SOME FURTHER CONSIDERATIONS IN THE DESIGN AND IMPLEMENTATION OF A LOW-POWER, 15-BIT DATA ACQUISITION SYSTEM

by

JEFFREY DARREN BRADLEY

B. S., Kansas State University, 1985

A MASTER'S THESIS

submitted in partial fulfillment of the
requirements for the degree

MASTER OF SCIENCE

Department of Electrical and Computer Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1986

Approved by:

M S P Lucas

Major Professor

Acknowledgements

This work was sponsored and funded by the Systems Engineering Division 5238, Sandia National Laboratories, Albuquerque, New Mexico.

I would especially like to thank my major professor, Dr. M. S. P. Lucas for his friendship and insight during my research work. I would also like to thank Dr. J. Garth Thompson and Dr. Ruth Dyer for serving as my committee members, and Michelle White for her assistance in the preparation of some of the figures that appear in this thesis.

Finally, I would like to thank my wife, Lois, for her patience and understanding during my studies at KSU.

CONTENTS

LIST OF FIGURES

SOME FURTHER CONSIDERATIONS IN THE DESIGN AND IMPLEMENTATION OF A LOW-POWER, 15-BIT DATA ACQUISITION SYSTEM

Introduction

A need for remote and/or battery powered instrumentation has created a demand for low-power, high-resolution data acquisition systems which are capable of functioning for extended periods of time. Applications for such systems can be found in satellite instrumentation packages, biological field experiments, and oil well logging equipment. Over the past several years, a low-power data acquisition system (DAS) has been developed by the Department of Electrical and Computer Engineering at Kansas State University.

The low-power DAS will simultaneously sample two analog voltage signals (within a $\pm$ 5 V range) and perform a 15-bit analog-to-digital conversion on both sampled inputs. The analog-to-digital conversion is performed by a sign-magnitude, successive approximation type analog-to-digital converter. For the application that the DAS was designed, the sampling rate is 128 samples/second for each of two channels. Control of the DAS is performed by a microprocessor. The presence of the microprocessor also gives the DAS the capability to do some of its own data processing and error correction. The calculated power consumption for the DAS at the 128 samples/second rate is 36.5 mW.

Since the development work on the DAS was done by several graduate students at KSU over a period of several years, the

documentation is somewhat fragmented. The purpose of this thesis is to present an overview of the DAS, and to analyze and suggest solutions to several error sources that are inherent in the design. The overview will serve to tie together previous work, (both documented and undocumented), and to form a background for the analysis. The analysis will point out the major sources of error in a prototype DAS and show how the microprocessor can be used to null out some of the errors.

The following topics are covered in this thesis and are presented in the order shown:

- A review of the existing low-power DAS hardware
- Inherent error sources in the low-power DAS
- A microprocessor-based error compensation technique
- Recommendations for future implementations of the DAS

Coverage of related material and software listings are left to the appendices. It is assumed that the reader is acquainted with the successive approximation technique of analog-to-digital conversion and with the terminology related to the specification of analog-to-digital converters.

CHAPTER 1. REVIEW OF THE EXISTING LOW-POWER DAS HARDWARE

Introduction

This section reviews the existing hardware of a low-power data acquisition system (DAS). The initial design requirements for the DAS were:[1]

1. conversion by successive approximation technique
2. 15-bit resolution
3. maximum differential linearity error of $\pm$ 1/2 LSB
4. maximum integral linearity error of $\pm$ 1 LSB
5. no missing codes
6. bipolar input voltage range, $\pm$ 5 V
7. maximum input frequency of 45 Hz
8. two analog input channels
9. two conversions (one per channel) every 1/128 sec.
10. digital output data in two's-complement form
11. microprocessor control
12. power supply voltages of $\pm$ 7.5 VDC
13. power consumption of less than 80 mW (analog section)

The hardware consists of two parts: 1) an analog section consisting of a two channel track-and-hold and a 15-bit successive approximation type analog-to-digital converter (ADC), and 2) a digital section consisting of a microprocessor controller and supporting digital hardware. The analog section performs the

sampling and digitizing of two analog input signals. The digital section performs the control and data manipulation functions of the DAS and provides an interface between the analog section and a more general purpose host computer system.

The analog section is reviewed first. Next the digital section is reviewed. Finally, the performance of a single channel prototype DAS is reported, and briefly compared to that of a PC board version of the DAS.

The successive approximation type ADC was designed by Ragsdale.[1] The two channel track-and-hold was designed by Reed.[2] The control logic and microprocessor controller were designed by Doerfler.[3-5] The methods for testing the prototype DAS were also developed and implemented by Doerfler.

Analog Section

The analog section of the low-power DAS consists of two major blocks, as shown in the block diagram, Figure 1: 1) a two channel track-and-hold, and 2) a 15-bit successive approximation type ADC. Two supporting blocks are also included in the analog section. These are the control logic and a power supply filtering and switching network. The following sub-sections discuss each of these blocks. A discussion of the theory of operation of the complete analog section is also given.

Two Channel Track-and-Hold

An open-loop voltage follower track-and-hold design is used in the low-power DAS. The advantages offered by this design are

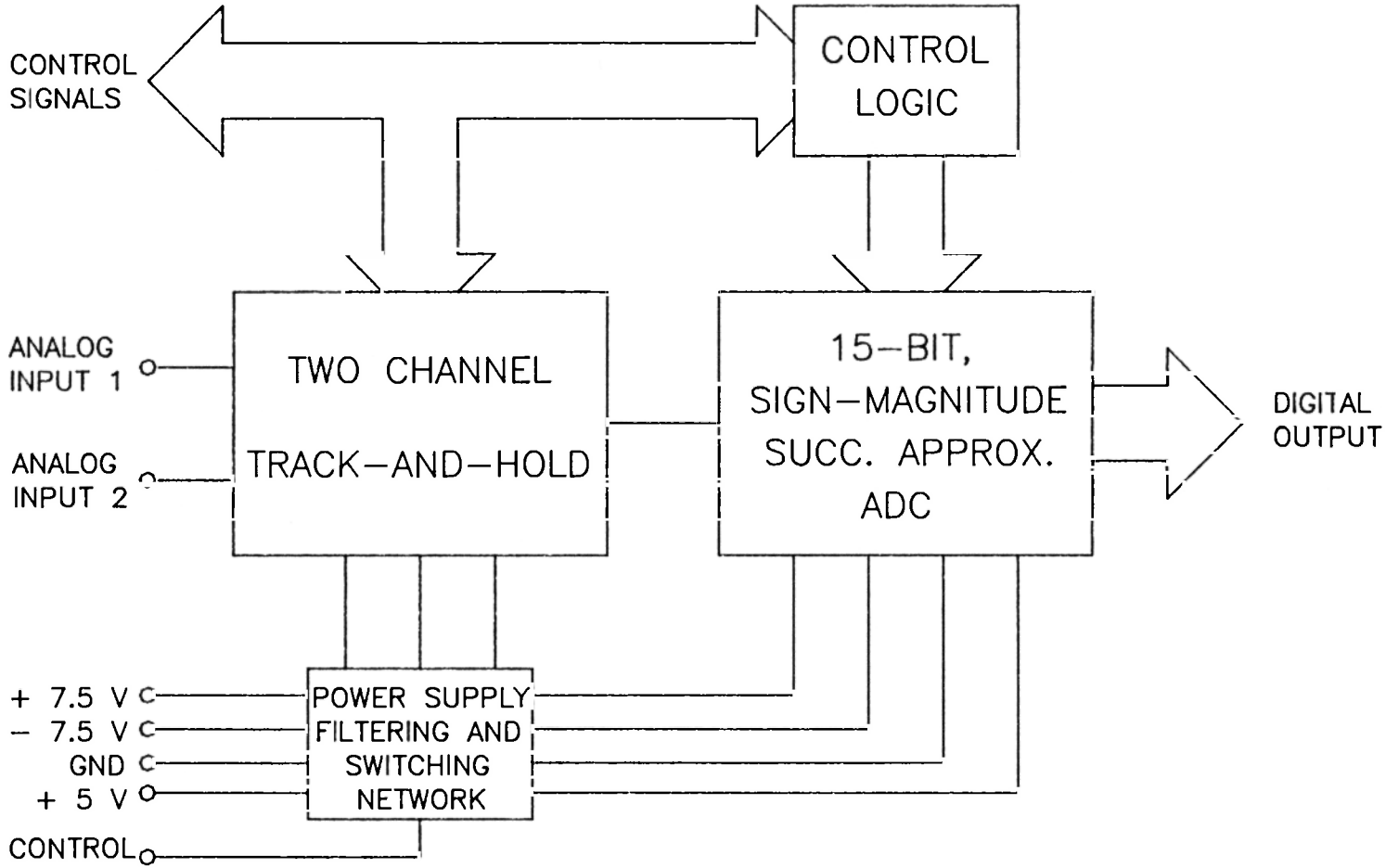


Figure 1. Block Diagram of the Analog Section

simplicity and stable performance. Since the DAS's application requires two input channels, two identical track-and-holds are placed in parallel. Their outputs are multiplexed so that the ADC can perform a separate conversion on each sampled input. The track-and-hold timing and multiplexing are handled by the microprocessor controller.

A circuit diagram of the two channel track-and-hold is shown in Figure 2. Several other features of the circuit should be noted. The inputs to the track-and-hold can be grounded so that the ADC can perform a ground test. Switches are cascaded to improve the off-isolation between the input and output. Also, a low-pass filter is used on the output of each track-and-hold channel to reduce output noise. The measured power consumption for the entire track-and-hold circuit is 3.28 mW.

Although a detailed discussion of the two channel track-and-hold can be found in Reed,[2] a few comments about the track-and-hold components are given here. The Precision Monolithics (PMI) OP-22 programmable micropower operational amplifiers were chosen for this application for their low power consumption, low offset voltage, high gain, and high CMRR. The Siliconix DG309 switch used for selecting the track/hold mode was chosen for its low power consumption, small gate-to-drain capacitance, and small charge transfer. Teflon capacitors are used for the hold capacitors and were chosen for their low dielectric absorption.

Successive Approximation ADC

The successive approximation type ADC that is used in the low-power DAS is designed to provide 15-bit resolution over a

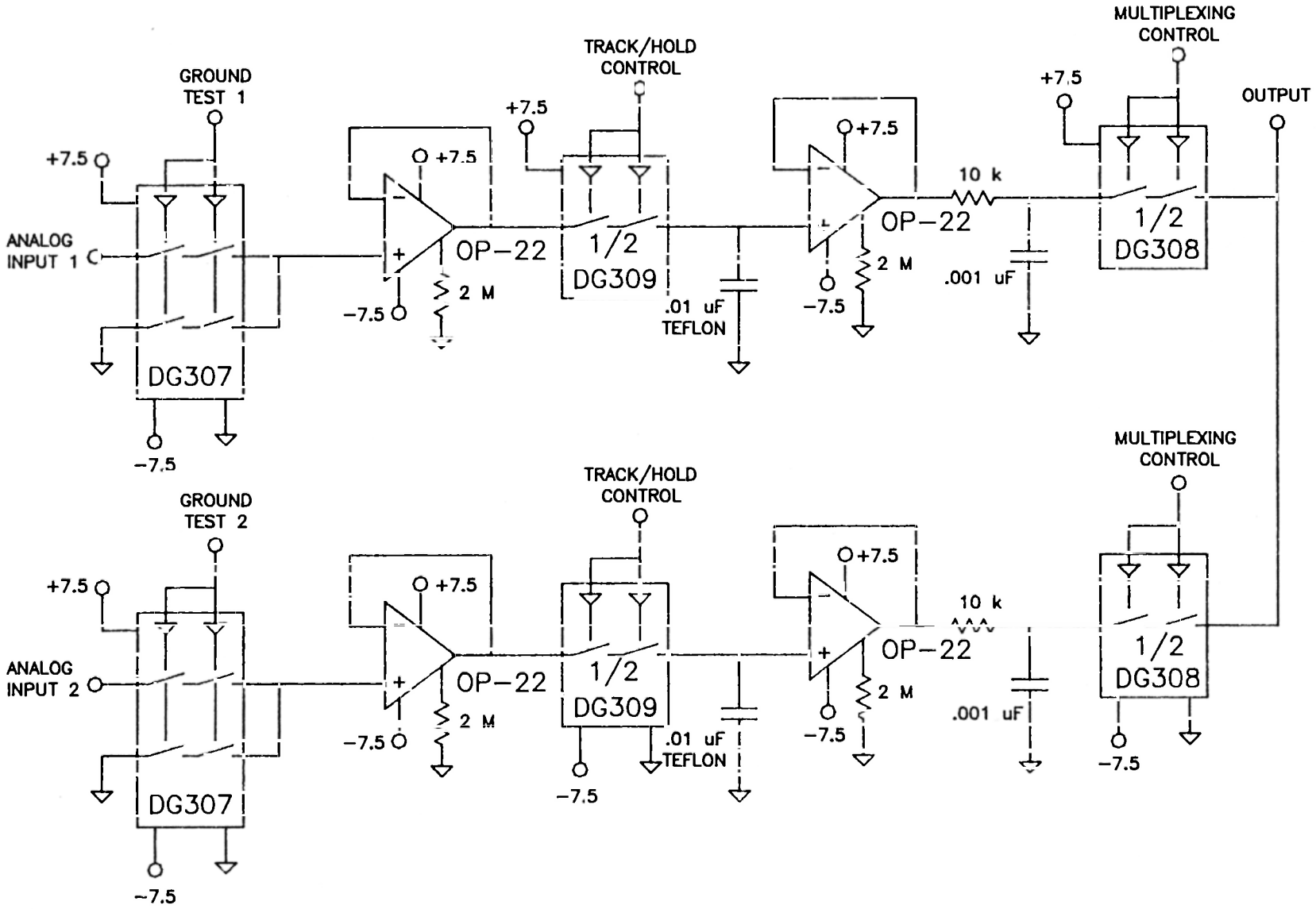


Figure 2. Circuit Diagram of the Track-and-Hold

± 5 V range. Operation of the ADC is handled by the microprocessor controller. The digital output data can be delivered to the microprocessor in either serial or parallel format. Low power consumption is achieved by turning off (power-switching) several of the components of the ADC when the conversion process is not taking place.

In this implementation, a unipolar 14-bit ADC is forced to act as a bipolar 15-bit ADC. This is done by first determining the polarity of the sampled input signal, and then performing a 14-bit conversion in the proper polarity range. The result of the 14-bit conversion represents the magnitude of the sampled input signal. The 15th bit (MSB) represents the polarity, or sign. This ADC therefore determines the digital output data in sign-magnitude form.

A circuit diagram of the successive approximation type ADC is shown in Figure 3. The heart of the ADC is the Datel Intersil DAC-HA14B 14-bit digital-to-analog converter (DAC). This DAC was chosen for its very low power consumption, wide supply voltage range, and rapid settling time. The last characteristic makes this DAC very useful in a power-switched application. A stable + 5 V reference for the DAC is provided by a PMI REF-02 precision voltage reference.

Digital data is delivered to the DAC by the successive approximation registers (SAR). Two 8-bit SARs are cascaded to form the 14 bits needed by the DAC. The SARs execute and control the binary search algorithm that is characteristic of the successive approximation technique. Once the SARs are started by the microprocessor controller, they automatically execute the binary

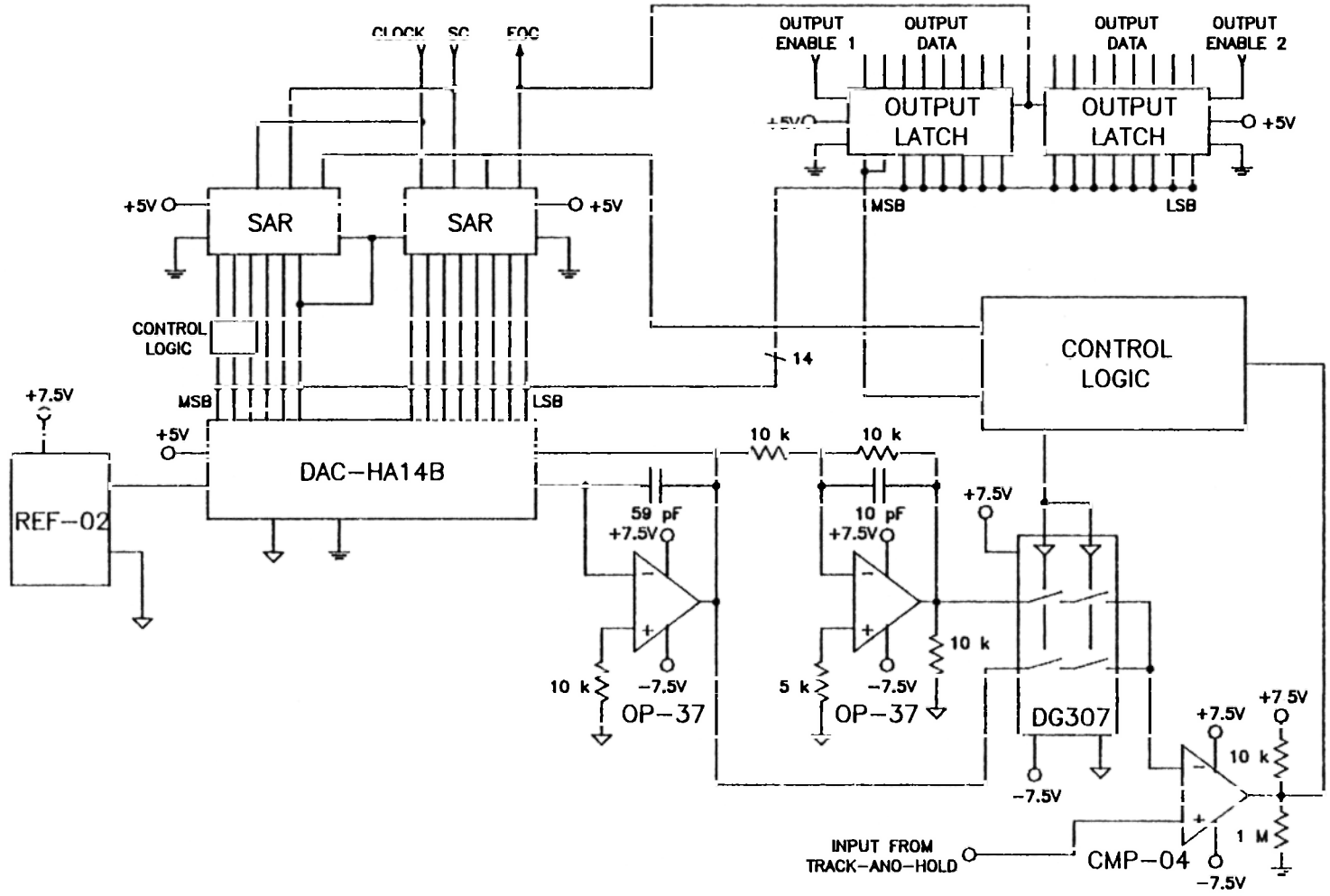


Figure 3. Circuit Diagram of the Successive Approximation ADC

search at a rate determined by the microprocessor provided system clock.

The current output of the DAC is converted to a negative voltage by a current-to-voltage circuit based on a PMI OP-37 precision high-speed operational amplifier. If the polarity of the sampled input signal voltage is negative, then the output of this op-amp is compared to the sampled input directly. If the polarity of the sampled input signal is positive, then the output of the current-to-voltage circuit is inverted by an inverting amplifier circuit also based on the OP-37. The output of the inverting amplifier is then compared to the sampled input. Feedback capacitors are included in both op-amp circuits to decrease their settling time. The OP-37 op-amp was chosen for its relatively low power consumption, high gain, high gain-bandwidth product, high slew rate, rapid settling time, and low offset voltages and currents.

A PMI CMP-04 low-power precision comparator makes all voltage comparisons for the determination of input signal polarity and for the binary search procedure. The CMP-04 was chosen for its low power consumption, high gain (high resolution), high CMRR, and rapid small signal response time. The output of the comparator is used by the SARs (after manipulation by control logic) to determine whether the bit being tested should be left set or should be reset.

As mentioned before, the digital output data can be delivered to the microprocessor controller in either serial or parallel format. If the data is to be delivered serially, then the bits are sent by the SARs to the microprocessor as they are

determined. If the data is to be delivered in parallel, then all bits are latched into a separate buffer at the end of the binary search. In this application, the output data is sent to the microprocessor in parallel format.

Several components of the ADC are power-switched to conserve power. These components are the REF-02 voltage reference, the OP-37 op-amps, and the CMP-04 comparator. Each of the power-switched components was carefully selected to gain the greatest benefit from power-switching. Their high performance, relatively low power consumption, and short turn-on settling time are all desirable characteristics for use in a power-switched mode. For this application, the sampling period is 1/128 s. Since the actual conversion process takes a small portion of this time, the power for these components can be turned off for most of the sampling period. For a sampling period of 1/128 s, the duty cycle for these components is roughly 15%. Obviously power-switching yields a significant savings in power consumption when the duty cycle is this low.

A detailed analysis of the design of the low-power, 15-bit successive approximation ADC can be found in Ragsdale.[1]

Control Logic

Very little control logic is required in the analog section. The logic performs two functions: 1) assisting the microprocessor controller in the determination of the input signal polarity, (sign bit), and 2) setting up the ADC to make the 14-bit magnitude determination in the proper polarity range. The control logic is shown in Figure 4. Note that the SARs and the

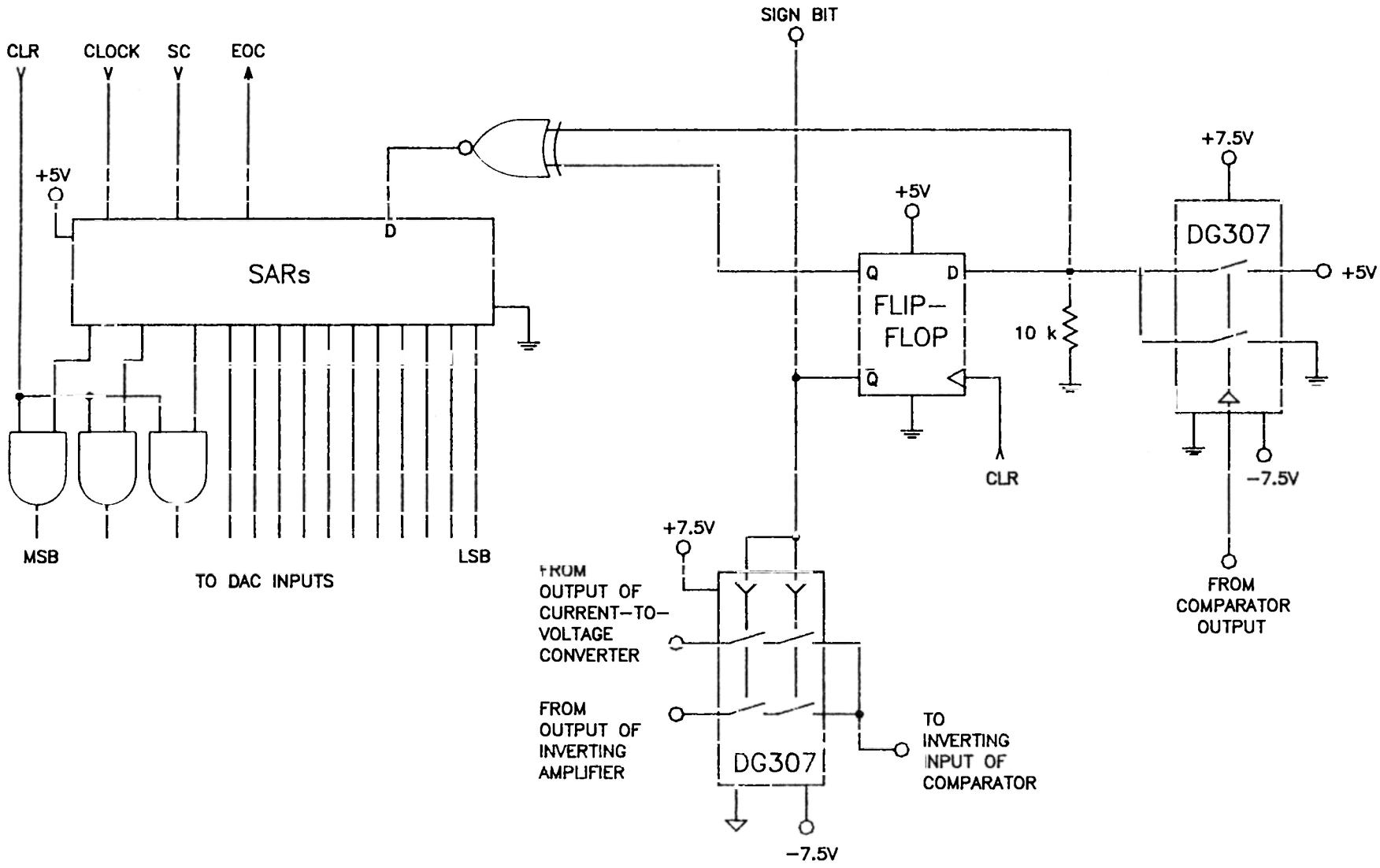


Figure 4. Circuit Diagram of the Control Logic

switch on the outputs of the OP-37 op-amps, which were shown previously in Figure 3, are also shown here for clarity. Also, in the following text, a logic high is represented as a "1" and a logic low is represented as a "0".

During the determination of the sign bit, a 0 V reference is needed at the inverting input of the comparator. To accomplish this, a 00...00 code is required at the DAC input. However, the SARs are not able to deliver a 00...00 code on demand. To get around this problem, the SARs are "false started" and all bits that might be set to a "1" by the SARs during the sign determination period are forced to "0". This is done with three 74C08 AND gates and the CLR line from the microprocessor controller. Setting the CLR line to "0" causes the output of the AND gates to be "0" regardless of the level of the other inputs.

A Siliconix DG307 analog switch is used as a level shifter to convert the + 7.5 V and - 7.5 V comparator output to + 5 V and 0 V logic levels respectively. The 74C74 D flip-flop is used to store the sign bit that results from the input signal polarity determination, and to set up the ADC for the magnitude determination. After the sign bit is stored, the Q output of the flip-flop and a 74HC266 exclusive NOR gate are used to provide the correct logic levels to the D inputs of the SARs. For magnitude determination in the positive range, the logic levels required at the D inputs of the SARs are a "1" if the comparator output is + 7.5 V and a "0" if the comparator output is a - 7.5 V. For magnitude determination in the negative range, the required logic levels are reversed. The $\overline{Q}$ output of the flip-flop is used to select the proper OP-37 op-amp output for the

magnitude determination. If the sampled input signal is a negative voltage, the output of the current-to-voltage converter circuit is selected. If the sampled input signal is a positive voltage, the output of the inverting amplifier circuit is selected. The $\overline{Q}$ output of the flip-flop is also made available to the output latches of the analog section for use as the two most significant bits, or sign bits.

Power Supply Filtering and Switching Network

The power supply filtering and switching network is shown in Figure 5. Power supply filtering is required for the analog section to reduce the effects of system noise on the conversion accuracy. This noise comes from the operation of both the digital and analog components of the DAS. A three-pole low-pass filter is placed between the external power supply and each analog component; a two-pole low-pass filter is placed between analog components.

Several components of the ADC are power-switched to reduce their power consumption. Switching is performed by analog switches (Siliconix DG308) under microprocessor control. More detail on the power supply filtering and switching network can be found in Ragsdale.[1]

Theory of Operation, Analog Section

A step-by-step example of the analog section's operation is given here. The example will cover one conversion cycle (conversions performed on both inputs). The digital output data will be in parallel format, and will be taken from the analog board in

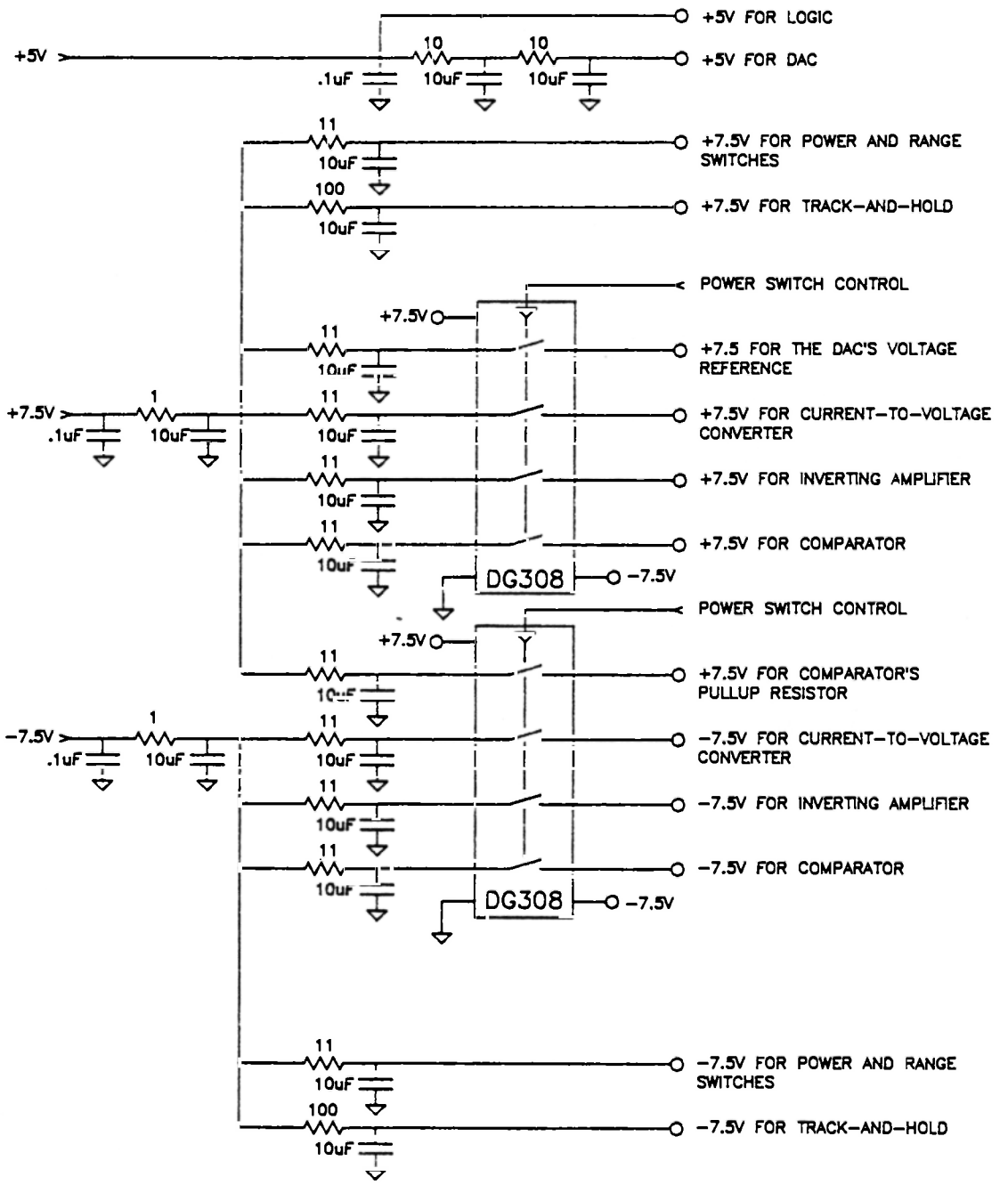


Figure 5. Power Supply Filtering and Switching Network

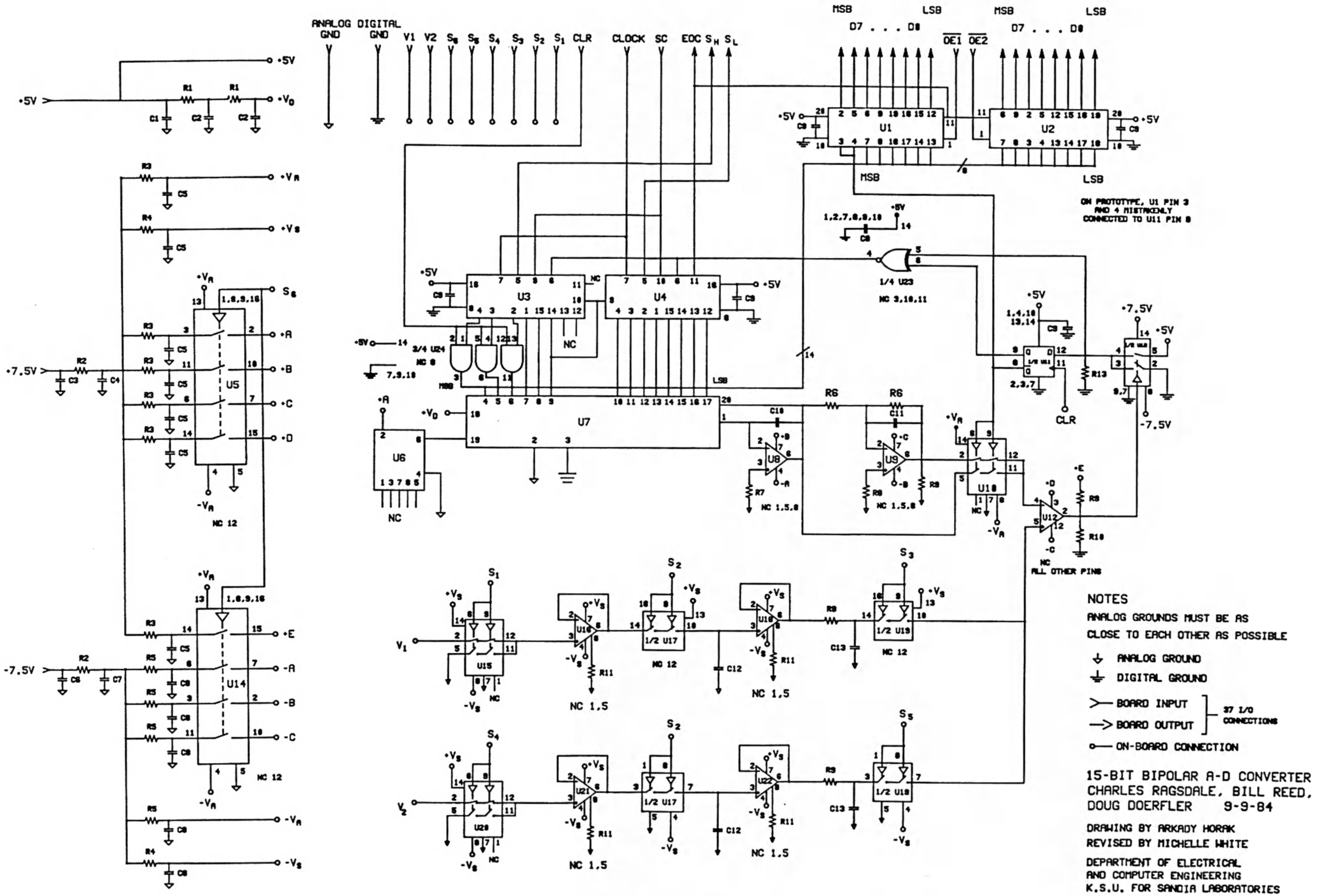
two bytes. Also, in the following text a logic high will be represented as a "1" and a logic low will be represented as a "0". The reader should refer to the circuit diagram for the complete analog section, Figure 6, and the accompanying timing diagram, Figure 7, while reading this section.

The microprocessor controller is required to provide ten control signals plus a clock signal to the analog section. The control signals are:

1. S_1 - ground test control for input channel 1
2. S_2 - track/hold mode control for the track-and-hold
3. S_3 - multiplexing control for input channel 1
4. S_4 - ground test control for input channel 2
5. S_5 - multiplexing control for input channel 2
6. S_6 - power-switching control
7. SC - start conversion control
8. CLR - sign determination control
9. $\overline{OE1}$ - latch output enable, most significant byte
10. $\overline{OE2}$ - latch output enable, least significant byte

In return, the analog section provides the microprocessor with an end-of-conversion flag (EOC), two serial data lines (S_H and S_L), and sixteen parallel data lines. Since the output data will be in parallel format for this example, S_H and S_L will be ignored here. Information on their use can be found in the literature.[6]

Before a conversion cycle begins, the control signals are in the following states (see page 19):



NOTES

ANALOG GROUNDS MUST BE AS CLOSE TO EACH OTHER AS POSSIBLE

⏚ ANALOG GROUND

⏚ DIGITAL GROUND

> BOARD INPUT

> BOARD OUTPUT

} 37 I/O CONNECTIONS

o— ON-BOARD CONNECTION

15-BIT BIPOLAR A-D CONVERTER
CHARLES RAGSDALE, BILL REED, DOUG DOERFLER 9-9-84

DRAWING BY ARKADY HORAK
REVISED BY MICHELLE WHITE

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
K.S.U. FOR SANDIA LABORATORIES

PARTS LIST IN APPENDIX A

Figure 6. Circuit Diagram of the Analog Section

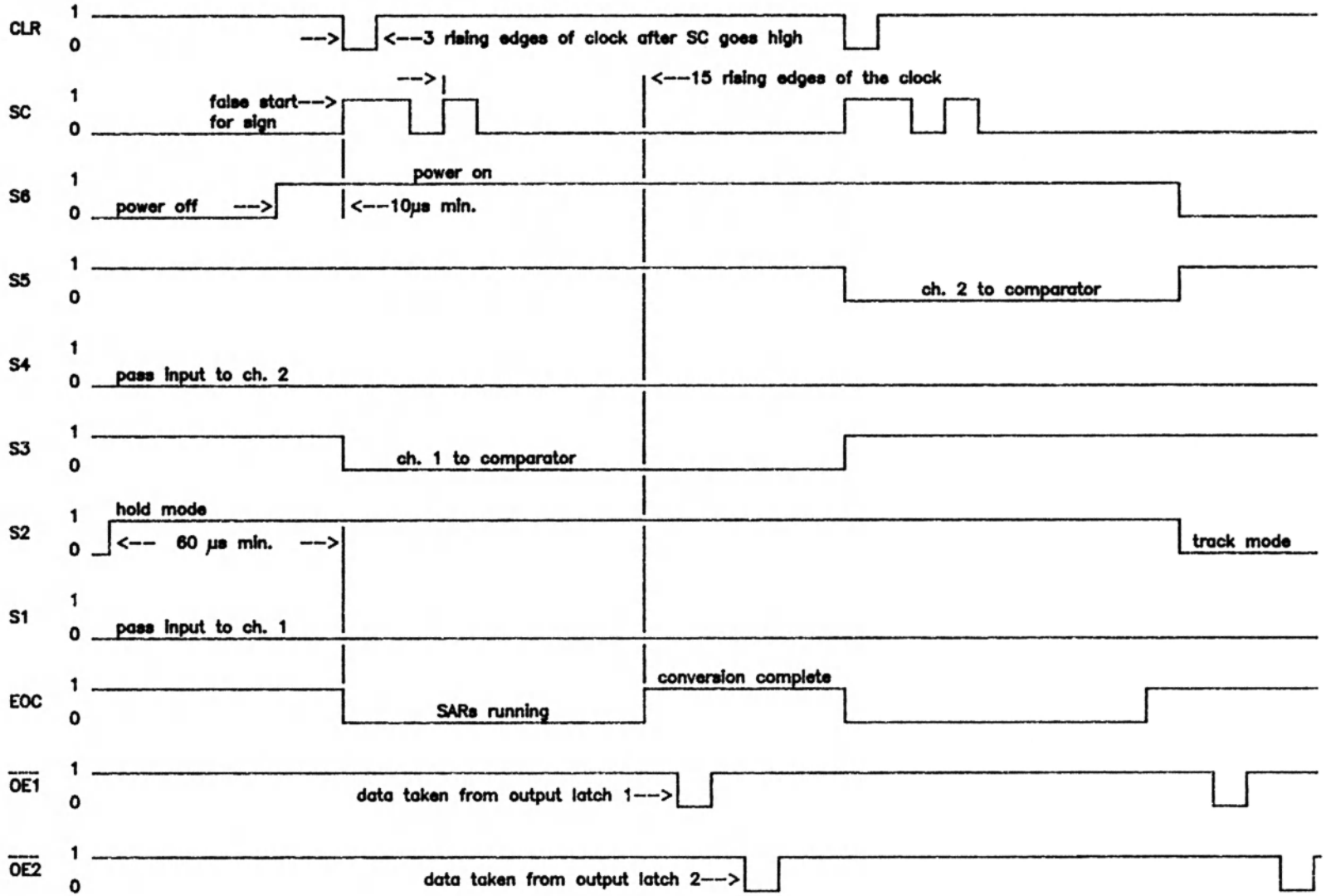


Figure 7. Timing Diagram for the Analog Section [5]

1. S_1 = "0", input signal passed to channel 1
2. S_2 = "0", track-and-hold in track mode
3. S_3 = "1", channel 1 cut off from ADC
4. S_4 = "0", input signal passed to channel 2
5. S_5 = "1", channel 2 cut off from ADC
6. S_6 = "0", power off
7. SC = "0", SARs waiting for the start signal
8. CLR = "1", normal state for the CLR line
9. $\overline{OE1}$ = "1", latch outputs in high impedance state
10. $\overline{OE2}$ = "1", latch outputs in high impedance state

The track-and-hold's buffer amplifiers U16 and U21 and the hold capacitors C12 are tracking the input signals. The ADC is in a quiescent state.

The conversion cycle is started by setting S_2 to "1". This puts both channels of the track-and-hold in the hold mode. The track-and-hold must then be allowed to settle for a minimum of 60 μs. Next the power for the power-switched components is turned on by setting S_6 to "1". The power-switched components are then allowed to settle for a minimum of 10 μs.

After the input signals have been sampled and the power to the ADC has been switched on, the polarity (sign) determination process begins. Several things now happen simultaneously. S_3 is set to "0", allowing the sampled signal from channel 1 to pass to the ADC's comparator. SC is set high, signalling the SARs to begin a conversion. This is the "false start" that was mentioned before in the Control Logic section. In response, the SARs set the EOC line to "0" and begin their usual binary search.

However, as discussed before, it is desired that the digital input to the DAC be 00...00 during the sign determination. This is accomplished by setting the CLR line to "0" at the same time that the SC line is set to "1".

During the sign determination process the sampled input is compared to a 0 V reference (ideally). The reference is taken from either the output of op-amp U8 or the output of op-amp U9, depending on whether the polarity of the last conversion was positive or negative. The comparator outputs + 7.5 V if the sampled signal is positive with respect to the reference or - 7.5 V if the sampled signal is negative with respect to the reference. These outputs are shifted to + 5 V and 0 V respectively.

The sign determination process is completed by setting the CLR line to "1". The shifted output of the comparator is latched into the flip-flop U11 by the rising edge of the CLR signal. A "1" is latched if the sampled input is positive; a "0" is latched if the sampled input is negative. Finally SC is set to "0" in preparation for the magnitude determination.

The magnitude determination is started by a pulse on the SC line. The conversion of the rest of the 14 bits now proceeds in the usual fashion for a unipolar 14-bit successive approximation converter with a 5 V span. The ADC is set up for the proper polarity range by the outputs of flip-flop U11, as was discussed in the Control Logic section. The SARs automatically control the binary search.

When the conversion is finished the SARs set EOC to "1". The rising edge of EOC clocks the output latches, causing the

results of the conversion to be held in the latches. The microprocessor controller is then free to retrieve the data from the output latches of the analog section. First the $\overline{OE1}$ line is set to "0", enabling the outputs of latch U1. The most significant byte of the data can then be read. After reading the first byte, $\overline{OE1}$ is returned to "1". Data is retrieved from latch U2 in a similar manner, using the $\overline{OE2}$ line to enable the outputs of U2. Conversion of the sampled signal from channel 1 is now complete.

To begin the conversion of the sampled signal from channel 2, S_3 is set to "1" to cut off channel 1 from the comparator and S_5 is set to "0" to pass the sampled signal from channel 2 to the comparator. The conversion of the sampled signal from channel 2 then proceeds in the same manner as described before for channel 1. At the end of the conversion, S_6 is set to "0" to turn the power to the ADC off and S_2 is set to "0" to put the two channel track-and-hold back into the track mode. The data for channel 2 is then retrieved from the analog section as described before. The two channel conversion cycle is now complete.

Digital Section

The digital section of the low-power DAS consists of a microprocessor controller and supporting digital hardware, as shown in the circuit diagram, Figure 8. The digital section performs two functions. The first function is to control the analog section. The second function is to provide an interface between the analog section and a host computer system.

The digital section's components and theory of operation are

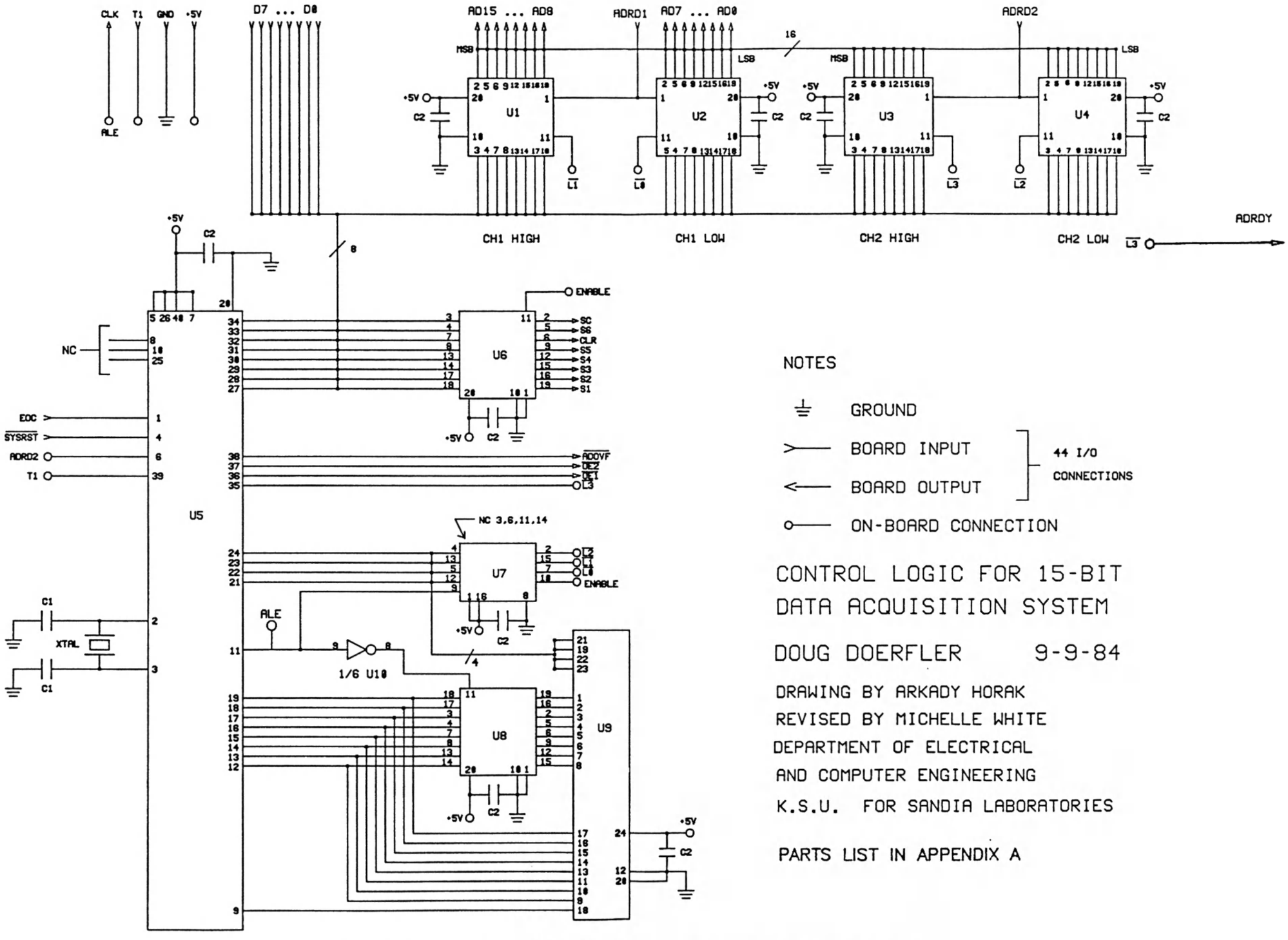


Figure 8. Circuit Diagram of the Digital Section

discussed in the following sub-sections. A discussion of the interfacing requirements for connection to a host computer system is also given.

Microprocessor Controller

The microprocessor used in the digital section is the Intel 80C39 CHMOS Single-Component 8-bit Microcomputer. The 80C39 is a low-power version of the Intel 8039 and is a component in the MCS-48 family. In addition to the CPU, the 80C39 has a 128 x 8 RAM, 27 I/O lines, and an 8 bit timer/counter on board. The 80C39 also provides a system clock for memory timing and other uses, at a frequency that is 1/15 of the crystal oscillator frequency. The low power consumption of the 80C39 makes this microprocessor ideal for battery powered operation. Near optimum power consumption for the combination of the analog and digital sections is achieved with a crystal oscillator frequency of 1 MHz. A 1 MHz crystal was chosen for its availability.

Twenty-four of the I/O lines are arranged into three 8-bit bidirectional ports, called BUS, PORT1, and PORT2. In this application, BUS is used as an address/data bus for an EPROM. PORT1 is used for sending control signals to the analog section, retrieving data from the analog section's output latches, and sending data to the digital section's output latches. PORT2 is used for controlling the latches on both the digital and analog sections, and for addressing the EPROM mentioned before.

The reader should refer to the literature for more specific information regarding the 80C39.[7]

Supporting Digital Hardware

The 80C39 is supported by six 8-bit latches, one 4-bit latch, and a 32K-bit UV-erasable EPROM. All supporting hardware is CMOS to reduce power consumption. The EPROM, a 27C32, contains the program for the DAS's operation. Four of the 8-bit latches, (U1 - U4), are used for holding the two 16-bit data words that have resulted from a conversion cycle. These data words are available to the host computer system. One of the 8-bit latches, (U6), is used to hold the ADC's control signals while the 80C39 is using PORT1 for data transfer. The last 8-bit latch, (U8), is used for holding the lower eight bits of the EPROM's address while the BUS is being used to retrieve instructions from the EPROM. The 4-bit latch, (U7), is used for holding four of the latch control signals while the lower four bits of PORT2 are being used as the upper four address lines for the EPROM.

Theory of Operation. Digital Section

A step-by-step example of the digital section's operation is given here. This example will parallel the one given previously for the analog section. Again, a logic high will be represented as a "1" and a logic low will be represented as a "0". The reader should refer to the circuit diagram for the digital section, Figure 8, the circuit diagram for the analog section, Figure 6, and the timing diagram, Figure 9. The assembly language software for this example can be found in Appendix C.

In addition to the ten control signals listed before in Theory of Operation, Analog Section, the microprocessor provides

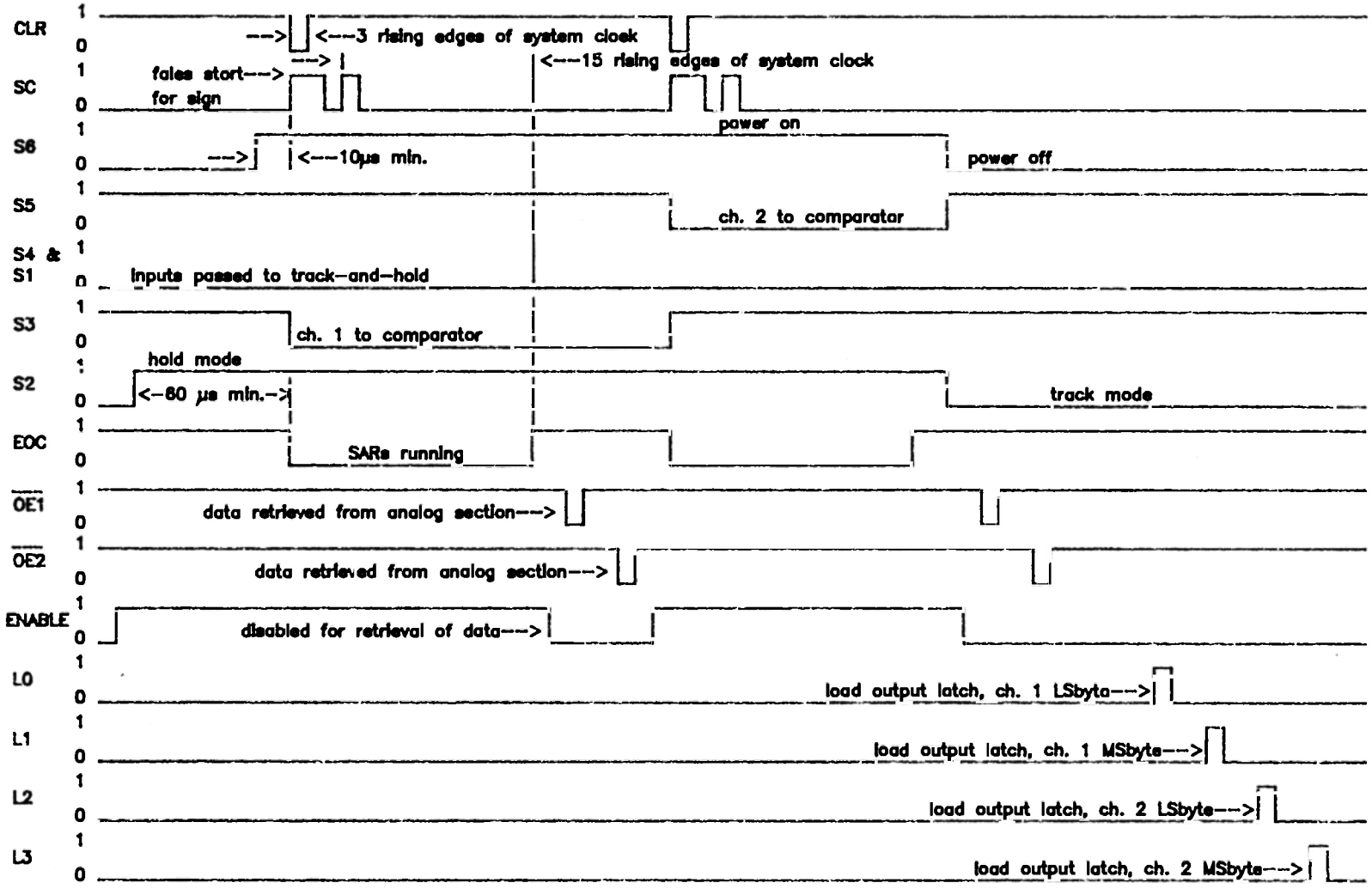


Figure 9. Timing Diagram for the Data Acquisition System [5]

five other control signals which are used within the digital section. They are:

1. ENABLE - latch enable for U6, ADC control signals
2. L0 - latch enable for U2, output data latch, channel 1, least significant byte
3. L1 - latch enable for U1, output data latch, channel 1, most significant byte
4. L2 - latch enable for U4, output data latch, channel 2, least significant byte
5. L3 - latch enable for U3, output data latch, channel 2, most significant byte

These five control signals, plus $\overline{OE1}$ and $\overline{OE2}$, are assigned to PORT2. The remaining eight control signals for the ADC are assigned to PORT1.

The microprocessor begins a conversion cycle by setting up its internal timer/counter to generate an interrupt at the beginning of the next conversion cycle. The timer/counter then runs by itself in the background while the microprocessor takes care of its duties. The microprocessor next sets ENABLE to "1" to allow the ADC control signals to pass through the latch U6.

After enabling latch U6, the microprocessor executes the control sequence described before in Theory of Operation, Analog Section. However, before the microprocessor can retrieve the data from the analog section, the latch U6 must be disabled by setting ENABLE to "0". This is done so that the control signals presently occupying PORT1 are not corrupted by the incoming data, which must also use PORT1. The data is retrieved in two 8-bit bytes as explained before. The two bytes are temporarily stored in RAM and the control signals for the ADC are reinitialized on

PORT1. Then the latch U6 is enabled again and the conversion cycle continues for channel 2 as described before.

When the conversion process for channel 2 is complete and the analog board has been returned to the quiescent state, latch U6 is disabled. The data is retrieved from the analog section again and temporarily stored in RAM. At this point the microprocessor is free to perform any processing on the data that is required. In this application the microprocessor must convert the sign-magnitude data into 2's complement form.

After the processing is done, the resulting data can be loaded into the four output latches, a byte at a time, through PORT1. A byte of data is loaded into an output latch by first setting its latch enable to "1", sending the data out through PORT1, and then setting the latch enable to "0". This is done for each of the four data bytes. Latch U3 must be loaded last, since its latch enable, L3, is also used to signal the host computer that data is present in the output latches. After the data has been loaded into the latches, the host system can retrieve the data from the output latches, either with or without handshaking.

At this point, the digital section is finished with its duties for the present conversion cycle. The microprocessor now waits in a loop until the timer/counter interrupts to signal the beginning of the next conversion cycle.

Interfacing with a Host System

To a host computer system, the DAS appears as a peripheral with a 16-bit data port and six control lines. The control lines

are:

1. ADRDY - flag signalling that data is present in the output latches, output from the digital section
2. ADRD1 - output enable for latches U1 and U2, input to the digital section
3. ADRD2 - output enable for latches U3 and U4,and flag signalling that data has been received, input to the digital section
4. T1 - arbitrary flag, input to the digital section
5. $\overline{\text{SYSRST}}$ - reset for the DAS, input to the digital section
6. $\overline{\text{ADCVF}}$ - flag signalling data overflow, output from the digital section

Passage of data from the DAS to the host system can be accomplished with or without handshaking.

An example interfacing scheme is given here. It involves passing two 16-bit data words using partial handshaking. Refer to the timing diagram, Figure 10.

The digital section first signals the host system that data is present in the output latches by setting ADRDY (L3) to "1". The host system receives the signal and enables the outputs of latches U1 and U2 by setting ADRD1 to "1". The host system then reads the data from these latches and returns ADRD1 to "0". The host system repeats the process for latches U3 and U4 using the ADRD2 line. While the host system is retrieving the data, the microprocessor in the digital section is waiting for the ADRD2 line to be set to "1". When this occurs, the microprocessor responds by going to an idle state and waiting for its timer/-counter to signal the beginning of the next conversion cycle.

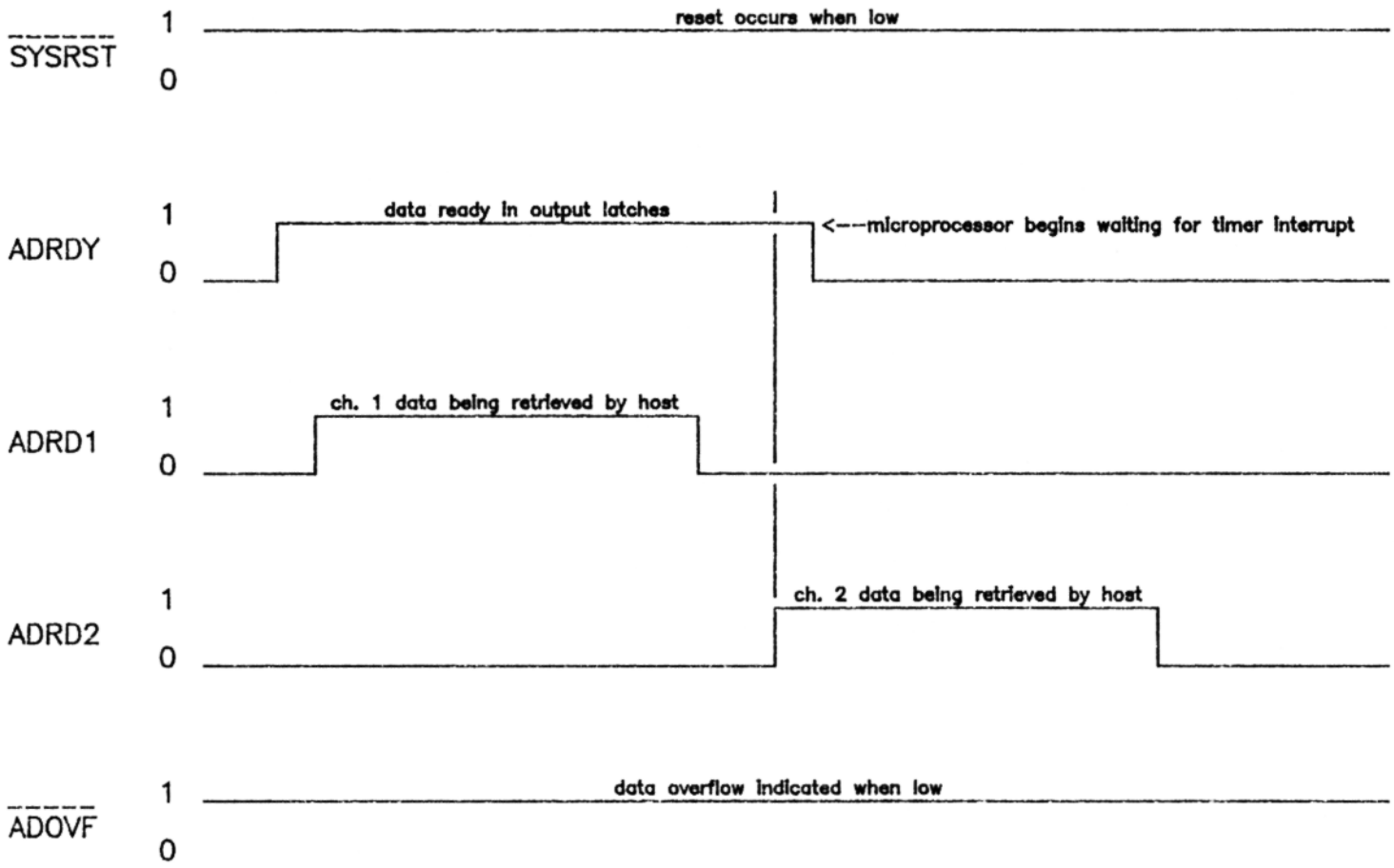


Figure 10. Timing Diagram for an Example Host System Interface

Alternatively, the T1 line could have been used to signal the microprocessor that the data had been retrieved by the host system. In another application, where continuous running of the DAS is not required, the T1 line could be used to signal the microprocessor to begin a conversion.

Performance of the Existing Low-Power DAS

The performance of the low-power DAS is reported here. A full explanation of the test techniques used to obtain the following performance data is beyond the scope of this thesis; however, the test techniques are discussed in detail in Doerfler.[4]

Briefly, a spectrally pure, 45 Hz, bipolar sine wave was applied to the input of a single channel, wire-wrapped prototype DAS, and using automated test techniques, exhaustive performance data was compiled and processed over the full span of the DAS. Such dynamic test techniques yield results that more closely represent the DAS's real world performance. In order to investigate the errors introduced by the DAC within the DAS's ADC, the tests were repeated three different times with DACs from different lot numbers. A plot of the differential linearity error over the full span of the DAS is shown in Figure 11a. A plot of an FFT that was calculated from the data is shown in Figure 11b. Results are summarized in Table 1.

A very large differential linearity error at the 00...00 code is evident in Figure 11a. Significant harmonic distortion is apparent in Figure 11b. The data in Table 1 indicates that the

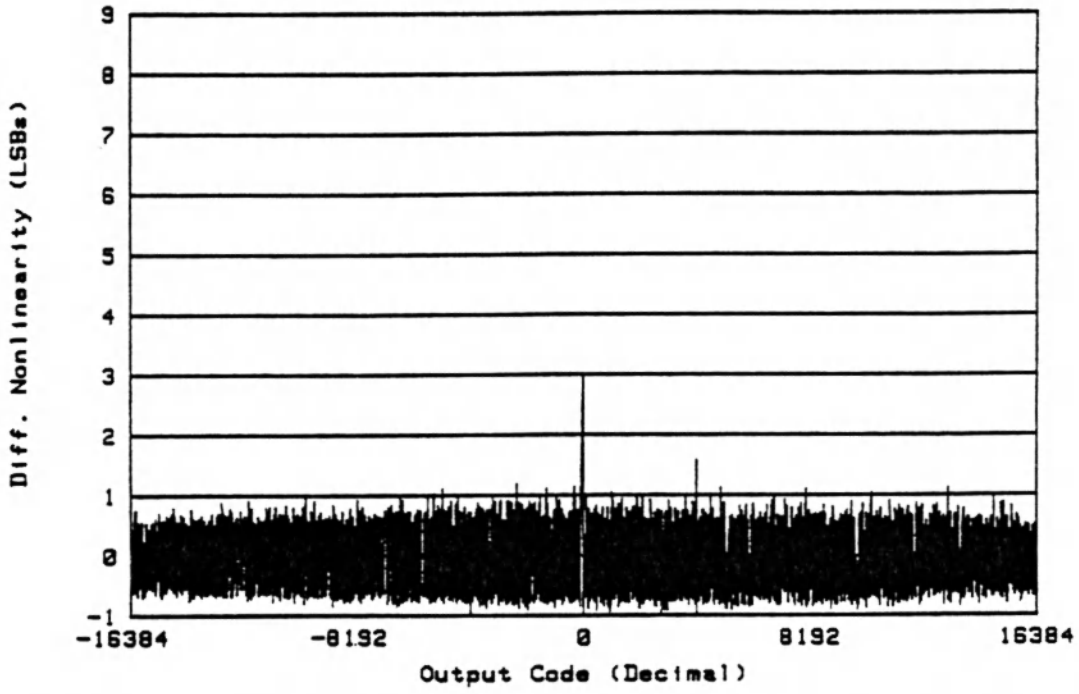


Figure 11a. Diff. Linearity Error as a Function of Output Code (DAC #1 used in ADC)

Source: Doerfler, "Techniques for Testing a 15-Bit Data Acquisition System", [4].

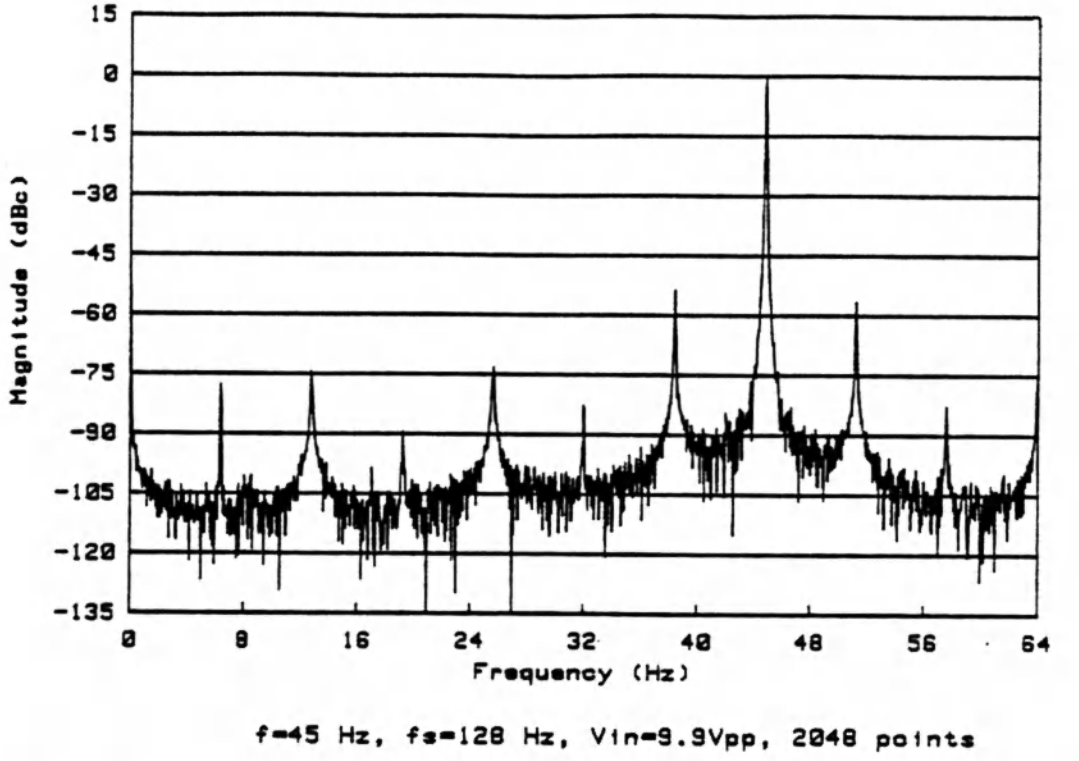


Figure 11b. Fourier Transform Test Results for a Bipolar Input

Source: Doerfler, "Techniques for Testing a 15-Bit Data Acquisition System", [4].

error at the 00...00 code is at least partially dependent on the DAC. Sources of the large 00...00 code error are examined in the next chapter.

Table 1

Performance Data for a Single Channel DAS Prototype that was Subjected to a Bipolar, Sine Wave Input

Performance Figure	DAC Used 1	2	3
Max. Diff. Linearity Error (LSBs)	3	3+	5+
# Missing Codes	5	44	32
Effective Linearity (# bits out of 15)	9.1	-	-

The previous data suggests that the DAS does not perform well when subjected to a bipolar signal. However, it performs much better when subjected to a unipolar signal. Figure 12 shows an FFT plot calculated from data taken from the DAS when the input of the DAS was subjected to a unipolar sine wave. The harmonic distortion is much lower than in the previous case. Other results are summarized in Table 2.

Table 2

Performance Data for a Single Channel DAS Prototype that was Subjected to a Unipolar, Sine Wave Input

Performance Figure	DAC Used 1	2	3
Max. Diff. Linearity Error (LSBs)	1+	2	2+
Effective Linearity (# bits out of 14)	13.3	-	-

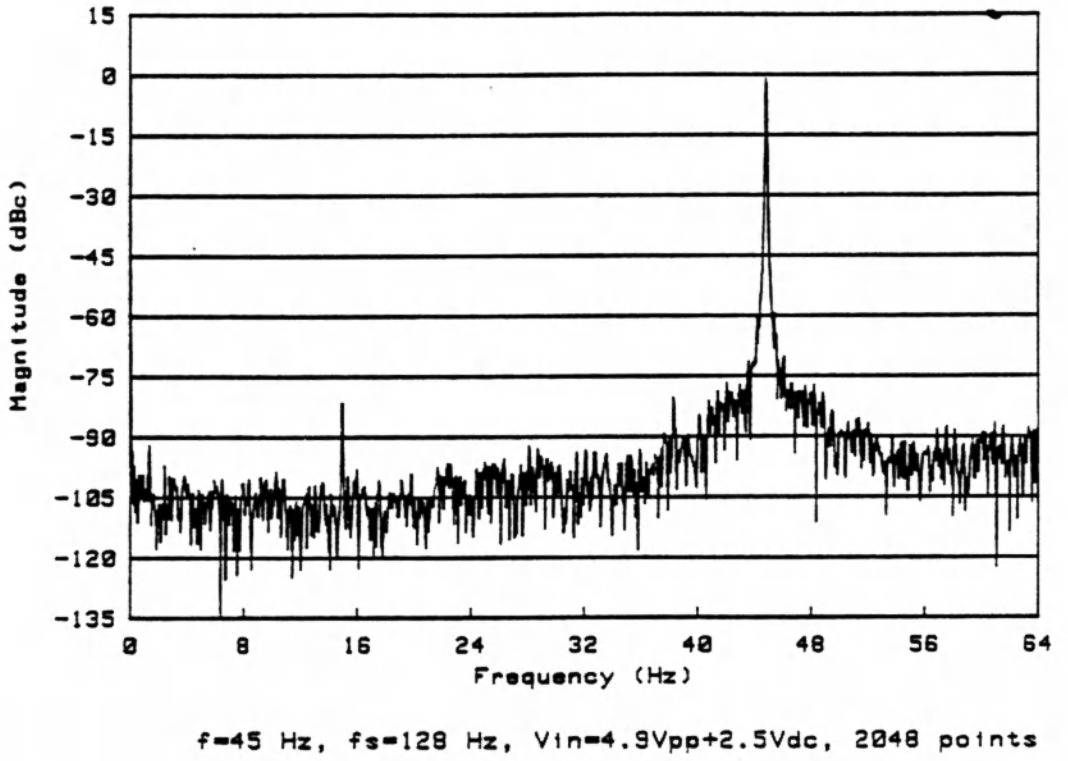


Figure 12. Fourier Transform Test Results for a Unipolar Input
Source: Doerfler, "Techniques for Testing a 15-Bit Data Acquisition System", [4].

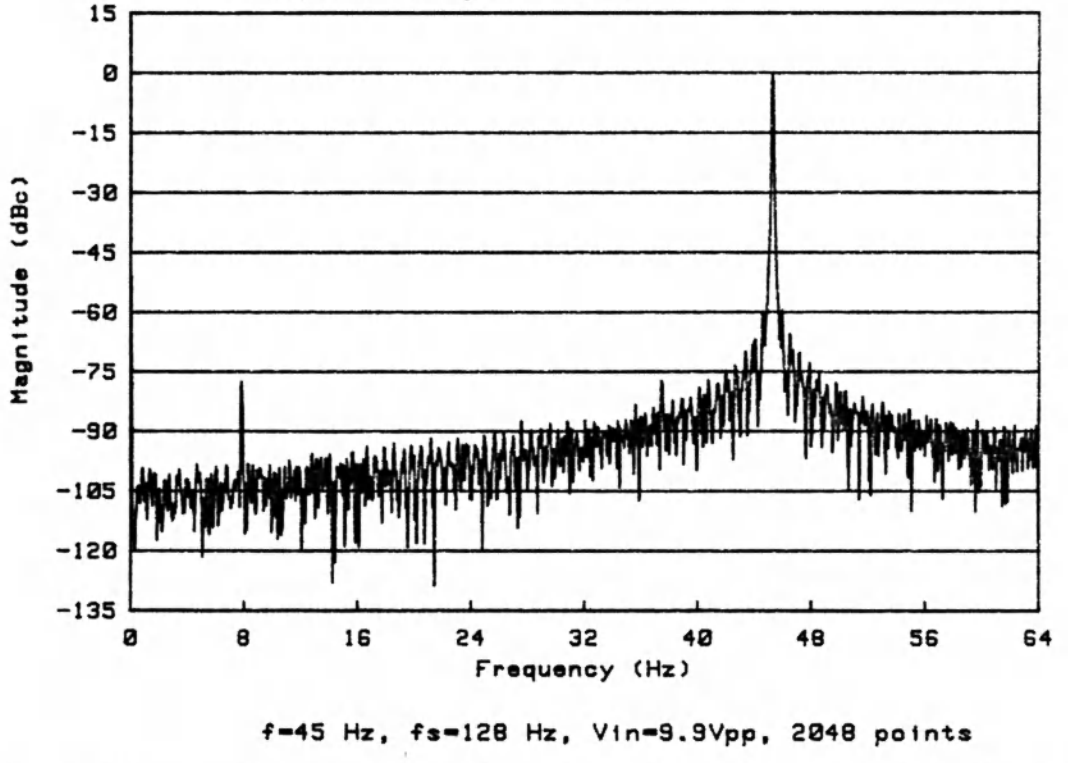


Figure 13. Fourier Transform Test Results for a Bipolar Input (Two channel PC board version)

Later testing of a PC board version of the DAS indicates that the harmonic distortion shown in Figure 11b is not directly related to the size of the differential linearity error at the 00...00 code. The FFT plot shown in Figure 13 was obtained from a PC board version of the DAS that has a differential linearity error at the 00...00 code of 8 LSB. The harmonic distortion shown in Figure 13 is much less apparent than the harmonic distortion shown in Figure 11b, and indicates that the effective linearity of the PC board DAS is about 13 bits. A possible explanation for the difference in performance between the prototype DAS and the PC board DAS is that the best straight lines drawn through the transfer characteristics of the negative and positive polarity ranges of each DAS are more colinear for the PC board DAS than for the prototype DAS. However, this theory has not been tested.

CHAPTER 2. INHERENT ERROR SOURCES IN THE LOW-POWER DAS

Introduction

Several inherent error sources in the low-power DAS which have not been formally considered before can now be discussed. The major source of error in this system is the mismatch between the polarity ranges. Other sources of error include track-and-hold droop, system noise, and comparator related problems. These error sources will be discussed in the following sections. Reference will be made to particular circuit components, so the reader should refer to Figure 6 while reading these sections.

Mismatch of Polarity Ranges

Preliminary tests performed by Doerfler showed that the existing DAS has a 3 - 5 LSB differential linearity error at the 00...00 code.[4] This error produces a mismatch of the positive and negative polarity ranges of the DAS, and can produce significant harmonic distortion. A mismatch of the polarity ranges will produce a transfer characteristic like the one shown in Figure 14.

There are several causes for the polarity range mismatch. The three major ones are 1) the DAC output has not been properly offset for use in a bipolar converter, 2) the DAC circuit's output voltage step between the 00...00 and 00...01 codes is greater than 1 LSB in width, and 3) the reference used for polarity determination depends on the polarity determined during

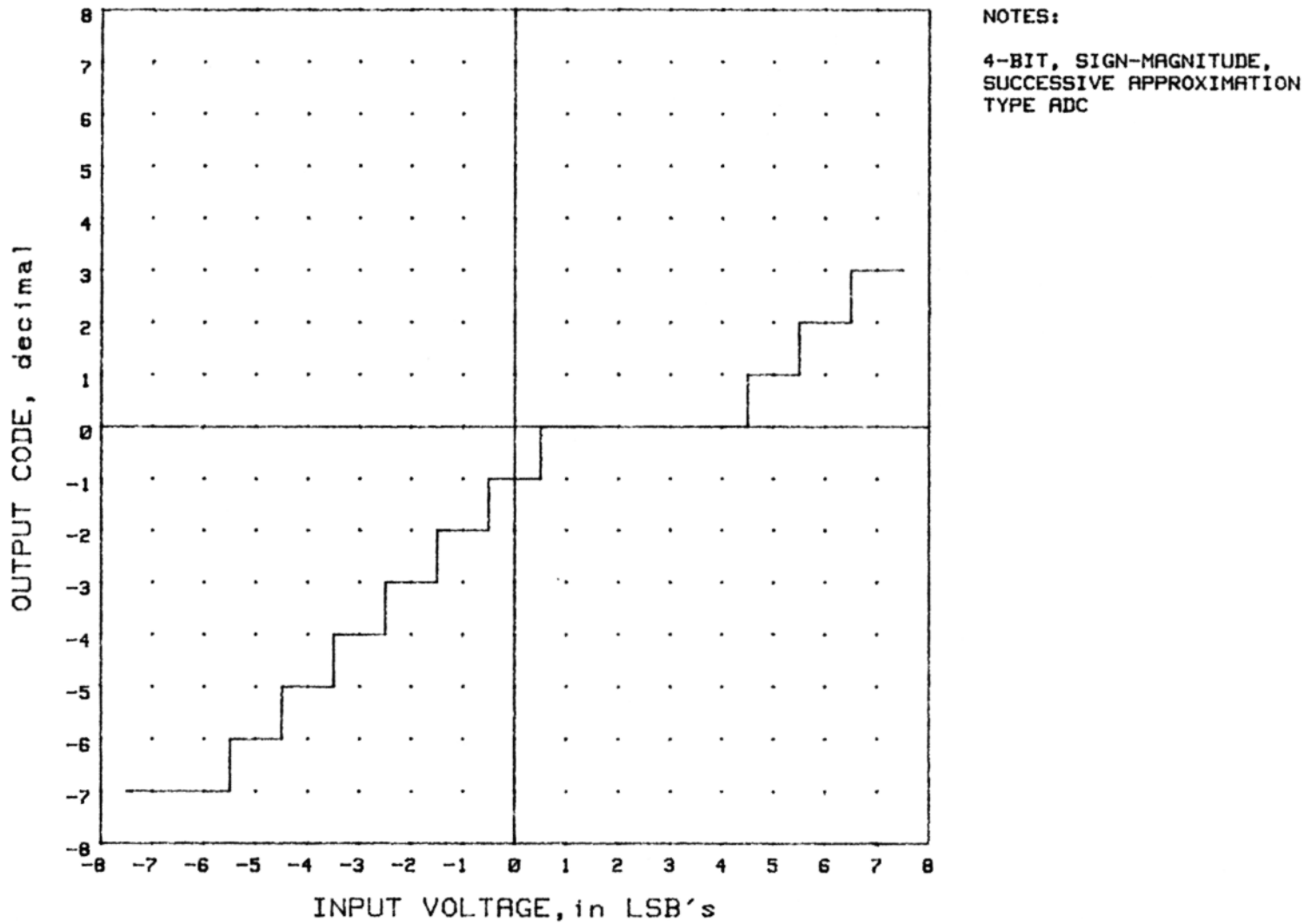


Figure 14. Transfer Characteristic of an ADC with Polarity Range Mismatch

the last conversion. These three causes will be considered here.

Improperly Offset DAC Output

For proper bipolar operation, the output of the DAC used in a sign-magnitude, successive approximation type ADC must be offset by 1/2 LSB. A correctly adjusted DAC will allow the ADC to produce the ideal transfer characteristic shown in Figure 15. The range of analog voltages represented by the 00...00 code is 1 LSB wide and centered on 0 V. When the DAC output is not offset, (which is the case with the low-power DAS), the ADC produces a transfer characteristic like the one shown in Figure 16. The range of analog voltages represented by the 00...00 code is now 2 LSB wide. The result is a built-in differential linearity error of 1 LSB for the 00...00 code. Accurately offsetting the DAC output is not easily accomplished in the DAC circuit used in the low-power DAS. Fortunately, another method of correction is available, and is discussed in the next chapter.

Unusually Large Voltage Step at DAC Circuit's Output

The previous cause of polarity range mismatch does not explain the 3 - 5 LSB differential linearity error at the 00...00 code, however. Another cause is that the voltage step at the output of the current-to-voltage circuit, which follows the DAC output current, is much greater than 1 LSB in width between the the DAC input codes of 00...00 and 00...01. This problem manifests itself in the following manner. The signal is, for example, a very small positive voltage. During polarity determination, the polarity is correctly determined to be

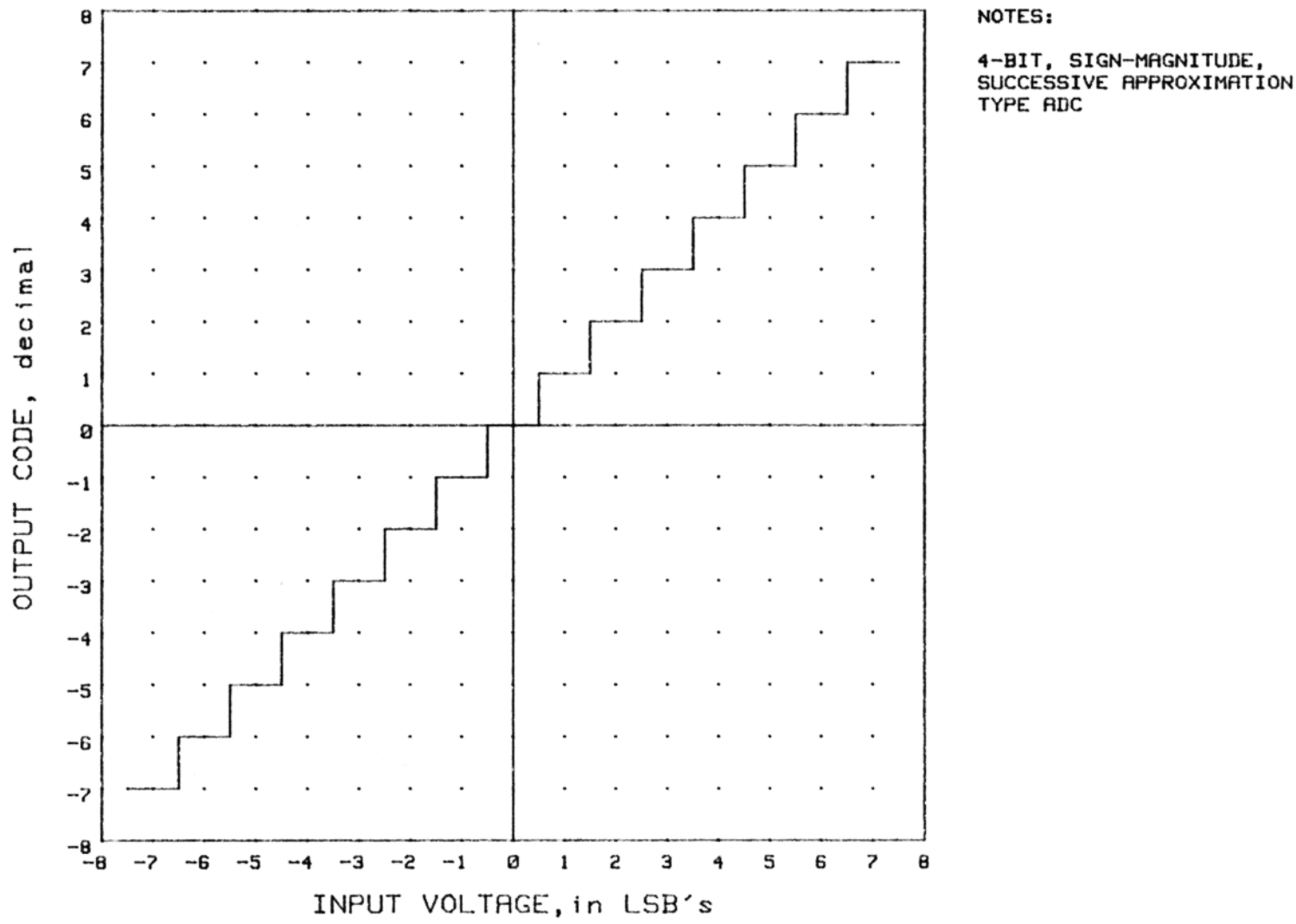


Figure 15. Transfer Characteristic of a Perfect ADC

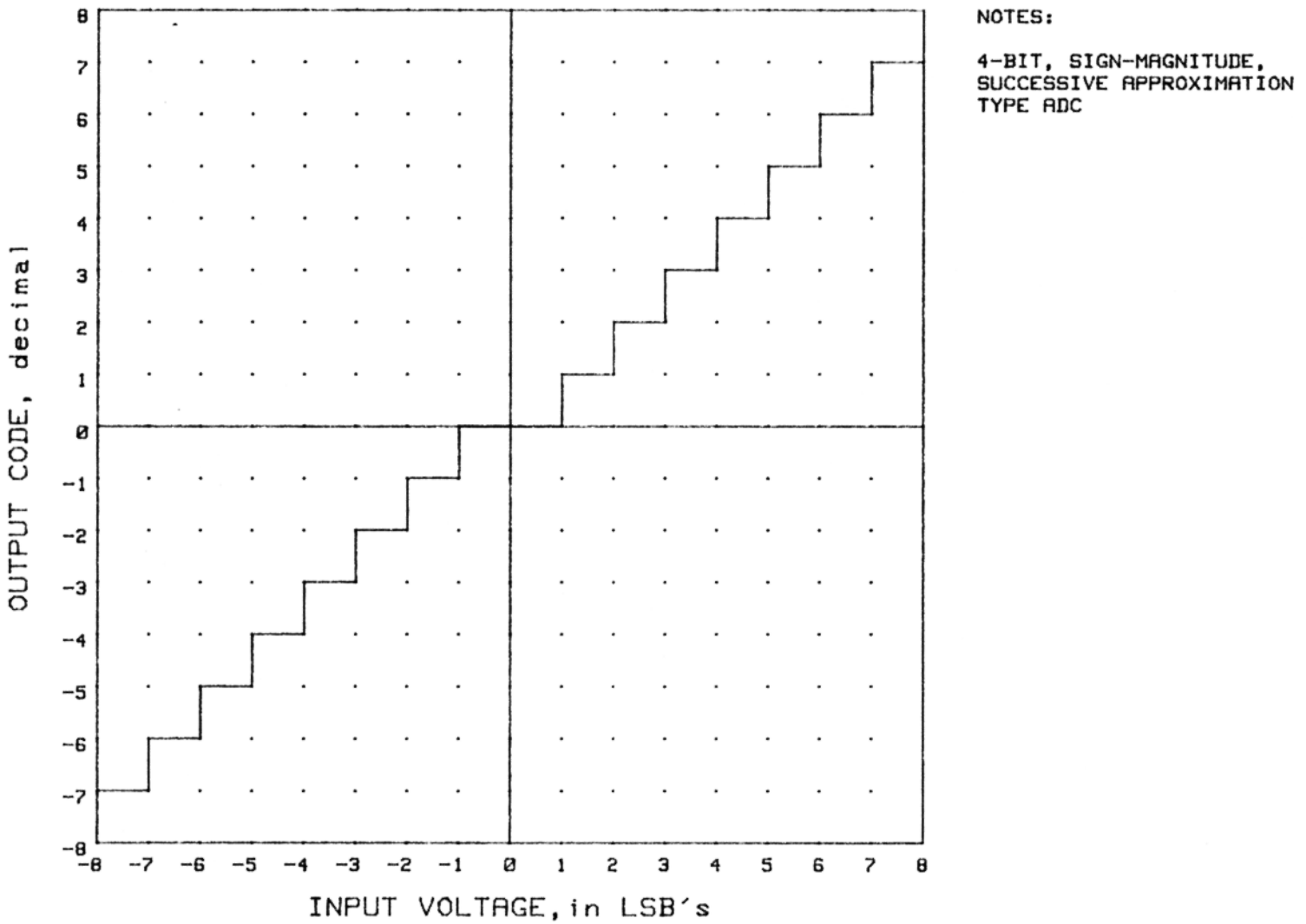


Figure 16. Transfer Characteristic of an ADC When its DAC Output is not Offset

positive. However, the signal is not sufficiently positive to result in any output code except 00...00. This occurs over an analog voltage range of several LSB, and results in a "dead zone" where no measurement can be made.

This problem has been shown to be dependent on the DAC, as DACs from different lots produce different "dead zone" widths. Also, this problem is aggravated by the input offset current of the op-amp U8. In the design of the ADC, nulling the effects of input offset current was accomplished by including the resistor R7 between the noninverting input of U8 and the analog common. The 10 kohm value for R7 was computed in the usual manner for the condition when the inverting input of U8 sees an open circuit (ideally) when looking into the DAC output. This condition exists when the DAC input code is 00...00, but for any other input code the inverting input of U8 sees a finite resistance which is dependent on the DAC input code. Therefore, the 10 kohm value for R7 is not strictly correct for any condition except when the DAC input code is 00...00. On the other hand, the computed value for R7 is probably the best compromise.

Since the DAC itself is at least partially responsible for the large differential linearity error at the 00...00 code, a partial solution to this cause of polarity range mismatch would be to individually test each DAC to ensure that its output current step between the DAC input codes 00...00 and 00...01 is as close to the ideal as possible. This cause of polarity range mismatch can also be compensated for by the microprocessor-based compensation technique that is presented in the next chapter.

Uncertain Reference for Polarity Determination

Another cause of polarity range mismatch is that the reference used to determine the polarity of the input signal depends on the polarity of the input signal during the last conversion. If the sampled input's polarity during the last conversion was negative, the output of op-amp U8 is used as the reference. If the sampled input's polarity during the last conversion was positive, the output of op-amp U9 is used as the reference. The uncertainty about which op-amp output is used as the reference creates a problem if there is a mismatch of the output offset voltages of the two OP-37 op-amps stemming from differences in their input offset voltages and currents. Since close matching of the outputs of U8 and U9 was strived for in the design of the DAS, the effect of this error source should be small. However, this problem is easily eliminated be resetting the flip-flop U11 at the beginning of each conversion. Another solution is to use the analog common as the polarity determination reference instead of the outputs of the op-amps U8 and U9. Circuit modifications which allow the analog common to be used as the polarity determination reference are presented in the next chapter.

Track-and-Hold Droop

Track-and-hold droop causes two problems. The first problem is that the sampled voltage being measured is changing during the successive approximation search. The second problem is that the sampled voltage held in the second channel is changing during the

conversion of the sampled voltage held in the first channel. The second problem arises from the fact that the two input channels are sampled simultaneously, but conversion of the two sampled voltages occurs in succession.

Droop During a Conversion

The droop rate for the track-and-hold was determined by Reed to be + 0.8 $\mu V/\mu s$.[2] The droop is caused by op-amp bias currents and switch leakage currents. During the conversion of a sampled voltage, approximately 27 microprocessor instruction cycles take place, with a period of 15 μs per instruction cycle. At the measured droop rate, the sampled voltage can change by 324 μV, or 1.06 LSB. This analysis assumes that the droop rate is constant over the entire range of input voltages; this is probably not a good assumption.

If the measured droop rate is accurate over the entire span of the DAS, then the first droop related problem appears to be a serious error because it causes uncertainty in the accuracy of the ADC's output code. A software simulation was written to investigate the effects of track-and-hold droop on the ADC's performance. The development of the simulation is shown in Appendix E. The simulation models the effects of the track-and-hold droop on an otherwise ideal bipolar 4-bit (3 bits plus sign) ADC over an input range of $\pm$ 5 V. The result of the simulation is shown in Figure 17. The plot shows the difference between the droop affected output code and the ideal output code, and indicates that the maximum error caused by the droop is 2 LSB.

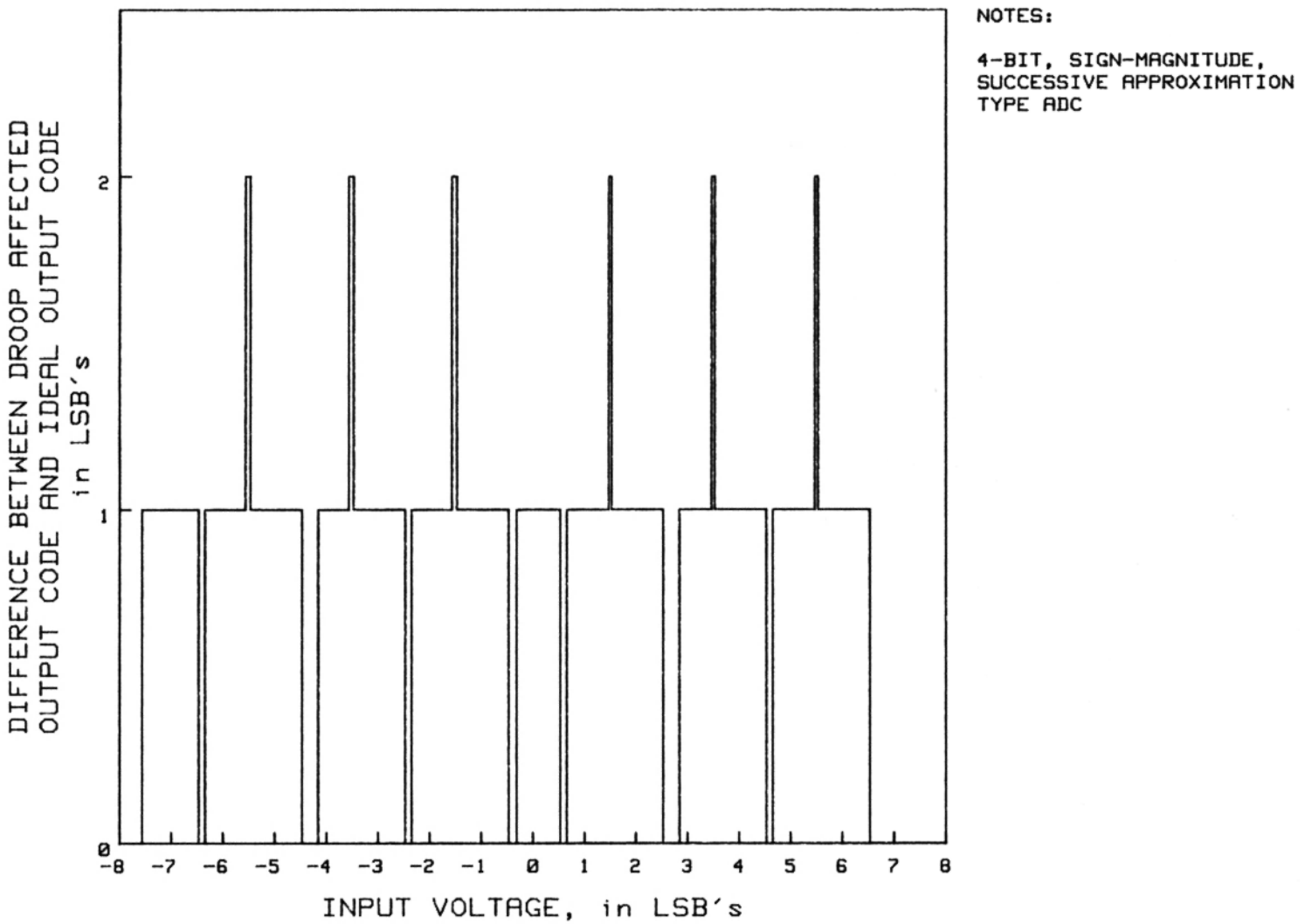


Figure 17. Results of Track-and-Hold Droop Error Simulation

Droop of Second Sampled Voltage During Conversion of First

Analysis of the second droop related problem is similar to the analysis of the first. During the conversion of the sampled voltage on channel 1 and the retrieval of the resulting data, approximately 52 microprocessor instruction cycles take place. At the measured droop rate, the sampled voltage on channel 2 would drift approximately 624 μV, or 2.05 LSB before conversion of the channel 2 voltage even begins. This results in an offset error between the channels. One solution would be to alter the channel 2 hold capacitor value to slow down the droop rate of the second channel relative to the first. However, care would have to be taken in the choice of the new capacitor value so that the signal acquisition performance of the second channel is not significantly degraded. Again another solution is available, and is presented in the next chapter.

System Noise

Since the voltage differences handled by the ADC are in the range of hundreds of microvolts, the noise that the analog section is subjected to should be minimized. This noise comes from the system environment, from the digital section, and from within the analog section itself.

Noise pick-up from the environment can be reduced by enclosing the analog section in a shielding container. This also reduces noise pick-up from the digital section. Other techniques for reducing noise pick-up include shielding of the power supply and signal lines to protect them from noise, shielding of the

clock line between the analog and digital sections to reduce noise output from this line, and ensuring that the entire system is properly grounded so that no "ground loops" are formed.

Steps should also be taken to reduce noise problems within the analog section itself. These include all the usual PC board tricks, such as use of ground planes, use of guard loops on component inputs, and physical separation of analog and digital components and traces. Special care should be given to the placement of the SAR clock trace to avoid coupling of the clock into any of the analog signals or supplies.

One particular improvement to the existing DAS design would be to place part of the power supply filter network on the component side of the power supply switches, as close to the components' power supply pins as possible. The reason for this is that at radio frequencies, (the SAR clock frequency), even a few inches of trace metal between the filters and the components can allow significant noise pick-up. Furthermore, at higher frequencies, the power supply rejection ratio (PSRR) of the components is low enough to allow a noise signal on the power supply pins to affect the components' performance. This is especially important for the comparator because it is a high gain device that is operated in an open loop configuration, and is therefore prone to noise induced error and oscillation. The comparator is discussed in more detail in the following section.

Comparator Related Problems

Proper functioning of the comparator is crucial to the performance of the ADC, since it is the central component in the determination of input signal polarity and in the successive approximation search. Since the comparator is a high gain device and is used in an open loop configuration, it is susceptible to unwanted feedback caused by parasitic effects. The solution to comparator related problems is to provide the comparator with the best circuit environment possible.[8 9]

Improper Power Supply Bypassing

The most important part of the comparator's circuit environment is proper power-supply bypassing. Ideally the comparator should be provided with a low impedance power source under all conditions. However, as the comparator switches, the changing internal characteristics of the comparator create rapidly changing demands for power supply current. These current demands appear on the power supply lines as high frequency signals. At these high frequencies the resistance and parasitic inductance of the power supply wires and traces become significant impedances. To the comparator, the power supply is no longer a low impedance power source. Since the PSRR of the comparator falls off at high frequencies, the resulting fluctuations of the power supply potentials have greater effect on the output of the comparator. A vicious circle develops between the fluctuations of the power supply potentials and the changing state of the comparator's internal circuitry, creating a sustained oscillatory condition.

The voltage signals generated on the power supply lines are also communicated to the power supply pins of other components, creating problems throughout the circuit. The solution to this problem is to place bypass capacitors between the comparator's power supply pins and the analog common. These capacitors act like small energy reservoirs that can provide a low impedance current source or sink for the comparator when one is needed. They also act as a short to ground for high frequency voltage noise on the power supply lines.

However, when bypass capacitors are used on several components, another problem is created. All the bypass capacitors are effectively paralleled and the combination of the capacitors and parasitic inductances can create a complicated, resonant circuit. The effects of the resonant circuit can be reduced by including a damping element in the power supply lines, such as a 10 ohm resistor. These resistors should be placed between a component's power supply pin and the power supply, on the power supply side of the bypass capacitor and as close to the component as possible. It should be noted that the resistor and bypass capacitor form a simple RC low-pass filter.

The existing low-power DAS has a power supply filtering network that performs the functions of the bypass capacitor and damping resistor. However, the filter network could be more effective if it were physically rearranged so that part of the filtering network was as close to the components as possible. In other words, a one or two pole low-pass filter could be placed at the edge of the board between the power supply and the power switches to reduce the amount of noise that is passed from the

power supply wires to the analog section. Then individual low-pass filters could be placed on the component side of the power switches, as close to the components as possible.

In the literature, the bypass circuit that is usually recommended is a 10 ohm resistor connected in series with the power supply line and a 10 μF tantalum capacitor in parallel with a 0.01 μF ceramic capacitor connected between the power supply pin and the analog common. The recommendation is based on experimentation which has shown this circuit to be adequate for most applications. The filtering network used in the low-power DAS is similar to the one mentioned above, and further experimental comparison of these two circuits would show which is better in this application.

Unwanted Feedback

Another common problem for high gain comparators is feedback from the output to the input through stray capacitance and inductance. This problem can result in net positive feedback or oscillation in the comparator circuit. The solution to the feedback problem is to keep the source impedances at the comparator inputs low and reduce the stray capacitance and inductance between the output and the inputs as much as possible. Source impedances should be kept low to reduce error voltages across the impedances. These error voltages are produced by currents induced on the inputs by stray inductances or by current spikes on the inputs that are caused by the switching of the comparator. The stray capacitance and inductance between the output and the inputs of the comparator can be reduced by routing the output

away from the inputs and by providing a ground plane under the comparator and its associated traces.

Input Noise

System noise that appears on the comparator inputs will create a "region of indecision" for the comparator. The region of indecision is a range of differential input voltages where the noise voltage amplitude is of the same order as the differential input voltage amplitude. In this region the noise will inhibit the comparator's ability to make a comparison of its input voltages, causing the output of the comparator to continuously wander between its rails in a random fashion. The effect on the ADC's performance is to obscure the ideally sharp transitions between the output codes in the ADC's transfer characteristic, as shown in Figure 18. In other words, there will be bands of analog input voltages where the ADC will not be able to precisely determine a digital representation, but will in successive conversion cycles produce several, (usually two), output codes for a given input voltage. Therefore, the system noise that the analog section is exposed to should be reduced as much as possible.

One technique used with comparators to increase their noise immunity is to introduce feedback to the noninverting input. This produces a hysteresis effect in the switching characteristic of the comparator, as shown in Figure 19. A comparator circuit which produces hysteresis is shown in Figure 20. A problem with this circuit, however, is that the placement of the hysteresis loop relative to the reference voltage is dependent on the

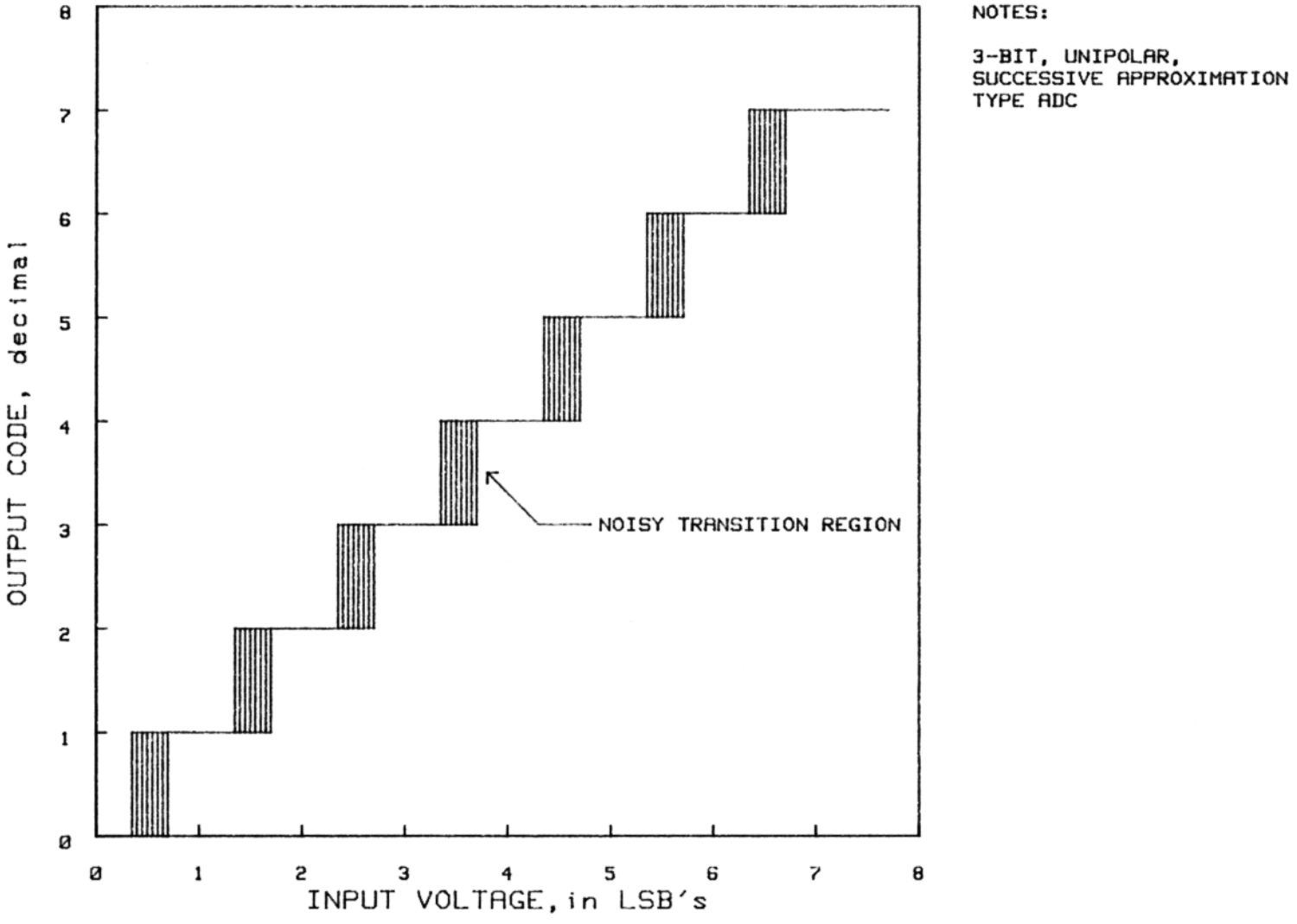


Figure 18. Transfer Characteristic of an ADC with Comparator Noise

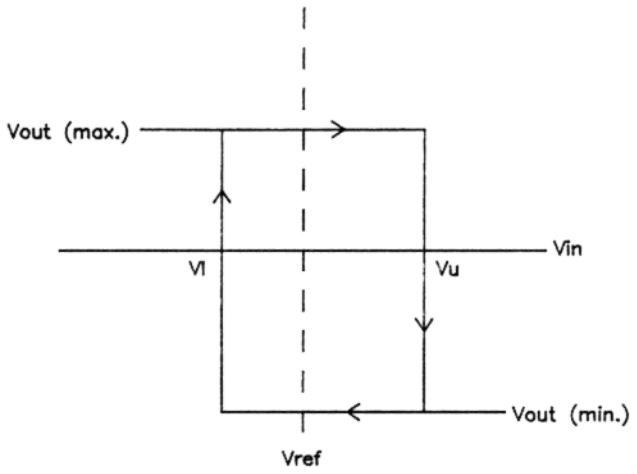


Figure 19. Switching Characteristic of a Comparator with Hysteresis

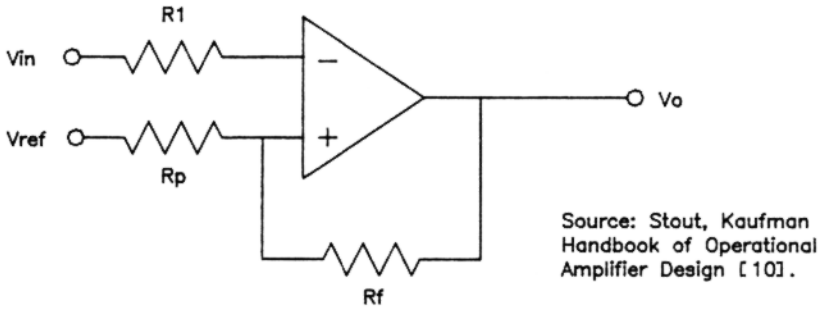


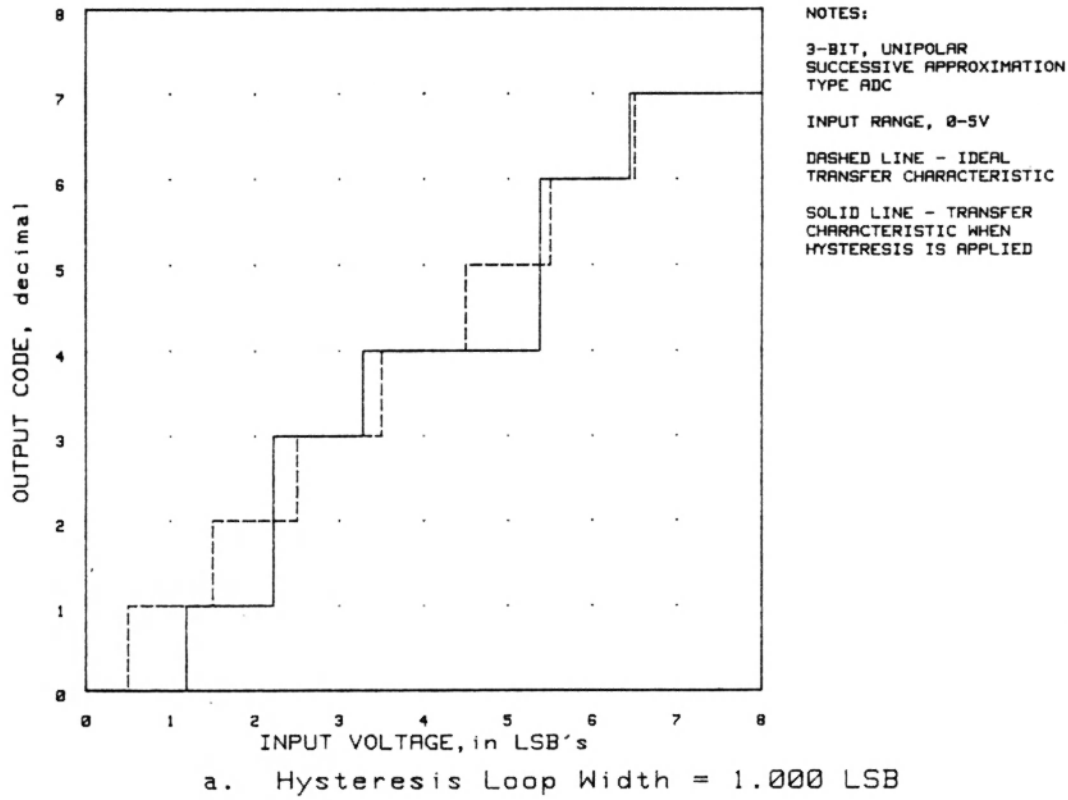
Figure 20. Circuit Diagram for a Comparator with Hysteresis

reference voltage. If used in a successive approximation ADC, this can cause differential linearity errors.

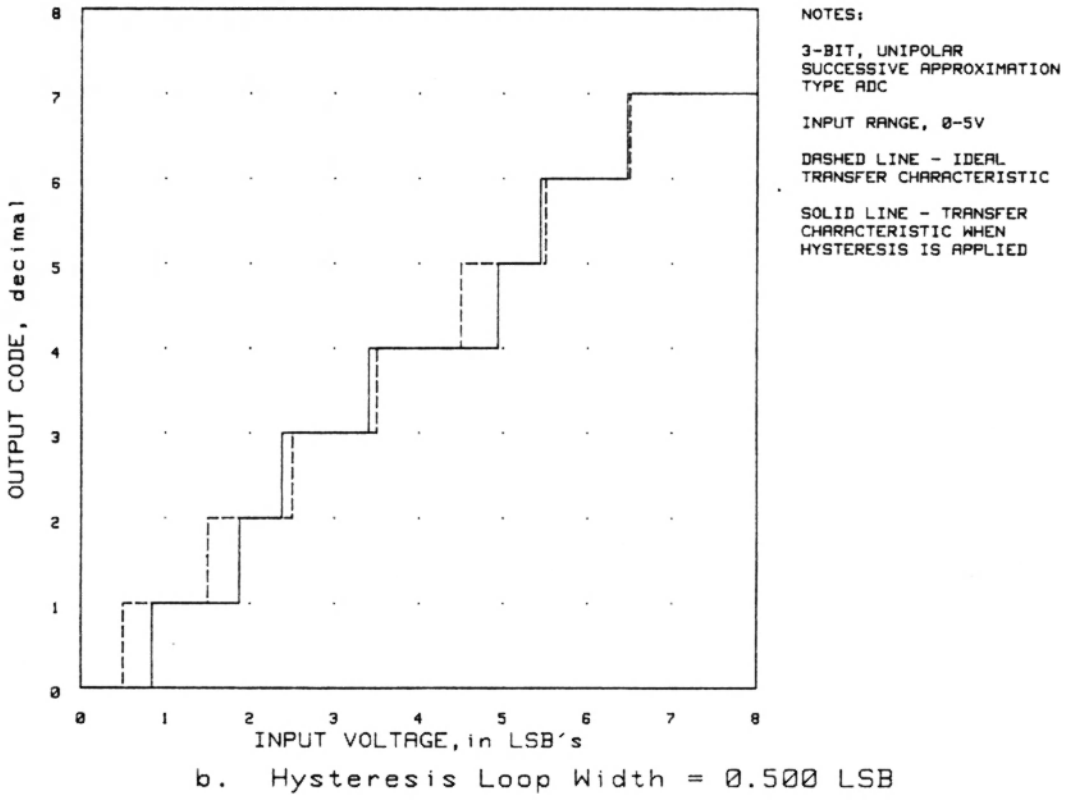
A software simulation was written to investigate the extent of the linearity error caused by comparator hysteresis in a successive approximation ADC. Development of the simulation is shown in Appendix F. The simulation models the effects of comparator hysteresis on an otherwise ideal 3-bit, unipolar successive approximation ADC with a 5 V span. The reference voltage shown in the comparator circuit, (Figure 19), is the output of the DAS's track-and-hold. From inspection of the results shown in Figures 21a - d, it appears that the hysteresis loop width would have to be less than 1/4 LSB so that the differential linearity error introduced by comparator hysteresis is relatively insignificant. For a 15-bit ADC with a 10 V span, 1/4 LSB corresponds to a voltage of 76.3 μV. To achieve a loop size this small, a feedback resistor value on the order of tens of megohms would be needed. Not only is a resistor this large not commonly available, but it could introduce significant thermal noise at the comparator input as well. Therefore, hysteresis is probably not a good solution for increasing the noise immunity of comparators used in high resolution successive approximation ADCs.

Input Offset Voltage

One other comparator related problem is the input offset voltage. The effect of the offset voltage is to create an offset error in the ADC's transfer characteristic. This is not a serious error and can be compensated for by the microprocessor-based error compensation technique presented in the next chapter.

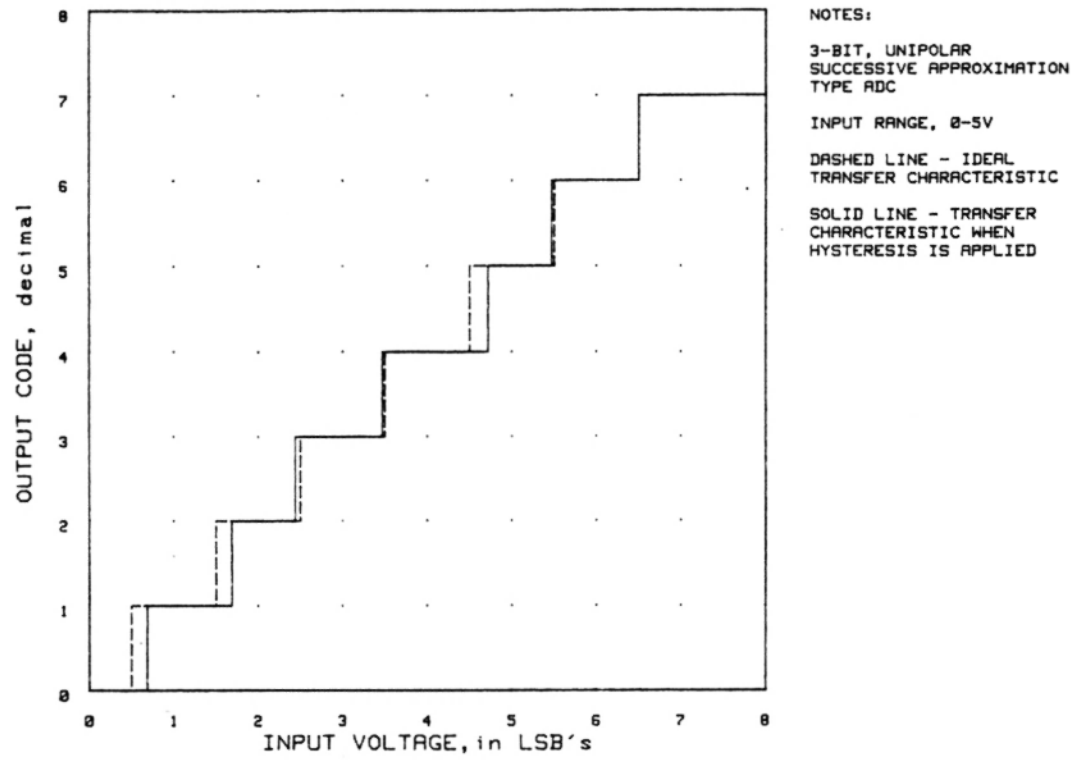


a. Hysteresis Loop Width = 1.000 LSB

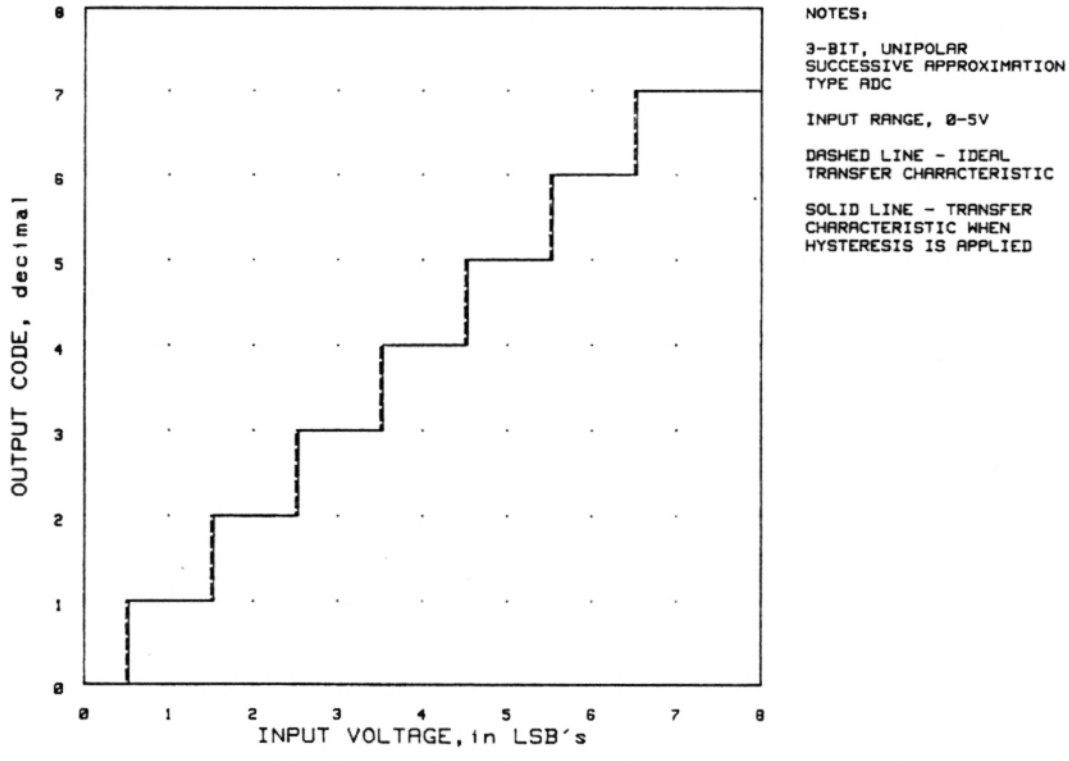


b. Hysteresis Loop Width = 0.500 LSB

Figure 21. Transfer Characteristic of an ADC with Comparator Hysteresis



c. Hysteresis Loop Width = 0.250 LSB



d. Hysteresis Loop Width = 0.000 LSB

Figure 21 (cont'd). Transfer Characteristic of an ADC with Comparator Hysteresis

CHAPTER 3. A MICROPROCESSOR-BASED ERROR COMPENSATION TECHNIQUE

Introduction

Since the DAS contains a microprocessor, it is desirable to use the microprocessor to correct for the DAS's errors. Several of the error sources discussed in the last chapter can be compensated for by the microprocessor-based technique that is discussed here. The errors that are compensated for are:

- polarity range mismatch
- offset between the track-and-hold's channels
- comparator input offset

A theoretical discussion of the compensation technique is given first. Next an implementation of the compensation technique is shown for the low-power DAS. Finally, some initial results obtained from static and dynamic tests of the error compensated DAS are reported.

Theory

If the positive and negative polarity ranges of the ADC overlap so that small analog input voltages can be measured in either range, as shown in Figure 22, then the overlap can be compensated for in the microprocessor's software. This is done by shifting all the positive range output codes down toward the ideal transfer characteristic and shifting all the negative range

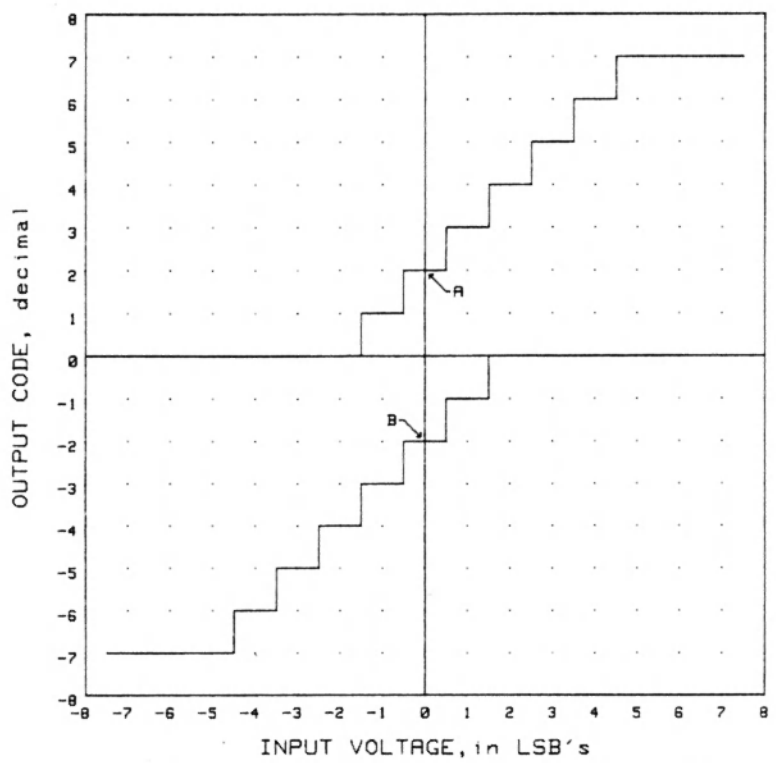


Figure 22. Transfer Characteristic of an ADC with Ranges Offset

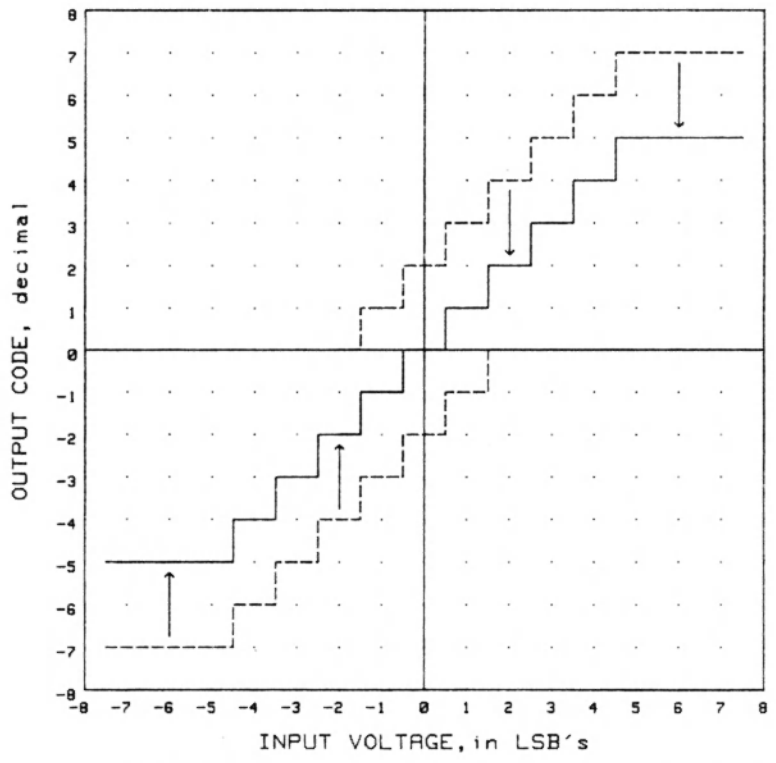


Figure 23. Transfer Characteristic of an ADC After Correction is Applied

output codes up toward the ideal transfer characteristic, as shown in Figure 23. The required amount of shift is determined by finding the output codes that correspond to an input voltage of 0 V for both the positive and negative polarity ranges, (points A and B in Figure 22). Ideally, only the part of the positive range that is used to measure input voltages greater than 0 V and the part of the negative range that is used to measure input voltages less than 0 V are retained; the rest of the output codes are thrown away so that the two parts of the ADC's corrected transfer characteristic meet at an input voltage of 0 V.

To determine the required amount of shift for both polarity ranges, the track-and-hold inputs are connected to the analog common and two analog-to-digital conversions are performed. The negative polarity range is used for one of the conversions and the positive polarity range is used for the other. The two resulting output codes become the correction codes for the negative and the positive polarity ranges. In subsequent conversion cycles on an input signal, the correction codes are used by the microprocessor to shift all the output codes toward their ideal values. If the output code that results from a conversion is a negative value, then the magnitude of the negative polarity range correction code is added to the output code. If the output code that results from a conversion is a positive value, then the magnitude of the positive polarity range correction code is subtracted from the output code. Since the low-power DAS is a two channel system, two sets of correction codes are needed (one set for each channel).

To achieve the overlapping transfer characteristic shown in Figure 22, offsets must be introduced in the ADC's hardware. The range of input voltages measurable by the negative polarity range of the ADC must be offset toward the positive direction; the range of input voltages measurable by the positive polarity range of the ADC must be offset toward the negative direction. The overlapping region should extend far enough on either side of 0 V to include the magnitudes of the offset voltage between the track-and-hold channels, the comparator's input offset voltage, and error voltages caused by aging of components. The overlapping region must include the analog common voltage, (0 V). The offsets would be fixed at the time that the DAS is manufactured, and would be set at a large enough value to meet the preceding criteria.

The accuracy of the induced offsets is somewhat important. The offsets should ideally be X + 1/2 LSB in magnitude, where X is an integer. Variation from the ideal offset value will result in a differential linearity error at the 00...00 code and an offset error in the ADC's overall transfer characteristic. The differential linearity error will be between - 1 and + 1 LSB. The offset error will be between - 1/2 and + 1/2 LSB. These errors are not overly significant when compared to errors in other parts of the transfer characteristic.

The uncertainty about whether the output of the current-to-voltage converter or the output of the inverting amplifier (op-amp circuits containing U8 and U9 in Figure 6) is used as the polarity determination reference voltage can be eliminated by using the analog common as the reference. The effect of the

comparator's input offset voltage on the accuracy of the polarity determination is then unimportant as long as the magnitude of the offset is less than the difference between the voltage at the edge of the overlapping region and the analog common voltage.

When the ADC's polarity ranges are forced to overlap and are then corrected for with the microprocessor, many of the low-power DAS's original errors are also compensated for. This hardware overlap/software correction scheme then forms the basis for a microprocessor-based error compensation technique for the DAS. The concept behind this error compensation technique involves the replacement of uncontrollable errors with an error that is controllable and correctable. For instance, the effect of the wide voltage step at the DAC circuit's output between the 00...00 and 00...01 DAC input codes is eliminated. This is because the DAC is no longer used to generate the polarity determination reference voltage (when the analog common is used for that purpose) and because the 00...00 and 00...01 codes are not used in the magnitude determination. The polarity ranges are forced to match by the software-based shifting of their transfer characteristics. Also, errors caused by the comparator's input offset voltage are nulled. Since two sets of correction values are used, (one for each channel), the offset error between the channels is nulled as well. Finally, noise induced errors in the outcome of the polarity determination for very small voltages become unimportant since both ranges can be used to measure those small voltages.

This compensation technique is not without cost. A few LSBs of span are lost at the extremes of each polarity range. The amount of span lost at the positive extreme is equal to the

magnitude of the positive range correction code. The amount of span lost at the negative extreme is equal to the magnitude of the negative range correction code. Another problem involves the effect of noise on the determination of the correction codes. If noise causes an error in the determination of the correction codes, then the output codes will not be shifted by the proper amounts. The result of this is that the transfer characteristics of the positive and negative polarity ranges will not match and an error at the 00...00 code will be created that could be as bad or worse than the original one. To minimize this possibility, the correction codes would have to be determined periodically. However, a trade-off exists between the power consumption of the DAS and the frequency of the determination of the correction codes. The trade-off exists because the determination of the correction codes must be accomplished during the extra time between normal conversion cycles; consequently the power-switched components of the DAS must be powered during that extra time. Another solution would be to use correction codes that are the result of a moving average of several previous correction codes. This solution is probably not desirable since it increases computation time and software complexity.

Implementation

Since board space is an important consideration for the DAS in the particular application that it was designed for, it is desirable to use as few extra components as possible in an implementation of an error compensation scheme. If possible,

some of the extra unused devices on the existing low-power DAS should be used. The extra devices include:

1. 3 EX-NOR (exclusive NOR) gates
2. 1 AND gate
3. 1 D type flip-flop
4. 1 SPDT switch
5. 3 comparators

The following sub-sections describe the circuit modifications and the theory of operation for an implementation of the previously discussed error compensation technique. In the following text, a logic high will be represented as a "1" and a logic low will be represented as a "0".

Circuit Modifications

An implementation of the hardware portion of the error compensation technique is shown in Figures 24, 25, and 26. Figure 24 shows a circuit that can be used to create the offsets in the ADC's polarity ranges. Figure 25 shows some modifications to the analog section's control logic and comparator circuit that will allow the microprocessor to find the correction codes and also will allow the analog common to be used as the polarity determination reference voltage. Figure 26 shows a complete circuit diagram of the analog section with all modifications in place.

The polarity range offsets are achieved by creating an offset voltage at the outputs of the op-amps U8 and U9. The offset voltage is created by introducing a small offset current

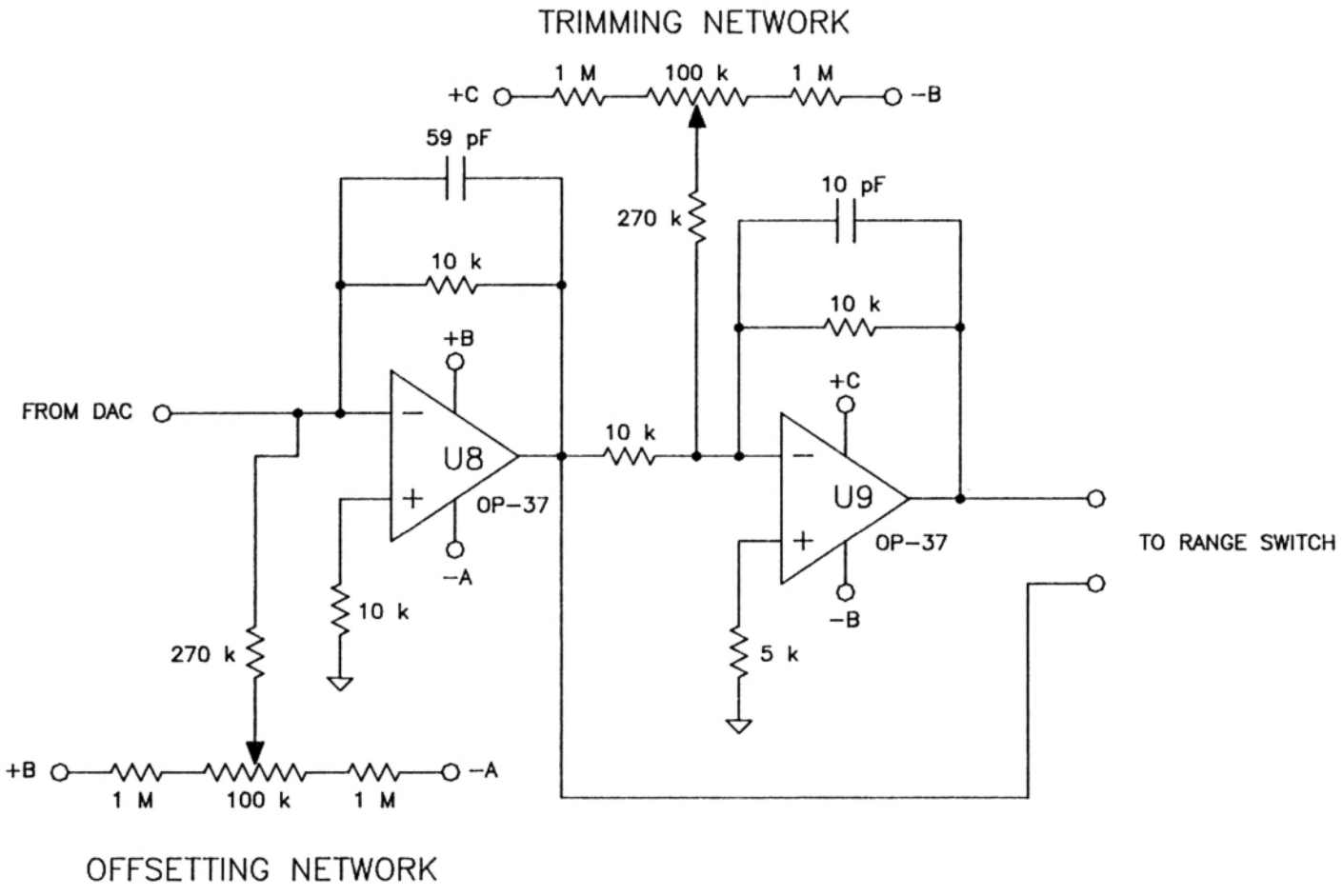


Figure 24. Resistor Divider Networks Used to Create Polarity Range Offsets

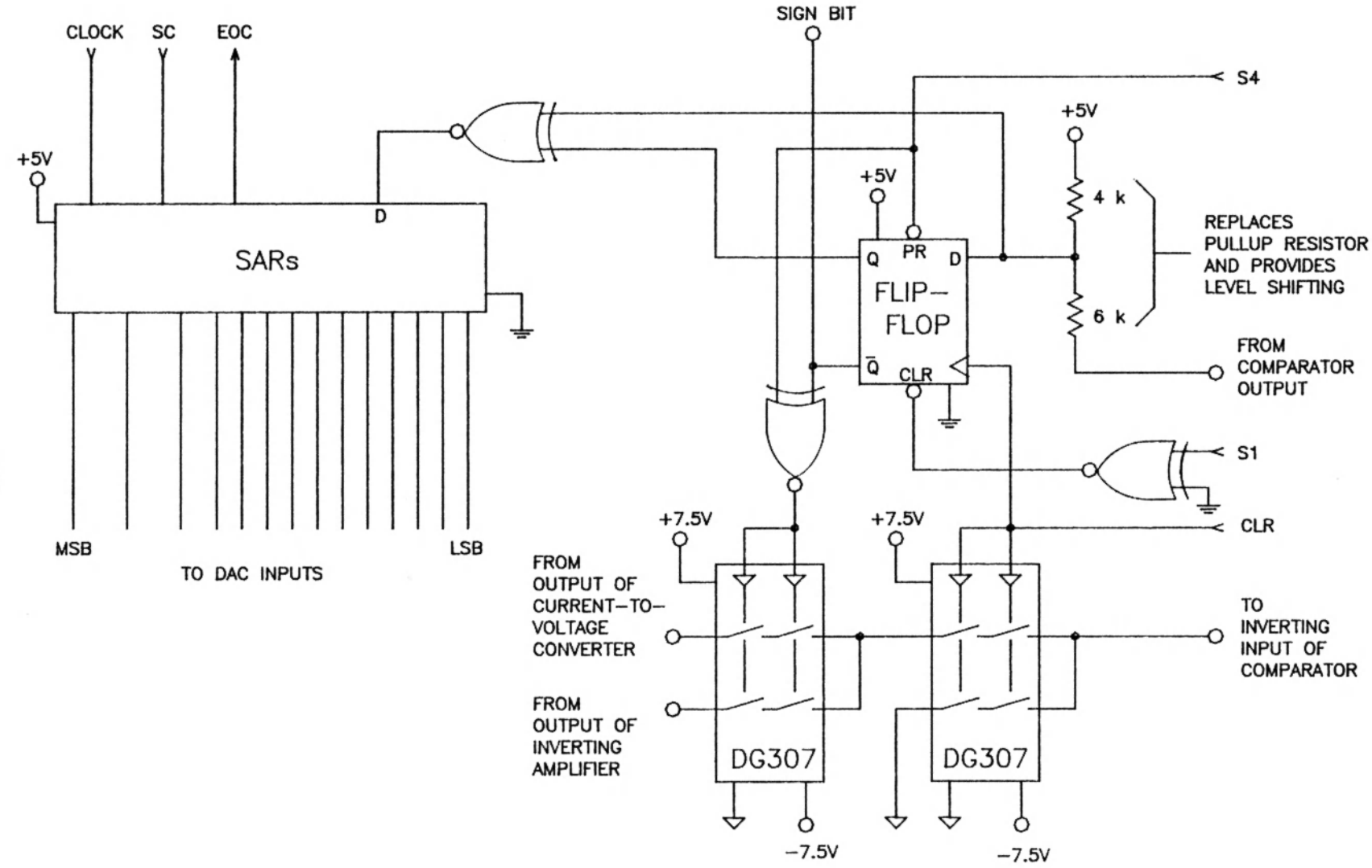


Figure 25. Modified Control Logic

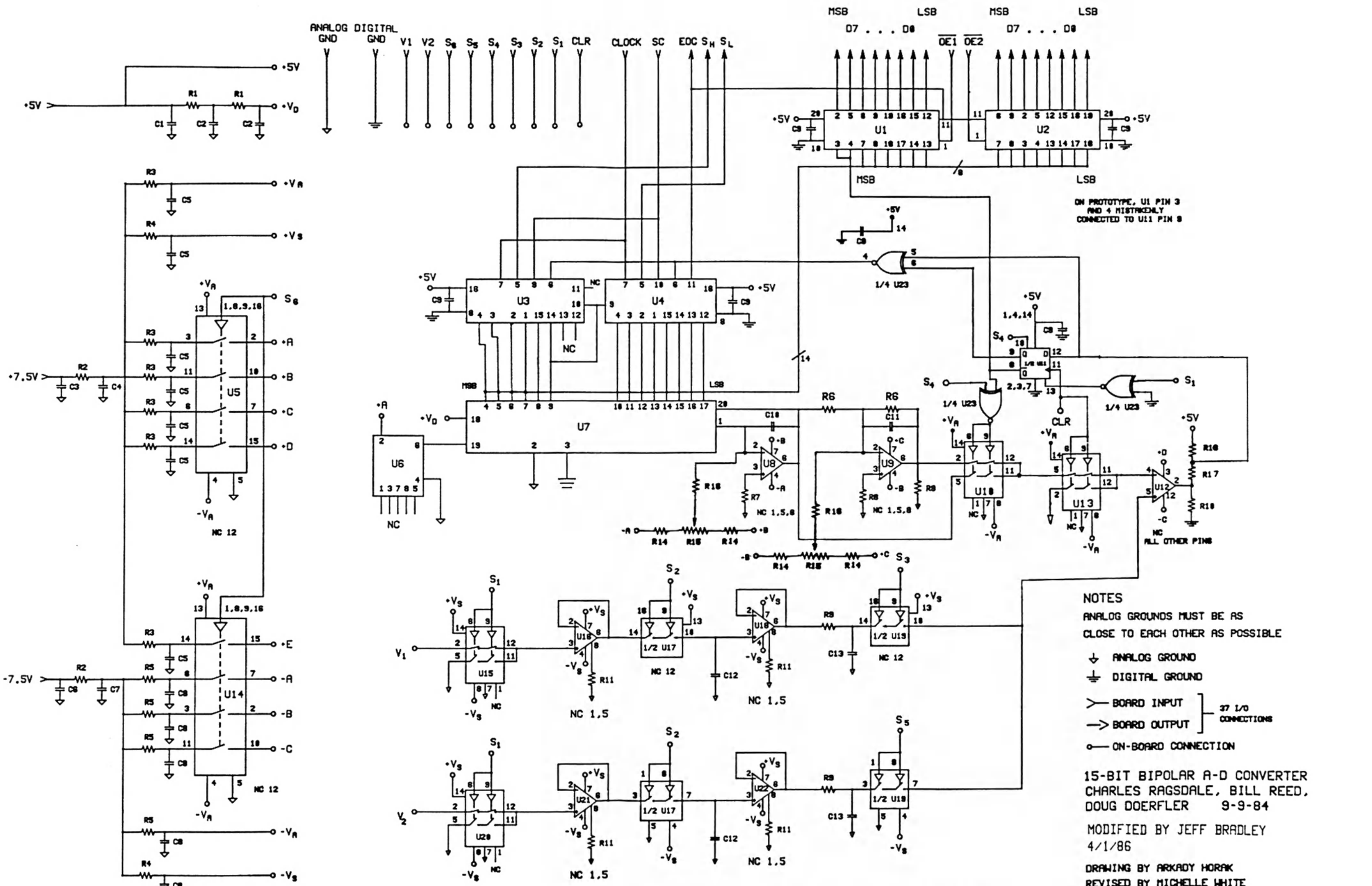


Figure 26. Circuit Diagram of the Modified Analog Section

at the inverting input of each op-amp with a resistor divider network. For the initial tests potentiometers were used to provide adjustibility. In the final product the divider network would be constructed with fixed value resistors. Since the output of the op-amp circuit containing U8 is inverted at the output of the op-amp circuit containing U9, the offset introduced in the circuit containing U8 is mirrored at the output of the circuit containing U9 and is fortunately in the correct direction. Therefore, the offset network used in conjunction with the circuit containing U8 creates the offsets for both ranges. The offset network used in conjunction with the circuit containing U9 is used to fine-tune the offset for the positive polarity range.

As shown in Figure 25, the polarity determination reference voltage is now the analog common instead of the virtual ground created by the DAC circuit at the output of either the op-amp U8 or the op-amp U9. The analog common is switched into the circuit using the SPDT analog switch U13. The original level shifting function of U13 is now performed by a resistor divider network. Switching control of U13 is performed by the CLR signal from the microprocessor. The AND gates (U24) are no longer needed since the DAC is not used to generate the reference voltage for the polarity determination.

To determine the correction codes, the control logic must allow the microprocessor to force the choice of polarity range. The circuit shown in Figure 25 gives the microprocessor this ability. The control signal S_4 is no longer used to switch the input of channel 2 of the track-and-hold to the analog common. It is instead used in combination with the control signal S_1 to

manipulate the outputs of the flip-flop U11. S_1 is also used to switch both of the inputs of the track-and-hold to the analog common. Two extra EX-NOR gates are used in the modification; one is used as an inverter and the other is used to combine S_4 and the flip-flop output $\bar{Q}$ into a control signal for the switch U10. The only component change required by this modification is the replacement of the 74C74 flip-flop with a 74HC74 flip-flop. This change is made so that the flip-flop output levels will respond properly when both S_1 and S_4 are set to "0".

Theory of Operation

A step-by-step example of the determination of the correction codes is given here. Assembly language software which performs the determination of the correction codes is included in Appendix D. A timing diagram for this example is shown in Figure 27.

The correction code cycle is similar to a normal conversion cycle, and is executed in the extra time between normal conversion cycles. Since the microprocessor is being run at a relatively slow rate, (the crystal oscillator frequency is 1 MHz), there is not enough time to determine all four correction codes in the extra time between normal conversion cycles. Therefore, the negative range correction codes must be determined after one normal conversion cycle and the positive range correction codes must be determined after the following normal conversion cycle. The negative range correction codes for both channels are obtained first. The control signals for the analog section are reinitialized as follows (see page 68):

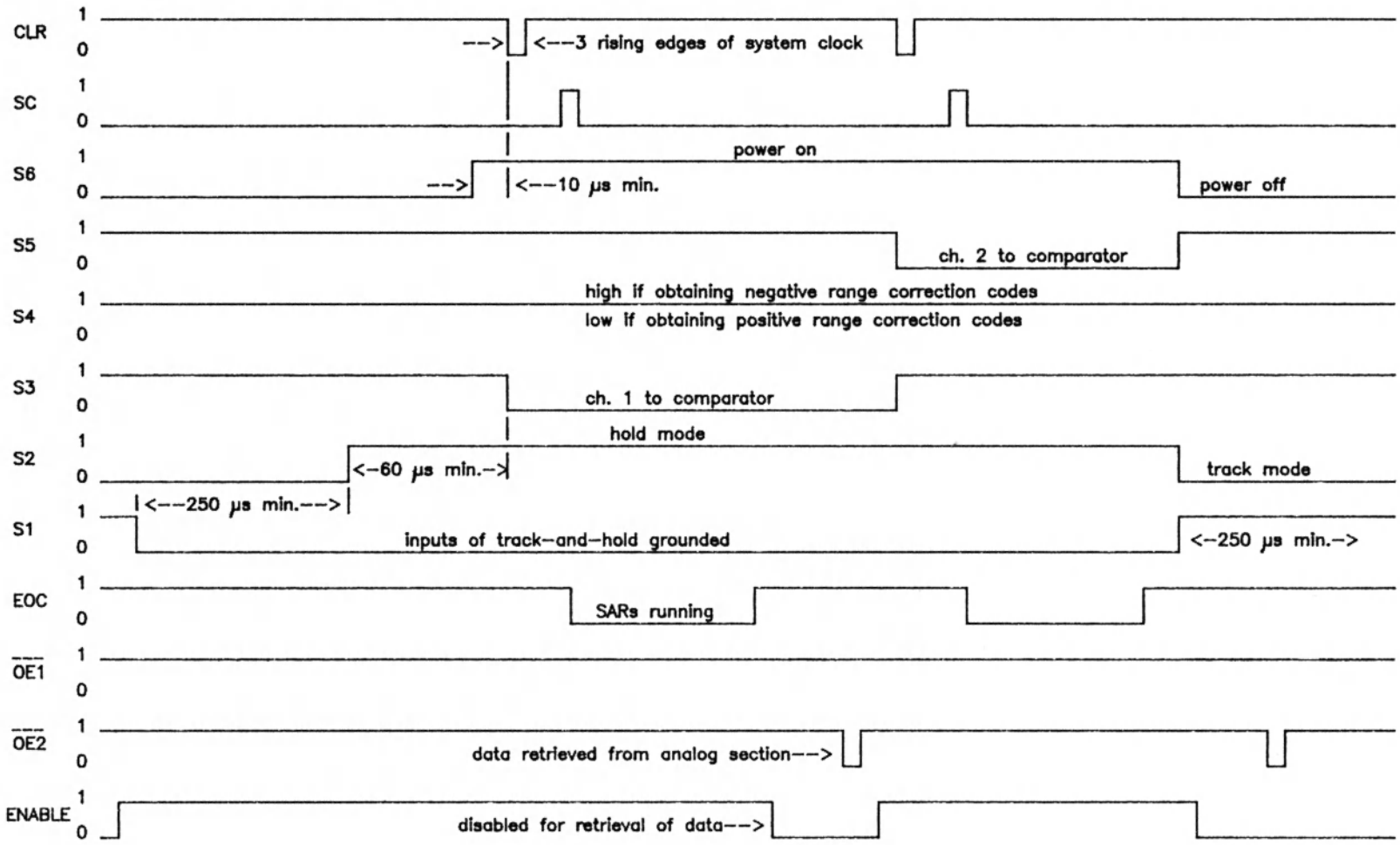


Figure 27. Timing Diagram for Obtaining Correction Codes

1. S_1 = "1", inputs to track-and-hold grounded
2. S_2 = "0", track-and-hold in track mode
3. S_3 = "1", channel 1 cut off from ADC
4. S_4 = "1", flip-flop U11 set for negative range
5. S_5 = "1", channel 2 cut off from ADC
6. S_6 = "0", power off
7. CLR = "1", normal state for the CLR line
8. SC = "0", SARs waiting for the start signal

Setting S_1 to "1" connects the inputs of the track-and-hold to the analog common and causes the "clear" input of the flip-flop U11 to be set to "0". Setting S_4 to "1" causes the "preset" input of the flip-flop U11 to be set to "1". The states of the "clear" and "preset" inputs cause the flip-flop outputs Q and $\bar{Q}$ to be set to "0" and "1" respectively. The EX-NOR combination of S_4 and the flip-flop output $\bar{Q}$ sets up the switch U10 so that the negative range of the ADC is chosen. The digital section's latch U6 is then enabled to allow the control signals to pass to the analog section.

Before the correction codes can be obtained, the track-and-hold must be allowed to stabilize after its inputs have been grounded. The measured slew rate of the track-and-hold is 0.02 V/μs.[2] Since the maximum voltage step that the track-and-hold is subjected to when its inputs are grounded is 5 V, the track-and-hold must be allowed to settle for (5 V) / (0.02 V/μs) = 250 μs. The correction codes can then be found.

The rest of the cycle is very similar to a normal conversion cycle, except that no polarity determination is required. S_2 is

set to "1" to put the track-and-hold in hold mode, and the track-and-hold is allowed to settle for a minimum of 60 μs. S_6 is then set to "1" to turn the power on for the power-switched components. The sampled voltage from channel 1 is then passed to the input of the comparator by setting S_3 to "0" and the 14-bit magnitude conversion is started by a pulse on the SC line. When the magnitude conversion is finished, channel 1 is cut off from the comparator and the digital section's latch U6 is disabled to allow data to be retrieved from the analog section's output latches. The negative range correction code for channel 1 is then retrieved and stored in one of the microprocessor's internal registers. After the data is stored, the digital section's latch U6 is reenabled and the process is repeated for channel 2. When the second conversion is complete, the digital section's latch U6 is disabled and the negative range correction code for channel 2 is retrieved and stored. After the two conversions are complete another 250 μs of delay time must be allowed for the track-and-hold to catch up with the input signals.

The positive range correction codes are obtained next in a similar manner. One difference is that S_4 is set to "0" to choose the positive range of the ADC. Setting S_4 to "0" sets the "preset" input of the flip-flop U11 to "0" and drives both of the outputs of the flip-flop to "1". The EX-NOR combination of S_4 and the flip-flop output $\bar{Q}$ sets up the switch U10 so that the positive range of the ADC is chosen. Another difference is that the positive range correction codes must be 2's complemented before they can be used. This is because the positive range correction codes are to be subtracted from the positive range

output codes, but the DAS's 80C39 microprocessor is not able to perform subtraction directly.

During normal conversion cycles, the correction codes are applied to the output codes after the output codes have been 2's complemented. If the output code is a positive value, the 2's complemented positive range correction code is added to the output code. If the output code is a negative value, the negative range correction code is added to the output code.

Since the offsets induced in the ADC are small, their magnitudes can be represented by 8-bit values. Therefore the software can be simplified if only the lower eight bits of the correction codes are retrieved from the analog section. When applying the correction codes to the output codes, 16-bit sign-extended versions of the 8-bit correction codes are used.

Initial Results

The result of an initial static test on the error compensated DAS is shown in Figure 28. The figure shows a plot of the transfer characteristic around the zero crossing point of the ADC when the error compensation technique discussed previously is applied. The result shown was obtained by subjecting the input of the DAS to one pass of a digitally controlled ramp. Therefore the plot only shows a rough estimate of the transfer characteristic. However, it is clear that the mismatch of the polarity ranges has been reduced considerably.

The results of an initial dynamic test on the error compensated DAS (PC board version) are shown in Figures 29b and 30b.

The results of a dynamic test on the unmodified DAS are also shown in Figures 29a and 30a for comparison. The dynamic tests performed on the unmodified and the error compensated DAS were the same as the ones performed on the prototype DAS, (results of which were reported in Chapter 1). A bipolar input signal was used.

Figure 29b shows a plot of the differential linearity error of the error compensated DAS. The differential linearity error at the 00...00 code is now acceptable. However, the differential linearity error at other points in the plot is not acceptable, especially in the negative range. Comparison of Figures 29a and 29b show that the differential linearity errors in the negative range are slightly worse for the error compensated DAS than for the unmodified DAS, and that the differential linearity errors in the positive range are slightly better for the error compensated DAS than for the unmodified DAS. This is a very puzzling occurance, and will require further investigation. Unfortunately, further debugging was not possible due to lack of time.

Figure 30b shows a plot of an FFT that was calculated from the test data. Harmonic distortion is very apparent, and is much worse than the distortion shown in Figure 30a for the unmodified version of the DAS. A possible reason for the extra distortion is that the modification circuitry adversely affects the performance of the DAC, the op-amps U8 and U9, or the comparator. Again, lack of time prevented further investigation.

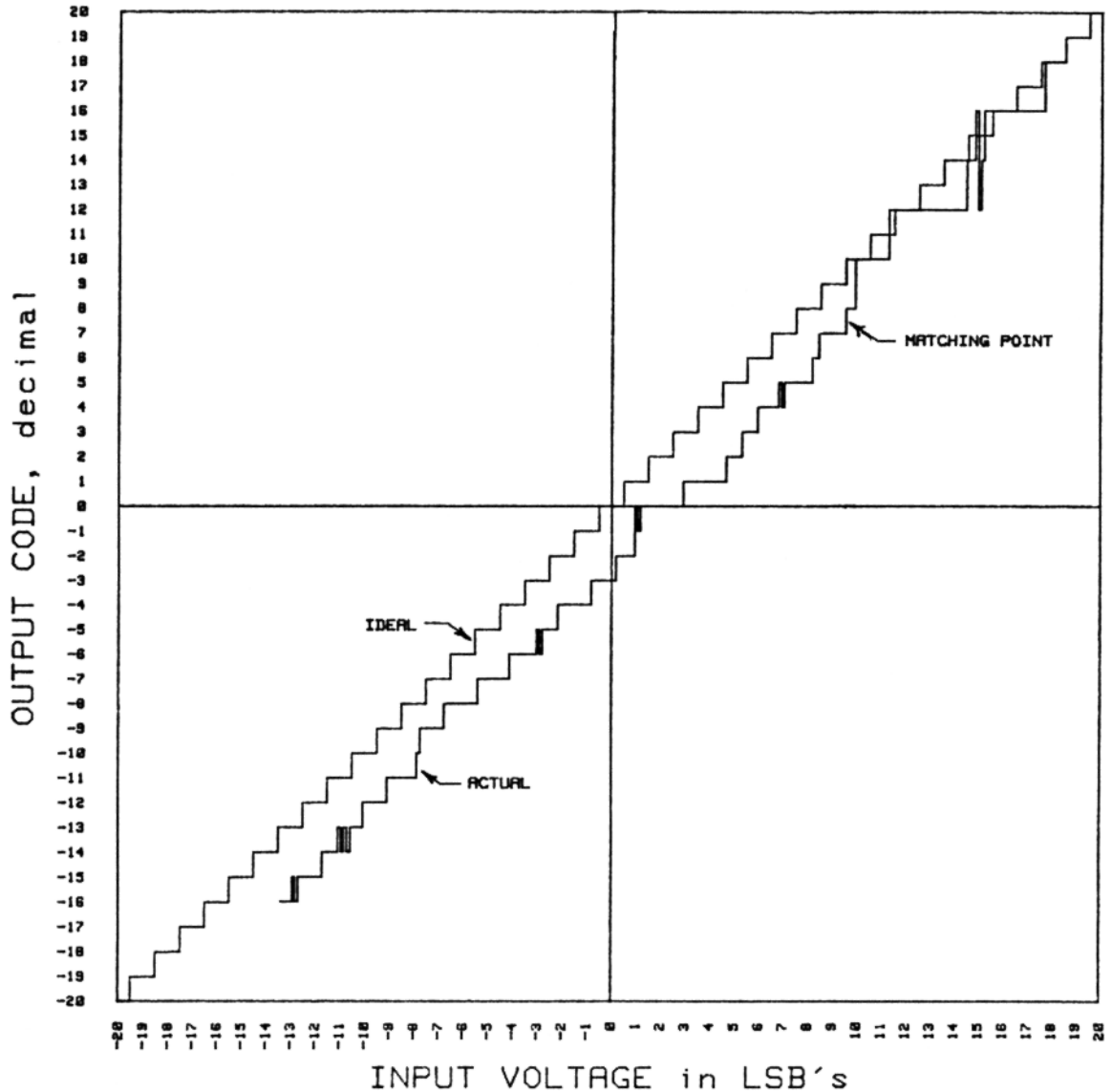
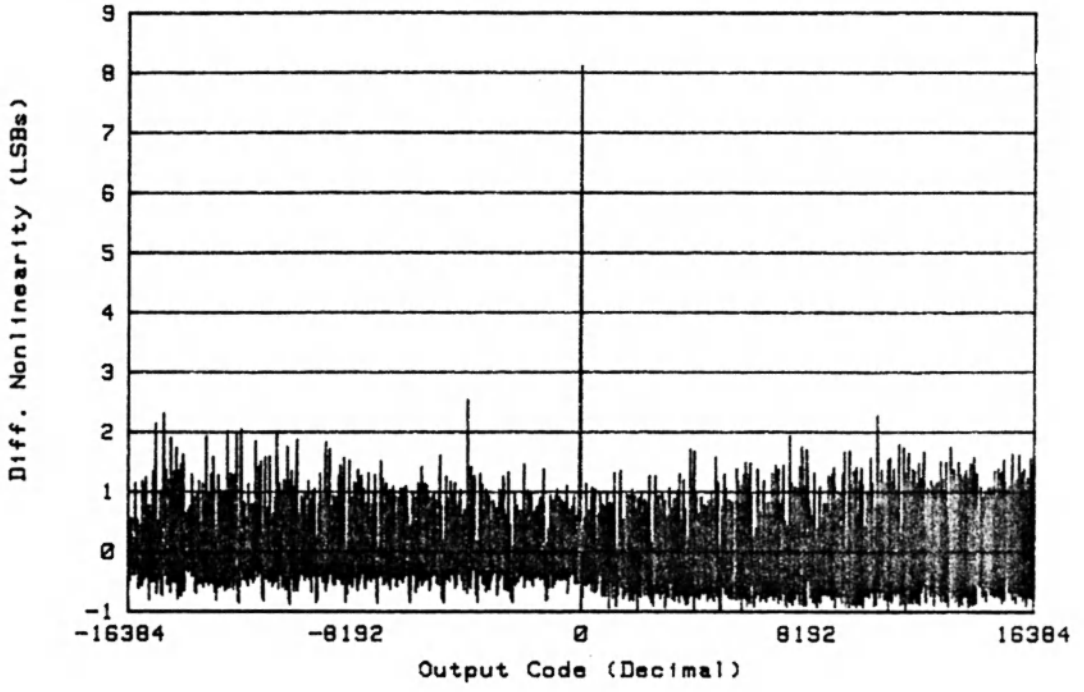
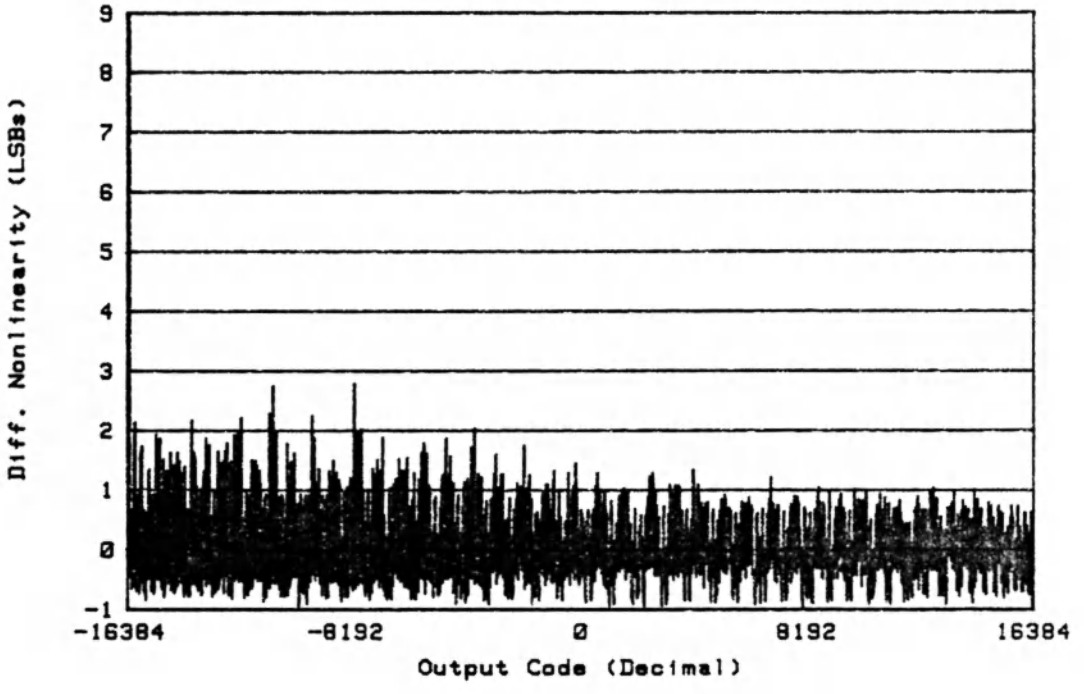


Figure 28. Rough Transfer Characteristic of the Error Compensated DAS Around the Zero Crossing

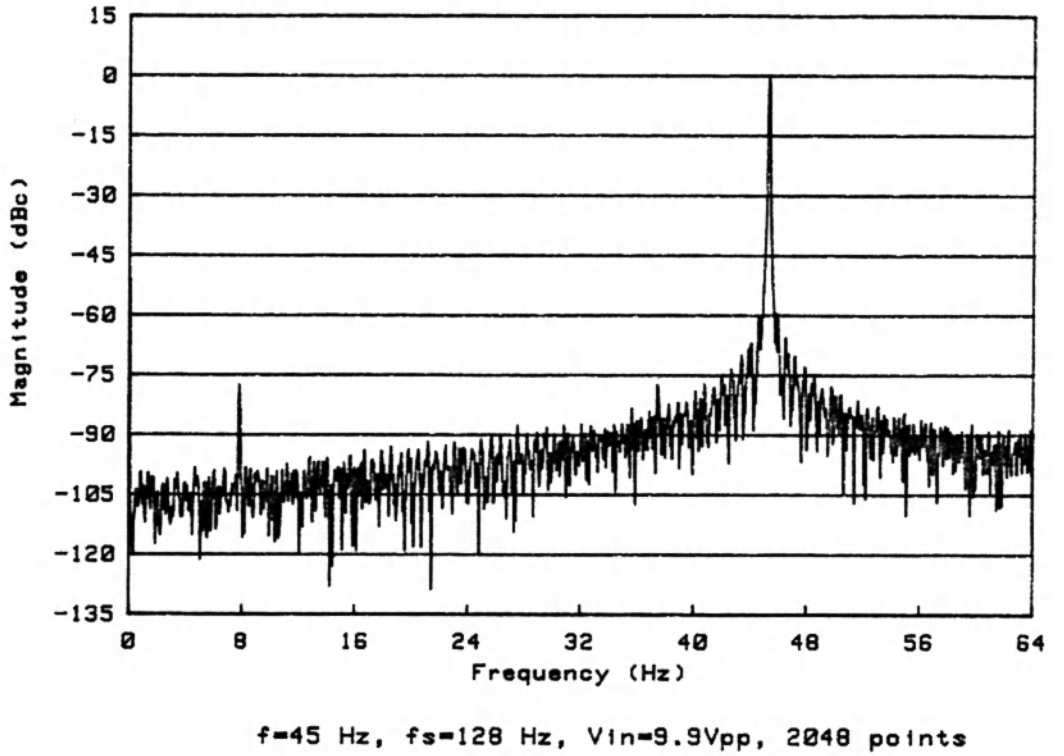


a. Unmodified DAS

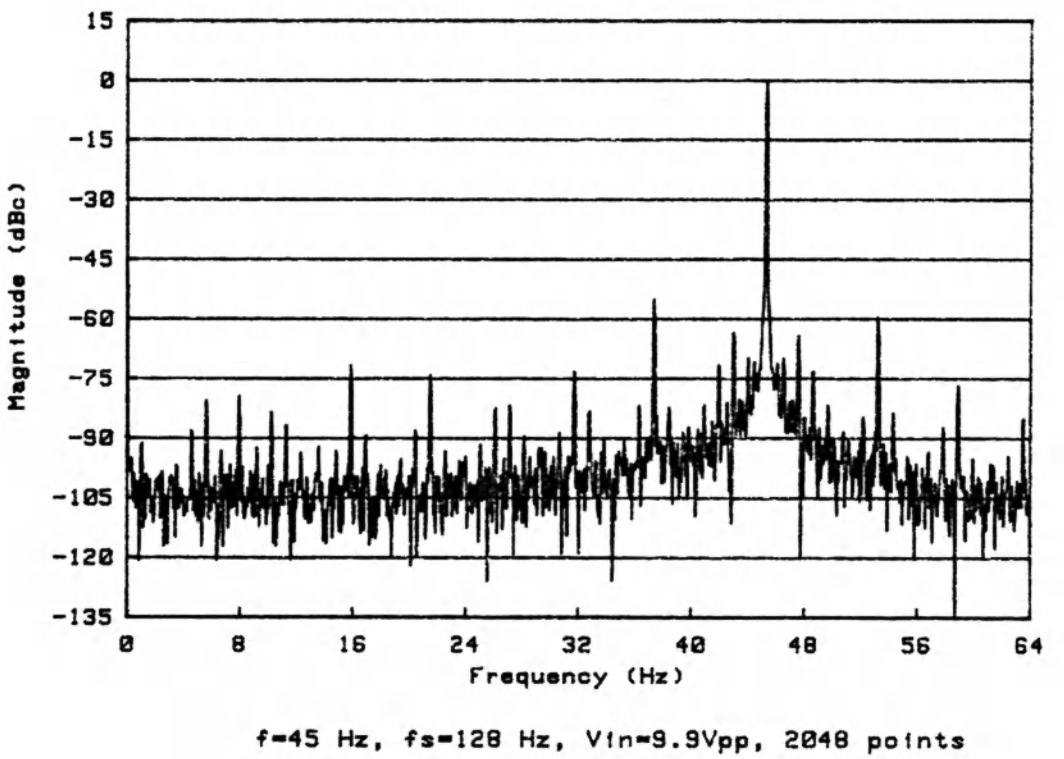


b. Error Compensated DAS

Figure 29. Diff. Linearity Error as a Function of Output Code for the Unmodified DAS and the Error Compensated DAS



a. Unmodified DAS



b. Error Compensated DAS

Figure 30. Fourier Transform Test Results for a Bipolar Input for the Unmodified DAS and the Error Compensated DAS

CHAPTER 4. RECOMMENDATIONS FOR FUTURE IMPLEMENTATIONS OF THE DAS

Introduction

Although the low-power DAS easily meets the requirement for power consumption that is listed at the beginning of Chapter 1, it does not meet the requirements for maximum differential and integral linearity error at this time. It is the author's opinion that these relatively stringent requirements will not be realizable with the present design. However, with further development the maximum differential and integral linearity errors could probably be limited to 1 LSB and 2 LSB respectively.

Several comments concerning future work on the low-power DAS are made in this chapter. There are two approaches that can be taken in future work: 1) continued development of the present design, realizing that some of the initial design requirements cannot quite be met, or 2) development of a "second generation" design using the same proven components. Each approach is briefly considered in the following text.

Continued Development of the Present Design

If the "continued development" approach is chosen, then there are several considerations that should be addressed. These considerations are outlined in the following sub-sections.

Quality of Components

Since the DAS is a high resolution system, the first and most important consideration is the quality of the analog section's DAC and analog components. Testing by both Doerfler and the author has shown that component quality is extremely critical to the individual DAS's performance, and to the consistency of performance from one DAS to the next. It is possible that each DAC and analog component would have to be individually tested to ensure that they meet their specifications, (an expensive and time consuming procedure). Substitution of cheaper, generic components for the specified components should not be attempted unless the cheaper part has been thoroughly tested and proven.

Differential Linearity Concerns

When the PC board version of the DAS was tested it was found that the differential linearity errors were slightly worse in the ADC's negative polarity range than in the positive polarity range. On the unmodified DAS this problem was not overly apparent, but when the error compensation hardware was added, the problem was aggravated enough to become significant. A possible cause for this problem is that the offset networks connected to the inputs of the op-amps U8 and U9 adversely affect the performance of the DAC or the op-amps. Another possibility is that the comparator is not performing as well with negative input voltages as with positive input voltages. This problem will require further investigation.

During testing of the DAS it was found that the differential linearity errors were reduced if the comparator and its pullup

resistor were not power-switched. At the 00...00 output code the reduction was slightly greater than 1 LSB; the average reduction was approximately 1/2 LSB. However, continuously powering the comparator and the pullup resistor will increase the power consumption of the DAS; the calculated increase is 29.2 mW. This calculation assumes that the comparator output is at - 7.5 V during the extra time between conversions, (worst case voltage across the pullup resistor), and that the pullup voltage is + 7.5 V. The power consumption increase nearly doubles the power consumption of the DAS. Since only a 1/2 LSB average reduction in the differential linearity error is achieved, continuously powering the comparator and its pullup resistor is probably not desirable. However, some reduction in the differential linearity error at the 00...00 output code could probably be achieved if more settling time for the comparator was allowed, both before and during the polarity determination.

Integral Linearity Concerns

The integral linearity of the prototype DAS was very bad (Figure 11b, Chapter 1); the integral linearity of the unmodified PC board version of the DAS was acceptable (Figure 13, Chapter 1). This is puzzling since these two versions of the DAS were almost identical, and since the differential linearity errors at the 00...00 code for the prototype DAS and the PC board DAS were 3 LSB and 8 LSB respectively. As mentioned before, a possible cause for the difference in performance is that the best straight lines drawn through the positive and negative polarity ranges are more colinear for the PC board DAS than for the prototype DAS.

If the compensation technique described in the last chapter is to be used, the poor integral linearity performance indicated by Figure 30 (Chapter 3) must be improved. The many harmonics introduced by the error compensated DAS could be devastating to the accuracy of any frequency domain signal processing performed on output data from the DAS. Possible causes would include software errors or adverse effects on the performance of the DAS's original components caused by the modification circuitry.

Crystal Oscillator Frequency

If the 80C39 microprocessor is required to perform additional processing, such as that required by the previously described compensation technique, then consideration should be given to increasing the crystal oscillator frequency to 1.5 MHz, (if a 1.5 MHz crystal can be found). This is suggested for the following reason. During the implementation of the software part of the error compensation technique, it was found that the extra processing took very nearly all of the available time between conversions. Very little time was left to send data to the host computer. Unfortunately, the handshaking requirements of the host computer required the 80C39 microprocessor to monitor the transfer of data to the host. This slowed down the transfer so much that data from only one channel could be sent to the host during the time available for data transfer.

When the oscillator frequency is increased, the calculated power consumption is actually decreased to 32.8 mW, (from 36.5 mW). The reason for the decrease is that the original 1MHz crystal does not quite optimize the power consumption for the

combination of the analog and digital sections; the 1 MHz value was chosen because it is widely available. Since the system clock frequency generated by the microprocessor is 1/15th of the oscillator frequency, the system clock period is decreased to 10 μs. The system clock can still be used by the ADC for the SAR clock since all components of the ADC were chosen to settle within 10 μs. One side benefit of increasing the oscillator frequency is that the effect of the track-and-hold's droop is decreased since a conversion will take less time to complete.

Other Considerations

Several other concerns which were discussed in Chapter 2 deserve mentioning here. The effects of the track-and-hold's droop should be verified and reduced if necessary. The analog section, signal lines, and power supply lines should be shielded. Consideration should also be given to the rearrangement of the power supply filtering network as discussed in Chapter 2.

Development of a "Second Generation" Design

The sign-magnitude configuration used for the successive approximation ADC is not the best configuration for a high resolution converter. The biggest problem with this configuration is that the ADC inherently posesses two ranges which must be closely matched. Making the ranges match is a complicated task for two reasons. The first reason is that the two ranges are generated by different circuitry. The second reason involves the ADC's internal DAC. The transition region between the DAC input codes

00...00 and 00...01 is a likely place for a healthy differential linearity error because the output impedance of the DAC changes from an ideally infinite value to a finite value. Since this transition region is used in both ranges, it occurs on both sides of (and adjacent to) the desired matching point of the two ranges. Also, since any error that is created in the ADC's transfer characteristic by the DAC's differential linearity error at the 00...00 input code occurs in the middle of the bipolar input range, it will more frequently affect the determination of a digital representation of the analog input.

In spite of the previously mentioned problems, it was necessary to use a sign-magnitude configuration for the ADC to achieve the required 15-bit resolution. This is because truly low-power DACs with resolutions of 15 bits or greater that can be run from $\pm$ 7.5 V supplies do not exist. The highest resolution obtainable in a low-power package was 14 bits; therefore, the 15th bit had to be obtained in a different manner. In the future, if low-power DACs with higher resolution become available, or if the requirements for ADC resolution or power supply voltages are relaxed, a different design approach could be taken. There are a myriad of other successive approximation ADC designs in the literature,[11] [12] and an extensive review of them is well beyond the scope of this thesis. However, three different alternatives are mentioned here. These alternative designs were chosen for their simplicity, their lack of multiple ranges, and their ability to use the components that have been proven for the present design. The three alternate designs are:

1. Unipolar successive approximation ADC
2. Bipolar successive approximation ADC, using a bipolar, offset binary DAC circuit
3. Bipolar successive approximation ADC, using a four quadrant multiplying DAC circuit

The original DAS design could be modified to accept any of these ADCs relatively easily. However, if the same 14-bit DAC is used, the resolution of the DAS would be reduced to 14 bits.

The unipolar ADC design is the simplest, but its use would require the bipolar input to be shifted into a unipolar range. It would also require a change in the power supply voltages and the reference voltage for the DAC. The ADC design that uses a bipolar, offset binary DAC circuit would require an additional voltage reference which produces a voltage equal to the negative of the DAC's reference voltage. The additional reference would be used to generate a current that would in turn be used to offset the DAC's output current. The ADC design that uses a four-quadrant multiplying DAC circuit would require no changes in the power supply voltages or the DAC reference voltage.

SUMMARY

Several implementation considerations for a low-power data acquisition system (DAS) have been qualitatively discussed. The low-power DAS discussed in this thesis will simultaneously sample two analog voltage signals (within a $\pm$ 5 V range) and perform a 15-bit analog-to-digital conversion on both sampled inputs. The analog-to-digital conversion is performed by a sign-magnitude, successive approximation type analog-to-digital converter. The sampling frequency is 128 Hz and the maximum input frequency is 45 Hz. The calculated power consumption is 36.5 mW.

The previously designed DAS was reviewed and its performance and major errors were reported in Chapter 1. The review included hardware descriptions of the DAS's analog and digital sections and a discussion of the operation of the system. The following major error sources were then discussed in Chapter 2:

- polarity range mismatch
- track-and-hold droop
- system noise
- comparator related problems

Next a microprocessor-based error compensation technique that corrects for the ADC's polarity range mismatch was presented in Chapter 3. Both the theoretical and practical aspects of the compensation technique were discussed. Initial test results for the error compensated DAS were also reported. The error compensation technique was found to reduce the ADC's polarity range

mismatch to an acceptable level. However, this improvement was not without cost; the power consumption was slightly increased, the input range was slightly reduced, and the system's software complexity was increased. There were also some problems with the implementation that will have to be solved before the compensation technique can be used. Finally, some recommendations for future development of the low-power DAS were given in Chapter 4. Two approaches to future development were briefly considered:

1) continued development of the present design

2) development of an alternate design, using the same proven components from the original DAS

Again, the purpose of this thesis was to present an overview of the low-power DAS and to analyze and suggest solutions to several error sources that are inherent in the design. It is hoped that this thesis has provided useful information and documentation to anyone interested in low-power data acquisition systems.

APPENDIX A

Circuit Diagrams and Parts Lists for the Existing DAS

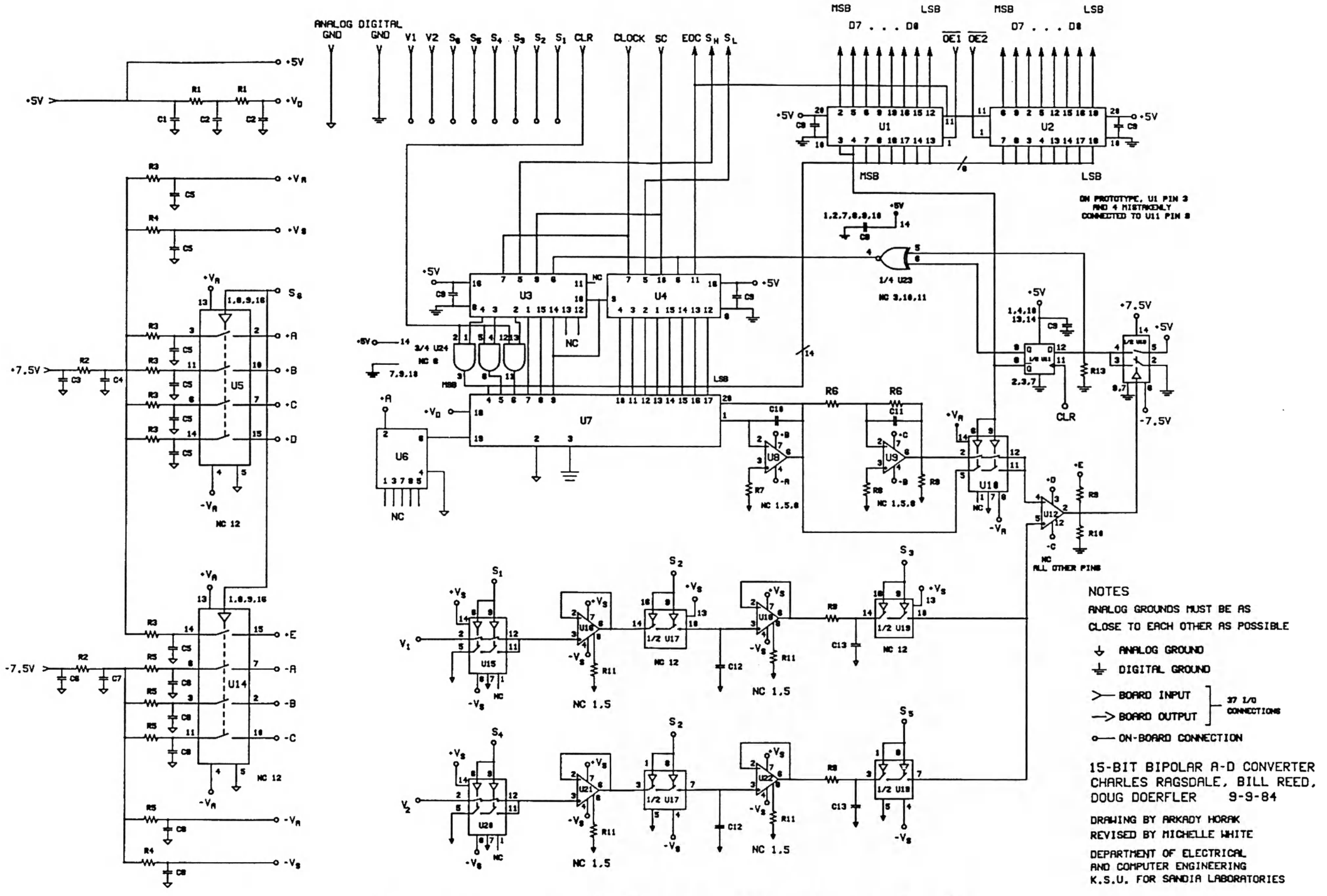


Figure A1. Circuit Diagram of the Analog Section

Parts List for the Analog Section of the Low-Power, 15-Bit Data Acquisition System

ICs

DEVICE	FUNCTION	PART#	MANUFACT.	#PINS
U1	Octal D Flip-Flop	MM74C374	Nat. Semi.	20
U2	Octal D Flip-Flop	MM74C374	Nat. Semi.	20
U3	Successive Approx. Reg.	MC14559B	Motorola	16
U4	Successive Approx. Reg.	MC14549B	Motorola	16
U5	Quad SPST Analog Switch	DG308CJ	Siliconix	16
U6	+5V Voltage Reference	REF-02EZ	PMI	8
U7	14-bit DAC	DAC-HA14B	Datel-Intersil	20
U8	Low-noise Op-Amp	OP-37EZ	PMI	8
U9	Low-noise Op-Amp	OP-37EZ	PMI	8
U10	Dual SPDT Analog Switch	DG307BP	Siliconix	14
U11	Dual D Flip-Flop	MM74C74	Nat. Semi.	14
U12	Quad Comparator	CMP-04FP	PMI	14
U13	Dual SPDT Analog Switch	DG307BP	Siliconix	14
U14	Quad SPST Analog Switch	DG308CJ	Siliconix	16
U16	Micropower Op-Amp	OP-22EZ	PMI	8
U17	Quad SPST Analog Switch	DG309CJ	Siliconix	16
U18	Micropower Op-Amp	OP-22EZ	PMI	8
U19	Quad SPST Analog Switch	DG308CJ	Siliconix	16
U20	Dual SPDT Analog Switch	DG307BP	Siliconix	14
U21	Micropower Op-Amp	OP-22EZ	PMI	8
U22	Micropower Op-Amp	OP-22EZ	PMI	8
U23	Quad 2 I/P EX-NOR	MM74HC266	Nat. Semi.	14
U24	Quad 2 I/P AND	MM74C08	Nat. Semi.	14

Resistors

DEVICE	FUNCTION	PART#	MANUFACT.	# USED	
R1	10 ohm discrete resistor	*		2	
R2	1 ohm discrete resistor	*		2	
R3	22 ohm resistor network	108A	Allen-Bradley	2	$
R4	100 ohm discrete resistor	*		2	
R5	22 ohm discrete resistor	106A	Allen-Bradley	2	$
R6	10 kohm resistor network matched to 0.005%	300190	Vishay	1	
R7	10 kohm 1% discr. resist.	*		1	
R8	5 kohm 1% discr. resist.	*		1	
R9	10 kohm resistor network	108B	Allen-Bradley	2	#
R10	1 Mohm discrete resistor	*		1	
R11	2 Mohm discrete resistor	*		4	
R13	10 kohm discrete resistor	*		1	

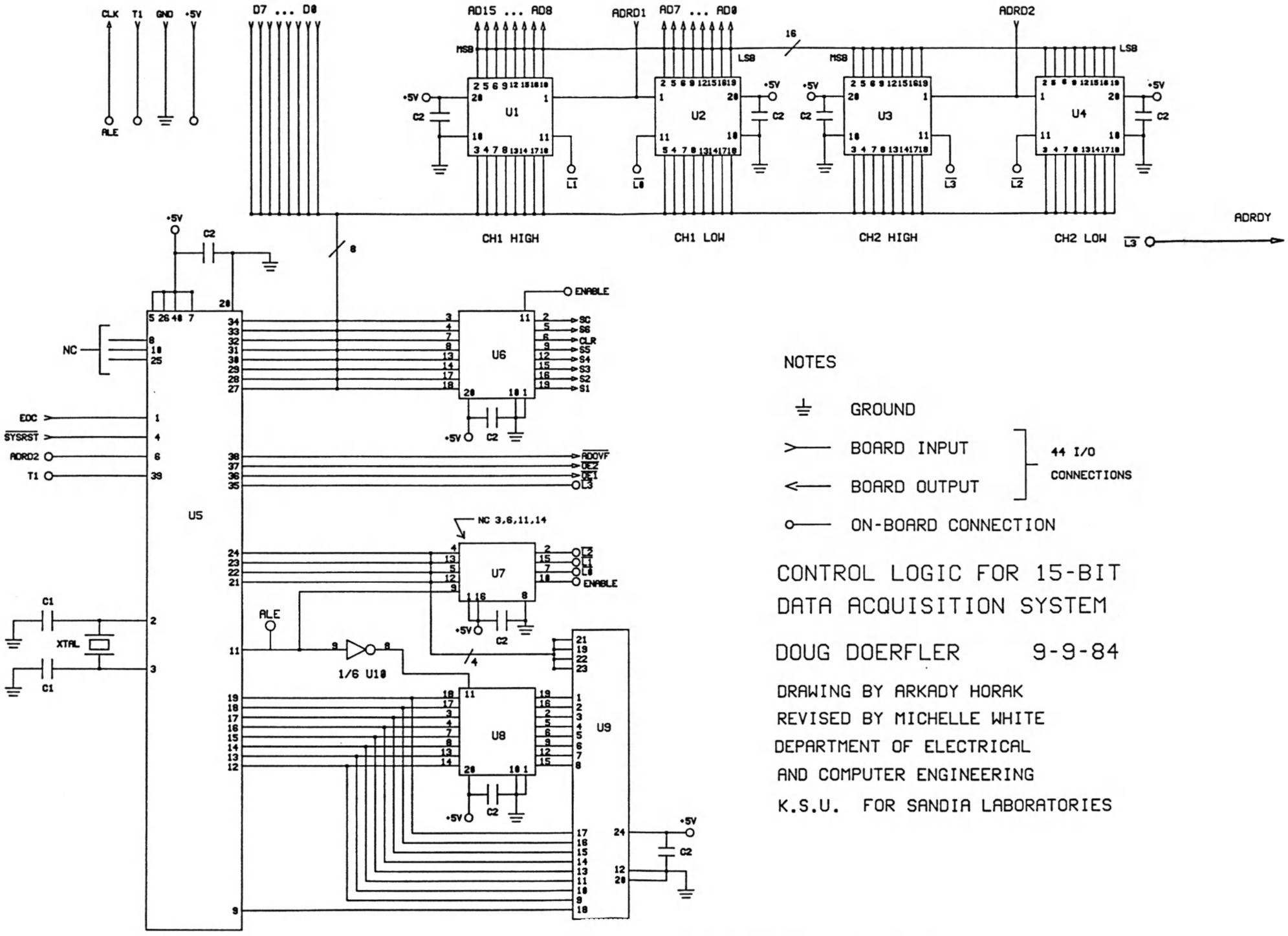
* use Allen-Bradley 1/4 watt, 5% or equivalent

$ R3 and R5 are packaged in 8 pin and 6 pin SIPs respectively, two SIPs are placed in parallel to achieve an 11 ohm value.

\# R9 is packaged in an 8 pin SIP

Capacitors

DEVICE	FUNCTION	# USED	
C1	0.1 µF Ceramic	1	All Ceramic capacitors are
C2	10 µF Tantalum	2	Panasonic disc ceramic
C3	0.1 µF Ceramic	1	caps or equivalent.
C4	10 µF Tantalum	1	
C5	10 µF Tantalum	7	All Tantalum capacitors
C6	0.1 µF Ceramic	1	are Panasonic SQ resin
C7	10 µF Tantalum	1	dipped solid tantalum
C8	10 µF Tantalum	5	caps or equivalent.
C9	0.01 µF Ceramic	6	
C10	59 pF Ceramic	1	
C11	10 pF Ceramic	1	
C12	0.01 µF Teflon	2	
C13	0.001 µF Ceramic	2	



NOTES

GROUND

BOARD INPUT

BOARD OUTPUT

44 I/O CONNECTIONS

ON-BOARD CONNECTION

CONTROL LOGIC FOR 15-BIT DATA ACQUISITION SYSTEM

DOUG DOERFLER 9-9-84

DRAWING BY ARKADY HORAK
REVISED BY MICHELLE WHITE
DEPARTMENT OF ELECTRICAL
AND COMPUTER ENGINEERING
K.S.U. FOR SANDIA LABORATORIES

Figure A2. Circuit Diagram of the Digital Section

Parts List for the Digital Section of the Low-Power, 15-Bit Data Acquisition System

ICs

DEVICE	FUNCTION	PART#	MANUFACT.	#PINS
U1	Octal D-type Latch	MM74C373	Nat. Semi.	20
U2	Octal D-type Latch	MM74C373	Nat. Semi.	20
U3	Octal D-type Latch	MM74C373	Nat. Semi.	20
U4	Octal D-type Latch	MM74C373	Nat. Semi.	20
U5	8-bit Microcomputer	80C39-7	Intel	40
U6	Octal D-type Latch	MM74C373	Nat. Semi.	20
U7	Quad D Flip-Flop	MM74C175	Nat. Semi.	16
U8	Octal D Flip-Flop	MM74C374	Nat. Semi.	20
U9	32k UV CMOS EPROM	MMC27C32	Nat. Semi.	24
U10	Hex Inverter	MM74C04	Nat. Semi.	14

Capacitors

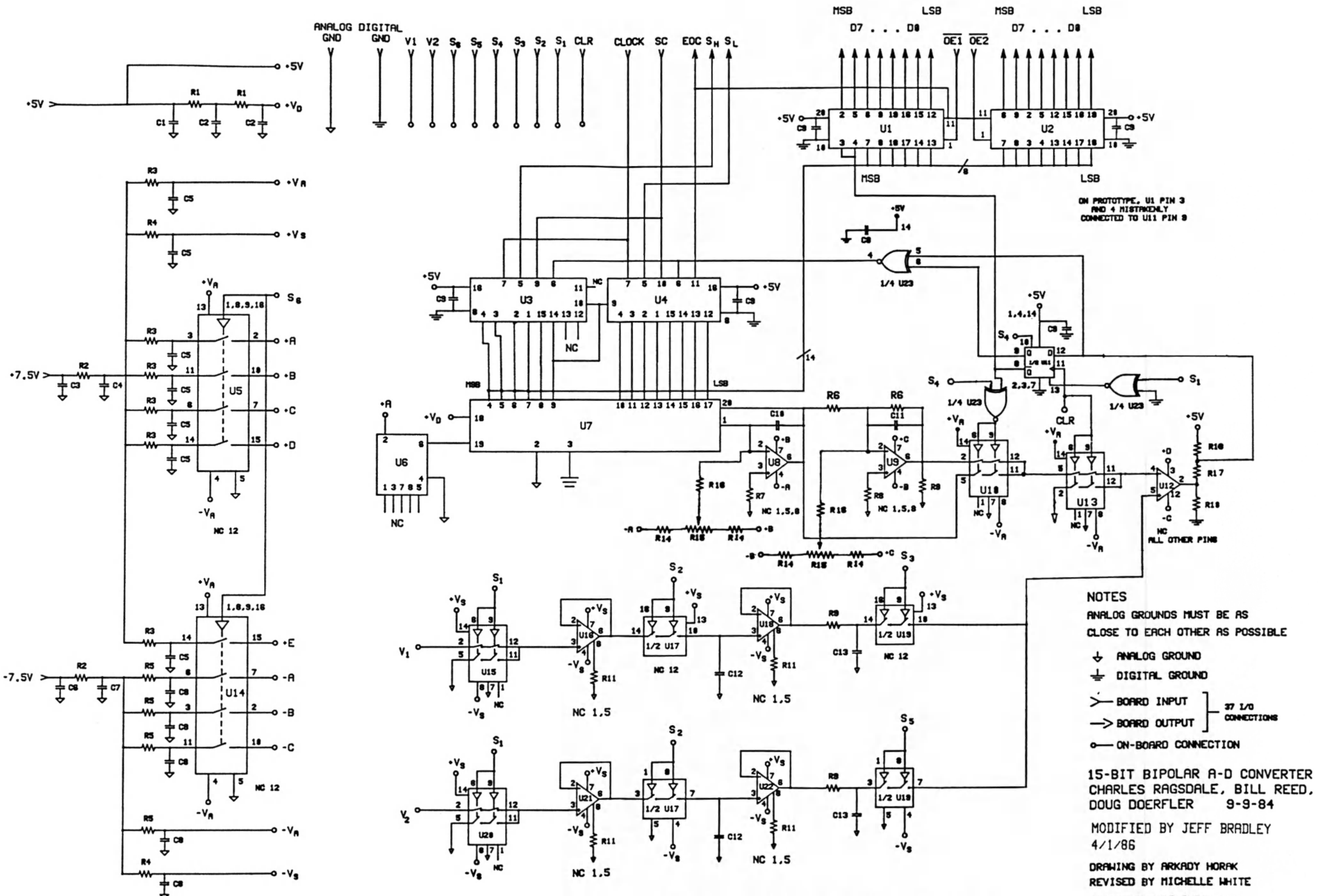
DEVICE	FUNCTION	# USED	
C1	20 pF Ceramic	10	Panasonic disc ceramic
C2	0.01 µF Ceramic	2	caps or equivalent.

Crystal

DEVICE	FUNCTION	
XTAL	1.0 MHz crystal	series resonant, AT cut, HC33 package

APPENDIX B

Circuit Diagram and Parts List for the Modified DAS



NOTES

ANALOG GROUNDS MUST BE AS CLOSE TO EACH OTHER AS POSSIBLE

⏚ ANALOG GROUND

⏚ DIGITAL GROUND

BOARD INPUT

BOARD OUTPUT } 37 I/O CONNECTIONS

ON-BOARD CONNECTION

15-BIT BIPOLAR A-D CONVERTER
CHARLES RAGSDALE, BILL REED, DOUG DOERFLER 9-9-84

MODIFIED BY JEFF BRADLEY
4/1/86

DRAWING BY ARKADY HORAK
REVISED BY MICHELLE WHITE

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING
K.S.U. FOR SANDIA LABORATORIES

Figure B1. Circuit Diagram of the Modified Analog Section

Parts List for the Modified Data Acquisition System

The parts lists of the analog and digital sections of the modified DAS are identical to the parts lists shown in Appendix A, with the following exceptions and additions.

Exceptions

1. U11: MM74C74 has been replaced by a MM74HC74.
2. U24: MM74C08 has been deleted.
3. R9: The 10 kohm resistor used as a pullup at the comparator's ouput is no longer used.
4. R13: The 10 kohm resistor has been deleted.

Additions

DEVICE	FUNCTION	PART#	MANUFACT.	# USED
R14	1 Mohm discrete resistor	*		4
R15	100 kohm potentiometer	unknown		2
R16	270 kohm discrete resist.	*		2
R17	6 kohm 1% disc. resistor	*		1
R18	4 kohm 1% disc. resistor	*		1

* use Allen-Bradley 1/4 watt, 5% or equivalent

APPENDIX C

Control Program for Existing Low-Power DAS

```
;
; PROGRAM: DYNAMIC 1.1
;
; PROGRAMMER: JEFF BRADLEY
; DATE: 4/3/86
;
; This program runs the two channel low-power DAS at a sampling
; frequency of 128 Hz.  It is a modification of a program written
; by Doerfler for the single channel prototype DAS.[5]  There is
; no data overflow check (new data ready before host has ret-
; rieved previous data) in this program.
;
; Register Usage:
;
;     R1 - low byte of channel 1 data
;     R2 - high byte of channel 1 data
;     R3 - low byte of channel 2 data
;     R4 - high byte of channel 2 data
;
; I/O Port Usage:
;
;     PORT 1 is used as an output port for the analog section's
;     control signals, as an input port for retrieval of data
;     from the analog section, and as an output port for sending
;     data to the digital section's output latches.
;
;     PORT 2 is used as an output port for control of latches on
;     both the analog and digital sections.
;
;     The port assignments for the control signals are as follows
;
;     PORT 1:  P17 - SC          PORT 2:  P27 - ADOVF
;              P16 - S6                   P26 - OE2
;              P15 - CLR                  P25 - OE1
;              P14 - S5                   P24 - L3
;              P13 - S4                   P23 - L2
;              P12 - S3                   P22 - L1
;              P11 - S2                   P21 - L0
;              P10 - S1                   P20 - ENABLE
;
Addr  Label      Mnemonic        Op Code          Comments
      ;
      ;****************
      ;* reset vector *
      ;****************
      ;
            ORG 000
000           EN TCNTI           25      ;enable timer interrupt
001           JMP Start          24 00
```

```
Addr  Label     Mnemonic          Op Code        Comments

      ;
      ;*************************
      ;* timer interrupt vector *
      ;*************************
      ;
            ORG 007
007         CLR A               27      ;clear flag
008         RETR                93
      ;
      ;******************
      ;* initialization *
      ;******************
      ;
            ORG 100                     ;initialize the 80C39's
                                        ;data ports
100   Start MOV A,#%00110100    23 34   ;initialize PORT 1
102         OUTL P1,A           39
103         MOV A,#%11100001    23 E1   ;initialize PORT 2
105         OUTL P2,A           3A
106         ANL P2,#%11111110   9A FE   ;disable latch U6 to
                                        ;hold ADC control status
      ;
108         NOP                 00      ;this space would be
109         NOP                 00      ;used to synchronize
10A         NOP                 00      ;the host system with
10B         NOP                 00      ;the DAS
10C         NOP                 00
10D         NOP                 00
10E         JMP Conv            24 70   ;begin operation
      ;
      ;************************
      ;* set up 128 Hz timer *
      ;************************
      ;
            ORG 170
170   Conv  NOP                 00
171         MOV A,#07H          23 07   ;delay loop to produce
173         CALL Delay          54 40   ;correct timing
175         STOP TCNT           65      ;stop the timer
176         MOV A,#F1H          23 F1   ;initialize timer
178         MOV T,A             62
179         STRT T              55      ;start timer
      ;
      ;*********************************
      ;* A/D conversion for channel 1 *
      ;*********************************
      ;
17A         MOV A,#%00110100    23 34   ;reinitialize PORT 1
17C         OUTL P1,A           39
17D         ORL P2,#%00000001   8A 01   ;enable latch U6 to
                                        ;allow ADC control
                                        ;signals to pass
17F         ORL P1,#%00000010   89 02   ;hold mode for t-and-h
```

```
Addr  Label     Mnemonic          Op Code     Comments
181             MOV A,#1          23 01   ;allow t-and-h to settle
183             CALL Delay        54 40
185             ORL P1,#%01000000 89 40   ;turn on power
187             MOV A,#%11010010  23 D2   ;connect ch. 1 to compar-
                                          ;ator, start sign det.
189             OUTL P1,A         39
18A             ORL P1,#%00100000 89 20   ;latch sign
18C             ANL P1,#%01111111 99 7F   ;clear SC
18E             ORL P1,#%10000000 89 80   ;start 14-bit conversion
190             ANL P1,#%01111111 99 7F   ;clear SC again
192             MOV A,#4          23 04   ;wait for conversion to
                                          ;complete
194             CALL Delay        54 40
196             ORL P1,#%00000100 89 04   ;cut off ch. 1 from com-
                                          ;parator
198             ANL P2,#%11111110 9A FE   ;disable latch U6 to hold
                                          ;ADC control signals
        ;
        ;*****************************
        ;* read result for channel 1 *
        ;*****************************
        ;
19A             ORL P1,#FFH       89 FF   ;set all PORT 1 bits for
                                          ;input
19C             ANL P2,#%11011111 9A DF   ;clear OE1 to enable
                                          ;latch U1
19E             IN A,P1           09      ;read ch. 1 high byte
19F             NOP               00      ;this space for toggling
1A0             NOP               00      ;sign bits because of
                                          ;board layout error on
                                          ;prototype PC boards
1A1             MOV R2,A          AA      ;store byte in reg. R2
1A2             ORL P2,#%00100000 8A 20   ;set OE1 to disable U1
1A4             ANL P2,#%10111111 9A BF   ;clear OE2 to enable
                                          ;latch U2
1A6             IN A,P1           09      ;read ch. 1 low byte
1A7             MOV R1,A          A9      ;store byte in reg. R1
1A8             ORL P2,#%01000000 8A 40   ;set OE2 to disable U2
        ;
        ;*********************************
        ;* A/D conversion for channel 2 *
        ;*********************************
        ;
1AA             MOV A,#%11000110  23 C6   ;reinitialize PORT 1
                                          ;connect ch. 2 to com-
                                          ;parator input, start
                                          ;sign determination
1AC             OUTL P1,A         39
1AD             ORL P2,#%00000001 8A 01   ;enable latch U6 to
                                          ;allow ADC control
                                          ;signals to pass
1AF             ORL P1,#%00100000 89 20   ;latch sign
1B1             ANL P1,#%01111111 99 7F   ;clear SC
```

```
Addr  Label      Mnemonic          Op Code      Comments
1B3              ORL P1,#%10000000  89 80   ;start 14-bit conversion
1B5              ANL P1,#%01111111  99 7F   ;clear SC again
1B7              MOV A,#4           23 04   ;wait for conversion to
                                            ;complete
1B9              CALL Delay         54 40
1BB              MOV A,#%00110100   23 34   ;cut off ch. 2 from com-
1BD              OUTL P1,A          39      ;parator, put t-and-h in
                                            ;track mode, power off
1BF              ANL P2,#%11111110  9A FE   ;disable latch U6 to hold
                                            ;ADC control signals
          ;
          ;*****************************
          ;* read result for channel 2 *
          ;*****************************
          ;
1C0              ORL P1,#FFH        89 FF   ;set all PORT 1 bits for
                                            ;input
1C2              ANL P2,#%11011111  9A DF   ;clear OE1 to enable
                                            ;latch U1
1C4              IN A,P1            09      ;read ch. 2 high byte
1C5              NOP                00      ;this space for toggling
1C6              NOP                00      ;sign bits because of
                                            ;board layout error on
                                            ;prototype PC boards
1C7              MOV R4,A           AC      ;store byte in reg. R4
1C8              ORL P2,#%00100000  8A 20   ;set OE1 to disable U1
1CA              ANL P2,#%10111111  9A BF   ;clear OE2 to enable
                                            ;latch U2
1CC              IN A,P1            09      ;read ch. 2 low byte
1CD              MOV R3,A           AB      ;store byte in reg. R3
1CE              ORL P2,#%01000000  8A 40   ;set OE2 to disable U2
          ;
          ;*************************************************
          ;* calculate 2's complement of negative data from *
          ;* channel 1                                      *
          ;*************************************************
          ;
                                            ;see if output code is -0
                                            ;if so, then replace with
                                            ;+0
                                            ;
1D0              MOV A,R1           F9      ;get ch. 1 low byte
1D1              JNZ Jump1          96 DB   ;jump out if low byte
                                            ;not 0
1D3              MOV A,R2           FA      ;get ch. 1 high byte
1D4              XRL A,#%11000000   D3 C0   ;check sign bits
1D6              JNZ Jump1          96 DB   ;if high byte wasn't
                                            ;equal to C0H then jump
1D8              MOV R2,A           AA      ;else store 0 into high
                                            ;byte
1D9              JMP Jump2          24 EC   ;don't need further
                                            ;processing, so jump
          ;
```

```
Addr  Label     Mnemonic          Op Code        Comments

                                                 ;see if code is negative
                                                 ;if so, take 2's comp.
                                                 ;
1DB   Jump1     MOV A,R2          FA             ;get ch. 1 high byte
1DC             JB6 Jump3         D2 E0          ;if not negative, no
1DE             JMP Jump2         24 EC          ;further processing
                                                 ;needed
1E0   Jump3     MOV A,R1          F9             ;get ch. 1 low byte
1E1             CPL A             37             ;complement low byte
1E2             ADD A,#01H        03 01          ;add 1
1E4             MOV R1,A          A9             ;restore the low byte
1E5             MOV A,R2          FA             ;get ch. 1 high byte
1E6             CPL A             37             ;complement high byte
1E7             ADDC A,#00H       13 00          ;add carry from last add
1E9             XRL A,#C0H        D3 C0          ;set 2 msb's
1EB             MOV R2,A          AA             ;restore high byte
        ;
        ;*********************************************************
        ;* calculate the 2's complement of negative data from *
        ;* channel 2                                           *
        ;*********************************************************
        ;
                                                 ;see if output code is -0
                                                 ;if so, then replace with
                                                 ;+0
                                                 ;
1EC   Jump2     MOV A,R3          FB             ;get ch. 2 low byte
1ED             JNZ Jump4         96 F7          ;jump out if low byte
                                                 ;not 0
1EF             MOV A,R4          FC             ;get ch. 2 high byte
1F0             XRL A,#%11000000  D3 C0          ;check sign bits
1F2             JNZ Jump4         96 F7          ;if high byte wasn't
                                                 ;equal to C0H then jump
1F4             MOV R4,A          AC             ;else store 0 into high
                                                 ;byte
1F5             JMP Jump5          44 08         ;don't need further
                                                 ;processing, so jump
        ;
                                                 ;see if code is negative
                                                 ;if so, take 2's comp.
                                                 ;
1F7   Jump4     MOV A,R4          FC             ;get ch. 2 high byte
1F8             JB6 Jump6         D2 FC          ;if not negative, no
1FA             JMP Jump5         44 08          ;further processing
                                                 ;needed
1FC   Jump6     MOV A,R3          FB             ;get ch. 2 low byte
1FD             CPL A             37             ;complement low byte
1FE             ADD A,#01H        03 01          ;add 1
200             MOV R3,A          AB             ;restore the low byte
201             MOV A,R4          FC             ;get ch. 2 high byte
202             CPL A             37             ;complement high byte
203             ADDC A,#00H       13 00          ;add carry from last add
205             XRL A,#C0H        D3 C0          ;set 2 msb's
```

```
Addr  Label     Mnemonic          Op Code      Comments

207             MOV R4,A          AC       ;restore high byte
      ;
      ;*************************************************
      ;* send data to output latches of digital section *
      ;*************************************************
      ;
208   Jump5     MOV A,R1          F9       ;get ch. 1 low byte
209             OUTL P1,A         39       ;send out to PORT 1
20A             ORL P2,#%00000010 8A 02    ;latch into digital
20C             ANL P2,#%11111101 9A FD    ;section latch U2
20E             MOV A,R2          FA       ;get ch. 1 high byte
20F             OUTL P1,A         39       ;send out to PORT 1
210             ORL P2,#%00000100 8A 04    ;latch into digital
212             ANL P2,#%11111011 9A FB    ;section latch U1
      ;
214             MOV A,R3          FB       ;get ch. 2 low byte
215             OUTL P1,A         39       ;send out to PORT 1
216             ORL P2,#%00001000 8A 08    ;latch into digital
218             ANL P2,#%11110111 9A F7    ;section latch U4
21A             MOV A,R4          FC       ;get ch. 2 high byte
21B             OUTL P1,A         39       ;send out to PORT 1
21C             ORL P2,#%00010000 8A 10    ;latch into digital
                                           ;section latch U3
21E             NOP               00       ;this space is for
21F             NOP               00       ;handshaking with the
220             NOP               00       ;host system
221             NOP               00
222             NOP               00
223             NOP               00
224             NOP               00
225             NOP               00
226             ANL P2,#%11101111 9A EF
      ;
      ;*************************************************
      ;* set flag indicating the cycle is done and wait *
      ;* for timer to interrupt to begin the next cycle *
      ;*************************************************
      ;
228             NOP               00
229             MOV A,#FFH        23 FF    ;fill accumulator with
                                           ;1's to indicate cycle
                                           ;complete
22B   Wait      JNZ Wait          96 2B    ;wait for timer interrupt
22D             JMP Conv          24 70    ;having returned from
                                           ;timer interrupt vector,
                                           ;begin another conversion
                                           ;cycle
;
;
;
;
;
;
```

```
Addr  Label     Mnemonic        Op Code        Comments
      ;
      ;*********************
      ;* delay subroutine *
      ;*********************
      ;
              ORG 240
240   Delay   DEC A             07         ;delay = 90 + 45 * A µs
241           JNZ Delay         96 40      ;including preceding MOV
243           RET               83         ;and CALL instructions
```

APPENDIX D

Control Program for Low-Power DAS Using Error Compensation

Control Program for Low-Power DAS Using Error Compensation

```
;
; PROGRAM: DYNAMIC 2.1
;
; PROGRAMMER: JEFF BRADLEY
; DATE: 4/11/86
;
; This program runs the two channel low-power DAS at a sampling
; frequency of 128 Hz.  It performs the error compensation
; described in chapter 3.  The analog common is used as the
; polarity determination reference.  There is no data overflow
; check (new data ready before host has retrieved previous data)
; in this program.  Note that this program is not guaranteed to
; be bug free!
;
; Register Usage:
;
;     R0  - pointer for indirect addressing
;     R1  - low byte of channel 1 data
;     R2  - high byte of channel 1 data
;     R3  - low byte of channel 2 data
;     R4  - high byte of channel 2 data
;     R5  - low byte of conversion cycle counter
;     R6  - high byte of conversion cycle counter
;     R24 - negative range correction code for channel 1
;     R25 - negative range correction code for channel 2
;     R26 - positive range correction code for channel 1
;     R27 - positive range correction code for channel 2
;
; I/O Port Usage:
;
;     PORT 1 is used as an output port for the analog section's
;     control signals, as an input port for retrieval of data
;     from the analog section, and as an output port for sending
;     data to the digital section's output latches.
;
;     PORT 2 is used as an output port for control of latches on
;     both the analog and digital sections.
;
;     The port assignments for the control signals are as follows
;
;     PORT 1:  P17 - SC          PORT 2:  P27 - ADOVF
;              P16 - S6                   P26 - OE2
;              P15 - CLR                  P25 - OE1
;              P14 - S5                   P24 - L3
;              P13 - S4                   P23 - L2
;              P12 - S3                   P22 - L1
;              P11 - S2                   P21 - L0
;              P10 - S1                   P20 - ENABLE
;
```

```
Addr  Label     Mnemonic          Op Code      Comments

      ;
      ;****************
      ;* reset vector *
      ;****************
      ;
            ORG 000
000         EN TCNTI            25           ;enable timer interrupt
001         JMP Start           24 00
      ;
      ;****************************
      ;* timer interrupt vector *
      ;****************************
      ;
            ORG 007
007         CLR A               27           ;clear flag
008         RETR                93
      ;
      ;******************
      ;* initialization *
      ;******************
      ;
            ORG 100                          ;initialize the 80C39's
                                             ;data ports
100   Start MOV A,#%00111100    23 3C        ;initialize PORT 1
102         OUTL P1,A           39
103         MOV A,#%11100001    23 E1        ;initialize PORT 2
105         OUTL P2,A           3A
106         ANL P2,#%11111110   9A FE        ;disable latch U6 to
                                             ;hold ADC control status
      ;
108         MOV R5,#00H         BD 00        ;initialize cycle counter
                                             ;low byte
10A         MOV R6,#00H         BE 00        ;initialize cycle counter
                                             ;high byte
10C         CLR F1              A5           ;initialize flag for
                                             ;correction cycle
10D         CALL Corr1          74 00        ;perform 2 correction
10F         CALL Corr2          74 80        ;cycles as a warm up
111         CALL Corr1          74 00
113         CALL Corr2          74 80
115         NOP                 00           ;this space would be
116         NOP                 00           ;used to synchronize
117         NOP                 00           ;the host system with
118         NOP                 00           ;the DAS
119         NOP                 00
11A         NOP                 00
11B         JMP Conv            24 70        ;begin operation
      ;
      ;************************
      ;* set up 128 Hz timer *
      ;************************
      ;
```

```
Addr  Label     Mnemonic          Op Code      Comments

            ORG 170
170   Conv    NOP               00
171           MOV A,#07H        23 07    ;delay loop to produce
173           CALL Delay        54 80    ;correct timing
175           STOP TCNT         65       ;stop the timer
176           MOV A,#F1H        23 F1    ;initialize timer
178           MOV T,A           62
179           STRT T            55       ;start timer
      ;
      ;*********************************
      ;* A/D conversion for channel 1 *
      ;*********************************
      ;
17A           MOV A,#%00111100  23 3C    ;reinitialize PORT 1
17C           OUTL P1,A         39
17D           ORL P2,#%00000001 8A 01    ;enable latch U6 to
                                         ;allow ADC control
                                         ;signals to pass
17F           ORL P1,#%00000010 89 02    ;hold mode for t-and-h
181           MOV A,#1          23 01    ;allow t-and-h to settle
183           CALL Delay        54 80
185           ORL P1,#%01000000 89 40    ;turn on power
187           ANL P1,#%11011011 99 DB    ;connect ch. 1 to compar-
                                         ;ator, start sign det.
189           NOP               00       ;delay
18A           ORL P1,#%00100000 89 20    ;latch sign
18C           NOP               00       ;delay
18D           ORL P1,#%10000000 89 80    ;start 14-bit conversion
18F           ANL P1,#%01111111 99 7F    ;clear SC again
191           MOV A,#4          23 04    ;wait for conversion to
                                         ;complete
193           CALL Delay        54 80
195           ORL P1,#%00000100 89 04    ;cut off ch. 1 from com-
                                         ;parator
197           ANL P2,#%11111110 9A FE    ;disable latch U6 to hold
                                         ;ADC control signals
      ;
      ;*******************************
      ;* read result for channel 1 *
      ;*******************************
      ;
199           ORL P1,#FFH       89 FF    ;set all PORT 1 bits for
                                         ;input
19B           ANL P2,#%11011111 9A DF    ;clear OE1 to enable
                                         ;latch U1
19D           IN A,P1           09       ;read ch. 1 high byte
19E           NOP               00       ;this space for toggling
19F           NOP               00       ;sign bits because of
                                         ;board layout error on
                                         ;prototype PC boards
1A0           MOV R2,A          AA       ;store byte in reg. R2
1A1           ORL P2,#%00100000 8A 20    ;set OE1 to disable U1
```

```
Addr  Label     Mnemonic          Op Code     Comments
1A3             ANL P2,#%10111111  9A BF   ;clear OE2 to enable
                                           ;latch U2
1A5             IN A,P1            09      ;read ch. 1 low byte
1A6             MOV R1,A           A9      ;store byte in reg. R1
1A7             ORL P2,#%01000000  8A 40   ;set OE2 to disable U2
        ;
        ;*********************************
        ;* A/D conversion for channel 2 *
        ;*********************************
        ;
1A9             MOV A,#%01001110   23 4E   ;reinitialize PORT 1
                                           ;connect ch. 2 to com-
                                           ;parator input, start
                                           ;sign determination
1AB             OUTL P1,A          39
1AC             ORL P2,#%00000001  8A 01   ;enable latch U6 to
                                           ;allow ADC control
                                           ;signals to pass
1AE             NOP                00      ;delay
1AF             ORL P1,#%00100000  89 20   ;latch sign
1B1             NOP                00      ;delay
1B2             ORL P1,#%10000000  89 80   ;start 14-bit conversion
1B4             ANL P1,#%01111111  99 7F   ;clear SC again
1B6             MOV A,#4           23 04   ;wait for conversion to
                                           ;complete
1B8             CALL Delay         54 80
1BA             MOV A,#%00111100   23 3C   ;cut off ch. 2 from com-
1BC             OUTL P1,A          39      ;parator, put t-and-h in
                                           ;track mode, power off
1BD             ANL P2,#%11111110  9A FE   ;disable latch U6 to hold
                                           ;ADC control signals
        ;
        ;*******************************
        ;* read result for channel 2 *
        ;*******************************
        ;
1BF             ORL P1,#FFH        89 FF   ;set all PORT 1 bits for
                                           ;input
1C1             ANL P2,#%11011111  9A DF   ;clear OE1 to enable
                                           ;latch U1
1C3             IN A,P1            09      ;read ch. 2 high byte
1C4             NOP                00      ;this space for toggling
1C5             NOP                00      ;sign bits because of
                                           ;board layout error on
                                           ;prototype PC boards
1C6             MOV R4,A           AC      ;store byte in reg. R4
1C7             ORL P2,#%00100000  8A 20   ;set OE1 to disable U1
1C9             ANL P2,#%10111111  9A BF   ;clear OE2 to enable
                                           ;latch U2
1CB             IN A,P1            09      ;read ch. 2 low byte
1CC             MOV R3,A           AB      ;store byte in reg. R3
1CD             ORL P2,#%01000000  8A 40   ;set OE2 to disable U2
```

```
Addr  Label     Mnemonic          Op Code      Comments
      ;
      ;*************************************************
      ;* calculate 2's complement of negative data from *
      ;* channel 1                                      *
      ;*************************************************
      ;
                                                ;see if output code is -0
                                                ;if so, then replace with
                                                ;+0
                                                ;
1CF             MOV A,R1          F9            ;get ch. 1 low byte
1D0             JNZ Jump1         96 DA         ;jump out if low byte
                                                ;not 0
1D2             MOV A,R2          FA            ;get ch. 1 high byte
1D3             XRL A,#%11000000  D3 C0         ;check sign bits
1D5             JNZ Jump1         96 DA         ;if high byte wasn't
                                                ;equal to C0H then jump
1D7             MOV R2,A          AA            ;else store 0 into high
                                                ;byte
1D8             JMP Jump2         24 EB         ;don't need further
                                                ;processing, so jump
      ;
                                                ;see if code is negative
                                                ;if so, take 2's comp.
                                                ;
1DA   Jump1     MOV A,R2          FA            ;get ch. 1 high byte
1DB             JB6 Jump3         D2 DF         ;if not negative, no
1DD             JMP Jump2         24 EB         ;further processing
                                                ;needed
1DF   Jump3     MOV A,R1          F9            ;get ch. 1 low byte
1E0             CPL A             37            ;complement low byte
1E1             ADD A,#01H        03 01         ;add 1
1E3             MOV R1,A          A9            ;restore the low byte
1E4             MOV A,R2          FA            ;get ch. 1 high byte
1E5             CPL A             37            ;complement high byte
1E6             ADDC A,#00H       13 00         ;add carry from last add
1E8             XRL A,#C0H        D3 C0         ;set 2 msb's
1EA             MOV R2,A          AA            ;restore high byte
      ;
      ;*************************************************
      ;* calculate the 2's complement of negative data from *
      ;* channel 2                                      *
      ;*************************************************
      ;
                                                ;see if output code is -0
                                                ;if so, then replace with
                                                ;+0
                                                ;
1EB   Jump2     MOV A,R3          FB            ;get ch. 2 low byte
1EC             JNZ Jump4         96 F6         ;jump out if low byte
                                                ;not 0
1EE             MOV A,R4          FC            ;get ch. 2 high byte
```

```
Addr  Label     Mnemonic          Op Code     Comments

1EF             XRL A,#%11000000  D3 C0   ;check sign bits
1F1             JNZ Jump4         96 F6   ;if high byte wasn't
                                          ;equal to C0H then jump
1F3             MOV R4,A          AC      ;else store 0 into high
                                          ;byte
1F4             JMP Jump5         44 07   ;don't need further
                                          ;processing, so jump
      ;
                                          ;see if code is negative
                                          ;if so, take 2's comp.
                                          ;
1F6   Jump4     MOV A,R4          FC      ;get ch. 2 high byte
1F7             JB6 Jump6         D2 FB   ;if not negative, no
1F9             JMP Jump5         44 07   ;further processing
                                          ;needed
1FB   Jump6     MOV A,R3          FB      ;get ch. 2 low byte
1FC             CPL A             37      ;complement low byte
1FD             ADD A,#01H        03 01   ;add 1
1FF             MOV R3,A          AB      ;restore the low byte
200             MOV A,R4          FC      ;get ch. 2 high byte
201             CPL A             37      ;complement high byte
202             ADDC A,#00H       13 00   ;add carry from last add
204             XRL A,#C0H        D3 C0   ;set 2 msb's
206             MOV R4,A          AC      ;restore high byte
      ;
      ;*****************************************************
      ;* apply correction codes to channel 1 output codes *
      ;*****************************************************
      ;
207   Jump5     MOV A,R2          FA      ;get ch. 1 high byte
208             JB6 Neg1          D2 15   ;check to see if code
                                          ;is negative
      ;
20A             MOV A,R1          F9      ;get ch. 1 low byte
20B             MOV R0,#1AH       B8 1A   ;point to ch. 1 pos.
                                          ;range corr. code
20D             ADD A,@R0         60      ;apply corr. code
20E             MOV R1,A          A9      ;restore low byte
20F             MOV A,R2          FA      ;get ch. 1 high byte
210             ADDC A,#FFH       13 FF   ;complete application
                                          ;of corr. code
212             MOV R2,A          AA      ;restore high byte
213             JMP Test2         44 1E   ;repeat process for ch. 2
      ;
215   Neg1      MOV A,R1          F9      ;get ch. 1 low byte
216             MOV R0,#18H       B8 18   ;point to ch. 1 neg.
                                          ;range corr. code
218             ADD A,@R0         60      ;apply corr. code
219             MOV R1,A          A9      ;restore low byte
21A             MOV A,R2          FA      ;get ch. 1 high byte
21B             ADDC A,#00H       13 00   ;complete application
                                          ;of corr. code
21D             MOV R2,A          AA      ;restore high byte
```

```
Addr  Label     Mnemonic        Op Code      Comments

      ;
      ;*****************************************************
      ;* apply correction codes to channel 2 output codes *
      ;*****************************************************
      ;
21E   Test2     MOV A,R4          FC         ;get ch. 2 high byte
21F             JB6 Neg2          D2 2D      ;check to see if code
                                             ;is negative
      ;
221             MOV A,R3          FB         ;get ch. 2 low byte
222             MOV R0,#1BH       B8 1B      ;point to ch. 2 pos.
                                             ;range corr. code
224             ADD A,@R0         60         ;apply corr. code
225             MOV R3,A          AB         ;restore low byte
226             MOV A,R4          FC         ;get ch. 2 high byte
227             ADDC A,#FFH       13 FF      ;complete application
                                             ;of corr. code
229             MOV R4,A          AC         ;restore high byte
22A             JMP Send          44 35      ;go send the data
      ;
22C   Neg2      MOV A,R3          FB         ;get ch. 2 low byte
22D             MOV R0,#19H       B8 19      ;point to ch. 2 neg.
                                             ;range corr. code
22F             ADD A,@R0         60         ;apply corr. code
230             MOV R3,A          AB         ;restore low byte
231             MOV A,R4          FC         ;get ch. 2 high byte
232             ADDC A,#00H       13 00      ;complete application
                                             ;of corr. code
234             MOV R4,A          AC         ;restore high byte
      ;
      ;*****************************************************
      ;* send data to output latches of digital section *
      ;*****************************************************
      ;
235   Send      MOV A,R1          F9         ;get ch. 1 low byte
236             OUTL P1,A         39         ;send out to PORT 1
237             ORL P2,#%00000010 8A 02      ;latch into digital
239             ANL P2,#%11111101 9A FD      ;section latch U2
23B             MOV A,R2          FA         ;get ch. 1 high byte
23C             OUTL P1,A         39         ;send out to PORT 1
23D             ORL P2,#%00000100 8A 04      ;latch into digital
23F             ANL P2,#%11111011 9A FB      ;section latch U1
      ;
241             MOV A,R3          FB         ;get ch. 2 low byte
242             OUTL P1,A         39         ;send out to PORT 1
243             ORL P2,#%00001000 8A 08      ;latch into digital
245             ANL P2,#%11110111 9A F7      ;section latch U4
247             MOV A,R4          FC         ;get ch. 2 high byte
248             OUTL P1,A         39         ;send out to PORT 1
249             ORL P2,#%00010000 8A 10      ;latch into digital
                                             ;section latch U3
24B             NOP               00         ;this space is for
24C             NOP               00         ;handshaking with the
```

```
Addr  Label     Mnemonic          Op Code      Comments

24D             NOP               00      ;host system
24E             NOP               00
24F             NOP               00
250             NOP               00
251             NOP               00
252             NOP               00
253             ANL P2,#%11101111 9A EF
        ;
        ;*************************************************
        ;* check to see if it is time to determine new   *
        ;* correction codes; they are found after every  *
        ;* 65536 conversion cycles, or every 8.5 minutes *
        ;*************************************************
        ;
255             JF1 Leap3         76 69   ;if F1 is set, get
                                          ;pos. range corr. codes
                                          ;
257             MOV A,R5          FD      ;increment cycle counter
                                          ;low byte
258             ADD A,#01H        03 01
25A             MOV R5,A          AD      ;restore low byte
25B             MOV A,R6          FE      ;increment cycle counter
                                          ;high byte
25C             ADDC A,#00H       13 00
25E             MOV R6,A          AE      ;restore high byte
        ;
25F             JNZ Out1          96 6C   ;test counter high byte
                                          ;to see if 0
261             MOV A,R5          FD      ;get low byte to test
262             JNZ Out1          96 6C   ;test to see if 0
        ;
        ;*************************************************
        ;* counter has been detemined to be 0 and flag F1 is *
        ;* 0, so get negative range correction codes     *
        ;*************************************************
        ;
264             CALL Corr1        74 00
266             CPL F1            B5      ;set F1 so that next
                                          ;time around the pos.
                                          ;range corr. codes will
                                          ;be determined
267             JMP Out1          44 6C
        ;
        ;*************************************************
        ;* flag F1 is 1, so get positive range correction codes *
        ;*************************************************
        ;
269 Leap3   CALL Corr2        74 80
26B             CLR F1            A5      ;clear F1 for next time
```

```
Addr  Label     Mnemonic          Op Code        Comments
      ;
      ;*************************************************
      ;* set flag indicating the cycle is done and wait *
      ;* for timer to interrupt to begin the next cycle *
      ;*************************************************
      ;
26C   Outl      NOP               00
26D             MOV A,#FFH        23 FF    ;fill accumulator with
                                           ;1's to indicate cycle
                                           ;complete
26F   Wait      JNZ Wait          96 6F    ;wait for timer interrupt
22D             JMP Conv          24 70    ;having returned from
                                           ;timer interrupt vector,
                                           ;begin another conversion
                                           ;cycle
      ;
      ;*********************
      ;* delay subroutine *
      ;*********************
      ;
                ORG 280
280   Delay     DEC A             07       ;delay = 90 + 45 * A µs
281             JNZ Delay         96 80    ;including preceding MOV
283             RET               83       ;and CALL instructions
      ;
      ;*************************************************
      ;* determination of negative range correction codes *
      ;* channel 1 first                                 *
      ;*************************************************
      ;
                ORG 300
300   Corr1     MOV A,#%00111101  23 3D    ;reinitialize PORT 1
302             OUTL P1,A         39       ;neg. range, inputs gnded
303             ORL P2,#%00000001 8A 01    ;enable latch U6 to
                                           ;allow ADC control
                                           ;signals to pass
305             MOV A,#5          23 05    ;wait for t-and-h to slew
307             CALL Delay        54 80
309             ORL P1,#%00000010 89 02    ;hold mode for t-and-h
30B             MOV A,#1          23 01    ;allow t-and-h to settle
30D             CALL Delay        54 80
30F             ORL P1,#%01000000 89 40    ;turn on power
311             MOV A,#%01111011  23 7B    ;connect ch. 1 to compar-
313             OUTL P1,A         39       ;ator
314             ORL P1,#%10000000 89 80    ;start 14-bit conversion
316             ANL P1,#%01111111 99 7F    ;clear SC again
318             MOV A,#4          23 04    ;wait for conversion to
                                           ;complete
31A             CALL Delay        54 80
31C             ORL P1,#%00000100 89 04    ;cut off ch. 1 from com-
                                           ;parator
31E             ANL P2,#%11111110 9A FE    ;disable latch U6 to hold
                                           ;ADC control signals
```

```
Addr  Label     Mnemonic        Op Code      Comments
      ;
      ;*************************************************************
      ;* read negative range correction code for channel 1         *
      ;* since the hardware offset is small, the important         *
      ;* part of the correction code is in the lower 8 bits.       *
      ;* therefore, only the lower 8 bits are read and stored.     *
      ;* if for some reason the hardware offset is so large        *
      ;* that 8 bits can't represent it, then an error will        *
      ;* occur.                                                    *
      ;*************************************************************
      ;
320           ORL P1,#FFH          89 FF   ;set all PORT 1 bits for
                                           ;input
322           ANL P2,#%10111111    9A BF   ;clear OE2 to enable
                                           ;latch U2
324           IN A,P1              09      ;read ch. 1 low byte
325           ORL P2,#%01000000    8A 40   ;set OE2 to disable U2
327           MOV R0,#18H          B8 18   ;point to R24
329           MOV @R0,A            A0      ;store neg. range corr.
                                           ;code in R24
      ;
      ;*************************************************
      ;* get channel 2's negative correction code now *
      ;*************************************************
      ;
32A           MOV A,#%01101111     23 6F   ;reinitialize PORT 1
32C           OUTL P1,A            39      ;neg. range, inputs gnded
                                           ;ch. 2 to comparator
32D           ORL P2,#%00000001    8A 01   ;enable latch U6 to
                                           ;allow ADC control
                                           ;signals to pass
32F           ORL P1,#%10000000    89 80   ;start 14-bit conversion
331           ANL P1,#%01111111    99 7F   ;clear SC again
333           MOV A,#4             23 04   ;wait for conversion to
                                           ;complete
335           CALL Delay           54 80
337           MOV A,#%00111100     23 3C   ;cut off ch. 2 from com-
339           OUTL P1,A            39      ;parator, t-and-h in
                                           ;track mode, power off
                                           ;inputs connected to
                                           ;signal
33A           ANL P2,#%11111110    9A FE   ;disable latch U6 to hold
                                           ;ADC control signals
      ;
      ;*************************************************************
      ;* read negative range correction code for channel 2         *
      ;* since the hardware offset is small, the important         *
      ;* part of the correction code is in the lower 8 bits.       *
      ;* therefore, only the lower 8 bits are read and stored.     *
      ;* if for some reason the hardware offset is so large        *
      ;* that 8 bits can't represent it, then an error will        *
      ;* occur.                                                    *
      ;*************************************************************
      ;
```

```
Addr  Label     Mnemonic          Op Code      Comments

33C             ORL P1,#FFH       89 FF   ;set all PORT 1 bits for
                                          ;input
33E             ANL P2,#%10111111 9A BF   ;clear OE2 to enable
                                          ;latch U2
340             IN A,P1           09      ;read ch. 1 low byte
341             ORL P2,#%01000000 8A 40   ;set OE2 to disable U2
343             MOV R0,#19H       B8 19   ;point to R25
345             MOV @R0,A         A0      ;store neg. range corr.
                                          ;code in R25
346             RET               83      ;return
        ;
        ;
        ;*****************************************************
        ;* determination of positive range correction codes *
        ;* channel 1 first                                   *
        ;*****************************************************
        ;
                ORG 380
380   Corr2     MOV A,#%00110101  23 35   ;reinitialize PORT 1
382             OUTL P1,A         39      ;pos. range, inputs gnded
383             ORL P2,#%00000001 8A 01   ;enable latch U6 to
                                          ;allow ADC control
                                          ;signals to pass
385             MOV A,#5          23 05   ;wait for t-and-h to slew
387             CALL Delay        54 80
389             ORL P1,#%00000010 89 02   ;hold mode for t-and-h
38B             MOV A,#1          23 01   ;allow t-and-h to settle
38D             CALL Delay        54 80
38F             ORL P1,#%01000000 89 40   ;turn on power
391             MOV A,#%01110011  23 73   ;connect ch. 1 to compar-
393             OUTL P1,A         39      ;ator
394             ORL P1,#%10000000 89 80   ;start 14-bit conversion
396             ANL P1,#%01111111 99 7F   ;clear SC again
398             MOV A,#4          23 04   ;wait for conversion to
                                          ;complete
39A             CALL Delay        54 80
39C             ORL P1,#%00000100 89 04   ;cut off ch. 1 from com-
                                          ;parator
39E             ANL P2,#%11111110 9A FE   ;disable latch U6 to hold
                                          ;ADC control signals
        ;
        ;*********************************************************
        ;* read positive range correction code for channel 1     *
        ;* since the hardware offset is small, the important     *
        ;* part of the correction code is in the lower 8 bits.   *
        ;* therefore, only the lower 8 bits are read and stored. *
        ;* if for some reason the hardware offset is so large    *
        ;* that 8 bits can't represent it, then an error will    *
        ;* occur.                                                *
        ;*********************************************************
        ;
3A0             ORL P1,#FFH       89 FF   ;set all PORT 1 bits for
                                          ;input
```

```
Addr  Label     Mnemonic          Op Code      Comments

3A2             ANL P2,#%10111111  9A BF   ;clear OE2 to enable
                                           ;latch U2
3A4             IN A,P1            09      ;read ch. 1 low byte
3A5             ORL P2,#%01000000  8A 40   ;set OE2 to disable U2
3A7             MOV R0,#1AH        B8 1A   ;point to R26
3A9             MOV @R0,A          A0      ;store neg. range corr.
                                           ;code in R26
        ;
        ;*************************************************
        ;* get channel 2's positive correction code now *
        ;*************************************************
        ;
3AA             MOV A,#%01100111   23 67   ;reinitialize PORT 1
3AC             OUTL P1,A          39      ;pos. range, inputs gnded
                                           ;ch. 2 to comparator
3AD             ORL P2,#%00000001  8A 01   ;enable latch U6 to
                                           ;allow ADC control
                                           ;signals to pass
3AF             ORL P1,#%10000000  89 80   ;start 14-bit conversion
3B1             ANL P1,#%01111111  99 7F   ;clear SC again
3B3             MOV A,#4           23 04   ;wait for conversion to
                                           ;complete
3B5             CALL Delay         54 80
3B7             MOV A,#%00111100   23 3C   ;cut off ch. 2 from com-
3B9             OUTL P1,A          39      ;parator, t-and-h in
                                           ;track mode, power off
                                           ;inputs connected to
                                           ;signal
3BA             ANL P2,#%11111110  9A FE   ;disable latch U6 to hold
                                           ;ADC control signals
        ;
        ;*************************************************************
        ;* read positive range correction code for channel 2         *
        ;* since the hardware offset is small, the important         *
        ;* part of the correction code is in the lower 8 bits.       *
        ;* therefore, only the lower 8 bits are read and stored.     *
        ;* if for some reason the hardware offset is so large        *
        ;* that 8 bits can't represent it, then an error will        *
        ;* occur.                                                    *
        ;*************************************************************
        ;
3BC             ORL P1,#FFH        89 FF   ;set all PORT 1 bits for
                                           ;input
3BE             ANL P2,#%10111111  9A BF   ;clear OE2 to enable
                                           ;latch U2
3C0             IN A,P1            09      ;read ch. 1 low byte
3C1             ORL P2,#%01000000  8A 40   ;set OE2 to disable U2
        ;
        ;*************************************************************
        ;* since the 80C39 doesn't have a subtract instruction,      *
        ;* the positive range correction codes must be 2's           *
        ;* complemented.                                             *
        ;*************************************************************
```

```
Addr  Label     Mnemonic          Op Code     Comments

3C3             JZ Leap1          C6 C8   ;don't 2's complement
                                          ;the 00..00 code
3C5             CPL A             37      ;2's complement
3C6             ADD A,#1          03 01
3C8   Leap1     MOV R0,#1BH       B8 1B   ;point to R27
3CA             MOV @R0,A         A0      ;store in R27
      ;
                                          ;now go back and comp.
                                          ;first corr. code
3CB             MOV R0,#1A        B8 1A   ;point to R26
3CD             MOV A,@R0         F0      ;get ch. 1 pos. range
                                          ;corr. code
3CE             JZ Leap2          C6 D3   ;don't 2's complement
                                          ;the 00..00 code
3D0             CPL A             37      ;2's complement
3D1             ADD A,#1          03 01
3D3   Leap2     MOV @R0,A         A0      ;store in R26
3D4             RET                       ;return
```

APPENDIX E

Development of the Track-and-Hold Droop Error Simulation

Development of the Track-and-Hold Droop Error Simulation

To investigate the effects of track-and-hold droop on the ADC's performance, a software simulation was developed. The simulation models the effects of droop on an otherwise ideal sign-magnitude, successive approximation type ADC over an input voltage range of $\pm$ 5 V. The simulation is simplified by using a 4-bit ADC model. All droop voltages for the 15-bit DAS are normalized to their equivalent LSB values. The LSB values are then used in the simulation with the 4-bit ADC model.

The measured droop rate for the track-and-hold is + 0.8 μV/μs.[2] During the conversion of a sampled voltage, approximately 27 microprocessor instruction cycles take place, with a period of 15 μs per instruction cycle. The first 8 cycles occur during polarity determination. Therefore, at the time that the sign bit is determined and latched, the sampled voltage drifts 8 x (15 μs) x (0.8 μV/μs) = 96 μV, or 0.3148 LSB. After the sign bit is latched, 4 more instruction cycles occur before the beginning of the magnitude determination. So by the beginning of the magnitude determination, the sampled voltage drifts an additional 4 x (15 μs) x (0.8 μV/μs) = 48 μV, or 0.1574 LSB. Then during the time that the SARs are determining the magnitude of the sampled voltage, approximately 15 more instruction cycles occur. This results in an additional drift of 15 x (15 μs) x (0.8 μV/μs) = 180 μV, or 0.5902 LSB. The total drift in the sampled voltage is then 1.0624 LSB.

In the simulation, the sign determination and successive

approximation search are modeled for a 4-bit ADC. The LSB voltage for the 4-bit ADC is 0.625 V for a ± 5 V span. For each input voltage, the output codes which result from conversion by an ideal ADC and by a droop affected ADC are calculated. To calculate the output code of the droop affected ADC, the value of the drooping input voltage must be known at several points in time. These voltages are given below, and are shown in Figure E1.

Voltage at time of sign latch:

$$V_{sign\_latch} = V_{in} + ((0.3148\ LSB) \times (0.625\ V/LSB)) \quad (E1)$$

Voltage at beginning of magnitude determination:

$$V_{start\_conv} = V_{in} + ((0.3148 + 0.1574\ LSB) \times (0.625\ V/LSB)) \quad (E2)$$

Voltage at determination of 1st magnitude bit:

$$V_{s1} = V_{start\_conv} + ((0.5902\ LSB / 3\ bits) \times (0.625\ V/LSB) \times (1)) \quad (E3)$$

Voltage at determination of 2nd magnitude bit:

$$V_{s2} = V_{start\_conv} + ((0.5902\ LSB / 3\ bits) \times (0.625\ V/LSB) \times (2)) \quad (E4)$$

Voltage at determination of 3rd magnitude bit:

$$V_{s3} = V_{start\_conv} + ((0.5902\ LSB / 3\ bits) \times (0.625\ V/LSB) \times (3)) \quad (E5)$$

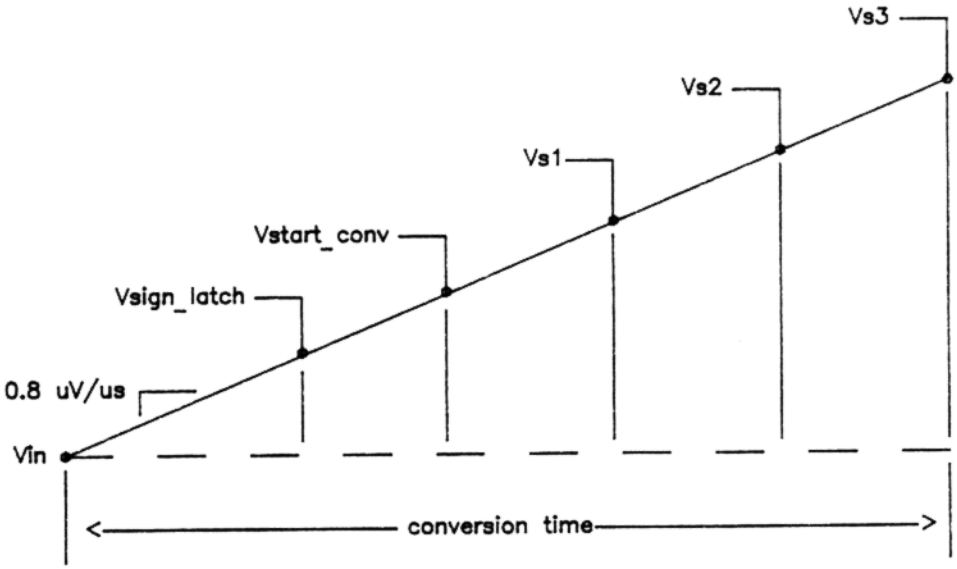


Figure E1. Input Voltage Droop for 4-Bit ADC Model

After the two output codes are calculated for each input voltage, the difference between the droop affected code and the ideal code for each input voltage is found. The difference data can then be displayed, as shown in Figure E2. The plot shows that the maximum error caused by the droop is 2 LSB for the conditions considered here.

A program listing for this simulation, (in the Hewlett Packard 9845 BASIC language), is shown at the end of this appendix. The plotting segment of the program has been left out for brevity.

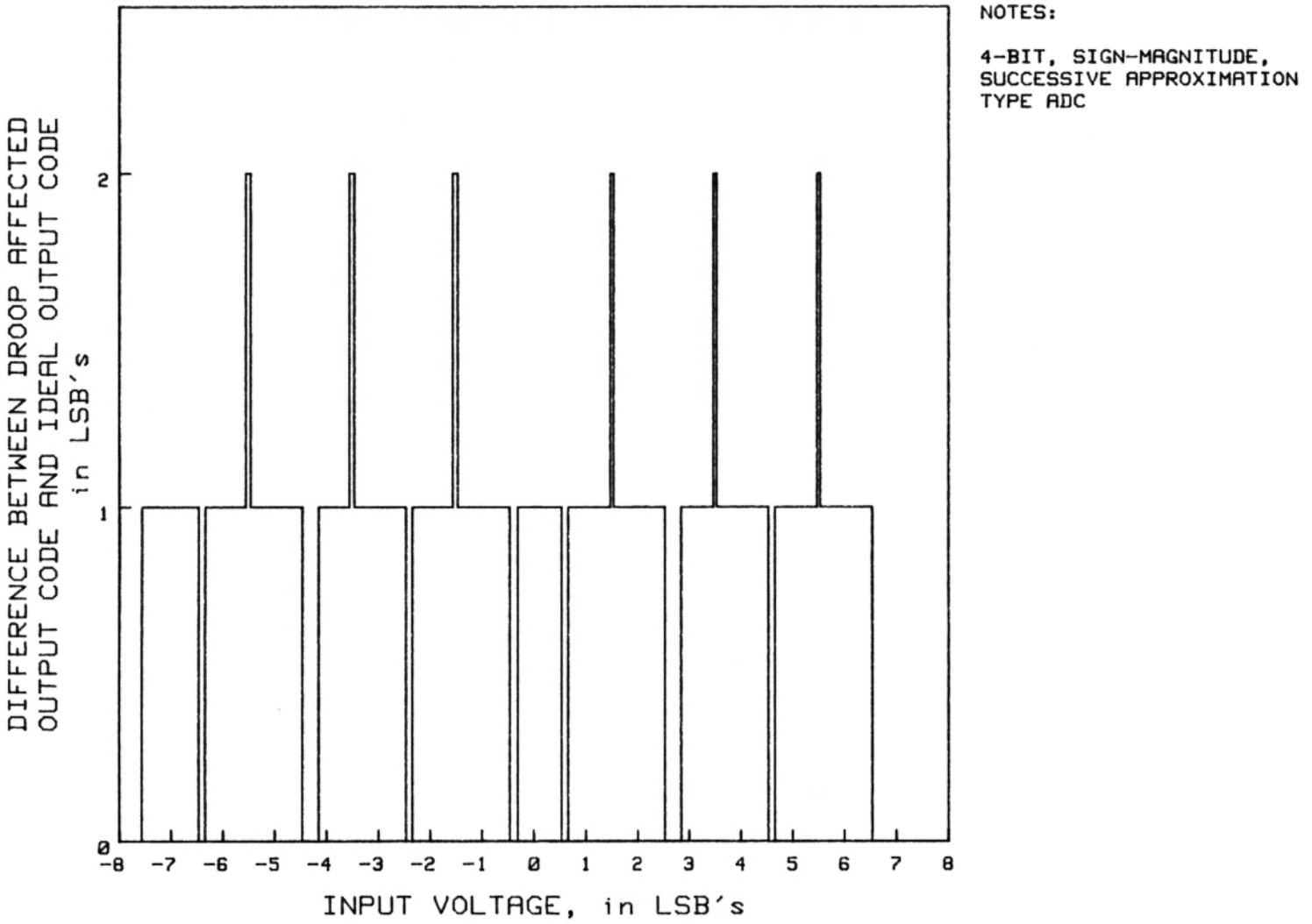


Figure E2. Results of Track-and-Hold Droop Error Simulation

```
10    ! ************************************************************************
11    !                                                                        *
20    ! PROGRAM: DRPSIM                                                        *
21    !                                                                        *
30    ! SIMULATES THE EFFECTS OF TRACK-AND-HOLD DROOP IN A 4-BIT,              *
31    ! SIGN-MAGNITUDE SUCCESSIVE APPROXIMATION ADC.                           *
32    !                                                                        *
40    ! WRITTEN BY JEFF BRADLEY, 3/31/86                                       *
50    !                                                                        *
60    ! ************************************************************************
61    !
62    ! INITIALIZATION
63    !
70    OPTION BASE 0
80    REAL Vin(513),Vref,Vsign_latch,Vstart_conv,Vs(3),Difference(513)
90    INTEGER D(3),Ideal_code(513),Real_code(513),I,J
100   Lsb=5/2^3                          ! VOLTAGE EQUIVALENT TO 1 LSB
110   Droop1=.3148*Lsb                   ! VOLTAGE DROOP AT SIGN LATCH
120   Droop2=.4722*Lsb                   ! VOLTAGE DROOP BY BEGINNING OF
121                                      ! MAGNITUDE CONVERSION
130   Slope=.1230                        ! DROOP SLOPE IN VOLTS PER BIT
131                                      ! BIT DETERMINED, DURING BINARY
132                                      ! SEARCH
133   ! ---------------------------------------------------------------------
134   !
140   ! SIMULATE A CONVERSION OF AN IDEAL ADC
141   !
150   FOR I=0 TO 512
160     Vin(I)=-5+I*.019531              ! INPUT VOLTAGE, STEPS 1/32 OF AN LSB
161       D(2)=1                         ! INITIALIZE THE BITS FOR THE BINARY
162       D(1)=0                         ! SEARCH
163       D(0)=0
170       IF Vin(I)>=0 THEN GOTO Pos1    ! JUMP IF INPUT IS A POSITIVE VOLTAGE
171       !
210       FOR J=2 TO 0 STEP -1           ! BINARY SEARCH FOR NEGATIVE RANGE
220         Vref=-1*(D(2)*2.5+D(1)*1.25+D(0)*.625-.5*Lsb)
221                                      ! CALCULATE THE DAC OUTPUT FOR EACH
222                                      ! TEST
230         IF Vin(I)>Vref THEN D(J)=0   ! TEST THE Jth BIT
240         IF J=0 THEN GOTO L1          ! IF LAST BIT,THEN JUMP
250         D(J-1)=1                     ! SET NEXT BIT
260 L1:   NEXT J
261       !
270     Ideal_code(I)=-1*(D(2)*4+D(1)*2+D(0)*1)
271                                      ! CALCULATE THE OUTPUT CODE FOR
272                                      ! THE NEGATIVE RANGE
280     GOTO L5
281   !
290 Pos1:!
291       FOR J=2 TO 0 STEP -1           ! BINARY SEARCH FOR POSITIVE RANGE
300         Vref=D(2)*2.5+D(1)*1.25+D(0)*.625-.5*Lsb
301                                      ! CALCULATE THE DAC OUTPUT FOR
302                                      ! EACH TEST
310         IF Vin(I)<Vref THEN D(J)=0   ! TEST THE Jth BIT
320         IF J=0 THEN GOTO L2          ! IF LAST BIT THEN JUMP
330         D(J-1)=1                     ! SET NEXT BIT
340 L2:   NEXT J
341       !
350     Ideal_code(I)=D(2)*4+D(1)*2+D(0)*1
351                                      ! CALCULATE THE OUTPUT CODE FOR
```

```
352                                     ! THE POSITIVE RANGE
380   ! ---------------------------------------------------------------------
381 L5:!
390   ! SIMULATE A CONVERSION OF A DROOP AFFECTED ADC
400   !
430     D(2)=1                          ! INITIALIZE THE BITS FOR THE BINARY
440     D(1)=0                          ! SEARCH
450     D(0)=0
451     Vsign_latch=Vin(I)+Droop1       ! INPUT VOLTAGE AT SIGN LATCH TIME
452     Vstart_conv=Vin(I)+Droop2       ! INPUT VOLTAGE AT START OF BINARY
453                                     ! SEARCH
456     Vs(1)=Vstart_conv+Slope*1       ! INPUT VOLTAGE AT 1ST BIT DECISION
457     Vs(2)=Vstart_conv+Slope*2       ! INPUT VOLTAGE AT 2ND BIT DECISION
458     Vs(3)=Vstart_conv+Slope*3       ! INPUT VOLTAGE AT 3RD BIT DECISION
459     !
460     IF Vsign_latch)=0 THEN GOTO Pos2
461                                     ! JUMP IF INPUT IS POSITIVE AT SIGN
462                                     ! LATCH TIME
470       FOR J=2 TO 0 STEP -1          ! BINARY SEARCH FOR NEGATIVE RANGE
480         Vref=-1*(D(2)*2.5+D(1)*1.25+D(0)*.625-.5*Lsb)
481                                     ! CALCULATE THE DAC OUTPUT FOR EACH
482                                     ! TEST
490         IF Vs(3-J))Vref THEN D(J)=0 ! TEST THE Jth BIT
500         IF J=0 THEN GOTO L3         ! IF LAST BIT THEN JUMP
510         D(J-1)=1                    ! SET NEXT BIT
520 L3:   NEXT J
521           !
530     Real_code(I)=-1*(D(2)*4+D(1)*2+D(0)*1)
531                                     ! CALCULATE THE OUTPUT CODE FOR
532                                     ! THE NEGATIVE RANGE
540   NEXT I                            ! REPEAT FOR NEXT INPUT VOLTAGE
549 Pos2:!
550       FOR J=2 TO 0 STEP -1          ! BINARY SEARCH FOR POSITIVE
551                                     ! RANGE
560         Vref=D(2)*2.5+D(1)*1.25+D(0)*.625-.5*Lsb
561                                     ! CALCULATE THE DAC OUTPUT FOR EACH
562                                     ! TEST
570         IF Vs(3-J)(Vref THEN D(J)=0 ! TEST THE Jth BIT
580             IF J=0 THEN GOTO L4     ! IF LAST BIT THEN JUMP
590             D(J-1)=1                ! SET NEXT BIT
600 L4:   NEXT J
610     Real_code(I)=D(2)*4+D(1)*2+D(0)*1
611                                     ! CALCULATE THE OUTPUT CODE FOR
612                                     ! THE POSITIVE RANGE
620   NEXT I                            ! REPEAT FOR NEXT INPUT VOLTAGE
630   ! ---------------------------------------------------------------------
640   !
650   ! FIND DIFFERENCE BETWEEN THE DROOP AFFECTED AND IDEAL OUTPUT CODES
660   !
670   FOR I=0 TO 512
680        Difference(I)=Real_code(I)-Ideal_code(I)
700   NEXT I
701   ! ---------------------------------------------------------------------
702   !
703   ! PLOTTING ROUTINE
704   !
710   ! THE PLOTTING ROUTINE HAS BEEN LEFT OUT FOR BREVITY
711   !
```

APPENDIX F

Development of the Comparator Hysteresis Error Simulation

Development of the Comparator Hysteresis Error Simulation

To investigate the effects of comparator hysteresis on the ADC's performance, a software simulation was developed. The simulation models the effects of comparator hysteresis on an otherwise ideal 3-bit, unipolar, successive approximation ADC with a 5 V span. A 3-bit ADC model is used to simplify the simulation. The desired hysteresis loop width voltage is normalized to its equivalent LSB value by the user. The LSB value is then entered by the user after starting the simulation.

The hysteresis equations used in the simulation were taken from Stout.[10] The equations were derived from a comparator circuit similar to the one shown in Figure F1. The switching characteristic for this circuit is shown in Figure F2. The input voltage Vin shown in Figure F1 is the output of the ADC's DAC/op-amp circuit. The input voltage Vref is the output of the track-and-hold.

The simulation assumes that the correction hardware described in Chapter 3 is in place. The resistor R1 is set to be 200 ohms. This value is the equivalent resistance of four closed analog switches in series. After the desired loop width is specified, the value for the feedback resistor Rf is calculated from

$$Rf = R1 \times (Vomax - Vomin) / Hystloop \qquad (F1)$$

where: Vomax = + 5 V (maximum comparator output voltage)

Vomin = - 7.5 V (minimum comparator output voltage)

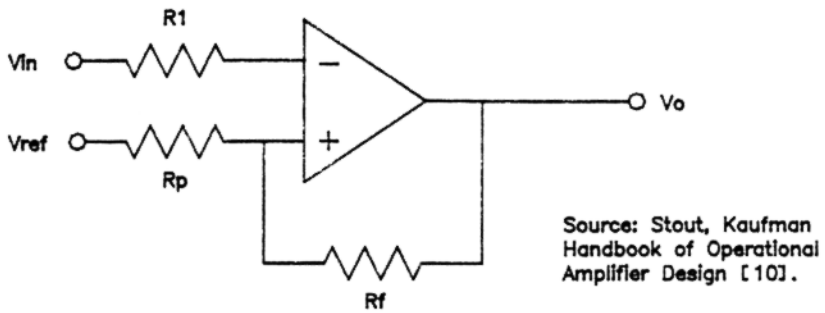


Figure F1. Circuit Diagram for a Comparator with Hysteresis

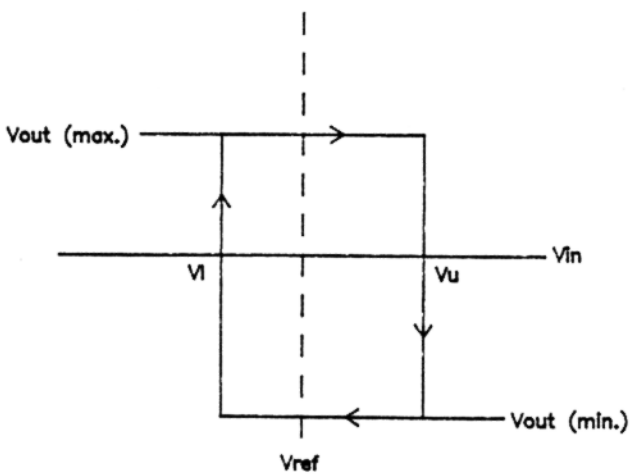


Figure F2. Switching Characteristic of a Comparator with Hysteresis

Hystloop = (hysteresis loop width in volts, calculated for the 3-bit ADC)

Then the value for the resistor Rp is calculated from

$$Rp = Rl \times Rf / (Rf - Rl) \qquad (F2)$$

After the resistor values have been calculated, The operation of the ADC is simulated. The operation of the ADC in this simulation differs from that of an ideal ADC in that the comparator now has two trip points to make comparisons against. The trip point that is used in a particular iteration depends on the state of the comparator output at the end of the last iteration. Since the position of the trip points relative to the reference voltage Vref (output of the track-and-hold) depends on Vref, the trip point values must be recalculated for every new Vref. The upper and lower trip points are calculated from

$$Vu = ((Rp / (Rp + Rf)) \times (Vomax - Vref)) + Vref \qquad (F3)$$

$$Vl = ((Rf / (Rp + Rf)) \times (Vref - Vomin)) + Vomin \qquad (F4)$$

The output code resulting from the simulated analog-to-digital conversion is calculated for each input voltage (Vref). After all the output codes have been compiled, they can be displayed as shown in Figure F3. A program listing for this simulation, (in the Hewlett Packard 9845 BASIC language), is shown at the end of this appendix. The plotting segment of the program has been left out for brevity.

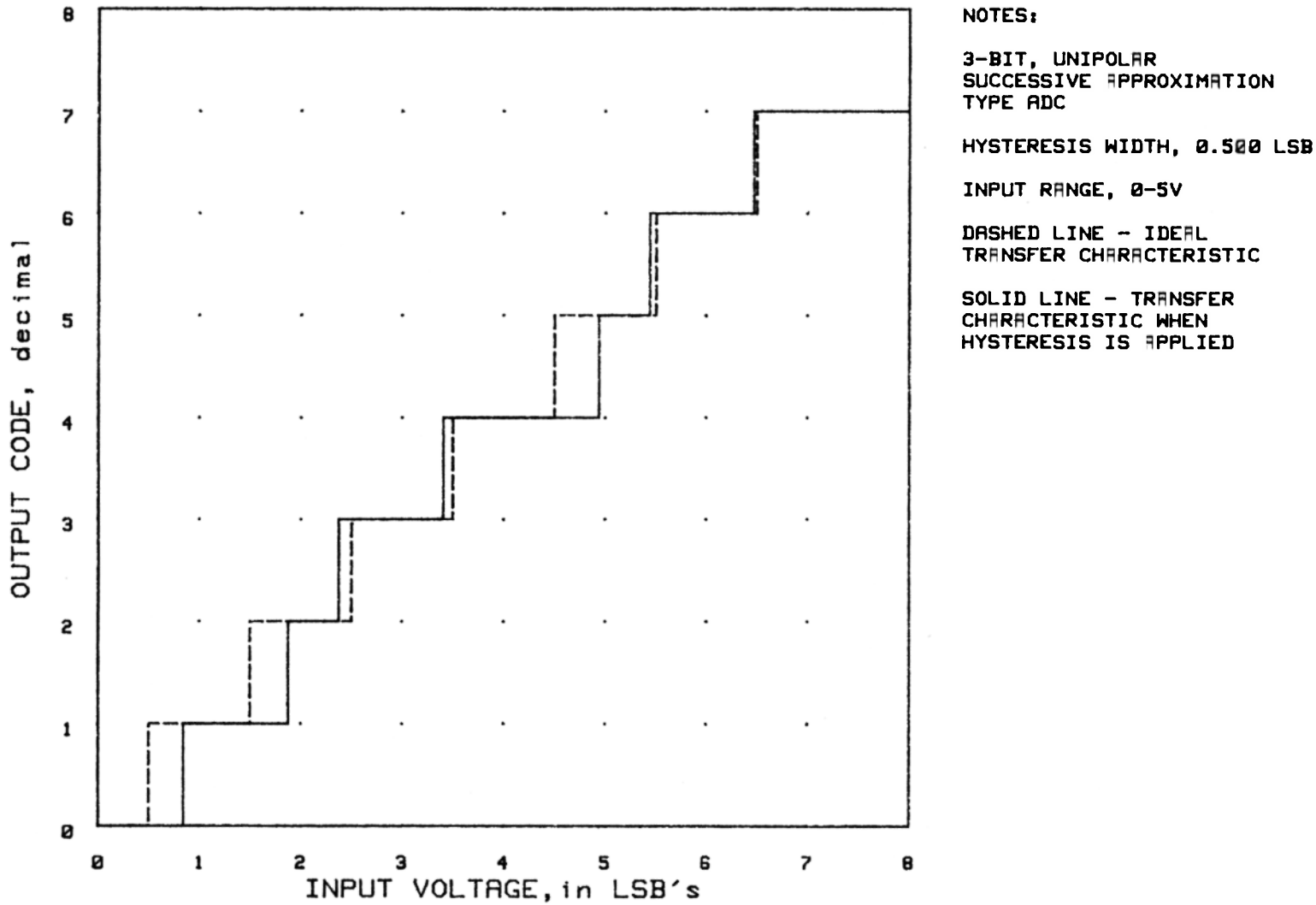


Figure F3. Transfer Characteristic of ADC with Comparator Hysteresis

```
10    ! ************************************************************************
20    !                                                                        *
30    ! PROGRAM:  HYSSIM                                                       *
40    !                                                                        *
50    ! SIMULATION OF A SUCCESSIVE APPROXIMATION ADC WHICH HAS A               *
60    ! COMPARATOR WITH HYSTERESIS.  THIS PROGRAM IS PRESENTLY SET             *
70    ! UP FOR A 3 BIT, 0 TO 5V ADC.                                          *
80    !                                                                        *
90    ! A FUTURE IMPROVEMENT WOULD BE TO ALLOW USER INPUT FOR INPUT            *
100   ! STEP SIZE, INPUT RANGE, AND NUMBER OF BITS.  ANOTHER IMPROVEMENT      *
110   ! WOULD BE TO CALCULATE THE DIFFERENTIAL AND INTEGRAL LINEARITY ERROR.   *
120   !                                                                        *
130   ! WRITTEN BY: JEFF BRADLEY, 3/11/86                                      *
140   ! ************************************************************************
150   !
160   ! INITIALIZATION
170   !
180   OPTION BASE 0
190   REAL Vr(257),Vin,Vu,Vl,Hystloop,Lsb,Lsbloop,Rf,R1,Rp,Vomax,Vomin
200   INTEGER D(3),Compout,I,J,Plotter,Code(257),Xmin,Xmax,Ymin,Ymax
210   Lsb=5/2^3
220   Vomax=5                                    ! MAX COMP OUTPUT VOLTAGE
230   Vomin=-7.5                                 ! MIN COMP OUTPUT VOLTAGE
240   ! ---------------------------------------------------------------------
250   PRINT PAGE
260   PRINT "PROGRAM:  HYSSIM"
270   PRINT LIN(1)
280   PRINT "THIS PROGRAM SIMULATES THE OPERATION OF A 3-BIT SUCCESSIVE"
290   PRINT "APPROXIMATION ADC WITH A 0 - 5V RANGE AND COMPARATOR HYSTERESIS."
300   PRINT "ITS PURPOSE IS TO INVESTIGATE THE EFFECTS OF THE HYSTERESIS ON"
310   PRINT "THE ADC's TRANSFER FUNCTION.  THE HYSTERESIS CIRCUIT AND EQUATIONS"
320   PRINT "COME FROM STOUT/KAUFMAN, HANDBOOK OF OPERATIONAL AMPLIFIER CIRCUIT"
330   PRINT "DESIGN."
340   ! ---------------------------------------------------------------------
350   !
360   ! MAKE CALCULATIONS FOR HYSTERESIS LOOP
370   !
380   INPUT "ENTER DESIRED HYSTERESIS LOOP WIDTH IN LSB's",Lsbloop
390   !
400   Hystloop=Lsbloop*.625                      ! WIDTH IN VOLTS
410   R1=200                                     ! - INPUT RESISTOR
420   Rf=R1*(Vomax-Vomin)/Hystloop               ! FEEDBACK RESISTOR
430   Rp=R1*Rf/(Rf-R1)                           ! + INPUT RESISTOR
440   DISP "MAKING CALCULATIONS"
450   ! ---------------------------------------------------------------------
460   !
470   ! SIMULATED CONVERSIONS
480   !
490   FOR I=0 TO 256                             ! FOR 65 ANALOG INPUTS
500        Vr(I)=I*1.9531E-2                     ! ANALOG INPUT, 32 STEPS/LSB
510        Vu=Rp/(Rp+Rf)*(Vomax-Vr(I))+Vr(I)     ! UPPER COMPARATOR TRIP PT.
520        Vl=Rf/(Rp+Rf)*(Vr(I)-Vomin)+Vomin     ! LOWER COMPARATOR TRIP PT.
530        D(2)=1                                ! INITIALIZE MSB
540        D(1)=0                                ! INITIALIZE OTHER BITS
550        D(0)=0
560        Compout=1                             ! INITIALIZE COMPARATOR O/P
570   ! ---------------------------------------------------------------------
580        FOR J=2 TO 0 STEP -1                  ! S-A CONVERSION
590             Vin=D(2)*2.5+D(1)*1.25+D(0)*.625-.5*Lsb
591                                              ! CALCULATE THE DAC OUTPUT
```

```
592                                                  ! VOLTAGE
610             IF (Vin)=Vu) AND (Compout=1) THEN Compout=0
620                                                  ! CONDITION FOR +/- TRANSITION
630             IF (Vin(=Vl) AND (Compout=0) THEN Compout=1
631                                                  ! CONDITION FOR -/+ TRANSITION
640             IF Compout=0 THEN D(J)=0             ! TEST WHETHER BIT SHOULD BE
641                                                  ! LEFT SET OR RESET
650             IF J=0 THEN GOTO L1                  ! IF LAST BIT THEN JUMP
660                  D(J-1)=1                        ! OTHERWISE SET NEXT BIT
670 L1:     NEXT J
680         Code(I)=D(2)*4+D(1)*2+D(0)*1             ! CALCULATE OUTPUT CODE
690   NEXT I
700   ! ------------------------------------------------------------------------
710   !
720   ! PLOTTING ROUTINE
721   !
722   ! THE PLOTTING ROUTINE HAS BEEN LEFT OUT FOR BREVITY
723   !
```

REFERENCES

[1]Charles R. Ragsdale, "Some Considerations in the Design of a Low-Power, 15-Bit Analog-to-Digital Converter," (MS thesis, Department of Electrical and Computer Engineering, Kansas State University, 1984).

[2]William G. Reed, "The Use of Low-Power Operational Amplifiers in Track-and-Hold Amplifiers," (MS thesis, Department of Electrical and Computer Engineering, Kansas State University, 1984).

[3]Personal research notes of Douglas W. Doerfler, dated 6/21/83 to 9/12/84, (Department of Electrical and Computer Engineering, Kansas State University).

[4]Douglas W. Doerfler, "Techniques for Testing a 15-Bit Data Acquisition System," (MS thesis, Department of Electrical and Computer Engineering, Kansas State University, 1985).

[5]Douglas W. Doerfler, "Program Documentation for Techniques for Testing a 15-Bit Data Acquisition System," (Department of Electrical and Computer Engineering, Kansas State University, 1985).

[6]Motorola CMOS Data, (Austin, TX: Motorola Technical Information Center, 1978).

[7]Intel Microcontroller Handbook, (Santa Clara, CA: Intel Literature Department, 1983).

[8]Jim Williams, "Follow Design Rules for Optimum Use of Fast Comparator IC," EDN. June 13, 1985, p 129.

[9]Paul Brokaw, "An I.C. Amplifier User's Guide to Decoupling, Grounding, and Making Things Go Right for a Change," (Analog Devices, Inc. Application Note, date unknown).

[10]David F. Stout, Handbook of Operational Amplifier Circuit Design, ed Milton Kaufman, (New York: McGraw-Hill, 1976).

[11]Analog-Digital Conversion Handbook, ed Daniel H. Sheingold, (3rd ed; Englewood Cliffs, NJ: Prentice Hall, 1986).

[12]Datel Data Conversion Components, (Mansfield, MA: Datel Intersil, 1983/84).

SOME FURTHER CONSIDERATIONS IN THE DESIGN AND IMPLEMENTATION OF A LOW-POWER, 15-BIT DATA ACQUISITION SYSTEM

by

JEFFREY DARREN BRADLEY

B. S., Kansas State University, 1985

AN ABSTRACT OF A MASTER'S THESIS

submitted in partial fulfillment of the
requirements for the degree

MASTER OF SCIENCE

Department of Electrical and Computer Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1986

Abstract

This thesis discusses several implementation considerations for a low-power data acquisition system (DAS). The system resolution is 15 bits, the sampling frequency is 128 Hz, the input range is $\pm$ 5 V, and the maximum input frequency is 45 Hz. The calculated power consumption is 36.5 mW. The analog-to-digital conversion is performed by a sign-magnitude, successive approximation type analog-to-digital converter (ADC).

The previously designed DAS hardware is reviewed and its performance and major errors are reported. The major error sources are discussed, including the ADC's polarity range mismatch, the track-and-hold's droop, system noise, and comparator related problems. Next a microprocessor-based error compensation technique that corrects for the ADC's polarity range mismatch is introduced. Initial test results for the error compensated DAS are then reported. Finally, recommendations are made for improvement of the low-power DAS in future implementations.

The microprocessor-based error compensation technique was found to reduce the ADC's polarity range mismatch to an acceptable level. However, the improvement in the DAS's performance is not without cost; the power-consumption is modestly increased, the input range is slightly reduced, and the system's software is more complex.