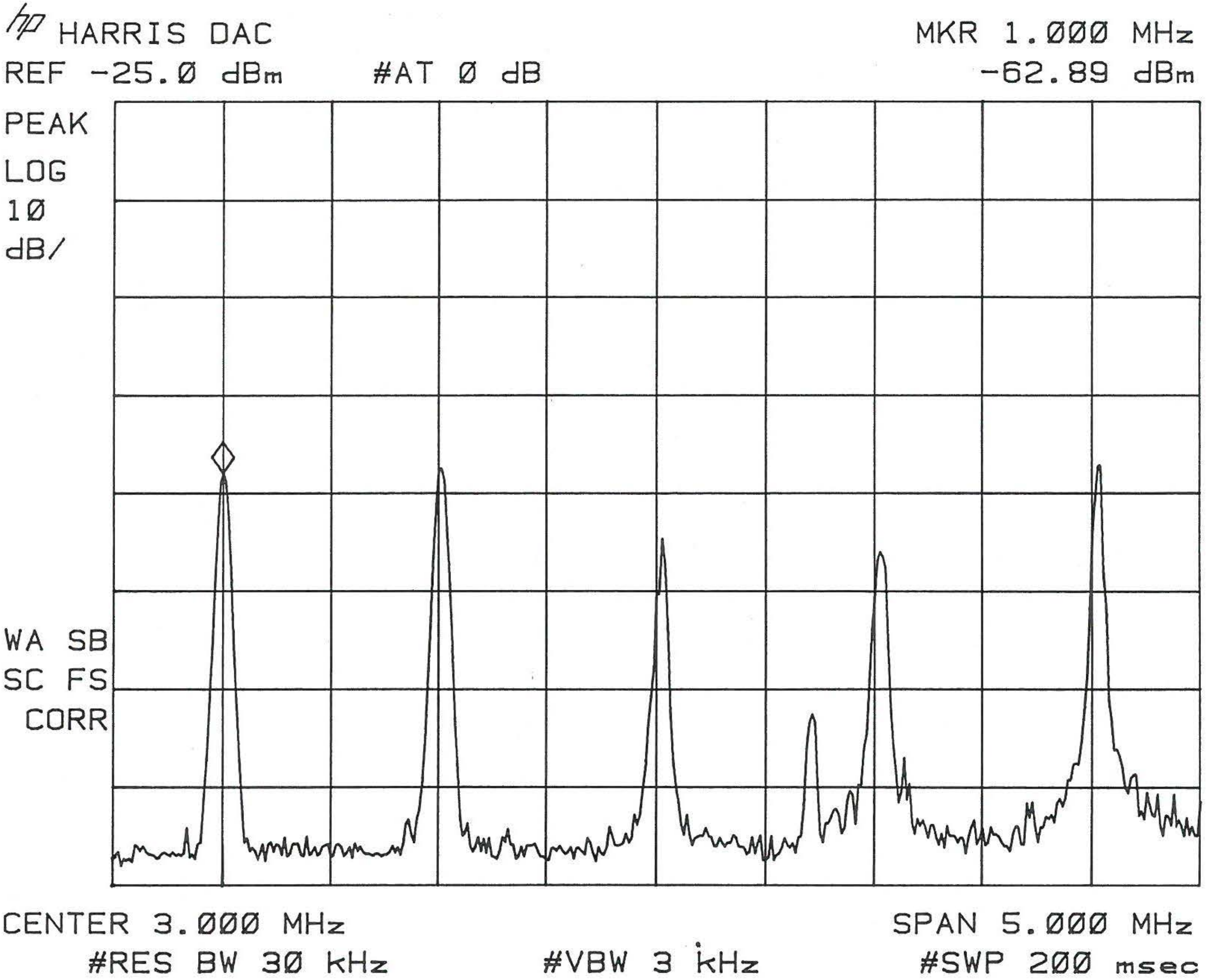


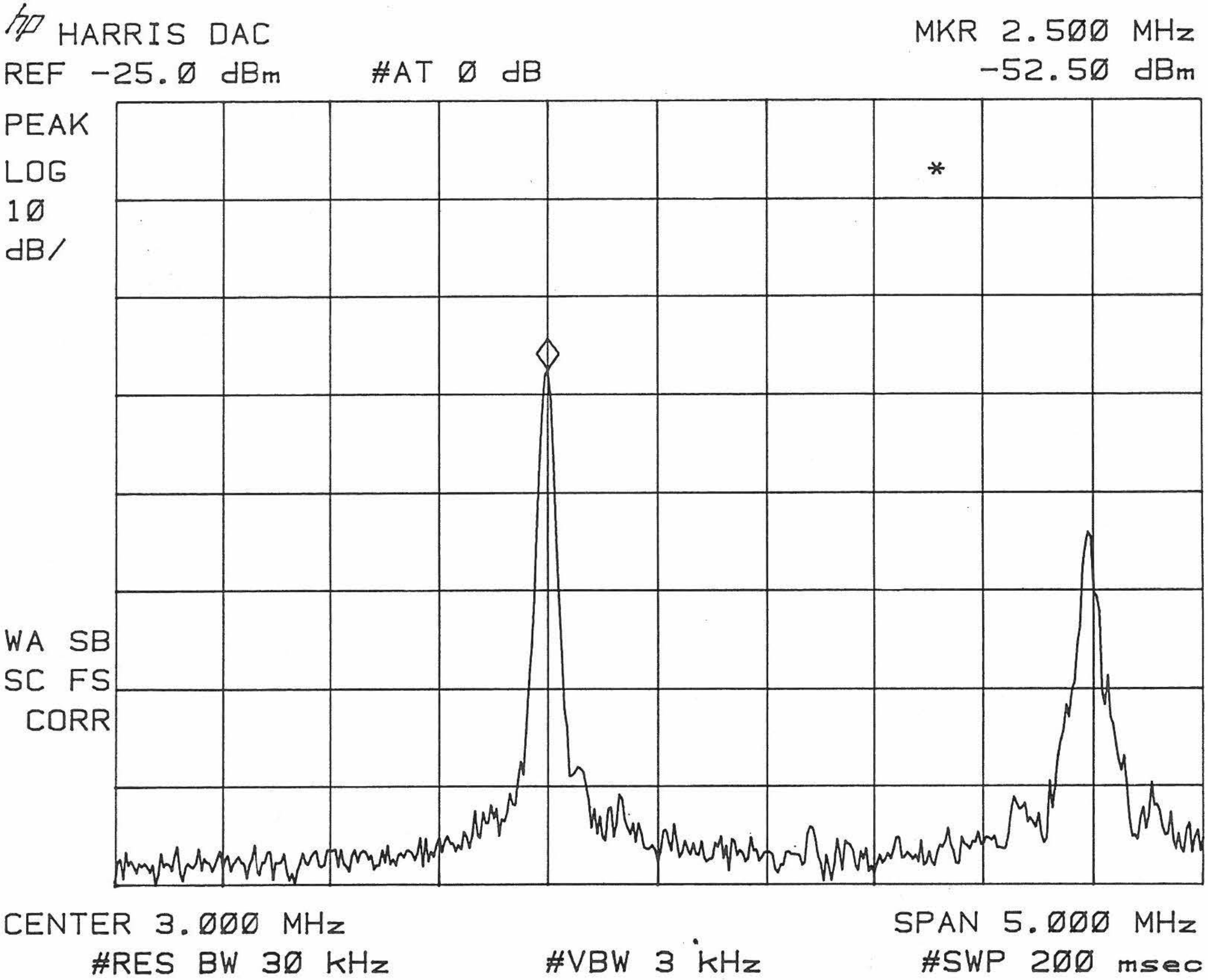
Appendix B1.1



Appendix B1.2

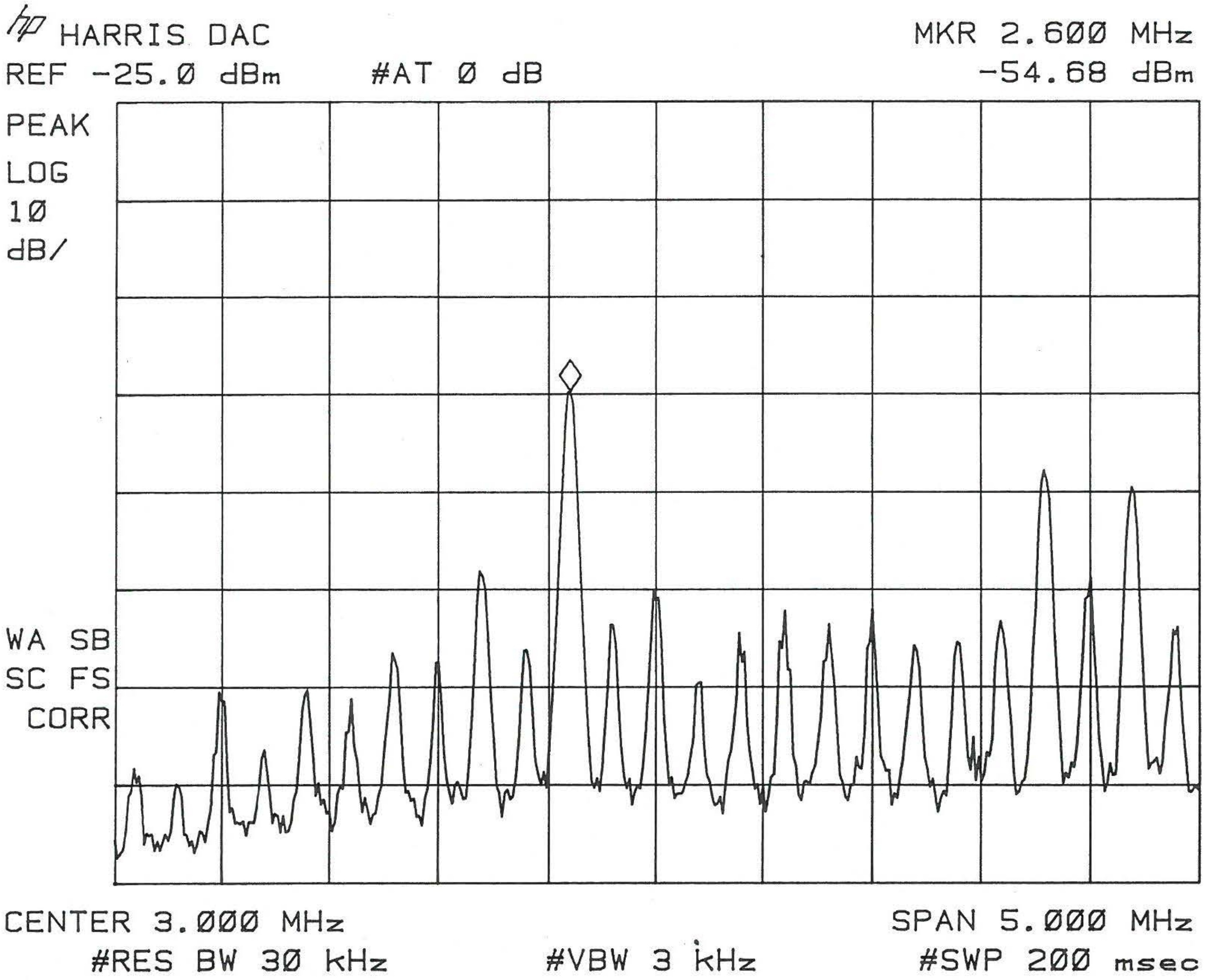
Appendix B1.3



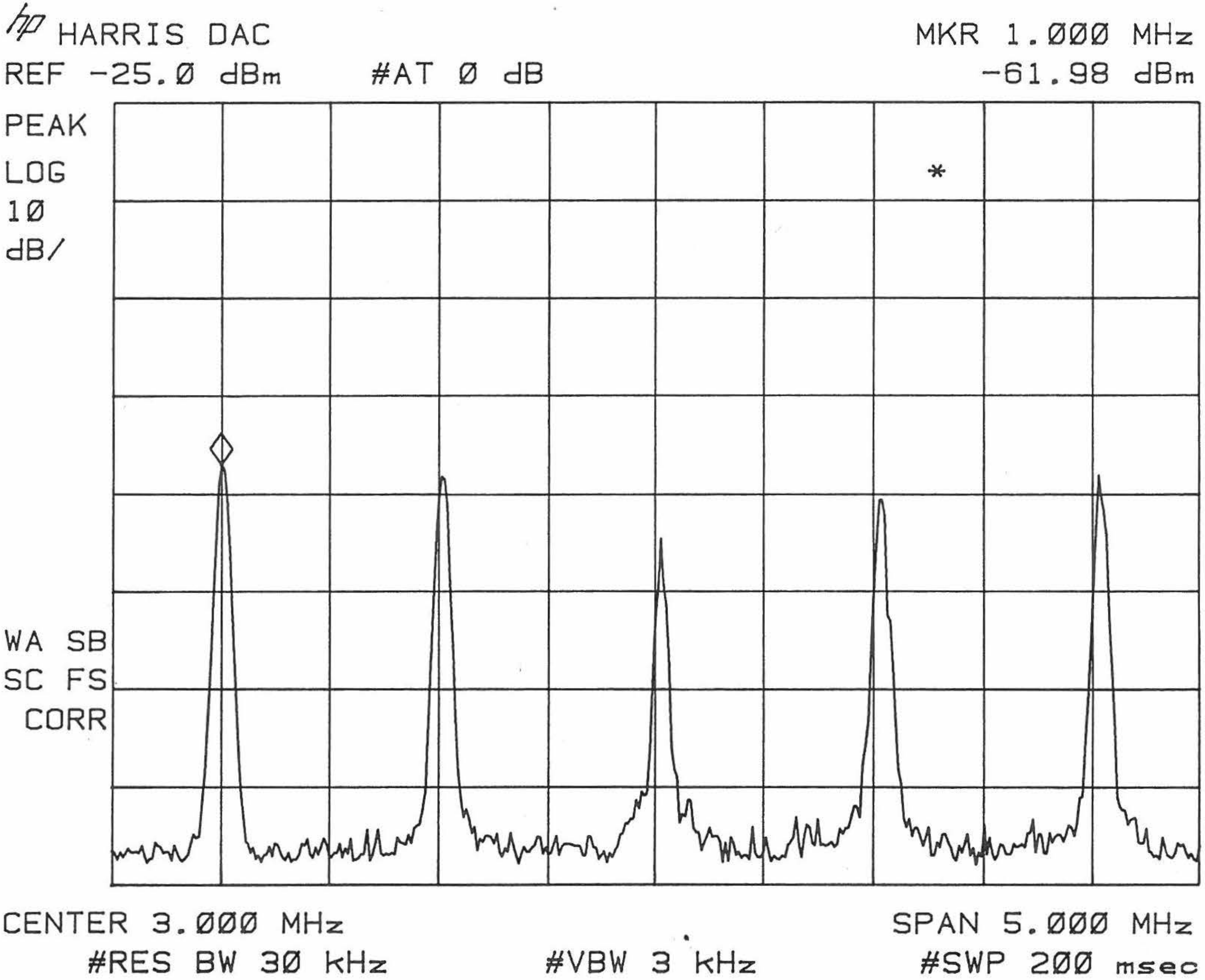


Appendix B2.1

Appendix B2.2



Appendix B2.3



HARRIS DAC

REF -25.Ø dBm #AT Ø dB

MKR 1.ØØØ MHz
-61.98 dBm

PEAK
LOG
1Ø
dB/

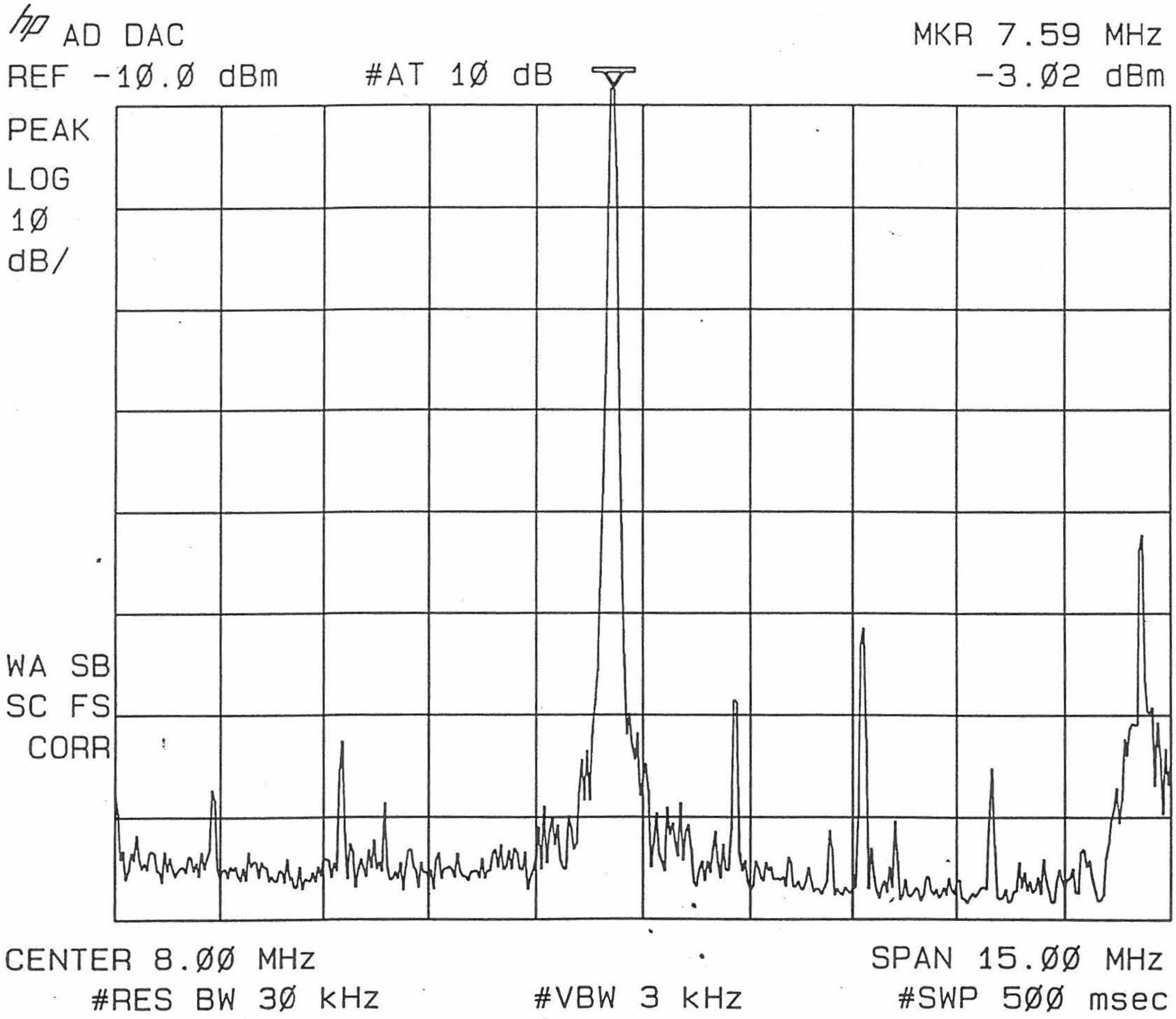
WA SB
SC FS
CORR

CENTER 3.ØØØ MHz

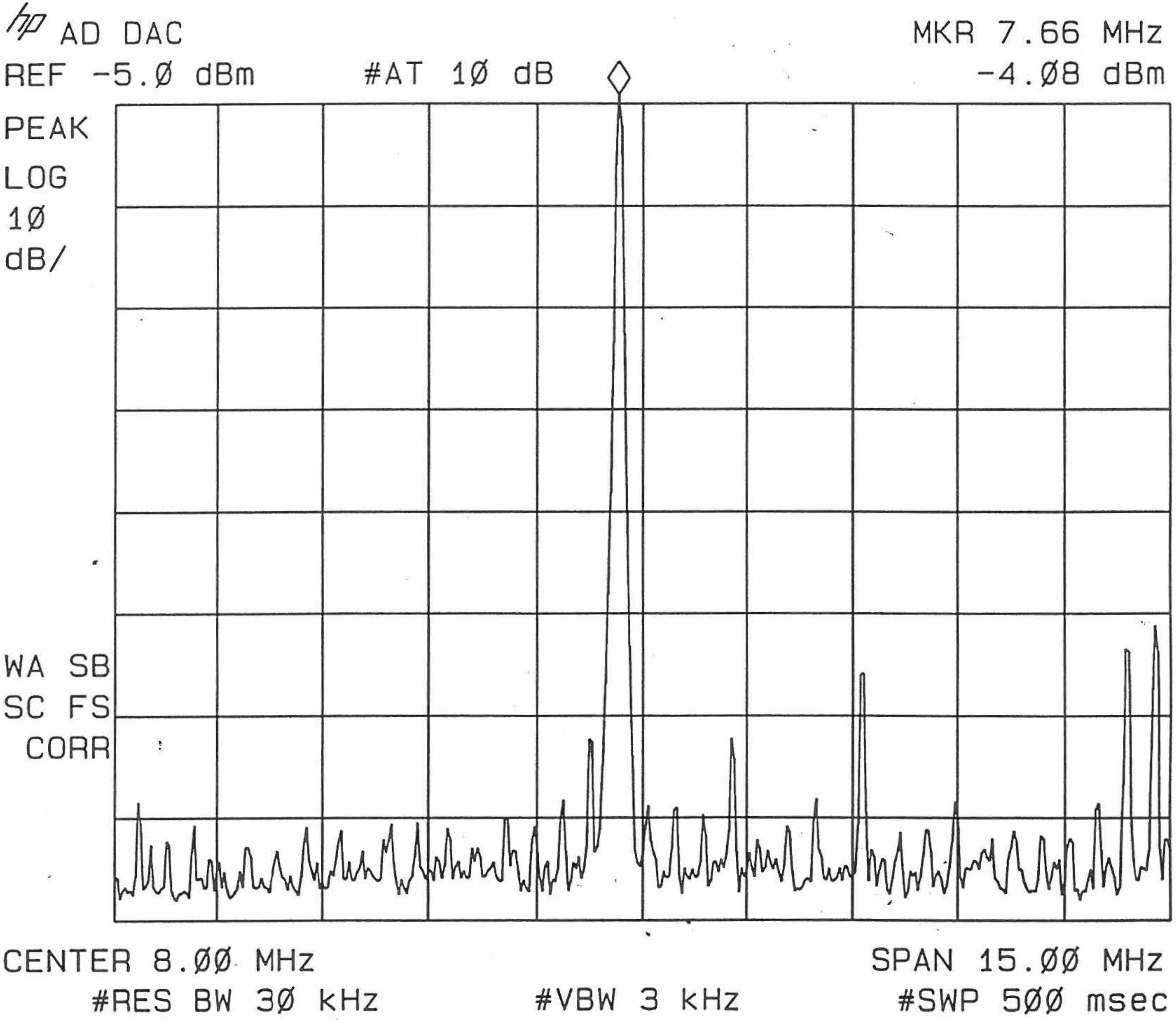
SPAN 5.ØØØ MHz

#RES BW 3Ø kHz #VBW 3 kHz #SWP 2ØØ msec

Appendix B3.1



Appendix B3.2

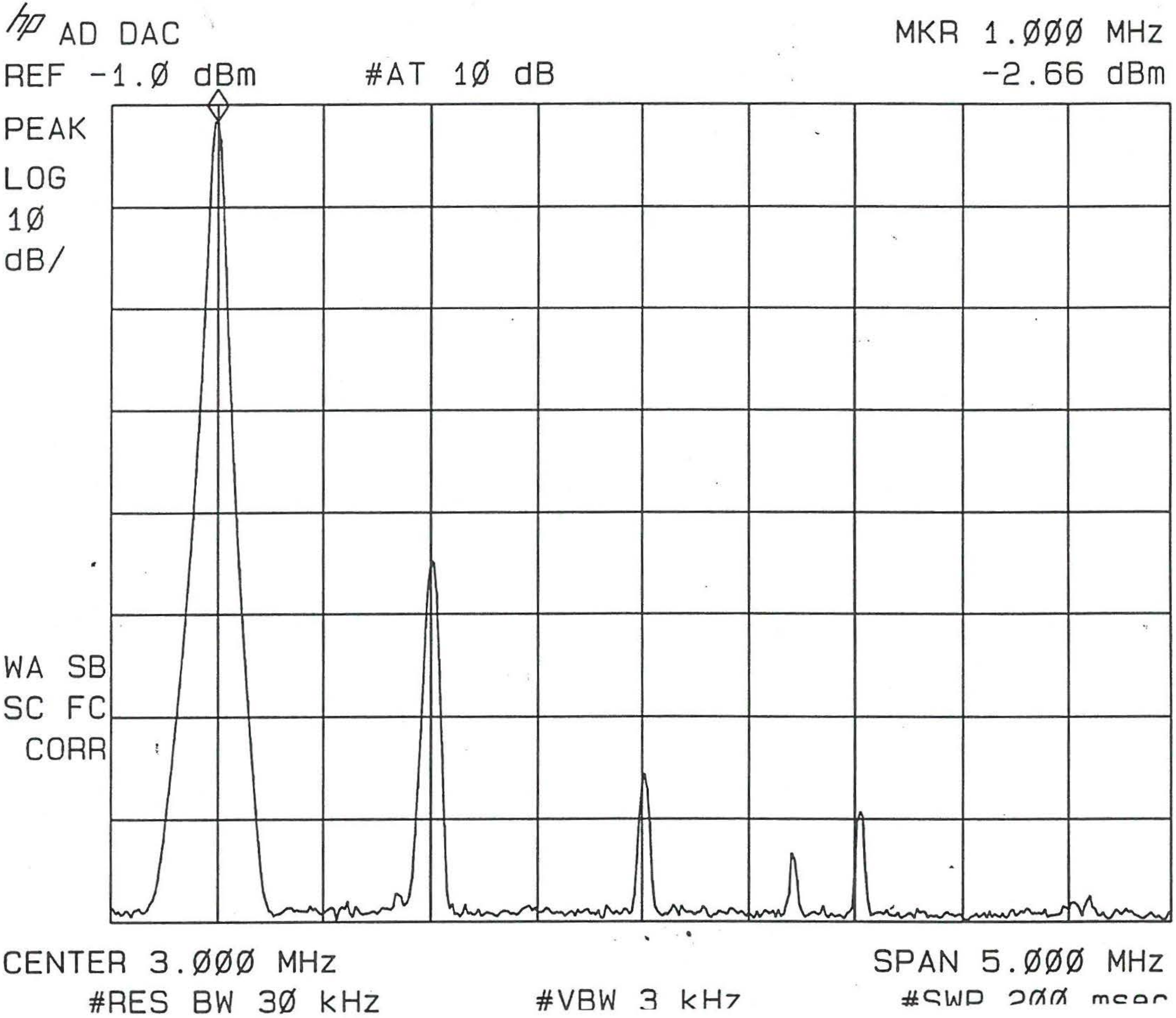


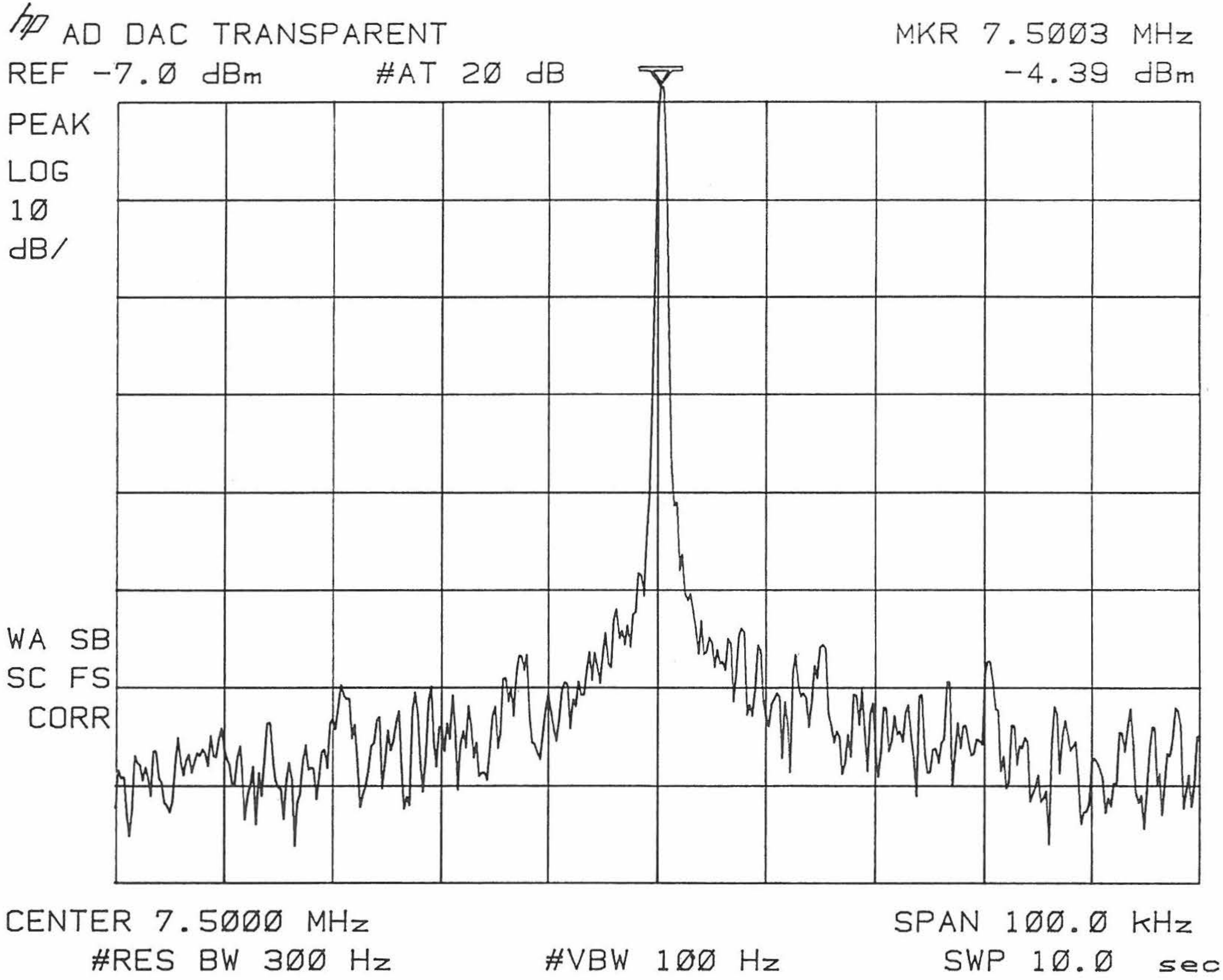
```
hp AD DAC                                   MKR 7.66 MHz
REF -5.Ø dBm        #AT 1Ø dB                  -4.Ø8 dBm
PEAK
LOG
1Ø
dB/

WA SB
SC FS
 CORR

CENTER 8.ØØ MHz                        SPAN 15.ØØ MHz
    #RES BW 3Ø kHz        #VBW 3 kHz    #SWP 5ØØ msec
```

Appendix B3.3





AD DAC TRANSPARENT

MKR 7.5ØØ3 MHz

REF -7.Ø dBm #AT 2Ø dB -4.39 dBm

PEAK
LOG
1Ø
dB/

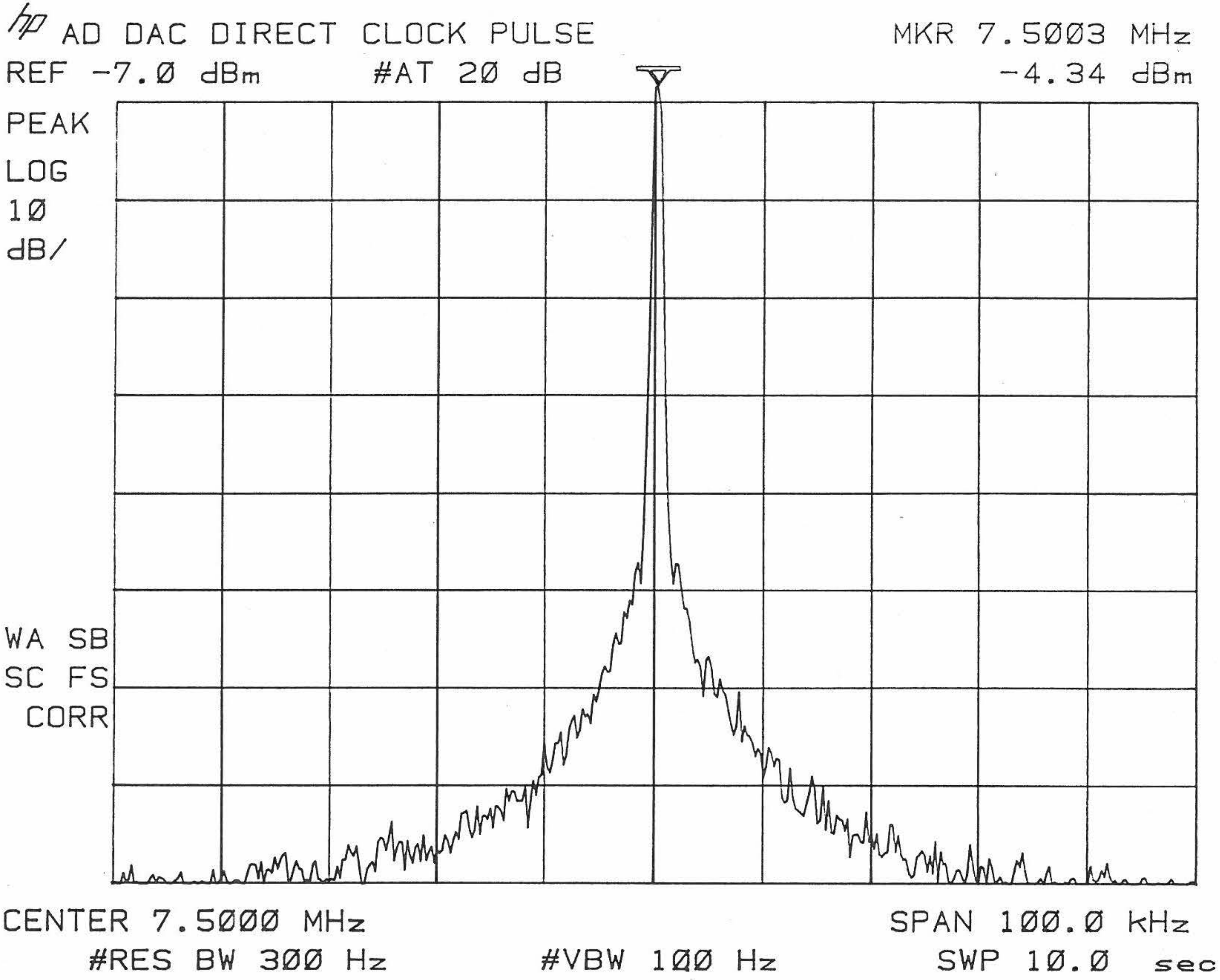
WA SB
SC FS
CORR

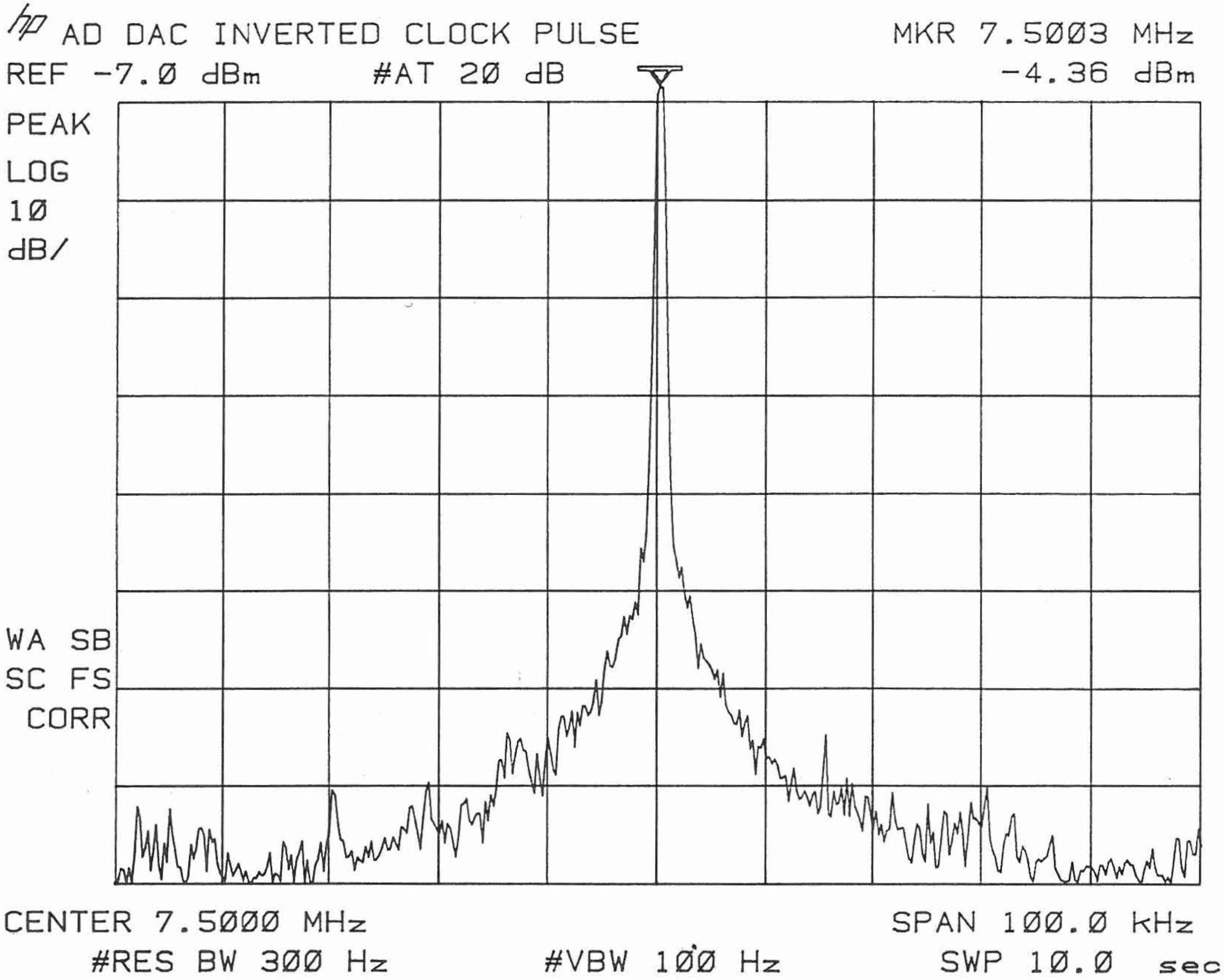
CENTER 7.5ØØØ MHz SPAN 1ØØ.Ø kHz

#RES BW 3ØØ Hz #VBW 1ØØ Hz SWP 1Ø.Ø sec

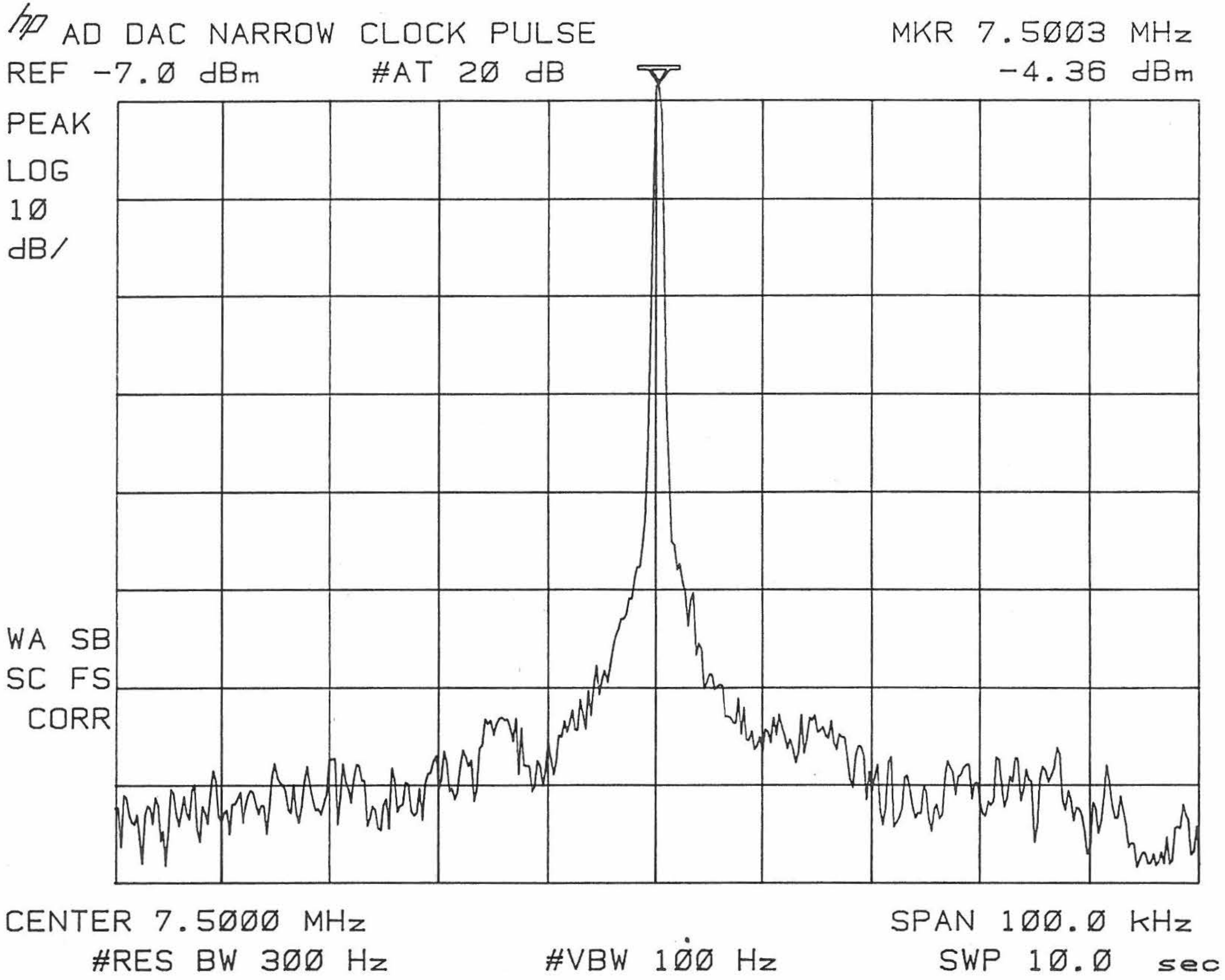
Appendix B4.1

Appendix B4.2

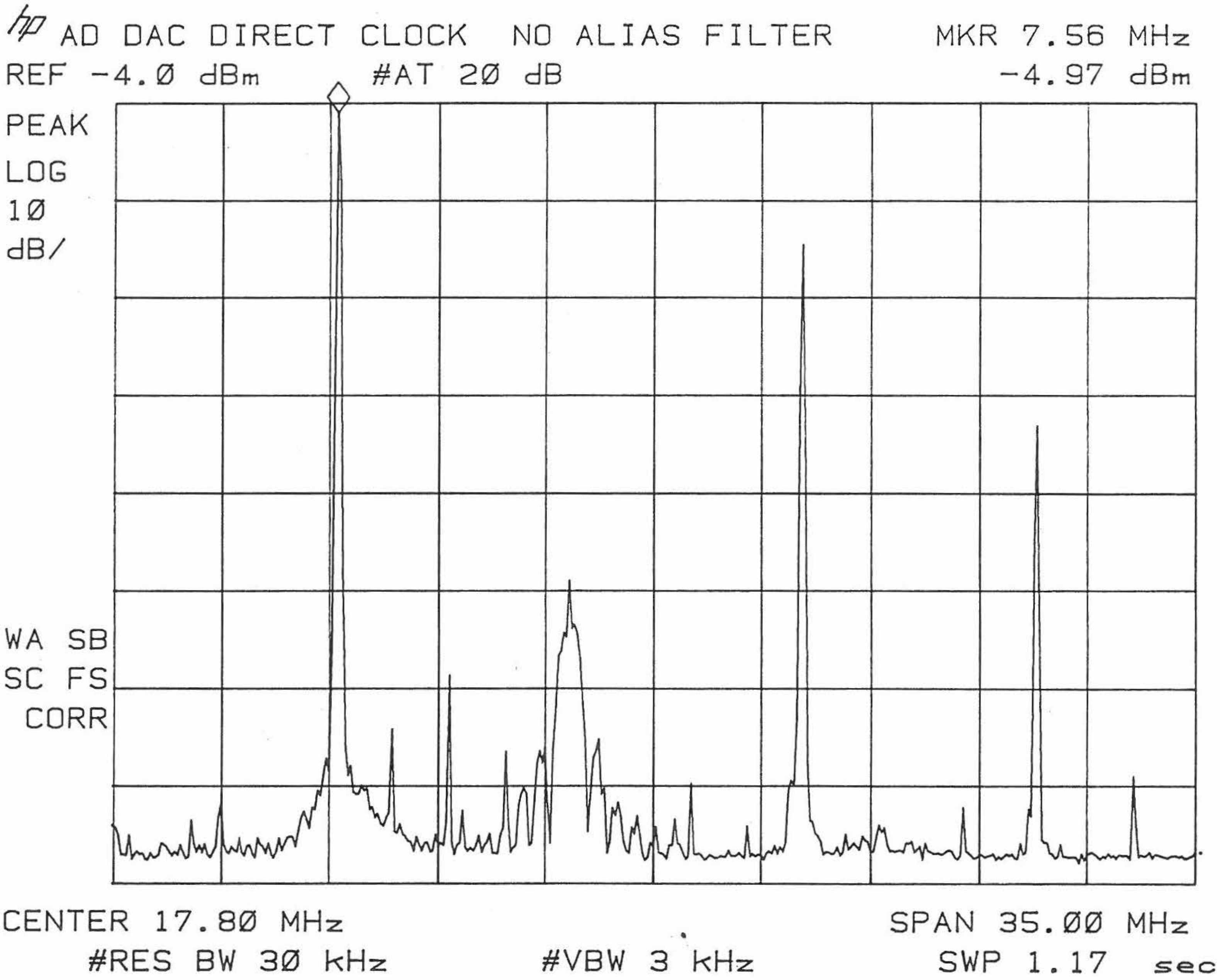




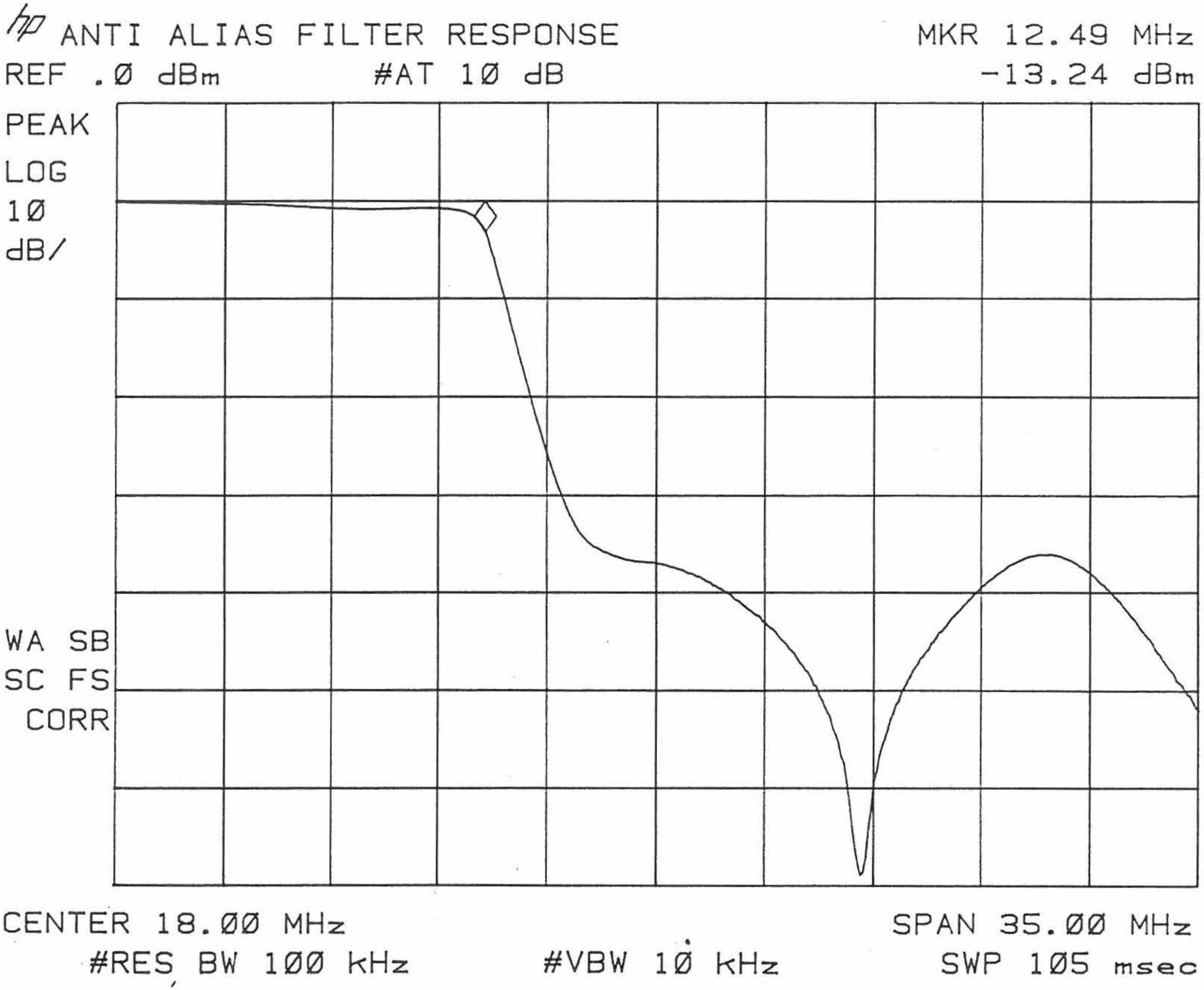
Appendix B4.3



Appendix B4.4

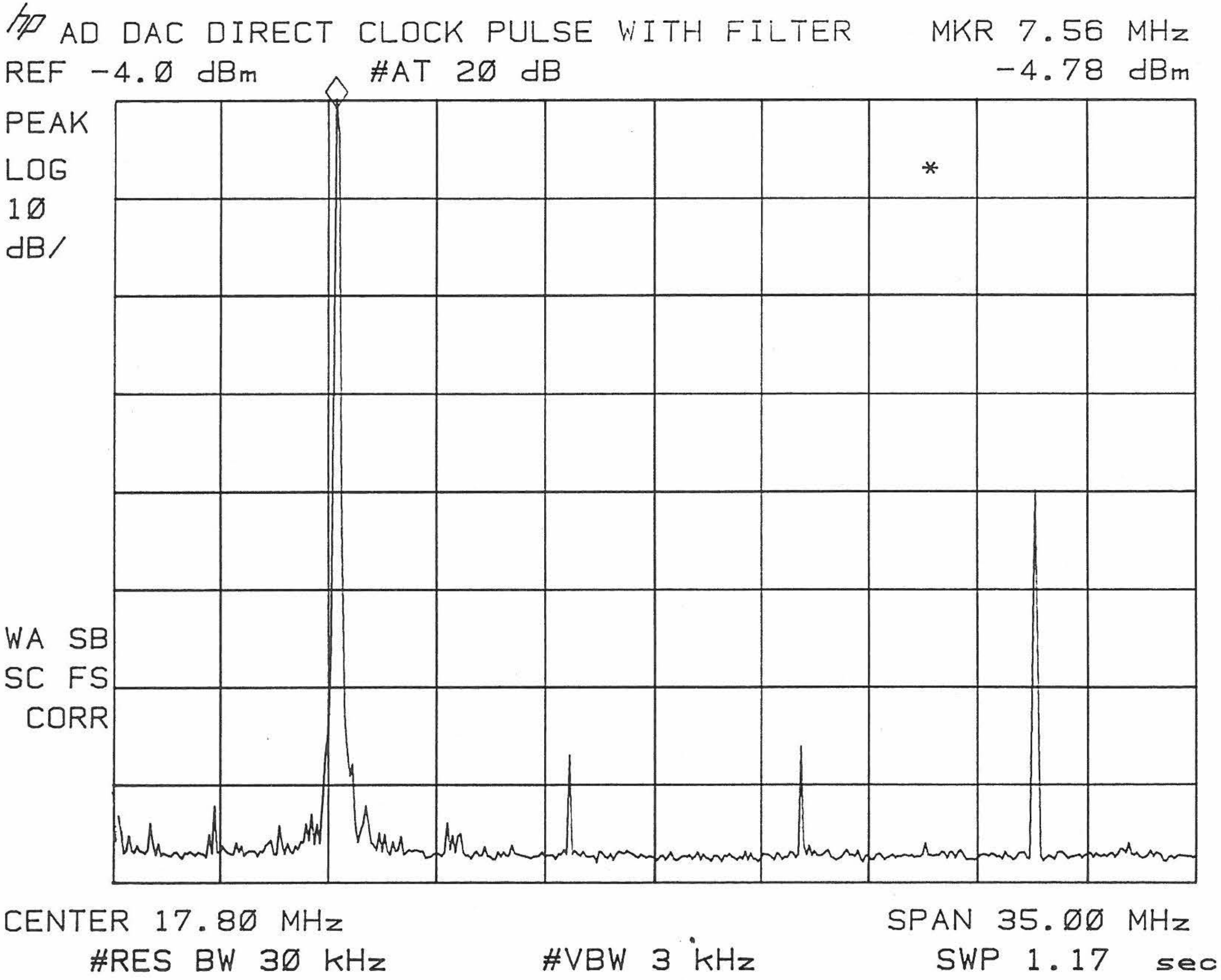


Appendix B5.1

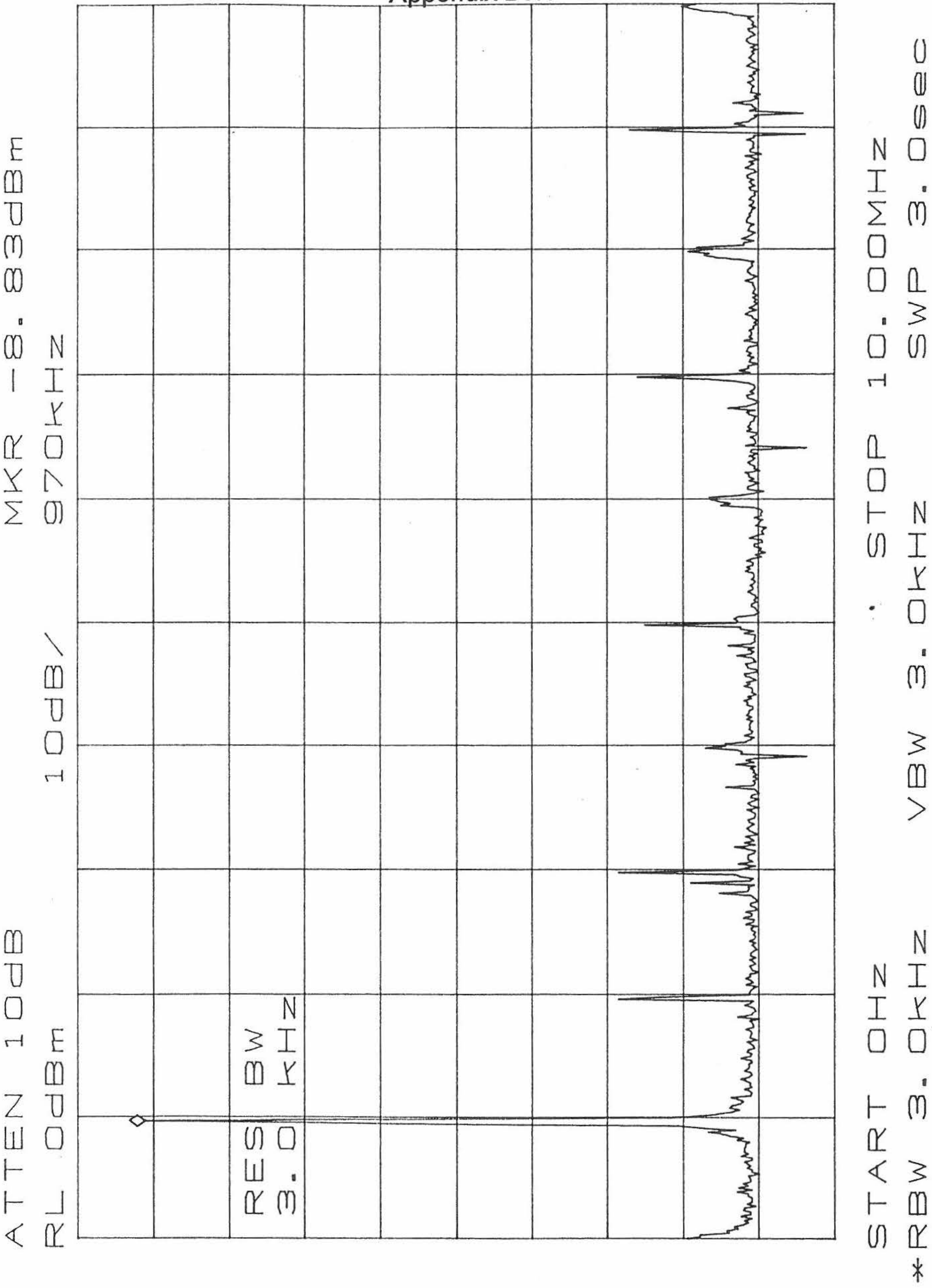


Appendix B5.2

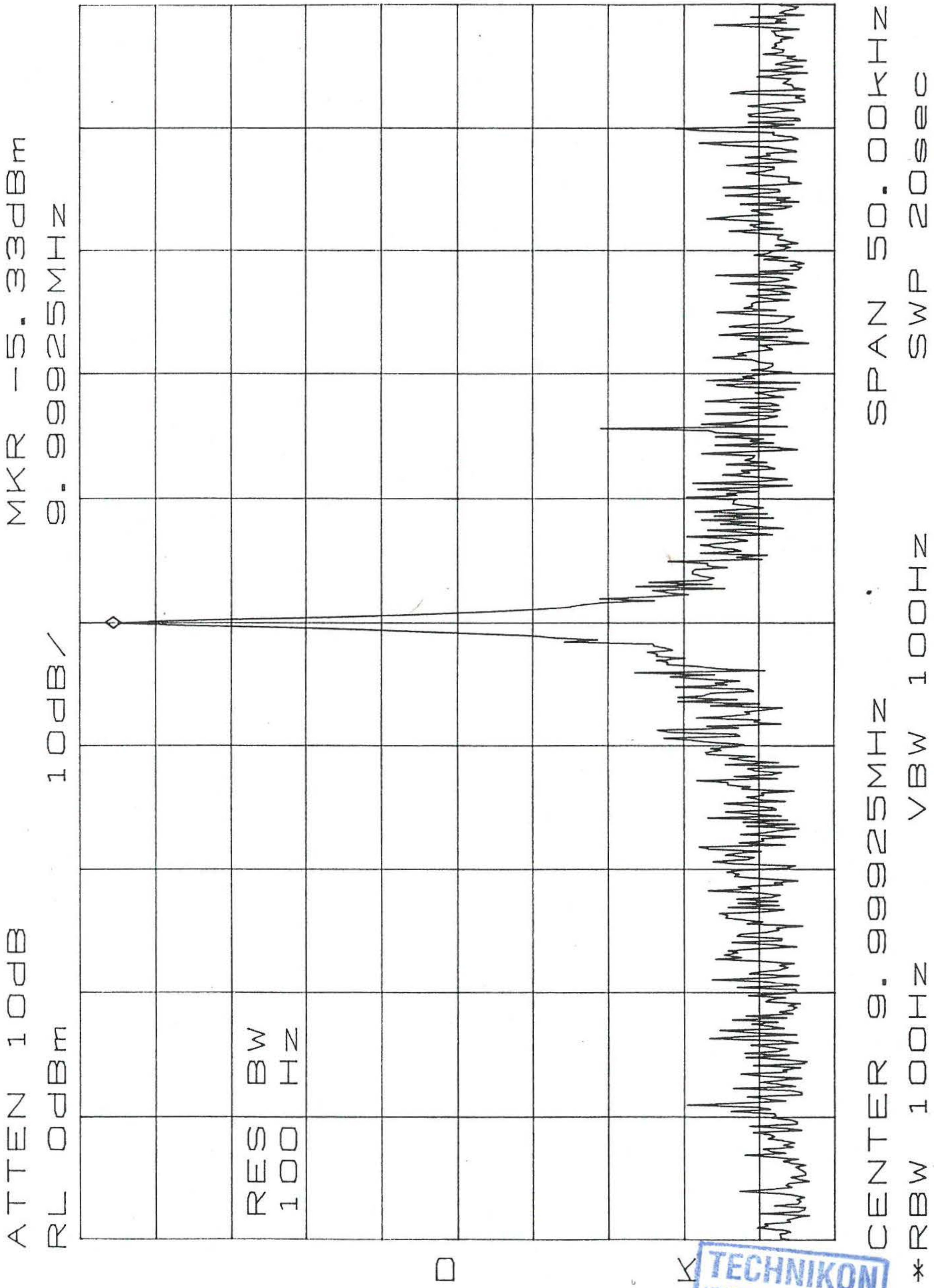
Appendix B5.3



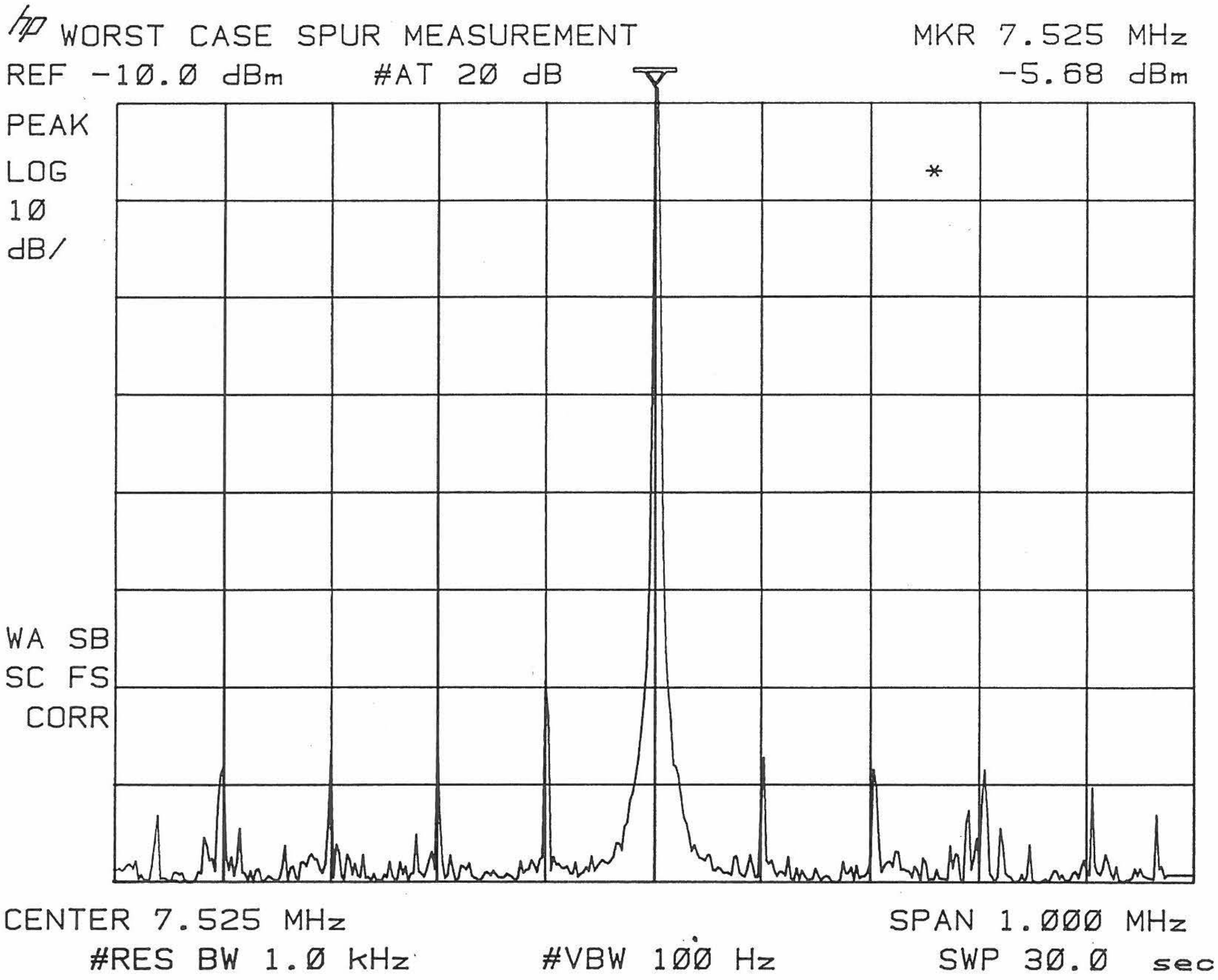
Appendix B6.1



Appendix B6.2



Appendix B6.3



APPENDIX C

Software source code listing

```
/****************************************************************************/
/* Program : Software to control HSP 45102 DDS with DS5000 Controller*/
/* Author  : Bernie van der Walt                                          */
/****************************************************************************/

#pragma DEBUG CODE                /* pragma lines can contain state C51     */
                                  /* command line directives                */

#pragma NOLC                      /* no list file made in compiling         */


#include <DS5000.h>               /* special function register declarations */


#include <stdio.h>                /* prototype declarations for I/O function*/


#include <stdlib.h>               /* general functions library              */


#include <string.h>               /* handles string functions               */

/****************************************************************************/
/**************************PIN ASSIGNMENTS***********************************/
/****************************************************************************/

sfr  PORT1  = 0x90;               /* Port 1 Assigned for keypad and LCD     */
sfr  PORT0  = 0x80;               /* Port 0 LCD                             */
sbit DAT_AV = 0xA6;               /* P2.6 Data Available signal from 74922  */
sbit KEY_EN = 0xA7;               /* P2.7 Output Enable signal to 74922     */
sbit E_LCD  = 0xB4;               /* P3.4 Enable pulse to LCD               */
sbit RW_LCD = 0xB5;               /* P3.5 Read/Write select to LCD          */
sbit RS_LCD = 0xB6;               /* P3.6 Register select to LCD            */
sbit SD     = 0xA5;               /* P2.5 Serial data to DDS - LSB first    */
sbit SCLK   = 0xB7;               /* P3.7 SCLK output to DDS                */
sbit SFTEN  = 0xA4;               /* P2.4 Shift Enable to DDS load register */
sbit LOAD   = 0xA3;               /* P2.3 LOAD Zero feedback in Phase Acc   */
sbit TXFR   = 0xA2;               /* P2.2 TXFR freq into Phase acc reg      */
sbit A_B    = 0xA1;               /* P2.1 FSK frequency select              */
sbit ENPHAC = 0xA0;               /* P2.0 Enable clocking of Phase accumulat*/
sbit ROE_ACT= 0xB3;               /* P3.3 Activity sense from ROE           */
sbit UP_DWN = 0xB2;               /* P3.2 Up or Down sense from ROE CCT     */

/****************************************************************************/
/***********************FUNCTION DECLARATIONS********************************/
/****************************************************************************/

void main_menu(void);             /* main menu display and choice           */
void menu_opt1(void);             /* makes up freq from keypad input        */
void menu_opt2(void);             /* Frequency Shift Keying option          */
void menu_opt3(void);             /* Sweep menu                             */
char keyboard(void);              /* converts keyvalue to ASCII             */
float freq_val(void);             /* array for frequency word               */
void float_to_str(float freq_LCD);/* converts floating point to string      */
```

```
char int_to_char(int int_byte);    /* changes integers to characters for LCD */
void delay(int time);              /* delay routine for general timing       */
void write_data(char ASCII_val);   /* writes data to LCD - byte at a time    */
void write_ins(int instr);         /* writes instruction to LCD              */
void write_str(char *word);        /* writes string to LCD                   */
void LCD_init(void);               /* initialization sequence for LCD        */
void LCD_add(add);                 /* select LCD start address               */
int  DDS_data(float freq_data);    /* converts freq in 32 bit binary stream  */
void DDS_ctrl(unsigned long PA);   /* controls DDS load sequence - send SD   */
char getkey();                     /* gets key input from Pc keyboard        */

/**********************************************************************/
/*******************************MAIN ROUTINE***************************/
/**********************************************************************/

main()

{
SCON  = 0x50;                    /* SCON:mode 1, 8-bit UART, enable rcvr     */
TMOD |= 0x20;                    /* TMOD:timer 1, mode 2, 8-bit reload       */
TH1   = 0xfD;                    /* TH1:reload value for 9600 baud           */
TR1   = 1;                       /* TR1:timer 1 run                          */
TI    = 1;                       /* TI: set TI to send first char of UART    */

LCD_init();                      /* Initialise LCD                           */
while(1){
      main_menu();
      }
}

/**********************************************************************/
/********************************MAIN MENU*****************************/
/**********************************************************************/

void main_menu(void)

{
char m;

ENPHAC=1;                           /* disable clocking of phase accumulator */
printf("\n\n--DDS SIGNAL GEN--\nSelect mode:\n1:SIG GEN 2:FSK 3:SWEEP\n");
write_ins(1);
LCD_add(128);
write_str("---DDS SIGNAL GEN---1:SIG GEN  2:FSK    ");
write_str("    Select mode:    3:SWEEP             ");
LCD_add(209);                                          /* sets cursor position*/

m = keyboard();
write_data(m);

switch (m)
      {
      case 49:{menu_opt1();
      break;}
      case 50:{menu_opt2();
      break;}
      case 51:{menu_opt3();
      break;}
      }
}




/*****************************************************************************/
/************************FREQUENCY LOAD ROUTINES*****************************/
/*****************************************************************************/

void menu_opt1(void)               /* Signal generator mode - Single freq    */

{
float freq;

write_ins(1);                                            /*clear LCD display*/
printf("\n\n--SIGNAL GENERATOR--(menu option 1 selected)\nFREQ:");
LCD_add(128);
write_str("--SIGNAL GENERATOR--FREQ:");
LCD_add(153);
freq=freq_val();
LCD_add(153); write_str("               ");
LCD_add(153);
printf("\n");

  if   (freq> 10.0)
      {
      write_ins(1);
      LCD_add(128); write_str(" * Limit 10 MHz!! * ");
      LCD_add(148); write_str("Freq above LPF limit");
      delay(8000);
      printf("\nLimit 10 MHz!!");
      write_ins(1);
      LCD_add(128); write_str("Exceeding LPF cutoff");
      LCD_add(192); write_str("frequency, test only");
      LCD_add(153);
      printf("\n");
      }

DDS_data(freq);
}
/*****************************************************************************/
void menu_opt2(void)                                         /* FSK mode      */

{
float freq1,freq2;
unsigned long PA1,PA2,TEMP_PA,N1,N2;
int t;
```

```
unsigned long bit_pos=1;                          /* checkbit shifted val */

write_ins(1);                                          /*clear LCD display*/
printf("\n\n--FSK Mode--\nFreq A:");
LCD_add(128);write_str("    ---FSK Mode---    ");
LCD_add(192);write_str("FREQ A: ");LCD_add(199);
freq1=freq_val(); write_str(" MHz");
delay(10000);
printf("\nFreq B:");
LCD_add(148);write_str("FREQ B:");LCD_add(155);
freq2=freq_val(); write_str(" MHz");
delay(10000);


      N1=(freq1 * 143.164139e6);
      PA1=(unsigned long)N1;                 /* converts float N1 to long int */
      N2=(freq2 * 143.164139e6);
      PA2=(unsigned long)N2;                 /* converts float N2 to long int */

     do        {
                DDS_ctrl(PA1);
                LCD_add(221);write_str("A ");
                printf("\nA");
                while(A_B){if ((DAT_AV)||(RI)) break;}
                DDS_ctrl(PA2);
                LCD_add(221);write_str(" B");
                printf("\nB");
                while(!A_B){if ((DAT_AV)||(RI)) break;}
                } while((!DAT_AV)&&(!RI));
}

/***************************************************************************/

void menu_opt3(void)                                     /* Sweep mode   */

{
float freq1,freq2,N,start_freq;
unsigned long PA;

write_ins(1);                                          /*clear LCD display*/
printf("\n\n--Sweep Mode--\nStart Freq:");
LCD_add(128);write_str("    --SWEEP MODE--    ");
LCD_add(192);write_str("START:");LCD_add(198);
freq1=freq_val(); write_str(" MHz");
delay(10000);
printf("\nStop Freq:");
LCD_add(148);write_str("STOP:");LCD_add(154);
freq2=freq_val(); write_str(" MHz");
delay(10000);
printf("\nSweeping....");
start_freq=freq1;

while((!DAT_AV)&&(!RI))
      {
        N=(freq1 * 143.164139e6);
        PA=(unsigned long)N;                      /* converts float N1 to long int */
        DDS_ctrl(PA);
        LCD_add(212);write_str("Sweeping....");
        freq1=freq1+0.001;

        if (freq1>=freq2) freq1=start_freq;
        }
}
/*****************************************************************************/
float freq_val()

{
char freq_word[8];
float freq=0;                                  /* freq value for DDS         */
int n=0;                                       /* n counter for freq array   */
char digit;                                    /* value of key pressed       */

do
   {
   write_ins(6);
   digit = keyboard();                         /* calls key input            */
   if(digit!=8)
     write_data (digit),
     freq_word[n]=digit,
     n++;

   if(digit==8)
      {
      n--;
      write_ins(4);                          /* decrement cursor no shift  */
      write_ins(16);
      }

   } while(n!=8);

freq=atof(freq_word);                          /* converts array to float    */

return(freq);

}

/*****************************************************************************/

int DDS_data(float freq_data) /* Calculates Phase acc word and controls ROE */

        {
        float N,step_size=0.000001;                        /* 1 Hz steps      */
        unsigned long PA;
        int key_val;


while(1) {
          float_to_str(freq_data);
      LCD_add(162);write_str(" MHz  ");
      LCD_add(153);
      N=(freq_data * 143.164139e6);
      PA=(unsigned long)N;            /* converts float N to long int */
      DDS_ctrl(PA);


while((!DAT_AV)&&(!RI)&&(!ROE_ACT)){}   /* waits in loop for updwn signals*/
   if (DAT_AV) {                                      /* keypad active*/
     KEY_EN=0,
     key_val=PORT1,key_val=key_val&15;

       if (key_val==3)freq_data=freq_data+step_size;

       if (key_val==7)freq_data=freq_data-step_size;

       if ((key_val==11)&&(step_size<0.100)) step_size=step_size*10,
 LCD_add(212),write_str("* Inc Step.."),delay(750),
 LCD_add(212),write_str("                    "),LCD_add(153);

       if ((key_val==11)&&(step_size>=0.100))
 LCD_add(212),write_str("* Maximum Stepsize! "),delay(4000),
 LCD_add(212),write_str("                    "),
 LCD_add(153);

       if ((key_val==15)&&(step_size>0.000001)) step_size=step_size/10,
 LCD_add(212),write_str("* Dec Step.."),delay(750),
 LCD_add(212),write_str("                    "), LCD_add(153);

       if ((key_val==15)&&(step_size<0.000002))
 LCD_add(212),write_str("* Minimum Stepsize! "),delay(4000),
 LCD_add(212),write_str("                    "),
 LCD_add(153);

       if (key_val==14) break;                   /* break to main_menu */
             }
   if (RI)       {                           /* pc active on RS232 port*/
       if (SBUF==85)freq_data=freq_data+step_size;          /* "U" on pc */
       if (SBUF==68)freq_data=freq_data-step_size;          /* "D" on pc */
       if (SBUF==81)  break;                                /* "Q" on pc */
       RI=0;   }                             /* resets RS232 active flag */

   if (ROE_ACT)  {                                /* ROE active signal */
       if (!UP_DWN)freq_data=freq_data+step_size;
       else freq_data=freq_data-step_size;
             }

   if (freq_data<=0) freq_data=0.0;
   if (freq_data>=30) freq_data=30;

      printf("\n");                             /* new line for pc display*/
      }
    }
```

```
/*****************************************************************************/

void DDS_ctrl(unsigned long PA)             /* DDS control for freq loading  */

{
int t;
unsigned long bit_pos=1;                    /* checkbit shifted val          */


      SFTEN=0;                              /* Enable Shift register         */

            for (t=0;t<32;t++)
            {
            bit_pos=bit_pos<<1;
            SCLK=0;
             if(PA&bit_pos)
               SD=1;
             else
               SD=0;
            SCLK=1;

            }
      SFTEN=1;                                     /* Disable Shift register */
      TXFR=0;                      /* Transfer from freq reg to Phase Acc reg*/
      ENPHAC=0;                                  /* Enable clocking of Phase Acc */
}

/*****************************************************************************/

void float_to_str(float freq_LCD_in)                   /* writes freq to LCD */

{
int n;
int byte;
char freq_LCD[10];
    double z=10;

printf("freq: %f",freq_LCD_in);

for (n=0;n<9;n++)
       {
       byte=(freq_LCD_in/z);
       freq_LCD[n]=int_to_char(byte);
        if(n==1)n++,freq_LCD[n]=46;
        if((n==0)&&byte==0)freq_LCD[0]=32;
        if(byte)freq_LCD_in=(freq_LCD_in-(z*byte));
       z=(z/10);
       }
write_str(freq_LCD);
LCD_add(153);
}

/*****************************************************************************/
```

```
char int_to_char(int int_byte)     /* converts integer to character for LCD */

{
char char_byte;

switch(int_byte)
                  {
                  case 1: {char_byte=49;break;}              /*1*/
                  case 2: {char_byte=50;break;}              /*2*/
                  case 3: {char_byte=51;break;}              /*3*/
                  case 4: {char_byte=52;break;}              /*4*/
                  case 5: {char_byte=53;break;}              /*5*/
                  case 6: {char_byte=54;break;}              /*6*/
                  case 7: {char_byte=55;break;}              /*7*/
                  case 8: {char_byte=56;break;}              /*8*/
                  case 9: {char_byte=57;break;}              /*9*/
                  case 0: {char_byte=48;break;}              /*0*/
                  }
return(char_byte);
}

/****************************************************************************/
/******************************KEYBOARD ROUTINES*****************************/
/****************************************************************************/

char keyboard()  /* reads in values from keypad and converts to ASCII values*/

{
int key_val;
char output;

while ((!DAT_AV)&&(!RI)) {}              /* infinite loop waits for DAT_AV=1*/

if(DAT_AV)
   do    {
      KEY_EN=0;                        /* enable 74922 output */
      key_val=PORT1,                   /* reads value at port1*/
      KEY_EN=1;                        /* disable 74922 output*/
      output = key_val&15;             /* mask upper 4 bits   */

      switch(output)                   /* Assigns ASCII values*/
                 {
                  case 0:   {output=49;break;}             /*1*/
                  case 1:   {output=50;break;}             /*2*/
                  case 2:   {output=51;break;}             /*3*/
                  case 4:   {output=52;break;}             /*4*/
                  case 5:   {output=53;break;}             /*5*/
                  case 6:   {output=54;break;}             /*6*/
                  case 8:   {output=55;break;}             /*7*/
                  case 9:   {output=56;break;}             /*8*/
                  case 10:  {output=57;break;}             /*9*/
                  case 12:  {output=46;break;}             /*.*/
                  case 13:  {output=48;break;}             /*0*/
                  default:  {output=48;break;}             /*0*/
```

```
                }


        } while(DAT_AV);

else output=getkey();                                /* pc key input*/

printf("%c",output);
return(output);
}

/*************************************************************************/

char getkey()                                /* Gets key input from Pc keyboard*/

{
char output;

output=SBUF;
RI=0;
        switch(output)                         /* Assigns ASCII values*/
                {
                 case 49:  {output=49;break;}           /*1*/
                 case 50:  {output=50;break;}           /*2*/
                 case 51:  {output=51;break;}           /*3*/
                 case 52:  {output=52;break;}           /*4*/
                 case 53:  {output=53;break;}           /*5*/
                 case 54:  {output=54;break;}           /*6*/
                 case 55:  {output=55;break;}           /*7*/
                 case 56:  {output=56;break;}           /*8*/
                 case 57:  {output=57;break;}           /*9*/
                 case 46:  {output=46;break;}           /*.*/
                 case 48:  {output=48;break;}           /*0*/
                 default:  {output=48;break;}           /*0*/
                 }

return(output);
}

/*************************************************************************/
/**********************LCD CONTROL AND LOAD ROUTINES**********************/
/*************************************************************************/

void write_data(char ASCII_val) /* writes byte at a time to LCD          */

            {
            RS_LCD = 1;                             /* select data mode    */
            RW_LCD = 0;                             /* select write        */
            E_LCD=1;                                /* enable pulse        */
            PORT0=ASCII_val;                        /* ASCII_val to PORT1  */
            delay(10);
            E_LCD  = 0;
            RW_LCD = 1;
            }
```

```
/*****************************************************************************/

void write_str(char *word)                    /* writes string to LCD        */

            {
            int length,                              /* length of string     */
                byte,                                /* byte out to LCD      */
                p;                                   /* pos counter in word  */

            length=strlen(word);                     /* determine str length */
            for (p=0;p<=length-1;p++)
                       {
                       byte=word[p];                 /* place pos val in byte*/
                       write_data(byte);             /* call write_data      */
                       }
            *word=("                   ");
            }
/*****************************************************************************/

void write_ins(int instr)                      /* Writes instruction to LCD */

            {
            RS_LCD=0;                                /* select instr mode    */
            RW_LCD=0;                                /* select write         */
            E_LCD=1;                                 /* enable pulse         */
            PORT0=instr;                             /* instr out to port 0  */
            delay(50);
            //printf("\nLCD_ins=%d",instr);
            E_LCD=0;
            RW_LCD=1;
            }
/*****************************************************************************/

void LCD_init(void)                    /* Initialisation string for the LCD */

            {
            delay(400);
            write_ins(48);                           /*func set 8 bits       */
            delay(200);
            write_ins(48);
            delay(200);
            write_ins(48);
            write_ins(56);                           /*func set 8bits, 5x7   */
            write_ins(8);                            /*LCD off               */
            write_ins(12);                           /*LCD on  (14 cursor on)*/
            write_ins(6);                            /*entry mode-increment, */
                                                     /*no shift              */
            write_ins(1);                            /*LCD clear             */
            write_ins(2);                            /*Cursor home           */
            }
/*****************************************************************************/

void LCD_add(int add)                    /* Address selection on LCD Matrix */
```

```
            {
            RS_LCD=0;                               /* select ins           */
            RW_LCD=0;                               /* select write         */
            E_LCD=1;                                /* enable pulse         */
            delay(50);
            PORT0=(add);                            /* Start address is 128 */
            E_LCD=0;
            RW_LCD=1;
            }
/****************************************************************************/

void delay(int time)                                         /* Delay timer */

            {
            long int x;                             /* x = timer variable   */
            for (x=0;x<time;x++);
            }

/****************************************************************************/
/****************************************************************************/
```

REFERENCES

1. **Analog Devices.** 1993: *CMOS DDS Modulator AD7008*, Analog Devices, Norwood MA.

2. **Analog Devices.** 1993: *AD9713B 12-Bit, 100 MSPS D/A Converters*, Analog Devices, Norwood MA.

3. **ARRL Handbook.** 1993: *The ARRL Handbook for the Radio Amateur - 70th Edition*, American Radio Relay League, Newington CT.

4. **Dallas Semiconductor.** 1992: *DS5000 SOFT MICROCONTROLLER* , Dallas, Texas.

5. **Gallant, John A.** 1989: Devices refine the art of frequency synthesis, *EDN*, November 1989, pp 95 - 104.

6. **Harris.** 1991: *HSP45102 12 Bit Numerically Controlled Oscillator* , September 1991, pp 1 - 10.

7. **Harris.** 1993: *HI565A High Speed Monolithic D/A Converter with Reference*, December 1993, pp 8-42 to 8-48.

8. **Hickman, I.** 1992: Direct Digital Synthesis, *Electronics World & Wireless World*, August 1992, pp 630 - 634.

9. **Ifeachor, E.C. & Jervis, B.W.** 1993: *Digital Signal Processing : A Practical Approach*, Addison Wesley Publishing Company, Workingham.

10. **Jauch.** 1993: *Jauch TTL-type Crystal Oscillator Datasheet*

11. **Qualcomm.** 1993: *Q2334 Dual Direct Digital Synthesizer*, Qualcomm Inc., San Diego CA, pp 5 - 18.

12. **Stanford Telecom.** 1990: Alias and Spurious Responses in DDS systems, *Application Note 102 Edition 2*, Stanford Telecom, Sunnyvale CA.

13. **Stanford Telecom.** 1992: Alias and Spurious Responses in DDS systems, *Application Note 102 Edition 7*, Stanford Telecom, Sunnyvale CA.

14. **Sqiteq electronics Inc.** 1988: 1200 to 2400 MHz Frequency Synthesizer with Direct-Digital Synthesis/Phase-Lock Loop Combination, Microwave Journal February 1988, 194-195.

15. **Zavrel, Robert J. Jr.** 1988: Digital Modulation Using the NCMO, *RF Design*, March 1988, pp 27 - 32.

16. **Zavrel, Robert J. Jr.** 1993: DIRECT DIGITAL SYNTHESIZERS (DDS), *ARRL Handbook*, 1993, pp 10-17 to 10-23.

17. **Zavrel, Robert J. Jr. and Mc Cune, Earl W Jr.** 1988: Low Spurious Techniques & Measurements for DDS Systems, Research Report, Proceedings of RF Expo East, *DIGITAL RF SOLUTIONS CORPORATION*, pp 1 - 12.

18. **Zavrel, Robert J. Jr. and Edwards, Gwyn.** 1990: *The DDS Handbook: Second Edition*, Stanford Telecom, Santa Clara CA.

ADDITIONAL REFERENCES

(Consulted but not referred to)

1. **Bramble, Albert L.** 1981: Direct Digital Frequency Synthesis, *Proc. 35th Ann. Freq. Control Symposium, USAERADCOM*, Ft. Monmouth, NJ, 07703, May 1981.

2. **Brynard, D.** 1992: Direkte Digitale Frekwensie Sintese. *Radio ZS* , Februarie 1992, pp 10 - 11.

3. **Dawson, J.H.J. 1981**: Direct Digital Frequency Synthesizer, *Wireless World*, December 1981.

4. **Edwin, A.** 1990: Direct Digital Synthesis Applications. *Microwave Journal,* January 1990, pp 149 - 151.

5. **Hack, Thomas.** 1992: An Improved FM Exciter Using Direct Digital Synthesis, *RF Design* , March 1992.

6. **Hawksford, M.O.J.** 1989: Chaos, Oversampling and Noise Shaping in Digital-to-Analog Conversion, *J Audio Eng Soc. Volume 37, No 12* , December 1989.

7. **Henry, T; Nicholas III; and Henry Samueli.** 1987: Analysis of the output spectrum of direct digital synthesisers in the presence of phase-accumulator truncation, *41st Annual Frequency Control Symposium*, 1987, IEEE.

8. **Pesec, John.** 1992: DDS Techniques Assist Exotic Waveform Generation, *Microwaves & RF*, September 1992.

9. **Shen, E.K; Eden R.C; Moacanin, M.A; & Lee, F.S.** 1985: Monolithic GaAs Variable Modulus divider for Frequency Synthesis Applications. *Microwave Journal*, February 1988, pp 194 - 195.

10. **Stanford Telecommunications.**1992: DDS Source Incorporates High-speed DAC, *Microwaves and RF*, July 1992.

11. **Takura Kida and Keiichi Akagawa.** 1989: Sampling Theorems Using Nonuniform Sampling Points and Interpolation Functions with Band-limited Continuos Spectra, *ISSN1042-0967/89/0006-0059$7.50/0, Electronics and Communications in Japan, Part 3, Vol. 72, No.6*, 1989.

12. **Wheatley III, C.E.** 1981: Spurious suppression in direct digital synthesizers, *Proc 35th Ann. Freq. Control Symposium, USAERADCOM* , Ft. Monmouth, NJ, 07703, May 1981.

13. **Zavrel, Robert J Jr.** 1991: HDTV and NTSC Transmission Using Direct Digital Synthesis. *SMPTE Journal* , December 1991, pp 961 - 962.