DATA LOGGER USING THE SYM-1 MICROCOMPUTER

by

MICHAEL DEAN SCHWARZ

B. S., Kansas State University, 1978

A MASTER'S THESIS

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Agricultural Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1984

Approved by:

Stanley J. Clark
Major Professor

ACKNOWLEDGEMENT

Special thanks to USDA for funding the Solar Demonstration projects that made development of the data logger possible. Thanks to Pat Murphy, Dennis Matteson, John Anschutz and Vic Robbins who were also involved in the Solar Demonstration projects.

Finally, for their guidance and support, I thank the members of the graduate committee, Dr. Mark Schrock, Prof. Ralph Lipper, Prof. Ralph Turnquist, and my major professor Dr. Stanley Clark.

CONTENTS

INTRODUCTION

The design of the SYM-1 based data logger resulted from two USDA sponsored Solar Demonstration Projects. In these projects a total of 19 solar collectors were to be monitored to determine the energy collected during the heating season. As funds were limited, it was not possible to purchase a commercial data logger for each solar collector so an alternative had to be chosen. The decision was made to build and program a data logger for the project that could be located at each solar collector.

In addition to the solar demonstration projects, several other uses of the data logger were made. These were for the ice freezing project (where the logger also served as the controller) and for engine testing.

LITERATURE REVIEW

In reviewing data loggers there are several types of data loggers available. They range from computer-based data loggers to commercial-dedicated data loggers to custom-built dedicated data loggers.

One type of data logging equipment is the computer-based data acquisition system. This type of system is sold as either a full computer with data acquisition interfaces or as data acquisition interfaces that can be used on an existing computer. In this case the computer is a complete single-user system with computer, terminal and disk drives. A disk operating system is available along with a selection of languages for programming. With this type of equipment, data can be acquired and processed on site with full interaction. This is the most flexible type of data logging system but also the most expensive. Since this system is a complete computer it almost always requires 120 volt AC electrical power for operation. Usually a power outage will completely stop operation and require an operator to restart the operation.

Representive of commercial-dedicated data loggers are the Campbell Scientific CR5 and CR21. The CR21 is built primarily as a meteorological data logger but can be used for various other types of data logging. The logger provides a fixed number of analog channels usable for thermocouples and similar inputs and pulse counting channels for wind speed and direction sensing.

The CR5 is a more general data logger providing expandable modules that can be added to meet the desired application. These systems can store data on either a printed paper tape or a cassette tape. In addition, a modem can be installed to allow remote interrogation of the data collected on the data logger over telephone lines. These data loggers operate on batteries so that it is not necessary that power is maintained for the logger.

The final type of data logger is the custom-built data acquisition system based on a computer. In this type of system a general purpose computer is selected and appropriate data acquisition circuits are added to acquire the data. The computer selected can range from a single-board computer with very limited or no operator input/output capability (such as the Synertek SYM-1) to a computer with full keyboard and monitor (such as the Radio Shack TRS8Ø). The data acquisition hardware is usually inexpensive and custom built for a specific task. Depending on the application, the data loggers can be designed for either battery operation or other electrical power as available.

SELECTION OF THE SYNERTEK SYM-1 MICROCOMPUTER

Several considerations were taken into account in the selection of a microcomputer for the data acquisition system. The required features of the microcomputer were as follows:

1. Ease of interfacing the analog to digital (A/D) converter.
2. Handling of power outages during unattended operation.
3. Reliable method of saving data.
4. Being able to transfer data to the Agricultural Engineering computer.
5. Easily programmable for the data logging task.

The first computer considered was the Radio Shack TRS-8Ø Model I with a cassette interface. In analyzing this system, it was found to satisfy requirements 3, 4, and 5 above. The cassette tape is a reliable storage method for data; interfacing to the Agricultural Engineering computer can be made through the RS232 interface; and Basic would provide an adequate programming environment. Requirement 1, interfacing to the A/D converter, is possible since the address and data lines are externally accessible; however, two or three parallel ports would need to be interfaced before the A/D could be connected and an additional power supply would be needed. Requirement 2, providing the ability to handle power outages, is not an easy problem to handle as the computer provides no EPROM sockets for auto-start operation. The only possible solution allowing for unattended operation is to provide battery backup.

The second computer under consideration was the Synertek SYM-1 single board computer. This computer satisfied requirements 1, 2, 3, and 4 easily. Six parallel ports are provided to allow interfacing of an A/D converter to the computer. Since a power supply must be added to operate the computer, the power supply selected would provide adequate power for both the computer and the A/D converter. Power outage handling is provided with EPROM sockets that can be configured to automatically restart data logging on power-up. A cassette interface provides data storage capability and the RS232 interface provides access to the Agricultural Engineering computer. The last requirement was the programmability of the computer. Since the Department has an Apple II computer which uses the same MOS Technology 65Ø2 microprocessor, programming could be done on the Apple II and then transferred to the SYM-1 in EPROM for execution. Although programming in Assembly language is not as desirable as in Basic, this was not considered a major deterrent.

A third computer that was briefly considered is the SD Systems Single Board Computer which uses the Z-8Ø microcomputer. The evaluation of this computer is similar to the SYM-1 as the computer provides similar features. Although it was felt the Z-8Ø is easier to program, this computer was not used due to the lack of a development system within the Department.

DESIGN OF THE MICROCOMPUTER DATA LOGGER

The development process for the data logger resulted in three distinct versions of the data logger being produced. These versions show a progression in capabilities from that of a simple data logger programmed for a specific task to a more general data logger that can handle a wide variety of data acquisition tasks.

All versions of the data logger use the same microcomputer, the Synertek SYM-1. The primary features of the SYM-1 microcomputer include a 6-digit LED display and hexadecimal keypad; three 6522 versatile interface adapters (VIA's) providing a total of six 8-bit parallel ports for interfacing the A/D converter; and a cassette tape port for reading and writing tapes used for data storage. The cassette interface also provides motor control automatic starting and stopping of the tape under software control.

The peripherals needed for the SYM-1 include a power supply and cassette tape recorder. The Power One HTAA-16W power supply provides 5 volts at 2 amperes and +/- 9 to 15 volts at Ø.4 amperes each. The Radio Shack CTR-8Ø cassette tape recorder was chosen since it is designed for computer use and therefore provided the necessary characteristics for data storage. All the components of the data logger are enclosed in an ABS plastic instrumentation case for protection.

The differences in the three versions of the data logger involve the data acquisition hardware, real time clock hardware/software, and control software. All programming was done on an Apple II-Plus computer with dual disk drives and an EPROM programmer in Assembly language. This worked well as both the Apple II-Plus and the SYM-1 share the MOS Technology 65Ø2 microprocessor. Initially the programs were transferred using the RS232 interface from the Apple II-Plus to the SYM-1 for testing, but as the program grew in size EPROMs were used for testing.

Development System

An Apple II-Plus computer was used as the software development system for the data logger. The Apple II-Plus has two disk drives, 48K Ram and a high speed serial interface. To complete the development system, the Macro Assembler Editor (MAE) from Eastern House Software was purchased. All programming was done using this assembler on the Apple II-Plus.

Initially the object code was down loaded through the Apple II-Plus and SYM I serial ports and then saved on cassette tapes. This allowed quick turn-around for program testing and debugging. Since this requires loading the program from tape and starting execution manually, automatic power-up operation was impossible.

To provide power-up operation on the SYM-1 an EPROM programmer was purchased for the Apple II-Plus. Burning the program into an EPROM allows automatic power-up operation of the SYM-1 and removes the dependency on loading the program from tape. The first programmer, an MP.PP III, had some reliability problems due to bad socket connectors along with the limitations of programming only 5 volt 2716 EPROMs and not being able to read or verify the EPROMs. This resulted in several EPROMs that were incorrectly programmed without indication of such until they malfunctioned. For these reasons the Apparat Apple PROM Blaster EPROM programmer was purchased which programmed a variety of EPROMs, and can also read and verify the EPROM.

Version 1

The first version of the data logger served as an experimental model to verify operating capabilities, resolution, and reliability of the microcomputer data logger. Its design was very specific to the task of taking temperature data every minute from an ice freezing project. Only one data logger was built with this design and was later upgraded to the Version 2 data logger.

Hardware configuration

The analog to digital (A/D) conversion was performed using an ADC1609 A/D converter chip. This chip provides a multiplexor for 16 analog input channels and an A/D converter with eight bits of resolution over the input range of 0 to 5 volts. For temperature sensing, the Analog Devices AD590 with a current to voltage conversion amplifier was used. Resistance of the water and ice mixture was measured with a voltage divider to determine time of freezing.

Software

On this first version of the program, the SYM-1 monitor was used extensively to reduce the amount of programming required. No information display was done from the data logging program, relying on the operator to be able to use the SYM-1 monitor to obtain current data to check operations. A timer in one of the three 6522 VIA's on the SYM-1 was used to provide interrupts at 50 millisecond (20 Hz) intervals. This provided a timing signal used to maintain a software real time clock and allow the processor to perform the data logging. With data logging proceeding as a background operation, the operator could use the monitor to check memory locations and proper operation of the program independently of the data logging.

The operation of the program was to maintain the software clock, sample the data every minute, and store the data in the memory array. Every hour the data array was stored twice on tape to provide duplicate storage in the event there was a defect in the tape.

The program was initially loaded from cassette into ram but later stored in an EPROM. In both cases, someone had to physically start the program, enter the date, and start the data logging. Any power outage would result in the data logger stopping operation and losing data until the operator returned to restart its operation.

Version 2

The second version used the experience gained from Version 1 and resulted in a data logger suitable for taking solar data in confinement livestock buildings. The additional capabilities needed were to make the data logger able to withstand power outages and to continue taking data after the power was turned back on. An operator interface was also needed to allow people without computer experience to obtain information about the current state of input data, time, and general operation.

Hardware

The analog to digital converter used on this data logger was the ADCØ8Ø9 which is an eight channel equivalent of the ADC16Ø9. The temperature sensing circuits were the same, but precision resistors with a low temperature coefficient of resistance were used instead of standard carbon resistors. This was required since the temperature sensors sense absolute temperature and any errors were a percentage of absolute temperature. For example, with a 2Ø degree Celsius temperature (293 degree Kelvin) a Ø.5% change in resistance would result in an unacceptable 1.5 degree Celsius temperature change. To measure air flow, an air flow sensor producing thermocouple level output was amplified by a commutating auto-zero amplifier.

In order for the data logger to handle power outages it needed either to have battery backup or to be able to automatically restart data logging upon power-up. Since a battery backup would be both costly and complex requiring three supply voltages, the automatic restart operation was selected. To automatically restart operation upon power-up requires both an auto-start EPROM and a real time clock since weather data would be of little use without the time of day. The Oki Semiconductor MSM5832 real time clock was used primarily because at the time it was the only microcomputer interfacable, real time clock chip available. This clock maintains year, month, day, hour, minute, second, and day of the week. Also provided by the clock chip are 1/1024th second, 1 second, 1 minute, and 1 hour interrupts of which the hour and minute interrupts usually didn't work as specified in the data sheet. The second interrupt was used to trigger data sampling under the interrupt routine. The clock is a single CMOS chip with a standby power requirement of 2.3 to 5 volts at 30 microamperes. Backup power for this clock was provided by 2 AA carbon cell batteries which provided power for at least 2 years.

An auto-start EPROM was used to provide the data logger with power-up restart capability. The SYM-1 provides jumpers to allow the reset address to be selected from one of the four ROM/EPROMs sockets. These jumpers were configured so that the reset address is obtained from the data logging EPROM which has its own copy of the initialization code after which it enters the data logging routine.

Another method used to measure air flow was with a vane anemometer. This anemometer is an AC generator providing a sinusoidal signal with a frequency and a voltage which are proportional to the air flow. This signal was buffered thru a comparator producing a square wave of the same frequency as the original signal. A pulse counter in the 6522 VIA is then used to count these pulses over one-second intervals.

Software

To provide a better user interface a display routine was developed that would display current data for several channel numbers. When a channel location is selected by keying in a 2-digit hexadecimal number, current data for that channel is then displayed. The first eight channels are the analog data converted to a decimal number representing degrees Fahrenheit. The time of day and the pulse counter channels were added to the end of the analog channels providing output for their values also. The normal state of the display is to cycle thru all the channels leaving each data value on the LED's for two seconds. The date and time are set by selecting special channel numbers that are not normally scanned.

Version 3

This data logger was designed for the solar grain drying demonstration project. In designing this data logger a more general purpose data logger was desired that could more easily adapt to different needs. The major change from Version 2 was the use of a 12-bit A/D converter and Programmable Gain Amplifier. This allows low-level signals such as those provided by thermocouples to be sampled without a special amplifier. Also by having 12 bits of resolution, the need to have a signal use the full range of the input to obtain acceptable resolution was eliminated.

A block diagram of the version 3 data logger is shown in Figure 1. Figure 2 shows the typical input connections to the data logger. A photograph of the data logger is shown in Figure 3.

Hardware

The Burr Brown SDM856JG Data Acquisition System was used for analog input. This chip provides two 8-channel multiplexors, a sample and hold, and a 12-bit successive approximation analog to digital converter. The multiplexors can be configured for either 8 differential channels or 16 single-ended channels. The analog input multiplexors will take a continuous input voltage of ±2ØV and provide reverse biased diodes for static discharge protection. Easy expansion is provided to 32 single-ended or 16 differential channels by the addition of two more 8-channel multiplexors. The A/D converter provides tri-state outputs in three 4-bit sections and will complete a conversion in 4Ø microseconds.

The Burr Brown 36Ø6AG Programmable Gain Amplifier is used to condition the signal from the multiplexor to the sample and hold. Four TTL lines select the gain providing eleven gain whose values are powers of two ranging from $2^{Ø}$ to $2^{1Ø}$ (1 to 1Ø24).

To supply the automatic power-up sequence with some configuration options, a bank of eight switches are interfaced to the

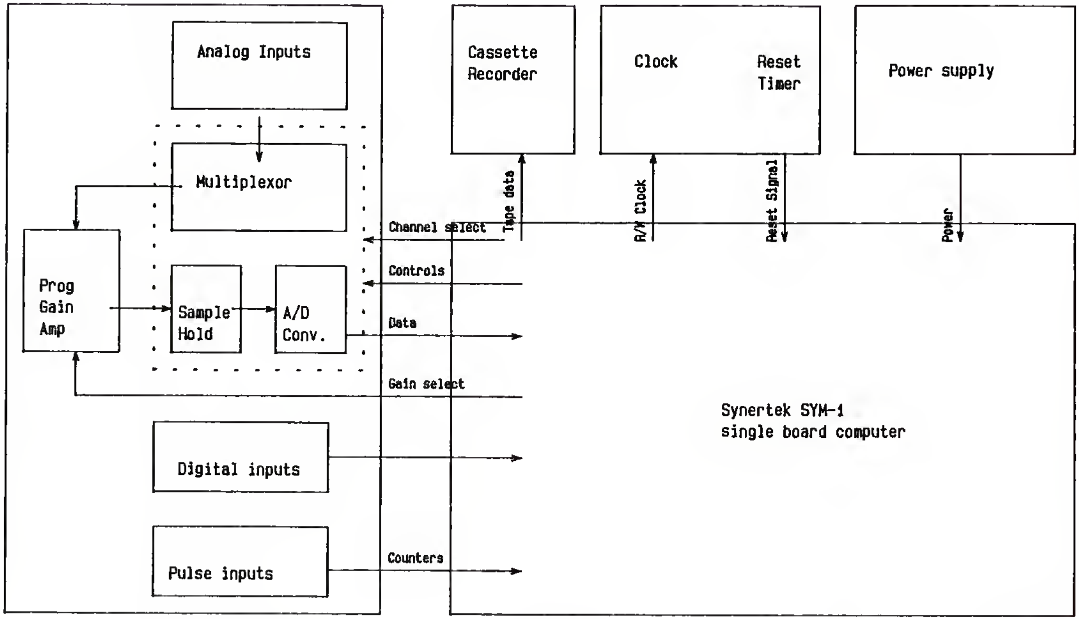


Figure 1. Data Logger Version 3 Block Diagram

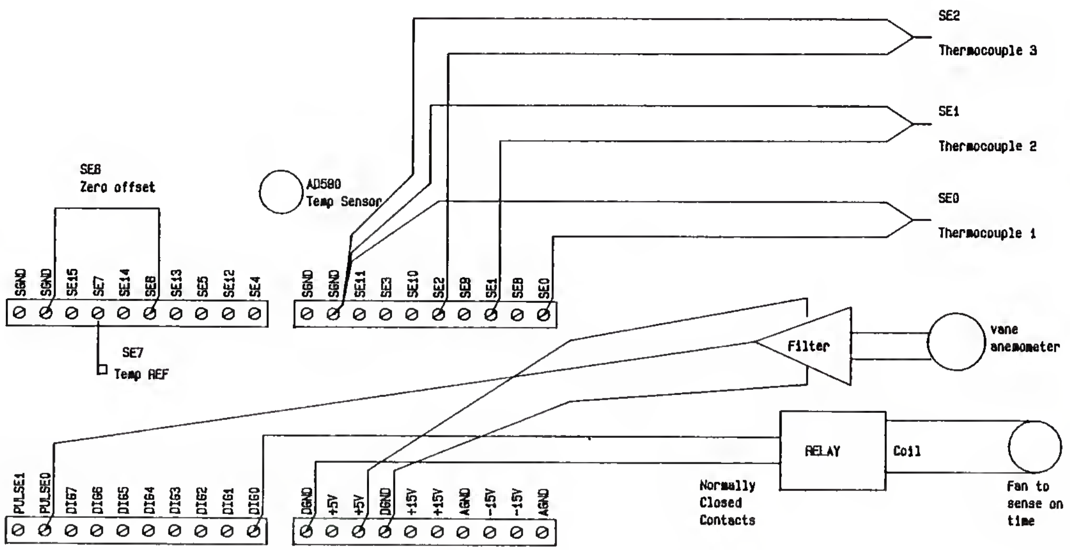


Figure 2. Data Logger Version 3 Example Inputs

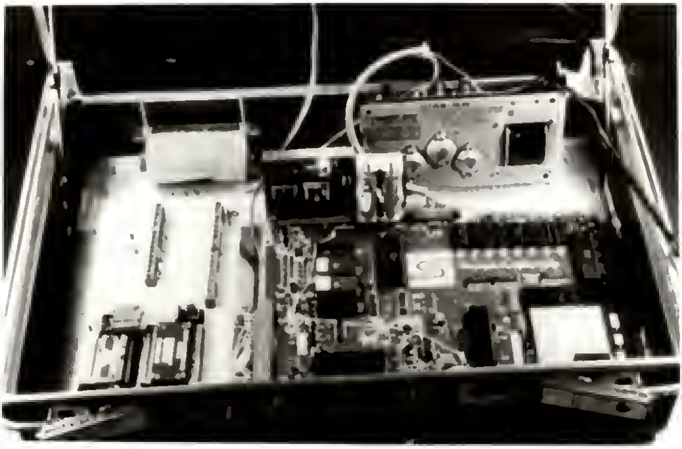


FIGURE 3. SYM-1 Version 3 Data Logger

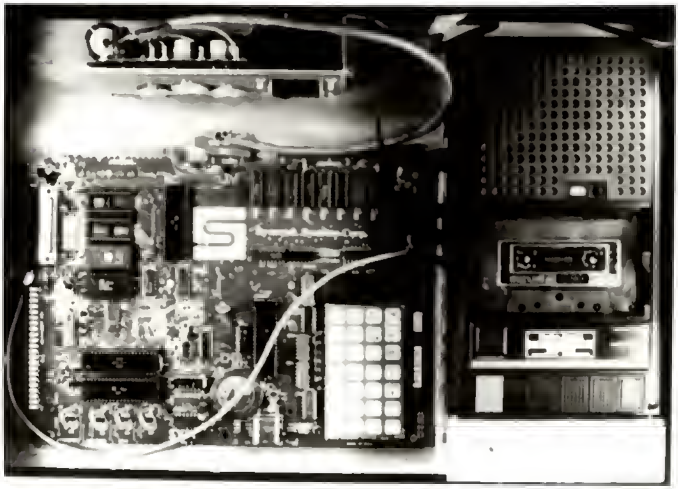


FIGURE 4. SYM-1 Tape Reader

SYM-1. These indicate the number of channels (multiples of 8 analog, 2 pulse and 8 digital) and the sampling frequency. The configuration setting capability gave this data logger much more versatility than the older data loggers.

The 6522 Versatile Interface Adaptor (VIA) has two pulse counters, one of which can be used to count an external signal. Two external pulse-counting inputs are available from the two 6522 VIA's that interface to the A/D board. The pulse counter inputs go directly to the 6522 VIA and therefore require a clean TTL signal to give accurate counts. For example, a single unbuffered switch closure will result in from 1 to 5Ø counts on the pulse counter.

Eight digital lines were left on the VIA's to provide eight digital input channels. These inputs can be used to measure on/off time for devices such as a fan.

The clock interface changed very little from the Version 2 data logger. The main change was to move it to another 6522 VIA that is partially used for the cassette interface. The reset circuit is also added on this circuit. A change was made to the reset circuit to sense cassette save information along with the clock reading. This allows shortening the delay time from 25 seconds (maximum time for a cassette save) to 2 seconds.

Software

As the program for the data logger is approximately 3,ØØØ lines of Assembler code, the program has been designed modularly to make it manageable. There is one large section of the code that provides floating point functions and is assembled separately from the main logger program with only the entry points referenced.

Upon power-up, the reset code is executed which initializes the SYM-1 and jumps to the data logging routine. The main routine initializes memory, stores the interrupt vector, and executes the show routine to display data on the SYM-1 LED display. Each input module has an entry point referenced by show that displays the appropriate values for each channel in that module.

All the data logging activities occur under interrupt control. The clock chip generates an interrupt signal each second causing the SYM-1 to stop what it is doing and enter the interrupt routine. The SYM-1 then reads the clock; calls the every-second routine; determines according to the switch settings whether it is time to sample data, average data, and/or save data on tape; and calls a routine to do each job in order. These routines are short and call the appropriate routines from each module.

The data manager routines define the format of the data array and manipulate it for averaging and saving of the data. The data block consists of a fixed-sized header which includes the logger ID number; the number of analog, digital, and pulse channels; and the coded times for sampling, averaging, and saving data. Following this header are multiple data blocks, the size determined by the number of channels given in the header.

The data blocks consist of the current time, the number of analog and digital samples, the sum of the analog samples, the sum of the digital samples, the number of seconds of pulse data, and total pulse counts. No averaging takes place in the data logger itself leaving the division for later. At the average time interval, the entire array of data blocks is moved up in memory and space is cleared for the next data block. This keeps the current data in a fixed location while the oldest data is erased from memory.

The analog section reads each channel at the sample time interval and adds its value to the summing registers in the data block. The A/D conversion routine selects the current channel, sets its gain, and does the conversion. The auto-ranging code then checks for overflow of which it reduces the gain and retries the conversion. After a good conversion, the converted value is checked to see what gain would be required to use all but the most significant bit. This gain is stored for the next conversion on that channel. Since the programmable gain amplifier has eleven gain levels which are powers of two, floating point data simply becomes the converted value with the negative of the gain as the exponent. The digital channels are implemented as on-time accumulators. For each bit set the appropriate register is incremented at the sample time internal. This input can be used to determine total time or percent of total time.

CASSETTE TAPE READER

The SYM-1 microcomputer used to read the data tapes recorded by the SYM-1 Data Loggers is pictured in Figure 4. The SYM-1 was used because cassette data recording formats are not standard and vary from computer to computer. The reader is designed so that the destination of the data can be any computer that has a RS232 interface such as the UNIX system, Apple II or Radio Shack TRS-80.

The serial interface on the SYM-1 is a software interface and provides no hardware modem or flow control signals. The software serial interface is implemented with one output bit and one input bit. For serial output, a program with precise timing loops puts each bit in succession on the output port. For serial input the reverse is done after receiving a start bit and then the program samples the input for each bit at the precise time. Since each operation takes the full dedication of the microprocessor, transfer in only one direction can occur at a time and any input that occurs while the processor is not ready for input is lost.

The first method of reading data from the SYM-1 was by using the existing SYM-1 monitor program. The SYM-1 provides the capability of using either the on-board keypad or the serial interface as the control and output device and the monitor provides commands to read a file from the cassette into memory and to do a hexadecimal dump of memory.

To get the data from the SYM-1 to the UNIX computer requires the connection of the SYM-1 to the UNIX computer on a serial line. In order to avoid writing a program on UNIX to directly control the SYM-1, a switch box (Figure 5) was built to allow a terminal to control both the SYM-1 and UNIX computer and direct the output of the SYM-1 to either the terminal or to UNIX. The switch box is configured as a loop between the terminal, the SYM-1, and the UNIX computer. When all three devices are in the loop, output from the terminal goes to the SYM-1, output from the SYM-1 goes to UNIX, and output from UNIX goes to the terminal. When a device is switched out, its input is connected to its output bypassing that device in the loop. For example, switching out the SYM-1 sends the output from the terminal to UNIX.

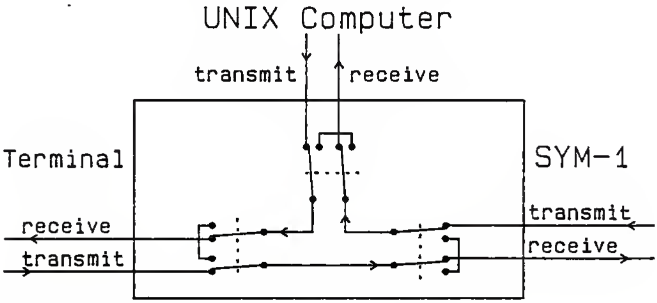


FIGURE 5. RS232 Switchbox.

All signal grounds are connected together.

The procedure to read a cassette file into a file with the switchbox on UNIX is as follows:

1. On the UNIX system (the SYM-1 switched out) edit the desired file and enter the mode to append text to the file.

2. On the SYM-1 (the UNIX switched out) read the file from the tape into memory.

3. Switch both the SYM-1 and UNIX into the loop (output from the SYM-1 goes to UNIX) and give the SYM-1 the command to do a hexadecimal dump on memory.

This process is repeated for each file on the tape.

While this procedure got the files transferred, there were several drawbacks to the procedure: (1) the process required someone to repeatedly control the transfer, (2) the file that is obtained on the computer is in hexadecimal, (3) this procedure requires that the destination be able to accept characters as fast as they are sent. This is not a problem for UNIX but may be a problem for other computers.

The solution to these problems is to write a control program for one computer or both. A single control program on UNIX would not solve the last problem and would need to be rewritten for each destination computer. A single program on the SYM-1 would need specific information about the editor at the destination and therefore also be destination specific. The third alternative is to put a program on both, with the SYM-1 side being as general as possible, but requiring a coordinated program on the destination computer.

Since there is no hardware flow control line available on the SYM-1, there are several software flow control features available in the reader program. The flow controls implemented do not include a conventional control such as X-ON/X-OFF since that cannot be achieved but instead include delays and request characters. Delays are available and can be programmed between every character and between every line. Request characters can be sent by the destination to trigger the transmission of each line or file, similar to ENQ-ACK. Because the SYM-1 serial interface is a software program and can miss input when it is not ready, such as in reading a file from tape, a character is sent every second while SYM-1 is expecting a request character. Each of these control parameters are programmable on the SYM-1 reader, with several configurations pre-programmed in a table.

The manual method resulted in a transfer of hexadecimal files in which the data was usually binary and needed to be transformed into ASCII format to be usable in either a program or for visual scanning. For the Version 3 Data Logger, the SYM-1 reader is programmed to interpret the data, eliminate redundancy, and transfer the data in ASCII character representation suitable for direct interpretation. This eliminates one step in the handling of the data that would otherwise have been even more complex with the use of floating point in Version 3. There is also a hexadecimal dump output available in this reader for the older data loggers and other data loggers that may be programmed for other data formats.

APPLICATION OF THE DATA LOGGER

The applications for which the data loggers were built were the two USDA Solar Demonstration Projects. Both Version 2 and Version 3 were used in these projects and each performed well. The data loggers were also used in an ice freezing project and in engine performance data logging. These projects indicate the versatility of the data logger in applications for which they were not intended.

Solar Livestock Demonstration

Version 2 Data Logger was designed and built for the USDA Solar Livestock Demonstration Project in which nine solar collectors were instrumented and data collected for two heating seasons. All the data loggers were located inside a confinement swine building, usually in an entry room.

Although there were a few problems with the data loggers, they collected data and generally operated maintenance free. A major problem was the occurrence of an ice storm in December of the first project year. This ice storm caused power fluctuations and returned power to the SYM-1 before it had completely powered down. On seven out of nine data loggers state wide this caused the computer to miss the power on reset signal and left the computer in an indeterminate state. On five of these the cassette recorder was left on. This resulted in the tape being used up so that no more data could be taken when power was finally returned properly. This prompted the design of the watch-dog timer that monitors correct operation of the computer and will generate a reset signal within a few seconds if nothing happens. Another concern was the corrosive environment in a confinement swine building. A few of the data loggers showed corrosion on the solder connections and traces, but the major problem was the metallic parts of the instrument cases. There were no failures in the two years of use due to any corrosion.

Solar Grain Drying Project

The Version 3 data logger was built for the USDA Solar Grain Drying Project in which solar collectors were instrumented for one drying season and possibly alternate use during the winter. These data loggers were generally located near grain drying bins and not near a building in which the data logger could be located. The first attempt at an outside installation was to use two plastic trash bags to keep out moisture. This didn't work as moisture would condense inside the case during the daily temperature fluctuations. Although the condensation did not cause failure, about a cup of water accumulated after a rain storm which did cause failure. After the logger was dried out and cleaned it continued to operate properly.

The major problem with the data logger was that 50% of the Burr Brown data acquisition system chips either didn't work initially or failed during initial operation. Although these failures were replaced (in four to six weeks) by Burr Brown there were frustrating. There were also human factor problems with the data loggers getting unplugged or circuit breakers being turned off inadvertently.

Ice Freezing

All three versions of the data loggers were modified for use on an ice freezing project to freeze ice during the winter. Each of these loggers controlled the ice freezing as well as taking data on the process. The control signals generated were two digital outputs to turn on a fan and water valve. These signals were both based on the current temperature.

The first and second versions of the data loggers required modification of the existing data logging code to obtain the appropriate control capability. Because of this need to modify several parts of the code, development and debugging time was longer than for Version 3. For the Version 3 data logger, the only changes needed in the logging code were to install two subroutine calls into the control code. The control code then used existing data already sampled, calculated the outputs, and stored them in the appropriate locations.

The main problem in the ice freezing controllers was caused by the control relays. Although bypass diodes were installed on the relays, occasionally a spike would be generated that was large enough to put the computer into an indeterminate state. In the second and third version, the watch-dog timer was used which reset the computer in the event of a shutdown of this type, allowing normal operation to continue.

Engine Performance Logging

The SYM-1 data logger was used to monitor engine performance in alternate fuels research. This use of the data logger was possible only through the versatility of the Version 3 Data Logger. Whereas long-term logging of solar collectors required continuous hourly sampling, engine performance needs data sampling at 4 second intervals during the test periods with no sampling during stabilization of the engine. All these requirements had been built into the data logger so no modification was needed for use in data logging. The only complication was is that the data logger is set at power-up to automatically save data at regular time intervals. This must be defeated to prevent logging a lot of data that is not needed.

The initial problem in using the data logger for engine testing was that the A/D converters were failing with no apparent cause. This was later attributed to having active high-level inputs to the A/D with the power off on the data logger.

DISCUSSION

Programming

The programming of the data logger was done entirely in Assembly language and resulted in approximately 3000 lines of code for Version 3. The investment in time for this programming is only worthwhile when there are several units that will use the same code, in this case for 10 loggers. This results in an equivalent of 300 lines of code for each logger which is a reasonable amount of code to write for a data logging task. In order for this type of programming to be useful for a fewer number of data loggers the task it is to perform must be very simple or a higher level language must be used.

With the availability of larger EPROMs and high-level languages such as C and Fortran it is now feasible to use high-level languages for programming. By using a high-level language, other advantages would also be realized. These would be more easily modifiable code and the ability to add features that are not worth the programming effort in Assembly language.

Data Acquisition Hardware

The 12 bit analog to digital converter is needed in any flexible data acquisition system. In Version 3, a single chip includes the multiplexors, sample and hold, voltage reference, analog to digital converter, and the control circuitry. This is therefore an expensive chip as it includes all the components needed for data acquisition. The problem with this system is that a failure of one component such as the multiplexor then requires the replacement of the whole circuit. In our experience the weakest part of the Burr Brown chip was the voltage reference, resulting in the replacement of a large component due to the failure of a simple component. By using multiple chips the individual chip cost, and therefore replacement cost, is reduced with very little added complexity. Interfacing the various chips in the system requires only the connection of the outputs of one chip to the inputs of the next chip.

The programmable gain amplifier is a very powerful device used on this data logger. It effectively extends the 12 bit analog to digital converter by the gain of the amplifier, in this case 1024 or equivalent to 10 bits. This allows simultaneous use of both low-level 0 to 5 millivolt signals such as thermocouples and high-level 0 to 5 volt signals such as presure transducers. In working with a mixture of devices there is a slight effect from the 5 volt signal on the low-voltage devices on the same multiplexor; so for full accuracy, signals should be divided among the channels according to full scale range.

A problem that showed up on the programmable gain amplifier after it was in use was the zero offset values. There is offset adjustment on the amplifier, but in using thermocouples the residual offset and temperature dependent offset can be significant. To correct for this a channel is connected to ground and subtracted from the thermocouple readings. For higher voltage inputs this offset is insignificant and the subtraction isn't needed. The problem appeared in reading a signal ranging from 270 to 310 millivolts. This signal needs the full resolution to achieve accuracy with the 270 millivolt offset voltage. In reading this signal two different amplifier gains are used by the auto-ranging software code and it was observed that the different gains have different offset voltages. Since the channel tied to ground is always sampled at the highest gain, it was not obtaining the true offset seen by this signal. To correct this the zero channel needs to be read at the same gain as the data channel and subtracted at sample time instead of later.

Cassette Tape Reader

There were some problems in reading the tapes written by the data loggers. Both the Radio Shack CTR-80 and CTR-80a tape recorders were used on the data loggers. The newer CTR-80a is much more reliable than the CTR-80 although both worked. The major problem was in the head alignment of the writing and reading recorders. If there was any difference in head alignment then the tape would not read properly until the head alignment was adjusted. Also interesting was the fact that the optimum volume level was different on the two recorder models.

RECOMMENDATIONS FOR FUTURE WORK

In designing a microcomputer-based data logger the two tradeoffs that must be balanced are hardware and software cost. In general the lower the cost of hardware, the more time that is required for programming. In the design of these data loggers the hardware cost was kept to a minimum at the expense of software cost. There have also been several developments that make designing and building a data logger simpler than before. The recommendations for a new data logger of this type are as follows:

1. Unless there is evidence of much higher reliability, use separate multiplexors, sample and hold, and analog to digital converter chips instead of the all-in-one type. The only real advantage of the all-in-one is that the control circuitry is included which can easily be handled in the microcomputer.

2. Put the terminal connectors on a separate board attached to the analog to digital converter board with a ribbon cable. This allows more flexible packaging of the system. Also by adding 3 or 4 channel select lines and locating second stage multiplexors on the connector board, up to 128 differential channels could be selected.

3. Possibly select a newer single board computer using a 16-bit microprocessor. This would greatly simplify programming of the data logger.

4. Select a development system that provides a high level language that can be used to program the microcomputer. For example a C compiler is now available for the Apple II and cross compilers are available for other computers. A high level language would produce larger code which must be allowed for but the development time would be much faster.

5. Use CMOS battery backup RAM for configuration storage and possibly data storage. This would add even more flexibility to the data logger.

6. Add a serial port that can be used for remote modem interrogation or possibly real time access by another computer.

SUMMARY

The data logger using the SYM-1 microcomputer collected data on solar collectors as well as other projects. This data logger was built for approximately one-third the hardware cost of comparable data loggers. The total parts cost of the data logger was $750 compared to over $2000 for commercial data loggers. Software development required approximately 1000 hours to program and debug the data logger.

Three versions of the data logger were designed and built during the two solar demonstration projects. These data loggers show a progression in capabilities from a simple data logger programmed for a specific task to a general purpose data logger. This was demonstrated by the use of the data logger in ice freezing and engine performance testing projects.

The 12-bit analog to digital converter and programmable gain amplifier provide analog input ranges such that both low level and high level sensors can be used concurrently. The real time clock provides time of day for time stamping of the data and allow automatic power-up operation where time is needed with the data. A watch-dog timer keeps the data logger function properly during power fluctuations that might otherwise cause the data logger to malfunction.

Another SYM-1 was used as a cassette tape reader to read the tapes produced by the data logger. The data is read from tape and then transmitted to another computer using the RS232 interface. Several flow control options are available to control the flow of data from the SYM-1.

CONCLUSIONS

Building a data logger using a single board microcomputer and readily available data acquisition components is an alternative to purchasing a commercially available data logger provided several data loggers are to be built alike. This can be for a lower cost while providing greater performance.

The data logging software is not a small programming task and represents the single largest cost item for the data logger. To be feasible the software cost must be spread over several units.

The Burr Brown Data Acquisition System and similar devices do not the provide much of an advantage over separate components. They are actually 4 or 5 separate devices on a single chip that must still be connected together for operation. There is little or no cost savings by using the single chip, while if the multiplexor goes bad the entire $175 package must be replaced instead of a single $12 multiplexor chip.

A common practice to correct for amplifier offset is to read an input channel that has its positive and negative inputs tied together and substracting the reading from the other channel. With the programmable gain amplifier this must be modified as there may be different offsets for each gain.

For continuous unattended operation of a computer, special considerations must be taken to properly handle power losses and fluctuations. The watch-dog timer provided for reliable recovery from power supply problems. After the watch-dog timer was installed there were no malfunctions due to power variations.

In programming for continuous operation where there may be power problems, special care must be taken to verify correct operation. There were instances where a memory location would get changed by some random event and result in improper logging operation. The software should provide extensive checks to verify consistency of operation and these should be run periodically rather than only upon reset.

REFERENCES

Anschutz, J. A. 1981. Ice Storage for Summer Cooling. Report of the 55th and 56th Annual Meeting of the Kansas Committee on the Relation of Electricity to Agriculture 1979-81:64-67.

Anschutz, J. A., M. C. Schwarz and J. P. Murphy. 1983. Kansas Final Report Solar Grain Drying Demonstrations. USDA Agreement No. 12-05-300-509 KSU Project No. 60810-0816.

Jones, D. L., V. M. Meyer, D. J. Korslund and F. C. Vosper. 1980. Microcomputers for Solar Demonstrations. ASAE Paper 80-5001.

Jordan, K. A. 1979. Information on data logger using the SDK-85 microcomputer. (Private Communication).

Jordan, K. A., V. J. Johnson, P. J. Stadick and K. A. Janni. 1980. Methods of Data Logging with Microcomputers. ASAE Paper 80-5527.

Murphy, J. P., M. D. Schwarz and F. V. Robbins. 1982. Kansas Final Report Solar Heating of on Farm Livestock Shelters. USDA Agreement No. 12-05-300-421 KSU Project No. 60810-0810.

Robbins, F. V., J. P. Murphy and M. D. Schwarz. 1983. Solar Energy for Livestock Shelters Kansas Demonstration. ASAE Paper MCR83-117.

Schrock, M. D., and S. J. Clark. 1983. Butanol-Acetone Blends as S.I. Engine Fuels. Transactions of the ASAE 26(3):723-727.

Schwarz M. D., D. Matteson, and J. P. Murphy. 1981. A Data Logging System Using a SYM-1 Microcomputer. ASAE Paper MCR81-512.

APPENDIX A. Data Logger Users Manual

USERS MANUAL FOR THE SYM-1 BASED DATA LOGGER VERSION 3 AND READER VERSION 2

BY

Michael D. Schwarz

1. SYM BASED DATA LOGGER

1.1 HARDWARE

1.1.1 Input connections Input connections to the data logger are made thru screw type terminal blocks on the interface board. For ease of use, a wire wrap pin is connected to each terminal pole to provide alternate on-board connections. The terminal blocks are arranged in 2 or 3 rows of 2 terminal blocks each. For discussion, the rows are numbered 1, 2, and 3 starting with the one closest to the SYM-1, and the blocks will be lettered A and B from the edge of the board. This number system corresponds to the assembly drawings showing the bottom view of the interface board. Due to availability of parts, a mixture of 8, 10, & 12 pole, terminal blocks were used. Therefore the actual configuration may be sightly different from that described here.

The first row contains power supply connectors and digital inputs to the SYM-1. Terminals 1A1 to 1A10 contain the power supply connections in the following order starting at 1A1:

A_GND, -15, -15, A_GND, +15, +15, D_GND, +5, +5, D_GND

A_GND is the analog ground and D_GND is the digital ground. Although these grounds are connected, they should be used separately to prevent ground loops and digital noise on the analog ground. The analog ground has been connected to pole 1A1 on this row to reduce the digital noise on the analog power supply lines.

The next two poles, either 1A11 & 1A12 or 1B1 & 1B2, are pulse counting inputs. The pulse counting inputs will measure either event counts or a frequency input up to 60kHz. At the higher frequencies some time periods may be discarded due to overrun during cassette saves. The pulse inputs are digital signals from 0 to 5 volts. This signal should be a clean signal with no noise as very fast noise due to contact bounce will be detected. For example, a simple switch may give as many as 30 counts on one switch closure. Signals may need filtered and/or buffered to eliminate noise for pulse inputs.

The last 8 poles, numbered 1B1 to 1B8 or 1B3 to 1B10, are digital inputs to the SYM-1. The digital channels are single bit channels sampled as either on or off. The channels are sampled at the sample time interval (normally once a second) and each of the counter registers contain the sum of the number of times the channel is on. This supplies data for slow devices such as a fan on percentage of on time.

Row 2 contains 16 analog inputs to the multiplexors. The input connectors are organized as follows:

POLE:	2A1	2A2	2A3	2A4	2A5	2A6	2A7	2A8	2B1	2B2	2B3	2B4	2B5	2B6	2B7	2B8
DIFF:	0H,	0L,	1H,	1L,	2H,	2L,	3H,	3L,	4H,	4L,	5H,	5L,	6H,	6L,	7H,	7L
SING:	00,	08,	01,	09,	02,	10,	03,	11,	04,	12,	05,	13,	06,	14,	07,	15

Any terminal poles numbered 9 through 12 are connected to the signal ground used for single ended inputs. Using A_GND or D_GND for the signal ground will result in a bias voltage of several millivolts if used as the signal ground. Although this bias exists it may be desirable if current sensors are used along with low voltage inputs, and the currents produce significant voltage drops.

An AD590 integrated circuit temperature sensor is on the board near the analog input connectors. This temperature sensor measures absolute temperature without a compensating junction and can be used as a reference temperature for thermocouples. The output of the on-board temperature sensor is a wire wrap pin near pole 2B7. This may be wire wrapped to any channel desired, with the low side tied to signal ground for differential mode.

Due to offset voltages in amplifiers, there exists a voltage offset of +/-2 millivolts imposed on the inputs that cannot be nulled. For low level inputs this is significant and should be compensated for by tying an input channel to the signal ground. The reading on this channel will be the offset voltage, which can then be subtracted from all channels during data processing.

Row 3, if installed, is identical to row 2 except for differential channels 8 through 15 and single ended channels 16 through 31.

1.1.2 Calibrating the analog to digital converter and amplifier.

1. The amplifier is calibrated first using a digital volt meter. It may be desireable to add a wire wrap pin connected to the amplifier output for easy access to this signal. Before beginning the calibration adjust the pots on header3 to center positions, and set the SPDT switch on the clock board to the up position.

   Set the SYM-1 to a fixed gain of 1 by entering a $80 in memory location $40; and disabling interrupts by entering $7F at $A00E. Do this by entering `CR' `MEM' 4 0 `CR' 8 0 `CR' `MEM' A 0 0 E `CR' 7 F `CR' `GO' `CR'. If there are problems consult the SYM-1 refrence manual on the memory command. Set the SYM-1 to display channel 0 and then proceed with the amplifier calibration.

   With the input to channel 0 tied to signal ground, adjust pot 1 (pot on pin 1 and 2 of header3) until the output of the amplifier reads as close to zero as possible. This should be done using a volt meter since the A/D has not been calibrated yet.

   Reset the SYM-1 so that maximum gain is selected by entering $8A into location $40 as above. Adjust pot 2 (pot on pin 3 and 4 of header 3) until the output of the amplifier reads as close to zero as possible.

2. Calibration of the analog to digital converter should proceed after the amplifier is calibrated. The SYM-1 should be set for a gain of zero by entering $80 in location $40.

   Zero adjustment is made first with the input left at zero as above sampling analog channel zero. Adjust pot 3 (pot on pin 5 and 6 of header3) until the SYM-1 reads 0 on the display.

   Adjust the gain by putting a known voltage to the input channel (The 5 volt power supply is usually below 5 volts at the interface board and works well). Knowing the SYM-1 reading and tbe actual reading from a volt meter adjust out twice the voltage difference. EX: The SYM-1 reads 4.7V and the voltmeter reads 4.75V. Adjust pot 4 so the sym reads 4.8V (4.7 + 2*(4.75-4.70)). The actual overadjustment should be ((5 + volt reading)/5). Go back and readjust the zero point as it will be off after tbis adjustment. The zero and 5 volt readings should converge on the proper values within about two adjustments.

3. An alternate adjustment is available if a voltage source or differential inputs are available. This adjustment is a replacment for tbe adjustments of pots 3 & 4 above.

   Instead of using zero volts in 3 above use -4.998V (-4.5 to -4.99 will also work ok). Adjust pot 3 until the SYM-1 reads the negative voltage.

   Now apply +4.998V (4.5 to 4.99) to input 0. Adjust the gain pot (4) to read the applyed voltage. This calibration is similar to the calibration given in the SDM856 data sheet.

1.2 SOFTWARE

The data logging program is set to auto start on reset or power up. This makes the logger able to deal with power outages reasonably. Any time power is applied or reset is pressed the logger restarts data logging. All memory is initialized unless the check variables are identical to the expected values, assuming memory has not been changed. After tbe initialization the program starts operation as if nothing happened. The data in memory that hasn't been saved (up to 3 bours max) will be lost during a power outage. No data on the tape can be lost.

1.2.1 Switches Switches have been included to give some configuration information to the data logger when it is powered up. The settings determine the number of analog channels and sample timing.

Switch useage

switch	Use
1	Not Used (leave off)
2,3	Number of analog channels
4	Digital channels
5	Pulse channels
6,7,8	Sample times

The analog channels are set using switches 2 and 3 as follows:

ANALOG CHANNELS

Switches		Number of
3	2	Channels
OFF	OFF	8 (0- 7)
OFF	ON	16 (0-15)
ON	OFF	24 (0-23)
ON	ON	32 (0-31)

The digital channels are selected by switch 4. When switch 4 is on, 8 digital channels are selected and will be sampled at the sample time interval. When switch 4 is off, no digital channels will be recorded or sampled.

Pulse channels are selected by switch 5. Two pulse counting channels are available and are sampled when switch 5 in on. When switch 5 is off, no pulse channels are sampled.

The sample times are determined by switches 6, 7, & 8. These switches determine the sample time interval, and data averaging time interval. Sample and average times are as follows:

Time Interval Selection Switches

Switches			Time Interval	
8	7	6	sample	average
OFF	OFF	OFF	1 SEC	1 SEC
OFF	OFF	ON	1 SEC	3.75 SEC*
OFF	ON	OFF	1 SEC	15 SEC
OFF	ON	ON	1 SEC	1 MIN
ON	OFF	OFF	1 SEC	3.75 MIN*
ON	OFF	ON	1 SEC	15 SEC
ON	ON	OFF	1 SEC	1 HR
ON	ON	ON	3.75 MIN*	3 HR

*These values are not exact but are rounded up to the next integer.

Internal representation of the time intervals is a single byte number from 0 to 12. This number is approximately the log base 2 of the time interval. This internal number appears in two places: the operating display and the data printouts from the reader. Time interval numbers and actual time intervals are as follows:

Internal Time Interval Representation

Internal number		Actual time	
Hexadecimal	Decimal	interval	
0	0	1	SEC
1	1	1.875	SEC
2	2	3.75	SEC
3	3	7.5	SEC
4	4	15	SEC
5	5	30	SEC
6	6	1	MIN
7	7	1.875	MIN
8	8	3.75	MIN
9	9	7.5	MIN
A	10	15	MIN
B	11	30	MIN
C	12	1	HOUR
D	13	1.5	HOUR
E	14	3	HOUR

The time intervals that have fractional parts are not constant time intervals but are rounded up to the nearest integer. For example, the 3.75 second interval should trigger at 0, 3.75, 7.5, 11.25, 15, 18.75, etc. second marks. Actual triggering times are at 1, 4, 8, 12, 15, 19, etc. second marks respectively.

The save time interval is calcuated using the following constraints:

1. All memory available is allocated and the maximum number of samples that can be held is calcuated. From this the longest save time interval is calcuated to save all data twice.

2. The maximum time interval is set as the maximum of 1 hour and the average time interval.

3. The save time interval is then set as the minimum of the intervals from 1 and 2 above.

4. The memory saved on tape at the sae interval is adjusted so that each data sample is saved no more than four times. This reduces tape useage when long average intervals are selected.

1.2.2 SYM-1 Operating Display The display during data logging operations is used to show current channels of the inputs to the data logger. The display normally cycles every 2 seconds and displays the next channel, wrapping around to the first channel when the last channel is reached. Channels to be displayed are divided into sections with related channels. Sections are numbered from 0 and the channels within the sections are also numbered from 0 with one exception.

Several methods are provided to change the display to allow more flexible use of the data logger in use and troubleshooting. The data logging process proceeds independently of the display program, and is unaltered by the manipulating of the display. There are six general control keys that manipulate the display. The `GO' key will cause the display to cycle in the normal default method. The `REG' key will cause the display to cycle only on the current section instead of all sections. The `MEM' key will stop the display entirely and redisplay the current value every .5 seconds. This also occures as the result of any other key except the `GO' and `REG' keys. The `->' key will advance to the next channel in the current section. The `<-' key will backup the display one channel in the current section. The `CR' causes the display program to jump to the SYM-1 monitor. This is not for general use but can be useful in debugging and checking out the interface boards. To resume the display program from the monitor, press `GO' `CR' or `RST'. `RST' should only be used as a last resort as some data may be lost by using `RST'. If that doesn't work then power off-on cycling is the last resort.

The general format to select a display channel randomly is to enter: `section number' `-' `channel number' `CR'. The format can be shortened to `channel number' `CR' if the desired section is the current one.

<u>Display Sections</u>.

0 - Section 0 contains the clock channels. Channel 0 is the date in "MM.DD.YY", 1 is the time in "HH.MM.SS". Channels 2 thru 9 are undefined as this is the only section where the channels are not selected consecutively. To set the date, select channel 10 of the clock section manually. The SYM will respond with " date?". To set the date enter `month' `-' `day' `-' `year'. Then, while holding down the black button on the clock board, press `CR' for the new time to take effect. The time is set similarly by selecting channel 11. The SYM will respond with " time?" and the time is entered as `hour (24-hour format)' `-' `minute'. Seconds are automatically set to zero when `CR' is pressed.

1 - Section 1 displays the analog channels currently being sampled. The display is either fixed or floating point representation depending on the value of the number. Fixed notation has a 4 digit mantissa with 1 decimal point and scientific notation has a decimal point, 3 mantissa digits, a decimal point, 1 base 10 exponent digit, and the sign bit of the exponent. The first two digits represent the channel number, while the remaining 4 digits display the numeric value of the data in millivolts. In fixed notation the decimal is placed in the correct place in the number with the sign bit being the point between the two channel number digits. In scientific notation the decimal point preceding the mantissa and the point preceding the exponent are displayed. Sign bit of the mantissa is the same as in fixed notation, while the sign of the exponent is the last decimal point.

EX:
```
0 1 1 2 4 1.   is  1241.0  millivolts on channel 1
0.1 1 2 4.1    is  -124.1  millivolts on channel 1
0 4.1 2 4 1    is  .1241   millivolts on channel 4
0 4.1 2 4.1    is  .124e+1 millivolts on channel 4
0 4.1 2 4.1.   is  .124E-1 millivolts on channel 4
0.4.1 2 4.1.   is -.124E-1 millivolts on channel 4
```

2 - Section 2 displays the digital channels if they are selected, and will skip to section 3 if not selected. The display is the channel number (d0 to d7) followed by an on or off indication of the channel.

3 - Section 3 displays the pulse channels if they are selected, and will skip to section 4 if not selected. The display consists of 4 channels indicated by P0 to P3 in the first two digits. The number is then displayed as total counts in the format described in the section on analog data. P0 and P1 are the current second count on the pulse channels, and P2 and P3 are the current minute count on the respective channels. This gives both a second count for high frequency pulses and a minute count for low frequency pulses. The P2 and P3 values are modulo 65536 so errors will result on these two values if the signal is greater than 1000kHz.

4 - Section 4 is a null section and causes the section number to be reset to zero.

5 - Section 5 displays some general configuration data. This section is not normally displayed and must be selected manually. The first two digits represent the channel number and the last 4 digits represent the value. No decimal points are on and all numbers except as noted are in decimal in this section. The channels are as follows:

0. This is the logger id # burnt into rom. This is a unique number on each logger and is stored in the data to provide an automatic method for data identification.

1. This is the number of analog channels being sampled and corresponds to the switch settings.

2. This is the number of digital channels being sampled and corresponds to the switch settings.

3. This is the number of pulse channels being sampled and corresponds to the switch settings.

4. This is the number of records that are actually being saved on tape. Data is maintained in all of the memory even though only part of it is saved on tape.

5. This is the number of bytes that are used to store a record of data. Each time data is saved on tape, this is the number of bytes that is recorded.

6. This is the last location of memory that is being stored on tape. Storing begins at hexidecimal location 200.

7. This is the end of consecutive memory in hexidecimal. Data is stored in all memory up to this location, but not all of it is saved. On power up the SYM-1 saves all consecutive memory starting at $200 that is installed.

8. This display shows the sample, average, and save time intervals heing used. These are coded hexadecimal numbers representing the switch settings and are described above.

9. The center two digits are the a/d error count and should be zero except when the a/d board is unplugged. The last two digits are the current tape label which is derived from the time as follows:
   Label = ((day of month) * 8) + ((hour of day) / 3)
   This display is in hexadecimal notation.

1.2.3 Data Storage Format The data is stored in a record format with a header at the beginning. The header is the first 8 bytes and consists of the following information, one binary byte each.

LOGGERID, SWITCHES, #ANALOG CHANNELS, #DIGITAL CHANNELS,
#PULSE CHANNELS, SAMPLE INTERVAL, AVERAGE INTERVAL, SAVE INTERVAL

The data records continue in the following format:

1. The record header occurs first and contains the time and count. The time is 6 BCD bytes in the following order: Year, Month, Day, Hour, Minute, Second. The next 2 bytes is a binary integer containing the number of samples averaged in this record.

2. The next series of hytes are the analog readings. The tape stores sums of samples of pure analog to digital conversions. Conversions result in a value from +/- 1.00 and therefore need multiplied hy 5 volts to give volts. The number of samples summed is the count in the record header. Therefore each number needs multiplied by 5 and divided by the count to give average volts. This is done automaticly by the reader unless the average flag is set on the reader.

   The analog data contains the number of channels as given in analog channels in the file header. Each channel consists of 4 bytes which is a floating point number converted hy the reader to ascii decimal numhers.

3. The next series of hytes contain the digital channels. Each channel is a 2 hyte integer containing the numher of times the channel was on out of the numher of counts in the total numher of samples given in the record header.

4. The next series of bytes is the pulse data. The first 2 bytes is the numher of seconds the pulse data was taken over. This may vary from the numher of counts in the record header. The pulse readings follow with a 4 hyte long integer each.

The next record follows immediately after the first. All data is pushed up in memory to make room for the next data item keeping the most current data at the heginning of the memory.

2. DATA CASSETTE TAPE READER

The reader is programmed to be a general reader that can transmit data to any device using RS232C data transfer. The data is converted to ascii character data before transmission so it is reasonable to use a printer to get a printout of the data directly. The options that can be programmed are as follows:

1. Delay between the transmission of characters. This is a binary number which represents a multiple of 4 milliseconds to delay between character transmission.

2. Delay between the transmission of each line. Also a multiple of 4 milliseconds to delay after a "CR" is sent.

3. Prompt waiting. If this byte is nonzero, this is a prompt character the SYM-1 will wait for before sending the next line. Any Line delay is inserted after the prompt character is received.

4. Linefeed and all data flag. If bit 7 is set ($80), a linefeed character will be sent after a "CR". If bit 6 is set ($40), then all file data will be sent and not just the nonrepeated data.

5. Sync and average flag. If bit 7 is set ($80), the SYM-1 will await a character before proceeding with the transmission of the file. This will allow a microcomputer to save the last file before the next one is sent. In this mode the SYM-1 reads a file and then sends a "CR" at 1-second intervals. When a character is received, the SYM delays .5 seconds and then sends the file with any other protocall discussed above. The receiving microcomputer should await the reception of a "CR" and then send a character to the SYM-1 indicating that it is ready to receive data. If bit 6 is set ($40), then the analog data will not be averaged before it is printed. Averaging multiplies by 5 and divides by the count to give the output average in volts.

6. Baud rate. The baud rate may be set to any value that the SYM-1 is capable of (110 to 4800). This is a coded number that can be found in the SYM REFRENCE MANUAL Page 4-23.

2.1 USING THE READER

2.1.1 Program execution

U0 (U0 is the shifted character on the 0 key. Press <shift> <0>). This command starts the reader program in the data logger format mode.

U1 This entry to the program does a hexadecimal dump of data on the tape. The data is in lines of 16 hex numbers exactly as on tape without any predefined format of the data.

U2 This entry is an end of data send. It may be used from the reader as a signal that the tape has ended. The output from this command is simply:

```
EOT
END
```

2.1.2 Table selection. The above commands can be entered with or without a table number to define the parameters to be used. Without a table number entered (U0 <cr>), the reader will prompt for a table number with 'table' on the L.E.D. To enter a table number with the command enter "U0 n <cr>".

The tables entered are as follows:

0 This is the user entered values located at $100 in memory. This is for custom parameters and are to be entered as 6 bytes described above at $100. Once entered, "Ux 0" will use these values for setup.

1 Values are: chardel=0, linedel=0, prompt=$11, linefeed=0, sync=0, baud=$01 (4800 baud). These are values used for the UNIX system and will also work for IBM/370 CMS.

2 Values are: chardel=6, linedel=5, prompt=0, linefeed=0, sync=$80, baud=$01 (4800 baud). These values will work for a small applesoft program on the apple.

3 These values are the same as 2 except with linedel=$10. This is for applesoft programs that need longer delays between lines.

4 These values are the same as 2 except with linedel=$38. This is for applesoft programs that need even longer delays between lines.

5 Values are: chardel=16, linedel=16, prompt=0, linefeed=0, sync=$80, bard=$01 (4800 baud). These values will work on the Radio Shack TRS-80.

There is room for 5 additional tables in the eprom that can be custom programmed. They are currently the same as table 0.

APPENDIX B. Data Logger Hardware Diagrams

SPECIFICATIONS AND FEATURES

Analog Input Channels

- 16 single-ended or 8 true differential inputs expandable to 32 single-ended or 16 true differential by adding two additional multiplexers and connectors.

- 12-bit resolution

- Autoranging Programmable gain amplifier providing 11 gain levels from 1 to 1024 (Powers of 2).

- Input Voltage range ±5V for gain=1, ±5mV for gain=1024.

- Each channel sampled once per second. Channel conversion time is 275 microsecond

- Sample averaging time interval switch selectable from 1 second to 3 hours.

- Accuracy .075% FSR, Temperature range 0 to 70°C.

Digital Inputs

- 8 digital input channels

- each input channel is sampled every second and summed into a counter

Pulse Inputs

- 2 pulse/event counting channels

- up to 65K Hz input frequency

- responds to 1M Hz burst

- Inputs must be debounced to be useable on switches

Parts list for Data Aquizition System 3

Quanity	Item	Cost	Vender
	GENERAL		
1	Case	55	Jensen Tools
1	SYM-1	210	AB Computers
1	Power Supply HTAA-16W	50	Power-one Inc.
1	Cassette Recorder CTR-80A	50	Radio Shack
	SYM-1 EXPANSION		
2	2114 Ram Chips	12	AB Computers
1	6522 VIA	6	AB Computers
2	2716 Eproms	18	AB Computers
2	Miniature Plugs	3	Radio Shack or Newerk
1	Subminiature Plug	1	Radio Shack or Newerk
18	WW Pins	.18	Newerk
	CLOCK BOARD		
1	Battery holder for 2 AA cells	3	Newerk
2	AA Batteries	.5	Newerk
1	NC momentary SP switch	1	Newerk
1	on - on SPDT switch	2	Newerk
1	MSM5832RS Clock Chip	7.5	Concord Computer
1	CX-1V-32.768 kHz Crystal	2.5	Concord or Staltek
1	MC14011B Quad NAND	1	Newerk
1	MC14047BE Timer	1	Newerk
2	.1 uF Capicator (cer)	.8	Newerk
2	4.7 uF Capicators (tant)	1	Newerk
2	15 pF Capicator (cer)	.8	Newerk
4	Diodes 4001 ?	1.5	Newerk
2	1M Resistor	.2	Newerk
4	2.2K Resistor	.8	Newerk
2	16 pin headers	1	LCOMP
2	16 pin sockets	1.5	LCOMP
2	14 pin sockets	1.5	LCOMP
1	18 pin socket	1	LCOMP
1	44 pin WW edge connector	6	Newerk
1	4.5" X 2.5" PC Board	1	Newerk
7	WW pins	.07	Newerk
	A/D BOARD		
1	SDM856JG data aquiz sys	175	Burr Brown (Midtec)
1	3606AG PGIA	95	Burr Brown (Midtec)
1	MC14028	1.2	Newerk
1	MC14049	1.2	Newerk
1	MC14174	1.2	Newerk
1	8 spst DIP switchs	4	Newerk
3	.01 uF Capicator (cer)	1.5	Newerk
3	1 uF Capicator (tant)	2	Newerk
1	1M Resistor	.2	Newerk
1	3900 pF Capicator (cer)	.5	Newerk
1	100K potentiometer	1	Newerk

1	10K potentiometer	1	Newerk
1	100 Ohm potentiometer	1	Newerk
1	50 Ohm potentiometer	1	Newerk
4	16 pin headers	2	LCOMP
6	16 pin sockets	4.4	LCOMP
6	25 pin socket strips	8.4	LCOMP
1	8.5" X 4.5" PC Board	2.5	Newerk
6	.75" Pexiglas, wood, or Spacers		
4	8 connector Terminal Blocks	12	Newerk
	<u>TEMPERATURE SENSOR REFRENCE</u>		
1	TL062 dual op amp	1.2	Newerk
1	8 pin socket	.7	LCOMP
1	16 pin Socket	.7	LCOMP
1	1K Potentiometer	1.2	Newerk
1	5K Potentiometer	1.2	Newerk
1	47.5K Precision Resistor	.2	Newerk
2	8.66K Precision Resistor	.2	Newerk
	<u>16 CHANNEL EXPANSION</u>		
2	MPC8S 8 Channel Mux	30	Burr Brown (Midtec)
2	16 pin sockets	1.4	LCOMP
2	8 connector terminal blocks	6	Newerk
	<u>MISC</u>		
8"	4 Strand wire for PS		
15"	6 Strand wire for Cassette		
	WW wire		
	Mounting board for Sym and Cassette		
1	Input Wire Clamp		

MODIFICATIONS ON SYM 1 BOARD

1. RESET JUMPER: This jumper is to allow reset of the sym 1 via an external clock circuit. The solder connection between pin 3 and pin 4 of "U3" must be cut. These are inputs to a 74LS27 NOR gate. A wire is to soldered to pin 3 of this chip and ran to point "41" of sym 1 board. This point is internally tied to "K" of the edge connector "A".

2. 15 VOLT POWER JUMPER This jumper supplies +15 volts to the clock circuit. Solder in a wire wrap post to pin 3 of connector "K", and another post to point "23" of sym 1 board (N of edge connector A). Wire wrap these two posts together.

3. DEBUG OFF JUMPER This jumper disables the debug function of the sym 1. Solder in wire wrap posts at pin 5 of connector "K" (ground) and to point "25" (input to NAND gate U8 pin 4).

4. RESET VECTOR These connectors allow sym 1 to be initialized and controled by user's EPROM upon power up. Point "19" is wire wrapped to point "N" . Point "20" is wire wrapped to point "P" . The wire wrap posts must be inserted and soldered.

5. INTERUPT CONNECTOR JUMPER This jumper allows an interupt to be sent in from clock circuit via a data line. A wire is to be soldered from pin 7 of "U25" (a 6522) to pin 18 of that same chip. These are pins PA5 AND CB1 respectively.

6. TRANSISTOR BUFFER BYPASS The output buffer transistors on PB4-PB7 must be bypassed if they are to be used as input ports. The transistors can be easily by-passed if a wired connections are made form points "B4" to "R9"; "B5" to "R10"; "B6" to "R11"; and "B7" to "R12".

7. CUT WPM JUMPER A the small wire jumper between points "45" and "MM" must be cut.

Analog Input Connections

Primary Channels on SDM856

Chip	Pin	Channel
U1	11	SE0/DIFF0H
U1	70	SE8/DIFF0L
U1	10	SE1/DIFF1H
U1	71	SE8/DIFF1L
U1	9	SE2/DIFF2H
U1	72	SE10/DIFF2L
U1	8	SE3/DIFF3H
U1	73	SE11/DIFF3L
U1	7	SE4/DIFF4H
U1	74	SE12/DIFF4L
U1	6	SE5/DIFF5H
U1	75	SE13/DIFF5L
U1	5	SE6/DIFF6H
U1	76	SE14/DIFF6L
U1	4	SE7/DIFF7H
U1	77	SE15/DIFF7L

Expansion Channels on 2 MCP8S's

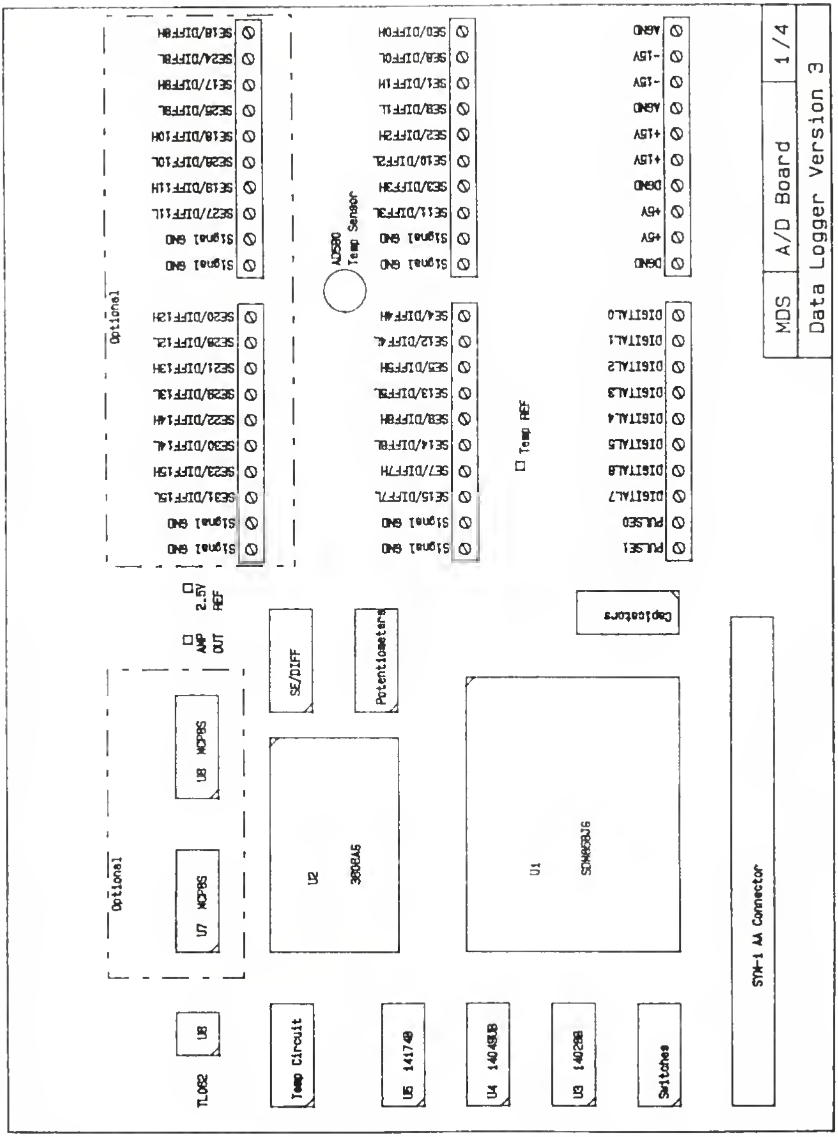
Chip	Pin	Channel
U7	4	SE16/DIFF8H
U8	4	SE24/DIFF8L
U7	5	SE17/DIFF9H
U8	5	SE25/DIFF9L
U7	6	SE18/DIFF10H
U8	6	SE26/DIFF10L
U7	7	SE19/DIFF11H
U8	7	SE27/DIFF11L
U7	12	SE20/DIFF12H
U8	12	SE28/DIFF12L
U7	11	SE21/DIFF13H
U8	11	SE29/DIFF13L
U7	10	SE22/DIFF14H
U8	10	SE30/DIFF14L
U7	9	SE23/DIFF15H
U8	9	SE31/DIFF15L

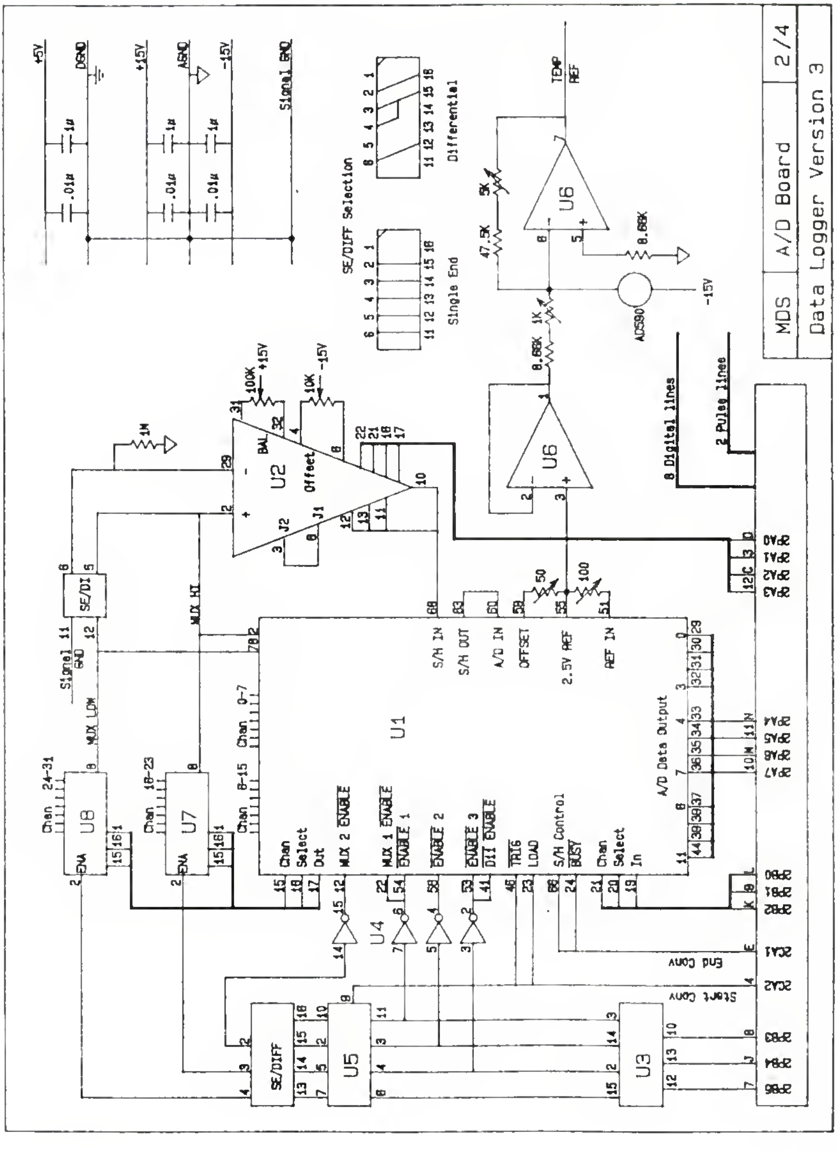
Digital and Pulse Channel Connections

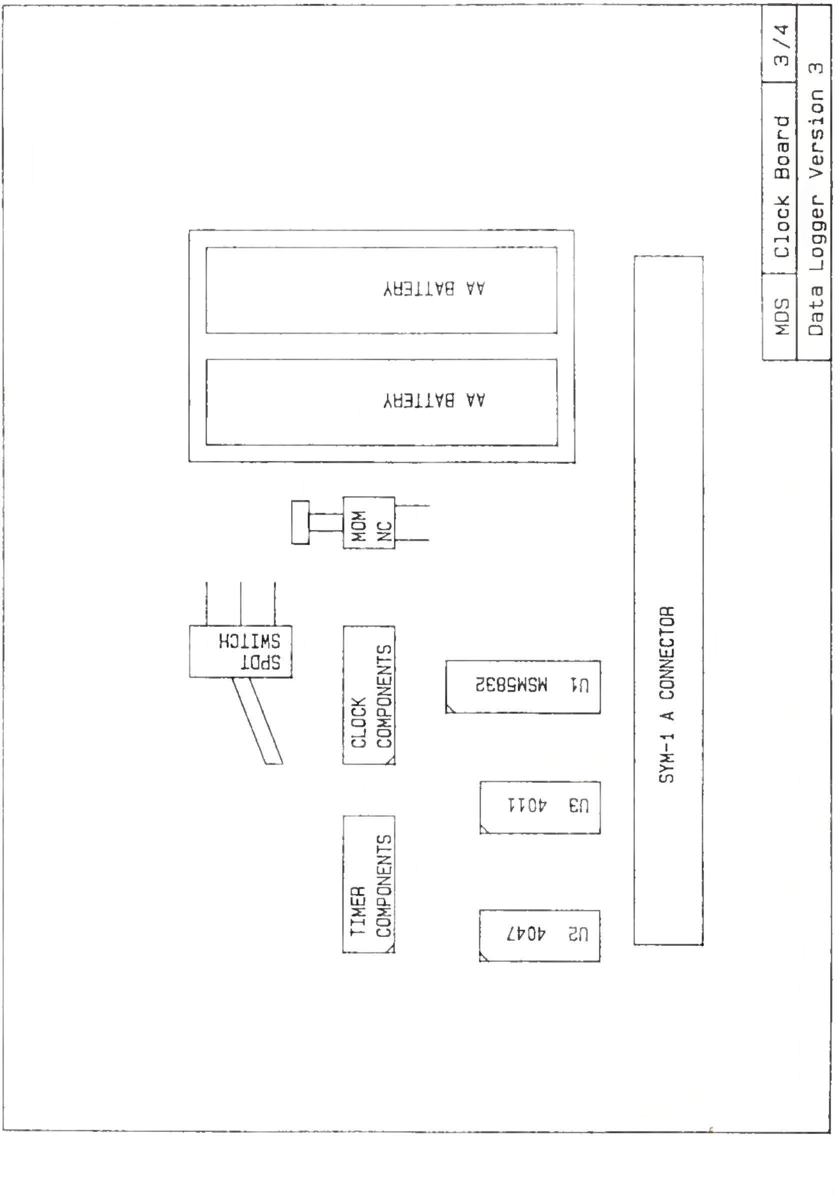
SYM-1 AA Connector		
Name	Pin	Channel
3PB0	16	DIGITAL0
3PB1	T	DIGITAL1
3PB2	15	DIGITAL2
3PB3	S	DIGITAL3
3PB4	Y	DIGITAL4
3PB5	21	DIGITAL5
2PB7	6	DIGITAL6
3PB7	22	DIGITAL7
2PB6	H	PULSE0
3PB6	Z	PULSE1

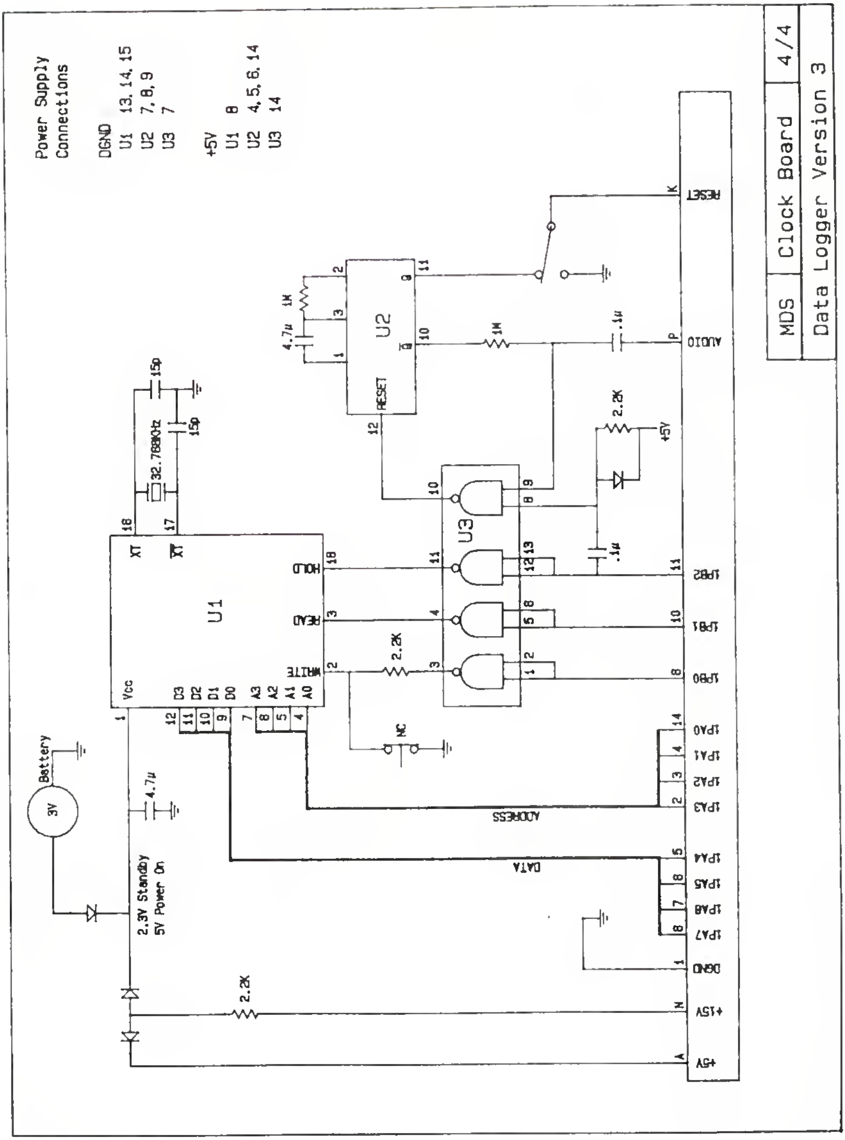
Switch Connections
One side of each switch is tied to DGND

SYM-1 AA Connector		
Name	Pin	Channel
3PA0	V	SW1
3PA1	W	SW2
3PA2	X	SW3
3PA3	18	SW4
3PA4	19	SW5
3PA5	20	SW6
3PA6	17	SW7
3PA7	U	SW8


MDS
A/D Board
1/4
Data Logger Version 3
Optional
SE16/DIFF8H
SE24/DIFF8L
SE17/DIFF9H
SE25/DIFF9L
SE18/DIFF10H
SE26/DIFF10L
SE19/DIFF11H
SE27/DIFF11L
Signal GND
Signal GND
SE20/DIFF12H
SE28/DIFF12L
SE21/DIFF13H
SE29/DIFF13L
SE22/DIFF14H
SE30/DIFF14L
SE23/DIFF15H
SE31/DIFF15L
Signal GND
Signal GND
AD590
Temp Sensor
SE0/DIFF0H
SE8/DIFF0L
SE1/DIFF1H
SE9/DIFF1L
SE2/DIFF2H
SE10/DIFF2L
SE3/DIFF3H
SE11/DIFF3L
Signal GND
Signal GND
SE4/DIFF4H
SE12/DIFF4L
SE5/DIFF5H
SE13/DIFF5L
SE6/DIFF6H
SE14/DIFF6L
SE7/DIFF7H
SE15/DIFF7L
Signal GND
Signal GND
Temp REF
AGND
-15V
-15V
AGND
+15V
+15V
DGND
+5V
+5V
DGND
DIGITAL0
DIGITAL1
DIGITAL2
DIGITAL3
DIGITAL4
DIGITAL5
DIGITAL6
DIGITAL7
PULSE0
PULSE1
2.5V REF
AMP OUT
SE/DIFF
Potentiometers
Capacitors
Optional
U8 MCP8S
U7 MCP8S
U2 380BA5
U1 SCN68BJ6
SYM-1 AA Connector
TL082
U6
Temp Circuit
U5 14174B
U4 14049UB
U3 14028B
Switches


MDS
A/D Board
2/4
Data Logger Version 3
+5V
DGND
+15V
AGND
-15V
Signal GND
.01µ
1µ
SE/DIFF Selection
Single End
Differential
6 5 4 3 2 1
11 12 13 14 15 16
TEMP
REF
U6
5K
47.5K
8.66K
1K
AD590
-15V
U2
BAL
Offset
100K
+15V
10K
1M
U1
S/H IN
S/H OUT
A/D IN
OFFSET
2.5V REF
REF IN
A/D Data Output
Chan 0-7
Chan 8-15
Chan 16-23
Chan 24-31
U7
U8
ENA
Chan Select Out
MUX 2 ENABLE
MUX 1 ENABLE
ENABLE 1
ENABLE 2
ENABLE 3
DI1 ENABLE
TRIG
LOAD
S/H Control
BUSY
Chan Select In
SE/DI
Signal GND
MUX LOW
MUX HI
U4
U5
U3
SE/DIFF
Start Conv
End Conv
8 Digital lines
2 Pulse lines
50
100
PPA0
PPA1
PPA2
PPA3
PPA7
PPA6
PPA5
PPA4
PPB0
PPB1
PPB2
PPB3
PPB4
PPB6
PC1
PC2


MDS
Clock Board
3/4
Data Logger Version 3
AA BATTERY
AA BATTERY
MOM NC
SPDT SWITCH
CLOCK COMPONENTS
TIMER COMPONENTS
U1 MSM5832
U3 4011
U2 4047
SYM-1 A CONNECTOR


Power Supply
Connections
DGND
U1 13, 14, 15
U2 7, 8, 9
U3 7
+5V
U1 8
U2 4, 5, 6, 14
U3 14
MDS
Clock Board
4/4
Data Logger Version 3
Battery
3V
4.7µ
2.3V Standby
5V Power On
2.2K
U1
Vcc
XT
XT
HOLD
READ
WRITE
D3 D2 D1 D0
A3 A2 A1 A0
32.768KHz
15p
15p
U2
RESET
1M
4.7µ
1M
.1µ
U3
.1µ
2.2K
+5V
NC
ADDRESS
DATA
RESET
AUDIO
1PB2
1PB1
1PB0
1PA0
1PA1
1PA2
1PA3
1PA4
1PA5
1PA6
1PA7
DGND
+15V
+5V

APPENDIX C. Data Logger Program Listings

Data Logger Source Listing

Data Logger Floating Point Routines

Cassette Tape Reader Source Listing

```
;PUT "L3LG.C"
            .LS

;SYM-1 DATA LOGGER 3.0 EXCEPT FP ROUTINES
;APRIL 1981
;BY MICHAEL D. SCHWARZ

;AG ENGG EXTENSION
;KANSAS STATE UNIVERSITY

;VERSION 3.1

;JULY 16,1981
;REMOVED AVERAGING OF ANALOG INFORMATION TO BE DONE
;IN THE READER PROGRAM. RESTRUCTURING. ETC.

;VERSION 3.2

;OCTOBER 21,1981
;CORRECTED CURVAL CALC ERROR
;CHANGED TO 1 HR SAVE DEFAULT
;CORRECTED ANALOG PROBLEM WITH ZERO
;SPLIT DISVEL FOR FIXED AND CYCLING
;REWROTE MOVE ROUTINE FOR INCREASED SPEED

            .CE
            .CT
            .OS
            .ES

            .PR "ASSEMBLY OF LOGGING ROUTINES"
            .PR "PUT FPDISK IN DRIVE 2"
            .PR "FIRST EPROM IS STORED AT $800"
            .PR "SECOND EPROM IS STORED AT $1000"

AVGANALOG   .DE 0                   ;0 FOR NO AVG 1 FOR AVG
LOWMEM      .DE $200

            .FI "SYM.L"
```

0289 3485-370E SYM.L

```
;PUT "SYM.L"

;MONITOR ADDRESSES

ACCESS      .DE $8B86
BLKMOV      .DE $8740
CONFIG      .DE $89A5
FILL3       .DE $8718
INBYTE      .DE $81D9
INCHR       .DE $8A1B
KSCONF      .DE $89A3
L2          .DE $84D3
MONITOR     .DE $8003
NACCESS     .DE $8B9C
OUTBYT      .DE $82FA
OUTCHR      .DE $8A47
OUTNIB      .DE $8A44
OUTXAH      .DE $82F4
PARM        .DE $8220
RESALL      .DE $81C4
RESTART     .DE $8000
SAVE2       .DE $87EA
SAVER       .DE $8188
SCAND       .DE $8906
SPACE       .DE $8342
VECSW       .DE $8BB7

DISBUF      .DE $A640
GOVEC       .DE $A659
ID          .DE $A64E
```

```
         0031 INTVEC      .DE $A678
         0032 PARN        .DE $A64A
         0033 PARNR       .DE $A649
         0034 RMDIG       .DE $A645
         0035 SDBYT       .DE $A651
         0036 TAPDEL      .DE $A630
         0037
         0038 VIA1        .DE $A000
         0039 VIA2        .DE $A800
         0040 VIA3        .DE $AC00
         0041
         0042 ;PUT "SYM.L"
         0040             .FI "L3FP.L"

013A  3485-35BF  L3FP.L

         0001 ;PUT "L3FP.L"
         0002
         0003 FPBASE      .DE $CC00
         0004 CLRMEM      .DE FPBASE+$000
         0005 MOVIND      .DE FPBASE+$00A
         0006 INCMEM      .DE FPBASE+$015
         0007 DECMEM      .DE FPBASE+$01E
         0008 FPNORM      .DE FPBASE+$071
         0009 FPADD       .DE FPBASE+$0B1
         0010 FPSUB       .DE FPBASE+$11C
         0011 FPMULT      .DE FPBASE+$126
         0012 FPDIV       .DE FPBASE+$1C9
         0013 FPOUT       .DE FPBASE+$2B8
         0014 FPDISP      .DE FPBASE+$38F
         0015
         0016 ;PUT "L3FP.L"
         0041             .FI "L3MACROS.M"

015F  3485-35E4  L3MACROS.M

         0001 ;PUT "L3MACROS.M"
         0002
         0003 ;MACROS FOR 16BIT MANIPULATIONS
         0004
         0005             .MG
         0006 !!!LDZI     .MD (...A ...B)
         0007             LDA #L,...B
         0008             STA *...A
         0009             LDA #H,...B
         0010             STA *...A+1
         0011             .ME
         0012 !!!LDZZ     .MD (...A ...B)
         0013             LDA *...B
         0014             STA *...A
         0015             LDA *...B+1
         0016             STA *...A+1
         0017             .ME
         0018 !!!INCD     .MD (...A ...B)
         0019             CLC
         0020             LDA #...B
         0021             ADC *...A
         0022             STA *...A
         0023             BCC =+3
         0024             INC *...A+1
         0025             .ME
         0026
         0027 ;PUT "L3MACROS.M"
         0042
         0043             .BA $00
         0044             .FI "L3ZPVARS.B"

03BA  35E4-399E  L3ZPVARS.B

         0001 ;PUT "L3ZPVARS.B"
         0002
         0003 ;ZERO PAGE VARIABLES
```

```
                 0004
                 0005 ;VARIABLES INITIALIZED BY INITIAL
                 0006
0000-            0007 VERIFY     .DS 2
0002-            0008 MEMEND     .DS 2
0004-            0009 DATAEND    .DS 2
0006-            0010 DATASTART  .DS 2
0008-            0011 DISVEL     .DS 2
                 0012
                 0013 ;***** UNINITIALIZED VARIABLES
                 0014
                 0015 ;         HEADER INFO
000A-            0016 SWITCH     .DS 1
000B-            0017 ADCHAN     .DS 1
000C-            0018 DGCHAN     .DS 1
000D-            0019 PLCHAN     .DS 1
000E-            0020 SMPTIME    .DS 1
000F-            0021 AVGTIME    .DS 1
0010-            0022 SAVTIME    .DS 1
                 0023 ;         ADDRESSES AND POINTERS
0011-            0024 ADADDR     .DS 2
0013-            0025 DGADDR     .DS 2
0015-            0026 JMPVEC     .DS 2
0017-            0027 PLADDR     .DS 2
0019-            0028 WORKPNT    .DS 2
                 0029 ;         SHOW VARIABLES
001B-            0030 CYCLE      .DS 1
001C-            0031 DISCNT     .DS 2
001E-            0032 KEYCNT     .DS 1
001F-            0033 LCNT1      .DS 1
0020-            0034 LCNT2      .DS 1
                 0035 ;         INTERRUPT VARS
0021-            0036 CURVAL     .DS 1
0022-            0037 DAY        .DS 6
                 0038 ;         DATA HANDLING VARS
                 0039            IFN AVGANALOG
                 0040 DIVTMP     .DS 4
                 0041            ***
0028-            0042 LABEL      .DS 1
0029-            0043 LENGTH     .DS 2
002B-            0044 NUMREC     .DS 1
                 0045 ;         PULSE VARABLES
002C-            0046 PLTMP      .DS 1
002D-            0047 PLLSTSEC   .DS 1
002E-            0048 PLINCSEC   .DS 1
002F-            0049 PLSEC      .DS 4
0033-            0050 PLMIN      .DS 4
0037-            0051 PLSUM      .DS 4
                 0052 ;         DIGITAL CHANNEL VARIABLES
003B-            0053 DGBYTE     .DS 1
                 0054 ;         ANALOG/DIGITAL CONVERTER VARS
003C-            0055 ADERR      .DS 1
003D-            0056 CHANN      .DS 1
003E-            0057 HEXAD      .DS 2
0040-            0058 GAIN       .DS 64                   ;(# OF CHANNELS)
                 0059
                 0060 ;PUT "L3ZPVARS.B"
                 0045            .BA $B8
                 0046            .FI D26 "FPVARS.B"

02BD  35E4-38A1  FPVARS.B

                 0001 ;PUT "FPVARS.B"
                 0002
                 0003 ;VARIABLES FOR MATH ROUTINES
                 0004
00B8-            0005 FMPNT      .DS 2
00BA-            0006 TOPNT      .DS 2
00BC-            0007 CNTR       .DS 1
00BD-            0008 TSIGN      .DS 1
00BE-            0009 SIGNS      .DS 1
00BF-            0010 FPLSWE     .DS 1
```

```
00C0-            0011 FPLSW      .DS 1
00C1-            0012 FPNSW      .DS 1
00C2-            0013 FPMSW      .DS 1
00C3-            0014 FPACCE     .DS 1
00C4-            0015 MCAND0     .DS 1
00C5-            0016 MCAND1     .DS 1
00C6-            0017 MCAND2     .DS 1
00C7-            0018 FOLSWE     .DS 1
00C8-            0019 FOPLSW     .DS 1
00C9-            0020 FOPNSW     .DS 1
00CA-            0021 FOPMSW     .DS 1
00CB-            0022 FOPEXP     .DS 1
00CC-            0023 WORK0      .DS 1
00CD-            0024 WORK1      .DS 1
00CE-            0025 WORK2      .DS 1
00CF-            0026 WORK3      .DS 1
00D0-            0027 WORK4      .DS 1
00D1-            0028 WORK5      .DS 1
00D2-            0029 WORK6      .DS 1
00D3-            0030 WORK7      .DS 1
                 0031 ;       VARIABLES FOR I/O ROUTINES
00D4-            0032 INMTAS     .DS 1
00D5-            0033 INEXPS     .DS 1
00D6-            0034 INPRDI     .DS 1
00D7-            0035 IOLSW      .DS 1
00D8-            0036 IONSW      .DS 1
00D9-            0037 IOMSW      .DS 1
00DA-            0038 IOEXP      .DS 1
00DB-            0039 IOSTR      .DS 1
00DC-            0040 IOSTR1     .DS 1
00DD-            0041 IOSTR2     .DS 1
00DE-            0042 IOSTR3     .DS 1
00DF-            0043 IOEXPD     .DS 1
00E0-            0044 TPLSW      .DS 1
00E1-            0045 TPNSW      .DS 1
00E2-            0046 TPMSW      .DS 1
00E3-            0047 TPEXP      .DS 1
00E4-            0048 TEMP1      .DS 1
00E5-            0049 OUTBUF     .DS 16
                 0050
                 0051 ;PUT "FPVARS.B"
                 0047
                 0048            .BA $C000           ;FIRST EPROM
                 0049            .MC $800
                 0050            .FI "L3MAIN.S"

05B6   35E4-3B9A  L3MAIN.S

                 0001 ;PUT "L3MAIN.S"
                 0002
                 0003 ;MAIN:  CALLS INIT, SETS UP THE ADC, SETS THE
                 0004 ;INTERRUPT VECTOR, STARTS INTERRUPTS, AND CALLS
                 0005 ;SHOW.  LEAVES A RETURN ADDRESS ON STACK FOR SHOW.
                 0006
                 0007 MAIN
C000- 20 85 C1   0008           JSR INIT            ;INITIAL DATA IF NECESSARY
C003- A9 00      0009           LDA #0
C005- 20 C8 C6   0010           JSR INPA/DV         ;INITIALIZE ADC
C008- A9 19      0011           LDA #L,INTERRUPT    ;STORE INTERUPT VECTOR
C00A- 8D 78 A6   0012           STA INTVEC
C00D- A9 C0      0013           LDA #H,INTERRUPT
C00F- 8D 79 A6   0014           STA INTVEC+1
C012- 20 B0 C5   0015           JSR SETUPCLOCK
C015- 58         0016           CLI
C016- 20 CF C2   0017           JSR SHOW            ;NO RETURN FROM THIS SUBRO
                 0018
                 0019 ;THIS ROUTINE SERVICES THE CLOCK INTERRUPTS.
                 0020 ;IT CALLS SAMPLE, AVERAGE, AND STORE AT THE
                 0021 ;PROPER TIMES
                 0022
                 0023 INTERRUPT
C019- 48         0024           PHA
```

```
C01A- 8A          0025             TXA
C01B- 48          0026             PHA
C01C- 98          0027             TYA
C01D- 48          0028             PHA
                  0029
C01E- AD 00 A0    0030             LDA VIA1
C021- 09 20       0031             ORA #$20           ;SET BIT TO INDICATE INTR
C023- 8D 00 A0    0032             STA VIA1
C026- AD 02 A0    0033             LDA VIA1+2
C029- 09 20       0034             ORA #$20           ;SET AS OUTPUT
C02B- 8D 02 A0    0035             STA VIA1+2
C02E- A9 00       0036             LDA #0
C030- 8D 03 AC    0037             STA VIA3+3         ;DEFETE NACCESS & ACCESS
                  0038
C033- 20 8B C0    0039             JSR EVERYSEC       ;READCLOCK AND OTHERS
                  0040
C036- A2 03       0041             LDX #3             ;SECONDS TO HOURS
C038- CA          0042             DEX
C039- B5 25       0043             LDA *DAY+3,X       ;TIME BYTE
C03B- D0 04       0044             BNE =+5            ;NONZERO TIME
C03D- E0 00       0045             CPX #0             ;LAST COMPARE
C03F- D0 F7       0046             BNE =-8            ;TRY NEXT TIME BYTE
                  0047
C041- 86 21       0048             STX *CURVAL        ;TMP
C043- BC 23 C9    0049             LDY LOGTAB2,X
C046- A2 05       0050             LDX #5             ;POSSIBLE LOGS RETURNED
C048- F8          0051             SED
                  0052 @LOGAA
C049- D9 14 C9    0053             CMP LOGSM,Y
C04C- 90 05       0054             BCC @LOGBB
C04E- F9 14 C9    0055             SBC LOGSM,Y
C051- F0 04       0056             BEQ @LOGCC
                  0057 @LOGBB
C053- 88          0058             DEY
C054- CA          0059             DEX
C055- D0 F2       0060             BNE @LOGAA
                  0061 @LOGCC
C057- 8A          0062             TXA
C058- D8          0063             CLD
C059- A6 21       0064             LDX *CURVAL        ;LOAD SAVED X
                  0065
C05B- 18          0066             CLC
C05C- 7D 20 C9    0067             ADC LOGTAB1,X
C05F- 85 21       0068             STA *CURVAL        ;LOG OF TIME
                  0069
C061- C5 0E       0070             CMP *SMPTIME
C063- 90 18       0071             BCC ENDINT
C065- 20 95 C0    0072             JSR SAMPLE
C068- A5 21       0073             LDA *CURVAL
C06A- C5 0F       0074             CMP *AVGTIME
C06C- 90 0F       0075             BCC ENDINT
C06E- 20 AC C0    0076             JSR AVERAGE
C071- A5 21       0077             LDA *CURVAL
C073- C5 10       0078             CMP *SAVTIME
C075- 90 03       0079             BCC =+4
C077- 20 B0 C0    0080             JSR SAVE
C07A- 20 B7 C0    0081             JSR MOVE
                  0082 ENDINT
C07D- AD 00 A0    0083             LDA VIA1
C080- 29 DF       0084             AND #$DF           ;CLR BIT FOR LEAVING INTR
C082- 8D 00 A0    0085             STA VIA1
                  0086
C085- 68          0087             PLA
C086- A8          0088             TAY
C087- 68          0089             PLA
C088- AA          0090             TAX
C089- 68          0091             PLA
C08A- 40          0092             RTI
                  0093
                  0094 ;PUT "L3MAIN.S"
                  0051             .FI "L3FILE.S"
```

0B1E 35E4-4102 L3FILE.S

```
              ;PUT "L3FILE.S"

              ;INIT STOREAGE (IN ORDER)
              ;        *DATAEND        2
              ;        *DATASTART      2
              ;        *MOVETO         2

              ;INTERFACE ROUTINES CALLED FROM THE INTERRUPT
              ;ROUTINE.  ANY ROUTINE TO BE CALLED AT THE APPROPIATE
              ;TIME SHOULD BE HERE.

              EVERYSEC
C08B- 20 29 C5              JSR READCLOCK       ;GET CURRENT TIME
C08E- 20 00 C8              JSR PLSAVE          ;READ AND SAVE PULSE INPUT
C091- 20 67 C7              JSR DGREAD          ;READ DIGITAL BITS
C094- 60                    RTS
              SAMPLE
C095- A0 0E                 LDY #HEADER+6       ;INCREMENT SAMPLE COUNT
C097- 18                    CLC
C098- B1 06                 LDA (DATASTART),Y
C09A- 69 01                 ADC #$01
C09C- 91 06                 STA (DATASTART),Y
C09E- C8                    INY
C09F- B1 06                 LDA (DATASTART),Y
C0A1- 69 00                 ADC #$00
C0A3- 91 06                 STA (DATASTART),Y
C0A5- 20 DF C6              JSR SUMADC          ;SUM A/D CHANNELS
C0A8- 20 79 C7              JSR DGSAVE          ;SAVE DG INPUTS
C0AB- 60                    RTS
              AVERAGE
C0AC- 20 66 C1              JSR MOVECLOCK       ;SAVE TIME
                            IFN AVGANALOG
                            JSR AVGADC          ;AVERAGE A/D CHANNELS
                            ***
C0AF- 60                    RTS
              SAVE
C0B0- 20 73 C1              JSR MOVEHEADER      ;SAVE HEADER IN DATA
C0B3- 20 27 C1              JSR STORE
C0B6- 60                    RTS
              MOVE
C0B7- 20 BE C0              JSR MOVEDATA
C0BA- 20 01 C1              JSR CLRDATA
C0BD- 60                    RTS

              ;MOVE DATA UP IN STOREAGE AREA SO NEWEST
              ;DATA IS KEPT AT THE BEGINNING

              MOVEDATA
C0BE- 38                    SEC
C0BF- A5 02                 LDA *MEMEND         ;SET POINTERS TO END
C0C1- 85 BA                 STA *TOPNT
C0C3- E5 29                 SBC *LENGTH         ;POINTER TO MOVE FROM
C0C5- 85 B8                 STA *FMPNT
C0C7- A5 03                 LDA *MEMEND+1
C0C9- 85 BB                 STA *TOPNT+1
C0CB- E5 2A                 SBC *LENGTH+1
C0CD- 85 B9                 STA *FMPNT+1

C0CF- 38                    SEC                 ;SUB REMAINDER OF A PAGE
C0D0- A5 B8                 LDA *FMPNT
C0D2- E5 06                 SBC *DATASTART
C0D4- 85 B8                 STA *FMPNT          ;TEMPORARY: PARTIAL PAGE
C0D6- A8                    TAY
C0D7- B0 02                 BCS =+3
C0D9- C6 B9                 DEC *FMPNT+1
C0DB- 38                    SEC
C0DC- A5 BA                 LDA *TOPNT
C0DE- E5 B8                 SBC *FMPNT
C0E0- 85 BA                 STA *TOPNT
C0E2- B0 02                 BCS =+3
```

```
C0E4- C6 BB                         DEC *TOPNT+1
C0E6- A5 06                         LDA *DATASTART
C0E8- 85 B8                         STA *FMPNT               ;FINISHED WITH POINTERS
C0EA- 4C EE C0                      JMP @MOVEDATA

C0ED- 88                            DEY
                @MOVEDATA
C0EE- B1 B8                         LDA (FMPNT),Y            ;TRANSFER A PAGE
C0F0- 91 BA                         STA (TOPNT),Y
C0F2- C0 00                         CPY #0
C0F4- D0 F7                         BNE @MOVEDATA-1
C0F6- C6 B9                         DEC *FMPNT+1             ;NEXT PAGE
C0F8- C6 BB                         DEC *TOPNT+1
C0FA- A5 B9                         LDA *FMPNT+1
C0FC- C5 07                         CMP *DATASTART+1         ;END OF TRANSFER
C0FE- B0 ED                         BCS @MOVEDATA-1
C100- 60                            RTS

                ;CLEAR THE DATA AERA IN WHICH THE DATA IS SUMMED

                CLRDATA
C101- 18                            CLC
C102- A9 08                         LDA #HEADER              ;START OF BUFFER
C104- 65 06                         ADC *DATASTART
C106- 85 BA                         STA *TOPNT
C108- A9 00                         LDA #0
C10A- 65 07                         ADC *DATASTART+1
C10C- 85 BB                         STA *TOPNT+1
C10E- A4 2A                         LDY *LENGTH+1
C110- 84 B8                         STY *FMPNT               ;USE AS TMP CNTR
C112- C6 B8                         DEC *FMPNT
C114- 30 0A                         BMI =+11
C116- A2 00                         LDX #0
C118- 20 00 CC                      JSR CLRMEM               ;CLEAR 256 BYTES
C11B- E6 BB                         INC *TOPNT+1             ;NEXT PAGE OF MEM
C11D- 4C 12 C1                      JMP =-11
C120- A6 29                         LDX *LENGTH
C122- F0 03                         BEQ =+4
C124- 4C 00 CC                      JMP CLRMEM

                ;THIS ROUTINE SAVES THE DATA ON CASSETTE
                ;IN HS FORMAT.

                STORE
C127- 20 45 C1                      JSR CALCLABEL
C12A- A5 28                         LDA *LABEL
C12C- 8D 4E A6                      STA PARN+4
C12F- A2 03                         LDX #3
C131- B5 04                         LDA *DATAEND,X
C133- 9D 4A A6                      STA PARN,X
C136- CA                            DEX
C137- 10 F8                         BPL =-7
C139- A9 01                         LDA #1
C13B- 8D 30 A6                      STA TAPDEL               ;SHORTEN LEADER TO 1.5 SEC
C13E- 08                            PHP
C13F- 78                            SEI
C140- 20 EA 87                      JSR SAVE2                ;SAVE ON TAPE
C143- 28                            PLP
C144- 60                            RTS

                CALCLABEL
C145- A5 24                         LDA *DAY+2               ;DAY OF MONTH
C147- 29 3F                         AND #$3F
C149- 20 70 C4                      JSR BINARY
C14C- 0A                            ASL A
C14D- 0A                            ASL A
C14E- 0A                            ASL A
C14F- 85 28                         STA *LABEL
C151- A5 25                         LDA *DAY+3               ;HOUR OF DAY
C153- 29 3F                         AND #$3F
C155- 20 70 C4                      JSR BINARY
C158- A2 FF                         LDX #$FF
```

```
C15A- E8                            INX
C15B- 38                            SEC
C15C- E9 03                         SBC #3              ;DIVIDE BY 3
C15E- B0 FA                         BCS =-5
C160- 8A                            TXA
C161- 05 28                         ORA *LABEL
C163- 85 28                         STA *LABEL
C165- 60                            RTS

               ;MOVE CLOCK TO DATA FIELD

               MOVECLOCK
C166- A2 05                         LDX #5
C168- A0 0D                         LDY #HEADER+5
C16A- B5 22                         LDA *DAY,X
C16C- 91 06                         STA (DATASTART),Y
C16E- 88                            DEY
C16F- CA                            DEX
C170- 10 F8                         BPL MOVECLOCK+4
C172- 60                            RTS

               ;SAVE HEADER STUFF IN DATA AREA

               MOVEHEADER
C173- A0 07                         LDY #HEADER-1
C175- A2 10                         LDX #SAVTIME        ;END OF HEADER DATA
C177- B5 00                         LDA *0,X
C179- 91 06                         STA (DATASTART),Y
C17B- CA                            DEX
C17C- 88                            DEY
C17D- D0 F8                         BNE MOVEHEADER+4    ;LOOP FOR ALL HEADER
C17F- AD FF C7                      LDA LOGGERID        ;UNIQUE ID FOR EACH LOGGER
C182- 91 06                         STA (DATASTART),Y   ;SAVE ID
C184- 60                            RTS

               ;PUT "L3FILE.S"
                                    .FI "L3INIT.S"

0F24  35E4-4508  L3INIT.S

                                    .CE
               ;PUT "L3INIT.S"

               ;INITIAL ROUTINE INITIALIZES DATA IF LOCATIONS
               ;00 & 01 ARE NOT 5A & A5 RESP.

               INIT
C185- A9 00                         LDA #$00            ;CHECK SWITCHES
C187- 8D 03 AC                      STA VIA3+3          ;SET FOR INPUT
C18A- AD 01 AC                      LDA VIA3+1
C18D- 49 FF                         EOR #$FF            ;INVERT FOR NICENESS
C18F- C5 0A                         CMP *SWITCH         ;ARE SWITCHES SAME
C191- D0 0D                         BNE @INITA
C193- A6 00                         LDX *VERIFY         ;IS INIT STILL INTACT?
C195- E0 5A                         CPX #$5A            ;VERIFY SHOULD BE 5A & A5
C197- D0 07                         BNE @INITA
C199- A6 01                         LDX *VERIFY+1
C19B- E0 A5                         CPX #$A5
C19D- D0 01                         BNE @INITA
C19F- 60                            RTS                 ;VERIFY HASN'T CHANGED

               @INITA
C1A0- 78                            SEI
C1A1- 20 B8 C1                      JSR ZEROMEM
C1A4- 20 E8 C1                      JSR DECODESW
C1A7- 20 15 C2                      JSR CALCLENGTH
C1AA- 20 36 C2                      JSR CALCADDR
C1AD- 20 63 C2                      JSR CALCMXREC
C1B0- 20 7E C2                      JSR CALCDATAEND
C1B3- 20 AF C2                      JSR CALCSAVE
C1B6- 58                            CLI
C1B7- 60                            RTS
```

```

                 ;ZEROMEM: ZERO ZP & DATA MEMORY, INITIALIZE
                 ;ZP VALUES, AND FIND END OF MEMORY.

                 ZEROMEM
C1B8- 48                   PHA                     ;SAVE SWITCHES
C1B9- A9 00                LDA #0                  ;SET ZERO PAGE MEMORY TO Z
C1BB- A2 F8                LDX #$F8
C1BD- CA                   DEX
C1BE- 95 00                STA *00,X
C1C0- D0 FB                BNE =-4

C1C2- A2 09                LDX #ICOUNT             ;# OF BYTES IN ZERO PAGE T
C1C4- BD 5D C9             LDA INITDATA,X          ;TRANSFER INITIAL VALUES
C1C7- 95 00                STA *$00,X
C1C9- CA                   DEX
C1CA- 10 F8                BPL =-7

C1CC- A9 00                LDA #0
C1CE- A0 00                LDY #0                  ;SET MEMEND ON LAST BYTE
C1D0- 91 02                STA (MEMEND),Y          ;CLR MEM
C1D2- D1 02                CMP (MEMEND),Y          ;MEMORY EXISTANT
C1D4- D0 08                BNE =+9
C1D6- E6 02                INC *MEMEND             ;POINT TO NEXT BYTE
C1D8- D0 F6                BNE =-9
C1DA- E6 03                INC *MEMEND+1
C1DC- D0 F2                BNE =-13
C1DE- A5 02                LDA *MEMEND
C1E0- D0 02                BNE =+3
C1E2- C6 03                DEC *MEMEND+1           ;BACKUP ONE BYTE
C1E4- C6 02                DEC *MEMEND

C1E6- 68                   PLA
C1E7- 60                   RTS

                 ; SAVE AND DECODE SWITCHES
                 ; SWITCH BITS ARE DIVIDED AS FOLLOWS:
                 ;   NOT USED          0
                 ;   ANALOGCHAN        1,2
                 ;   DIGITALCHAN       3
                 ;   PULSECHAN         4
                 ;   SAMPLETIMES       5,6,7

                 DECODESW
C1E8- 85 0A                STA *SWITCH             ;SET SWITCHES
C1EA- 29 06                AND #$06                ;PICK ADCHANS
C1EC- 0A                   ASL A
C1ED- 0A                   ASL A
C1EE- 69 08                ADC #$08
C1F0- 85 0B                STA *ADCHAN
C1F2- A5 0A                LDA *SWITCH             ;GET DIGITAL CHANNELS
C1F4- 29 08                AND #$08
C1F6- 85 0C                STA *DGCHAN
C1F8- A5 0A                LDA *SWITCH             ;GET PULSE CHANNELS
C1FA- 29 10                AND #$10
C1FC- 4A                   LSR A
C1FD- 4A                   LSR A
C1FE- 4A                   LSR A
C1FF- 85 0D                STA *PLCHAN
C201- A5 0A                LDA *SWITCH             ;GET INTERVAL TIMES
C203- 29 E0                AND #$E0
C205- 4A                   LSR A
C206- 4A                   LSR A
C207- 4A                   LSR A
C208- 4A                   LSR A
C209- AA                   TAX
C20A- BD 67 C9             LDA TIMES,X             ;SAMPLE TIME
C20D- 85 0E                STA *SMPTIME
C20F- BD 68 C9             LDA TIMES+1,X           ;AVERAGE TIME
C212- 85 0F                STA *AVGTIME
C214- 60                   RTS

```

```
                  ;    CALCUATE LENGTH OF RECORD & STORE AT LENGTH

                  CALCLENGTH
C215- 18                         CLC
C216- A5 0B                      LDA *ADCHAN
C218- 65 0D                      ADC *PLCHAN
C21A- 69 02                      ADC #$02            ;8 BYTES FOR CLOCK AND COU
C21C- 0A                         ASL A               ;MULT BY 2
C21D- 85 29                      STA *LENGTH
C21F- 26 2A                      ROL *LENGTH+1       ;SHIFT IN CARRY
C221- 18                         CLC
C222- A5 0D                      LDA *PLCHAN         ;MAKE SPACE FOR CLOCK
C224- F0 01                      BEQ =+2
C226- 38                         SEC                 ;ADD 2 FOR CLOCK
C227- A5 0C                      LDA *DGCHAN
C229- 65 29                      ADC *LENGTH
C22B- 85 29                      STA *LENGTH
C22D- 90 02                      BCC =+3
C22F- E6 2A                      INC *LENGTH+1
C231- 06 29                      ASL *LENGTH         ;MULT BY 2
C233- 26 2A                      ROL *LENGTH+1
C235- 60                         RTS

                  ;   CALCUATE ADDRESSES OF THE DATA

                  CALCADDR
C236- 18                         CLC                 ;CALC ADADDR
C237- A9 10                      LDA #HEADER+8       ;HEADER + TIME AND LENGTH
C239- 65 06                      ADC *DATASTART
C23B- 85 11                      STA *ADADDR
C23D- A9 00                      LDA #$00
C23F- 65 07                      ADC *DATASTART+1
C241- 85 12                      STA *ADADDR+1

C243- A5 0B                      LDA *ADCHAN         ;CALC DGADDR
C245- 0A                         ASL A
C246- 26 14                      ROL *DGADDR+1
C248- 0A                         ASL A
C249- 26 14                      ROL *DGADDR+1
C24B- 65 11                      ADC *ADADDR
C24D- 85 13                      STA *DGADDR
C24F- A5 14                      LDA *DGADDR+1
C251- 65 12                      ADC *ADADDR+1
C253- 85 14                      STA *DGADDR+1

C255- A5 0C                      LDA *DGCHAN         ;CALC PLADDR
C257- 0A                         ASL A
C258- 65 13                      ADC *DGADDR
C25A- 85 17                      STA *PLADDR
C25C- A9 00                      LDA #$00
C25E- 65 14                      ADC *DGADDR+1
C260- 85 18                      STA *PLADDR+1
C262- 60                         RTS

                  ;    FIND LENGTH OF MEMORY AND CALCUATE
                  ;    DATA END AND MOVE ADDRESSES

                  CALCMXREC
C263- 38                         SEC
C264- A9 0C                      LDA #MAXSAVE        ;MAX SAVE INTERVAL
C266- E5 0F                      SBC *AVGTIME        ;AVERAGE INTERVAL
C268- B0 02                      BCS =+3             ;0 IF MAX < AVG
C26A- A9 00                      LDA #0
C26C- C9 06                      CMP #6              ;LOG OF #REC PER SAVE
C26E- B0 09                      BCS @CALCMXREC
C270- AA                         TAX
C271- A9 02                      LDA #2              ;REP FACTOR OF 4
C273- 0A                         ASL A
C274- CA                         DEX
C275- 10 FC                      BPL =-3
C277- 30 02                      BMI =+3             ;ALWAYS
                  @CALCMXREC
```

```
C279- A9 FF                         LDA #$FF
C27B- 85 21                         STA *CURVAL
C27D- 60                            RTS

              CALCDATAEND
C27E- 18                            CLC                  ;ADD LENGTH TO DATAEND
C27F- A5 04                         LDA *DATAEND
C281- 65 29                         ADC *LENGTH
C283- 85 04                         STA *DATAEND
C285- A5 05                         LDA *DATAEND+1
C287- 65 2A                         ADC *LENGTH+1
C289- 85 05                         STA *DATAEND+1

C28B- A5 03                         LDA *MEMEND+1        ;CHECK FOR VALID MEMORY
C28D- C5 05                         CMP *DATAEND+1
C28F- D0 04                         BNE =+5
C291- A5 02                         LDA *MEMEND
C293- C5 04                         CMP *DATAEND
C295- 90 0A                         BCC @CALCDATAEND1

C297- E6 2B                         INC *NUMREC
C299- A5 2B                         LDA *NUMREC
C29B- C5 21                         CMP *CURVAL          ;MAX # REC
C29D- F0 0F                         BEQ @CALCDATAEND2
C29F- D0 DD                         BNE CALCDATAEND      ;ALWAYS
              @CALCDATAEND1
C2A1- 38                            SEC
C2A2- A5 04                         LDA *DATAEND
C2A4- E5 29                         SBC *LENGTH
C2A6- 85 04                         STA *DATAEND
C2A8- A5 05                         LDA *DATAEND+1
C2AA- E5 2A                         SBC *LENGTH+1
C2AC- 85 05                         STA *DATAEND+1
              @CALCDATAEND2
C2AE- 60                            RTS

              ;    CALCUATE SAVE TIME INTERVAL

              CALCSAVE
C2AF- A6 0F                         LDX *AVGTIME
C2B1- A5 2B                         LDA *NUMREC
C2B3- 4A                            LSR A                ;SAVE AT LEAST TWICE
C2B4- 4A                            LSR A
C2B5- F0 03                         BEQ =+4
C2B7- E8                            INX
C2B8- D0 FA                         BNE =-5              ;ALWAYS
C2BA- E0 0C                         CPX #MAXSAVE         ;SAVE AT LEAST EVERY 3 HOU
C2BC- 90 02                         BCC =+3
C2BE- A2 0C                         LDX #MAXSAVE
C2C0- E0 06                         CPX #MINSAVE         ;MINIMUN SAVE TIME
C2C2- B0 02                         BCS =+3
C2C4- A2 06                         LDX #MINSAVE
C2C6- E4 0F                         CPX *AVGTIME         ;DON'T SAVE MORE THAN AVG
C2C8- B0 02                         BCS =+3
C2CA- A6 0F                         LDX *AVGTIME
C2CC- 86 10                         STX *SAVTIME
C2CE- 60                            RTS

              ;PUT "L3INIT.S"
                                    .FI "L3SHOW.S"

0AF3  35E4-40D7  L3SHOW.S

              ;PUT "L3SHOW.S"

              ;DISPLAY ROUTINE.  THIS ROUTINE
              ;WILL DISPLAY ONE VALUE OR CYCLE
              ;THROUGH ALL VALUES ON DEMAND.
              ;INDIVIDUAL ROUTINES ARE CALLED FROM
              ;JUMPTABLE TO PUT THE DIFFERENT
              ;PARAMETERS INTO THE DISPLAY BUFFER

```

```
                   0010 SHOW
C2CF- A9 00        0011           LDA #0
C2D1- 8D 03 AC     0012           STA VIA3+3          ;DEFETE ACCESS & NACCESS
                   0013
C2D4- A5 1C        0014           LDA *DISCNT
C2D6- 0A           0015           ASL A               ;MULT BY 2
C2D7- AA           0016           TAX
C2D8- BD 44 C9     0017           LDA JMPTAB,X        ;GET ROUTINE ADDRESS
C2DB- 85 15        0018           STA *JMPVEC
C2DD- BD 45 C9     0019           LDA JMPTAB+1,X      ;2ND BYTE
C2E0- 85 16        0020           STA *JMPVEC+1
C2E2- C9 FF        0021           CMP #$FF
C2E4- D0 08        0022           BNE =+9
C2E6- A9 00        0023           LDA #$00
C2E8- 85 1C        0024           STA *DISCNT         ;START AT ZERO IF AT END
C2EA- 85 1D        0025           STA *DISCNT+1
C2EC- F0 E1        0026           BEQ SHOW
                   0027
C2EE- 78           0028           SEI
C2EF- 20 12 C3     0029           JSR SHOWDISP
C2F2- 58           0030           CLI
C2F3- 90 11        0031           BCC @SHOWLOOP       ;VALID IF CARRY CLEAR
C2F5- A9 00        0032           LDA #$00
C2F7- C5 1D        0033           CMP *DISCNT+1       ;INVALID ON CHAN 0?
C2F9- F0 06        0034           BEQ =+7             ;INC NOMATER WHAT CYCLE
C2FB- 85 1D        0035           STA *DISCNT+1       ;INCREMENT TO NEXT #
C2FD- C5 1B        0036           CMP *CYCLE
C2FF- D0 CE        0037           BNE SHOW            ;SAME ROUTINE IF NONZERO
C301- E6 1C        0038           INC *DISCNT
C303- 4C CF C2     0039           JMP SHOW
                   0040 @SHOWLOOP
C306- 20 15 C3     0041           JSR LOOP            ;DISPLAY VALUES
C309- 24 1B        0042           BIT *CYCLE
C30B- 30 C2        0043           BMI SHOW
C30D- E6 1D        0044           INC *DISCNT+1
C30F- 4C CF C2     0045           JMP SHOW
                   0046 SHOWDISP
C312- 6C 15 00     0047           JMP (JMPVEC)        ;PUT VALUES IN DISPLAY BUF
                   0048
                   0049 ;THIS ROUTINE SCANS DISPLAY WITH
                   0050 ;VALUES STORED IN DISPLAY BUFFER
                   0051
                   0052 LOOP
C315- A5 08        0053           LDA *DISVEL         ;CYCLING DISPLAY VELOCITY
C317- 24 1B        0054           BIT *CYCLE          ;CURRENTLY CYCLING
C319- 10 02        0055           BPL =+3
C31B- A5 09        0056           LDA *DISVEL+1       ;STEADY DISPLAY VELOCITY
C31D- 85 20        0057           STA *LCNT2
                   0058 @LOOP1
C31F- A9 38        0059           LDA #56
C321- 85 1F        0060           STA *LCNT1
                   0061 @LOOP2
C323- 20 06 89     0062           JSR SCAND           ;SCAN DISPLAY
C326- D0 0D        0063           BNE KEY             ;IF KEY DEPRESSED, KEY
C328- A9 28        0064           LDA #40
C32A- 85 1E        0065           STA *KEYCNT
C32C- C6 1F        0066           DEC *LCNT1
C32E- D0 F3        0067           BNE @LOOP2
C330- C6 20        0068           DEC *LCNT2
C332- D0 EB        0069           BNE @LOOP1
C334- 60           0070           RTS
                   0071
                   0072 ;KEY ROUTINE. CHECKS FOR VALID
                   0073 ;KEY INPUTS. AND DETERMINES WHAT
                   0074 ;KEY IS DEPRESSED.
                   0075
                   0076 KEY
C335- C6 1E        0077           DEC *KEYCNT
C337- D0 EA        0078           BNE @LOOP2
C339- A9 FF        0079           LDA #$FF
C33B- 85 1B        0080           STA *CYCLE          ;STOP CYCLING
C33D- A2 17        0081           LDX #23
```

```
C33F- 20 63 C4  0082             JSR LEDMESG          ;DISPLAY CHAN?
C342- 20 20 82  0083             JSR PARM             ;GET PARAMETERS
C345- D0 22     0084             BNE @KEY2            ;CR? NO,@KEY2
C347- AC 49 A6  0085             LDY PARNR            ;GET NUMBER OF PARAMETERS
C34A- F0 6F     0086             BEQ GOMON            ;IF NO PARAM THEN GOTO MON
C34C- AD 4A A6  0087             LDA PARN
C34F- 20 70 C4  0088             JSR BINARY
C352- 85 1D     0089             STA *DISCNT+1
C354- 88        0090             DEY
C355- D0 01     0091             BNE =+2              ;IF NOT 0 SKIP RTS STATEME
C357- 60        0092             RTS
C358- AD 4C A6  0093             LDA PARN+2
C35B- 20 70 C4  0094             JSR BINARY
C35E- 0A        0095             ASL A                ;MULT BY 2
C35F- C9 0E     0096             CMP #L,NUCHAN        ;OUT OF RANGE?
C361- 30 02     0097             BMI =+3
C363- 10 4A     0098             BPL @SERROR
C365- 4A        0099             LSR A
C366- 85 1C     0100             STA *DISCNT          ;SET ROUTINE #
C368- 60        0101             RTS
                0102 @KEY2
C369- C9 3C     0103             CMP #'<              ;BACK UP ONE CHANNEL
C36B- D0 07     0104             BNE @K1
C36D- A6 1D     0105             LDX *DISCNT+1        ;DON'T DEC IF ZERRO
C36F- F0 02     0106             BEQ =+3
C371- C6 1D     0107             DEC *DISCNT+1
C373- 60        0108             RTS
                0109 @K1
C374- C9 3E     0110             CMP #'>              ;NEXT CHANNEL
C376- D0 03     0111             BNE @K2
C378- E6 1D     0112             INC *DISCNT+1
C37A- 60        0113             RTS
                0114 @K2
C37B- C9 4D     0115             CMP #'M              ;STOP CYCLING
C37D- D0 01     0116             BNE @K3
C37F- 60        0117             RTS
                0118 @K3
C380- C9 52     0119             CMP #'R              ;CYCLE ONLY ONE TYPE
C382- D0 05     0120             BNE @K4
C384- A9 40     0121             LDA #$40
C386- 85 1B     0122             STA *CYCLE
C388- 60        0123             RTS
                0124 @K4
C389- C9 1E     0125             CMP #$1E             ;S2: SAVE DATA
C38B- D0 0C     0126             BNE @K5
C38D- 78        0127             SEI
C38E- 20 AC C0  0128             JSR AVERAGE
C391- 20 B0 C0  0129             JSR SAVE
C394- 20 B7 C0  0130             JSR MOVE
C397- 58        0131             CLI
C398- 60        0132             RTS
                0133 @K5
C399- C9 13     0134             CMP #$13             ;L2: AVERAGE A READING
C39B- D0 09     0135             BNE @K6
C39D- 78        0136             SEI
C39E- 20 AC C0  0137             JSR AVERAGE
C3A1- 20 B7 C0  0138             JSR MOVE
C3A4- 58        0139             CLI
C3A5- 60        0140             RTS
                0141 @K6
C3A6- C9 47     0142             CMP #'G              ;CYCLE ON EVERYTHING
C3A8- D0 05     0143             BNE @SERROR
C3AA- A9 00     0144             LDA #$00
C3AC- 85 1B     0145             STA *CYCLE
C3AE- 60        0146             RTS
                0147
                0148 @SERROR
C3AF- A2 11     0149             LDX #17              ;END OF MESSAGE
C3B1- 20 63 C4  0150             JSR LEDMESG          ;DISPLAY MESSAGE ON LEDS
C3B4- A9 80     0151             LDA #$80
C3B6- 85 20     0152             STA *LCNT2
C3B8- 4C 1F C3  0153             JMP @LOOP1
```

```
                GOMON
C3BB- A2 1D                     LDX #29
C3BD- 20 63 C4                  JSR LEDMESG          ;DISPLAY SY1.1.
C3C0- 38                        SEC
C3C1- 68                        PLA                  ;PULL LOOP RETURN
C3C2- 68                        PLA
C3C3- 68                        PLA                  ;SHOW RETURN
C3C4- E9 02                     SBC #$02             ;BACKUP TO CALL
C3C6- 8D 59 A6                  STA GOVEC
C3C9- 68                        PLA
C3CA- E9 00                     SBC #$00             ;SUB ANY BORROW
C3CC- 8D 5A A6                  STA GOVEC+1
C3CF- 4C 03 80                  JMP MONITOR          ;GO CR WILL CONT.

                ;PUT "L3SHOW.S"
                                .FI "L3GENERAL.S"

07D7  35E4-3DBB  L3GENERAL.S

                ;PUT "L3GENERAL.S"

                ;THIS ROUTINE DISPLAYS SOME GENERAL VARIABLES USED
                ;IN THE DATA LOGGING PROGRAM

                GENDISP
C3D2- A5 1D                     LDA *DISCNT+1        ;CHANNEL REQUESTED
C3D4- 20 FA 82                  JSR OUTBYT           ;DISPLAY CHANNEL

C3D7- C9 00                     CMP #0               ;CH00 LOGGER ID
C3D9- D0 06                     BNE @G1
C3DB- AD FF C7                  LDA LOGGERID
C3DE- 4C 02 C4                  JMP @GENBCD
                @G1
C3E1- C9 01                     CMP #1               ;CH01 ADCHANNELS
C3E3- D0 05                     BNE @G2
C3E5- A5 0B                     LDA *ADCHAN
C3E7- 4C 02 C4                  JMP @GENBCD
                @G2
C3EA- C9 02                     CMP #2               ;CH02 DGCHANNELS
C3EC- D0 05                     BNE @G3
C3EE- A5 0C                     LDA *DGCHAN
C3F0- 4C 02 C4                  JMP @GENBCD
                @G3
C3F3- C9 03                     CMP #3               ;CH03 PLCHANN
C3F5- D0 05                     BNE @G4
C3F7- A5 0D                     LDA *PLCHAN
C3F9- 4C 02 C4                  JMP @GENBCD
                @G4
C3FC- C9 04                     CMP #4               ;CH04 NUREC
C3FE- D0 0A                     BNE @G5
C400- A5 2B                     LDA *NUMREC
                @GENBCD
C402- 20 80 C4                  JSR BINDEC           ;CONVERT TO DECIMAL
                @GENOUT
C405- 20 F4 82                  JSR OUTXAH
C408- 18                        CLC
C409- 60                        RTS
                @G5
C40A- C9 05                     CMP #5               ;CH05 LENGTH
C40C- D0 19                     BNE @G6
C40E- A5 29                     LDA *LENGTH          ;LOW BYTE
C410- 20 80 C4                  JSR BINDEC
C413- F8                        SED
C414- A4 2A                     LDY *LENGTH+1        ;HIGH BYTE
C416- 88                        DEY
C417- 10 04                     BPL =+5
C419- D8                        CLD
C41A- 4C 05 C4                  JMP @GENOUT          ;PRINT #
C41D- 18                        CLC
C41E- 69 56                     ADC #$56             ;LOW BYTE OF 256
C420- 90 01                     BCC =+2
```

```
C422- E8         0053            INX                 ;CARRY
C423- E8         0054            INX                 ;200
C424- E8         0055            INX
C425- D0 EF      0056            BNE @G5+12          ;TO DEY
                 0057 @G6
C427- C9 06      0058            CMP #6              ;CH06 DATAEND
C429- D0 07      0059            BNE @G7
C42B- A6 05      0060            LDX *DATAEND+1
C42D- A5 04      0061            LDA *DATAEND
C42F- 4C 05 C4   0062            JMP @GENOUT
                 0063 @G7
C432- C9 07      0064            CMP #7              ;CH07 MEMEND
C434- D0 07      0065            BNE @G8
C436- A6 03      0066            LDX *MEMEND+1
C438- A5 02      0067            LDA *MEMEND
C43A- 4C 05 C4   0068            JMP @GENOUT
                 0069 @G8
C43D- C9 08      0070            CMP #8              ;CH08 TIMES SAMP,AVG,SAVE
C43F- D0 14      0071            BNE @G9
C441- 20 42 83   0072            JSR SPACE
C444- A5 0E      0073            LDA *SMPTIME
C446- 20 44 8A   0074            JSR OUTNIB
C449- A5 0F      0075            LDA *AVGTIME
C44B- 20 44 8A   0076            JSR OUTNIB
C44E- A5 10      0077            LDA *SAVTIME
C450- 20 44 8A   0078            JSR OUTNIB
C453- 18         0079            CLC
C454- 60         0080            RTS
                 0081 @G9
C455- C9 09      0082            CMP #9              ;CH09 A/DERR & LABEL
C457- D0 08      0083            BNE @G10
C459- 20 45 C1   0084            JSR CALCLABEL       ;GET LABEL
C45C- A6 3C      0085            LDX *ADERR
C45E- 4C 05 C4   0086            JMP @GENOUT
                 0087 @G10
C461- 38         0088            SEC
C462- 60         0089            RTS
                 0090
                 0091 ;LEDMESG: DISPLAY A HEX CODED MESSAGE ON THE
                 0092 ;LED DISPLAY LOCATED AT OFFSET IN X AND RETURN.
                 0093
                 0094 LEDMESG
C463- A0 05      0095            LDY #$05
C465- BD 26 C9   0096            LDA LEDMESSAGES,X
C468- 99 40 A6   0097            STA DISBUF,Y
C46B- CA         0098            DEX
C46C- 88         0099            DEY
C46D- 10 F6      0100            BPL LEDMESG+2
C46F- 60         0101            RTS
                 0102
                 0103 ;DO A BCD TO BINARY CONVERSION
                 0104
                 0105 BINARY
C470- 48         0106            PHA
C471- 4A         0107            LSR A
C472- 4A         0108            LSR A
C473- 4A         0109            LSR A
C474- 4A         0110            LSR A
C475- AA         0111            TAX
C476- 68         0112            PLA
C477- CA         0113            DEX
C478- 10 01      0114            BPL =+2
C47A- 60         0115            RTS
C47B- 38         0116            SEC
C47C- E9 06      0117            SBC #$06
C47E- B0 F7      0118            BCS =-8             ;ALWAYS
                 0119
                 0120 ;CONVERT BINARY NUMBER IN A TO DECIMAL IN X AND A
                 0121
                 0122 BINDEC
C480- 38         0123            SEC
C481- A2 FF      0124            LDX #$FF            ;FOR COMPENSATION
```

```
C483- E8         0125            INX
C484- E9 64      0126            SBC #100          ;SUBTRACT 100
C486- B0 FB      0127            BCS BINDEC+3
C488- 69 64      0128            ADC #100          ;ADD BACK 100.
C48A- 48         0129            PHA
C48B- 4A         0130            LSR A             ;GET TOP NIBBLE
C48C- 4A         0131            LSR A
C48D- 4A         0132            LSR A
C48E- 4A         0133            LSR A
C48F- A8         0134            TAY               ;SAVE COUNTER
C490- 68         0135            PLA
C491- 18         0136            CLC
C492- F8         0137            SED
C493- 69 00      0138            ADC #$00          ;ADJUST TO DECIMAL
C495- 88         0139            DEY
C496- 30 06      0140            BMI =+7           ;TO CLD
C498- 18         0141            CLC
C499- 69 06      0142            ADC #$06          ;ADD Y*6
C49B- 4C 95 C4   0143            JMP =-6           ;TO DEY
C49E- D8         0144            CLD
C49F- 60         0145            RTS
                 0146
                 0147 ;PUT "L3GENERAL.S"
                 0055            .FI "L3CLOCK.S"

0D11  35E4-42F5  L3CLOCK.S

                 0001 ;PUT "L3CLOCK.S"
                 0002
                 0003 CLKDISP
C4A0- A5 1D      0004            LDA *DISCNT+1
C4A2- C9 00      0005            CMP #0
C4A4- F0 15      0006            BEQ @DATEDISP
C4A6- C9 01      0007            CMP #1
C4A8- F0 24      0008            BEQ @TIMEDISP
C4AA- C9 0A      0009            CMP #10
C4AC- F0 06      0010            BEQ @DATEDISP-7
C4AE- C9 0B      0011            CMP #11
C4B0- F0 15      0012            BEQ @TIMEDISP-7
C4B2- 38         0013            SEC
C4B3- 60         0014            RTS
                 0015
C4B4- 20 09 C5   0016            JSR SETDATE
C4B7- A9 00      0017            LDA #$00          ;RESET CNTR TO DATEDISP
C4B9- 85 1D      0018            STA *DISCNT+1
                 0019 @DATEDISP
C4BB- A5 23      0020            LDA *DAY+1
C4BD- 20 FA 82   0021            JSR OUTBYT
C4C0- A6 24      0022            LDX *DAY+2
C4C2- A5 22      0023            LDA *DAY
C4C4- 4C D7 C4   0024            JMP @TIMEDISP+9
                 0025
C4C7- 20 EC C4   0026            JSR SETIME
C4CA- A9 01      0027            LDA #$01          ;RESET CNTR TO TIMEDISP
C4CC- 85 1D      0028            STA *DISCNT+1
                 0029 @TIMEDISP
C4CE- A5 25      0030            LDA *DAY+3
C4D0- 20 FA 82   0031            JSR OUTBYT
C4D3- A6 26      0032            LDX *DAY+4
C4D5- A5 27      0033            LDA *DAY+5
C4D7- 20 F4 82   0034            JSR OUTXAH
C4DA- AD 41 A6   0035            LDA DISBUF+1
C4DD- 09 80      0036            ORA #$80
C4DF- 8D 41 A6   0037            STA DISBUF+1
C4E2- AD 43 A6   0038            LDA DISBUF+3
C4E5- 09 80      0039            ORA #$80
C4E7- 8D 43 A6   0040            STA DISBUF+3
C4EA- 18         0041            CLC               ;DISCNT OK
C4EB- 60         0042            RTS
                 0043
                 0044 ;SET THE TIME HOURS AND MINUTES AND
                 0045 ;RESET THE SECONDS TO ZERO

                 SETIME
C4EC- A2 0B                     LDX #11
C4EE- 20 D5 C5                  JSR @GETSET          ;GET TIME
C4F1- C9 02                     CMP #$02
C4F3- D0 31                     BNE READCLOCK-3
C4F5- A2 05                     LDX #$05
C4F7- AD 4C A6                  LDA PARN+2           ;HOUR
C4FA- 09 80                     ORA #$80             ;SET 24 HOUR FORMAT
C4FC- 20 6C C5                  JSR @SETBYTE
C4FF- AD 4A A6                  LDA PARN             ;MINUTE
C502- 20 6C C5                  JSR @SETBYTE
C505- A9 00                     LDA #00              ;SECONDS
C507- F0 1A                     BEQ READCLOCK-6

                 ;SET THE YEAR,MONTH,AND DAY TO THE CLOCK
                 ;WITHOUT THE HOLD.

                 SETDATE
C509- A2 05                     LDX #5
C50B- 20 D5 C5                  JSR @GETSET
C50E- C9 03                     CMP #$03
C510- D0 14                     BNE READCLOCK-3
C512- A2 0C                     LDX #12
C514- AD 4A A6                  LDA PARN             ;YEAR
C517- 20 6C C5                  JSR @SETBYTE
C51A- AD 4E A6                  LDA PARN+4           ;MONTH
C51D- 20 6C C5                  JSR @SETBYTE
C520- AD 4C A6                  LDA PARN+2           ;DAY
C523- 20 6C C5                  JSR @SETBYTE
C526- 20 B0 C5                  JSR SETUPCLOCK

                 ;SETIME AND SETDATE FALL THRU TO READCLOCK HERE
                 ;SUBROUTINE TO READ TIME FROM CLOCK
                 ;AND STORE IT IN 6 BYTES AT DAY

                 READCLOCK
C529- A9 06                     LDA #$06             ;HOLD & READ
C52B- 20 8D C5                  JSR @STOPCLOCK
C52E- A2 05                     LDX #5
C530- A9 00                     LDA #0
C532- 8D 01 A0                  STA VIA1+1
C535- 20 4D C5                  JSR @READ3           ;READ HH:MM:SS
C538- EE 01 A0                  INC VIA1+1           ;SKIP DAY OF WK
C53B- 20 4D C5                  JSR @READ3           ;READ YY:MM:DD
C53E- A5 25                     LDA *DAY+3           ;CLEAR 24 HR FLAG
C540- 29 3F                     AND #$3F
C542- 85 25                     STA *DAY+3
C544- A5 24                     LDA *DAY+2           ;CLEAR LEAPYEAR FLAG
C546- 29 3F                     AND #$3F
C548- 85 24                     STA *DAY+2
C54A- 4C B0 C5                  JMP SETUPCLOCK

                 ;READ 3 BYTES OF DATA FROM THE CLOCK
                 ;AND PUT IT AT DAY,X TO DAY,X-2
                 ;LEAVE X = X-3.

                 @READ3
C54D- A0 03                     LDY #$03
C54F- AD 01 A0                  LDA VIA1+1
C552- EE 01 A0                  INC VIA1+1           ;NEXT NIBBLE
C555- 4A                        LSR A
C556- 4A                        LSR A
C557- 4A                        LSR A
C558- 4A                        LSR A
C559- 95 22                     STA *DAY,X           ;LOWER NIBBLE
C55B- AD 01 A0                  LDA VIA1+1
C55E- EE 01 A0                  INC VIA1+1           ;NEXT NIBBLE
C561- 29 F0                     AND #%11110000       ;HIGH NIBBLE
C563- 15 22                     ORA *DAY,X           ;MERGE NIBBLES
C565- 95 22                     STA *DAY,X
C567- CA                        DEX
```

```
C568- 88                        DEY
C569- D0 E4                     BNE @READ3+2
C56B- 60                        RTS

            ;SET 1 BYTE TO CLOCK AT LOCATION IN X
            ;AND X-1.

            @SETBYTE
C56C- 48                        PHA
C56D- 29 F0                     AND #$F0
C56F- 20 77 C5                  JSR =+8
C572- 68                        PLA
C573- 0A                        ASL A
C574- 0A                        ASL A
C575- 0A                        ASL A
C576- 0A                        ASL A
C577- 86 22                     STX *DAY             ;USE AS TMP LOC
C579- 05 22                     ORA *DAY
C57B- 8D 01 A0                  STA VIA1+1
C57E- AD 00 A0                  LDA VIA1
C581- 29 FE                     AND #%11111110       ;WRITE PULSE
C583- 8D 00 A0                  STA VIA1             ;WRITE
C586- 09 01                     ORA #%00000001       ;CLR WRITE(INV)
C588- 8D 00 A0                  STA VIA1
C58B- CA                        DEX
C58C- 60                        RTS

            ;THIS ROUTINES DISABLES THE INTERRUPTS
            ;AND SETS THE OUTPUT LINES AS SET IN A
            ;2 PB IS SET TO OUTPUT IF WRITE FLAG IS SET

            @STOPCLOCK
C58D- 78                        SEI
C58E- 85 22                     STA *DAY
C590- A9 10                     LDA #$10             ;DISABLE CB1 INT
C592- 8D 0E A0                  STA VIA1+14          ;DISABLE INTERRUPTS
C595- AD 00 A0                  LDA VIA1
C598- 09 07                     ORA #$07             ;SET STATUS BITS
C59A- 45 22                     EOR *DAY             ;INV NEW STAT BITS
C59C- 8D 00 A0                  STA VIA1
C59F- A9 02                     LDA #$02             ;READ BIT
C5A1- 24 22                     BIT *DAY             ;SET FOR READ?
C5A3- D0 05                     BNE =+6
C5A5- A9 FF                     LDA #$FF
C5A7- 8D 03 A0                  STA VIA1+3           ;ALL OUTPUTS
C5AA- A2 1E                     LDX #30              ;DELAY FOR HOLD SETUP
C5AC- CA                        DEX
C5AD- D0 FD                     BNE =-2
C5AF- 60                        RTS

            ;ROUTINE TO SETUP THE VIA'S FOR THE CLOCK
            ;SUBROUTINES

            SETUPCLOCK
C5B0- AD 00 A0                  LDA VIA1
C5B3- 29 F8                     AND #%11111000       ;CLR STAT
C5B5- 09 05                     ORA #%00000101       ;SET NEG STAT
C5B7- 8D 00 A0                  STA VIA1             ;SET READ ONLY MODE
C5BA- AD 02 A0                  LDA VIA1+2
C5BD- 09 07                     ORA #%00000111       ;STAT BITS OUT
C5BF- 8D 02 A0                  STA VIA1+2
C5C2- A9 0F                     LDA #$0F
C5C4- 8D 01 A0                  STA VIA1+1           ;SET FOR INTERRUPTS
C5C7- 8D 03 A0                  STA VIA1+3           ;DATA DIRECTION REG.
C5CA- A9 10                     LDA #$10             ;CB1 INT
C5CC- 8D 0D A0                  STA VIA1+13          ;RESET INTERRUPTS
C5CF- A9 90                     LDA #$90             ;CB1 INT
C5D1- 8D 0E A0                  STA VIA1+14          ;ENABLE INTERRUPTS
C5D4- 60                        RTS

            ;PRINT A MESSAGE ON DISPLAY & GET PARAMETERS

```

```
                   0190 @GETSET
C5D5- 20 63 C4     0191            JSR LEDMESG        ;DISPLAY MESSAGE ON LEDS
C5D8- 08           0192            PHP                ;SAVE FLAGS
C5D9- 58           0193            CLI                ;ENABLE INTER DURING ENTRY
C5DA- 20 20 82     0194            JSR PARM           ;GET TIME DATA
C5DD- A9 04        0195            LDA #$04           ;SET FOR READ
C5DF- 20 8D C5     0196            JSR @STOPCLOCK
C5E2- AD 49 A6     0197            LDA PARNR          ;NUMBER OF PARMS
C5E5- 28           0198            PLP                ;RESTORE FLAGS
C5E6- 60           0199            RTS
                   0200
                   0201 ;PUT "L3CLOCK.S"
                   0056            .FI "L3ANALOG1.S"

0AEC  35E4-40D0  L3ANALOG1.S

                   0001 ;PUT "L3ANALOG1.S"
                   0002
                   0003 ;THIS SUBROUTINE STARTS A/D CONVERSION.  IT ASSUMES
                   0004 ;THAT THE CHANNEL NUMBER WAS IN THE ACCUMULATOR
                   0005 ;BEFORE CALLING THIS ROUTINE.
                   0006
                   0007 STRTCONV
C5E7- 29 3F        0008            AND #$3F           ;64 CHANNEL LIMIT
C5E9- 85 3D        0009            STA *CHANN         ;STORE CHAN#
C5EB- AA           0010            TAX
C5EC- 20 10 C6     0011            JSR SETUPA/D
C5EF- 8E 00 A8     0012            STX VIA2+0         ;OUT CHAN# TO MUX & ETC
C5F2- B5 40        0013            LDA *GAIN,X
C5F4- 29 0F        0014            AND #$0F           ;CLR HIGH BITS
C5F6- A8           0015            TAY
C5F7- B9 52 C9     0016            LDA OUTGAIN,Y      ;LOOK UP CODE FOR GAIN
C5FA- 8D 01 A8     0017            STA VIA2+1         ;SET GAIN
C5FD- AD 0C A8     0018            LDA VIA2+12        ;START CONVERSION
C600- 09 02        0019            ORA #$02
C602- 8D 0C A8     0020            STA VIA2+12
C605- 29 FD        0021            AND #$FD
C607- A2 32        0022            LDX #50            ;250 MICROSEC DELAY
C609- CA           0023            DEX
C60A- D0 FD        0024            BNE =-2
C60C- 8D 0C A8     0025            STA VIA2+12
C60F- 60           0026            RTS                ;RETURN
                   0027
                   0028 ;THIS ROUTINE SETS UP I/O FOR A/D ROUTINE
                   0029
                   0030 SETUPA/D
C610- AD 0C A8     0031            LDA VIA2+12
C613- 29 F0        0032            AND #$F0
C615- 09 0D        0033            ORA #$0D           ;SET CAI TO POS EDGE
C617- 8D 0C A8     0034            STA VIA2+12        ;SET FOR LOW OUTPUT
C61A- A9 0F        0035            LDA #$0F           ;SET PA I/O
C61C- 8D 03 A8     0036            STA VIA2+3
C61F- AD 02 A8     0037            LDA VIA2+2
C622- 09 3F        0038            ORA #$3F
C624- 8D 02 A8     0039            STA VIA2+2         ;SET PB I/O
C627- 60           0040            RTS
                   0041
                   0042 ;THIS ROUTINE INPUTS A/D DATA AND PUTS IT IN THE PROPER
                   0043 ;FLOATING PNT LOCATIONS.
                   0044
                   0045 INPA/D
C628- A2 73        0046            LDX #115           ;2MS MAX WAIT TIME
C62A- A9 02        0047            LDA #$02
C62C- E6 3C        0048            INC *ADERR         ;ERROR BYTE
C62E- CA           0049            DEX
C62F- F0 07        0050            BEQ @NOA/D
C631- C6 3C        0051            DEC *ADERR         ;NO ERROR RESET
C633- 2C 0D A8     0052            BIT VIA2+13
C636- F0 F4        0053            BEQ INPA/D+4
                   0054 @NOA/D
C638- A2 10        0055            LDX #$10           ;GET MS NIBBLE
C63A- 20 A4 C6     0056            JSR @INPNIB        ;GET HIGH NIBBLE
```

```
C63D- 85 C2                STA *FPMSW
C63F- A2 08                LDX #$08               ;GET NEXT SIG NIBBLE
C641- 20 A4 C6             JSR @INPNIB            ;GET 2ND NIBBLE
C644- 4A                   LSR A
C645- 4A                   LSR A
C646- 4A                   LSR A
C647- 4A                   LSR A
C648- 05 C2                ORA *FPMSW
C64A- 85 C2                STA *FPMSW
C64C- 85 3F                STA *HEXAD+1           ;KEEP HEX CONVERSION
C64E- A2 00                LDX #$00               ;GET 3RD NIBBLE
C650- 20 A4 C6             JSR @INPNIB
C653- 85 C1                STA *FPNSW
C655- 85 3E                STA *HEXAD
                ;CHECK FOR OVERFLOW
C657- A6 3D                LDX *CHANN             ;GET CHAN#
C659- B5 40                LDA *GAIN,X            ;LOOK UP GAIN FOR CHAN#
C65B- F0 1B                BEQ @ADOK              ;GAIN AT LOWEST LEVEL
C65D- 30 19                BMI @ADOK              ;DONT CHANGE GAIN IF MINUS
C65F- A4 C0                LDY *FPLSW             ;CHECK FOR OVERFLOW
C661- C0 F0                CPY #$F0
C663- F0 06                BEQ =+7
C665- A4 BF                LDY *FPLSWE            ;CHECK FOR UNDERFLOW
C667- C0 00                CPY #$00
C669- D0 0D                BNE @ADOK
C66B- D6 40                DEC *GAIN,X            ;OUT OF RANGE, REDUCE GAIN
C66D- F0 02                BEQ =+3
C66F- D6 40                DEC *GAIN,X
C671- 8A                   TXA                    ;LDA WITH CHANNEL #
C672- 20 E7 C5             JSR STRTCONV
C675- 4C 28 C6             JMP INPA/D
                @ADOK
C678- 29 0F                AND #$0F               ;CLR ANY HIGH BITS
C67A- 49 FF                EOR #$FF               ;2'S COMPLEMENT
C67C- 85 C3                STA *FPACCE            ;PUT IN EXPONENT
C67E- E6 C3                INC *FPACCE
C680- A9 00                LDA #$00
C682- 85 C0                STA *FPLSW
C684- 85 BF                STA *FPLSWE            ;4 BYTES FOR FPNORM
C686- 20 71 CC             JSR FPNORM
C689- A6 3D                LDX *CHANN
C68B- B5 40                LDA *GAIN,X            ;CHECK GAIN
C68D- 30 14                BMI @SKIPGAIN
C68F- A5 C2                LDA *FPMSW             ;IS CONVERSION ZERO?
C691- F0 0C                BEQ @SKIPGAIN-4        ;THEN MAXGAIN
C693- A5 C3                LDA *FPACCE
C695- 49 FF                EOR #$FF               ;COMPLEMENT
C697- 10 02                BPL =+3                ;TEST FOR GAIN TOO LOW
C699- A9 00                LDA #$00               ;GAIN LIMIT
C69B- C9 0B                CMP #$0A+1             ;UPPER GAIN LIMIT
C69D- 90 02                BCC =+3
C69F- A9 0A                LDA #$0A
C6A1- 95 40                STA *GAIN,X
                @SKIPGAIN
C6A3- 60                   RTS

                ;INPUT A NIBBLE FROM THE ADC WITH THE ADDRESS IN X

                @INPNIB
C6A4- A9 38                LDA #$38               ;DESLECT INPUT LINES
C6A6- 8D 00 A8             STA VIA2+0
C6A9- 8E 00 A8             STX VIA2+0
C6AC- AD 01 A8             LDA VIA2+1             ;GET DATA
C6AF- 29 F0                AND #$F0
C6B1- A8                   TAY                    ;SAVE DATA
C6B2- E0 10                CPX #$10
C6B4- D0 08                BNE =+9
C6B6- 49 80                EOR #$80
C6B8- 85 C0                STA *FPLSW             ;TMP LOCS FOR OVFL CHK
C6BA- 85 BF                STA *FPLSWE
C6BC- 98                   TYA
C6BD- 60                   RTS
```

```
C6BE- 25 C0     0129            AND *FPLSW
C6C0- 85 C0     0130            STA *FPLSW
C6C2- 98        0131            TYA
C6C3- 05 BF     0132            ORA *FPLSWE
C6C5- 4C BA C6  0133            JMP =-11
                0134
                0135 ;THIS ROUTINE PUTS TOGETHER THE
                0136 ;INPA/D, FPNORM, & MULT BY 5 ROUTINES.
                0137
                0138 INPA/DV
C6C8- 20 E7 C5  0139            JSR STRTCONV         ;START A/D CONV
C6CB- A9 50     0140            LDA #$50
C6CD- 85 CA     0141            STA *FOPMSW          ;PUT 5 IN FPOP
C6CF- A9 00     0142            LDA #$00
C6D1- 85 C9     0143            STA *FOPNSW
C6D3- 85 C8     0144            STA *FOPLSW
C6D5- A9 03     0145            LDA #$03
C6D7- 85 CB     0146            STA *FOPEXP          ;PUT 3 IN EXPONENT
C6D9- 20 28 C6  0147            JSR INPA/D
C6DC- 4C 26 CD  0148            JMP FPMULT
                0149
                0150 ;PUT "L3ANALOG1.S"
                0057            .FI "L3ANALOG2.S"

0820  35E4-3E04  L3ANALOG2.S

                0010 ;PUT "L3ANALOG2.S"
                0020
                0030 ;SUM ADC CHANNELS INTO THE DATA AREA, DON'T DIVIDE
                0040 ;BY 5. THAT IS DONE ONCE LATER IN AVERAGE.
                0050
                0060 SUMADC
C6DF- A5 0B     0070            LDA *ADCHAN          ;CHECK FOR CHANNELS
C6E1- D0 01     0080            BNE =+2
C6E3- 60        0090            RTS
C6E4- A9 00     0100            LDA #0
C6E6- 20 E7 C5  0110            JSR STRTCONV         ;STARTS FIRST CHANNEL CONV
                0120            LDZZ (WORKPNT ADADDR)

C6E9- A5 11
C6EB- 85 19
C6ED- A5 12
C6EF- 85 1A

                0130 @SUMADC2
                0140            LDZI (TOPNT FOPLSW)

C6F1- A9 C8
C6F3- 85 BA
C6F5- A9 00
C6F7- 85 BB

                0150            LDZZ (FMPNT WORKPNT)

C6F9- A5 19
C6FB- 85 B8
C6FD- A5 1A
C6FF- 85 B9

C701- A2 04     0160            LDX #4
C703- 20 0A CC  0170            JSR MOVIND           ;GET SUM
C706- 20 28 C6  0180            JSR INPA/D           ;GET NUMBER WITHOUT MULT B
C709- E6 3D     0190            INC *CHANN
C70B- A5 3D     0200            LDA *CHANN
C70D- C5 0B     0210            CMP *ADCHAN          ;END OF CHANNELS
C70F- B0 03     0220            BCS =+4
C711- 20 E7 C5  0230            JSR STRTCONV         ;START NEXT CONV WHILE PRO
C714- 20 B1 CC  0240            JSR FPADD
                0250            LDZI (FMPNT FPLSW)

C717- A9 C0
C719- 85 B8
```

```
C71B- A9 00
C71D- 85 B9

                0260            LDZZ (TOPNT WORKPNT)

C71F- A5 19
C721- 85 BA
C723- A5 1A
C725- 85 BB

C727- A2 04     0270            LDX #4
C729- 20 0A CC  0280            JSR MOVIND            ;SAVE SUM
                0290            INCD (WORKPNT 4)

C72C- 18
C72D- A9 04
C72F- 65 19
C731- 85 19
C733- 90 02
C735- E6 1A

C737- A5 3D     0300            LDA *CHANN            ;IS THERE ANOTHER CONV
C739- C5 0B     0310            CMP *ADCHAN
C73B- 90 B4     0320            BCC @SUMADC2
C73D- 60        0330            RTS
                0340
                0350            IFN AVGANALOG
                0360
                0370 ;AVERAGE THE ADC DATA. ALSO MULTIPLY BY
                0380 ; 5 TO CONVERT TO VOLTS
                0390
                0400 AVGADC
                0410            LDA #$50              ;5 VOLTS
                0420            STA *FOPMSW
                0430            LDA #3
                0440            STA *FOPEXP
                0450            LDY #HEADER+6
                0460            LDA (DATASTART),Y     ;COUNT TO DIVIDE BY
                0470            STA *FPNSW
                0480            INY
                0490            LDA (DATASTART),Y
                0500            STA *FPMSW            ;ITS LESS THAN 32000
                0510            LDA #15
                0520            STA *FPACCE           ;15 BITS WITHOUT SIGN
                0530            LDA #0
                0540            STA *FPLSW
                0550            STA *FOPLSW
                0560            STA *FOPNSW
                0570            JSR FPNORM            ;NORMALIZE COUNT TO FP
                0580            JSR FPDIV
                0590            LDZI (FMPNT FPLSW)
                0600            LDZI (TOPNT DIVTMP)
                0610            LDX #4
                0620            JSR MOVIND
                0630            LDA #0
                0640            STA *CHANN
                0650            LDZZ (WORKPNT ADADDR)
                0660 @AVGADC2
                0670            LDZZ (FMPNT WORKPNT)
                0680            LDZI (TOPNT FPLSW)
                0690            LDX #4
                0700            JSR MOVIND
                0710            LDZI (FMPNT DIVTMP)
                0720            LDZI (TOPNT FOPLSW)
                0730            LDX #4
                0740            JSR MOVIND
                0750            JSR FPMULT            ;SCALE NUMBER
                0760            LDZI (FMPNT FPLSW)
                0770            LDZZ (TOPNT WORKPNT)
                0780            LDX #4
                0790            JSR MOVIND            ;SAVE AVERAGE
                0800            INCD (WORKPNT 4)
```

```
                   0810            INC *CHANN
                   0820            LDA *CHANN
                   0830            CMP *ADCHAN          ;LAST NUMBER
                   0840            BCC @AVGADC2
                   0850            RTS
                   0860            ***
                   0870
                   0880 ;THIS ROUTINE CALLS A/DV & FPOUT TO DISPLAY THE A/D
                   0890 ;CONVERSION ON THE CHANNEL IN *DISBUF+1.  THE VALUE
                   0900 ;IS MULTIPLIED BY 1000 TO DISPLAY THE NUMBER IN
                   0910 ;MILLIVOLTS.
                   0920
                   0930 ADDISP
C73E- A5 1D        0940            LDA *DISCNT+1        ;CHANNEL #
C740- C5 0B        0950            CMP *ADCHAN          ;CHECK
C742- 90 02        0960            BCC =+3
C744- 38           0970            SEC
C745- 60           0980            RTS
C746- 20 C8 C6     0990            JSR INPA/DV          ;GET CONVERSION
C749- A9 7D        1000            LDA #$7D
C74B- 85 CA        1010            STA *FOPMSW
C74D- A9 00        1020            LDA #$00
C74F- 85 C9        1030            STA *FOPNSW
C751- 85 C8        1040            STA *FOPLSW
C753- A9 0A        1050            LDA #$0A
C755- 85 CB        1060            STA *FOPEXP
C757- 20 26 CD     1070            JSR FPMULT           ;CHANGE TO MILLIVOLTS
C75A- A5 1D        1080            LDA *DISCNT+1        ;DISPLAY CHANNEL
C75C- 20 80 C4     1090            JSR BINDEC
C75F- 20 FA 82     1100            JSR OUTBYT
C762- 20 8F CF     1110            JSR FPDISP
C765- 18           1120            CLC
C766- 60           1130            RTS
                   1140
                   1150 ;PUT "L3ANALOG2.S"
                   0058            .FI "L3DIGITAL.S"

04AF  35E4-3A93  L3DIGITAL.S

                   0001 ;PUT "L3DIGITAL.S"
                   0002
                   0003 ;DGREAD -- READ DIGITAL CHANNELS & STORE IN DGBYTE
                   0004 ;ASSUME DDR'S ARE SET RIGHT FROM RESET DEFAULTS.
                   0005
                   0006 DGREAD
C767- AD 00 AC     0007            LDA VIA3
C76A- 29 BF        0008            AND #$BF             ;CLR PB6
C76C- 85 3B        0009            STA *DGBYTE
C76E- AD 00 A8     0010            LDA VIA2
C771- 4A           0011            LSR A
C772- 29 40        0012            AND #$40
C774- 05 3B        0013            ORA *DGBYTE          ;MERGE BYTES
C776- 85 3B        0014            STA *DGBYTE          ;SAVE CHANNEL READINGS
C778- 60           0015            RTS
                   0016
                   0017 ;DGSAVE -- SUM DIGITAL BITS TO THE LOGGER STOREAGE.
                   0018
                   0019 DGSAVE
C779- A6 0C        0020            LDX *DGCHAN
C77B- F0 FB        0021            BEQ DGSAVE-1         ;NO CHANNELS - RETURN
C77D- A0 00        0022            LDY #0
C77F- A5 3B        0023            LDA *DGBYTE          ;BITS
C781- 4A           0024            LSR A
C782- 48           0025            PHA
C783- B1 13        0026            LDA (DGADDR),Y       ;ADD BIT
C785- 69 00        0027            ADC #0               ;ADD CARRY
C787- 91 13        0028            STA (DGADDR),Y
C789- C8           0029            INY
C78A- B1 13        0030            LDA (DGADDR),Y       ;ADD CARRY FROM LOW BYTE
C78C- 69 00        0031            ADC #0               ;ADD CARRY
C78E- 91 13        0032            STA (DGADDR),Y
C790- C8           0033            INY
```

```
C791- 68        0034            PLA
C792- CA        0035            DEX
C793- D0 EC     0036            BNE DGSAVE+8          ;GO DO NEXT
C795- 60        0037            RTS
                0038
                0039 ;DGDISP -- DISPLAY DIGITAL CHANNELS EITHER
                0040 ;ON OR OFF ON THE DISPLAY.
                0041
                0042 DGDISP
C796- A5 1D     0043            LDA *DISCNT+1         ;CHANNEL #
C798- C5 0C     0044            CMP *DGCHAN           ;LIMIT
C79A- 90 01     0045            BCC =+2
C79C- 60        0046            RTS
C79D- AA        0047            TAX
C79E- 20 44 8A  0048            JSR OUTNIB            ;OUTPUT CHAN #
C7A1- A9 5E     0049            LDA #$5E              ;OUTPUT "D"
C7A3- 8D 40 A6  0050            STA DISBUF
C7A6- AD 45 A6  0051            LDA RMDIG
C7A9- 8D 41 A6  0052            STA DISBUF+1          ;CHANNEL # CONVERTED
C7AC- A9 00     0053            LDA #0
C7AE- 8D 42 A6  0054            STA DISBUF+2          ;SPACE
C7B1- 8D 45 A6  0055            STA DISBUF+5
C7B4- A9 3F     0056            LDA #$3F
C7B6- 8D 43 A6  0057            STA DISBUF+3          ;OUTPUT "O"
C7B9- 20 67 C7  0058            JSR DGREAD            ;GET CURRENT BITS
C7BC- 4A        0059            LSR A
C7BD- CA        0060            DEX
C7BE- 10 FC     0061            BPL =-3
C7C0- B0 09     0062            BCS =+10              ;CARRY IS BIT HERE
C7C2- A9 71     0063            LDA #$71              ;OUTPUT "F"
C7C4- 8D 45 A6  0064            STA DISBUF+5
C7C7- 8D 44 A6  0065            STA DISBUF+4
C7CA- 60        0066            RTS
C7CB- 18        0067            CLC
C7CC- A9 54     0068            LDA #$54              ;OUTPUT "N"
C7CE- D0 F7     0069            BNE =-8
                0070
                0071 ;PUT "L3DIGITAL.S"
                0059
                0060            .BA $C800             ;SECOND EPROM
                0061            .MC $1000
                0062            .FI "L3PULSE.S"

103C  35E4-4620  L3PULSE.S

                0001 ;PUT "L3PULSE.S"
                0002
                0003 ;THESE ROUTINES ARE THE PULSE COUNTING PACKAGE
                0004 ;FOR THE LOGGER
                0005
                0006 ;PLSAVE -- CONTROL PULSE SAVING IF THERE IS
                0007 ;PULSE CNTRS FOR DISPLAY AND DATA LOGGING
                0008
                0009 PLSAVE
C800- A5 0D     0010            LDA *PLCHAN
C802- F0 11     0011            BEQ =+18              ;SKIP IF THERE IS NONE
C804- 20 BF C8  0012            JSR MRKSEC            ;MARK TIME FOR PULSES
C807- 20 16 C8  0013            JSR PLCHK             ;READ PULSES
C80A- D0 09     0014            BNE =+10              ;BAD COUNT, DONT ADD
C80C- 20 70 C8  0015            JSR PLSAVEDIS         ;SAVE FOR DISPLAY
C80F- 20 97 C8  0016            JSR PLSAVCNT          ;SAVE FOR DATA LOGGING
C812- 20 D3 C8  0017            JSR PLTIMESAV         ;SAVE CNTER TIME
C815- 60        0018            RTS
                0019
                0020 ;PLCHK -- READS THE PULSE CHANNELS INTO MEMORY
                0021 ;AND RESETS THE TIMERS TO DO PULSE COUNTING
                0022 ;N=1 IF RESULTS VALID OTHERWISE N=0
                0023
                0024 PLCHK
C816- A2 08     0025            LDX #8                ;REGISTER OF VIA
C818- 86 BA     0026            STX *TOPNT
C81A- E8        0027            INX
```

```
C81B- 86 B8                      STX *FMPNT          ;HIGH COUNTER
C81D- A2 A8                      LDX #H,VIA2
C81F- 86 BB                      STX *TOPNT+1
C821- 86 B9                      STX *FMPNT+1
C823- A2 00                      LDX #0              ;CHANNEL OFFSET
C825- 20 38 C8                   JSR GOPULSE         ;GET PULSE DATA & RESET
C828- 85 2C                      STA *PLTMP          ;SAVE ERROR STATUS
C82A- A2 AC                      LDX #H,VIA3         ;SECOND PULSE CNTER
C82C- 86 BB                      STX *TOPNT+1
C82E- 86 B9                      STX *FMPNT+1
C830- A2 02                      LDX #2
C832- 20 38 C8                   JSR GOPULSE
C835- 05 2C                      ORA *PLTMP
C837- 60                         RTS

            ;GOPULSE -- READ THE PULSE COUNTERS AND STORE
            ;THE VALUES IN PLSEC OFFSET BY PLTMP. IF THE
            ;COUNTER OVER RAN THE RANGE OR WAS NOT SETUP
            ;RETURN WITH A != 0 ELSE RETURN A=0

            GOPULSE
C838- A0 03                      LDY #3              ;OFFSET FROM T2
C83A- B1 BA                      LDA (TOPNT),Y       ;AUX CNTL BYTE
C83C- 48                         PHA
C83D- 09 20                      ORA #$20            ;SET T2 BIT
C83F- 91 BA                      STA (TOPNT),Y       ;TIMER INPUT ON PB6
C841- 68                         PLA
C842- 49 20                      EOR #$20            ;WAS T2 BIT SET
C844- A0 05                      LDY #5
C846- 11 BA                      ORA (TOPNT),Y       ;OR INT FLG SET
C848- 29 20                      AND #$20            ;ONLY T2 BITS
C84A- 48                         PHA
C84B- 20 56 C8                   JSR READPL          ;GET READINGS ON PULSE
C84E- A9 FF                      LDA #$FF
C850- 91 BA                      STA (TOPNT),Y       ;RESTART CNTERS
C852- 91 B8                      STA (FMPNT),Y
C854- 68                         PLA
C855- 60                         RTS

            ;READPL -- READ PULSE COUNTER.  IF COUNT HAS
            ;CHANGED FROM READING LOW BYTE TO HIGH BYTE
            ;REREAD THE COUNT.

            READPL
C856- A0 00                      LDY #0
C858- 20 63 C8                   JSR =+11            ;READ CNTR
C85B- B1 BA                      LDA (TOPNT),Y       ;LOW BYTE CHANGED?
C85D- 49 FF                      EOR #$FF            ;INVERT #
C85F- D5 2F                      CMP *PLSEC,X        ;SAVE AS LAST READING
C861- F0 0C                      BEQ =+13            ;BRANCH IN NOT

C863- B1 BA                      LDA (TOPNT),Y       ;READ LOW BYTE FIRST
C865- 49 FF                      EOR #$FF            ;INVERT
C867- 95 2F                      STA *PLSEC,X        ;SAVE LOW BYTE
C869- B1 B8                      LDA (FMPNT),Y       ;READ HIGH BYTE
C86B- 49 FF                      EOR #$FF
C86D- 95 30                      STA *PLSEC+1,X      ;SAVE HIGH BYTE
C86F- 60                         RTS

            ;PLSAVEDIS -- SAVE PULSE COUNT FOR DISPLAY
            ;SAVE THE SECOND COUNT, PREVIOUS MINUITE CNT,
            ;AND KEEP THE SUM FOR THE CURRENT MINUITE.

            PLSAVEDIS
C870- A5 27                      LDA *DAY+5          ;SEC =0
C872- D0 0D                      BNE =+14            ;TO LDX
C874- A2 03                      LDX #3              ;MOVE SUM TO MIN CNT
C876- A0 00                      LDY #0              ;ZERO SUM
C878- B5 37                      LDA *PLSUM,X
C87A- 95 33                      STA *PLMIN,X
C87C- 94 37                      STY *PLSUM,X
C87E- CA                         DEX
```

```
C87F- 10 F7                     BPL =-8            ;TO LDA
C881- A2 00                     LDX #0
C883- 18                        CLC
C884- B5 2F                     LDA *PLSEC,X
C886- 75 37                     ADC *PLSUM,X       ;SUM SECONDS FOR MIN
C888- 95 37                     STA *PLSUM,X
C88A- E8                        INX
C88B- B5 2F                     LDA *PLSEC,X
C88D- 75 37                     ADC *PLSUM,X
C88F- 95 37                     STA *PLSUM,X       ;16BIT SUM ONLY
C891- E8                        INX
C892- E0 04                     CPX #4             ;TWO CNTRS
C894- D0 ED                     BNE =-18           ;TO CLC
C896- 60                        RTS

              ;PLSAVCNT -- SAVE PULSE CNT IN THE DATA SUMMING
              ;LOCATIONS. SAVE AS A 4BYTE INTEGER LOW BYTE FIRST

              PLSAVCNT
C897- A0 02                     LDY #2             ;FIRST 2 BYTES CLOCK CNT
C899- A2 00                     LDX #0
C89B- 18                        CLC
C89C- B1 17                     LDA (PLADDR),Y
C89E- 75 2F                     ADC *PLSEC,X
C8A0- 91 17                     STA (PLADDR),Y
C8A2- E8                        INX
C8A3- C8                        INY
C8A4- B1 17                     LDA (PLADDR),Y
C8A6- 75 2F                     ADC *PLSEC,X
C8A8- 91 17                     STA (PLADDR),Y
C8AA- E8                        INX
C8AB- C8                        INY
C8AC- B1 17                     LDA (PLADDR),Y     ;ADD CARRY TO 3RD BYTE
C8AE- 69 00                     ADC #0
C8B0- 91 17                     STA (PLADDR),Y
C8B2- C8                        INY                ;CARRY FROM 3RD BYTE
C8B3- B1 17                     LDA (PLADDR),Y     ;ADD CARRY TO HIGH BYTE
C8B5- 69 00                     ADC #0
C8B7- 91 17                     STA (PLADDR),Y
C8B9- C8                        INY
C8BA- E0 04                     CPX #4             ;2 CNTRS
C8BC- D0 DD                     BNE PLSAVCNT+4
C8BE- 60                        RTS

              ;MRKSEC -- CALCUATE ELASPED TIME IN SECONDS FOR
              ;THE PULSE CNTRS.  60 SECONDS IS THE MAX ELASPED
              ;TIME. SAVE THE CURRENT TIME FOR NEXT TIME.

              MRKSEC
C8BF- F8                        SED
C8C0- A5 27                     LDA *DAY+5
C8C2- AA                        TAX                ;COPY TO X FOR LATER
C8C3- 38                        SEC
C8C4- E5 2D                     SBC *PLLSTSEC
C8C6- B0 02                     BCS =+3
C8C8- 69 60                     ADC #$60           ;CYCLED OVER 1 MINUITE
C8CA- 86 2D                     STX *PLLSTSEC      ;SAVE CURRENT SEC
C8CC- D8                        CLD
C8CD- 20 70 C4                  JSR BINARY         ;CONVERT TO BINARY
C8D0- 85 2E                     STA *PLINCSEC      ;SAVE INCREMENTAL SEC
C8D2- 60                        RTS

              ;PLTIMESAV -- ADD SECOND COUNT TO THE TIME
              ;COUNTER AT THE FIRST TWO BYTES OF PULSE DATA
              ;STOREAGE.

              PLTIMESAV
C8D3- A0 00                     LDY #0
C8D5- 18                        CLC
C8D6- A5 2E                     LDA *PLINCSEC
C8D8- 71 17                     ADC (PLADDR),Y
C8DA- 91 17                     STA (PLADDR),Y
```

```
C8DC- C8        0172          INY
C8DD- B1 17     0173          LDA (PLADDR),Y       ;ADD CARRY TO HIGH BYTE
C8DF- 69 00     0174          ADC #0
C8E1- 91 17     0175          STA (PLADDR),Y
C8E3- 60        0176          RTS
                0177
                0178 ;PLDISP -- CALLED FROM SHOW TO DISPLAY PULSE
                0179 ;DATA ON THE DISPLAY. P0 & P1 ARE SECOND COUNTS,
                0180 ;P2 & P3 ARE MINUITE COUNTS RESPECT.
                0181 ;WRITTEN FOR 0 OR 2 CHANNELS ONLY
                0182
                0183 PLDISP
C8E4- A5 0D     0184          LDA *PLCHAN
C8E6- D0 02     0185          BNE =+3
C8E8- 38        0186          SEC
C8E9- 60        0187          RTS
C8EA- A5 1D     0188          LDA *DISCNT+1
C8EC- C9 04     0189          CMP #4
C8EE- B0 F9     0190          BCS =-6              ;OVER RANGE
C8F0- 20 FA 82  0191          JSR OUTBYT           ;OUTPUT CHAN #
C8F3- A2 73     0192          LDX #$73             ;OUTPUT "P"
C8F5- 8E 44 A6  0193          STX DISBUF+4         ;BEFORE CHANNEL #
C8F8- 0A        0194          ASL A                ;MULT BY 2
C8F9- AA        0195          TAX
C8FA- B5 2F     0196          LDA *PLSEC,X
C8FC- 85 C0     0197          STA *FPLSW           ;LOW BYTE
C8FE- B5 30     0198          LDA *PLSEC+1,X
C900- 85 C1     0199          STA *FPNSW           ;HIGH BYTE OF CNT
C902- A9 00     0200          LDA #0
C904- 85 BF     0201          STA *FPLSWE
C906- 85 C2     0202          STA *FPMSW
C908- A9 17     0203          LDA #23
C90A- 85 C3     0204          STA *FPACCE          ;STORE EXPONENT
C90C- 20 71 CC  0205          JSR FPNORM           ;ADJUST NUMBER TO FP
C90F- 20 8F CF  0206          JSR FPDISP           ;DISPLAY VALUE
C912- 18        0207          CLC
C913- 60        0208          RTS
                0209
                0210 ;PUT "L3PULSE.S"
                0063          .FI "L3DATA.D"

0507  35E4-3AEB  L3DATA.D

                0001 ;PUT "L3DATA.D"
                0002
                0003 ;CONSTANT DEFINITIONS
                0004
                0005 HEADER     .DE 8
                0006 LOGGERID   .DE $C7FF          ;ADDRESS OF ID NUMBER
                0007 MAXSAVE    .DE 12             ;MAXIMUM SAVE FREQUENCY (1
                0008 MINSAVE    .DE 06             ;MINIMUM SAVE FREQUENCY (1
                0009
C914- 01 02 04  0010 LOGSM      .BY 1 2 4 8 $15 $30
C917- 08 15 30
C91A- 01 02 03  0011 LOGHR      .BY 1 2 3 6 $12 0
C91D- 06 12 00
C920- 0C 06 00  0012 LOGTAB1    .BY 12 6 0
C923- 0B 05 05  0013 LOGTAB2    .BY LOGHR-LOGSM+5 5 5
                0014
                0015 LEDMESSAGES
C926- 00 5E 77  0016            .BY $00 $5E $77 $78 $79 $53 ;DATE?
C929- 78 79 53
C92C- 00 78 06  0017            .BY $00 $78 $06 $54 $79 $53 ;TIME?
C92F- 54 79 53
C932- 79 50 50  0018            .BY $79 $50 $50 $5C $50 $00 ;ERROR
C935- 5C 50 00
C938- 00 39 76  0019            .BY $00 $39 $76 $77 $54 $53 ;CHAN?
C93B- 77 54 53
C93E- 00 6D 6E  0020            .BY $00 $6D $6E $86 $06 $80 ;SY1.1.
C941- 86 06 80
                0021
                0022 JMPTAB
```

```
C944- A0 C4      0023              .SE CLKDISP
C946- 3E C7      0024              .SE ADDISP
C948- 96 C7      0025              .SE DGDISP
C94A- E4 C8      0026              .SE PLDISP
C94C- FF FF      0027              .SE $FFFF
C94E- D2 C3      0028              .SE GENDISP
C950- FF FF      0029              .SE $FFFF
                 0030 NUCHAN       .DE =-JMPTAB
                 0031
C952- 00 01 04   0032 OUTGAIN      .BY 0 1 4 5 6 8 9 10 12 13 14
C955- 05 06 08
C958- 09 0A 0C
C95B- 0D 0E
                 0033
                 0034 ;INITIAL DATA FOR ZERO PAGE AERA
                 0035
                 0036 INITDATA
C95D- 5A A5      0037              .BY $5A $A5              ;VERIFY
C95F- 00 02      0038              .SE LOWMEM               ;MEMEND
C961- 07 02      0039              .SE LOWMEM+HEADER-1      ;DATAEND
C963- 00 02      0040              .SE LOWMEM               ;DATASTART
C965- 20 08      0041              .BY $20 $08              ;DISVEL
                 0042 ICOUNT       .DE =-INITDATA-1
                 0043
                 0044 TIMES
C967- 00 00      0045              .BY 0 0                  ; 1     SEC : 1     SEC
C969- 00 02      0046              .BY 0 2                  ; 1     SEC : 3.75  SEC
C96B- 00 04      0047              .BY 0 4                  ; 1     SEC :15     SEC
C96D- 00 06      0048              .BY 0 6                  ; 1     SEC : 1     MIN
C96F- 00 08      0049              .BY 0 8                  ; 1     SEC : 3.75  MIN
C971- 00 0A      0050              .BY 0 10                 ; 1     SEC :15     MIN
C973- 00 0C      0051              .BY 0 12                 ; 1     SEC : 1     HR
C975- 02 0E      0052              .BY 2 14                 ; 3.75  SEC : 3     HR
                 0053
                 0054 ;PUT "L3DATA.D"
                 0064              .EN
END OF MAE PASS!

 ---  LABEL FILE:  ---

@ADOK =C678              @CALCDATAEND1 =C2A1      @CALCDATAEND2 =C2AE
@CALCMXREC =C279         @DATEDISP =C4BB          @G1 =C3E1
@G10 =C461               @G2 =C3EA                @G3 =C3F3
@G4 =C3FC                @G5 =C40A                @G6 =C427
@G7 =C432                @G8 =C43D                @G9 =C455
@GENBCD =C402            @GENOUT =C405            @GETSET =C5D5
@INITA =C1A0             @INPNIB =C6A4            @K1 =C374
@K2 =C37B                @K3 =C380                @K4 =C389
@K5 =C399                @K6 =C3A6                @KEY2 =C369
@LOGAA =C049             @LOGBB =C053             @LOGCC =C057
@LOOP1 =C31F             @LOOP2 =C323             @MOVEDATA =C0EE
@NOA/D =C638             @READ3 =C54D             @SERROR =C3AF
@SETBYTE =C56C           @SHOWLOOP =C306          @SKIPGAIN =C6A3
@STOPCLOCK =C58D         @SUMADC2 =C6F1           @TIMEDISP =C4CE
ACCESS =8B86             ADADDR =0011             ADCHAN =000B
ADDISP =C73E             ADERR =003C              AVERAGE =C0AC
AVGANALOG =0000          AVGTIME =000F            BINARY =C470
BINDEC =C480             BLKMOV =8740             CALCADDR =C236
CALCDATAEND =C27E        CALCLABEL =C145          CALCLENGTH =C215
CALCMXREC =C263          CALCSAVE =C2AF           CHANN =003D
CLKDISP =C4A0            CLRDATA =C101            CLRMEM =CC00
CNTR =00BC               CONFIG =89A5             CURVAL =0021
CYCLE =001B              DATAEND =0004            DATASTART =0006
DAY =0022                DECMEM =CC1E             DECODESW =C1E8
DGADDR =0013             DGBYTE =003B             DGCHAN =000C
DGDISP =C796             DGREAD =C767             DGSAVE =C779
DISBUF =A640             DISCNT =001C             DISVEL =0008
ENDINT =C07D             EVERYSEC =C08B           FILL3 =8718
FMPNT =00B8              FOLSWE =00C7             FOPEXP =00CB
FOPLSW =00C8             FOPMSW =00CA             FOPNSW =00C9
FPACCE =00C3             FPADD =CCB1              FPBASE =CC00
FPDISP =CF8F             FPDIV =CDC9              FPLSW =00C0
```

```
FPLSWE =00BF            FPMSW =00C2             FPMULT =CD26
FPNORM =CC71            FPNSW =00C1             FPOUT =CEB8
FPSUB =CD1C             GAIN =0040              GENDISP =C3D2
GOMON =C3BB             GOPULSE =C838           GOVEC =A659
HEADER =0008            HEXAD =003E             ICOUNT =0009
ID =A64E                INBYTE =81D9            INCHR =8A1B
INCMEM =CC15            INEXPS =00D5            INIT =C185
INITDATA =C95D          INMTAS =00D4            INPA/D =C628
INPA/DV =C6C8           INPRDI =00D6            INTERRUPT =C019
INTVEC =A678            IOEXP =00DA             IOEXPD =00DF
IOLSW =00D7             IOMSW =00D9             IONSW =00D8
IOSTR =00DB             IOSTR1 =00DC            IOSTR2 =00DD
IOSTR3 =00DE            JMPTAB =C944            JMPVEC =0015
KEY =C335               KEYCNT =001E            KSCONF =89A3
L2 =84D3                LABEL =0028             LCNT1 =001F
LCNT2 =0020             LEDMESG =C463           LEDMESSAGES =C926
LENGTH =0029            LOGGERID =C7FF          LOGHR =C91A
LOGSM =C914             LOGTAB1 =C920           LOGTAB2 =C923
LOOP =C315              LOWMEM =0200            MAIN =C000
MAXSAVE =000C           MCAND0 =00C4            MCAND1 =00C5
MCAND2 =00C6            MEMEND =0002            MINSAVE =0006
MONITOR =8003           MOVE =C0B7              MOVECLOCK =C166
MOVEDATA =C0BE          MOVEHEADER =C173        MOVIND =CC0A
MRKSEC =C8BF            NACCESS =8B9C           NUCHAN =000E
NUMREC =002B            OUTBUF =00E5            OUTBYT =82FA
OUTCHR =8A47            OUTGAIN =C952           OUTNIB =8A44
OUTXAH =82F4            PARM =8220              PARN =A64A
PARNR =A649             PLADDR =0017            PLCHAN =000D
PLCHK =C816             PLDISP =C8E4            PLINCSEC =002E
PLLSTSEC =002D          PLMIN =0033             PLSAVCNT =C897
PLSAVE =C800            PLSAVEDIS =C870         PLSEC =002F
PLSUM =0037             PLTIMESAV =C8D3         PLTMP =002C
READCLOCK =C529         READPL =C856            RESALL =81C4
RESTART =8000           RMDIG =A645             SAMPLE =C095
SAVE =C0B0              SAVE2 =87EA             SAVER =8188
SAVTIME =0010           SCAND =8906             SDBYT =A651
SETDATE =C509           SETIME =C4EC            SETUPA/D =C610
SETUPCLOCK =C5B0        SHOW =C2CF              SHOWDISP =C312
SIGNS =00BE             SMPTIME =000E           SPACE =8342
STORE =C127             STRTCONV =C5E7          SUMADC =C6DF
SWITCH =000A            TAPDEL =A630            TEMP1 =00E4
TIMES =C967             TOPNT =00BA             TPEXP =00E3
TPLSW =00E0             TPMSW =00E2             TPNSW =00E1
TSIGN =00BD             VECSW =8BB7             VERIFY =0000
VIA1 =A000              VIA2 =A800              VIA3 =AC00
WORK0 =00CC             WORK1 =00CD             WORK2 =00CE
WORK3 =00CF             WORK4 =00D0             WORK5 =00D1
WORK6 =00D2             WORK7 =00D3             WORKPNT =0019
ZEROMEM =C1B8
//0000,C977,1177
```

```
                 0001 ;PUT "L3FP.C"
                 0002              .LS
                 0003
                 0004 ;SYM-1 DATA LOGGER 3.0 FP ROUTINES
                 0005 ;APRIL 1981
                 0006 ;BY MICHAEL D. SCHWARZ
                 0007
                 0008 ;AG ENGG EXTENSION
                 0009 ;KANSAS STATE UNIVERSITY
                 0010
                 0011              .CE
                 0012              .CT
                 0013              .OS
                 0014
                 0015              .PR "ASSEMBLY OF ONLY FP ROUTINES"
                 0016              .PR "PUT FPDISK IN DRIVE 2"
                 0017 BA           .PR "ENTER BEGINNING ADDRESS"
                 0018              .IN BA
                 0019 MC           0020          .IN MC
                 0021
                 0022              .FI "SYM.L"

0289  3250-34D9  SYM.L

                 0001 ;PUT "SYM.L"
                 0002
                 0003 ;MONITOR ADDRESSES
                 0004
                 0005 ACCESS      .DE $8B86
                 0006 BLKMOV      .DE $8740
                 0007 CONFIG      .DE $89A5
                 0008 FILL3       .DE $8718
                 0009 INBYTE      .DE $81D9
                 0010 INCHR       .DE $8A1B
                 0011 KSCONF      .DE $89A3
                 0012 L2          .DE $84D3
                 0013 MONITOR     .DE $8003
                 0014 NACCESS     .DE $8B9C
                 0015 OUTBYT      .DE $82FA
                 0016 OUTCHR      .DE $8A47
                 0017 OUTNIB      .DE $8A44
                 0018 OUTXAH      .DE $82F4
                 0019 PARM        .DE $8220
                 0020 RESALL      .DE $81C4
                 0021 RESTART     .DE $8000
                 0022 SAVE2       .DE $87EA
                 0023 SAVER       .DE $8188
                 0024 SCAND       .DE $8906
                 0025 SPACE       .DE $8342
                 0026 VECSW       .DE $8BB7
                 0027
                 0028 DISBUF      .DE $A640
                 0029 GOVEC       .DE $A659
                 0030 ID          .DE $A64E
                 0031 INTVEC      .DE $A678
                 0032 PARN        .DE $A64A
                 0033 PARNR       .DE $A649
                 0034 RMDIG       .DE $A645
                 0035 SDBYT       .DE $A651
                 0036 TAPDEL      .DE $A630
                 0037
                 0038 VIA1        .DE $A000
                 0039 VIA2        .DE $A800
                 0040 VIA3        .DE $AC00
                 0041
                 0042 ;PUT "SYM.L"
                 0023             .BA $B8
                 0024             .FI D26 "FPVARS.B"

02BD  3250-350D  FPVARS.B

                 0001 ;PUT "FPVARS.B"
```

```
                     0002
                     0003 ;VARIABLES FOR MATH ROUTINES
                     0004
00B8-                0005 FMPNT         .DS 2
00BA-                0006 TOPNT         .DS 2
00BC-                0007 CNTR          .DS 1
00BD-                0008 TSIGN         .DS 1
00BE-                0009 SIGNS         .DS 1
00BF-                0010 FPLSWE        .DS 1
00C0-                0011 FPLSW         .DS 1
00C1-                0012 FPNSW         .DS 1
00C2-                0013 FPMSW         .DS 1
00C3-                0014 FPACCE        .DS 1
00C4-                0015 MCANDO        .DS 1
00C5-                0016 MCAND1        .DS 1
00C6-                0017 MCAND2        .DS 1
00C7-                0018 FOLSWE        .DS 1
00C8-                0019 FOPLSW        .DS 1
00C9-                0020 FOPNSW        .DS 1
00CA-                0021 FOPMSW        .DS 1
00CB-                0022 FOPEXP        .DS 1
00CC-                0023 WORKO         .DS 1
00CD-                0024 WORK1         .DS 1
00CE-                0025 WORK2         .DS 1
00CF-                0026 WORK3         .DS 1
00D0-                0027 WORK4         .DS 1
00D1-                0028 WORK5         .DS 1
00D2-                0029 WORK6         .DS 1
00D3-                0030 WORK7         .DS 1
                     0031 ;      VARIABLES FOR I/O ROUTINES
00D4-                0032 INMTAS        .DS 1
00D5-                0033 INEXPS        .DS 1
00D6-                0034 INPRDI        .DS 1
00D7-                0035 IOLSW         .DS 1
00D8-                0036 IONSW         .DS 1
00D9-                0037 IOMSW         .DS 1
00DA-                0038 IOEXP         .DS 1
00DB-                0039 IOSTR         .DS 1
00DC-                0040 IOSTR1        .DS 1
00DD-                0041 IOSTR2        .DS 1
00DE-                0042 IOSTR3        .DS 1
00DF-                0043 IOEXPD        .DS 1
00E0-                0044 TPLSW         .DS 1
00E1-                0045 TPNSW         .DS 1
00E2-                0046 TPMSW         .DS 1
00E3-                0047 TPEXP         .DS 1
00E4-                0048 TEMP1         .DS 1
00E5-                0049 OUTBUF        .DS 16
                     0050
                     0051 ;PUT "FPVARS.B"
                     0025
                     0026               .BA BA
                     0027               .MC MC
                     0028               .FI D26 "GPSUBS.S"

0883  3250-3AD3  GPSUBS.S

                     0001 ;PUT "GPSUBS.S"
                     0002
                     0003 ;CLEAR MEMORY. A,X,Y. CLEARS X BYTES
                     0004 ;OF MEMORY TO ZERO AT #TOPNT.
                     0005
                     0006 CLRMEM
CC00- A9 00          0007               LDA #$00
CC02- A8             0008               TAY
CC03- 91 BA          0009               STA (TOPNT),Y          ;CLEAR MEM LOC
CC05- C8             0010               INY
CC06- CA             0011               DEX
CC07- D0 FA          0012               BNE CLRMEM+3           ;NOT ZERO, CONT CLR
CC09- 60             0013               RTS
                     0014
                     0015 ;MOVE MEMORY. A,X,Y. MOVE X BYTES
```

```
                ;FROM *FMPNT TO *TOPNT. LIMIT=256.

                MOVIND
CC0A- A0 00                LDY #$00
CC0C- B1 B8                LDA (FMPNT),Y
CC0E- 91 BA                STA (TOPNT),Y
CC10- C8                   INY
CC11- CA                   DEX
CC12- D0 F8                BNE MOVIND+2
CC14- 60                   RTS

                ;INCREMENT MEMORY. X,Y. INCREMENT A
                ;MULTI PREC NUMBER Y BYTES LONG AT X
                ;Z FLAG SET IF RESULT = 0

                INCMEM
CC15- F6 00                INC *$00,X
CC17- D0 04                BNE INCMEM+8
CC19- E8                   INX
CC1A- 88                   DEY
CC1B- D0 F8                BNE INCMEM
CC1D- 60                   RTS

                ;DECREMENT MEMORY. A,X,Y. DEC MULTI
                ;PREC NUMBER OF Y BYTES AT X.
                ;CARRY CLR IF RESULT=-1

                DECMEM
CC1E- 38                   SEC
CC1F- B5 00                LDA *$00,X
CC21- E9 01                SBC #$01
CC23- 95 00                STA *$00,X
CC25- B0 04                BCS DECMEM+13
CC27- E8                   INX
CC28- 88                   DEY
CC29- D0 F3                BNE DECMEM
CC2B- 60                   RTS

                ;ROTATE LEFT. X,Y. ROTATE MULTI PREC
                ;NUMBER LEFT 1 BIT. X IS LOC OF LSB
                ; AND Y IS # OF BYTES.

                ROTATL
CC2C- 18                   CLC                     ;ROTATE IN ZERO BIT
                ROTL
CC2D- 36 00                ROL *$00,X
CC2F- E8                   INX
CC30- 88                   DEY
CC31- D0 FA                BNE ROTL
CC33- 60                   RTS

                ;ROTATE OR ARITH SHIFT RIGHT. X,Y.
                ;ROTATE BYTE RIGHT 1 BIT. Y IS # OF
                ;BYTES, X IS PNTR TO MSBYTE

                ROTATR
CC34- 18                   CLC
CC35- 90 06                BCC ARSHR+6             ;SHIFT IN ZERO BIT
                ARSHR
CC37- 36 00                ROL *$00,X              ;SHIFT IN SIGN BIT
CC39- 08                   PHP
CC3A- 76 00                ROR *$00,X
CC3C- 28                   PLP                     ;SIGN BIT IN CARRY
CC3D- 76 00                ROR *$00,X              ;ROTATE THE BYTE
CC3F- CA                   DEX
CC40- 88                   DEY
CC41- D0 FA                BNE ARSHR+6
CC43- 60                   RTS

                ;COMPLEMENT MEMORY (1 OR 2). A,X,Y.
                ;COMPLEMENT MULTI PREC BYTE OF Y
                ;BYTES, AT LOC X.
```

```

             COMPL1
CC44- 18                CLC                 ;CLEAR CARRY FOR 1'S COMP
CC45- 90 01             BCC COMPLM+1
             COMPLM
CC47- 38                SEC                 ;SET CARRY FOR 2'S COMP
CC48- A9 FF             LDA #$FF
CC4A- 55 00             EOR *$00,X
CC4C- 69 00             ADC #$00
CC4E- 95 00             STA *$00,X
CC50- E8                INX
CC51- 88                DEY
CC52- D0 F4             BNE COMPLM+1
CC54- 60                RTS

             ;ADDER. A,X,Y. ADD MULTI PREC NUMS
             ;IN *TOPNT AND *FMPNT. STORE RESULT
             ;IN *TOPNT.

             ADDER
CC55- A0 00             LDY #$00
CC57- 18                CLC
CC58- B1 BA             LDA (TOPNT),Y
CC5A- 71 B8             ADC (FMPNT),Y
CC5C- 91 BA             STA (TOPNT),Y
CC5E- C8                INY
CC5F- CA                DEX
CC60- D0 F6             BNE ADDER+3
CC62- 60                RTS

             ;SUBBER. A,X,Y. SUBTRACTS NUM IN
             ;*FMPNT FROM *TOPNT AND LEAVES RESULT
             ;IN *TOPNT.

             SUBBER
CC63- A0 00             LDY #$00
CC65- 38                SEC
CC66- B1 BA             LDA (TOPNT),Y
CC68- F1 B8             SBC (FMPNT),Y
CC6A- 91 BA             STA (TOPNT),Y
CC6C- C8                INY
CC6D- CA                DEX
CC6E- D0 F6             BNE SUBBER+3
CC70- 60                RTS

             ;COMPARE MEMORY. A,Y. Y IS LENGTH
             ;OF NUMS, *FMPNT & *TOPNT ARE STRING
             ;PNTRS.  CMPS LAST BYTE FIRST ASSUME
             ;NUMBERS.

             ; TYA
             ; BEQ CMPMEM+7 ;END OF DATA
             ;CMPMEM
             ; DEY
             ; LDA (FMPNT),Y
             ; CMP (TOPNT),Y
             ; BEQ CMPMEM-3 ;IF EQUAL CONTINUE
             ; RTS ;C & Z FLGS SET

             ;PUT "GPSUBS.S"
                        .FI D26 "FPNORM.S"

032F  3250-357F  FPNORM.S

             ;PUT "FPNORM.S"

             ;FLOATING POINT NORMALIZATION. A,X,Y.
             ;NORMALIZES FPNUM IN FPACC WITH 4
             ;BYTES IN THE MANTISSA.

             FPNORM
CC71- A5 C2             LDA *FPMSW
```

```
CC73- 85 BD     0009            STA *TSIGN         ;SAVE SIGN BIT
CC75- 10 07     0010            BPL @ACZERT
CC77- A0 04     0011            LDY #$04           ;COMPLMENT IF NEGATIVE
CC79- A2 BF     0012            LDX #FPLSWE
CC7B- 20 47 CC  0013            JSR COMPLM
                0014 @ACZERT
CC7E- A5 C2     0015            LDA *FPMSW         ;CHECK FOR FPACC == 0
CC80- 05 C1     0016            ORA *FPNSW
CC82- 05 C0     0017            ORA *FPLSW
CC84- 05 BF     0018            ORA *FPLSWE
CC86- D0 03     0019            BNE @ACNONZ
CC88- 85 C3     0020            STA *FPACCE        ;FPACC=0 CLR EXP
                0021 @NORMEX
CC8A- 60        0022            RTS                ;EXIT RESULT = 0
                0023 @ACNONZ
CC8B- 24 C2     0024            BIT *FPMSW
CC8D- 30 0E     0025            BMI @MINUS1
CC8F- 70 15     0026            BVS @ACCSET        ;IF SO, JUSTIFIED
CC91- A2 BF     0027            LDX #FPLSWE        ;SHIFT LEFT 1 BIT
CC93- A0 04     0028            LDY #$04
CC95- 20 2C CC  0029            JSR ROTATL
CC98- C6 C3     0030            DEC *FPACCE        ;ADJUST EXP
CC9A- 4C 8B CC  0031            JMP @ACNONZ        ;CONTINUE ROTATING
                0032 @MINUS1
CC9D- A2 C2     0033            LDX #FPMSW         ;THE NUMBER IS -1.0000000
CC9F- A0 03     0034            LDY #3
CCA1- 20 34 CC  0035            JSR ROTATR
CCA4- E6 C3     0036            INC *FPACCE
                0037 @ACCSET
CCA6- A5 BD     0038            LDA *TSIGN         ;IS SIGN POS
CCA8- 10 E0     0039            BPL @NORMEX        ;YES, RETURN
CCAA- A0 03     0040            LDY #$03           ;NO, COMPLEMENT AGAIN
CCAC- A2 C0     0041            LDX #FPLSW
CCAE- 4C 47 CC  0042            JMP COMPLM
                0043
                0044 ;PUT "FPNORM.S"
                0030            .FI D26 "FPADDSUB.S"

060A  3250-385A  FPADDSUB.S

                0001 ;PUT "FPADDSUB.S"
                0002 ;FLOATING POINT ADD. A,X,Y. ADDS NUM
                0003 ;IN FPACC TO FPOP AND LEAVES RESULT
                0004 ;IN FPACC.
                0005
                0006 FPADD
CCB1- A5 C2     0007            LDA *FPMSW         ;FPACC = 0?
CCB3- D0 0A     0008            BNE @NONZAC
                0009 @MOVOP
CCB5- A2 03     0010            LDX #3             ;MOVE FOP TO FPACC & RET
CCB7- B5 C8     0011            LDA *FOPLSW,X
CCB9- 95 C0     0012            STA *FPLSW,X
CCBB- CA        0013            DEX
CCBC- 10 F9     0014            BPL =-6
CCBE- 60        0015            RTS
                0016 @NONZAC
CCBF- A5 CA     0017            LDA *FOPMSW        ;SEE IF FPOP = 0
CCC1- D0 01     0018            BNE @CKEQEX
CCC3- 60        0019            RTS
                0020 @CKEQEX
CCC4- 38        0021            SEC
CCC5- A5 CB     0022            LDA *FOPEXP
CCC7- E5 C3     0023            SBC *FPACCE        ;GET EXP DIFF
CCC9- 50 03     0024            BVC @NOVFL         ;DIFF > 128
CCCB- 90 E8     0025            BCC @MOVOP         ;RESULT = FPOP
                0026 @CKRET
CCCD- 60        0027            RTS                ;RESULT = FPACC
                0028 @NOVFL
CCCE- F0 1D     0029            BEQ @SHACOP
CCD0- A8        0030            TAY                ;PUT DIFF IN Y
CCD1- 30 0E     0031            BMI @SKPNEG        ;FPACC >, SHIFT FPOP
CCD3- C9 18     0032            CMP #$18           ;DIFF > $18
```

```
CCD5- 10 DE     0033            BPL @MOVOP          ;RESULT IS FPOP
                0034 @MORACC
CCD7- A2 C3     0035            LDX #FPACCE         ;SET PNTR TO FPACC EXP
CCD9- 20 0F CD  0036            JSR SHLOOP          ;SHIFT FPACC >>
CCDC- 88        0037            DEY                 ;DEC DIFF CNTR
CCDD- D0 F8     0038            BNE @MORACC
CCDF- F0 0C     0039            BEQ @SHACOP
                0040 @SKPNEG
CCE1- C9 E8     0041            CMP #$00-$18        ;DIFF > $18
CCE3- 30 E8     0042            BMI @CKRET          ;RESULT = FPACC
                0043 @SHIFTO
CCE5- A2 CB     0044            LDX #FOPEXP         ;SET PNTR TO FPOP EXP
CCE7- 20 0F CD  0045            JSR SHLOOP          ;SHIFT FPOP RIGHT
CCEA- C8        0046            INY                 ;INC DIFF CNTR
CCEB- D0 F8     0047            BNE @SHIFTO
                0048 @SHACOP
CCED- A9 00     0049            LDA #$00            ;PREPARE ADDITION
CCEF- 85 BF     0050            STA *FPLSWE
CCF1- 85 C7     0051            STA *FOLSWE
CCF3- A2 C3     0052            LDX #FPACCE         ;COMPENSATE FOR OVFL
CCF5- 20 0F CD  0053            JSR SHLOOP
CCF8- A2 CB     0054            LDX #FOPEXP
CCFA- 20 0F CD  0055            JSR SHLOOP
CCFD- 18        0056            CLC                 ;ADD BYTES
CCFE- A0 04     0057            LDY #4              ;4 BYTES
CD00- A2 00     0058            LDX #0
CD02- B5 C7     0059            LDA *FOLSWE,X
CD04- 75 BF     0060            ADC *FPLSWE,X
CD06- 95 BF     0061            STA *FPLSWE,X
CD08- E8        0062            INX
CD09- 88        0063            DEY
CD0A- D0 F6     0064            BNE =-9
CD0C- 4C 71 CC  0065            JMP FPNORM          ;NORMALIZE RESULT
                0066
                0067 SHLOOP
CD0F- F6 00     0068            INC *$00,X          ;INC EXP
CD11- CA        0069            DEX                 ;DEC PNTR TO MSB
CD12- 98        0070            TYA                 ;SAVE DIFF
CD13- 48        0071            PHA
CD14- A0 04     0072            LDY #$04
CD16- 20 37 CC  0073            JSR ARSHR           ;ARITH SHIFT RIGHT
CD19- 68        0074            PLA
CD1A- A8        0075            TAY                 ;RESTORE IN Y
CD1B- 60        0076            RTS
                0077
                0078 ;FLOATING SUBTRACT. A,X,Y. SUBTRACTS
                0079 ;FPOP FROM FPACC BY TAKING THE NEGATIVE
                0080 ;OF FPOP AND ADDING.
                0081
                0082 FPSUB
CD1C- A2 C0     0083            LDX #FPLSW          ;SET PNTR TO FPOP
CD1E- A0 03     0084            LDY #$03
CD20- 20 47 CC  0085            JSR COMPLM          ;NEGATE
CD23- 4C B1 CC  0086            JMP FPADD           ;ADD THE VALUES
                0087
                0088 ;PUT "FPADDSUB.S"
                0031            .FI D26 "FPMUL.S"

0AFE  3250-3D4E  FPMUL.S

                0001 ;PUT "FPMUL.S"
                0002 ;FLOATING POINT MULTIPLY. A,X,Y.
                0003 ;MULTIPLY NUMBER IN FPACC TO FPOP
                0004 ;AND LEAVES THE RESULT IN FPACC
                0005 ;
                0006 FPMULT
CD26- 20 92 CD  0007            JSR CKSIGN          ;SETUP & CHK SIGN
CD29- A5 CB     0008            LDA *FOPEXP         ;GET FPOP EXP
CD2B- 38        0009            SEC                 ;ADD 1 FOR COMPEN
CD2C- 65 C3     0010            ADC *FPACCE         ;ADD FPOP EXP
CD2E- 85 C3     0011            STA *FPACCE         ;SAVE RESULT EXP
CD30- A9 17     0012            LDA #$17            ;SET BIT COUNTER
```

```
CD32- 85 BC      0013              STA *CNTR       ;STORE BIT CNTR
                 0014 MULTIP
CD34- A2 C2      0015              LDX #FPMSW      ;SET PNTR TO FPACC MS BYTE
CD36- A0 03      0016              LDY #$3         ;SET PRECISION CNTR
CD38- 20 34 CC   0017              JSR ROTATR      ;ROTATE FPACC RT
CD3B- 90 0D      0018              BCC NADOPP      ;CARRY=0,DON'T ADD PARTIAL
                 0019 ADOPP
CD3D- A2 C8      0020              LDX #FOPLSW     ;PNTR TO LS BYTE OF MULTIP
CD3F- 86 B8      0021              STX *FMPNT      ;STORE POINTER
CD41- A2 D0      0022              LDX #WORK4      ;PNTR TO LS BYTE OF PARTIA
CD43- 86 BA      0023              STX *TOPNT
CD45- A2 03      0024              LDX #$3         ;ONLY 3 BYTES NEED ADDED
CD47- 20 55 CC   0025              JSR ADDER       ;ADD MULTIPLICAND TO PARTI
                 0026 NADOPP
CD4A- A2 D2      0027              LDX #WORK6      ;SET PNTR TO MS BYTE OF PA
CD4C- A0 04      0028              LDY #$4         ;SET PREC CNTR
CD4E- 20 34 CC   0029              JSR ROTATR      ;ROTATE PART PROD RT
CD51- C6 BC      0030              DEC *CNTR
CD53- D0 DF      0031              BNE MULTIP      ;NOT 0, CONT MULTIPLYING
CD55- A2 D2      0032              LDX #WORK6      ;ELSE, SET PNTR TO PART PR
CD57- A0 04      0033              LDY #$4         ;SET PREC CNTR
CD59- 20 34 CC   0034              JSR ROTATR      ;MAKE ROOM FOR ROUNDING
CD5C- A2 CF      0035              LDX #WORK3      ;SET PNTR TO 24TH BIT OF P
CD5E- A0 04      0036              LDY #$4         ;SET PREC CNTR
CD60- 18         0037              CLC             ;CLEAR FOR ADDITION
CD61- A9 80      0038              LDA #$80        ;24TH BIT TO RND IF >.5
CD63- 24 D2      0039              BIT *WORK6      ;IS MANTISA >.5
CD65- 70 01      0040              BVS CKRND       ;IF SO THEN ROUND
CD67- 4A         0041              LSR A           ;24BIT IS RT ONE
                 0042 CKRND
CD68- 24 CF      0043              BIT *WORK3      ;NEED RNDING?
CD6A- F0 0A      0044              BEQ PREXFR      ;IF ZERO SKIP RND
                 0045 CROUND
CD6C- 75 00      0046              ADC *$00,X      ;ADD WITH CARRY TO PROPAGA
CD6E- 95 00      0047              STA *$00,X      ;STORE PART PROD
CD70- A9 00      0048              LDA #$0         ;CLR A FOR NEXT ADD
CD72- E8         0049              INX             ;INCREMENT INDEX PNTR
CD73- 88         0050              DEY             ;DECREMENT CNTR
CD74- D0 F6      0051              BNE CROUND      ;NOT 0, ADD NEXT BYTE
                 0052 PREXFR
CD76- A2 BF      0053              LDX #FPLSWE     ;SET PNTR TO FPACC LSW-1
CD78- 86 BA      0054              STX *TOPNT      ;STORE IN TOPNT
CD7A- A2 CF      0055              LDX #WORK3      ;SET PNTR TO PART PROD
CD7C- 86 B8      0056              STX *FMPNT      ;STORE IN FMPNT
CD7E- A2 04      0057              LDX #$4         ;SET PREC CNTR
                 0058 EXMLDV
CD80- 20 0A CC   0059              JSR MOVIND      ;MOVE PART PROD TO FPACC
CD83- 20 71 CC   0060              JSR FPNORM      ;NORMALIZE RESULT
CD86- A5 BE      0061              LDA *SIGNS      ;GET SIGN STORAGE
CD88- D0 07      0062              BNE MULTEX      ;IF NOT 0, SIGN IS POS
CD8A- A2 C0      0063              LDX #FPLSW      ;ELSE, SET PNTR TO FPACC L
CD8C- A0 03      0064              LDY #$3         ;SET PREC CNTR
CD8E- 20 47 CC   0065              JSR COMPLM      ;COMPLEMENT RESULT
                 0066 MULTEX
CD91- 60         0067              RTS             ;EXIT FPMULT
                 0068 CKSIGN
CD92- A9 00      0069              LDA #$0         ;SET PAGE PORTION OF POINT
CD94- 85 BB      0070              STA *TOPNT+1    ;STORE IN TOPNT
CD96- 85 B9      0071              STA *FMPNT+1    ;STORE IN FMPNT
CD98- A9 CC      0072              LDA #WORK0      ;SET PNTR TO WORK AREA
CD9A- 85 BA      0073              STA *TOPNT      ;STORE IN TOPNT
CD9C- A2 08      0074              LDX #$8         ;SET PREC CNTR
CD9E- 20 00 CC   0075              JSR CLRMEM      ;SET WORK AREA
CDA1- A9 C4      0076              LDA #MCAND0     ;SET PNTR TO MULTIPLICAND
CDA3- 85 BA      0077              STA *TOPNT      ;STORE IN TOPNT
CDA5- A2 04      0078              LDX #$4         ;SET PREC CNTR
CDA7- 20 00 CC   0079              JSR CLRMEM      ;CLEAR MULTIPLICAND STORAG
CDAA- A9 01      0080              LDA #$1         ;INITIALIZE SIGN INDICATOR
CDAC- 85 BE      0081              STA *SIGNS      ;BY STORING 1 IN SIGNS
CDAE- A5 C2      0082              LDA *FPMSW      ;FETCH FPACC MS BYTE
CDB0- 10 09      0083              BPL OPSGNT      ;POSITIVE, CHECK FPOP
                 0084 NEGFPA
```

```
CDB2- C6 BE                 DEC *SIGNS        ;IF NEG, DECREMENT SIGNS
CDB4- A2 C0                 LDX #FPLSW        ;SET PNTR TO FPACC LS BYTE
CDB6- A0 03                 LDY #$3           ;SET PREC CNTR
CDB8- 20 47 CC              JSR COMPLM        ;MAKE POS FOR MULT
                 OPSGNT
CDBB- A5 CA                 LDA *FOPMSW       ;IS FPOP NEG?
CDBD- 30 01                 BMI NEGOP         ;TES,COMPLEMENT VALUE
CDBF- 60                    RTS               ;ELSE RETURN
                 NEGOP
CDC0- C6 BE                 DEC *SIGNS        ;DECREMENT SIGNS INDICATOR
CDC2- A2 C8                 LDX #FOPLSW       ;SET PNTR TO FPOP LS BYTE
CDC4- A0 03                 LDY #$3           ;SET PREC CNTR
CDC6- 4C 47 CC              JMP COMPLM        ;COMPLEMENT FPOP AND RETUR
                 ;PUT "FPMUL.S"
                            .FI D26 "FPDIV.S"
```

```
0919  3250-3B69  FPDIV.S

                 ;PUT "FPDIV.S"
                 ;FLOATING POINT DIVIDE. A,X,Y.
                 ;DIVIDES NUMBER IN FPACC BY NUMBER
                 ;IN FPOP AND LEAVES RESULT IN FPACC
                 FPDIV
CDC9- 20 92 CD              JSR CKSIGN        ;CLEAR WORK AREA
CDCC- A5 C2                 LDA *FPMSW        ;CHCK FOR DIV BY 0
CDCE- F0 22                 BEQ DERROR        ;DIV=0,DIV BY 0 ERROR
                 SUBEXP
CDD0- A5 CB                 LDA *FOPEXP       ;FET DIVIDEND EXPONENT
CDD2- 38                    SEC               ;SET CARRY
CDD3- E5 C3                 SBC *FPACCE       ;SUBTRACT DIVISOR EXP
CDD5- 85 C3                 STA *FPACCE       ;SORE IN FPACC EXP
CDD7- E6 C3                 INC *FPACCE       ;COMPENSATE FOR DIV ALGORI
                 SETDCT
CDD9- A9 17                 LDA #$17          ;SET BIT CNTR STORAGE
CDDB- 85 BC                 STA *CNTR
                 DIVIDE
CDDD- 20 36 CE              JSR SETSUB        ;SUB DIVISOR FROM DIVIDEND
CDE0- 30 11                 BMI NOGO
CDE2- A2 C8                 LDX #FOPLSW       ;SET PNTR TO DIVIDEND
CDE4- 86 BA                 STX *TOPNT        ;STORE IN TOPNT
CDE6- A2 CC                 LDX #WORK0        ;SET PNTR TO QUOTIENT
CDE8- 86 B8                 STX *FMPNT        ;STORE IN FMPNT
CDEA- A2 03                 LDX #$3           ;SET PREC CNTR
CDEC- 20 0A CC              JSR MOVIND        ;MOV QUOT TO DIVIDEND
CDEF- 38                    SEC               ;SET CARRY
CDF0- B0 02                 BCS QUOROT
                 DERROR
CDF2- 60                    RTS               ;DIV BY 0 IS 0
                 NOGO
CDF3- 18                    CLC               ;NEG RESULT, CLEAR CARRY
                 QUOROT
CDF4- A2 D0                 LDX #WORK4        ;SET PNTR TO QUOTIENT LS B
CDF6- A0 03                 LDY #$3           ;SET PREC CNTR
CDF8- 20 2D CC              JSR ROTL          ;ROTATE CARRY IN LSB OF QU
CDFB- A2 C8                 LDX #FOPLSW       ;SET PNTR TO DIVID LS BYTE
CDFD- A0 03                 LDY #$3           ;SET PREC CNTR
CDFF- 20 2C CC              JSR ROTATL        ;ROTATE DIVIDEND L
CE02- C6 BC                 DEC *CNTR         ;DEC BIT CNTR
CE04- D0 D7                 BNE DIVIDE        ;NOT 0,CONT
CE06- 20 36 CE              JSR SETSUB        ;DO AGAIN FOR RNDING
CE09- 30 1E                 BMI DVEXIT        ;-,NO RNDING
CE0B- A9 01                 LDA #$1           ;ELSE ADD 1 TO 23RD BIT
CE0D- 18                    CLC               ;CLEAR CARRY FOR ADD
CE0E- 65 D0                 ADC *WORK4        ;RND OFF LS BYTE OF QUOT
CE10- 85 D0                 STA *WORK4        ;RESTORE BYTE
CE12- A9 00                 LDA #$0           ;CLEAR A NOT CARRY
CE14- 65 D1                 ADC *WORK5        ;ADD CARRY
CE16- 85 D1                 STA *WORK5        ;STOR RESULT
CE18- A9 00                 LDA #$0           ;CLEAR A NOT CARRY
CE1A- 65 D2                 ADC *WORK6        ;ADD CARRY TO MS BYTE OF Q
CE1C- 85 D2                 STA *WORK6        ;STORE RESULT
CE1E- 10 09                 BPL DVEXIT
```

```
CE20- A2 D2      0055            LDX #WORK6
CE22- A0 03      0056            LDY #$3                 ;SET PREC CNTR
CE24- 20 34 CC   0057            JSR ROTATR              ;CLEAR SIGN BIT CNTR
CE27- E6 C3      0058            INC *FPACCE             ;COMPENSATE EXP FOR ROTATE
                 0059 DVEXIT
CE29- A2 BF      0060            LDX #FPLSWE             ;SET POINTER TO FPACC
CE2B- 86 BA      0061            STX *TOPNT              ;STORE IN TOPNT
CE2D- A2 CF      0062            LDX #WORK3              ;SET PNTR TO QUOTIENT
CE2F- 86 B8      0063            STX *FMPNT              ;SOTRE IN FMPNT
CE31- A2 04      0064            LDX #$4                 ;SET PREC CNTR
CE33- 4C 80 CD   0065            JMP EXMLDV              ;MOVE QUOTIENT TO FPACC
                 0066 SETSUB
CE36- A2 CC      0067            LDX #WORK0              ;SET PNTR
CE38- 86 BA      0068            STX *TOPNT
CE3A- A2 C0      0069            LDX #FPLSW              ;SET PNTR TO FPACC
CE3C- 86 B8      0070            STX *FMPNT
CE3E- A2 03      0071            LDX #$3                 ;SET PREC CNTR
CE40- 20 0A CC   0072            JSR MOVIND              ;MOVE FPACC TO WORK AREA
CE43- A2 CC      0073            LDX #WORK0              ;PREP FOR SUBT
CE45- 86 BA      0074            STX *TOPNT              ;STORE PNTR TO DIVISOR
CE47- A2 C8      0075            LDX #FOPLSW             ;SET PNTR TO FPOP LS BYTE-
CE49- 86 B8      0076            STX *FMPNT              ;STORE PNTR TO DIVIDEND
CE4B- A0 00      0077            LDY #$0                 ;INITIALIZE INDEX PNTR
CE4D- A2 03      0078            LDX #3                  ;SET PREC CNTR
CE4F- 38         0079            SEC
                 0080 SUBR1
CE50- B1 B8      0081            LDA (FMPNT),Y           ;FETCH FPOP
CE52- F1 BA      0082            SBC (TOPNT),Y           ;SUB FPACC BYTE
CE54- 91 BA      0083            STA (TOPNT),Y           ;STORE IN PLACE OF DIVISOR
CE56- C8         0084            INY
CE57- CA         0085            DEX
CE58- D0 F6      0086            BNE SUBR1               ;NOT 0 CONT SUBT
CE5A- A5 CE      0087            LDA *WORK2              ;SET SIGN BIT RESULT IN N
CE5C- 60         0088            RTS
                 0089 ;
                 0090 ;PUT "FPDIV.S"
                 0033            .FI D26 "FPIOSUBS.S"

04E1  3250-3731  FPIOSUBS.S

                 0001 ;PUT "FPIOSUBS.S"
                 0002 ;SUBROUTINES USED BY THE FPIN AND FPOUT ROUTINES
                 0003 ;
                 0004 FPX10
CE5D- A9 04      0005            LDA #$04
CE5F- 85 CB      0006            STA *FOPEXP             ;LOAD FPOP WITH TEN
CE61- A9 50      0007            LDA #$50                ;BY SETTING EXP TO 4
CE63- 85 CA      0008            STA *FOPMSW             ;AND THE MANTISSA TO $50,$
CE65- A9 00      0009            LDA #$00
CE67- 85 C9      0010            STA *FOPNSW
CE69- 85 C8      0011            STA *FOPLSW
CE6B- 20 26 CD   0012            JSR FPMULT              ;MULT FPACC BY FPOP
CE6E- C6 DF      0013            DEC *IOEXPD             ;DECR DEC EXP
CE70- 60         0014            RTS                     ;RETURN TO TEST FOR COMPLE
                 0015 FPD10
CE71- A9 FD      0016            LDA #$FD                ;PLACE .1 IN FPOP BY
CE73- 85 CB      0017            STA *FOPEXP             ;SETTING FPOP EXP TO -3
CE75- A9 66      0018            LDA #$66                ;AND LOADING NANTISSA WITH
CE77- 85 CA      0019            STA *FOPMSW
CE79- 85 C9      0020            STA *FOPNSW
CE7B- A9 67      0021            LDA #$67
CE7D- 85 C8      0022            STA *FOPLSW
CE7F- 20 26 CD   0023            JSR FPMULT              ;MULT FPACC BY FPOP
CE82- E6 DF      0024            INC *IOEXPD
CE84- 60         0025            RTS
                 0026 DECBIN
CE85- A9 00      0027            LDA #$00
CE87- 85 DE      0028            STA *IOSTR3             ;CLEAR MS BYTE+1 OF RESULT
CE89- A2 D7      0029            LDX #IOLSW              ;SET PNTR TO I/O STOR
CE8B- 86 BA      0030            STX *TOPNT
CE8D- A2 DB      0031            LDX #IOSTR              ;SET PNTR TO I/O STOR
CE8F- 86 B8      0032            STX *FMPNT
```

```
CE91- A2 04     0033            LDX #$04          ;SET PREC CNTR
CE93- 20 0A CC  0034            JSR MOVIND        ;MOVE I/O STOR TO WORK ARE
CE96- A2 DB     0035            LDX #IOSTR        ;SET PNTR TO ORIG VAL
CE98- A0 04     0036            LDY #$04          ;SET PREC CNTR
CE9A- 20 2C CC  0037            JSR ROTATL        ;START *10 ROUT
CE9D- A2 DB     0038            LDX #IOSTR        ;RESET PNTR
CE9F- A0 04     0039            LDY #$04          ;SET PREC CNTR
CEA1- 20 2C CC  0040            JSR ROTATL        ;MULT BY 2 AGAIN
CEA4- A2 D7     0041            LDX #IOLSW        ;SET PNTR TO I/O WORK AREA
CEA6- 86 B8     0042            STX *FMPNT        ;STORE IN FMPNT
CEA8- A2 DB     0043            LDX #IOSTR        ;SET PNTR TO I/O STOR
CEAA- 86 BA     0044            STX *TOPNT        ;STORE IN TOPNT
CEAC- A2 04     0045            LDX #$04          ;SET PREC CNTR
CEAE- 20 55 CC  0046            JSR ADDER         ;ADD ORIGINAL TO ROTATED
CEB1- A2 DB     0047            LDX #IOSTR        ;RESET PNTR
CEB3- A0 04     0048            LDY #$04          ;SET PREC CNTR
CEB5- 4C 2C CC  0049            JMP ROTATL
                0050 ;
                0051 ;PUT "FPIOSUBS.S"
                0034            .FI D26 "FPBUFOUT.S"

0CB1  3250-3F01  FPBUFOUT.S

                0001 ;PUT "FPBUFOUT.S"
                0002 ;FLOATING POINT OUTPUT ROUTINE
                0003 ;CONVERTS THE FLOATING POINT BINARY
                0004 ;NUMBER IN FPACC TO ITS DECIMAL
                0005 ;EQUIVALENT AND OUTPUTS IT TO THE
                0006 ;DISPLAY DEVICE AS ASCII CHARACTERS
                0007 ;IN THE FOLLOWING FORMAT
                0008 ;  0.1234567 E+07
                0009 ;
                0010 FPOUT
CEB8- A9 E5     0011            LDA #OUTBUF       ;CLEAR OUTPUT BUFFER
CEBA- 85 BA     0012            STA *TOPNT
CEBC- A9 00     0013            LDA #$00
CEBE- 85 BB     0014            STA *TOPNT+1
CEC0- 85 B9     0015            STA *FMPNT+1
CEC2- 85 DF     0016            STA *IOEXPD       ;CLEAR DEC EXP STOR
CEC4- A2 16     0017            LDX #$16
CEC6- 20 00 CC  0018            JSR CLRMEM
CEC9- A5 C2     0019            LDA *FPMSW        ;IS NEG
CECB- 30 04     0020            BMI OUTNEG        ;YES,MAKE POS AND OUT "-"
CECD- A9 2B     0021            LDA #'+           ;ELSE, SET ASCII FOR "+"
CECF- D0 09     0022            BNE AHEAD1        ;DISPLAY +
                0023 OUTNEG
CED1- A2 C0     0024            LDX #FPLSW        ;SET PNTR TO LS BYTE OF FP
CED3- A0 03     0025            LDY #$3           ;SET PREC CNTR
CED5- 20 47 CC  0026            JSR COMPLM        ;MAKE FPACC POS
CED8- A9 2D     0027            LDA #'-           ;SET ASCII FOR "-"
                0028 AHEAD1
CEDA- 20 88 CF  0029            JSR BUFECHO       ;OUT SIGN OF RESULT
CEDD- A9 30     0030            LDA #'0           ;SET UP ASCII 0
CEDF- 20 88 CF  0031            JSR BUFECHO       ;OUT 0
CEE2- A9 2E     0032            LDA #'.           ;SET UP ASCII "."
CEE4- 20 88 CF  0033            JSR BUFECHO       ;OUT .
CEE7- C6 C3     0034            DEC *FPACCE
                0035 DECEXT
CEE9- 10 0F     0036            BPL DECEXD        ;IF COMPENSATED,EXP>=0
CEEB- A9 04     0037            LDA #$4           ;EXP NEG,ADD 4 TO FPACC
CEED- 18        0038            CLC
CEEE- 65 C3     0039            ADC *FPACCE       ;ADD 4 TO FPACC EXP
CEF0- 10 0E     0040            BPL DECOUT        ;IF EXP>=0,OUT MANTISSA
CEF2- 20 5D CE  0041            JSR FPX10         ;ELSE, MULT MANTISSA BY 10
                0042 DECREP
CEF5- A5 C3     0043            LDA *FPACCE       ;GET EXP
CEF7- 4C E9 CE  0044            JMP DECEXT        ;REPEAT TEST FOR >=0
                0045 DECEXD
CEFA- 20 71 CE  0046            JSR FPD10         ;MULT FPACC BY .1
CEFD- 4C F5 CE  0047            JMP DECREP        ;CHK STATUS OF FPACC EXP
                0048 DECOUT
CF00- A2 DB     0049            LDX #IOSTR        ;SET UP FOR MOVE OPER
```

```
CF02- 86 BA     0050              STX *TOPNT       ;SET TOPNT TO WORK REG
CF04- A2 C0     0051              LDX #FPLSW       ;SET PNTR TO FPACC LS BYTE
CF06- 86 B8     0052              STX *FMPNT
CF08- A2 03     0053              LDX #$3          ;SET PREC CNTR
CF0A- 20 0A CC  0054              JSR MOVIND       ; MOVE FPACC TO OUT REGS
CF0D- A9 00     0055              LDA #$0
CF0F- 85 DE     0056              STA *IOSTR3      ;CLEAR OUT REG MS BYTE+1
CF11- A2 DB     0057              LDX #IOSTR       ;SET PNTR TO OUT LS BYTE
CF13- A0 03     0058              LDY #$3          ;SET PREC CNTR
CF15- 20 2C CC  0059              JSR ROTATL       ;ROT TO COMP FOR SIGN BIT
CF18- 20 85 CE  0060              JSR DECBIN       ;OUT REG X10,OVERFLOW
                0061 COMPEN
CF1B- E6 C3     0062              INC *FPACCE      ;INCR FPACC EXP
CF1D- F0 0A     0063              BEQ OUTDIG       ;OUT DIGIT
CF1F- A2 DE     0064              LDX #IOSTR3      ;ELSE,ROTATE R TO COMP
CF21- A0 04     0065              LDY #$4          ;FOR ANY REMAINDER IN BIN
CF23- 20 34 CC  0066              JSR ROTATR       ;ROT R
CF26- 4C 1B CF  0067              JMP COMPEN       ;REPEAT LOOP UNTIL EXP=0
                0068 OUTDIG
CF29- A9 07     0069              LDA #$7          ;SET DIGIT CNTR
CF2B- 85 BC     0070              STA *CNTR
CF2D- A5 DE     0071              LDA *IOSTR3      ;FETCH BCD,SEE IF 1ST DIG
CF2F- F0 11     0072              BEQ ZERODG       ;YES, CHECK REM OF DIGITS
                0073 OUTDGS
CF31- A5 DE     0074              LDA *IOSTR3      ;GET BCD FROM OUT REG
CF33- 09 30     0075              ORA #'0          ;FORM ASCII CODE
CF35- 20 88 CF  0076              JSR BUFECHO      ;OUT DIGIT
                0077 DECRDG
CF38- C6 BC     0078              DEC *CNTR        ;DEC DIGIT CNTR
CF3A- F0 1A     0079              BEQ EXPOUT       ;=0,DONE OUT EXP
CF3C- 20 85 CE  0080              JSR DECBIN       ;ELSE,GET NEXT DIGIT
CF3F- 4C 31 CF  0081              JMP OUTDGS       ;FORM ASCII AND OUT
                0082 ZERODG
CF42- C6 DF     0083              DEC *IOEXPD      ;DEC EXP FOR SKIPPING DISP
CF44- A5 DD     0084              LDA *IOSTR2      ;CHECK IF MANTISSA =0
CF46- D0 F0     0085              BNE DECRDG       ;IF NOT 0,CONT OUT
CF48- A5 DC     0086              LDA *IOSTR1
CF4A- D0 EC     0087              BNE DECRDG
CF4C- A5 DB     0088              LDA *IOSTR
CF4E- D0 E8     0089              BNE DECRDG
CF50- A9 00     0090              LDA #$0          ;MANTISSA 0,CLEAR EXP
CF52- 85 DF     0091              STA *IOEXPD
CF54- F0 E2     0092              BEQ DECRDG       ;BEFORE FINISHING DISPLAY
                0093 EXPOUT
CF56- A9 45     0094              LDA #'E          ;SET ASCII FOR E
CF58- 20 88 CF  0095              JSR BUFECHO      ;DISPLAY E
CF5B- A5 DF     0096              LDA *IOEXPD      ;TEST IF NEG
CF5D- 30 05     0097              BMI EXOUTN       ;YES, DISPLAY - AND NEGATE
CF5F- A9 2B     0098              LDA #'+          ;NO, SET ASCII FOR +
CF61- 4C 6C CF  0099              JMP AHEAD2       ;DISPLAY EXP VAL
                0100 EXOUTN
CF64- 49 FF     0101              EOR #$FF         ;TWO'S COMP EXP
CF66- 85 DF     0102              STA *IOEXPD      ;TO MAKE NEG POS
CF68- E6 DF     0103              INC *IOEXPD      ;FOR OUT OF EXP VALUE
CF6A- A9 2D     0104              LDA #'-          ;SET ASCII FOR -
                0105 AHEAD2
CF6C- 20 88 CF  0106              JSR BUFECHO      ;OUT SIGN OF EXP
CF6F- A0 00     0107              LDY #$0
CF71- A5 DF     0108              LDA *IOEXPD      ;FETCH EXP
                0109 SUB12
CF73- 38        0110              SEC
CF74- E9 0A     0111              SBC #$0A         ;SUB TEN'S FROM EXP
CF76- 30 06     0112              BMI TOMUCH       ;IF MINUS,READY FOR OUT
CF78- 85 DF     0113              STA *IOEXPD      ;RESTORE POS RESULT
CF7A- C8        0114              INY              ;INC TEN'S CNTR
CF7B- 4C 73 CF  0115              JMP SUB12        ;CONT SUB
                0116 TOMUCH
CF7E- 98        0117              TYA              ;PUT MS DIGIT INTO A
CF7F- 09 30     0118              ORA #'0          ;FORM ASCII
CF81- 20 88 CF  0119              JSR BUFECHO      ;OUT TEN'S DIG TO DISPLAY
CF84- A5 DF     0120              LDA *IOEXPD      ;FETCH UNIT'S DIGIT
CF86- 09 30     0121              ORA #'0          ;FORM ASCII
```

```
                  ;
                  ;OUTBUF[0] IS A POINTER TO THE LAST CHAR IN OUTBUF
                  ;OUTBUF[1] TO OUTBUF[15] CONTAINS THE CHARS.
                  BUFECHO
CF88- E6 E5                      INC *OUTBUF          ;POINT TO NEXT FREE SPOT
CF8A- A6 E5                      LDX *OUTBUF
CF8C- 95 E5                      STA *OUTBUF,X        ;STORE CHAR
CF8E- 60                         RTS
                  ;
                  ;PUT "FPBUFOUT.S"
                                 .FI D26 "FPDISP.S"

03FC  3250-364C  FPDISP.S

                  ;PUT "FPDISP.S"

                  ;THIS ROUTINE DISPLAYS THE NUMBER IN FPACC
                  ;ON 4 DIGITS.  FOR NUMBERS FROM .1 TO 1000
                  ;THE NUMBER IS FIXED.  OTHERWISE IT IS .XXX.E
                  ;WITH THE DECIMAL ON E BEING THE SIGN BIT OF E.

                  FPDISP
CF8F- 20 B8 CE                   JSR FPOUT            ;CONV IN ASCII
CF92- A5 E6                      LDA *OUTBUF+1        ;SIGN
CF94- C9 2D                      CMP #'-
CF96- D0 08                      BNE =+9
CF98- AD 44 A6                   LDA DISBUF+4
CF9B- 09 80                      ORA #$80
CF9D- 8D 44 A6                   STA DISBUF+4
CFA0- A2 04                      LDX #$04             ;DISPLAY FIRST 3 DIGS.
CFA2- B5 E5                      LDA *OUTBUF,X
CFA4- 20 47 8A                   JSR OUTCHR           ;OUTPUT ASCII CHR
CFA7- E8                         INX
CFA8- E0 07                      CPX #$07
CFAA- D0 F6                      BNE =-9
CFAC- A6 E5                      LDX *OUTBUF
CFAE- B5 E5                      LDA *OUTBUF,X
CFB0- 29 0F                      AND #$0F             ;MASK OFF DIGIT
CFB2- A8                         TAY
CFB3- C9 05                      CMP #$05
CFB5- B0 14                      BCS @SCINOT          ;@SCINOT IF >4
CFB7- B5 E3                      LDA *OUTBUF-2,X
CFB9- C9 2D                      CMP #'-
CFBB- F0 0E                      BEQ @SCINOT          ;SCIENTIFIC NOTATION IF NE
CFBD- A5 EC                      LDA *OUTBUF+7        ;FIXED
CFBF- 20 47 8A                   JSR OUTCHR           ;OUTPUT LAST CHAR
CFC2- B9 41 A6                   LDA DISBUF+1,Y
CFC5- 09 80                      ORA #$80
CFC7- 99 41 A6                   STA DISBUF+1,Y
CFCA- 60                         RTS
                  @SCINOT
CFCB- 98                         TYA
CFCC- 20 44 8A                   JSR OUTNIB           ;OUTPUT NIBBLE IN A
CFCF- A6 E5                      LDX *OUTBUF
CFD1- B5 E3                      LDA *OUTBUF-2,X      ;SIGN BYTE
CFD3- C9 2D                      CMP #'-
CFD5- D0 08                      BNE =+9
CFD7- AD 45 A6                   LDA RMDIG
CFDA- 09 80                      ORA #$80             ;SET SIGN FLAG
CFDC- 8D 45 A6                   STA RMDIG
CFDF- AD 41 A6                   LDA DISBUF+1
CFE2- 09 80                      ORA #$80             ;SET MANTISSA DEC PNT
CFE4- 8D 41 A6                   STA DISBUF+1
CFE7- AD 44 A6                   LDA DISBUF+4
CFEA- 09 80                      ORA #$80             ;SET SCI NOT FLAG
CFEC- 8D 44 A6                   STA DISBUF+4
CFEF- 60                         RTS

                  ;PUT "FPDISP.S"
                                 .EN
END OF MAE PASS!
```

```
 ---  LABEL FILE:  ---

@ACCSET =CCA6          @ACNONZ =CC8B          @ACZERT =CC7E
@CKEQEX =CCC4          @CKRET =CCCD           @MINUS1 =CC9D
@MORACC =CCD7          @MOVOP =CCB5           @NONZAC =CCBF
@NORMEX =CC8A          @NOVFL =CCCE           @SCINOT =CFCB
@SHACOP =CCED          @SHIFTO =CCE5          @SKPNEG =CCE1
ACCESS =8B86           ADDER =CC55            ADOPP =CD3D
AHEAD1 =CEDA           AHEAD2 =CF6C           ARSHR =CC37
BA =CC00               BLKMOV =8740           BUFECHO =CF88
CKRND =CD68            CKSIGN =CD92           CLRMEM =CC00
CNTR =00BC             COMPEN =CF1B           COMPL1 =CC44
COMPLM =CC47           CONFIG =89A5           CROUND =CD6C
DECBIN =CE85           DECEXD =CEFA           DECEXT =CEE9
DECMEM =CC1E           DECOUT =CF00           DECRDG =CF38
DECREP =CEF5           DERROR =CDF2           DISBUF =A640
DIVIDE =CDDD           DVEXIT =CE29           EXMLDV =CD80
EXOUTN =CF64           EXPOUT =CF56           FILL3 =8718
FMPNT =00B8            FOLSWE =00C7           FOPEXP =00CB
FOPLSW =00C8           FOPMSW =00CA           FOPNSW =00C9
FPACCE =00C3           FPADD =CCB1            FPD10 =CE71
FPDISP =CF8F           FPDIV =CDC9            FPLSW =00C0
FPLSWE =00BF           FPMSW =00C2            FPMULT =CD26
FPNORM =CC71           FPNSW =00C1            FPOUT =CEB8
FPSUB =CD1C            FPX10 =CE5D            GOVEC =A659
ID =A64E               INBYTE =81D9           INCHR =8A1B
INCMEM =CC15           INEXPS =00D5           INMTAS =00D4
INPRDI =00D6           INTVEC =A678           IOEXP =00DA
IOEXPD =00DF           IOLSW =00D7            IOMSW =00D9
IONSW =00D8            IOSTR =00DB            IOSTR1 =00DC
IOSTR2 =00DD           IOSTR3 =00DE           KSCONF =89A3
L2 =84D3               MC =1000               MCANDO =00C4
MCAND1 =00C5           MCAND2 =00C6           MONITOR =8003
MOVIND =CC0A           MULTEX =CD91           MULTIP =CD34
NACCESS =8B9C          NADOPP =CD4A           NEGFPA =CDB2
NEGOP =CDC0            NOGO =CDF3             OPSGNT =CDBB
OUTBUF =00E5           OUTBYT =82FA           OUTCHR =8A47
OUTDGS =CF31           OUTDIG =CF29           OUTNEG =CED1
OUTNIB =8A44           OUTXAH =82F4           PARM =8220
PARN =A64A             PARNR =A649            PREXFR =CD76
QUOROT =CDF4           RESALL =81C4           RESTART =8000
RMDIG =A645            ROTATL =CC2C           ROTATR =CC34
ROTL =CC2D             SAVE2 =87EA            SAVER =8188
SCAND =8906            SDBYT =A651            SETDCT =CDD9
SETSUB =CE36           SHLOOP =CD0F           SIGNS =00BE
SPACE =8342            SUB12 =CF73            SUBBER =CC63
SUBEXP =CDD0           SUBR1 =CE50            TAPDEL =A630
TEMP1 =00E4            TOMUCH =CF7E           TOPNT =00BA
TPEXP =00E3            TPLSW =00E0            TPMSW =00E2
TPNSW =00E1            TSIGN =00BD            VECSW =8BB7
VIA1 =A000             VIA2 =A800             VIA3 =AC00
WORK0 =00CC            WORK1 =00CD            WORK2 =00CE
WORK3 =00CF            WORK4 =00D0            WORK5 =00D1
WORK6 =00D2            WORK7 =00D3            ZERODG =CF42

//0000,CFF0,13F0
```

```
;PUT "R2.1.C"
            .LS

;SYM CASSETTE READER FOR DATA LOGGER 3.0
;JULY 16,1981
;BY MICHAEL D. SCHWARZ

;AG ENGG EXTENSION
;KANSAS STATE UNIVERSITY

;VERSION 2.1

;AUGUST 22,1981
;MODIFIED TO GIVE DATA IN CRONOLOGICAL ORDER
;AND SKIP REPEATED DATA FROM BEFORE. REDONE
;INITIAL STRUCTURE.
;ADDED HEXDUMP MAIN PROGRAM.

;VERSION 2.2

;OCTOBER 12,1981
;ENTRY AND CONFIG TABLE CHANGED TO USR0-2 SELECTS
;MAIN ROUTINE AND NOT TABLE. ADDED USR2 ENTRY TO
;PRINT EOT AND JMP TO RESET ROUTINE.
;REONE INITIAL STRUCTURE.
;CORRECT PROBLEM WITH BLANK LINES FROM BLANK & NEWLINE

            .CE
            .CT
            .OS
            .ES
            .PR "PUT FPDISK IN DRIVE 2"
BA          .PR "ENTER BEGINNING ADDRESS"
            .IN BA
DATASTART   .PR "INPUT DATA STARTING PAGE"
            .IN DATASTART
PAGES       .PR "INPUT NUMBER OF MEMORY PAGES IN HEX"
            .IN PAGES
            .FI "SYM.L"
```

02BD 3494-3751 SYM.L

```
;PUT "SYM.L"

;MONITOR ADDRESSES

ACCESS     .DE $8B86
ASCNIB     .DE $8275
BLKMOV     .DE $8740
CONFIG     .DE $89A5
FILL3      .DE $8718
INBYTE     .DE $81D9
INCHR      .DE $8A1B
KSCONF     .DE $89A3
L2         .DE $84D3
MONITOR    .DE $8003
NACCESS    .DE $8B9C
OUTBYT     .DE $82FA
OUTCHR     .DE $8A47
OUTNIB     .DE $8A44
OUTXAH     .DE $82F4
PARM       .DE $8220
RESALL     .DE $81C4
RESTART    .DE $8000
SAVE2      .DE $87EA
SAVER      .DE $8188
SCAND      .DE $8906
SPACE      .DE $8342
TSTAT      .DE $8B3C
VECSW      .DE $8BB7

DISBUF     .DE $A640
```

```
0031 GOVEC      .DE $A659
0032 ID         .DE $A64E
0033 INTVEC     .DE $A678
0034 PARN       .DE $A64A
0035 PARNR      .DE $A649
0036 RMDIG      .DE $A645
0037 SDBYT      .DE $A651
0038 TAPDEL     .DE $A630
0039 TECHO      .DE $A653
0040
0041 VIA1       .DE $A000
0042 VIA2       .DE $A800
0043 VIA3       .DE $AC00
0044
0045 ;PUT "SYM.L"
0040            .FI "MACROS.M"
```

```
0225   3494-36B9  MACROS.M
```

```
0001 ;PUT "MACROS.M"
0002
0003            .MG
0004 !!!LDZI    .MD (...A ...B)
0005            LDA #L,...B
0006            STA *...A
0007            LDA #H,...B
0008            STA *...A+1
0009            .ME
0010
0011 !!!LDZZ    .MD (...A ...B)
0012            LDA *...B
0013            STA *...A
0014            LDA *...B+1
0015            STA *...A+1
0016            .ME
0017
0018 !!!ADZZ    .MD (...A ...B)
0019            CLC
0020            LDA *...A
0021            ADC *...B
0022            STA *...A
0023            LDA *...A+1
0024            ADC *...B+1
0025            STA *...A+1
0026            .ME
0027
0028 !!!SBZZ    .MD (...A ...B)
0029            SEC
0030            LDA *...A
0031            SBC *...B
0032            STA *...A
0033            LDA *...A+1
0034            SBC *...B+1
0035            STA *...A+1
0036            .ME
0037
0038 !!!CPZZ    .MD (...A ...B)
0039            LDA *...A+1
0040            CMP *...B+1
0041            BNE =+5
0042            LDA *...A
0043            CMP *...B
0044            .ME
0045
0046 ;PUT "MACROS.M"
0041            .BA $00
0042            .FI "R2ZPVARS.B"
```

```
0267   36B9-3920  R2ZPVARS.B
```

```
0001 ;PUT "R2ZPVARS.B"
0002
```

```
                0003 ;CONSTANTS
                0004
                0005 MAXAD       .DE 65
                0006 MAXDG       .DE 70
                0007 MAXPL       .DE 65
                0008 MAXDUMP     .DE 47
                0009 NODAT       .DE $EE
                0010
                0011 ;ZERO PAGE VARIABLES
                0012
0000-           0013 DATA        .DS 2
0002-           0014 END         .DS 2
0004-           0015 CURRENT     .DS 2
0006-           0016 START       .DS 2
0008-           0017 LENGTH      .DS 2
000A-           0018 LSTDAY      .DS 6
0010-           0019 LABEL       .DS 1
0011-           0020 ERROR       .DS 1
0012-           0021 DIVTMP      .DS 4
0016-           0022 BINARY      .DS 4
001A-           0023 BCD         .DS 5
001F-           0024 NBINARY     .DS 1
0020-           0025 NBCD        .DS 1
0021-           0026 ZERO        .DS 1
0022-           0027 COL         .DS 2
0024-           0028 ADCHAN      .DS 1
0025-           0029 DGCHAN      .DS 1
0026-           0030 PLCHAN      .DS 1
0027-           0031 CNTR1       .DS 1
0028-           0032 CNTR2       .DS 1
0029-           0033 STACK       .DS 1
002A-           0034 COUNT       .DS 1
002B-           0035 DLOOP1      .DS 1
002C-           0036 DLOOP2      .DS 1
                0037 ;   STRUCTURE OF THE CONFIG TABLE
002D-           0038 CHARDEL     .DS 1
002E-           0039 LINEDEL     .DS 1
002F-           0040 PROMPT      .DS 1
0030-           0041 LINEFEED    .DS 1
0031-           0042 SYNCFLAG    .DS 1
0032-           0043 BAUD        .DS 1
0033-           0044 MAINENTRY   .DS 2
                0045
                0046 ;PUT "R2ZPVARS.B"
                0043             .FI D26 "FPVARS.B"

02BD   36B9-3976   FPVARS.B

                0001 ;PUT "FPVARS.B"
                0002
                0003 ;VARIABLES FOR MATH ROUTINES
                0004
0035-           0005 FMPNT       .DS 2
0037-           0006 TOPNT       .DS 2
0039-           0007 CNTR        .DS 1
003A-           0008 TSIGN       .DS 1
003B-           0009 SIGNS       .DS 1
003C-           0010 FPLSWE      .DS 1
003D-           0011 FPLSW       .DS 1
003E-           0012 FPNSW       .DS 1
003F-           0013 FPMSW       .DS 1
0040-           0014 FPACCE      .DS 1
0041-           0015 MCAND0      .DS 1
0042-           0016 MCAND1      .DS 1
0043-           0017 MCAND2      .DS 1
0044-           0018 FOLSWE      .DS 1
0045-           0019 FOPLSW      .DS 1
0046-           0020 FOPNSW      .DS 1
0047-           0021 FOPMSW      .DS 1
0048-           0022 FOPEXP      .DS 1
0049-           0023 WORK0       .DS 1
004A-           0024 WORK1       .DS 1
```

```
004B-           0025 WORK2         .DS 1
004C-           0026 WORK3         .DS 1
004D-           0027 WORK4         .DS 1
004E-           0028 WORK5         .DS 1
004F-           0029 WORK6         .DS 1
0050-           0030 WORK7         .DS 1
                0031 ;       VARIABLES FOR I/O ROUTINES
0051-           0032 INMTAS        .DS 1
0052-           0033 INEXPS        .DS 1
0053-           0034 INPRDI        .DS 1
0054-           0035 IOLSW         .DS 1
0055-           0036 IONSW         .DS 1
0056-           0037 IOMSW         .DS 1
0057-           0038 IOEXP         .DS 1
0058-           0039 IOSTR         .DS 1
0059-           0040 IOSTR1        .DS 1
005A-           0041 IOSTR2        .DS 1
005B-           0042 IOSTR3        .DS 1
005C-           0043 IOEXPD        .DS 1
005D-           0044 TPLSW         .DS 1
005E-           0045 TPNSW         .DS 1
005F-           0046 TPMSW         .DS 1
0060-           0047 TPEXP         .DS 1
0061-           0048 TEMP1         .DS 1
0062-           0049 OUTBUF        .DS 16
                0050
                0051 ;PUT "FPVARS.B"
                0044               .BA BA
                0045               .MC $1000
                0046               .FI "R2MAIN.S"

0BC3  36B9-427C  R2MAIN.S

                0001 ;PUT "R2MAIN.S"
                0002
                0003 ;USRENTRY -- CALLED FROM THE MONITOR AS THE
                0004 ;UNRECOGNIZED COMMAND VECTOR.  USR0, USR1 & USR2
                0005 ;WITH 0 OR 1 PARAMETER ARE RECOGNIZED HERE AND
                0006 ;VECTORED TO THE APPROPIATE ROUTINE.
                0007
                0008 USRENTRY
C000- C9 14     0009              CMP #$14             ;USR0
C002- D0 09     0010              BNE =+10             ;NEXT COMMAND
C004- 20 24 C0  0011              JSR INIT             ;INITIALIZE
C007- 20 84 C0  0012              JSR MAIN             ;CALL MAIN ROUTINE
C00A- 4C 07 C0  0013              JMP =-3              ;RECALL MAIN
C00D- C9 15     0014              CMP #$15             ;USER 1
C00F- D0 09     0015              BNE =+10
C011- 20 24 C0  0016              JSR INIT
C014- 20 B1 C0  0017              JSR MAINDUMP
C017- 4C 14 C0  0018              JMP =-3
C01A- C9 16     0019              CMP #$16             ;USER 2
C01C- D0 2E     0020              BNE NEXT             ;END OF COMMANDS
C01E- 20 24 C0  0021              JSR INIT
C021- 4C C0 C0  0022              JMP PEOT             ;NO RE ENTRY HERE
                0023
                0024 ;INIT: CONTROL INITIALIZATION OF ZERO PAGE
                0025 ;GET PARAMETERS AND PREPARE FOR MAIN ROUTINE.
                0026
                0027 INIT
C024- 20 86 8B  0028              JSR ACCESS
C027- E0 00     0029              CPX #0
C029- D0 15     0030              BNE INITZP
                0031
                0032 ;GETABLE: PRINT MESSAGE ON LED'S AND ASK FOR
                0033 ;A SINGLE HEX CHAR TABLE NUMBER.  IF INVALID
                0034 ;ASK AGAIN.
                0035 ;STORE HEX NUMBER AS PARN 1 AND RETURN WITH X=1.
                0036
                0037 GETABLE
C02B- A2 05     0038              LDX #5
C02D- BD 6C C7  0039              LDA MESSAGE,X
```

```
C030- 9D 40 A6  0040                STA DISBUF,X        ;MESSAGE
C033- CA        0041                DEX
C034- 10 F7     0042                BPL =-8
                0043
C036- 20 1B 8A  0044                JSR INCHR           ;GET ASCII CHARACTER
C039- 20 75 82  0045                JSR ASCNIB          ;CONVERT TO HEX NIBBLE
C03C- B0 ED     0046                BCS GETABLE         ;RESTART IN NOT HEX
C03E- 90 03     0047                BCC INITZP+3        ;ALWAYS
                0048
                0049 ;INITZP: CHECK THAT TABLE NUMBER IS VALID,
                0050 ;ZERO ZP, TRANSFER TABLE, TRANSFER USER TABLE
                0051 ;IF SELECTED, AND SETUP FOR RS232 TRANSMITION.
                0052
                0053 INITZP
C040- AD 4A A6  0054                LDA PARN            ;1 PARN, TABLE ENTRY
C043- 0A        0055                ASL A
C044- 0A        0056                ASL A
C045- 0A        0057                ASL A
C046- C9 58     0058                CMP #TABLENGTH      ;OVERRUN?
C048- 90 04     0059                BCC NEXT+2
                0060
C04A- 68        0061                PLA                 ;ERROR RETURN
C04B- 68        0062                PLA                 ;BYPASS CALLING ROUTINE
                0063 NEXT
C04C- 38        0064                SEC
C04D- 60        0065                RTS
                0066
C04E- A8        0067                TAY
C04F- A9 00     0068                LDA #0              ;CLR ZERO PAGE MEMORY
C051- A2 F8     0069                LDX #$F8
C053- CA        0070                DEX
C054- 95 00     0071                STA *0,X
C056- D0 FB     0072                BNE =-4
                0073
C058- B9 7A C7  0074                LDA CONFIGTAB,Y
C05B- 95 2D     0075                STA *CHARDEL,X      ;SAVE IN ZERO PAGE
C05D- C8        0076                INY
C05E- E8        0077                INX
C05F- E0 08     0078                CPX #8              ;8 VALUES
C061- D0 F5     0079                BNE =-10
                0080
C063- A5 32     0081                LDA *BAUD           ;USE USR VALUES?
C065- C9 FF     0082                CMP #$FF
C067- D0 0A     0083                BNE @RI             ;USER VALUES IF BAUD=0
                0084
C069- A2 07     0085                LDX #7              ;8 USR VALUES
C06B- BD 00 01  0086                LDA $100,X          ;USR VALUE LOCATION
C06E- 95 2D     0087                STA *CHARDEL,X
C070- CA        0088                DEX
C071- 10 F8     0089                BPL =-7
                0090 @RI
C073- 20 B7 8B  0091                JSR VECSW           ;SET RS232
C076- A9 00     0092                LDA #0
C078- 8D 53 A6  0093                STA TECHO
C07B- A5 32     0094                LDA *BAUD           ;SET BAUD RATE
C07D- 8D 51 A6  0095                STA SDBYT
C080- BA        0096                TSX
C081- 86 29     0097                STX *STACK          ;SAVE STACK FOR FIXUP
C083- 60        0098                RTS                 ;NON ERROR RETURN
                0099
                0100 ;THIS IS THE MAIN ROUTINE THAT CONTROLS THE TRANSFER
                0101 ;IT IS CALLED AFTER THE SETUP ROUTINE FINNISHES AND
                0102 ;CONTROL IS TRANSFERED HERE FROM EOF WHEN THE END
                0103 ;OF DATA IS REACHED.
                0104
                0105 MAIN
C084- 20 CE C0  0106                JSR LOAD            ;GET A RECORD FROM TAPE
C087- 20 46 C2  0107                JSR SYNC            ;GET IN SYNC WITH RECIEVER
C08A- 20 46 C3  0108                JSR OUTHEAD         ;PRINT HEADER
C08D- 20 22 C1  0109                JSR CALCLENGTH      ;CALCUATE RECORD LENGTH
C090- 20 45 C1  0110                JSR FINDFIRST       ;FIND LAST RECORD IN MEMOR
                0111 MAINLOOP
```

```
C093- 20 9C C1              JSR NEXTRECORD        ;CALC NEXT RECORD
C096- 20 F6 C2              JSR AVGFACT           ;CALCUATE AVERAGE FACTOR
C099- 20 66 C3              JSR OUTDATE           ;PRINT RECORD DATE
C09C- A5 24                 LDA *ADCHAN
C09E- 20 95 C3              JSR OUTAD             ;OUTPUT ADCHANNELS
C0A1- A5 25                 LDA *DGCHAN
C0A3- 20 E9 C3              JSR OUTDG             ;OUTPUT DIGITAL CHANNELS
C0A6- A5 26                 LDA *PLCHAN
C0A8- 20 0C C4              JSR OUTPL             ;OUTPUT PULSE CHANNELS
C0AB- 20 09 C2              JSR EOR               ;END OF RECORD
C0AE- 4C 93 C0              JMP MAINLOOP          ;CONTINUE

            ;MAINDUMP: THIS MAIN ROUTINE CONTROLS A HEXIDECIMAL
            ;DUMP OF THE TAPE.  THE FILE NAME HEADER AND END
            ;AT THE END OF FILE ARE ADDED. THE LENGTH OF LINE
            ;IS CONTROLED BY MAXDUMP.

            MAINDUMP
C0B1- 20 CE C0              JSR LOAD              ;GET FILE
C0B4- 20 46 C2              JSR SYNC              ;GET READY
C0B7- 20 25 C3              JSR OUTLAB            ;PRINT FILE NAME
C0BA- 20 09 C2              JSR NEWLINE           ;END OF FILE NAME
C0BD- 4C 3D C4              JMP OUTDUMP           ;DUMP, EOF WILL RETURN.

            ;PEOT: PRINT END OF TAPE AND EXIT TO RESET ROUTINE

            PEOT
C0C0- 20 46 C2              JSR SYNC
C0C3- A2 04                 LDX #4                ;EOT
C0C5- 20 E9 C1              JSR WMESG+2
C0C8- 20 E7 C1              JSR WMESG             ;END
C0CB- 6C FC C7              JMP ($C7FC)           ;RESET VECTOR ON CHIP

            ;PUT "R2MAIN.S"
                            .FI "R2FILE.S"

0B79  36B9-4232  R2FILE.S

            ;PUT "R2FILE.S"

            ;LOAD IS CALLED TO READ A FILE FROM TAPE.  IT SETS
            ;POINTERS AS TO THE BEGINNING AND END OF THE DATA,
            ;A FLAG FOR A READ ERROR, AND THE FILE LABEL.  THE
            ;BEGINNING POINTER IS NOT SAVED FROM THE LOAD SO THE
            ;MEMORY IS SET TO A UNIQUE EMPTY BYTE AND SEARCHED
            ;FOR A DIFFERENT BYTE AFTER THE READ.

            LOAD
C0CE- 20 14 C1              JSR EMPTY             ;ZERO MEMORY
C0D1- A9 00                 LDA #$00
C0D3- 8D 4E A6              STA ID                ;TAPE LABEL (ANY FILE)
C0D6- 85 11                 STA *ERROR            ;NO ERROR YET
C0D8- 20 D3 84              JSR L2                ;READ FILE FROM TAPE
C0DB- AD 00 A4              LDA $A400             ;TAPE LABEL IS HERE
C0DE- 85 10                 STA *LABEL
C0E0- 90 02                 BCC =+3               ;READ ERROR IF CARRY SET
C0E2- C6 11                 DEC *ERROR            ;READ ERROR
            ;JMP FIND ;SET POINTERS AND RETURN

            ;FIND THE FIRST LOCATION THAT DOESN'T HAVE
            ;A EMPTY BYTE IN IT AND ASSUME THAT IS THE FIRST
            ;LOCATION READ OFF TAPE. IF NOT FOUND SET
            ;BOTH DATA AND END TO 0.

            FIND
C0E4- 20 07 C1              JSR SETDAT
C0E7- D1 00                 CMP (DATA),Y
C0E9- D0 11                 BNE FOUND
C0EB- C8                    INY
C0EC- D0 F9                 BNE FIND+3
C0EE- E6 01                 INC *DATA+1
C0F0- CA                    DEX
```

```
C0F1- D0 F4                   BNE FIND+3
C0F3- A9 00                   LDA #0              ;FOUND NO DATA CLR POINTER
C0F5- 85 01                   STA *DATA+1
C0F7- 85 02                   STA *END
C0F9- 85 03                   STA *END+1
C0FB- 60                      RTS
                  FOUND
C0FC- 84 00                   STY *DATA           ;LOW BYTE OF BEGINNING ADD
                              LDZZ (END $FE)

C0FE- A5 FE
C100- 85 02
C102- A5 FF
C104- 85 03

C106- 60                      RTS

                  ;EMPTY, AND FIND ARE CALLED FROM LOAD. SETDAT IS
                  ;CALLED FROM EMPTY AND FIND.

                  SETDAT
C107- A0 00                   LDY #0
C109- 84 00                   STY *DATA
C10B- A9 02                   LDA #DATASTART      ;STARTING PAGE OF MEMORY
C10D- 85 01                   STA *DATA+1
C10F- A9 EE                   LDA #NODAT          ;BYTE TO USE FOR EMPTY MEM
C111- A2 0E                   LDX #PAGES          ;PAGES OF MEMORY USED
C113- 60                      RTS

                  ;FILL MEMORY FROM START FOR PAGES WITH EMPTY

                  EMPTY
C114- 20 07 C1                JSR SETDAT
C117- 91 00                   STA (DATA),Y
C119- C8                      INY
C11A- D0 FB                   BNE EMPTY+3
C11C- E6 01                   INC *DATA+1
C11E- CA                      DEX
C11F- D0 F6                   BNE EMPTY+3
C121- 60                      RTS

                  ;CALCLENGTH: CALCUATE LENGTH OF RECORD FROM THE
                  ;NUMBER OF AD, DG, AND PL CHANNELS.

                  CALCLENGTH
C122- A9 00                   LDA #0              ;INITIALIZE
C124- 85 09                   STA *LENGTH+1
C126- 18                      CLC
C127- A9 02                   LDA #2              ;TIME & CNT
C129- 65 26                   ADC *PLCHAN         ;NO CARRY
C12B- 65 24                   ADC *ADCHAN         ;4 EACH AT THIS POINT
C12D- 90 02                   BCC =+3
C12F- E6 09                   INC *LENGTH+1
C131- 0A                      ASL A
C132- 26 09                   ROL *LENGTH+1
C134- A6 26                   LDX *PLCHAN
C136- F0 01                   BEQ =+2
C138- 38                      SEC                 ;ADD IN SPACE FOR PLCLOCK
C139- 65 25                   ADC *DGCHAN
C13B- 90 02                   BCC =+3
C13D- E6 09                   INC *LENGTH+1       ;ADD CARRY
C13F- 0A                      ASL A
C140- 26 09                   ROL *LENGTH+1       ;MULT BY 2
C142- 85 08                   STA *LENGTH
C144- 60                      RTS

                  ;FINDFIRST: FIND THE EARLIEST COMPLETE RECORD
                  ;THAT WAS TAKEN AFTER THE LAST RECORD FROM THE
                  ;PREVIOUS DATA.

                  FINDFIRST
C145- A5 00                   LDA *DATA           ;INITIALIZE START AND
```

```
C147- 85 06     0101            STA *START          ;CURRENT WITH DATA
C149- 85 04     0102            STA *CURRENT
C14B- A5 01     0103            LDA *DATA+1
C14D- 85 07     0104            STA *START+1
C14F- 85 05     0105            STA *CURRENT+1
                0106 @RA
C151- 20 67 C1  0107            JSR TESTREC         ;END YET?
C154- B0 01     0108            BCS =+2
C156- 60        0109            RTS
                0110            ADZZ (CURRENT LENGTH)

C157- 18
C158- A5 04
C15A- 65 08
C15C- 85 04
C15E- A5 05
C160- 65 09
C162- 85 05

C164- 4C 51 C1  0111            JMP @RA             ;TRY AGAIN
                0112
                0113 TESTREC
                0114            CPZZ (END CURRENT)

C167- A5 03
C169- C5 05
C16B- D0 04
C16D- A5 02
C16F- C5 04

C171- B0 0F     0115            BCS @RB             ;OUT OF DATA?
                0116            SBZZ (CURRENT LENGTH)

C173- 38
C174- A5 04
C176- E5 08
C178- 85 04
C17A- A5 05
C17C- E5 09
C17E- 85 05

C180- 18        0117            CLC
C181- 60        0118            RTS
                0119 @RB
C182- 24 30     0120            BIT *LINEFEED       ;STRIP REPEATS??
C184- 70 FB     0121            BVS @RB-1           ;CARRY SET, RETURN
C186- A0 01     0122            LDY #1
C188- B1 04     0123            LDA (CURRENT),Y     ;MONTH
C18A- F0 F4     0124            BEQ @RB-2           ;END IF ZERO MONTH
C18C- A0 05     0125            LDY #5              ;CHECK FOR OLD DATA
C18E- B1 04     0126            LDA (CURRENT),Y
C190- D9 0A 00  0127            CMP LSTDAY,Y
C193- F0 02     0128            BEQ =+3
C195- 38        0129            SEC
C196- 60        0130            RTS
C197- 88        0131            DEY
C198- 10 F4     0132            BPL =-11
C19A- 18        0133            CLC
C19B- 60        0134            RTS
                0135
                0136 ;NEXTRECORD: SET POINTERS TO NEXT RECORD AND
                0137 ;TEST FOR DONE
                0138
                0139 NEXTRECORD
C19C- 38        0140            SEC
C19D- A5 04     0141            LDA *CURRENT        ;ADD CURRENT AND LENGTH
C19F- E5 08     0142            SBC *LENGTH         ;AND STORE AT CURRENT
C1A1- 85 04     0143            STA *CURRENT
C1A3- 85 00     0144            STA *DATA           ;ALSO AT DATA
C1A5- A5 05     0145            LDA *CURRENT+1
C1A7- E5 09     0146            SBC *LENGTH+1
C1A9- 85 05     0147            STA *CURRENT+1
```

```
C1AB- 85 01                        STA *DATA+1
C1AD- C5 07                        CMP *START+1          ;AT BEGINNING OF DATA?
C1AF- D0 04                        BNE =+5
C1B1- A5 04                        LDA *CURRENT
C1B3- C5 06                        CMP *START
C1B5- 90 18                        BCC EOF               ;YES END OF FILE
C1B7- 60                           RTS

                  ;PUT "R2FILE.S"
                                   .FI "R2IOSUBS.S"

08EC  36B9-3FA5  R2IOSUBS.S

                  ;PUT "R2IOSUBS.S"

                  ;GETB: GETS THE NEXT BYTE OF DATA. EXITS TO
                  ;EOF IF NO MORE DATA.

                  GETB
                                   CPZZ (DATA END)

C1B8- A5 01
C1BA- C5 03
C1BC- D0 04
C1BE- A5 00
C1C0- C5 02

C1C2- B0 0B                        BCS EOF
C1C4- A0 00                        LDY #0
C1C6- B1 00                        LDA (DATA),Y          ;GET NEXT DATA BYTE
C1C8- E6 00                        INC *DATA             ;INCREMENT POINTER
C1CA- D0 02                        BNE =+3
C1CC- E6 01                        INC *DATA+1
C1CE- 60                           RTS                   ;RETURN WITH DATA

                  ;EOF: CALLED WHEN END OF FILE. PRINTS END
                  ;PATCHES UP THE STACK AND JMPS TO MAIN

                  EOF
C1CF- A5 11                        LDA *ERROR
C1D1- D0 0A                        BNE =+11              ;DON'T UPDATE IF ERROR
C1D3- A0 05                        LDY #5
C1D5- B1 06                        LDA (START),Y
C1D7- 99 0A 00                     STA LSTDAY,Y
C1DA- 88                           DEY
C1DB- 10 F8                        BPL =-7

C1DD- 20 09 C2                     JSR NEWLINE           ;PRINT CR IF NEEDED
C1E0- 20 E7 C1                     JSR WMESG             ;WRITE END
C1E3- A6 29                        LDX *STACK            ;PACHUP STACK
C1E5- 9A                           TXS
C1E6- 60                           RTS                   ;RETURN TO FIRST CALL

                  WMESG
C1E7- A2 00                        LDX #0                ;DEFAULT MESG
C1E9- BD 72 C7                     LDA MESGS,X           ;LOAD CHAR
C1EC- F0 1B                        BEQ NEWLINE           ;END OF MESG
C1EE- 20 FF C1                     JSR OUTCHAR
C1F1- E8                           INX
C1F2- D0 F5                        BNE WMESG+2           ;UNCONDITIONAL

                  BLANK
C1F4- A5 23                        LDA *COL+1            ;DON'T PRINT SPACE ON COL
C1F6- 05 22                        ORA *COL              ;PRINT SPACE IF NOT COL 1
C1F8- D0 03                        BNE =+4
C1FA- E6 23                        INC *COL+1            ;ONLY SKIP FIRST BLANK
C1FC- 60                           RTS
C1FD- A9 20                        LDA #'                ;PRINT A SPACE
                  OUTCHAR
C1FF- E6 22                        INC *COL              ;INCREMENT COLUMN COUNT
C201- 20 47 8A                     JSR OUTCHR
C204- A5 2D                        LDA *CHARDEL          ;DELAY BETWEEN CHARACTERS
```

```
C206- 4C 26 C2                JMP DELAY

               EOR
               ;JSR NEWLINE
               ;ANY END OF RECORD DELAY PROCESS GOES HERE
               ;RTS

               NEWLINE
C209- A5 22                   LDA *COL               ;CHECK COLUMN
C20B- F0 2C                   BEQ RTS                ;DON'T PRINT BLANK LINES
C20D- A9 0D                   LDA #$0D               ;OUTPUT A CR
C20F- 20 FF C1                JSR OUTCHAR
C212- 24 30                   BIT *LINEFEED          ;NEED A LF OUTPUT
C214- 10 05                   BPL =+6
C216- A9 0A                   LDA #$0A               ;OUTPUT A LF
C218- 20 FF C1                JSR OUTCHAR
C21B- 20 3A C2                JSR WAIT               ;GET PROMPT IF ANY
C21E- A9 00                   LDA #0
C220- 85 23                   STA *COL+1
C222- 85 22                   STA *COL               ;START OF LINE
C224- A5 2E                   LDA *LINEDEL           ;END OF LINE DELAY
               ;JSR DELAY
               ;RTS

               ;DELAY: DELAY FOR 4*A MS.  IF A IS 0 THEN RETURN

               DELAY
C226- F0 11                   BEQ RTS
C228- 85 2B                   STA *DLOOP1
C22A- A9 C8                   LDA #200               ;20 MICRO SEC EACH
C22C- 85 2C                   STA *DLOOP2
C22E- 20 39 C2                JSR RTS                ;EATUP TIME
C231- C6 2C                   DEC *DLOOP2
C233- D0 F9                   BNE DELAY+8
C235- C6 2B                   DEC *DLOOP1
C237- D0 F1                   BNE DELAY+4
C239- 60       RTS            RTS

               ;WAIT: WAIT FOR A PROMPT CHARACTER BEFORE CONTINUING
               ;TO OUTPUT DATA

               WAIT
C23A- A5 2F                   LDA *PROMPT
C23C- F0 FB                   BEQ RTS
C23E- 20 1B 8A                JSR INCHR
C241- C5 2F                   CMP *PROMPT
C243- D0 F9                   BNE WAIT+4
C245- 60                      RTS

               ;SYNC: GET IN SYNC WITH THE RECIVER IF NECESSARY
               ;PROTOCALL:  TRANSMITTER: SEND A CR EVERY SECOND
               ;UNTIL A CHARACTER IS RECIEVED, THEN SEND A NEWLINE
               ;AND START DATA. RECIEVER: WAIT FOR A CR, THEN OUTPUT
               ;A CHARACTER TO SIGNAL TO SIGNAL READY FOR DATA.

               SYNC
C246- 24 31                   BIT *SYNCFLAG          ;DO A SYNC?
C248- 10 EF                   BPL RTS                ;NO.
C24A- A9 0D                   LDA #$0D               ;OUTPUT A CR
C24C- 20 47 8A                JSR OUTCHR
C24F- A0 20                   LDY #$20
               @SYNC2
C251- 20 3C 8B                JSR TSTAT              ;GET TERMINAL STATUS
C254- B0 06                   BCS =+7
C256- CA                      DEX
C257- D0 F8                   BNE @SYNC2
C259- 88                      DEY
C25A- D0 F5                   BNE @SYNC2

C25C- 90 EC                   BCC SYNC+4
C25E- A9 80                   LDA #$80               ;A SHORT DELAY
C260- 4C 26 C2                JMP DELAY              ;THEN PRINT DATA
```

```
                     
                     ;PUT "R2IOSUBS.S"
                               .FI "R2CONV.S"

09DE  36B9-4097  R2CONV.S

                     ;PUT "R2CONV.S"
                     
                     ;OUTDOUBLE: OUTPUT 2 BYTES AS 6 DIGITS BCD
                     
                     OUTDOUBLE
C263- A2 02                    LDX #2               ;2 IN AND 3 OUT
C265- D0 08                    BNE OUTBIN+4
                     
                     ;OUTQUAD: OUTPUT 4 BYTES AS 11 DIGITS BCD
                     
                     OUTQUAD
C267- A2 04                    LDX #4               ;4 IN AND 5 OUT
C269- D0 04                    BNE OUTBIN+4
                     
                     ;OUTBIN: OUTPUT A BINARY BYTE AS A 4 DIGIT BCD
                     ;NUMBER LEAVING OFF LEADING ZEROS.
                     
                     OUTBIN
C26B- 85 16                    STA *BINARY          ;STORE IN LOW BYTE
C26D- A2 01                    LDX #1
C26F- 86 1F                    STX *NBINARY         ;1 BYTE IN
                     ;BNE CONVERT ;ALWAYS, JUMP TO CONVERT
                     
                     ;OUTPUT NBINARY BYTE NUMBER AS (NBINARY+1)*2 DIGIT
                     ;BCD LEAVING OFF LEADING ZEROS.  OUTBIN CALLS
                     ;THIS ROUTINE
                     
                     CONVERT
C271- A9 00                    LDA #0
C273- 85 21                    STA *ZERO            ;LEADING ZERO FLAG
C275- A5 1F                    LDA *NBINARY
C277- 0A                       ASL A
C278- 0A                       ASL A
C279- 0A                       ASL A                ;NUMBER OF BITS
C27A- 85 28                    STA *CNTR2           ;BIT COUNTER
C27C- A9 00                    LDA #0               ;ZERO RESULT
C27E- A2 04                    LDX #4
C280- 95 1A                    STA *BCD,X
C282- CA                       DEX
C283- 10 FB                    BPL =-4
C285- F8                       SED                  ;START CONVERSION
                     @RD
C286- A4 1F                    LDY *NBINARY         ;NUMBER OF BYTES
C288- A2 00                    LDX #0
C28A- 36 16                    ROL *BINARY,X        ;SHIFT OUT MSB
C28C- E8                       INX
C28D- 88                       DEY
C28E- D0 FA                    BNE @RD+4
C290- A2 00                    LDX #0               ;ADD CARRY AND
C292- A4 1F                    LDY *NBINARY         ;MULT BY 2 IN BCD
C294- B5 1A                    LDA *BCD,X
C296- 75 1A                    ADC *BCD,X
C298- 95 1A                    STA *BCD,X
C29A- E8                       INX
C29B- 88                       DEY
C29C- 10 F6                    BPL =-9              ;NBINARY+1 BCD DIGITS
C29E- C6 28                    DEC *CNTR2
C2A0- D0 E4                    BNE @RD
                     ;    PRINT
C2A2- D8                       CLD
C2A3- A6 1F                    LDX *NBINARY         ;NUMBER OF BYTES-1
C2A5- B5 1A                    LDA *BCD,X
C2A7- 20 B9 C2                 JSR OUTNUM
C2AA- CA                       DEX
C2AB- D0 F8                    BNE =-7
C2AD- A5 1A                    LDA *BCD             ;LAST 2 DIGITS
```

```
C2AF- 24 21                         BIT *ZERO          ;PRINT AT LEAST 1 DIGIT
C2B1- 30 03                         BMI =+4
C2B3- 4C D8 C2                      JMP OUTHEX
C2B6- 4C D0 C2                      JMP OUTHEXZ        ;HAS BEEN ZEROS TO HERE

                  OUTNUM
C2B9- 48                            PHA
C2BA- 4A                            LSR A
C2BB- 4A                            LSR A
C2BC- 4A                            LSR A
C2BD- 4A                            LSR A
C2BE- 20 C2 C2                      JSR =+4            ;HIGH CHAR
C2C1- 68                            PLA                ;LOW CHAR

C2C2- D0 07                         BNE =+8
C2C4- 24 21                         BIT *ZERO          ;CHECK IF 0 CAN BE PRINTED
C2C6- 30 03                         BMI =+4            ;OUTPUT THE DIGIT EVEN 0
C2C8- 4C F4 C1                      JMP BLANK          ;OUTPUT CHAR AND RETURN
C2CB- C6 21                         DEC *ZERO          ;OUTPUT ZEROS FROM NOW ON
C2CD- 4C E9 C2                      JMP NIBASC         ;OUTPUT CHAR AND RETURN

                  ;OUTHEXZ: OUTPUT A BYTE WITH LEADING ZEROS

                  OUTHEXZ
C2D0- 48                            PHA
C2D1- 4A                            LSR A              ;HIGH NIBBLE
C2D2- 4A                            LSR A
C2D3- 4A                            LSR A
C2D4- 4A                            LSR A
C2D5- 4C E5 C2                      JMP OUTHEX2

                  ;OUTHEX: OUTPUT A BYTE WITHOUT LEADING ZEROS

                  OUTHEX
C2D8- 48                            PHA
C2D9- 4A                            LSR A              ;HIGH NIBBLE
C2DA- 4A                            LSR A
C2DB- 4A                            LSR A
C2DC- 4A                            LSR A
C2DD- D0 06                         BNE OUTHEX2        ;SKIP SPACE CALL
C2DF- 20 F4 C1                      JSR BLANK
C2E2- 4C E8 C2                      JMP OUTHEX2+3      ;SKIP NIBASC CALL
                  OUTHEX2
C2E5- 20 E9 C2                      JSR NIBASC         ;OUTPUT HEX BYTE
C2E8- 68                            PLA
                  NIBASC
C2E9- 29 0F                         AND #$0F
C2EB- C9 0A                         CMP #$0A           ;HEX A-F DIGIT
C2ED- 90 02                         BCC =+3
C2EF- 69 06                         ADC #$06           ;A-F OFFSET
C2F1- 69 30                         ADC #'0            ;CONVERT TO ASCII
C2F3- 4C FF C1                      JMP OUTCHAR        ;OUTPUT THE DIGIT

                  ;AVGFACT: CALCUATE THE FACTOR TO MULTIPLY
                  ;BY THE ANALOG SUMS TO GET AVERAGE VOLTS.

                  AVGFACT
C2F6- A9 50                         LDA #$50           ;5 VOLTS
C2F8- 85 47                         STA *FOPMSW
C2FA- A9 03                         LDA #3
C2FC- 85 48                         STA *FOPEXP
C2FE- A0 06                         LDY #6
C300- B1 04                         LDA (CURRENT),Y    ;COUNT TO DIVIDE BY
C302- 85 3E                         STA *FPNSW
C304- C8                            INY
C305- B1 04                         LDA (CURRENT),Y
C307- 85 3F                         STA *FPMSW         ;ITS LESS THAN 32000
C309- A9 0F                         LDA #15
C30B- 85 40                         STA *FPACCE        ;15 BITS WITHOUT SIGN
C30D- A9 00                         LDA #0
C30F- 85 3D                         STA *FPLSW
C311- 85 45                         STA *FOPLSW
```

```
C313- 85 46     0139                STA *FOPNSW
C315- 20 C3 C4  0140                JSR FPNORM              ;NORMALIZE COUNT TO FP
C318- 20 A6 C5  0141                JSR FPDIV
C31B- A2 03     0142                LDX #3                  ;4 BYTES
C31D- B5 3D     0143                LDA *FPLSW,X
C31F- 95 12     0144                STA *DIVTMP,X
C321- CA        0145                DEX
C322- 10 F9     0146                BPL =-6
C324- 60        0147                RTS
                0148
                0149 ;PUT "R2CONV.S"
                0050                .FI "R2DATOUT.S"

0BE8  36B9-42A1  R2DATOUT.S

                0001 ;PUT "R2DATOUT.S"
                0002
                0003 ;OUTHEAD: PRINTS THE HEADER INFO
                0004
                0005 OUTLAB
C325- A9 46     0006                LDA #'F                 ;PRINT FILE NAME
C327- 20 FF C1  0007                JSR OUTCHAR
C32A- A5 2A     0008                LDA *COUNT
C32C- E6 2A     0009                INC *COUNT              ;SIMPLE FILE COUNTER
C32E- 20 D0 C2  0010                JSR OUTHEXZ
C331- A9 4C     0011                LDA #'L
C333- 20 FF C1  0012                JSR OUTCHAR
C336- A5 10     0013                LDA *LABEL              ;TAPE LABEL
C338- 20 D0 C2  0014                JSR OUTHEXZ
C33B- A9 20     0015                LDA #'                  ;SPACE
C33D- 24 11     0016                BIT *ERROR              ;TAPE READ ERROR?
C33F- 10 02     0017                BPL =+3
C341- A9 45     0018                LDA #'E                 ;INDICATE ERROR
C343- 4C FF C1  0019                JMP OUTCHAR
                0020 OUTHEAD
C346- 20 25 C3  0021                JSR OUTLAB              ;PRINT LABEL
C349- A0 04     0022                LDY #4                  ;SAVE DATA COUNTS FOR LATE
C34B- A2 02     0023                LDX #2
C34D- B1 00     0024                LDA (DATA),Y
C34F- 95 24     0025                STA *ADCHAN,X
C351- 88        0026                DEY
C352- CA        0027                DEX
C353- 10 F8     0028                BPL =-7
C355- A9 08     0029                LDA #8                  ;OUTPUT 8 BYTES OF HEADER
C357- 85 27     0030                STA *CNTR1
C359- 20 B8 C1  0031                JSR GETB                ;GET NEXT BYTE
C35C- 20 6B C2  0032                JSR OUTBIN              ;OUTPUT AS BCD
C35F- C6 27     0033                DEC *CNTR1
C361- D0 F6     0034                BNE =-9
C363- 4C 09 C2  0035                JMP NEWLINE             ;PUT A CR
                0036
                0037 ;OUTDATE: PTINTS THE DATE AND COUNT INFO FROM A RECORD
                0038
                0039 OUTDATE
C366- A2 02     0040                LDX #2                  ;2 SECTIONS
C368- 20 F4 C1  0041                JSR BLANK               ;SEPERATE DATE
C36B- 20 B8 C1  0042                JSR GETB                ;HOUR
C36E- 20 D8 C2  0043                JSR OUTHEX
C371- 20 8A C3  0044                JSR OUTDTB              ;MINUTE
C374- 20 8A C3  0045                JSR OUTDTB              ;SECOND
C377- CA        0046                DEX
C378- D0 EE     0047                BNE OUTDATE+2
C37A- 20 B8 C1  0048                JSR GETB
C37D- 85 16     0049                STA *BINARY             ;COUNT
C37F- 20 B8 C1  0050                JSR GETB
C382- 85 17     0051                STA *BINARY+1
C384- 20 63 C2  0052                JSR OUTDOUBLE
C387- 4C 09 C2  0053                JMP NEWLINE
                0054
                0055 OUTDTB
C38A- A9 2F     0056                LDA #'/                 ;BYTE SEPERATOR
C38C- 20 FF C1  0057                JSR OUTCHAR             ;OUTPUT SEPERATION CHAR
```

```
C38F- 20 B8 C1  0058                JSR GETB             ;GET NEXT BYTE
C392- 4C D0 C2  0059                JMP OUTHEXZ          ;OUTPUT BYTE
                0060
                0061 ;OUTAD: AVERAGES AND PRINTS FLOATING POINT VALUES
                0062 ;FROM THE ADC READINGS.
                0063
                0064 OUTAD
C395- 85 27     0065                STA *CNTR1           ;NUMBER OF VALUES
C397- D0 01     0066                BNE =+2
C399- 60        0067                RTS
C39A- A2 00     0068                LDX #0               ;FP INDEX PNTR
C39C- 20 B8 C1  0069                JSR GETB
C39F- 95 3D     0070                STA *FPLSW,X         ;PUT IN FPACC
C3A1- E8        0071                INX
C3A2- E0 04     0072                CPX #4               ;4 BYTES PER FP NUM
C3A4- D0 F6     0073                BNE =-9
C3A6- 24 31     0074                BIT *SYNCFLAG        ;AVERAGE?
C3A8- 70 0C     0075                BVS @RE              ;NO.
C3AA- A2 03     0076                LDX #3               ;LOAD DIVIDE FACTOR
C3AC- B5 12     0077                LDA *DIVTMP,X
C3AE- 95 45     0078                STA *FOPLSW,X
C3B0- CA        0079                DEX
C3B1- 10 F9     0080                BPL =-6
C3B3- 20 03 C5  0081                JSR FPMULT           ;AVERAGE
                0082 @RE
C3B6- 20 C9 C3  0083                JSR FPRINT           ;PRINT RESULT
C3B9- A5 22     0084                LDA *COL
C3BB- C9 41     0085                CMP #MAXAD           ;END OF LINE YET?
C3BD- 90 03     0086                BCC =+4
C3BF- 20 09 C2  0087                JSR NEWLINE          ;START NEW LINE
C3C2- C6 27     0088                DEC *CNTR1
C3C4- D0 D4     0089                BNE OUTAD+5
C3C6- 4C 09 C2  0090                JMP NEWLINE          ;END OF LINE
                0091
                0092 ;FPRINT: PRINT THE NUMBER IN THE FPACC IN FLOATING
                0093 ;POINT FORMAT.
                0094
                0095 FPRINT
C3C9- 20 95 C6  0096                JSR FPOUT            ;CONVERT TO DECIMAL
C3CC- 20 F4 C1  0097                JSR BLANK            ;PUT SPACE
C3CF- A2 01     0098                LDX #1               ;PRINT OUT MANTISA
C3D1- B5 62     0099                LDA *OUTBUF,X
C3D3- 20 FF C1  0100                JSR OUTCHAR          ;OUTPUT CHAR
C3D6- E8        0101                INX
C3D7- E0 08     0102                CPX #8               ;END OF VALID DATA
C3D9- D0 F6     0103                BNE =-9
C3DB- A6 62     0104                LDX *OUTBUF          ;LOCATION OF EXP
C3DD- A0 04     0105                LDY #4
C3DF- B5 5F     0106                LDA *OUTBUF-3,X      ;EXPONENT
C3E1- 20 FF C1  0107                JSR OUTCHAR
C3E4- E8        0108                INX
C3E5- 88        0109                DEY
C3E6- D0 F7     0110                BNE =-8
C3E8- 60        0111                RTS
                0112
                0113 ;OUTDG: PRINTS DIGITAL CHANNEL COUNTS
                0114
                0115 OUTDG
C3E9- 85 27     0116                STA *CNTR1           ;NUMBER OF VALUES
C3EB- D0 01     0117                BNE =+2
C3ED- 60        0118                RTS
C3EE- 20 B8 C1  0119                JSR GETB
C3F1- 85 16     0120                STA *BINARY          ;GET 16BIT WORK
C3F3- 20 B8 C1  0121                JSR GETB
C3F6- 8D 17 00  0122                STA BINARY+1
C3F9- 20 63 C2  0123                JSR OUTDOUBLE        ;OUTPUT NUMBER IN BCD
C3FC- A5 22     0124                LDA *COL             ;CHECK COLUMN NUMBER
C3FE- C9 46     0125                CMP #MAXDG
C400- 90 03     0126                BCC =+4
C402- 20 09 C2  0127                JSR NEWLINE          ;START NEW LINE
C405- C6 27     0128                DEC *CNTR1           ;NUMBER LEFT
C407- D0 E5     0129                BNE OUTDG+5          ;OUTPUT NEXT #
```

```
C409- 4C 09 C2               JMP NEWLINE

               ;OUTPL: OUTPUT PULSE CNTERS. THE FIRST NUMBER
               ;IS A 2 BYTE CLOCK CNT FOLLOWED BY 4 BYTE PULSE
               ;COUNTS FOR EACH CHANNEL

               OUTPL
C40C- 85 27                  STA *CNTR1
C40E- D0 01                  BNE =+2
C410- 60                     RTS
C411- 20 B8 C1               JSR GETB
C414- 85 16                  STA *BINARY
C416- 20 B8 C1               JSR GETB
C419- 85 17                  STA *BINARY+1
C41B- 20 63 C2               JSR OUTDOUBLE        ;OUTPUT NUMBER
               @RC
C41E- A2 00                  LDX #0
C420- 20 B8 C1               JSR GETB
C423- 95 16                  STA *BINARY,X
C425- E8                     INX
C426- E0 04                  CPX #4
C428- D0 F6                  BNE @RC+2
C42A- 20 67 C2               JSR OUTQUAD          ;OUTPUT 4 BYTES
C42D- A5 22                  LDA *COL
C42F- C9 41                  CMP #MAXPL
C431- 90 03                  BCC =+4
C433- 20 09 C2               JSR NEWLINE
C436- C6 27                  DEC *CNTR1
C438- D0 E4                  BNE @RC
C43A- 4C 09 C2               JMP NEWLINE

               ;OUTDUMP -- DUMP ENTIRE TAPE FILE CONTENTS IN
               ;HEXADECIMAL FORMAT

               OUTDUMP
C43D- 20 F4 C1               JSR BLANK            ;SPACE BETWEEN BYTES
C440- 20 B8 C1               JSR GETB             ;GET BYTE
C443- 20 D0 C2               JSR OUTHEXZ          ;OUTPUT BYTE WITH ZEROS
C446- A5 22                  LDA *COL
C448- C9 2F                  CMP #MAXDUMP         ;MAX LINE LENGTH
C44A- 90 F1                  BCC OUTDUMP
C44C- 20 09 C2               JSR NEWLINE          ;NEWLINE
C44F- 4C 3D C4               JMP OUTDUMP          ;NEXT BYTE

               ;PUT "R2DATOUT.S"
                             .FI D26 "GPSUBS.S"

0883   36B9-3F3C   GPSUBS.S

               ;PUT "GPSUBS.S"

               ;CLEAR MEMORY. A,X,Y. CLEARS X BYTES
               ;OF MEMORY TO ZERO AT *TOPNT.

               CLRMEM
C452- A9 00                  LDA #$00
C454- A8                     TAY
C455- 91 37                  STA (TOPNT),Y        ;CLEAR MEM LOC
C457- C8                     INY
C458- CA                     DEX
C459- D0 FA                  BNE CLRMEM+3         ;NOT ZERO, CONT CLR
C45B- 60                     RTS

               ;MOVE MEMORY. A,X,Y. MOVE X BYTES
               ;FROM *FMPNT TO *TOPNT. LIMIT=256.

               MOVIND
C45C- A0 00                  LDY #$00
C45E- B1 35                  LDA (FMPNT),Y
C460- 91 37                  STA (TOPNT),Y
C462- C8                     INY
C463- CA                     DEX
```

```
C464- D0 F8                  BNE MOVIND+2
C466- 60                     RTS

               ;INCREMENT MEMORY. X,Y. INCREMENT A
               ;MULTI PREC NUMBER Y BYTES LONG AT X
               ;Z FLAG SET IF RESULT = 0

               INCMEM
C467- F6 00                  INC *$00,X
C469- D0 04                  BNE INCMEM+8
C46B- E8                     INX
C46C- 88                     DEY
C46D- D0 F8                  BNE INCMEM
C46F- 60                     RTS

               ;DECREMENT MEMORY. A,X,Y. DEC MULTI
               ;PREC NUMBER OF Y BYTES AT X.
               ;CARRY CLR IF RESULT=-1

               DECMEM
C470- 38                     SEC
C471- B5 00                  LDA *$00,X
C473- E9 01                  SBC #$01
C475- 95 00                  STA *$00,X
C477- B0 04                  BCS DECMEM+13
C479- E8                     INX
C47A- 88                     DEY
C47B- D0 F3                  BNE DECMEM
C47D- 60                     RTS

               ;ROTATE LEFT. X,Y. ROTATE MULTI PREC
               ;NUMBER LEFT 1 BIT. X IS LOC OF LSB
               ; AND Y IS # OF BYTES.

               ROTATL
C47E- 18                     CLC                    ;ROTATE IN ZERO BIT
               ROTL
C47F- 36 00                  ROL *$00,X
C481- E8                     INX
C482- 88                     DEY
C483- D0 FA                  BNE ROTL
C485- 60                     RTS

               ;ROTATE OR ARITH SHIFT RIGHT. X,Y.
               ;ROTATE BYTE RIGHT 1 BIT. Y IS # OF
               ;BYTES, X IS PNTR TO MSBYTE

               ROTATR
C486- 18                     CLC
C487- 90 06                  BCC ARSHR+6            ;SHIFT IN ZERO BIT
               ARSHR
C489- 36 00                  ROL *$00,X             ;SHIFT IN SIGN BIT
C48B- 08                     PHP
C48C- 76 00                  ROR *$00,X
C48E- 28                     PLP                    ;SIGN BIT IN CARRY
C48F- 76 00                  ROR *$00,X             ;ROTATE THE BYTE
C491- CA                     DEX
C492- 88                     DEY
C493- D0 FA                  BNE ARSHR+6
C495- 60                     RTS

               ;COMPLEMENT MEMORY (1 OR 2). A,X,Y.
               ;COMPLEMENT MULTI PREC BYTE OF Y
               ;BYTES, AT LOC X.

               COMPL1
C496- 18                     CLC                    ;CLEAR CARRY FOR 1'S COMP
C497- 90 01                  BCC COMPLM+1
               COMPLM
C499- 38                     SEC                    ;SET CARRY FOR 2'S COMP
C49A- A9 FF                  LDA #$FF
C49C- 55 00                  EOR *$00,X
```

```
C49E- 69 00             ADC #$00
C4A0- 95 00             STA *$00,X
C4A2- E8                INX
C4A3- 88                DEY
C4A4- D0 F4             BNE COMPLM+1
C4A6- 60                RTS

              ;ADDER. A,X,Y. ADD MULTI PREC NUMS
              ;IN *TOPNT AND *FMPNT. STORE RESULT
              ;IN *TOPNT.

              ADDER
C4A7- A0 00             LDY #$00
C4A9- 18                CLC
C4AA- B1 37             LDA (TOPNT),Y
C4AC- 71 35             ADC (FMPNT),Y
C4AE- 91 37             STA (TOPNT),Y
C4B0- C8                INY
C4B1- CA                DEX
C4B2- D0 F6             BNE ADDER+3
C4B4- 60                RTS

              ;SUBBER. A,X,Y. SUBTRACTS NUM IN
              ;*FMPNT FROM *TOPNT AND LEAVES RESULT
              ;IN *TOPNT.

              SUBBER
C4B5- A0 00             LDY #$00
C4B7- 38                SEC
C4B8- B1 37             LDA (TOPNT),Y
C4BA- F1 35             SBC (FMPNT),Y
C4BC- 91 37             STA (TOPNT),Y
C4BE- C8                INY
C4BF- CA                DEX
C4C0- D0 F6             BNE SUBBER+3
C4C2- 60                RTS

              ;COMPARE MEMORY. A,Y. Y IS LENGTH
              ;OF NUMS, *FMPNT & *TOPNT ARE STRING
              ;PNTRS.  CMPS LAST BYTE FIRST ASSUME
              ;NUMBERS.

              ; TYA
              ; BEQ CMPMEM+7 ;END OF DATA
              ;CMPMEM
              ; DEY
              ; LDA (FMPNT),Y
              ; CMP (TOPNT),Y
              ; BEQ CMPMEM-3 ;IF EQUAL CONTINUE
              ; RTS ;C & Z FLGS SET

              ;PUT "GPSUBS.S"
                        .FI D26 "FPNORM.S"

032F  36B9-39E8  FPNORM.S

              ;PUT "FPNORM.S"

              ;FLOATING POINT NORMALIZATION. A,X,Y.
              ;NORMALIZES FPNUM IN FPACC WITH 4
              ;BYTES IN THE MANTISSA.

              FPNORM
C4C3- A5 3F             LDA *FPMSW
C4C5- 85 3A             STA *TSIGN          ;SAVE SIGN BIT
C4C7- 10 07             BPL @ACZERT
C4C9- A0 04             LDY #$04            ;COMPLMENT IF NEGATIVE
C4CB- A2 3C             LDX #FPLSWE
C4CD- 20 99 C4          JSR COMPLM
              @ACZERT
C4D0- A5 3F             LDA *FPMSW          ;CHECK FOR FPACC == 0
C4D2- 05 3E             ORA *FPNSW
```

```
C4D4- 05 3D     0017            ORA *FPLSW
C4D6- 05 3C     0018            ORA *FPLSWE
C4D8- D0 03     0019            BNE @ACNONZ
C4DA- 85 40     0020            STA *FPACCE          ;FPACC=0 CLR EXP
                0021 @NORMEX
C4DC- 60        0022            RTS                  ;EXIT RESULT = 0
                0023 @ACNONZ
C4DD- 24 3F     0024            BIT *FPMSW
C4DF- 30 0E     0025            BMI @MINUS1
C4E1- 70 15     0026            BVS @ACCSET          ;IF SO, JUSTIFIED
C4E3- A2 3C     0027            LDX #FPLSWE          ;SHIFT LEFT 1 BIT
C4E5- A0 04     0028            LDY #$04
C4E7- 20 7E C4  0029            JSR ROTATL
C4EA- C6 40     0030            DEC *FPACCE          ;ADJUST EXP
C4EC- 4C DD C4  0031            JMP @ACNONZ          ;CONTINUE ROTATING
                0032 @MINUS1
C4EF- A2 3F     0033            LDX #FPMSW           ;THE NUMBER IS -1.0000000
C4F1- A0 03     0034            LDY #3
C4F3- 20 86 C4  0035            JSR ROTATR
C4F6- E6 40     0036            INC *FPACCE
                0037 @ACCSET
C4F8- A5 3A     0038            LDA *TSIGN           ;IS SIGN POS
C4FA- 10 E0     0039            BPL @NORMEX          ;YES, RETURN
C4FC- A0 03     0040            LDY #$03             ;NO, COMPLEMENT AGAIN
C4FE- A2 3D     0041            LDX #FPLSW
C500- 4C 99 C4  0042            JMP COMPLM
                0043
                0044 ;PUT "FPNORM.S"
                0053            .FI D26 "FPMUL.S"

0AFE  36B9-41B7  FPMUL.S

                0001 ;PUT "FPMUL.S"
                0002 ;FLOATING POINT MULTIPLY. A,X,Y.
                0003 ;MULTIPLY NUMBER IN FPACC TO FPOP
                0004 ;AND LEAVES THE RESULT IN FPACC
                0005 ;
                0006 FPMULT
C503- 20 6F C5  0007            JSR CKSIGN           ;SETUP & CHK SIGN
C506- A5 48     0008            LDA *FOPEXP          ;GET FPOP EXP
C508- 38        0009            SEC                  ;ADD 1 FOR COMPEN
C509- 65 40     0010            ADC *FPACCE          ;ADD FPOP EXP
C50B- 85 40     0011            STA *FPACCE          ;SAVE RESULT EXP
C50D- A9 17     0012            LDA #$17             ;SET BIT COUNTER
C50F- 85 39     0013            STA *CNTR            ;STORE BIT CNTR
                0014 MULTIP
C511- A2 3F     0015            LDX #FPMSW           ;SET PNTR TO FPACC MS BYTE
C513- A0 03     0016            LDY #$3              ;SET PRECISION CNTR
C515- 20 86 C4  0017            JSR ROTATR           ;ROTATE FPACC RT
C518- 90 0D     0018            BCC NADOPP           ;CARRY=0,DON'T ADD PARTIAL
                0019 ADOPP
C51A- A2 45     0020            LDX #FOPLSW          ;PNTR TO LS BYTE OF MULTIP
C51C- 86 35     0021            STX *FMPNT           ;STORE POINTER
C51E- A2 4D     0022            LDX #WORK4           ;PNTR TO LS BYTE OF PARTIA
C520- 86 37     0023            STX *TOPNT
C522- A2 03     0024            LDX #$3              ;ONLY 3 BYTES NEED ADDED
C524- 20 A7 C4  0025            JSR ADDER            ;ADD MULTIPLICAND TO PARTI
                0026 NADOPP
C527- A2 4F     0027            LDX #WORK6           ;SET PNTR TO MS BYTE OF PA
C529- A0 04     0028            LDY #$4              ;SET PREC CNTR
C52B- 20 86 C4  0029            JSR ROTATR           ;ROTATE PART PROD RT
C52E- C6 39     0030            DEC *CNTR
C530- D0 DF     0031            BNE MULTIP           ;NOT 0, CONT MULTIPLYING
C532- A2 4F     0032            LDX #WORK6           ;ELSE, SET PNTR TO PART PR
C534- A0 04     0033            LDY #$4              ;SET PREC CNTR
C536- 20 86 C4  0034            JSR ROTATR           ;MAKE ROOM FOR ROUNDING
C539- A2 4C     0035            LDX #WORK3           ;SET PNTR TO 24TH BIT OF P
C53B- A0 04     0036            LDY #$4              ;SET PREC CNTR
C53D- 18        0037            CLC                  ;CLEAR FOR ADDITION
C53E- A9 80     0038            LDA #$80             ;24TH BIT TO RND IF >.5
C540- 24 4F     0039            BIT *WORK6           ;IS MANTISA >.5
C542- 70 01     0040            BVS CKRND            ;IF SO THEN ROUND
```

```
C544- 4A         0041            LSR A               ;24BIT IS RT ONE
                 0042 CKRND
C545- 24 4C      0043            BIT *WORK3          ;NEED RNDING?
C547- F0 0A      0044            BEQ PREXFR          ;IF ZERO SKIP RND
                 0045 CROUND
C549- 75 00      0046            ADC *$00,X          ;ADD WITH CARRY TO PROPAGA
C54B- 95 00      0047            STA *$00,X          ;STORE PART PROD
C54D- A9 00      0048            LDA #$0             ;CLR A FOR NEXT ADD
C54F- E8         0049            INX                 ;INCREMENT INDEX PNTR
C550- 88         0050            DEY                 ;DECREMENT CNTR
C551- D0 F6      0051            BNE CROUND          ;NOT 0, ADD NEXT BYTE
                 0052 PREXFR
C553- A2 3C      0053            LDX #FPLSWE         ;SET PNTR TO FPACC LSW-1
C555- 86 37      0054            STX *TOPNT          ;STORE IN TOPNT
C557- A2 4C      0055            LDX #WORK3          ;SET PNTR TO PART PROD
C559- 86 35      0056            STX *FMPNT          ;STORE IN FMPNT
C55B- A2 04      0057            LDX #$4             ;SET PREC CNTR
                 0058 EXMLDV
C55D- 20 5C C4   0059            JSR MOVIND          ;MOVE PART PROD TO FPACC
C560- 20 C3 C4   0060            JSR FPNORM          ;NORMALIZE RESULT
C563- A5 3B      0061            LDA *SIGNS          ;GET SIGN STORAGE
C565- D0 07      0062            BNE MULTEX          ;IF NOT 0, SIGN IS POS
C567- A2 3D      0063            LDX #FPLSW          ;ELSE, SET PNTR TO FPACC L
C569- A0 03      0064            LDY #$3             ;SET PREC CNTR
C56B- 20 99 C4   0065            JSR COMPLM          ;COMPLEMENT RESULT
                 0066 MULTEX
C56E- 60         0067            RTS                 ;EXIT FPMULT
                 0068 CKSIGN
C56F- A9 00      0069            LDA #$0             ;SET PAGE PORTION OF POINT
C571- 85 38      0070            STA *TOPNT+1        ;STORE IN TOPNT
C573- 85 36      0071            STA *FMPNT+1        ;STORE IN FMPNT
C575- A9 49      0072            LDA #WORK0          ;SET PNTR TO WORK AREA
C577- 85 37      0073            STA *TOPNT          ;STORE IN TOPNT
C579- A2 08      0074            LDX #$8             ;SET PREC CNTR
C57B- 20 52 C4   0075            JSR CLRMEM          ;SET WORK AREA
C57E- A9 41      0076            LDA #MCAND0         ;SET PNTR TO MULTIPLICAND
C580- 85 37      0077            STA *TOPNT          ;STORE IN TOPNT
C582- A2 04      0078            LDX #$4             ;SET PREC CNTR
C584- 20 52 C4   0079            JSR CLRMEM          ;CLEAR MULTIPLICAND STORAG
C587- A9 01      0080            LDA #$1             ;INITIALIZE SIGN INDICATOR
C589- 85 3B      0081            STA *SIGNS          ;BY STORING 1 IN SIGNS
C58B- A5 3F      0082            LDA *FPMSW          ;FETCH FPACC MS BYTE
C58D- 10 09      0083            BPL OPSGNT          ;POSITIVE, CHECK FPOP
                 0084 NEGFPA
C58F- C6 3B      0085            DEC *SIGNS          ;IF NEG, DECREMENT SIGNS
C591- A2 3D      0086            LDX #FPLSW          ;SET PNTR TO FPACC LS BYTE
C593- A0 03      0087            LDY #$3             ;SET PREC CNTR
C595- 20 99 C4   0088            JSR COMPLM          ;MAKE POS FOR MULT
                 0089 OPSGNT
C598- A5 47      0090            LDA *FOPMSW         ;IS FPOP NEG?
C59A- 30 01      0091            BMI NEGOP           ;TES,COMPLEMENT VALUE
C59C- 60         0092            RTS                 ;ELSE RETURN
                 0093 NEGOP
C59D- C6 3B      0094            DEC *SIGNS          ;DECREMENT SIGNS INDICATOR
C59F- A2 45      0095            LDX #FOPLSW         ;SET PNTR TO FPOP LS BYTE
C5A1- A0 03      0096            LDY #$3             ;SET PREC CNTR
C5A3- 4C 99 C4   0097            JMP COMPLM          ;COMPLEMENT FPOP AND RETUR
                 0098 ;PUT "FPMUL.S"
                 0054            .FI D26 "FPDIV.S"

0919  36B9-3FD2  FPDIV.S

                 0001 ;PUT "FPDIV.S"
                 0002 ;FLOATING POINT DIVIDE. A,X,Y.
                 0003 ;DIVIDES NUMBER IN FPACC BY NUMBER
                 0004 ;IN FPOP AND LEAVES RESULT IN FPACC
                 0005 FPDIV
C5A6- 20 6F C5   0006            JSR CKSIGN          ;CLEAR WORK AREA
C5A9- A5 3F      0007            LDA *FPMSW          ;CHCK FOR DIV BY 0
C5AB- F0 22      0008            BEQ DERROR          ;DIV=0,DIV BY 0 ERROR
                 0009 SUBEXP
C5AD- A5 48      0010            LDA *FOPEXP         ;FET DIVIDEND EXPONENT
```

```
C5AF- 38                       SEC                ;SET CARRY
C5B0- E5 40                    SBC *FPACCE        ;SUBTRACT DIVISOR EXP
C5B2- 85 40                    STA *FPACCE        ;SORE IN FPACC EXP
C5B4- E6 40                    INC *FPACCE        ;COMPENSATE FOR DIV ALGORI
                     SETDCT
C5B6- A9 17                    LDA #$17           ;SET BIT CNTR STORAGE
C5B8- 85 39                    STA *CNTR
                     DIVIDE
C5BA- 20 13 C6                 JSR SETSUB         ;SUB DIVISOR FROM DIVIDEND
C5BD- 30 11                    BMI NOGO
C5BF- A2 45                    LDX #FOPLSW        ;SET PNTR TO DIVIDEND
C5C1- 86 37                    STX *TOPNT         ;STORE IN TOPNT
C5C3- A2 49                    LDX #WORK0         ;SET PNTR TO QUOTIENT
C5C5- 86 35                    STX *FMPNT         ;STORE IN FMPNT
C5C7- A2 03                    LDX #$3            ;SET PREC CNTR
C5C9- 20 5C C4                 JSR MOVIND         ;MOV QUOT TO DIVIDEND
C5CC- 38                       SEC                ;SET CARRY
C5CD- B0 02                    BCS QUOROT
                     DERROR
C5CF- 60                       RTS                ;DIV BY 0 IS 0
                     NOGO
C5D0- 18                       CLC                ;NEG RESULT, CLEAR CARRY
                     QUOROT
C5D1- A2 4D                    LDX #WORK4         ;SET PNTR TO QUOTIENT LS B
C5D3- A0 03                    LDY #$3            ;SET PREC CNTR
C5D5- 20 7F C4                 JSR ROTL           ;ROTATE CARRY IN LSB OF QU
C5D8- A2 45                    LDX #FOPLSW        ;SET PNTR TO DIVID LS BYTE
C5DA- A0 03                    LDY #$3            ;SET PREC CNTR
C5DC- 20 7E C4                 JSR ROTATL         ;ROTATE DIVIDEND L
C5DF- C6 39                    DEC *CNTR          ;DEC BIT CNTR
C5E1- D0 D7                    BNE DIVIDE         ;NOT 0,CONT
C5E3- 20 13 C6                 JSR SETSUB         ;DO AGAIN FOR RNDING
C5E6- 30 1E                    BMI DVEXIT         ;-,NO RNDING
C5E8- A9 01                    LDA #$1            ;ELSE ADD 1 TO 23RD BIT
C5EA- 18                       CLC                ;CLEAR CARRY FOR ADD
C5EB- 65 4D                    ADC *WORK4         ;RND OFF LS BYTE OF QUOT
C5ED- 85 4D                    STA *WORK4         ;RESTORE BYTE
C5EF- A9 00                    LDA #$0            ;CLEAR A NOT CARRY
C5F1- 65 4E                    ADC *WORK5         ;ADD CARRY
C5F3- 85 4E                    STA *WORK5         ;STOR RESULT
C5F5- A9 00                    LDA #$0            ;CLEAR A NOT CARRY
C5F7- 65 4F                    ADC *WORK6         ;ADD CARRY TO MS BYTE OF Q
C5F9- 85 4F                    STA *WORK6         ;STORE RESULT
C5FB- 10 09                    BPL DVEXIT
C5FD- A2 4F                    LDX #WORK6
C5FF- A0 03                    LDY #$3            ;SET PREC CNTR
C601- 20 86 C4                 JSR ROTATR         ;CLEAR SIGN BIT CNTR
C604- E6 40                    INC *FPACCE        ;COMPENSATE EXP FOR ROTATE
                     DVEXIT
C606- A2 3C                    LDX #FPLSWE        ;SET POINTER TO FPACC
C608- 86 37                    STX *TOPNT         ;STORE IN TOPNT
C60A- A2 4C                    LDX #WORK3         ;SET PNTR TO QUOTIENT
C60C- 86 35                    STX *FMPNT         ;SOTRE IN FMPNT
C60E- A2 04                    LDX #$4            ;SET PREC CNTR
C610- 4C 5D C5                 JMP EXMLDV         ;MOVE QUOTIENT TO FPACC
                     SETSUB
C613- A2 49                    LDX #WORK0         ;SET PNTR
C615- 86 37                    STX *TOPNT
C617- A2 3D                    LDX #FPLSW         ;SET PNTR TO FPACC
C619- 86 35                    STX *FMPNT
C61B- A2 03                    LDX #$3            ;SET PREC CNTR
C61D- 20 5C C4                 JSR MOVIND         ;MOVE FPACC TO WORK AREA
C620- A2 49                    LDX #WORK0         ;PREP FOR SUBT
C622- 86 37                    STX *TOPNT         ;STORE PNTR TO DIVISOR
C624- A2 45                    LDX #FOPLSW        ;SET PNTR TO FPOP LS BYTE-
C626- 86 35                    STX *FMPNT         ;STORE PNTR TO DIVIDEND
C628- A0 00                    LDY #$0            ;INITIALIZE INDEX PNTR
C62A- A2 03                    LDX #3             ;SET PREC CNTR
C62C- 38                       SEC
                     SUBR1
C62D- B1 35                    LDA (FMPNT),Y      ;FETCH FPOP
C62F- F1 37                    SBC (TOPNT),Y      ;SUB FPACC BYTE
```

```
C631- 91 37     0083            STA (TOPNT),Y       ;STORE IN PLACE OF DIVISOR
C633- C8        0084            INY
C634- CA        0085            DEX
C635- D0 F6     0086            BNE SUBR1           ;NOT 0 CONT SUBT
C637- A5 4B     0087            LDA *WORK2          ;SET SIGN BIT RESULT IN N
C639- 60        0088            RTS
                0089 ;
                0090 ;PUT "FPDIV.S"
                0055            .FI D26 "FPIOSUBS.S"

04E1  36B9-3B9A  FPIOSUBS.S

                0001 ;PUT "FPIOSUBS.S"
                0002 ;SUBROUTINES USED BY THE FPIN AND FPOUT ROUTINES
                0003 ;
                0004 FPX10
C63A- A9 04     0005            LDA #$04
C63C- 85 48     0006            STA *FOPEXP         ;LOAD FPOP WITH TEN
C63E- A9 50     0007            LDA #$50            ;BY SETTING EXP TO 4
C640- 85 47     0008            STA *FOPMSW         ;AND THE MANTISSA TO $50,$
C642- A9 00     0009            LDA #$00
C644- 85 46     0010            STA *FOPNSW
C646- 85 45     0011            STA *FOPLSW
C648- 20 03 C5  0012            JSR FPMULT          ;MULT FPACC BY FPOP
C64B- C6 5C     0013            DEC *IOEXPD         ;DECR DEC EXP
C64D- 60        0014            RTS                 ;RETURN TO TEST FOR COMPLE
                0015 FPD10
C64E- A9 FD     0016            LDA #$FD            ;PLACE .1 IN FPOP BY
C650- 85 48     0017            STA *FOPEXP         ;SETTING FPOP EXP TO -3
C652- A9 66     0018            LDA #$66            ;AND LOADING NANTISSA WITH
C654- 85 47     0019            STA *FOPMSW
C656- 85 46     0020            STA *FOPNSW
C658- A9 67     0021            LDA #$67
C65A- 85 45     0022            STA *FOPLSW
C65C- 20 03 C5  0023            JSR FPMULT          ;MULT FPACC BY FPOP
C65F- E6 5C     0024            INC *IOEXPD
C661- 60        0025            RTS
                0026 DECBIN
C662- A9 00     0027            LDA #$00
C664- 85 5B     0028            STA *IOSTR3         ;CLEAR MS BYTE+1 OF RESULT
C666- A2 54     0029            LDX #IOLSW          ;SET PNTR TO I/O STOR
C668- 86 37     0030            STX *TOPNT
C66A- A2 58     0031            LDX #IOSTR          ;SET PNTR TO I/O STOR
C66C- 86 35     0032            STX *FMPNT
C66E- A2 04     0033            LDX #$04            ;SET PREC CNTR
C670- 20 5C C4  0034            JSR MOVIND          ;MOVE I/O STOR TO WORK ARE
C673- A2 58     0035            LDX #IOSTR          ;SET PNTR TO ORIG VAL
C675- A0 04     0036            LDY #$04            ;SET PREC CNTR
C677- 20 7E C4  0037            JSR ROTATL          ;START *10 ROUT
C67A- A2 58     0038            LDX #IOSTR          ;RESET PNTR
C67C- A0 04     0039            LDY #$04            ;SET PREC CNTR
C67E- 20 7E C4  0040            JSR ROTATL          ;MULT BY 2 AGAIN
C681- A2 54     0041            LDX #IOLSW          ;SET PNTR TO I/O WORK AREA
C683- 86 35     0042            STX *FMPNT          ;STORE IN FMPNT
C685- A2 58     0043            LDX #IOSTR          ;SET PNTR TO I/O STOR
C687- 86 37     0044            STX *TOPNT          ;STORE IN TOPNT
C689- A2 04     0045            LDX #$04            ;SET PREC CNTR
C68B- 20 A7 C4  0046            JSR ADDER           ;ADD ORIGINAL TO ROTATED
C68E- A2 58     0047            LDX #IOSTR          ;RESET PNTR
C690- A0 04     0048            LDY #$04            ;SET PREC CNTR
C692- 4C 7E C4  0049            JMP ROTATL
                0050 ;
                0051 ;PUT "FPIOSUBS.S"
                0056            .FI D26 "FPBUFOUT.S"

0CB1  36B9-436A  FPBUFOUT.S

                0001 ;PUT "FPBUFOUT.S"
                0002 ;FLOATING POINT OUTPUT ROUTINE
                0003 ;CONVERTS THE FLOATING POINT BINARY
                0004 ;NUMBER IN FPACC TO ITS DECIMAL
                0005 ;EQUIVALENT AND OUTPUTS IT TO THE
```

```
                   0006 ;DISPLAY DEVICE AS ASCII CHARACTERS
                   0007 ;IN THE FOLLOWING FORMAT
                   0008 ;  0.1234567 E+07
                   0009 ;
                   0010 FPOUT
C695- A9 62        0011                LDA #OUTBUF          ;CLEAR OUTPUT BUFFER
C697- 85 37        0012                STA *TOPNT
C699- A9 00        0013                LDA #$00
C69B- 85 38        0014                STA *TOPNT+1
C69D- 85 36        0015                STA *FMPNT+1
C69F- 85 5C        0016                STA *IOEXPD          ;CLEAR DEC EXP STOR
C6A1- A2 16        0017                LDX #$16
C6A3- 20 52 C4     0018                JSR CLRMEM
C6A6- A5 3F        0019                LDA *FPMSW           ;IS NEG
C6A8- 30 04        0020                BMI OUTNEG           ;YES,MAKE POS AND OUT "-"
C6AA- A9 2B        0021                LDA #'+              ;ELSE, SET ASCII FOR "+"
C6AC- D0 09        0022                BNE AHEAD1           ;DISPLAY +
                   0023 OUTNEG
C6AE- A2 3D        0024                LDX #FPLSW           ;SET PNTR TO LS BYTE OF FP
C6B0- A0 03        0025                LDY #$3              ;SET PREC CNTR
C6B2- 20 99 C4     0026                JSR COMPLM           ;MAKE FPACC POS
C6B5- A9 2D        0027                LDA #'-              ;SET ASCII FOR "-"
                   0028 AHEAD1
C6B7- 20 65 C7     0029                JSR BUFECHO          ;OUT SIGN OF RESULT
C6BA- A9 30        0030                LDA #'0              ;SET UP ASCII 0
C6BC- 20 65 C7     0031                JSR BUFECHO          ;OUT 0
C6BF- A9 2E        0032                LDA #'.              ;SET UP ASCII "."
C6C1- 20 65 C7     0033                JSR BUFECHO          ;OUT .
C6C4- C6 40        0034                DEC *FPACCE
                   0035 DECEXT
C6C6- 10 0F        0036                BPL DECEXD           ;IF COMPENSATED,EXP>=0
C6C8- A9 04        0037                LDA #$4              ;EXP NEG,ADD 4 TO FPACC
C6CA- 18           0038                CLC
C6CB- 65 40        0039                ADC *FPACCE          ;ADD 4 TO FPACC EXP
C6CD- 10 0E        0040                BPL DECOUT           ;IF EXP>=0,OUT MANTISSA
C6CF- 20 3A C6     0041                JSR FPX10            ;ELSE, MULT MANTISSA BY 10
                   0042 DECREP
C6D2- A5 40        0043                LDA *FPACCE          ;GET EXP
C6D4- 4C C6 C6     0044                JMP DECEXT           ;REPEAT TEST FOR >=0
                   0045 DECEXD
C6D7- 20 4E C6     0046                JSR FPD10            ;MULT FPACC BY .1
C6DA- 4C D2 C6     0047                JMP DECREP           ;CHK STATUS OF FPACC EXP
                   0048 DECOUT
C6DD- A2 58        0049                LDX #IOSTR           ;SET UP FOR MOVE OPER
C6DF- 86 37        0050                STX *TOPNT           ;SET TOPNT TO WORK REG
C6E1- A2 3D        0051                LDX #FPLSW           ;SET PNTR TO FPACC LS BYTE
C6E3- 86 35        0052                STX *FMPNT
C6E5- A2 03        0053                LDX #$3              ;SET PREC CNTR
C6E7- 20 5C C4     0054                JSR MOVIND           ; MOVE FPACC TO OUT REGS
C6EA- A9 00        0055                LDA #$0
C6EC- 85 5B        0056                STA *IOSTR3          ;CLEAR OUT REG MS BYTE+1
C6EE- A2 58        0057                LDX #IOSTR           ;SET PNTR TO OUT LS BYTE
C6F0- A0 03        0058                LDY #$3              ;SET PREC CNTR
C6F2- 20 7E C4     0059                JSR ROTATL           ;ROT TO COMP FOR SIGN BIT
C6F5- 20 62 C6     0060                JSR DECBIN           ;OUT REG X10,OVERFLOW
                   0061 COMPEN
C6F8- E6 40        0062                INC *FPACCE          ;INCR FPACC EXP
C6FA- F0 0A        0063                BEQ OUTDIG           ;OUT DIGIT
C6FC- A2 5B        0064                LDX #IOSTR3          ;ELSE,ROTATE R TO COMP
C6FE- A0 04        0065                LDY #$4              ;FOR ANY REMAINDER IN BIN
C700- 20 86 C4     0066                JSR ROTATR           ;ROT R
C703- 4C F8 C6     0067                JMP COMPEN           ;REPEAT LOOP UNTIL EXP=0
                   0068 OUTDIG
C706- A9 07        0069                LDA #$7              ;SET DIGIT CNTR
C708- 85 39        0070                STA *CNTR
C70A- A5 5B        0071                LDA *IOSTR3          ;FETCH BCD,SEE IF 1ST DIG
C70C- F0 11        0072                BEQ ZERODG           ;YES, CHECK REM OF DIGITS
                   0073 OUTDGS
C70E- A5 5B        0074                LDA *IOSTR3          ;GET BCD FROM OUT REG
C710- 09 30        0075                ORA #'0              ;FORM ASCII CODE
C712- 20 65 C7     0076                JSR BUFECHO          ;OUT DIGIT
                   0077 DECRDG
```

```
C715- C6 39                    DEC *CNTR           ;DEC DIGIT CNTR
C717- F0 1A                    BEQ EXPOUT          ;=0,DONE OUT EXP
C719- 20 62 C6                 JSR DECBIN          ;ELSE,GET NEXT DIGIT
C71C- 4C 0E C7                 JMP OUTDGS          ;FORM ASCII AND OUT
                ZERODG
C71F- C6 5C                    DEC *IOEXPD         ;DEC EXP FOR SKIPPING DISP
C721- A5 5A                    LDA *IOSTR2         ;CHECK IF MANTISSA =0
C723- D0 F0                    BNE DECRDG          ;IF NOT 0,CONT OUT
C725- A5 59                    LDA *IOSTR1
C727- D0 EC                    BNE DECRDG
C729- A5 58                    LDA *IOSTR
C72B- D0 E8                    BNE DECRDG
C72D- A9 00                    LDA #$0             ;MANTISSA 0,CLEAR EXP
C72F- 85 5C                    STA *IOEXPD
C731- F0 E2                    BEQ DECRDG          ;BEFORE FINISHING DISPLAY
                EXPOUT
C733- A9 45                    LDA #'E             ;SET ASCII FOR E
C735- 20 65 C7                 JSR BUFECHO         ;DISPLAY E
C738- A5 5C                    LDA *IOEXPD         ;TEST IF NEG
C73A- 30 05                    BMI EXOUTN          ;YES, DISPLAY - AND NEGATE
C73C- A9 2B                    LDA #'+             ;NO, SET ASCII FOR +
C73E- 4C 49 C7                 JMP AHEAD2          ;DISPLAY EXP VAL
                EXOUTN
C741- 49 FF                    EOR #$FF            ;TWO'S COMP EXP
C743- 85 5C                    STA *IOEXPD         ;TO MAKE NEG POS
C745- E6 5C                    INC *IOEXPD         ;FOR OUT OF EXP VALUE
C747- A9 2D                    LDA #'-             ;SET ASCII FOR -
                AHEAD2
C749- 20 65 C7                 JSR BUFECHO         ;OUT SIGN OF EXP
C74C- A0 00                    LDY #$0
C74E- A5 5C                    LDA *IOEXPD         ;FETCH EXP
                SUB12
C750- 38                       SEC
C751- E9 0A                    SBC #$0A            ;SUB TEN'S FROM EXP
C753- 30 06                    BMI TOMUCH          ;IF MINUS,READY FOR OUT
C755- 85 5C                    STA *IOEXPD         ;RESTORE POS RESULT
C757- C8                       INY                 ;INC TEN'S CNTR
C758- 4C 50 C7                 JMP SUB12           ;CONT SUB
                TOMUCH
C75B- 98                       TYA                 ;PUT MS DIGIT INTO A
C75C- 09 30                    ORA #'0             ;FORM ASCII
C75E- 20 65 C7                 JSR BUFECHO         ;OUT TEN'S DIG TO DISPLAY
C761- A5 5C                    LDA *IOEXPD         ;FETCH UNIT'S DIGIT
C763- 09 30                    ORA #'0             ;FORM ASCII
                ;
                ;OUTBUF[0] IS A POINTER TO THE LAST CHAR IN OUTBUF
                ;OUTBUF[1] TO OUTBUF[15] CONTAINS THE CHARS.
                BUFECHO
C765- E6 62                    INC *OUTBUF         ;POINT TO NEXT FREE SPOT
C767- A6 62                    LDX *OUTBUF
C769- 95 62                    STA *OUTBUF,X       ;STORE CHAR
C76B- 60                       RTS
                ;
                ;PUT "FPBUFOUT.S"
                               .FI "R2DATA.D"

0270  36B9-3929  R2DATA.D

                ;PUT "R2DATA.D"

                MESSAGE
C76C- 78 77 7C                 .BY $78 $77 $7C $38 $79 $53  ;TABLE?
C76F- 38 79 53

                MESGS
C772- 45 4E 44                 .BY 'END' $00
C775- 00
C776- 45 4F 54                 .BY 'EOT' $00
C779- 00

                CONFIGTAB
C77A- FF FF FF                 .BY $FF $FF $FF $FF $FF $FF $FF $FF
```

```
C77D- FF FF FF
C780- FF FF
C782- 00 00 11  0012           .BY $00 $00 $11 $00 $00 $01 $FF $FF ;UNIX
C785- 00 00 01
C788- FF FF
C78A- 06 05 00  0013           .BY $06 $05 $00 $00 $80 $01 $FF $FF ;APPLE
C78D- 00 80 01
C790- FF FF
C792- 06 10 00  0014           .BY $06 $10 $00 $00 $80 $01 $FF $FF ;APPLE
C795- 00 80 01
C798- FF FF
C79A- 06 38 00  0015           .BY $06 $38 $00 $00 $80 $01 $FF $FF ;APPLE
C79D- 00 80 01
C7A0- FF FF
C7A2- 10 10 00  0016           .BY $10 $10 $00 $00 $80 $01 $FF $FF ;TRS-80
C7A5- 00 80 01
C7A8- FF FF
C7AA- FF FF FF  0017           .BY $FF $FF $FF $FF $FF $FF $FF $FF
C7AD- FF FF FF
C7B0- FF FF
C7B2- FF FF FF  0018           .BY $FF $FF $FF $FF $FF $FF $FF $FF
C7B5- FF FF FF
C7B8- FF FF
C7BA- FF FF FF  0019           .BY $FF $FF $FF $FF $FF $FF $FF $FF
C7BD- FF FF FF
C7C0- FF FF
C7C2- FF FF FF  0020           .BY $FF $FF $FF $FF $FF $FF $FF $FF
C7C5- FF FF FF
C7C8- FF FF
C7CA- FF FF FF  0021           .BY $FF $FF $FF $FF $FF $FF $FF $FF
C7CD- FF FF FF
C7D0- FF FF
                0022 TABLENGTH .DE =-CONFIGTAB
                0023
                0024 ;PUT "R2DATA.D"
                0058           .EN
END OF MAE PASS!

 ---  LABEL FILE:  ---

@ACCSET =C4F8            @ACNONZ =C4DD            @ACZERT =C4D0
@MINUS1 =C4EF            @NORMEX =C4DC            @RA =C151
@RB =C182                @RC =C41E                @RD =C286
@RE =C3B6                @RI =C073                @SYNC2 =C251
ACCESS =8B86             ADCHAN =0024             ADDER =C4A7
ADOPP =C51A              AHEAD1 =C6B7             AHEAD2 =C749
ARSHR =C489              ASCNIB =8275             AVGFACT =C2F6
BA =C000                 BAUD =0032               BCD =001A
BINARY =0016             BLANK =C1F4              BLKMOV =8740
BUFECHO =C765            CALCLENGTH =C122         CHARDEL =002D
CKRND =C545              CKSIGN =C56F             CLRMEM =C452
CNTR =0039               CNTR1 =0027              CNTR2 =0028
COL =0022                COMPEN =C6F8             COMPL1 =C496
COMPLM =C499             CONFIG =89A5             CONFIGTAB =C77A
CONVERT =C271            COUNT =002A              CROUND =C549
CURRENT =0004            DATA =0000               DATASTART =0002
DECBIN =C662             DECEXD =C6D7             DECEXT =C6C6
DECMEM =C470             DECOUT =C6DD             DECRDG =C715
DECREP =C6D2             DELAY =C226              DERROR =C5CF
DGCHAN =0025             DISBUF =A640             DIVIDE =C5BA
DIVTMP =0012             DLOOP1 =002B             DLOOP2 =002C
DVEXIT =C606             EMPTY =C114              END =0002
EOF =C1CF                EOR =C209                ERROR =0011
EXMLDV =C55D             EXOUTN =C741             EXPOUT =C733
FILL3 =8718              FIND =C0E4               FINDFIRST =C145
FMPNT =0035              FOLSWE =0044             FOPEXP =0048
FOPLSW =0045             FOPMSW =0047             FOPNSW =0046
FOUND =C0FC              FPACCE =0040             FPD10 =C64E
FPDIV =C5A6              FPLSW =003D              FPLSWE =003C
FPMSW =003F              FPMULT =C503             FPNORM =C4C3
FPNSW =003E              FPOUT =C695              FPRINT =C3C9
FPX10 =C63A              GETABLE =C02B            GETB =C1B8
```

```
GOVEC =A659             ID =A64E                INBYTE =81D9
INCHR =8A1B             INCMEM =C467            INEXPS =0052
INIT =C024              INITZP =C040            INMTAS =0051
INPRDI =0053            INTVEC =A678            IOEXP =0057
IOEXPD =005C            IOLSW =0054             IOMSW =0056
IONSW =0055             IOSTR =0058             IOSTR1 =0059
IOSTR2 =005A            IOSTR3 =005B            KSCONF =89A3
L2 =84D3                LABEL =0010             LENGTH =0008
LINEDEL =002E           LINEFEED =0030          LOAD =C0CE
LSTDAY =000A            MAIN =C084              MAINDUMP =C0B1
MAINENTRY =0033         MAINLOOP =C093          MAXAD =0041
MAXDG =0046             MAXDUMP =002F           MAXPL =0041
MCAND0 =0041            MCAND1 =0042            MCAND2 =0043
MESGS =C772             MESSAGE =C76C           MONITOR =8003
MOVIND =C45C            MULTEX =C56E            MULTIP =C511
NACCESS =8B9C           NADOPP =C527            NBCD =0020
NBINARY =001F           NEGFPA =C58F            NEGOP =C59D
NEWLINE =C209           NEXT =C04C              NEXTRECORD =C19C
NIBASC =C2E9            NODAT =00EE             NOGO =C5D0
OPSGNT =C598            OUTAD =C395             OUTBIN =C26B
OUTBUF =0062            OUTBYT =82FA            OUTCHAR =C1FF
OUTCHR =8A47            OUTDATE =C366           OUTDG =C3E9
OUTDGS =C70E            OUTDIG =C706            OUTDOUBLE =C263
OUTDTB =C38A            OUTDUMP =C43D           OUTHEAD =C346
OUTHEX =C2D8            OUTHEX2 =C2E5           OUTHEXZ =C2D0
OUTLAB =C325            OUTNEG =C6AE            OUTNIB =8A44
OUTNUM =C2B9            OUTPL =C40C             OUTQUAD =C267
OUTXAH =82F4            PAGES =000E             PARM =8220
PARN =A64A              PARNR =A649             PEOT =C0C0
PLCHAN =0026            PREXFR =C553            PROMPT =002F
QUOROT =C5D1            RESALL =81C4            RESTART =8000
RMDIG =A645             ROTATL =C47E            ROTATR =C486
ROTL =C47F              RTS =C239               SAVE2 =87EA
SAVER =8188             SCAND =8906             SDBYT =A651
SETDAT =C107            SETDCT =C5B6            SETSUB =C613
SIGNS =003B             SPACE =8342             STACK =0029
START =0006             SUB12 =C750             SUBBER =C4B5
SUBEXP =C5AD            SUBR1 =C62D             SYNC =C246
SYNCFLAG =0031          TABLENGTH =0058         TAPDEL =A630
TECHO =A653             TEMP1 =0061             TESTREC =C167
TOMUCH =C75B            TOPNT =0037             TPEXP =0060
TPLSW =005D             TPMSW =005F             TPNSW =005E
TSIGN =003A             TSTAT =8B3C             USRENTRY =C000
VECSW =8BB7             VIA1 =A000              VIA2 =A800
VIA3 =AC00              WAIT =C23A              WMESG =C1E7
WORK0 =0049             WORK1 =004A             WORK2 =004B
WORK3 =004C             WORK4 =004D             WORK5 =004E
WORK6 =004F             WORK7 =0050             ZERO =0021
ZERODG =C71F
//0000,C7D2,17D2
```

APPENDIX D. Apple II Reader Program

```
80  DIM X$(300)
90 D$ =  CHR$ (4)
100  PRINT : PRINT "READY":CNT = 0
110  PRINT D$;"IN#1": PRINT D$;"PR#1": REM   RS232 INPUT
120  INPUT A$: PRINT " ": REM SYNC
130  PRINT D$;"PR#0"
140  INPUT X$(CNT)
150  IF X$(CNT) = "END" THEN 300
160 CNT = CNT + 1
170  GOTO 140
300  PRINT D$;"IN#0"
310  FOR I = 0 TO CNT
320  PRINT X$(I)
330  NEXT I
340  GOTO 100
350  END
```

APPENDIX E. Component Data Sheets

SDM856 Data Acquisition System

3606 Programmable Gain Amplifier

MPC8S Multiplexor

MSM5832 Clock

HTAA-16W Power Supply

BURR-BROWN

SDM856
SDM857

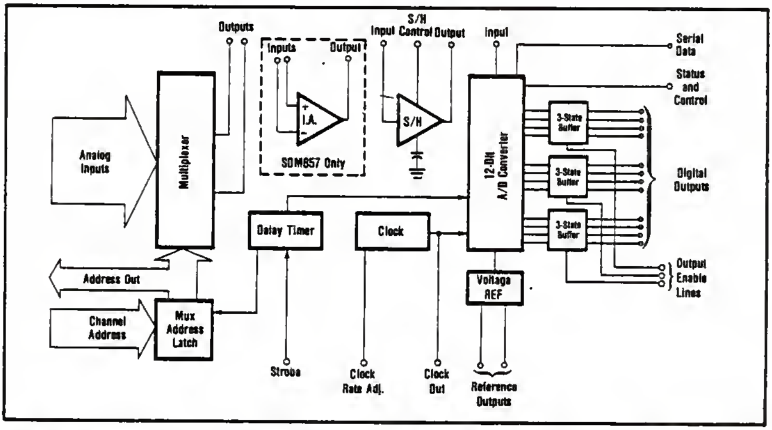
HYBRID DATA ACQUISITION SYSTEM

FEATURES

- MINIATURE SIZE
- LOW COST
- 12-BIT, ±0.012% LINEARITY ERROR
- INSTRUMENT AMP OPTION
- LOW LEVEL INPUTS (SDM857)
- SELECTABLE 16 SINGLE, 8 DIFFERENTIAL INPUTS
- THREE-STATE OUTPUT BUFFERS
- THROUGHPUT RATES (SDM857 Overlap Mode)
  8-Bit Accuracy: 70kHz
  10-Bit Accuracy: 32kHz
  12-Bit Accuracy: 29kHz

DESCRIPTION

The SDM856 and SDM857 are complete data acquisition systems contained in a miniature 2.2" x 1.7" x 0.22" ceramic package. These systems offer all the functions available in large modular data acquisition systems and are available with an optional internal instrumentation amplifier (SDM857). Inputs as low as ±10mV can be accepted by the SDM857; thermocouples, strain gages, and other low level signal sensors don't require external signal conditioning. Both models are fully expandable from the basic 16 channel single-ended or 8 channel differential input capability. Digital resolution is 12 bits with accuracy of ±0.024% at a throughput rate of 29kHz (SDM856KG).



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

Printed in U.S.A. October, 1978

TABLE OF CONTENTS

SYSTEM DESCRIPTION

SDM857 contains all components necessary to multiplex and convert analog signals as low as ±10mV and as high as ±5V into equivalent digital outputs. Throughput sampling rates are from 29kHz (12-bit resolution) to 70kHz (8-bit resolution) in the overlap mode of operation. A complete low drift instrumentation amplifier allows selection of gains from 2 to 500 with one external resistor. SDM856 is identical to SDM857, but does not include the instrumentation amplifier. Tbis provides the option of adding an external instrumentation amplifier for specific requirements such as high speed, digital programming, etc. Both models can be configured to accept either 8-channel differential or 16-channel single-ended signals and can be expanded almost without limit with external multiplexers. Three-state outputs are provided for easy interface to microprocessor and other bus-structured systems. Figure 1 illustrates all system components which are described in the following paragraphs.

ANALDG MULTIPLEXER

The analog multiplexer consists of two CMOS integrated circuits. Pin interconnects are used to select 16-channel single-ended or 8-channel differential operation. In single-ended operation the multiplexer can be used in a pseudo-differential mode by connecting the amplifier inverting input to common remote signal ground. Channel selection is made by an internally latched 3- or 4-bit binary word, for differential or single-ended operation respectively.

INSTRUMENTATION AMPLIFIER (SDM857 only)

Offering low drift and high accuracy, the internal instrumentation amplifier may be programmed by a single external resistor for gains from 2 to 500. With gain programming pins open, the gain is 2.

SAMPLE/HOLD

A complete stand-alone circuit, the sample/hold amplifier features buffered output, 10μsec acquisition time, and 100nsec aperture time.

Input, output, and mode control lines are brought out to separate pins. This allows maximum system flexibility for performing functions, such as automatic gain ranging, with no loss of aperture time.

ANALOG-TO-DIGITAL CONVERTER

The ADC is a 12-bit, 25μsec converter with 0.01% linearity error. Its features include positive and negative reference voltage outputs, external gain and offset adjustments, straight binary or two's complement output, serial data and clock outputs, status output, a short cycle feature, and a clock rate control for higher throughput rates at lower resolution or accuracy.

THREE-STATE OUTPUT BUFFERS

Digital outputs of the ADC are internally buffered by LSTTL three-state buffers. Three separate enable lines are brought out for easy interfacing to 4-, 8- or 16-bit data buses. $\overline{\text{MSB}}$ and BUSY are also buffered by separate three-state devices, each with its own enable line.

ADDRESS LATCH

Outputs of the 4-bit TTL register latch are connected to the address inputs of the multiplexer. This latch serves as an address storage register for the selected analog input. It may be loaded through 4 address inputs. Other inputs are LOAD and $\overline{\text{CLEAR}}$. The 3 least significant bits are used for 8-channel differential mode addressing.

DELAY TIMER

A delay timer allows settling time for the multiplexer, amplifier, and sample/hold circuits before conversion begins. The delay is adjustable over a wide range by use of an external resistor or capacitor. This allows for longer settling time of the instrumentation amplifier when operating at high gains, or shorter settling time for lower resolution operation.

CHANNEL EXPANSION

The number of analog input channels of the SDM856 and SDM857 can be easily increased by using Burr-Brown's MPC8D (8-channel differential) and MPC16S (16-channel single-ended) multiplexers. These are latch-free devices which contain internal binary decoding at TTL or MOS levels and may be integrated into a system with minimal external logic.

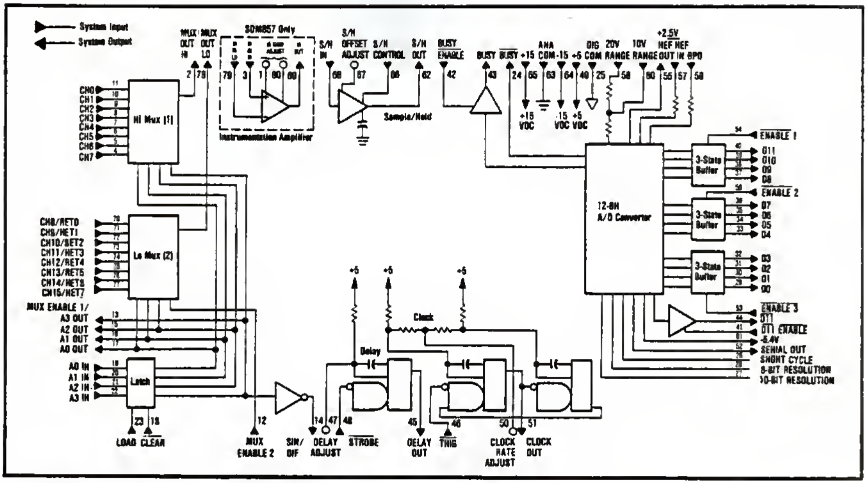


FIGURE 1. SDM856/857 Block Diagram.

SYSTEM PERFORMANCE

SDM856 and SDM857 are configured for random channel selection. With the addition of an external counter they can be configured to continuously sequence through all analog channels or sequence through all analog channels on command from an external trigger.

With the appropriate 4-bit (single-ended) or 3-bit (differential) channel address on the latcb inputs, and DELAY OUT (pin 45) tied to the LOAD input (pin 23), a negative going edge is applied to the STROBE input (pin 48). This starts the delay timer, latcbes the multiplexer address, and allows the input signal to pass through the multiplexer, instrumentation amplifier and sample/hold and settle to its final value before starting the A/D conversion. The DELAY OUT signal (pin 45) is also connected to the TRIG input (pin 46) and the A/D conversion is initiated on the negative-going edge. The S/H CONTROL input (pin 66) is connected to BUSY (pin 24) so that the sample/hold is in the HOLD mode during the A/D conversion.

By using overlap programming the settling time effects of the analog multiplexer and instrumentation amplifier can be reduced, extending throughput sampling rates up to 29kHz for 12-bit and 70kHz for 8-bit resolution (ADC short-cycled). This mode of operation is most useful when converting low level inputs to accommodate the increased settling time of the instrumentation amplifier. Overlap programming is accomplished by connecting BUSY to STROBE and S/H CONTROL; DELAY OUT to LOAD and TRIG. In this mode of operation the address of the next channel to be converted is latched and the output of the instrumentation amplifier allowed to settle to a new value during the present conversion.

DIGITAL INPUT SPECIFICATIONS

Address Inputs (A0 - A3)	One standard LSTTL load, positive true
Address Coding	4-bit binary
LOAD	One staodard LSTTL load, positive true, address loaded on positive edge.
CLEAR	One standard LSTTL load, negative true, low level clears address latch.
STROBE	One standard LSTTL load, high-to-low transition triggers the delay timer.
TRIG	One standard LSTTL load, a negative going edge initiates the A/D conversion.
SHORT CYCLE	One standard LSTTL load, logic 1 for 12-bit resolution. Connect to "8-bit" or "10-bit" for 8- or 10-bit resolution.
ENABLE 1, ENABLE 2, ENABLE 3 D11 ENABLE BUSY ENABLE	One standard LSTTL load, a low level enables the 3-state output.
S/H CONTROL	TTL compatible, 10μA maximum input current. Logic 0 = Hold mode, Logic 1 = Sample (track) mode.
MUX ENABLE 2	TTL compatible, 2μA input current. Logic 0 enables multiplexer 2 (channels 8-15).

DIGITAL OUTPUT SPECIFICATIONS

Parallel Data Outputs	5 standard TTL loads, positive true, 3-state.
Serial Output	2 standard TTL loads, positive true, NRZ, time serial data output beginning with O11 (see Timing Diagram).
D11	5 standard TTL loads, positive true, 3-state.
BUSY	5 standard TTL loads, low during A/D conversion.
BUSY	5 standard TTL loads, high during A/O conversion, 3-state
CLOCK OUT	5 standard TTL loads, for synchronizing serial oui data (see Timing Oiagram).
Address Outputs (A0 - A3)	5 standard TTL loads, positive true
OELAY OUT	5 standard TTL loads, high during delay period, triggered by Strobe input.
SIN/DIF	5 standard TTL loads, high while addressing channels 0-7, low while addressing channels 8-15.

SPECIFICATIONS

ELECTRICAL

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

MODEL	SDM856/SDM857			
TRANSFER CHARACTERISTICS	**MIN**	**TYP**	**MAX**	**UNITS**
Resolution	12			Bits
Number of Analog Channels	16SIN/8DIF			
Throughput Rate (Normal mode)				
SDM856JG	33	35		kHz
SDM856KG	25	27		kHz
SDM857JG	22	24		kHz
SDM857KG	18	20		kHz
Throughput Rate (Overlap mode)				
SDM856JG	38	40		kHz
SDM856KG	27	29		kHz
SDM857JG	38	40		kHz
SDM857KG	27	29		kHz
ANALOG INPUTS				
ADC Input Voltage Ranges	0 to +10, ±5, ±10			V
Mux Input Voltage Range				
Absolute max without damage			±20	V
For linear operation			±6	V
Mux Input Impedance, OFF Channel		$5 \times 10^9 \| 10$		Ω‖pF
Mux Input Impedance, ON Channel		1800‖7		Ω‖pF
Amplifier Characteristies (SDM857 only)				
Input Impedance		$5 \times 10^9 \| 3$		Ω‖pF
Gain Range	2		500	
Gain Equation	G = 2 + (20kΩ/R_{EXT}[1])			
Input Bias Current at +25°C			±50	nA
0°C to +70°C		±1.1		nA/°C
Offset Current at +25°C			±20	nA
0°C to +70°C		±0.6		nA/°C
Input Offset Voltage		±0.1		mV
Input Offset Voltage Drift (G > 100)		±4	±6	μV/°C
Output Noise (10Hz - 10kHz)				
G = 100, R_S = 500Ω		400		μV, rms
Common-mode Rejection (DC)				
G = 2		90		dB
G = 1000		97		dB
Sample/Hold DC Characteristics				
Input Impedance		$10^{10} \| 3$		Ω‖pF
Bias Current		50		nA
Output Offset Voltage		7		mV
REFERENCE VOLTAGES				
Positive Output	+2.490	+2.500	+2.510	V
Positive Output Drift		±5	±10	ppm/°C
Negative Output	-6.0	-6.4	-6.8	V
Negative Output Drift		±5	±10	ppm/°C
ACCURACY				
Throughput Accuracy				
0 to +5V, ±5V ranges JG			±0.048	% of FSR[2]
0 to +5V, ±5V ranges KG			±0.024	% of FSR
0 to +20mV, ±10mV JG (SDM857 only)			±0.11	% of FSR
0 to +20mV, ±10mV KG (SDM857 only)			±0.08	% of FSR
Linearity (G = 1)				
JG			±0.024	% of FSR
KG			±0.012	% of FSR
Differential Linearity (G = 1)				
JG		±0.024	±0.048	% of FSR
KG		±0.012	±0.024	% of FSR
Quantizing Error			±0.012	% of FSR
System Gain Error[3]		±0.1	±0.3	%
System Offset Error[3]		±0.1	±0.3	% of FSR
Power Supply Sensitivity +15V		±0.0007		%/%ΔV
Power Supply Sensitivity -15V		±0.0007		%/%ΔV
Power Supply Sensitivity +5V		±0.001		%/%ΔV

TEMPERATURE STABILITY	MIN	TYP	MAX	UNITS
System Accuracy Drift[4] Unipolar			±25	ppm/°C
System Accuracy Drift[4] Bipolar			±20	ppm/°C
Linearity Drift			±2	ppm of FSR/°C
DYNAMIC ACCURACY				
Sample/Hold Characteristics				
Aperture Time		100		nsec
Acquisition time		10		μsec
Feedthrough (10V step)		±1.4		mV
Amplifier CMRR at 60Hz G = 2		90		dB
Amplifier CMRR at 60Hz G = 500		95		dB
Amplifier Overload Recovery Time		200		μsec
OUTPUTS				
Digital Output Coding	Binary, Offset Binary, Two's Complement			
Serial Output Coding	Non-return to zero (NRZ)			
ADC Conversion Time[5]		25	30	μsec
Clock Frequency[5]		520		kHz
Delay[6] SDM856		15		μsec
Delay[6] SDM857		30		μsec
POWER REOUIREMENTS				
Rated Voltage for Specified Accuracy	±14.5	±15	±15.5	
	+4.75	+5	+5.25	V
Quiescent Current				
SDM856, +15VDC		+10	+20	mA
SDM856, -15VDC		-35	-50	mA
SDM856, +5VDC		+120	+140	mA
SDM857, +15VDC		+15	+25	mA
SDM857, -15VDC		-40	-55	mA
SDM857, +5VDC		+120	+140	mA
Power Dissipation SDM856		1300	1750	mW
Power Dissipation SDM857		1400	1900	mW
ENVIRONMENTAL				
Specification Temperature Range	0		+70	°C
Operating Temperature Range	-25		+85	°C
Storage Temperature Range	-55		+100	°C
PRICES	1-24	25-99	100-249	
SDM856JG	149.00	115.00	99.00	$
SDM856KG	179.00	138.00	125.00	$
SDM857JG	179.00	138.00	125.00	$
SDM857KG	197.00	152.00	138.00	$

NOTES:

1. R_{EXT} is the external gain-setting resistor. (Connect between pins 1 and 80).
2. FSR means Full Scale Range, e.g., FSR is 10V for ±5V range.
3. Adjustable to zero.
4. Includes gain, offset, and linearity drifts.
5. Conversion time and clock frequency can be externally adjusted from 13μsec (f_{clock} = 1.0MHz) to 110μsec (f_{clock} = 118kHz). (Conv. times are for 12-bit resolution.)
6. Can be externally adjusted from 3μsec to 300μsec.

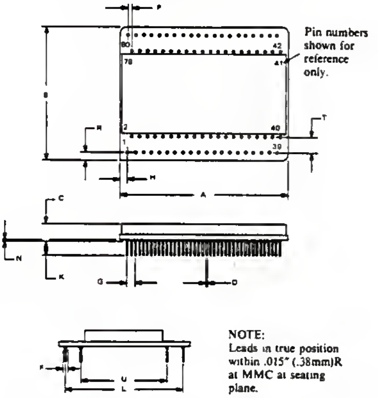
The information in this publication has been carefully checked and is believed to be reliable; however, no responsibility is assumed for possible inaccuracies or omissions. Prices and specifications are subject to change without notice. No patent rights are granted to any of the circuits described herein.

PIN DESIGNATIONS

IA GAIN AOJUST	•1	•80	IA GAIN ADJUST
MUX OUT HI	2	•79	IA IN LO
IA IN HI	•3	78	MUX OUT LD
CH7	4	77	CH15/RET 7
CH6	5	76	CH14/RET6
CH5	6	75	CH13/RET5
CH4	7	74	CH12/RET4
CH3	8	73	CH11/RET3
CH2	9	72	CH10/RET2
CH1	10	71	CH9/RET1
CH0	11	70	CH8/RET0
MUX ENABLE 2	12	•69	IA DUT
MUX ENABLE 1/A3 DUT	13	68	S/H IN
SIN/DIF	14	67	S/H DFFSET ADJUST
A2 OUT	15	66	S/H CDNTROL
A1 OUT	16	65	+15VDC
A0 OUT	17	64	-15VDC
$\overline{\text{CLEAR}}$	18	63	ANA COM
A0 IN	19	62	S/H DUT
A1 IN	20	61	-6.4V REF OUT
A3 IN	21	60	10V RANGE
A3 IN	22	59	BPO
LDAD	23	58	20V RANGE
$\overline{\text{BUSY}}$	24	57	+2.5V REF IN
DIG COM	25	56	$\overline{\text{ENABLE 2}}$
SHDRT CYCLE	26	55	+2.5V REF OUT
10-BIT RESOLUTION	27	54	$\overline{\text{ENABLE 1}}$
8-BIT RESOLUTION	28	53	$\overline{\text{ENABLE 3}}$
D0 (LSB)	29	52	SERIAL DUT
D1	30	51	CLOCK DUT
O2	31	50	CLOCK RATE ADJUST
O3	32	49	+5VDC
D4	33	48	$\overline{\text{STROBE}}$
D5	34	47	OELAY AOJUST
D6	35	46	$\overline{\text{TRIG}}$
D7	36	45	OELAY DUT
D8	37	44	$\overline{\text{D11}}$
D9	38	43	BUSY
O10	39	42	BUSY $\overline{\text{ENABLE}}$
D11 (MSB)	40	41	$\overline{\text{D11}}$ $\overline{\text{ENABLE}}$

*For SDM857 only. Make no connection in SDM856.

MECHANICAL



Pin numbers shown for reference only.

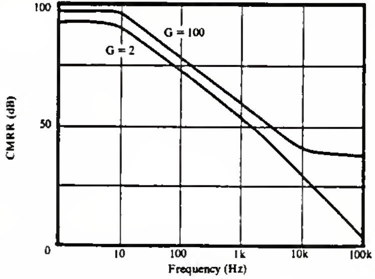
NOTE:
Leads in true position within .015" (.38mm)R at MMC at seating plane.

DIM	INCHES		MILLIMETERS	
	MIN	MAX	MIN	MAX
A	2.120	2.180	53.85	55.37
B	1.670	1 720	42 42	43 68
C	.170	.230	4.32	5.84
D	018	021	0.46	0.53
F	035	050	0.88	1.27
G	100 BASIC		2 54 BASIC	
H	100 BASIC		2.54 BASIC	
K	150	.250	3.81	6.35
L	1 500 BASIC		38.1 BASIC	
N	002	010	0.05	0.25
P	050 BASIC		1 27 BASIC	
R	100 BASIC		2.54 BASIC	
T	200 BASIC		5.08 BASIC	
U	1 100 BASIC		27 94 BASIC	

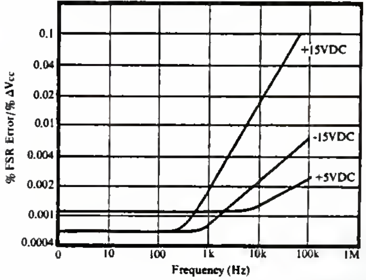
MATERIAL: Alumina
WEIGHT: 32 grams (1.2 oz.)
PINS: Pin material and plating composition conform to Method 2003 (solderability) of Mil-Std-883 (except paragraph 3.2)
MATING CONNECTOR: 2350MC (Set of four 20-pin strips) or 0422MC (assembled unit).

TYPICAL PERFORMANCE CURVES

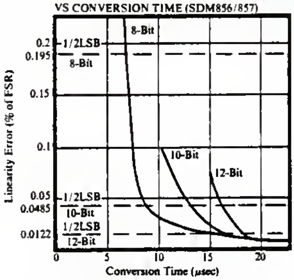
INSTRUMENTATION AMPLIFIER/MULTIPLEXER CMRR VS FREQUENCY (SDM857)



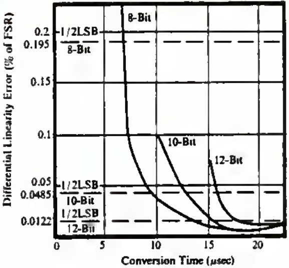
POWER SUPPLY REJECTION VS POWER SUPPLY RIPPLE FREQUENCY (SDM856/857)



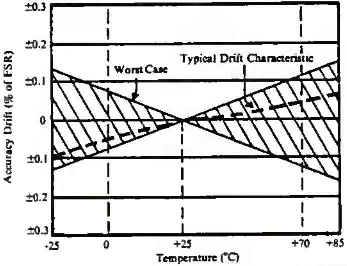
A/D CONVERTER LINEARITY ERROR VS CONVERSION TIME (SDM856/857)



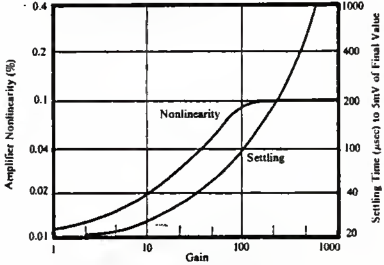
A/D CONVERTER DIFFERENTIAL LINEARITY ERROR VS CONVERSION TIME (SDM856/857)



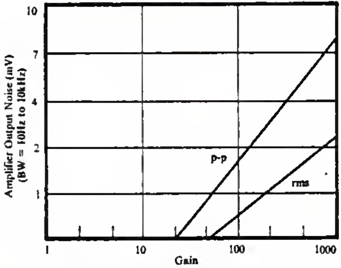
ACCURACY DRIFT VS TEMPERATURE (SDM856/857)



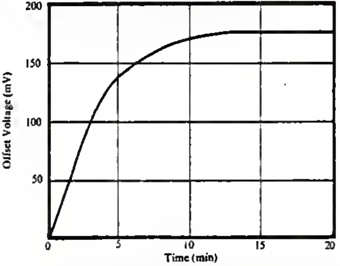
NONLINEARITY AND SETTLING TIME VS AMPLIFIER GAIN (SDM857)



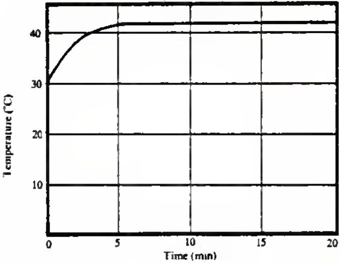
OUTPUT NOISE VS AMPLIFIER GAIN (SDM857)



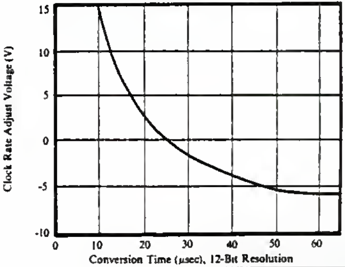
INSTRUMENTATION AMPLIFIER OFFSET WARM-UP (Gain = 500) (SDM857)



CASE TEMPERATURE VS TIME WITH NO HEAT SINK OR AIR FLOW (SDM856/857)



A/D CONVERSION TIME VS CLOCK RATE CONTROL VOLTAGE, 12-BIT RESOLUTION (SDM856/857)

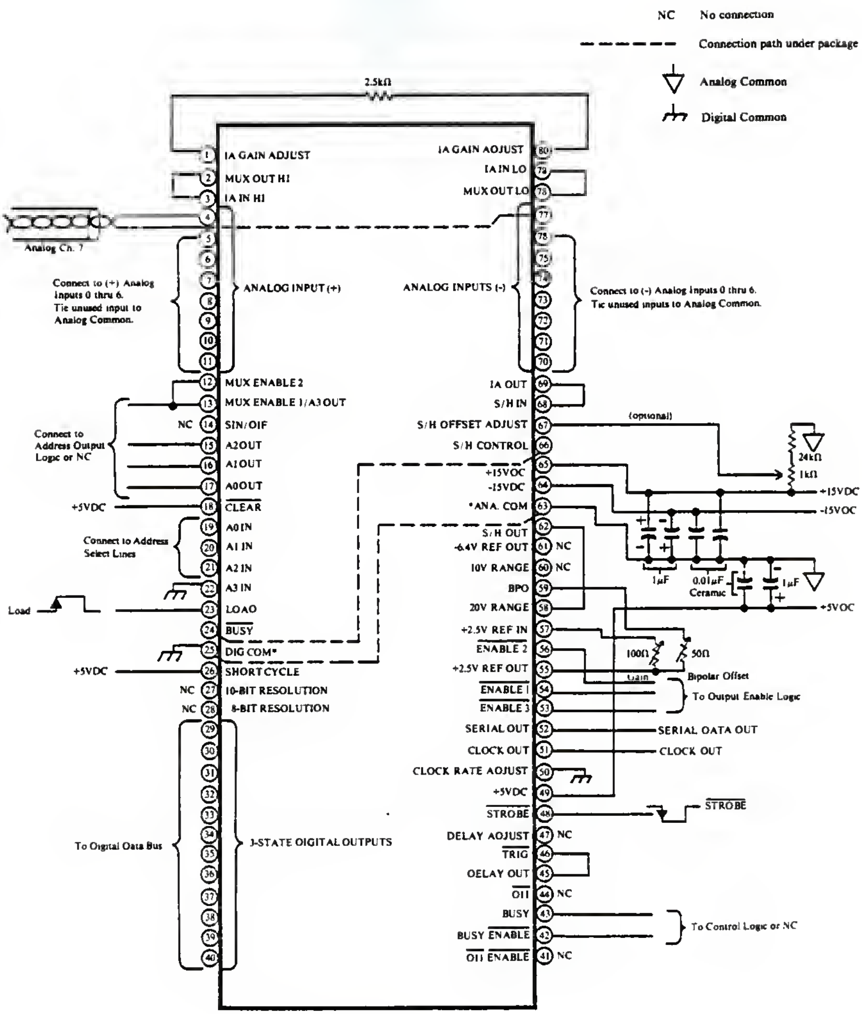


DESCRIPTION OF PIN FUNCTIONS

NUMBER	OESIGNATION	OESCRIPTION
Pin 1	IA GAIN AOJUST	(SOM857 only). By connecting a resistor between pin 1 and pin 80 tbe gain of the internal instrumentation amplifier can be varied as follows: $\text{Gain} = 2 + (20\text{k}\Omega/R_{EXT})$ where R_{EXT} is the gain setting resistor. The IA is factory adjusted for a gain of 2 without any external resistor. Important: If a gain greater than 10 is required an external capacitor must be eonnected from "OELAY AOJ." (pin 47) to +5VDC. This increases an internal delay to allow for the increased settling time of the instrumentation amplifier (see page 12). For SOM856 make no connection.
Pin 2	MUX OUT HI	High output of the analog input multiplexer. Connect to pin 3 (IA IN HI) for differential operatinn. Connect to pin 78 (MUX OUT LO) and pin 3 (SOM857) or (S/H IN) pin 68 (SOM856) for single-ended input operation.
Pin 3	IA IN HI	(SOM857 only). Positive input of the internal instrumentation amplifier. Connect to pin 2 (MUX OUT HI) for normal operation. For SOM856 make no connection.
Pins 4 thru 11	CH7-CH0	The first 8 (of 16) analog inputs for single-ended operation or the 8 positive inputs for 8-channel differential input operation.
Pin 12	MUX ENABLE 2	Connect to pin 14 (SIN/OIF) for single-ended input operation. Connect to pin 13 (MUX ENABLE 1) for differential input operatinn.
Pin 13	MUX ENABLE 1/ A3 OUT	Leave open for single-ended input operation. Connect to pin 12 (MUX ENABLE 2) for differential input operation. Also, A3 output line.
Pin 14	SIN/OIF	Single/Oifferential input operation. Connect to pin 12 (MUX ENABLE 2) for single-ended operation. Lcave open for differential input operation.
Pins 15, 16, 17	A0 OUT - A2 OUT	Output lines from input channel address latch (A3 OUT is on pin 13).
Pin 18	$\overline{\text{CLEAR}}$	A low on this line elears the address latch causing the SOM856/857 to address channel 0 regardless of the information present on AO IN - A3 IN. Connect to +5VDC or to user logic circuitry.
Pins 19, 20, 21, 22	AO IN - A3 IN	Address lines that select one of 16 analog input signals (CH0-CH15). 0000 selects channels 0 and 1111 selects channel 15. Connect A3 to ground for 8-channel differential operation. The address is latehed with a positive TTL edge on the LOAO (pin 23).
Pin 23	LOAO	A positive TTL edge on this pin latehes the input ehannel address present on AO IN - A3 IN (pins 19, 20, 21, 22).
Pin 24	$\overline{\text{BUSY}}$	This signal will be low during the A/O conversion (≈25μsec). Output data is not valid while this signal is low. Connect to S/H CONTROL (pin 66).
Pin 25	OIG COM	Oigital common. Connect to ANA COM (pin 63) as elose to the SOM856/857 as possible.
Pin 26	SHORT CYCLE	This pin allows short cyeling the A/O converter for lower resolutions thereby obtaining faster conversion times. Connect to +5VDC (pin 49) for 12-bit resolution, (pin 27) for 10-bit resolution, or (pin 28) for 8-bit resolution.
Pin 27	10-BIT RESOLUTION	To short cycle to 10-bit resolution connect to pin 26. Otherwise, make no connection.
Pin 28	8-BIT RESOLUTION	To short cycle to 8-bit resolution, connect to pin 26. Otherwise, make no eonnection.
Pins 29 thru 40	D0-O11	12-bit data bus. 3-state low power Schottky TTL-compatible.
Pin 41	$\overline{\text{O11}}$ $\overline{\text{ENABLE}}$	$\overline{\text{D11}}$ (pin 44) is enabled when $\overline{\text{D11}}$ $\overline{\text{ENABLE}}$ is low.
Pin 42	BUSY $\overline{\text{ENABLE}}$	BUSY (pin 43) is enabled when BUSY $\overline{\text{ENABLE}}$ is low.
Pin 43	BUSY	3-state output that will be high only while an A/O conversion is in process. Output data is not valid while this signal is high.
Pin 44	$\overline{\text{O11}}$	$\overline{\text{MSB}}$. Use instead of O11 when two's complement output is required.
Pin 45	OELAY OUT	This pulse is used to delay the beginning of the A/O conversion to allow for the settling of the multiplexer, instrumentation amplifier, and sample/hold.
Pin 46	$\overline{\text{TRIG}}$	A negative TTL edge on this pin initiates the A/O conversion. Connect to OELAY OUT (pin 45).
Pin 47	OELAY AOJUST	When the SOM856/857 is addressed, an internal delay of approximately 30μsec (SOM857) or 15μsec (SOM856) is initiated to allow for multiplexer, instrumentation amplifier, and sample/hold settling time. When the IA is operated with gain > 10 this delay must be increased to allow for the increased settling time of the IA. (see Table IV and page 14. The delay can also be shortened for faster lower-resolution operation.
Pin 48	$\overline{\text{STROBE}}$	A negative TTL edge on this pin initiates the OELAY OUT pulse.
Pin 49	+5VDC	+5VOC at 140mA maximum, 120mA typical.
Pin 50	CLOCK RATE AOJUST	Varying the voltage at this pin changes the elock frequency and thereby ehanges the conversion speed of the A/O converter. Connect to OIG COM (pin 25) for 12-bit operation (25μsec A/O conversion time). Connect to +5VDC for 10-bit operation and eonnect to +15VOC for 8-bit operation. (see page 13).
Pin 51	CLOCK OUT	A/O converter elock output. Output is present only during A/O conversion. N + 1TTL pulses are output at a 520kHz rate where N is the resolution.
Pin 52	SERIAL OUT	Serial output data in NRZ format is synchronous with CLOCK OUT (pin 51) signal. Use negative edge of CLOCK OUT to strobe caeh bit.
Pins 53, 54, 56	$\overline{\text{ENABLE 3}}$/ $\overline{\text{ENABLE 1}}$/ $\overline{\text{ENABLE 2}}$	3-state enable lines for data bus O11 - D0 (MSB = O11). $\overline{\text{ENABLE 1}}$ (pin 54) enables O11 - O8; $\overline{\text{ENABLE 2}}$ (pin 56) enables O7 - D4; $\overline{\text{ENABLE 3}}$ (pin 53) enables O3 - D0. A low on the enable line enables data outputs.
Pin 55	+2.5V REF OUT	Positive voltage reference output. Connect to REF IN (pin 57) (through 50Ω) for unipolar or bipolar operation (unless an external reference is used). Also connect to BPO (pin 59) (through 25Ω) for bipolar operation.
Pin 57	+2.5V REF IN	Reference voltage input. Connect to +2.5V REF OUT (pin 55) (through 50Ω resistor or 100Ω pot) or use external +2.5V reference. (+2.5V ±10mV at 0.5mA required).

DESCRIPTION OF PIN FUNCTIONS

NUMBER	DESIGNATION	DESCRIPTIDN
Pin 58	20V RANGE	A/D converter input resistor. Using SDM857: connect to S/H OUT (pin 62) for ±5V operation. Leave open for other input ranges. Using SDM856: leave open unless an external IA with a gain greater than 2 is used. (Input multiplexers are limited to ±6V maximum input voltage.)
Pin 59	BPO	A/D converter bipolar offset. Connect to REF DUT (pin 55) through a 25Ω resistor or a 50Ω pot for bipolar operation. Leave open for unipolar operation.
Pin 60	10V RANGE	A/D converter input resistor. Using SDM857 with internal instrumentation amplifier with a minimum gain of 2: connect to S/H OUT (pin 62) for 0 to +5V mux input unipolar operation or ±2.5V mux input bipolar operation. Leave open for ±5V input bipolar operation. Using SDM856 without IA: connect to S/H DUT (pin 62) for ±5V mux input operation.
Pin 61	-6.4V REF OUT	Negative voltage reference output. Maximum current drain from this point without degradation of specifications is 200μA.
Pin 62	S/H OUT	Sample/hold output. Connect to 10V RANGE (pin 60) or 20V RANGE (pin 58) for normal operations.
Pin 63	ANA COM	Analog common. Connect to DIG COM (pin 25) as close to the SDM856/857 as possible.
Pin 64	-15VDC	-15VDC at 30mA typical.
Pin 65	+15VDC	+15VDC at 30mA typical.
Pin 66	S/H CONTRDL	A low signal on this line causes the sample/hold to enter the hold mode. Connect to $\overline{\text{BUSY}}$ (pin 24).
Pin 67	S/H OFFSET ADJUST	Dffset adjust for sample/hold (see Figure 10).
Pin 68	S/H IN	Input to sample/hold amplifier. Connect to IA OUT (pin 69 - SDM857) or MUX OUT HI (pin 2) and MUX OUT LO (pin 78 - SDM856).
Pin 69	IA OUT	(SDM857 only). Instrumentation amplifier output. Connect to S/H IN (pin 68) for normal operation. For SDM856 make no connection.
Pins 70 thru 77	CH8-CH15 RET0 - RET7	Analog inputs 8 through 15 for single-ended operation or analog returns 0 through 7 for differential input operation.
Pin 78	MUX DUT LO	Multiplexer output for CH8-CH15 (single-ended) or RET0-RET7 (differential). Connect to MUX DUT HI (pin 2) and IA IN HI (pin 3 - SDM857) or S/H IN (pin 68 - SDM856) for single-ended input operation or connect to IA IN LD (pin 79) for differential input operation.
Pin 79	IA IN LD	(SDM857 only). Negative input of instrumentation amplifier. Connect to ANA COM (pin 63) for single-ended input operation or MUX DUT LD (pin 78) for differential input operation. For SDM856 make no connection.
Pin 80	IA GAIN ADJUST	See pin 1 description.



*Analog and Digital Common should be connected together close to the unit.

FIGURE 2. Connection Diagram for SDM857 Operating Under These Conditions:
IA Gain = 10
Analog Input: Bipolar, differential
Reference Voltage: Internal
Resolution: 12-bits
Mode: Normal
Digital Output: Binary

SETUP PROCEDURE

INPUT CONNECTIONS

Unused analog inputs must be connected to ANA COM, pin 63. When long leads are connected to the inputs, care must be taken that leads do not pick up excessive noise from external equipment and wiring. When low level applications are undertaken, it is usually advisable to operate the system as an 8-channel, differential input system (the SDM856 requires an external differential amplifier to operate in this mode). In this way any noise will be common to both input wires, and will be rejected by the instrumentation amplifier. For best noise rejection use twisted shielded pair cable. The inputs of the SDM856/857 are protected from damage by voltage as high as 15.5 volts and from short spikes well in excess of this for a few microseconds; however, careful wiring and cable routing practices are recommended.

Single-Ended Inputs

Two configurations may be used with 16 single-ended channels. They are single-ended with local ground or remote signal ground.

Local Ground:	Connect pins 2 and 78 to 3 (SDM857) or 68 (SDM856), 79 to 63, unused inputs to 63, and all signal returns to 63.
Remote Ground:	Same as local ground except connect 79 to remote signal ground. (SDM857 only).

Differential Inputs (SDM857)

Connect the signal inputs to pins 4 through 11, and their returns to 77 through 70. Connect pin 12 to 13, 2 to 3, 78 to 79, and 69 to 68.

Differential Inputs (SDM856 With External Instrumentation Amplifier)

Connect the signal inputs to pins 4 through 11, and their returns to 77 through 70. Connect pin 12 to 13. Connect pins 2 and 78 to the noninverting and inverting input of the amplifier respectively. The output of the amplifier is connected to pin 68.

AMPLIFIER GAIN (SDM857)

The instrumentation amplifier gain may be set to any value between 2 and 500 by connecting an external gain resistor between pins 1 and 80. The gain is determined by the formula: $G = 2 + (20k\Omega/R_{EXT})$. Internal gain determining resistors have an accuracy of ±0.1% and a maximum temperature coefficient of ±10ppm/°C. In normal operation IA OUT, pin 69 is connected to S/H IN, pin 68.

SAMPLE/HOLD

Connect S/H CONTROL, pin 66, to the ADC $\overline{\text{BUSY}}$ output, pin 24.

ANALOG-TO-DIGITAL CONVERTER INPUT VOLTAGE RANGE

The analog-to-digital converter is essentially a current input device having a current input range of 0 to 2mA. The input may be considered a virtual ground summing point. To convert voltage to current, a center tapped 10kΩ resistor is internally connected to this summing point. This is illustrated in Figure 3.

The interconnections of the ADC pins and the S/H OUT, pin 62, are shown in Table I.

TABLE I. ADC Range Jumpers.

Input Range (V)	Jumper
0 to +10	59 Open, 60 to 62, 58 Open
-5 to +5	59 to 55, 60 to 62, 58 Open
-10 to +10	59 to 55, 58 to 62, 60 Open

NOTE: Input ranges in Table 1 apply to ADC only. The input multiplexer is limited to ±6V maximum.

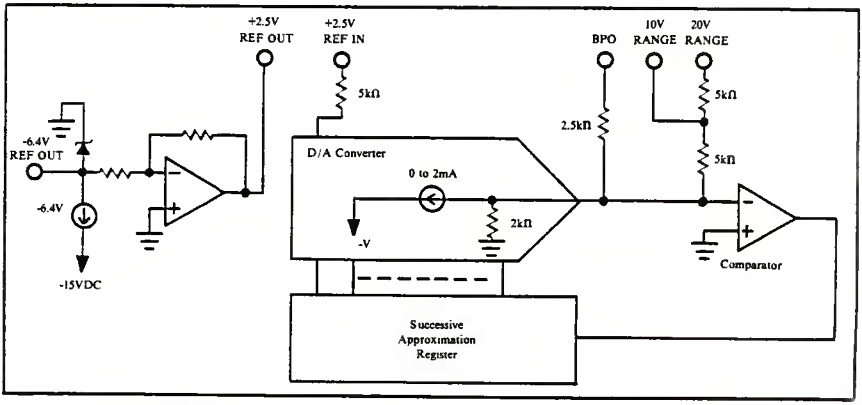


FIGURE 3. Analog-to-Digital Converter.

OUTPUT CODE

For unipolar binary and offset binary use D11 (pin 40) for the most significant bit. Two's complement binary is obtained by using pin 44, $\overline{\text{D11}}$, as the most significant bit. One's complement code may be obtained by a different offset adjustment in the calibration procedure. Two's complement and one's complement codes are usually used only for bipolar signal ranges. For 12-bit resolution, SHORT CYCLE (pin 26) is left open or taken to +5VDC. Connect pin 26 to pin 27 (10-bit) or pin 28 (8-bit) to obtain lower resolution. The conversion time will be shortened by the following formula:

(Conversion Time) = (25μsec) x [1 - (12-R/13)]
Where R is the resolution desired.

NORMAL AND OVERLAP MODE

The two basic modes of system operation are normal and overlap. In normal operation the channel address, N, is loaded or clocked into tbe address latch. The addressed channel will remain selected during its analog-to-digital conversion. In overlap mode channel N + 1 is selected while channel N is being converted. This can be used to increase the system throughput rate by allowing the multiplexer and instrumentation amplifier to settle while a conversion is being made. In this way the throughput rate is limited by the sample/hold acquisition time and the analog-to-digital converter conversion time. This will be true except for low level operation where the instrumentation amplifier's settling time has been increased to a value greater than that required for the sample/hold and converter. For this reason, the overlap mode is more desirable for low level signals. Table II and Figures 4 and 5 provide additional timing details. At high signal levels a high source resistance may increase the multiplexer settling time to an extent which makes the overlap mode desirable (see page 12).

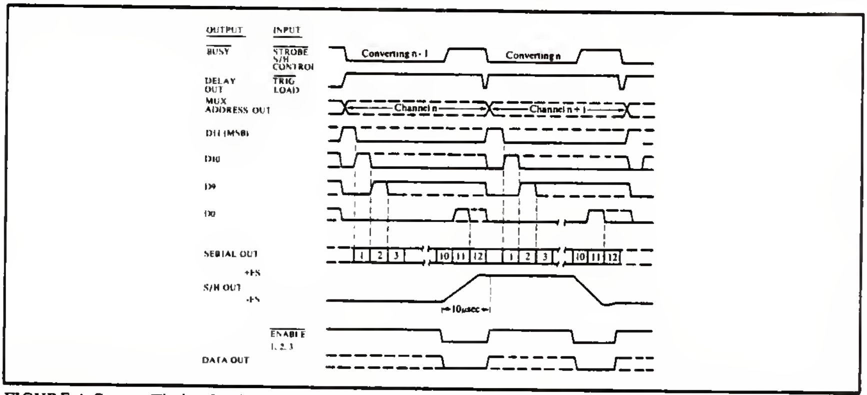


FIGURE 4. System Timing for Overlap Operation.

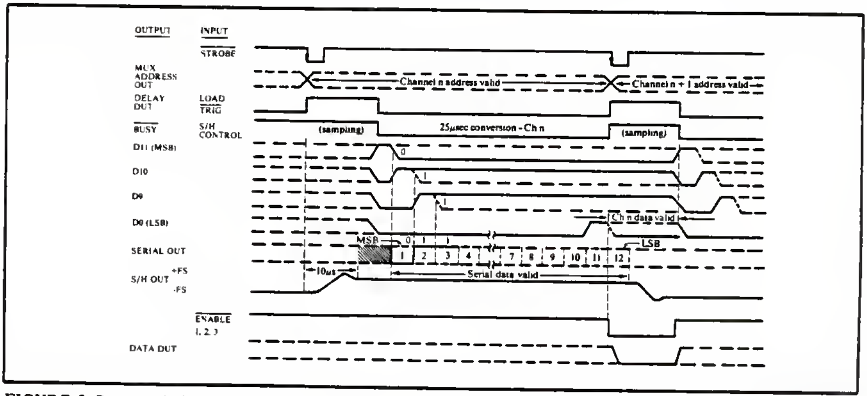


FIGURE 5. System Timing for Normal Operation.

Normal Mode Connections

Connect DELAY OUT, pin 45 to $\overline{\text{TRIG}}$, pin 46.

Overlap Mode Connections

Connect $\overline{\text{BUSY}}$, pin 24 to $\overline{\text{STROBE}}$, pin 48, and DELAY OUT, pin 45 to $\overline{\text{TRIG}}$, pin 46. Adjust the delay as shown in Table II and described in the following paragraph.

DELAY ADJUSTMENT

The delay timer may be adjusted with an external capacitor or resistor from DELAY ADJUST (pin 47) to +5VDC. A capacitor will increase the delay to allow for the increased settling time of the instrumentation amplifier at high gains while a resistor will decrease the delay to allow for increased throughput rate with an external high speed instrumentation amplifier or lower resolution operation.

The values of R and C versus delay for both the SDM856 and SDM857 are shown in Figures 6, 7, 8 and 9.

TABLE II. Throughput Rate and Delay Time vs Gain for Normal and Overlap Modes.

System Gain		System Accuracy		Throughput Rate (min) (Channels/sec)				Delay Time (μsec)		
				Normal		Overlap		Normal	Overlap	
V/V		KG	JG	JG	KG	JG	KG	JG and KG	JG	KG
1	856 only	±0.024%	±0.048%	33k	25k	38k	27k	15	26	35
2	857 only	±0.024%	±0.048%	22k	18k	38k	27k	30	26	35
10	857 only	±0.035%	±0.06%	22k	18k	38k	27k	30	26	35
100	857 only	±0.08%	±0.11%	10k	9k	11k	11k	90	90	
500	857 only	±0.1%	±0.15%	2.5k	2.4k	2.6k	2.6k	390	390	

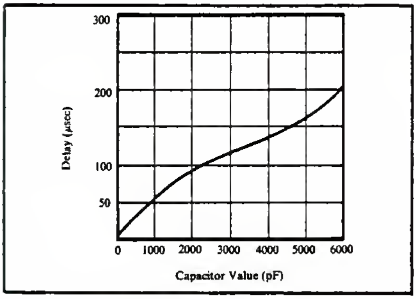


FIGURE 6. Typical Capacitor Value to Increase Delay Time (SDM857).*

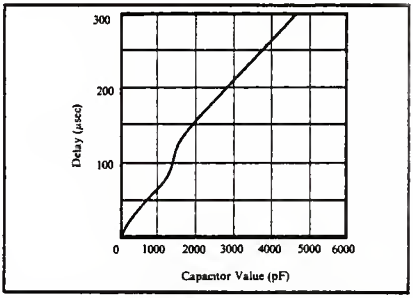


FIGURE 7. Typical Capacitor Value to Increase Delay Time (SDM856). *

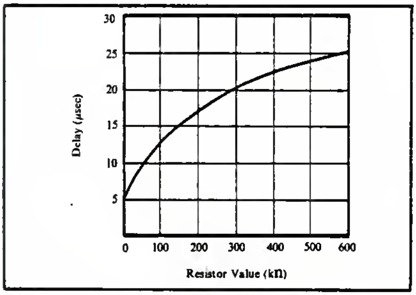


FIGURE 8. Typical Resistor Value to Decrease Delay Time (SDM857). *

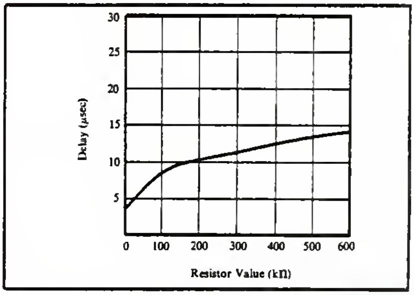


FIGURE 9. Typical Resistor Value to Decrease Delay Time (SDM856). *

*Capacitor or resistor is connected from pin 47 to +5V supply.

GROUNDING CONSIDERATIONS

The circuit configuration of a high speed successive approximation A/D converter is such that low level analog and digital signals are in close proximity. In fact the two circuits are actually interconnected; for this reason no AC noise voltage should be allowed to exist between digital and analog ground. Digital and analog ground should be connected as close to the unit as possible. In a typical application an SDM module will be used near a computer. For best results the SDM digital ground should be connected to the computer's +5VDC supply ground at the supply terminal. The ±15VDC supply ground should be left floating, if possible. The Model 546 +5VDC to ±15VDC DC/DC converter is a convenient way to do this. For single-ended systems, signal returns are connected to analog ground; or if a common remote signal ground is available, the inverting input of the differential amplifier (SDM857) should be used for the signal return.

CALIBRATION PROCEDURE

GAIN AND OFFSET ADJUSTMENT

External gain and offset adjust potentiometers are shown in Figure 10. Cermet pots with a T.C.R. of ±100ppm/°C or less should be used. The adjustments shown each have a range of ±0.3% of the Full Scale Range.

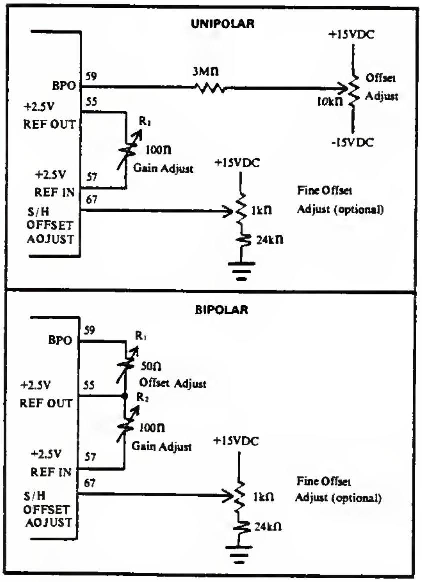


FIGURE 10. External Gain and Offset Adjustment.

If adjustment of gain and offset is not required R1 and R2 should be replaced with 25Ω and 50Ω resistors respectively. These resistors should be low T.C. (<±100ppm/°C) metal film or equivalent.

The S/H OFFSET ADJUST (pin 67) may be used as a fine offset adjustment.

The easiest way to calibrate the device is to connect a voltage source to multiplexer input CH0 (either differential or single-ended input operation may be used). Channel zero will be addressed by simply connecting $\overline{\text{CLEAR}}$ to DIG COM.

After the CH0 voltage source has been addressed, set it to the most negative value of the input range being used plus 1/2LSB. Twelve-bit LSB voltage values are given in Table III. Connect a triggering source to $\overline{\text{STROBE}}$ and adjust the offset potentiometer until all output bits are logic zero with bit DØ dithering between logic zero and one. Change the source voltage to the most positive value of the input range minus 3/2LSB. Adjust the gain potentiometer until all output bits are logic one with bit D0 dithering between logic one and zero. When a resolution less than 12 bits is used, the LSB voltage is given by the formula in Table III where N is the number of output bits. One's complement coding is obtained by shifting the previous adjustments up by 1/2LSB using the offset potentiometer.

TABLE III. LSB Values for 12-Bit Resolution.

LSB (Volts) = (Range)/(2^N)

Range	LSB Voltage (12-Bits)
5V	1.22mV
10V	2.44mV

CLOCK RATE ADJUSTMENT

To obtain higher throughput rates at lower accuracy the A/D clock rate can be adjusted by varying the voltage on the clock rate adjust pin. This point should be connected to digital common for 12-bit accuracy, +5VDC for 10-bit accuracy, or +15VDC for 8-bit accuracy giving conversion times of 25μsec, 15μsec and 10μsec respectively. The conversion speed can also be continuously varied from about 13μsec to 110μsec (12-bit resolution) with a potentiometer as shown in Figure 11.

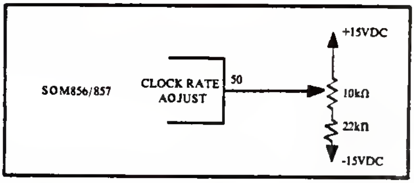


FIGURE 11. Clock Rate Adjustment.

CHECKOUT PROCEDURE

Checkout is essentially accomplished by the calibration procedure. Before the unit is plugged into a new installation, it is well to go over the pin connection list to be sure that all 80 pins have been properly connected in the setup. Linearity and monotonicity may be verified by varying the input voltage over the complete range during the calibration procedure.

LATCH

Latch operation can be verified by connecting a pulse generator to the LOAD input. The address inputs (A0 IN - A3 IN) should appear at the address outputs (A0 OUT - A3 OUT).

AMPLIFIER AND MULTIPLEXER

To check amplifier operation, connect a voltmeter to IA OUT (pin 69) and observe that the output follows the input voltage as in the calibration procedure. Check the multiplexer in the same way noting that the output cbanges when the address is changed.

SAMPLE/HOLD

The sample/hold circuit can be checked during the calibration procedure by observing the output of the S/H OUT (pin 62) with an oscilloscope. The waveform should be approximately as in Figure 12.

The charge offset will vary in a linear manner from about 10mV for -10V to 30mV for +10V. Tbis is compensated for by the offset and gain adjustments of tbe A/D converter. The spikes during conversion are normal noise caused by the converter operation.

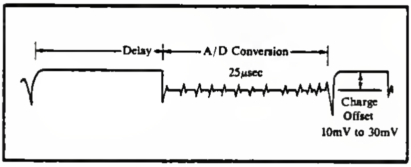


FIGURE 12. Sample/Hold Output Waveform.

ANALOG-TO-DIGITAL CONVERTER

The ADC can be checked out as an individual circuit element. Connect a fixed voltage to either 20V RANGE (pin 58) or 10V RANGE (pin 60). After adjusting the gain and offset errors as described on page 13, the digital output should represent the analog input as shown in Table IV. To enable the three-state buffers, pins 53, 54 and 56 should be connected to logic 0.

In overlap, when the amplifier/multiplexer settling time is less than the ADC conversion time, set the delay timer for the ADC conversion time plus the sample/hold acquisition time (30μsec plus 10μsec). When the amplifier/multiplexer settling time exceeds the ADC conversion time plus the S/H acquisition time, set the delay timer for the amplifier/multiplexer settling time.

TABLE IV. Delay Timer Settings for Specified Settling Time Accuracies of the Instrumentation Amplifier.

Full Scale Input Range	ADC Range	Amplifier Gain	Resolution	Amplifier/Multiplexer Settling Time (μsec)		
				To ±0.2%	To ±0.05%	To ±0.01%
10V	-10 to +10	2	2.44mV	8	10	20
1V	0 to +10	10	244μV	12	14	24
0.1V	0 to +10	100	24.4μV	65	80	90
20mV	0 to +10	500	4.88μV	320	390	450

APPLICATION NOTES

CHANNEL CAPACITY EXPANSION

The SDM856/857 may be easily expanded to any number of channels by using Burr-Brown Models MPC8D and MPC16S. The MPC8D is an 8-channel double-ended multiplexer, and tbe MPC16S is a 16-channel single-ended multiplexer. These devices are CMOS FET units which can operate from supply voltages up to ±20VDC. They feature latch-free operation with full input protection. Binary decoding and level sbifting circuits are included. Logic levels are jumper selectable for TTL or CMOS. Packaging is a 28-pin DIP.

There are two methods for using these devices for channel capacity expansion. The SDM856/857 multiplexer may be expanded by shunt or series connected multiplexers. Shunt connection refers to connecting the output of several multiplexers togetber and enabling each in sequence. The disabled devices present a very high resistance to the common output line. The disadvantages to this scbeme are increased leakage current and output capacitance. For these reasons sbunt connections are usually used only when it is desired to expand the capacity by a factor of two or three. A shunt connected system logic diagram is shown in Figure 13. Forty-eight single-ended channels are indicated; however 24 double-ended channels could easily be realized by using two MPC8D's and connecting the two-sided outputs appropriately. For large systems series connected expansion is usually used. In this method tbe outputs of a second tier of multiplexers are connected to the inputs of the SDM856/857 multiplexer. This allows up to 256 single-ended or 128 double-ended channels to be addressed. A third tier can be used for 4096 single or 2048 double-ended channels. A logic diagram of a series system is shown in Figure 14. Double-ended operation can be obtained by using the MPC8D instead of the MPC16S and connecting the SDM856/857 for double-ended operation.

SEQUENTIAL ADDRESSING

Simply adding an external counter will allow sequential addressing of all 16 input channels (see Figure 15).

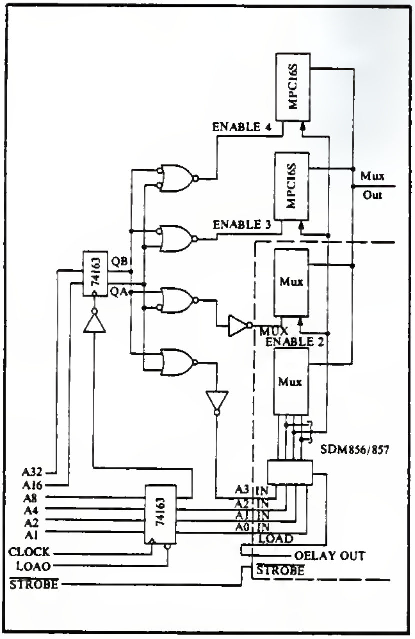


FIGURE 13. Shunt Connected Multiplexer System 32 Single-Ended Channels.

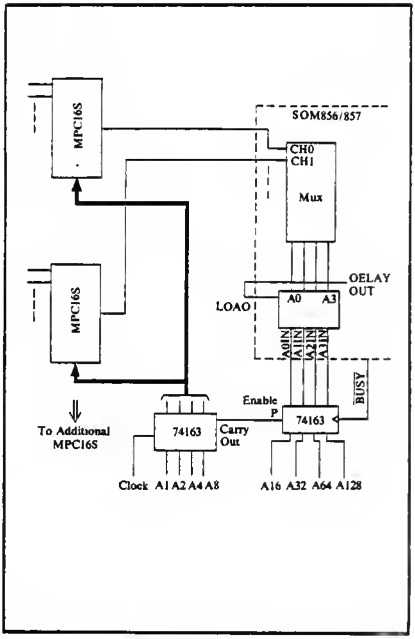


FIGURE 14. Series Connected Multiplexers, 256 Single-Ended Channels. Sequential Addressing.

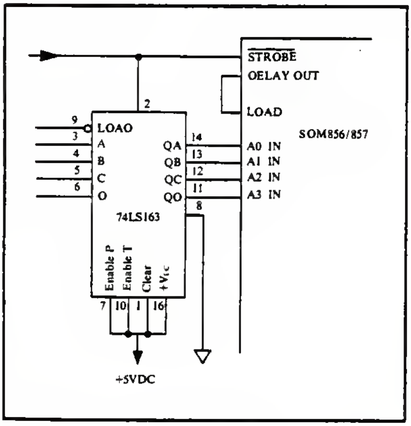


FIGURE 15. Sequential Addressing.

MULTIPLEXER CIRCUIT OPERATION

At the address and enahle inputs a voltage is interpreted as a logic "1" if it is greater than 2.4 volts; and "0" if less than 0.8 volts.

When an input channel has been selected the "on resistance" from input to output is approximately 1.8kΩ. The input capacitance for each channel is approximately 7pF; while the output capacitance is approximately 25pF for each 8-channel multiplexer. A circuit model of an ON channel is shown in Figure 16.

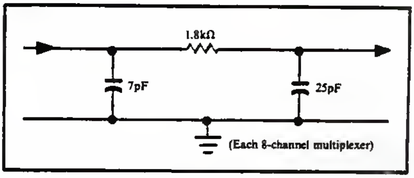


FIGURE 16. ON Channel Circuit Model.

This model is very important when high speed switching of high output impedance sources is required. For example, if the full accuracy and resolution of the system is required, the signal at the output of the multiplexer must be allowed to settle to about 0.01%. If the source impedance is 1kΩ, the 7pF can be neglected and the multiplexer has a time constant of 2.8kΩ x 50pF = 140nsec. It requires approximately 9 time constants to settle to 0.01%; 1.26μsec is well within the 30μsec (SDM857) or 15μsec (SDM856) of the delay timer. However, if the source impedance had been 10kΩ, the 0.01% settling time would have approached 6μsec. For high speed multiplexing of higher impedance sources, it will usually be desirable to parallel the 7pF input capacitor with a large capacitor; however, this could limit the source bandwidth. In any case there is no point in making it any larger than 10^4 times the output capacitance, or 0.5μF. When this size storage capacitor is used, the output time contant is 1.8kΩ x 50pF = 90nsec. This means that the system settling time is essentially determined by the settling time of the differential amplifier and sample/hold circuit. For switching of large signals it must be remembered that the ON resistance is the channel resistance of a FET, and, as such, it is a nonlinear function of the applied voltages. Any FET will current limit at its I_{DSS} value. As a result, the previous calculations are only an approximation derived from a linearized model. The settling time to 0.01% for a 20V step is approximately 4.0μsec for source impedance less than 1kΩ.

The analog and digital inputs have reverse biased diode circuits which prevent damage from discharge of static electricity. However, it is still wise to take reasonable preeaution against static discharge.

BINARY SCALING

Binary sealing of the A/D converter provides LSB voltages of 2.5mV, 2.5mV, and 5.0mV for voltage ranges of 0 to 10.24V, -5.12V to +5.12V, and -10.24V to +10.24 respectively. These may be obtained by adding external resistors in series with input resistors of the A/D converter. Metal film resistors with temperature coefficients of less than 100ppm/°C are recommended. This is shown in Figure 17.

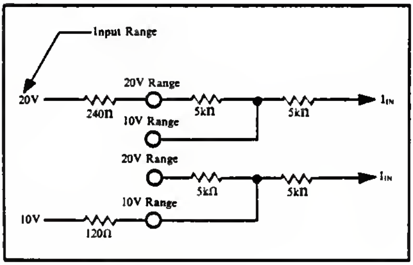


FIGURE 17. Binary Sealing.

THERMOCOUPLE TEMPERATURE ACQUISITION

Thermocouples are often used as temperature sensors for process control systems. Thermocouples are characterized by temperature coefficients of 10 to 70μV/°C and operating ranges of minus hundreds to plus thousands of degrees centigrade. When the SDM857 is operated with a instrumentation amplifier gain of 100 to 500, it may be connected directly to these devices. However, electronic instrumentation is usually mounted in a temperature controlled environment with long runs of thermocouple wire to the actual point of temperature measurement. These long wire runs often pick up large common-mode noise signals of 60Hz or higher frequencies. When the SDM857 is used as an 8-channel differential input system, the high common-mode rejection of the instrument amplifier will reject common-mode noise. To minimize differential mode noise, signal wires should be twisted and possibly shielded. As a rule, an open twisted pair is better than a coax, and a shielded, twisted pair better still. In applieations where these wiring practices eannot always be observed, a differential RC filter may be used (see Figure 19).

The 10kΩ resistors and a 10μF capacitor provide low-pass filtering (f_c = 0.8Hz) while the 1MΩ resistors supply bias current to the instrumentation amplifier. The remote sensor should be earth grounded to prevent common-mode voltages from exceeding the ±10V range of the multiplexer. This will usually supply bias current; however, the resistors provide a back up. It is not obvious what resistance the bias currents of the amplifiers will see. The 1MΩ resistors do not enter into an error calculation for input drift because the low resistance of the sensor shorts any differential voltage that might be caused by the offset or difference current of the amplifier. Offset or difference current is merely the difference between the bias currents of each input. See page 17 for a worst case error analysis of the input filter for multiplexed data acquisition systems. The 1MΩ resistors could have been put on the output side of the multiplexer eliminating the need for repeating them for each input; however, this would have loaded the 10kΩ resistors of the filter causing a possible 1% error for static conditions.

To complete a thermocouple system it is necessary to terminate all thermocouple wire pairs at an isothermal box or connector strip of some type. An ordinary barrier strip in an enclosed cabinet with even air circulation is usually adequate. The temperature of this barrier strip must be monitored to allow the observed thermocouple emf to be cold junction compensated. Figure 18 shows an excellent circuit for this purpose. Its output is connected to one of the input channels to supply ambient temperature data to the system computer.

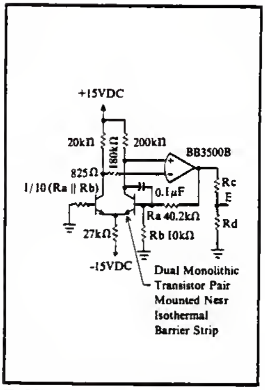


FIGURE 18. Ambient Temperature Sensor.

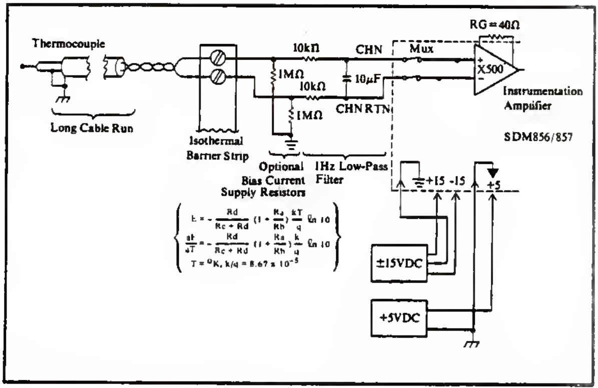

Thermocouple
Long Cable Run
Isothermal Barrier Strip
10kΩ
1MΩ
10kΩ
1MΩ
CHN
10μF
CHN RTN
Mux
RG=40Ω
X500
Instrumentation Amplifier
SDM856/857
Optional Bias Current Supply Resistors
1Hz Low-Pass Filter
+15 -15
+5
±15VDC
+5VDC
E = - Rd/(Rc + Rd) (1 + Ra/Rb) kT/q ℓn 10
dE/dT = - Rd/(Rc + Rd) (1 + Ra/Rb) k/q ℓn 10
T = °K, k/q = 8.67 x 10^-5

FIGURE 19. Thermocouple Inputs.

INPUT FILTER DESIGN FOR LOW LEVEL SYSTEMS

When the SDM856/857 is used to acquire low level sensor data, it is often desired to place a low-pass, passive filter on each input. This is usually done to reduce any differential mode, power line frequency pickup. Figure 20 shows such a circuit.

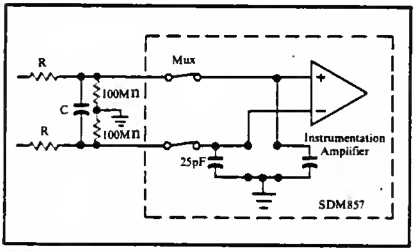


FIGURE 20. Input Filter Design for Low Level System.

This circuit is deceptive in its simplicity. Actually four errors sources should be considered in its design. They are loading, offset current, charge transfer, and pump out current.

The static loading error is simply the resistive divider created by the filter resistors and the 100MΩ input resistance. For low level sensors, 0.1% system accuracy is usually adequate. Thus R should be less than 10^{-3} x (100MΩ) = 100kΩ. However, if the inputs are scanned at a high speed, and between scans the multiplexer can be addressed to a unique channel having a lower resistance, higher filter resistances can be tolerated because the large filter capacitor will act as a voltage source during the 30μsec to 100μsec period required to read each channel. The filter capacitors will then recharge between scans.

The input offset current caused by the bias currents of the instrument amplifier as well as any leakage current of the multiplexer will cause an error voltage proportional to the size of the filter resistors ($E = I_{OS} \times 2R$). Of course, this is a static error and as for loading error, may not be important for some operating conditions. If all channels have the same resistance most of this error may be corrected by the offset adjustment of the analog-to-digital converter. If the offset current drift is 0.1nA/°C the error is 2R x 0.1nV/°C. For 10kΩ resistors this would be 2μV/°C.

When the multiplexer scans, charge will be transferred from the filter capacitor to the 25pF output capacitance of the multiplexer. For less than 0.1% of full scale error, the filter capacitor must be larger than 25000pF. This assumes that adjacent channels may differ by the full scale voltage.

Pumpout current refers to charge being transferred from the filter capacitor to the multiplexer capacitance at time intervals short enough that the filter capacitor does not have time to recharge between scans. At high scan rates this may be considered a DC current which may add to the offset current. Assume a 10μF capacitor sampled once per millisecond. For a 20mV full scale range, the maximum effective current is (20mV x 25pF)/1msec = 0.5nA. If the filter resistors are 10kΩ, a 0.5nA x 20kΩ = 10μV error is created.

When no input filter is used, the signal source must be able to charge the multiplexers and any cable capacitance during the channel acquisition time of the multiplexer and amplifier. This is discussed on page 16. When all of these errors as well as the basic 2.0μV/°C input offset voltage drift of the amplifier are considered, the overall system accuracy may be estimated.

INSTRUMENTATION AMPLIFIER OVERLOAD RECOVERY

If an analog input channel is left open or is opened due to a sensor failure it is possible for the instrumentation amplifier to saturate when that channel is addressed. Since the overload recovery time of the IA is usually much longer than the settling time specification, this can cause an error on the NEXT channel that is addressed. One way to avoid this problem is to connect 100MΩ resistors from the instrumentation amplifier inputs to analog common as shown in Figure 21. This technique will generally work for high level input (gain ≤ 10). With low-level inputs the offset current of the IA will usually cause saturation to occur anyway.

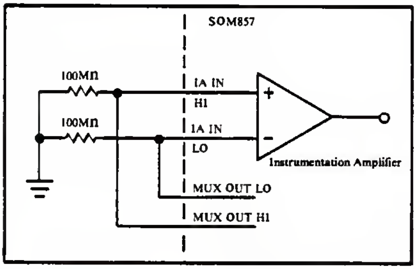


FIGURE 21. Circuit to Prevent High Level Saturation.

The SDM856 and SDM857 can be readily interfaced to operate with microprocessors. The following circuit diagrams illustrate several typical applications. The logic functions are all TTL.

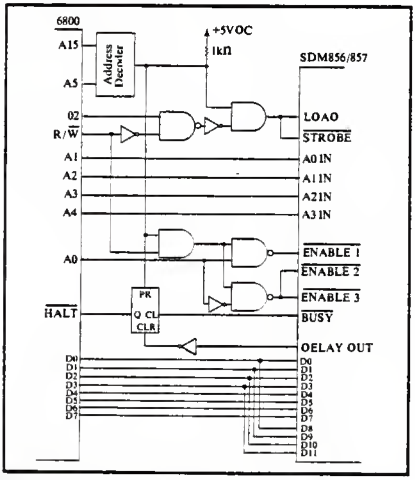


FIGURE 22. SDM856/857 Interfaced to 6800 Microprocessor.

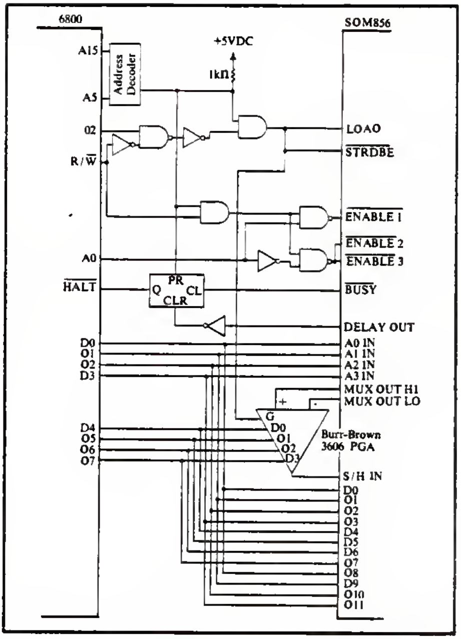


FIGURE 23. SDM856 and 3606 PGA Interfaced to 6800.

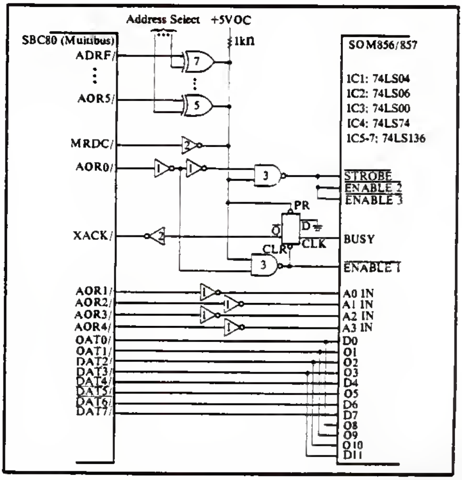


FIGURE 24. SDM856/857 Interfaced to SBC80 Multibus.

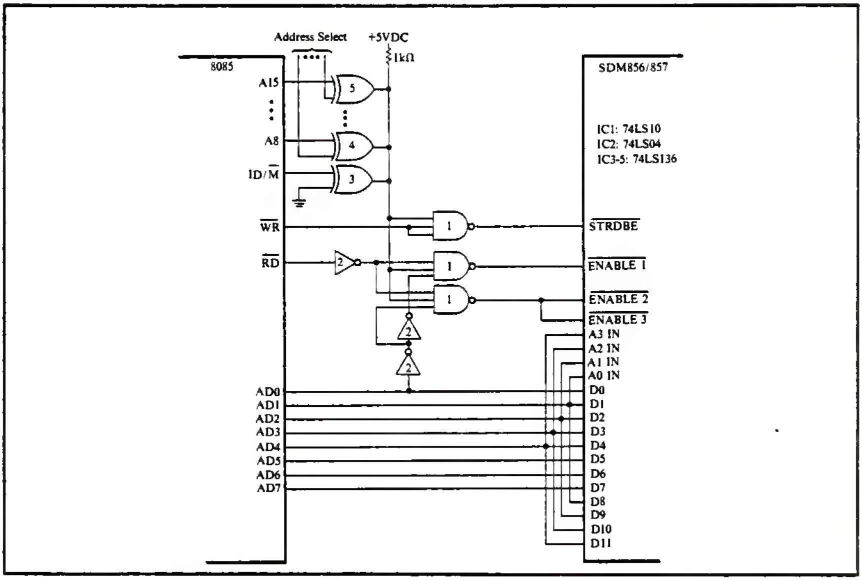
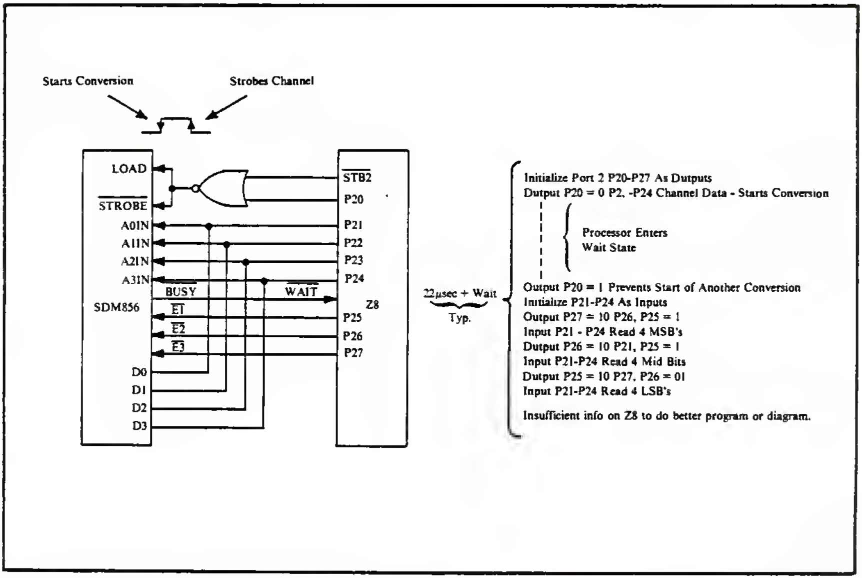


FIGURE 25. SDM856/857 Interfaced tD 8085.



Initialize Port 2 P20-P27 As Dutputs
Dutput P20 = 0 P2, -P24 Channel Data - Starts Conversion

Processor Enters
Wait State

Output P20 = 1 Prevents Start of Another Conversion
Initialize P21-P24 As Inputs
Output P27 = 10 P26, P25 = 1
Input P21 - P24 Read 4 MSB's
Dutput P26 = 10 P21, P25 = 1
Input P21-P24 Read 4 Mid Bits
Dutput P25 = 10 P27, P26 = 01
Input P21-P24 Read 4 LSB's

Insufficient info on Z8 to do better program or diagram.

FIGURE 26. SDM856/857 Interfaced to Z8.

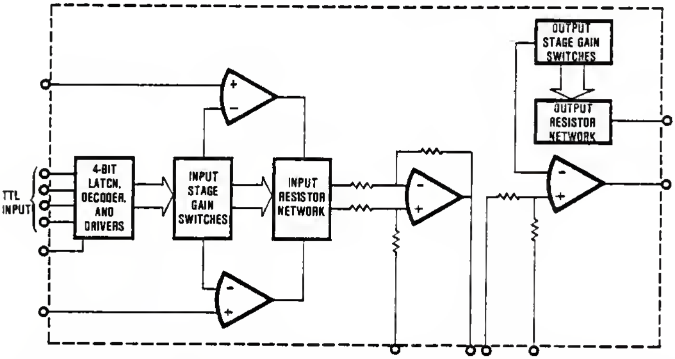
BURR-BROWN

3606

Digitally Controlled PROGRAMMABLE GAIN INSTRUMENTATION AMPLIFIER

FEATURES

- 11 BINARY GAINS - 1, 2, 4, 8, 16, 32 64, 128, 256, 512, 1024 V/V
- 4-BIT TTL GAIN CONTROL
- EXCELLENT GAIN NONLINEARITY 0.01% max at G = 1024 V/V
- LOW GAIN ERRORS - 0.02% MAX
- LOW GAIN DRIFT - 10ppm/°C MAX
- LOW VOLTAGE DRIFT - 1µV/°C MAX RTI, G = 1024V/V
- HIGH CMR - 110dB MIN, G = 1024V/V
- HIGH INPUT IMPEDANCE - $10 \times 10^9\Omega$
- LOW OFFSET VOLTAGE
  22µV max RTI, G = 1024V/V
  2mV max RTI, G = 1V/V


TTL INPUT
4-BIT LATCH, DECODER, AND DRIVERS
INPUT STAGE GAIN SWITCHES
INPUT RESISTOR NETWORK
OUTPUT STAGE GAIN SWITCHES
OUTPUT RESISTOR NETWORK
+
-

GIVES SYSTEM WIDE RANGE AND HIGH RESOLUTION

Printed in U.S.A. April, 1978

DESCRIPTION

The 3606 is a self-contained, Programmable Gain Instrumentation Amplifier (PGIA) whose gain can be changed in 11 binary weighted steps from 1 to 1024 V/V. The gain control is accomplished through a 4-bit TTL input.

The PGIA function allows the user to deal with wide dynamic range signals while maintaining high system resolution. For example: when used with a 10-bit A to D converter in a "floating point" system, the 2^{10} gain range of the 3606, plus tbe 2^{10} range of the converter produces a total system resolution of 2^{20} (≅ 1,000,000:1).

Desirable characteristics of a high performance instrumentation amplifier are offered by the 3606: high input impedance (10GΩ), excellent gain nonlinearity (0.01% max, G = 1024V/V; 0.002% max, G = 1V/V), high common-mode rejection (100dB min, G ⩾ 4V/V), low gain error (0.02% max with no trimming required), low gain temperature coefficient (10ppm/°C max), and low offset voltage drift vs temperature (1μV/°C max, RTI, G = 1024).

Added to these outstanding instrumentation amplifier characteristics is the ability to change the 3606's gain under control of a 4-bit TTL input word. An important characteristic of the 3606 PGIA is its low change in offset voltage when gain is changed. A unique design approach, plus laser trimming minimizes this change to a maximum of ±25mV with no external adjustments. With two simple offset adjustments the change can be limited to less than 2mV (1mV typ) at the output over the entire 1V/V to 1024V/V gain range.

A simplified schematic of the 3606 is shown in Figure 1. The circuit consists of a variable gain high input impedance voltage follower input stage (A1 and A2) followed by a unity gain difference amplifier (A3) with a variable gain output stage (A4).

Common-mode voltage is derived for active guard drive to improve system common-mode rejection. Two-pole, low pass filtering can easily be implemented on the output stage to reduce noise bandwidth and improve system signal-to-noise operation. A latch function is provided to inhibit gain changes while the digital gain control input is changed.

Burr-Brown's instrumentation grade monolithic operational amplifiers, high stability precision thin-film resistor networks and advanced laser trimming techniques are used by the 3606 to achieve a performance, size and cost combination never before achieved in a PGIA. It is available in a 32-pin dual-in-line package in either ceramic or metal (hermetic) configurations.

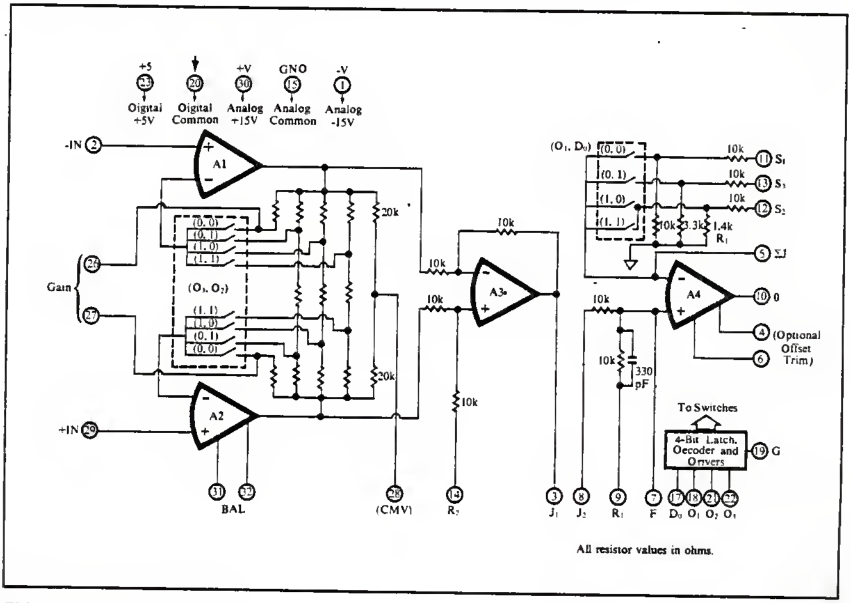


FIGURE 1. Simplified Schematic

ELECTRICAL SPECIFICATIONS

All specifications typical at 25°C unless otherwise noted.

PARAMETER	Conditions	3606A(7)			3606B(7)			Units
		MIN	TYP	MAX	MIN	TYP	MAX	
GAIN, G(1)								
Inaccuracy	G = 1 to 1024, I_o = 1mA		±0.02	±0.05		±0.01	±0.02	%
Nonlinearity (2)	G = 1 to 16		0.001	0.002		•	•	%(5)
	G = 32 to 128		0.003	0.004		•	•	%
	G = 256 to 1024		0.005	0.01		•	•	%
Drift vs Temperature	G = 1 to 1024		±5	±10		•	•	ppm/°C
vs Time	G = 1 to 1024		±0.01			•		%/1000 hrs.
RATED OUTPUT								
Voltage	I_o = ±5mA	±10	±12		•	•		V
Current	V_o = ±10V	±5	±10		•	•		mA
Impedance			0.05			•		Ω
INPUT CHARACTERISTICS								
Absolute Max Voltage	No damage			$\pm V_{CC}$			•	V
Common-mode Voltage Range	Linear operation	±10	±10.5		•	•		V
Differential Impedance			10 ∥ 3			•		10^9Ω ∥ pF
Common-mode Impedance			10 ∥ 3			•		10^9Ω ∥ pF
OFFSET VOLTAGE, RTO(3)								
Initial at 25°C (4)			±(0.02G +1)	±(0.04G +2)		±(0.01G +1)	±(0.02G +2)	mV
vs Temperature	-25°C to +85°C		(±0.0015G ±0.03G_2)	(±0.003G ±0.05G_2)		(±0.0005G ±0.01G_2)	(±0.001G ±0.02G_2)	mV/°C
vs Time			(±0.001G ±0.01G_2)			•		mV/mo
vs Supply			(±0.002G ±0.04G_2)			•		mV/V
vs Gain (5)	With trimming		±1	±2		•	•	mV
INPUT BIAS CURRENT								
Initial	25°C		±15	±50		±5	±20	nA
vs Temperature	-25°C to +85°C		±0.3			•		nA/°C
vs Supply Voltage			±0.1			•		nA/V
INPUT DIFFERENCE CURRENT								
Initial	25°C		±15	±50		±5	±20	nA
vs Temperature	-25°C to +85°C		±0.5			•		nA/°C
vs Supply Voltage			±0.1			•		nA/V
INPUT NOISE								
Voltage	R source ≤ 5kΩ							
0.01Hz to 10Hz	G = 1024		1.4			•		μV p-p
10Hz to 1kHz			1.0			•		μV rms
Current								
0.01Hz to 10Hz			70			•		nA p-p
10Hz to 1kHz			20			•		nA rms
COMMON-MODE REJECTION								
DC, 1kΩ Source Imbalance								
G = 1, 2		80	90		90	100		dB
G = 4 to 16		90	100		100	110		dB
G = 32 to 1024		100	114		110	114		dB
60Hz, 1kΩ Source Imbalance								
G = 1, 2		80	86		•	•		dB
G = 4 to 16		90	96		•	•		dB
G = 32 to 1024		100	106		•	•		dB
DYNAMIC RESPONSE								
±3dB Response	Small Signal							kHz
G = 1			100			•		kHz
G = 32 to 128			40			•		kHz
G = 256 to 1024			10			•		kHz
±1% Response	Small Signal							
G = 1			40			•		kHz
G = 32 to 128			8			•		kHz
G = 256 to 1024			3			•		kHz
Slew Rate	G = 1	0.2	0.5		•	•		V/μs
Settling Time	G = 128							
to 1%			75			•		μs
to 0.1%			100			•		μs
to 0.01%			200			•		μs
LOGIC VOLTAGES								
"0" Level(6)			0	+0.4		•	•	V
"1" Level(6)		+2.4	+5.0		•	•		V
Absolute Max	No damage			+7			•	V
ANALOG SUPPLY								
Rated Voltage			±15			•		VDC
Voltage Range, Derated Performance		±8		±18	•		•	VDC
Current, quiescent			±10	±20		•	•	mA

ELECTRICAL SPECIFICATIONS CONTINUED

All specifications typical at 25°C unless otherwise noted.

PARAMETER	Conditions	3606A (7)			3606B (7)			Units
		MIN	TYP	MAX	MIN	TYP	MAX	
OIGITAL SUPPLY								
Rated Voltage			+5			*		VDC
Voltage Range		+4.5		+5.5	*		*	VDC
Current, quiescent			10			*		mA
TEMPERATURE RANGE								
Specification		-25		+85	*		*	°C
Storage		-40		+100	*		*	°C
PRICE		"G" Pkg		"M" Pkg	"G" Pkg		"M" Pkg	
1 - 24		$78.00		$97.50	$100.00		$125.00	
25 - 99		$60.00		$75.00	$80.00		$100.00	
100's		$51.50		$64.00	$70.00		$87.50	

NOTES:

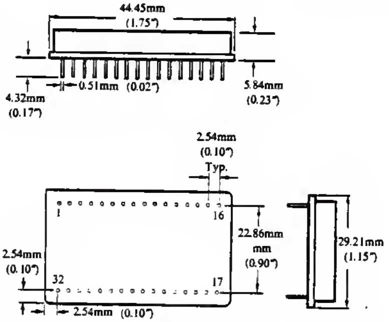
*Specifications same as 3606A.

1. $G = G_1 \cdot G_2$
2. Nonlinearity is the maximum peak deviation from the best straightline as a percent of full scale peak-to-peak output.
3. "RTO" = Referred To Output. May be referred to input by dividing by gain G.
4. May be adjusted to zero.
5. Trimmed according to Figure 8.
6. All digital inputa are 1 TTL unit load.
7. Specify 3606AG or 3606BG for ceramic package and 3606AM or 3606BM for metal package - see below.

The information in this publication has been carefully checked and is believed to be reliable; however, no responsibility is assumed for possible inaccuracies or omissions. Prices and specifications are subject to change without notice. No patent rights are granted to any of the circuits described herein.

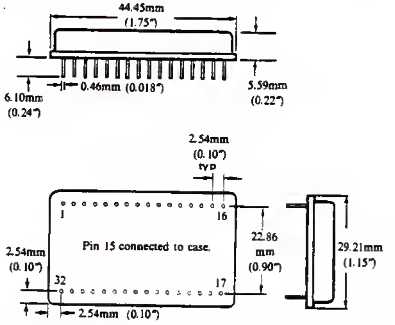
MECHANICAL

"G" PACKAGE



PINS: Pin material and plating composition conform to Method 2003 (solderability) of Mil-Std-883 (except paragraph 3.2). CASE: Ceramic MATING CONNECTOR: 2302MC - Set of two 16 pin strips. WEIGHT: 13 grams (0.46 oz.).

"M" PACKAGE



PINS: Pin material and plating composition conform to Method 2003 (solderability) of Mil-Std-883 (except paragraph 3.2). CASE: Metal MATING CONNECTOR: 2302MC - Set of two 16 pin strips. WEIGHT: 13 grams (0.46 oz.).

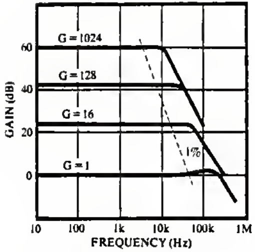
PIN DESIGNATIONS

PIN NO.	OESIG.	FUNCTION	PIN NO.	OESIG.	FUNCTION
1	-V	-15V Analog Supply	17	D_0	Oigital Input, LSB
2	-IN	Inverting Input	18	D_1	Oigital Input, next LSB
3	J_1	Output of A_3	19	G	Latch
4	(None)	Optional A_4 Offset Trim	20	⏚	Oigital Common
5	ΣJ	Summing Junction of A_4	21	O_2	Oigital Input, next MSB
6	(None)	Optional A_4 Offset Trim	22	O_3	Oigital Input, MSB
7	F	Low Pass Filter Pin	23	+5	+5V Oigital Supply
8	J_2	Input to A_4	24	(None)	No Internal Connection
9	R_1	Output Reference	25	(None)	No Internal Connection
10	O	Output	26	Gain	Optional External Gain
11	S_1	Sense G = 1	27	Gain	Optional External Gain
12	S_2	Sense G = 4	28	(None)	Input CMV
13	S_3	Sense G = 2	29	+IN	Noninverting Input
14	R_2	Output Reference	30	+V	+15V Analog Supply
15	GND	Analog Common	31	BAL	Optional Input Stage
16	(None)	No Internal Connection	32	BAL	Offset Null

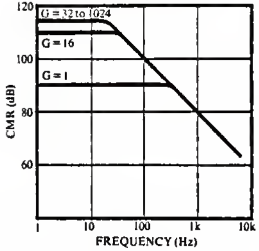
TYPICAL PERFORMANCE CURVES

All specifications typical at 25°C unless otherwise noted.

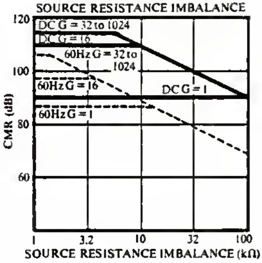
SMALL SIGNAL FREQUENCY RESPONSE



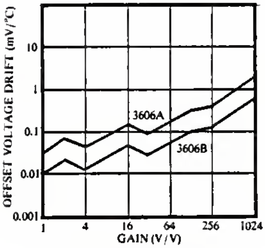
COMMON-MOOE REJECTION VS FREQUENCY



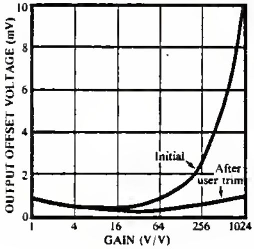
COMMON-MOOE REJECTION VS SOURCE RESISTANCE IMBALANCE



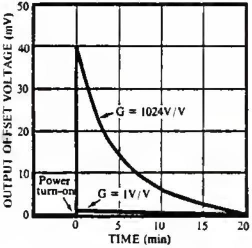
OUTPUT OFFSET VOLTAGE DRIFT VS GAIN



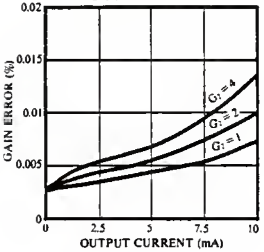
OUTPUT OFFSET VOLTAGE VS GAIN



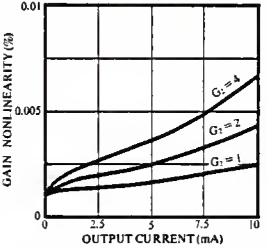
WARM-UP RESPONSE



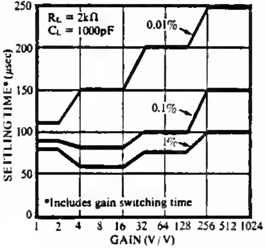
OUTPUT STAGE GAIN ERROR VS OUTPUT CURRENT



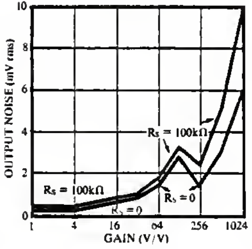
OUTPUT STAGE GAIN NONLINEARITY VS OUTPUT CURRENT



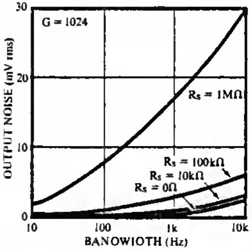
SETTLING TIME VS GAIN



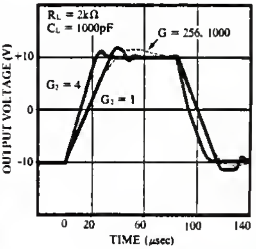
WIOEBANO OUTPUT NOISE VS GAIN



OUTPUT NOISE VS BANOWIOTH



STEP RESPONSE



INSTALLATION AND OPERATING INSTRUCTIONS

POWER SUPPLY CONNECTIONS

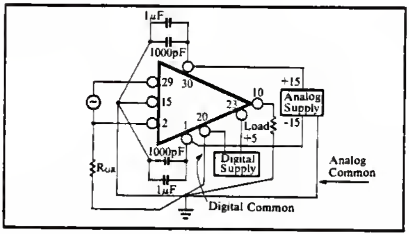


FIGURE 2. Power Supply and Ground Connections

Figure 2 shows the proper analog and digital power supply connections. The analog supplies should be decoupled with 1μF tantalum and 1000pF ceramic capacitors as close to the amplifier as possible. Because the amplifier is direct-coupled it must have a ground return path for the bias currents associated with the amplifier inputs at pins 2 and 29. If the ground return path is not inherent in the signal source (floating source) it must be provided externally. The ground return resistance (R_{GR}) should be kept as low as practical. An upper limit of approximately 50MΩ is established by the input bias currents of the amplifier and its common-mode voltage.

SIGNAL CONNECTIONS

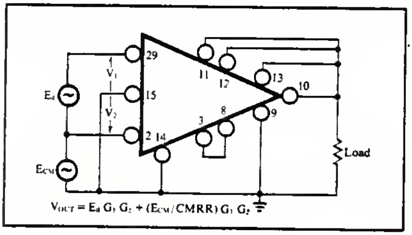


FIGURE 3. Basic Signal Connections

Basic signal connections are shown in Figure 3. The connection to pin 14 completes the difference amplifier of A_3 (see Figure 1). The 3 to 8 jumper connects the output stage. The pin 9 connection provides a divide-by-two attenuator for the A_4 stage. This is necessary to limit the signal on the output stage switches to maintain signal linearity. The pin 11, 12 and 13 connections to pin 10 close the feedback loop around A_4.

In the equation shown in Figure 3, G_1 is the input stage gain and G_2 is the output stage gain. CMRR is the common-mode rejection ratio [CMR (in dB) = 20 log CMRR (in V/V)]. Common-mode voltage shown as E_{CM} is actually the average of the two voltages appearing at the two inputs (pins 29 and 2) with respect to pin 15 (V_1 and V_2).

GAIN SETTING

Gain is determined by a 4-bit digital word applied to the input D_0 through D_3 (see Figure 1). Pin 19 provides a latch function for the inputs. When pin 19 is a logic 0, changes on the D_0 through D_3 inputs are inhibited. Pin 19 should be at +5V if the latch is not used.

A gain state truth table is shown in Table I. Gains are determined by the resistor networks shown in Figure 1. For the state D_3, D_2 = 0, 0, the input stage gain is a function of the gain setting resistor R_G connected between pins 26 and 27. If gains of 1, 2 and 4 are desired, no connection should be made to pins 26 and 27 and the resistance across these pins should be kept high with respect to 40kΩ (> 400MΩ).

Gain accuracy is established by laser-trimming the thin-film resistor networks during assembly. No external, user trimming is required.

OUTPUT OFFSET

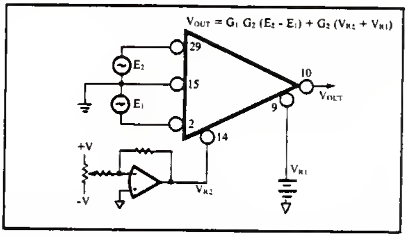


FIGURE 4. Output Offsetting

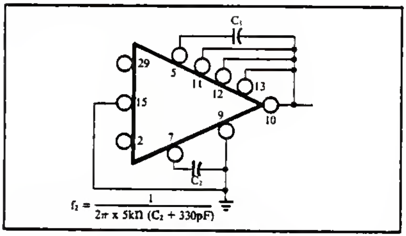
Output offset may be varied by either of two methods shown in Figure 4. Sources at pin 9 and pin 14 apply voltages to the noninverting inputs of A_4 and A_3 respectively (see Figure 1). Since the output stage gain occurs after these points, the output voltage bias established with V_{R1} and V_{R2} will vary with the output gain, G_2. Sources connected at pins 9 and 14 must have resistances low with respect to 10kΩ in order not to disturb gain accuracy and common-mode rejection.

Digital Inputs (G_1)		(G_2)		G_1 (A_1 and A_2) (Pins 2 & 29 to 3)	G_2 (A_4) (Pin 8 to Pin 10)	$G_1 \cdot G_2$ (R_G* = ∞)	$G_1 \cdot G_2$ (R_G* ≠ ∞)
D_3	D_2	D_1	D_0				
0	0	0	0	1 + 40k/R_G	1	1	1(1 + 40k/R_G)
0	0	0	1		2	2	2(1 + 40k/R_G)
0	0	1	0		4	4	4(1 + 40k/R_G)
0	0	1	1		4	4	4(1 + 40k/R_G)
0	1	0	0	4	1	4	4
0	1	0	1		2	8	8
0	1	1	0		4	16	16
0	1	1	1		4	16	16
1	0	0	0	32	1	32	32
1	0	0	1		2	64	64
1	0	1	0		4	128	128
1	0	1	1		4	128	128
1	1	0	0	256	1	256	256
1	1	0	1		2	512	512
1	1	1	0		4	1024	1024
1	1	1	1		4	1024	1024

*R_G connected between pins 26 and 27.

TABLE I. Gain State Truth Table

LOW PASS FILTER



$$f_2 = \frac{1}{2\pi \times 5k\Omega\ (C_2 + 330pF)}$$

FIGURE 5. Low Pass Filter Connections

For low frequency signals, system performance may be improved by reducing noise bandwidth in the amplifier. This may be accomplished with the addition of one or two external capacitors as shown in Figure 5. C_2 is connected to a 10k/10k attenuator and C_1 is connected as a feedback element across A4 (see Figures 1 and 5). The transfer function is:

$$\frac{V_o}{V_{IN}} = \left[\frac{10 \times 10^3}{100 \times 10^6\ S\ (C_2 + 330 \times 10^{-12}) + 20 \times 10^3}\right]\left[1 + \frac{10 \times 10^3}{10 \times 10^3\ R_1\ SC_1 + R_1}\right]$$

The first term is a first order filter. The second term is more complex. R_1 varies with the output stage gain - 1.4k for $G_2 = 4$ (see Figure 1). The "1 + ..." nature of the transfer function prevents a true first order filter rolloff.

For most applications, tbe first order low pass filter obtained by C_2 provides sufficient filtering. The value of C_2 required for a desired cutoff frequency (f_2 in Hz) is obtained by the equation shown in Figure 5.

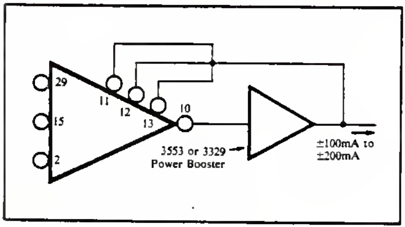


FIGURE 6. Output Current Booster

LARGER OUTPUT CURRENT

The output current rating of the 3606 is a minimum of ±5mA. The linearity of the gain is affected by output current. See TYPICAL PERFORMANCE CURVES. Optimum linearity is achieved with $I_o \leqslant 1mA$. $I_o \leqslant 5mA$ is acceptable. Above 5mA it may be desirable to use a power or current booster as shown in Figure 6. Burr-Brown's 3329 will provide ±100mA output while Burr-Brown's 3553 will supply ±200mA. When either booster is placed inside the feedback loop as shown, the booster's offset voltage produces no significant errors since it is divided by the open loop gain of the output stage.

GUARD DRIVE CONNECTIONS

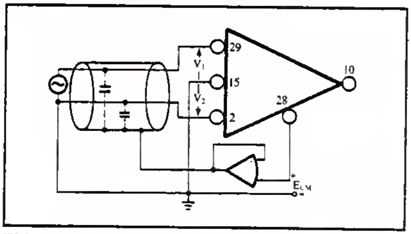


FIGURE 7. Guard Drive Connections

Use of the guard drive connection shown in Figure 7 can improve system common-mode rejection when the distributed capacitance of the input lines is significant. The common-mode voltage which appears on the input lines and on pins 29 and 2 is computed by the 3606 $[(V_1 + V_2)/2]$ and appears at pin 28. It is then fed back to the shield so that the voltage across the distributed capacitances is minimized. This reduces the common-mode current and improves common-mode rejection. The operational amplifier in the voltage follower configuration is used to supply more current than can be obtained from the 20k resistors connected internally to pin 28 (see Figure 1).

OFFSET TRIM

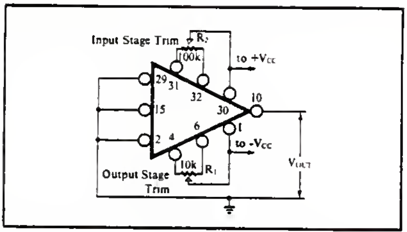


FIGURE 8. Optional Offset Trim

Offset voltages of the 3606 are reduced hy laser-trimming during assemhly. This reduces the initial offset voltage and the offset voltage change with gain change to levels that are acceptahle for most applications. For more critical applications the offset voltages can he externally nulled to zero. The following steps should be followed (see Figure 8).

1. Adjust hoth R_1 and R_2 to mid range
2. Set the gain to minimum (1V/V)
3. Adjust R_1 to make V_{OUT} equal zero
4. Set the gain to maximum (1024 V/V)
5. Adjust R_2 to make V_{OUT} equal zero

By using this technique, the change in output offset voltage caused hy a gain change of 1V/V to 1024V/V may be reduced to, typically 1mV instead of 10mV with no external trimming. Trimming may cause the offset voltage drift vs temperature to increase slightly.

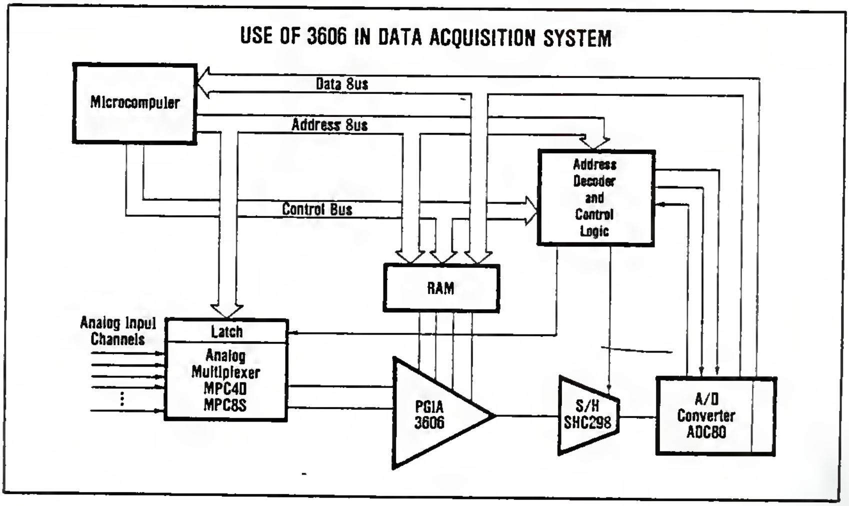
APPLICATIONS

A typical application of 3606 in a microcomputer hased data acquisition system is shown in the block diagram helow.

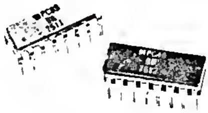
The purpose of this system is to be ahle to acquire data from a specific analog input channel, suitably condition it (amplify it and convert it to digital form) and store it or transmit it for further processing.

Initially the Microcomputer loads the RAM (random access memory) with the required coding for various desired gains via Data Bus. The coding associates the gain state truth tahle for 3606 with corresponding address locations in the computer memory. So when the computer puts out an instruction to multiplex a specific analog input channel through the multiplexer via the Address Bus, the RAM also receives the same address information and puts out corresponding gain code to the PGIA 3606. The 3606 amplifies the multiplexed signal hy the programmed gain value, and outputs it to S/H (sample and hold). The S/H holds the output value when it receives the control signal from the computer and the A/D converts it and outputs it to the computer via the Data Bus under computer control.

The PGIA 3606 allows the system user to modify and reprogram gain values for different analog input channels merely by changing the software computer program. Since different dedicated instruments are not required for various input channels, the PGIA also saves space and overall system costs.


USE OF 3606 IN DATA ACQUISITION SYSTEM

BURR-BROWN®



MPC4D
MPC8S

CMOS ANALOG MULTIPLEXERS

FEATURES

- LOW POWER CONSUMPTION
  CMOS analog switches
  15mW at 100kHz
- PROTECTS SIGNAL SOURCES
  Break-before-make switching
- HIGH THROUGHPUT RATE
- RELIABLE MONOLITHIC CONSTRUCTION

DESCRIPTION

The MPC8S is a single-ended monolithic 8-channel analog multiplexer and the MPC4D is a monolithic 4-channel differential input output multiplexer. The digital and analog inputs are protected from overvoltage inputs that exceed either power supply. These CMOS devices feature self-contained binary channel address decoding and are compatible with DTL, TTL, or CMOS input levels. Channel interaction is eliminated during overvoltage conditions and also in the event of a power loss. They are packaged in a 16-pin DIP and dissipate typically 7.5mW.

DESCRIPTION

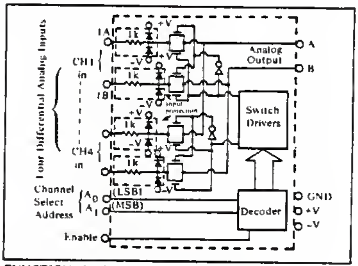
The MPC8S is a single-ended monolithic 8-channel analog multiplexer and the MPC4D is a monolithic differential input output channel analog multiplexer constructed with failure-protected CMOS devices. Transfer accuracies of better than 0.01% can be aehieved at sampling rates up to 200kHz from signal sources of up to ±10 volts amplitude.

These DTL TTL CMOS compatible devices feature self-contained binary channel address decoding. An ENABLE line is also made available which allows the user to individually enable an 8-channel group (MPC8S) or a 4-channel group (MPC4D) facilitating channel expansion in either single-mode or multitiered matrix configurations.

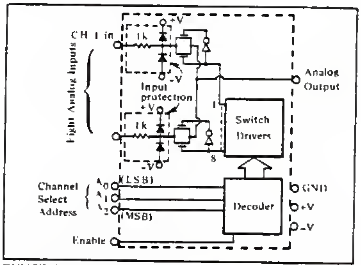
Digital and analog inputs are failure proteeted from either overvoltages that exceed the power supplies or from the loss of power.

High quality processing is employed to produce CMOS FET analog channel switches which have low leakage current, high OFF resistanee, low feedthrough eapacitance and fast settling time.

These devices are housed in compaet 16-pin dual-in-line packages, and are specified for operation over a 0°C to +75°C temperature range. They are pin and package compatible with the 508 509 series.



FUNCTIONAL BLOCK OIAGRAM – MPC4D



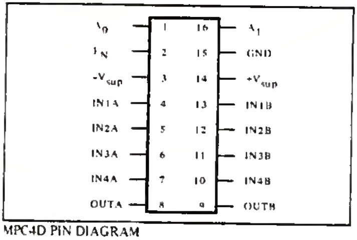
FUNCTIONAL BLOCK DIAGRAM – MPC8S

A_1	A_0	E_N	"On" Switch Pair
X	X	L	None
L	L	H	1
L	H	H	2
H	L	H	3
H	H	H	4

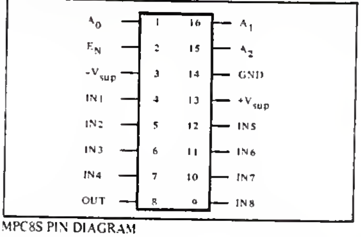
TRUTH TABLE –MPC4D

A_2	A_1	A_0	E_N	On Switch
X	X	X	L	None
L	L	L	H	1
L	L	H	H	2
L	H	L	H	3
L	H	H	H	4
H	L	L	H	5
H	L	H	H	6
H	H	L	H	7
H	H	H	H	8

TRUTH TABLE – MPC8S



MPC4D PIN DIAGRAM



MPC8S PIN DIAGRAM

SPECIFICATIONS

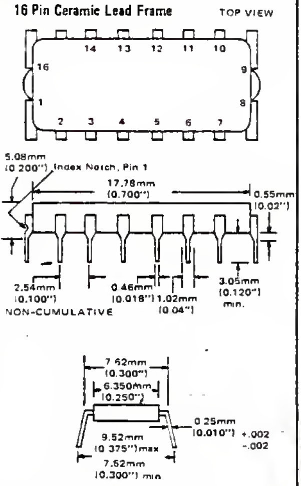
Typical for following conditions: V + = +15V, V - = -15V, $R_{source} \leq 1000\ \Omega$, $T_A = 25°C$ unless otherwise noted.

ELECTRICAL

MODELS	MPC8S	MPC4D	Units
INPUT			
ANALOG INPUT			
Voltage Range	±15		V
Maximum Overvoltage	+V supply +20		V
	-V supply -20		V
Current at Maximum Overvoltage per channel[1]	±18		mA
Number of Input Channels			
Single-ended	8		
Differential		4	
ON Characteristics			
ON Resistance (R_{ON})			
Typical	1.5		kΩ
Maximum	1.8		kΩ
R_{ON} Drift vs. Temperature (0°C to +75°C)	0.25		%/°C
R_{ON} Mismatch			
Channel-to-channel	50	50	Ω
Differential	N/A	50	Ω
Input Leakage (I_I)	0.1		nA
Input Leakage Drift	See Figure 9		
OFF Characteristics			
OFF Resistance	10^{11}		Ω
Output Leakage (All channels disabled)	0.2		nA
Input Leakage[6]	0.02		nA
Leakage Drift	See Figure 9		
Output Leakage with Input Overvoltage			
of +35V	1		nA
of -35V	1		μA
DIGITAL INPUTS			
Logic "0" (V_L)[1&2]	-V supply ≤ V_L < 0.8 at 1 nA		V
Logic "1" (V_H)[1&2]	+4V ≤ V_H ≤ + V supply at 1 nA		V
Channel Select	3 bit binary code - one of eight	2 bit binary code - one of four	
Enable	Logic "0" (low) disables all channels Logic "1" (high) enables channel select to turn on selected channel		
POWER REQUIREMENTS			
Rated Power Supply Voltages	±15		V
Supply Range			
+Supply	+10 to ±20		V
-Supply	-10 to -20		V
Supply Drain			
At 1 MHz Switching Speed	+4, -2		mA
AT 100 kHz Switching Speed	±0.5		mA
Typical Power Consumption			
DC to 10 kHz	7.5		mW
DYNAMIC CHARACTERISTICS			
Gain Error (20 MΩ load) maximum	0.01		%
Crosstalk[3]	0.005		% of OFF channel signal
Settling Time[4]			
To ±2mV (±0.01%)	5		μs
To ±20mV (±0.10%)	2		μs
Common-mode Rejection (minimum)	N/A	120	dB
Switching Time			
Turn ON	0.5		μs
Turn OFF	0.3		μs
Recovery Time from Input Overvoltage Pulse of 35V for 100 μsec			
To 0.01%	150		μs
To 0.10%	15		μs
OUTPUT			
Voltage Range	±15		V
Capacitance to Ground	25	12[5]	pF
Capacitance Mismatch	N/A	±10	%
TEMPERATURE			
Specification	0 to +75		°C
Storage	-65 to +150		°C

MPC4D AND MPC8S

16 Pin Ceramic Lead Frame



NOTES:

1. Total power dissipation due to input overvoltage current flowing in the input protection circuitry must be limited to 0.75 watt for both (a) normal operation with power supplies turned on or (b) during a fault condition when the supplies are shorted to ground.
2. Maximum overvoltage is ±Vsupply ±4 volts at ±15 mA.
3. 20 volt peak-to-peak 1000 Hz sinewave; R_{SOURCE} = 1000Ω, same signal on all unused channels.
4. For 20 volts between switched channels, R_{SOURCE} = 1000Ω. See Figure 5 for settling time vs. source impedance (R_S).
5. From each side of MPC4D to ground.
6. Leakage measurement made with all OFF channel inputs fed in parallel to +20 volts.

DISCUSSION OF PERFORMANCE

Static Transfer Accuracy

The static or DC transfer accuracy of transmitting the multiplexer input voltage to the output depends on the channel ON resistance (R_{ON}), the load impedance, the source impedance, the load bias current and the multiplexer leakage current.

SINGLE-ENDED MULTIPLEXER STATIC ACCURACY

The major contributors to static transfer accuracy for single-ended multiplexers are:

Source resistance loading error
Multiplexer ON resistance error
DC offset error caused by both load bias current and multiplexer leakage current.

Resistive Loading Errors

The source and load impedances will determine the input resistive loading errors. To minimize these errors:

- Keep loading impedance as high as possible. This minimizes the resistive loading effects of the source resistance and multiplexer ON resistance. As a guideline, load impedances of 10^8 ohms or greater will keep resistive loading errors to 0.002% or less for 1000 ohm source impedances. A 10^6 ohm load impedance will increase source loading error to 0.2% or more.
- Use sources with impedances as low as possible. A 1000 ohm source resistance will present less than 0.001% loading error and 10,000 ohm source resistance will increase source loading error to 0.01% with a 10^8 ohm load impedance.

Input resistive loading errors are determined by the following relationship: (see Figure 1)

Source and Multiplexer Resistive Loading Error

$$\epsilon_{(R_S + R_{ON})} = \frac{R_S + R_{ON}}{R_S + R_{ON} + R_L} \times 100\%$$

where R_S = source resistance
R_L = load resistance
R_{ON} = multiplaxer ON resistance

INPUT OFFSET VOLTAGE

Bias current generates an input OFFSET voltage as result of tbe IR drop across the multiplexer ON resistance and source resistance. A load bias current of 10 nanoamperes will generate an offset voltage of 20μV if a 1000 ohm source is used, and 200μV if a 10,000 ohm source is used. In general, for the MPC8S, the OFFSET voltage at the output is determined by:

$$V_{OFFSET} = (I_b + I_L)(R_{ON} + R_S)$$

where I_b = Bias current of device multiplexer is driving
I_L = Multiplexer leakage current
R_{ON} = Multiplexer ON resistance
R_{SOURCE} = Source resistance

DIFFERENTIAL MULTIPLEXER STATIC ACCURACY

Static accuracy errors in a differential multiplexer are difficult to control, especially when it is used for multiplexing low-level signals with full scale ranges of 10 to 100 millivolts.

The matching properties of the multiplexer, source and output load play a very important part in determining the transfer accuracy of the multiplexer. The source impedance unbalance, common-mode impedance, load bias current mismatch, load differential impedance mismatch, and common-mode impedance of the load all contribute errors to the multiplexer. The multiplexer ON resistance mismatch, leakage current mismatch and ON resistance also contribute to differential errors.

The effects of these errors can be minimized by following the general guidelines described in this section, especially for low level multiplexing applications. Refer to Figure 2.

LOAD (OUTPUT DEVICE) CHARACTERISTICS

- Use devices with very low bias current. Generaly, FET input amplifiers should be used for low level signals less than 50mV RSR. Low bias current bipolar input amplifiers are acceptable for signal ranges higher than 50mV FSR. Bias current matching will determine the input offset.
- The system DC common-mode rejection (CMR) can never be better than the combined CMR of the multiplexer and driven load. System CMR will be less than the device which has the lower CMR figure.
- Load impedances, differential and common-mode, should be 10^{10} ohms or higher.

SOURCE CHARACTERISTICS

- The source impedance unbalance will produce offset, common-mode and channel-to-channel gain-scatter errors. Use sources which do not have large impedance unbalances if at all possible.
- Keep source impedances as low as possible to minimize resistive loading errors.
- Minimize ground loops. If signal lines are shielded, ground all shields to a common point at the system analog common.

If the MPC4D is used for multiplexing high-level signals of ±1 volt to ±10 volts full scale ranges, the foregoing precautions should still be taken, but the parameters are not as critical as for low-level signal applications.

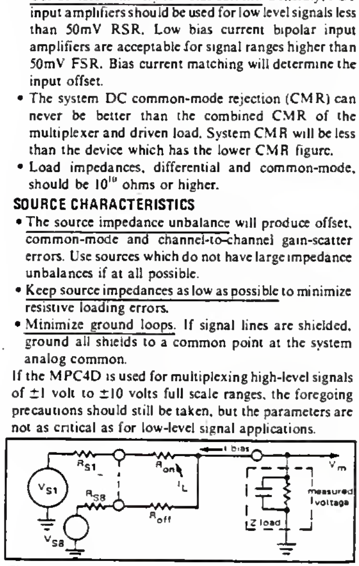


FIGURE 1. MPC8S Static Accuracy Equivalent Circuit.

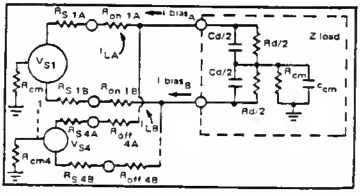


FIGURE 2: MPC4D Static Accuracy Equivalent Circuit.

SETTLING TIME

The gate-to-source and gate-to-drain capacitance of the CMOS FET switches, the RC time constants of the source and the load determine the settling time of the multiplexer.

Governed by the charge transfer relation $i = C\frac{dV}{dt}$, the charge currents transferred to both load and source by the analog switches are determined by the amplitude and rise time of the signal driving the CMOS FET switches and the gate-to-drain and gate-to-source junction capacitances as shown in Figure 3 and 4. Using this relationship, one can see that the amplitude of the switching transients seen at the source and load decrease proportionally as the capacitance of the load and source increase. The tradeoff for reduced switching transient amplitude is increased settling time. If effect, the amplitude of the transients seen at the source and load are:

$$dV_{load} = \frac{i}{C}\,dt$$

where $i = C\frac{dV}{dt}$ of the CMOS FET switches

C = load or source capacitance

The source must then redistribute this charge, and the effect of source resistance on settling time is shown in Figure 5. This graph shows the settling time for a 20 volt step change on the input. The settling time for smaller step changes on the input will be less than that shown in Figure 5.

SWITCHING TIME

This is the time required for the CMOS FET to turn ON after a new digital code has been applied to the Channel Address inputs. It is measured from the 50 percent point of the address input signal to the 90 percent point of the analog signal seen at the output for a 10 volt signal change between channels.

CROSSTALK

Crosstalk is the amount of signal feedthrough from the three (MPC4D) or seven (MPC8S) OFF channels appearing at the multiplexer output. Crosstalk is caused by the voltage divider effect of the OFF channel OFF resistance and junction capacitances in series with the R_{DS} and R_{SOURCE} impedances of the ON channel. Crosstalk is measured with a 20 volt pk-pk 1000 Hertz sine wave applied to all OFF channels. The crosstalk for these multiplexers is shown in Figure 6.

COMMON-MOOE REJECTION (MPC40 ONLY)

The matching properties of the load, multiplexer and source affect the common-mode rejection (CMR) capability of a differentially multiplexed system. CMR is the ability of the multiplexer and input amplifier to reject signals that are common to both inputs, and to pass on only the signal difference to the output. For the MPC4D, protection is provided for common-mode signals of ±20 volts above the power supply voltages with no damage to the analog switches.

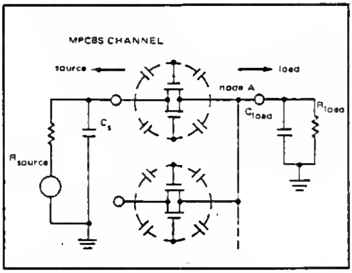


FIGURE 3: Settling Time Effects – MPC8S

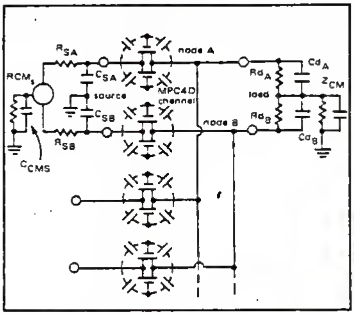


FIGURE 4: Settling & Common-Mode Effects – MPC4D.

The CMR of the MPC4D and Burr-Brown's model 3660 Instrumentation Amplifier is 120 dB at DC to 1 Hz with a 6 dB/octave rolloff to 70 dB at 1000 Hz. This measurement of CMR is shown in Figure 8 and is made with a Burr-Brown model 3660 Instrumentation Amplifier connected for a gain of 1000 and with source unbalance of 1kΩ and no unbalance.

Factors which will degrade multiplexer and system DC CMR are:

- Amplifier bias current and differential impedance mismatch
- Load impedance mismatch
- Multiplexer impedance and leakage current mismatch
- Load and source common-mode impedance

AC CMR rolloff is determined by the amount of common-mode capacitances (absolute and mismatch) from each signal line to ground. Larger capacitances will limit CMR at higher frequencies; thus, if good CMR is desired at higher frequencies, the common-mode capacitances and unbalance of signal lines and multiplexer to amplifier wiring must be minimized. Use twisted-shielded pair signal lines wherever possible.

TYPICAL PERFORMANCE CURVES

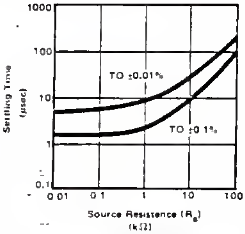


FIGURE 5. Settling time vs source resistance for 20 volt step change.

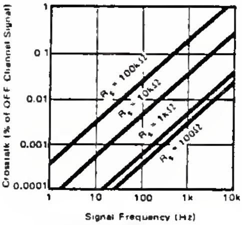


FIGURE 6. Crosstalk vs signal frequency.

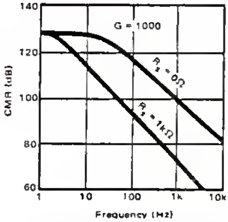


FIGURE 7. CMR vs frequency for Model 3660 IA and MPC4D (G = 1000).

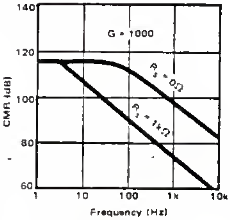


FIGURE 8. Combined CMR vs frequency for Model 3670 IA (G = 1000) and MPC4D.

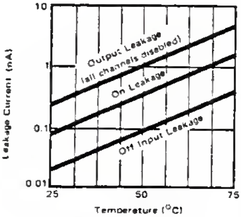


FIGURE 9. Leakage current vs temperature.

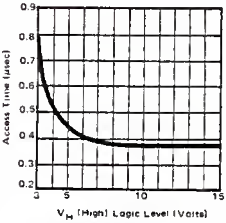


FIGURE 10. Access time vs logic level (high).

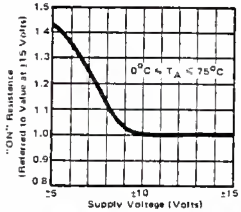


FIGURE 11. Normalized "ON" resistance vs supply voltage.

OPERATION & INSTALLATION INSTRUCTIONS

The ENABLE input, pin 2, is included for expansion of the number of channels on a single node as illustrated in Figure 12. With ENABLE line at a logic 1, the channel is selected by the 2 bit (MPC4D) or 3 bit (MPC8S) Channel Select Address (shown in the Truth Tables on page 9-4) If ENABLE is at logic 0, all channels are turned OFF, even if the Channel Address lines are active. If the ENABLE line is not to be used, simply tie it to +V supply.

If the +15 volt and or -15 volt supply voltage is absent or shorted to ground, the MPC4D and MPC8S multiplexers will not be damaged; however, some signal feedthrough to the output will occur. Total package power dissipation must not be exceeded (see Footnote 1, page 9-5).

For best settling speed, the input wiring and interconnections between multiplexer output and driven devices should be kept as short as possible. When driving the digital inputs from TTL, open collector output with pull-up resistors are recommended. See Figure 10 (access time).

To preserve common-mode rejection of the MPC4D, use twisted-shielded pair wire for signal lines and inter-tier connections and or multiplexer output lines. This will help common-mode capacitance balance and reduce stray signal pickup. If shields are used, all shields should be connected as closely as possible to system analog common or to the common-mode guard driver.

CHANNEL EXPANSION

SINGLE-ENDED MULTIPLEXER (MPC8S)

Up to 32 channels (4 multiplexers) can be connected to a single node, or up to 64 channels using 9 MPC8S multiplexers on a two-tiered structure as shown in Figure 12 and 13.

DIFFERENTIAL MULTIPLEXER (MPC4D)

Single or multi-tiered configurations can be used to expand multiplexer channel capacity up to 32 channels using a 32 x 1 or 16 channels using a 4 x 4 configuration.

SINGLE NODE EXPANSION

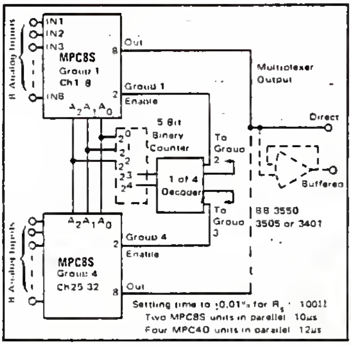
The 32 x 1 configuration is simply eight MPC4D units tied to a single node. Programming is accomplished with a 5 bit counter, using the 2 LSB's of the counter to control Channel Address inputs A₀ and A₁ and the 3 MSB's of the counter to drive a 1 of 8 decoder. The 1 of 8 decoder then is used to drive the ENABLE inputs (pin 2) of the MPC4D multiplexers.

TWO TIER EXPANSION

Using a 4 x 4 2-tier structure for expansion to 16 channels, the programming is simplified A 4-bit counter output does not require a 1 of 8 decoder. The 2 LSB's of the counter drive the A₀ and A₁ inputs of the four first tier multiplexers and the 2 MSB's of the counter are applied to the A₀ and A₁ inputs of the second tier multiplexer.

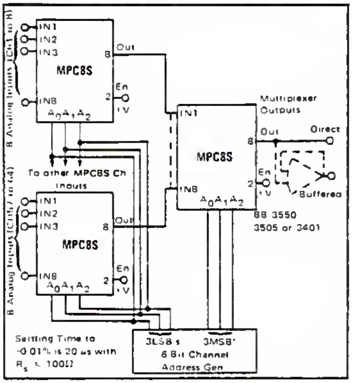
Single vs. Multi-Tiered Channel Expansion

In addition to reducing programming complexity, two-tier configuration offers the added advantages over single node expansion of reduced OFF channel current leakage (reduced OFFSET), better CMR, and a more reliable configuration if a channel should fail in the ON condition (short). Should a channel fail ON in the single node configuration, data cannot be taken from any channel, where as only one channel group is failed (4 or 8) in the multi-tiered configuration.



Settling time to ±0.01% for R_s · 100Ω
Two MPC8S units in parallel 10μs
Four MPC4D units in parallel 12μs

FIGURE 12. 32 Channel. Single-Tier Expansion.



Settling Time to ±0.01% is 20 μs with R_s ≤ 100Ω

FIGURE 13. Channel Expansion Up to 64 Channels Using 8X8 Two-Tiered Expansion.

OKI
semiconductor

MSM5832 MICROPROCESSOR REAL-TIME CLOCK/CALENDAR

GENERAL DESCRIPTION

The MSM5832 is a monolithic, metal-gate CMOS integrated circuit that functions as a real time clock/calendar for use in bus-oriented microprocessor applications. The on-chip 32,768 Hz crystal controlled oscillator time base is counted down to provide addressable 4-bit I/O data of SECONOS, MINUTES, HOURS, OAY-OF-WEEK, DATE, MONTH, and YEAR. Data access is controlled by 4-bit address, chip select, read, write and hold inputs. Other functions include 12H/24H format selection, leap year identification and manual ±30 second correction.

The MSM5832 normally operates from a 5 volt ±5% supply. Battery back-up operation down to 2.2 volts allows continuation of time keeping when main power is off. One test input facilitates rapid testing of the time keeping operations. The MSM5832 is offered in an 18-lead dual-in-line plastic (RS suffix) package.

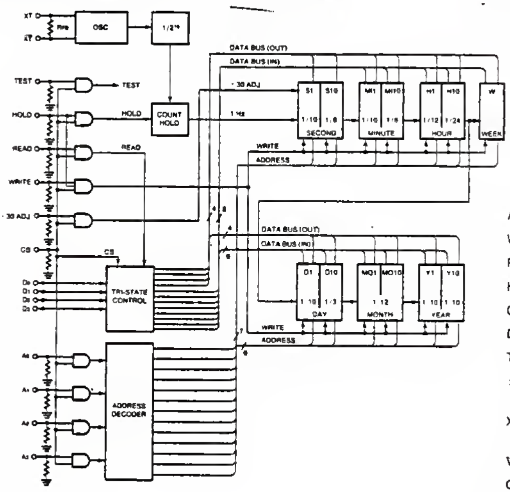
FEATURES

- Microprocessor bus-oriented
-

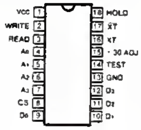
TIME	MONTH		DATE		YEAR		DAY OF WEEK
23:59:59	12	-	31	-	99	-	7

- 4-BIT DATA BUS
- 4-BIT AODRESS
- Raad, Wnte, Hold, Chip select inputs
- Interrupt signal outputs—1024, 1, 1/60, 1/3600 Hz
- 32.768 KHz crystal controlled operation
- Laap year register bit
- 12 or 24 hour format
- ±30 second error correction
- Single 5 volt power supply
- Back-up battery operation to VCC=2.2 V
- Low Power Dissipation
  90 μw Max. at VCC=3 V
  2.5 mw Max. at VCC=5 V
- High Density 300 mil 18-Pin Package

FUNCTIONAL BLOCK DIAGRAM



PIN CONFIGURATION



A_0 to A_3: Address Inputs

WRITE: Write Enable

READ: Read Enable

HOLD: Count Hold Enable

CS: Chip Select

D_0 to D_3: Data Input/Output

TEST: Test Input

±30 ADJ: ±30 Second Correction Input

XT & $\overline{XT}$: xtal oscillator connections

Vcc: +5 V Supply

GND: Ground

ICROPROCESSOR REAL-TIME CLOCK/CALENDAR

FUNCTION TABLE

FIGURE 1

ADDRESS INPUTS				INTERNAL COUNTER	DATA I/O				DATA LIMITS	NOTES
A_0	A_1	A_2	A_2		D_0	D_1	D_2	D_3		
0	0	0	0	S 1	•	•	•	•	0 ~ 9	S_1 or S_{10} are reset to zero irrespective of input data D_0~D_3 when write instruction is executed with address selection
1	0	0	0	S 10	•	•	•		0 ~ 5	
0	1	0	0	MI 1	•	•	•	•	0 ~ 9	
1	1	0	0	MI 10	•	•	•		0 ~ 5	
0	0	1	0	H 1	•	•	•	•	0 ~ 9	
1	0	1	0	H 10	•	•	†	†	0~1 / 0~2	D_2="1" for PM D_3="1" for 24 hour format D_2="0" for AM D_3="0" for 12 hour format
0	1	1	0	W	•	•	•		0 ~ 6	
1	1	1	0	D 1	•	•	•	•	0 ~ 9	
0	0	0	1	D 10	•	•	†		0 ~ 3	D_2 ="1" for 29 days in month 2 D_2 ="0" for 28 days in month 2 (2)
1	0	0	1	MO 1	•	•	•	•	0 ~ 9	
0	1	0	1	MO 10	•				'0 ~ 1	
1	1	0	1	Y 1	•	•	•	•	0 ~ 9	
0	0	1	1	Y 10	•	•	•	•	0 ~ 9	

(1) * data valid as "0" or "1"
blank does not exist (unrecognized during a write and held at "0" during a read)
† data bits used for AM/PM, 12/24 HOUR and leap year

(2) If D2 previously set to "1", upon completion of month 2 day 29, D2 will be internally reset to "0"

TYPICAL CHARACTERISTICS—Oscillator Frequency Deviations

Frequency Deviation vs Temperature

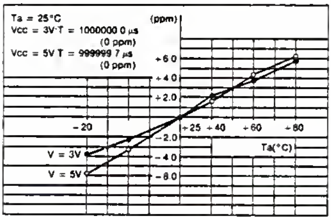


FIGURE 2

Frequency Deviation vs Supply Voltage

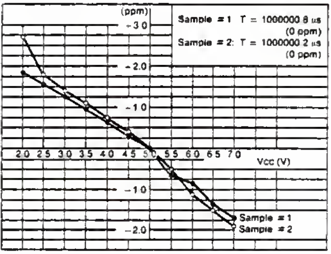


FIGURE 3

ABSOLUTE MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{CC}	−0.3 ~ 7.0	V
Input Voltage	V_{IN}	−0.3 ~ V_{CC} +0.3	V
Data I/O Voltage	V_D	−0.3 ~ V_{CC} +0.3	V
Storage Temperature	T_{stg}	−55 ~ 150	°C

Note: Stresses above those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or at any other condition above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

OPERATING CONDITIONS

Parameter	Symbol	Min.	Typ.	Max.	Unit	Conditions
Supply Voltage	V_{CC}	4.75	5	5.25	V	5V ± 5%
Standby Supply Voltage	V_{CCS}	2.2	5	7	V	
Input Signal Level	V_{IH}	3.6	5	V_{CC}	V	V_{CC} = 5V ± 5% Raspect to Gnd
	V_{IL}	−0.3	0	0.8	V	
Crystal Oscillator Freq.	f(XT)		32.768		KHz	
Operating Temperature	Ta	−30		+85	°C	

DC CHARACTERISTICS

(V_{CC} = 5V ± 5%; T_A = −30 to +85°C, unless otherwise noted)

Parameter	Symbol	Min.	Typ.	Max.	Unit	Conditions
Input Current (1)	I_{TH}	10	25	50	μA	V_{IN} = 5V
	I_{IL}	−1		1	μA	V_{IN} = 0V
Data I/O Laakage Current	I_{LD}	−1		1	μA	$V_{I/O}$ = 0 to V_{CC}, CS = "0"
Output Low Voltage	V_{OL}			0.4	V	Io = 1.6 ma, CS = "1", READ = "1"
Output Low Current	I_{OL}	1.6			mA	Vo = 0.4V, CS = "1", READ = "1"
Operating Supply Current	I_{CCS}			30	μA	V_{CC} = 3V, Ta = 25°C
	I_{CC}			500	μA	V_{CC} = 5V, Ta = 25°C

(1) XT, $\overline{XT}$ and D_0~D_3 excluded.

AC CHARACTERISTICS

CAPACITANCE

T_A = 25°C, f = 1 MHz

Parameter	Symbol	Min.	Typ.	Max.	Unit
Input/Output Capacitance	$C_{I/O}$			8	pF
Input Capacitance	C_{IN}			5	pF

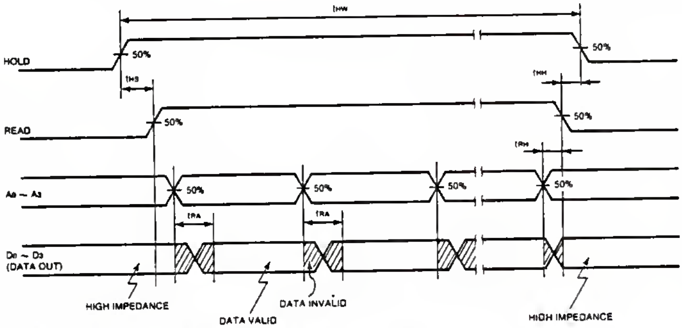
Note: This parameter is periodically sampled and not 100% tested.

READ CYCLE

(V_{CC} = 5V ± 5%; Ta = 25°C)

Parameter	Symbol	Min.	Typ.	Max.	Unit
HOLD Set-up Time	t_{HS}	150			μS
HOLD Hold Time	t_{HH}	0			μS
HOLD Pulse Width	t_{HW}			1	SEC
READ Hold Time	t_{RH}	0			μS
READ Access Time	t_{RA}			8	μS

READ CYCLE **FIGURE 4**



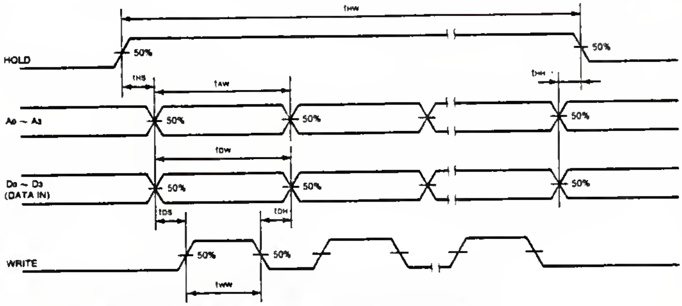
Notes: 1. A Read occurs during the overlap of a high CS and a high READ
2. Output Load: 1 TTL Gate, C_L = 50 pf and R_L = 4.7 KΩ
3. CS may be a permanent "1", or may be coincident with HOLD pulse

WRITE CYCLE

(Vcc = 5V ± 5%; Ta = 25°C)

Parameter	Symbol	Min.	Typ.	Max.	Unit
HOLD Set-up Time	t_{HS}	150			µS
HOLD Hold Time	t_{HH}	0			µS
HOLD Pulse Width	t_{HW}			1	SEC
ADDRESS Pulse Width	t_{AW}	1.7			µS
DATA Pulse Width	t_{DW}	1.7			µS
DATA Set-up Time	t_{DS}	0.5			µS
DATA Hold Time	t_{DH}	0.2			µS
WRITE Pulse Width	t_{WW}	1.0			µS

WRITE CYCLE **FIGURE 5**



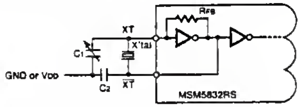
Notes: 1. A WRITE occurs during the overlap of a high CS, a high HOLD and a high WRITE
2. CS may be a permanent "1", or may be coincident with HOLD pulse

FUNCTIONAL DESCRIPTION

A block diagram ot the MSM5832 microprocessor real-time clock/calendar and s package connection diagram srs shown on the first page. Figure 9 illustrates s mathod of interfacing between the clock/calendar circuit and a micro processor. Figures 9, 10 and 11 illustrats alternative standby power supply circuits. A function table listing relationships between sddress inputs, data input/output and internal counter selection is shown in Figure 1. Unless otherwise indicated, the following descriptions ara based on the block diagram.

32.768 K Hz OSCILLATOR (pins 16 and 17): An internal inverting amplifier with feedback resistor, Rfb, is connected with a crystal snd two capacitors as *shown in Figure* 6 to torm a stabla, accurate oscillator—which serves as the precision time base of the circuit. Capacitors C1 and C2 in series provide the parallel load capacitanca required for precise tuning of ths quartz crystal. Typical oscillstor parformance as a function of ambient temperature and supply voltage is shown in Figures 2 and 3 respectively.

OSCILLATOR CIRCUIT FIGURE 6



$C_1 = C_2 = 15 \sim 30$ pF

CHIP SELECT (pin 8): Connecting CS input to Vcc enables all inputs and outputs. Unconnected—pull-down to GND is provided by an internal resistor—or connecting CS to GND will disable HOLD, WRITE, READ, ±30 ADJ, D0~D3, A0~D3 and TEST.
As shown in *Figure* 9 CS can be used to detect system power failure by connecting system power (+5V) to CS, so that when system power is on, all inputs and outputs will be enabled, and when system power is *off*, all inputs and outputs will be disabled. The threshold voltsge of CS is higher than all other inputs to insure correct operation of this function.

HOLD (pin 18): Switching this input to Vcc inhibits the internal 1Hz clock to the S1 counter. After the specified HOLD set-up time (150 μs), all counters will be in a static state, thus allowing error-free read or write operstions. So long as the HOLD pulsa width is less then 1 second, accuracy of the real time will be undisturbed. Pull-down to GND is provided by an internal resistor.

READ (pin 3): Reed function as *shown* in *Figure 4* is enabled when READ is switched to Vcc. Pull-down to GND is provided by an internal resistor.

WRITE (pin 2): Write function as *shown* in *Figure 5* is enabled when WRITE is switched to Vcc. Pull-down to GND is provided by an internal resistor.

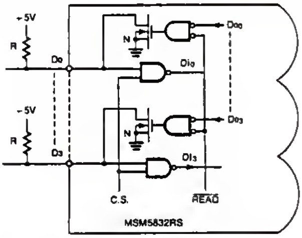
±30 ADJ (Pin 15): Momantarily connecting this input to Vcc (>31.25 ms) will reset seconds (S1, S10 counters and $2^{11} \sim 2^{15}$ frequency dividers) to 00; if seconds wera 30 or more, one minute is added to the minutes (MI 1 counter) and if seconds were less than 30, the minutes ara unchanged. Pull-down to GND is provided by an internal resistor.

A0 ~ A3 (pins 4 ~ 7): Address inputs, used to select internal counters tor read/writs operstions (see tunction table—*Figure 1*). A "1" is defined as Vcc; a "0" is GND. Pull-down to GND is provided by internal resistors.

D0 ~ D3 (pins 9 ~ 12): Data Inputs/Outputs, two-way bus lines controlled by READ and WRITE inputs. *As shown in Figure 7* external pull-up resistors of 4.7K or higher are required by the open-drein N-channel MOS outputs. D3 is the MSB; D0 is the LSB.

TEST (pin 14): Normally this input is unconnected—pull-down to GND is provided by an internal resiator—or connected to GND. With CS at Vcc, pulses to Vcc on the TEST input will directly clock ths S1, MI10, W, D1 and Y1 counters, depending on which counter is addressed (W and D1 are selected by D1 address in this mods only). Roll-over to next countsr is anabled in this mode.

DATA I/O CIRCUIT FIGURE 7



REFERENCE SIGNAL OUTPUT
Reference signals are available as outputs on D0 ~D3 if CS, READ and A0 ~ A3 are at Vcc. Refer to *Figure 8* for specifics. *As shown in Figure 9* these signals may be used to generste interrupts for the microprocessor.

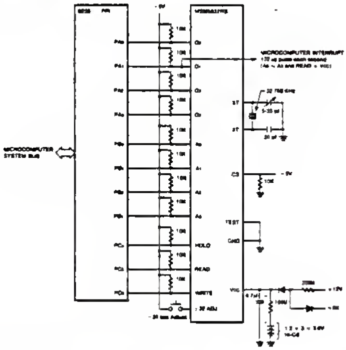
REFERENCE SIGNAL OUTPUTS FIGURE 8

CONDITIONS	OUTPUT	REFERENCE FREQUENCY	PULSE WIDTH
HOLD = L	D0 (1)	1024 Hz	duty 50%
READ = H	D1	1 Hz	122.1 μs
C.S. = H	D2	1/60 Hz	122.1 μs
A0 ~ A3 = H	D3	1/3600 Hz	122.1 μs

(1) 1024 Hz signal at D0 not dependent on HOLD input level

TYPICAL APPLICATION—Use with Programmable Peripheral Interface (PPI)

FIGURE 9



TYPICAL APPLICATIONS—Alternative Standby Power Supply Circuits

FIGURE 10

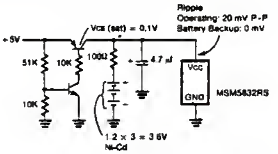
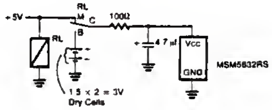
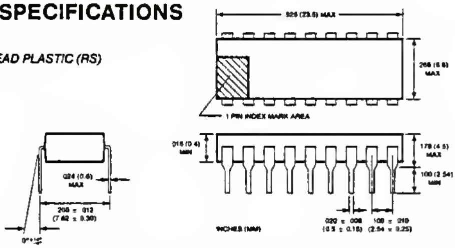


FIGURE 11



PACKAGE SPECIFICATIONS

18 LEAD PLASTIC (RS)



OKI SEMICONDUCTOR 1333 LAWRENCE EXPRESSWAY, SANTA CLARA, CALIF. 95051

TELEPHONE: (408) 984–4842 TELEX (25) 910–3380508

OKI Semiconductor reserves the right to make changes in specifications at any time and without notice. The information furnished by OKI Semiconductor in this publication is believed to be accurate and reliable. However, no responsibility is assumed by OKI Semiconductor for its use; nor for any infringements of patents or other rights of third parties resulting from its use. No license is granted under any patents or patent rights of OKI.

PRINTED IN U.S.A.

POWER-ONE INC.
DC POWER SUPPLIES

APPLICATION DATA SHEET

INFORMATION CONTAINED:
1. SCHEMATIC
2. PARTS LIST
3. SPECIFICATIONS
4. OUTLINE AND MOUHTING
5. GENERAL USER INFORMATION

MODEL HTAA-16W

EDITION NO. 2

A.C. INPUT	115/230VAC±10% 47-440HZ (DERATE OUTPUT CURRENT 10% FOR 50HZ OPERATION)
D.C. OUTPUT	SEE OUTPUT RATING CHART. ADJUSTMENT RANGE, ±5% MINIMUM
LINE REGULATION	±.05% FOR A 10% LINE CHANGE.
LOAO REGULATION	±.05% FOR A 50% LOAD CHANGE.
OUTPUT RIPPLE	2 TO 15V UNIT: 5.0mv. PK-PK MAXIMUM.
SHORT CIRCUIT & OVERLOAD PROTECTION	AUTOMATIC CURRENT LIMIT / FOLDBACK.
TEMPERATURE RATING	D° TO 50°C FULL RATED, DERATE LINEARLY TO 40% AT 70°C.
TEMP. COEFFICIENT	±.03%/ °C MAXIMUM .010% / °C TYPICAL.
EFFICIENCY	5V. UNITS: 45%, 12 & 15V. UNITS: 55%,
VIBRATION	PER MIL-STD-810C; METHOO 514, PROCEDURE X.
SHOCK	PER MIL-STD-810C; METHOO 516, PROCEDURE V.
OVERVOLTAGE PROTECTION	BUILT IN ON ALL 5V. MODELS SET@6.2V±.4V. OTHER MODELS USE OPTIONAL OVERVOLTAGE PROTECTION.
TRANSIENT RESPONSE	50 μ SECONDS FOR A 50% LOAD CHANGE
STABILITY	±.3% FOR 24HOURS AFTER WARM UP.
COOLING	CONVECTION COOLING IS ADEOUATE WHERE NON RESTRICTED AIR FLOW IS AVAILABLE. WHEN OPERATING IN A CONFINED AREA, MOVING AIR OR CONOUCTION COOLING IS RECOMMENDED.

*NEG. OUTPUT USABLE AT -5V AT .4A BY JUMPERING E1-E2 AND RESETTING R19.

SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE.

AC CONNECTION TABLE

FOR USE AT	115V A.C.	230V A.C.
JUMPER	1&3, 2 4	2&3
APPLY A.C. AT	1 & 4	1 & 4
FUSE INPUT AT	0.8A	0.25A

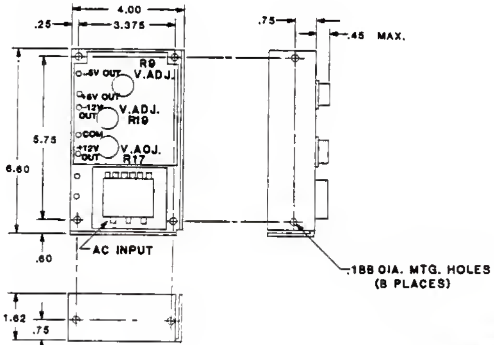
OUTPUT RATING CHART

MODEL	VOLTS	AMPS STEADY	SURGE	OVP
HTAA-16W	5	2.0		6.2±.4VDC
	+9to15	0.4		
	-9to15	0.4		
*—OR	-5to10	0.4		

WARRANTY

POWER-ONE WARRANTS EACH POWER SUPPLY OF ITS MANUFACTURE THAT DOES NOT PERFORM TO PUBLISHED SPECIFICATIONS, AS A RESULT OF OEFECTIVE MATERIALS OR WORKMANSHIP, FOR A PERIOD OF TWO (2) FULL YEARS FROM THE DATE OF ORIGINAL OELIVERY. RETURNS MUST BE FREIGHT PREPAID.

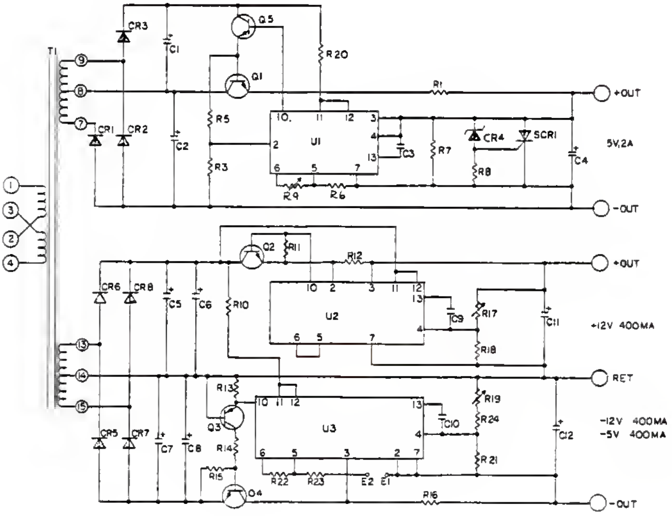
POWER-ONE ASSUMES NO LIABILITIES FOR CONSEOUENTIAL DAMAGES OF ANY KINO THROUGH THE USE OR MISUSE OF ITS PRODUCTS BY THE PURCHASER OR OTHERS. NO OTHER OBLIGATIONS OR LIABILITIES ARE EXPRESSED OR IMPLIED.



'HTAA' CHASSIS
UNIT WEIGHT 2LBS.

REF. DES.	PART OR IDENTIFYING NO.	NOMENCLATURE OR DESCRIPTION	POWER-ONE STD. P/N
C1,4	220/16	CAPACITOR ELECT.	101-10107
C2	9000/15	ELECT.	102-10097
C5,6,7,8	330/35	ELECT.	101-10109
C11,12	100/35	ELECT.	101-10110
C9	.001/100	MYLAR	104-10093
C3	.01/100	MYLAR	104-10095
C10	.0033/100	CAPACITOR MYLAR	104-10092
CR1,2	AE3B	DIODE 3A 100V	111-10252
CR3,5,6,7,8	AE1C	DIODE 1A 200V	111-10251
CR4	IN752A	DIODE ZENER	112-10006
SCR1	S0303LS3	SCR 3A	160-10258
Q1,2,4	2N6569	TRANSISTOR	171-10261
Q3	2N2907A	TRANSISTOR	172-10248
Q5	2N6551	TRANSISTOR	172-10249
R3	1.6K	RESISTOR ½W 5% CF	151-10370
R11,13,15,18	1K	RESISTOR ½W 5% CF	151-10365
R7	270Ω		10351
R5	300Ω		10352
R6	2.2K		10373
R8,14,20	47Ω		10333
R10	750Ω		10362
R24	150Ω		10345
R21	1.2K	½W 5% CF	151-10367
R22	2.4K	¼W 2% MF	152-10514
R23	2.7K	½W 2% MF	152-10515
R1	.22	RESISTOR 2W WW BWH	158-10079
R12,16	1.0Ω	RESISTOR ½W 5% CF	151-10300
R9,17,19	1.5K	POT. 2W WW	155-10085
U1,2,3	μA 723	IC VOLTAGE REGULATOR	130-10287
T1		TRANSFORMER	082-12465
PCB		P.C. BOARD	505-12466
CHASSIS		CHASSIS	412-11091

PARTS LIST



DATA LOGGER USING THE SYM-1 MICROCOMPUTER

by

MICHAEL DEAN SCHWARZ

B. S., Kansas State University, 1978

AN ABSTRACT OF A MASTER'S THESIS

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Agricultural Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

1984

ABSTRACT

The development of the SYM-1 based data logger resulted from two USDA Solar Demonstration Projects. These projects required instrumenting nine solar collectors on confinement swine buildings and ten solar collectors on grain drying installations to determine energy use. In evaluating the available funds for instrumentation the conclusion was made that commercial data loggers were too expensive to allow instrumenting each solar collector. The alternative to commercial data loggers was to build a data logger using a microcomputer.

The SYM-1 was selected as the microcomputer to use in instrumenting the solar collectors. At first an 8-bit analog to digital converter was used which required the input signal range to be as close to full scale as possible. Later a 12-bit analog to digital converter and programmable gain amplifier with gains from 1 to 1024 was used to obtain 12-bits of resolution for inputs ranging from 5 millivolts to 5 volts. A CMOS real-time clock was used to provide the correct time for the data logging. A battery was provided to provide power to the clock chip during power outages.

In response to problems where power fluctuations would stop the processor, a watch-dog timer was developed for the data logger. The watch-dog timer is an astable timer with its output connected to the computer reset. Under normal operation the SYM-1 produces a signal that retriggers the timer every second,

preventing a reset. In the event the processor stops because of a power glitch, the timer will not be retriggered and generate a reset signal for the computer.

Data is recorded onto an audio cassette tape which is then read by another SYM-1 and sent to a minicomputer for analysis thru the RS232 serial interface. To deal with possible tape errors, each data observation is recorded on the tape two to four times depending on the configuration of the data logger. The SYM-1 reader removes this redundant data and marks the data in the event of a checksum error.

The data loggers collected data for two heating seasons for the Solar Livestock Demonstration and one drying season for the Solar Grain Drying Demonstration. After a few initial electronic problems the data loggers functioned as designed.

The SYM-1 data logger was used on to instrument two projects other than solar collectors demonstrating the versatility of the data logger. An ice freezing project was conducted where the SYM-1 data controlled the air flow and water additions as well as collecting data. Engine preformance data was also collected with this data logger. This required fast sampling rates for short periods of time with no data logging during during engine stabilization. In both of these cases the SYM-1 data logger preformed with no modifications to the data logging code, although a control program was added for the ice freezing project.