P.107

1 GHZ DIGITIZER FOR SPACE BASED LASER ALTIMETER

FINAL REPORT

16 October 1991

Prepared For:

NASA GODDARD Space Flight Center
Greenbelt, Md. 20771

Contract No.: NAS5-30626
Project Manager: Jack Bufton

Submitted By:
Amerasia Technology, Inc.
2248 Townsgate Road
Westlake Village, CA 91361

Principal Investigator
Edward J. Staples

TABLE OF CONTENTS

1.0 PHASE II OBJECTIVE

The overall objective of this Phase II Program is to develop, design, fabricate, and test a feasibility model of a low power 1 GHz Waveform Digitizer. The Digitizer is intended for analyzing data collected via a space based Laser Altimeter. It has a 6-bit resolution, and is equipped with a 1 GHz surface acoustic wave (SAW) oscillator, and a random access buffer memory to interface with the 8-bit parallel bus of the altimeter system computer. Low power consumption is obtained by cutting off the power supply during the absence of data from the altimeter, thus lowering the duty cycle of power utilization.

The following technical objectives were specified for the implementation of the 1 GHz Waveform Digitizer feasibility model.

Functions:
(1) Perform analog-to-digital conversion
(2) Transfer digital data to system computer via a random access memory and an 8-bit parallel bus (DMA)
(3) Provide a 1 GHz system timing clock

Inputs:

(1) Signal bandwidth:	DC to 350 MHz
(2) Signal pulsewidth:	4 to 10 Nanoseconds
(3) Repetition frequency:	40 Hz
(4) Input impedance:	50 Ohms, nominal

Outputs:

(1) Sampling rate:	1 Gigasamples/second
(2) Resolution:	6 bits
(3) Clock frequency:	1 Ghz
(4) Number of samples:	128/second

Power supply:

(1) Average power consumption:	1.5 Watts
(2) Duty cycle:	2.5 %
(3) Available supply voltages:	5V, -5.2V, -2V, and 12V

2.0 PROGRAM SUMMARY

2.1 Initial Technical Approach

A system diagram of the initial approach is shown in Figure 2-1. Two Tektronix flash 8-bit A/D-converters (TKAD10C) were selected to digitize the input signal. Each is capable of 500 Megasamples/second operation. By paralleling these converters an effective 1 Gigasample processor is obtained. The converters have 7-bit accuracy, as specified by its manufacturer. The digitized data can be stored in 32 RAMs, each being an 8-bit 30 MHz device. Alternatively, eight 4-bit 250 MHz RAMs were considered. Write and Read activities in the RAMs are controlled by an address counter as shown. Sampled data is transferred to the system computer via DMA type bus interface. Timing is derived from a 1 GHz SAW oscillator and distributed to the converters counters, and computer interface bus.

2.2 Technical Problems encountered and Possible Solutions

It was found later that the total set-up and hold time of the high speed memories (RAM) is equal to the access time (4 nanoseconds), hence there is no allowance for any timing errors during acquisition at full speed. The following solutions were considered to overcome this problem:

(1) Replace the RAMS with high speed multistage (128) shift registers utilizing gate array technology
(2) Use 16 RAMS in place of the original 8, allowing for half speed clocking.
(3) Incorporate hybrid memory technology.

The first approach was investigated and found to be technically risky and not cost effective, due to the associated nonrecurring engineering cost. The second approach would increase the systems parts count, resulting in higher power consumption and reduced reliability. An existing hybrid memory and data sampling device (AN1000H) was found that is useable and met the technical requirements. This device is manufactured by Analytek and used in their Gigasample data acquisition systems. Therefore, there is minimal technical risk involved with the application of this device. In addition, the devices are readily available.

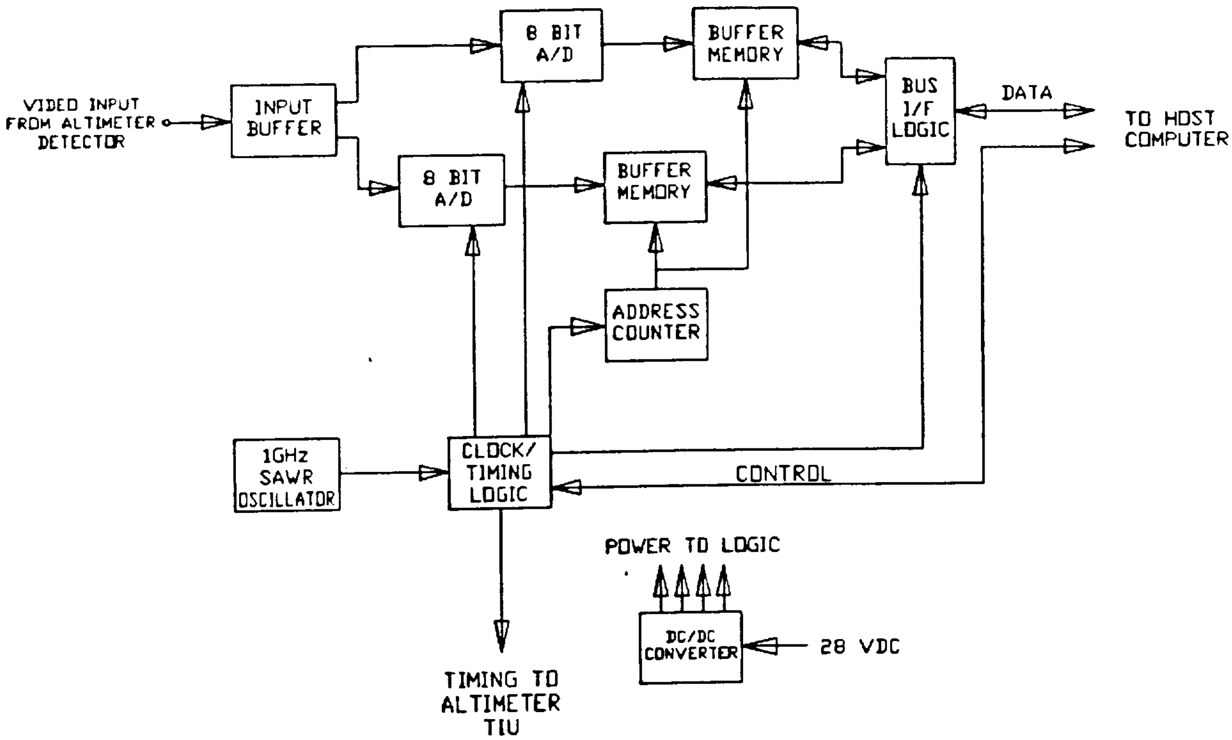


Figure 2-1, 1 Gigasample/sec. Digitizer System Diagram

2.3 Selected Technical Approach

A system diagram of the selected technical solution is shown in Figure 2-2. In this approach fast analog storage is provided by the AN1000H hybrids, which is followed by a slow readback via an A/D Converter at a moderate speed. The analog memory is made up of 4 hybrids. Each hybrid consists of four channels containing 256 randomly addressable charge storage capacitors capable of storing an analog voltage of ±2 volts with a resolution of ±1 millivolt. Each channel can operate at 62.5 Megasamples/second. By delaying the data and clock signals appropriately the system can be designed for 16 channels, equally spaced within one 16 nanosecond window, giving 1 nanosecond resolution. Fast sample/hold presamplers on each channel insure adequate aperture parameters to make the acquired signal meaningful at this rate.

Initially the input signal was delayed in increments of one nanosecond via a 16-tap delay line. This approach was not satisfactory because of differences in losses among multiple taps, although small between two adjacent taps, are excessively large thus deteriorating the acquisition accuracy of the digitizer. Moreover, stripline implementation of the tapped delay line is adversely effected by temperature due to expansion and contraction. This problem was resolved by offsetting the acquisition instants of each channel of the hybrids by one nanosecond, respectively.

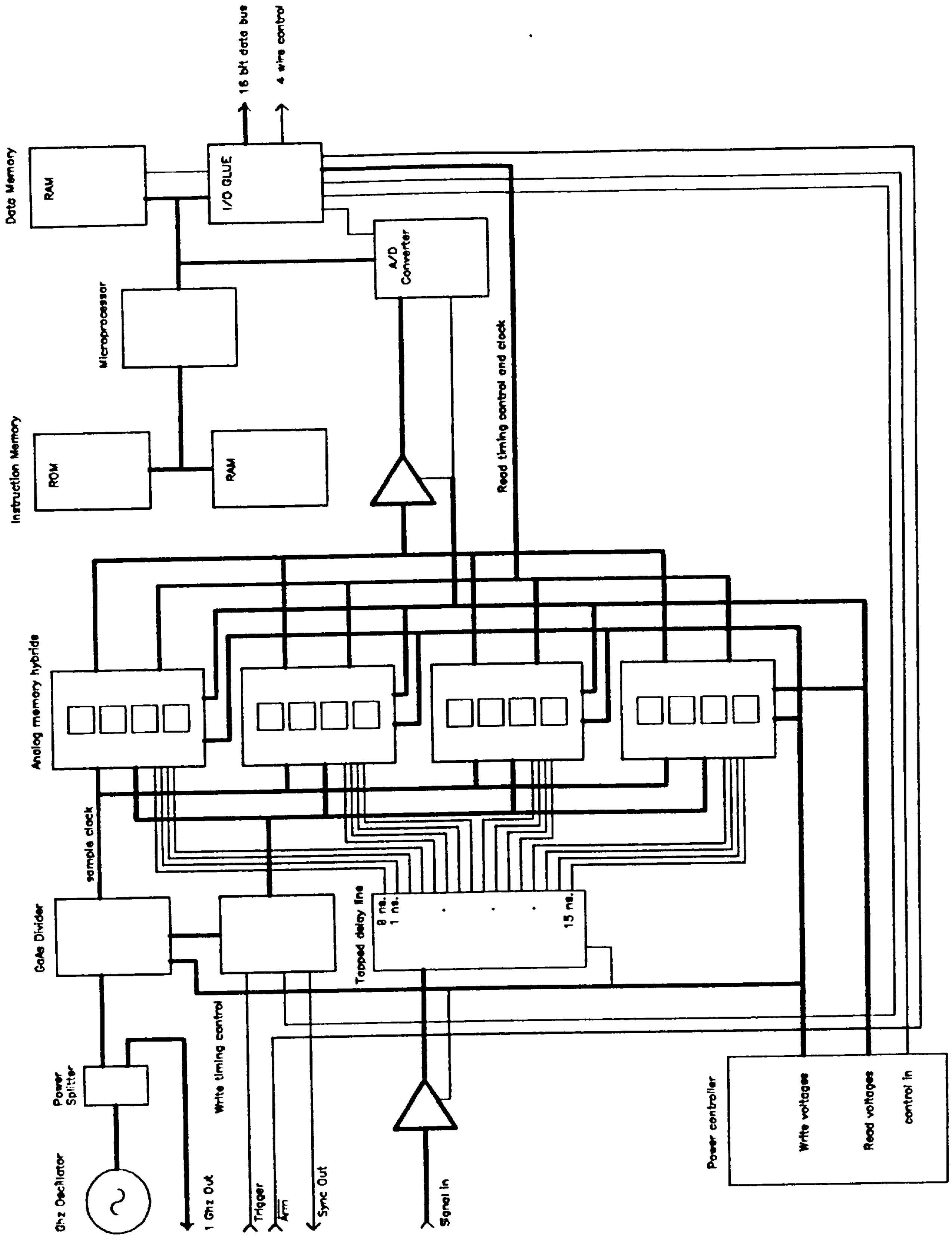


Figure 2-2, Lola A/D Converter Block Diagram

3.0 WORK ACCOMPLISHED

3.1 Summary

During the Phase II development program a feasibility model of the 1 GHz Digitizer was designed, fabricated and tested. Hardware system testing was not fully completed due to limited funding. Preliminary testing of the A/D function of the hardware demonstrated satisfactory results. A software package for data processing and interface with the host computer was developed but not tested. Source code listing of the software is attached in Appendix 1.

The 1 GHz SAW clock required for system timing is included, and a timing interface circuit The 1 GHz SAW clock required for system timing is included, and a timing interface circuit for clock distribution was designed and built as part of the hardware printed circuit board. Average power consumption is within the specified limit, while power up and power down timing were satisfactorily tested.

As mentioned earlier, initially the design was based on utilizing Tektronix's flash A/D converters (TKAD10C). For this purpose a clock distribution circuit was designed, the results of which are shown in Appendix 2. The 1GHz oscillator which was acquired for this purpose is also useable in the final design.

The hybrid devices (AN1000H) used in the final design were characterized. For this purpose a test circuit was designed and built. A schematic diagram showing the test circuit and the printed circuit board are shown in Figures 3-1 and 3-2. The results are shown in Appendix 3. During this effort one hybrid, containing 4 channels, was tested by characterizing its transfer function and input-output signal waveforms from 150 KHz to 1 GHz. Each channel of the hybrid contains 256 cells. Each of these cells have variations in gain and offset, consequently the output (sampled data) waveform becomes noisy. Corrections were made and the accompanying result shown, where the waveform is considerably cleaner.

A tapped delay line for the purpose of distributing the input signals with one nanosecond offsets to all the 16 channels of the hybrids was designed and built. This implementation is shown in Figure 3-3 and 3-4; it is basically a transmission line with 16 one nanosecond taps, constructed on epoxy fiberglass (G-10) material utilizing microstrip technique. The result of this implementation was not satisfactory due to dielectric and copper losses. A second version was built utilizing duroid printed circuit board, as shown in Figure 3-5. This implementation was also not satisfactory. Finally the approach was abandoned and this problem resolved by staggering the timing of each channel by 1 nanosecond. The test results of the delay line implementations are shown in Appendix 4.

In what follows the development of the 1 GHz Digitizer utilizing the AN1000H hybrid chips are described. It includes the characterization of these chips prior to hardware design, design of the Digitizer circuit, and software for control and interface with the system computer.

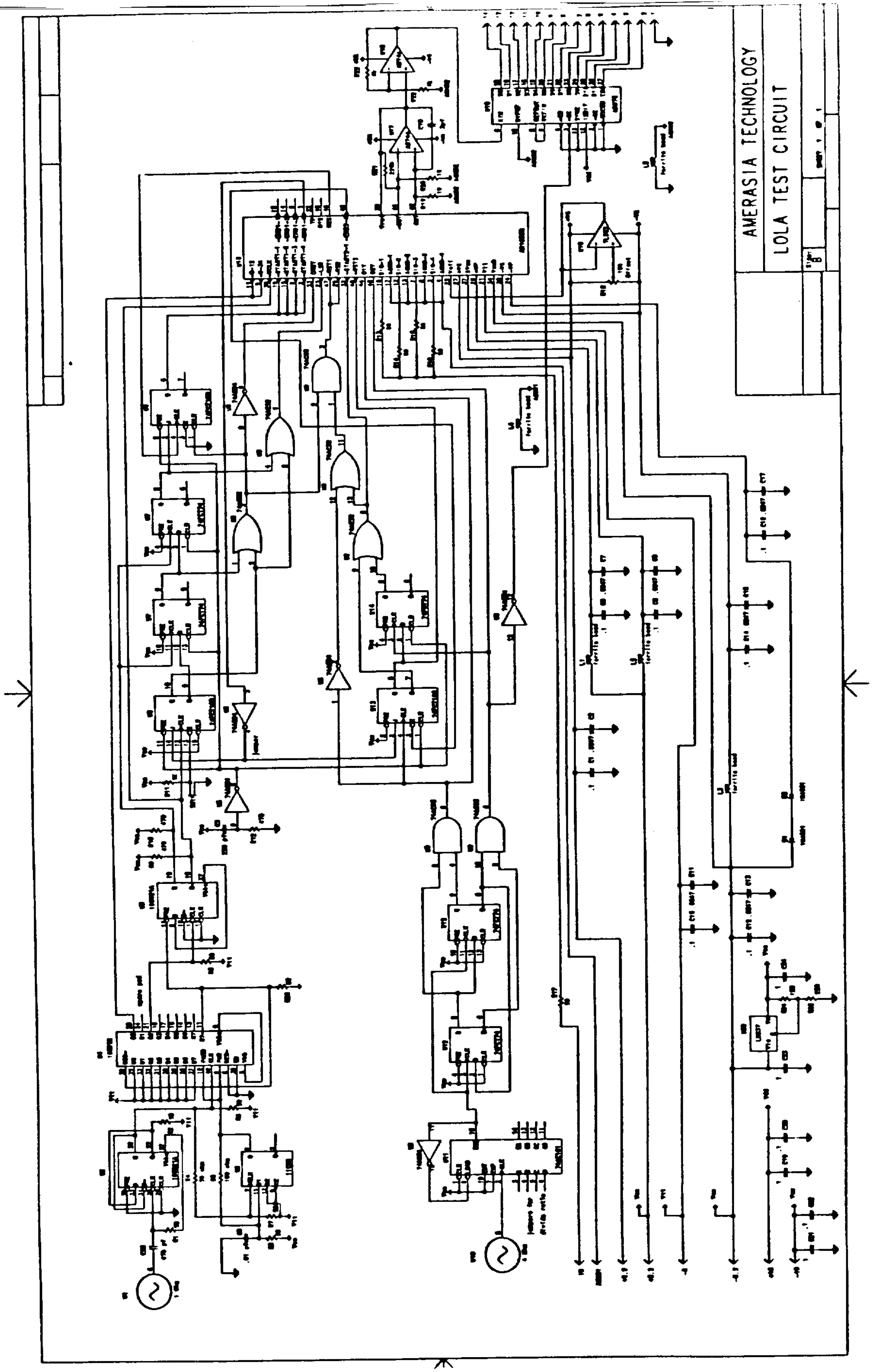


Figure 3-1, Lola Test Circuit

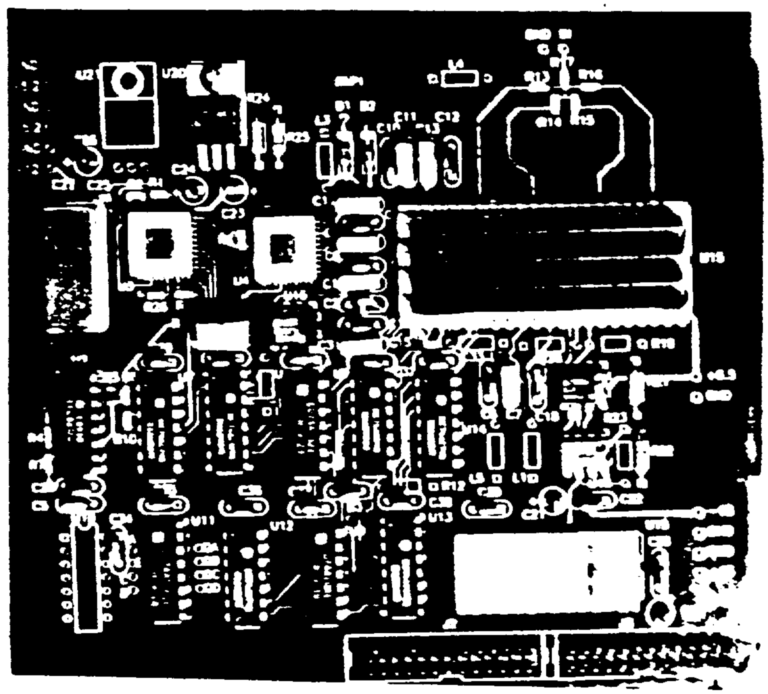


Figure 3-2, Test Circuit Layout

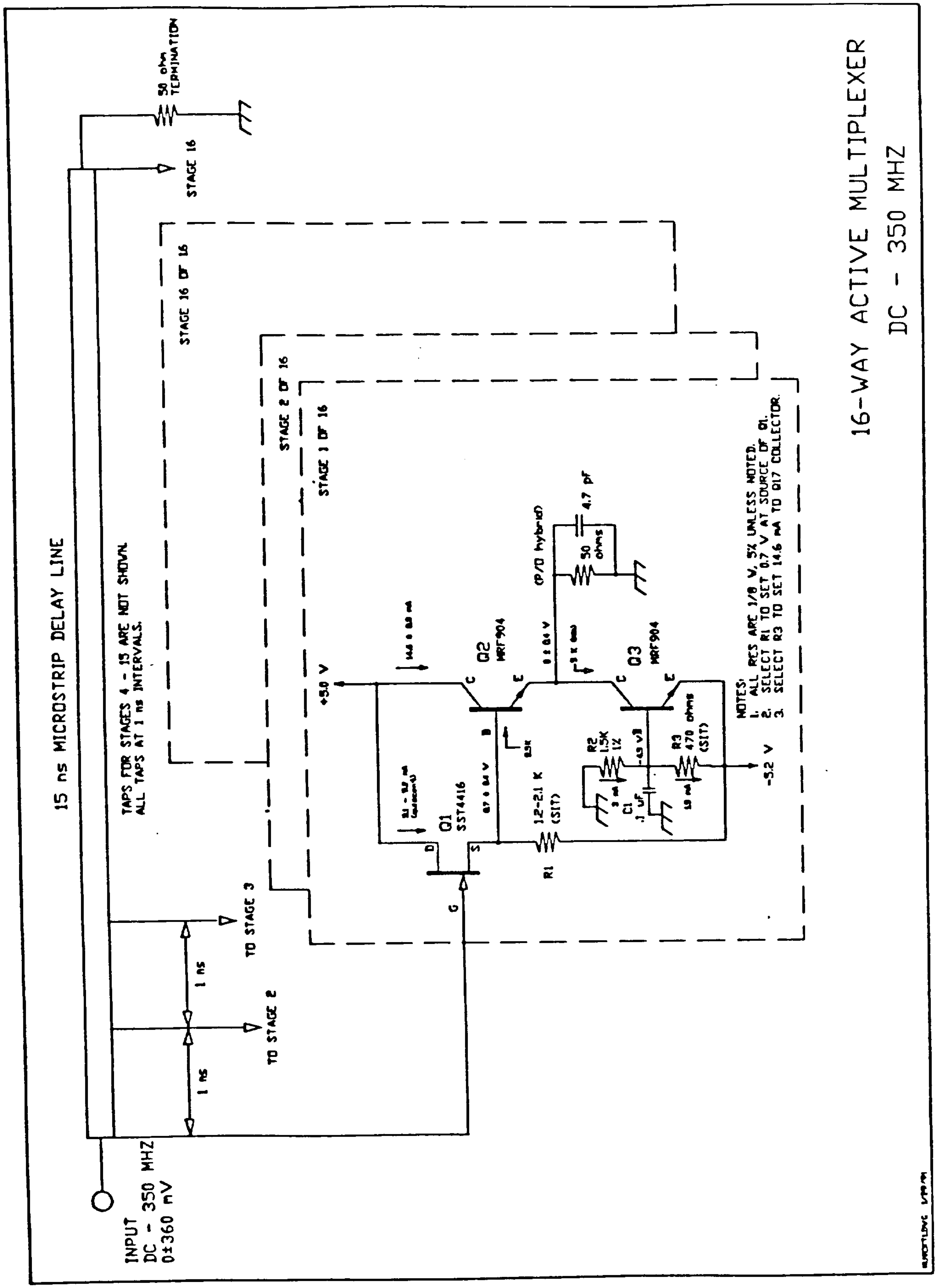


Figure 3-3, 16-Way Active Multiplexer

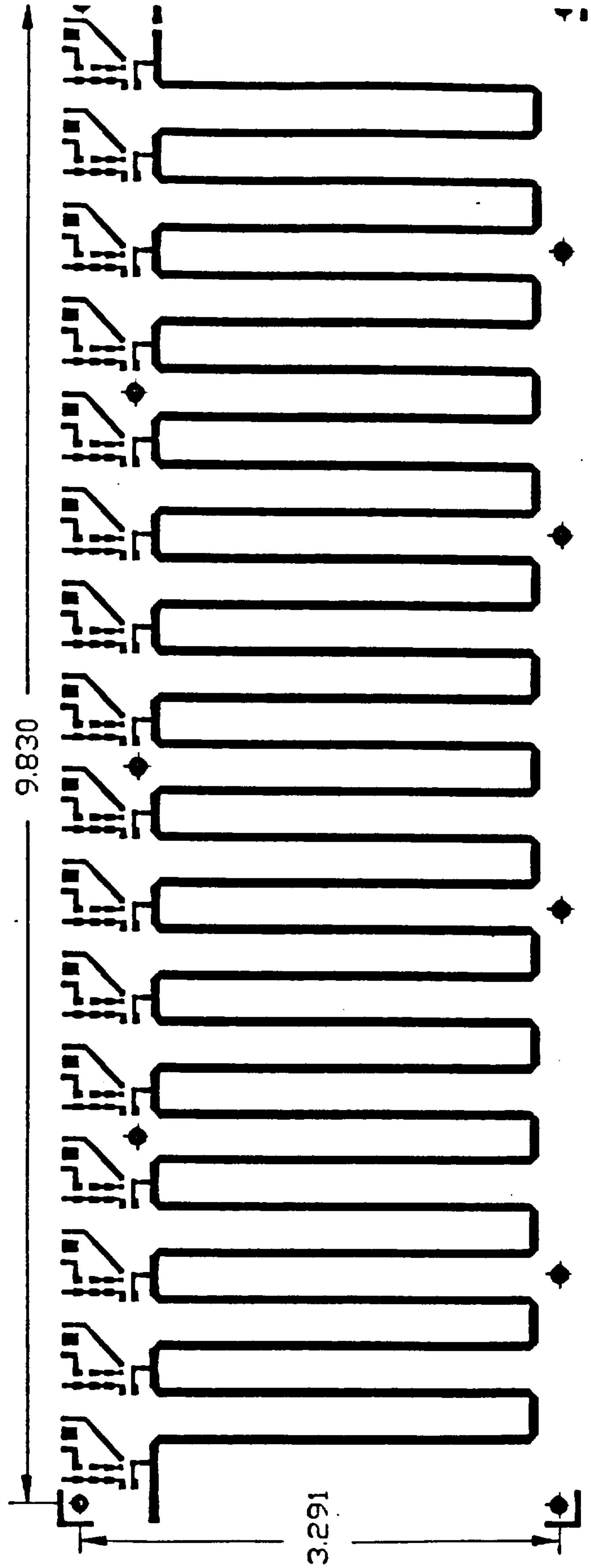


Figure 3-4, Microstrip Delay Line with pads for JFET mounting.

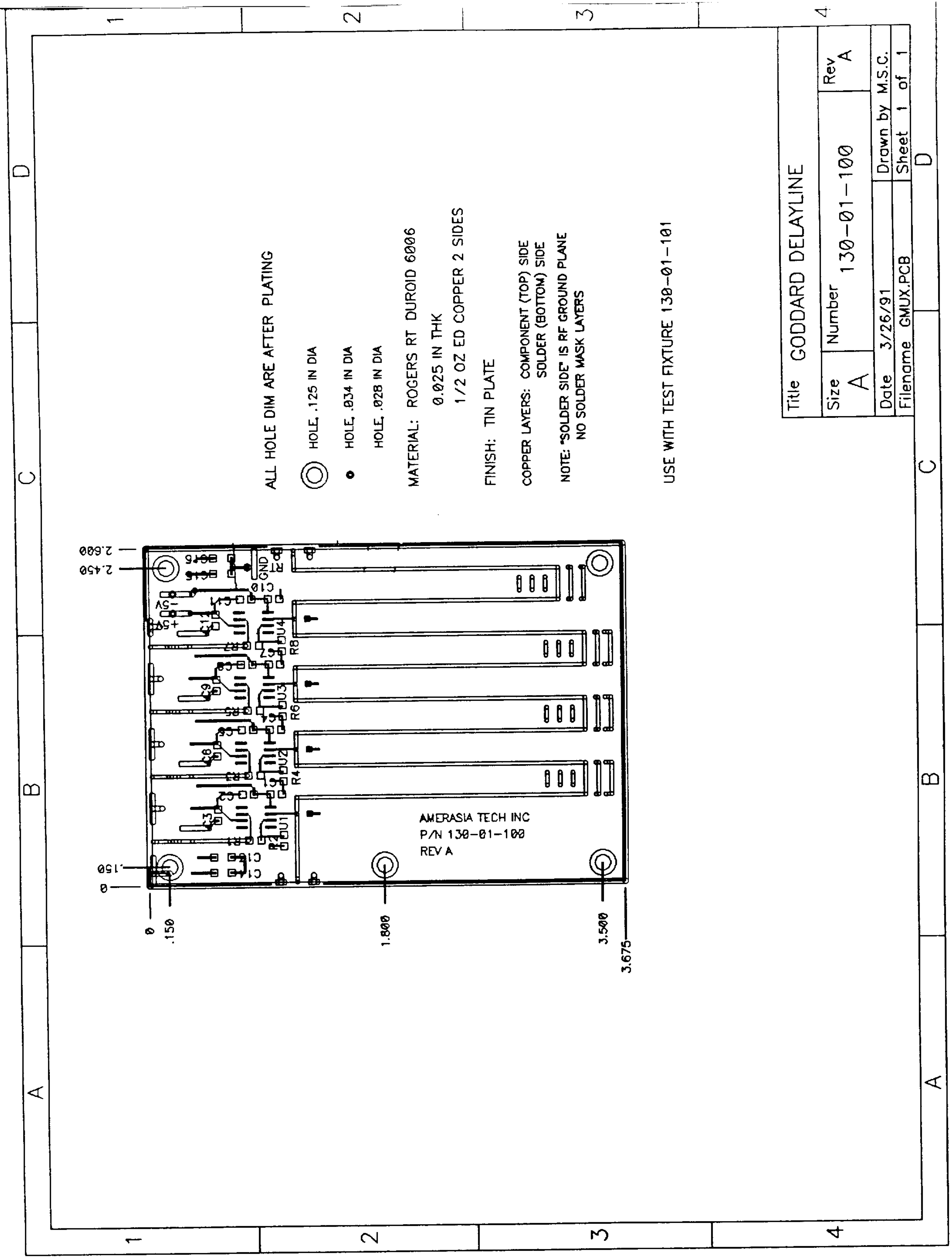


Figure 3-5, Goddard Delayline

3.2 Characterization of the AN1000H Hybrid

3.2.1 Design of the Test Circuit

The test circuit is shown in Figure 3-1 and 3-2. Since the objective is to characterize the AN1000H hybrid analog memory module, the design includes only one chip. Each hybrid has four inputs; in the 1 Gigasample Digitizer four hybrids are used and data is acquired via a 16-way multiplexer. Multiplexing is accomplished by a combination of delays in the timing and data paths such that the data latched by each of the inputs is staggered by one nanosecond referenced to time of actual occurrence. In order to achieve this, the presamplers in the hybrids must be fast enough to accurately take their samples. In addition the internal delay times must be known and included in the overall timing delay consideration.

The sampling window and timing is determined accurately in a single hybrid. The sampling waveforms should be the same as if it were one of four hybrids in the final system. The delays should be identical or at one nanosecond intervals so as to represent one or more noncontiguous repetitive samples out of the full set. Identical timing was chosen since it would provide the easiest method of determining channel to channel variations in both timing and amplitude.

In this test circuit on-board microprocessor is not included. The final circuit, however, will be controlled by a combination of microprocessor and dedicated logic, which will also unload and correct the data. Part of our objective is to determine the necessary corrections, therefore raw data is being collected. The on-board A/D converter is connected to a general purpose parallel port on an MS-DOS computer where simple programs in BASIC or C can be used to unload and save the data for further analysis. Analysis will include gain, offset and timing errors for each channel, offset and gain, offset and linearity errors for each memory cell, as well as input frequency versus amplitude errors for each memory cell, to check the sampling accuracy.

The power and control circuitry is not needed in the test board. It has been designed with adequate power and heat dissipation capacity for 100% duty cycle. Power down circuitry will be added later to determine its effect on performance and to determine minimum turn-on and turn-off times.

Since only four channels are being driven, a resistive divider is used to provide adequate impedance match. A signal function generator is used as a test source.

Referring to the circuit in Figure 3-1, several power voltages are required and a number of timing signals as shown in Figure 3-6. The hybrid needs Vtt for termination of the ECL inputs, Vsub for substrate biasing and Vofs as a fixed DC offset to the signal paths.

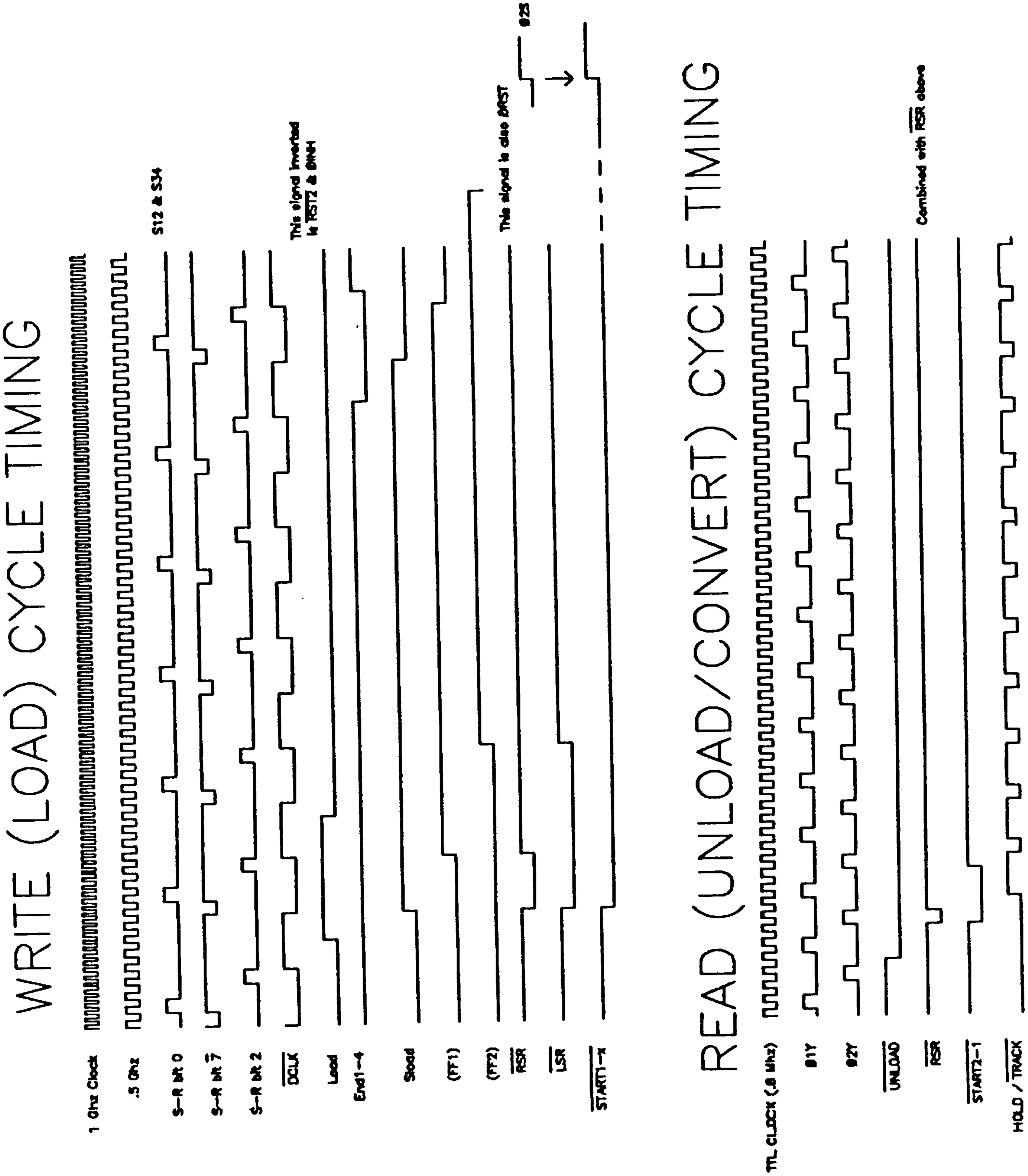


Figure 3-6, Timing Diagram

Voltages needed for sensitive analog portions of the circuit are isolated from noise of the digital circuitry by ferrite beads and bypass capacitors. There are two analog grounds, one at the input of the hybrid and the other between the hybrid and the A/D converter. Isolated analog sections of each plane have been created by partitioning the planes.

The timing signals are summarized in the timing diagram shown in Figure 3-6. The AN1000H collects samples during the high portion of the S12 an S34 signals. These samples are latched in the presampler on the falling edge of S12 an S34. The data in the presampler is transferred to a memory cell during DCLK high, and latched on the falling edge of the DCLK. The relationship between S12, S34 and DCLK should be such as to maximize the time from the falling edge of DCLK. DCLK also clocks the shift registers used to select the next memory cell for writing. These registers are cleared when RSR and RST1 are low and start counting when LSR and START-X go low, loading a 1 bit into the fast row and slow column shift registers respectively. ORST is the same signal as RSR. All analog memory locations are reset on the rising edge of ORST. END1-4 is used to determine the end of the collection (WRITE) phase, so that the unload phase can be started. O2S is used to synchronize the START-X signals. RST2 is held high to prevent read attempts during the collection phase.

3.2.2 16-Way Active Multiplexer Design

Distribution of signals and time delay to drive the hybrids is accomplished by a 16-Way Multiplexer. System requirements and characteristics of the AN1000H hybrid dictate the performance requirement of the multiplexer. Following are the design objectives:

Input parameters:
- 0 ± 360 mV (720 mV P-P)
- DC-350 MHz
- 50 Ohms, Impedance
- VSWR 1:8, maximum

Output parameters:
- 16 Outputs
- 1 ns delay between each output
- 0 ± 360 mV
- DC-360 MHz, flat frequency response
- 50-Ohm impedance load (50 Ohms shunted by 4.7 pF)
- VSWR 1.8:1 maximum

Power Supply:
5 VDC
-5.2 VDC

The design of the multiplexer is shown in Figure 3-3. The signal distribution and delay requirements are achieved by tapping a microstrip transmission line at 16 points, each 1 ns apart. The line is meandered to reduce layout length and is adequately spaced to reduce coupling between adjacent segments. Circuit loading at the taps is minimized by using JFET buffer amplifiers with high input resistance and low gate-source capacitance. Current gain is supplied by a bipolar transistor output stage.

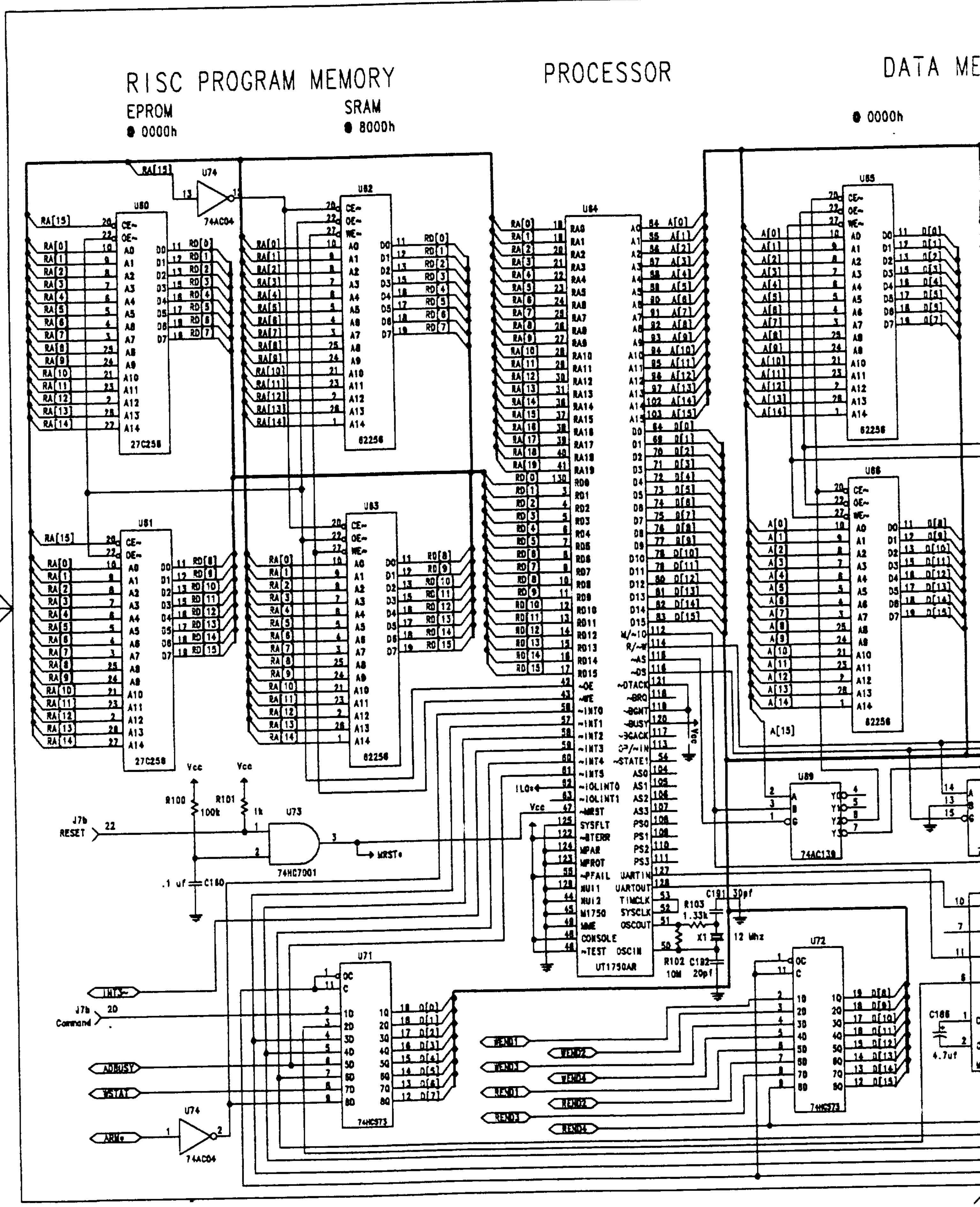
The circuit is built on a 31-mill microstrip board utilizing G10 dielectric material. The delay line is shown in Figure 3-4. Surface mount packaged JFETs are used. Initially, test data is taken with 3.3 pF chip capacitors simulating the gate capacitance of the JFETs. The test results, showing group delay and insertion loss as a function of frequency, are shown in Appendix 4. These measurements show that the transfer characteristic of the delay line has a 3dB slope down to 350 MHz, which is too large to meet system performance objectives. A second design, shown in Figure 3-5, was tested. In this design lower loss dielectric material (Duroid) was used. A slope of 1.2 dB was obtained, which is smaller but still considered unacceptable. Therefore, this approach was abandoned, and the problem is resolved by sampling the input signal at sampling times staggered by 1 ns from channel to channel.

3.3 System Description

The 1 GHz Digitizer system block diagram is shown in Figure 2-2, and the detailed circuit diagrams are shown in Figures 3-7 through 3-10. A short duration of the input signal is stored as discrete analog samples in a set of AN1000H Hybrids. There are four hybrids, each of which has four channels. Each channel can capture a signal with picosecond precision using fast pre-samplers, but requires a 16 ns cycle to store the sample. The four hybrids provide 16 channels, each staggered by 1 ns thus acquiring a new sample every nanosecond. The pre-sampler clocks are brought out of the hybrids in pairs, allowing for eight different clocks (two nanoseconds apart). The two channels sharing the same clock are connected to different signal paths, one having an extra 1 ns of coax delay in it. Each channel is 256 cells deep, resulting in a total acquisition of 4096 samples, or over 4 microseconds of data. No attempt has been made to shorten the acquisition cycle. After the samples are collected, a readout phase begins, which transfers the successive sample of each channel to a fast 12 bit A/D converter. Only a small fraction of the 4096 samples are actually needed, but the four channels on each hybrid are arranged such that one must be completely read before the next one can be started. The time (and power consumption) of this requirement is minimized by rapidly clocking out the samples before and after the desired segment, and slowing the clock to the speed of the A/D converter for the samples which are actually needed. This process is currently implemented using fixed constants, but could be modified to identify the region of interest, with a small increase in on time. The cells beyond the end of the region of interest in the fourth channel in each hybrid do not need to be clocked out at all, so the closer the desired data is to the beginning of the buffer, the less time will be required in the unload phase. The readout clock and A/D converter control are generated in software in the microprocessor. This is a full time task during that interval, and could not have even been considered on a slower processor. For this reason, no processing is done until the readout is completed. Upon completion, the data is scaled by gain and offset values stored from a calibration sequence and transferred to the host system. Calibration consists of starting the calibration sequence, providing a fixed voltage input, doing an acquisition cycle, providing the value of the fixed voltage, changing the voltage, and repeating the process as many times as desired (3 minimum, 16 maximum, 6 to 10 suggested). A final call to the calibration routine converts the accumulated summations into slope and intercept parameters for each cell. Separate calibration data is necessary for each cell because of variations in the sample and hold capacitors which store the samples. The host system communicates with the A/D system by means of a 16-bit bi-directional data bus with 6 handshaking wires. A command set is provided that allows not only calibration and operation, but also allows reading and writing (except EPROM) all data memory and code memory addresses, individually or by blocks, and input or output to any I/O address. Extra commands are reserved which are initially NOP's but can be patched to provide added functionality, even in flight.

In order to minimize power consumption and heat dissipation, power duty cycling has been implemented. Many of the power requirements, including all of the GaAs and ECL logic are needed only during the four microseconds of acquisition. These are known as the write loads. Several other voltages are needed only during acquisition and readout. These are known as the read loads. All switched voltages turn on within 5 microseconds, and are designed to be stable within 10 microseconds. Another 10 microseconds is allowed for the circuitry drawing the power to stabilize. Ten microseconds is allowed as a trigger window, and another 10 microseconds for the acquisition and shutdown of the write loads. The read loads remain in for several milliseconds, depending on the size and position of the data to be unloaded.

FOLDOUT FRAME



FOLDOUT FRAME 2.

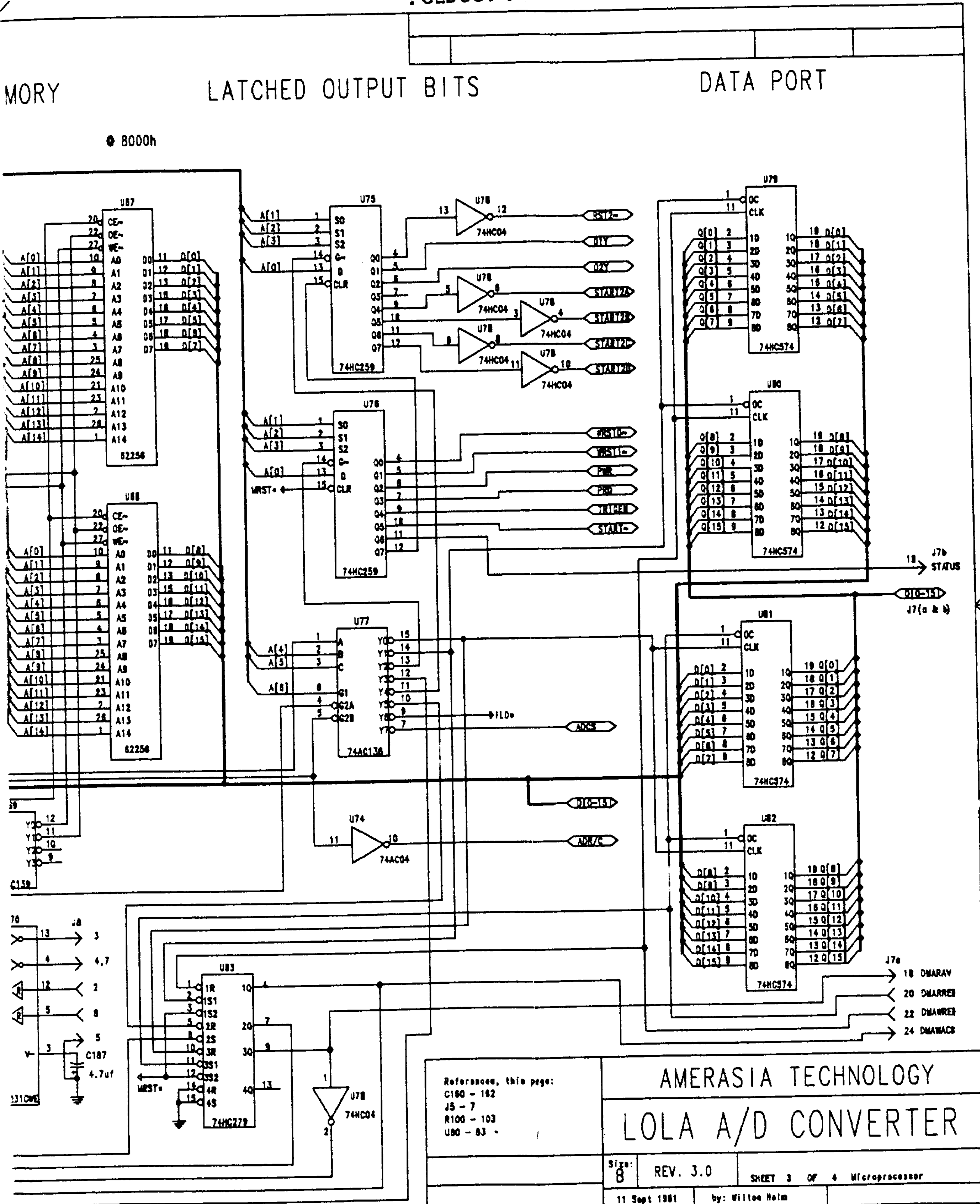
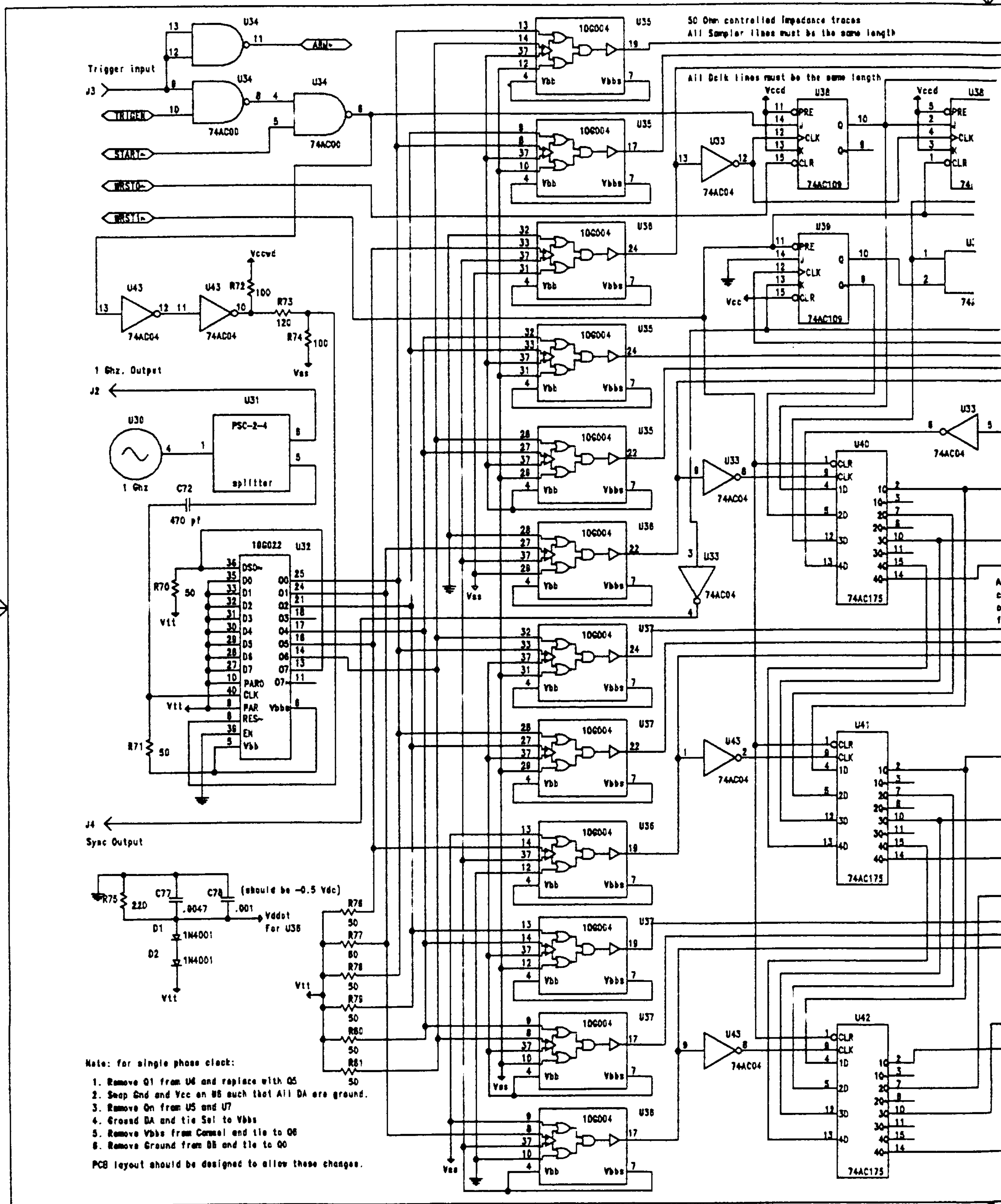


Figure 3-9, Lola A/D Converter

FOLDOUT FRAME /



FOLDOUT FRAME 2

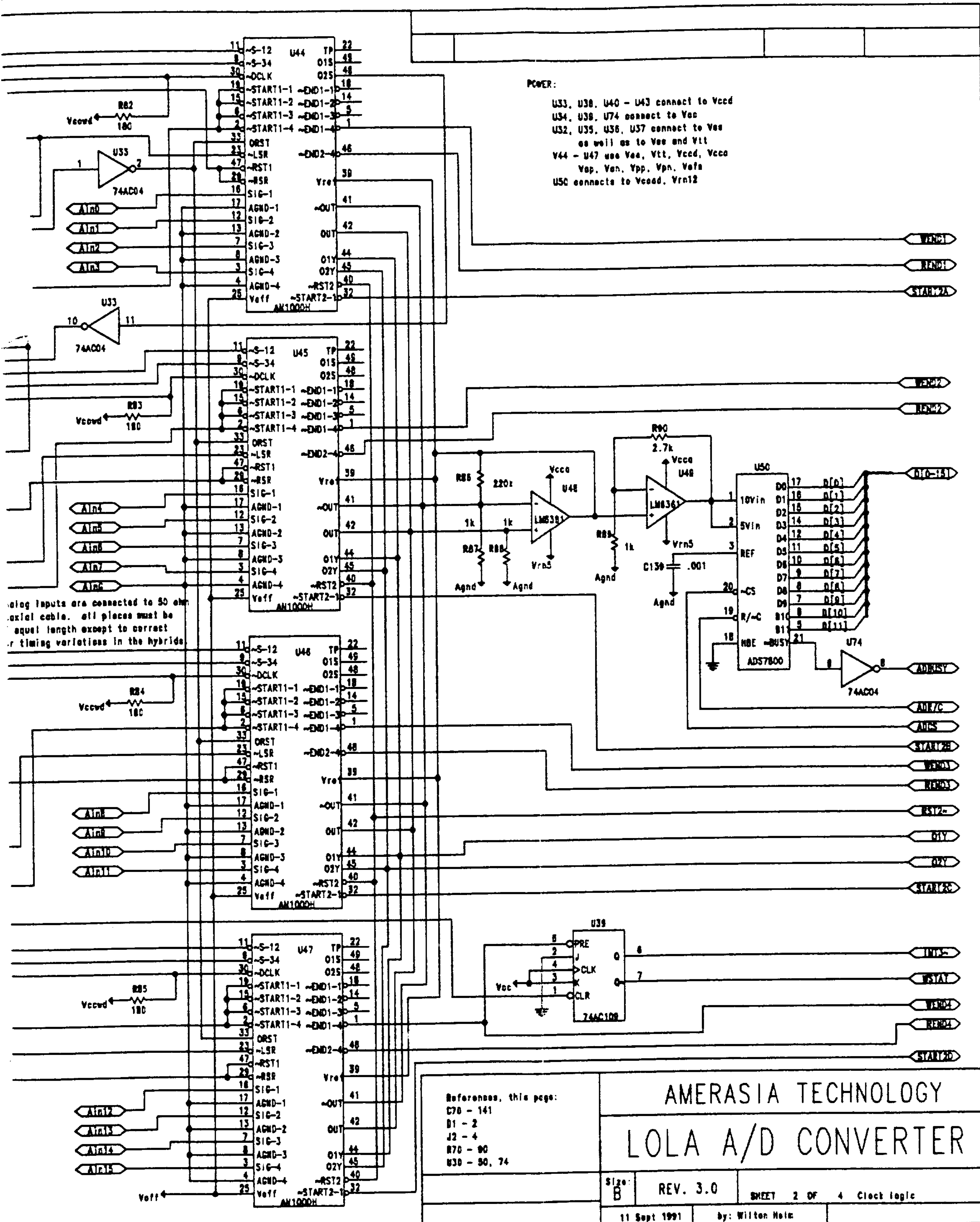
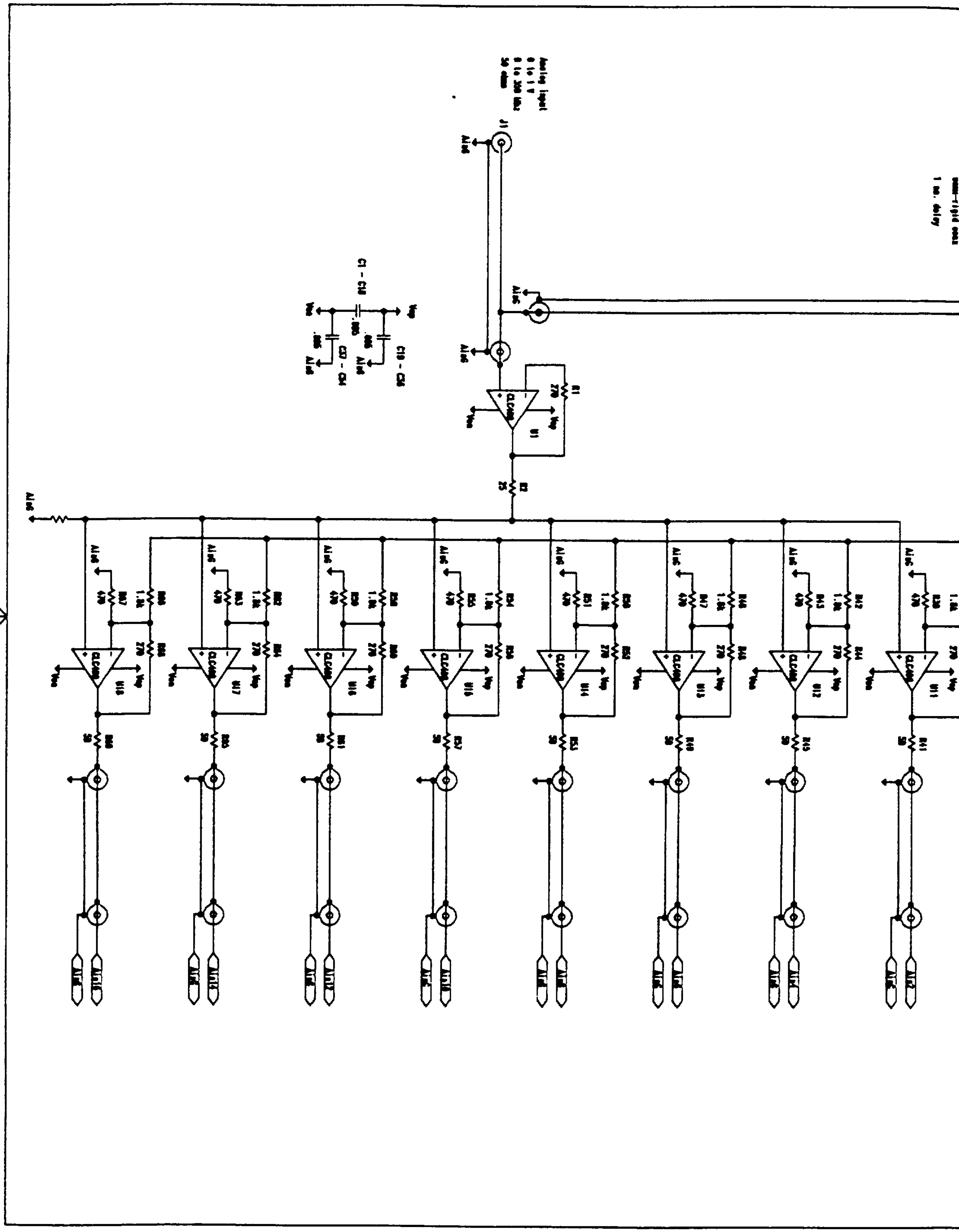
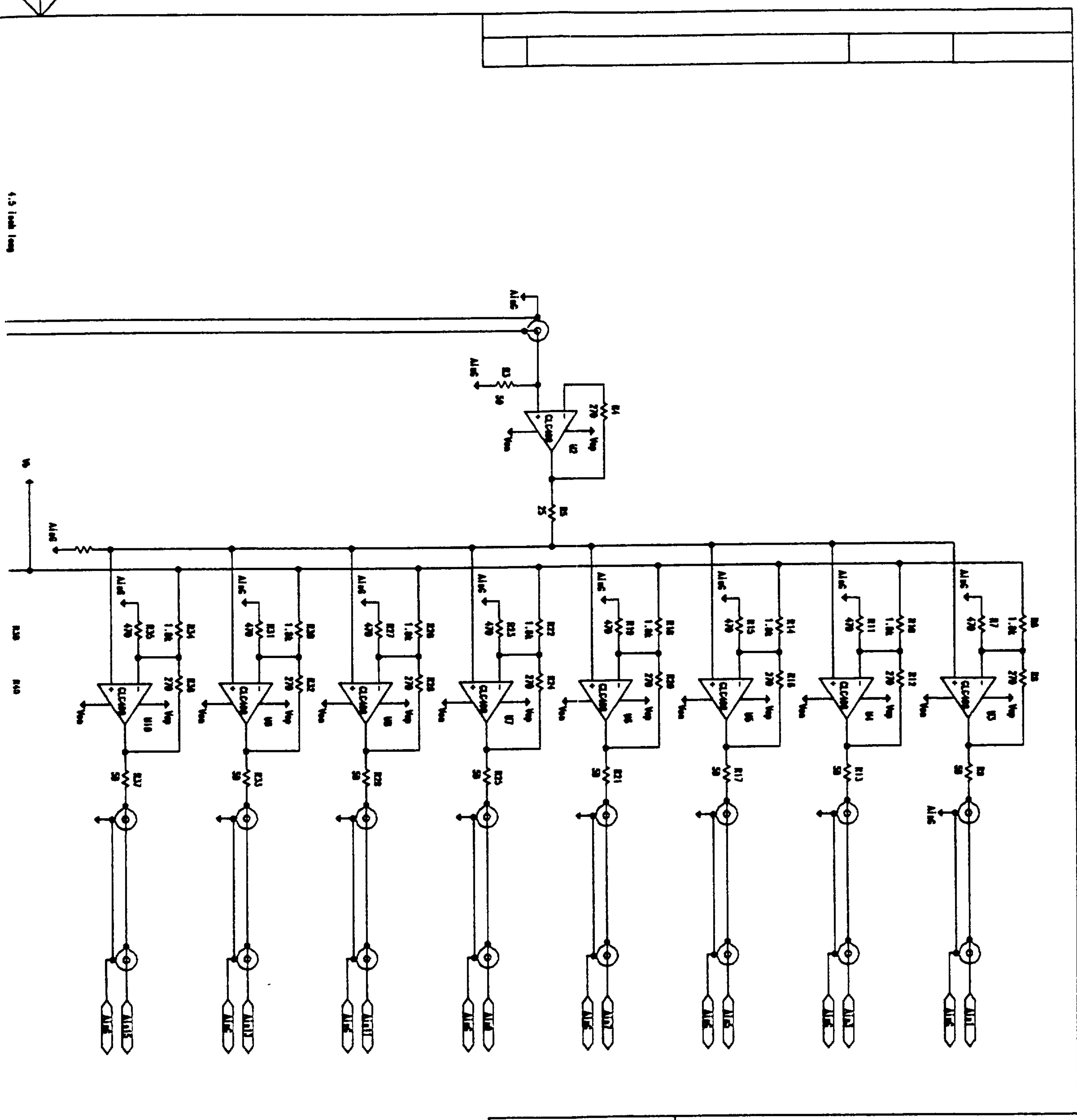


Figure 3-8, Lola A/D Coonverter

FOLDOUT FRAME /



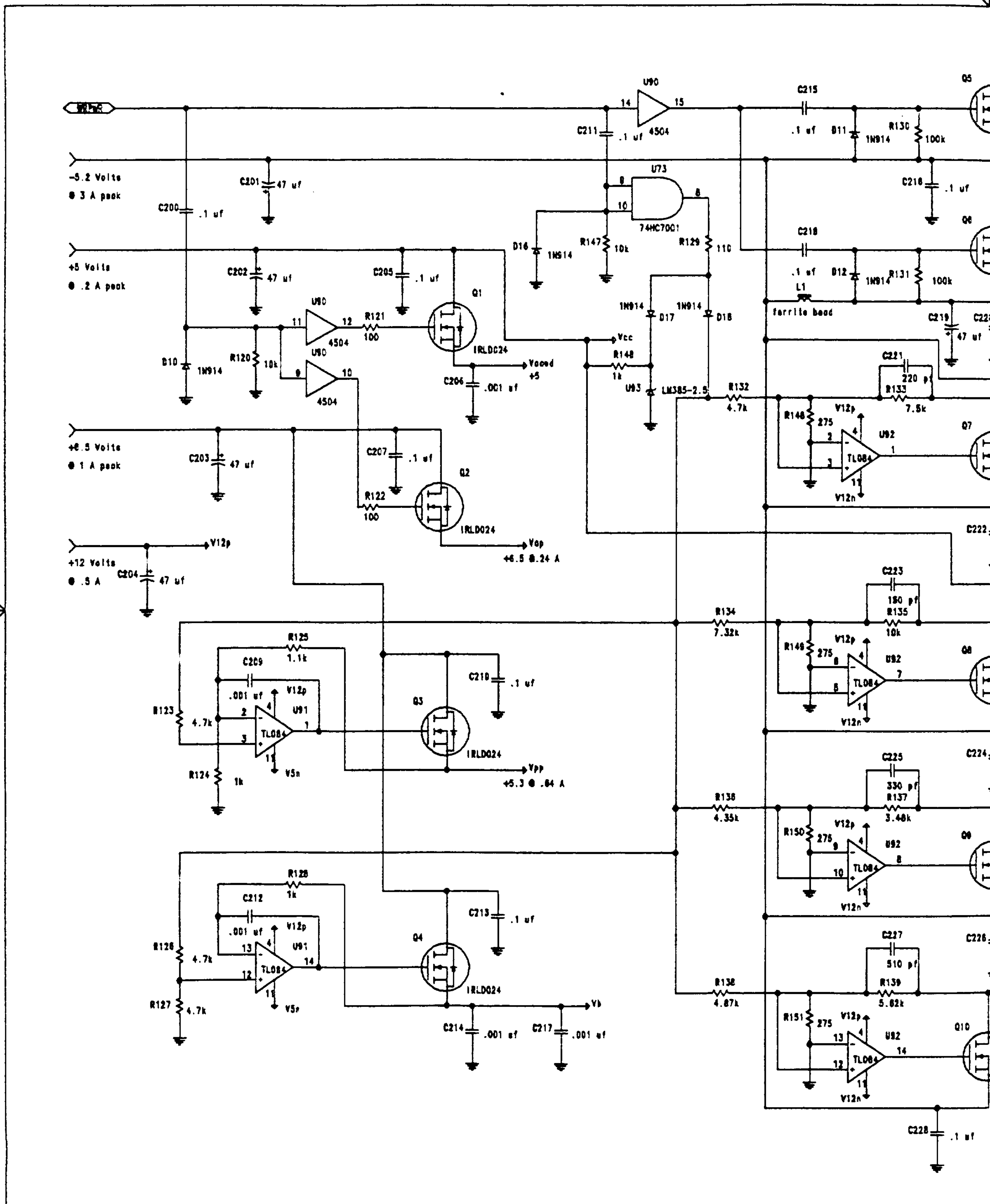
FOLDOUT FRAME 2



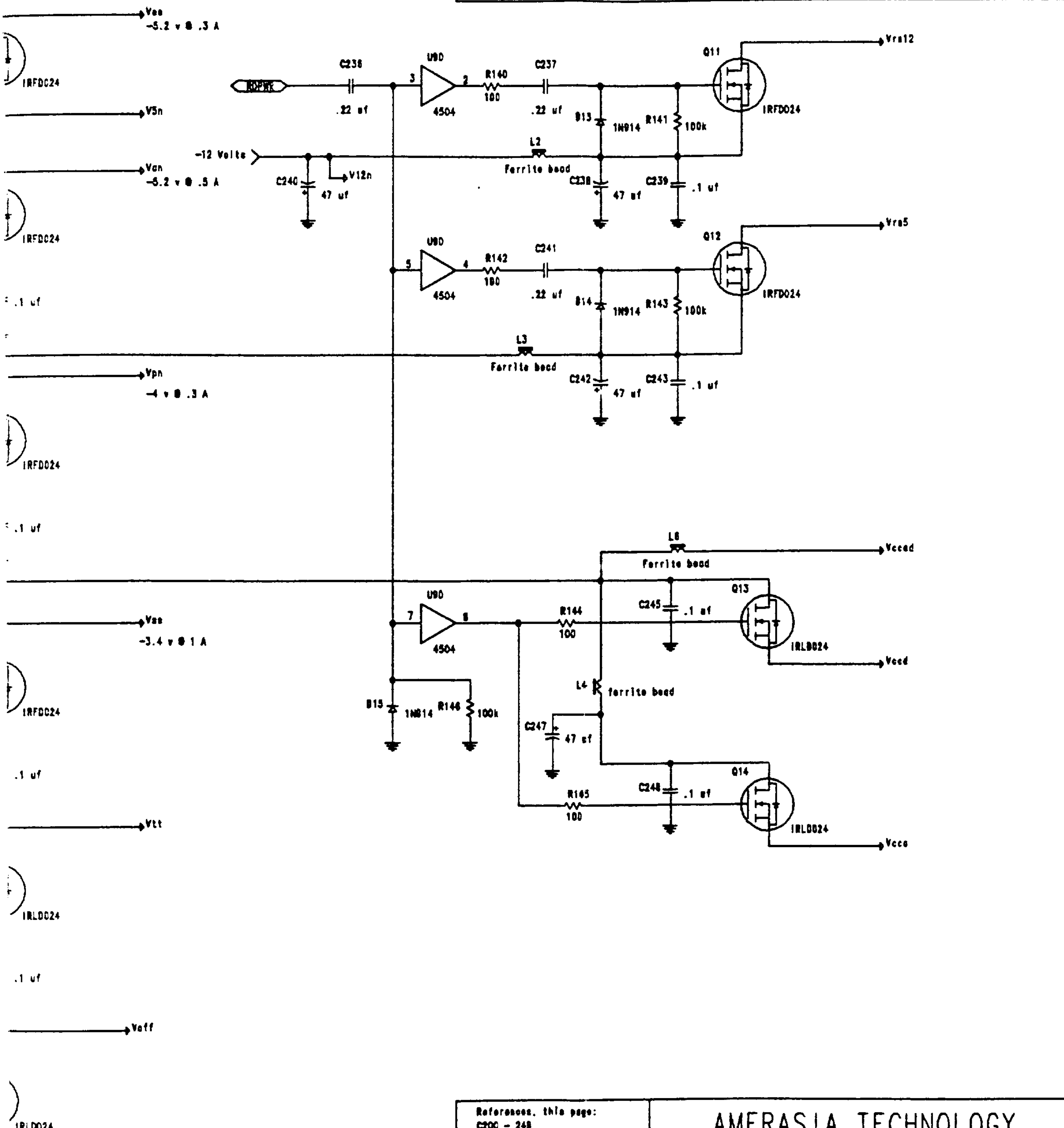
Last reference: C64 J1 R68 U18	AMERASIA TECHNOLOGY
	LOLA A/D CONVERTER
	Size: B · REV. 3.0 · SHEET 1 OF 4 · Analog Input
	20 August 1991 · by: Wilton Hale

Figure 3-7, Lola A/D Converter

FOLDOUT FRAME /



FOLDOUT FRAME 2.



References, this page: C200 - 248, D10 - 15, Q1 - 14, R120 - 146, U9C - 83	AMERASIA TECHNOLOGY
	LOLA A/D CONVERTER
	Size: B · REV. 3.0 · SHEET 4 OF 4 Power Switching
	11 Sept 1991 · by: Wilton Helm

Figure 3-10. Lola A/D Converter

3.3.1 Signal Input Stage

The input amplifier is constructed in two section, as shown in Figure 3-7, each having two stages. The sections are identical except that one section has no delay, the other a one nanosecond coaxial delay line. The first stage is a buffer amplifier which provides minimal loading of the input signal to allow bridging of the two stages, and low enough output impedance to drive the eight second stage amplifiers in parallel. The input signal is terminated in 50 ohms at the end of the 1 nanosecond length of coax. The second stage provides minimal loading to the first stage and a 50 ohm output to match the hybrids. A wideband amplifier having a frequency response which is flat from DC to 350 MHz pushes the state of the art in semiconductors and integrated circuits. It would be much simpler if DC coupling were not required, or if a DC offset could at least be tolerated. DC coupling was a requirement partly to simplify calibration, and partly because any coupling capacitors which could charge and stabilize during the 20 microsecond turn on time would not pass low enough frequencies to leave the pulses being measured undistorted. Operational amplifiers represent the most straightforward method of providing DC coupling without significant offset. The recent development of current feedback op-amps allows their use to frequencies in excess of 300 MHz. The CLC 409 was chosen for this application. With careful construction practices it can be built into a system which is fairly flat to 350 MHz.

3.3.2 Staggered Clock Generation

In order to acquire samples with 1 ns resolution, a 1 Ghz clock with fast rise and fall times must be the basis of the timing logic. A SAW oscillator was used for this purpose. The oscillator is connected to a power splitter which provides an external output for other uses, and a clock signal to a GaAs shift register. The shift register is connected to circulate a pattern of 8 zeros followed by 8 ones. By tapping this signal at various stages, the four 16 ns clocks were provided (one for each hybrid, four nanoseconds apart). The eight pre-sampler clocks come from GaAs gates which generate pulses based on the time it takes for the basic clock transition to pass from one tap of the shift register to a tap two stages later. This provides a 2 ns wide clock pulse every 16 ns. Eight of these gates provide pre-sampler clocks staggered by 2 ns intervals.

3.3.3 Analog Memory and Presampler

The heart of the system is the set of AN1000H hybrids. Each of these memories have four signal inputs, two pre-sampler clock inputs, one data transfer clock input, and various control signals, as shown in Figure 3-8. The control signals are generated by TTL logic derived from the trigger signal and the data transfer clock for the first hybrid. The control signals must be delayed by 4 ns per hybrid. This is done by re-clocking them for each of the other hybrids, using a 47AC174 quad D flip-flop clocked from that hybrid's data clock. All pre-sampler and data clocks are routed over 50 ohm controlled impedance traces whose paths are match to within 2/1000 inches to minimize external channel to channel skew. The internals of the hybrids exhibit some skew, as was shown earlier. It has been assumed that this skew is consistent from device to device. The signal inputs to the hybrids from the front-end amplifier are connected via pieces of semi-rigid .141" coax cable. The pieces are all the same length, except for variations calculated to compensate for the internal skew. The following table shows this:

CHANNEL	RELATIVE DELAY	CABLE LENGTH
1	200 ps	+.9"
2	0 ps	
3	135 ps	+.6"
4	260 ps	+1.2"

The hybrids require numerous voltages (see Figure 3-8), all of which are duty cycle switched, some write cycle only, other stay on until the end of the read cycle. The read cycle also has its own clock and control signals which are derived from I/O circuitry on the microprocessor. This approach simplifies the hardware, and adds flexibility. The analog output is shared between all sixteen channels on all four hybrids. It consists of a differential amplifier with feedback, followed by a gain stage, which drives an A/D converter. The A/D converter has 5 volt and 10 volt inputs which are tied together to give a three volt full scale sensitivity. Its input is bipolar, which fits well with the output of the hybrids. No offset was included. The A/D converter has a built in sample and hold, and is very easy to use. Each read starts a new conversion. This does mean that a priming read must be given, or conversely, the first data element is discarded, and an extra read is done at the end. The 5 volt supply to the A/D converter had to be left on all the time because the converter is connected to the computer data bus. A bus buffer chip would have solved this had it been anticipated. The -12V supply is switched. Because the original intention was to switch the

+5V supply also, the reference voltage is filtered with a much smaller capacitor than the recommended 47 uf. This may need to be changed if it adds too much noise.

3.3.4 Microprocessor Interface and Communications Circuit with Host Computer

The microprocessor is a UT1750AR RAD hard RISC processor, as shown in Figure 3-9. It uses a Harvard architecture, having separate code and data memory. Only the lower 16 bits of the code address are decoded, with the first 32k being EPROM and the last 32k SRAM. The only thing in SRAM by design is a command jump table and a few small utility code fragments. These are copied from the EPROM. The remainder of the SRAM is reserved for extensions, including in-flight modifications. The data memory consists of 64k bytes of fast CMOS SRAM. The original intention was only to populate the first half. Most of this is used up by calibration tables and the data buffer. The second half is unused at present. An RS-232 serial port is built into the processor, but is of limited use because it does not generate interrupts. It is not needed in this application, although a level translator and connector have been added to allow for its use. If it is not used, it may be desirable to cut the power jumpers to this chip to lower the power consumption. I/O decoding is done with a 74AC138. In addition two 74HC259's provide latched output discretes which are set by writing (anything) to a specific I/O address, and cleared by writing to one address lower. U75 deserved particular mention because its clear input comes from U76-Q7. This means that when U76-Q7 is high, U75 acts like a decoder, with all outputs low, except for the one being addressed at a particular moment (if any). This allows short positive pulses to be output instead of levels, which is used to generate the fast two phase read clock used to skip cells. The other outputs of U75 have been inverted as necessary so that their cleared state would be compatible with this operation. Two 74HC573's provide a 16 bit status word. The low byte contains various handshake signals, while the upper byte is a copy of the read and write end signals of all four hybrids. These latter signals are not likely to be used in operation, but are useful for testing. U72 may be optionally removed for in flight use to conserve power, although its consumption is minimal, mostly derived from the extra capacitive loading it provides to the bus. Host communication is provided by U79 - U82 (74HC574's) and U83 (status latches). The protocol for this is described elsewhere, and a listing of the software is given in Appendix 1. Notice that U83 is level sensitive, and responds to the leading edge of the pulse. This means that pulses should be less than 500 ns wide so the micro does not have time to start a new operation before the old operation is completed externally.

3.3.5 Power Supply Duty Cycle Control

The power supply consists of three sections. The first is fixed voltages. The microprocessor subsystem needs a constant source of +5V. The 1 Ghz oscillator and the RS-232 level shifter need a constant source of + 12 V. The second category is switched versions of input voltages. In addition to the above voltages, 6.5V, -5.2V, and -12V, are provided to the system. These voltages and +5V are switched by power mosfets as needed for the write loads and read loads. In each case the switching signal passes through a capacitor to insure that it will time-out in hardware in case of a microprocessor upset or glitch. The third category is switched derived voltages. The write loads need, in addition to the above voltages, +5.3 V, -2 V, -3.4 V, and -4 V, as well as two offset bias voltages. These are generated by power mosfets controlled by op amps. An LM385-2.5 is used as a precision reference. The original intention was to power it on and off for the 40 microseconds of the write cycle. However, it has several internal capacitors which cannot charge that rapidly, so it is left on, lightly loaded all the time. A copy of the write power control signal is current limited in a 110 ohm resistor and clamped by this regulator. This signal provides a regulated pulse to the op amps controlling the mosfets. When this signal is off, the op-amps shut off the mosfets, and when it is on they regulate the output of the mosfets to the desired voltage. Extra resistors and capacitors were required to provide closed loop stability of the op-amp circuits, while at the same time providing the rapid turn on and settling needed by the loads. The pulse which is clamped by the LM385-2.5 is also capacitor coupled to provide a hardware time-out. This is especially important here because the mosfets which are regulating must dissipate power and are not heat sunk for continuous dissipation. Also the write loads, especially the GaAs, are not heat sunk for continuous dissipation. IT IS VERY IMPORTANT NOT TO BYPASS THIS DUTY CYCLE LIMITING.

The decision of which voltages to require as inputs and which to derive was based on power consumption considerations. All constant power sources were made inputs, since any regulator drops in these would represent full time power consumption and heat losses. It is assumed that these voltages can be generated externally with high efficiency switching power converters. Second, all voltages that would represent a major drop from the next higher available voltage are required as inputs. The +6.5V input for instance would be very inefficient to derive from +12V. The remainder of the voltages are derived from these voltages, using the mosfet regulator/switches. The voltages are all write cycle loads having a 40 microsecond duration and a .1% duty cycle, so the losses in the regulator represent a very small fraction of the total power consumption.

3.4 Testing

3.4.1 Characterization of the AN1000H Hybrid

The hybrid is characterized in the test circuit shown in Figure 3-1. It is clocked at 62.5 MHz, which is the DCLK clock. The same clock rate is used for the presampler inputs, which are 2 ns wide. The difference between this test circuit and the final implementation of the 1 GHz Digitizer is that the latter consists of four hybrids, for a total of 16 inputs, while the signal to each input are delayed by a different amount, with the differences being exactly 1 ns apart. This provides 16 equally spaced (1 ns) signals every 16 ns (at 62.5 MHz). For initial testing the test circuit was connected to a 20 MHz function generator, set to produce SINE, SQUARE, and TRIANGLE waveforms at various frequencies. The analog memories (in the hybrids) is filled from this source (all four channels in parallel), and then read out (sequentially) using a 400 Msamples/s 6-bit digital oscilloscope to observe the waveform. Plots from the digital oscilloscope are given in Appendix 3. A full set of 256 samples requires 4.096 microseconds to collect, and sequentially reading back the 1024 samples takes 9 milliseconds. Therefore the frequency of the input signal is reduced by approximately 500 to 1.

In this procedure there are several limitations. The four inputs are resistively isolated, causing significant loss of signal, hence requiring maximum output from the function generator. Consequently, little room is left for baseline offset, and may have contributed to some clipping of the waveform. In addition, the oscilloscope resolution of 6 bits tends to exaggerate the noise content of the waveform. The signal would normally be sampled by a sampling A/D converter just before each transition, when it is most stable. The oscilloscope, however, shows all the waveform settling and glitches, which otherwise would not be digitized. To isolate the digital from the analog signals two planes are used, which are segmented. The test results are given in Appendix 3. Figure A3-1 shows the input signal (lower trace) captured by the test circuit. The upper trace is a frame sync signal which defines the beginning and end of the capture interval. The input signal was a portion of the rising half of a 100 KHz triangular wave. The noise on the signal shows the typical resolution and noise of the oscilloscope.

Figure A3-2 shows the output (readback) signal resulting from the input signal. Similarly, the upper trace shows the frame sync (read frame) signal used to define the beginning and end of the readback interval. Notice that there are four copies of the input signal, corresponding to the sequential readback of the four channels. These four copies should be identical. The end purpose of this test circuit is to quantify channel to channel variations. In Figure A3-3 the same signal as in Figure A3-2 is shown using a faster timebase so that the first channel is given an expanded view. This allows a detailed look at signal variations and noise. Figure A3-4 shows an even faster timebase, so that the individual cell readout is clearly visible. Cell changes occur at 9-microsecond intervals. Notice in the latter two cases, while there is considerable noise on the signal, the largest spike occurs immediately after a cell transition followed by one occurring near the midpoint of the cell. The last microsecond before a transition remains relatively noise free. This is the point for optimum sampling and digitization.

A 2 MHz triangular input signal is shown in Figure A3-5, and its corresponding output waveform is shown in Figure A3-6. The three discontinuities appearing at the output are not errors, but represent the transition from one channel to another. In order to show this boundary, the input signal frequency is chosen not to be a multiple of the sampling window. With a 1 MHz input sinewave, the output is shown in Figure A3-7. The channel to channel transitions are even more clearly visible. In Figure A3-8, the same output is shown with a faster timebase to show only one channel output. The input sinewave frequency is increased to 20 MHz and the resulting output is shown in Figure A3-9. Since there is no output filtering, the beat frequency between this signal and the 62.5 MHz clock is perceptible in the output waveform. The top trace belongs to the sample/hold clock used by the A/D converter which samples the output. The sample phase ends with the negative edge of this clock, which occurs immediately before the cell transition of the hybrid. The first two clock pulses are part of the initialization process, thus do not present valid data.

From the test results several conclusions are drawn. First, the basic concept is viable. The test circuit captures and reproduce the waveforms. Second, the signal-to-noise ratio is adequate for the intended application. Third, there is some channel to channel offset which needs to be dealt with during calibration. In the remaining test results, given in Appendix 3, further test and analysis are performed with input signal frequencies up to 1 GHz. Output waveforms are displayed with errors due to offset and gain variations. Their corrected versions are also given.

3.4.2 System Tests

A number of tests and considerable debugging of the system were performed, although it was not 100% functional or tested when the funding ran out. The microprocessor and support logic and memory have been fully tested and are fully functional. The intended software has been written, a listing is given in Appendix 1. The startup and monitor code have been fully tested and debugged. The one addition which might be made in this area would be self-test routines, such as RAM test, EPROM checksum, and as much I/O testing as the design allows for. The conversion cycle routines have been partially tested. The cycle progresses properly without any time-out or errors. The write cycle appears to initialize the system properly, and respond to status properly. The read cycle seems to generate the proper clock and control signals, operate the A/D converter properly, store data in the correct place and scale and output it correctly. Because no calibration has been done and not useful data retrieved, this section is less than fully tested. It has been determined that output data is dependent on input signal, but with the limited amount of testing done, it was not obvious that the output data was a reasonable representation of the input signal. The calibration routine has not been tested at all, and is not guaranteed crash free at this time. A copy of the C code (known to work) from which this routine was derived will be provided with the system as an aid to understanding and debugging this routine. The C code is based on processing one cell at a time from several files, each containing data at a different calibration point. The 1750 code by contrast was designed to accumulate the necessary intermediate values for each cell, one calibration point at a time, and then transform the result to slope intercept form afterward. The power circuitry has been tested. All power supply voltages achieve nominal value within 5 microseconds after turn on. Turn off decay depends on the current drawn by the load, and the number and size of the bypass capacitors across it. Current from the source stops within a microsecond or two after the shutdown command. The hardware time-out circuitry has also been tested and works properly. The signal input circuitry has been tested, both with static power and duty cycled. The value of the feedback resistors has been optimized for maximally flat frequency response. The trace between the first and second stages was found to be inadequate at 350 MHz, and had to be supplemented with a wire. The amplifiers furthest from the source were showing more attenuation. No testing has been done on the coax path length. This testing cannot be done until calibrated signals are available from the system. The test procedure should consist of calibrating the system (DC) and then placing a sine or triangle wave of 20 to 50 MHz and full scale amplitude into the system and examining the output data graphically. Timing errors will result in certain segments of the resulting waveform being higher or lower than they should be. A short path will cause a rising slope to appear higher than expected, and a falling slope to appear lower. This is in contrast with gain errors which will always appear closer to (or further from) the midpoint than expected, or offset errors which will always appear higher (or lower) than expected. If every other channel is off, the 1 ns delay coax will need trimming. If random channels are off, the coax between the input amplifier and that channel must be trimmed.

The clock and control signals have been examined. It is not possible with the available equipment to fully check out the GaAs clock signals for duty cycle and phasing in a pulsed system. This is because the available oscilloscope is not fast enough for real time sampling, and the pulsed signal was not long enough to be captured using repetitive equivalent sampling. The signals were verified to be present and on frequency, while the control signals were verified to be present and appropriately timed. The write cycle was measured at 40 microseconds, and the read cycle at 2.8 ms (this is sample size dependent). The read clock was checked, the fast clock was 840 ns and the slow clock was 3.3 microseconds. Both had the expected number of pulses for the skip and sample values used, and were checked with different values. The START-2 signals were check and found to be inserted properly. No check was made of any end signals, although since the software does not read out the trailing skipped values from channel four of the hybrids, no end signal would have been visible unless all channels were read.

3.4.3 Special Operations Instructions and Notes

The power inputs are assumed to turn-on simultaneously. If it is necessary to sequence them for external reasons, the order should be +5 and +12 first -5 second and then +6.5. The -12 can be turned on anytime with or after +12. There is hardware and partial software support for software triggering of a conversion cycle. This is included for testing and calibration purposes. The variable SoftTrig is set to 0 (disabled) on a cold reset. It may be changed to non-zero using the monitor write instruction. When it is non-zero, the software does not wait for a hardware arm signal, and supplies an internal hardware trigger to start an acquisition cycle. This trigger is generated approximately 30 microseconds after the acquisition command is given by the host, when SoftTrig is non-zero. All testing if the system has been done using this method of triggering. It is possible, and may prove desirable, to eliminate the hardware ARM command. Since the host system is likely to have control over the subsystem responsible for the data to be collected, it could issue the acquisition command at the time the system needs to be armed. One easy way to implement this change is to separate the bits of the SoftTrig variable such that one bit is used to skip the ARM check and another to force a software trigger. Presently both tests simply check for a non-zero value. In Appendix 5 the description of hardware interface with the system computer is given.

4.0 CONCLUSIONS AND RECOMMENDATIONS

The design of the 1 GHz Digitizer for Space Based Laser Altimeter has been completed. A feasibility model was built, and partially tested. Testing was not completed because of the limited funding available at this time. However, a unique method of digitizing wideband signals (350 MHz) with very low average power consumption was developed and proven to be feasible. The heart of the system is a state-of-the-art hybrid memory chip (AN1000H) with built in presamplers. Sufficient hardware testing has been performed to give assurance that the developed technique is feasible. Because of the much lower power consumption achieved as compared to the initially expected amount, a continuous 1 GHz timing output (from a SAW oscillator) is afforded within a limited power budget. A software package to control various functions within the digitizer and to communicate with a host system computer was also developed. Unfortunately, the allowable budget would not permit debugging and testing of the software. Because the status of the digitizer is so close to completion, it is recommended to extend funding of this project to complete all necessary testing and packaging to obtain a working model. At this point technical risk is minimal if not nonexistent. At the completion of this feasibility model, the logical action is to develop a space qualified unit for future space exploratory missions.

APPENDIX #1

```
                 ;This file is designed to be assembled with CROSS16 meta-assembler
                 ;  version 2.0

                 ;File created 9 August 1991
                 ;  by Wilton Helm
                 ;  last modified on 10 September 1991

                 ;Contains stand-alone UT1750AR assembly language code in native
                 ;  mode to control a 1 Gs/s A/D converter system using Analytek
                 ;  hybrid analog memory modules in a 16-way interleaved fashion.

                 ;Code memory map is as follows (addresses are 16 bit words):
                 ;  0000h to 7FFFh  is EPROM containing startup and core code.
                 ;  8000h to FFFFh  is SRAM containing linking tables loaded from
                 ;                  EPROM and dynamically added code modules.
                 ;  10000h and up are not decoded and wrap into above areas.

                 ;A separate 64k x 16 bit data address space is populated with
                 ;  SRAM, and used as described by data equates below.

                 ;A separate I/O space is decoded as described by I/O equates below.

                 ;Special note should be given to the hardware signals O1Y and O2Y
                 ;  which are a two-phase, non-overlapping readback clock
                 ;  generated in software.  There are two modes of generating this
                 ;  signal.  For skipping cells, a high speed mode is used where
                 ;  U75 is placed in non-latching decode mode (by asserting its
                 ;  CLEAR pin).  Writes to O1YHi and O2YHi in this mode generate
                 ;  pulses on the respective lines.  For reading cells, the CLEAR
                 ;  is removed and O1Y and O2Y become latching outputs which must
                 ;  be set by O1YHi and O2YHi respectively, and cleared by O1YLow
                 ;  and O2YLow respectively. These commands are interspersed with
                 ;  suitable timing waits, A/D reads and loop control commands.

                 ;Note that the assembler used makes no distinction between
                 ;  code and data objects.  Both share the same symbol table space.
                 ;  It is up to the programmer to use symbols appropriately.  Code
                 ;  space symbols are used only with LRI and STRI instructions.

                 ;The assembler uses one possibly non-standard mnemonic form.  I
                 ;  could not see a way to differentiate between:
                 ;        sar R0,R4           ;shift right 4 places
                 ;  and
                 ;        sar R0,R4           ;shift right by amount specified
                 ;                            ;in register 4
                 ;  so I adopted the following convention for the former case:
                 ;        sar R0,R+4          ;shift right 4 places
                 ;  and for consistency:
                 ;        sar R0,L+6          ;shift left 6 places

  00000                   WDLN 2    ;word length (width in bytes) for the 1750
  00000                   CPU "UT1750AR.TBL"  ;table of mnemonics to use for 1750

                          ;CODE SPACE SRAM EQUATES
00008000     =   ImgDst:  equ    8000h        ;start of RAM code space
00008000     =   CmdJmp:  equ    8000h        ;address table for command execution
00008010     =   CmdRtn:  equ    8010h        ;address to return to after command execution
00008011     =   WarmLink: equ   8011h        ;address of warm start routine.
00008012     =   Postchk: equ    8012h        ;routine to post checksum changes
0000FFFF     =   CRAMe:   equ   0FFFFh        ;end of RAM code space
```

```
                          ;DATA MEMORY EQUATES
00000000    =    CodeChk: equ    0h       ;32 bit sum of code RAM
00000002    =    Samples: equ    2h       ;number of samples to keep (per chan)
00000003    =    Skip:    equ    3h       ;number to skip before starting (per chan)
00000004    =    SoftTrig: equ   4h       ;non-zero if software trigger should be given
00000005    =    CalFlag: equ    5h       ;non-zero if calibration in process
00000006    =    CalVSum: equ    6h       ;summation of correct calibration values
00000007    =    CalCnt:  equ    7h       ;count of number of calibration values used
00000008    =    CalSkH:  equ    8h       ;Hold value for skip, used during calib.
00000009    =    CalSaH:  equ    9h       ;Hold value for samples, used during calib.

00001000    =    ADBuf:   equ    1000h
00002000    =    ADGain:  equ    2000h    ;m from y = m * x + b
00003000    =    ADOfs:   equ    3000h    ;b
00002000    =    CalSums: equ    2000h    ;used for curve fitting, overlays gain
00003000    =    CalSqrs: equ    3000h    ;used for curve fitting, overlays offset
00005000    =    CalVctr: equ    5000h    ;used for curve fitting. Sqrs and Vctr
                 ;        ending at 6FFFh ;  values are two words wide

                          ;I/O ADDRESS EQUATES
00000040    =    DMA:     equ    40h      ;DMA write and read word to host
00000050    =    Status:  equ    50h      ;read to get system peripheral status:
0000001F    =    Cmd:     equ    31       ; b0 = 1 when host is giving command
0000001E    =    CTS:     equ    30       ; b1 is RS-232 handshake (CTS)
0000001D    =    InEmpty: equ    29       ; b2 is DMAWACK (0 = DATA read by host)
0000001C    =    OutFull: equ    28       ; b3 is DMARAV (0 = DATA ready to read)
0000001B    =    ADBusy:  equ    27       ; b4 = 0 if A/D conversion finished
0000001A    =    ConvBusy: equ   26       ; b5 = 0 at end of read cycle
                                          ;      set to 1 by reading I/O 60h
00000019    =    AcqBusy: equ    25       ; b6 = 1 when triggered and 0 at
                                          ;        end of write cycle
00000018    =    NotArm:  equ    24       ; b7 = 0 for ARM system request
00000017    =    NotREnd1: equ   23       ; b8 is *END2-1 from hybrid
00000016    =    NotREnd2: equ   22       ; b9 is *END2-2 from hybrid
00000015    =    NotREnd3: equ   21       ; b10 is *END2-3 from hybrid
00000014    =    NotREnd4: equ   20       ; b11 is *END2-4 from hybrid
00000013    =    NotWend1: equ   19       ; b12 is *END1-1 from hybrid
00000012    =    NotWend2: equ   18       ; b13 is *END1-2 from hybrid
00000011    =    NotWend3: equ   17       ; b14 is *END1-3 from hybrid
00000010    =    NotWend4: equ   16       ; b15 is *END1-4 from hybrid

                                 ;The following addresses are activated by
                                 ;writing. The value written is ignored,
                                 ;only the address matters.
00000050    =    RstWr0En: equ   50h      ;resets the write logic which is not
                                          ;needed during read
00000051    =    RstWr0Dis: equ  51h      ;releases reset
00000052    =    RstWr1En: equ   52h      ;resets write logic which must not
                                          ;  be reset during read
00000053    =    RstWr1Dis: equ  53h      ;releases reset
00000054    =    WrPwrDn: equ    54h      ;removes power from write logic
00000055    =    WrPwrUp: equ    55h      ;applies power to write logic
00000056    =    RdPwrDn: equ    56h      ;removes power from read logic
00000057    =    RdPwrUp: equ    57h      ;applies power to read logic
                          ;      (NOTE: RdPwr is needed for Write as well)
00000058    =    TrigDis: equ    58h      ;prevents trigger signal from
                                          ;  starting a conversion
00000059    =    TrigEn:  equ    59h      ;allows trigger signal to start a conversion
0000005A    =    StartEn: equ    5Ah      ;software trigger
```

```
0000005B    =        StartDis: equ   5Bh        ;end of software trigger
0000005C    =        StatOff: equ    5Ch        ;indicates next word to host is data
0000005D    =        StatOn:  equ    5Dh        ;indicates next word to host is status
0000005E    =        FastRead: equ   5Eh        ;turns 6X writes into pulses
                                                ;this allows O1Y and O2Y to operate
                                                ;faster for skipping cells not needed in hybrids
0000005F    =        SlowRead: equ   5Fh        ;changes 6X writes into latched data
                                                ;this mode is needed for all
                                                ;handshaking to hybrids, and for actual
                                                ;data reading

00000060    =        ResRdSt: equ    60h        ;reading this address resets (1) status b3

                                 ;more write strobe addresses
00000060    =        RstRDis: equ    60h        ;releases RST2 reset
00000061    =        RstREn:  equ    61h        ;resets hybrid read logic
00000062    =        O1YLow:  equ    62h        ;sets O1Y clock line low
00000063    =        O1YHi:   equ    63h        ;sets O1Y clock line high
                                                ;during fast read, pulse this for O1Y
00000064    =        O2YLow:  equ    64h        ;sets O2Y clock line low
00000065    =        O2YHi:   equ    65h        ;sets O2Y clock line high
                                                ;during fast read, pulse this for O2Y
00000068    =        Rd1Dis:  equ    68h        ;trailing edge of START2-1 signal to hybrid
00000069    =        Rd1En:   equ    69h        ;leading edge of START2-1 signal
0000006A    =        Rd2Dis:  equ    6Ah        ;trailing edge of START2-2 signal to hybrid
0000006B    =        Rd2En:   equ    6Bh        ;leading edge of START2-2 signal
0000006C    =        Rd3Dis:  equ    6Ch        ;trailing edge of START2-3 signal to hybrid
0000006D    =        Rd3En:   equ    6Dh        ;leading edge of START2-3 signal
0000006E    =        Rd4Dis:  equ    6Eh        ;trailing edge of START2-4 signal to hybrid
0000006F    =        Rd4En:   equ    6Fh        ;leading edge of START2-4 signal

00000070    =        AD:      equ    70h        ;reads A/D converter and starts next conversion
00000070    =        SWINT:   equ    70h        ;write this address to generate a
                                                ;software interrupt - used to disable ints.

                              ;REGISTER USEAGES
                     ;   R0          Short term temporary use
                     ;   R16         Subroutine return address register
                     ;   R18         Interrupt return address register
                     ;        (return registers need only be 16 bit in this system,
                     ;        since there is only 64k of code space.)

                              ;PROGRAM CONSTANTS
00000000    =        IntMask: equ    0          ;change later to allow desired ints.

                              ;ERROR CODES
00000000    =        OKStat:  equ    0h         ;OK status word
00000001    =        WarmStat: equ   1h         ;Reset caused a warm start
00000002    =        ColdStat: equ   2h         ;Reset caused a cold start
00000004    =        Cmpltd:  equ    4h         ;Command completed
00000005    =        IllData: equ    5h         ;Data rcvd when command expected
00000006    =        NoData:  equ    6h         ;Command rcvd when data expected
00000007    =        WrInt:   equ    7h         ;Command rcvd before requested data sent
00000008    =        Aborted: equ    8h         ;Command rcvd during A/D cycle
00000009    =        NoTrig:  equ    9h         ;status word: no trigger after arm
0000000A    =        NoLoad:  equ    0Ah        ;Analog write cycle failure
0000000B    =        AcqCal:  equ    0Bh        ;A/D cycle completed, but no data because
                                                ;calibration cycles have been requested

   00000                      org    0
```

```
00000 1BFF0050      Init:    otr    ACC,RstWr0En ;value in ACC doesn't matter
00002 1BFF0052               otr    ACC,RstWr1En ;set all resets and power down
00004 1BFF0054               otr    ACC,WrPwrDn
00006 1BFF0056               otr    ACC,RDPwrDn
00008 1BFF0058               otr    ACC,TrigDis  ;disable triggering
0000A 1BFF005B               otr    ACC,StartDis ;  and software trigger
0000C 1BFF005C               otr    ACC,StatOff  ;no status word ready
0000E 1BFF005E               otr    ACC,FastRead ;this resets RstRDis, 01YLow, 02YLow,
                                                 ;  St2-1Dis, St2-2Dis, St2-3Dis, St2-4Dis
00010 1BFF005F               otr    ACC,SlowRead ;undo above resets
00012 1BFF0061               otr    ACC,RstREn   ;Leave Read reset active
00014 97F0                   inr    ACC,TAH      ;stop timers
00015 97F1                   inr    ACC,TBH
                    ;   set up any hardware needed for interrupts
00016 141F0050               inr    R0,STATUS    ;get status word and check Command bit
00018 D81F                   tbr    R0,Cmd
00019 FE8B                   br     NE,cold      ;if set, force cold start
0001A 0000                   nop
0001B 0E3F02C0               call   R16,Chksum   ;test code RAM
0001D 045F0000               lr     R2,CodeChk
0001F 047F0001               lr     R3,CodeChk+1 ;get comparison value
00021 3E12                   cmp    XR0,XR2
00022 7D1F0020               jc     EQ,warm      ; OK, matches

00024 0000                   nop                 ; no match, restart from scratch
00025 001F02CB      cold:    mov    R0,Image     ;block move RAM constants from EPROM
00027 003F8000               mov    R1,ImgDst
00029 03E0          coldlp:  mov    ACC,R0
0002A 845F                   lri    R2
0002B 03E1                   mov    ACC,R1
0002C 8840                   stri   R2
0002D A001                   add    R0,1
0002E 3C1F02E7               cmp    R0,ImgEnd
00030 FC98                   br     LT,coldlp
00031 A021                   add    R1,1         ;(used if br)
00032 0E3F02C0               call   R16,Chksum   ;checksum entire code RAM
00034 081F0000               str    R0,CodeChk   ;save result
00036 083F0001               str    R1,CodeChk+1
00038 83E0                   mov    ACC,IntMask  ;set up interrupt mask
00039 9BE5                   otr    ACC,MK
0003A 0E3F0205               call   R16,Cal0     ;initialize calibration table
0003C 8000                   mov    R0,0
0003D 081F0003               str    R0,Skip
0003F 081F0004               str    R0,SoftTrig
00041 8002                   mov    R0,ColdStat
00042 FF86                   jc     X,startcom
00043 0000                   nop

00044 03FF8011      warm:    mov    ACC,WarmLink ;get address of user warm start
00046 841F                   lri    R0
00047 0E20                   call   R16,R0       ;execute it (will not return
                                                 ;  but will jump (call) warmend)
00048 8001          warmend: mov    R0,WarmStat

00049               startcom:
                    ;   omit for now         otr    ACC,ENBL     ;turn on interrupts
00049 1BFF005D               otr    ACC,StatOn   ;set status flag
0004B 181F0040               otr    R0,DMA       ;output status word regardless of
0004D 181F0040               otr    R0,DMA       ;handshaking
```

```
                    ;        REGISTER USEAGE BY MONITOR
                    ;   R1          low portion of address being accumulated
                    ;   XR2         starting address for range
                    ;   XR4         ending address for range
                    ;   R6          non-zero if address is in code space
                    ;   R7          data portion of command word
                    ;   ACC         command / jmp address

0004F 8020          monitor: mov    R1,0          ;set low portion to zero initially
00050 8241                   mov    XR2,1         ;set starting address of 1
00051 8280                   mov    XR4,0         ;  and ending address of 0 (done)
00052 80C0                   mov    R6,0          ;set to data space
00053 141F0050      nextword: inr   R0,Status
00055 D81D                   tbr    R0,InEmpty    ;see if data available
00056 FE9C                   br     ne,nextword   ; no, wait
00057 D81F                   tbr    R0,Cmd        ; yes, test for command sequence
00058 14FF0040               inr    R7,DMA        ;get command
0005A 7D1F0093               jc     eq,statout    ; not command, error
0005C 8005                   mov    R0,IllData    ;   (error msg - needed if jc)
0005D 0007                   mov    R0,R7         ;copy to command area
0005E 40FF0FFF               and    R7,0FFFh      ;strip data section
00060 401FF000               and    R0,0F000h     ;strip command section
00062 E803                   scr    R0,L+4        ;shift command down
00063 03E0                   mov    ACC,R0
00064 47FF8000               or     ACC,CmdJmp    ;form table address
00066 841F                   lri    R0            ;get jump address
00067 0C00                   call   R0,R0         ;go there, does not return here

                    ; CMD 0
00068 80CF          CodeLow: mov    R6,15         ;set code space flag          (0)
00069 7F9FFFE8               jc     x,nextword    ;get rest of command
0006B 0027                   mov    R1,R7         ;set 12 low bits (jump taken)

                    ; CMD 1
0006C 80C0          DataLow: mov    R6,0          ;set data space flag          (1)
0006D 7F9FFFE4               jc     x,nextword    ;get rest of command
0006F 0027                   mov    R1,R7         ;set 12 low bits (jump taken)

                    ; CMD 2
00070 0047          StHiRng: mov    R2,R7         ;place upper bits in starting address (2)
00071 8060                   mov    R3,0
00072 E25C                   slr    XR2,R+4       ;shift right 4
00073 4641                   or     XR2,R1        ;and combine with low bits
00074 7F9FFFDD               jc     x,nextword    ;get rest of command
00076 8020                   mov    R1,0          ;clear low bits for end adr (jump taken)

                    ; CMD 3
00077 0047          StHiRd:  mov    R2,R7         ;place upper bits in starting address (3)
00078 8060                   mov    R3,0
00079 E25C                   slr    XR2,R+4       ;shift right 4
0007A 4641                   or     XR2,R1        ;and combine with low bits
0007B 7F9F003F               jc     x,MonRd       ;go read it and put on bus
0007D 0292                   mov    XR4,XR2       ;set end = beg for one word (jump taken)

                    ; CMD 4
0007E 0047          StHiWr:  mov    R2,R7         ;place upper bits in starting address (4)
0007F 8060                   mov    R3,0
00080 E25C                   slr    XR2,R+4       ;shift right 4
00081 4641                   or     XR2,R1        ;and combine with low bits
00082 7F9F0055               jc     x,MonWr       ;wait for data and write it
```

```
00084 0292                  mov   XR4,XR2       ;set end = beg for one word (jump taken)

                  ; CMD 5
00085 0047        Execute:  mov   R2,R7         ;place upper bits in starting address (5)
00086 8060                  mov   R3,0
00087 E25C                  slr   XR2,R+4       ;shift right 4
00088 4641                  or    XR2,R1        ;and combine with low bits
00089 023F004F              mov   R16,monitor   ;set up return address
0008B 0C92                  call  R4,XR2        ;indirect jump, return is:
                                                ;         CALL  R16,R16
                                                ;which will re-enter the monitor

                  ; CMD 6
0008C 0087        EndHiRd:  mov   R4,R7         ;place upper bits in ending address (6)
0008D 80A0                  mov   R5,0
0008E E29C                  slr   XR4,R+4       ;shift right 4
0008F 7F9F002B              jc    x,MonRd       ;go read it and put on bus
00091 4681                  or    XR4,R1        ;combine with low bits (jump taken)

                  ; CMD 7
00092 0087        EndHiWr:  mov   R4,R7          ;place upper bits in ending address (7)
00093 80A0                  mov   R5,0
00094 E29C                  slr   XR4,R+4       ;shift right 4
00095 7F9F0042              jc    x,MonWr       ;wait for data and write it
00097 4681                  or    XR4,R1        ;combine with low bits (jump taken)

                  ; CMD 8
00098 7F9F002E    IORd:     jc    x,MonSnd      ;                         (8)
0009A 1407                  inr   R0,R7         ;read I/O, and put on bus (jump taken)

                  ; CMD 9
0009B 141F0050    IOWr:     inr   R0,Status     ;see if data available    (9)
0009D D81D                  tbr   R0,InEmpty
0009E FE9C                  br    ne,IOWr       ; no, wait
0009F D81F                  tbr   R0,Cmd        ; yes, is it data or command
000A0 7E9F004D              jc    ne,statout    ;  command, error - don't continue
000A2 8006                  mov   R0,NoData     ;    (error msg - needed if jc)
000A3 141F0040              inr   R0,DMA        ;  data, read it
000A5 7F9FFFA8              jc    x,monitor
000A7 1807                  otr   R0,R7         ;output data to I/O address (jump taken)

                  ; CMD A
000A8 E0FC        SetSkp:   slr   R7,R+4        ;truncate modulo 16       (10)
000A9 08FF0003              str   R7,Skip       ;  and divide by 16 to get
000AB 7F9FFFA2              jc    x,monitor     ;  skip value per channel
000AD 0000                  nop

                  ; CMD B
000AE A0EF        ManAcq:   add   R7,15         ;round up modulo 16       (11)
000AF E0FC                  slr   R7,R+4        ;  and divide by 16 to derive
000B0 BCE0                  cmp   R7,0
000B1 FD84                  br    le,ManLim     ;don't allow 0 or negative value
000B2 3CFF0100              cmp   R7,100h       ;  or more than 4096 (/16)
000B4 FD82                  br    le,ManOk
000B5 0000                  nop
000B6 80E7        ManLim:   mov   R7,7          ;default to 128 (7 * 16) if invalid
000B7 08FF0002    ManOk:    str   R7,Samples    ;  number of cells per channel
000B9 7F9F0043              jc    x,Arm         ;start conversion cycle
000BB 0000                  nop
```

```
                    ; CMD C
0000004F    =       AutoAcq: equ monitor          ;reserved

  000BC 3E54        MonRd:    cmp   XR2,XR4       ;more data to send?
  000BD 7E1F0030              jc    GT,statout    ; no, return completed status
  000BF 8004                  mov   R0,Cmpltd     ;    (used if jc)
  000C0 44C6                  or    R6,R6         ;code or data?
  000C1 FD04                  br    eq,MonRDta    ; data
  000C2 03F2                  mov   ACC,XR2       ; code, read next word to send
  000C3 841F                  lri   R0
  000C4 FF82                  br    x,MonNxtR
  000C5 0000                  nop
  000C6 0412        MonRDta:  lr    R0,XR2        ; data, read next word to send
  000C7 A241        MonNxtR:  add   XR2,1         ;ready for next pass
  000C8 143F0050    MonSnd:   inr   R1,Status     ;get status word
  000CA D83D                  tbr   R1,InEmpty    ;see if data rcvd
  000CB FD09                  jc    eq,snderr     ; yes, should not have been
  000CC D83C                  tbr   R1,OutFull    ;buffer empty?
  000CD FE9A                  jc    ne,MonSnd     ; no, wait
  000CE 0000                  nop
  000CF 181F005C              otr   R0,StatOff    ;place in data mode
  000D1 181F0040              otr   R0,DMA        ;send data
  000D3 7F9FFFE7              jc    x,MonRd       ;try again
  000D5 8005        snderr:   mov   R0,IllData
  000D6 7F9F0017              jc    x,statout
  000D8 0000                  nop

  000D9 141F0050    MonWr:    inr   R0,Status     ;get status word
  000DB D81D                  tbr   R0,InEmpty    ;see if data rcvd
  000DC FE9C                  br    ne,MonWr      ; no, wait for it
  000DD D81F                  tbr   R0,Cmd        ; yes, data or command?
  000DE 7E9F000F              jc    ne,statout    ;   command, error
  000E0 8006                  mov   R0,NoData     ;     (error msg - needed if jc)
  000E1 141F0040              inr   R0,DMA        ;   data, read it
  000E3 44C6                  or    R6,R6         ;code or data memory?
  000E4 FD04                  br    eq,MonWDta    ; data
  000E5 03F2                  mov   ACC,XR2       ; code, write word
  000E6 8800                  stri  R0
  000E7 FF82                  br    x,MonWrNxt
  000E8 0000                  nop
  000E9 0812        MonWDta:  str   R0,XR2        ; data, write word
  000EA A241        MonWrNxt: add   XR2,1         ;ready for next pass
  000EB 3E54                  cmp   XR2,XR4       ;more to come?
  000EC 7D9FFFEB              jc    LE,MonWr      ; yes, continue
  000EE 8004                  mov   R0,Cmpltd     ; no, return completion

  000EF 8040        statout:  mov   R2,0          ;timeout in case host not unloading buffer
  000F0 143F0050              inr   R1,Status     ;get status word
  000F2 D83C                  tbr   R1,Outfull    ;buffer empty?
  000F3 FD03                  jc    eq,statol     ; yes, proceed
  000F4 A041                  add   R2,1          ; no, increment timer (65536 is fail)
  000F5 FE99                  jc    ne,statout    ;   100 ms. time limit, not out
  000F6 0000                  nop                 ;   ran out, force output
  000F7 181F005D    statol:   otr   R0,StatOn     ;place in status mode
  000F9 181F0040              otr   R0,DMA        ;send data
  000FB 7F9FFF52              jc    x,monitor     ;status sent, return to monitor
  000FD 0000                  nop                 ;  as a recovery

                    ;   R0   temp
                    ;   R1   temp
```

```
                        ;  R8   I/O address of O1YHi
                        ;  R9   I/O address of O2YHi or O2YLow as needed
                        ;  R10  Buffer address pointer
                        ;  R11  Channel number (1 - 4) within a hybrid
                        ;  R12  Hybrid number (1 - 4)
                        ;  R13  Temp cell counter for skip and read

000FE 041F0004  Arm:      lr    R0,SoftTrig  ;If Software triggered, don't
00100 4400                or    R0,R0        ;  wait for hardware arm
00101 FE8E                br    NE,PwrUp
00102 0000                nop
00103 141F0050            inr   R0,Status    ;Wait for hardware Arm status
00105 D818                tbr   R0,NotArm
00106 FD09                br    eq,PwrUp     ; Arm request received
00107 D81D                tbr   R0,InEmpty   ; No Arm request yet, check for abort
00108 FE95                br    ne,Arm       ;  No data ready, loop
00109 D81F                tbr   R0,Cmd       ;  Data ready, is it command
0010A 7E9FFFE3            jc    ne,statout   ;   Command, give abort status
0010C 8008                mov   R0,Aborted
0010D 7F9FFFE0            jc    x,statout    ;   Data, abort with data error
0010F 8005                mov   R0,IllData
00110 1BFF0055  PwrUp:    otr   ACC,WrPwrUp  ;Turn on power to all of A/D system
00112 1BFF0057            otr   ACC,RdPwrUp
00114 1BFF0050            otr   ACC,RstWr0En ;Reset all of A/D system
00116 1BFF0052            otr   ACC,RstWr1En
00118 1BFF0058            otr   ACC,TrigDis  ;disable triggering
0011A 1BFF005B            otr   ACC,StartDis ;disable SW trigger initially
0011C 1BFF005E            otr   ACC,FastRead ;Reset read latches
0011E 101FFFD2            movc  R0,-46       ;wait 20 us for power to settle
00120 FC9F      ArmLp:    br    lt,ArmLp     ;loop time 4 cycles = 1/3 us.
00121 A001                add   R0,1         ;  (less 16 cycles allowed for setup below)

00122 1BFF005F  Enable:   otr   ACC,SlowRead ;end read latch reset
00124 1BFF0061            otr   ACC,RstREn   ;hold read logic reset
00126 1BFF0053            otr   ACC,RstWr1Dis ;end write reset condition
00128 1BFF0051            otr   ACC,RstWr0Dis
0012A 1BFF0059            otr   ACC,TrigEn   ;allow triggering
0012C 041F0004            lr    R0,SoftTrig  ;see if SW trigger needed
0012E 4400                or    R0,R0
0012F FD07                br    EQ,Trigger   ; no
00130 0000                nop
00131 101FFFE2            movc  R0,-30       ; yes, wait 10 more us.
00133 FC9F      EnLp:     br    lt,EnLp
00134 A001                add   R0,1
00135 1BFF005A            otr   ACC,StartEn  ;    and then trigger it
00137 003FFD12  Trigger:  mov   R1,-750      ;start 1 ms loop waiting for trigger
00139 141F0050  TrigLp:   inr   R0,Status    ;16 cycle loop
0013B D819                tbr   R0,AcqBusy   ;see if triggered
0013C 7E9F0014            jc    NE,Collect   ; yes
0013E A021                add   R1,1         ; no, count time.  Time out?
0013F FC99                br    LT,TrigLp    ;  no, loop
00140 8009                mov   R0,NoTrig    ;  yes, exit with error code
00141 1BFF005B  shutdn:   otr   ACC,StartDis ;end software trigger
00143 1BFF0058            otr   ACC,TrigDis  ;disable triggering
00145 1BFF0050            otr   ACC,RstWr0En ;Reset and power down everytning
00147 1BFF0052            otr   ACC,RstWr1En
00149 1BFF0061            otr   ACC,RstREn
0014B 1BFF0054            otr   ACC,WrPwrDn
0014D 1BFF0056            otr   ACC,RdPwrDn
0014F 7F9FFF9E            jc    X,statout
```

```
00151 0000                  nop

00152 003FFFEF    Collect: mov   R1,-17        ;start 20 us. timing
00154 141F0050    ColLp:   inr   R0,Status     ;14 cycle loop
00156 D819                 tbr   R0,AcqBusy    ;still acquiring?
00157 FD06                 br    EQ,Unload     ; no, unload it
00158 A021                 add   R1,1          ; yes, time it
00159 FE9A                 br    NE,ColLp      ;  not out
0015A 800A                 mov   R0,NoLoad     ;  timed out, error
0015B 7F9FFFE4             jc    x,shutdn      ;    shut stuff off and post error
0015D 0000                 nop

0015E 1BFF0050    Unload:  otr   ACC,RstWrOEn ;Reset write logic
00160 1BFF0058             otr   ACC,TrigDis   ;prevent further triggering
                                               ; (this must occur after RstWrOEn)
00162 1BFF005B             otr   ACC,StartDis ;end software trigger pulse if given
00164 1BFF0054             otr   ACC,WrPwrDn   ;Turn off write power
00166 1BFF0060             otr   ACC,RstRDis   ;End Read Reset
00168 8184                 mov   R12,4         ;number of hybrids
00169 015F1000             mov   R10,ADBuf     ;set ptr to buffer
0016B 011F0063             mov   R8,O1YHi      ;load O1YHi address into register
0016D 001F0071    NewHyb:  mov   R0,Rd4En+2    ;set up start address for this hybrid
0016F 300C                 sub   R0,R12
00170 300C                 sub   R0,R12
                  ;The next section of code . . .
00171 1BE8                 otr   ACC,R8        ;Do one clock cycle
00172 1BFF0062             otr   ACC,O1YLow
00174 1BFF0065             otr   ACC,O2YHi
00176 1BFF0064             otr   ACC,O2YLow
00178 1BE0                 otr   ACC,R0        ;initiate start on channel
00179 1BFF0063             otr   ACC,O1YHi     ;Do second cycle
0017B B001                 sub   R0,1          ;convert address to disable
0017C 1BFF0062             otr   ACC,O1YLow
0017E 1BE0                 otr   ACC,R0        ;end start pulse
0017F 1BFF0065             otr   ACC,O2YHi
00181 1BFF0064             otr   ACC,O2YLow
                  ;which ends here, may be able to be simplified.
                  ;  some of the Analytec documentation indicates that
                  ;  the start pulse is positive edge triggered inside the
                  ;  analog memory chip.  If this is so, the above code
                  ;  need only trigger it initially and the go directly
                  ;  into fast read mode to skip unused cells at the beginning.
                  ;  It would look like this:
                  ;
                  ;        otr   ACC,FastRead ;this eliminates the need for
                  ;                            RstRDis as well
                  ;        otr   ACC,R0        ;send out start pulse
                  ;
                  ;  then skip 0 or more cells using fast read, which is already
                  ;  set up.

00183 8164                 mov   R11,4         ;number of channels per hybrid
00184 05BF0003             lr    R13,skip      ;number of cells to skip initially
00186 45AD                 or    R13,R13       ;test for zero
00187 FD09        SkipAgn: jc    eq,Read
00188 0000                 nop
00189 013F0065             mov   R9,O2YHi      ;set up second register needed
0018B 1BFF005E             otr   ACC,FastRead ;set to fast mode
0018D B1A1        SkipLp:  sub   R13,1         ;decrement and test counter     (2 cy)
0018E 1BE8                 otr   ACC,R8        ;Phase 1 pulse                  (3 cy)
```

```
0018F FE9D              br    NE,SkipLp     ;loop as needed              (2 cy)
00190 1BE9              otr   ACC,R9        ;Phase 2 pulse, even if br taken (3 cy)
                                            ;this loop is 10 cycles long, so
                                            ;skips a cell every 833 ns.

00191 1BFF005F   Read:  otr   ACC,SlowRead  ;return to slow mode
00193 1BE8              otr   ACC,R8        ;start a cycle to prime the A/D
00194 013F0064          mov   R9,02YLow     ;set up address needed by slow read
00196 1BFF0062          otr   ACC,01YLow
00198 1BFF0065          otr   ACC,02YHi     ;                    (1 + 3 cy)
0019A 141F0070          inr   R0,AD         ;                    (4 cy)
0019C 0000              nop                 ;                    (2 cy)
0019D 1BE9              otr   ACC,R9        ;                    (3 cy)
0019E 05BF0002          lr    R13,Samples   ;# to read           (3 cy)
001A0 05BF0002   RdAll: lr    R13,Samples   ;waste time          (3 cy)
001A2 1BE8       RPh0:  otr   ACC,R8        ;Phase 1Y High       (3 cy)
001A3 0000              nop                 ;waste time          (2 cy)
001A4 0000              nop                 ;                    (2 cy)
001A5 001F0062          mov   R0,01YLow     ;                    (2 cy)
001A7 1BE0       RPh1:  otr   ACC,R0        ;Phase 1Y Lo         (3 cy)
001A8 001F0065          mov   R0,02YHi      ;                    (2 cy)
001AA 003F0070          mov   R1,AD         ;                    (2 cy)
001AC B1A1              sub   R13,1         ;loop counter        (2 cy)
001AD 1BE0       RPh2:  otr   ACC,R0        ;Phase 2Y Hi         (3 cy)
001AE 1401              inr   R0,R1         ;Read A/D & restart  (3 cy)
001AF 080A              str   R0,R10        ;Store value in buf  (3 cy)
                 ;         NOTE: full 16 bits stored, top four are not
                 ;               valid and must be stripped before use.
001B0 1BE9       RPh3:  otr   Acc,R9        ;Phase 2Y Low        (3 cy)
001B1 FE90              jc    NE,RPh0       ;jmp based on R13    (4 cy)
001B2 A141              add   R10,1         ;update bufad        (2 cy) always

001B3 B161              sub   R11,1         ;see if more channels in this hybrid
001B4 FD0C              jc    EQ,NxtHyb     ; no, go to next one
001B5 0000              nop                 ; yes
001B6 041F0002          lr    R0,Samples    ;Number of cells skipped is 256 -
001B8 01BF00FF          mov   R13,255       ;  the number to be read, - 1 for bogus
001BA 31A0              sub   R13,R0        ;  read used to unload A/D last time
001BB 7C9FFFE3          jc    LT,RdAll      ;if all cells to be read, don't skip
001BD 0000              nop                 ;part of the skipped area will be the
001BE 7F9FFFC7          jc    x,SkipAgn     ;  remainder of this channel, the rest
001C0 0000              nop                 ;  will be the first of the next chan.
001C1 B181       NxtHyb: sub  R12,1
001C2 7E9FFFA9          jc    NE,NewHyb
001C4 0000              nop

001C5 1BFF0061   ConvEnd: otr ACC,RstREn    ;reset read logic and
001C7 1BFF0052          otr   ACC,RstWrlEn  ;  remaining write logic,
001C9 1BFF0056          otr   ACC,RdPwrDn   ;  and power it down
001CB 041F0005          lr    R0,CalFlag    ;Is calibration in progress?
001CD 7E9FFF20          jc    NE,statout    ; yes, return status
001CF 800B              mov   R0,AcqCal     ;  (if jc)
                                            ;DO NOT attempt to correct data and
                                            ;  return it if calibration is in process.
                                            ;  The results will be at best garbage.
                                            ;  At worst it might cause an overflow
                                            ;  or underflow error.

                 ;Now clean up and output the values using the following pseudo-code:
```

```
                    ;   FOR cell = 0 to Samples - 1        ;cell within each channel
                    ;       For chan = 0 to 15             ;channels multiplexed
                    ;            data = ADBuf[chan * Samples + cell]  ;raw data
                    ;            gain = ADGain[chan * 256 + cell + skip]  ;gain correction
                    ;            ofs = ADOfs[chan * 256 + cell + skip] ;offset correction
                    ;            res = (data * gain) / 4096 - offset  ;corrected value
                    ;
                    ;Notes:
                    ; 1.  Gain and offset values exist for all cells, data values
                    ;     exist only for requested cells, thus providing the tables
                    ;     with two different size multipliers.
                    ; 2.  Values were collected in Hybrid unload order for maximum
                    ;     unload speed, and thus minimum power consumption.  The
                    ;     data is interleaved throughout the 16 channels on the four
                    ;     hybrids, and thus must be retrieved in a different order
                    ;     so as to come out in true time order.
                    ; 3.  The gain values have been multiplied by 4096 to allow
                    ;     scaling both upward and downward.  the multiply should be
                    ;     done in a 32 bit register and then the result divided by
                    ;     4096 (or shift right 12).
                    ; 4.  The offsets are subtracted off after scaling, not before.
                    ;     This affects the way the offsets are calculated.

001D0 8100                  mov    R8,0          ;cell number (0 to Samples - 1)
001D1 053F0002              lr     R9,Samples    ;number of cells collected per channel
001D3 055F0003              lr     R10,Skip
001D5 8160          UnlLp:  mov    R11,0         ;channel number (0 to 15)
001D6 018B          UnlLp2: mov    R12,R11       ;calculate index to ADBuf
001D7 6D89                  muls   R12,R9        ;  chan * Samples
001D8 2188                  add    R12,R8        ;  +  cell
001D9 219F1000              add    R12,ADBuf     ;  +  base address
001DB 042C                  lr     R1,R12        ;get raw data
001DC 421F0FFF              and    XR0,0FFFh     ;mask garbage and setup double word
001DE 018B                  mov    R12,R11       ;calculate index to corrections
001DF E187                  slr    R12,L+8       ;  chan * 256
001E0 2188                  add    R12,R8        ;  + cell
001E1 218A                  add    R12,R10       ;  + skip
001E2 219F2000              add    R12,ADGain    ;  + base address
001E4 05AC                  lr     R13,R12       ;get gain factor
001E5 6E0D                  muls   XR0,R13       ;scale it
001E6 E614                  sar    XR0,R+12      ;correct shift
001E7 219F1000              add    R12,ADOfs-ADGain ;new base address, for offset
001E9 05AC                  lr     R13,R12       ;get offset correction
001EA 302D                  sub    R1,R13        ;subtract off, only 16 bit needed here
                    ;  now send word to host

001EB 141F0050      UnlSnd: inr    R0,Status     ;get status word
001ED D81D                  tbr    R0,InEmpty    ;see if data rcvd
001EE 7D1F0012              jc     eq,unlerr     ; yes, should not have been
001F0 D81C                  tbr    R0,OutFull    ;buffer empty?
001F1 FE99                  br     ne,UnlSnd     ; no, wait
001F2 BD6F                  cmp    R11,15        ;more channels at this cell offset?
001F3 183F005C              otr    R1,StatOff    ;place in data mode
001F5 183F0040              otr    R1,DMA        ;send data
001F7 7C9FFFDD              jc     LT,UnlLp2     ; yes (more channels)
001F9 A161                  add    R11,1         ;next channel (if jc)
001FA A101                  add    R8,1          ;next cell offset
001FB 3D09                  cmp    R8,R9         ;end of cells?
001FC 7C9FFFD7              jc     LT,UnlLp      ; no, continue
001FE 8004                  mov    R0,Cmpltd     ; yes, finished
```

```
001FF 7F9FFEEE            jc     x,statout
00201 0000                nop

00202 7F9FFEEB   unlerr:  jc     x,statout
00204 8005                mov    R0,IllData    ;     (used if jc)

                 ;  Cal0 loads the entire gain table with 1 (unity gain)
                 ;     and the offset table with 0.  These are obviously not
                 ;     accurate values, but serve as a starting point until
                 ;     calibration can be done.  This routine is called on
                 ;     cold start.  It also clears the CalFlag variable.

00205 001F2000   Cal0:    mov    R0,ADGain     ;address of beginning of gain table
00207 003F3000            mov    R1,ADOfs      ;address of beginning of offset table
00209 005F1000            mov    R2,4096       ;table size (256 * 16)
0020B 007F1000            mov    R3,4096       ;set gains to 1 ( * 4096)
0020D 8080                mov    R4,0          ;and offsets to 0
0020E 0860       CalLoop: str    R3,R0         ;store gain in table
0020F 0881                str    R4,R1         ;store offset in table
00210 A001                add    R0,1          ;next gain address
00211 B041                sub    R2,1          ;how many left
00212 FE9B                br     NE,CalLoop
00213 A021                add    R1,1          ;next offset address (done before br)
00214 8000                mov    R0,0
00215 081F0005            str    R0,CalFlag
00217 0E31                call   R16,R16

                 ;Routine to generate table of gains and offsets.  There are three
                 ;ways to call this routine.  The first call (which notices that
                 ;CalFlag is not set) sets it and initializes the work areas.
                 ;While CalFlag is set, this call cannot be repeated.  The A/D
                 ;cycle will not return data when CalFlag is set, only a
                 ;completion code.  This initial call also sets the skip value to
                 ;zero and Samples to 256 to provide a complete calibration table.
                 ;(This may not be a good idea, as timing and related temerature
                 ;variations may make calibration under actual skip and sample
                 ;conditions more accurate.)

                 ;Subsequent calls to calibrate should occur after each A/D cycle.
                 ;They provide the value (0 to 4000) representing the voltage used
                 ;in that calibration cycle.  These calls cause a summation of data
                 ;values to occur which will be used later by the curve fitter.
                 ;The actual values given will determine the scaling of subsequent
                 ;data.  i.e. 0 to 1000 for 0 to 1 v would yield 1 mv per count.

                 ;The final call to calibrate is done after all calibrate cycles
                 ;have been completed, and the summations have been done.  The
                 ;call is done with a data value of 4095 (0FFFh) which signals
                 ;completion of the calibration.  A least squares fit is done,
                 ;fitting the data to the equation y = m * x + b.  The values of m
                 ;and b are used as AdGain and AdOfs respectively.  CalFlag is
                 ;again cleared, allowing normal processing to resume.

                 ;Note that this routine destroys AdGain and AdOfs by useing the
                 ;same area of memory as workspace.  It should not be aborted
                 ;without completing.

                 ;        REGISTER USEAGE BY CALIBRATION ROUTINE
                 ;        R0 through R15 used temporarily
```

```
00218 041F0005     Calib:   lr    R0,CalFlag    ;is calibration in process?
0021A 7E9F002D              jc    NE,CalCont    ; yes
0021C 001F1000              mov   R0,4096       ;loop counter used in all cases
0021E 808F                  mov   R4,15         ; no, set up to start
0021F 089F0005              str   R4,CalFlag
00221 8080                  mov   R4,0
00222 089F0006              str   R4,CalVSum
00224 089F0007              str   R4,CalCnt
00226 003F2000              mov   R1,CalSums
00228 005F3000              mov   R2,CalSqrs
0022A 007F5000              mov   R3,CalVctr
0022C 0881         CalZer:  str   R4,R1         ;clear array area
0022D A021                  add   R1,1
0022E 0882                  str   R4,R2
0022F A041                  add   R2,1
00230 0882                  str   R4,R2         ;squares and vectors are double prec.
00231 A041                  add   R2,1
00232 0883                  str   R4,R3
00233 A061                  add   R3,1
00234 0883                  str   R4,R3
00235 B001                  sub   R0,1
00236 FE95                  br    NE,CalZer
00237 A061                  add   R3,1          ;  (if br)
00238 043F0003              lr    R1,Skip
0023A 083F0008              str   R1,CalSkH
0023C 089F0003              str   R4,Skip       ;skip nothing
0023E 041F0002              lr    R0,Samples
00240 081F0009              str   R0,CalSaH
00242 001F0100              mov   R0,256        ;read all
00244 081F0002              str   R0,Samples
00246 7F9FFEA7              jc    x,statout     ;Return with completed status
00248 8004                  mov   R0,Cmpltd     ;  (if jc)

00249 3CFF0FFF     CalCont: cmp   R7,0FFFh      ;termination code?
0024B 003F2000              mov   R1,CalSums    ;set up pointers to arrays
0024D 005F3000              mov   R2,CalSqrs
0024F 7D1F002F              jc    eq,CalFin     ; yes, compute new corrections
00251 007F5000              mov   R3,CalVctr    ;  (either way)
00253 049F0006              lr    R4,CalVSum    ;add current value to sum
00255 2087                  add   R4,R7
00256 089F0006              str   R4,CalVSum
00258 049F0007              lr    R4,CalCnt     ;and increment count
0025A A081                  add   R4,1
0025B 089F0007              str   R4,CalCnt
0025D 009F1000              mov   R4,ADBuf
0025F 04A4         CalLp:   lr    R5,R4         ;get raw value
00260 40BF0FFF              and   R5,0FFFh      ;strip off garbage
00262 04C1                  lr    R6,R1         ;add it to sum
00263 20C5                  add   R6,R5
00264 08C1                  str   R6,R1
00265 A021                  add   R1,1          ;ready for next pass
00266 0345                  mov   XR10,R5        ;square raw value
00267 6F45                  muls  XR10,R5
00268 0502                  lr    R8,R2         ;add square to sum of squares
00269 A041                  add   R2,1          ;  (squares is double word)
0026A 0522                  lr    R9,R2
0026B 231A                  add   XR8,XR10
0026C 0922                  str   R9,R2
0026D B041                  sub   R2,1
```

```
0026E 0902              str   R8,R2
0026F A042              add   R2,2          ;ready for next pass
00270 0345              mov   XR10,R5       ;calculate raw value * correct value
00271 6F47              muls  XR10,R7
00272 0503              lr    R8,R3         ;add to total
00273 A061              add   R3,1
00274 0523              lr    R9,R3
00275 231A              add   XR8,XR10
00276 0923              str   R9,R3
00277 B061              sub   R3,1
00278 0903              str   R8,R3
00279 B001              sub   R0,1          ;loop counter
0027A 7E9FFFE3          jc    NE,CalLp      ; repeat until done
0027C A062              add   R3,2          ;   (if jc)
0027D 7F9FFE70          jc    x,statout     ;then exit with completion code
0027F 8004              mov   R0,Cmpltd     ;  (if jc)

00280 009F2000   CalFin: mov   R4,AdGain     ;Set up pointers specific to this part
00282 00BF3000          mov   R5,AdOfs
00284 04DF0007          lr    R6,CalCnt
00286 04FF0006          lr    R7,CalVSum
00288 0501       CFLp:  lr    R8,R1         ;get sum of data
00289 A021              add   R1,1          ;ready for next
0028A 0168              mov   R11,R8        ;copy it
0028B 8140              mov   R10,0         ;  (unsigned to double precision)
0028C 7B46              divs  XR10,R6       ;divide by number of points.
0028D 012B              mov   R9,R11        ;R9 is an important intermediate
0028E 0168              mov   R11,R8        ;sum * R9
0028F 8140              mov   R10,0
00290 6F49              muls  XR10,R9
00291 0582              lr    R12,R2        ;subtract from sum of squares
00292 A041              add   R2,1
00293 05A2              lr    R13,R2
00294 A041              add   R2,1
00295 339A              sub   XR12,XR10     ;denominator of gain term
00296 0543              lr    R10,R3        ;get constant vector
00297 A061              add   R3,1
00298 0563              lr    R11,R3
00299 A061              add   R3,1
0029A 01E9              mov   R15,R9
0029B 81C0              mov   R14,0
0029C 6FC7              muls  XR14,R7       ;subtr. sum of correct values * R9
0029D 335E              sub   XR10,XR14     ;numerator of gain term
0029E 454A              or    R10,R10       ;if numerator isn't too large,
0029F FE84              br    NE,CFSD
002A0 0000              nop
002A1 E74B              sar   XR10,L+12     ; multiply it by 4096 to scale gain
002A2 FF82              br    x,CFSC
002A3 0000              nop
002A4 E794       CFSD:  sar   XR12,R+12     ; otherwise divide denominator by 4096
002A5 7B5C       CFSC:  divs  XR10,XR12     ;gain
002A6 0964              str   R11,R4        ;  and store it
002A7 A081              add   R4,1
002A8 6F48              muls  XR10,R8       ;gain * sum of data
002A9 E754              sar   XR10,R+12     ;  undo scaling for this
002AA 0387              mov   XR12,R7       ;sum of correct values
002AB 339A              sub   XR12,XR10
002AC 7B86              divs  XR12,R6       ;divide by number of data samples
002AD 09A5              str   R13,R5        ;offset
002AE B001              sub   R0,1          ;loop counter
```

```
002AF 7E9FFFD7          jc      NE,CFLp
002B1 A0A1              add     R5,1          ;  (if jc)
002B2 043F0008          lr      R1,CalSkH
002B4 083F0003          str     R1,Skip       ;Restore skip and sample values
002B6 043F0009          lr      R1,CalSaH
002B8 083F0002          str     R1,Samples
002BA 8020              mov     R1,0
002BB 083F0005          str     R1,CalFlag    ;Clear CalFlag
002BD 7F9FFE30          jc      x,statout
002BF 8004              mov     R0,Cmpltd     ;  (if jc)

                ;Routine to calculate checksum of code RAM to determine if it
                ;has been corrupted.
                ;This routine should be called, followed by "str XR0,CodeChk"
                ;when the system is cold started, and after any modification
                ;to the code RAM (stri to 8000h to FFFFh).  The code write
                ;command does NOT do this automatically.  It is necessary to
                ;do a code execute at address 8011h after code writes.
                ;Chksum may be called anytime the integrity of the code RAM is
                ;in question.  The value in XR0 should agree with the value
                ;stored in CodeChk.  Chksum is automatically called before a warm
                ;start and if code RAM doesn't check, a cold start is done
                ;instead.
                ;Destroys ACC and R2, returns to R16

002C0 03FF8000  Chksum: mov     ACC,ImgDst    ;where to check
002C2 8200              mov     XR0,0         ;initial sum
002C3 845F      cklp:   lri     R2            ;value at address
002C4 2202              add     XR0,R2        ;add to sum
002C5 964E              inr     XR2,ACC       ;copy to 16 bit register so compare will work
002C6 3C7FFFFF          cmp     R3,CRAMe      ;done?
002C8 FE9A              br      NE,cklp       ; no, cont
002C9 A3E1              add     ACC,1         ;next place to check
002CA 0E31              call    R16,R16       ;return with results in XR0

                ;RAM IMAGE DATA TO LOAD AT 8000h in code area
002CB           Image:
002CB 0068              dwm     CodeLow       ;CMD =  0 (0xxxh) - set CODE flag & 12 low bits
002CC 006C              dwm     DataLow       ;CMD =  1 (1xxxh) - set DATA flag & 12 low bits
002CD 0070              dwm     StHiRng       ;CMD =  2 (2xxxh) - hi bits, form start adr
002CE 0077              dwm     StHiRd        ;CMD =  3 (3xxxh) - hi bits, form adr and rd 1
002CF 007E              dwm     StHiWr        ;CMD =  4 (4xxxh) - hi bits, form adr and wr 1
002D0 0085              dwm     Execute       ;CMD =  5 (5xxxh) - hi bits, form adr and ex
002D1 008C              dwm     EndHiRd       ;CMD =  6 (6xxxh) - hi bits, form end adr & rd rng
002D2 0092              dwm     EndHiWr       ;CMD =  7 (7xxxh) - hi bits, form end adr & wr rng
002D3 0098              dwm     IORd          ;CMD =  8 (8xxxh) - 12 bit adr and Input word
002D4 009B              dwm     IOWr          ;CMD =  9 (9xxxh) - 12 bit adr and Output word
002D5 00A8              dwm     SetSkp        ;CMD = 10 (Axxxh) - 12 bit skip count (mod 16)
002D6 00AE              dwm     ManAcq        ;CMD = 11 (Bxxxh) - 12 bit cnvt cnt (mod 16) and arm
002D7 004F              dwm     AutoAcq       ;CMD = 12 (Cxxxh) - reserved for auto skip and cnt
002D8 0218              dwm     Calib         ;CMD = 13 (Dxxxh) - calibration routine
002D9 004F              dwm     monitor       ;CMD = 14 (Exxxh) - reserved
002DA 004F              dwm     monitor       ;CMD = 15 (Fxxxh) - reserved

002DB 004F              dwm     monitor       ;return address for execute, etc
002DC 0048              dwm     warmend       ;link address for warm start
002DD 0E3F02C0          call    R16,Chksum    ;callable routine to update checksum
002DF 081F0000          str     R0,CodeChk    ;  resides at 8012h
002E1 083F0001          str     R1,CodeChk+1
```

```
002E3 03FF8010            mov   ACC,CmdRtn
002E5 863F                lri   R16
002E6 0E31                call  R16,R16    ;(return to monitor)
002E7           imgend:
00000                     END
```

00000008	ABORTED	00000019	ACQBUSY	0000000B	ACQCAL
00000070	AD	00001000	ADBUF	0000001B	ADBUSY
00002000	ADGAIN	00003000	ADOFS	000000FE	ARM
00000120	ARMLP	0000004F	AUTOACQ	00000205	CAL0
00000007	CALCNT	00000249	CALCONT	00000280	CALFIN
00000005	CALFLAG	00000218	CALIB	0000020E	CALLOOP
0000025F	CALLP	00000009	CALSAH	00000008	CALSKH
00003000	CALSQRS	00002000	CALSUMS	00005000	CALVCTR
00000006	CALVSUM	0000022C	CALZER	00000288	CFLP
000002A5	CFSC	000002A4	CFSD	000002C0	CHKSUM
000002C3	CKLP	0000001F	CMD	00008000	CMDJMP
00008010	CMDRTN	00000004	CMPLTD	00000000	CODECHK
00000068	CODELOW	00000025	COLD	00000029	COLDLP
00000002	COLDSTAT	00000152	COLLECT	00000154	COLLP
0000001A	CONVBUSY	000001C5	CONVEND	0000FFFF	CRAME
0000001E	CTS	0000006C	DATALOW	00000040	DMA
00000122	ENABLE	0000008C	ENDHIRD	00000092	ENDHIWR
00000133	ENLP	00000085	EXECUTE	0000005E	FASTREAD
00000005	ILLDATA	000002CB	IMAGE	00008000	IMGDST
000002E7	IMGEND	0000001D	INEMPTY	00000000	INIT
00000000	INTMASK	00000098	IORD	0000009B	IOWR
000000AE	MANACQ	000000B6	MANLIM	000000B7	MANOK
0000004F	MONITOR	000000C7	MONNXTR	000000BC	MONRD
000000C6	MONRDTA	000000C8	MONSND	000000E9	MONWDTA
000000D9	MONWR	000000EA	MONWRNXT	0000016D	NEWHYB
00000053	NEXTWORD	00000006	NODATA	0000000A	NOLOAD
00000018	NOTARM	00000017	NOTREND1	00000016	NOTREND2
00000015	NOTREND3	00000014	NOTREND4	00000009	NOTRIG
00000013	NOTWEND1	00000012	NOTWEND2	00000011	NOTWEND3
00000010	NOTWEND4	000001C1	NXTHYB	00000063	O1YHI
00000062	O1YLOW	00000065	O2YHI	00000064	O2YLOW
00000000	OKSTAT	0000001C	OUTFULL	00008012	POSTCHK
00000110	PWRUP	00000068	RD1DIS	00000069	RD1EN
0000006A	RD2DIS	0000006B	RD2EN	0000006C	RD3DIS
0000006D	RD3EN	0000006E	RD4DIS	0000006F	RD4EN
000001A0	RDALL	00000056	RDPWRDN	00000057	RDPWRUP
00000191	READ	00000060	RESRDST	000001A2	RPH0
000001A7	RPH1	000001AD	RPH2	000001B0	RPH3
00000060	RSTRDIS	00000061	RSTREN	00000051	RSTWR0DIS
00000050	RSTWR0EN	00000053	RSTWR1DIS	00000052	RSTWR1EN
00000002	SAMPLES	000000A8	SETSKP	00000141	SHUTDN
00000003	SKIP	00000187	SKIPAGN	0000018D	SKIPLP
0000005F	SLOWREAD	000000D5	SNDERR	00000004	SOFTTRIG
00000049	STARTCOM	0000005B	STARTDIS	0000005A	STARTEN
000000F7	STAT01	0000005C	STATOFF	0000005D	STATON
000000EF	STATOUT	00000050	STATUS	00000077	STHIRD
00000070	STHIRNG	0000007E	STHIWR	00000070	SWINT
00000058	TRIGDIS	00000059	TRIGEN	00000137	TRIGGER
00000139	TRIGLP	00000202	UNLERR	000001D5	UNLLP
000001D6	UNLLP2	0000015E	UNLOAD	000001EB	UNLSND
00000044	WARM	00000048	WARMEND	00008011	WARMLINK
00000001	WARMSTAT	00000007	WRINT	00000054	WRPWRDN
00000055	WRPWRUP				

APPENDIX #2

CLOCK GENERATOR UNIT

PROGRAM REVIEW

Binneg Lao

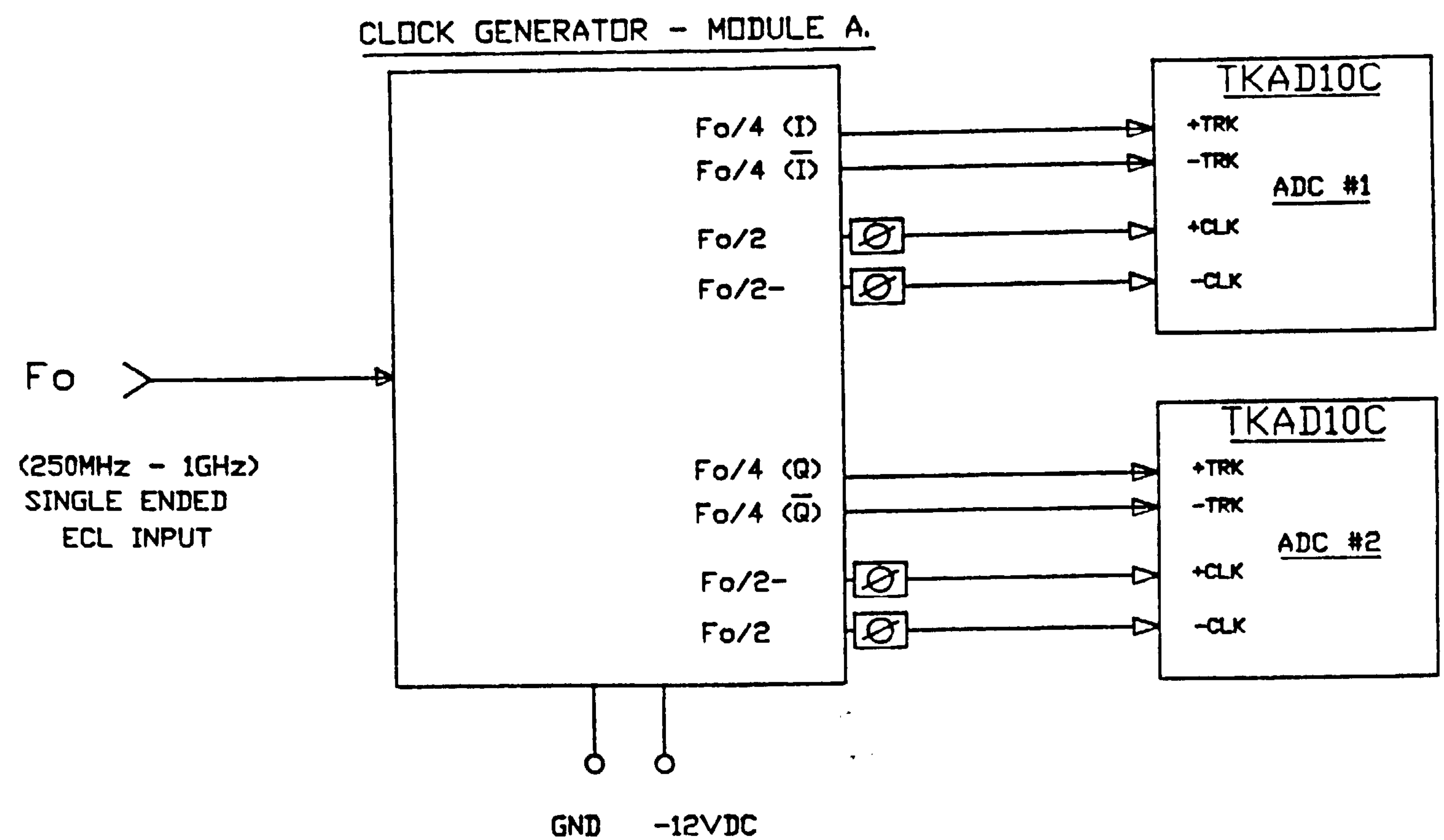
Carl Price

Dec 07, 1989

AMERASIA: CLOCK GENERATOR - MODULE A.
PROPOSED SYSTEM APPLICATION.

Sierra Monolithics

12/5/89 CP_CLKA2



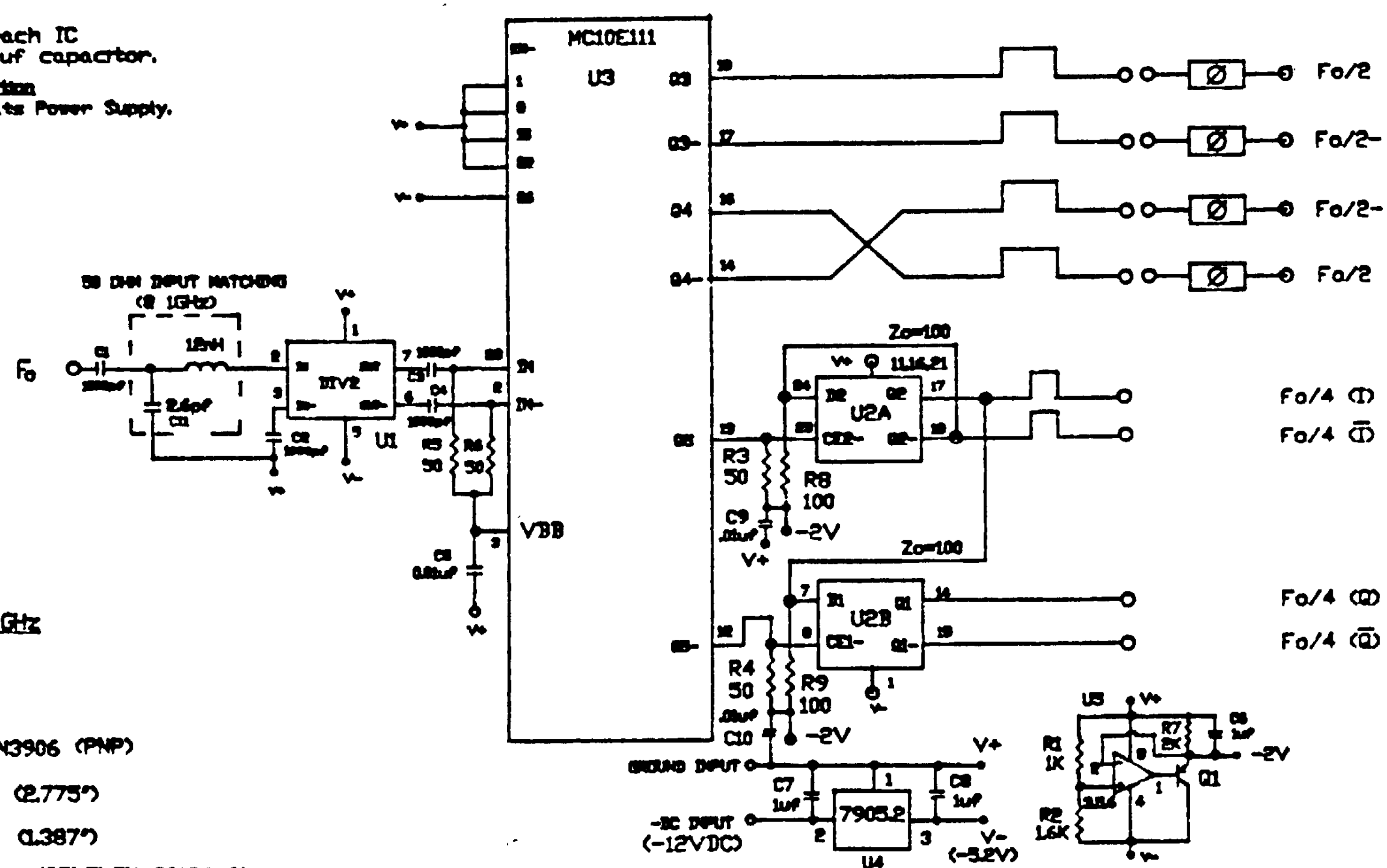
AMERASIA: CLOCK GENERATOR – MODULE A. | Sierra Monolithics

12/04/89 (CP_CLKA)

PCB = G10, 15MIL THICKNESS
50 OHMS = 28MIL LINE WIDTH
100 OHMS = 5MIL LINE WIDTH
ALL SIGNAL LINES ARE 50 OHMS
EXCEPT AS SPECIFIED

NOTE: Decouple each IC
with a 0.01uf capacitor.

Total Power Consumption
= 2.1 Watts @ -12Volts Power Supply.



F_o = 250MHz to 1GHz
U1 = SP8822B1/DG
U2 = MC10E131FN
U4 = MC7905.2CT
U5 = MC1458U
Q1 = 2N6191 or 2N3906 (PNP)

= 400ps delay (2.775")

= 200ps delay (1.387")

Ø = PHASE Trimmer (SEMFLEX 30136-01)
(198ps ± 16ps)

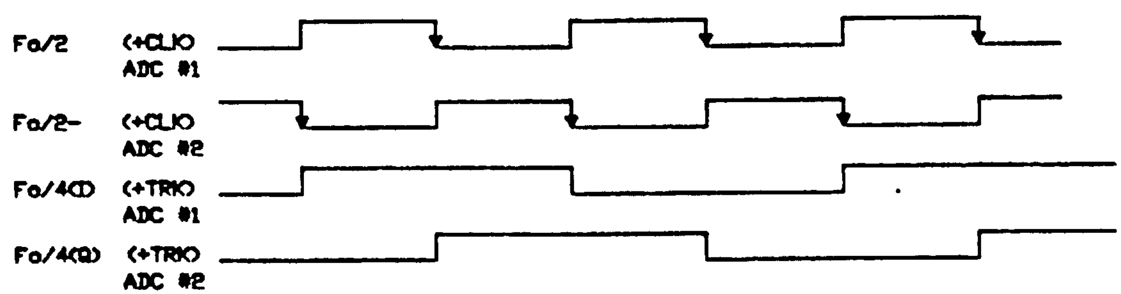
o = 50 OHM SMA CNX
(Sealectro50-651-0000-31)

AMERASIA: CLOCK GENERATOR – MODULE A.
GENERATED OUTPUT WAVEFORMS.

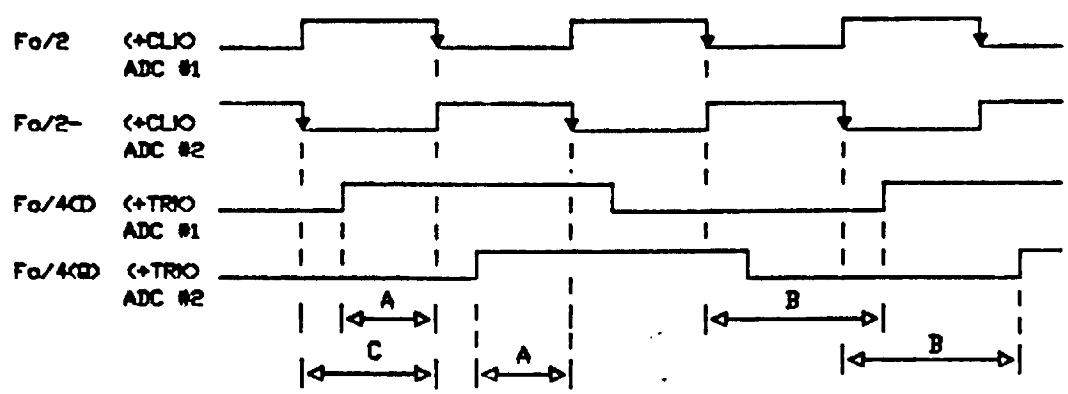
Sierra Monolithics

12/4/89 (CP_CLKA1)

OUTPUT WAVEFORMS with Fo = 250MHz.



OUTPUT WAVEFORMS with Fo = 1GHz.



SET-UP TIME:
$A \geq 725$ ps

HOLD-TIME:
$B \geq 1.275$ ns

PULSE-WIDTH
C = 1 ns PW
@ 1GHz

NOTE: The Waveforms drawn for Fo/2 & Fo/2- are the Outputs from the PHASE TRIMMERS.

12/05/89 (clka1.doc)

SIERRA MONOLITHIC'S EXPECTED FUNCTIONAL & PERFORMANCE OPERATION FOR THE MODULE A CLOCK GENERATOR.

Module A will take an unbalanced ECL Input signal (Fo)ranging in frequency from 250Mhz to 1Ghz, then produce as outputs, FOUR balanced ECL compatible signals, namely Fo/2, Fo/2-, Fo/4 (I), & Fo/4 (Q).

Fo/4 (I) shall be in phase with the POSITIVE edge of Fo/2, & Fo/4 (Q) shall be in phase with the NEGATIVE edge of Fo/2. Hence, Fo/4 (Q) is always 90 degrees out of phase with respect to Fo/4 (I).

Over the Fo INPUT frequency range, Fo/4(I) output changes logic states at least 725ps before and at least 725ps after the negative clock edge of Fo/2. Likewise, it is true for Fo/4(Q) relative to the negative clock edge of Fo/2-.

ELECTRICAL CHARACTERISTICS:

POWER CONSUMPTION:
2.1 Watts (approx).

INPUT SIGNAL PORT (Fo):

Input Sensitivity:	100mV (min) Single-Ended.
Input Impedance:	50 ohms at 1Ghz - with matching.
	20 - j50 at 1GHz - w/o matching.
	100 - j175 at 250MHz - w/o matching.

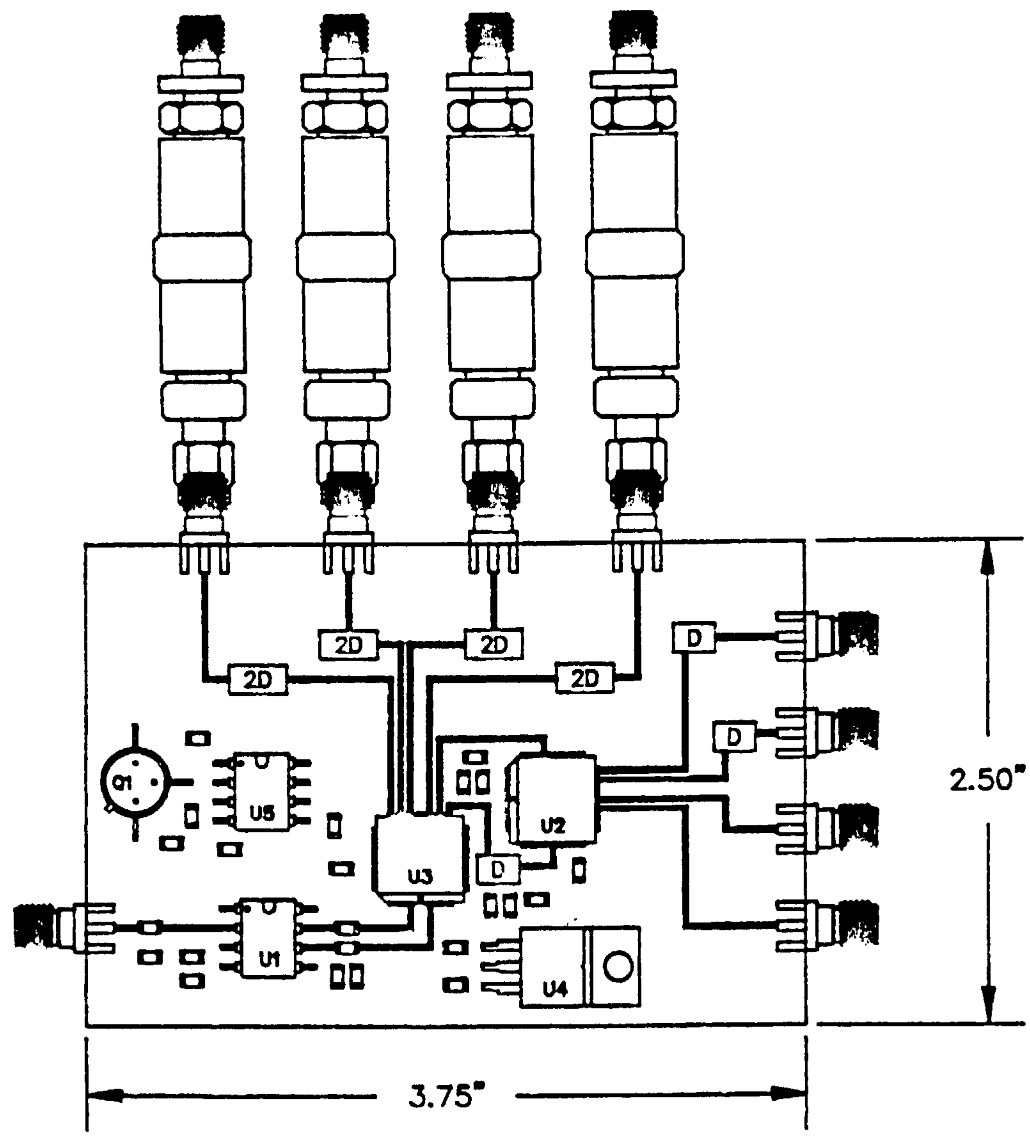
OUTPUT SIGNAL LEVELS: (Fo/2, Fo/2-, Fo/4(I), Fo/4(Q)

	0C		25C		75C	
	Min	Max	Min	Max	Min	Max
Output HIGH Volt	-1.02	-0.84	-0.98	-0.81	-0.92	-0.735
Output LOW Volt	-1.95	-1.63	-1.95	-1.63	-1.95	-1.595
Input HIGH Volt	-1.17	-0.84	-1.13	-0.81	-1.07	-0.735
Input LOW Volt	-1.95	-1.48	-1.95	-1.48	-1.95	-1.450
Output Rise/Fall time (20-80%)	300ps	800ps	300ps	800ps	300ps	800ps
SKEW (U3 outputs)		100ps		100ps		100ps

NOTE: Motorola states that 10E Series type IC's do not have a problem driving 100E or 100K chips.

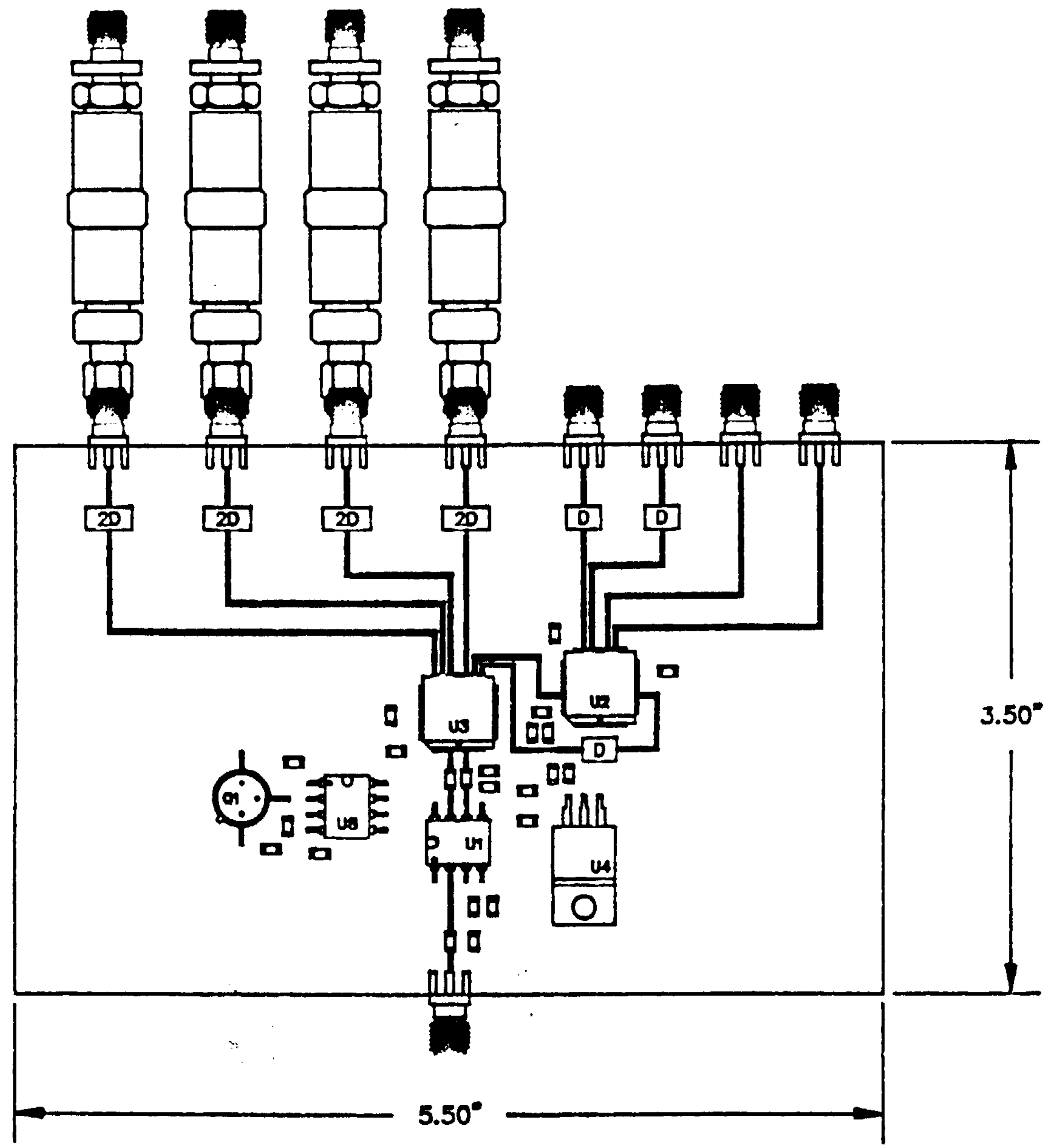
MODULE A CONCEPTUAL LAYOUT

Sierra Monolithics



MODULE A CONCEPTUAL LAYOUT

Sierra Monolithics



DKS0108-1289

PARTS LIST - AMERASIA: CLOCK GENERATOR - MODULE A.

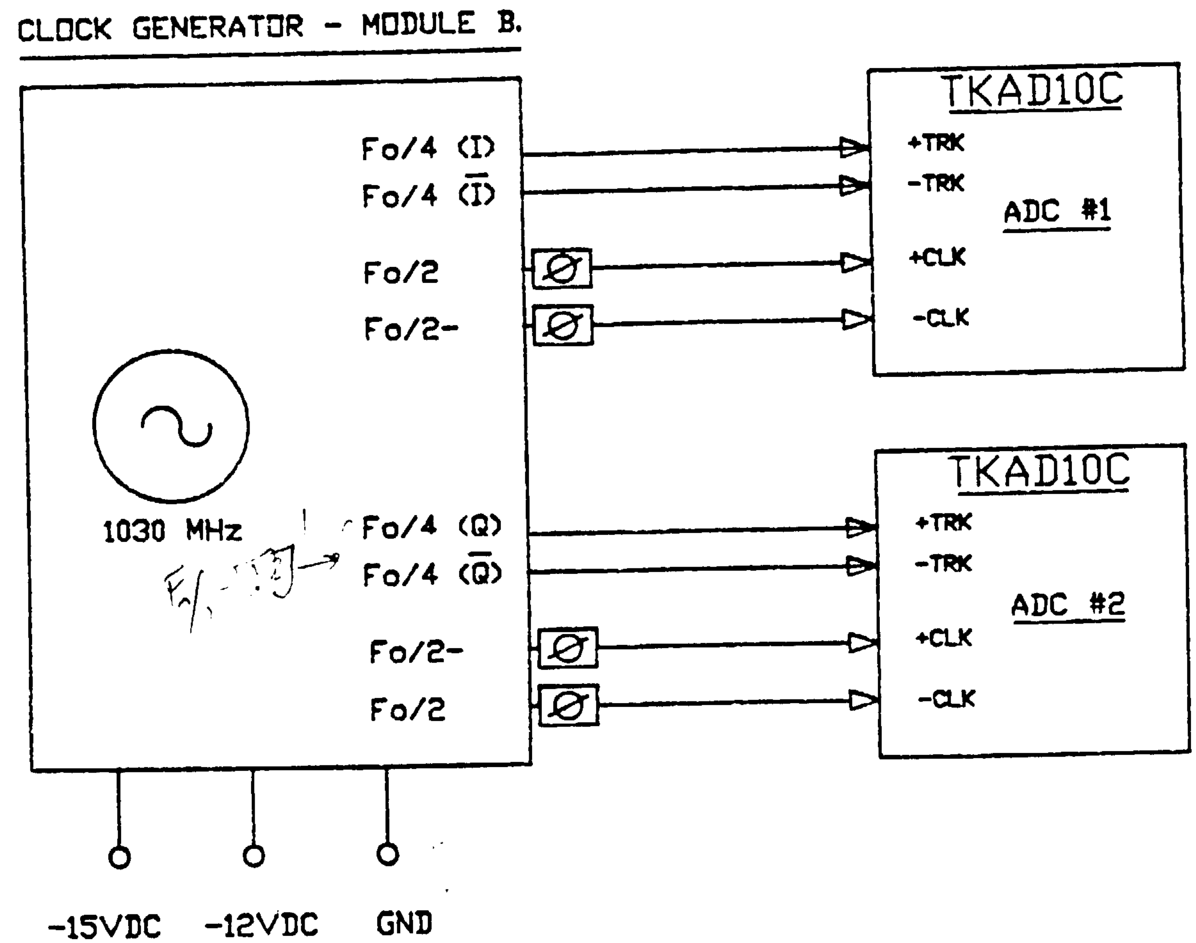
REF	PART NO.	DISTRIBUTOR	SHIP (WEEKS)	QTY	PRICE (EACH)	PRICE (TOTAL)
U1	SP8822B1/DG (Plessey)	Select Electronics (Arlene:714 739-8891) Sample requested(11/8/89)	14	1	$49.76	$49.76
U2	MC10E131FN (Motorolla)	WYLE	12	1	$17.31	$17.31
U3	MC10E111FN (Motorolla)	HAMILTON AVNET (213)217-6830	STOCK	1	$28.23	$28.23
U4	MC7905.2CT (Motorolla)	Hamilton Avnet (213)217-6830 call local distr...	12	1	$ .54	$ .54
U5	MC1458U (Motorolla)	Hamilton Avnet (213)217-6830	STOCK	1	$ .96	$ .96
Q1 ~~D1~~	2N6191 (PNP) ~~1N914~~	Local Distr...	STOCK	1	$ 1.00	$ 1.00
-	30136-01 (Semflex)	Cain Technology (213)326-5236 (plus $200 setup charge?)	?	4	$101.00	$404.00
SMA	50-651-0000-31	Selectro Corp (914)698-5600) Sealectro West (213)990-8131	?	9	$ 10.00	$ 90.00

TOTAL ESTIMATED COST: **$542.04**

AMERASIA: CLOCK GENERATOR - MODULE B.
PROPOSED SYSTEM APPLICATION.

Sierra Monolithics

12/5/89 CP_CLKB2



AMERASIA: CLOCK GENERATOR - MODULE B. PRELIMINARY DESIGN.

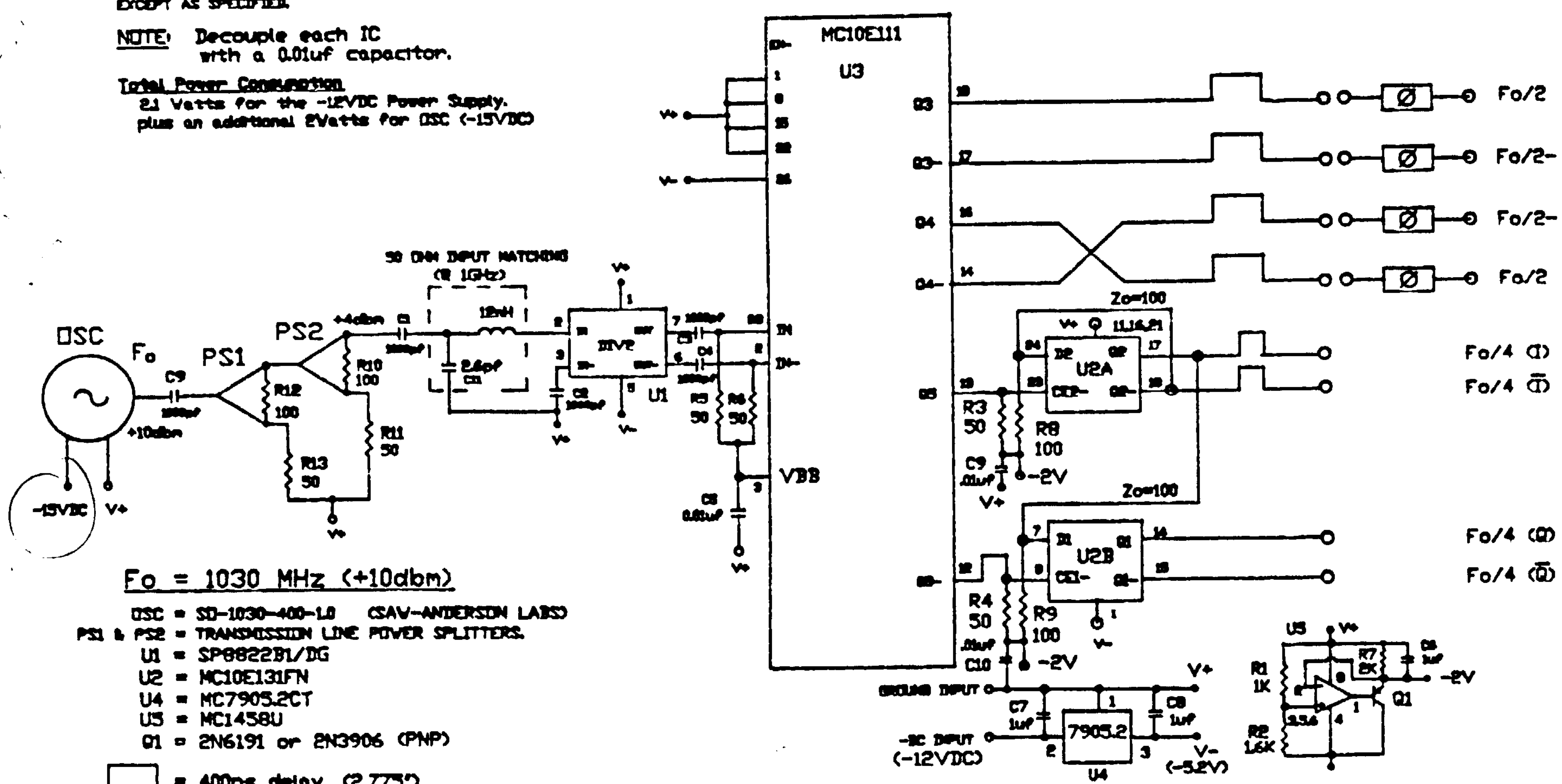
Sierra Monolithics

12/04/89 (CP_CLKB)

PCB = G10, 15MIL THICKNESS
50 OHMS = 26MIL LINE WIDTH.
100 OHMS = 5MIL LINE WIDTH.
ALL SIGNAL LINES ARE 50 OHMS
EXCEPT AS SPECIFIED.

NOTE: Decouple each IC with a 0.01uf capacitor.

Total Power Consumption
2.1 Watts for the -12VDC Power Supply.
plus an additional 2Watts for OSC (-15VDC)



Fo = 1030 MHz (+10dbm)

OSC = SD-1030-400-1.0 (SAV-ANDERSON LABS)
PS1 & PS2 = TRANSMISSION LINE POWER SPLITTERS.
U1 = SP8822B1/DG
U2 = MC10E131FN
U4 = MC7905.2CT
U5 = MC1458U
Q1 = 2N6191 or 2N3906 (PNP)

= 400ps delay (2.775")

= 200ps delay (1.387")

Ø = PHASE Trimmer (SEMFLEX 30136-01) (198ps ± 16ps)

O = 50 OHM SMA CNX (Sealectro 50-651-0000-31)

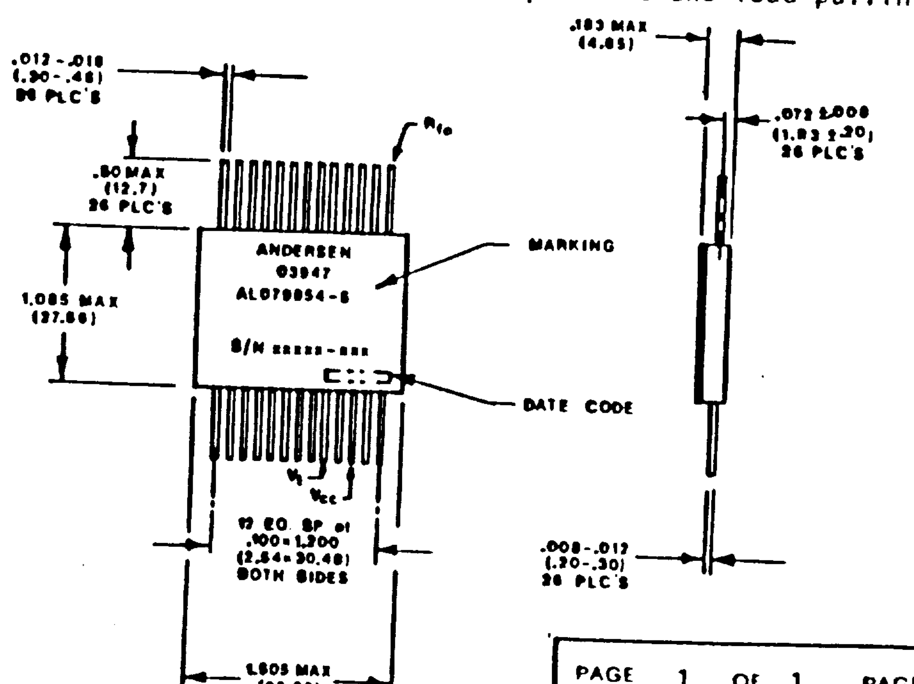
Pag. 7

ⓐ ANDERSEN LABORATORIES CODE IDENT 03947	SPECIFICATION	SPEC NUMBER	REV
PREPARED BY / DATE	TYPE SURFACE ACOUSTIC WAVE HYBRID OSCILLATOR		
TITLE STANDARD SPECIFICATIONS			
APPROVED BY / DATE	REV. APPROVED BY / DATE	MODEL NO. SO-1030-400-1.0	

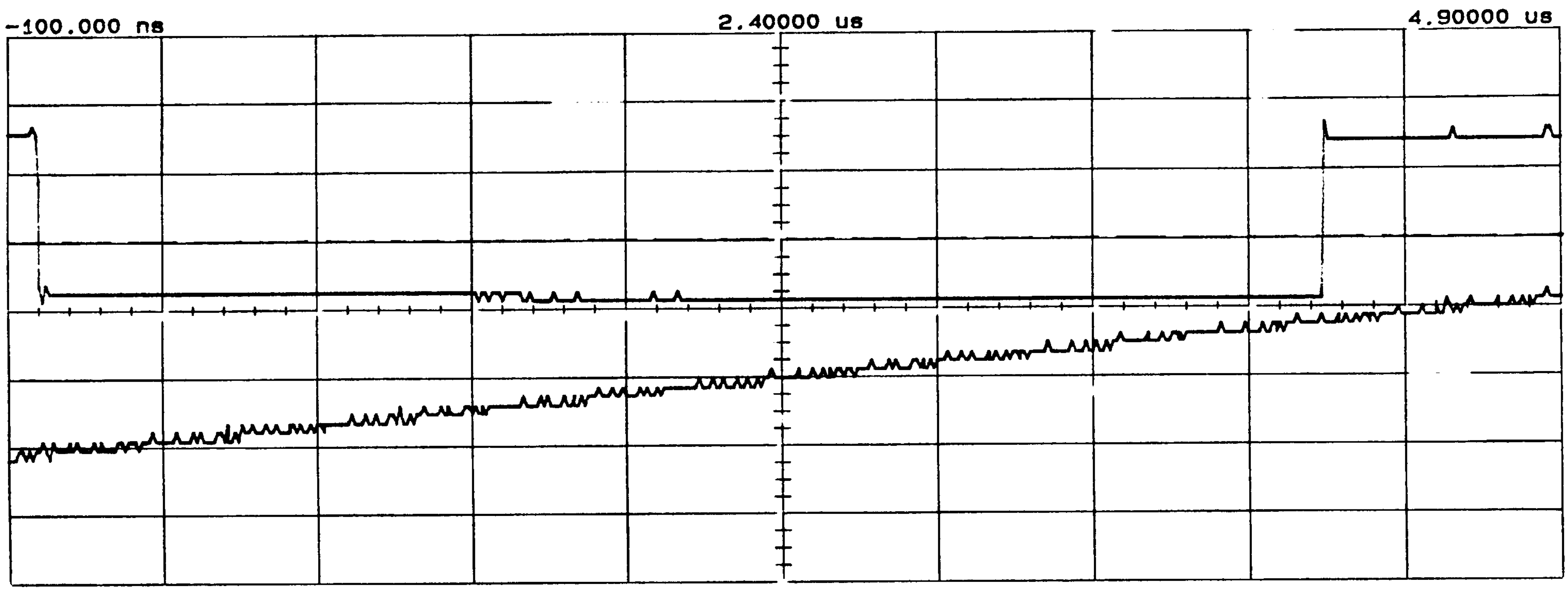
Carrier Frequency (fo)	1030 MHz
Frequency Deviation	.04 $(T\text{-}Tref)^2$ ppm max., Tref = 25°C nominal
Output Power	+10dBm ± 2dBm (1v pk) into 50Ω
Operating Temperature Range	-45°C to +85°C
Tuning Range (Δf)	400 KHz minimum
Tuning Voltage (Δv)	0 - 12 V
Spurious - Harmonic	30dBc
- Non-harmonic	-60dBc
Phase Noise	-65dBc/Hz @ 100 Hz
	-85dBc/Hz @ 1 KHz
	-105dBc/Hz @ 10 KHz
Power Supply	+15 V DC ± 5%, 125 mA

Note(s): 1) Tuning range supplied is sufficient to maintain the specified carrier frequency over the effects of temperature and load pulling.

Outline:



APPENDIX #3

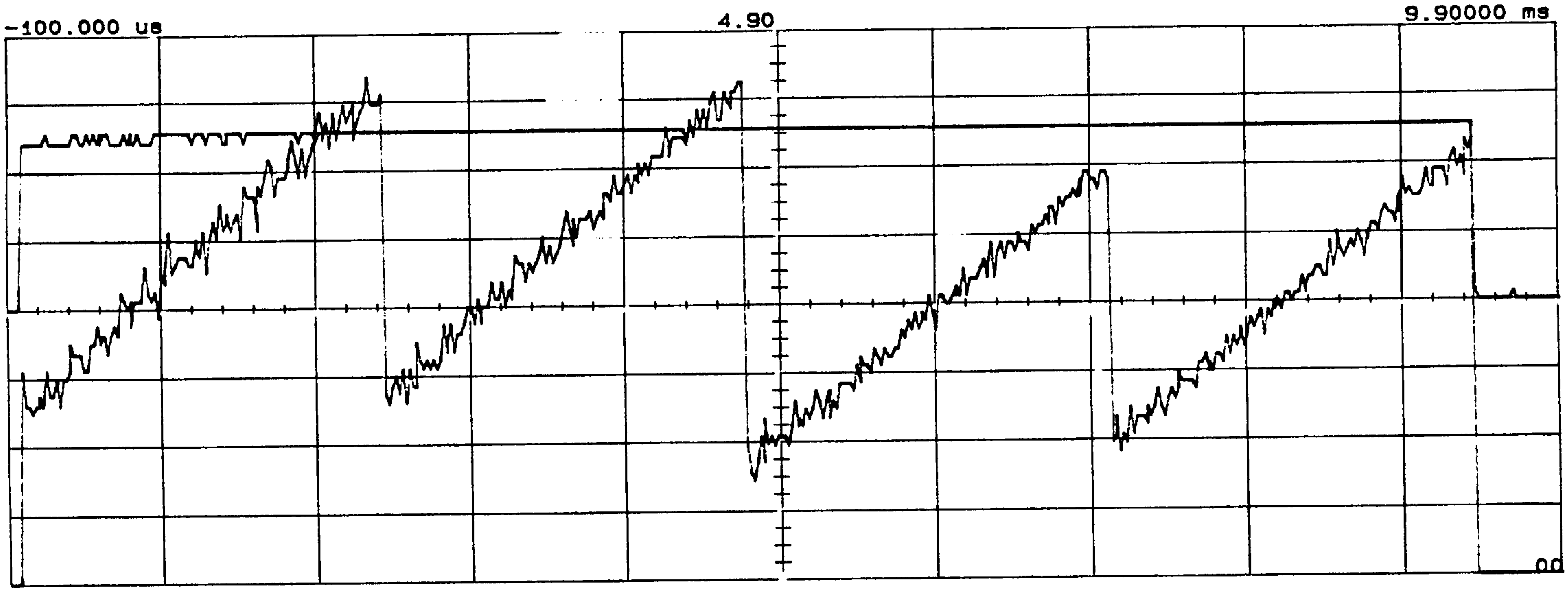


	Timebase	Delay/Pos	Reference	Mode
Main	500 ns/div	-100.000 ns	Left	Realtime (NORMAL)

	Sensitivity	Offset	Probe	Coupling
Channel 1	2.00 V/div	0.00000 V	10.00 : 1	dc (1M ohm)
Channel 2	1.00 V/div	-1.00000 V	10.00 : 1	dc (1M ohm)

Trigger mode : Edge
On Negative Edge Of Chan1
Trigger Level
Chan1 = 2.00000 V (noise reject OFF)
Holdoff = 40.000 ns

FIGURE A3-1, Input Signal

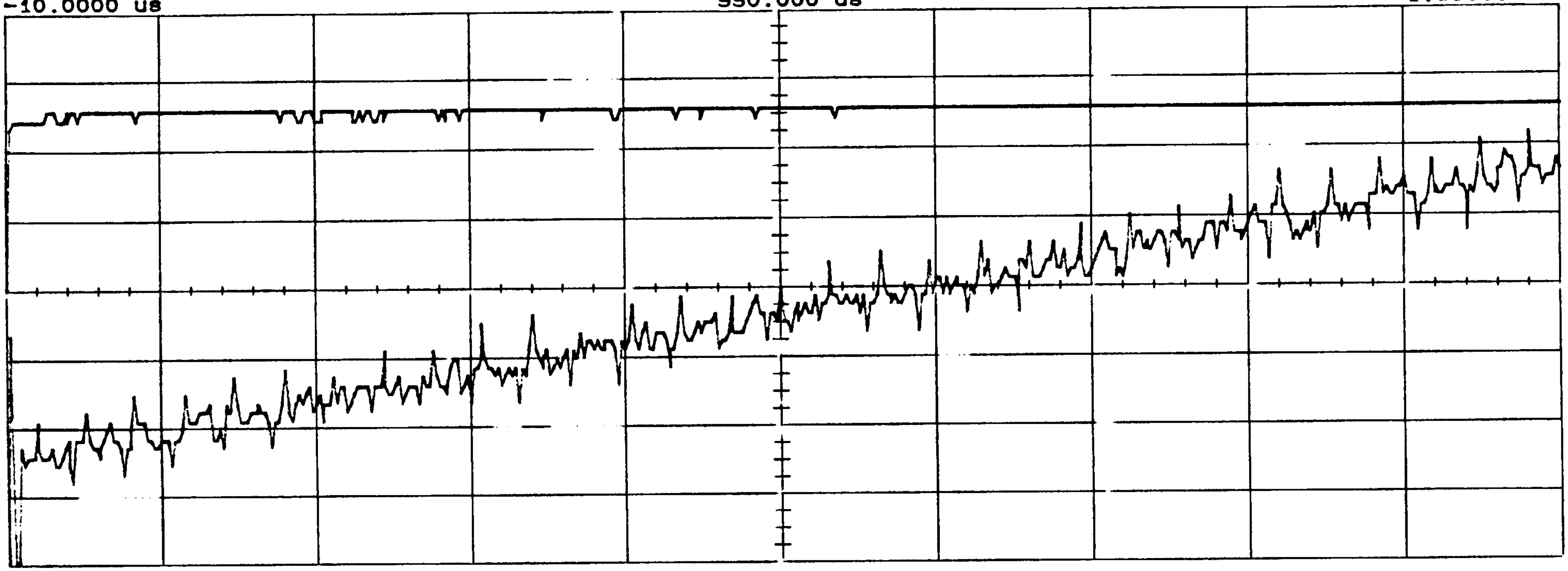


	Timebase	Delay/Pos	Reference	Mode
Main	1.00 ms/div	-100.000 us	Left	Realtime (NORMAL)

	Sensitivity	Offset	Probe	Coupling
Channel 1	2.00 V/div	0.00000 V	10.00 : 1	dc (1M ohm)
Channel 2	1.00 V/div	3.00000 V	10.00 : 1	dc (1M ohm)

Trigger mode : Edge
On Positive Edge Of Chan1
Trigger Level
Chan1 = 2.00000 V (noise reject OFF)
Holdoff = 40.000 ns

FIGURE A3-2, Output Signal (Readback)

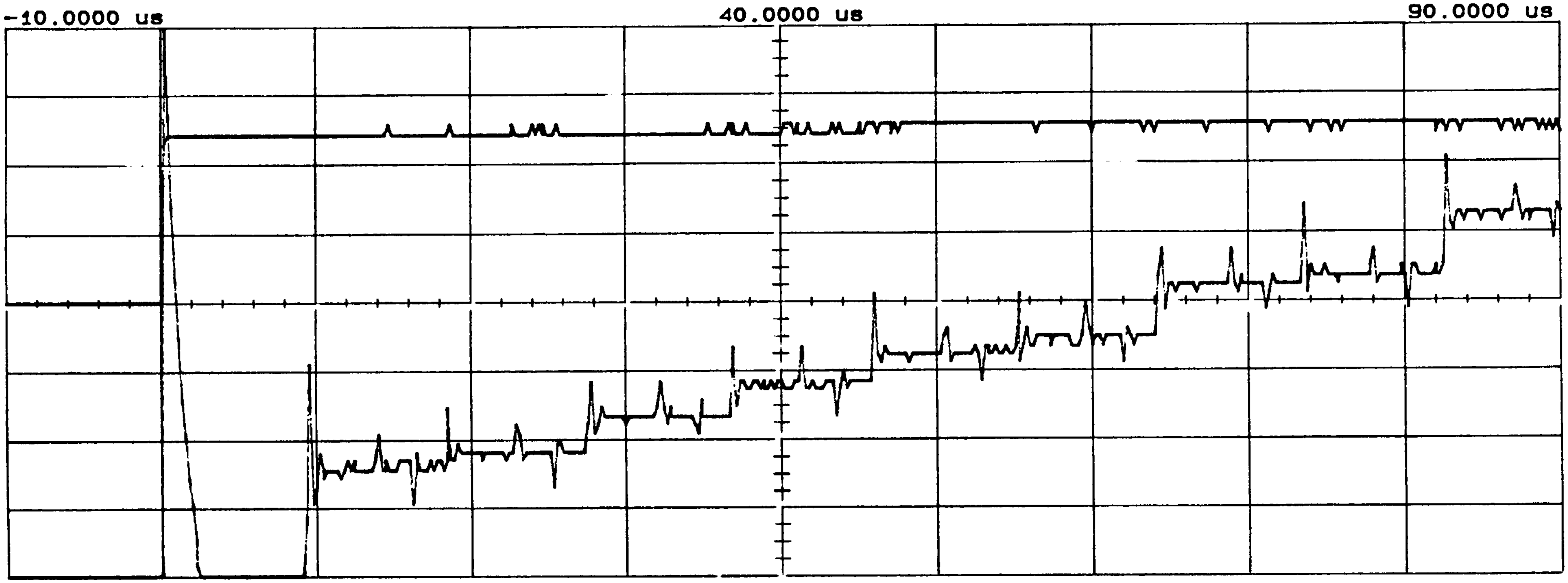


	Timebase	Delay/Pos	Reference	Mode
Main	200 us/div	-10.0000 us	Left	Realtime (NORMAL)

	Sensitivity	Offset	Probe	Coupling
Channel 1	2.00 V/div	0.00000 V	10.00 : 1	dc (1M ohm)
Channel 2	1.00 V/div	3.00000 V	10.00 : 1	dc (1M ohm)

Trigger mode : Edge
On Positive Edge Of Chan1
Trigger Level
Chan1 = 2.00000 V (noise reject OFF)
Holdoff = 40.000 ns

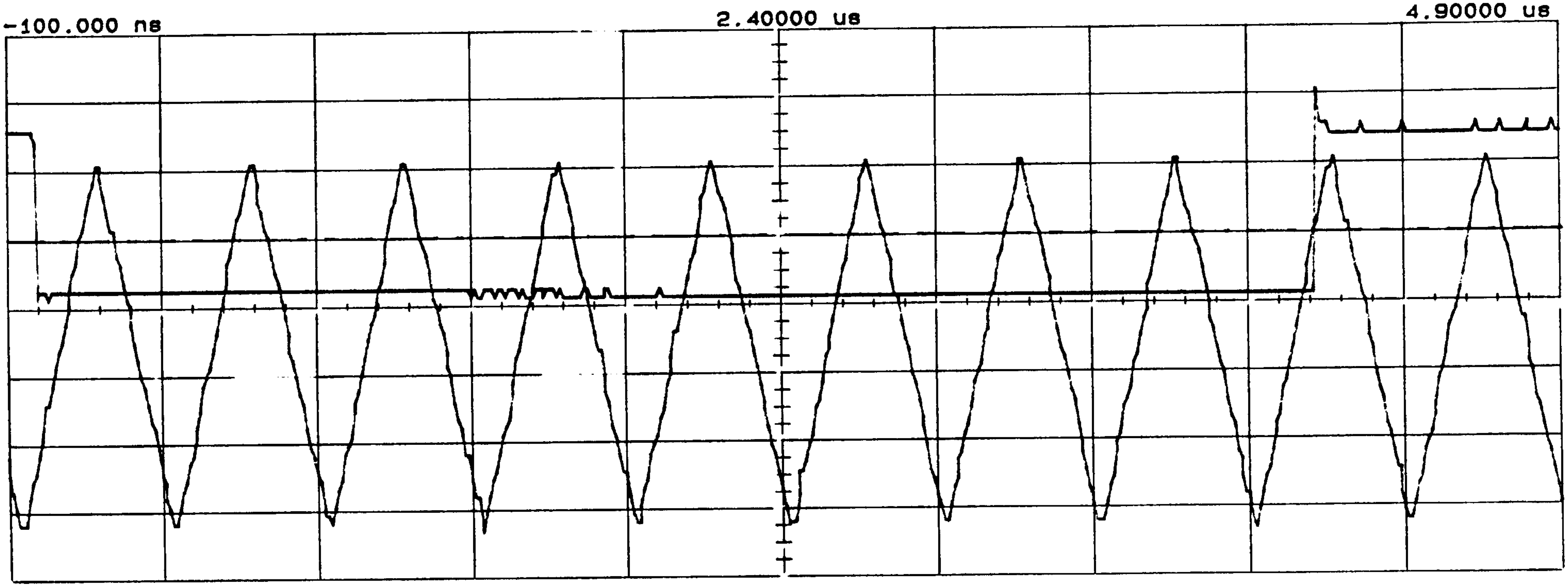
FIGURE A3-3, Output Signal Observed with Faster Timebase



	Timebase	Delay/Pos	Reference	Mode
Main	10.0 us/div	-10.0000 us	Left	Realtime (NORMAL)
	Sensitivity	Offset	Probe	Coupling
Channel 1	2.00 V/div	0.00000 V	10.00 : 1	dc (1M ohm)
Channel 2	1.00 V/div	3.00000 V	10.00 : 1	dc (1M ohm)

Trigger mode : Edge
On Positive Edge Of Chan1
Trigger Level
Chan1 = 2.00000 V (noise reject OFF)
Holdoff = 40.000 ns

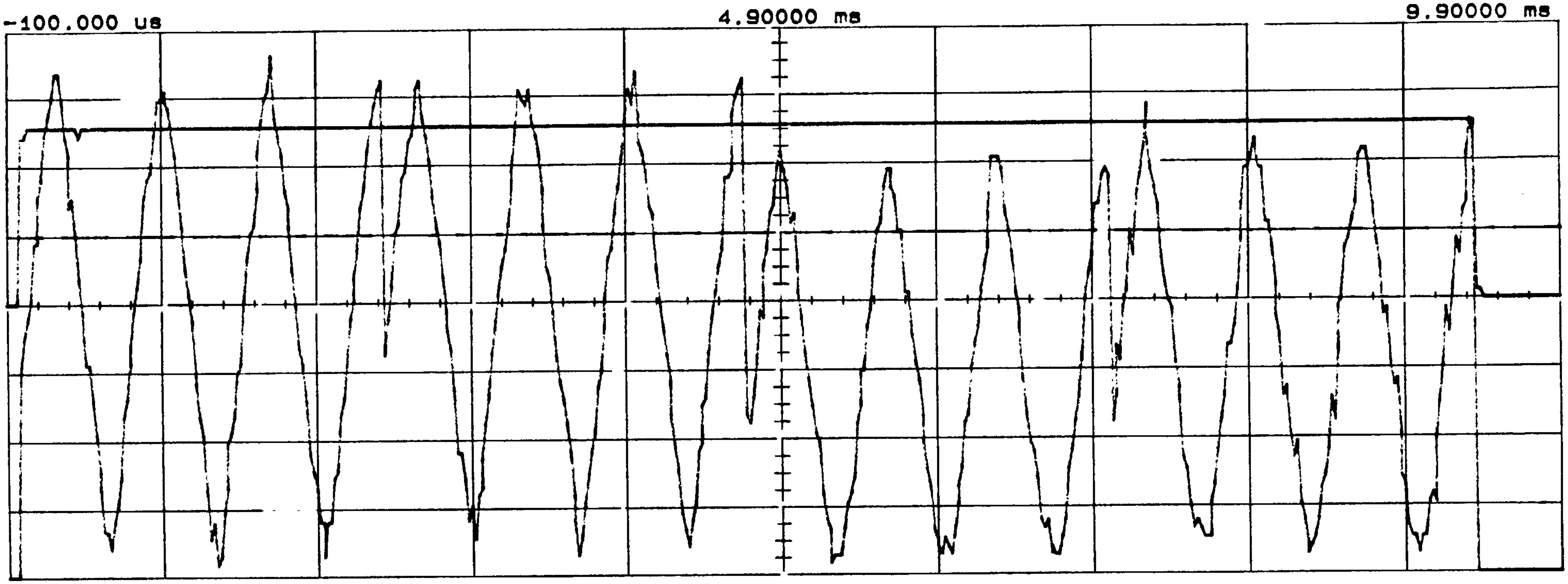
FIGURE A3-4, Output Signal Observed with Faster Timebase



	Timebase	Delay/Pos	Reference	Mode
Main	500 ns/div	-100.000 ns	Left	Realtime (NORMAL)
	Sensitivity	Offset	Probe	Coupling
Channel 1	2.00 V/div	0.00000 V	10.00 : 1	dc (1M ohm)
Channel 2	1.00 V/div	-1.00000 V	10.00 : 1	dc (1M ohm)

Trigger mode : Edge
On Negative Edge Of Chan1
Trigger Level
Chan1 = 2.00000 V (noise reject OFF)
Holdoff = 40.000 ns

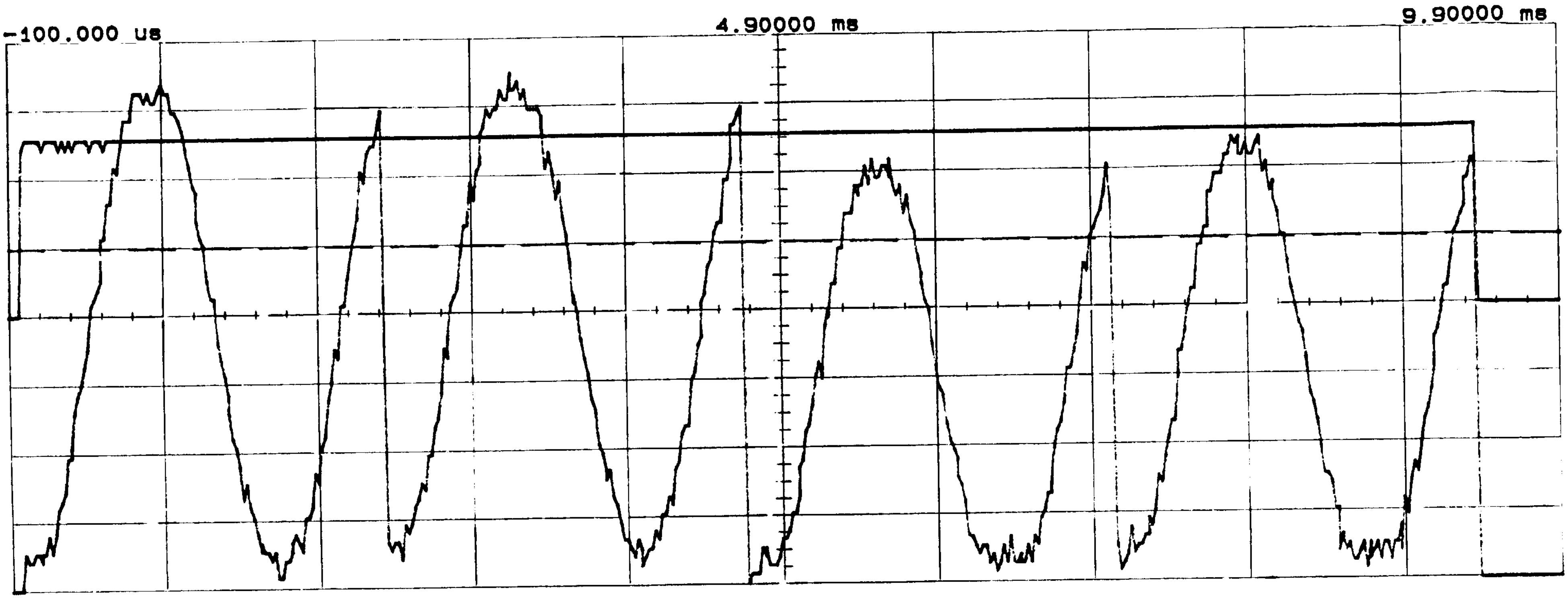
FIGURE A3-5, 2 MHz Triangular Input Signal



	Timebase	Delay/Pos	Reference	Mode
Main	1.00 ms/div	-100.000 us	Left	Realtime (NORMAL)
	Sensitivity	**Offset**	**Probe**	**Coupling**
Channel 1	2.00 V/div	0.00000 V	10.00 : 1	dc (1M ohm)
Channel 2	1.00 V/div	3.00000 V	10.00 : 1	dc (1M ohm)

Trigger mode : Edge
On Positive Edge Of Chan1
Trigger Level
Chan1 = 2.00000 V (noise reject OFF)
Holdoff = 40.000 ns

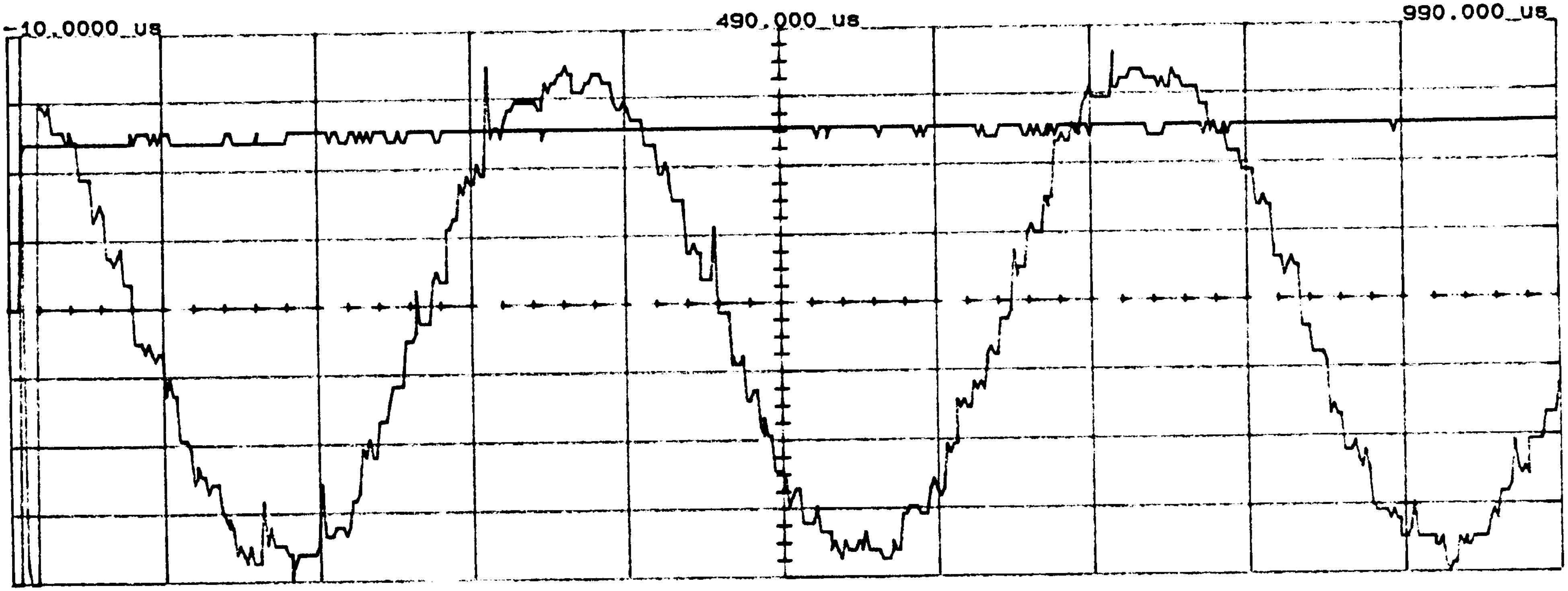
FIGURE A3-6, Output Signal with Triangular Signal Input



	Timebase	Delay/Pos	Reference	Mode
Main	1.00 ms/div	-100.000 us	Left	Realtime (NORMAL)
	Sensitivity	Offset	Probe	Coupling
Channel 1	2.00 V/div	0.00000 V	10.00 : 1	dc (1M ohm)
Channel 2	1.00 V/div	3.00000 V	10.00 : 1	dc (1M ohm)

Trigger mode : Edge
On Positive Edge Of Chan1
Trigger Level
Chan1 = 2.00000 V (noise reject OFF)
Holdoff = 40.000 ns

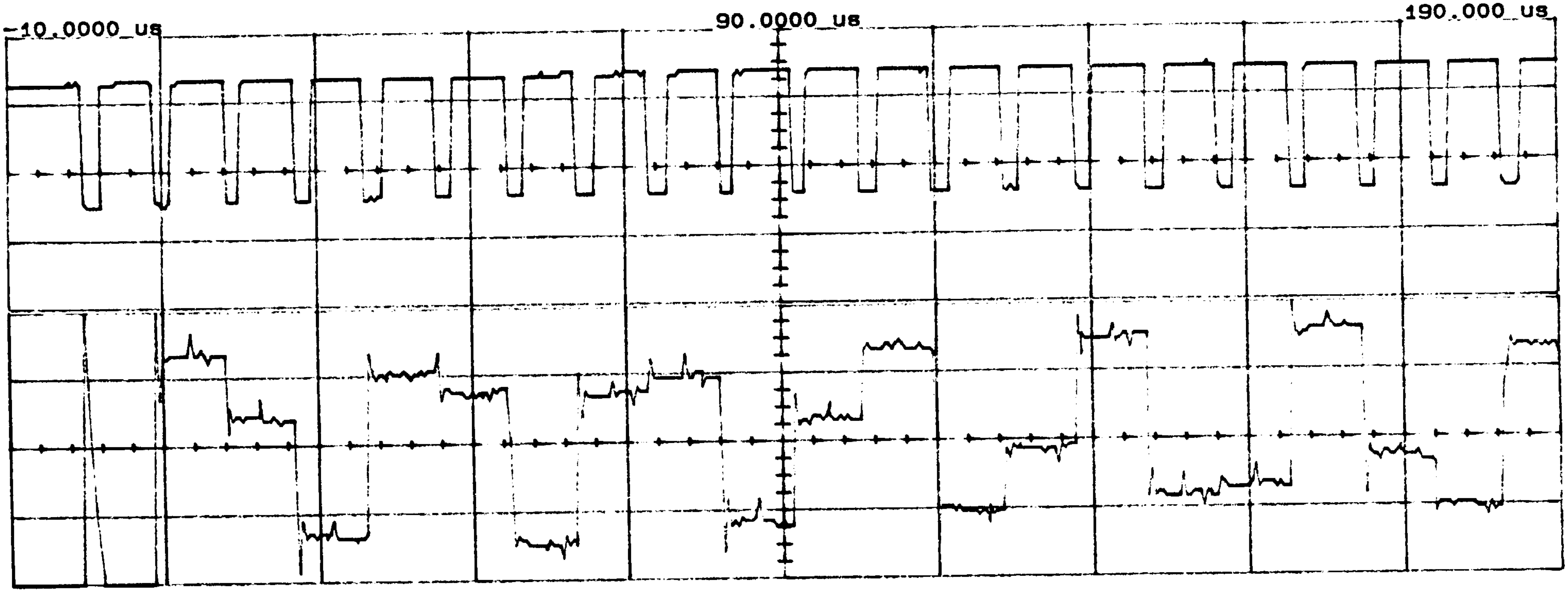
FIGURE A3-7, 1 MHz Sinewave Input Signal



	Timebase	Delay/Pos	Reference	Mode
Main	100 us/div	-10.0000 us	Left	Realtime (NORMAL)
	Sensitivity	**Offset**	**Probe**	**Coupling**
Channel 1	2.00 V/div	0.00000 V	10.00 : 1	dc (1M ohm)
Channel 2	1.00 V/div	3.00000 V	10.00 : 1	dc (1M ohm)

Trigger mode : Edge
On Positive Edge Of Chan1
Trigger Level
Chan1 = 2.00000 V (noise reject OFF)
Holdoff = 40.000 ns

FIGURE A3-8, Output Signal with Sinewave Input



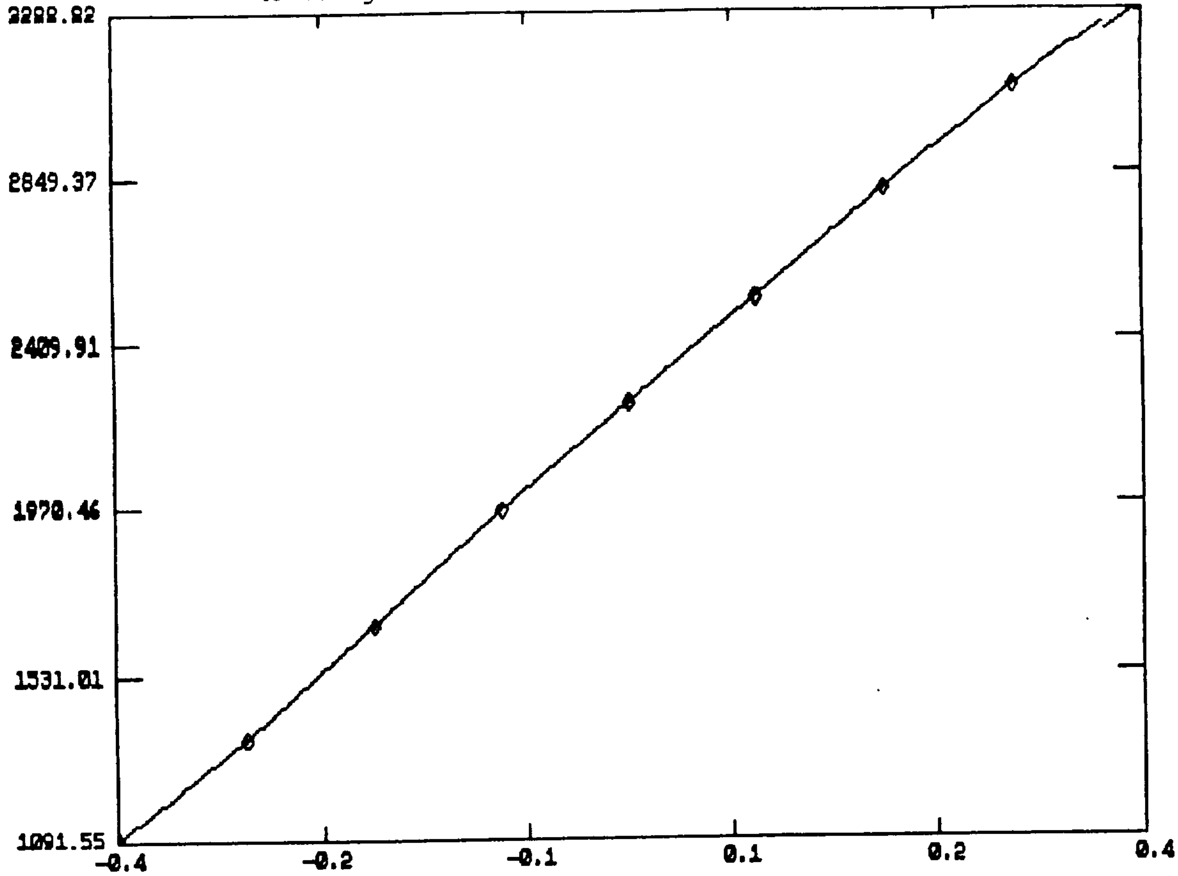
	Timebase	Delay/Pos	Reference	Mode
Main	20.0 us/div	-10.0000 us	Left	Realtime (NORMAL)

	Sensitivity	Offset	Probe	Coupling
Channel 1	4.00 V/div	0.00000 V	10.00 : 1	dc (1M ohm)
Channel 2	2.00 V/div	3.00000 V	10.00 : 1	dc (1M ohm)

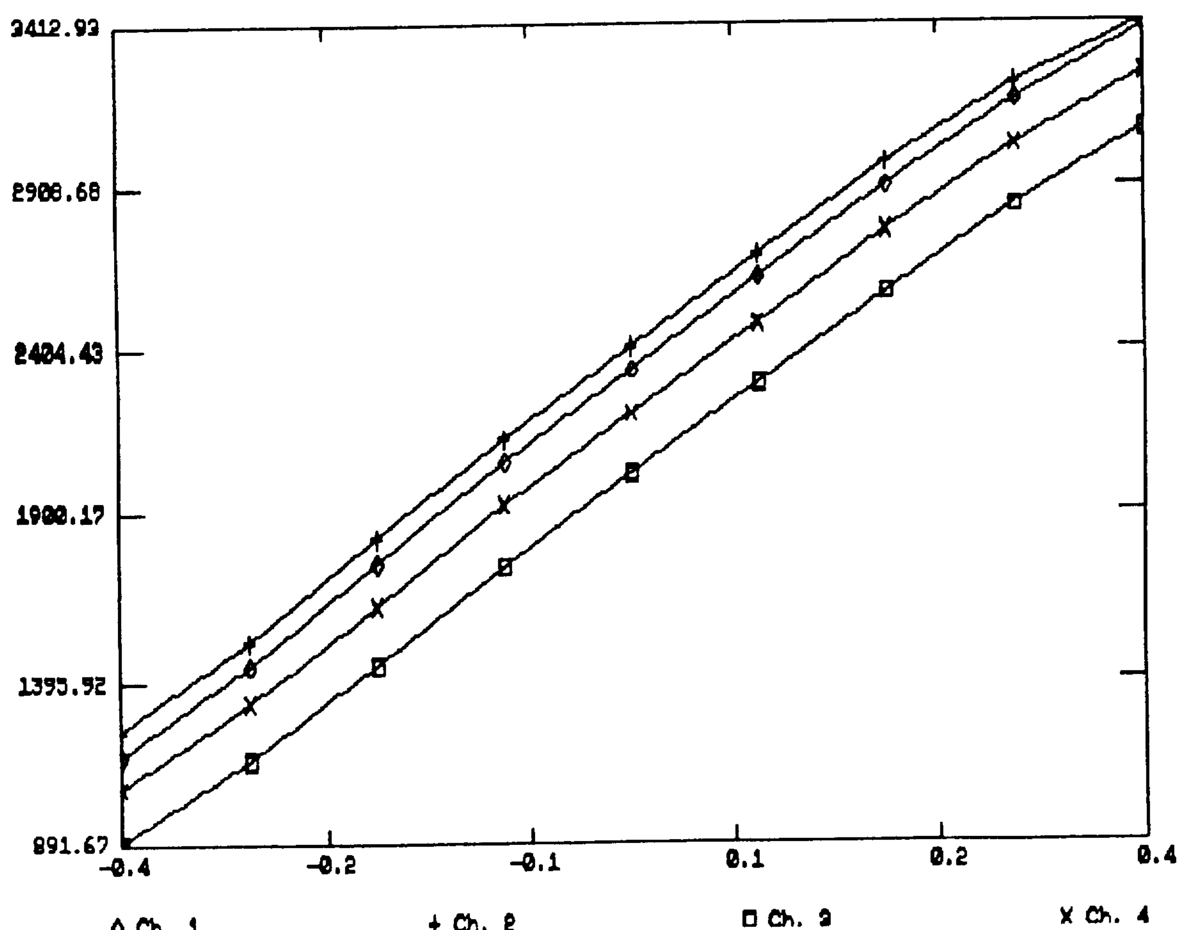
Trigger mode : Edge
On Positive Edge Of Ext1
Trigger Level
Ext1 = 1.87500 V (noise reject OFF)
Holdoff = 40.000 ns

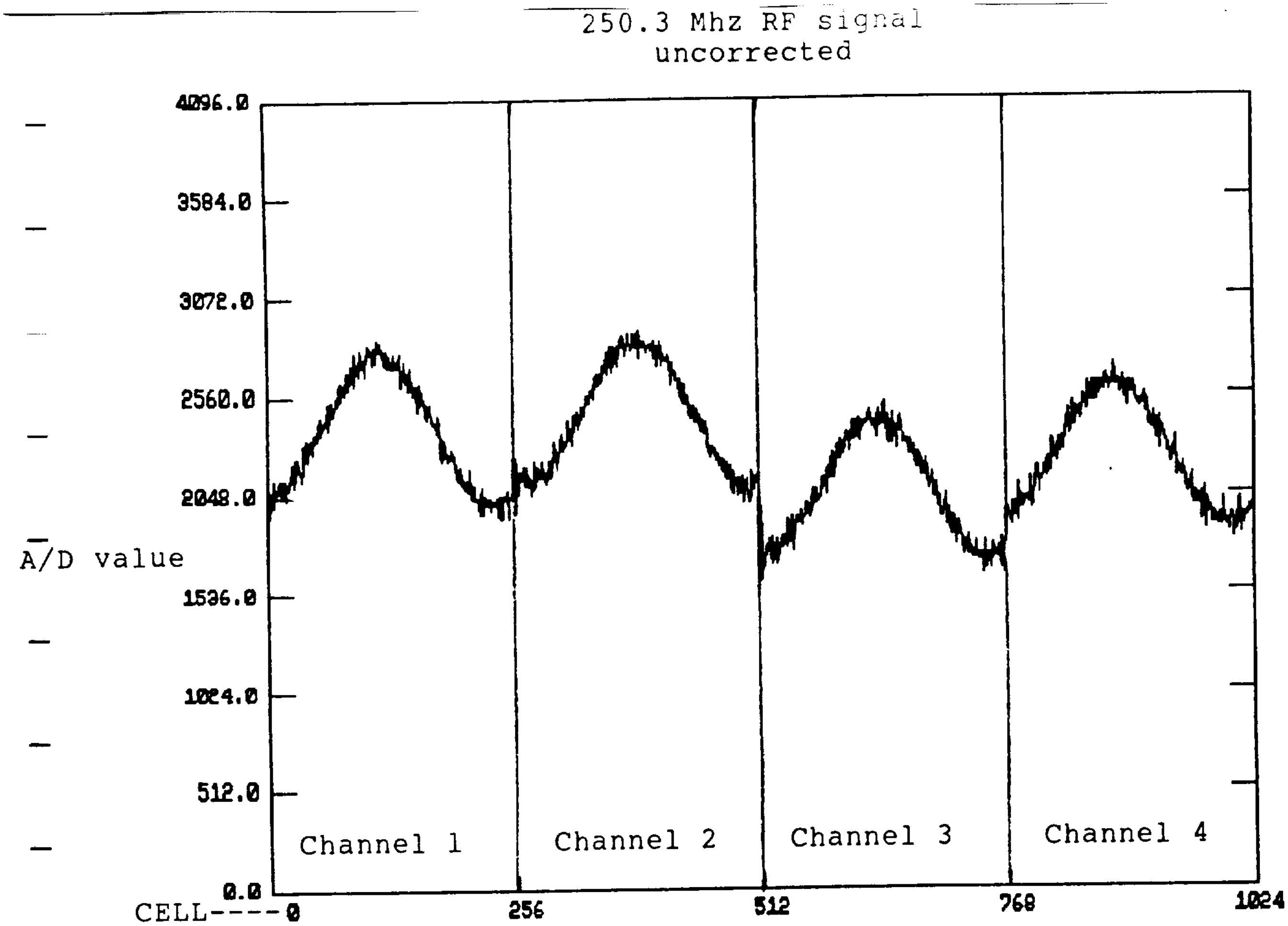
FIGURE A3-9, Output Signal with 20 MHz Sinewave Input

TRANSFER FUNCTION
Averaged over all channels and cells

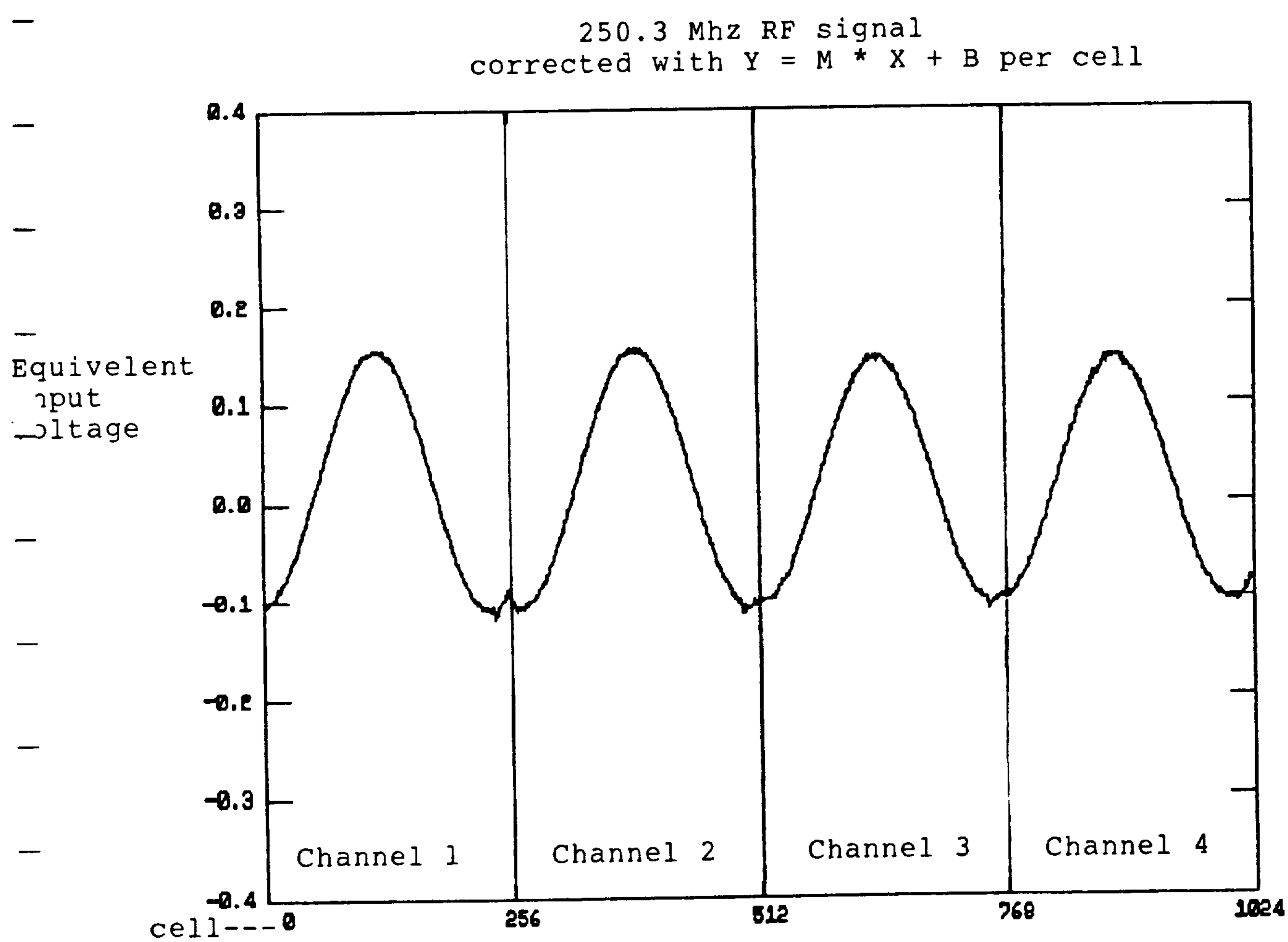


TRANSFER FUNCTION
Averaged over all cells, per channel



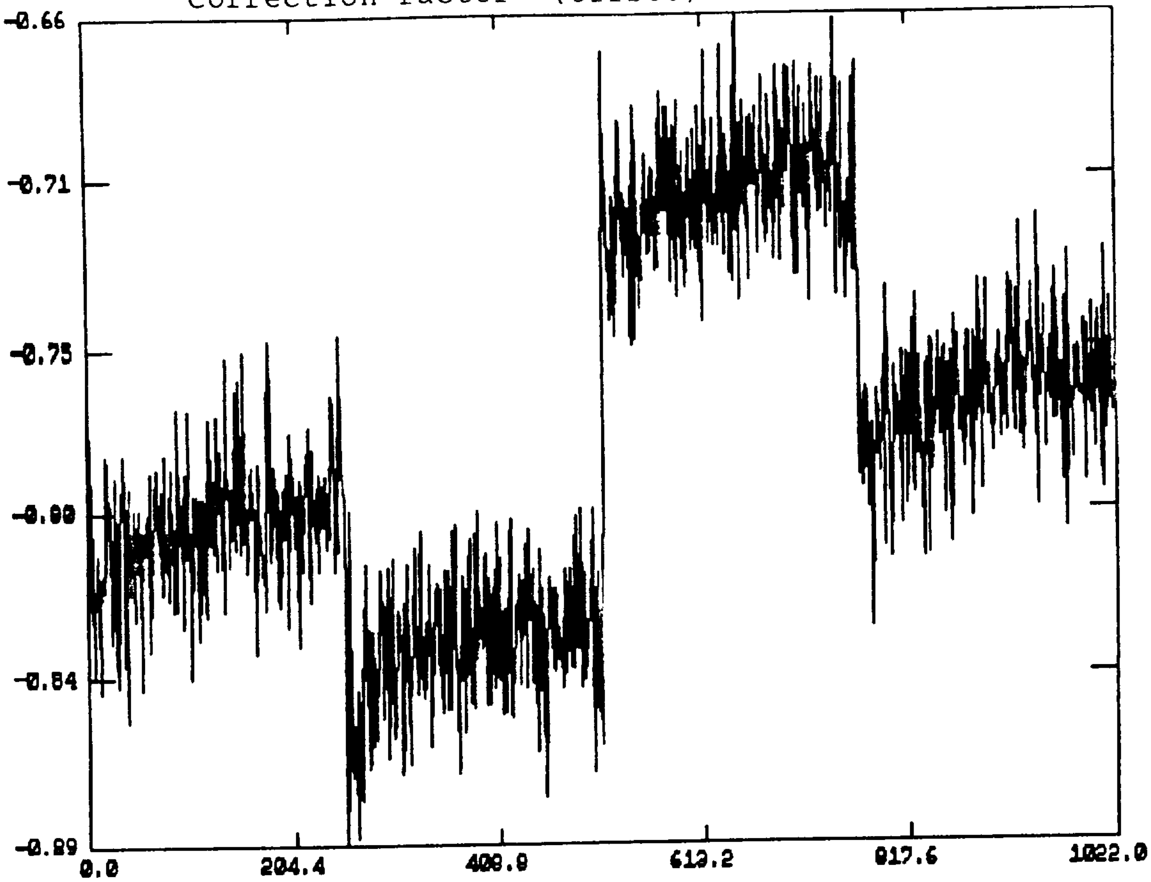


250.3 Mhz RF signal
uncorrected
4096.0
3584.0
3072.0
2560.0
2048.0
A/D value
1536.0
1024.0
512.0
0.0
Channel 1
Channel 2
Channel 3
Channel 4
CELL----0
256
512
768
1024

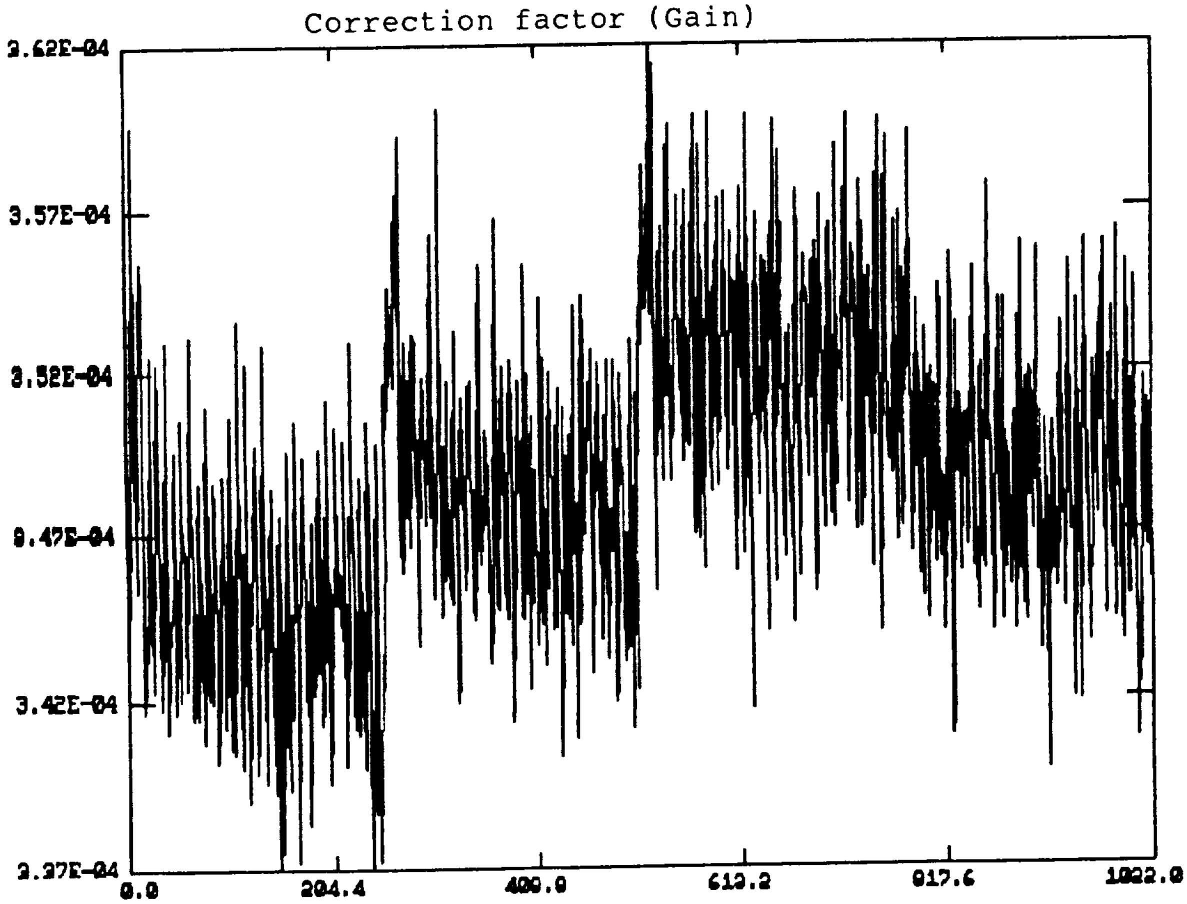


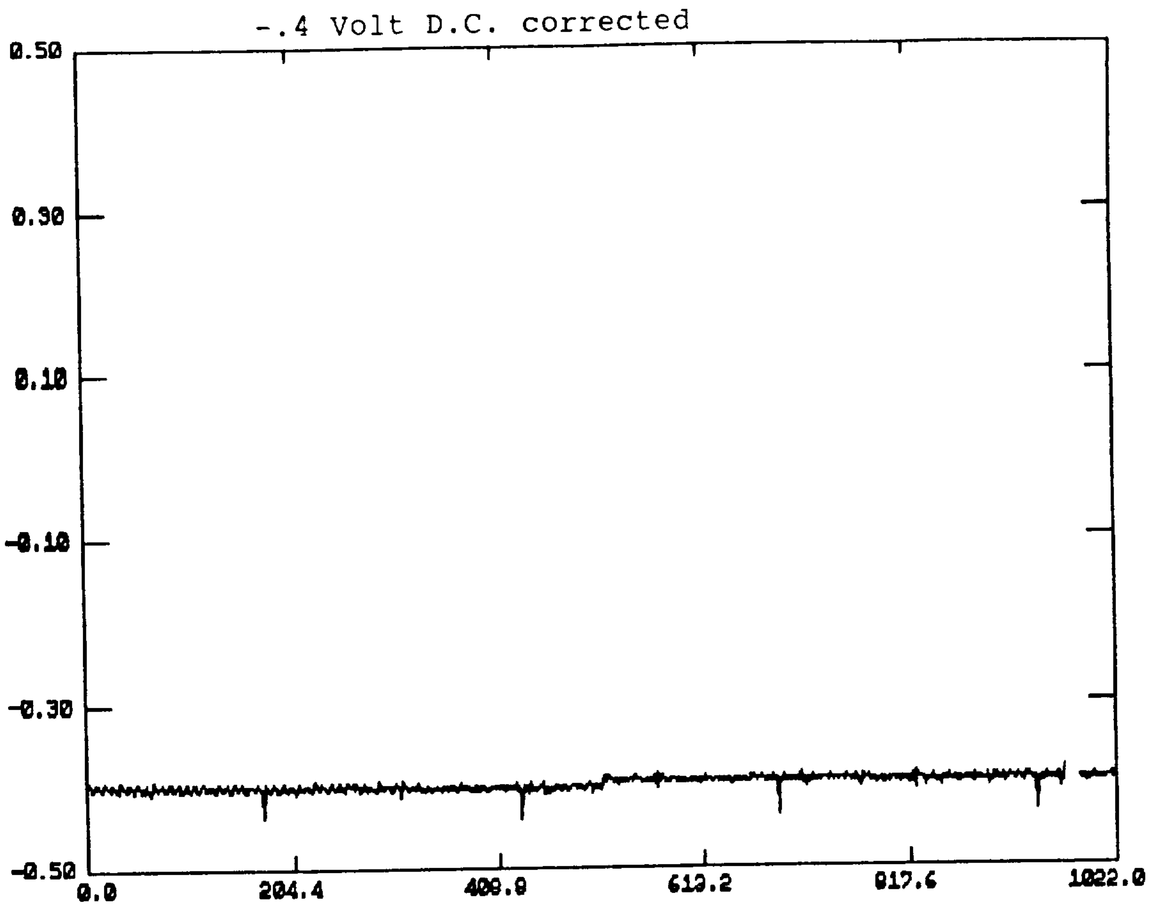
250.3 Mhz RF signal
corrected with Y = M * X + B per cell
0.4
0.3
0.2
Equivelent
input
voltage
0.1
0.0
-0.1
-0.2
-0.3
-0.4
Channel 1
Channel 2
Channel 3
Channel 4
cell---0
256
512
768
1024

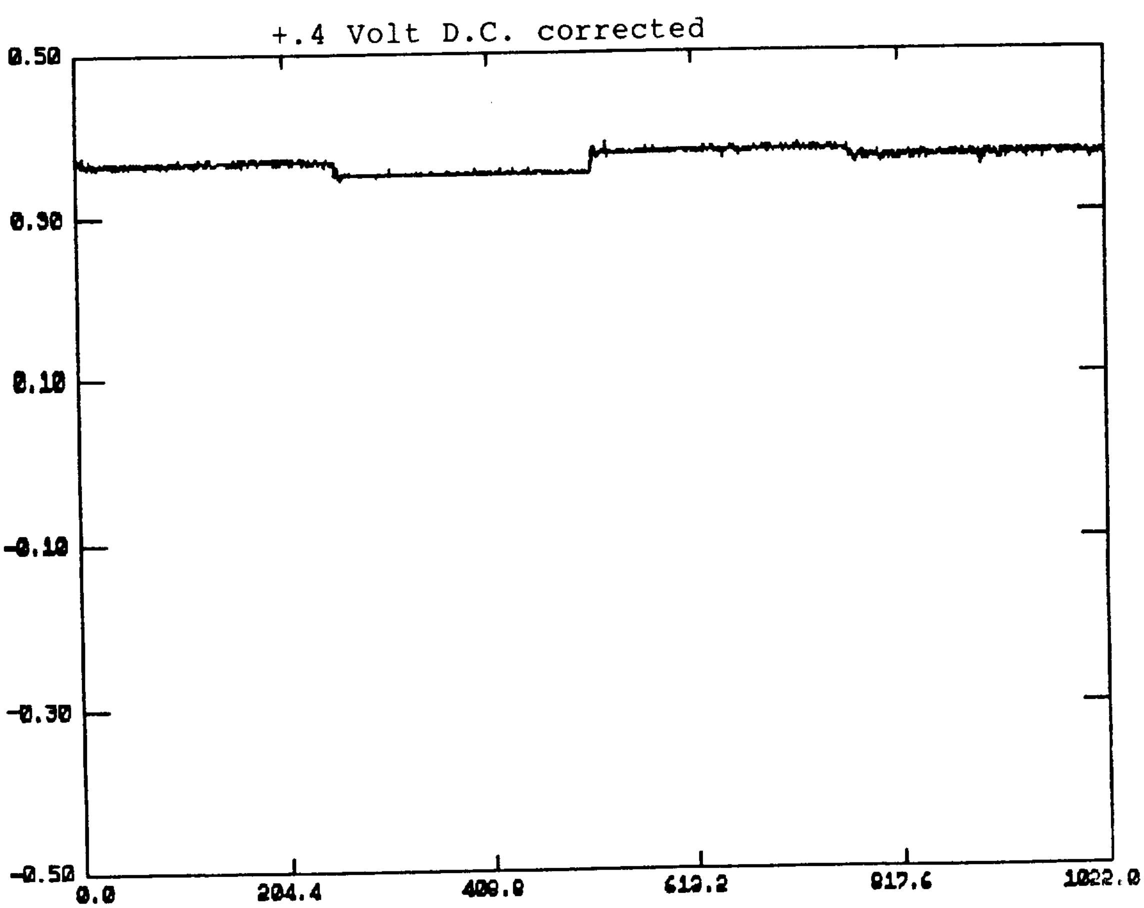
Correction factor (Offset)

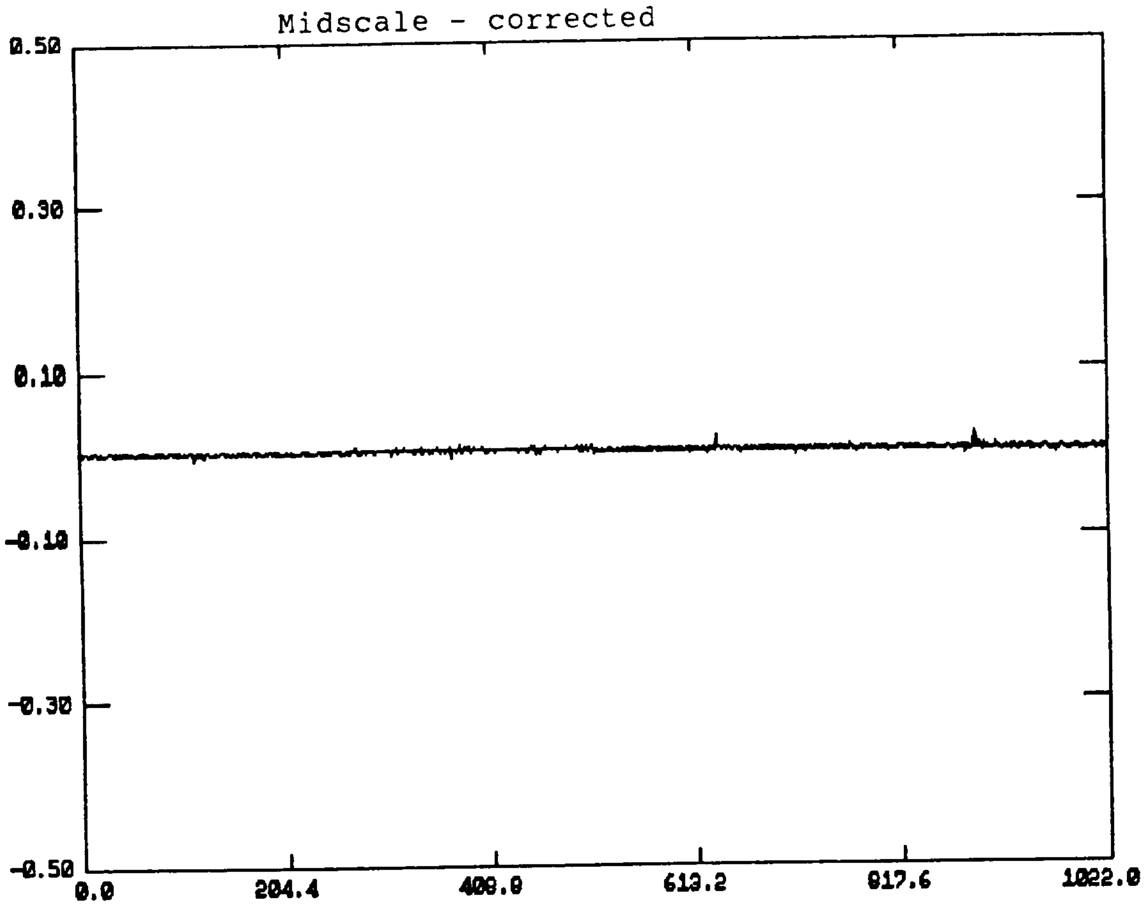


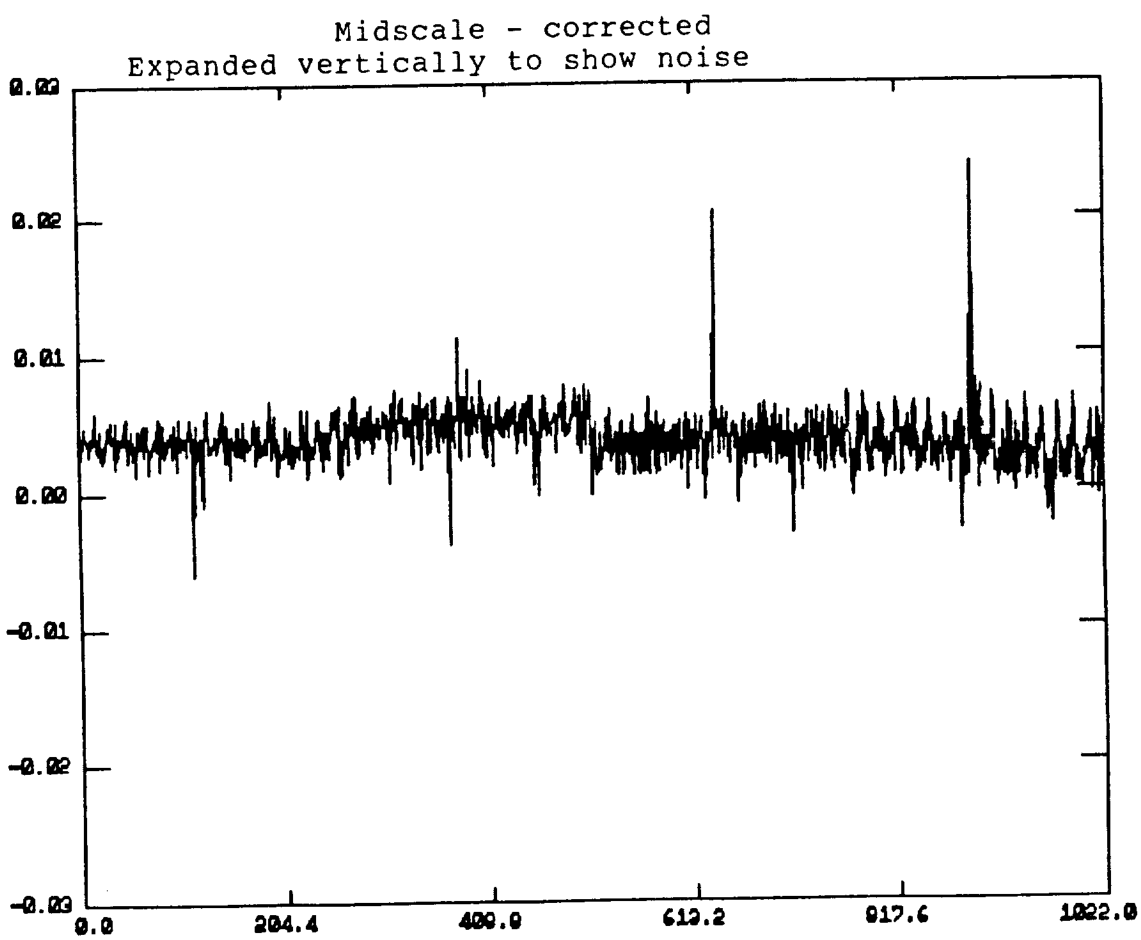
Correction factor (Gain)



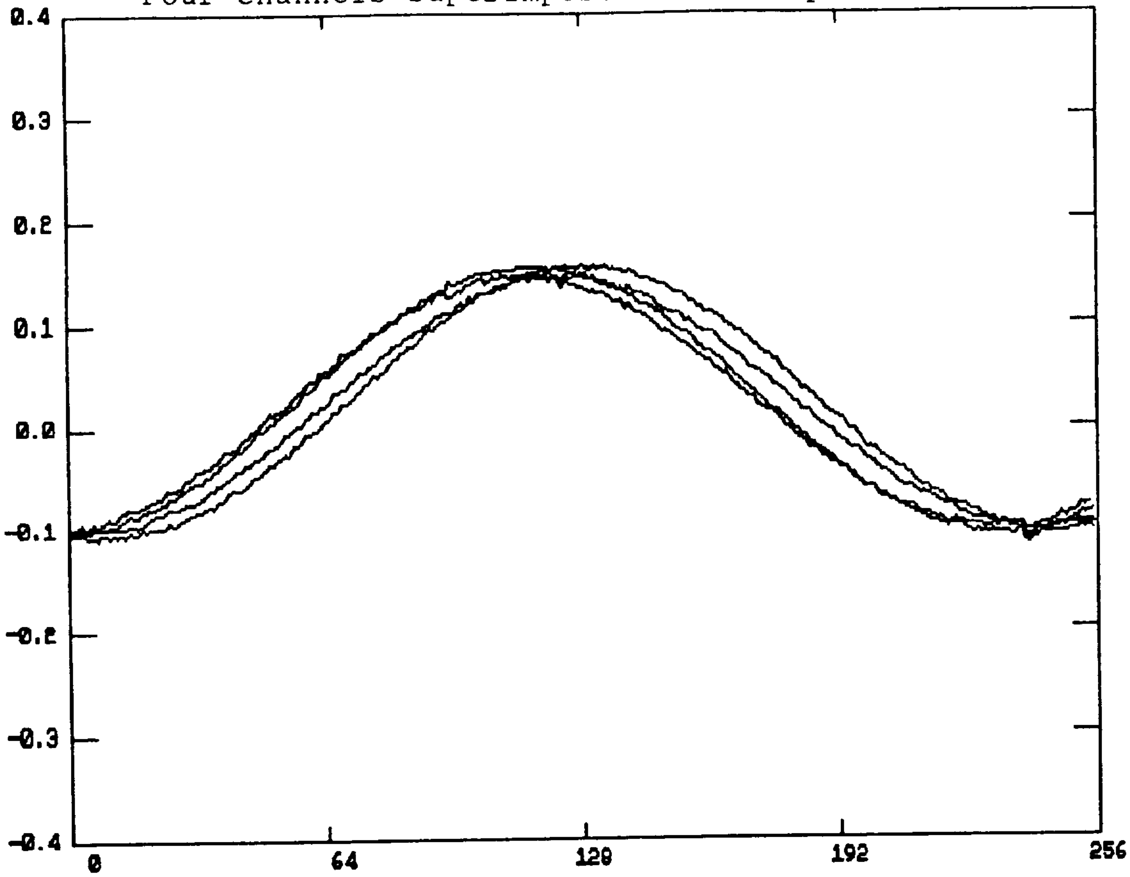

-.4 Volt D.C. corrected
0.50
0.30
0.10
-0.10
-0.30
-0.50
0.0
204.4
408.8
613.2
817.6
1022.0


+.4 Volt D.C. corrected
0.50
0.30
0.10
-0.10
-0.30
-0.50
0.0
204.4
408.8
613.2
817.6
1022.0

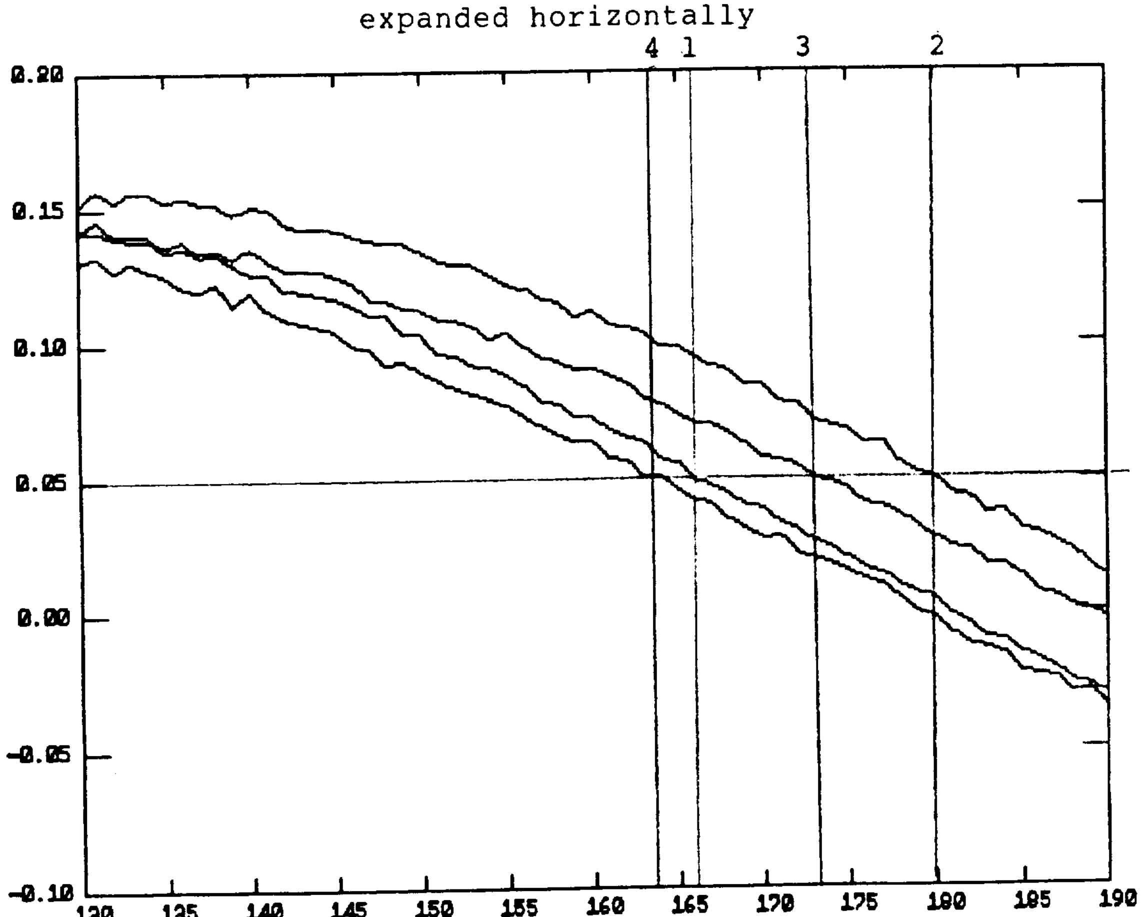

Midscale - corrected
0.50
0.30
0.10
-0.10
-0.30
-0.50
0.0
204.4
408.8
613.2
817.6
1022.0


Midscale - corrected
Expanded vertically to show noise
0.03
0.02
0.01
0.00
-0.01
-0.02
-0.03
0.0
204.4
408.8
613.2
817.6
1022.0

250.3 Mhz RF
Four channels superimposed to show phase difference

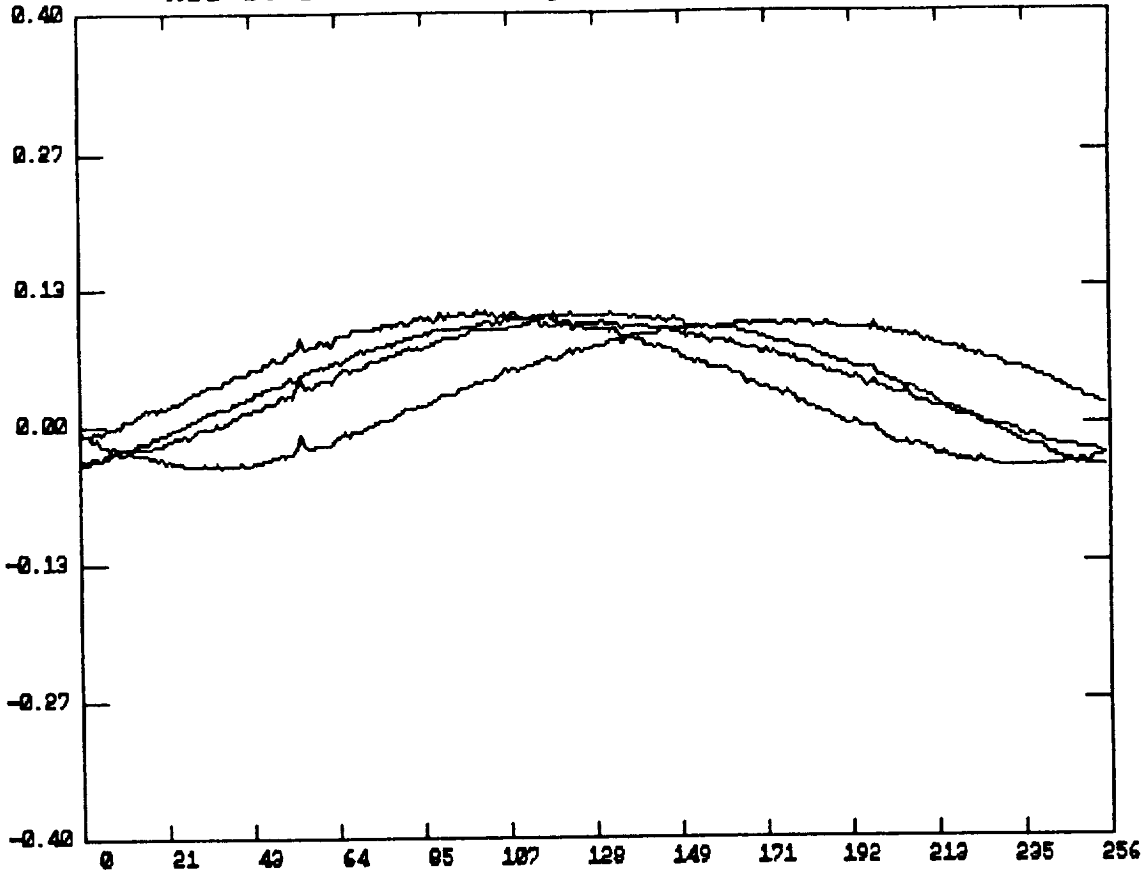


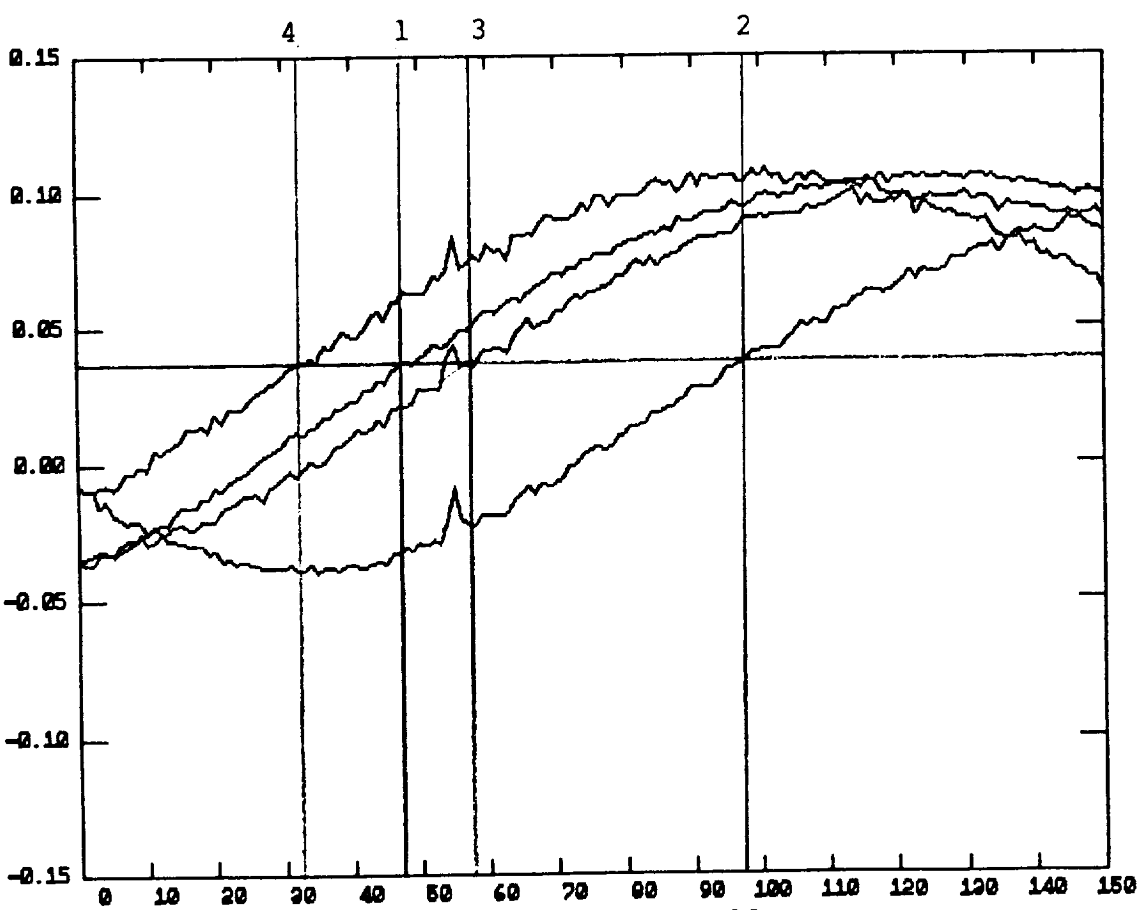
Same graph
expanded horizontally



163	166	173	180	cell number
0	3	10	17	Relative cell
0	47	156	265	ps delay

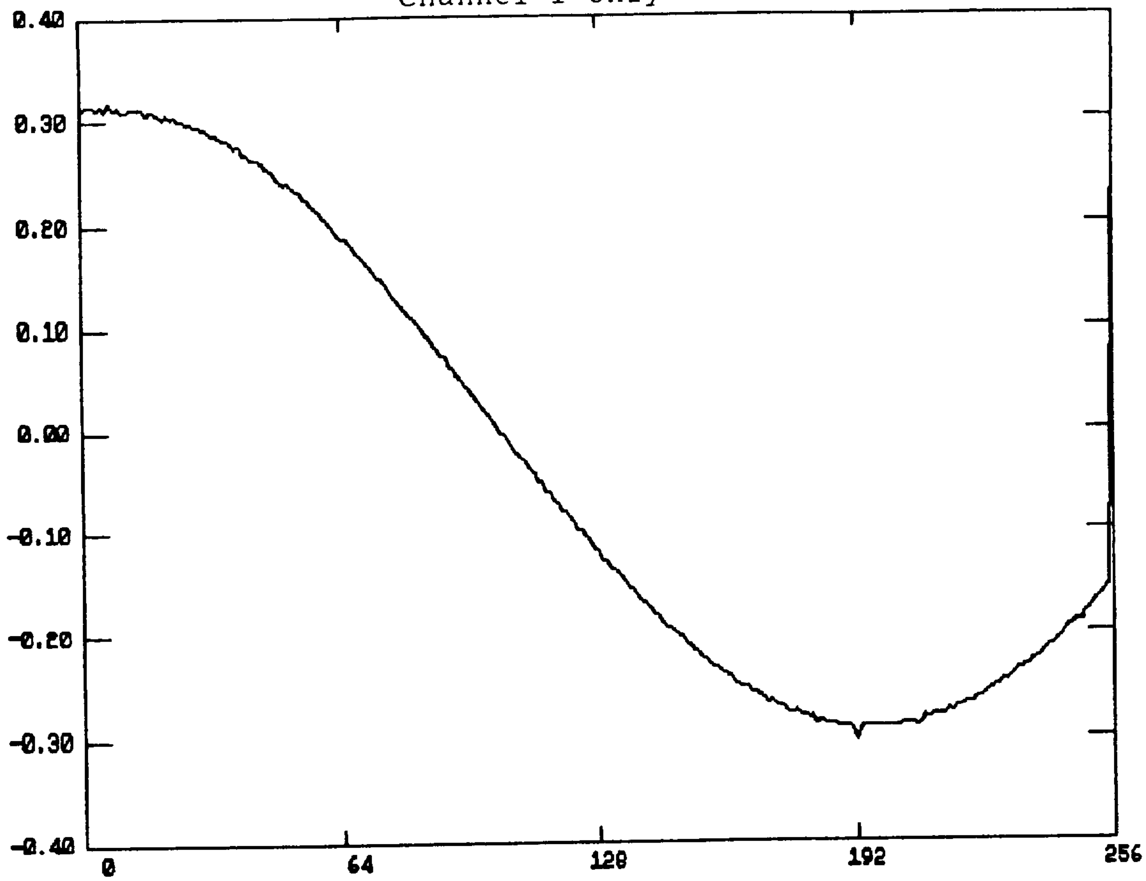
1.0003 Ghz RF signal
All four channels superimposed to show phase



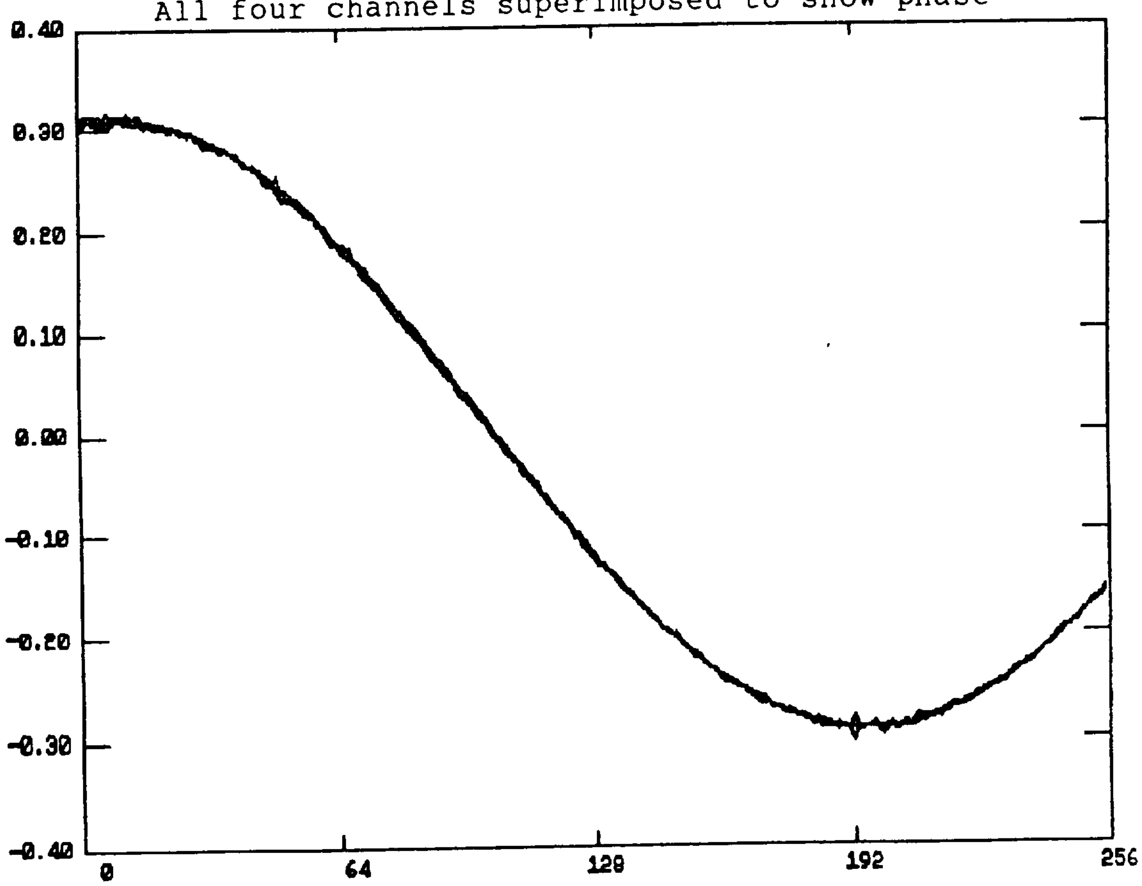


32	48	58	98	Cell
0	16	26	66	Relative
0	62	102	258	

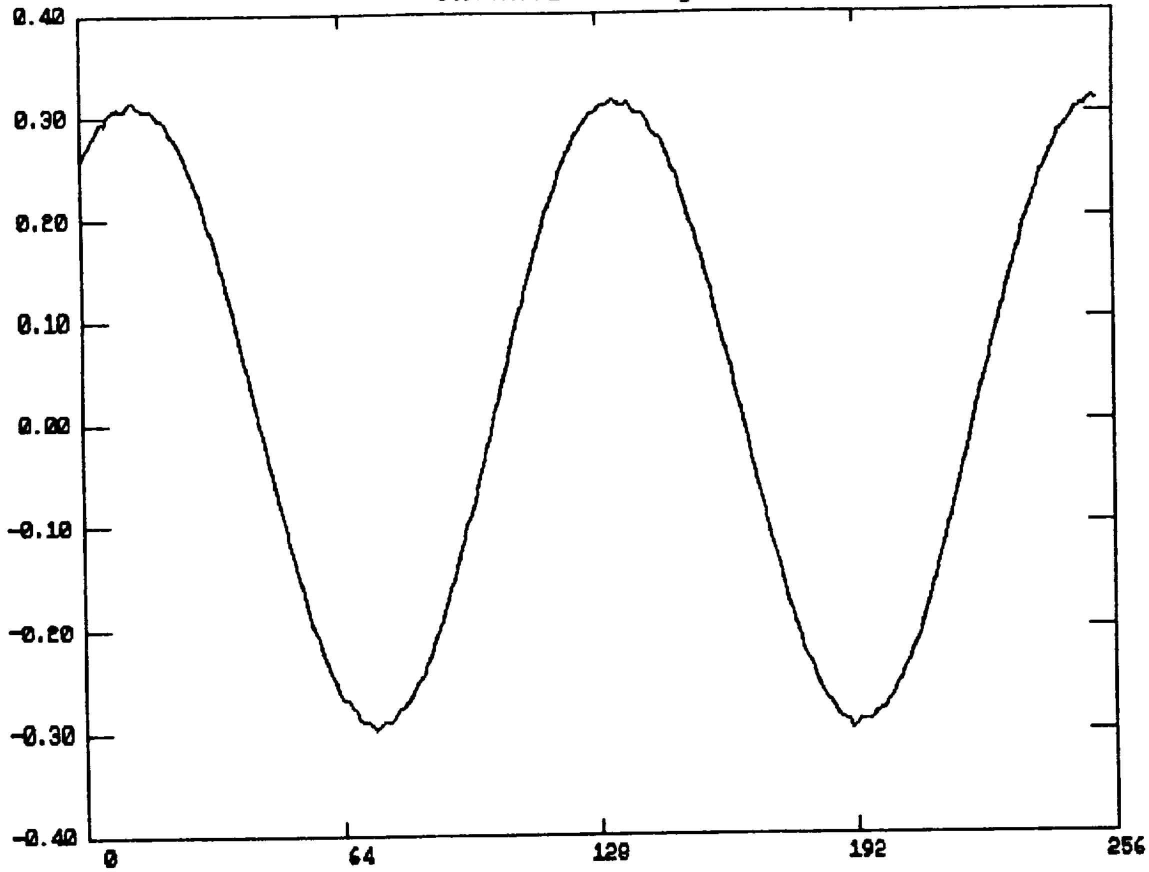
150 Khz Sine wave (+/- .3 v pk)
Channel 1 only



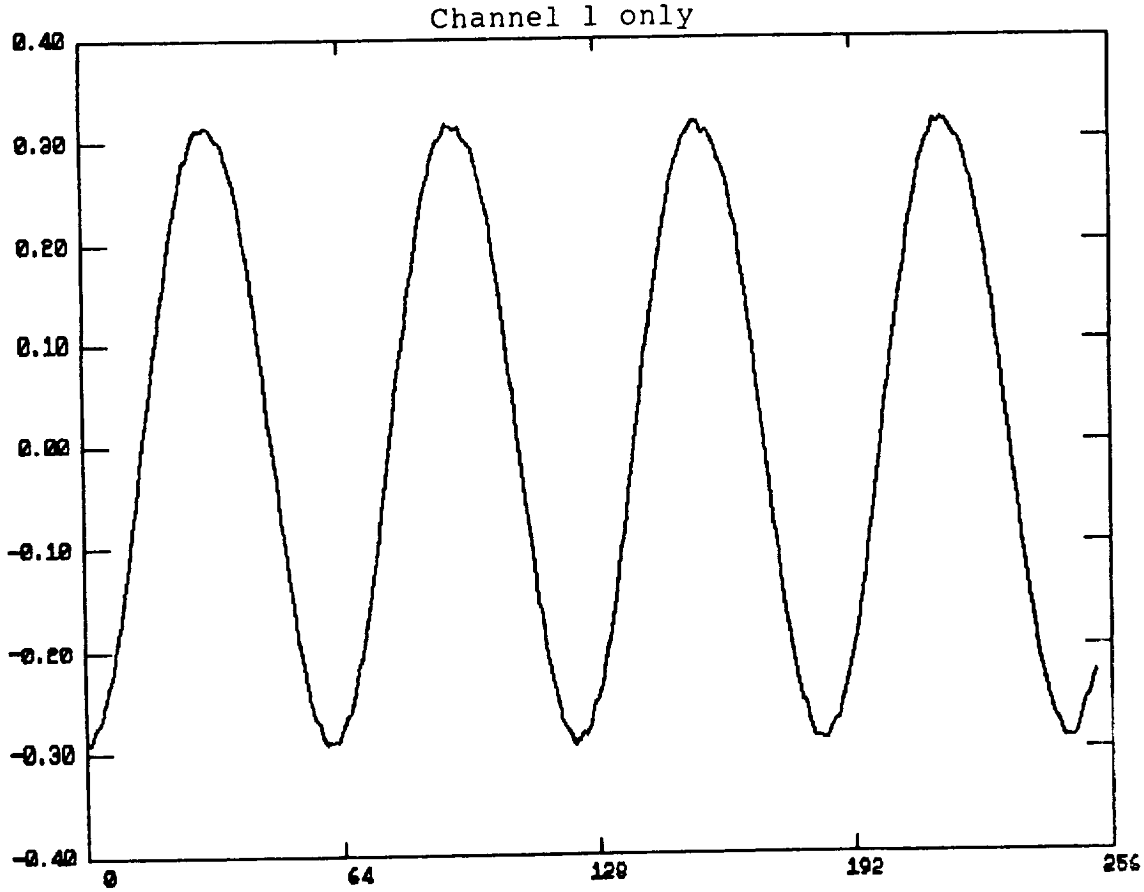
150 Khz Sine wave (+/- .3v pk)
All four channels superimposed to show phase

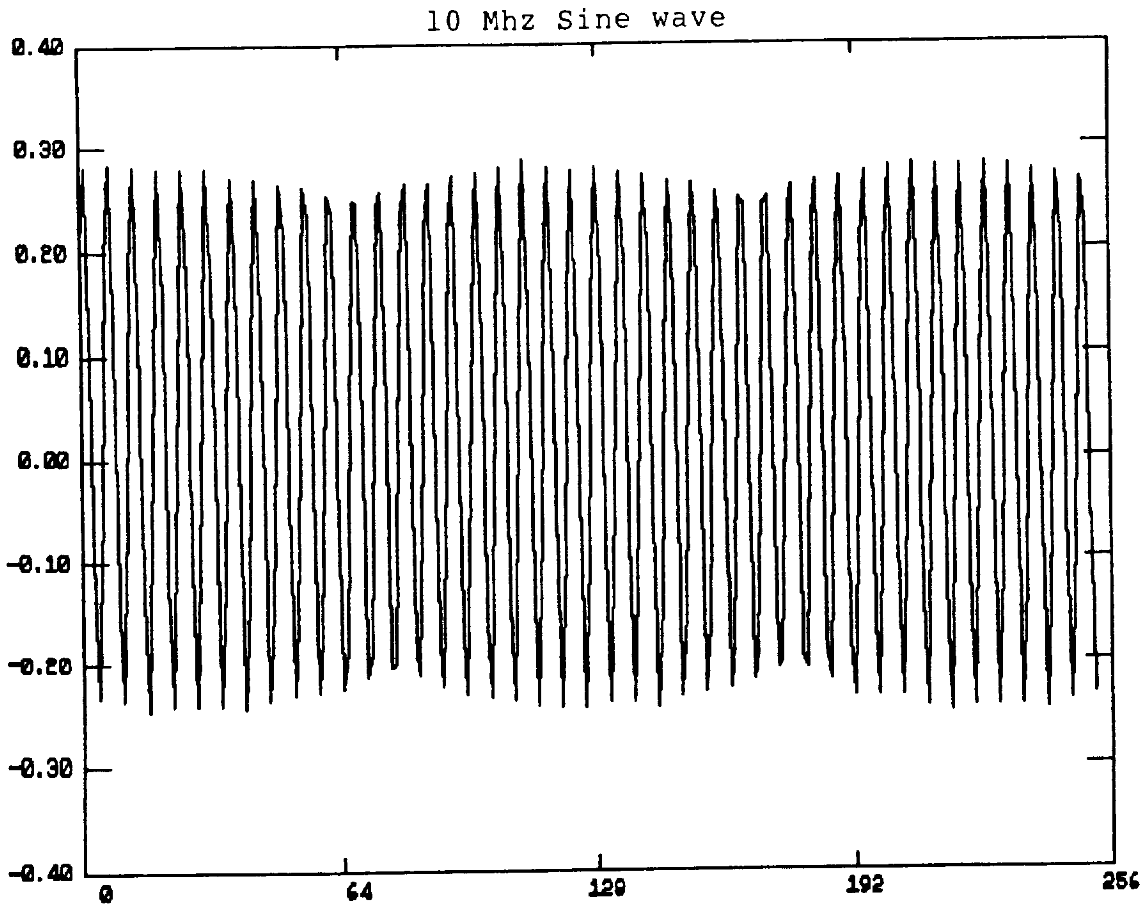


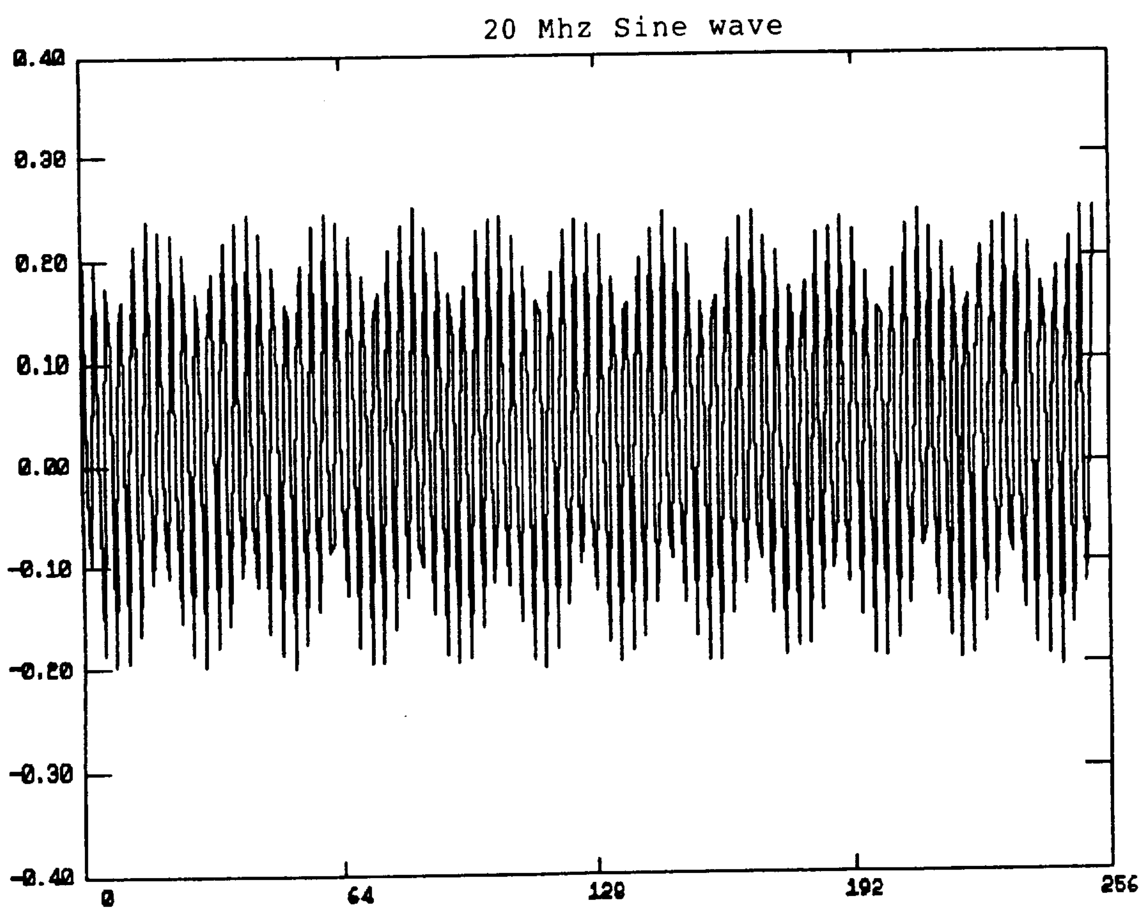
500 Khz Sine wave
Channel 1 only



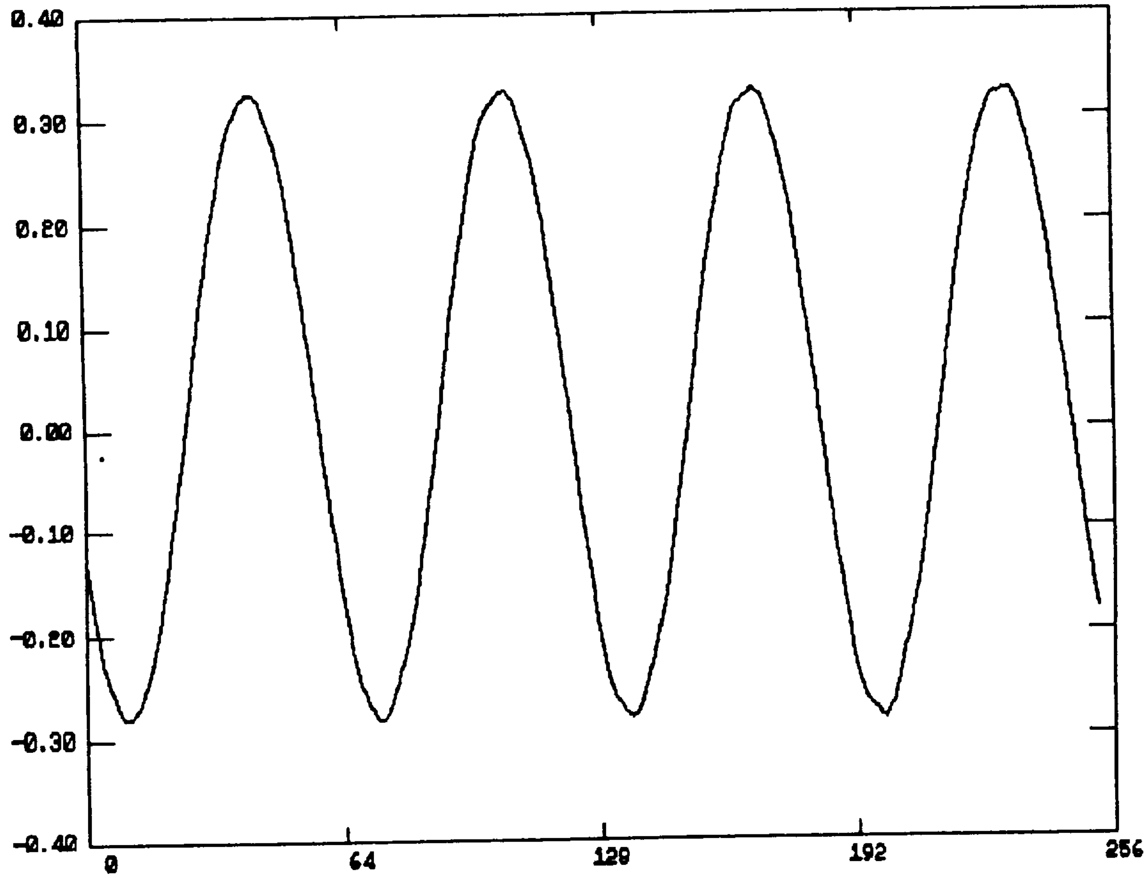
1 Mhz Sine wave
Channel 1 only



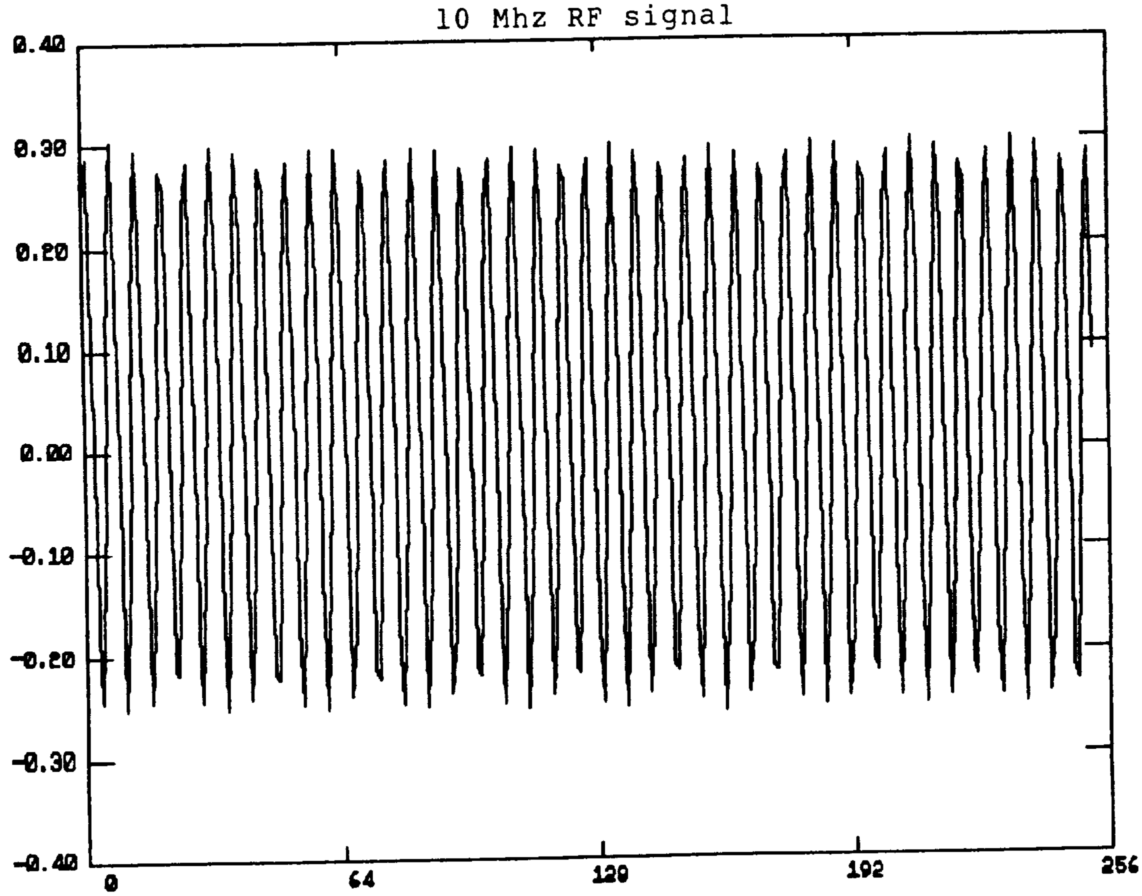

10 Mhz Sine wave
0.40
0.30
0.20
0.10
0.00
-0.10
-0.20
-0.30
-0.40
0
64
128
192
256


20 Mhz Sine wave
0.40
0.30
0.20
0.10
0.00
-0.10
-0.20
-0.30
-0.40
0
64
128
192
256

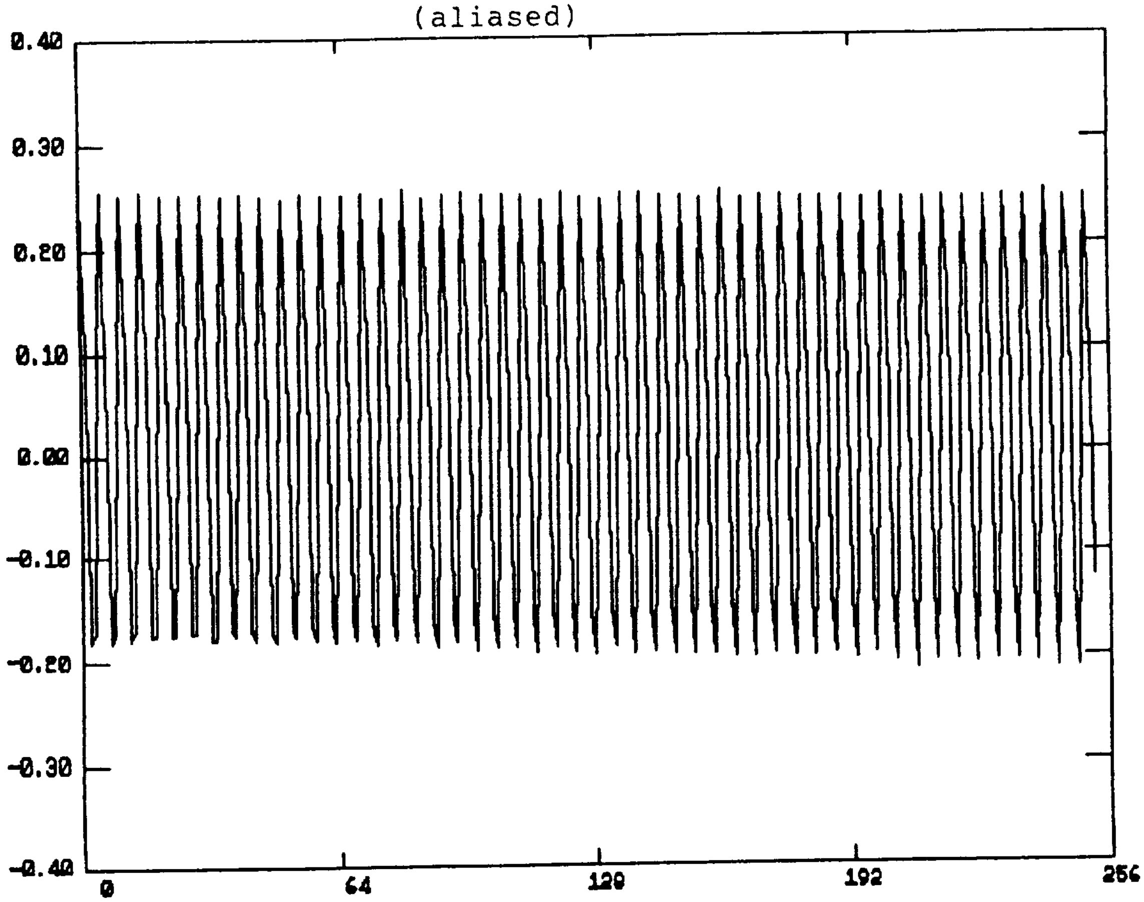
1 Mhz RF signal (+/- .3v peak)



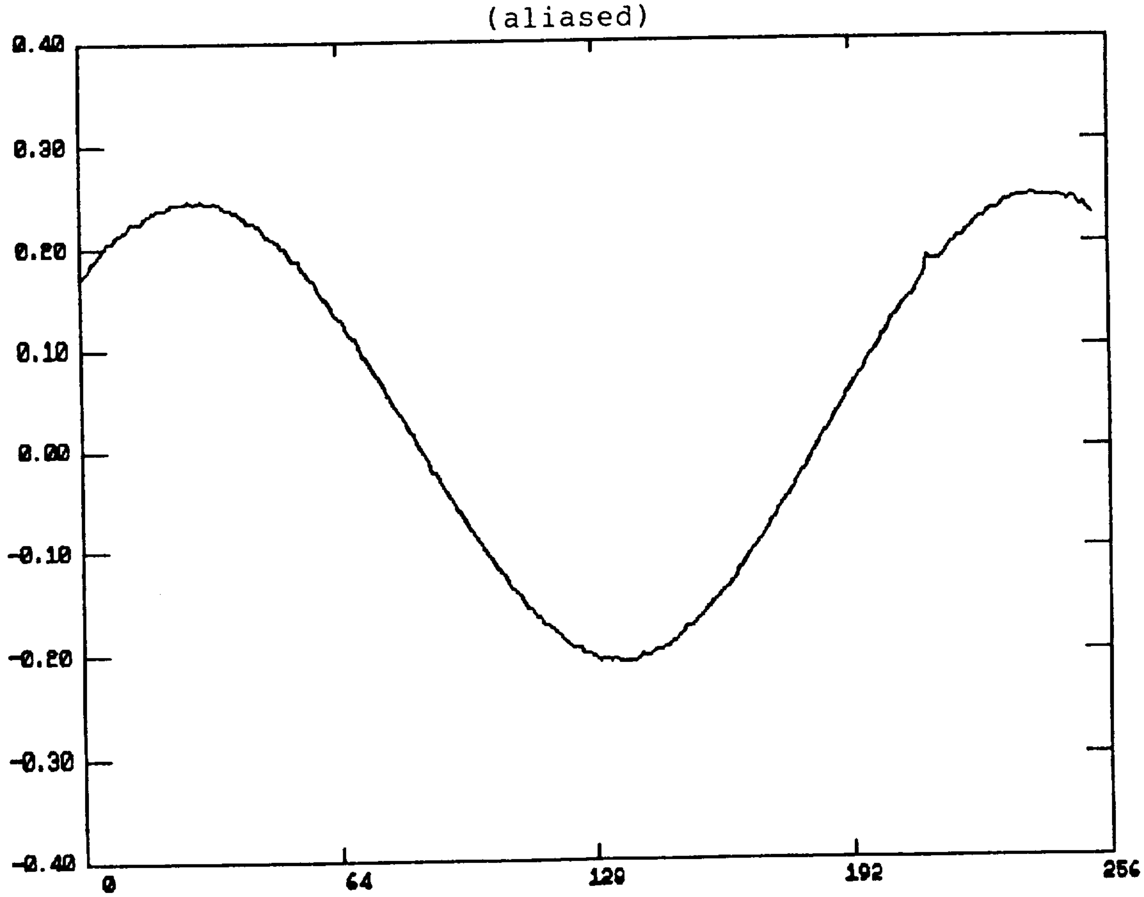
10 Mhz RF signal



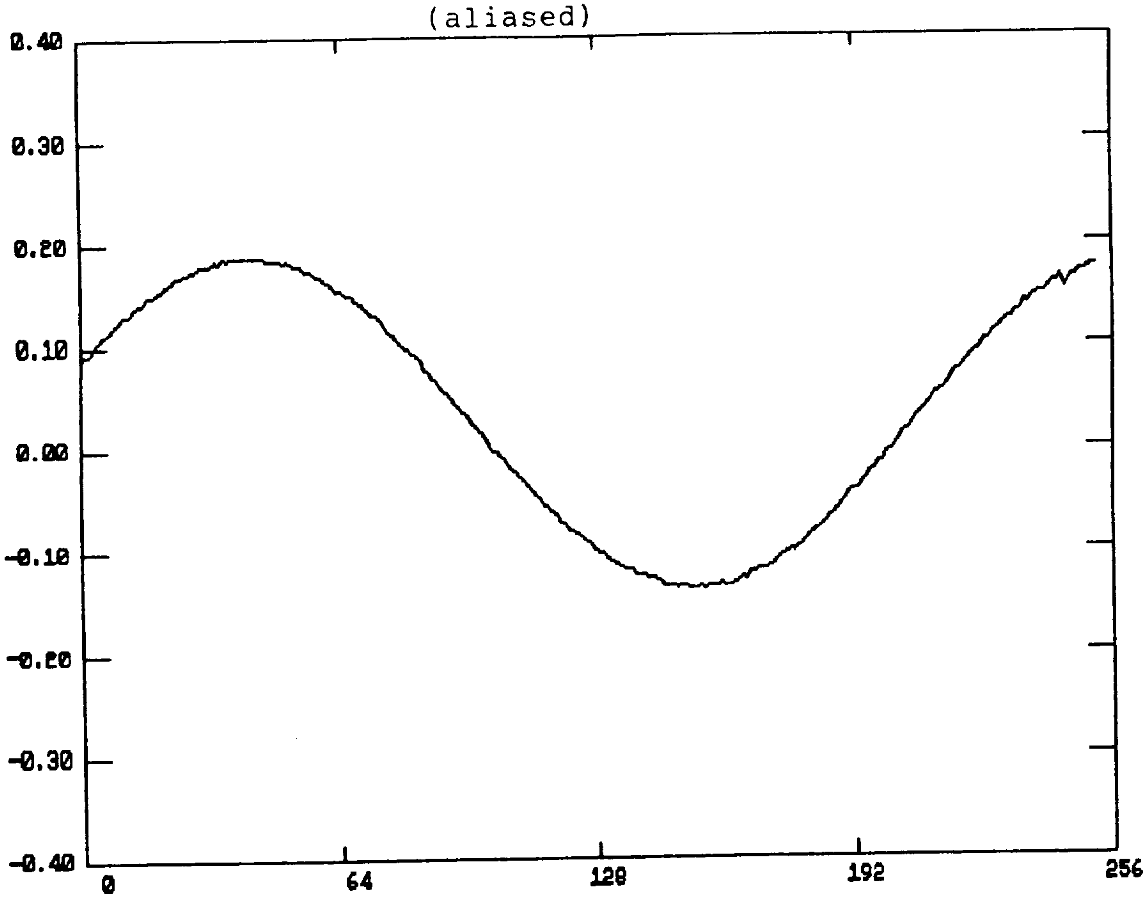
50 Mhz RF signal
(aliased)



62.8 Mhz. RF signal
(aliased)



125.3 Mhz RF signal
(aliased)



APPENDIX #4

Figure 3. Measured end-to-end Group Delay of Microstrip Delay Line loaded with 16 each 3.3 pf capacitors.

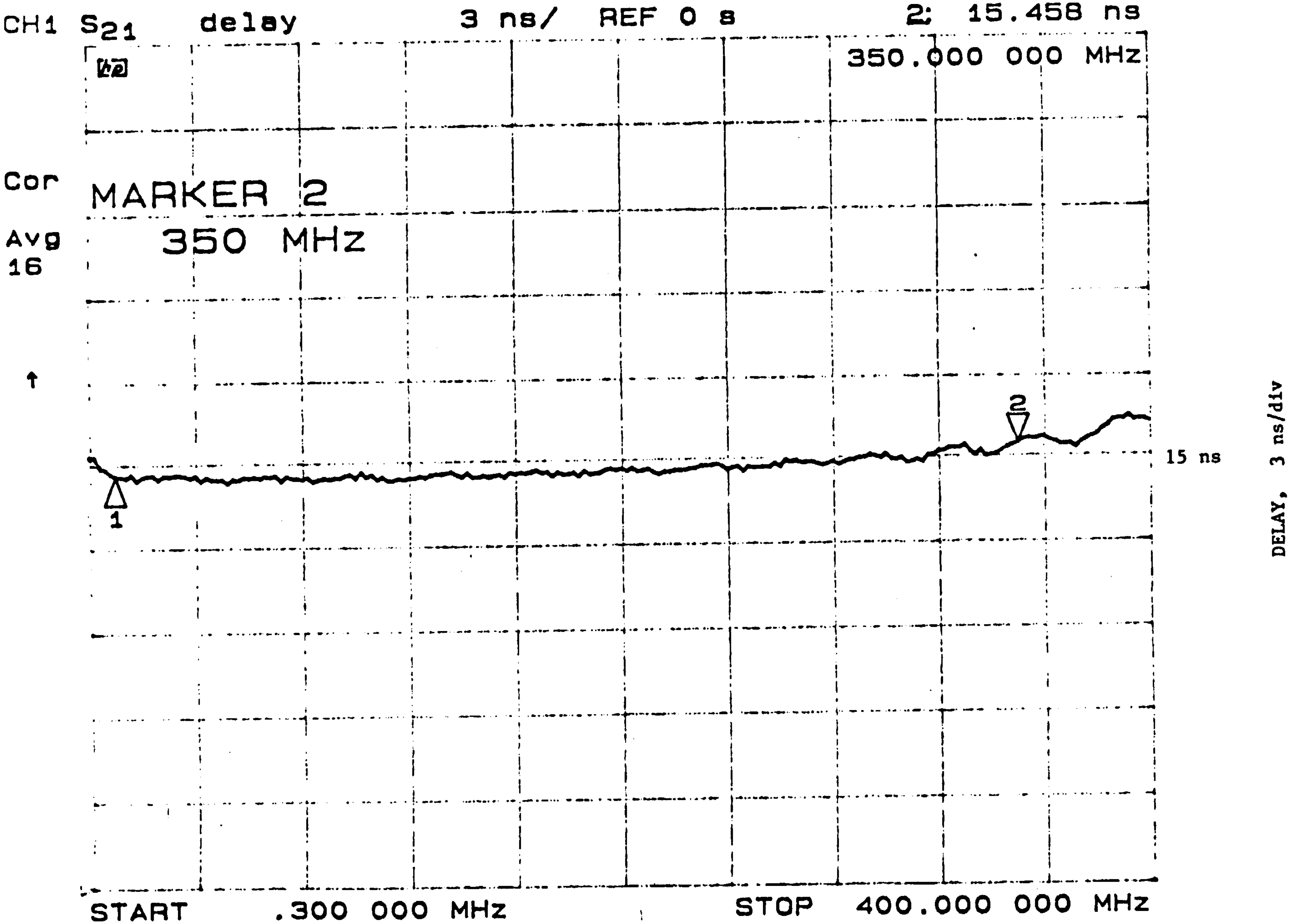


Figure 4. Measured end-to-end Insertion Loss of Microstrip Delay Line loaded with 16 each 3.3 pf capacitors.

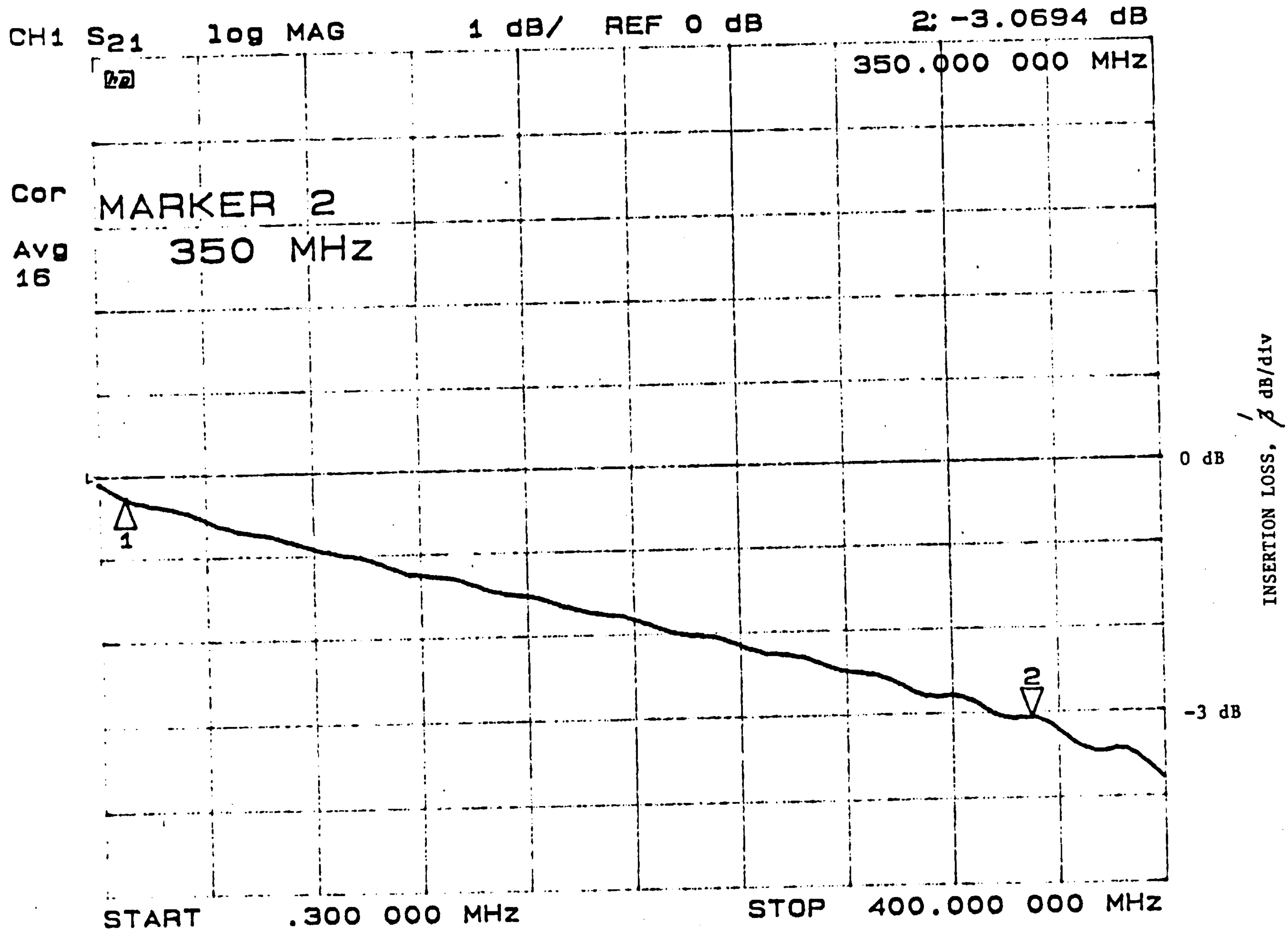
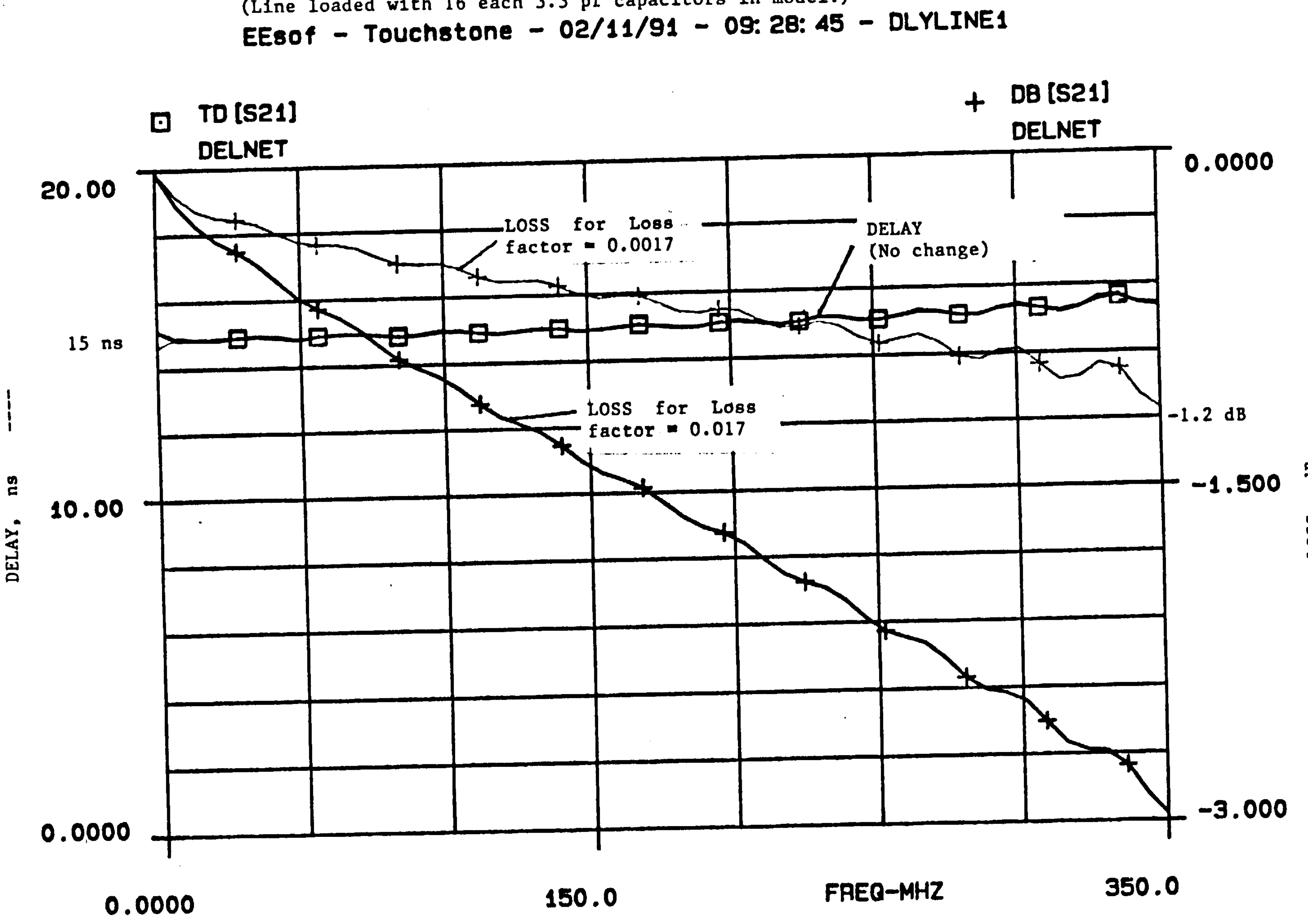
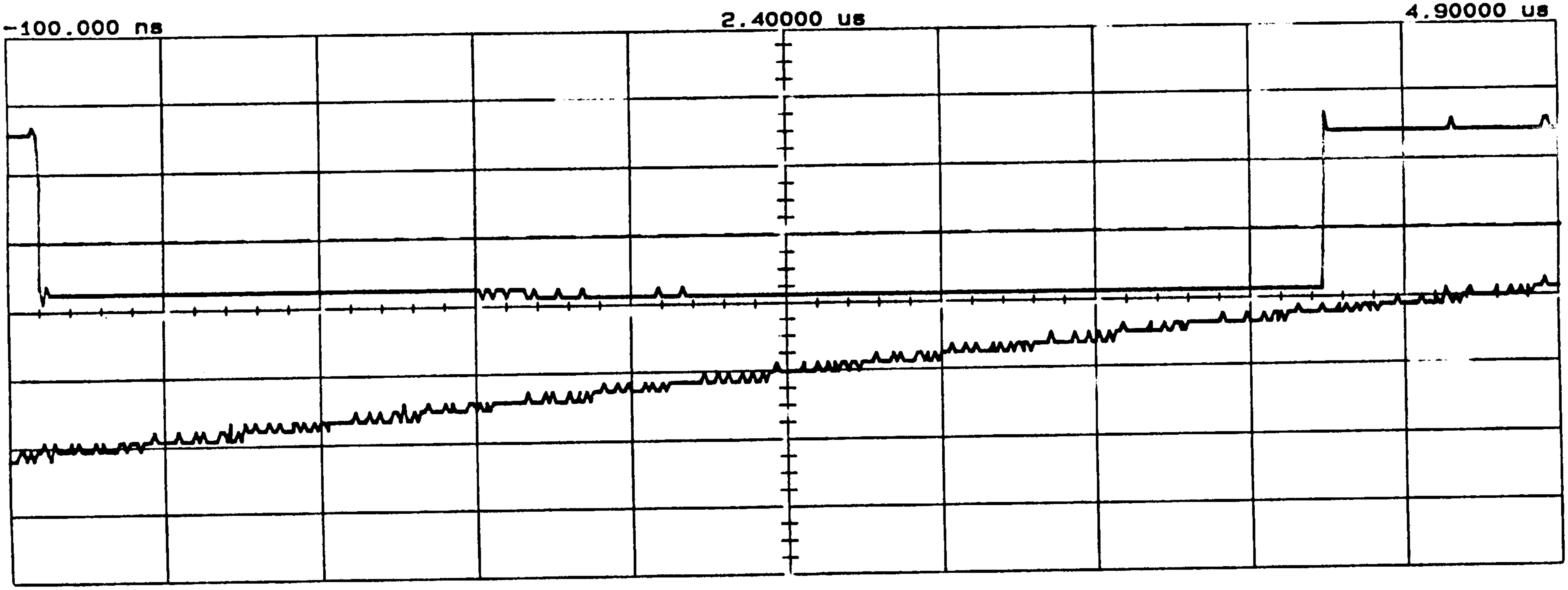


Figure 5a, Predicted end-to-end Delay and Insertion Loss of Microstrip Delay Line.
(Line loaded with 16 each 3.3 pf capacitors in model.)



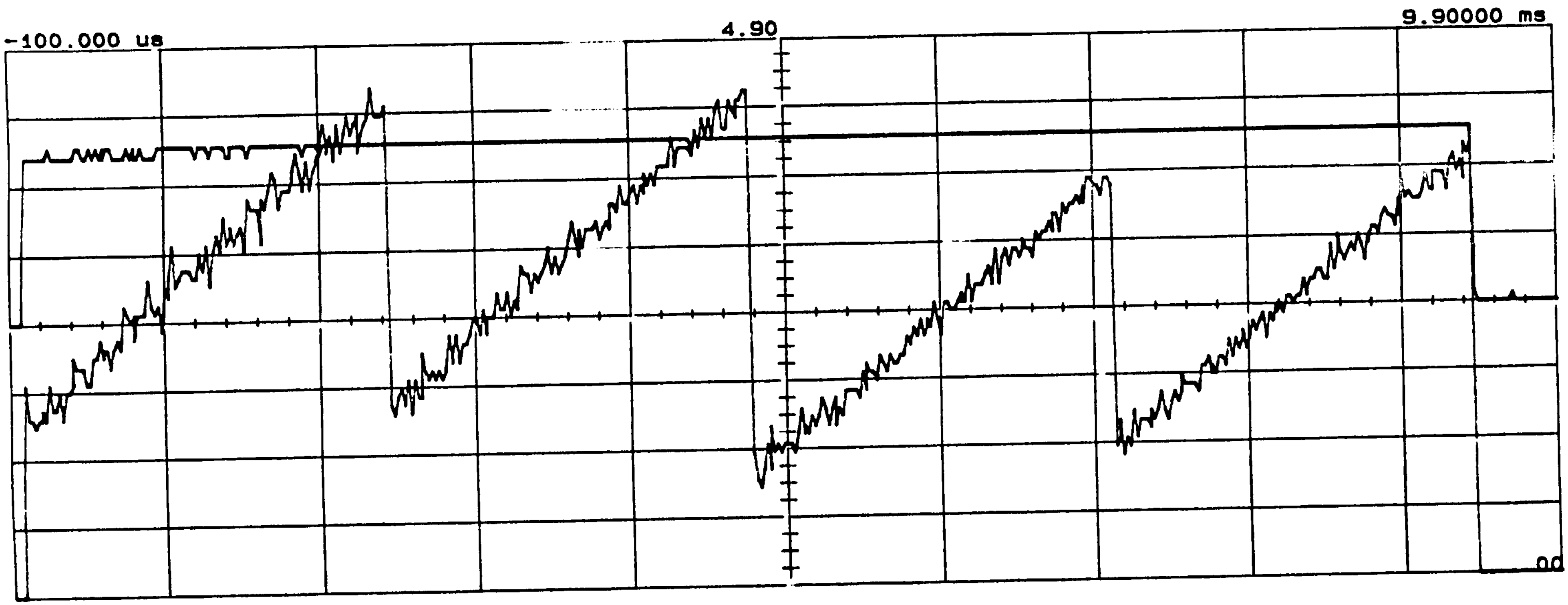


	Timebase	Delay/Pos	Reference	Mode
Main	500 ns/div	-100.000 ns	Left	Realtime (NORMAL)

	Sensitivity	Offset	Probe	Coupling
Channel 1	2.00 V/div	0.00000 V	10.00 : 1	dc (1M ohm)
Channel 2	1.00 V/div	-1.00000 V	10.00 : 1	dc (1M ohm)

Trigger mode : Edge
On Negative Edge Of Chan1
Trigger Level
Chan1 = 2.00000 V (noise reject OFF)
Holdoff = 40.000 ns

FIGURE 5

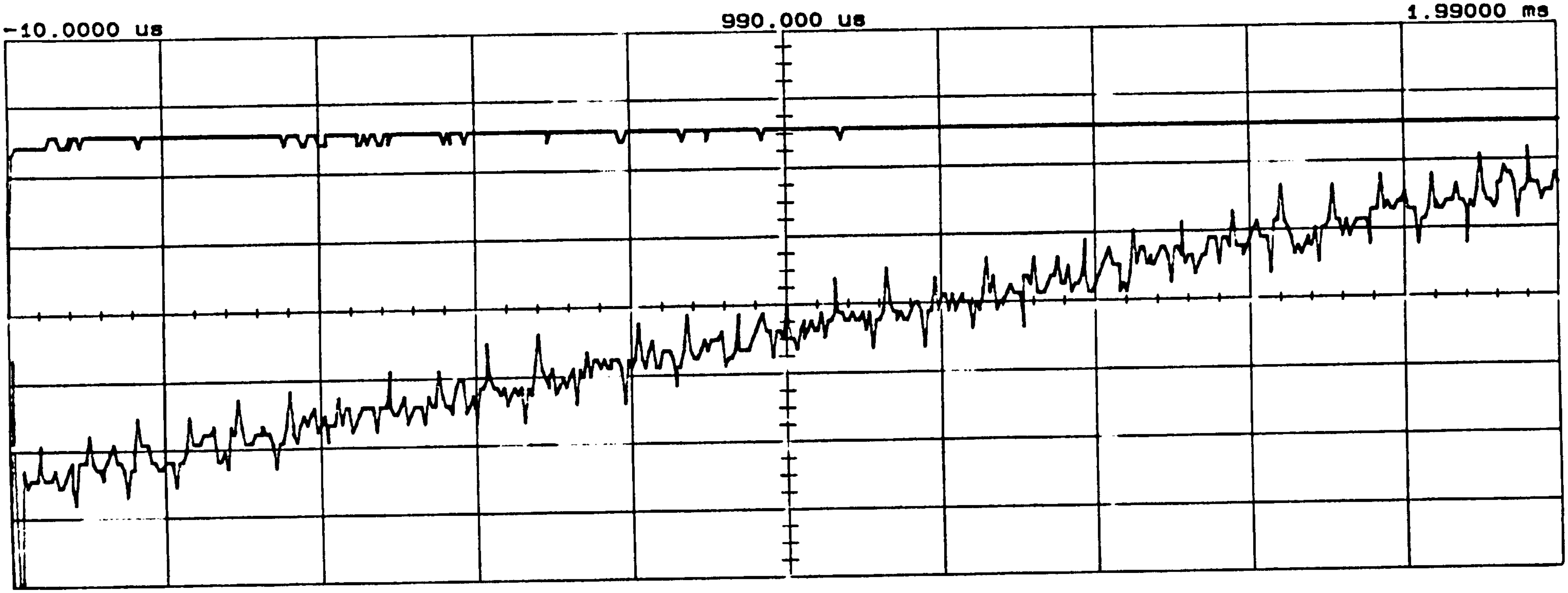


	Timebase	Delay/Pos	Reference	Mode
Main	1.00 ms/div	-100.000 us	Left	Realtime (NORMAL)

	Sensitivity	Offset	Probe	Coupling
Channel 1	2.00 V/div	0.00000 V	10.00 : 1	dc (1M ohm)
Channel 2	1.00 V/div	3.00000 V	10.00 : 1	dc (1M ohm)

Trigger mode : Edge
On Positive Edge Of Chan1
Trigger Level
Chan1 = 2.00000 V (noise reject OFF)
Holdoff = 40.000 ns

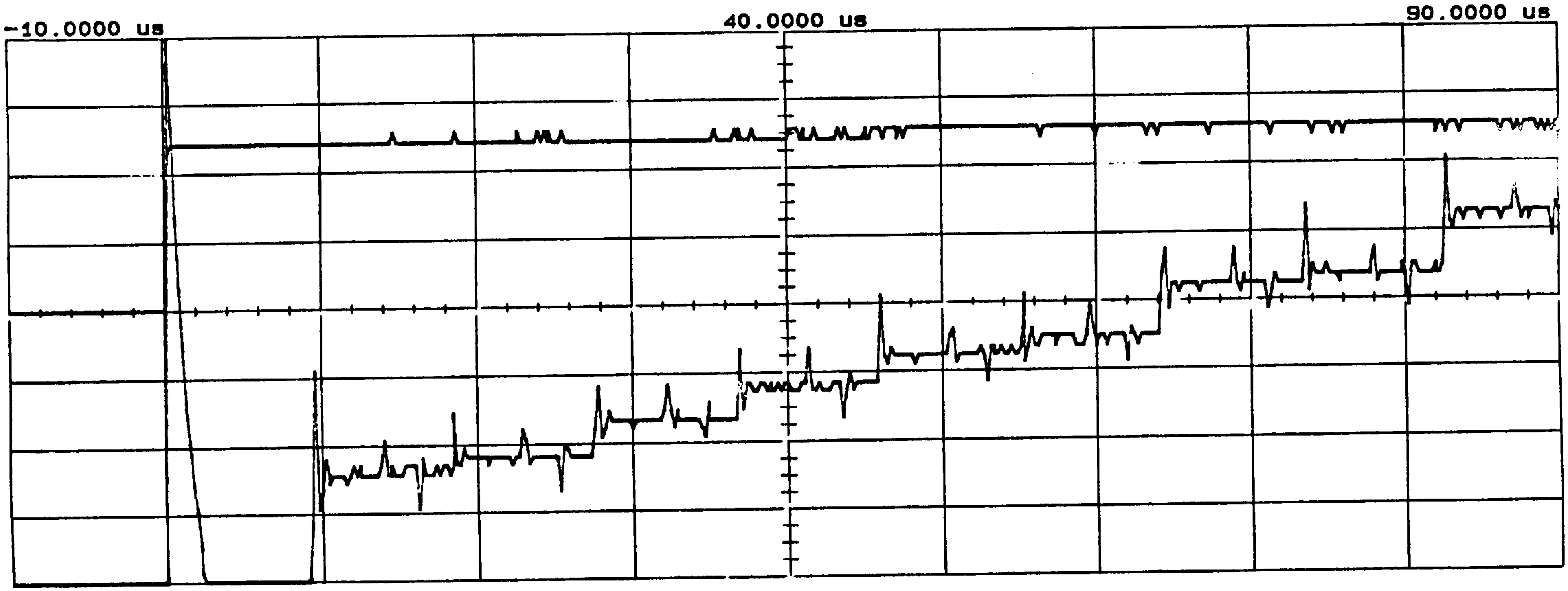
FIGURE 6



	Timebase	Delay/Pos	Reference	Mode
Main	200 us/div	-10.0000 us	Left	Realtime (NORMAL)
	Sensitivity	**Offset**	**Probe**	**Coupling**
Channel 1	2.00 V/div	0.00000 V	10.00 : 1	dc (1M ohm)
Channel 2	1.00 V/div	3.00000 V	10.00 : 1	dc (1M ohm)

Trigger mode : Edge
On Positive Edge Of Chan1
Trigger Level
Chan1 = 2.00000 V (noise reject OFF)
Holdoff = 40.000 ns

FIGURE 7



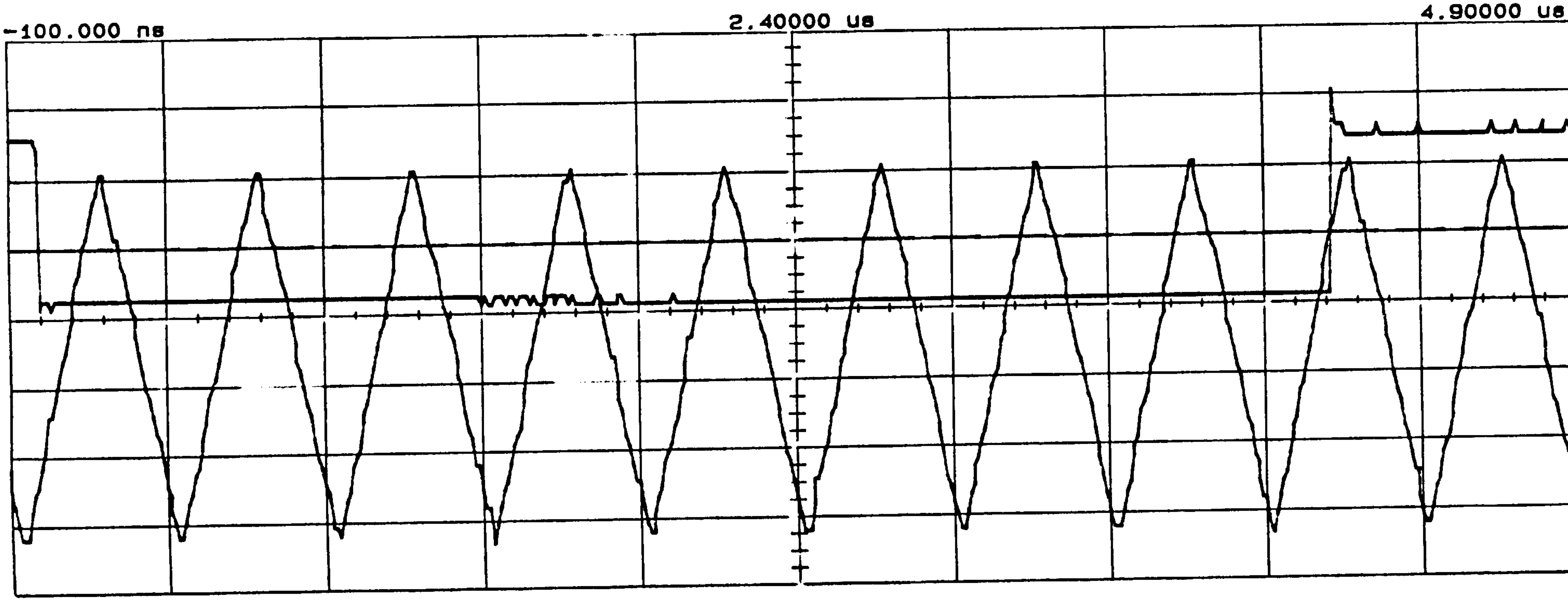
C-2

	Timebase	Delay/Pos	Reference	Mode
Main	10.0 us/div	-10.0000 us	Left	Realtime (NORMAL)

	Sensitivity	Offset	Probe	Coupling
Channel 1	2.00 V/div	0.00000 V	10.00 : 1	dc (1M ohm)
Channel 2	1.00 V/div	3.00000 V	10.00 : 1	dc (1M ohm)

Trigger mode : Edge
On Positive Edge Of Chan1
Trigger Level
Chan1 = 2.00000 V (noise reject OFF)
Holdoff = 40.000 ns

FIGURE 8



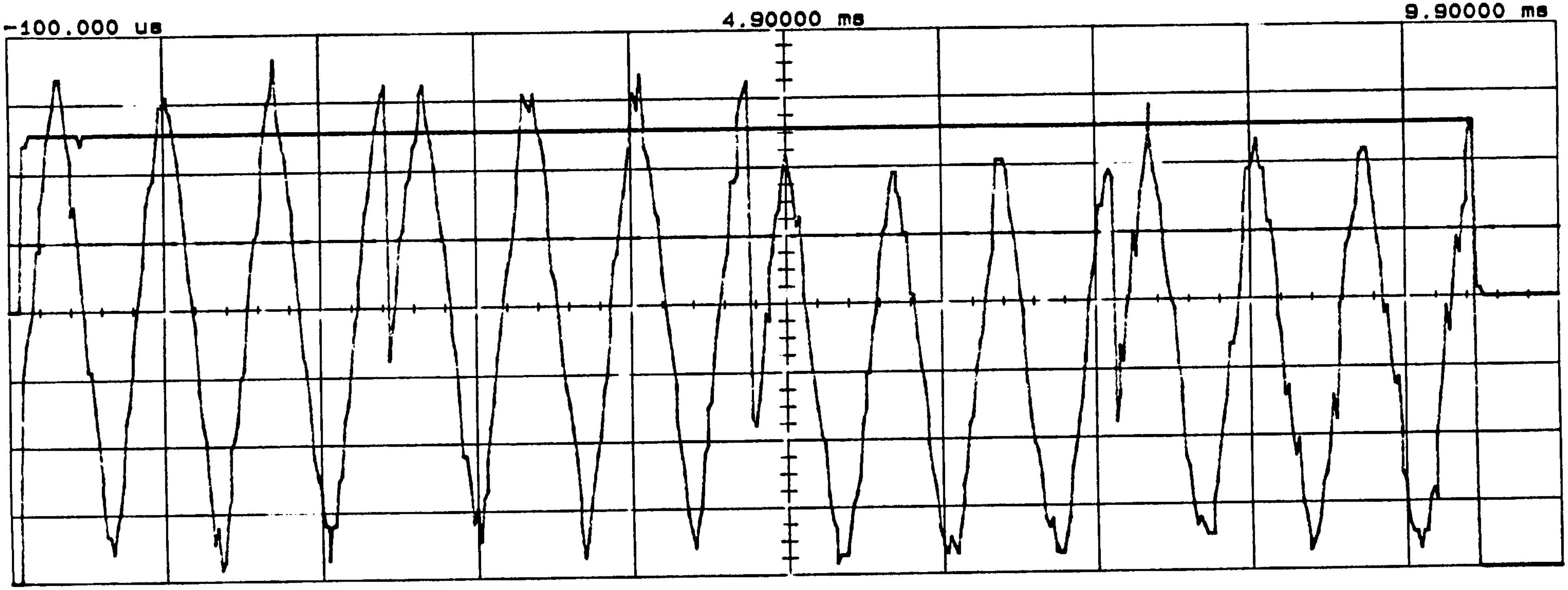
	Timebase	Delay/Pos	Reference	Mode
Main	500 ns/div	-100.000 ns	Left	Realtime (NORMAL)

	Sensitivity	Offset	Probe	Coupling
Channel 1	2.00 V/div	0.00000 V	10.00 : 1	dc (1M ohm)
Channel 2	1.00 V/div	-1.00000 V	10.00 : 1	dc (1M ohm)

Trigger mode : Edge
On Negative Edge Of Chan1
Trigger Level
Chan1 = 2.00000 V (noise reject OFF)
Holdoff = 40.000 ns

FIGURE 9

figure 9

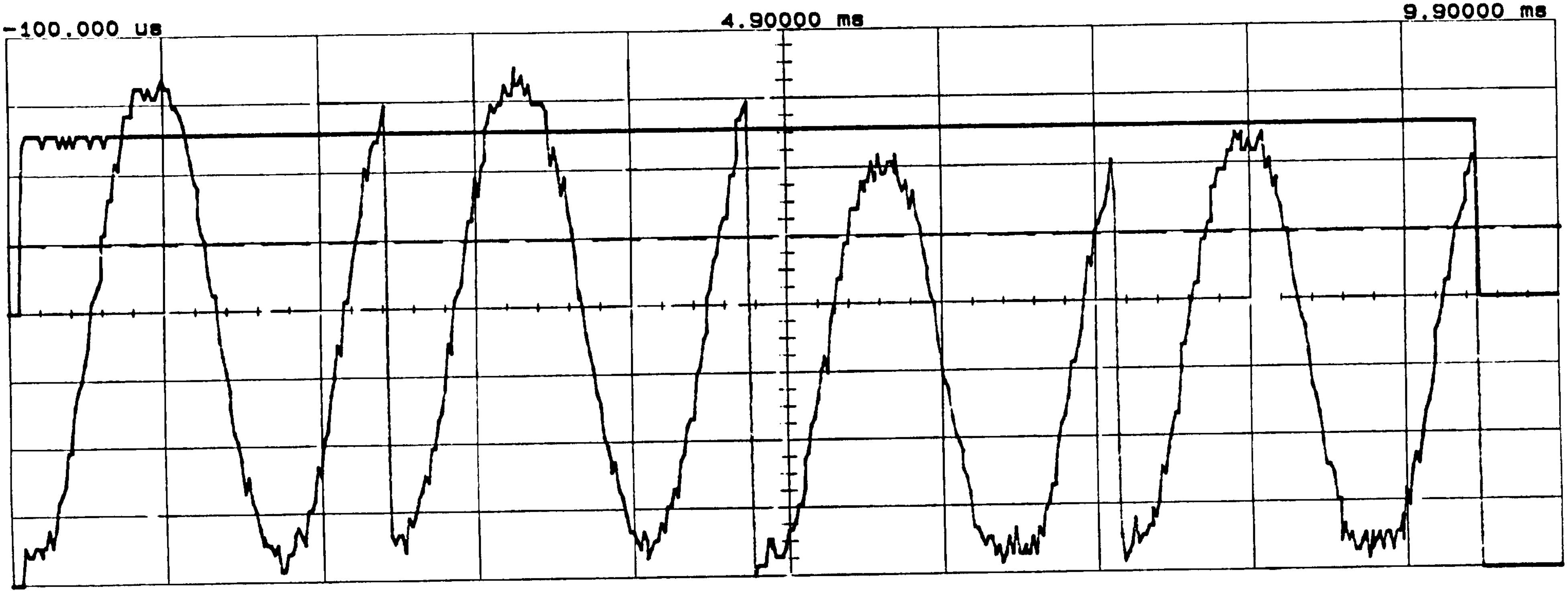


	Timebase	Delay/Pos	Reference	Mode
Main	1.00 ms/div	-100.000 us	Left	Realtime (NORMAL)

	Sensitivity	Offset	Probe	Coupling
Channel 1	2.00 V/div	0.00000 V	10.00 : 1	dc (1M ohm)
Channel 2	1.00 V/div	3.00000 V	10.00 : 1	dc (1M ohm)

Trigger mode : Edge
On Positive Edge Of Chan1
Trigger Level
Chan1 = 2.00000 V (noise reject OFF)
Holdoff = 40.000 ns

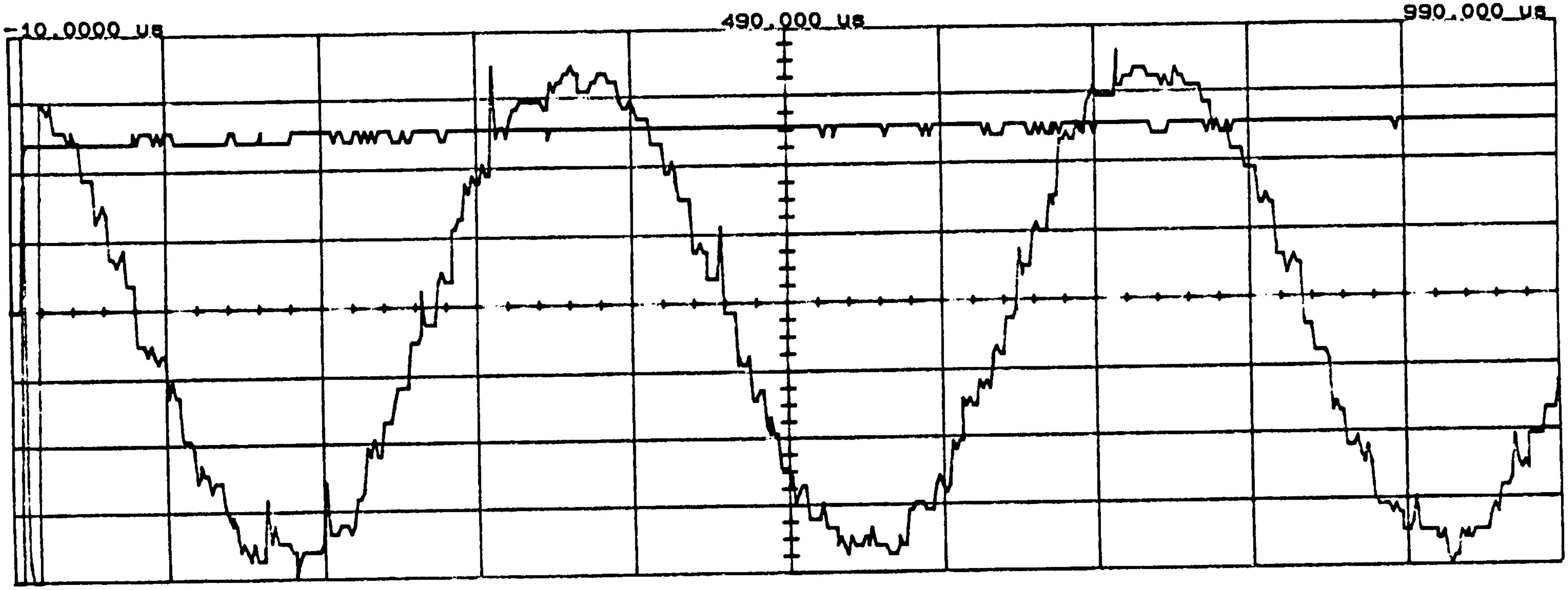
FIGURE 10



	Timebase	Delay/Pos	Reference	Mode
Main	1.00 ms/div	-100.000 us	Left	Realtime (NORMAL)
	Sensitivity	**Offset**	**Probe**	**Coupling**
Channel 1	2.00 V/div	0.00000 V	10.00 : 1	dc (1M ohm)
Channel 2	1.00 V/div	3.00000 V	10.00 : 1	dc (1M ohm)

Trigger mode : Edge
On Positive Edge Of Chan1
Trigger Level
Chan1 = 2.00000 V (noise reject OFF)
Holdoff = 40.000 ns

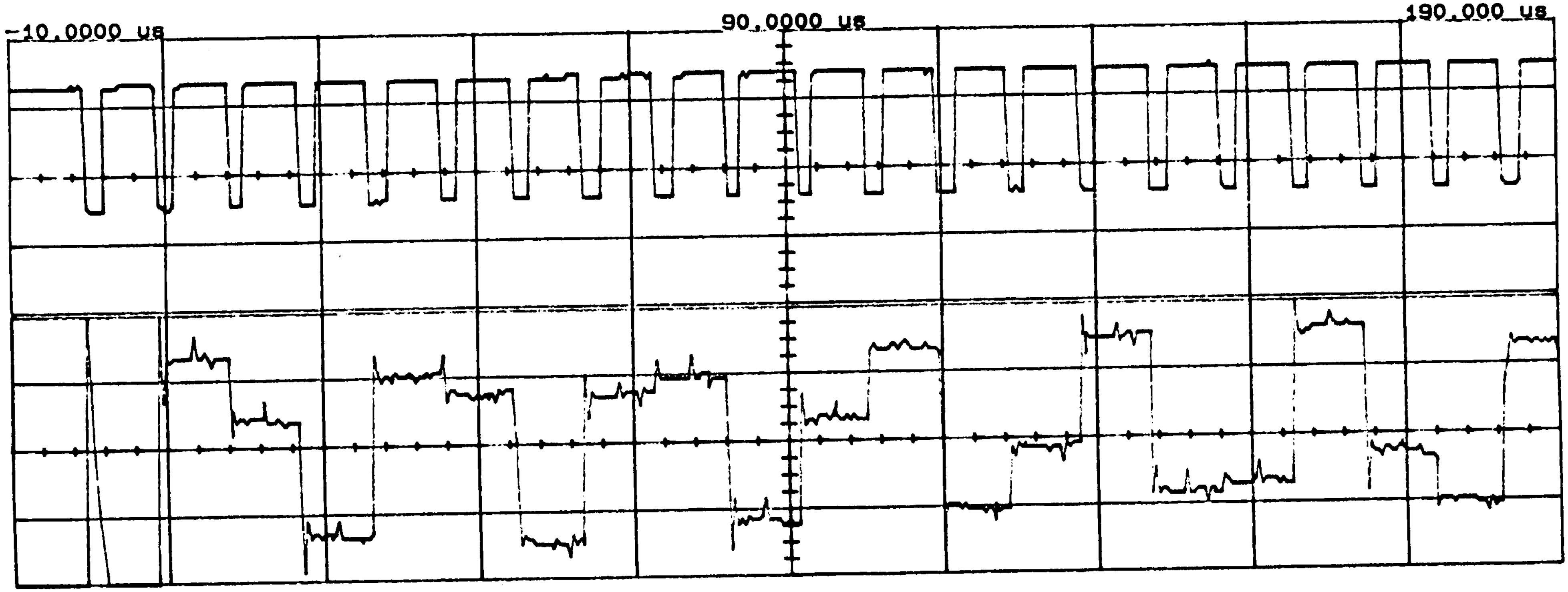
FIGURE 11



	Timebase	Delay/Pos	Reference	Mode
Main	100 us/div	-10.0000 us	Left	Realtime (NORMAL)
	Sensitivity	**Offset**	**Probe**	**Coupling**
Channel 1	2.00 V/div	0.00000 V	10.00 : 1	dc (1M ohm)
Channel 2	1.00 V/div	3.00000 V	10.00 : 1	dc (1M ohm)

Trigger mode : Edge
On Positive Edge Of Chan1
Trigger Level
Chan1 = 2.00000 V (noise reject OFF)
Holdoff = 40.000 ns

FIGURE 12

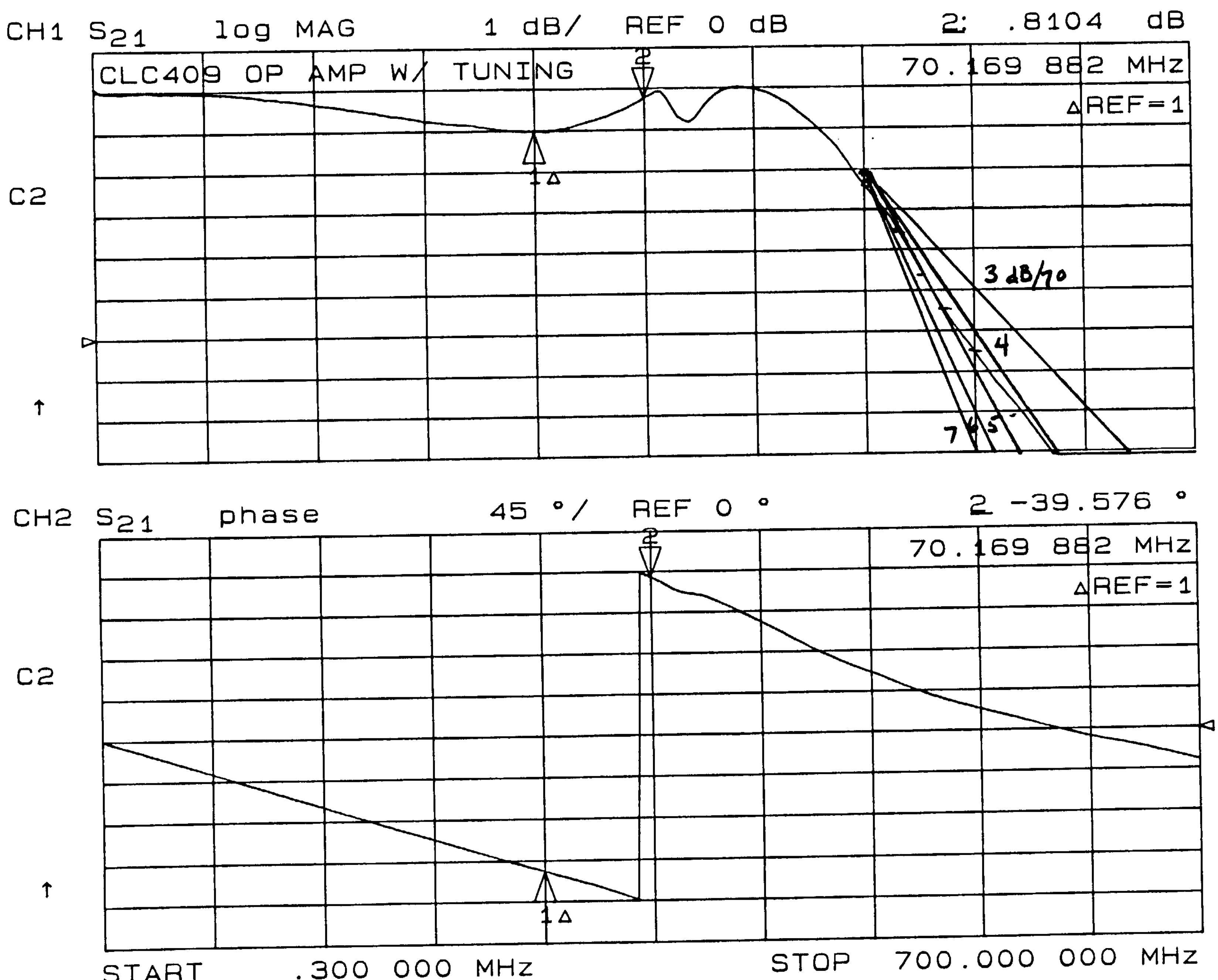


	Timebase	Delay/Pos	Reference	Mode
Main	20.0 us/div	-10.0000 us	Left	Realtime (NORMAL)

	Sensitivity	Offset	Probe	Coupling
Channel 1	4.00 V/div	0.00000 V	10.00 : 1	dc (1M ohm)
Channel 2	2.00 V/div	3.00000 V	10.00 : 1	dc (1M ohm)

Trigger mode : Edge
On Positive Edge Of Ext1
Trigger Level
Ext1 = 1.87500 V (noise reject OFF)
Holdoff = 40.000 ns

FIGURE 13



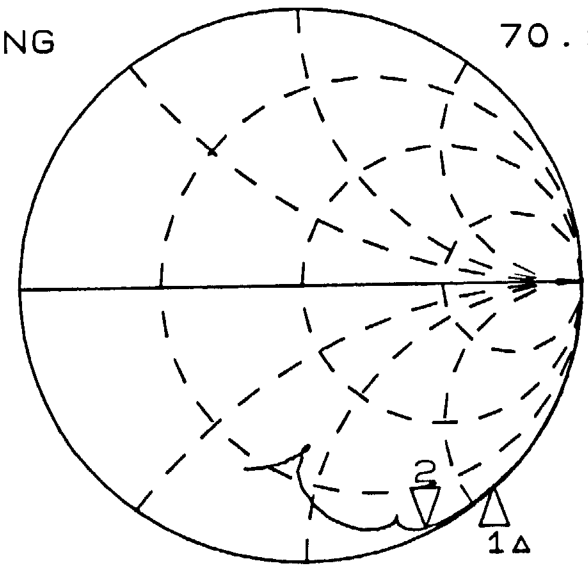
CH1 S21 log MAG 1 dB/ REF 0 dB 2: .8104 dB
CLC409 OP AMP W/ TUNING
70.169 882 MHz
ΔREF=1
C2
3 dB/70
4
7 6 5
CH2 S21 phase 45 °/ REF 0 ° 2 -39.576 °
70.169 882 MHz
ΔREF=1
C2
START .300 000 MHz
STOP 700.000 000 MHz

CH1 S_{11} 1 U FS 2 328.13 mΩ 34.395 Ω 78.012 nH

CLC409 OP AMP W/ TUNING 70.169 882 MHz

ΔREF=1

C2

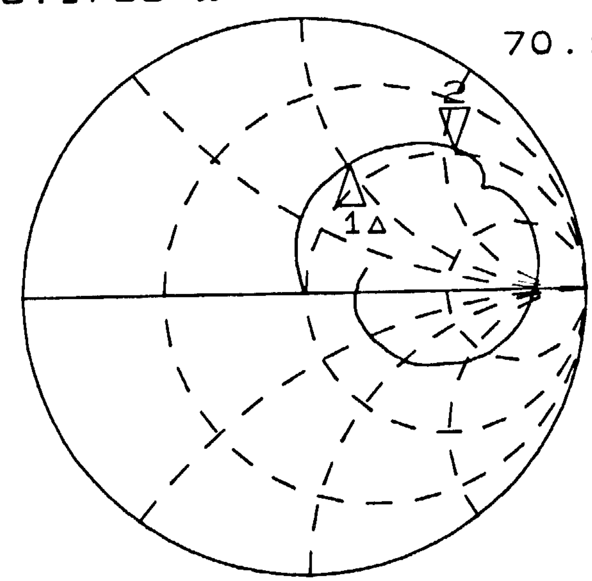


CH2 S_{22} 1 U FS 2: 5.1758 Ω 57.52 Ω 130.46 nH

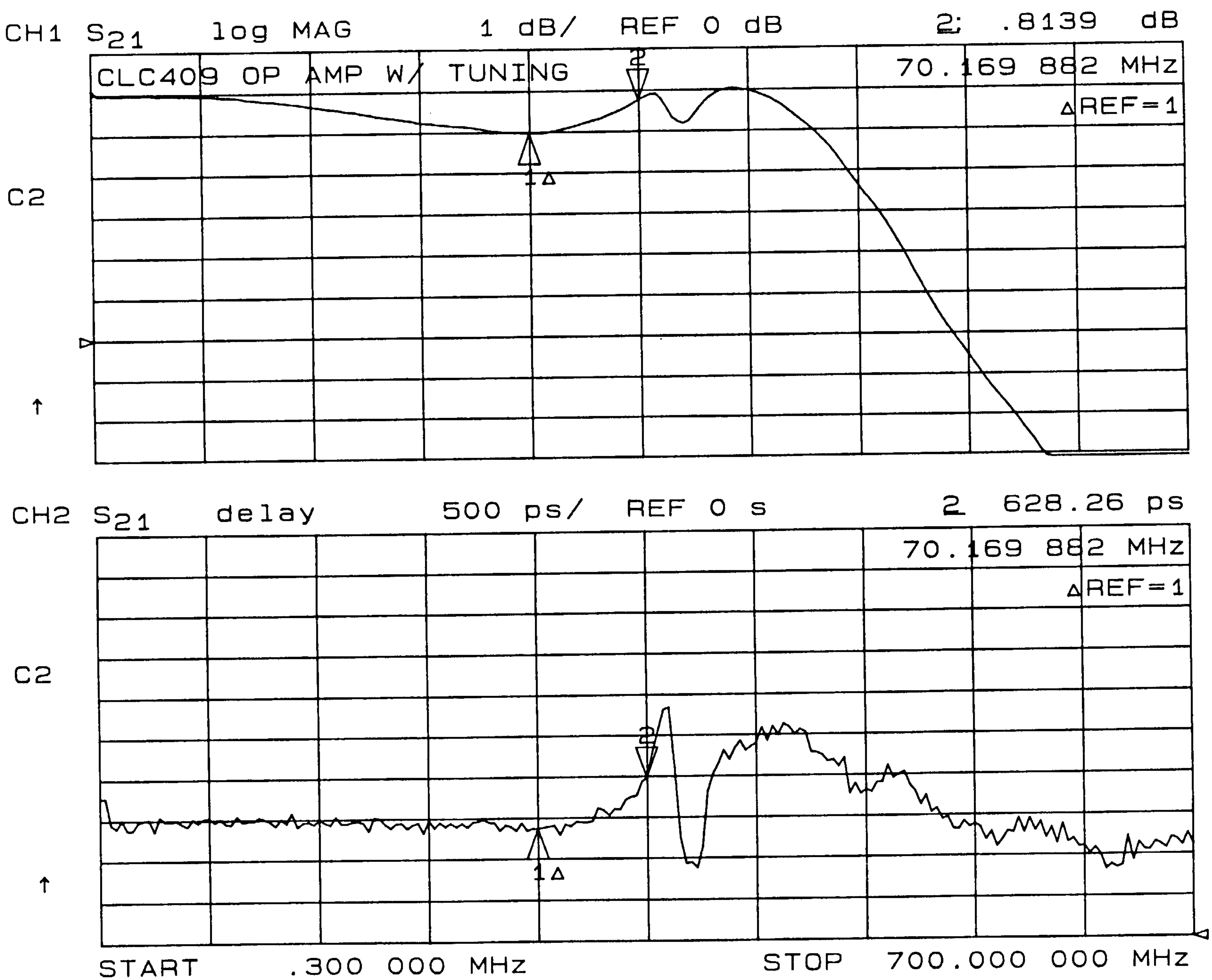
70.169 882 MHz

ΔREF=1

C2



START .300 000 MHz STOP 700.000 000 MHz


CH1 S21 log MAG 1 dB/ REF 0 dB 2: .8139 dB
CLC409 OP AMP W/ TUNING
70.169 882 MHz
ΔREF=1
C2
CH2 S21 delay 500 ps/ REF 0 s 2 628.26 ps
70.169 882 MHz
ΔREF=1
C2
START .300 000 MHz
STOP 700.000 000 MHz

APPENDIX #5

APPENDIX 5

HARDWARE INTERFACE

1. 16 bit bi-directional parallel data

2. Four wire handshake
   a. DMAWREQ - negative pulse by host initiates transfer
   b. DMAWACK - 0 after DMAREQ, 1 when accepted by μP.
   c. DMARAV - 0 after μP loads latch, 1 after read by host
   d. DMARREQ - negative pulse by host reads data and clears status

3. Write transaction (Host to μP)
   a. data is latched on trailing (+) edge of DMAWREQ
   b. low level on DMAWREQ sets WR status FF (DMAWACK = 0)
   c. μP read clears WR status FF (DMAWACK = 1)

4. Read transaction (μP to Host)
   a. Generally occurs as a result of request from host
   b. Only exception is error condition status
   c. μP writes data to latch
   d. this sets read status FF (DMARACK = 0)
   e. DMARREQ low places data on Q bus
   f. DMARREQ low level resets read status FF (DMARACK = 1)

5. commands are 16 bit data
   a. 12 lsb is value
   b. 4 msb is command
      0 = code address low - first word of Read, Write or Execute in code memory. (with low 12 bits of address)
      1 = data address low - first word of Read or Write in data memory. (with low 12 bits of address)
      2 = start address high - second word of Read or Write range (upper 8 bits of code address or 4 bits of

code address right justified in lower 12 bits)

3 = start address high - second (final) word to read one word from memory. Next transaction is a single read transaction.

4 = start address high - second word to write one word to memory. Next transaction is a single write transaction with a 16 bit data word.

5 = execution address high - second (final) word to start execution at address (upper 8 bits of address right justified in lower 12 bits)

6 = end address high - fourth (final) word to read an inclusive range of memory. This is preceded by a second low address (0 or 1). This is followed by N + 1 read transactions.

7 = end address high - fourth word to write an inclusive range of memory. This is preceded by a second low address (0 or 1). This is followed by N + 1 write transactions.

8 = I/O read - 12 lower bits are I/O address. Next transaction is a single read transaction with data word read (no prefix command needed).

9 = I/O write - 12 lower bits are I/O address. Next transaction is a single write transaction with data word to be output (no prefix command needed).

A = skip value - 12 lower bits contain number of cells to skip at the beginning of an acquire. This should be a multiple of 16.

B = acquire value - 12 lower bits contain number of cells to read and convert. This command also does a software arm. This should be followed by a hardware arm and trigger. The hardware will respond with a sync strobe, and the μP will initiate N read cycles with data.

C = auto acquire (optional) Number of cells to skip and

read determined by the μP by inspecting the data.
12 lower bits are ignored. Otherwise works like command 9.

D = calibrate. First call set mode, remaining calls give value of last acquire. Final call with value of 0FFFh calculates coeficients.

E - F reserved

a. These go to jump vectors in code RAM

b. Initially they are programmed as NOP's

6. Timing

a. When host writes data to system, data is latched on trailing edge of strobe. Status is level sensitive, so responds to leading edge. In order to prevent multiple reads by system, pulse width must be less than 500 ns.

b. When host reads data, both data enable and status are level sensitive. The strobe pulse width must be less than 500 ns in order to prevent μP from placing second word on bus before end of pulse.

c. Read data is valid from 30 ns after DMARREQ goes low until a small time after DMARREQ goes high (20 ns typ.)

d. Write data must be valid at least 30 ns before the rising edge of DMAWREQ and remain valid at least 5 ns after the rising edge of DMAWREQ.

e. The Status output bit (Data/Status, 0/1) is valid from the time DMARAV goes low until it goes high again.

f. The Command input bit (Data/Command, 0/1) must be valid from the falling edge of DMAWREQ until the subsequent rising edge of DMAWACK. This is best accomplished by providing an non-Tristate ff whose state is changed just before data is written.

g. The command bit is interrogated during a reset (either power up reset, or external reset). If it is a 1, a cold reset is forced (all variables initialized to default, calibration values all set to gain = 1 and offset = 0, all code extensions eliminated and a new code memory checksum computed. If Command = 0 a warm start is done instead, provided that the code checksum agrees with the value stored in the data memory.

7. Error and status codes (Status = 1)

0. OK
1. Warm start (after reset, unsolicited)
2. Cold start (after reset, unsolicited)
3. (reserved)
4. Completed (generally at end of block reads)
5. Illegal data - data written when not expected
6. No data - Command written when more data was expected, or during operation.
7. Write interrupt (not used)
8. Aborted - Command received after Acquire command, before arm signal received.
9. Not Triggered - A/D acquire cycle did not start, probably due to lack of a trigger.

0A. No Acquire - A/D acquire cycle did not complete.

0B. In Calibration - A/D cycle completed. No data will be sent because data is being used for calibration.
NOTE: attempt has been made to keep error codes in the range of 0 - 0Fh so that they can be loaded with a single word mov instruction.

NASA
National Aeronautics and Space Administration

Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.

4. Title and Subtitle

1 GHz Digitizer for Space Based Laser Altimeter

5. Report Date

16 October 1992

6. Performing Organization Code

7. Author(s)

E.J. Staples

8. Performing Organization Report No.

SR0130-9

10. Work Unit No.

9. Performing Organization Name and Address

Amerasia Technology Inc.
2248 Townsgate Road
Westlake Village, CA 91361

11. Contract or Grant No.

NAS5-30626

12. Sponsoring Agency Name and Address

National Aeronautics and Space Administration
Goddard Spaceflight Center
Greenbelt, MD 20771

13. Type of Report and Period Covered

Final Report
April 1989 - August 1991

14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

This is the final report for the research and development of the 1 GHz Digitizer fo Spaced Based Laser Altimeter. A Feasability model was designed, built, and tested. Only partial testing of essential functions of the Digitizer was completed, due to limited funding available at this time. Hybrid technology was incorporated which allows analog storage (memory) of the digitally sampled data. The actual sampling rate is 62.5 MHz, but executed in 16 parallel channels, to provide an effective sampling rate of 1 GHz. The average power consumption of the 1 GHz Digitizer is not more than 1.5 Watts. A 1 GHz SAW oscillator is incorporated for timing purposes. This signal is also made available externally for system timing. A software package was also developed for internal use (controls, commands, etc.) and for data communications with the host computer. The Digitizer is equipped with an on-board microprocessor for this purpose.

17. Key Words (Suggested by Author(s))

1 GHz Digitizer, SAW, Hybrid memory.

18. Distribution Statement

1- Contract Specialist
5- Technical Officer
1- NASA Scientific & Tech Info.Facility
(Accessioning Department)

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of pages	22. Price
Unclassified	Unclassified	99	

NASA FORM 1626 OCT 86