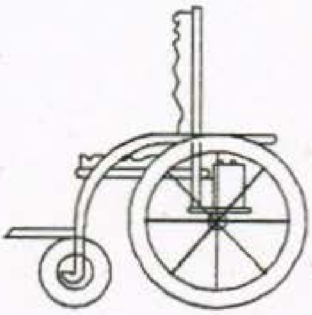
THE EASY CHAIR



DESIGNED BY:

JAMES P. WILLIAMS

GREGORY F. WELCH

E.E.T. 490/491 SENIOR DESIGN PROJECT

THE EASY CHAIR

INDEX

TOPIC PAGE

E.E.T. 490/491 SENIOR DESIGN PROJECT

THE EASY CHAIR

FIGURE LIST

TITLE FIGURE

ABSTRACT

The following report is a general synopsis of ideas and designs used in the development of the Easy Chair, a microprocessor controlled wheelchair for small children with muscular disorders.

The initial wheelchair comes equipped with a Damaco D88 Add-On power unit. This unit comes complete with batteries, the drive units (motors and controllers), and a proportional joystick controller. The touch-pad, ultrasonic kit, and the computer are the three extra components to be added for additional control and safety.

Specifications for the Easy Chair were outlined by an Occupational Therapist, Physical Therapist, and a classroom teacher from The Wabash Center in Lafayette, Indiana. This outlining was assisted by George Karlin, Special Education project coordinator at Purdue University, Lafayette, Indiana.

The original idea for the wheelchair was conceived by George Karlin, while working with small handicapped children both at Purdue University and The Wabash Center. George Karlin also acted as a go-between for the designers and therapists, throughout the design.

The touch-pad is valued at around $213, the unltrasonic system at around $342, and the computer at around $363, with a total cost of around $1053. All of the development components are being being paid for by The Wabash Center, with the final prototype being released to them.

The three sections reported on hereafter, all work very well seperately. The ultrasonics presently convey perimeter information, the touch-pad can be used to configure the system, and the computer is running, controlling the other systems.

What remains in the project is basically to complete the motor control system, to combine, test and modify the seperate components, to package the resulting hardware, and to polish off the system software so that it will allow the users to configure the chair for their specific needs.

INTRODUCTION

For many years, small children with muscular disorders have had severly limited opportunities to acquire any amount of mobility. Because of this lack of mobility, they have also had limited opportunities to initiate communication with others, limiting further their learning capabilities.

The idea behind a microprocessor controlled wheelchair (the Easy Chair) is to provide a mode of transportation for very young children with muscular disorders. Because the users will be so young, typically two to six years old, the chair should be equipped with a variety of devices which will not only allow them to control movement with limited muscular force, but will also protect them from any undesireable circumstances.

Such devices include a method of input such as a touch-pad, (requiring minimal or no muscular force to actuate) , an ultrasonic ranging system to monitor the chair's perimeters, and a computer to control these devices in a fashion which is transparent to the user, (see Figure 1).

From this point on, there will three major sections to the report. The first section will cover the touch-pad, the second will cover the ultrasonic ranging, and the third will cover the computer and the motor control.

THE INFRARED TOUCH-PAD

SCOPE

The infrared touch-pad is to be known as the input system for the control of the chair. It is currently thought of as the only direct method of input which will be associated with the final wheelchair. Therefore, it must meet many requirements which allow it to alter the current system configurations, or just to control the chair.

In conceiving the idea for the touch-pad, the following specifications were used as guidelines to facilitate design.

It was determined that a touch sensitive input surface requiring minimal pressure would best suit the needs of the small children. The system needed to be adaptable to different children, some of whom are incapable of generating high response force.

The touch-pad should use a common medium for set-up, to increase the independance of the system and its users. This is to say that it should be possible to simply plug in or unplug the touch-pad, and to switch between the pad and the current joystick with little or no effort. It should be totally self-contained as a unit, electronics and all. Again, this would increase the independance of the system.

The touch-pad should be constructed in such a way that it could be attached to the current center off-set mounting arm of the wheelchair (which swings out of the way of the user), with the option of resting on the lap tray of the chair. These two methods will result in the touch-pad being as ambidextrous as possible.

The unit should be large enough to be easily viewed and touched, but small enough so as not to be obtrusive to the user and the wheelchair. A general touch-pad area of ten inches by ten inches was set for initial dimensions.

The size and locations of the symbols on the touch-pad (used to control the wheelchair) must be programmable. This will accomodate different ranges of motion.

The touch-pad must be moisture proof. Children with such handicaps as cerebral palsy frequently have oral motor problems which result in excessive drooling. Any reasonable amount of moisture should not cause the wheelchair to malfunction.

In the past, it had been thought that a total hardware solution was the most reliable and consistant route to take. However, after carefully studying that route, and testing the results, it was determined that a combination of approximately equal amounts of hardware and software would allow the most flexible design. The following sections describe the present solution, and how it is implemented.

BODY

BLOCK DIAGRAM

The block diagram for the touch-pad consists of six main blocks. These blocks include the row decoding (selecting) block, the column decoding block, the extra decoding block (which includes the menu-select decoding and the ultrasound direction light decoding), the touch-pad block, the row/column detect block, and the menu-select detect block. Each of these blocks will be discussed further in the following sections (see also figure 2, Block Diagram).

I. THE ROW DECODING BLOCK

The row decoding block is one such block where the seven bit control word which is sent to the touch-pad circuitry is interpreted to select a certain LED/phototransistor pair.

The decoding is accomplished by sending the least significant four of the seven bits as a nibble which gives a zero through fifteen (F Hex) count, and then bringing one of three chip select lines high, in particular the row decoder chip select line (see figure 3-1). To accomplish this, a 74154 4 to 16 line decoder is used. The outputs of this 74154 are low when they are selected, so they are used to provide a ground path for the infrared LEDs and phototransistors, thus allowing them to be turned on only when they are selected.

It is appropriate at this time to re-state the fact that the select lines are used to select both an LED and a phototransistor. With this scheme, if there is nothing blocking the beam path from the LED to the phototransistor, then the phototransistor should be turned on.

II. THE COLUMN DECODING BLOCK

The column decoding block functions in almost the same fashion as the row decoding block. The only difference is that of the select line which is used to select the column decoding chip, also a 74154. Of the three selct lines (bits) from the seven bit word mentioned, one is used to select the row decoding chip, one the column decoding chip, and one the extra decoding chip. To select the column pairs, the column select bit must be high.

Again, in the same fashion as the row decoding, this block selects certain LED/phototransistor pairs to be observed by the detection circuitry.

III. THE EXTRA DECODING BLOCK

Again, the basic function of the extra decoding block is the same as that of the row and column decoding blocks. However, this block serves no one single function such as row or column decoding.

The term extra is meant to reflect the odd or `extra´ decoding that is done by this block. At the present time, it serves two main functions; to select one of the five menu-select LED/phototransistor pairs for observation, and also to momentarily select other devices such as lights which will assist the user in determining which perimeters are being warned about by the ultrasonics.

In refering to figure 3-1, it should be noted that the five `menu select´ lines are passed through tri-state buffers before they are connected to the LED/phototransistor pairs. This is because smaller LEDs and phototransistors had to be used for the five menu select pairs (to fit between the column pairs in the pad). These smaller phototransistors had lower off-state resistance, which caused problems when they were not selected. Normally when a pair is not selected, +5 volts is connected to the cathode of the LED and to the emitter of the phototransistor. This would not allow either to be turned on. With these five menu select pairs however, the +5 volts (seen when not selected) caused the menu-select detect circuitry to send a touch message to the computer. Therefore, the tri-state buffers were used, which present an open circuit in their non-selected state.

This extra decoding device could be thought of as an extra computer port, with the only difference (which is a disadvantage) being that the outputs are not latched in their selected states. However, for the present time, this is not necessary, and momentary selction will work fine.

IV. THE TOUCH PAD BLOCK

This block contains the actual touch-pad with the LEDs and phototransistors mounted in it, and the slot for the selected menus to be inserted into (see figure 4). Along the vertical and horizontal sides of the sunken touch area, are alternately mounted 32 infrared LEDs and 32 phototransistors, one across from each LED. These pairs were alternated to reduce the amount of light being received in error. The LEDs and phototransistors were carefully aligned so as to achieve the maximum signal recieved when a signal is sent. Each of the cathodes of the LEDs along with the emitters of the phototransistors across from them, are tied to the select lines of the 74154s (see also The Row Decoder Block and The Column Decoder Block).

The touch-pad also contains five seperate pairs which are mounted perpendicular to the row and column pairs, along the edge of the pad. These serve the purpose of allowing the computer to detect which menu is in the pad. The paper menus have five corresponding holes which can be cut open or left intact (closed), representing zeros and ones.

The anodes of all of the infrared LEDs (both row/column LEDs and menu-select LEDs) are tied high through a single series limiting resistor. Therefore, again when the pair is selected, and the cathode is taken to ground, the LED turns on.

Eventually, all of the select and the detect circuitry will be packaged along the left and right side of the touch-pad, and all of this will be enclosed in one case. This will allow the touch-pad to be totally self-contained (independant). It will be tied to the computer by a ten conductor cable which will include the four pair select lines, the three chip select bits, one touch return line, +5 volts and ground.

V. THE ROW/COLUMN DETECT BLOCK

This block is where the phototransistor status is transformed into a level that can be interpreted by the computer.

The collectors of all of the phototransistors are tied together, because only one is selected at a time. These are then pulled high through a single pull-up resistor (100k ohms). When any one of the 32 row/column phototransistors is selected, an infrared beam of light from the paired LED should turn it on, putting the collector voltage somewhere near ground. If while one is selected, the beam is blocked, the phototransistor will be turned off. When off, the collector voltage approaches +5 volts because of the pull-up resistor.

Because of the change in collector voltage from when a beam is blocked to when one is not blocked, the collectors are the input to the row/column detect circuitry. This circuitry uses a pair of comparators, with references set by a 20k ohm potentiometer set up as a voltage divider.

The first comparator is set up in an inverting fashion, so that when any collector voltage is below the reference (no beam blocked), the output of the comparator is at positive saturation. However, if any collector voltage swings above the reference, the output goes to negative saturation (close to ground). This output is then used as the input to the second comparator.

This second comparator has the same reference voltage as the first one, however, it is set up in a non-inverting fashion. It is used to clean-up the comparator signal. When the selected beam is not broken, the output of the first comparator (which is the input to the second) is high, which also sends the second conparator into positive saturation. This second output signal, called RCRET (row/column return), is then passed through an OR gate which has one input tied low, to clean it up.

This conditioned RCRET signal is then combined with the MSRET signal (menu-select return) to provide one single RET (return) signal for the computer. This signal does not provide a hardware interrupt, but is instead polled by the software as a single bit input to a port.

VI. THE MENU-SELECT DETECT BLOCK

The menu-select detect block has almost the same circuitry as the row/column detect block, with the only real difference being the size of the pull-up resistor needed for the five smaller phototransistors (menu-select pairs). Otherwise, the operation is the same, with the same circuitry repeated simply to isolate the menu-select return (MSRET) from the row/column return (RCRET).

It is appropriate at this time to note the reason for combining the three decoding chip selects with both the row/column detect and the menu-select detect (see figure 4, SCHEMATIC). The reason is that if the neither the row or column chip is selected, then the RCRET signal is high, falsely signaling a beam being broken. The same problem is encountered when the menu-select chip is not selected, the MSRET signal is high, falsely signaling a beam being broken. To eliviate this problem, the row and column chip selects are AND'ed with the RCRET signal, and the extra chip select is AND'ed with the MSRET signal. With this method, RCRET can only go high when either the row or column chips are selected. Also, MSRET can only go high when the extra chip is selected.

The resulting signals are OR'd together to form a single RET line which is high whenever a selected beam is broken. This leaves the computer free to select either a row, column or menu-select beam, and then determine with one line (RET) whether or not that beam is being broken.

GENERAL DISCUSSION

As was mentioned earlier in the scope of the project, the original thought had been that a total hardware system would be best. With such a system, the computer would only have to respond to an interrupt by the touch-pad, and during its service request, check the pad to see which location had been touched.

All of this could have been provided by setting up a hardware clock which ran several counters. These counters would in turn select each row pair, then each column pair, and finally each menu-select pair. The major disadvantage to this method was that the scan process would be set in one certain fashion, unable to change as better processes were discovered. With the present method, the computer supplies the count to the pad, so if it sees that the RET (return) line is high, then it knows that the beam (pair) selected has been interrupted or blocked.

The current method of using infrared light beams, was decided upon for various reasons. First of all, other touch-pad schemes such as capicitive touch sensing, and pressure sensitive membrane type keypads, are all open to problems because they are affected by water, or saliva in this case. Secondly (and most important), breaking a light beam requires the least amount of pressure of any method studied.

The approach of using identical circuits for the RCRET and the MSRET may at first seem redundant. However, because of the limited amount of physical space between the column LEDs and phototransistors, smaller versions had to be used. These smaller versions required the same type of detection circuitry, with only a change in one resistor. So, because the two blocks need to be electronically isolated, and because the needed gates and comparators (for duplicate circuitry) were in fact available, it was decided to duplicate the row/column detection for the menu-select detection.

Other reasons for choosing to duplicate the detection scheme are, not only the fact that no additional components were required, but also that the original scheme was tested and working well.

It is thought that in the future, there might be the possibility of interfacing a small lap-top computer which would allow the users to much more readily re-configure the touch-pad. With such a device, programs could be written in BASIC to make programming much more user-friendly.

ULTRASONIC RANGING SYSTEM

SCOPE

The ultrasonic ranging system is considered a protective device. Its major function is to prevent damage to the chair or injury to its operator. It is also necesary to protect other young children who might be in the operating area of the chair.

In conceiving the idea for the ultrasonic system, the following specifications were used as guidelines to facilitate design.

The system is not intended to be an intelligent system. That is, it is not to take offensive control at anytime as this would deter the user from learning to be in complete control of the wheelchair. It is hoped eventually the devlopment of the users skills will allow the user full control without perimeter sensing.

The system should have some kind of audio and visual feedback, warning the user of obstacles, causing the chair to slow or stop. As loud noises can become bothersome, this option should be selectable.

The system should sense any obstacle entering into an approxamantly 2 foot distance surounding the chair, and should slow down accordingly to allow the chair user to have more time to make corrective actions. If corrective actions are not made in time, the chair will stop just before contacting the obstacle (less than 4 inches).

The ultrasonic system as well as the other systems should not destroy or deface the wheelchair in any manner. If any one part of the chair is rendered inopperative, the chair itself cannot become useless. If a major failure occured, it should be easily possible to remove and retire the complete sytem.

The ultrasonic system, as specified, performs two functions. It provides feedback to the user as to the approach of obstacles and it provides a failsafe for stopping chair movement if the child does not respond to the approach warning.

BODY

BLOCK DIAGRAM

The block diagram for the ultrasonic system consists of four principal parts. These include four directional transducers, the tone generator, the time base generator for distance calculation, and the interface to the computer system. Each of these blocks will be discussed in the following sections (see also figure 1, EASYCHAIR BLOCK DIAGRAM).

I. THE DIRECTIONAL TRANSDUCER BLOCK

The directional transducer block is the heart of the ranging system. It consits of four complete and seperate ranging transducers. Each of which contains a 5Ø-kHz 3ØØ-volt electrostatic transducer and a small amount of drive circuity. Each transducer is capable of ranging from 4 inches to approxatly 35 feet with less than 2% maximum error. (see figure 5)

The drive circuity consits of Texas Instruments SN28827 sonar ranging module. This module provides the 15Ø-volt bias for the transducer and pulses the transducer with 16 cycles of 5Ø-kHz 3ØØ-volt waveform. (see figure 6). This manifests itself as short audiable click. This ultrasonic click travel at the speed of sound (Ø.9 ms/foot) until it strikes an obstacle and its echo returns to the transducer at the same speed. The module provides a controlable blanking period to allow transducer vibration to disapate before it is enabled to wait for a returning echo. All control signals are TTL compatible, but the echo output is of open collector type and needs a pull-up resistor in order to get a reliable TTL signal.

There are three main control signals. The INIT* input starts the ranging process by sending out the click. The BLNK* input defeates the internal echo blanking. And the ECHO* output signals when the click is returned. All three signals are active low and their relationship to all the rest of the control line is shown in figure 6.

The only devation from Texas Instruments design was in adding a large capacitor in parallel with the power supply as it enters each transducer's driver. This was done in order to supply the rated 2ØØØ mA each transducer needs during the 326·uS transmit period. This is such a rappid drain that the power supply could not source it through 6 ft of cabling.

II. THE TONE GENERATOR BLOCK

The tone generator block consists of the XR22Ø6 function generator chip which is capabale of switching between two selected tones, and an LM2ØØ2 8 watt audio power amplifier chip that amplifies the tone signal and drives the 8 ohm speaker. (see figure 8).

The XR22Ø6 has the ablitiy to output a stable tone and change to another tone by switching the TTL level at the FSK input. This allows several types of warnings to be generated. The two tones are seperatly adjustable and independent. These adjustments are made to R4 and R6 in figure 8. The potentiometer (R7) in the figure is a volume adjustment allowing the overall loudness to be changed.

Turning the tone off all togther is done with the Amplitude Modulation input witch if held at half the supply voltage to the chip will stop the output of the tone. What was done here was to build a voltage divider with two equal resistances therefore a voltage at half the supply, then parrallel a 2N39Ø4 to ground. now the base of the transistor can accept a TTL signal and switch the tone on or off.

III. THE ADDITONAL PIA AND TIMER BLOCKS

The interface block necessatated a second 8255 programable port. It is configured to have 24 bits of output and 4 bits of input. With port A and B being output ports along with the higher 4 bits of port C. The lower 4 bits of port C are the input bits. Port A controls the ultrasonics INIT* and BLNK* of each transducer. Port B output a digital word to be use by the motor control circiuts for direction and speed control. Port C controls the tone generator with its upper half and receives the ECHO* from the transducers on the lower half. (see figure 15)

The time base block consists of three programmable counter/timers in the 8253 on the SCCS-85. The fist timer is configured to count down from 65,535 (ØfffH) and is used as a stop watch during the ranging cycle. The second function of the 8253 is generation of the 16*baud clock needed for RS-232C communacation. The last counter is used for a heartbeat interupt .This will return the chair to the joystick configuration if the computer becomes inopperative or is turned off.

GENERAL DISCUSSION

The ultrasonic system and it parts have all been breadboarded and tested. All parts work as expected, and the ranging system, in particular, out performs what was expected of it. The ultrasonic system is very easy to use and is extremly accurate and reliable. The one and only disadvantage to ultrasonics as opposed to other ranging methods would be the perceivable click when the transducer fires.

From a designers standpoint, using a prebuilt module for the units was definitely better than trying to design the modules themselves. This made troubleshooting the modules harder if they failed to work (they often did) because of not being exactly sure of what the module was tring to do. A lot of the solutions to those problems came about from trial and error and a bit of luck.

The design of the tone generator and additonal PIA/timer configuration was much more straight forward and the results more along the lines of what was expected. The only problem arising here was driving the 8 ohm load of the speaker. After trying to use voltage and current amps (741 and 3900), and transformers and push-pull amps, it was decided to use the LM2002 which is made for such a purpose.

What is left for these parts is for a single PC board for the PIA, tone generator, power supply and motor control circuits to be made and tested. The software for the control of these circuits has been done to the extent that testing required, but has a long way to go before the Easy Chair is completed.

THE COMPUTER AND MOTOR CONTROL BLOCKS

SCOPE

The computer and motor control systems are possibly the most improtant parts of the Easy Chair system. A failure in either of these two systems could render the entire system inopperative. Therefore, durability and usablity are two major concerns. The computer system was chosen due to its ablities and because of the knowledge and familiarity of the EET staff with this product. So far it has filled the need and lived up to its expectations.

The motor control system is the weakest part of the total system as it stands now. This was due to the limited amount of time spent with the wheelchair itself. Arrangements have already been made to speed a great deal of time on this portion next semester.

BODY

I. THE COMPUTER BLOCK

The computer block is made from the 8Ø85 based single card computer system avalible from Purdue. The computer was built according to the manual provided. After opperation was verified, the following changes were made. Clock speed was increased to speed execution time but no appreciable increase has been noted. The memory configuration for the computer consists of three types: 8K of EPROM for startup sequence and monitor, 8K of static RAM for data storage and program development, and 2K of EEPROM memory used to store the menu information and other 'hard' variables. The EEPROM is expected to be configured to allow anyone to make easy and permanent changes in the menus or parameters.(see figures 1Ø-16)

II. THE MOTOR CONTROL BLOCK

The motor control block contains all the necessary electronics to switch control of the chair over to the Easy Chair controller. When this happens, the light pad and ultrasonic systems become the controller replacing the joystick. The motor control circuit uses a single 2 digit hexadecimal value to control both motors in approximately eight speeds forward, and eight speeds reverse. This should allow not only for smooth speed changes, but also, starting and stopping should not be rough or jerky.

Operation of the controller is fairly straight forward. Two AD558 digital to analog converters are used to create a digitally controlled voltage variable from Ø-2 volts. This output is summed with a 7 volt reference to produce an overall output controllable from 7 to 9 volts. The joystick pots have been measured to be at these potentials during operation of the chair. Although this has not been fully tested, it is believed to be a sound design.(see figure 9)

GENERAL DISCUSSION

The computer system is working as expected and software is the only thing planned to be added to it at this time. It is exepected that an additional EEPROM will be add in the future to allow greater program flexibility and possibly an increase in the number and quality of the menus. The addition of the extra 8255 caused no problems with the system and was easily added by using the available selects on the computer board.

The motor control system is the part of the Easy Chair which the most effort is currently being put forth on. The design should work theoretically, but there are concerns such as noise and drift, which must be addressed next semester. Along with software, motor control is where most efforts will be concentrated next semester.

CONCLUSION

The project as a whole seems to be running very smoothly, in fact, ahead of schedule. Each of the seperate blocks is independently working, with almost all functioning together as a system.

As far as software is concerned, the original monitor program used in the SCCS-85 computer has been modified to include several small test routines. These routines currently exercise only the seperate blocks to assure that they are working correctly.

The designers felt that at this point, the software was only in an experimental stage, and that the more serious software work would take place during the second semester of the 1985-86 school year at Purdue. For that reason, this report only includes a single listing of the current monitor program (see Appendix B) and no in-depth discussion concerning each seperate routine. There is, however, general discussion in the form of comments within the code.

A major reccomendation for the future would be to always check second vendors for supplies. For instance, after checking with Polaroid for the ultrasonic transducers, they were later found for almost one third the original cost at another vendor. Also, the cost of LEDs and phototransistors could be kept down by buying from a large wholesale distributor, because of the quantity.

Another thought would be that if the touch-pad were constructed just slightly larger, the same LEDs and transistors could be used for all of the detection. This would eliminate the need for special menu-select detect circuitry, and obviously the special LEDs and transistors.

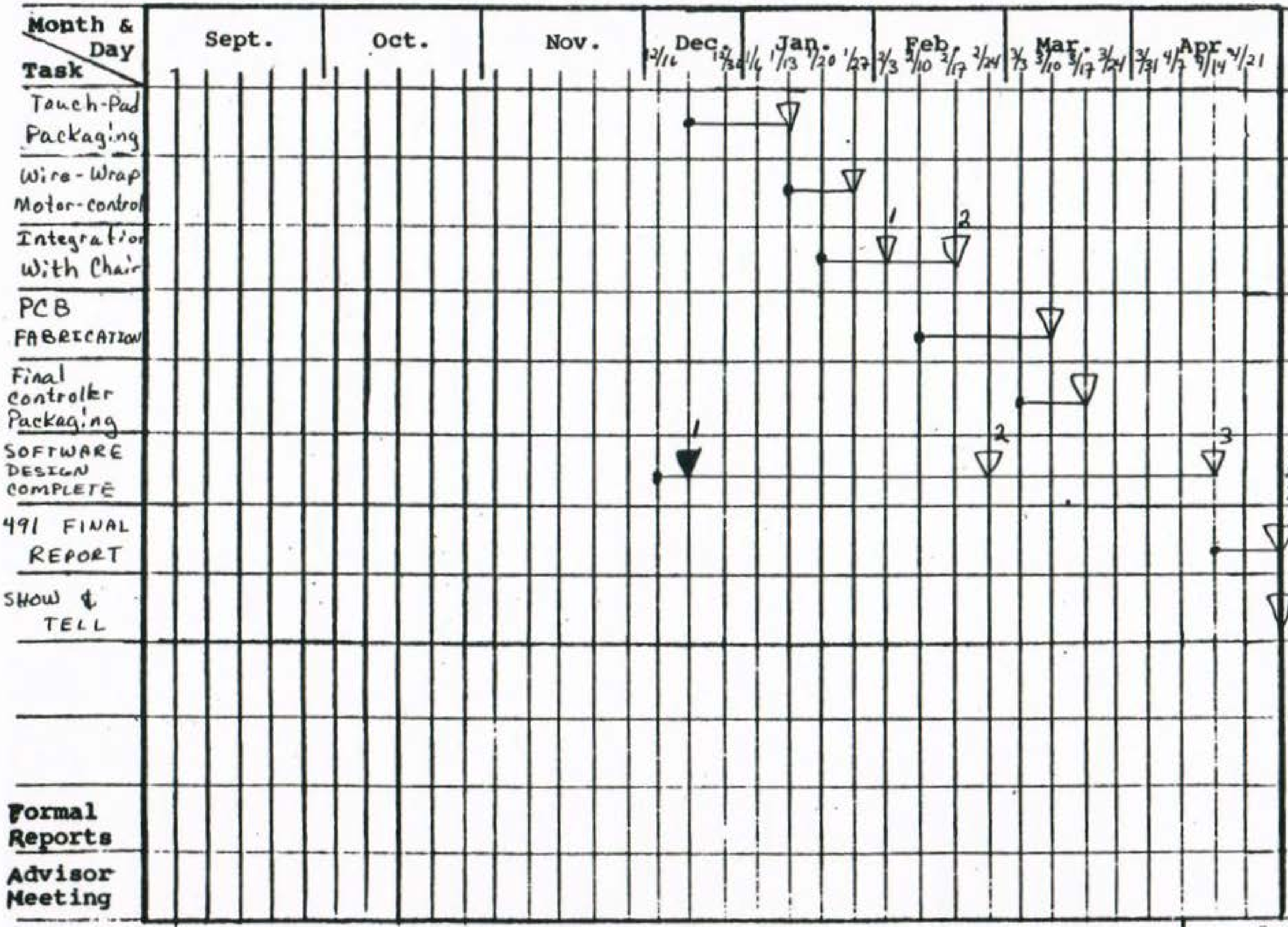
Again, it would seem worthwhile to mention the fact that the project seems to be really running ahead of schedule. Not only has the class goal for EET 490 been met, but most of the system blocks are already integrated together in test programs, achieving a good head-start into EET 491.

Overall for the project, having two people working together seems to greatly enhance not only productivity, but also the enthusiasm. The only major difficulty encountered with a joint project was a time problem when integrating the individual reports to form one single, flowing, and concise report.

EET 490-491

TIME ACTION PLAN
PROJECT MILESTONE

Project Title: Microprocessor Controlled Wheelchair Project Leader Greg Welch / James Williams Date 11/12/85



Milestone Definition

1 - Mounted On Chair (Te
2 - Tested Fully (Basic

1 - Test Routines (Gener
2 - Major Completions
3 - Extra Features Comp

Comments Timeline for Spring Semester

Project Advisor: Prof. Tom Schultz (See advisor once per week)

Legend
__Plan ▽ Milestone
----Actual ▼ Completed Milestone

E.E.T. 490/491 SENIOR DESIGN PROJECT

THE EASY CHAIR

APPENDIX A: FIGURES (PICTORIALS)

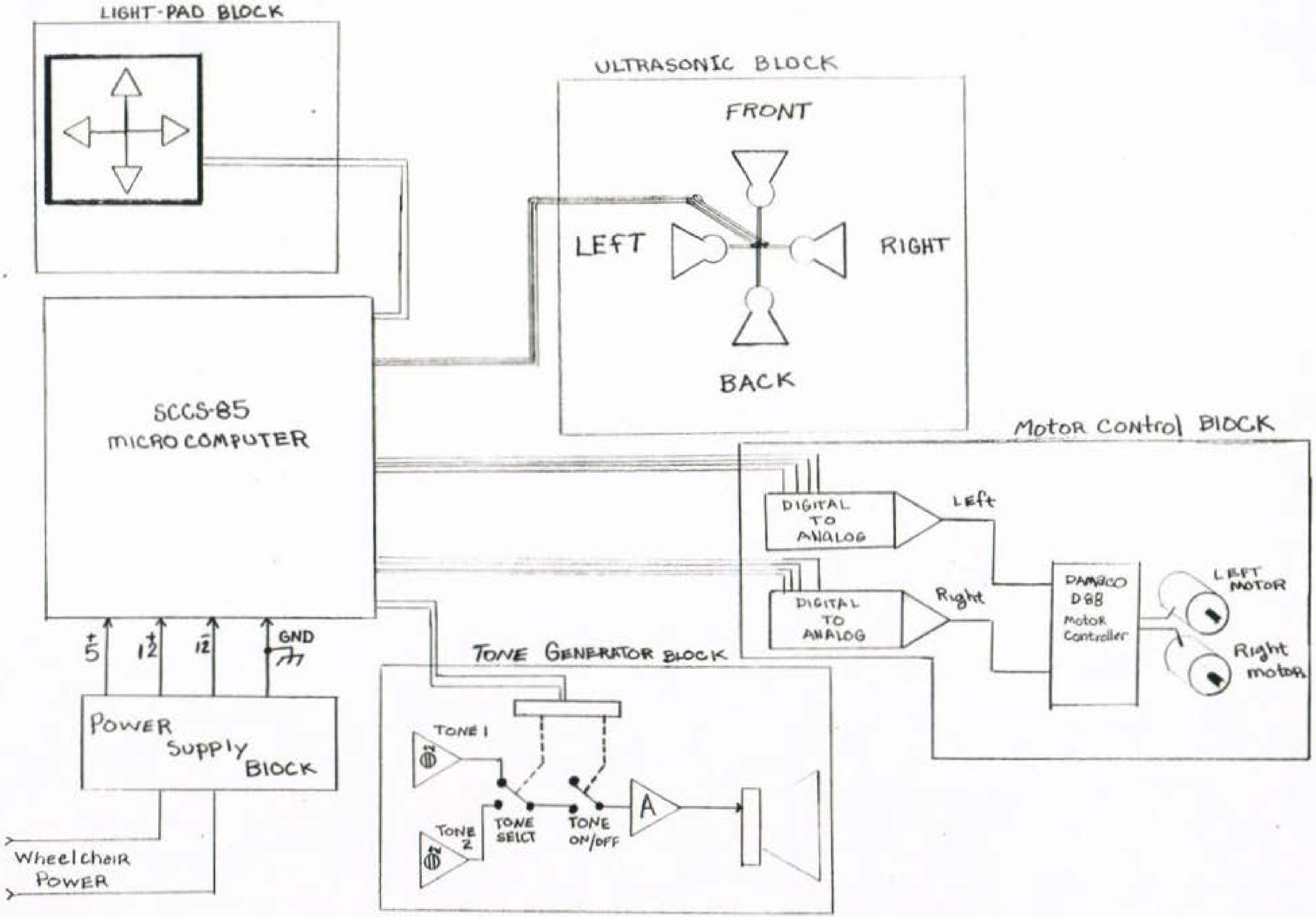


FIG: 1	DESIGN BY: Y.P.N	TITLE: EASY CHAIR BLOCK DIAGRAM
DATE: 12/17/85	DRAWN BY: Y.P.W	

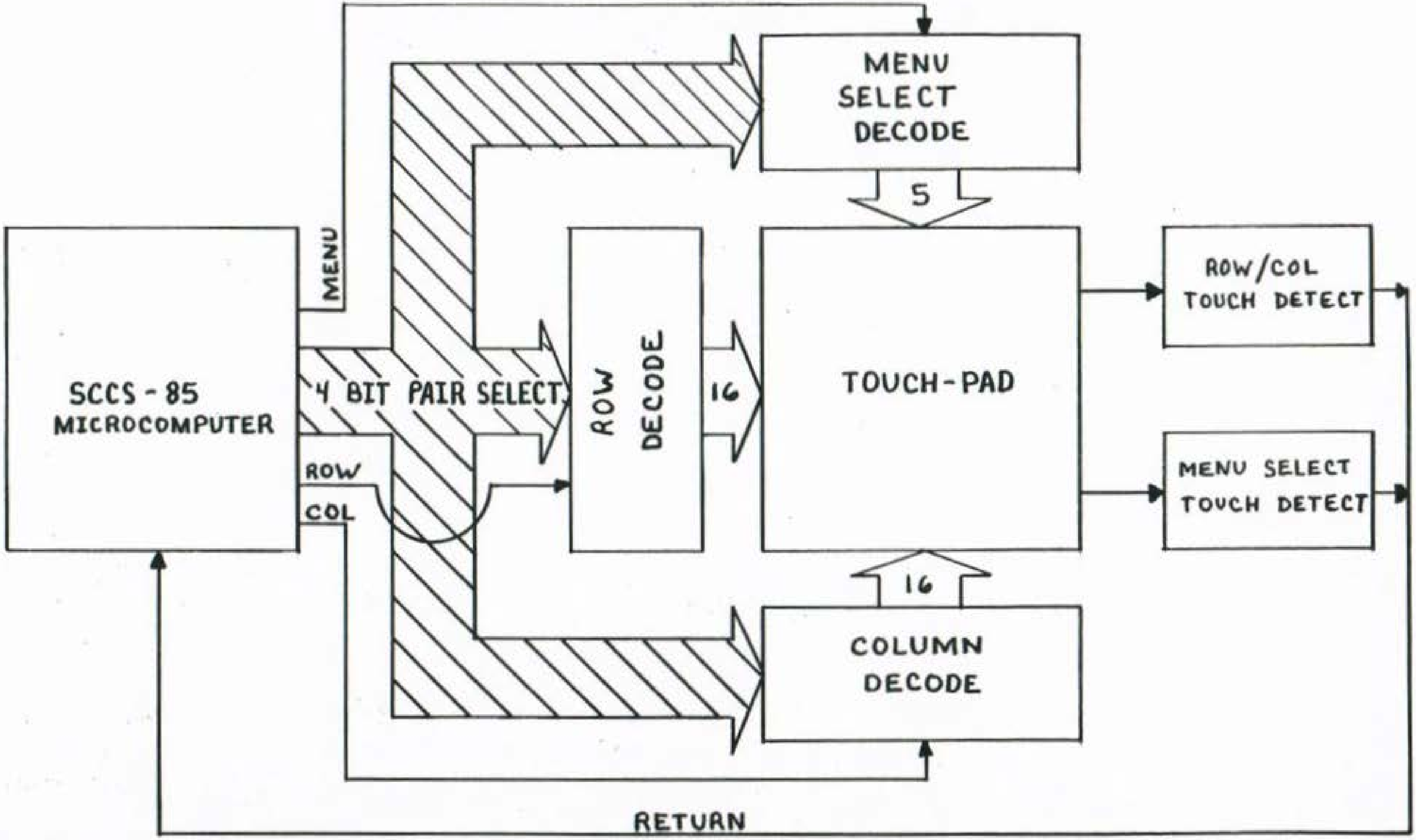
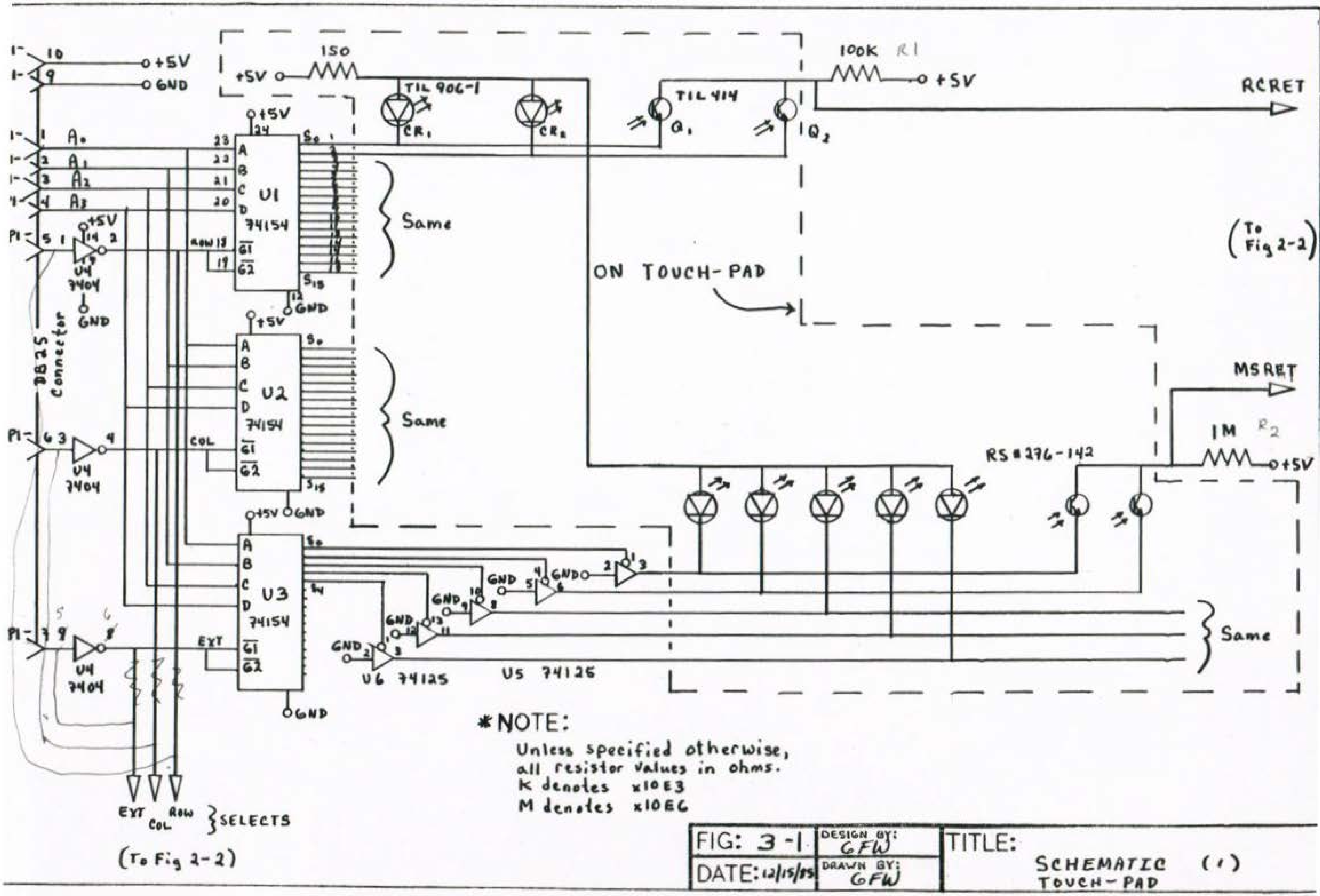


FIG: 2	DESIGN BY: GFW	TITLE: TOUCH - PAD BLOCK DIAGRAM
DATE: 11/14/85	DRAWN BY: GFW	



*NOTE:

Unless specified otherwise,
all resistor values in ohms.
K denotes x10E3
M denotes x10E6

FIG: 3-1	DESIGN BY: GFW	TITLE:
DATE: 12/15/85	DRAWN BY: GFW	SCHEMATIC (1) TOUCH-PAD

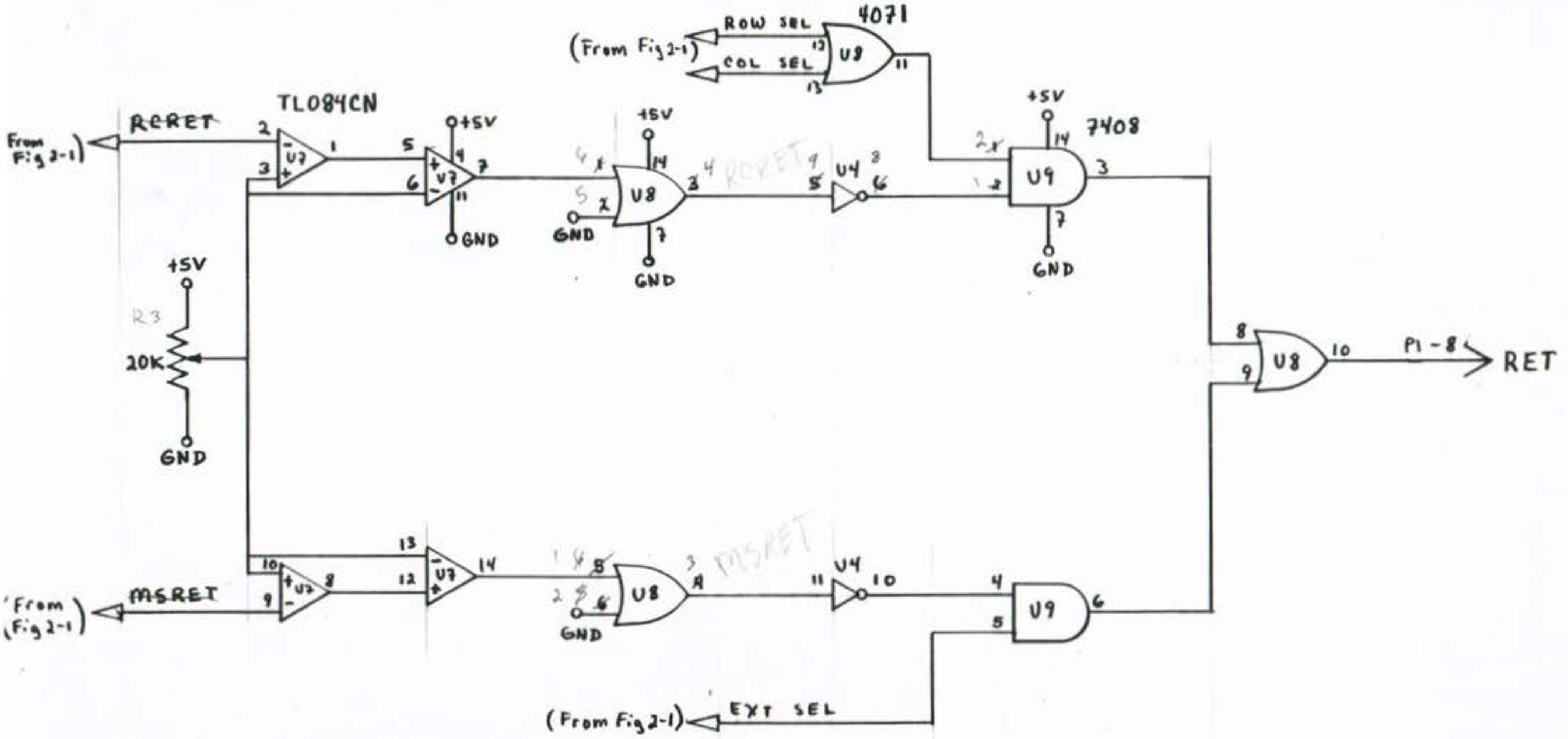


FIG: 3 - 2	DESIGN BY: GFW	TITLE:
DATE: 12/13/85	DRAWN BY: GFW	SCHEMATIC (2) TOUCH-PAD

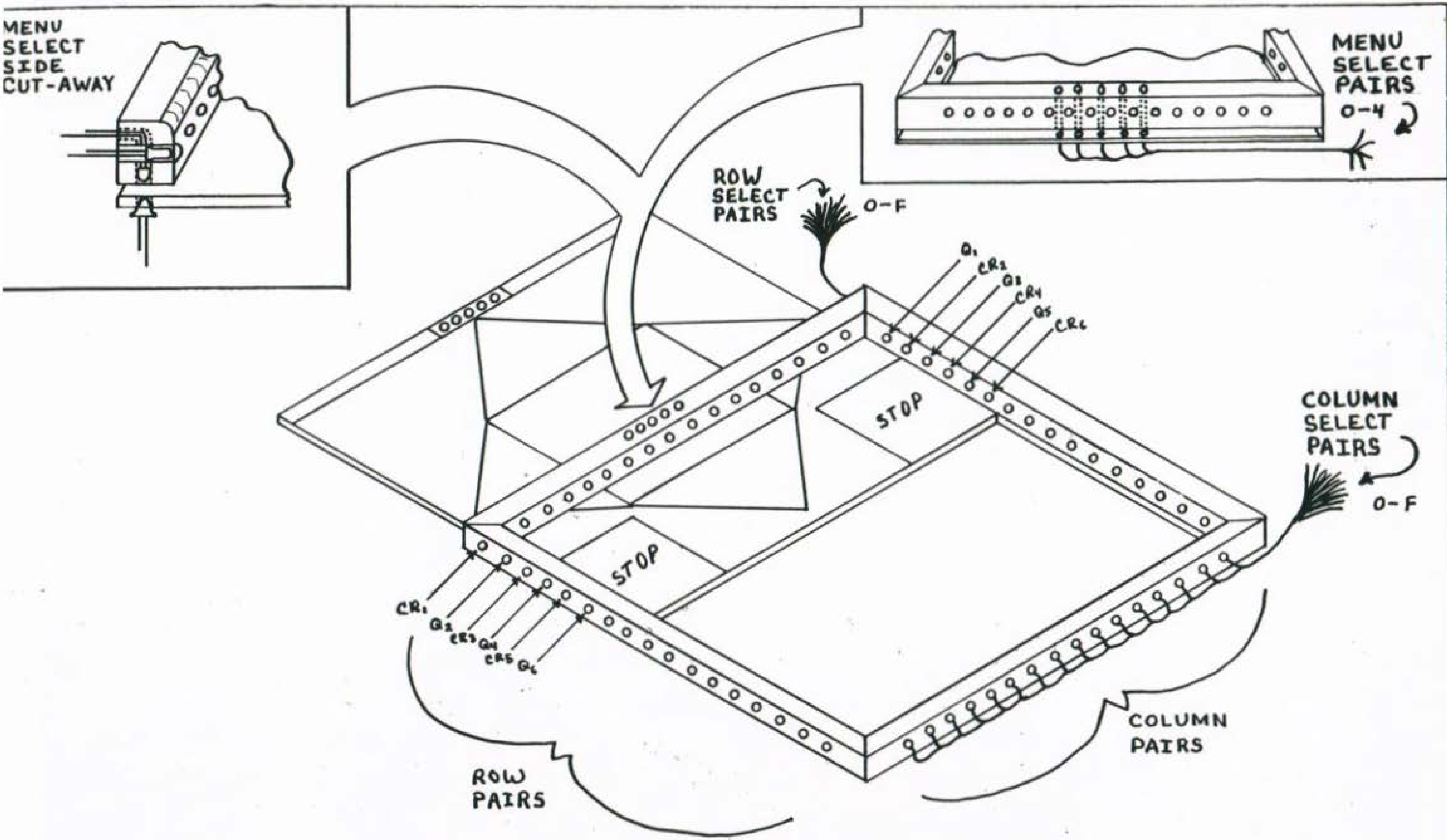
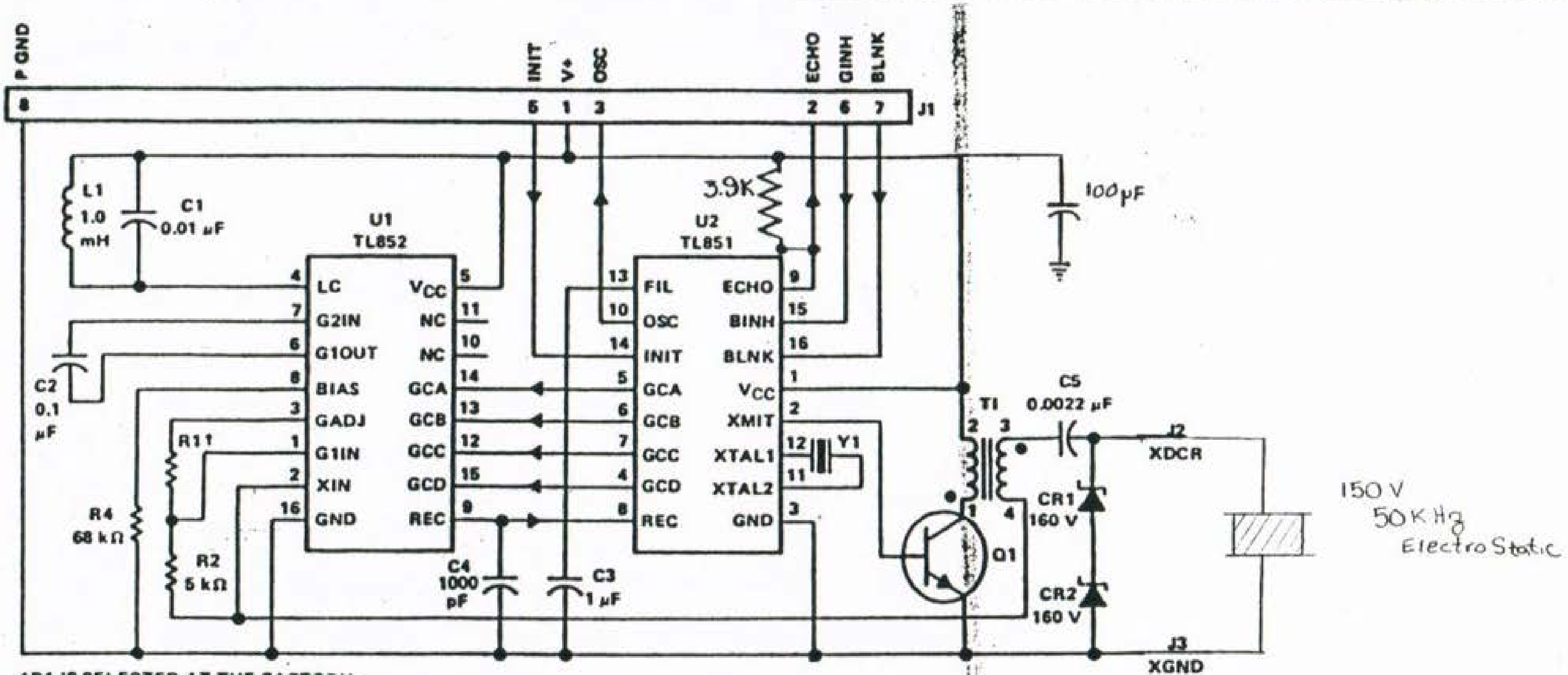
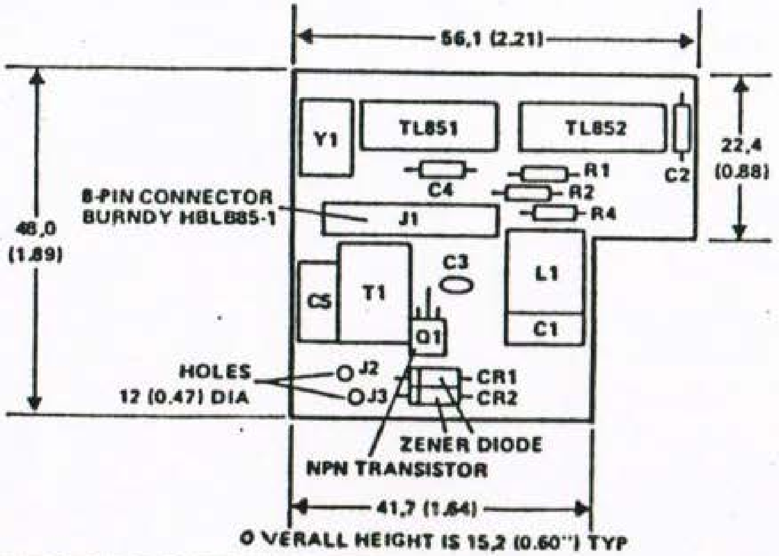


FIG: 4	DESIGN BY: GFW	TITLE: THE TOUCH PAD
DATE: 12/13/85	DRAWN BY: GFW	



†R1 IS SELECTED AT THE FACTORY.



OVERALL HEIGHT IS 15,2 (0.60") TYP

UltreSonic driVER.

* ONE OF FOOR *

Contained inside eACH
Ranging Unit

FIG: 5
DESIGN BY: VPW
DATE: 12/18/85
DRAWN BY: V.P.W
TITLE: SN28827 Sonar
RANGING Module

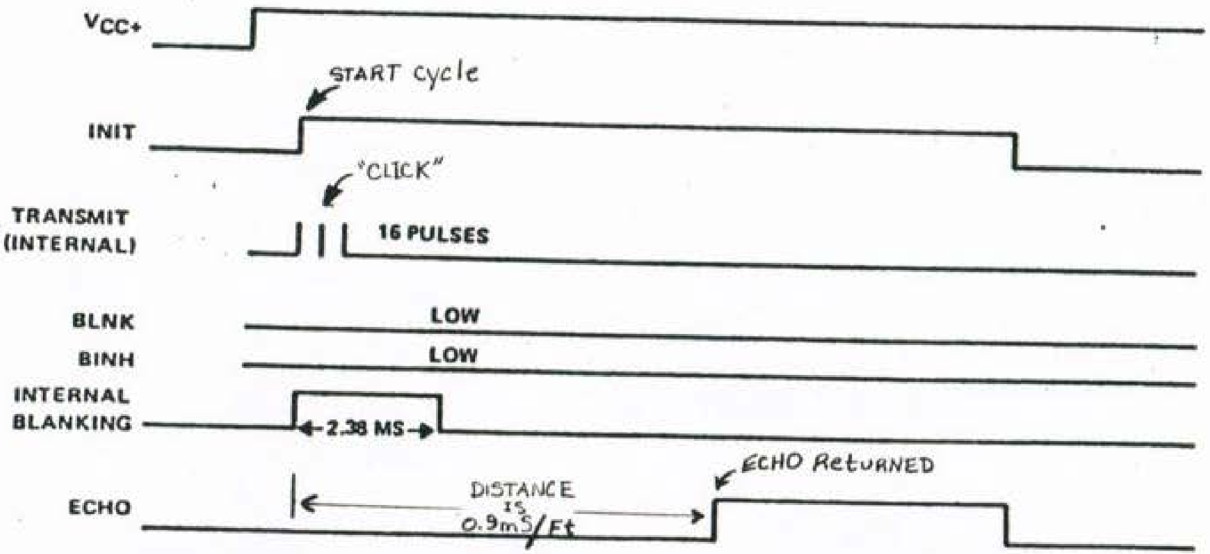
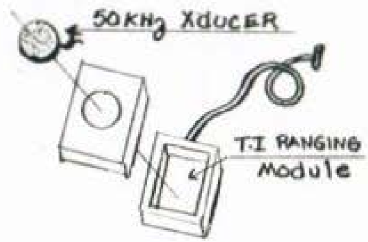
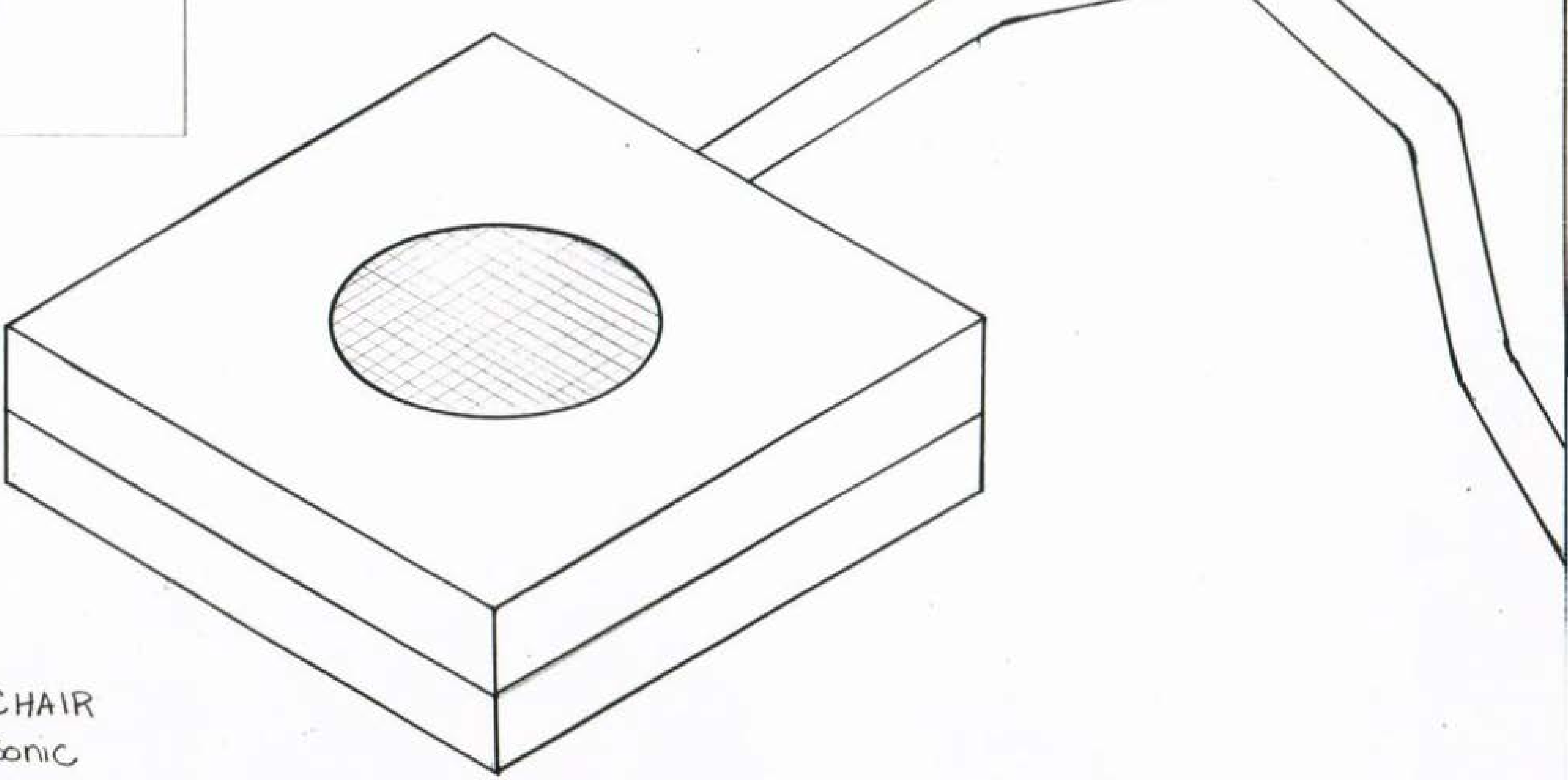


FIGURE 1—EXAMPLE OF A SINGLE-ECHO-MODE CYCLE WITHOUT BLANKING INPUT

FIG: 6	DESIGN BY: YP.W	TITLE: ULTRASONIC Control Signals
DATE: 12/18/85	DRAWN BY: YPW	





EASY CHAIR
Ultra Sonic
RANgING
Unit

FIG: 7	DESIGN BY: V.P.W.	TITLE: ULTRASONIC Pictorial
DATE: 12/17/05	DRAWN BY: V.P.W.	

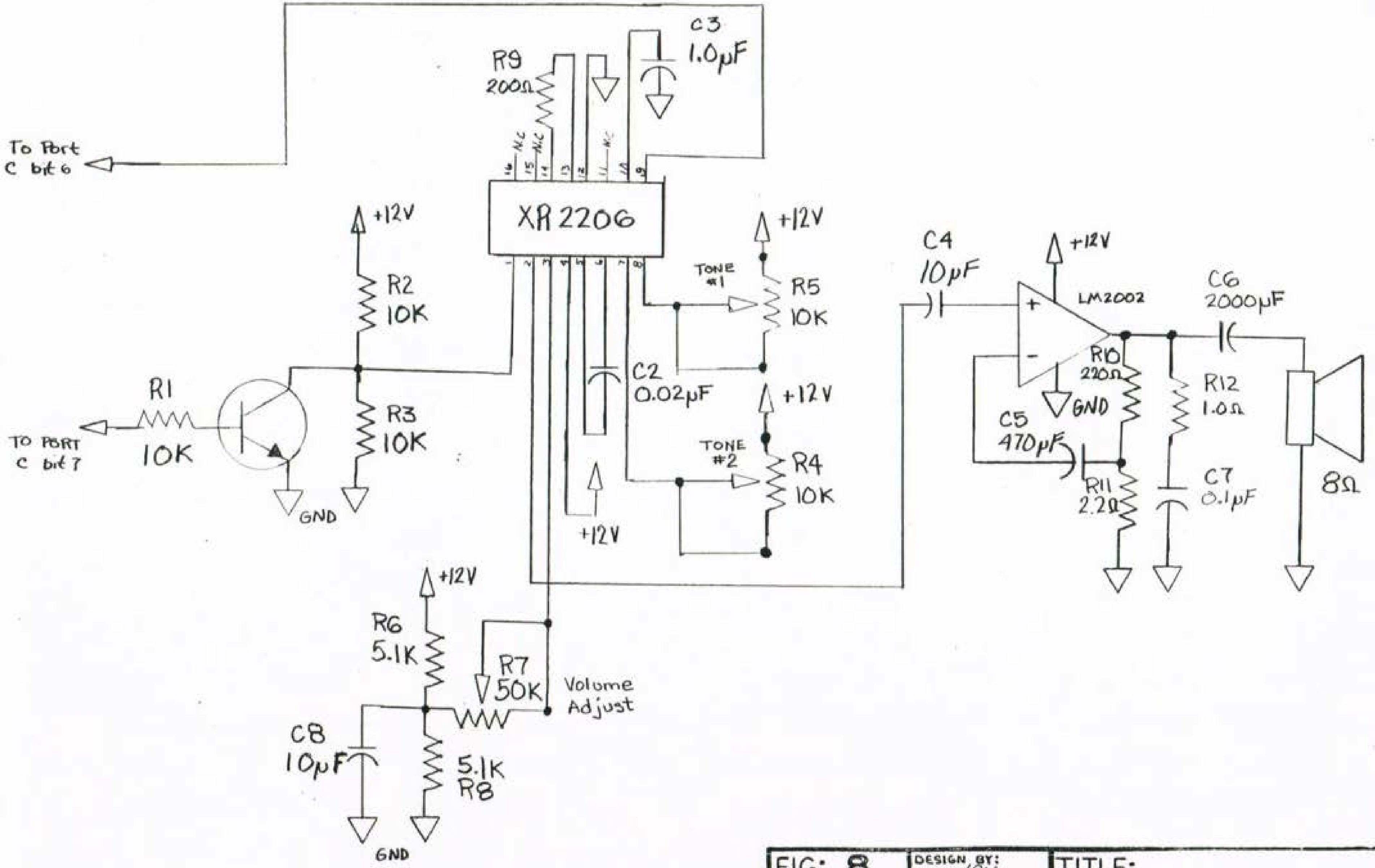
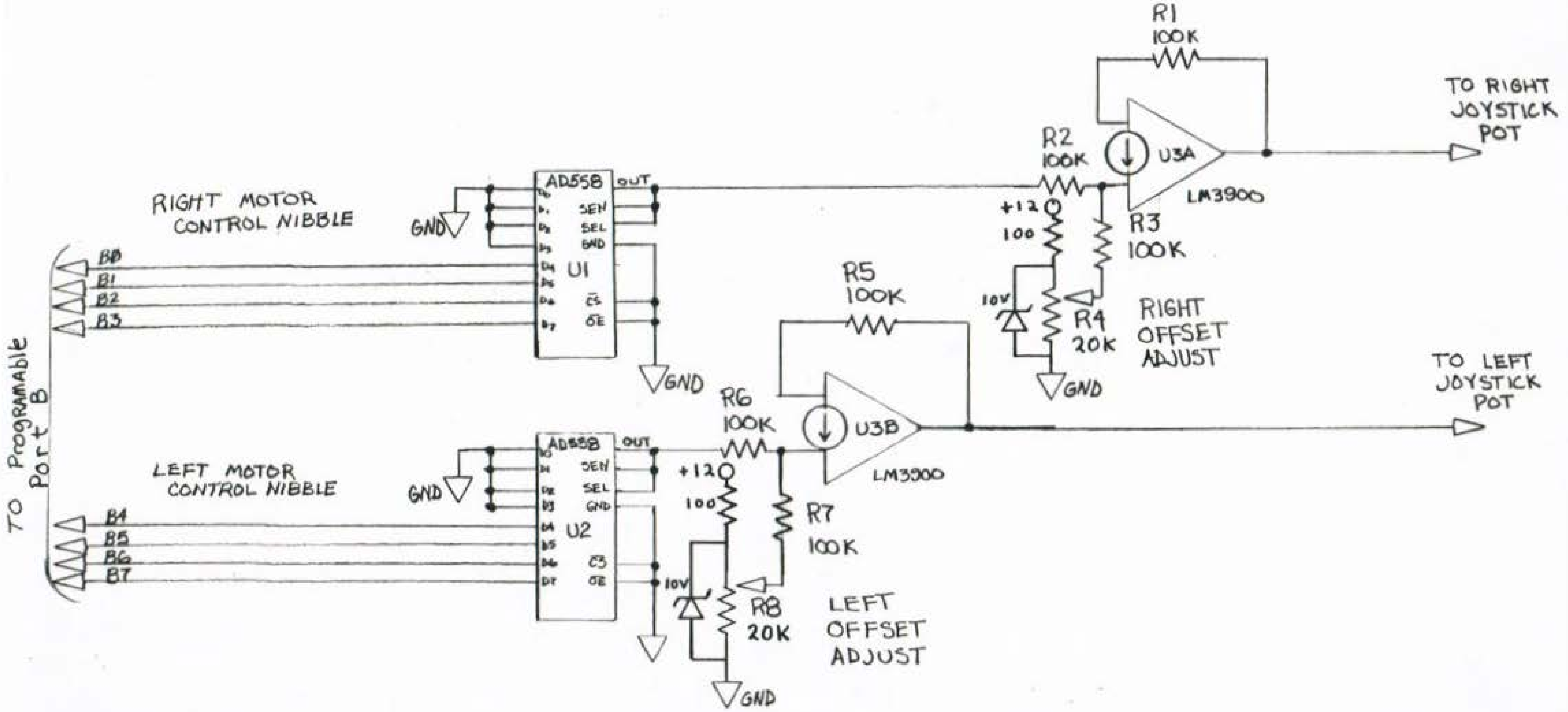
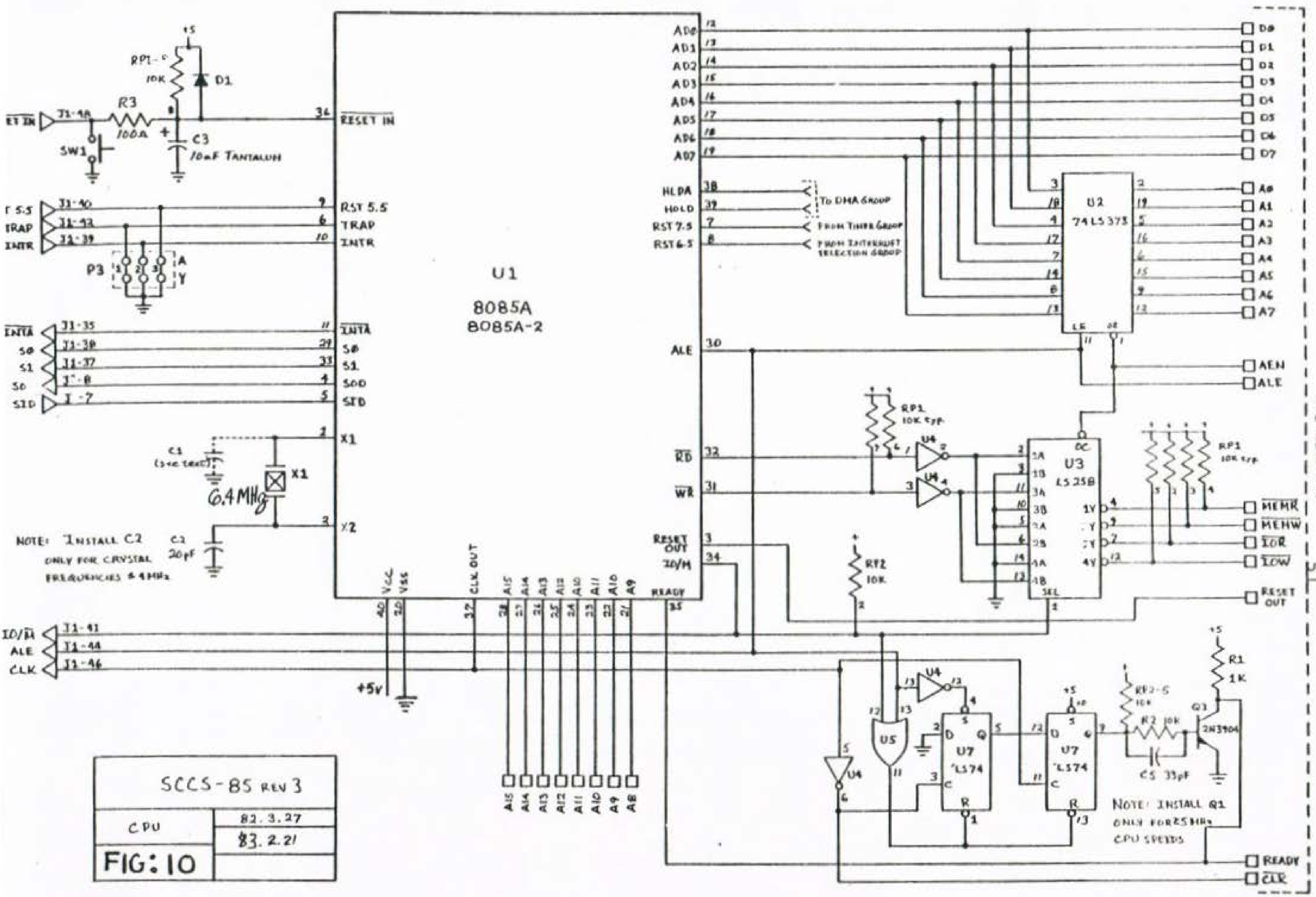


FIG: 8	DESIGN BY: YPW	TITLE: Tone Generator
DATE: 12/18/85	DRAWN BY: Y.PW	



* NOTE: ALL RESISTANCES ARE IN Ohms.
I.C. Power CONNECTIONS Not Shown.

FIG: 9	DESIGN BY: YPW	TITLE: Motor Control
DATE: 12/18/85	DRAWN BY: YPW	



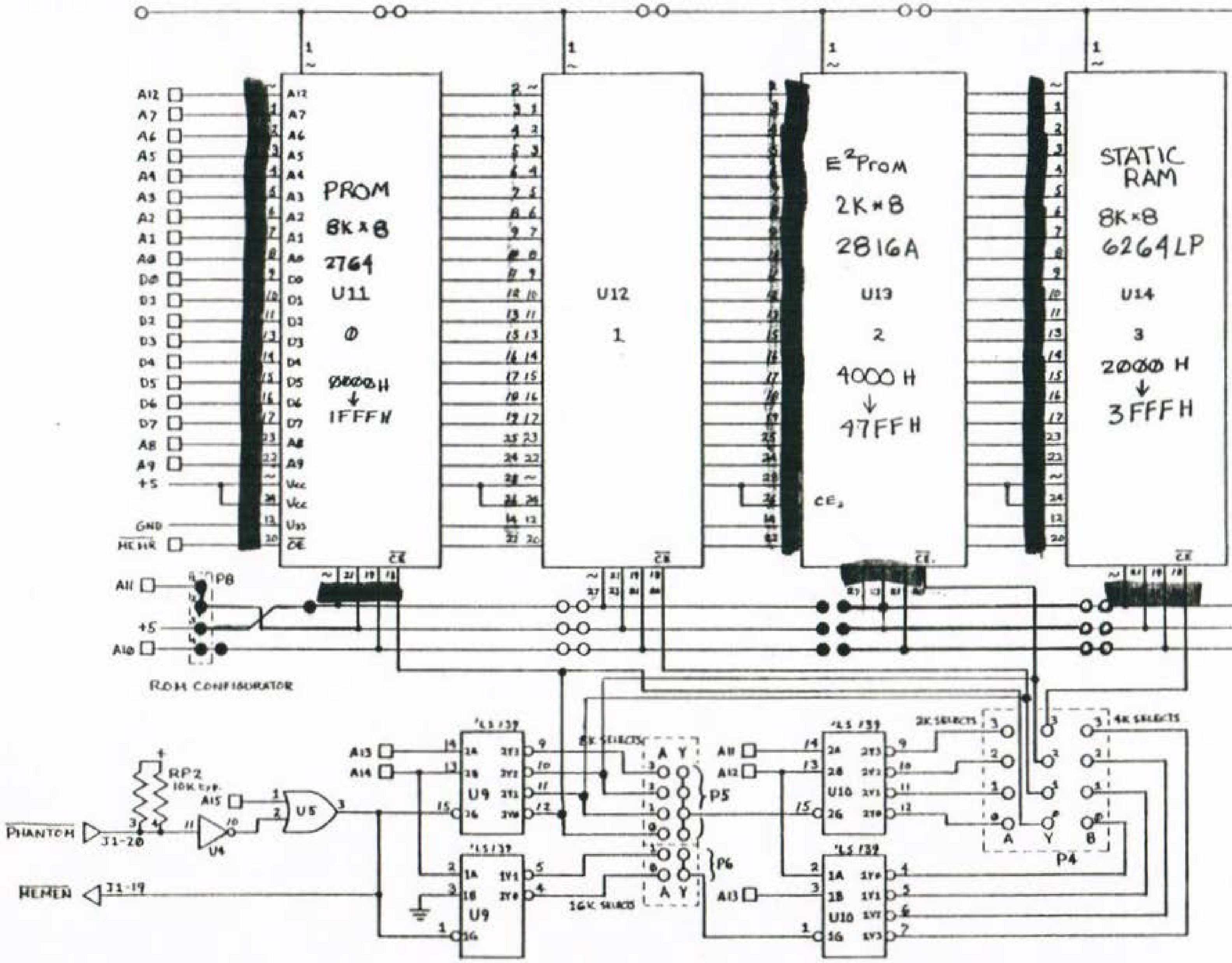
SCCS-85 REV 3

CPU

82.3.27

83.2.21

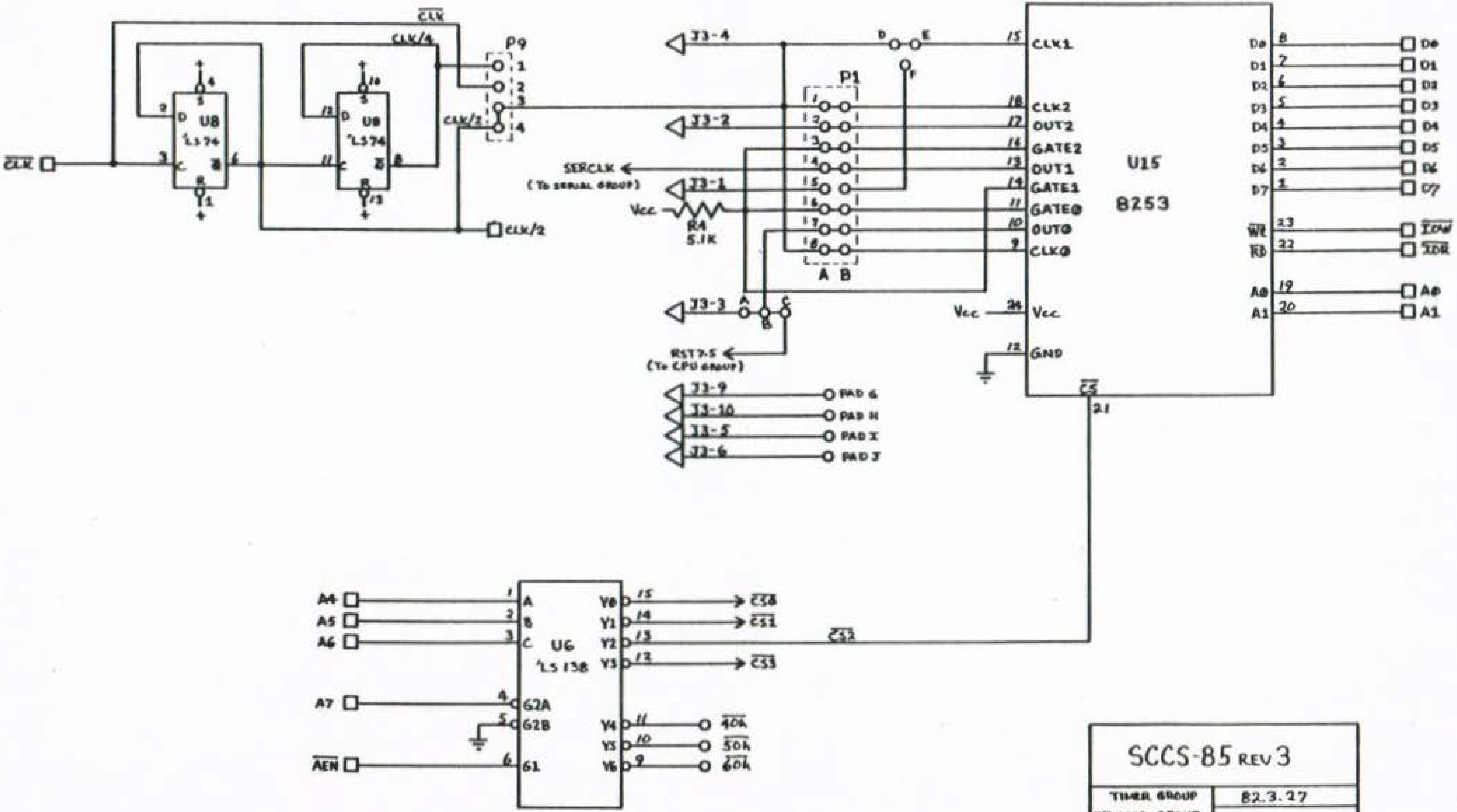
FIG: 10



SCCS-85 REV 3

MEMORY 82.3.27

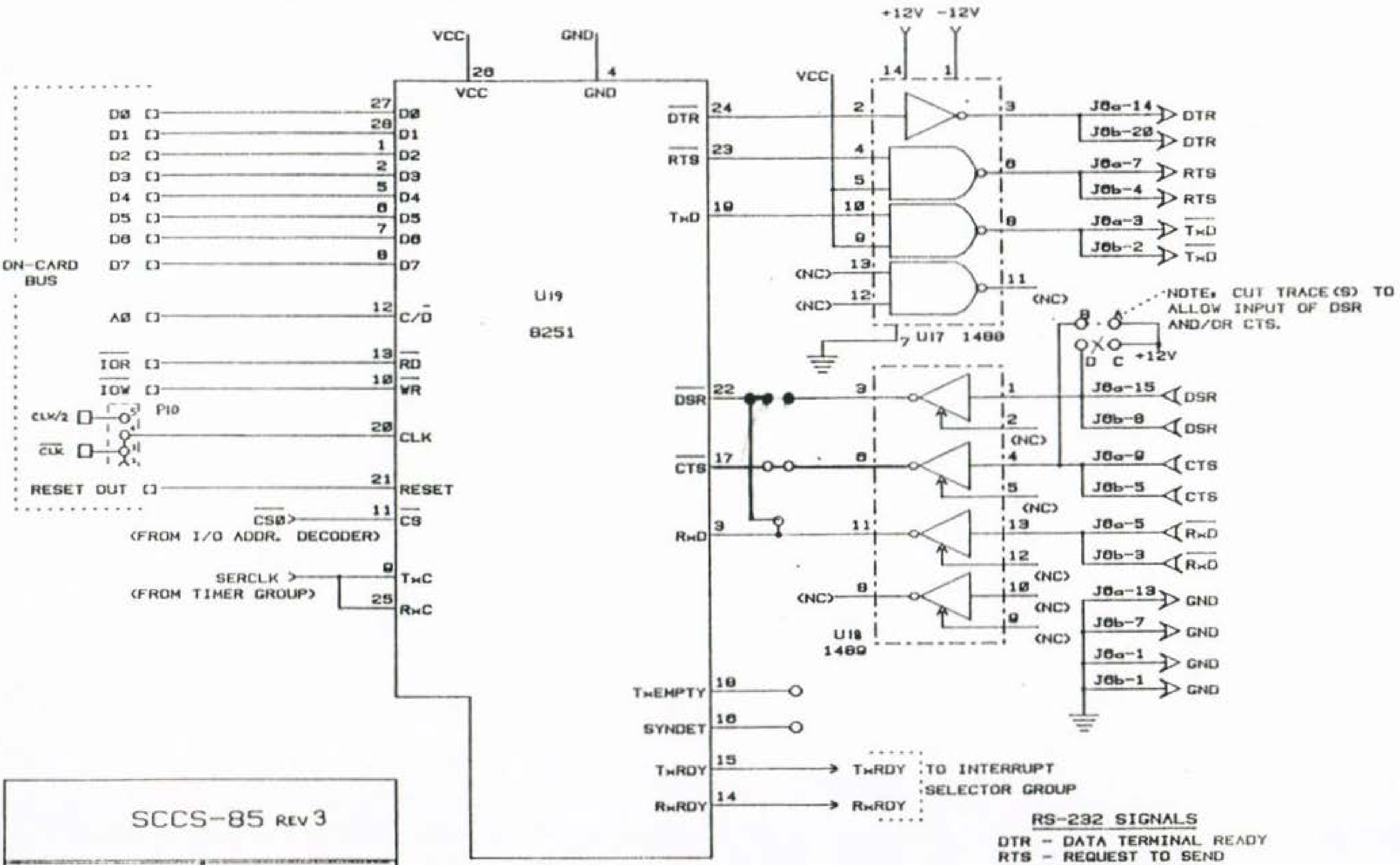
FIG: 11



SCCS-85 REV 3

TIMER GROUP I/O ADDR. GROUP	82.3.27

FIG:12



RS-232 SIGNALS

DTR - DATA TERMINAL READY
RTS - REQUEST TO SEND
TxD - TRANSMIT DATA
DSR - DATA SET READY
CTS - CLEAR TO SEND
RxD - RECEIVED DATA

SCCS-85 REV 3		
SERIAL GROUP	80. 1. 31	
	CORRECTED CTS PIN NO. 81.1.1	UPDATED FOR REV 3 83.3.27
FIG: 13		

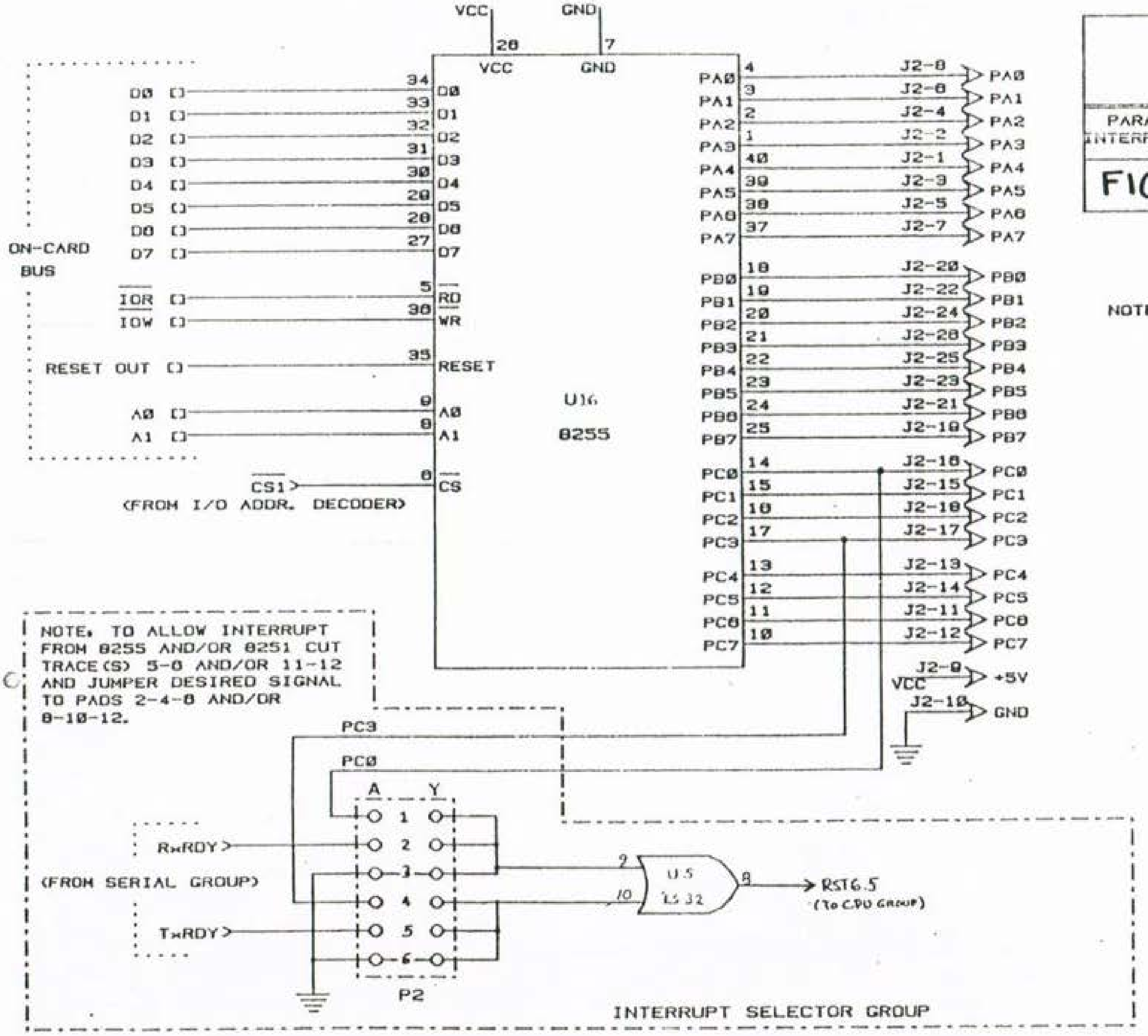
SCCS-85 REV 3

PARALLEL GROUP. INTERRUPT SELECTOR GROUP

FIG:14

80. 1. 31

UPDATED FOR REV 3 83.1.27



NOTE. PIN NUMBERS (SHOWN FOR J2) ALSO APPLY TO J4.

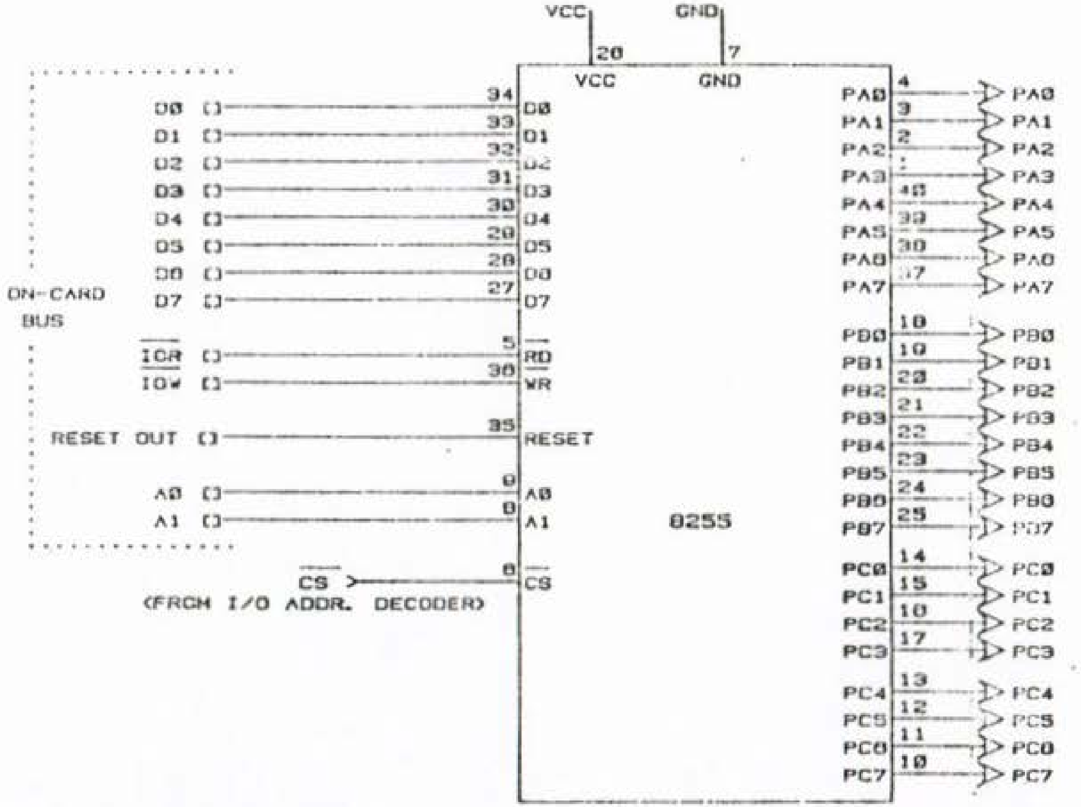
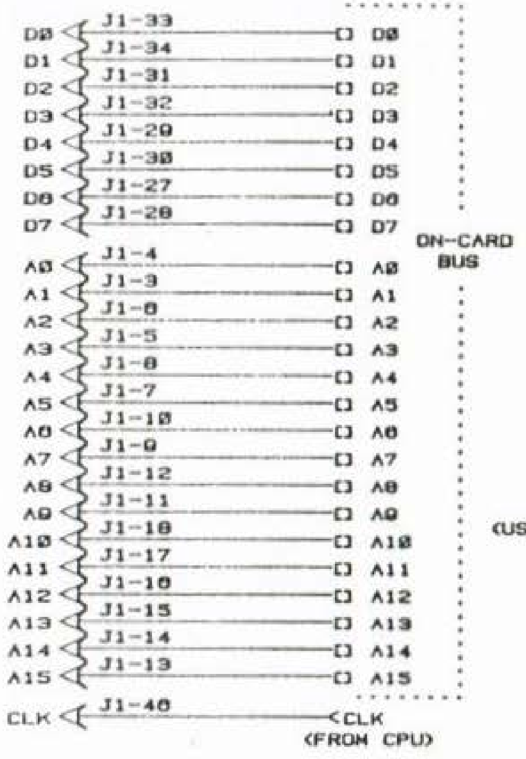
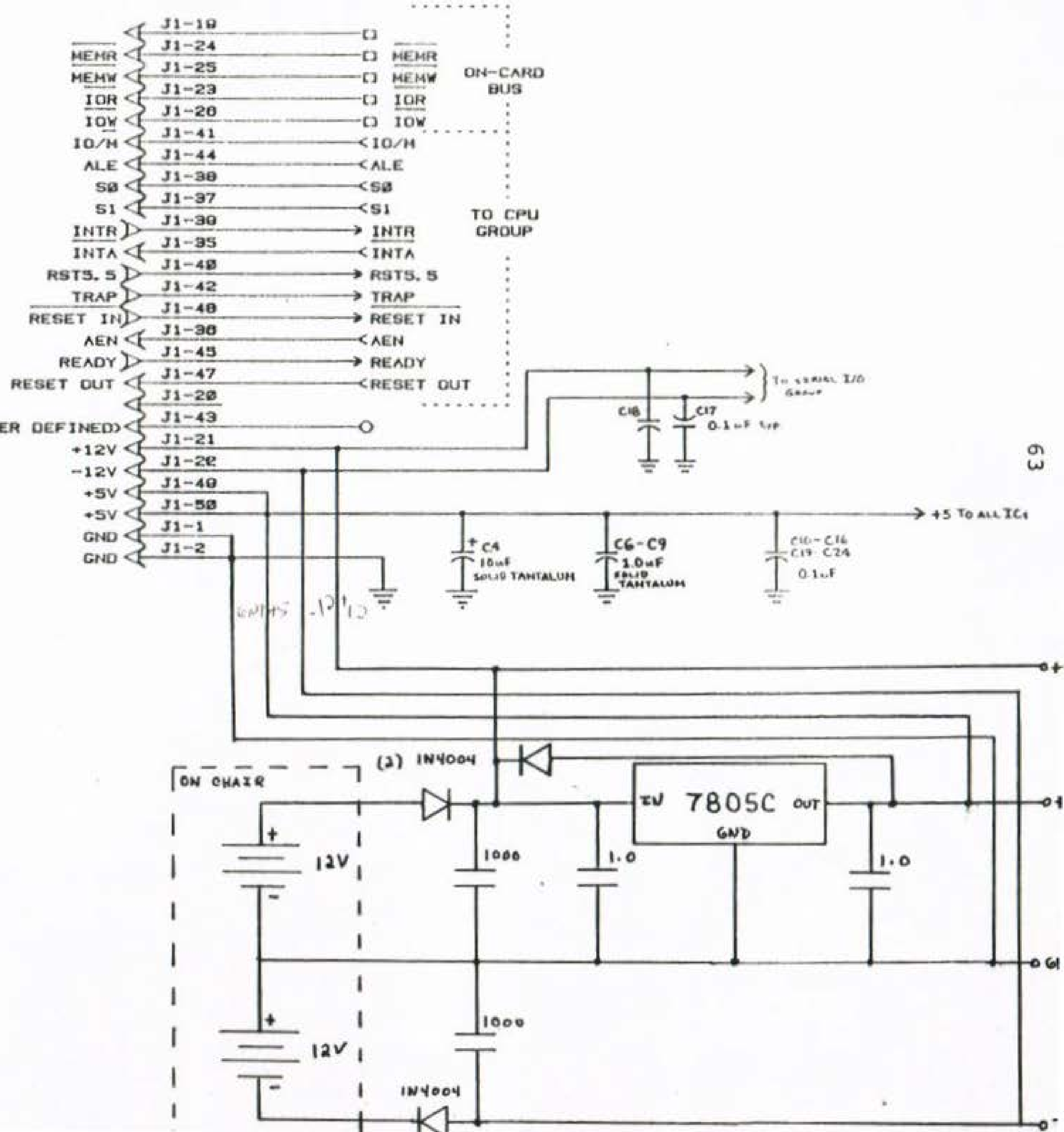


FIG: 15	DESIGN BY: JPW	TITLE: SCHEMATIC:
DATE: 12/15/85	DRAWN BY: GFW	ADDITIONAL PARALLEL GROUP





SCCS-85 REV 3

BUS CONNECTOR, POWER SUPPLY	80.2.1	
FIG: 16	ADDED DECOUPLING CAPACITORS 82.12.3	UPDATED FOR REV 3 83.3.27

E.E.T. 490/491 SENIOR DESIGN PROJECT

THE EASY CHAIR

APPENDIX B: CURRENT SOFTWARE

```
                ;***********************************************************
                ;*          EASYCHAIR THE BEST IN CHAIRS                  *
                ;***********************************************************
                ;
21ØØ =          BASE   EQU     21ØØH   ;BASE ADDRESS OF MONITOR
37ØØ =          MONRAM EQU     37ØØH   ;BASE ADDRESS OF MRSIZ BYTES FOR MONITOR
3FFF =          ENDRAM EQU     3FFFH   ;END OF RAM MEMORY
Ø1ØØ =          MRSIZ  EQU     Ø1ØØH   ;MONITOR RAM SIZE
38ØØ =          USRRAM EQU     MONRAM+1ØØH   ;FIRST BYTE OF USER RAM
ØØFF =          EOL    EQU     ØFFH    ;END OF STRING (LINE) CHARACTER
ØØØ7 =          BEL    EQU     Ø7H     ;BEEEEEEEEEEEEEEP
ØØØD =          CR     EQU     ØDH     ;CARRIAGE RETURN
ØØØA =          LF     EQU     ØAH     ;LINE FEED
ØØ1C =          HOME   EQU     Ø1CH    ;CURSOR UP AND LEFT
ØØ1B =          ESC    EQU     Ø1BH    ;ESCAPE
ØØ7F =          RUB    EQU     Ø7FH    ;RUBOUT
ØØ13 =          XOFF   EQU     Ø13H    ;DC3 (X-OFF)
ØØ11 =          XON    EQU     Ø11H    ;DC1 (X-ON)
ØØØF =          MWIDTH EQU     ØFH     ;CONTROLS THE WIDTH OF "DUMP" "PUNCH"
                ;                      ;COMMANDS:
                ;                      ;           ØFH = 16 BYTES, 52 COLUMNS
                ;                      ;           Ø7H = 8 BYTES, 28 COLUMNS
ØØ2Ø =          TIMEØ  EQU     2ØH     ;8253 TIMER ZERO
ØØ21 =          TIME1  EQU     21H     ; TIMER ONE
ØØ22 =          TIME2  EQU     22H     ;  TIMER TWO
ØØ23 =          TIMCTL EQU     23H     ;8253 CONTROL REGISTER
ØØ1Ø =          PIAA   EQU     Ø1ØH    ;PIA A DATA REGISTER
ØØ11 =          PIAB   EQU     Ø11H    ;PIA B DATA REGISTER
ØØ12 =          PIAC   EQU     Ø12H    ;PIA C DATA REGISTER
ØØ4Ø =          PIAD   EQU     Ø4ØH    ;PIA D DATA REGISTER
ØØ41 =          PIAE   EQU     Ø41H    ;PIA E DATA REGISTER
ØØ42 =          PIAF   EQU     Ø42H    ;PIA F DATA REGISTER
ØØ43 =          PIBCNTL EQU    Ø43H    ;#2 PIA CONTROL REGISTER
ØØ13 =          PIACNTL EQU    Ø13H    ;#1 PIA CONTROL REGISTER
ØØØ1 =          SERCON EQU     Ø1H     ;ACIA CONTROL REGISTER
ØØØØ =          SERDAT EQU     ØØH     ;ASIA DATA REGISTER
ØØØ1 =          PROMSK EQU     ØØØØØØØ1B       ;PROGRAM MENU DETECT
ØØØ1 =          BEAMSK EQU     ØØØØØØØ1B       ;MASK FOR DETECT (PIAB BØ)
ØØ8Ø =          BADMSK EQU     1ØØØØØØØB       ;MASK FOR PAD ERROR (PIAA B7)
ØØ1Ø =          ROWMSK EQU     ØØØ1ØØØØB       ;MASK FOR ROW SELECT (PIAA)
ØØ2Ø =          COLMSK EQU     ØØ1ØØØØØB       ;    COLUMN SELECT (PIAA)
ØØ4Ø =          EXTMSK  EQU    Ø1ØØØØØØB       ;    EXTRA SELECT (PIAA)
                                               ;EXTRA SELECT INCLUDES:
                                               ;MENU SELECT LEDS/TRANS.
                                               ;ULTRASOUND DIRECTION LEDS
ØØ8Ø =          TOUCH  EQU     1ØØØØØØØB       ;MASK FOR A TOUCH [HL]
ØØ4Ø =          MENERR EQU     Ø1ØØØØØØB       ;MASK MENU ERROR
ØØ8Ø =          ERRLED EQU     1ØØØØØØØB       ;MENU ERROR LED MASK
ØØ2Ø =          PADERR EQU     ØØ1ØØØØØB       ;MASK LED/TRANS. ERROR
ØØFF =          TRUE   EQU     ØFFH            ;TRUE IS FF HEX
ØØØØ =          FALSE  EQU     ØØH             ;FALSE IS ØØ HEX
ØØ18 =          ZERO   EQU     18H             ;ZERO LOCATION
ØØ1A =          ONE    EQU     1AH             ;ONE
ØØ1C =          TWO    EQU     1CH             ;TWO
ØØ1E =          THREE  EQU     1EH             ;THREE
ØØ38 =          FOUR   EQU     38H             ;FOUR
```

```
003A =          FIVE   EQU    3AH             ;FIVE
003C =          SIX    EQU    3CH             ;SIX
003E =          SEVEN  EQU    3EH             ;SEVEN
0058 =          EIGHT  EQU    58H             ;EIGHT
005A =          NINE   EQU    5AH             ;NINE
005C =          AHEX   EQU    5CH             ;A (TEN) IN HEX
005E =          BHEX   EQU    5EH             ;B (ELEVEN)
0078 =          CHEX   EQU    78H             ;C (TWELVE)
007A =          DHEX   EQU    7AH             ;D (THIRTEEN)
007C =          EHEX   EQU    7CH             ;E (FOURTEEN)
007E =          FHEX   EQU    7EH             ;F (FIFTEEN)
0072 =          TOGON  EQU    72H
0075 =          TOGOFF EQU    75H
00AD =          RIGHT  EQU    0ADH
00A9 =          LEFT   EQU    0A9H
00C9 =          FRONT  EQU    0C9H
00CD =          BACK   EQU    0CDH
0036 =          RANGE  EQU    36H
0056 =          SPEED  EQU    56H
0016 =          SOUND  EQU    16H
0096 =          RRATE  EQU    96H
00B6 =          SDELAY EQU    0B6H
                ;=============================================================
                ; VECTORS FOR HARDWARE INTERRUPTS
3800 =          RST0   EQU    USRRAM+   000H   ; NOT USED - MONITOR RESET
3808 =          RST1   EQU    USRRAM+   008H   ;
3810 =          RST2   EQU    USRRAM+   010H   ;
3818 =          RST3   EQU    USRRAM+   018H   ;
3820 =          RST4   EQU    USRRAM+   020H   ;
3824 =          TRAP   EQU    USRRAM+.  024H   ;
3828 =          RST5   EQU    USRRAM+   028H   ;
382C =          RST55  EQU    USRRAM+   02CH   ;
3830 =          RST6   EQU    USRRAM+   030H   ;
3834 =          RST65  EQU    USRRAM+   034H   ;
3838 =          RST7   EQU    USRRAM+   038H   ;
383C =          RST75  EQU    USRRAM+   03CH   ;
                ;=============================================================
                ; RST 0 ENTRY POINT - POWER UP RESET   ;RST 0
2100                   ORG    BASE+0
2100 310038            LXI    SP,MONRAM+MRSIZ        ;TEMP INIT OF SP
2103 C39721            JMP    ENTRY
2106 00                NOP
2107 00                NOP
                ;==============================================================
                ; RST 1 ENTRY POINT
2108                   ORG    BASE+08H               ; RST 1
2108 C30838            JMP    RST1
210B 0000000000        DB     0,0,0,0,0
                ;==============================================================
                ; RST 2 ENTRY POINT
2110                   ORG    BASE+10H               ; RST 2
2110 C31038            JMP    RST2
2113 0000000000        DB     0,0,0,0,0
                ;=============================================================
                ; RST 3 ENTRY POINT
2118                   ORG    BASE+18H               ; RST 3
```

```
2118 C31838          JMP     RST3
211B 0000000000      DB      0,0,0,0,0
                ;===============================================================
                ; RST 4 ENTRY POINT
2120                 ORG     BASE+20H                ; RST 4
2120 C32038          JMP     RST4
2123 00              NOP
                ;===============================================================
                ; TRAP ENTRY POINT
2124                 ORG     BASE+24H                ; TRAP
2124 C32438          JMP     TRAP
2127 00              NOP
                ;===============================================================
                ; RST 5 ENTRY POINT
2128                 ORG     BASE+28H                ; RST 5
2128 C32838          JMP     RST5
212B 00              NOP
                ;===============================================================
                ; RST 5.5 ENTRY POINT
212C                 ORG     BASE+2CH                ; RST 5.5
212C C32C38          JMP     RST55
212F 00              NOP
                ;===============================================================
                ; RST 6 ENTRY POINT
2130                 ORG     BASE+30H                ; RST 6
2130 C33038          JMP     RST6
2133 00              NOP
                ;===============================================================
                ; RST 6.5 ENTRY POINT
2134                 ORG     BASE+34H                ; RST 6.5
2134 C33438          JMP     RST65
2137 00              NOP
                ;===============================================================
                ; RST 7 ENTRY POINT
2138                 ORG     BASE+38H                ; RST 7
2138 C33838          JMP     RST7
213B 00              NOP
                ;===============================================================
                ; RST 7.5 ENTRY POINT                        ;RST 5.5
213C                 ORG     BASE+3CH
213C C33C38          JMP     RST75
213F 00              NOP
                ;===============================================================
2140                 ORG     BASE+40H
                ; JUMP TABLE FOR MONITOR SUBROUTINES
                ;        ALL REFERENCES TO THESE LABELS SHOULD GO THROUGH THIS
                ;        SO THAT CHANGES IN THE ACTUAL ROUTINE'S LOCATION IN
                ;        FUTURE VERSIONS OF THE MONITOR DO NOT EFFECT NON-MONITOR
                ;        PROGRAMS.  THESE LOCATIONS WILL NEVER CHANGE.
2140 C3B32B          JMP     CI
2143 C3D32B          JMP     CO
2146 C32D2C          JMP     CRLF    ;PRINTS (CR) (LF)
2149 C3452D          JMP     GHW     ;WORD RET IN H&L OR CY=1 & BAD CHAR IN A
214C C35C2D          JMP     GHB     ;BYTE RET IN A OR CY=1 & BAD CHAR IN A
214F C3712D          JMP     GHD     ;DIGIT RET IN A OR CY=1 & BAD CHAR IN A
2152 C3AB2D          JMP     MSG     ;ADDRESS OF EOL TERMINATED MSG IN D&E
```

```
2155 C3E42D          JMP     PHW     ;WORD PASSED IN H&L
2158 C3EF2D          JMP     PHB     ;BYTE PASSED IN A
215B C3012E          JMP     PHD     ;DIGIT PASSED IN A
215E C35A2E          JMP     SPACE   ;PRINT SPACE
2161 C3742E          JMP     SUB16   ;(H&L) <- (H&L) - (D&E)
2164 C3802E          JMP     UCASE   ;UPPER TO LOWER CASE CONVERSION
2167 C3742D          JMP     ATH     ;ASCII TO HEX CONVERSION
216A C3ED21          JMP     WARMST  ;BEGINNING OF MONITOR COMMAND LOOP
               ;                     ;NOTHING RESET - STACK OR ANY MON RAM
216D C3202C          JMP     CMP16   ;UNCOMMENT WHEN CMP16 ROUTINE INCLUDED
2170 C37E24          JMP     DUMP1   ;MEMORY DUMP
2173 C3F223          JMP     LOAD1   ;INTEL LOADER
2176 C32B23          JMP     MEMED   ;MEMORY EDITOR
2179 C3C92B          JMP     CISTAT  ;RETURNS NON-ZERO IF REC BUFFER FULL
217C C3122E          JMP     POPPC   ;RETURNS WITH RETURN ADDRESS IN H&L
217F C3B22B          JMP     CALLIN  ;INDIRECT CALL TO (H&L)
2182 C3362E          JMP     SHRHL   ;SHIFT RIGHT H&L
2185 C32B2E          JMP     RNDHL   ;ADD CARRY FLAG TO H&L
2188 C3832B          JMP     BCDTBIN ;CONVERT BCD IN H&L TO BINARY IN H&L
218B C3182A          JMP     PADCK   ;PAD DIAG.
218E C39827          JMP     ULTRA   ;ULTRASONIC BUMPER
2191 C3F122          JMP     TSTBRD  ;CHECK BOARD
2194 C37E22          JMP     MEMTST  ;RAM MEMORY TEST


               ;
               ; POWER-UP AND RESET INITIALIZATION
               ;
               ; NOW INITIALIZE USART CHIP
               ;
2197 3E82      ENTRY: MVI    A,082H  ;FORCE USART TO EXPECT CMND WORD
2199 D301            OUT     SERCON
219B 3E40            MVI     A,040H  ;NOW MAKE USART TO EXPECT MODE WORD
219D D301            OUT     SERCON
219F 3ECE            MVI     A,0CEH  ;MODE BYTE -
21A1 D301            OUT     SERCON  ;  11 00 11 10
21A3 3E37            MVI     A,037H  ;COMMAND BYTE -
21A5 D301            OUT     SERCON  ;  0 0 1 1 0 1 1 1
               ;
               ; INITIALIZE TIMER CHIP TO GENERATE 16X BAUDRATE FOR
               ;
21A7 210E00          LXI     H,000EH         ; 7200 BAUD
                                             ;[1/(16*7200)]/[1/3.2 MHZ]
21AA 3E76            MVI     A,76H           ;INIT TIMER 1 TO DIVIDE BY N
21AC D323            OUT     TIMCTL          ;
21AE 7D              MOV     A,L             ;
21AF D321            OUT     TIME1           ;
21B1 7C              MOV     A,H             ;
21B2 D321            OUT     TIME1           ;
               ;
               ; INITIALIZE MONITOR RAM PERTAINING TO CONSOLE I/O
               ;
21B4 AF              XRA     A               ;MAKE A ZERO
21B5 320137          STA     DLYRAM          ; NUMBER OF 10MS DELAYS ON <CR>
21B8 320237          STA     ECHOFL          ; 0=ECHO 1=NO ECHO
21BB 3E0F            MVI     A,MWIDTH        ; INITIALIZE WIDTH
21BD 320337          STA     WIDTH           ;   .
```

```
21C0 320037            STA    COCOOK          ; 0=COOKED 1=RAW
                ;
                ; PRINT STARTUP MESSAGE - ALSO EFFECTIVE WAY TO WAIT A FEW
                ;                       CHAR PERIODS WHILE DOUBLE BUFFERED
                ;                       INPUT SETTLES.
21C3 118A31            LXI    D,START         ;PRINT STARTUP MESSAGE
21C6 CDAB2D            CALL   MSG             ;
21C9 DB00              IN     SERDAT          ;EAT POSSIBLE GARBAGE CHARACTER
                ;
                ; INITIALIZE REMAINDER OF MONITOR RAM AND STACK POINTER
                ;
21CB AF                XRA    A               ; ON POWER UP SET TO LOAD
21CC 320637            STA    VFYFLG          ; 0=LOAD, 1=VERIFY
21CF 320037            STA    COCOOK          ; 0=COOKED 1=RAW
21D2 3EFF              MVI    A,EOL           ; ON POWER UP NO ANSWER
21D4 322637            STA    MISCBF          ;
21D7 210020            LXI    H,2000H         ; INITIALIZE
21DA 220F37            SHLD   CLKBCD          ;CLOCK FREQ IN BCD
21DD 21D007            LXI    H,2000          ;
21E0 221137            SHLD   CLKBIN          ;       AND   BINARY
21E3 23                INX    H               ; PULSE TIMING VERY SMALL IN
21E4 220D37            SHLD   D50DIV          ;   CASE SOMETHING GOES WRONG
21E7 CD182A            CALL   PADCK           ; PAD TEST ON POWERUP
                ;      LXI    H,ENDRAM        ;UNCOMMENT FOR MEM TEST
                ;      LXI    D,USRRAM        ; ON RESET/POWER UP
                ;      CALL   MT0
21EA CD622E            CALL   STACKI          ;REAL INITIALIZATION OF SP
                ;
                ; PRINT WARMSTART MESSAGE...
                ;
                ;      NOTHING INITIALIZED
                ;
21ED 114E33     WARMST: LXI   D,STKAT         ;PRINT LOCATION OF STACK
21F0 CDAB2D            CALL   MSG             ;
21F3 210000            LXI    H,0             ;
21F6 39                DAD    SP              ;
21F7 CDE42D            CALL   PHW             ;
                ;
                ; COMMAND LEVEL - GET CHARACTER; JUMP TO APPROPRIATE ROUTINE


21FA CD4B2E     COMND: CALL   SETJMP          ;RUBOUT ABORTED COMNDS COME HERE

21FD 117E31            LXI    D,PRMPT         ;PRINT COMMAND PROMPT
2200 CDAB2D            CALL   MSG
2203 CDB32B            CALL   CI      ;
                ;      ANI    7FH     ;PUT IN IF UCASE TAKEN OUT
2206 CD802E            CALL   UCASE   ;CONVERT LOW TO UP CASE & STRIPS PARITY
                ;
                ; SEQUENCE BELOW IS KLUDGE TO ALLOW CR AND ? AS ONE CHAR COMNDS
                ;
2209 FE0D              CPI    CR      ;SPECIAL CASE, (CR) IS NOP THAT DOES NOT
220B CAFA21            JZ     COMND   ;  CLEAR THE ANSWER
220E 11FA21            LXI    D,COMND ;ADDR FOR PSEUDO CALL COMPLETED BY PCHL
2211 D5                PUSH   D       ;
2212 FE3F              CPI    '?'     ;SPECIAL CASE '?', MUST NOT CLEAR
```

```
2214 CA4922         JZ      ASK     ; ANSWER FIRST.
              ;
              ; NOW FOR THE REAL COMMANDS...
              ;
2217 67             MOV     H,A     ;PUT FIRST CHAR INTO H
2218 CDB32B         CALL    CI      ;GET SECOND CHAR
              ;     ANI     Ø7FH    ;UNCOMMENT IF CALL UCASE REMOVED
221B CD8Ø2E         CALL    UCASE   ;
221E 6F             MOV     L,A     ;PUT SECOND CHAR INTO L
221F CD5A2E         CALL    SPACE   ;GOD KNOWS WHAT FOR...
2222 Ø18C2E         LXI     B,CMDS  ;SCAN COMMAND TABLE...COMND IN H&L
2225 ØA       CMDNXT: LDAX  B       ;GET COMMAND FROM TABLE
2226 57             MOV     D,A     ;  GET FIRST LETTER
2227 Ø3             INX     B       ;  POINT TO SECOND LETTER
2228 ØA             LDAX    B       ;  GET SECOND LETTER
2229 5F             MOV     E,A     ;  .
222A Ø3             INX     B       ;  POINT TO LOWER BYTE OF ADDRESS
222B CD2Ø2C         CALL    CMP16   ;COMPARE TO COMND TYPED
222E CA3C22         JZ      CMDFND  ;FOUND IT
2231 Ø3             INX     B       ;SKIP OVER ADDR OF COMMAND JUST CHECKED
2232 Ø3             INX     B       ;POINT TO UPPER BYTE OF ADDR THEN NXT CMD
2233 7A             MOV     A,D     ;CHECK FOR END OF TABLE
2234 B3             ORA     E       ;
2235 C22522         JNZ     CMDNXT  ;NOT END...TRY NEXT ENTRY
2238 CD142E   ERRER: CALL   PRBAD   ;PRINT ERRER MESSAGE AND RETURN.  "COMND"
223B C9             RET             ;  IS ON STACK AS RETURN ADDR FOR COMMAND
              ;                     ;  NOTE ALL THE COMMANDS USE ERRER LABEL.
              ;
223C 3EFF     CMDFND: MVI   A,EOL   ;CLEAR ANSWER
223E 322637         STA     MISCBF  ;
2241 ØA             LDAX    B       ;GET LOWER BYTE OF ADDRESS
2242 5F             MOV     E,A     ; .
2243 Ø3             INX     B       ;POINT TO LOWER BYTE
2244 ØA             LDAX    B       ;GET UPPER BYTE
2245 57             MOV     D,A     ; .
2246 7C             MOV     A,H     ;COMMAND EXPECTS FIRST LETTER IN A REG
2247 EB             XCHG            ;
2248 E9             PCHL            ;
              ;
              ;****************** END OF COMMAND LEVEL ************************

              ;************************BEGINNING OF ASK*************************
              ;
              ;      PRINT   ONE BYTE NOTE LEFT BY LAST COMMAND
              ;
2249 CD5A2E   ASK:   CALL   SPACE
224C 112637          LXI    D,MISCBF
224F CDAB2D          CALL   MSG
2252 C9              RET
              ;*************************END OF ASK********************************

              ;*************************BEGINNING OF HELP************************
              ;
              ; HELP
2253 11Ø73Ø   HELP:  LXI    D,PHELP
2256 CDAB2D          CALL   MSG
```

```
2259 C9                RET
                ;************************END OF HELP**********************************

                ;************************BEGINNING OF GOTO******************************
                ;
                ; GOTO ROUTINE - STARTS EXECUTION IN MEMORY LOCATION

225A CD452D     GOTO:  CALL    GHW     ;GET HEX WORD
225D DA3822            JC      ERRER   ;
226Ø CDC42D            CALL    OKCK    ;
2263 D8                RC              ;
2264 E5                PUSH    H       ;SAVE GOTO ADDRESS
2265 114E33            LXI     D,STKAT ;GET STACKPOINTER AND PRINT
2268 CDAB2D            CALL    MSG     ;
226B 21Ø2ØØ            LXI     H,Ø2    ;
226E 39                DAD     SP      ;
226F CDE42D            CALL    PHW     ;
2272 CD2D2C            CALL    CRLF    ;
2275 AF                XRA     A       ;PRINT DUMMY CHARACTER SO THAT PROGRAM
2276 CDD32B            CALL    CO      ;CANNOT PREVENT END OF CRLF FROM PRINTING
2279 CDD32B            CALL    CO      ;
227C E1                POP     H       ;GET ADDRESS...
227D E9                PCHL            ;               AND GO
                ;************************END OF GOTO***********************************
                ;
                ;************************BEGINNING OF MEMTST*************************
                ;
227E CD8F2C     MEMTST: CALL   FROMTO          ;GET FROM AND TO ADDRESSES
2281 DA3822            JC      ERRER           ;
2284 EB                XCHG                    ;
2285 CDC42D            CALL    OKCK            ;CHECK WITH USER BEFORE STARTING
2288 DAFØ22            JC      MTEND           ;
228B 4C         MTØ:   MOV     C,H             ;STOP AT XX?? WHERE XX-1 IS THE
228C ØC                INR     C               ;UPPER BYTE OF THE USERS TO ADDR
228D Ø6ØØ              MVI     B,ØØH           ; ALSO USE OF COUNTER
228F C5                PUSH    B
229Ø Ø6ØØ              MVI     B,Ø             ;CLEAR B PATTERN MODIFIER
2292 62         MT1:   MOV     H,D             ;
2293 6B                MOV     L,E             ;
2294 7D         MTFILL: MOV    A,L             ;LOW BYTE TO ACCUM.
2295 AC                XRA     H               ;XOR WITH HIGH BYTE
2296 A8                XRA     B               ;XOR WITH PATTERN
2297 77                MOV     M,A             ;STORE IN ADDR
2298 23                INX     H               ;INCREMENT ADDR
2299 7C                MOV     A,H             ;LOAD HIGH BYTE OF ADDR
229A B9                CMP     C               ;COMPARE WITH STOP ADDR
229B C29422            JNZ     MTFILL          ;LOOP IF NOT DONE
                ;
                ; READ AND CHECK TEST DATA
                ;
229E 62                MOV     H,D
229F 6B                MOV     L,E             ;GET STARTING ADDR
22AØ 7D         MTTST: MOV     A,L             ;GET LOW BYTE
22A1 AC                XRA     H               ;XOR WITH HIGH BYTE
22A2 A8                XRA     B               ;XOR WITH MODIFIER
22A3 C5                PUSH    B               ;
```

```
22A4 47              MOV     B,A
22A5 7E              MOV     A,M
22A6 B8              CMP     B              ;COMPARE WITH MEMORY LOCATION
22A7 C2D222          JNZ     MTFXIT         ;ERRER EXIT
22AA C1              POP     B
22AB 23              INX     H              ;UPDATE MEMORY ADDRESS
22AC 7C              MOV     A,H            ;GET HIGH BYTE
22AD B9              CMP     C              ;COMPARE WITH STOP ADDR
22AE C2A022          JNZ     MTTST          ;LOOP BACK
22B1 3A0337          LDA     WIDTH          ;GENERATE ((WIDTH+1)*4)-1
22B4 37              STC                    ; .
22B5 17              RAL                    ; .
22B6 37              STC                    ; .
22B7 17              RAL                    ; .
22B8 A0              ANA     B              ;CHECK FOR TIME FOR CRLF
22B9 CC2D2C          CZ      CRLF           ;CRLF IF RUNNING OUT OF LINE
22BC 04              INR     B              ;UPDATE MODIFIER
22BD EB              XCHG
22BE 3E21            MVI     A,'!'          ;PRINT PASS DONE MESSAGE
22C0 CDD32B          CALL    CO             ;
22C3 EB              XCHG
22C4 C1              POP     B
22C5 05              DCR     B
22C6 C5              PUSH    B
22C7 C29222          JNZ     MT1            ;RESTART WITH NEW MODIFIER
22CA C1              POP     B
22CB 11AA2F          LXI     D,MTGOOD
22CE CDAB2D          CALL    MSG
22D1 C9              RET                    ; FOR 255 TIMES THEN TO CMDS
22D2 11C12F  MTFXIT: LXI     D,MTERR.       ;PRINT ERRER ADDRESS
22D5 CDAB2D          CALL    MSG
22D8 CDE42D          CALL    PHW
22DB 11E32F          LXI     D,MTREAD
22DE CDAB2D          CALL    MSG
22E1 CDEF2D          CALL    PHB
22E4 11DA2F          LXI     D,MTWROT
22E7 CDAB2D          CALL    MSG
22EA 78              MOV     A,B
22EB CDEF2D          CALL    PHB
22EE C1              POP     B
22EF C1              POP     B

             ;
22F0 C9      MTEND: RET                     ;RETURN TO COMMAND LOOP
             ;
             ;
             ;******************END OF MEMTST*****************************

             ;******************BEGINNING OF TEST BOARD******************

22F1 CD2D2C  TSTBRD:         CALL    CRLF
22F4 11DE2E          LXI     D,MTSBRD
22F7 CDAB2D          CALL    MSG
22FA 3E37            MVI     A,37H
22FC D323            OUT     TIMCTL
22FE 3E77            MVI     A,77H
```

```
23Ø0 D323          OUT     TIMCTL
23Ø2 3EB7          MVI     A,ØB7H
23Ø4 D323          OUT     TIMCTL
23Ø6 97            SUB     A
23Ø7 D32Ø          OUT     TIMEØ
23Ø9 D321          OUT     TIME1
23ØB D322          OUT     TIME2
23ØD 3E2Ø          MVI     A,2ØH
23ØF D32Ø          OUT     TIMEØ
2311 D321          OUT     TIME1
2313 D322          OUT     TIME2

2315 3E8Ø          MVI     A,8ØH
2317 D313          OUT     PIACNTL
2319 D343          OUT     PIBCNTL
231B D31Ø  LOOPA:  OUT     PIAA
231D D311          OUT     PIAB
231F D312          OUT     PIAC
2321 D34Ø          OUT     PIAD
2323 D341          OUT     PIAE
2325 D34Ø          OUT     PIAD
2327 ØF            RRC
2328 C31B23        JMP     LOOPA
           ;*****************END OF TEST BOARD *********************

           ;*****************BEGINNING OF MEMED***********************
           ;
           ; MEMED - HEXADECIMAL MEMORY EDITOR
           ;
232B 11DA2E  MEMED: LXI     D,EDM2 .;PRINT "CR, LF, ("
232E CDAB2D         CALL    MSG

2331 CD452D         CALL    GHW
2334 D24Ø23         JNC     OK      ;GET HEX WORD INTO HL, JUMP IF VALID

2337 FE2F           CPI     '/'     ;BAD CHAR RECEIVED - WAS IT "/"
2339 C8             RZ              ;GO BACK TO COMMAND LEVEL IF SO

233A CD142E         CALL    PRBAD   ;PRINT "WHAT ?"
233D C32B23         JMP     MEMED   ;THEN TRY AGAIN

234Ø CDB623  OK:    CALL    DISCON  ;DISPLAY CONTENTS OF LOCATION
2343 CD4923         CALL    EDIT    ;THEN BEGIN EDITING
2346 C32B23         JMP     MEMED   ;LOUPE IF EDIT RETURNS
             ;
             ;      END     MEMED
             ;
             ;
             ; GET EITHER A NEW HEX BYTE TO BE WRITTEN WHERE HL POINTS,
             ; FOLLOWED BY ANOTHER COMMAND, OR JUST ANOTHER COMMAND.
             ;
2349 CD5C2D  EDIT:  CALL    GHB     ;GET THE NEW HEX BYTE IF TYPED
234C D27423         JNC     EDBYTE  ;GOOD BYTE TYPED - PUT IN MEMORY
234F FE27           CPI     Ø27H    ;DOES USER WANT LITERAL CHARACTER ?
2351 CA6F23         JZ      EDLIT   ;  YEP...
2354 FE5E           CPI     '^'     ;DOES USER WANT CONTROL CHARACTER ?
```

```
2356 C27D23          JNZ     NEXT    ;NOPE...MUST BE COMMAND OR ERRER...
2359 CDB32B          CALL    CI      ;GET CHAR
235C E67F            ANI     Ø7FH    ;STRIP PARITY
235E FE4Ø            CPI     Ø4ØH    ;SEE IF MAKES SENSE...
236Ø DAAØ23          JC      EDBAD   ;DUMMY
2363 FE6Ø            CPI     Ø6ØH    ;FIGURE OUT WHAT TO SUBTRACT...
2365 DA6A23          JC      EDUC    ;IS UPPER CASE...OK AS IS
2368 D62Ø            SUI     Ø2ØH    ;LOWER CASE...MUST BE MOVED DOWN
236A D64Ø    EDUC:   SUI     Ø4ØH    ;CONVERT TO CONTROL CHAR
236C C37423          JMP     EDBYTE  ;
236F CDB32B  EDLIT:  CALL    CI      ;GET CHAR
2372 E67F            ANI     Ø7FH    ;BETTER STRIP PARITY
2374 77      EDBYTE: MOV     M,A     ;ELSE STORE IT IN MEMORY
2375 CD5A2E          CALL    SPACE   ;SPACE TO REINFORCE THAT ONCE TWO DIGITS
             ;                       ;  ARE ENTERED, LOCATION IS CHANGED.
2378 CDB32B          CALL    CI      ;AND GET ANOTHER CHAR & ECHO IT
237B E67F            ANI     7FH     ;KILL TOP BIT
237D FEØD    NEXT:   CPI     CR      ;CARRIAGE RETURN?
237F C28623          JNZ     E1
2382 23              INX     H
2383 C3A323          JMP     PR      ;YES- PRINT NEXT LOCATION
2386 FE2Ø    E1:     CPI     ' '     ;OR BLANK
2388 C28F23          JNZ     E2
238B 23              INX     H
238C C3A323          JMP     PR      ;YES- DO THE SAME
238F FE2E    E2:     CPI     '.'     ; PERIOD?
2391 CAA323          JZ      PR      ;PRINT CURRENT LOCATION
2394 FE2D    E3:     CPI     '-'     ; DASH?
2396 C29D23          JNZ     E4
2399 2B              DCX     H
239A C3A323          JMP     PR      ;YES - PRINT PREVIOUS LOCATION
239D FE2F    E4:     CPI     '/'     ;SLASH?
239F C8              RZ              ;EDIT ALL DONE IF SO
23AØ CD142E  EDBAD:  CALL    PRBAD   ;IF NONE OF THE ABOVE, PRINT "WHAT ?"
23A3 CDA923  PR:     CALL    DISMEM  ;DISPLAY THE NEW CURRENT MEMORY LOCATION
23A6 C34923          JMP     EDIT    ;AND LOOP

             ; PRINT CR, LF THEN AN ( FOLLOWED BY THE CONTENTS OF HL IN HEX.

23A9 11DA2E  DISMEM: LXI     D,EDM2  ;DO CR,LF, "("
23AC CDAB2D          CALL    MSG
23AF CDE42D          CALL    PHW
23B2 CDB623          CALL    DISCON
23B5 C9              RET
             ; **** DISCON ****
             ;
             ; PRINT ') = ' FOLLOWED BY THE CONTENTS OF THE MEMORY LOC.
             ; POINTED TO BY HL
             ;
23B6 11D52E  DISCON: LXI     D,EDM1  ;
23B9 CDAB2D          CALL    MSG     ;
23BC 7E              MOV     A,M     ;GET CONTENTS OF MEM LOC.
23BD CDEF2D          CALL    PHB     ;PRINT IT
23CØ 11D62E          LXI     D,EDM3  ;
23C3 CDAB2D          CALL    MSG     ;
23C6 E5              PUSH    H       ;SAVE ADDRESS
```

```
23C7 CD662C             CALL    DISASC  ;CONVERT TO PRINTABLE
23CA 7C                 MOV     A,H     ;PRINT ' ' OR '^'
23CB CDD32B             CALL    CO      ;
23CE 7D                 MOV     A,L     ;PRINT CHARACTER
23CF CDD32B             CALL    CO      ;
23D2 E1                 POP     H
23D3 CD5A2E             CALL    SPACE   ;
23D6 C9                 RET
                ;
                ;***********************END OF MEMED******************************

                ;***********************BEGINNING OF LOADER***********************
                ;
                ; HEX-FORMAT LOADER
                ;       NOTE:   RECORD LENGTH = 00 TAKEN AS EOF
23D7 CDB42C     LOADER: CALL    GBIAS   ;GET BIAS
23DA DA3822             JC      ERRER   ;BAD CHAR - QUIT
23DD 220437             SHLD    BIAS    ;STORE BIAS
23E0 CDC42D             CALL    OKCK    ;CHECK WITH USER BEFORE JUMPING
23E3 D8                 RC              ;
23E4 3A0237             LDA     ECHOFL  ;SAVE ECHO FLAG
23E7 322837             STA     MISCBF+2;MISCBF & MISCBF+1 USED BY ANSWER
23EA 3E11               MVI     A,XON   ;START DATA COMING
23EC 320237             STA     ECHOFL  ;NON-ZERO VALUE (XON) TURNS OFF ECHO
23EF CDD32B             CALL    CO      ;
23F2 CD1A24     LOAD1:  CALL    GETREC  ;READ IN ONE REC, (A) = RECORD LENGTH
23F5 B7                 ORA     A       ;SET Z-FLAG ON RECORD LENGTH
23F6 3E47               MVI     A,'G'   ;ANSWER TO QUESTION = GOOD
23F8 CA0224             JZ      DONE    ;IF LENGTH = 0 THEN DONE
23FB 7A                 MOV     A,D     ;(D) = ERRER FLAG ON GETREC RETURN
23FC B7                 ORA     A       ;SEE IF THE "ERRER" FLAG IS NON-ZERO.
23FD CAF223             JZ      LOAD1   ;IF NOT, GO DO NEXT RECORD
2400 3E42               MVI     A,'B'   ;STORE "BAD" FLAG IN ANSWER TO QUESTION
2402 322637     DONE:   STA     MISCBF  ;STORE GOOD/BAD STRING
2405 3EFF               MVI     A,EOL   ;  .
2407 322737             STA     MISCBF+1;  .
240A 3A2837             LDA     MISCBF+2;RESTORE ECHO FLAG
240D 320237             STA     ECHOFL  ;  .
2410 AF                 XRA     A       ;SET BACK TO "LOAD" MODE
2411 320637             STA     VFYFLG  ;  .
2414 3E13               MVI     A,XOFF  ;STOP FURTHER OUTPUT
2416 CDD32B             CALL    CO      ;
2419 C9                 RET             ;RETURN TO COMMAND LEVEL
                ;
                ;       END     LOADER
                ;
                ;
                ; *** GETREC *** READ IN ONE RECORD
                ;
241A CD3B24     GETREC: CALL    FNDMRK  ;SKIP TO RECORD MARK
                ;
241D CD6C24             CALL    LGHB    ;GET THE RECORD LENGTH
2420 4F                 MOV     C,A     ;  INTO THE C REG.
2421 CD6C24             CALL    LGHB    ;GET LOAD ADDRESS FIELD INTO H & L
2424 67                 MOV     H,A     ;
2425 CD6C24             CALL    LGHB    ;
```

```
2428 6F                 MOV     L,A     ;
2429 D5                 PUSH    D       ;SAVE D&E
242A EB                 XCHG            ;
242B 2AØ437             LHLD    BIAS    ;ADD BIAS
242E 19                 DAD     D       ;
242F D1                 POP     D       ;RESTORE D&E
243Ø CD6C24             CALL    LGHB    ;GET THE RECORD-TYPE BYTE AND IGNORE
2433 CD4824             CALL    DATA    ;PUT THE NEXT (C) BYTES INTO MEMORY
                                        ;STARTING WHERE HL POINTS
2436 CD6C24             CALL    LGHB    ;READ THE CHECKSUM BYTE
2439 79                 MOV     A,C     ;PUT THE RECORD LENGTH BACK INTO A REG.
243A C9                 RET             ;RETURN FROM GETREC. (D) CONTAINS THE
                                        ;  SUM OFF ALL HEX BYTES READ, AND SO
                                        ;  IS EFFECTIVELY AN ERRER FLAG
                ;       END     GETREC
                ;
                ;
                ; *** FNDMRK *** - FIND RECORD MARK
                ;                   IGNORES ALL TEXT UNTIL ":" FOUND, THEN RET
                ;
243B CDB32B     FNDMRK: CALL    CI      ;GET CHARACTER
243E E67F               ANI     Ø7FH    ;STRIP OFF 8TH BIT
244Ø FE3A               CPI     ':'     ;
2442 C23B24             JNZ     FNDMRK  ;NOT RECORD MARK - GET NEXT CHAR
2445 16ØØ               MVI     D,Ø     ;CLEAR D REGISTER (ERRER ACCUMULATOR)
2447 C9                 RET             ;
                ;
                ;       END     FNDMRK
                ;
                ; *** DATA *** - INPUT ALL DATA BYTES
                ;                 (C) = NUMBER OF BYTES TO READ IN
                ;                 (D) = ERRER FLAG ACCUMULATOR MAINTAINED BY LGHB
                ;
2448 41         DATA:   MOV     B,C     ;COPY C REG. TO B
2449 78         LOOP:   MOV     A,B     ;GET REMAINING BYTE COUNT
244A B7                 ORA     A       ;GET FLAGS
244B C8                 RZ              ;RETURN FROM SUBR. IF NONE LEFT
244C Ø5                 DCR     B       ;ELSE DECREMENT B REG.
244D 3AØ637             LDA     VFYFLG  ;NON-ZERO MEANS VERIFY ONLY
245Ø B7                 ORA     A       ;
2451 C25B24             JNZ     LVFY    ;
2454 CD6C24             CALL    LGHB    ;GET BYTE FROM DATA FIELD
2457 77                 MOV     M,A     ;STORE IN MEMORY
2458 C36824             JMP     DATA1   ;
245B CD6C24     LVFY:   CALL    LGHB    ;GET BYTE FROM DATA FIELD
245E 96                 SUB     M       ;COMPARE TO MEMORY
245F CA6824             JZ      DATA1   ;GOOD...
2462 322937             STA     MISCBF+3;FLAG WHERE WE ARE COMING FROM
                                        ;IS NONZERO OR WOULDN'T BE HERE
2465 CD1B2E             CALL    RETJMP  ;
2468 23         DATA1:  INX     H       ;BUMP POINTER
2469 C34924             JMP     LOOP    ;GO BACK FOR NEXT CHAR.
                ;
                ;       END     DATA
                ;
                ;
```

```
               ; *** LGHB *** - LOADER GET HEX BYTE
               ;               SAME AS GHB EXCEPT ADDS BYTE GOTTEN TO ERRER
               ;               ACCUMULATOR IN D REGISTER
               ;
246C CD5C2D    LGHB:  CALL    GHB     ;GET BYTE
246F F5               PUSH    PSW     ;SAVE BYTE
2470 82               ADD     D       ;ADD TO (D)
2471 57               MOV     D,A     ;PUT SUM IN D-REG
2472 F1               POP     PSW     ;RESTORE BYTE
2473 C9               RET             ;
               ;
               ;      END     LGHB
               ;
               ;***********************END OF LOADER****************************
               ;
               ;***********************BEGINNING OF DUMP*************************
               ;
               ; DUMP1 IS AN ENTRY POINT FOR EXTERNAL USE OF ROUTINE
               ;
2474 CD8F2C    DUMP:  CALL    FROMTO  ;GET BEGINNING ADDRESS AND BYTE COUNT
2477 DA3822           JC      ERRER   ;NON HEX CHAR TYPED - WHAT ?? ? ?? ?
247A CDC42D           CALL    OKCK    ;CHECK WITH USER BEFORE CONTINUING
247D D8               RC              ;
247E 3A0337    DUMP1: LDA     WIDTH   ;GET WIDTH
2481 47               MOV     B,A     ;
2482 2F               CMA             ;ROUND DOWN STARTING ADDRESS
2483 A5               ANA     L       ;
2484 6F               MOV     L,A     ;
2485 7B               MOV     A,E     ;ROUND UP ENDING ADDRESS
2486 B0               ORA     B       ;
2487 5F               MOV     E,A     ;
2488 E5               PUSH    H       ;D&E=START-ENDING-1
2489 CD742E           CALL    SUB16   ;
248C 2B               DCX     H       ;
248D D1               POP     D       ;
248E EB               XCHG            ;
248F CD2D2C           CALL    CRLF    ;GO TO NEW LINE
2492 CDE42D           CALL    PHW     ;PRINT MEMORY ADDRESS
2495 E5               PUSH    H       ;PUT RAM ADDRESS ON STACK
2496 212637           LXI     H,MISCBF;GET BUFFER ADDRESS
2499 E3               XTHL            ;PUT BUFFER ADDRESS ON STACK
               ;                      ;GET RAM ADDRESS OFF
               ;
               ; AT THIS POINT TOP OF STACK HAS BUFFER ADDRESS
               ;              H&L HAS RAM ADDRESS
               ;
249A 7E        DI1:   MOV     A,M     ;GET BYTE
249B 23               INX     H       ;POINT TO NEXT BYTE IN RAM
249C CD5A2E           CALL    SPACE   ;
249F CDEF2D           CALL    PHB     ;PRINT BYTE IN HEX
24A2 E67F             ANI     07FH    ;STRIP PARITY
24A4 FE20             CPI     020H    ;CHECK FOR PRINTABLE
24A6 DAAE24           JC      DI3     ;NOT PRINTABLE - PRINT '.'
24A9 FE7F      DI2:   CPI     07FH    ;MAY BE PRINTABLE - CHECK FOR RUBOUT
24AB C2B024           JNZ     DI4     ;NOPE..OK
24AE 3E2E      DI3:   MVI     A,'.'   ;NOT PRINTABLE - REPLACE WITH SPACE
```

```
2480 E3         DI4:   XTHL             ;GET BUFFER ADDRESS
2481 77                MOV     M,A      ;PUT CHAR OR SPACE IN BUFFER
2482 23                INX     H        ;
2483 E3                XTHL             ;PUT BUFFER ADDRESS BACK
2484 13                INX     D        ;DECREMENT COUNT OF NUMBER OF BYTES LEFT
2485 7D                MOV     A,L      ;
2486 A0                ANA     B        ;END OF LINE - PRINT ASCII AND CRLF
2487 C29A24            JNZ     DI1      ;KEEP GOING IF NOT AT END OF LINE
24BA E3         DMPLIN: XTHL            ;GET BUFFER ADDRESS
24BB 36FF              MVI     M,EOL    ;TERMINATE STRING
24BD 212637            LXI     H,MISCBF;POINT BACK TO START OF BUFFER
24C0 E3                XTHL             ;PUT BUFFER ADDRESS BACK ON STACK
24C1 CD5A2E            CALL    SPACE    ;SPACE OVER A COUPLE
24C4 CD5A2E            CALL    SPACE    ;
24C7 D5                PUSH    D        ;
24C8 112637            LXI     D,MISCBF;POINT TO BEGINNING OF ASCII BUFFER
24CB CDAB2D            CALL    MSG      ;PRINT ASCII BUFFER
24CE D1                POP     D        ;
24CF 7B                MOV     A,E      ;
24D0 B2                ORA     D        ;
24D1 CADD24            JZ      DMPEND   ;DONE
24D4 CD2D2C            CALL    CRLF     ;
24D7 CDE42D            CALL    PHW      ;PRINT MEMORY ADDRESS
24DA C39A24            JMP     DI1      ;
24DD E1         DMPEND: POP    H        ;CLEAN OFF STACK
24DE 3EFF              MVI     A,EOL    ;CLEAR ANSWER...
24E0 322637            STA     MISCBF   ;
24E3 C9                RET              ;
                ;
                ;*************************END OF DUMP*********************************
                ;
                ;*************************BEGINNING OF IOPORT*************************
                ;
                ; IO - I/O PORT MANIPULATION
                ;
24E4 CD5C2D     IOPORT: CALL   GHB              ;GET PORT NUMBER
24E7 DA3822            JC      ERRER            ;
24EA 322837            STA     MISCBF+2         ;DON'T TROMP ON EOL
24ED 3EC9              MVI     A,0C9H           ;STORE RETURN
24EF 322937            STA     MISCBF+3         ;
24F2 CD5A2E            CALL    SPACE            ;
24F5 CDB32B            CALL    CI               ;GET IOPORT COMMAND
24F8 CD802E            CALL    UCASE            ;STRIP PARITY
24FB CD5A2E            CALL    SPACE            ;
24FE FE52              CPI     'R'              ;IF NOT R, CHECK OTHERS
2500 C20725            JNZ     IOP1             ;
2503 CD1925            CALL    IOPR             ;IOPORT READ ROUTINE
2506 C9                RET                      ;
2507 FE57       IOP1:  CPI     'W'              ;IF NOT W, CHECK M
2509 C21025            JNZ     IOP2             ;
250C CD3925            CALL    IOPW             ;IOPORT WRITE ROUTINE
250F C9                RET                      ;
2510 FE4D       IOP2:  CPI     'M'              ;IF NOT M, THEN WHAT DO
2512 C23822            JNZ     ERRER            ;  YOU WANT ?
2515 CD5425            CALL    IOPM             ;IOPORT MONITOR ROUTINE
2518 C9                RET                      ;
```

```
                ;       END     IOPORT          ;MAIN PROGRAM
                ;
                ; IOPR - IOPORT READ SUBCOMMAND
                ;
2519 3EDB       IOPR:   MVI     A,ØDBH          ;STORE "IN" INST
251B 322737             STA     MISCBF+1        ;
251E CD2737             CALL    MISCBF+1        ;GET BYTE FROM PORT
2521 116F2F             LXI     D,IOPDA         ;PRINT 'DATA= '
2524 CDAB2D             CALL    MSG             ;
2527 CDEF2D             CALL    PHB             ;PRINT BYTE IN HEX
252A CD5A2E             CALL    SPACE           ;
252D CD662C             CALL    DISASC          ;PRINT BYTE IN ASCII
253Ø 7C                 MOV     A,H             ;
2531 CDD32B             CALL    CO              ;
2534 7D                 MOV     A,L             ;
2535 CDD32B             CALL    CO              ;
2538 C9                 RET                     ;
                ;
                ; IOPW - IOPORT WRITE COMMAND
                ;
2539 116F2F     IOPW:   LXI     D,IOPDA         ;PRINT 'DATA= '
253C CDAB2D             CALL    MSG             ;
253F CD5C2D             CALL    GHB             ;
2542 DA3822             JC      ERRER           ;BAD CHAR TYPED...
2545 CDC42D             CALL    OKCK            ;CHECK TO BE SURE
2548 D8                 RC                      ;MUST HAVE GOOFED...
2549 F5                 PUSH    PSW             ;SAVE DATA
254A 3ED3               MVI     A,ØD3H          ;STORE "OUT" INST
254C 322737             STA     MISCBF+1        ;
254F F1                 POP     PSW             ;GET DATA BACK
255Ø CD2737             CALL    MISCBF+1        ;WRITE DATA
2553 C9                 RET                     ;
                ;
                ; IOPM - IOPORT MONITOR COMMAND
                ;
2554 11762F     IOPM:   LXI     D,IOPMM         ;PRINT '@ 5ØMS # '
2557 CDAB2D             CALL    MSG             ;
255A CD5C2D             CALL    GHB             ;
255D DA3822             JC      ERRER           ;BAD CHAR...
256Ø CDC42D             CALL    OKCK            ;GIVE ESCAPE A CHANCE...
2563 D8                 RC                      ;
2564 4F                 MOV     C,A             ;WOULD YOU BELEIVE C FOR COUNTER?
2565 CDED2C             CALL    GCLKFQ          ;CHECK TO SEE IS WE CAN TIME IT...
2568 CD2D2C             CALL    CRLF            ;
256B 3EDB               MVI     A,ØDBH          ;STORE "IN" INST
256D 322737             STA     MISCBF+1        ;
257Ø 16ØØ               MVI     D,Ø             ;
2572 CD2737     IOPM1:  CALL    MISCBF+1        ;GET BYTE FROM PORT
2575 CDEF2D             CALL    PHB             ;PRINT BYTE IN HEX
2578 CD5A2E             CALL    SPACE           ;
257B CD662C             CALL    DISASC          ;PRINT BYTE IN ASCII
257E 7C                 MOV     A,H             ;
257F CDD32B             CALL    CO              ;
2582 7D                 MOV     A,L             ;
2583 CDD32B             CALL    CO              ;
2586 11802F             LXI     D,IOPSM         ;PRINT ',  '
```

```
2589 CDAB2D          CALL    MSG             ;
258C 41              MOV     B,C             ;WAIT (C)*5ØMS
258D Ø4              INR     B               ;CHECK FOR ZERO
258E Ø5      IOPM2:  DCR     B               ;
258F CA9825          JZ      IOPM3           ;
2592 CD362C          CALL    D5ØMS           ;
2595 C38E25          JMP     IOPM2           ;
2598 14      IOPM3:  INR     D               ;CHECK TO SEE IF IT IS TIME
2599 3AØ337          LDA     WIDTH           ;  FOR A ROUSING ROUND OF CRLF
259C B7              ORA     A               ;CLEAR CARRY
259D 1F              RAR                     ;CUT DOWN ONE
259E A2              ANA     D               ;
259F CC2D2C          CZ      CRLF            ;
25A2 C37225          JMP     IOPM1           ;
                     ;
                     ;*************************END OF IO PORT COMMAND*************
                     ;
                     ;***********************************************************
                     ;*          BEGINNING OF ULTRASONIC ROUTINE                *
                     ;***********************************************************

25A5 3EØØ    USFNT:  MVI     A,ØØH
25A7 D34Ø            OUT     PIAD            ;RESET INIT LINE ON SONICS
25A9 D322            OUT     TIME2           ;ZERO MSB OF COUNT
25AB D322            OUT     TIME2           ;     LSB OF COUNT
25AD 3EØ1            MVI     A,Ø1H
25AF D34Ø            OUT     PIAD            ;SEND OUT SONIC BOOM
25B1 115ØØØ          LXI     D,ØØ5ØH         ;DELAY FOR < 1 MILLISEC.
25B4 CD8C2A          CALL    DELAYD          ; OFF TO DELAY
25B7 3EØ3            MVI     A,Ø3H           ;SEND OUT BLANK INHIBIT
25B9 D34Ø            OUT     PIAD            ; BUT KEEP BOOM HIGH
25BB 3A1537  LOOPD:  LDA     MAXFNT          ;GET MAX FRONT DIST.
25BE 47              MOV     B,A
25BF CDD626          CALL    CNTCK           ;FIND OUT HOW LONG
25C2 7C              MOV     A,H
25C3 B8              CMP     B               ; BOOM HAS BEEN GONE
25C4 DAD225          JC      NEXTA           ; IF SO FORGET IT
25C7 3EØØ            MVI     A,ØØH
25C9 D34Ø            OUT     PIAD            ;RESET EVERYTHING
25CB 21ØØØØ          LXI     H,ØØØØH         ; CLEAR DIST.
25CE 221337          SHLD    FNTDST
25D1 C9              RET

25D2 DB42    NEXTA:  IN      PIAF            ;TEST FOR BOOM
25D4 E6Ø1            ANI     Ø1H             ;MASK OFF DIRECTION
25D6 FEØ1            CPI     Ø1H             ;TEST FOR DIRECTION
25D8 C2BB25          JNZ     LOOPD           ;IF NOT BOOM THEN WAIT
25DB 3EØØ            MVI     A,ØØH
25DD D34Ø            OUT     PIAD            ;RESET INIT LINE
25DF CDD626          CALL    CNTCK           ;GET COUNTER IN HL
25E2 CDE526          CALL    BEEP
25E5 CDE526          CALL    BEEP
25E8 CD1627          CALL    FNDDT
25EB 221337          SHLD    FNTDST
25EE C9              RET
```

```
25EF 3EB0     USBACK:       MVI      A,0B0H          ;INITIALIZE 8253 COUNTER
25F1 D323           OUT     TIMCTL          ;TIMER2 BINARY COUNT MODE 0
25F3 3E00           MVI     A,00H
25F5 D340           OUT     PIAD            ;RESET INIT LINE ON SONICS
25F7 D322           OUT     TIME2           ;ZERO MSB OF COUNT
25F9 D322           OUT     TIME2           ;     LSB OF COUNT
25FB 3E04           MVI     A,04H
25FD D340           OUT     PIAD            ;SEND OUT SONIC BOOM
25FF 115000         LXI     D,0050H         ;DELAY FOR < 1 MILLISEC.
2602 CD8C2A         CALL    DELAYD          ; OFF TO DELAY
2605 3E0C           MVI     A,0CH           ;SEND OUT BLANK INHIBIT
2607 D340           OUT     PIAD            ; BUT KEEP BOOM HIGH
2609 3A1837   LOOPF: LDA    MAXBAK          ;GET MAX BACK DIST.
260C 47             MOV     B,A
260D CDD626         CALL    CNTCK           ;FIND OUT HOW LONG
2610 7C             MOV     A,H
2611 B8             CMP     B               ; BOOM HAS BEEN GONE
2612 DA2026         JC      NEXTB           ; IF SO FORGET IT
2615 3E00           MVI     A,00H
2617 D340           OUT     PIAD            ;RESET EVERYTHING
2619 210000         LXI     H,0000H
261C 221637         SHLD    BAKDST
261F C9             RET

2620 DB42     NEXTB: IN     PIAF            ;TEST FOR BOOM
2622 E602           ANI     02H             ;MASK OFF DIRECTION
2624 FE02           CPI     02H             ;TEST FOR DIRECTION
2626 C20926         JNZ     LOOPF           ;IF NOT BOOM THEN WAIT
2629 3E00           MVI     A,00H
262B D340           OUT     PIAD            ;RESET INIT LINE
262D CDD626         CALL    CNTCK           ;GET COUNTER IN HL
2630 CDE526         CALL    BEEP
2633 CDE526         CALL    BEEP
2636 CD1627         CALL    FNDDT
2639 221637         SHLD    BAKDST
263C C9             RET

263D 3EB0     USRT:  MVI    A,0B0H          ;INITIALIZE 8253 COUNTER
263F D323           OUT     TIMCTL          ;TIMER2 BINARY COUNT MODE 0
2641 3E00           MVI     A,00H
2643 D340           OUT     PIAD            ;RESET INIT LINE ON SONICS
2645 D322           OUT     TIME2           ;ZERO MSB OF COUNT
2647 D322           OUT     TIME2           ;     LSB OF COUNT
2649 3E10           MVI     A,10H
264B D340           OUT     PIAD            ;SEND OUT SONIC BOOM
264D 115000         LXI     D,0050H         ;DELAY FOR < 1 MILLISEC.
2650 CD8C2A         CALL    DELAYD          ; OFF TO DELAY
2653 3E30           MVI     A,30H           ;SEND OUT BLANK INHIBIT
2655 D340           OUT     PIAD            ; BUT KEEP BOOM HIGH
2657 3A1B37   LOOPH: LDA    MAXRT           ;GET MAX RIGHT DIST.
265A 47             MOV     B,A
265B CDD626         CALL    CNTCK           ;FIND OUT HOW LONG
265E 7C             MOV     A,H
265F B8             CMP     B               ; BOOM HAS BEEN GONE
2660 DA6E26         JC      NEXTC           ; IF SO FORGET IT
2663 3E00           MVI     A,00H
```

```
2665 D340             OUT    PIAD          ;RESET EVERYTHING
2667 210000           LXI    H,0000H
266A 221937           SHLD   RTDST
266D C9               RET

266E DB42     NEXTC:  IN     PIAF          ;TEST FOR BOOM
2670 E604             ANI    04H           ;MASK OFF DIRECTION
2672 FE04             CPI    04H           ;TEST FOR DIRECTION
2674 C25726           JNZ    LOOPH         ;IF NOT BOOM THEN WAIT
2677 3E00             MVI    A,00H
2679 D340             OUT    PIAD          ;RESET INIT LINE
267B CDD626           CALL   CNTCK         ;GET COUNTER IN HL
267E CDE526           CALL   BEEP
2681 CD1627           CALL   FNDDT
2684 221937           SHLD   RTDST
2687 C9               RET

2688 3EB0     USLFT:  MVI    A,0B0H        ;INITIALIZE 8253 COUNTER
268A D323             OUT    TIMCTL        ;TIMER2 BINARY COUNT MODE 0
268C 3E00             MVI    A,00H
268E D340             OUT    PIAD          ;RESET INIT LINE ON SONICS
2690 D322             OUT    TIME2         ;ZERO MSB OF COUNT
2692 D322             OUT    TIME2         ;     LSB OF COUNT
2694 3E40             MVI    A,40H
2696 D340             OUT    PIAD          ;SEND OUT SONIC BOOM
2698 115000           LXI    D,0050H       ;DELAY FOR < 1 MILLISEC.
269B CD8C2A           CALL   DELAYD        ; OFF TO DELAY
269E 3EC0             MVI    A,0C0H        ;SEND OUT BLANK INHIBIT
26A0 D340             OUT    PIAD          ; BUT KEEP BOOM HIGH
26A2 3A1E37   LOOPJ:  LDA    MAXLFT        ;GET MAX LEFT DIST.
26A5 47               MOV    B,A
26A6 CDD626           CALL   CNTCK         ;FIND OUT HOW LONG
26A9 7C               MOV    A,H
26AA B8               CMP    B             ; IF GEATER
26AB DAB926           JC     NEXTD         ; IF SO FORGET IT
26AE 3E00             MVI    A,00H
26B0 D340             OUT    PIAD          ;RESET EVERYTHING
26B2 210000           LXI    H,0000H
26B5 221C37           SHLD   LFTDST
26B8 C9               RET

26B9 DB42     NEXTD:  IN     PIAF          ;TEST FOR BOOM
26BB E608             ANI    08H           ;MASK OFF DIRECTION
26BD FE08             CPI    08H           ;TEST FOR DIRECTION
26BF C2A226           JNZ    LOOPJ         ;IF NOT BOOM THEN WAIT
26C2 3E00             MVI    A,00H
26C4 D340             OUT    PIAD          ;RESET INIT LINE
26C6 CDD626           CALL   CNTCK         ;GET COUNTER IN HL
26C9 CDE526           CALL   BEEP
26CC CDE526           CALL   BEEP
26CF CD1627           CALL   FNDDT
26D2 221C37           SHLD   LFTDST
26D5 C9               RET

26D6 F5       CNTCK:  PUSH   PSW
26D7 3E80             MVI    A,80H
```

```
26D9 D323              OUT     TIMCTL             ;LATCH CURRENT COUNT
26DB DB22              IN      TIME2              ;GET LSB
26DD 2F                CMA                        ; FLIP IT TO REAL TIME
26DE 6F                MOV     L,A
26DF DB22              IN      TIME2              ;GET MSB
26E1 2F                CMA                        ; FLIP TO REAL TIME
26E2 67                MOV     H,A
26E3 F1                POP     PSW
26E4 C9                RET

26E5 3A2137    BEEP:   LDA     SONOFF
26E8 FEFF              CPI     TRUE
26EA CØ                RNZ
26EB 3E4Ø              MVI     A,4ØH
26ED D342              OUT     PIAF               ;TURN ON TONE
26EF 54                MOV     D,H
26FØ 5D                MOV     E,L
26F1 CD8C2A            CALL    DELAYD             ;WAIT FOR IT
26F4 3ECØ              MVI     A,ØCØH             ;CHANGE TONE
26F6 D342              OUT     PIAF
26F8 54                MOV     D,H
26F9 5D                MOV     E,L
26FA CD8C2A            CALL    DELAYD
26FD 3EØØ              MVI     A,ØØH              ;NOW TURN EVERYTHING OFF
26FF D342              OUT     PIAF
27Ø1 C9                RET

27Ø2 3ECØ      HORN1:  MVI     A,ØCØH             ;TURN ON TONE 1
27Ø4 C3Ø927            JMP     HORNA
27Ø7 3E4Ø      HORN:   MVI     A,4ØH              ;TURN ON TONE 2
27Ø9 D342      HORNA:  OUT     PIAF
27ØB 11ØØ5Ø            LXI     D,5ØØØH
27ØE CD8C2A            CALL    DELAYD
2711 3EØØ              MVI     A,ØØH              ;TURN IT OFF
2713 D342              OUT     PIAF
2715 C9                RET


2716 11FØØØ    FNDDT:  LXI     D,ØØFØH            ;COUNT TO DIST. RATIO
2719 Ø1FFFF            LXI     B,ØFFFFH           ; ZERO BC
271C Ø3        LOOPM:  INX     B
271D CD742E            CALL    SUB16              ;HL=HL-DE
272Ø D21C27            JNC     LOOPM              ;DONE YET?/
2723 69                MOV     L,C
2724 6Ø                MOV     H,B
2725 C9                RET

2726 21ØØØ1    SETDEF:         LXI     H,Ø1ØØH
2729 221F37            SHLD    TIMDLY
272C 3E2Ø              MVI     A,2ØH
272E 321537            STA     MAXFNT
2731 3E2Ø              MVI     A,2ØH
2733 321837            STA     MAXBAK
2736 3E2Ø              MVI     A,2ØH
2738 321B37            STA     MAXRT
273B 3E2Ø              MVI     A,2ØH
```

```
273D 321E37             STA     MAXLFT
2740 C9                 RET

2741 11EE33     SETTIM:         LXI     D,TIMQUE
2744 CDAB2D             CALL    MSG
2747 CD932A             CALL    INPAD
274A 61                 MOV     H,C
274B E5                 PUSH    H
274C CD932A             CALL    INPAD
274F E1                 POP     H
2750 69                 MOV     L,C
2751 221F37             SHLD    TIMDLY
2754 CD2D2C             CALL    CRLF
2757 C9                 RET

2758 110534     SETFNT:         LXI     D,FNTQUE
275B CDAB2D             CALL    MSG
275E CD932A             CALL    INPAD
2761 321537             STA     MAXFNT
2764 CD2D2C             CALL    CRLF
2767 C9                 RET

2768 111834     SETBAK:         LXI     D,BAKQUE
276B CDAB2D             CALL    MSG
276E CD932A             CALL    INPAD
2771 321837             STA     MAXBAK
2774 CD2D2C             CALL    CRLF
2777 C9                 RET

2778 112A34     SETRT:  LXI     D,RTQUE.
277B CDAB2D             CALL    MSG
277E CD932A             CALL    INPAD
2781 321B37             STA     MAXRT
2784 CD2D2C             CALL    CRLF
2787 C9                 RET

2788 113D34     SETLFT:         LXI     D,LFTQUE
278B CDAB2D             CALL    MSG
278E CD932A             CALL    INPAD
2791 321E37             STA     MAXLFT
2794 CD2D2C             CALL    CRLF
2797 C9                 RET

2798 CDA525     ULTRA:  CALL    USFNT
279B 2A1F37             LHLD    TIMDLY
279E EB                 XCHG
279F CD8C2A             CALL    DELAYD
27A2 2A1337             LHLD    FNTDST
27A5 7C                 MOV     A,H
27A6 B5                 ORA     L
27A7 FE00               CPI     00H
27A9 CAB827             JZ      ULTRA1
27AC 11B533             LXI     D,FNTMSG
27AF CDAB2D             CALL    MSG
27B2 CDE42D             CALL    PHW
27B5 C3B827             JMP     ULTRA1
```

```
27B8 CDEF25     ULTRA1:         CALL    USBACK
27BB 2A1F37             LHLD    TIMDLY
27BE EB                 XCHG
27BF CD8C2A             CALL    DELAYD
27C2 2A1637             LHLD    BAKDST
27C5 7C                 MOV     A,H
27C6 B5                 ORA     L
27C7 FE00               CPI     00H
27C9 CAD527             JZ      ULTRA2
27CC 11BF33             LXI     D,BAKMSG
27CF CDAB2D             CALL    MSG
27D2 CDE42D             CALL    PHW
27D5 CD3D26     ULTRA2:         CALL    USRT
27D8 2A1F37             LHLD    TIMDLY
27DB EB                 XCHG
27DC CD8C2A             CALL    DELAYD
27DF 2A1937             LHLD    RTDST
27E2 7C                 MOV     A,H
27E3 B5                 ORA     L
27E4 FE00               CPI     00H
27E6 CAF227             JZ      ULTRA3
27E9 11CA33             LXI     D,RTMSG
27EC CDAB2D             CALL    MSG
27EF CDE42D             CALL    PHW
27F2 CD8826     ULTRA3:         CALL    USLFT
27F5 2A1F37             LHLD    TIMDLY
27F8 EB                 XCHG
27F9 CD8C2A             CALL    DELAYD
27FC 2A1C37             LHLD    LFTDST
27FF 7C                 MOV     A,H
2800 B5                 ORA     L
2801 FE00               CPI     00H
2803 CA0F28             JZ      ULTRA4
2806 11D833             LXI     D,LFTMSG
2809 CDAB2D             CALL    MSG
280C CDE42D             CALL    PHW
280F CD2D2C     ULTRA4:         CALL    CRLF
2812 C9                 RET


                ;****************** END OF SONICS ROUTINE ****************
                ;
                ;*****************  BEGINNING OF CHAIR PROGRAMS ***********

2813 3E82       RUNCHR:         MVI     A,10000010B     ;PORT A: OUTPUT
2815 D313               OUT     PIACNTL         ;PORT B: INPUT
                                                ;PORT C (UPPER): OUTPUT
                                                ;PORT C (LOWER): OUTPUT
2817 3EB0               MVI     A,0B0H          ;INITIALIZE 8253 COUNTER
2819 D323               OUT     TIMCTL          ;TIMER2 BINARY COUNT MODE 0
281B 3E81               MVI     A,81H           ;8255 PIA D=IN E=IN
281D D343               OUT     PIBCNTL         ;F=OUT

281F CD2627             CALL    SETDEF          ;SET UP ULTRASONIC PARAM.
2822 3E00               MVI     A,FALSE
2824 322537             STA     FLGERR
```

```
2827 3EFF             MVI    A,TRUE
2829 322237           STA    RONOFF
282C 322137           STA    SONOFF
282F 3A2237   CHAIR1: LDA    RONOFF
2832 FEFF             CPI    TRUE
2834 C23A28           JNZ    CHAIR2
2837 CD9827           CALL   ULTRA              ; CALL U.S.RANGING
283A CD5E29   CHAIR2:        CALL     PADRD
283D 7C               MOV    A,H
283E E64Ø             ANI    MENERR             ;CHECK FOR MENU ERROR
284Ø FE4Ø             CPI    MENERR
2842 C24B28           JNZ    MENOK              ;MENU IS OK ?
2845 CD4229           CALL   MENUER             ;FLASH LED/HORN
2848 C32F28           JMP    CHAIR1             ;IF NOT START OVER

284B 7C       MENOK: MOV     A,H
284C FEØ1             CPI    PROMSK
284E C22F28           JNZ    CHAIR1             ;IF NOT COUNTINUE SCAN
2851 CD5728           CALL   PROMEN             ;CALL PROGRAMMING MENU PROG
2854 C32F28           JMP    CHAIR1             ;REPEAT WHOLE PROCESS

2857 CD5E29   PROMEN:        CALL     PADRD             ;GET A Ø-F PAD INPUT
285A 7C               MOV    A,H
285B E64Ø             ANI    MENERR             ;
285D FE4Ø             CPI    MENERR             ;
285F C26628           JNZ    MENOK1             ;MENU OK
2862 CD4229            CALL  MENUER             ;BEEP HORN OR LIGHT LED
2865 C9               RET                       ;REPEAT CHAIR1 PROCESS

2866 7C       MENOK1:        MOV      A,H               ;CHECK FOR CORRECT PROGRAM
2867 E68Ø             ANI    TOUCH              ;
2869 FE8Ø             CPI    TOUCH              ;
286B C25728           JNZ    PROMEN             ;IF NOT VALID TOUCH & MENU
286E 7D               MOV    A,L                ;MOVE TOUCH LOC. TO A REG

286F FE16     SNDCHK:        CPI      SOUND             ;CHECK FOR SOUND ON/OFF
2871 C28328           JNZ    RNGCHK             ;
2874 CDØ727           CALL   HORN               ;
2877 212137           LXI    H,SONOFF           ;POINT TO SOUND FLAG
287A CD1F29           CALL   ONOFF              ;SELECT ON/OFF
287D CDØ227           CALL   HORN1              ;
288Ø C35728           JMP    PROMEN

2883 FE36     RNGCHK:        CPI      RANGE             ;CHECK FOR RANGING ON/OFF
2885 C29728           JNZ    SPDCHK             ;
2888 CDØ727           CALL   HORN               ;BEEP
288B 212237           LXI    H,RONOFF           ;POINT TO RANGING FLAG
288E CD1F29           CALL   ONOFF              ;WAIT FOR ON OR OFF TOUCH
2891 CDØ227           CALL   HORN1              ;LOW BEEOP
2894 C35728           JMP    PROMEN             ;

2897 FE56     SPDCHK:        CPI      SPEED             ;CHECK FOR HIGH SPEED ON/OFF
2899 C2AB28           JNZ    RMPCHK             ;
289C CDØ727           CALL   HORN               ;BEEP
289F 212337           LXI    H,HONOFF           ;POINT HL TO HIGH SPEED FLAG
28A2 CD1F29           CALL   ONOFF              ;CHECK FOR ON OR OFF TOUCH
```

```
28A5 CDØ227          CALL    HORN1           ;LOW BEEP
28A8 C35728          JMP     PROMEN          ;RETURN FOR NEW INP

28AB FE96     RMPCHK:        CPI     RRATE          ;CHECK FOR RAMP RATE
28AD C2CØ28          JNZ     DELCHK          ;
28BØ CDØ727          CALL    HORN
28B3 CD932A          CALL    INPAD           ;GET SINGLE BYTE FROM HEX PAD
28B6 212437          LXI     H,RAMP          ;POINT TO RAMP RATE VARIABLE
28B9 71              MOV     M,C             ;X-FER INPUT TO RAMP VARIABLE
28BA CDØ227          CALL    HORN1           ;LOW BEEP
28BD C35728          JMP     PROMEN

28CØ FEB6     DELCHK:        CPI     SDELAY         ;CHECK F FOR RAMP RATE CHOICE
28C2 C2DB28          JNZ     LCHK            ;
28C5 CDØ727          CALL    HORN            ;BEEP
28C8 CD932A          CALL    INPAD           ;GET ONE BYTE HEX INPUT
28CB 61              MOV     H,C             ;
28CC E5              PUSH    H
28CD CD932A          CALL    INPAD           ;GET NEXT HEX BYTE
28DØ E1              POP     H
28D1 69              MOV     L,C             ;
28D2 221F37          SHLD    TIMDLY          ;STORE DELAY TIME
28D5 CDØ227          CALL    HORN1           ;LOW BEEP
28D8 C35728          JMP     PROMEN          ;RETURN FOR NEW INPUT

28DB FEA9     LCHK:  CPI     LEFT            ;CHECK FOR LEFT DIST. INPUT
28DD C2EC28          JNZ     RCHK            ;
28EØ CDØ727          CALL    HORN            ;BEEP
28E3 CD8827          CALL    SETLFT          ;GET LEFT RANGE DIST.
28E6 CDØ227          CALL    HORN1           ;LOW BEEP
28E9 C35728          JMP     PROMEN          ;RETURN FOR NEW INPUT

28EC FEAD     RCHK:  CPI     RIGHT           ;CHECK FOR RIGHT DIST. INPUT
28EE C2FD28          JNZ     FCHK            ;
28F1 CDØ727          CALL    HORN
28F4 CD7827          CALL    SETRT           ;GET RIGHT DIST.
28F7 CDØ227          CALL    HORN1           ;LOW BEEP
28FA C35728          JMP     PROMEN          ;RETURN FOR NEW INPUT

28FD FEC9     FCHK:  CPI     FRONT           ;CHELEFT RANGE DIST.
28FF C2ØE29          JNZ     BCHK
29Ø2 CDØ727          CALL    HORN            ;BEEP
29Ø5 CD5827          CALL    SETFNT          ;GET FRONT RANGING DIST.
29Ø8 CDØ227          CALL    HORN1           ;LOW BEEP
29ØB C35728          JMP     PROMEN          ;

29ØE FECD     BCHK:  CPI     BACK            ;CHECK FOR BACK DIST.
291Ø C25728          JNZ     PROMEN
                                             ; IF NONE OF THE ABOVE
                                             ; BACK TO CHAIR1
2913 CDØ727          CALL    HORN            ;BEEP
2916 CD6827          CALL    SETBAK          ;SET BACK FOR FRONT DIST. INPUT
2919 CDØ227          CALL    HORN1           ;LOW BWEEP
291C C35728          JMP     PROMEN          ;RETURN FOR NEW INPUT

291F E5       ONOFF: PUSH    H
```

```
2920 CD5E29   ONOFF2:       CALL    PADRD          ;GET A POSITION INPUT
2923 7D              MOV    A,L            ;XFER TOUCH LOC TO A

2924 FE72     ONCHK: CPI    TOGON          ;CHECK FOR ON TOUCH
2926 C23329          JNZ    OFFCHK         ;
2929 CD0727          CALL   HORN           ;BEEP
292C E1              POP    H
292D 36FF            MVI    M,0FFH         ;SET FLAG CK RANGING DIST.
292F CD0227          CALL   HORN1          ;LOW BEEP
2932 C9              RET

2933 FE75     OFFCHK:       CPI     TOGOFF         ;CHECK FOR OFF TOUCH
2935 C22029          JNZ    ONOFF2         ;
2938 CD0727          CALL   HORN           ;BEEP
293B E1              POP    H
293C 3600            MVI    M,00H          ;SET FLAG ALL LO,L
                                           ;XFER TOUCH LOCATION TO A
293E CD0227          CALL   HORN1          ;LOW BEEP
2941 C9              RET

2942 CD5E29   MENUER:       CALL    PADRD          ;GET MENU STATUS
2945 7C              MOV    A,H            ;X-FER STATUS BITS
2946 E640            ANI    MENERR         ;MASK FOR MENU ERROR
2948 FE40            CPI    MENERR         ;
294A CA5329          JZ     FLASH          ;CONTINUE TO FLASH IF ERROR
294D 3E00            MVI    A,FALSE        ;TURN OFF LED
294F 322537          STA    FLGERR
2952 C9              RET                   ;RETURN TO PROGRAM
2953 CD0727   FLASH: CALL   HORN           ;OTHERWISE, BEEP
2956 3EFF            MVI    A,TRUE
2958 322537          STA    FLGERR
295B C34229          JMP    MENUER         ;REPEAT ERROR CHECK
              ;
              ;
              ;
              ;*************************************
              ;* PROGRAM NAME: SCAN.SRC            *
              ;*   THE PURPOSE OF THIS PROGRAM IS  *
              ;*   SCAN THE TOUCH PAD BY PLACING   *
              ;*   AN LED SELECT ON THE OUTPUT, AND *
              ;*   READING THE STATUS OF THE COR-  *
              ;*   RESPONDING TRANSISTOR.  IT WILL *
              ;*   THEN PRINT THE LOCATION ON THE  *
              ;*   MONITOR.                        *
              ;*   NOTE:                           *
              ;*   THE VOLTAGE REFERENCE SHOULD BE *
              ;*   SET AT 3.0 VOLTS.               *
              ;*************************************

295E 3E82     PADRD: MVI    A,10000010B    ;PORT A: OUTPUT
2960 D313            OUT    PIACNTL        ;PORT B: INPUT
                                           ;PORT C (UPPER): OUTPUT
                                           ;PORT C (LOWER): OUTPUT

2962 210000   MENU:  LXI    H,00H          ;RESET HL FOR NEW DATA/STATUS INFO
2965 0E00            MVI    C,00H          ;RESET (C) FOR NEW TOUCH LOCATION
```

```
2967 Ø6Ø5           MVI    B,Ø5H          ;LOAD MENU SELECT COUNTER+1
2969 Ø5      LOOP3: DCR    B              ;DECREMENT COUNTER OF MENU SELECT BITS
296A 3A2537         LDA    FLGERR
296D FEØØ           CPI    FALSE
296F CA7829         JZ     OUT1
2972 78             MOV    A,B
2973 F68Ø           ORI    ERRLED
2975 C37929         JMP    OUT2
2978 78      OUT1:  MOV    A,B            ;TRANSFER (B) TO (A) FOR OUTPUT
2979 F64Ø    OUT2:  ORI    EXTMSK         ;MASK FOR EXTRA DEMUX SELECT
297B D31Ø           OUT    PIAA           ;OUTPUT COUNT TO SELECT MENU SELECT BIT
297D 11AØØØ         LXI    D,ØAØH         ;SET UP DELAY COUNT
298Ø CD8C2A         CALL   DELAYD         ;SHORT DELAY
2983 DB11           IN     PIAB           ;INPUT TRANSISTOR STATUS
2985 E6Ø1           ANI    BEAMSK         ;PREPARE INPUT DATA (MASK)
2987 B4             ORA    H              ;OR CURRENT (H) DATA WITH LED STATUS
2988 17             RAL                   ;ROTATE THE (A) LEFT TO MOVE BITS ONE
2989 67             MOV    H,A            ;TRANSFER RESULT TO (H) AGAIN
298A 78             MOV    A,B            ;CHECK COUNT TO SEE IF = Ø
298B FEØØ           CPI    ØØH            ;
298D C26929         JNZ    LOOP3          ;REPEAT PROCESS IF 5 PAIRS NOT YET SCANNED
299Ø 7C             MOV    A,H            ;VALIDATE MENU DATA
2991 1F             RAR                   ;REPOSITION THE MENU DATA (ROTATED)
2992 67             MOV    H,A            ;
2993 FEØØ    ERR1:  CPI    ØØH            ;CHECK FOR NO BEAMS BLOCKED (NO MENU)
2995 C29D29         JNZ    ERR2           ;CHECK FOR NEXT ERROR IF NOT ERROR 1
2998 264Ø           MVI    H,MENERR       ;SIGNAL MENU ERROR
299A C3162A         JMP    PNTDAT         ;FINISH AND PRINT MSGS
299D FE1F    ERR2:  CPI    1FH            ;CHECK FOR ALL BEAMS BROKEN (FALSE MENU)
299F C2A729         JNZ    SCAN           ;CONTINUE SCAN IF NO MENU ERRORS
29A2 264Ø           MVI    H,MENERR       ;SIGNAL MENU ERROR
29A4 C3162A         JMP    PNTDAT         ;FINISH AND PRINT MSGS

29A7 ØEØØ    SCAN:   MVI   C,ØØH          ;CLEAR ROW/COL REGISTER

29A9 Ø61Ø    ROW:    MVI   B,1ØH          ;INITIAL COUNTER VALUE OF 16 LEDS + 1

29AB Ø5      LOOP4: DCR    B              ;DECREMENT COUNTER
29AC 3A2537         LDA    FLGERR
29AF FEØØ           CPI    FALSE
29B1 CABA29         JZ     OUT3
29B4 78             MOV    A,B
29B5 F68Ø           ORI    ERRLED
29B7 C3BB29         JMP    OUT4
29BA 78      OUT3:  MOV    A,B            ;TRANSFER COUNT TO ACCUM
29BB F61Ø    OUT4:  ORI    ROWMSK         ;PREPARE FOR ROW SELECT (MASK)
29BD D31Ø           OUT    PIAA           ;OUTPUT ROW LED/TRANSISTOR SELECT
29BF 11AØØØ         LXI    D,ØAØH         ;LOAD DELAY COUNTER
29C2 CD8C2A         CALL   DELAYD         ;SHORT DELAY
29C5 DB11           IN     PIAB           ;GET TRANSISTOR STATUS
29C7 E6Ø1           ANI    BEAMSK         ;PREPARE INPUT FROM TRANSISTOR (MASK)
29C9 FEØØ           CPI    ØØH            ;SET ZERO FLAG
29CB CAD729         JZ     COUNT3         ;CONTINUE LOOP IF NO TOUCH ('1'=TOUCH)
29CE 78             MOV    A,B            ;TRANSFER COUNT TO ACCUM
29CF 17             RAL                   ;ROTATE COUNT VALUE TO MS NIBBLE
29DØ 17             RAL
```

```
29D1 17               RAL
29D2 17               RAL
29D3 4F               MOV     C,A              ;SAVE ROW IN ROW/COL REGISTER
29D4 C3DE29           JMP     COL              ;JUMP TO COL SCAN BECAUSE ROW TOUCHED

29D7 78       COUNT3: MOV     A,B              ;MOVE COUNT TO 'A' TO DO ZERO CHECK
29D8 FE00             CPI     00H              ;REPEAT UNLESS CURRENTLY ZERO
29DA C2AB29           JNZ     LOOP4            ;CONTINUE LOOP IF NOT COUNTED OUT
29DD C9               RET                      ;RETURN IF LOOP COMPLETED W/NO TOUCH

29DE 06FF     COL:    MVI     B,0FFH           ;LOAD COLUMN COUNTER - 1

29E0 04       LOOP2: INR      B                ;INCREMENT COLUMN COUNTER
29E1 3A2537           LDA     FLGERR
29E4 FE00             CPI     FALSE
29E6 CAEF29           JZ      OUT5
29E9 78               MOV     A,B
29EA F680             ORI     ERRLED
29EC C3F029           JMP     OUT6
29EF 78       OUT5:   MOV     A,B              ;TRANSFER COL COUNT TO ACCUM
29F0 F620     OUT6:   ORI     COLMSK           ;PREPARE CN LED/TRANSISTOR SELECT (MASK)
29F2 D310             OUT     PIAA             ;SELECT LED/TRANSISTOR
29F4 11A000           LXI     D,0A0H           ;LOAD DELAY COUNTER
29F7 CD8C2A           CALL    DELAYD           ;CALL SHORT DELAY
29FA DB11             IN      PIAB             ;INPUT TRANSISTOR STATUS
29FC E601             ANI     BEAMSK           ;PREPARE INPUT FOR USE (MASK)
29FE FE00             CPI     00H              ;SET ZERO FLAG
2A00 CA0D2A           JZ      COUNT4           ;REPEAT LOOP IF NO TOUCH ('1'=TOUCH)
2A03 78               MOV     A,B              ;TRANSFER COUNT TO ACCUM
2A04 B1               ORA     C                ;COMPLETE ROW/COL DATA IN ACCUM
2A05 4F               MOV     C,A              ;SAVE ROW/COL DATA IN 'C'
                                               ;HIGH NIBBLE: ROW
                                               ;LOW NIBBLE: COLUMN
2A06 7C               MOV     A,H              ;MASK (H) TO SHOW A VALID TOUCH
2A07 F680             ORI     TOUCH            ;
2A09 67               MOV     H,A              ;
2A0A C3162A           JMP     PNTDAT           ;PRINT MESSAGE

2A0D 78       COUNT4: MOV     A,B              ;CHECK TO SEE IF COUNT=16 DECIMAL
2A0E FE0F             CPI     0FH              ;SCANNED ALL 16 LEDS?
2A10 CA6229           JZ      MENU
2A13 C2E029           JNZ     LOOP2            ;CONTINUE LOOP 2 TO CHECK FOR COL TOUCH
2A16 69       PNTDAT: MOV     L,C
2A17 C9               RET
              ;
              ;################### END OF PAD READ ROUTINE ##############
              ;
              ;
              ;######################################
              ;# BEGINNING OF PAD CHECK ROUTINE     #
              ;######################################


2A18 3E82     PADCK: MVI     A,10000010B      ;PORT A: OUTPUT
2A1A D313            OUT     PIACNTL          ;PORT B: INPUT
                                              ;PORT C (UPPER): OUTPUT
                                              ;PORT C (LOWER): OUTPUT
2A1C CD2D2C           CALL   CRLF
2A1F CD2D2C           CALL   CRLF
2A22 11B434           LXI    D,INTMSG         ;PRINT INTRO MESSAGE
2A25 CDAB2D           CALL   MSG              ;
2A28 CD2D2C           CALL   CRLF

2A2B Ø61Ø             MVI    B,1ØH            ;INITIAL COUNTER VALUE OF 16 LEDS + 1

2A2D Ø5       LOOPB:  DCR    B                ;DECREMENT COUNTER
2A2E 78               MOV    A,B              ;TRANSFER COUNT TO ACCUM
2A2F F61Ø             ORI    ROWMSK           ;PREPARE FOR ROW SELECT (MASK)
2A31 D31Ø             OUT    PIAA             ;OUTPUT ROW LED/TRANS.SELECT
2A33 11AØØØ           LXI    D,ØAØH           ;LOAD DELAY COUNTER VALUE
2A36 CD8C2A           CALL   DELAYD           ;SHORT DELAY
2A39 DB11             IN     PIAB             ;GET TRANSISTOR STATUS
2A3B E6Ø1             ANI    BEAMSK           ;PREPARE INPUT FROM TRANS.
2A3D FEØØ             CPI    ØØH              ;SET ZERO FLAG
2A3F CA4F2A           JZ     COUNT1           ;CONTINUE LOOP IF LED/TRANS.OK
2A42 114F34           LXI    D,ROWERR         ;PRINT ROW PAIR ERROR MSG
2A45 CDAB2D           CALL   MSG              ;(A '1' SIGNALING A TOUCH)
2A48 78               MOV    A,B              ;TRANSFER COUNT TO (A)
2A49 CDEF2D           CALL   PHB              ;PRINT HEX COUNTER (WORD)
2A4C CD2D2C           CALL   CRLF

2A4F 78       COUNT1: MOV    A,B              ; DO ZERO CHECK
2A5Ø FEØØ             CPI    ØØH              ;SET FLAGS WITH A COMPARE
2A52 C22D2A           JNZ    LOOPB            ;REPEAT UNLESS CURRENTLY ZERO

2A55 Ø6FF             MVI    B,ØFFH           ;LOAD COLUMN COUNTER - 1

2A57 Ø4       LOOPC:  INR    B                ;INCREMENT COLUMN COUNTER
2A58 78               MOV    A,B              ;TRANSFER COL COUNT TO ACCUM
2A59 F62Ø             ORI    COLMSK           ;COLUMN LED/TRANSISTOR SELECT
2A5B D31Ø             OUT    PIAA             ;SELECT LED/TRANSISTOR
2A5D 11AØØØ           LXI    D,ØAØH           ;LOAD DELAY COUNTER VALUE
2A6Ø CD8C2A           CALL   DELAYD           ;CALL SHORT DELAY
2A63 DB11             IN     PIAB             ;INPUT TRANSISTOR STATUS
2A65 E6Ø1             ANI    BEAMSK           ;PREPARE INPUT FOR USE (MASK)
2A67 FEØØ             CPI    ØØH              ;SET ZERO FLAGS
2A69 CA792A           JZ     COUNT2           ;LOOP IF TRANS./LED PAIR OK
2A6C 117334           LXI    D,COLERR         ;PRINT COLUMN PAIR ERROR
2A6F CDAB2D           CALL   MSG              ;
2A72 78               MOV    A,B              ;MOVE COUNT TO (A)
2A73 CDEF2D           CALL   PHB              ;PRINT HEX WORD (COUNT)
2A76 CD2D2C           CALL   CRLF
2A79 78       COUNT2: MOV    A,B              ;CHECK TO SEE IF COUNT=16 DECIMAL
2A7A FEØF             CPI    ØFH              ;SCANNED ALL 16 LEDS?
2A7C C2572A           JNZ    LOOPC            ;CONT.LOOP 2  CHECK FOR COL ERROR

2A7F 11DA34           LXI    D,ENDMSG         ;PRINT ENDING MESSAGE
2A82 CDAB2D           CALL   MSG              ;
2A85 CD2D2C           CALL   CRLF
2A88 CD2D2C           CALL   CRLF
2A8B C9               RET
```

```
2A8C 1B         DELAYD:     DCX     D           ;DECREMENT DELAY COUNT
2A8D 7A                MOV     A,D             ;COMPARE D AND E
2A8E B3                ORA     E               ;CHECK TO SEE IF DE=Ø
2A8F C28C2A            JNZ     DELAYD          ;REPEAT IF <>Ø
2A92 C9                RET

                ;**********************************************************
                ;       END                     ;END OF PAD CHECK
                ;**********************************************************



                ;*************************
                ;*                       *
                ;* FILE: INPAD.ASM       *
                ;*                       *
                ;* CREATED: DEC 1Ø, 1985 *
                ;* UPDATED:              *
                ;*                       *
                ;* PURPOSE:              *
                ;*   TO ALLOW INPUT OF A *
                ;* HEX VALUE OF ONE BYTE *
                ;* FROM THE TOUCH PAD    *
                ;* NUMERIC KEYPAD, PUTS  *
                ;* IT IN (C)             *
                ;*                       *
                ;*************************
2A93 Ø6Ø1       INPAD: MVI     B,Ø1H           ;ALLOW TWO NIBBLE INPUT
2A95 ØEØØ              MVI     C,ØØH           ;CLEAR (C) FOR USE
2A97 C5         NIB:   PUSH    B               ; SAVE BC REGS.
2A98 CD5E29            CALL    PADRD           ;GET A Ø-F PAD INPUT
2A9B C1                POP     B               ;RESORE BC
2A9C 7C                MOV     A,H             ;CHECK FOR MENU ERROR
2A9D E64Ø              ANI     MENERR          ;
2A9F FE4Ø              CPI     MENERR          ;
2AA1 CA552B            JZ      EXIT            ;EXIT WITH A PAD ERROR
2AA4 7C                MOV     A,H
2AA5 E6Ø1              ANI     PROMSK
2AA7 FEØ1              CPI     PROMSK
2AA9 C2552B            JNZ     EXIT
2AAC 7C                MOV     A,H             ;CHECK FOR VALID TOUCH
2AAD E68Ø              ANI     TOUCH           ;
2AAF FE8Ø              CPI     TOUCH           ;
2AB1 C2972A            JNZ     NIB             ;IF NOT VALID TOUCH & MENU
                                               ;THEN REPEAT UNTIL VALID
2AB4 7D                MOV     A,L             ;IF VALID, MOVE TOUCH
                                               ;LOCATION TO (A)
2AB5 FE18       COMP:  CPI     ZERO            ;ZERO CHECK (COMPARES)
2AB7 C2BF2A            JNZ     NEXT1           ;IF NOT A ZERO, CHECK NEXT
2ABA 3EØØ              MVI     A,ØØH           ;PUT Ø IN
2ABC C3632B            JMP     PUT             ;STORE VALUE, ANOTHER ?
2ABF FE1A       NEXT1: CPI     ONE             ;ONE CHECK
2AC1 C2C92A            JNZ     NEXT2           ;
2AC4 3EØ1              MVI     A,Ø1H           ;
2AC6 C3632B            JMP     PUT             ;
2AC9 FE1C       NEXT2: CPI     TWO             ;TWO CHECK
2ACB C2D32A            JNZ     NEXT3           ;
```

```
2ACE 3EØ2              MVI     A,Ø2H               ;
2ADØ C3632B            JMP     PUT                 ;
2AD3 FE1E      NEXT3:  CPI     THREE               ;THREE CHECK
2AD5 C2DD2A            JNZ     NEXT4               ;
2AD8 3EØ3              MVI     A,Ø3H               ;
2ADA C3632B            JMP     PUT                 ;
2ADD FE38      NEXT4:  CPI     FOUR                ;FOUR CHECK
2ADF C2E72A            JNZ     NEXT5               ;
2AE2 3EØ4              MVI     A,Ø4H               ;
2AE4 C3632B            JMP     PUT                 ;
2AE7 FE3A      NEXT5:  CPI     FIVE                ;FIVE CHECK
2AE9 C2F12A            JNZ     NEXT6               ;
2AEC 3EØ5              MVI     A,Ø5H               ;
2AEE C3632B            JMP     PUT                 ;
2AF1 FE3C      NEXT6:  CPI     SIX                 ;SIX CHECK
2AF3 C2FB2A            JNZ     NEXT7               ;
2AF6 3EØ6              MVI     A,Ø6H               ;
2AF8 C3632B            JMP     PUT                 ;
2AFB FE3E      NEXT7:  CPI     SEVEN               ;SEVEN CHECK
2AFD C2Ø52B            JNZ     NEXT8               ;
2BØØ 3EØ7              MVI     A,Ø7H               ;
2BØ2 C3632B            JMP     PUT                 ;
2BØ5 FE58      NEXT8:  CPI     EIGHT               ;EIGHT CHECK
2BØ7 C2ØF2B            JNZ     NEXT9               ;
2BØA 3EØ8              MVI     A,Ø8H               ;
2BØC C3632B            JMP     PUT                 ;
2BØF FE5A      NEXT9:  CPI     NINE                ;NINE CHECK
2B11 C2192B            JNZ     NEXT1Ø              ;
2B14 3EØ9              MVI     A,Ø9H               ;
2B16 C3632B            JMP     PUT                 ;
2B19 FE5C      NEXT1Ø:         CPI     AHEX                ;A CHECK
2B1B C2232B            JNZ     NEXT11              ;
2B1E 3EØA              MVI     A,ØAH               ;
2B2Ø C3632B            JMP     PUT                 ;
2B23 FE5E      NEXT11:         CPI     BHEX                ;B CHECK
2B25 C22D2B            JNZ     NEXT12              ;
2B28 3EØB              MVI     A,ØBH               ;
2B2A C3632B            JMP     PUT                 ;
2B2D FE78      NEXT12:         CPI     CHEX                ;C CHECK
2B2F C2372B            JNZ     NEXT13              ;
2B32 3EØC              MVI     A,ØCH               ;
2B34 C3632B            JMP     PUT                 ;
2B37 FE7A      NEXT13:         CPI     DHEX                ;D CHECK
2B39 C2412B            JNZ     NEXT14              ;
2B3C 3EØD              MVI     A,ØDH               ;
2B3E C3632B            JMP     PUT                 ;
2B41 FE7C      NEXT14:         CPI     EHEX                ;E CHECK
2B43 C24B2B            JNZ     NEXT15              ;
2B46 3EØE              MVI     A,ØEH               ;
2B48 C3632B            JMP     PUT                 ;
2B4B FE7E      NEXT15:         CPI     FHEX                ;F CHECK
2B4D C2972A            JNZ     NIB                 ;IF NOT Ø-F THEN NOT VALID
                                                   ;SO REPEAT WAIT FOR VALID
2B5Ø 3EØF              MVI     A,ØFH               ;
2B52 C3632B            JMP     PUT                 ;
```

```
2B55 2640      EXIT:  MVI    H,MENERR        ;SET MENU ERROR FLAG
2B57 2E00             MVI    L,00H           ;
2B59 F5               PUSH   PSW
2B5A CD0227           CALL   HORN1
2B5D F1               POP    PSW
2B5E 79        LEAVE: MOV    A,C
2B5F CDEF2D           CALL   PHB             ;PRINT VALUE FROM PAD
2B62 C9               RET                    ;EXIT PROGRAM

2B63 F5        PUT:   PUSH   PSW             ;SAVE NIBBLE DATA
2B64 CD0727           CALL   HORN            ; BEEP IF KEY PRESSED
2B67 F1               POP    PSW             ;RESTORE NIBBLE DATA
2B68 B1               ORA    C               ;COMBINE CURRENT C VALUE
2B69 4F               MOV    C,A             ;XFER BACK TO (C)
2B6A 78               MOV    A,B             ;CHECK TO SEE IF TWO
                                             ;NIBBLES ARE IN
2B6B FE00             CPI    00H             ;IS COUNTER (B) ZERO?
2B6D CA5E2B           JZ     LEAVE           ;LEAVE PROGRAM IF BYTE IN
2B70 05               DCR    B               ;DECREMENT (B) COUNTER
2B71 79               MOV    A,C             ;ROTATE FIRST NIBBLE LEFT
2B72 17               RAL
2B73 17               RAL
2B74 17               RAL
2B75 17               RAL
2B76 4F               MOV    C,A             ;RESTORE (C) VALUE ROTATED
2B77 DB11      POLL:  IN     PIAB            ;POLL FOR NO TOUCH
2B79 E601             ANI    BEAMSK          ;
2B7B FE01             CPI    BEAMSK          ;
2B7D CA772B           JZ     POLL            ;REPEAT UNTIL NO TOUCH
2B80 C3972A           JMP    NIB             ;RETURN FOR NEXT NIBBLE

               ;#############################################################
               ;
               ;      UTILITY ROUTINES - IN ALPHABETICAL ORDER (SORT OF)
               ;
               ;#############################################################
               ;
               ; BCDTBIN - CONVERT BCD IN H&L TO BINARY IN H&L
               ;          ONLY H&L CHANGED
               ;
2B83 C5        BCDTBIN:PUSH  B       ;
2B84 D5               PUSH   D       ;
2B85 54               MOV    D,H     ;COPY ORIGINAL
2B86 5D               MOV    E,L     ;
2B87 2600             MVI    H,0     ;INITIALIZE UPPER PART OF RESULT
2B89 0600             MVI    B,0     ;INITIAL UPPER PART OF B&C
2B8B 7A               MOV    A,D     ;GET UPPER DIGIT
2B8C 0F               RRC            ;
2B8D 0F               RRC            ;
2B8E 0F               RRC            ;
2B8F 0F               RRC            ;
2B90 E60F             ANI    0FH     ;
2B92 6F               MOV    L,A     ;START RESULT
2B93 CDBB2D           CALL   MULT10  ;SHIFT UP ONE DIGIT IN BASE 10
2B96 7A               MOV    A,D     ;GET NEXT TO TOP DIGIT
2B97 E60F             ANI    0FH     ;
```

```
2B99 4F                 MOV     C,A     ;
2B9A Ø9                 DAD     B       ;COMBINE WITH TOP DIGIT
2B9B CDBB2D             CALL    MULT1Ø  ;SHIFT UP ONE DIGIT IN BASE 1Ø
2B9E 7B                 MOV     A,E     ;GET NEXT TO BOTTOM DIGIT
2B9F ØF                 RRC             ;
2BAØ ØF                 RRC             ;
2BA1 ØF                 RRC             ;
2BA2 ØF                 RRC             ;
2BA3 E6ØF               ANI     ØFH     ;
2BA5 4F                 MOV     C,A     ;
2BA6 Ø9                 DAD     B       ;COMBINE WITH TOP TWO DIGITS
2BA7 CDBB2D             CALL    MULT1Ø  ;SHIFT UP ONE DIGIT IN BASE 1Ø
2BAA 7B                 MOV     A,E     ;GET BOTTOM DIGIT
2BAB E6ØF               ANI     ØFH     ;
2BAD 4F                 MOV     C,A     ;
2BAE Ø9                 DAD     B       ;COMBINE WITH TOP THREE DIGITS
2BAF D1                 POP     D       ;
2BBØ C1                 POP     B       ;
2BB1 C9                 RET             ;
                ;       END     BCDTBIN ;
                ;
                ;
                ; CALLIN - INDIRECT CALL TO (H&L)
                ;
2BB2 E9         CALLIN: PCHL            ;
                ;                       ;
                ;       END     CALLIN  ;
                ;
                ; I/O ROUTINES
                ;
2BB3 DBØ1       CI:     IN      SERCON          ;WAIT FOR DATA READY
2BB5 E6Ø2               ANI     2               ;
2BB7 CAB32B             JZ      CI              ;
2BBA DBØØ               IN      SERDAT          ;GET BYTE
2BBC F5                 PUSH    PSW             ;SAVE PSW
2BBD 3AØ237             LDA     ECHOFL          ;CHECK ECHO FLAG
2BCØ B7                 ORA     A               ;
2BC1 C21E2C             JNZ     COEND           ;IF NOT ZERO ECHO-RET ON CO
2BC4 F1                 POP     PSW             ;ECHO CHARACTER
2BC5 F5                 PUSH    PSW             ;
2BC6 C3Ø22C             JMP     C1              ;GO ECHO CHARACTER
                ;
                ;
                ; CISTAT - RETURNS NON-ZERO IN A IF RECIEVER BUFFER HAS A CHAR
                ;
2BC9 C5         CISTAT: PUSH    B               ;
2BCA F5                 PUSH    PSW             ;
2BCB DBØ1               IN      SERCON          ;
2BCD E6Ø2               ANI     2               ;
2BCF C1                 POP     B               ;
2BDØ 78                 MOV     A,B             ;
2BD1 C1                 POP     B               ;
2BD2 C9                 RET                     ;
                ;
                ;**** CO CONSOLE OUTPUT - DESTROYS ONLY FLAGS...
                ;
```

```
2BD3 F5         CO:     PUSH    PSW                 ;
2BD4 3AØØ37             LDA     COCOOK              ;IF IN RAW MODE JUST OUTPUT
2BD7 B7                 ORA     A                   ;
2BD8 C2Ø22C             JNZ     C1                  ;
2BDB CDC92B             CALL    CISTAT              ;IS CHAR WAITING ?
2BDE CAØ22C             JZ      C1                  ;NOPE...

2BE1 CDB32B             CALL    CI                  ;YEP...
2BE4 E67F               ANI     Ø7FH                ;
2BE6 FE13               CPI     XOFF                ;TURN OFF XMIT ?
2BE8 C2FA2B             JNZ     C5                  ;NO - TEST FOR RUBOUT
2BEB CDB32B     C4:     CALL    CI                  ;WAIT FOR XON
2BEE E67F               ANI     Ø7FH                ;

2BFØ FE7F               CPI     RUB                 ;QUIT IF RUBOUT
2BF2 CAFF2B             JZ      C6                  ;
2BF5 FE11               CPI     XON                 ;
2BF7 C2EB2B             JNZ     C4                  ;JUST DROP THROUGH..NOT RUB ANYWAY
2BFA FE7F       C5:     CPI     RUB                 ;INTERRUPT ?
2BFC C2Ø22C             JNZ     C1                  ;NO...IGNORE
2BFF CD182E     C6:     CALL    RETJMP              ;
2CØ2 DBØ1       C1:     IN      SERCON              ;
2CØ4 ØF                 RRC                         ;
2CØ5 D2Ø22C             JNC     C1                  ;
2CØ8 F1                 POP     PSW                 ;
2CØ9 F5                 PUSH    PSW                 ;
2CØA D3ØØ               OUT     SERDAT              ;
2CØC FEØD               CPI     CR                  ;IF CR THEN DELAY
2CØE C21E2C             JNZ     COEND               ;NOT CR - QUIT
2C11 3AØ137             LDA     DLYRAM              ;
2C14 3D         C2:     DCR     A                   ;
2C15 FA1E2C             JM      COEND               ;
2C18 CD4C2C             CALL    D1ØMS               ;DELAY 1ØMS
2C1B C3142C             JMP     C2                  ;
2C1E F1         COEND:  POP     PSW                 ;
2C1F C9                 RET                         ;
                ;****** CMP16 ** 16 BIT COMPARE H&L AND D&E ********************
                ;
                ;       IF( H&L = D&E ) Z=1, CY=Ø
                ;       IF( H&L > D&E ) Z=Ø, CY=Ø
                ;       IF( H&L < D&E ) Z=Ø, CY=1
                ;
2C2Ø E5         CMP16:  PUSH    H                   ;SAVE PSW & H&L
2C21 F5                 PUSH    PSW                 ;
2C22 7C                 MOV     A,H                 ;IF H  = D ENOUGH INFO FOUND
2C23 92                 SUB     D                   ;
2C24 C2292C             JNZ     CMP16E              ;
2C27 7D                 MOV     A,L                 ;IF H=D THEN COMPARE LOWER BYTES
2C28 93                 SUB     E                   ;
2C29 E1         CMP16E: POP     H                   ;
2C2A 7C                 MOV     A,H                 ;
2C2B E1                 POP     H                   ;
2C2C C9                 RET                         ;
                ;       END     CMD16               ;
2C2D D5         CRLF:   PUSH    D
2C2E 114B33             LXI     D,MCRLF
2C31 CDA82D             CALL    MSG
2C34 D1                 POP     D
```

```
2C35 C9                RET
               ; D5ØMS - ASSUMES ECKLDV HAS BEEN SET BY SOMEBODY
               ;
2C36 F5        D5ØMS:  PUSH    PSW            ;SAVE PSW
2C37 E5                PUSH    H              ;SAVE H&L
2C38 2AØD37            LHLD    D5ØDIV         ;
2C3B E3        D5ØMSL: XTHL                   ;18
2C3C E3                XTHL                   ;18
2C3D E3                XTHL                   ;18
2C3E E3                XTHL                   ;18
2C3F E5                PUSH    H              ;11
2C4Ø E1                POP     H              ;1Ø
2C41 2B                DCX     H              ; 5
2C42 23                INX     H              ; 5
2C43 2B                DCX     H              ; 5
2C44 7C                MOV     A,H            ; 5
2C45 B5                ORA     L              ; 4
2C46 C23B2C            JNZ     D5ØMSL         ;11
2C49 E1                POP     H              ;
2C4A F1                POP     PSW            ;
2C4B C9                RET
               ;
               ; D1ØMS - DELAY 1Ø MS
               ;
2C4C E5        D1ØMS:  PUSH    H       ;
2C4D F5                PUSH    PSW     ;
2C4E 21Ø1Ø3            LXI     H,769   ;
2C51 7D        DTWIDL: MOV     A,L     ;   ;~Ø.Ø1 SECONDS ON A 2 MHZ 8Ø85      5
2C52 B4                ORA     H       ;   ;  (CPU CLOCK FREQ)                  4
2C53 2B                DCX     H       ;   ;                                   1Ø
2C54 C2512C            JNZ     DTWIDL  ;   ;                   8Ø85/8Ø8Ø     7/1Ø
2C57 F1                POP     PSW     ;   ;                   TOTAL        26/29
2C58 E1                POP     H       ;
2C59 C9                RET             ;
               ;       END     D1ØMS   ;
               ;
               ; D5SEC - DELAY 5 SECONDS
               ;
2C5A C5        D5SEC:  PUSH    B              ;
2C5B Ø664              MVI     B,Ø64H         ;WAIT 5 SECOND FOR +25 SWITCHING
2C5D CD362C    ON16W1: CALL    D5ØMS          ;REGULATOR TO TURN ON OR OFF.
2C6Ø Ø5                DCR     B              ;
2C61 C25D2C            JNZ     ON16W1         ;
2C64 C1                POP     B              ;
2C65 C9                RET                    ;
               ;       END     D5SEC          ;
               ;
               ; DISASC - DISPLAY ASCII A-REG INTO H&L
               ;
2C66 F5        DISASC: PUSH    PSW     ;SAVE PSW
2C67 E67F              ANI     Ø7FH    ;STRIP PARITY
2C69 262Ø              MVI     H,Ø2ØH  ;PUT SPACE IN H-REG
2C6B FE2Ø              CPI     Ø2ØH    ;CHAR < Ø2ØH ?
2C6D D2742C            JNC     DA1     ;NO-IS PRINTABLE
2C7Ø 265E              MVI     H,Ø5EH  ;NOT PRINTABLE - C = '^'
2C72 C64Ø              ADI     Ø4ØH    ;MAKE PRINTABLE
```

```
2C74 FE7F       DA1:    CPI     Ø7FH    ;IS RUBOUT ?
2C76 C27B2C             JNZ     DA2     ;NOPE...AOK
2C79 3E2Ø               MVI     A,Ø2ØH  ;YEP-MAKE SPACE
2C7B 6F         DA2:    MOV     L,A     ;
2C7C F1                 POP     PSW     ;RESTORE PSW
2C7D C9                 RET             ;
                ;
                ;#############################################################
                ;
                ;       FRMCNT - ASKS " FROM "XXXX" TO "YYYY
                ;
2C7E D5         FRMCNT: PUSH    D       ;
2C7F E5                 PUSH    H       ;
2C8Ø CD8F2C             CALL    FROMTO  ;
2C83 DAB12C             JC      FRTOE   ;
2C86 E5                 PUSH    H       ;
2C87 CD742E             CALL    SUB16   ;CALC NUMBER OF BYTES TO BE PROCESSED
2C8A D1                 POP     D       ;
2C8B 2B                 DCX     H       ;H&L = NEGATIVE OF NUMBER OF BYTES
2C8C C3AA2C             JMP     FRCLN   ;THIS DOES XCHG & CLEANS OFF STACK...
                ;       END     FROMTO
                ;
                ; FROMTO - " FROM "XXXX" TO "YYYY
                ;
2C8F D5         FROMTO: PUSH    D       ;
2C9Ø E5                 PUSH    H       ;
2C91 11FØ2F             LXI     D,PLO   ;PROMPT FOR LO LIMIT
2C94 CDAB2D             CALL    MSG     ;
2C97 CD452D             CALL    GHW     ;
2C9A DAB12C             JC      FRTOE   ;RETURN IF ERRER
2C9D 11EB2F             LXI     D,PHI   ;PROMPT FOR HI LIMIT
2CAØ CDAB2D             CALL    MSG     ;
2CA3 EB                 XCHG            ;
2CA4 CD452D             CALL    GHW     ;
2CA7 DAB12C             JC      FRTOE   ;
2CAA EB         FRCLN:  XCHG            ;
2CAB E3                 XTHL            ;GET CRAP OFF OF STACK
2CAC E1                 POP     H       ;
2CAD E3                 XTHL            ;
2CAE E1                 POP     H       ;
2CAF B7                 ORA     A       ;BETTER BE SURE CARRY IS CLEAR
2CBØ C9                 RET             ;RETURN...
2CB1 E1         FRTOE:  POP     H       ;
2CB2 D1                 POP     D       ;
2CB3 C9                 RET             ;
                ; GBIAS - GET 16 BIT BIAS
                ;
2CB4 F5         GBIAS:  PUSH    PSW             ;SAVE PSW
2CB5 E5                 PUSH    H               ;  AND H&L
2CB6 D5                 PUSH    D               ;  AND D&E
2CB7 11F72F             LXI     D,PBIAS         ;PRINT BIAS MESSAGE
2CBA CDAB2D             CALL    MSG             ;
2CBD CD452D             CALL    GHW             ;GET BIAS
2CCØ D2EØ2C             JNC     GBIAS2          ;IF NO CARRY GOOD BIAS ENTERED
2CC3 FE2D               CPI     '-'             ;CHECK FOR NEGATIVE BIAS
2CC5 CAD32C             JZ      GBIAS1          ;OHHH- WANT NEGATIVE NUMBER ...
```

```
2CC8 FEØD               CPI     CR                ;CARRIAGE RETURN ?
2CCA C2E82C             JNZ     GBIASE            ;NOPE ERRER
2CCD 21ØØØØ             LXI     H,Ø               ;AHHHH - NO BIAS
2CDØ C3EØ2C             JMP     GBIAS2            ;
2CD3 CD452D     GBIAS1: CALL    GHW               ;GET NEGATIVE BIAS
2CD6 DAE82C             JC      GBIASE            ;BAD CHAR...BYE
2CD9 11ØØØØ             LXI     D,Ø               ;
2CDC EB                 XCHG                      ;SET UP SUBTRACTION FROM ZERO
2CDD CD742E             CALL    SUB16             ;NEGATE BIAS
2CEØ CD2D2C     GBIAS2: CALL    CRLF              ;PREVENT A MESS
2CE3 D1                 POP     D                 ;RESTORE D
2CE4 F1                 POP     PSW               ;LOOSE ORIGINAL H&L
2CE5 F1                 POP     PSW               ;RESTORE PSW
2CE6 B7                 ORA     A                 ;CLEAR CARRY
2CE7 C9                 RET
2CE8 D1         GBIASE: POP     D                 ;RESTORE D&E
2CE9 E1                 POP     H                 ;RESTORE ORIGINAL H&L
2CEA F1                 POP     PSW               ;RESTORE PSW
2CEB 37                 STC                       ;SET CARRY
2CEC C9                 RET
                ;################################################################
                ;
                ; GCLKFQ - AND WHAT FREQUENCY IS YOUR CLOCK TODAY, LITTLE BOY ?
                ;
2CED F5         GCLKFQ: PUSH    PSW     ;SAVE PSW
2CEE C5                 PUSH    B       ;SAVE B&C
2CEF D5                 PUSH    D       ;SAVE D
2CFØ E5                 PUSH    H       ;SAVE H&L
2CF1 2AØF37             LHLD    CLKBCD            ;GET BCD CLOCK FREQUENCY
2CF4 7C                 MOV     A,H               ;
2CF5 FE1Ø               CPI     Ø1ØH              ;Ø1ØØØH SMALLEST
2CF7 DA192D             JC      GCLK1             ;
2CFA FE9A               CPI     Ø9AH              ;Ø9999H BIGGEST
2CFC D2192D             JNC     GCLK1             ;
2CFF CD832B             CALL    BCDTBIN           ;CHECK TO SEE IF BINARY
2DØ2 EB                 XCHG                      ;  CLOCK FREQ AGREES
2DØ3 2A1137             LHLD    CLKBIN            ;
2DØ6 CD2Ø2C             CALL    CMP16             ;
2DØ9 C2192D             JNZ     GCLK1             ;NOT SAME
2DØC CD8B2D             CALL    M5Ø128            ;CHECK DIVISOR TO SEE IF
2DØF EB                 XCHG                      ;  IT AGREES
2D1Ø 2AØD37             LHLD    D5ØDIV            ;
2D13 CD2Ø2C             CALL    CMP16             ;
2D16 CA4Ø2D             JZ      GCLK2             ;EVERYTHING CONSISTENT-
                ;                                 ;  HOPE IT IS GOOD
2D19 11842F     GCLK1:  LXI     D,GCLKM ;GET CLK FREQ
2D1C CDAB2D             CALL    MSG     ;
2D1F CD452D             CALL    GHW     ;GET FREQUENCY IN KHZ
2D22 DC1B2E             CC      RETJMP  ;TAKE NO CRAP HERE...
2D25 7C                 MOV     A,H     ;CHECK FOR POSSIBLE
2D26 E6FØ               ANI     ØFØH    ;MUST BE AT LEAST 1 MHZ
2D28 CA192D             JZ      GCLK1   ;--ASK UNTIL THEY GET IT RIGHT
2D2B CDC42D             CALL    OKCK    ;
2D2E 22ØF37             SHLD    CLKBCD  ;SAVE BCD CLOCK FREQ
2D31 DC1B2E             CC      RETJMP  ;TAKE NO CRAP HERE...
2D34 CD832B             CALL    BCDTBIN ;CONVERT H&L TO BINARY
```

```
2D37 221137             SHLD    CLKBIN  ;SAVE BINARY CLOCK FREQ
2D3A CD8B2D             CALL    M5Ø128  ;MULTIPLY BY 5Ø/128
2D3D 22ØD37             SHLD    D5ØDIV  ;*12.5/32 (5Ø/128)
2D4Ø E1         GCLK2:  POP     H       ;IF YOU DON'T KNOW WHAT THESE ARE FOR BY
2D41 D1                 POP     D       ;
2D42 C1                 POP     B       ;
2D43 F1                 POP     PSW     ;  NOW YOU'RE A LOST CAUSE...
2D44 C9                 RET             ;
                ;
                ; GHW - GET HEX WORD
                ;
2D45 C5         GHW:    PUSH    B
2D46 F5                 PUSH    PSW
2D47 CD5C2D             CALL    GHB             ; GET FIRST BYTE IN A-REGISTER
2D4A DA592D             JC      GHWEND          ; RETURN IF BAD CHAR
2D4D 67                 MOV     H,A             ; MOVE BYTE TO FINAL DESTINATION
2D4E CD5C2D             CALL    GHB             ; GET SECOND BYTE
2D51 DA592D             JC      GHWEND          ;
2D54 6F                 MOV     L,A             ;
2D55 C1                 POP     B               ;
2D56 78                 MOV     A,B             ;
2D57 C1                 POP     B               ;
2D58 C9                 RET                     ;
2D59 C1         GHWEND: POP     B               ;
2D5A C1                 POP     B               ; DO NOT RESTORE A
2D5B C9                 RET                     ;
                ;       END     GHW             ;
                ;
                ; GHB - GET HEX BYTE
                ;
2D5C C5         GHB:    PUSH    B               ; SAVE B&C
2D5D CD712D             CALL    GHD             ; GET FIRST HEX DIGIT IN A-REG
2D6Ø DA6F2D             JC      GHBEND          ; IF BAD CHAR QUIT AND PASS BACK
2D63 Ø7                 RLC                     ; SHIFT TO UPPER HALF OF BYTE
2D64 Ø7                 RLC                     ;   .
2D65 Ø7                 RLC                     ;   .
2D66 Ø7                 RLC                     ;   .
2D67 47                 MOV     B,A             ; SAVE FIRST DIGIT
2D68 CD712D             CALL    GHD             ; GET SECOND DIGIT
2D6B DA6F2D             JC      GHBEND          ; BAD CHAR READ, RET IT TO CALLER
2D6E BØ                 ORA     B               ; COMBINE FIRST AND SECOND DIGITS
2D6F C1         GHBEND: POP     B               ; RESTORE ORIGINAL B&C
2D7Ø C9                 RET                     ;
                ;       END     GHB             ;
                ;
                ; GHD - GET HEX DIGIT
                ;
2D71 CDB32B     GHD:    CALL    CI              ; GET CHARACTER & ECHO
                ;       ANI     Ø7FH            ; PUT IN IF UCASE TAKEN OUT
2D74 CD8Ø2E     ATH:    CALL    UCASE           ; MAP LOWER TO UPPER CASE AND
                ;                               ;   STRIP PARITY.
2D77 FE3Ø               CPI     'Ø'
2D79 D8                 RC                      ;       NON-HEX CHARACTER
2D7A FE3A               CPI     ':'             ; IF  (A) =< '9'+1
2D7C DA882D             JC      GHD2            ;       'Ø'-'9' TYPED - CONVERT
2D7F FE41               CPI     'A'             ; IF  (A) < 'A'
```

```
2D81 D8                 RC                      ;       NON-HEX CHARACTER
2D82 FE47               CPI     'G'             ; IF  (A) >= 'G'
2D84 3F                 CMC                     ;       .
2D85 D8                 RC                      ;       NON-HEX CHARACTER
2D86 D607               SUI     07H             ; SHIFT 'A'-'F' DOWN
2D88 D630       GHD2:   SUI     '0'             ; CONVERT
2D8A C9                 RET                     ;
                ;       END     GHD             ;
                ;
                ; M50128 - MULTIPLY BY 50/128
                ;
2D8B 0601       M50128: MVI     B,1     ;DIVIDE BY TWO SO * 12.5
2D8D CD362E             CALL    SHRHL   ;  WILL FIT IN 16 BITS.
2D90 CD2B2E             CALL    RNDHL   ;  AND ROUND
2D93 54                 MOV     D,H     ;SAVE *1
2D94 5D                 MOV     E,L     ; .
2D95 29                 DAD     H       ;*2
2D96 29                 DAD     H       ;*4
2D97 44                 MOV     B,H     ;SAVE *4 IN D&E
2D98 4D                 MOV     C,L     ;
2D99 29                 DAD     H       ;*8
2D9A 09                 DAD     B       ;*12
2D9B EB                 XCHG            ;GENERATE * 0.5
2D9C 0601               MVI     B,1     ; .
2D9E CD362E             CALL    SHRHL   ; .
2DA1 19                 DAD     D       ;*12 + *0.5
2DA2 0604               MVI     B,4     ;DIVIDE H&L BY 16
2DA4 CD362E             CALL    SHRHL   ;
2DA7 CD2B2E             CALL    RNDHL   ;ROUND
2DAA C9                 RET
                ;
                ;       END     M50128
                ;
                ; MSG -
2DAB F5         MSG:    PUSH    PSW
2DAC 1A         LOUPE:  LDAX    D       ;GET CHAR
2DAD FEFF               CPI     EOL     ;END OF STRING?
2DAF 13                 INX     D       ;BUMP POINTER
2DB0 CAB92D             JZ      MDN     ;JUMP IF SO
2DB3 CDD32B             CALL    CO      ;ELSE PRINT IT
2DB6 C3AC2D             JMP     LOUPE   ;DO IT AGAIN
2DB9 F1         MDN:    POP     PSW
2DBA C9                 RET
                ;
                ; MULT10 - MULTIPLY H&L BY 10
                ;
2DBB D5         MULT10: PUSH    D       ;
2DBC 29                 DAD     H       ;*2
2DBD 54                 MOV     D,H     ;SAVE *2
2DBE 5D                 MOV     E,L     ;
2DBF 29                 DAD     H       ;*4
2DC0 29                 DAD     H       ;*8
2DC1 19                 DAD     D       ;*10
2DC2 D1                 POP     D       ;
2DC3 C9                 RET             ;
                ;
```

```
2DC4 D5         OKCK:   PUSH    D
2DC5 F5                 PUSH    PSW
2DC6 11A42F             LXI     D,MOK
2DC9 CDAB2D             CALL    MSG
2DCC CDB32B             CALL    CI
2DCF E67F               ANI     07FH
2DD1 FE0D               CPI     CR
2DD3 CADD2D             JZ      OKCKEND
2DD6 11C22E             LXI     D,ABORT
2DD9 CDAB2D             CALL    MSG
2DDC 37                 STC
2DDD CD2D2C     OKCKEND:CALL    CRLF
2DE0 D1                 POP     D
2DE1 7A                 MOV     A,D
2DE2 D1                 POP     D
2DE3 C9                 RET
                ;       END     OKCK
                ;
                ; PHW - PRINT HEX WORD
                ;
2DE4 F5         PHW:    PUSH    PSW                     ; SAVE A-REGISTER AND FLAGS
2DE5 7C                 MOV     A,H                     ;
2DE6 CDEF2D             CALL    PHB                     ; PRINT HIGH-ORDER BYTE
2DE9 7D                 MOV     A,L                     ;
2DEA CDEF2D             CALL    PHB                     ; PRINT LOW-ORDER BYTE
2DED F1                 POP     PSW                     ; RESTORE A-REGISTER AND FLAGS
2DEE C9                 RET                             ;
                ;       END     PHW                     ;
                ;
                ; PHB - PRINT HEX BYTE
                ;
2DEF F5         PHB:    PUSH    PSW                     ; SAVE PSW
2DF0 C5                 PUSH    B                       ; SAVE B&C
2DF1 47                 MOV     B,A                     ; SAVE LOWER NIBBLE
2DF2 0F                 RRC                             ; SHIFT TO LOWER HALF OF BYTE
2DF3 0F                 RRC                             ;   .
2DF4 0F                 RRC                             ;   .
2DF5 0F                 RRC                             ;   .
2DF6 CD012E             CALL    PHD                     ; PRINT UPPER HEX DIGIT
2DF9 78                 MOV     A,B                     ; GET LOWER NIBBLE
2DFA CD012E             CALL    PHD                     ; ...AND PRINT
2DFD 78                 MOV     A,B                     ; RESTORE ORIGINAL BYTE TO A
2DFE C1                 POP     B                       ; RESTORE B&C
2DFF F1                 POP     PSW                     ; RESTORE PSW
2E00 C9                 RET                             ;
                ;       END     PHB                     ;
                ;
                ; PHD - PRINT HEX DIGIT
                ;
2E01 F5         PHD:    PUSH    PSW                     ;SAVE PSW
2E02 E60F               ANI     0FH                     ; MASK OFF LOWER NIBBLE
2E04 C630               ADI     '0'                     ; CONVERT '0'-'9' TO ASCII
2E06 FE3A               CPI     '9'+1                   ; IF '0'-'9'
2E08 DA0D2E             JC      PHD1                    ;       THEN DONE
2E0B C607               ADI     'A'-':'                 ; CONVERT 'A'-'F'
2E0D CDD32B     PHD1:   CALL    CO                      ; PRINT DIGIT
```

```
2E1Ø F1                 POP     PSW             ;
2E11 C9                 RET                     ;
                ;       END     PHD             ;
                ;
                ; POPPC - POP THE PC INTO H&L
                ;       - ON RETURN (H&L) = ADDRESS RETURNED TO
                ;
2E12 E1         POPPC:  POP     H               ;
2E13 E9                 PCHL                    ;
                ;
                ;
                ; ***** PRBAD - PRINT ' WHAT ?' **** DESTROYS D&E ****
                ;
2E14 11CD2E     PRBAD:  LXI     D,BAD   ;
2E17 CDAB2D             CALL    MSG     ;
2E1A C9                 RET             ;
                ;       END     PRBAD
                ;
                ; RETJMP - RETURN JUMP
                ;
                ;       SETS STACK POINTER TO (RJSP) AND PC TO (RJVECT)
                ;       DOES NOT DESTROY ANY REGISTERS
                ;
2E1B 22Ø737     RETJMP: SHLD    RJSAV   ;
2E1E 2AØ937             LHLD    RJSP    ;
2E21 F9                 SPHL            ;
2E22 2AØ737             LHLD    RJSAV   ;
2E25 E5                 PUSH    H       ;
2E26 2AØB37             LHLD    RJVECT  ;
2E29 E3                 XTHL            ;
2E2A C9                 RET
                ;
                ; RNDHL - ADD CARRY FLAG TO H&L TO ROUND AFTER USING
                ;         SHRHL TO DIVIDE BY A POWER OF 2
                ;
2E2B F5         RNDHL:  PUSH    PSW             ;
2E2C 7D                 MOV     A,L             ;
2E2D CEØØ               ACI     Ø               ;ROUND
2E2F 6F                 MOV     L,A             ;
2E3Ø 7C                 MOV     A,H             ;PROPAGATE POSSIBLE ROUND-UP
2E31 CEØØ               ACI     Ø               ;  CARRY INTO H.
2E33 67                 MOV     H,A             ;
2E34 F1                 POP     PSW             ;
2E35 C9                 RET                     ;
                ;
                ;
                ; SHRHL - SHIFT RIGHT H&L - ZERO FILL ON LEFT
                ;         SHIFTS (B) BITS
                ;
                ;       ONLY H&L AND FLAGS CHANGED.
                ;       ON RETURN CARRY FLAG IS LAST BIT SHIFTED OUT
                ;       RIGHT END.
                ;
2E36 C5         SHRHL:  PUSH    B       ;SAVE B
2E37 F5                 PUSH    PSW     ;SAVE A
2E38 Ø4                 INR     B       ;CHECK FOR NO MORE BITS TO SHIFT
```

```
2E39 Ø5        SHRHLL: DCR    B       ;
2E3A CA472E            JZ     SHRHLE  ;
2E3D B7                ORA    A       ;CLEAR CARRY FLAG
2E3E 7C                MOV    A,H     ;GET H
2E3F 1F                RAR            ;SHIFT RIGHT
2E4Ø 67                MOV    H,A     ;PUT H BACK
2E41 7D                MOV    A,L     ;GET L
2E42 1F                RAR            ;SHIFT RIGHT
2E43 6F                MOV    L,A     ;PUT L BACK
2E44 C3392E            JMP    SHRHLL  ;BACK...
2E47 C1        SHRHLE: POP    B       ;RESTORE A
2E48 78                MOV    A,B     ;  .
2E49 C1                POP    B       ;RESTORE B
2E4A C9                RET            ;BYE...
               ;
               ;       END    SHRHL   ;
               ;
               ;
               ; SETJMP - SET SP AND PC FOR RETJMP
               ;          DOES NOT DESTROY ANY REGISTERS
               ;
2E4B E5        SETJMP: PUSH   H       ;
2E4C 21Ø4ØØ            LXI    H,Ø4    ;GET SP BEFORE PUSH H AND RET ADDR
2E4F 39                DAD    SP      ;
2E5Ø 22Ø937            SHLD   RJSP    ;
2E53 E1                POP    H       ;GET H&L BACK
2E54 E3                XTHL           ;GET RET ADDR
2E55 22ØB37            SHLD   RJVECT  ;SQUIREL AWAY
2E58 E3                XTHL           ;PUT RET ADDR BACK
2E59 C9                RET            ;
               ;
               ;
               ; ***** SPACE ***** PRINT SPACE
               ;
2E5A F5        SPACE:  PUSH   PSW
2E5B 3E2Ø              MVI    A,' '
2E5D CDD32B            CALL   CO
2E6Ø F1                POP    PSW
2E61 C9                RET
               ;
               ;
               ; STACKI - INITIALIZE STACK POINTER TO HIGHEST MEMORY LOCATION
               ;
2E62 D1        STACKI: POP    D       ;GET RETURN ADDRESS
2E63 F5                PUSH   PSW     ;SAVE PSW
2E64 21ØØØØ            LXI    H,Ø     ;START LOOKING AT ØFFFFH
2E67 2B        STKI1:  DCX    H       ;TRY NEXT LOWER LOCATION
2E68 7E                MOV    A,M     ;GET CONTENTS
2E69 2F                CMA            ;COMPLEMENT AND WRITE BACK
2E6A 77                MOV    M,A     ;
2E6B BE                CMP    M       ;SEE IF IT REALLY CHANGED
2E6C C2672E            JNZ    STKI1   ;NOPE - STILL WOM
2E6F 23                INX    H       ;EUREKA   RAM
2E7Ø F1                POP    PSW     ;GET PSW BACK...
2E71 F9                SPHL           ;SET STACK POINTER
2E72 EB                XCHG           ;GET RETURN ADDRESS
```

```
2E73 E9                 PCHL            ;RETURN
               ;
               ;        END     STACKI  ;
               ;
               ;
               ; ***** SUB16 ***** 16 BIT SUBTRACT (H&L) <- (H&L) - (D&E)
               ;
               ;        IF      (D&E)  < (H&L)     CY = 1
               ;        IF      (D&E) >= (H&L)     CY = Ø
               ;
2E74 D5        SUB16:   PUSH    D       ;
2E75 F5                 PUSH    PSW     ;
2E76 7D                 MOV     A,L     ;
2E77 93                 SUB     E       ;
2E78 6F                 MOV     L,A     ;
2E79 7C                 MOV     A,H     ;
2E7A 9A                 SBB     D       ;
2E7B 67                 MOV     H,A     ;
2E7C D1                 POP     D       ;
2E7D 7A                 MOV     A,D     ;
2E7E D1                 POP     D       ;
2E7F C9                 RET             ;
               ;
               ; UCASE - SUBROUTINE WHICH CHECKS THE A REG FOR A LOWER CASE
               ; ASCII LETTER. IF ONE PRESENT, IT IS CONVERTED TO UPPER CASE.
               ; IF NOT PRESENT, NOTHING DONE.  STRIPS PARITY FIRST.
2E8Ø E67F      UCASE:   ANI     Ø7FH    ;STRIP PARITY
2E82 FE61               CPI     61H
2E84 3F                 CMC
2E85 DØ                 RNC             ;DON'T CONVERT IF BEFORE 'A'
2E86 FE7B               CPI     7BH
2E88 DØ                 RNC             ;DON'T CONVERT IF AFTER 'Z'
2E89 D62Ø               SUI     2ØH     ;CONVERT LOWER TO UPPER
2E8B C9                 RET
               ;
               ; ROM CONSTANT ALLOCATION - ALPHABETICAL ORDER (SORTOF)
               ;                         - FUNCTIONAL ORDER TOO
               ;
               ; COMMAND TABLE
               ;
2E8C 444D      CMDS:    DB      'DM'            ;DUMP MEMORY
2E8E 7424               DW      DUMP            ;
2E9Ø 444C               DB      'DL'            ;DOWN LOAD
2E92 D723               DW      LOADER          ;
2E94 454D               DB      'EM'            ;EDIT MEMORY
2E96 2B23               DW      MEMED           ;
2E98 454B               DB      'EK'            ;ENABLE LIGHT BOARD
2E9A 932A               DW      INPAD           ;
2E9C 474F               DB      'GO'            ;GO
2E9E 5A22               DW      GOTO            ;
2EAØ 4845               DB      'HE'            ;HELP COMMAND
2EA2 5322               DW      HELP            ;
2EA4 494F               DB      'IO'            ;IO PORT R/W/M
2EA6 E424               DW      IOPORT          ;
2EA8 5442               DB      'TB'            ;TEST TIMERS AND PORTS ON BOARD
2EAA F122               DW      TSTBRD          ;
```

```
2EAC 544D               DB      'TM'            ;TEST MEMORY
2EAE 7E22               DW      MEMTST          ;
2EBØ 5243               DB      'RC'            ;RUN WHEELCHAIR
2EB2 1328               DW      RUNCHR          ;
2EB4 524D               DB      'RM'            ;RUN MENU
2EB6 3737               DW      RUNMU           ;
2EB8 5349               DB      'SI'            ;STACKPOINTER INITIALIZATION
2EBA 622E               DW      STACKI          ;
2EBC 5Ø43               DB      'PC'            ;CHECK LIGHT PAD
2EBE 182A               DW      PADCK           ;
2ECØ ØØØØ               DB      Ø,Ø             ;END OF TABLE MARK
                ;
                ; MESSAGES...
                ;
2EC2 2Ø41424F52ABORT:   DB      ' ABORTED  '
2ECC FF                 DB      EOL
2ECD 2Ø57484154BAD:     DB      ' WHAT ?'
2ED4 FF                 DB      EOL
2ED5 29         EDM1:   DB      ')'
2ED6 2Ø3D2Ø     EDM3:   DB      ' = '
2ED9 FF                 DB      EOL
2EDA ØDØA       EDM2:   DB      CR,LF
2EDC 28                 DB      '('
2EDD FF                 DB      EOL
2EDE ØDØA       MTSBRD  DB      CR,LF
2EEØ 5445535449         DB      'TESTING TIMERS AND PIA PORTS',CR,LF
2EFE 4C4F4F4B2Ø         DB      'LOOK FOR 1ØØØ HZ SQUAREWAVE ON TIMER OUTPUTS',CR,LF
2F2C 444154412Ø         DB      'DATA ANALIZER SHOULD SHOW PORTS COUNTING',CR,LF
2F56 494E2Ø412Ø         DB      'IN A STAIRSTEP FASTION',CR,LF
2F6E FF                 DB      EOL
2F6F 444154413DIOPDA:   DB      'DATA= '
2F75 FF                 DB      EOL
2F76 4Ø2Ø35306DIOPMM:   DB      '@ 5ØmS # '
2F7F FF                 DB      EOL
2F8Ø 2C2Ø2Ø     IOPSM:  DB      ',  '
2F83 FF                 DB      EOL
2F84 ØDØA       GCLKM:  DB      CR,LF
2F86 454E544552         DB      'ENTER CPU CLK FREQ XXXX KHZ: '
2FA3 FF                 DB      EOL
2FA4 2Ø4F4B2Ø3FMOK:     DB      ' OK ?'
2FA9 FF                 DB      EOL
2FAA ØDØA       MTGOOD: DB      CR,LF
2FAC 4D454D4F52         DB      'MEMORY TEST PASSED'
2FBE ØDØAFF             DB      CR,LF,EOL
2FC1 ØDØA       MTERR:  DB      CR,LF
2FC3 4D454D4F52         DB      'MEMORY TEST FAILED AT '
2FD9 FF                 DB      EOL
2FDA 3A2Ø57524FMTWROT:  DB      ': WROTE '
2FE2 FF                 DB      EOL
2FE3 2C2Ø524541MTREAD:  DB      ', READ '
2FEA FF                 DB      EOL
2FEB 2Ø544F2Ø   PHI:    DB      ' TO '
2FEF FF                 DB      EOL
2FFØ 2Ø46524F4DPLO:     DB      ' FROM '
2FF6 FF                 DB      EOL
2FF7 4F46465345PBIAS:   DB      'OFFSET VALUE ? '
```

```
3006 FF                DB      EOL
3007 0D0A0A    PHELP:  DB      CR,LF,LF
300A 2054484520        DB      ' THE FOLLOWING TWO CHARACTER COMMANDS'
302F 0D0A              DB      CR,LF
3031 2020202020        DB      '          ARE AVAILIBLE : '
304B 0D0A0D0A          DB      CR,LF,CR,LF
304F 444D202044        DB      'DM  Dump Memory'
305E 0D0A              DB      CR,LF
3060 444C202044        DB      'DL  Down Load from dev. system'
307E 0D0A              DB      CR,LF
3080 454D202045        DB      'EM  Edit Memory'
308F 0D0A              DB      CR,LF
3091 454B202045        DB      'EK  Enable Keyboard'
30A4 0D0A              DB      CR,LF
30A6 474F202047        DB      'GO  GOto'
30AE 0D0A              DB      CR,LF
30B0 494F202049        DB      'IO  I/O port r/w/m'
30C2 0D0A              DB      CR,LF
30C4 5349202053        DB      'SI  Sp Init'
30CF 0D0A              DB      CR,LF
30D1 5442202054        DB      'TB  Test Board utitily'
30E7 0D0A              DB      CR,LF
30E9 544D202054        DB      'TM  Test Memory'
30F8 0D0A              DB      CR,LF
30FA 5243202052        DB      'RC  Run Chair program'
310F 0D0A              DB      CR,LF
3111 524D202052        DB      'RM  Run Menu select program'
312C 0D0A              DB      CR,LF
312E 5043202050        DB      'PC  Pad Check'
313B 0D0A              DB      CR,LF
313D 3F20202070        DB      '?   print answer'
314D 0D0A0A0A          DB      CR,LF,LF,LF
3151 5255422069        DB      'RUB interrupts, ^S/^Q turns output off/on'
317A 0D0A0A            DB      CR,LF,LF
317D FF                DB      EOL
317E 0D0A      PRMPT:  DB      CR,LF
3180 5245414459        DB      'READY'
3185 0D0A              DB      CR,LF
3187 203E              DB      ' >'
3189 FF                DB      EOL
318A 0D0A      START:  DB      CR,LF
318C 2A2A2A2A2A        DB      '*****************************************'
31B5 0D0A              DB      CR,LF
31B7 2A2A2A2020        DB      '***    EASY CHAIR CONTROLER V 2.0     ***'
31E0 0D0A              DB      CR,LF
31E2 2A2A2A2A2A        DB      '*****************************************'
320B 0D0A              DB      CR,LF
320D 0D0A              DB      CR,LF
320F 2020544849        DB      '  THIS PROGRAM OPERATES THE EASY CHAIR'
3235 0D0A              DB      CR,LF
3237 2020434F4E        DB      '  CONTROLER, ULTRASONICS, LIGHT BOARD,'
325D 0D0A              DB      CR,LF
325F 2020414E44        DB      '  AND MOTORS. THIS PROGRAM ALSO ALLOWS '
3286 0D0A              DB      CR,LF
3288 2020544845        DB      '  THE MENUS FOR THE LIGHT BOARD TO BE '
32AE 0D0A              DB      CR,LF
```

```
32BØ 2Ø2Ø2Ø2Ø2Ø         DB      '     ADDED TO AND CHANGED AS NEEDED.'
32D4 ØDØA               DB      CR,LF
32D6 ØDØA               DB      CR,LF
32D8 2Ø414C4C2Ø         DB      ' ALL ATTEMPTS WERE MADE TO FORESEE ALL'
32FE ØDØA               DB      CR,LF
33ØØ 2Ø5448452Ø         DB      ' THE POSSIBLE PROBLEMS THAT MAY ARISE,'
3326 ØDØA               DB      CR,LF
3328 2Ø2Ø2Ø2Ø2Ø         DB      '       HOWEVER, -NO- PROMISES.'
3346 ØDØA               DB      CR,LF
3348 ØDØAFF             DB      CR,LF,EOL
334B ØDØAFF    MCRLF:   DB      CR,LF,EOL
334E 535441434BSTKAT:   DB      'STACK AT '
3357 FF                 DB      EOL
3358 544841542ØMNOGO:   DB      'THAT PROGRAM IS CURRENTLY OFF-LINE'
337A ØDØA               DB      CR,LF
337C 2Ø2Ø2Ø2Ø2Ø         DB      '      IT WILL BE ADDED SOON'
3397 ØDØA               DB      CR,LF
3399 FF                 DB      EOL
339A ØAØAØAØAØACLS      DB      LF,LF,LF,LF,LF,LF,LF,LF,LF,LF,LF,LF,LF,LF
33A8 ØAØAØAØAØA         DB      LF,LF,LF,LF,LF,LF,LF,LF,LF,LF,LF,HOME,EOL
33B5 2Ø46524F4EFNTMSG   DB      ' FRONT = ',EOL
33BF 2Ø2Ø2Ø4241BAKMSG   DB      '   BACK = ',EOL
33CA 2Ø2Ø2Ø2Ø2ØRTMSG    DB      '     RIGHT = ',EOL
33D8 2Ø2Ø2Ø2Ø2ØLFTMSG   DB      '       LEFT = ',EOL
33E8 434C454152ALLCLR   DB      'CLEAR',EOL
33EE 44454C4159TIMQUE   DB      'DELAY BETWEEN SCANS ? ',EOL
34Ø5 4D41582Ø46FNTQUE   DB      'MAX FRONT DIST. ? ',EOL
3418 4D41582Ø42BAKQUE   DB      'MAX BACK DIST. ? ',EOL
342A 4D41582Ø52RTQUE    DB      'MAX RIGHT DIST. ? ',EOL
343D 4D41582Ø4CLFTQUE   DB      'MAX LEFT DIST. ? ',EOL
344F 4C45442F54ROWERR   DB      'LED/TRANSISTOR ERROR IN ROW (Ø-F): ',EOL
3473 4C45442F54COLERR   DB      'LED/TRANSISTOR ERROR IN COLUMN (Ø-F): ',EOL
349A 5Ø41442Ø54TCHMSG   DB      'PAD TOUCHED AT LOCATION: ',EOL
34B4 424547494EINTMSG   DB      'BEGIN INFRA-RED TOUCH PAD DIAGNOSTICS',EOL
34DA 454E44204FENDMSG   DB      'END OF INFRA-RED TOUCH PAD DIAGNOSTICS',EOL
35Ø1 444F55472Ø         DB      'DOUG HEINTZ ',EOL


                  ;
                  ; RAM ALLOCATION IN ALPHABETICAL AND FUNCTIONAL ORDER
                  ;
37ØØ                      ORG     MONRAM            ;BEGINNING OF MONITOR RAM
                  ;
37ØØ              COCOOK: DS      1                 ;Ø=COOKED, 1=RAW
37Ø1              DLYRAM: DS      1                 ;NUMBER OF 1ØMS DELAYS ON <CR>
37Ø2              ECHOFL: DS      1                 ;ECHO FLAG: Ø=ECHO 1=NO ECHO
37Ø3              WIDTH:  DS      1                 ;WIDTH+1 = NUMBER OF BYTES PER LINE
                  ;                                 ;FOR PUNCH AND DUMP
37Ø4              BIAS:   DS      2                 ;BIAS FOR PUNCH AND LOAD
37Ø6              VFYFLG: DS      1                 ;Ø=LOAD, 1=VERIFY (HEX TAPE)
                  ;                                 ;RANGE: ØØH TO 8ØH
37Ø7              RJSAV:  DS      2                 ;TEMP SAVE AREA FOR RETJMP
37Ø9              RJSP:   DS      2                 ;RETURN JUMP STACK POINTER
37ØB              RJVECT: DS      2                 ;RETURN JUMP VECTOR (PC)
37ØD              D5ØDIV: DS      2                 ;COUNTER FOR TIMING OF 5ØMS PULSE
37ØF              CLKBCD: DS      2                 ;CLOCK FREQUENCY IN BCD
3711              CLKBIN: DS      2                 ;CLOCK FREQUENCY IN BINARY
```

```
3713           FNTDST: DS      2              ;ULTRASONIC FNT DIST.
3715           MAXFNT: DS      1              ; MAX FRONT DIST.
3716           BAKDST: DS      2              ;          BACK DIST.
3718           MAXBAK: DS      1              ; MAX BACK DIST.
3719           RTDST:  DS      2              ;         RIGHT DIST.
371B           MAXRT:  DS      1              ; MAX RIGHT DIST.
371C           LFTDST: DS      2              ;          LEFT DIST.
371E           MAXLFT: DS      1              ; MAX LEFT DIST.
371F           TIMDLY: DS      2              ;DELAY TIME
3721           SONOFF: DS      1              ;SOUND FLAG
3722           RONOFF: DS      1              ;RANGING FLAG
3723           HONOFF: DS      1              ;HIGH SPEED FLAG
3724           RAMP:   DS      1              ;RAMP RATE
3725           FLGERR  DS      1              ; ERROR LED FLAG
3726           MISCBF: DS      17             ;BUFFER FOR USE BY COMMANDS
               ;                              ;PUT LAST SO AN OVERRUN WON'T BOMB
               ;                              ;SYSTEM
                               ;END OF MONITOR

3737 00        RUNMU:  NOP
3738 115833            LXI     D,MNOGO        ;SUB NOT AVAL. MESSAGE
373B CDAB2D            CALL    MSG
373E C9                RET

               ;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++
373F                   END

A>
```

E.E.T. 490/491 SENIOR DESIGN PROJECT

THE EASY CHAIR

APPENDIX C: COSTING

INFRARED TOUCH-PAD	
40 - Infrared LEDS	24.00
40 - Infrared phototransistors	22.00
1 - Miscellaneous wood/plastic	30.00
1 - Electronic components	55.00
1 - Electronic cable	17.00
1 - Miscellaneous hardware	65.00
	213.00
ULTRASONIC RANGING	
4 - Ultrasonic transducers	300.00
1 - Electronic components	30.00
1 - Electronic cable	12.00
	342.00
COMPUTER AND MOTOR CONTROL	
1 - Working 8085 based computer	300.00
1 - Additional 8255 PIA	7.00
1 - 2816A EEPROM	16.00
2 - AD558 D/A Converters	15.00
1 - Electronic components	15.00
1 - Power supply components	10.00
	363.00
MISCELLANEOUS COSTS	
1 - Shipping and handling charges	60.00
1 - Phone calls (parts and consulting)	75.00
	TOTAL $ 1053.00

E.E.T. 490/491 SENIOR DESIGN PROJECT

THE EASY CHAIR

APPENDIX D: BIBLIOGRAPHY

---

1) Lotto W., Milner M., "Evaluations and Development Of Powered Mobility Aids For Two-To-Five Year Olds With Neuromusculoskeletal Disorders", Ontario Crippled Children's Center, 1984

2) Jaffe, David L., "Polaroid Ultrasonic Ranging Sensors In Robotic Applications", Robotics Age, March, 1985

3) Jaffe, David L., "A Design/Development Methodology For Rehabilitation Devices Using Embedded Microcomputers", Rehabilatation Research and Development Center, Palo Alto Veterans Administration Medical Center, 1983

4) Mims, Forrest M., "Making Your Own Pressure-Sensitive Resistors", Computers and Electronics, 1983

5) Mims, Forrest M., "Ultrasonic Sound Polaroid Rangefinder, LM3905 Ap Note Lower Supply Voltages Device Developments", Computer and Electronics, June, 1983

6) Byers, T.J., "Keyboards: The Power At Your Fingertips", Computers and Electronics, September, 1984

7) Welch, Gregory F., Williams, James P., "The Pressure Sensitive Touch-Pad", Purdue Universtiy, school of Electrical Engineering Technology, 1985

8) Jaffe, David L., "Ultrasonic Head Control Unit", Rehabilatation Research and Development Center, Palo Alto Veterans Administration Medical Center, 1983

9) Ciarcias, "An Ultrasonic Ranging System", Byte Magazine, October, 1984