A MULTICHANNEL, GENERAL-PURPOSE DATA LOGGER

MICHAEL EDWIN GARDENER
**This book is due for return on the last date.**
Hierdie boek moet op die laaste datum terugbesorg word.

<table>
<thead>
<tr>
<th>Return Date</th>
<th>Form 40</th>
</tr>
</thead>
<tbody>
<tr>
<td>1987, 11, 10</td>
<td></td>
</tr>
<tr>
<td>11 -04- 1988</td>
<td></td>
</tr>
<tr>
<td>22 -11- 1988</td>
<td></td>
</tr>
<tr>
<td>28 FEB 1990</td>
<td></td>
</tr>
</tbody>
</table>
A MULTICHANNEL, GENERAL-PURPOSE DATA LOGGER

by

MICHAEL EDWIN GARDENER

Thesis submitted in partial fulfilment of the requirements for the Diploma in Technology to the Department of Electrical Engineering (light current) at the Cape Technikon.

Institute for Maritime Technology
Part of the Armscor Organisation

Simon's Town
South Africa
January, 1986
ABSTRACT

This thesis describes the implementation of a general-purpose, microprocessor-based Data Logger.

The Hardware allows analog data acquisition from one to thirty two channels with 12 bit resolution and at a data throughput of up to 2KHz.

The data is logged directly to a Buffer memory and from there, at the end of each log, it is dumped to an integral cassette data recorder. The recorded data can be transferred from the logger to a desk-top computer, via the IEEE 488 port, for further processing and display.

All log parameters are user selectable by means of menu prompted keyboard entry and a Real-Time clock (RTC) provides date and time information automatically.
I would like to thank my supervisor, Mr C Rudolph, and Mr D Meyer for their encouragement and assistance during the course of this project.

I would especially like to thank Mr S Koster for his patience and friendly advice in helping me come to grips with the microprocessor development tasks as well as acting as a sounding-board for my design ideas.

I would like to acknowledge the circuits designed by my colleagues Mr S Koster and Mr G de Waal used in this project.

I would also like to thank the various members of the Institute for Maritime Technology who assisted me in the preparation of this document with special thanks to Miss T Miles for her word processing effort.

I am indebted to the Institute for Maritime Technology for the financial and material support extended to me during my Diploma in Technology Studies.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. INTRODUCTION</td>
<td>7</td>
</tr>
<tr>
<td>2. DESIGN CRITERIA</td>
<td>8</td>
</tr>
<tr>
<td>3. SYSTEM DESCRIPTION</td>
<td>10</td>
</tr>
<tr>
<td>3.1. Keyboard</td>
<td>11</td>
</tr>
<tr>
<td>3.2. Display</td>
<td>11</td>
</tr>
<tr>
<td>3.3. Real-Time Clock</td>
<td>11</td>
</tr>
<tr>
<td>3.4. Data Acquisition</td>
<td>12</td>
</tr>
<tr>
<td>3.5. Buffer Memory</td>
<td>13</td>
</tr>
<tr>
<td>3.6. IEEE-488 Interface</td>
<td>13</td>
</tr>
<tr>
<td>3.7. Digital Cassette Recorder</td>
<td>13</td>
</tr>
<tr>
<td>3.8. Power Supply</td>
<td>14</td>
</tr>
<tr>
<td>3.9. Controller</td>
<td>14</td>
</tr>
<tr>
<td>4. SYSTEM SPECIFICATIONS</td>
<td>17</td>
</tr>
<tr>
<td>5. SYSTEM SOFTWARE</td>
<td>18</td>
</tr>
<tr>
<td>5.1. BIN$ASCII $ASCII$BIN MODULE</td>
<td>19</td>
</tr>
<tr>
<td>5.2. SET$LOG$PARAMETER MODULE</td>
<td>19</td>
</tr>
<tr>
<td>5.3. DATA$LOGGER MODULE</td>
<td>21</td>
</tr>
<tr>
<td>6. THEORY OF OPERATION</td>
<td>23</td>
</tr>
<tr>
<td>6.1. IMT 8085 CPU BOARD</td>
<td>23</td>
</tr>
<tr>
<td>6.1.1. General</td>
<td>23</td>
</tr>
<tr>
<td>6.1.2. Features</td>
<td>24</td>
</tr>
<tr>
<td>6.1.3. Hardware</td>
<td>24</td>
</tr>
<tr>
<td>6.1.4. Memory</td>
<td>28</td>
</tr>
<tr>
<td>6.1.5. Parallel I/O</td>
<td>28</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>6.1.6. Display</td>
<td>29</td>
</tr>
<tr>
<td>6.1.6.1. Hardware</td>
<td></td>
</tr>
<tr>
<td>6.1.6.2. Software – DISPLAY$MODULE</td>
<td></td>
</tr>
<tr>
<td>6.1.7. Keyboard</td>
<td>32</td>
</tr>
<tr>
<td>6.1.7.1. General</td>
<td></td>
</tr>
<tr>
<td>6.1.7.2. Hardware</td>
<td></td>
</tr>
<tr>
<td>6.1.7.3. Software – KEY$BOARD$MODULE</td>
<td></td>
</tr>
<tr>
<td>6.1.8. Real-Time Clock</td>
<td>34</td>
</tr>
<tr>
<td>6.1.8.1. General</td>
<td></td>
</tr>
<tr>
<td>6.1.8.2. Hardware</td>
<td></td>
</tr>
<tr>
<td>6.1.8.3. Software – REAL$TIME$CLOCK$MODULE</td>
<td></td>
</tr>
<tr>
<td>6.1.9. Specifications</td>
<td>36</td>
</tr>
<tr>
<td>6.2. IMT DATA ACQUISITION BOARD</td>
<td>37</td>
</tr>
<tr>
<td>6.2.1. General</td>
<td>37</td>
</tr>
<tr>
<td>6.2.2. Data Throughput</td>
<td>38</td>
</tr>
<tr>
<td>6.2.3. Printed Circuit Layout and Power Supply Requirements</td>
<td>39</td>
</tr>
<tr>
<td>6.2.4. Hardware</td>
<td>40</td>
</tr>
<tr>
<td>6.2.5. Software – D$ACQ$MODULE</td>
<td>41</td>
</tr>
<tr>
<td>6.3. IMT MFE TAPE BOARD</td>
<td>43</td>
</tr>
<tr>
<td>6.3.1. General</td>
<td>43</td>
</tr>
<tr>
<td>6.3.2. Hardware</td>
<td>44</td>
</tr>
<tr>
<td>6.3.3. Software – TAPE$MODULE</td>
<td>44</td>
</tr>
<tr>
<td>6.4. IMT IEEE-488 BOARD</td>
<td>49</td>
</tr>
<tr>
<td>6.4.1. General</td>
<td>49</td>
</tr>
<tr>
<td>6.4.2. Hardware</td>
<td>49</td>
</tr>
<tr>
<td>6.4.3. Software – IEEE$488$MODULE</td>
<td>52</td>
</tr>
</tbody>
</table>
6.5. Microcom 32K Memory Board                                  53
6.6. Data Logger P.S.U.                                         53

7. OPERATING INSTRUCTIONS                                       54

7.1. Initial Power up                                           56
7.2. Set Date and Time                                         56
7.3. Set Logger Parameters                                     57
7.4. Data Acquisition Test and Calibration                     60
7.5. IEEE-488 Port Test                                        62
7.6. Cassette Loading in Record Mode                           62
7.7. Tape Transport Prompts                                    63
7.8. Data Playback                                             63

8. SERVICING                                                   65

9. SOFTWARE PROBLEMS AND ENHANCEMENTS                          66

10. CONCLUSION                                                 70

11. REFERENCES                                                 71

APPENDIX A: Version V 1.2 software listings
APPENDIX B: Version V 1.3 software listings
APPENDIX C: Version V 1.4 software listings
APPENDIX D: Data Sheets
APPENDIX E: Data Processing by HP85
APPENDIX F: Tape Header and Data Format
APPENDIX G: Component Lists, Component Overlays
            and Schematic Diagrams
1. INTRODUCTION

As part of an ongoing data gathering exercise situated in Durban, a requirement arose for a self-activating, stand-alone, Data Logger.

A system specification for the logger was drawn up and a survey of commercially available equipment established that none would meet the particular needs of this application. It was therefore decided to build a data logger that would meet the specifications; the design and development of which forms the basis of this thesis.

The main objective of this thesis is to provide a comprehensive document on the Data Logger that will enable the potential user to understand its theory of operation, design limitations and operating instructions. Enough information is included to allow repairs and modifications to be carried out when required.
2. DESIGN CRITERIA

The following prominent criteria were taken into consideration during the design stage of the Data Logger.

1. The design should be based on the Intel 8085 microprocessor family as it is well supported at the Institute for Maritime Technology (IMT) with regard to development system, high-level language (PL/M) and in circuit emulator.

2. The logger should be designed on a modular basis to allow for future expansion or modification and to be easily adaptable to other IMT requirements.

3. The Data Logger should be of a general-purpose, as opposed to dedicated, design to ensure flexibility in operating role.

4. In keeping with the modular concept it was decided that the system should be implemented using the S.A.Bus standard.

The S.A.Bus is a South African microprocessor bus standard that was first formulated in 1978 and has the following objectives.

1. Uniform card size and edge connectors.
2. Interchangeability of system modules between various organizations.
4. Access to relevant information concerning system modules.

It also has the following advantages.

1. The S.A. bus favours the 8085 microprocessor family.
2. Most boards are fully buffered and only require a single +5V supply.
3. There are at least two commercial concerns that support the S.A. bus with a selection of off-the-shelf boards.

5. The design should incorporate the M.F.E. Digital Cassette Tape Transport which was available at IMT.
6. A tape read-back unit would be required to recover the logged data from the digital cassettes. It was decided that this function should be incorporated into the Data Logger. Because the required Data Logger would be remotely sited and in continuous use, a second identical instrument should be constructed which would fulfil several useful roles:

1. A back up system for the remotely sited unit.
2. A unit on which to develop or modify the system hardware/software.
3. A useful instrument for general-purpose data logging at IMT.
4. A tape read-back unit.

7. Because at least two units were required it was decided to make use of printed circuit boards (p.c.b's) for all in-house electronics.

8. The system was required to be easy to operate and therefore the design should incorporate a menu-driven operating system with an interactive intelligent display to facilitate entry of system log parameters.

9. A real-time read out of selected data was also required. This was to be implemented in two ways:

1. A read out of a single logged channel to be displayed on an integral display should be provided.
2. An interface should be provided to allow data to be displayed on a HP-85 computer in real-time.

10. Battery back up of system parameters as well as clock/calander data should be incorporated so that in the event of a mains power interruption the system could perform a warm-start.
3. SYSTEM DESCRIPTION

The function of the Data Logger is to sample analog signals at a pre-determined scan rate; convert the analog signals to digital data and store this data in a buffer memory. This process is initiated by certain trigger conditions being met and when the required number of scans has been completed the data is dumped from the buffer onto digital cassette tape.

The logger also has a data retrieval or play back mode to allow data stored on tape to be transferred, via the IEEE 488 port, to a computer for processing.

The Data Logger comprises of nine functional blocks (Fig 1):

---

FIG. 1

DATA LOGGER FUNCTIONAL BLOCK DIAGRAM
3.1. Keyboard:

The Keyboard is part of the user/instrument interface and consists of 16 keys which can be divided into three groups.

1. Numeric keys:

There are ten keys which allow the numbers 0 – 9 to be entered as variables.

2. Control keys:

There are three control keys:

1. [<-] and [->] which allow the cursor position on the display to be shifted left or right or parameters to be decremented or incremented i.e. channel number.
2. [E] allows the selected parameters to be entered into the program.

3. Soft-keys

There are three soft-keys that have functions which are uniquely defined by the current program and change under program control. The actual function that a particular soft-key will perform at any time is labelled on the lower line of the display.

3.2. Display

The intelligent Liquid Crystal Display (L.C.D.) provides the other half of the user/instrument interface. The device can display two lines of sixteen characters each. Data is usually displayed on the top line with the soft key labels on the lower line.

3.3. Real-Time Clock

The real-time clock (RTC) provides the date and time data for the log. It also provides an alarm clock type interrupt for triggering a log and a periodic interrupt which governs the logger scan rate.
The scan rate is user selectable via the keyboard by entering a number between 1 and 15. This represents scan rates from 30,517 $\mu$ sec to 500 m sec with the scan rate doubling as the number is decremented by one. ($15 = 500\text{m sec}, 14 = 250\text{m sec}$ etc.)

The maximum scan rate is governed by the data acquisition hardware and software, and in practice is limited to scan rate 10 (15,625 m sec). This means that each of 32 channels can be scanned once every 15,625 m sec and the data for one selected channel displayed on the LCD. The clock chip also provides 50 bytes of battery backed up RAM which is used to store the log parameters. This enables the logger to recover from a mains power interruption with all the log parameters intact and with the correct date and time.

3.4. Data Acquisition

The Data Acquisition board is capable of servicing 32 analog signals on a multiplexed basis.

When a log is initiated each channel, starting with channel #1, is connected in turn via a Sample and Hold amplifier (S & H amp.) to the Analog to Digital Converter (A/D). The A/D converter takes 35 $\mu$ secs to convert the analog signal to a 12 bit digital word. These 12 bits are stored in the Buffer memory as two consecutive 8 bit bytes. (First 8 msb's, then 4 lsb's with 4 trailing zeros.)

The Multiplexer (MUX) is then incremented, by one, to the next channel and the process repeated until all the selected channels have been serviced. The program then waits for a scan interrupt from the RTC to begin the next scan. The program keeps scanning until the required number of scans to complete the log is achieved. The maximum number of scans is governed by the buffer memory size, which is 16K in this instance, and can be determined by

$$\text{Number of Scans}_{(\text{max})} \leq \frac{16384}{(2 \times \text{Number of channels})}.$$
3.5. Buffer Memory

The buffer memory consists of 16k x 8 bit CMOS static RAM. It is used as a means of buffering the data before transferring the log onto digital cassette. This is necessary to increase the storage capacity of the tape as well as making the logger independent of the Digital Cassette transports data transfer rate.

3.6. IEEE-488 Interface

The IEEE-488 interface provides a convenient means of linking the logger to a computer or other instrumentation. Specifically, it can be used to allow the logged data for one pre-selected channel to be displayed on a HP85 computer in real-time. It is also used to transport the data from the digital cassette to the HP85 or HP9816 for processing. (see Appendix E)

For Data Format see Appendix F

3.7. Digital Cassette Recorder

The Digital Cassette recorder is used for the mass storage of logged data. It has three modes of operation selectable from the keyboard:

1. Record Mode

This is the default mode and is selected whenever it is required to save data on the cassette tape. In this mode the logged data stored in the Buffer memory is dumped to tape at the end of each log. This data is preceded by header data which contains the following information. Date, Time, Scan Rate, Number of channels, Number of Scans. The header and data combined, form one record.

(see Appendix F)

2. Play Mode

This mode allows data to be retrieved from the data cassette. When this mode is initially selected the cassette is rewound and then positioned at the beginning of the first record (record #1). The program then allows the operator to select any record either by using the [-], [+] keys to increment or decrement the record number or via the [ENTER] soft-key. When the desired record number is displayed the [READ] soft-key allows the program to find the record and read it into the Buffer memory. If the IEEE ON mode has been selected then data will be read out of the Buffer to the IEEE-488 bus under IEEE listener control.
3. Off Mode

This mode ensures that no data dump takes place at the end of a log and the data is therefore lost.

3.8. Power Supply

This unit converts 220V AC mains to +5V @ 3A, ±15V @ 85mA and -5V @ 500mA. Because of the battery back-up feature a power-cut of several hours can be tolerated although the Data Logger will not be operational during this time.

3.9. Controller

The controller manages all the above functions using the software to determine the course of events.

```
START

INITIALIZE

MAIN

MODE CHANGE?

YES

ALARM FLAG SET?

YES

DISPLAY DATE & TIME

NO

NO

SET MODE

LOG MODE

TEST MODE

EXTERNAL TRIG. MODE

ALARM TRIG. MODE

KEY = 1 2

3 4

FIG. 2

SIMPLIFIED DATA LOGGER FLOW DIAGRAM
```
Fig 2 shows a simplified flow diagram of the Data Logger program.

The program first initializes all the hardware modules. It then enters the loop MAIN which checks to see whether a soft-key depression or external trigger contact closure has occurred. If so the program will change mode to service the particular need before returning to the beginning of MAIN. These modes are:

1. Set Mode

This mode allows either Date and Time, or Log Parameters to be set. The log parameters include:

1. IEEE mode selection {ON or OFF}
2. TAPE mode selection {Record, Play or OFF}
3. Trigger mode selection {Alarm, Manual or External}
4. Scan Rate
5. Number of Channels
6. Number of Scans

2. Log Mode

The response to this selection depends on the Trigger mode selected above:

1. Alarm Mode
   In this mode the selected log START TIME is displayed when the [LOG-A] soft-key is depressed.

2. Manual Mode
   The [LOG-M] soft-key starts a log immediately.

3. External Mode
   In this mode, if [LOG-X] is depressed, the user is asked if he wishes to override the external mode and if the response is yes, a log is started.

3. Test Mode

This mode allows either the IEEE Interface or the Data Acquisition board to be tested.
1. IEEE-Test

If selected all the values from 0 - 255 are output consecutively to the IEEE bus. An intelligent listener on the bus could use this data to ascertain whether the IEEE interface is functioning correctly or not. (see Appendix E)

2. Data acquisition Test

This function allows the Data Logger to be used as a digital volt meter. It enables input signals to be monitored or set up as each channel is individually selectable via the keyboard and displayed on the LCD.

4. External Trigger Mode

If either key [E] or the external trigger contacts are made a log is started. The contacts are connected in parallel with key [E] and must be closed for least 100m sec to be accepted as valid.

If the Alarm Trigger mode has been selected then the alarm flag is checked and if it is set then a log is started. Lastly the current date and time are output to the LCD display before the program loops back to the beginning of MAIN. This loop is executed ad infinitum.
4. SYSTEM SPECIFICATIONS

* Number of input channels: 1 to 32.

* Scan Rate: 2 scans / sec minimum
  64 scans / sec maximum (see text)

* Number of Scans: 1 to 9999

* Log Activation: External contact (X)
  Manual (M)
  Alarm (A)

Analog input: +5V bipolar (10Vp to p)

Resolution: 12 bit A/D (2.44 mV per bit)

Data Buffer: 16k (expandable to 40k)

Data Storage: Integral M.F.E. Digital Cassette Recorder
  (676k bytes both sides)

Data Replay: Transfer from Digital recorder to IEEE-488 interface port.

Date and Time: From battery backed up Real-Time Clock.

Operator Interface: 16 Key matrix keyboard.
  2 line by 16 character intelligent alphanumeric liquid crystal display.

Power Supply: 220V A.C.

* These parameters are set via the keyboard.
5. SYSTEM SOFTWARE

At the outset of the project a choice had to be made between Assembler and PL/M as the medium for software development. The high-level language PL/M was chosen because it encourages and in fact enforces good software engineering practices like structured programming, modular program design and top-down design implementation. It also improves programmer productivity and is more reliable, maintainable and portable than Assembler. It can also serve as documentation because it is easy and logical to follow.

Assembler, on the other hand, has few of these advantages but could make more efficient use of the hardware capabilities and perhaps reduce code size. It requires the programmer to have above average programming skills and experience.

The Data Logger system software comprises of 9 PL/M modules:

1. KEYBOARD MODULE
2. DISPLAY MODULE
3. REAL TIME CLOCK MODULE
4. DACQ MODULE
5. IEEE 488 MODULE
6. TAPE MODULE
7. BIN ASCII ASCII BIN MODULE
8. SET LOG PARAMETER MODULE
9. DATA LOGGER MODULE (The main module)

Each module is a collection of procedures with closely related program functions. I.e. DISPLAY MODULE holds all the procedures pertaining to the control of the display hardware.

The first 6 modules hold utility programs which are specific to the hardware they service and they will therefore be discussed along with the hardware under the heading Theory of Operation.

The last 3 modules will be discussed below.
5.1. **BIN$ASCII$ASCII$BIN MODULE:**

This module consists of two procedures used to convert an ASCII string to a binary word and visa versa.

**BIN$TO$ASCII:** Public

Converts a 16 bit binary word to four ASCII numeric characters and returns to the calling program with them in the string variable CHAR.

**USE:**

```
CALL BIN$ TO $ASCII (READING, SIZE).
```

Where

READING is a 16 bit binary word and SIZE is 0 TO 4.

If SIZE=2 then the string CHAR is left justified as follows: 0023 becomes 2323.

**ASCII$ TO$BIN:** Public

In this procedure the number of ASCII numeric characters indicated by SIZE are converted to a 16 bit binary word and stored in the variable BIN$DATA$. This value can be accessed by the calling program.

**Use**

```
CALL ASCII$TO$BIN (ASCII$ADDRESS, SIZE)
```

where

ASCII$ADDRESS$ holds the starting address of an ASCII string and SIZE is the number of characters to be converted. (limits 1 - 4)

5.2. **SET $ LOG $ PARAMETERS $ MODULE:**

This module allows the log parameters to be entered into the program via menu prompted interactive keyboard entry.

**LOG$PARAM$2:**

Outputs the first two characters of the string CHAR to the LCD and positions the cursor beneath the most significant digit (m.s.d.)

**LOG$PARAM$4:**

Same as LOG$PARAM$2 but for 4 characters.
ALARM$DISP:

Sets up the LCD to display the Alarm time.

X$ OVERRIDE:

This procedure sets up the LCD to display a question and a menu for the external trigger override function.

LOG$TRIG:

Is entered whenever the [LOG-?] soft key is depressed. It responds differently depending on which choice of trigger mode has previously been selected. i.e. depends on the value of the variable TRIG$TYPE.

If the Alarm mode was selected the actual alarm time is displayed on the LCD until either the alarm interrupt occurs or the [EXIT] soft-key is depressed.

If the External trigger mode was selected then the procedure X$OVERRIDE is called. This displays "X-TRIG OVERRIDE?" on line 1 and "NO  YES  EXIT" on line 2 of the LCD. Depending on the operators response, either a log will be initiated or the program will return to the calling program.

If the Manual trigger was selected then a log is initiated.

CHAN$DISP$SELECT:

Allows the channel to be displayed or output to the IEEE port to be selected.

SET$LOG$PARAMETERS:

Supplies the prompts and menus that allow the log parameters to be entered into the program. In order, they are:

1   Number of channels.
2   Scan Rate.
3   Number of Scans.
4   Channel to be displayed.
5   Trigger mode.
5.3. \texttt{DATA$LOGGER$MODULE}:

This is the main module and consists of the following procedures:

**TEST:**

Outputs the message "TEST MODE" and the menu D-ACQ IEEE EXIT". It then waits for one of the soft-keys to be depressed before calling the required procedure or exiting.

For "D-ACQ" response see DACQ$TEST in DACQ$MODULE.
For "IEEE" response see IEEE$488$TEST in IEEE$488$MODULE.

**SET$LOGGER$MODE:**

Allows the IEEE MODE to be set ON or OFF and the TAPE MODE to RECORD, PLAY or OFF.

It also calls the procedure SET$LOG$PARAMETERS before returning to the calling program.

**SET$MODE:**

Allows the user to set either the RTC time and date or the log parameters and calls the procedures SET$DATE$TIME or SET$LOG$MODE respectively before returning to the calling program.

**MAIN:**

This is the main procedure and does the following:

Firstly it checks to see if TAPE$MODE$FLAG is set to PLAY (1). If it is then a call to the procedure TAPE$PLAY$ is made which allows the logger to act as a data cassette replay unit. (see TAPE$MODULE$ further). If not, then a check is made to see if any one of the three soft-keys has been depressed or if the external contact, which is connected in parallel with the [E] key, has been made (KEY = 0 to 3). The following responses are made to each key.
[SET] - The set mode is entered by a call to the SET$MODE procedure.

[LOG-M] - The log mode is entered via a call to the LOG$TRIG procedure.

[TEST] - The test mode is entered by a call to the TEST procedure.

[E] or External contact - The logger will enter the log mode via the LOG$SCAN procedure

Note: All the above will return to the beginning of MAIN on completion. (see Figure 2)

Next, the main logger menu will be output to the display. "SET LOG-M TEST". In the LOG-M message the M indicates that the manual trigger option has been selected and will change to A for Alarm and X for external.

If the Alarm trigger mode has been selected a check is made to see if an alarm interrupt has occurred. (RST$55$FLAG set). If it has, then a call to LOG$SCAN is made. If the TAPE$MODE$FLAG is not off (2) then a check is made to see if the TAPE$ERROR$FLAG has been set. If it has, then a call is made to the TAPE$ERROR procedure. The Real-Time clock is then interrogated by a call to RTC$DATA. This returns the current date and time in the two strings RTC$DATE and RTC$TIME. The day of month is output to line 1 of the display and then the month word. This is after the month number has been converted into a 3 character month word via the look-up-table, DATE$MES. The time is then output to the display.

Lastly the 8085 interrupt register is read to determine whether either RST7.5 (keyboard interrupt) or RST 5.5 (RTC interrupt) has occurred. If the former then the keyboard register is read into the variable KEY and if the latter then RST$55$FLAG is set to OFFH. The Program then loops back to the beginning of MAIN.
6. THEORY OF OPERATION

Because the system was designed on a modular basis the theory of operation of each module making up the system will be discussed under the following headings:

1. IMT 8085 C.P.U. board.
2. IMT DATA ACQUISITION board.
3. IMT MFE TAPE board.
4. IMT IEEE-488 INTERFACE board.
5. Microcom 32K Memory board.
6. Data Logger P.S.U.

The discussion will encompass both the hardware and, were applicable, the software modules pertaining to each board.

It should be noted that, with the exception of the 32k memory board, all the above hardware was designed and developed inhouse specifically so that the Data Logger could be implemented.

6.1. IMT 8085 C.P.U. board

6.1.1. General

The IMT 8085 C.P.U. board is in fact a S.A. bus compatible single-board computer.

The design was conceived when it was realize that although IMT had a comprehensive 8085 microprocessor development system there was no easy and cheap solution for the implementation of small microprocessor controllers or systems. It was designed to be S.A. bus compatible so that the use of off-the-shelf boards would allow for system expansion if necessary, i.e. memory, input/output, data acquisition etc.

This single-board computer will enable many microprocessor applications to be tackled that would previously not have been viable due to the design tasks and their related cost and time requirements. It will also enable the designer to concentrate on the application of the computer rather than the hardware implementation details.
6.1.2. Features:

The IMT 8085 C.P.U. board has the following features:

- On board, battery backed up, CMOS Real-Time Clock.
- 50 bytes of battery backed up CMOS static RAM.
- Single +5V supply @ 300mA
- 2k or 4k (link selectable) JEDEC EPROM socket (2716 or 2732 type)
- 2k CMOS static RAM in JEDEC socket (6116 type)
- Two 8 bit 1/0 ports, one 6 bit port and a 14 bit counter/timer as well as 256 bytes of static RAM. (8155)
- 14 Pin header for strobed input keyboard (up to 8 data lines)
- 14 Pin header for intelligent Liquid Crystal Alphanuercis display (Densitron LMA2C16DC type)
- Power ON and Manual Reset.
- Board designed to be S.A. Bus compatible.
- All bus signals are buffered and will load the external S.A. Bus by a maximum of one LS TTL load.

6.1.3. Hardware:

The two prime design goals were:

- To be S.A. bus compatible.
- To be a complete stand-alone single board computer.

The design meets both these requirements.

With Reference to the 8085 CPU Board circuit diagram (Appendix G) the circuitry will be discussed in detail.

The 8085 (IC3) is the heart of the circuitry and is run at an input clock frequency of 4,194,30 MHz. This signal is derived from either the MC 146818 Real-time clock chip (IC 15) or by directly applying the crystal to the X1, X2 inputs of the 8085. The board will function at higher frequencies (say 6MHz) but the Intelligent display could not be used without modification as it requires a minimum 450 n sec pulse on 'E' for correct operation. (see data sheet for wait state modification). The following S.A. Bus output control lines are implemented by buffering and/or inverting the standard 8085 control signals: CLK, RESET, HOLDA, INTA and SYNC.
M1 is achieved by buffering (S1.50) and is active for an opcode fetch or INTA.

OW, IR, MW and MR are derived from the 8085 signals WR, RD and 10/M being decoded by a 74LS42 (IC 9). The RD and WR lines have 4k7 pull-up resistors on them to prevent spurious signals from being generated when the microprocessor is in the hold state.

The hold state is initiated by the S.A. bus HOLD line being taken low and indicates to the 8085 that another device is requesting the use of the address and data busses. e.g. for DMA. When the 8085 is ready to relinquish control it outputs the HOLDA signal. This signal together with RESET OUT and CLK (OUT) from the 8085 is used by a 74LS74 (IC 4) to implement a control signal for putting IC's 5, 6, 7 and 8 into the high impedance state. These IC's are buffers for the data, address and some control lines and are re-activated when HOLDA from the Bus is taken high.

IC 8 demultiplexes the low order address lines from the 8085 multiplexed address/data bus and buffers them onto the S.A. Bus.

IC 6 is a bidirectional three state buffer and is used to control data flow to and from the S.A. Bus. Pin 1 of IC 6 is the direction control pin and is normally held high allowing data to flow from the board to the S.A. Bus. The circuitry in the dashed block (IC 17 and IC 13) controls this line and will only allow data to be read from the S.A. Bus for memory addresses 4000H and above and 1/0 addresses 40H and above. i.e. external memory must be mapped above 4000H and external 1/0 above 40H. This circuitry was included because the first 16k of the microprocessor address space is absolutely decoded into 2k blocks by IC 12. The first two blocks are anded together by ICs 13 and 17 to form a contiguous 4k block for the 2732 type EPROM. Although use is not made of the full 16k it simplifies the design and still leaves room for 48k of external memory and 192 1/0 locations. (See System Memory and 1/0 Maps. FIG 3 and FIG 4)
FIG 3

SYSTEM MEMORY MAP
<table>
<thead>
<tr>
<th>Board/Port</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPEN</td>
<td>FFH</td>
</tr>
<tr>
<td>IEEE-488 BOARD</td>
<td>64H</td>
</tr>
<tr>
<td></td>
<td>63H</td>
</tr>
<tr>
<td>OPEN</td>
<td>60H</td>
</tr>
<tr>
<td>DATA ACQUISITION</td>
<td>5FH</td>
</tr>
<tr>
<td>BOARD</td>
<td>54H</td>
</tr>
<tr>
<td></td>
<td>53H</td>
</tr>
<tr>
<td>OPEN</td>
<td>50H</td>
</tr>
<tr>
<td>TAPE BOARD</td>
<td>4FH</td>
</tr>
<tr>
<td></td>
<td>48H</td>
</tr>
<tr>
<td></td>
<td>47H</td>
</tr>
<tr>
<td>OPEN</td>
<td>44H</td>
</tr>
<tr>
<td></td>
<td>43H</td>
</tr>
<tr>
<td>NOT USED</td>
<td>40H</td>
</tr>
<tr>
<td></td>
<td>3FH</td>
</tr>
<tr>
<td>L.C.D. DISPLAY</td>
<td>38H</td>
</tr>
<tr>
<td></td>
<td>37H</td>
</tr>
<tr>
<td>KEYBOARD</td>
<td>30H</td>
</tr>
<tr>
<td></td>
<td>2FH</td>
</tr>
<tr>
<td>NOT USED</td>
<td>28H</td>
</tr>
<tr>
<td></td>
<td>27H</td>
</tr>
<tr>
<td>I/O PORTS [8155]</td>
<td>20H</td>
</tr>
<tr>
<td></td>
<td>1FH</td>
</tr>
<tr>
<td>NOT USED</td>
<td>18H</td>
</tr>
<tr>
<td></td>
<td>17H</td>
</tr>
<tr>
<td>NOT USED</td>
<td>0CH</td>
</tr>
<tr>
<td></td>
<td>0BH</td>
</tr>
<tr>
<td>NOT USED</td>
<td>08H</td>
</tr>
<tr>
<td></td>
<td>07H</td>
</tr>
<tr>
<td>NOT USED</td>
<td>00H</td>
</tr>
</tbody>
</table>

**MAPPED TO CPU BOARD**

**FIG 4**

**SYSTEM I/O MAP**
6.1.4. Memory:

Two 28 pin JEDEC compatible sockets are provided, one each for RAM and EPROM. The address decoding allows for 2k or 4k of on board EPROM of the 2716 or 2732 type (link selectable) and 2k of CMOS static RAM of the 6116 or 5516 type. By cutting the track to pin 23 of IC 11 (RAM) and connecting pin 23 to +5V the RAM socket can be used for an extra 2k of EPROM (2716) if required.

6.1.5. Parallel I/O:

The board makes use of an Intel 8155 RAM/I/O/TIMER which has 22 programmable parallel I/O lines i.e. two 8 bit and one 6 bit port. These lines are defined by a software write to the 8155 command/status register. A further two lines are available for the programmable timer function.

<table>
<thead>
<tr>
<th>I/O ADDRESS</th>
<th>8155 SELECTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>18H</td>
<td>Internal command / Status Register</td>
</tr>
<tr>
<td>19H</td>
<td>PORT A 8 bit</td>
</tr>
<tr>
<td>1AH</td>
<td>PORT B 8 bit</td>
</tr>
<tr>
<td>1BH</td>
<td>PORT C 6 bit</td>
</tr>
<tr>
<td>1CH</td>
<td>Low order timer count 8 bit</td>
</tr>
<tr>
<td>1DH</td>
<td>High order timer count 6 bit</td>
</tr>
<tr>
<td></td>
<td>+2 mode bits</td>
</tr>
</tbody>
</table>

8155 I/O addressing scheme

The timer is basically a 14-bit down counter but has four different modes of operation allowing the following to be generated under software control.
A single square wave or programmable pulse.
A continuous square wave.
A single pulse on terminal count (rate generator)
A continuous pulse train or programmable strobe.

There are also 256 bytes of static RAM available starting at address 1800H.

6.1.6. Display:

6.1.6.1. Hardware

An intelligent 2 line by 16 character dot matrix, alphanumeric, L.C.D. display by the Densitron Corporation is used to implement the display function. It interfaces directly with the 8085 bus lines, with a minimum of hardware being required for the strobe enable ‘E’ pin. This signal is derived by the circuitry of IC 14 which combines \( \overline{OE}, \overline{OR} \) and \( \overline{CE} \) to produce a positive going pulse conforming to the enable pulse requirements. Note: This pulse length may not be less than 450n sec for correct operation therefore the 8085 input clock frequency should not be greater than 5MHz. i.e. system clock < 2.5MHz.

The display module has an on board CMOS microprocessor which takes care of all the display update, refresh and control functions.

The top line (line 1) of the display is generally used to display the relevant data and the bottom line (line 2) is used for soft-key function display.

The voltage divider \( R_6, V_{RI} \) is used to adjust the viewing angle of the display.

The display connects to the board via a 14 pin ribbon-cable header.

6.1.6.2. Software:

Associated with the display hardware is a PL/M software monitor module, DISPLAY$MODULE. This module has five Public procedures and two Public variables which can be accessed by any external program to facilitate using the display.
LCO$INIT: Public

Initializes the display as follows:

- 8 bit word length.
- Two line display.
- Cursor ON.
- Increment cursor position on entry.
- Clear display.

C$OUT: Public

Outputs one character to the LCD at the current cursor position.

Use: CALL C$OUT ('X')
where X is the ASCII character to be output.
or CALL C$OUT (0F3H)
where the hexadecimal value of the character to be output is used. (see data sheet in Appendix D for font table).

MENU: Public

Outputs three soft-key labels of 5 characters each to line 2 of the display.

where .X, .Y and .Z are the starting address locations of 3 separate 5 character strings X, Y and Z.

Note: Character 5 of Z is not displayed because it is off screen.

TEXT$OUT: Public

Outputs a string of characters starting at the current cursor position.

Use: CALL TEXT$OUT (.M, S)
where .M is the address of the first character in the message M and S is the size or number of characters to be displayed.
OR CALL TEXT$OUT (.M, LEN (M))
This will display the entire string M.
INST$OUT: Public

Outputs one instruction to the display instruction register.

Use: CALL INST$OUT (I)
where I is an instruction code from the instruction set below.

Instruction set:

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clear display</td>
<td>01H</td>
</tr>
<tr>
<td>Home</td>
<td>02H</td>
</tr>
<tr>
<td>Blink Cursor</td>
<td>0FH</td>
</tr>
<tr>
<td>Cursor ON</td>
<td>0EH</td>
</tr>
<tr>
<td>Cursor OFF</td>
<td>0CH</td>
</tr>
<tr>
<td>HOME 2</td>
<td>COH</td>
</tr>
<tr>
<td>Shift cursor left</td>
<td>10H</td>
</tr>
<tr>
<td>Shift cursor right</td>
<td>14H</td>
</tr>
<tr>
<td>Shift display left</td>
<td>18H</td>
</tr>
<tr>
<td>Shift display right</td>
<td>1CH</td>
</tr>
<tr>
<td>DISPLAY OFF</td>
<td>08H</td>
</tr>
</tbody>
</table>

(see data sheet further)

The two Public variables are:

. CHAR: This is a 16 character temporary storage location:

. POSITION: This variable holds the current cursor position at all times.

LCD$BUSY:

Is used within the module to determine when the display microprocessor is busy. I.e. processing an instruction. The procedure is called whenever a character or instruction is sent to the display module. The procedure returns to the calling program only once the busy flag is cleared. Before it returns it enters the new cursor position into POSITION.

Lastly, the display data and instruction registers are mapped to I/O locations 31H and 30H respectively.
6.1.7. Keyboard:

6.1.7.1. General

The keyboard is a 16 key x - y matrix type.
The keys have the following functions:

1. Ten have the standard numeric functions 0 through 9.
2. Three have the dedicated functions:
   1. Left shift or decrement [-].
      
   This key allows the cursor to be shifted one position to the left or a number to be decremented by one for each key depression. Which of the two functions it performs depends on the current program.
   2. Right shift or increment [+].
      
   This key performs the same as above but in the opposite direction.
   3. Enter [E]
      
   This key allows a variable to be entered into the program once it has been correctly keyed in.

3. The last three keys are soft-keys.

These keys change roles from time to time under software control. Their designated function at any one time is displayed in the lower line of the L.C.D. as a label of maximum 5 character length. The keys are colour coded for ease of association with the labels which have matching colours on the L.C.D. bezel.

This function allows a great deal of flexibility in the system/operator interface, especially with regard to the input and selection of parameters, and enhances the menu programming facility.
6.1.7.2. Hardware:

The keyboard has associated with it a circuit called Matrix Keyboard Encoder which was designed by Mr S Koster. This circuitry resides on a P.C. board which is mounted on the back of the keyboard and is connected directly to the C.P.U. bus lines via a ribbon cable. There are four data lines which convey the 16 possible key values to the C.P.U. (Allowance has been made on the CPU board for 8 bit data.) Of the remaining four lines two are used for +5V and ground, one for keyboard output enable and the last for the Data Available signal which is connected to RST 7.5 on the 8085.

The circuitry uses a 74C922 16 key encoder chip which has the following built in functions:

- On chip scanning.
- 2 Key roll over.
- Keybounce elimination.
- Tristate outputs.
- Last key register.

When a valid keyboard entry has been made a Data Available signal is generated. This signal stays high until the key is released and is used to generate a positive going pulse on RST 7.5 of the 8085. It also enables an oscillator which drives circuitry to cause a Beep sound for user feedback.

6.1.7.3. Software:

The software drivers for the keyboard circuitry are found in KEYBOARD$MODULE. This module has four Public procedures to facilitate its use.

KEY$INIT: Public

Sets up the RST 7.5 jump vector and the interrupt mask. It also enables the RST 7.5 interrupt.
Closely associated with this procedure is a procedure called RST$75.

RST$75: is an interrupt routine and is invoked when the Data Available strobe (RST 7.5 interrupt) is received from the keyboard encoder circuitry by the 8085. The procedure returns with the depressed key value in KEY (0 - F Hex) and an ASCII value in KEY$ASCII which is generated by a look-up table. An interrupt flag (FLAG1) is set and the program control is returned to where it left off.

KEY$ENTRY:    Public

Waits for a key depression to occur which is indicated by FLAG1 being set. Once this happens FLAG1 is reset and the procedure returns to the calling program with the keys value in KEY.

KEY$IN:    Public

Allows a string of up to 16 characters (usually numeric) to be assembled in the external variable CHAR which are usually displayed on the LCD for feedback. It also allows the cursor to be shifted right or left for editing purposes.

This procedure will only return once [E] is received. At this point a correctly entered string of numerals will be available in CHAR and a count variable (l) will hold the number of characters entered.

6.1.8. Real-Time Clock:

6.1.8.1. General:

The time-of-day clock with calendar function was implemented using a single CMOS chip (MC 146818). It has the following main features:

- Battery back-up for the clock functions as well as for 50 bytes of static RAM. This RAM is used for all variables and input parameters that need to be protected from power-cuts.

- A Programmable periodic interrupt mode.

- An alarm interrupt mode.
See data sheets in Appendix D further.

The clock is set via menu-promtented keyboard entry.

6.1.8.2. Hardware:

The clock chip interfaces directly with the CPU bus lines. The clock out (pin 21) which is derived from the 4 MHz crystal is used as the microprocessor clock input. An on-board Ni-CAD battery provides the 3, 6 V required for battery back-up. To implement a glitch free change over from +5V to battery it was found necessary to include a comparator circuit which isolates the CE line on the RTC chip as power fails. This circuit is set to operate when the +5V rail falls below 4,8V. When the +5V rail falls below 3,6V the battery takes over via a diode switch. This circuitry is hard wired into the system. (IC's 18 and 19)

6.1.8.3. Software

REAL$TIME$CLOCK$MODULE contains the software drivers for the above hardware. They are:

RTC$INIT: Public

Initializes the RST5.5 jump vector and sets the interrupt mask on the 8085. It also sets the initial Periodic interrupt rate at 0,5 sec and leaves it in the disabled state.

RST$55: Public

Is entered via the RST 5.5 interrupt and sets RST$55$FLAG before returning to where the program left off. The flag is used to indicated that an interrupt on the RST 5.5 line has occurred.

SET$DATE$TIME: Public

Puts the RTC chip in the set mode and allows the date and time or the alarm time to be set. The above is accomplished via menu-promtented keyboard entry and once completed the RTC chip is put into the run mode and the display is cleared before returning to the calling program.
The procedure RTC$INIT$DATA is used by the above to convert ASCII characters into packed BCD which in turn is used to load the relevant RTC registers with the new data.

RTC$DATA: Public

This procedure waits for the RTC internal update cycle to be completed before it reads the data registers and converts them to two 6 character ASCII strings, RTC$DATE and RTC$TIME.

Their format is:

RTC$DATE: DD MM YY
RTC$TIME: HH MM SS

This data is used to output to the LCD or used whenever current date and time data are required.

6.1.9. Specifications:

1. System Clock: 2,09715 MHz

2. Memory:

On Board: 2k or 4k bytes EPROM; start address 0H (2716 or 2732)
2k bytes static RAM; start address 1000H (6116 type)
256 bytes static RAM; start address 1800H (8155)
50 bytes battery backed up static RAM; start address 2000H (MC 146818).
Expansion: 48k bytes. ROM or RAM starting at 4000H.

3. Input/Output:

Parallel: 22 programmable 1/0 lines.
Serial: The SID and SOD pins can be used for serial 1/0.
Expansion: 192 1/0 locations available starting at 40H
4. Interrupts

<table>
<thead>
<tr>
<th>NAME</th>
<th>PRIORITY</th>
<th>USE</th>
<th>RESTART ADDRESS</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRAP</td>
<td>1</td>
<td>S.A. BUS</td>
<td>NMI</td>
</tr>
<tr>
<td>RST 7.5</td>
<td>2</td>
<td>Keyboard interrupt</td>
<td>3CH</td>
</tr>
<tr>
<td>RST 6.5</td>
<td>3</td>
<td>Not used</td>
<td>34H</td>
</tr>
<tr>
<td>RST 5.5</td>
<td>4</td>
<td>RTC interrupt</td>
<td>2CH</td>
</tr>
<tr>
<td>INTR</td>
<td>5</td>
<td>S.A. BUS</td>
<td>INTR</td>
</tr>
</tbody>
</table>

5. Timers: Time of day clock/calendar

- Periodic interrupt generator on RST 5.5 (MC 146818)
- Alarm clock interrupt on RST 5.5 (MC 146818)
- 14 bit general purpose timer (8155)

6. Electrical:

Requires 5V dc at 300mA for completely populated board.

6.2. IMT Data Acquisition Board

6.2.1. General:

This board was designed with the following basic requirements in mind.

1. S.A. Bus compatibility.
2. 12 bit resolution.
3. 32 multiplexed analog input channels.
4. Data throughput of at least 64 samples/sec.

The AD 574A, Analog to Digital (A/D) converter by Analog Devices was chosen to implement this board. It is a 12 bit (or 8 bit) successive-approximation A/D converter that can easily be interfaced to the S.A. bus. It is configured as a bipolar input device with the following input voltage ranges: -5 to +5V or -10 to +10V which are link selectable.
6.2.2. Data Throughput:

For a successive approximation A/D the input signal is required to remain constant during the conversion time. This implies that it must not change by more than half a least significant bit (LSB) during this time.

The maximum input frequency \( f_{\text{max}} \) for a given precision (n bits) is given by:

\[
 f_{\text{max}} / n\text{bit} = \frac{1}{2\pi x 2^n \times \text{Sampling time}}
\]

Ref. 4 Page 15

Sampling time for the AD574 is:

12 bit conversion: - 35\( \mu \)sec
8 bit conversion: - 24\( \mu \)sec

\[
 \therefore f_{\text{max}} / 12\text{bit} = \frac{1}{2\pi x 2^{12} x 35 \times 10^{-6}}
\]

\[
= 1,11 \text{ Hz}
\]

similarly \( f_{\text{max}} / 8\text{bit} = 25.9 \text{ Hz} \)

Because these Figures are rather low a Sample and Hold (S&H) amplifier (AD 582) is included in the design to improve them. \( f_{\text{max}} \) of the system is now governed by the S&H aperture jitter Figure which is 15\( \mu \) sec for the AD 582.

Now \( f_{\text{max}} / n\text{bit} = \frac{1}{2\pi x 2^n \times \text{Aperture jitter}} \)

Ref. 3 Page 14-34

\[
 \therefore f_{\text{max}} / 12\text{bit} = \frac{1}{2\pi x 12^{12} x 15 \times 10^{-9}}
\]

\[
= 2,590 \text{ kHz}
\]

and \( f_{\text{max}} / 8\text{bit} = 41,446 \text{ KHz} \) but in practice it is limited to the reciprocal of the sum of A/D conversion time (8 bit = 24\( \mu \) sec) added to the S&H acquisition time (6\( \mu \) sec)
If the sampling time is not referenced to an absolute time then the $f_{\text{max}}$ is again the reciprocal of the sum of the conversion time ($35\mu\text{ sec for 12 bit}$) plus the acquisition time of the S&H amp ($25\mu\text{ sec for 12 bit}$)

\[
\text{i.e. } f_{\text{max}} / 12\text{ bit} = \frac{1}{(25 + 25) \times 10^{-6}} = 16,67 \text{ KHz}
\]

Throughput for the Data acquisition board, using overlapped multiplexing is therefore:

12 bit resolution: - $60\mu\text{ sec.}$
8 bit resolution: - $30\mu\text{ sec.}$

6.2.3. Printed Circuit layout and Power supply requirements:

The layout of a 12 bit data acquisition board is fairly critical if full use is to be made of the available resolution. Consider that if the A/D is configured for a 10V input voltage range then one bit represents 10V divided by $2^{12}$ or 2.44mV. This means that any noise signals (and power supply ripple) must be kept well below this value if 12 bit accuracy is to be achieved.

Some points to consider during P.C. layout are:

1. Separate digital and analog circuits by "ground" tracks.
2. Reduce edge connector contact resistance effect by using parallel connections.
3. Use thick track and run separate ground lines for each DC supply.
4. De-couple DC supplies at the input to the PCB as well as at each IC. Use a combination of low value ceramic and high value tantalum bead capacitors to achieve this. i.e. 0.1\mu\text{F} ceramic in parallel with 47\mu\text{F} tantalum.
5. Use ground guard's on sensitive input lines.
6. Leave as much ground plane on the PCB as possible.
7. Connect all unused IC pins to ground.
8 Connect the analog and digital grounds together locally on the board i.e. at A/D converter.
9 Site A/D and S&H as close as possible physically.
10 If unavoidable, analog and digital lines should cross at 90° to minimise interference.

A design for a low noise ±15V tracking regulator supply was found in the National Semiconductor Linear Databook (Ref. 13 Page 1-45) which gave the required regulation and output ripple Figures. This design uses an opamp in a feedback configuration to improved the output ripple.

6.2.2. Hardware:

With reference to the IMT Data Acquisition board circuit diagram (Appendix G). IC1 buffers most of the incoming lines from the S.A. bus. IC 2 buffers the rest and decodes the board address (absolutely). The board occupies eight consecutive port locations (50H to 57H) and the DIL switch S1 and pull-up resistors select the actual board start address.

IC4 (c and d) decode the chip select (CS) line to the A/D converter (IC5). IC3 (c) has a similar function for IC6 (8255). IC3 (a and b) control the chip enable (G) of IC7 (74LS245). This ensures that the bi-directional data lines are only selected when ((IR OR OW) AND board select from IC2) is true. The DIR pin of IC7 is controlled by OW.

The analog signals of the Data Acquisition board are multiplexed to the S&H amp by two Multiplexer (MUX) chips (IC9 and 10). The channel to be sampled is selected by IC6 (Port B) under software control. IC4(a) ensures that only one of the MUX IC's is operative at any time.

IC8 is the Sample and Hold (S&H) amplifier.

The sample/hold mode is selected automatically by the status (STS) output of the A/D. When the A/D is triggered, by a software command, the conversion cycle begins by taking the STS line high which puts the S&H amp into the hold mode. It remains in the hold mode until the A/D has completed it's conversion. STS is then taken low which allows the S&H amp to revert to the sample or track mode.
Provision has been made on the PCB to accommodate different size hold capacitors (C2) and there are also two 8 bit, software configurable, digital input/output ports available for digital data acquisition. (Port A and Port C of IC6).

Analog input connection to the board is made either by the on board terminal blocks or via the 34 way ribbon cable header (P3).

To calibrate the board VR1 and VR3 can either be set to 50Ω (midway) or can be adjusted as per the procedure “Bipolar Operation” on Page 7 of the AD574 Circuit Details (Appendix D). VR3 is used as an offset null pot for the S&H amp. (IC8).

6.2.3. Software

The software module DACQ$MODULE has been written to facilitate the use of the Data Acquisition board. It has the following procedures:

D$ACQ$INIT: Public

Initializes the on-board 8255 (IC6) with all its ports set to the output mode.

MUX$35:

Is used to ensure that the A/D converter has a full 35 micro seconds in which to complete a 12 bit conversion before being read. It also increments the channel pointer, CHAN, and outputs this new value to Port B of IC6 which selects the new MUX channel. The MUX increment is done during this time to speed up system throughout by "overlapping" some of the system settling times and is allowed because the S/H amp is always in the "Hold" mode before this procedure is called.

The procedure returns to the calling program after 35μ sec with the MUX switch incremented by one.

LOG$DISPLAY$PROCEDURE:

This procedure configures the display to output "CHAN.#" on line 1 and a soft-key menu on line 2 which depends on whether the TEST flag is set or not

TEST = 0  "...........  .............  EXIT"
TEST = 1  "...........  ENTRY  EXIT"
This flag is supplied by the calling program.

**LOG$DATA$NORMALIZE:**

Normalizes the 12-bit data (DATA$HI and DATA$LO) for a selected channel, to a value in the range 0 - 10000. The value is stored in the 16 bit variable AD$DATA after the LSB has been rounded up and represents milli Volts.

If the IEEE$MODE$FLAG is set then AD$DATA$ is output (high byte first) to the IEEE-488 port provided that the IEEE listener indicates via the NRFD line that it is ready for data (NRFD=1) and that the Data ready buffer (DRB) line is low. AD$DATA$ is then converted first to a bipolar (0-5000mV) value and then this binary value is converted to four ASCII numerals with sign and output to the LCD with "mV" appended.

**LOG$SCAN:**

This is the main procedure and is called when the Data Logger is triggered.

First the buffer pointer (BUFF$POINT), which points to the next location in the Buffer memory to be filled, is initialized to zero and then the MUX switch is set to channel #1.

The periodic interrupt function from the Real-Time clock is set to the selected scan rate and enabled.

The current value of Date and Time is read from the RTC to be used in the Data Header.

A wait loop is then entered until the periodic interrupt occurs. When this happens the A/D conversion cycle is started by a dummy output to Port 50H. The time delay procedure (MUX$35) is called and when it returns, two bytes of data are read into tempory storage arrays (DATA$HI and DATA$LO) and then transfered to the Buffer memory (high byte first). BUFF$POINT is then incremented by two. Note that the MUX switch is incremented by one during the MUX$35 call.

When the required number of channels has been converted the MUX switch is reset to channel #1 and LOG$DATA$NORMALIZE is called. This procedure displays the value of the analog channel initially selected, on the LCD in millivolts.
If the required number of scans has not yet been met the program loops back to wait for the next periodic interrupt, otherwise the interrupt is disabled.

If the TAPE\$MODE\$FLAG is set, a tape dump of all the logged data, preceded by the Data Header (DATE, TIME, etc) is made to the Digital Tape recorder.

If the ALARM TRIGGER mode was active then the alarm is re-enabled before returning to the calling program.

D\$ACQ\$TEST: Public

Allows a specific, selected channel to be displayed and output to the IEEE port. The data is updated as fast as the software can run.

The selection of the channel to be displayed depends on the value in DISP. This value can be modified while the program is running by using the [\(-\)], [\(+\)] or [ENTRY] Keys. The procedure will run until either the [EXIT] key is pressed or the Data Logger is reset.

6.3. IMT M.F.E. Tape Board:

6.3.1. General:

The Mass storage media chosen for the Data Logger was Digital Cassette. This choice was made for two reasons:

1. A data logger typically has to operate in a harsh environment and therefore floppy disk storage, for instance, would not perform satisfactorily. A digital cassette (same format as the common audio cassette) is robust and will survive transportation fairly well.

2. Two M.F.E. digital cassette tape transports were available.

The MFE 450B tape transport has the following features:

- Robust design and good environmental characteristics.
- High data reliability is achieved by using an optically encoded clock to regulate the transport motor speed. (better than $1 \times 10^{-7}$ error rate.)
- Transport reliability is indicated by the 15,000 Hours MTBF Figure.
- Storage capacity is 676k bytes using both sides of a 300 foot cassette at 800 BPI.
6.3.2. Hardware:

The MFE 450B type transport was fitted with the MFE 414 parallel option which made interfacing it to the S.A. Bus a fairly simple task. This circuitry is called the IMT MFE Tape Interface and will be discussed with reference to the circuit diagram (see Appendix G).

IC1 is used to buffer the board from the S A Bus. IC2 is a 6 bit binary comparator which is employed as an address decoder. The board takes up four consecutive (fully decoded) input/output locations with the start address being set up on S1 (selected to start at 44H for Data Logger).

When the I/O address and switch value match, IC2 (pin 9) is allowed to be pulled high by a 4k7, resistor. (IR1 pin 7) which causes the CS line on IC4 (pin 6) to be taken low by the inverter IC3 (c).

The data lines of IC4 (8255) are buffered from the SA Bus via a 74LS245 (IC5) bi-directional buffer. The direction of data flow is controlled by the SA Bus OW line. IC5 chip enable (G) is only low (enabled) when IC2 (pin 9) is high AND OW OR IR are active (low). At all other times IC5 is in the high impedance state.

IC4 Port A and Port B are used for data input and output respectively (from the tape recorder's point of view). Port C is used for controlling the tape drive via the control lines CE1 to CE7. The Port C line along with R1, R2 and T1 was designed to drive a LED to be used as a tape transport motion or busy indicator but was not implemented. All the above lines as well as the power lines required by the tape drive were terminated in a 40 way ribbon cable header on the board (P2).

The power required by the tape transport comes from the Data Logger P.S.U. via P3 except for the -12V which is generated on board by a DC to DC converter (CB 3811).

6.3.3. Software:

The procedures used to control the tape transport are to be found in TAPE$MODULE:

```
TAPE$INIT: Public
```

Initializes the 8255 on the M.F.E. Tape Board to the required configuration. (Port A = output, Port B = input and Port C = output)
TAPE$STROBE$OUT:

Is used to strobe a control line (CE1 - CE7) from high to low and back to high.

USE: CALL TAPE$STROBE$OUT (CE4)

This would reset the EOT flag in the tape transport.

TAPE$STROBE$IN:

Strobes data into a variable T$DATA.

USE: CALL TAPE$STROBE$IN (CE6)

This would cause the USRT status register to be read into T$DATA.

DELAY:

Produces a delay of approximately 1 milli second.

USE: CALL DELAY (25)

This would cause the program to be delayed by 25m sec before proceeding.

PORT$A$OUT:

Outputs data to the tape transport USRT.

USE: CALL PORT$A$OUT (CONTROL, DATA)

This outputs DATA to the control register designated by CONTROL.

TBMT

This procedure waits for the transmit buffer of the USRT to be empty before returning to the calling program. This is achieved by repeatedly reading the USRT's status register until the TBMT bit is high.
RDA:

Reads the USRT status register repeatedly until the Receive Data Available flag is set. It then returns to the calling program.

TAPE$ERROR$MES:

Outputs a message to the L.C.D. which depends on the error condition prevailing at the time. The error condition is determined by reading the USRT status register and the message could be one of the following:

1 "END OF TAPE!"
2 "WRITE PROTECTED!"
3 "LOAD CASSETTE!"

These messages allow the user to quickly determine why the program has aborted.

REWIND:

Checks whether a cassette is loaded and if so rewinds it to the beginning. It then pauses for 360 m sec, as required by the cassette mechanics, before returning to the calling program.

POSITION$TAPE:

Allows the tape to be positioned at the beginning of a particular record which is denoted by the value in RECORD.

GET$OFF$LEADER:

Positions the tape head just after the clear lead-in tape (if called from TAPE$WRITE$INIT) or just before record #1 (if called from TAPE$READ$INIT).

RECORD$DISPLAY

Displays the next record to be read or written to, on the LCD.

TAPE$BLOCK$DUMP: Public

Dumps a block of data, preceded by a Data header onto tape. The header contains: Date, Time, Scan Rate, Number of Channels, Number of Scans (high byte), Number of Scans (low byte). The size of the block is determined by:
Block size = (No. of Chans x No of Scans x 2) + 16 bytes.
i.e. for 27 channels and 240 scans
Block size = (27 x 240 x 2) + 16
= 12,976 bytes.

:. Tape capacity in records is approximately = 676,000 ÷ Block size.

Using the above Figures for Block size:

Tape capacity = 52 records.

The procedure checks for tape errors at all times during and before the dump process and sets the TAPE$ERROR$FLAG if necessary. After all the data has been dumped onto tape an inter-record gap is generated and the variable RECORD is incremented before the procedure returns.

If the procedure is called from TAPE$WRITE$INIT then RECORD = 0 and a dummy block of 256 bytes is initially dumped onto tape. This block is never read but is required by the transport logic for error free operation. Because a USRT is used in the tape drive electronics a SYNC word (AAH) is required at the beginning and end of each record and is provided for in the procedure.

TAPE$BLOCK$READ:

This procedure positions the tape head before the selected record and then reads the header data into the Buffer memory. This data is also used to calculate the amount of data to be read into the Buffer memory. Tape errors are monitored throughout the procedure and the RECORD counter is incremented before returning to the calling program.

RECORD$SELECT:

Allows any record (limit 1 to 99) to be selected via the keyboard and its number to be displayed on the LCD prior to reading it.

The program allows for incrementing or decrementing the record number by one for each depression of the [+]* or [-] keys respectively. Any record number can be entered via the [ENTRY] soft key.
TAPE$READ$INIT:

Ensures that a cassette is loaded and then rewinds the tape and positions the tape head at the beginning for record #1.

BUFF$TO$IEEE$DUMP:

Dumps the data header and then the log data from the Buffer memory to the IEEE-488 port. The data will not be output unless the IEEE listener indicates that it is ready to receive data (NRFD = 1).

TAPE$PLAY$: Public

Checks keyboard entries to determine whether:

- A record is to be read.
- The present record number is to be incremented or decremented.
- The program is to exit or returned to the calling program.

TAPE$WRITE$INIT:

Initializes the data cassette in the transport by rewinding it, getting off the clear leader and executing a dummy dump. The procedure returns to the calling program with the tape head positioned in front of Record #1 and with the variable RECORD = 1.

TAPE$ERROR$: Public

This procedure calls TAPE$ERROR$MES which outputs the relevant error message to the LCD and then waits for a response from the user. The user must respond by removing and/or inserting a cassette depending on the prompt displayed. TAPE$WRITE$INIT is then called and the TAPE$ERROR$FLAG reset before returning to the calling program.
6.4. IMT IEEE-488 Board

6.4.1. General:

This board was designed and developed by Mr G de Waal to primarily interface a discreet logic circuit to the IEEE-488 Bus. For this application the Fairchild 96LS488 LSI circuit was ideally suited as it required no other intelligent hardware to implement.

The writer was involved with the project during the development stage and suggested extending the circuitry to include S.A. Bus compatibility. He also undertook to design the extra circuitry required. As the suggestion was not in conflict with the original design goals and would make the final board more versatile, it was decided to incorporate it.

Mr de Waal left IMT before being able to document his work and this was left to the writer, hence the inclusion of this material in this document.

The Board implements the IEEE-488 bus Talk, Listen and TALK/LISTEN modes (but not the controller mode) and either interfaces to the S.A. Bus as a fully buffered I/O peripheral with minimal software overhead requirements or to discreet logic in a stand-alone configuration (Talk mode only). It does not cater for the Extended Address mode or Parallel Pole mode.

6.4.2. Hardware:

The GPIB or IEEE-488 bus is an internationally recognized bus standard for instrumentation interface which defines mechanical, electrical and functional specifications. It is organized as a byte-serial, bit-parallel communications structure with 16 active bus lines which can be divided into 3 functional groups:

. 8 Data lines \( D_{10} \) to \( D_{10}^B \) which require negative logic signals.

. 3 handshake lines for data transfer \( NRFD, NDAC \) and \( DAV \).

. 5 control lines \( ATN, REN, EOI, IFC \) and \( SRQ \).

With reference to the IMT IEEE-488 board circuit diagram (Appendix G).
The handshake and control lines are implemented by IC1 (96LS488) and conform to the IEEE-488 standard of 1980. The data lines are buffered by IC3 or IC4 (74ALS245). These are bi-directional bus driver/receivers and capable of sinking 48mA which complies with the IEEE-488 bus specifications.

The 96LS488 can be operated in 14 different modes selectable by D.I.L. switch SW2. Table 1 (page 9) of the 96LS488 data sheet (Appendix D) defines these modes. (Note that the extended address option is not available.)

The IEEE device address is set up by D.I.L. switch SW1 (1 to 5). This allows selection of any of the 31 address codes or unlisten. (see table 2 on page 10 of the 96LS488 data sheet). The remaining switches of SW1 (6 to 8) are used to connect inverted "strobe output" signals to their "ready input" signals. This allows the required handshakes to be automatically generated if the peripheral electronics cannot supply them.

The 96LS488 clock is derived from a crystal and R1, C4 and C5 or the RC network R1, C5. (See data sheet, in Appendix D, page 12 fig 6). The crystal option should be chosen if precise timing is required otherwise the RC oscillator is adequate. Both should produce a 10MHz clock frequency which gives correct source handshake delays. A "power-on" reset function is provided by R6 and C7 which are connected to the master reset pin (MR) of the 96LS488.

There is space on the board (P2) for 3 LED's which are driven by the following 96LS488 status lines:

- **R/L**: this diode is on when the device is in the Remote mode.
- **LAD**: this diode is on when the 96LS488 is addressed to listen.
- **TAD**: this diode is on when the 96LS488 is addressed to talk.

Also on P2 are two points to be connected to a "remote/local" switch which requests a return to local input (RTL = 0) when depressed.

In the stand alone configuration IC4 buffers the signal lines from IC's 5 and 6 to the D10 bus lines. IC's 5 and 6 are 8 line to 4 line switches and allow the 96LS488 to select either a data or status read from the external electronics by means of the D/S/E line. Only the IEEE TALK mode is implemented with this hardware configuration.
In the stand alone configuration only IC's 1 to 6 but excluding IC3 are used. The board is connected to the external electronics via P3.

In the S A Bus configuration only IC1 (and its peripheral components) and IC3 are required of the above mentioned circuitry. (i.e. not IC's 2, 4, 5 and 6).

IC3 allows for bi-directional data transfer and buffers the data to and from the 8255 (IC9). The direction pin of IC3 is controlled by DRB (from the 96LS488) gated with Port C2 (IC9). IC12 is used to buffer the board from the S A Bus. IC11 is a 6 bit binary comparator which is employed as an address decoder. The board takes up four consecutive (fully decoded) input/output locations with the start address being set up on SW3. (Selected to start at 60H for the IMI Data Logger).

When the I/O address and switch values match, IC11 (pin 9) is allowed to be pulled high by a 4k7 resistor (IR2 pin 7) which causes the CS line on IC9 (pin 6) to be taken low by the inverter IC13c. The data lines of IC9 (8255) are buffered from the S A Bus via a 74LS245 (IC10) bi-directional buffer. The direction of data flow is controlled by the SA Bus OW line. IC10 chip enable (G) is only low (enabled) when IC11 (pin 9) is high AND OW OR IR are active (low). At all other times IC10 is in the high impedance state.

IC9 is a Programmable Peripheral Interface (PPI) and has three 8 bit ports. In this application IC9 Port A was configured in mode 2 (bi-directional I/O with handshakes) to simplify software design. To work correctly in this mode a hardware modification had to be made because the TXST or STST signals from the 96LS488 were not compatible with the ACKA requirements of IC9. The track was cut between IC8 pin 10 and IC9 pin 11 and the IEEE bus line, DAV (IC1 pin 28), was connected to IC9 pin 11.

A further modification was made by connecting, the IEEE bus line, NRFD (IC1 pin 26) to IC9 pin 23 (PB5). This line is used to test whether the IEEE bus is clear for data exchange (by the S A Bus).

Port B is used to input status data from the 96LS488 to the S A Bus and the remainder of the Port C lines (PC0 and PC2) are used as output "bit" ports.

Most of the 8255 port lines connect directly to the 96LS488 but the handshake lines require inverting for proper operation. This is accomplished by IC's 7 and 8. The
signal from PC7 (OBF) is gated with D/S/E from the 96LS488 by IC7 (a, b and c) to produce either a TXRDY or STRDY signal for IC1.

Port C2 is used to control the chip enable pin (G) of IC3. This signal is gated with DRB from IC1 by IC8 (a and d) to control the direction pin (DIR) of IC3.

6.4.3. Software:

The software drivers for this board are to be found in IEEE$488$MODULE:

IEEE$488$INT: Public

Initializes Port A of IC9 to mode 2 and enables IC3 via Port C2.

IEEE$488$TEST: Public

Outputs 0 to 255 as data to the IEEE-488 bus. It is used by an external computer (i.e. HP85) to determine if the interconnection between itself and the DATA LOGGER has been correctly made and is functional.

After the board has been initialized by IEEE$488$INIT it is treated as a bi-directional I/O port situated at (the selected start address) 60H. All hand shaking between 96LS488 (IC1) and the 8255 (IC9) is automatic and the 96LS488 or NRFD status can be read from Port B (location 61H.)

Status Data: Port B

PB0 = LAD from 96LS488
PB1 = CLR from 96LS488
PB2 = RQS from 96LS488
PB3 = DRB from 96LS488
PB4 = D/S/E from 96LS488
PB5 = NRFD from IEEE-BUS

To use: Establish whether theIEEE-488 bus is ready to receive data by waiting for DRB (from Port B) to equal 0 and then output a byte of data to Port A.

In the IMT Data Logger the IEEE-488 board is only used for outputting data to the IEEE-Bus. This occurs in the LOG$DATA$NORMALIZE and BUFF$TO$IEEE$DUMP procedures.
6.5. Microcom 32k Memory board

The Buffer and Program Memory requirements for the Data Logger were met by using an "off-the-shelf" 32k byte S A Bus Memory board from Microcom (see Appendix G).

This board can accommodate a mixture of 2716 EPROM's and 6116 CMOS static RAM and is therefore very useful in this application.

The board start address was chosen to be 8000H which is the start address of the Buffer RAM. The Buffer RAM is 16k bytes long and extends from 8000H - BFFFH.

The CPU restarts at OH but is immediately vectored to the program located at C000H. The program is approximately 8k long and therefore extends from C000H - DFFFH.

6.6. Data Logger P.S.U.

The power requirements for the Data Logger circuitry are as follows:

+5V at 3A - Tape drive and logic.
-5V at 500mA - Tape drive.
±15V at 35mA - Low noise for Data Acquisition board.

The +5V is supplied by an OLV15 (UGLY) power supply with the power transistor re-located to an external heat sink.

The -5V and ±15V requirements are supplied by the Data Logger P.S.U. board circuitry (see Appendix G).
7. OPERATING INSTRUCTIONS:

The Data Logger is set-up and controlled by the user by means of the 16 key keyboard. (See System description 3.1)

The program displays either an explicit prompt like "Set Channel to be Displayed" or "Load Cassette" and then waits for the user to respond, or a menu like "SET LOG-M TEST" from which the user may select his choice via soft-key entry. The keyboard and display functions are illustrated below (Fig 5).

![Keyboard & Display Functions Diagram](image)

KEYBOARD & DISPLAY FUNCTIONS

FIG 5

The IEEE-488 interface connector, External Trigger connector block and the Data Acquisition connector along with the 220V mains plug, fuses and on/off switch are all located on the Data Logger back panel. (Fig 6)

The pin allocation for the Data Acquisition connector is tabulated below (Fig 7). The DATA LOGGER IEEE-488 address is set at 10 (see Appendix G for switch settings).
### BACK PANEL

**FIG 6**

<table>
<thead>
<tr>
<th>CHANNEL</th>
<th>DEE CONNECTOR PIN</th>
<th>CHANNEL</th>
<th>DEE CONNECTOR PIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20</td>
<td>17</td>
<td>28</td>
</tr>
<tr>
<td>2</td>
<td>21</td>
<td>18</td>
<td>29</td>
</tr>
<tr>
<td>3</td>
<td>22</td>
<td>19</td>
<td>30</td>
</tr>
<tr>
<td>4</td>
<td>23</td>
<td>20</td>
<td>31</td>
</tr>
<tr>
<td>5</td>
<td>24</td>
<td>21</td>
<td>32</td>
</tr>
<tr>
<td>6</td>
<td>25</td>
<td>22</td>
<td>33</td>
</tr>
<tr>
<td>7</td>
<td>26</td>
<td>23</td>
<td>34</td>
</tr>
<tr>
<td>8</td>
<td>27</td>
<td>24</td>
<td>35</td>
</tr>
<tr>
<td>9</td>
<td>25</td>
<td>25</td>
<td>9</td>
</tr>
<tr>
<td>10</td>
<td>2</td>
<td>26</td>
<td>10</td>
</tr>
<tr>
<td>11</td>
<td>3</td>
<td>27</td>
<td>11</td>
</tr>
<tr>
<td>12</td>
<td>4</td>
<td>28</td>
<td>12</td>
</tr>
<tr>
<td>13</td>
<td>5</td>
<td>29</td>
<td>13</td>
</tr>
<tr>
<td>14</td>
<td>6</td>
<td>30</td>
<td>14</td>
</tr>
<tr>
<td>15</td>
<td>7</td>
<td>31</td>
<td>15</td>
</tr>
<tr>
<td>16</td>
<td>8</td>
<td>32</td>
<td>16</td>
</tr>
<tr>
<td>GND</td>
<td>36</td>
<td>GND</td>
<td>17</td>
</tr>
</tbody>
</table>

**DATA ACQUISITION CONNECTOR PIN ALLOCATION**

**FIG 7**
To operate the Data Logger in the log mode the following sequence of steps should be taken.

1. Initial Power up (7.1)
2. Set Date and Time (7.2)
3. Set Logger Parameters (7.3)
4. Data Acquisition Test and Calibration (7.4) if required.
5. IEEE-488 Port Test (7.5) if required.
6. Cassette loading in record mode (7.6), referring to Tape Transport Prompts (7.7) when necessary.

To operate in the Data Playback mode, Initial Power up (7.1) and then Data Playback (7.8) should be followed.

7.1. Initial Power up

On initial power up and switch on the Main menu and a very mixed up date and time should be displayed on the LCD.

```
?? ??? ???:???:??
```

Note: The ?'s represent random characters at this stage.

If the above display is not in evidence then the mains switch should be switched off and then on once more. If the Main menu is still not displayed then the instrument is faulty.

7.2. Set Date and Time

To set the required date and time into the Data Logger the following steps must be followed. From the main menu push [SET] to get

"SET MODE"
TIME LOG EXIT

push [TIME] for

```
SET DATE
DDMMYY
```
Key in the required date in the format displayed using the [←] or [→] keys to edit any incorrectly keyed value.

For example: 1st March 1986 is keyed in as

\([0], [1], [0], [3], [8], [6]\).

When you are satisfied that the date is correct push [E] to get

```
SET TIME
HHMMSS
```

The required time, in 24 hour clock format, is now keyed in.

example: 1pm is keyed in as \([1], [3], [0], [0], [0], [0]\).

Wait for your time reference to reach the set value before pushing [E]. The program now returns to the main display.

Using the values in the above examples the display would look as follows:

```
01 MAR.  13:00:00
SET LOG=? TEST
```

With the time updating each second.

7.3. Set Logger Parameters:

Starting with main display push [SET] to get

```
"SET MODE"
TIME LOG EXIT
```

Push [LOG] for

```
IEEE MODE SET
OFF  ON
```
[OFF] = Program does not output data to the IEEE-488 port.
[ON] = Program outputs the "Channel to be Displayed" data (see below) to the IEEE-488 port.

Note: The program will hang up in the Log or TEST-DACQ modes if no IEEE listener is connected to the Data Logger and [ON] is selected.

Upon selection of [OFF] or [ON] you get

TaPe MoDe Set
ReC PLaY OFF

[REC] = Program records data on cassette at the end of each log.

[PLAY] = Program enters the Tape Play-Back mode.

[OFF] = Program does not record data.

Upon selection of [REC] or [OFF] you get:

SeT
No Of ChAns: ??

Key in the required number of channels to be logged (limits 1-32), e.g. 9 is entered as [0], [9] and then [E] to get

SeT
ScAn RaTe: ??

Key in one of the codes tabulated below.

<table>
<thead>
<tr>
<th>code</th>
<th>scan rate</th>
<th>code</th>
<th>scan rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
<td>500m sec</td>
<td>12</td>
<td>162.5m sec</td>
</tr>
<tr>
<td>14</td>
<td>250m sec</td>
<td>11</td>
<td>131.25m sec</td>
</tr>
<tr>
<td>13</td>
<td>125m sec</td>
<td>10</td>
<td>115.625m sec</td>
</tr>
</tbody>
</table>

e.g. If code 15 was used each selected channel would be scanned once every 500m sec.
Key in [E] to get

\[\text{SET} \]
\[\text{NO OF SCANS: ???} \]

Key in the required number of scans for the log (limits 1-9999). e.g. 240 is keyed in as [0], [2], [4], [0] and then [E] to get

\[\text{SET CHAN. TO BE DISPLAYED: #??} \]

The real-time logged data of the channel selected here is displayed on the LCD and (if the IEEE mode is "ON") is output to the IEEE-488 bus during the log.

Key in the required channel number (two digits) and then [E] to get.

\[\text{SET TRIGGER ALARM EXT MAN} \]

This procedure allows the user to select one of three alternative ways to trigger a log:
1. [ALARM] When the [ALARM] key is pushed the program displays:

\[\text{SET TIME ALARM HHMMSS} \]

Key in the required log start time (24 hour clock) and then push [E] to return to the main display which will look as follows.

\[\text{01 MAR. 13:00:00} \]
\[\text{SET LOG-A TEST} \]

[LOG-A] can be used to display the ALARM time as follows:

\[\text{LOG STARTS AT 12:34 EXIT} \]

Push [EXT] to return to the main display.

Note: A log is started at the Alarm time and every 24 hours thereafter.
2. **[EXIT]** - This key is pushed to select the external trigger mode. Once it has been selected the program returns to the main display as follows:

```
01 MAR 13:00:00
SET LOG-X TEST
```

The program now allows a log to start every time the external contact points are shorted together.

**[LOG-X]** can be used to manually override the external trigger from the following display:

```
X-TRIG OVERRIDE?
NO YES EXIT
```

[NO] or [EXIT] allow the program to return to the main display.
[YES] starts a log and returns to the main display upon completion.

3. **[MAN]** - When this key is selected the program returns to the main display.

```
01 MAR 13:00:00
SET LOG-M TEST
```

In this mode a log is started every time **[LOG-M]** is pushed.

7.4. **Data Acquisition Test and Calibration:** Ensure that nothing is connected to the Data Acquisition port. From the main display select [TEST] to get

```
"TEST MODE"
D-ACQ IEEE EXIT
```

Push [D-ACQ] to get

```
CHAN # ?? -5000mV
ENTRY EXIT
```

Note: If instead of -5000mV the display is blank, then the IEEE mode is ON and no IEEE listener is connected to the IEEE port. Either the IEEE mode must be switched off (use [EXIT] and then 7.3 above) or an IEEE listener must be provided.
Select channel 1 by either using the [←], [→] or [ENTRY] keys. If [ENTRY] is pushed then the following will be displayed.

\[
\text{SET CHAN. TO BE DISPLAYED: \# ??}
\]

The key sequence [0], [1] and then [E] will result in the following display.

\[
\text{CHAN \# 01 -5000mV ENTRY EXIT}
\]

If a known voltage (between \(\pm 5\) V) is now applied between the CH1 and ground pins (see Fig 7) on the Data Acquisitions connector this voltage should be displayed (\(\pm 3\)mV) on the LCD in place of -5000mV.

Any channel can be selected and the above process repeated to ensure that all the Data Acquisition electronics are working correctly.

If the displayed voltages are incorrect follow the calibration procedure set out below:

Data Acquisition board Calibration:

1. Select Test Mode: [D-ACQ], CHAN \# 01.
2. The display should read -5000mV with the input open circuit (no connection to the Data Acquisition connector).
3. Short CH1 input to Analog input ground. (pin 20 to pin 17 of Data Acquisition connector). Display should now read approximately 0000mV.
4. Check \(-15\) V rail for \(-15,000\) V using a high input impedance DVM (FLUKE 8502A or similar.) and analog ground as reference. Adjust VR1 on the P.S.U. board if necessary.
5. Connect the DVM between analog ground and the "link" between S&H and A/D on the Data Acquisition board.
6. Adjust the S&H pot. (VR3) for 0000mV on the DVM.
7. Remove short on input and attach a stable, adjustable DC voltage source between CH1 input and analog ground (pin 20 and 17 resp.)
8. Set input voltage to \(-49988\)V and adjust VR2 for a transition on the Data Logger LCD from \(-4998\) to -5000mV.
9. Set input voltage to \(+49955\)V and adjust VR1 for a transition on the LCD from 4994 to 4997mV.
10. Do some random checks with various input voltage for confidence.
Note: The resolution of the A/D converter is 2.44mV per bit for the Data Logger.

To Return to the main display push [EXIT].

7.5. IEEE-488 Port Test

From the main display push [TEST] to get

"TEST MODE"
D-ACQ IEEE EXIT

Push [IEEE]. This program immediately outputs from 0 - 255 to the IEEE-488 port before returning to the main display. This test can be used to establish whether the Data Logger / IEEE-488 computer link is correctly made. (see Appendix E)

7.6. Cassette loading in Record Mode

If a cassette is being loaded for the first time or if "LOAD CASSETTE" is not at present being displayed then the following procedure must be used to load and initialize a cassette.

Follow the Set Logger Parameters procedure (7.3) until

TAPE MODE SET
REC PLAY OFF

With no cassette in the tape transport push [PLAY]. "LOAD CASSETTE" will now be displayed.

Insert a Data cassette (with record tabs intact).

TAPE INITIALIZE
IN PROGRESS

Will now be displayed until initialization is complete and then the program will return to the main display to await the first log trigger.
7.7. Tape Transport Prompts:

In the record mode the program will attempt to dump data from the Buffer memory to tape at the end of each log. If a tape transport error occurs at this point one of the following tape error messages will be displayed:

"END OF TAPE!" - requires the operator to remove the tape at which time "LOAD CASSETTE!" will be displayed.

"LOAD CASSETTE!" - requires the operator to insert a tape (either new or side B of current tape) into the tape drive.

TAPE INITIALIZE
IN PROGRESS

will now be displayed until initialization is complete and then the program will return to the main display to await the next log trigger.

"WRITE PROTECTED!" - requires the operator to replace the write protect tab or replace the tape before the program will continue. When the tape is removed from the tape transport "LOAD CASSETTE!" will be displayed.

7.8. Data Playback

The Data Logger must be connected to a computer (HP85 or HP9816) via the IEEE-488 port for data playback. The computer program must be waiting to receive the data before the [READ] key is pushed (see below) to initiate the data transfer sequence.

To select the Tape playback mode the Set Logger Parameters procedure (7.3) should be followed:
From the main display select [SET], [LOG], [ON] to get

TAPE MODE SET
REC PLAY OFF

Insert the pre-recorded Data Cassette and push [PLAY]

TAPE INITIALIZE
IN PROGRESS

will now be displayed until the tape is rewound and then

NEXT RECORD #01
READ ENTER EXIT

is displayed.

The record number can be incremented or decremented by the [\-\-], [\+\+] or [ENTER] keys.

When the desired record is displayed as the "Next Record" push [READ]. The data header and then the data will be output to the IEEE-488 port. Once the data transfer is complete the Record number is incremented by one and the display reverts to

NEXT RECORD #02
READ ENTER EXIT

The [EXIT] key returns you to the main display.
8. **SERVICING**

The Data Logger requires a minimal amount of servicing. This is limited to tape head cleaning at regular intervals (see MFE Manual) and recalibration of the data acquisition circuitry when deemed necessary. (see section 7.4).
9. SOFTWARE PROBLEMS AND ENHANCEMENTS

9.1. Problems

Initially, when the Data Logger was put into service a problem was discovered with the software (version V1.1). The problem was that the program would randomly "hang-up" as a log was triggered. This seemed to be caused by the external trigger interrupt occurring while the program was processing an "IF" statement. The actual cause was never found but would appear to be a peculiarity of the PL/M compiler.

The software was modified (Version V1.2) to get round this problem by masking out all interrupts during the sensitive part of the program. This is the current software version used in the IMT General Purpose Data Logger.

9.2. Enhancements

In the particular application for which the IMT Data Logger was originally designed the external log trigger was provided by a microwave doppler-shift movement detector. This device sometimes produced false triggers which resulted in large amounts of unwanted "data" being logged.

To overcome this problem the software was modified to include a qualify routine to determine whether the external trigger coincided with a signal before dumping to tape. (Version V1.3).

This was achieved by allowing all external triggers to initiate a log and then performing a "qualifying" test on the data in the Buffer memory.

The data of a pre-selected channel is passed through a software low-pass filter (see 9.3) and then the filtered minimum and maximum signal values are determined. The difference between maximum and minimum is then compared to a preset threshold value and only if the threshold value is exceeded is the data dumped to tape.

The above enhancement worked well but, because the external trigger device was range and velocity sensitive, the log trigger was sometimes too early and at other times too late to start a log. This resulted in the wanted signal being truncated. To overcome this problem it was decided to do away with the external trigger device and to design a software threshold trigger which would use the actual data signal to initiate a log. (Version V1.4).
The operation of the software trigger is similar to that of a digital oscilloscope with provision being made for selecting the threshold level (in mV) as well as the pre-trigger length (in samples) from the keyboard.

The software trigger is implemented by allowing the data logger to be continuously in the log mode. The logged data is stored in a revolving buffer memory which is set up to be a specific size determined by the number of scans and the number of channels selected. Data is stored in consecutive memory locations starting at the beginning of the buffer. A buffer-pointer is used to keep track of the next location to be filled and when the buffer is completely full the buffer-pointer is reset to 0.

It can be seen that this method allows the buffer to always contain "history" data equal to one complete log after one initial log period has been completed.

The pre-trigger function allows a selectable amount of this "history" data to be included at the beginning of a log so that the log data contains the complete signal and not only that portion after the threshold trigger has occurred.

A test is done at the end of each scan to determine if the pre-selected threshold on the specified channel has been reached. If it has, then a log is started with the pre-trigger data included at the beginning.

The trigger reference voltage, in this case, is not fixed but is a filtered version of the ambient background signal on the specified channel. A dynamic reference was chosen because the input signal d.c. level drifts due to its sensitive electronics and other background physical effects. To allow for this variation, but still have a constant threshold level, a software filter was designed to smooth the reference voltage.

9.3. Software Filter Design:

The filter is a first order Recursive low-pass filter of the form

\[ y(n) = ax(n) + by(n - 1) \]
or diagramatically

![Diagram](image)

The general transfer function for the above is:

\[ H(z) = \frac{a}{1 - b z^{-1}} \]

For a - 3dB cut off frequency \( f_o \)
and sampling interval \( T \)

\[ b = e^{-2\pi f_o T} \text{ for } b < 1 \quad \ldots \quad (1) \]

let \( \omega_0 = 2\pi f_o \) and due to the Sampling Theorem \( f_o \leq \frac{1}{2T} \)

or \( \omega_0 T \leq \pi \)

substituting in (1) \[ b = e^{-\omega_0 T} \]

Note: for unity gain \( a + b = 1 \)

For a required \( f_o \) of \( 1 \times 10^{-3} \)Hz with \( T = 500 \) msec.

\[ b = 0.9969 \]
\[ a = 0.0031 \]

but to simplify the software arithmetic the following values for \( a \) and \( b \) were chosen

\[ b = 0.99; \ a = 0.01 \]

Working back \( f_o = 1.39 \times 10^{-3} \) Hz
This simplification allowed 16 bit integer arithmetic to be used and gave satisfactory results.

Briefly, the filter allows 1% of the most recent input signal to be summed with 99% of the history signal to provide a filtered output. The output takes up the value of the slowly changing d.c. offsets but filters out rapid fluctuations due to signal or noise.
10. CONCLUSION

A General Purpose Data Logger has been successfully designed, developed and built that meets the original system requirements. In several aspects it is more sophisticated than the original requirement called for but this increases its flexibility in its general purpose role.

Two Data Loggers have been in service for over a year, with one in continuous twenty four hour use. They have proved to be extremely reliable with only one component failure occurring during this time even though the remote unit is sited in a very hostile environment.

The operation of the remote instrument has been successfully undertaken by non-technical personnel who are required to change the cassette tapes when necessary and to occasionally reset the logger after a long power cut or an excessive power line surge or glitch.

The time and effort spent on designing modular electronics and software has been justified by the fact that, to date, six further projects have made use of the 8085 CPU board and at least half of these have also used the IEEE-488 and/or Data Acquisition boards. These projects would most probably not have been undertaken if not for the availability and built in flexibility of the hardware/software.

The experience and knowledge gained by undertaking this project cannot be quantified but will, I am sure, contribute significantly towards future IMT project goals.
REFERENCES

5. Hewlett Packard HP85 Computer Manuals
APPENDIX A

Software Version V1.2
Program listings
I{)
co
In
o-

/*
 * DATA LOGGER MAIN MODULE
 */

DATA_LOGGER:
DO:
DECLARE DEC LITERALLY 'DECLARE';
DEC LIT LITERALLY 'LITERALLY';
DEC LEN LIT 'LENGTH';
DEC CLEAR#DISP LIT '01H';
DEC RETURN#HOME LIT '02H';
DEC HOME#2 LIT '00H';
DEC CURSOR#OFF LIT '0CH';
DEC CURSOR#OFF LIT '0CH';
DEC MESS1 (**) BYTE DATA (0F3H,'SET',0F3H);
DEC MESS2 (**) BYTE DATA (0F3H,'LOG',0F3H);
DEC MESS3 (**) BYTE DATA ('TEST ');
DEC MESS4 (**) BYTE DATA ('TIME',0F3H);
DEC MESS5 (**) BYTE DATA ('EXIT ');
DEC MESS6 (**) BYTE DATA ('SET MODE');
DEC MESS7 (**) BYTE DATA ('IEEE ');
DEC MESS8 (**) BYTE DATA ('D-ACO');
DEC MESS9 (**) BYTE DATA ('TEST MODE');
DEC MESS10 (**) BYTE DATA ('LOG- ');
DEC MESS11 (**) BYTE DATA ('JANFEBMARAPRMIYJUNJULAugSEPOTNOVDEC');
DEC (FLAG1,KEY) BYTE EXTERNAL;
DEC RTC#DATE(6) BYTE EXTERNAL;
DEC RTC#TIME(6) BYTE EXTERNAL;
DEC (POSITION) BYTE EXTERNAL;
DEC (MONTH BYTE EXTERNAL;
DEC TRIG#TYPE BYTE EXTERNAL;
DEC TAPE#MODE#FLAG BYTE PUBLIC AT (2016H); /* 0=RECORD,1=PLAY,2=OFF */
DEC IEEE#MODE#FLAG BYTE PUBLIC AT (2017H); /* 0=OFF,1=ON */
DEC RTI##55#FLAG BYTE EXTERNAL;
DEC TAPE#ERROR#FLAG BYTE EXTERNAL;
DEC (I,T) BYTE;
SINASK: PROCEDURE (MASK) EXTERNAL;
DEC MASK BYTE;
END SINASK;
RINASK: PROCEDURE BYTE EXTERNAL;
44 2 END REMASK;
45 1 KEY*INIT: PROCEDURE EXTERNAL;
46 2 END KEY*INIT;
47 1 RTC*INIT: PROCEDURE EXTERNAL;
48 2 END RTC*INIT;
49 1 INST*OUT: PROCEDURE (INST) EXTERNAL;
50 2 DEC INST BYTE;
51 2 END INST*OUT;
52 1 C*OUT: PROCEDURE (CHAR) EXTERNAL;
53 2 DEC CHAR BYTE;
54 2 END C*OUT;
55 1 TEXT*OUT: PROCEDURE (P4, SIZE) EXTERNAL;
56 2 DEC P4 ADDRESS;
57 2 DEC SIZE BYTE;
58 2 END TEXT*OUT;
59 1 LCD*INIT: PROCEDURE EXTERNAL;
60 2 END LCD*INIT;
61 1 KEY*ENTRY: PROCEDURE EXTERNAL;
62 2 END KEY*ENTRY;
63 1 SET*LOG*PARAMETERS: PROCEDURE EXTERNAL;
64 2 END SET*LOG*PARAMETERS;
65 1 TAPE*PLAY: PROCEDURE EXTERNAL;
66 2 END TAPE*PLAY;
67 1 SET*DATE*TIME: PROCEDURE (A) EXTERNAL;
68 2 DEC A BYTE;
69 2 END SET*DATE*TIME;
70 1 RTC*DATA: PROCEDURE (A) EXTERNAL;
71 2 DEC A BYTE;
72 2 END RTC*DATA;
73 1 MENU: PROCEDURE (F1, P2, P3) EXTERNAL;
74 2 DEC (F1, P2, P3) ADDRESS;
75 2 END MENU;
76 1 TAPE*INIT: PROCEDURE EXTERNAL;
77 2 END TAPE*INIT;
78 1 D#AC0*INIT: PROCEDURE EXTERNAL;
79 2 END D#AC0*INIT;
80 1 D#AC0*TEST: PROCEDURE EXTERNAL;
81 2 END D#AC0*TEST;
82 1 IEEE*X88*INIT: PROCEDURE EXTERNAL;
83 2 END IEEE*X88*INIT;
01/05/85 PAGE 3

IEEE$480$TEST: PROCEDURE EXTERNAL;
END IEEE$480$TEST;

LOG$TRIG: PROCEDURE EXTERNAL;
END LOG$TRIG;

LOG$SCAN: PROCEDURE EXTERNAL;
END LOG$SCAN;

TAPE$ERROR: PROCEDURE EXTERNAL;
END TAPE$ERROR;

TEST: PROCEDURE;
CALL INST$OUT(CLEAR#DISP); /* ADVANCE CURSOR TO POSITION 2 */
CALL INST$OUT(62H);
CALL TEXT$OUT(MES$21,LEN(MES$21)); /* TEST MODE */
CALL MENU(MES$20,MES$17,MES$6);
KEY=0FH;
DO WHILE KEY>2;
   FLAGI=0;
   CALL KEY$ENTRY;
   END;
   DO CASE KEY;
   CALL D$ACO$TEST;
   CALL IEEE$480$TEST;
   END;
   END TEST;

SET$LOGGER$MODE: PROCEDURE;
CALL INST$OUT(CLEAR#DISP);
CALL INST$OUT(81H);
CALL TEXT$OUT(MES$17,LEN(MES$17)); /* IEEE */
CALL TEXT$OUT(MES$29,LEN(MES$29)); /* MODE SET */
CALL MENU(MES$32,MES$33,BLANK); /* OFF ON */
KEY=0FH;
DO WHILE KEY>1;
   FLAGI=0;
   CALL KEY$ENTRY;
   END;
   IEEE$MODE$FLAG$KEY;
   CALL INST$OUT(CLEAR#DISP);
   CALL INST$OUT(81H);
   CALL TEXT$OUT(MES$27,LEN(MES$27)); /* TAPE */
   CALL TEXT$OUT(MES$29,LEN(MES$29)); /* MODE SET */
   CALL MENU(MES$30,MES$31,MES$32); /* REC PLAY OFF */
   KEY=0FH;
   DO WHILE KEY>2;
   FLAGI=0;
   CALL KEY$ENTRY;
   END;
   IEEE$MODE$FLAG$KEY;
   IF TAPE$MODE$FLAG=1 THEN
   CALL SET$LOGGER$PARAMETERS;
   END SET$LOGGER$MODE;
187  4    IF (I = 2) OR (I = 4) THEN
188  4    DO;
189  5    CALL C*OUT(':');
190  5    CALL C*OUT(RTC*TIME(I)); /* 0/P TIME */
191  5    END;
192  4    ELSE
193  4    CALL C*OUT(RTC*TIME(I));
194  4    END;
195  3    T = R#MASK;
196  3    IF (T AND 40H) = 40H THEN /* ON RST 7.5 */
197  3    KEY=INPUT(028H) AND 0FH;
198  3    IF (T AND 10H) = 10H THEN /* ON RST 5.5 */
199  3    RST#55#FLAG = OFFH;
200  3    END;
201  2    CALL #MK50(01AH);
202  2    DO CASE(KEY);
203  3    CALL SET#MODE;
204  3    CALL LOG#TRIG;
205  3    CALL TEST;
206  3    CALL LOG#SCAN;
207  2    END MAIN;

/* MAIN PROGRAM */

208  1    CALL LCD#INIT;
209  1    CALL KEY#INIT;
210  1    CALL RTC#INIT;
211  1    CALL TAPE#INIT;
212  1    CALL DR#ACO#INIT;
213  1    CALL IEEE#488#INIT;
214  1    TAPE#ERROR#FLAG=0;
215  1    RST#55#FLAG=0;
216  1    DO I=0 TO 4;
217  2    MES#22(I) = MES#22#INIT(I);
218  2    END;
219  1    DO WHILE 1;
220  2    CALL MAIN;
221  2    CALL #MK50(01AH); /* ENABLE 7.5/5.5,RESET 7.5 F/F */
222  2    END;
223  1    END DATA#LOGGER;

MODULE INFORMATION:

CODE AREA SIZE = 03EOH  992D
VARIABLE AREA SIZE = 0009H  9D
MAXIMUM STACK SIZE = 000AH 10D
259 LINES READ
0 PROGRAM ERROR(S)

END OF PL/M-80 COMPILATION
/* KEYBOARD MODULE */

1 KEYBOARDMODULE:
2   DO;
3      DECLARE KEYBOARD LITERALLY '028H'; /* KEYBOARD PORT ADDRESS */
4      DECLARE SLDCURSOR LITERALLY '10H';
5      DECLARE SRDCURSOR LITERALLY '14H';
6      DECLARE KEY#ASCII BYTE PUBLIC; /* DESIGNATED ASCII VALUE OF LAST KEY DEPRESSED */
7      DECLARE FLAGI BYTE PUBLIC; /* RST 7.5 INTERRUPT FLAG (SET=FFH,CLEAR=0H) */
8      DECLARE KEY BYTE PUBLIC; /* BINARY VALUE OF LAST KEY DEPRESSED */
9      DECLARE CHAR (16) BYTE EXTERNAL;
10     S#MASK: PROCEDURE (MASK) EXTERNAL;
11     DECLARE MASK BYTE;
12     END S#MASK;
13      KEY#INIT: PROCEDURE PUBLIC;
14      DECLARE JUMP BYTE AT (03CH);
15      DECLARE VECTOR ADDRESS AT (03DH);
16      JUMP = 03CH; /* JMP OF CODE */
17      VECTOR = .RST#75; /* INIT. RST7.5 JUMP VECTOR */
18      CALL S#MASK(01BH);
19      END KEY#INIT;
20      INSTOUT: PROCEDURE (INSTRUCTION) EXTERNAL;
21      DECLARE INSTRUCTION BYTE;
22      END INSTOUT;
23      C#OUT: PROCEDURE (CHARACTER) EXTERNAL;
24      DECLARE CHARACTER BYTE;
25      END C#OUT;
26      KEY#ENTRY: PROCEDURE PUBLIC;
27      ENDABLE;
28      DO WHILE FLAGI = 0H; /* WAIT FOR RST7.5 INTERRUPT */
29      END;
30      FLAGI = 0H; /* KEY DEPRESSION DETECTED */
31      END KEY#ENTRY;
32      RST#75: PROCEDURE;
33      DISABLE;
34      KEY = INPUT (KEY#BOARD) AND 0FH; /* MASK OFF 4 LSB'S OF I/P VALUE */
35      DO CASE (KEY);
36      KEY#ASCII = 'G'; /* ALLOCATE AN ASCII VALUE TO "KEY#ASCII" */
37      KEY#ASCII = 'B'; /* DEPENDING ON VALUE OF "KEY" */
38      KEY#ASCII = 'E';
39      END CASE (KEY);
40   END RST#75;
MODULE INFORMATION:

CODE AREA SIZE = 0035H 1970
VARIABLE AREA SIZE = 0000H 130
MAXIMUM STACK SIZE = 0002H 20
40 LINES READ
0 PROGRAM ERROR(S)
```plaintext
ISIS-II PL/MI-80 V3.1 COMPILATION OF MODULE DISPLAYMODULE
OBJECT MODULE PLACED IN :F1:DSMOD.OBJ
COMPILER INVOKED BY: PLMI80 :F1:DSMOD.SRC DEBUG PRINT (:F1:DSMOD.LST) DATE(14/01/85) PAGELENGTH(59)

/***************************************************************/
/* DISPLAY LANGUAGE */
/***************************************************************/

DISPLAYMODULE:
1  DO;
2  DECLARE CLEAR#DISP LITERALLY '01H';
3 DECLARE RETURN#HOME LITERALLY '02H'; /* MOVE CURSOR TO LINE 1, POSITION 1 */
4 DECLARE INST#REG LITERALLY '030H';
5 DECLARE D#REG LITERALLY '031H';
6 DECLARE ENTRY#MODE#INC LITERALLY '036H';
7 DECLARE CURSOR#ON LITERALLY '06H';
8 DECLARE CURSOR#OFF LITERALLY '0CH';
9 DECLARE TWO#LINE LITERALLY '038H';
10 DECLARE W#WORD#LENGTH LITERALLY '030H';
11 DECLARE BLINK LITERALLY '0FH';
12 DECLARE SL#CURSOR LITERALLY '10H';
13 DECLARE SR#CURSOR LITERALLY '14H';
14 DECLARE HOME#2 LITERALLY '00CH'; /* MOVE CURSOR TO LINE 2, POSITION 1 */
15 DECLARE CHAR (16) BYTE PUBLIC; /* TEMP. STRING STORAGE */
16 DECLARE (POSITION) BYTE PUBLIC; /* CURRENT CURSOR POSITION */
17 DECLARE DopROCE;URE;
18 DECLARE FLAG BYTE;
19 DECLARE 1S Disable:
20 DECLARE 1S OFFH;
21 DECLARE WHILE ((FLAG > 0H); /* WAIT WHILE LCD PROCESSOR BUSY */
22 DECLARE 1S INPUT(INST#REG) AND 080H;
23 DECLARE 1S END;
24 DECLARE 1S POSITION = INPUT(INST#REG) AND 07FH; /* UPDATE (CURSOR) POSITION */
25 DECLARE 1S ENABLE;
26 DECLARE EndLcoPROCE;URE;
27 DECLARE INST#OUT(1S INST) BYTE PUBLIC;
28 DECLARE INST BYTE;
29 DECLARE OUTPUT(INST#REG) = INST; /* 0/P 1 INSTRUCTION TO LCD PROCESSOR */
30 DECLARE CALL LCD#BUSY;
31 DECLARE 1S END INST#OUT;
32 DECLARE LCD#INIT:PUBLIC;
33 DECLARE 1S BYTE;
34 DECLARE LCD#INIT#CONDITIONS(*)(*) BYTE DATA (W#WORD#LENGTH,W#WORD#LENGTH,TWO#LINE,
CURSOR#ON,ENTRY#MODE#INC,CLEAR#DISP);
35 DECLARE DO I=0 TO LAST(LCD#INIT#CONDITIONS);
36 DECLARE CALL INST#OUT (LCD#INIT#CONDITIONS(I));
37 DECLARE END;
38 DECLARE 1S END LCD#INIT;
39 DECLARE CHAR#OUT:PUBLIC; /* 0/P 1 CHARACTER TO LCD AT PRESENT */
40 DECLARE CHAR BYTE; /* CURSOR POSITION & INC "POSITION" */
```
PL/M-80 COMPILER

41  2  OUTPUT(D$REG) = CHAR;
42  2  CALL LCD$BUSY;
43  2  END C$OUT;

44  1  TEXT#OUT: PROCEDURE(P4,SIZE) PUBLIC;
45  2  DECLARE P4 ADDRESS;
46  2  DECLARE (SIZE,1) BYTE;
47  2  DECLARE (TEXT#4 BASED P4)(1) BYTE;
48  2  DO I=0 TO (SIZE-1);
49  3  CALL C$OUT(TEXT#4(I)));
50  3  END;
51  2  END TEXT#OUT;

52  1  MENU: PROCEDURE(P1,P2,P3) PUBLIC;
53  2  DECLARE (P1,P2,P3)ADDRESS;
54  2  DECLARE (TEXT1 BASED P1)(5) BYTE;
55  2  DECLARE (TEXT2 BASED P2)(5) BYTE;
56  2  DECLARE (TEXT3 BASED P3)(5) BYTE;
57  2  DECLARE (I,N) BYTE;
58  2  CALL INST#OUT(HOME$2);
59  2  I,N=0;
60  2  DO N=0 TO 2;
61  3  DO I=0 TO 4;
62  4  DO CASE(N);
63  5  CALL C#OUT(TEXT1(I)));
64  5  CALL C#OUT(TEXT2(I)));
65  5  CALL C#OUT(TEXT3(I)));
66  5  END;
67  4  END;
68  3  CALL C#OUT(0F3H);
69  3  END;
70  2  END MENU;

71  1  END DISPLAY#MODULE;

MODULE INFORMATION:

CODE AREA SIZE = 0127H 295D
VARIABLE AREA SIZE = 0021H 33D
MAXIMUM STACK SIZE = 0004H 4D
84 LINES READ
0 PROGRAM ERROR(S)

END OF PL/M-80 COMPILATION
REAL-TIME CLOCK MODULE

DECLARE CLEARDISP LITERALLY '01H';
DECLARE DISPCURSORON LITERALLY '0EH';
DECLARE BLINK LITERALLY '0F'H;
DECLARE HOME2 LITERALLY '0CH';
DECLARE HES14 BYTES DATA ('ALARM');
DECLARE DATEMENU BYTES DATA ('SET DATE DDMMYY');
DECLARE TIMEHMENU BYTES DATA ('SET TIME HHMMSS');
DECLARE (FLAG1,KEY) BYTE EXTERNAL;
DECLARE RTCDATE BYTES PUBLIC;
DECLARE RTCTIME BYTES PUBLIC;
DECLARE (N) BYTE;
DECLARE CHAR BYTES EXTERNAL;
DECLARE POSBYTE EXTERNAL;
DECLARE TEMP BYTE;
DECLARE SECONDS BYTE AT (2000H);
DECLARE MINUTES BYTE AT (2002H);
DECLARE HOURS BYTE AT (2004H);
DECLARE DAYOFWEEK BYTE AT (2006H);
DECLARE DAYOFMONTH BYTE AT (2007H);
DECLARE MONTH BYTE PUBLIC AT (2008H);
DECLARE YEAR BYTE AT (2009H);
DECLARE SECONDS ALARM BYTE AT (2001H);
DECLARE HOURS ALARM BYTE AT (2003H);
DECLARE REGA BYTE PUBLIC AT (200AH);
DECLARE REGB BYTE PUBLIC AT (200BH);
DECLARE REGC BYTE AT (200CH);
DECLARE REGD BYTE AT (200DH);
DECLARE RST55 FLAG BYTES PUBLIC;  /* CLOCK INTERRUPT FLAG */
DECLARE SCANRATE BYTE EXTERNAL;

DECLARE INST BYTES EXTERNAL;

DECLARE INST BYTE;
END INST;
KEY*IN:  PROCEDURE EXTERNAL;
END KEY*IN;

RST*55:  PROCEDURE PUBLIC;
DECLARE C BYTE;
C = REG#C;
RST*55#FLAG=0FFH;
ENABLE;
END RST*55;

RTCINIT:  PROCEDURE PUBLIC;
DECLARE JUMP*55 BYTE AT (02CH);
DECLARE VECTOR*55 ADDRESS AT (02DH); /* INITIALIZE RST 5.5 JUMP VECTOR */
JUMP*55 = 03H; /* JUMP OP CODE */
VECTOR*55 = RST*55;
REG#A = SCAN*RATE AND 0FH; /* INIT SCAN RATE (PIE RATE) */
REG#B = 02H; /* PIE DISABLE */
CALL $*MASK(01AH); /* SET INTERRUPT MASK (ENABLE 7.5 & 5.5, RESET 7.5 F/F) */
END RTCINIT;

RTC*INIT*DATA:PROCEDURE(A); /* SETS RTC INITIAL DATA */
DECLARE A BYTE;
DO I=0 TO 4 BY 2;
  TEMP = (CHAR(I+1) AND 0FH) OR SHL((CHAR(I) AND 0FH),4); /* CONVERT 2 ASCII CHARs. */
  /* TO PACKED BCD */
  IF A=1 THEN /* IF A=1 THEN SET ALARM */
  DO CASE (I+1);
  1: HOURS*ALARM = TEMP;
  2: MINUTES*ALARM = TEMP;
  3: SECONDS*ALARM = TEMP;
  END;
  ELSE
  DO CASE (I+1);
  1: DAYOFMONTH = TEMP;
  2: HOURS = TEMP;
  3: MONTH = TEMP;
  4: MINUTES = TEMP;
  5: YEAR = TEMP;
  6: SECONDS = TEMP;
  END;
END;
END RTC*INIT*DATA;

SET*DATE*TIME: PROCEDURE(A) PUBLIC; /* A=1 FOR ALARM SET */
DECLARE A BYTE;
IF A=0 THEN
DO:
  REG#B = 10000010B; /* SET,BCD,24 HR */
  REG#A = 00H; /* INITIALIZE REG A */
  END;
  DO N = 0 TO 1;
  IF A=1 THEN N=1;
  CALL INST*OUT(CLEAR*DISP); /*SET ONLY TIME IN ALARM MODE!! */
DO CASE A;
  CALL INST#OUT(04H);
  CALL INST#OUT(01H);
END;
DO CASE N;
  CALL TEXT#OUT(.DATE#MENU,9); /* SET DATE */
  CALL TEXT#OUT(.TIME#MENU,9); /* SET TIME */
END;
DO CASE A;
  CALL TEXT#OUT(.HES#14,5); /* ALARM */
END;
  CALL INST#OUT(HOME#2 +S); /* MOVE CURSOR TO LINE 2 POSITION 5 */
DO CASE N;
  CALL TEXT#OUT(.DATE#MENU(9),6); /* DDMMYY */
  CALL TEXT#OUT(.TIME#MENU(9),6); /* HHMMSS */
END;
  CALL INST#OUT((POSITION=6)OR 80H); /* MOVE CURSOR ~6 PLACES */
  CALL INST#OUT(BLINK); /* BLINK ON */
  CALL KEY#IN;
  CALL RTC#INIT#DATA(A);
END;
  REG#B = 000000010B; /* GET OUT OF CLOCK SET MODE */
  CALL INST#OUT(DISPLAY#CURSOR#ON); /* BLINK OFF */
  CALL INST#OUT(CLEAR#DISP);
END SET#DATE#TIME;

RTC#DATA:PROCEDURE(A) PUBLIC;
DECLARE (1,N,A) BYTE;
DECLARE ASCII (6) BYTE;
DISABLE;
DO WHILE (REG#A AND 80H)<0;
END;
DO N = 0 TO 1;
DO I = 0 TO 4 BY 2;
IF A=1 THEN
  DO CASE (1+N);
  TEMP = 0;
  TEMP = HOURS$#ALARM;
  TEMP = 0;
  TEMP = MINUTES$#ALARM;
  TEMP = 0;
  TEMP = SECONDS$#ALARM;
END;
ELSE
  DO CASE (1+N);
  TEMP = DAY$OF#MONTH;
  TEMP = HOURS;
  TEMP = MONTH;
  TEMP = MINUTES;
  TEMP = YEAR;
  TEMP = SECONDS;
END;
ASCII(I) = SHR(TEMP,4) OR 30H; /* CONVERTS PACKED BCD TO 2 ASCII CHARACTERS */
ASCII(I+1) = (TEMP AND 0FH) OR 30H; /* */
END;
END REAL$\text{CLOCK}\$MODULE:

MODULE INFORMATION:

CODE AREA SIZE = 030H

INSTRUCTION+STACK SIZE = 004H

9750

250 LINES READ

0 PROGRAM ERROR(S)
.isSuccess status
1. DECLARE DEC LITERALLY 'DECLARE';
2. DEC LIT LITERALLY 'LITERALLY';
3. DEC LIT LITERALLY 'LENGTH';
4. DEC CLEAR=DISP LIT '01H';
5. DEC RETURN=HOME LIT '02H';
6. DEC HOUR2 LIT '00H';
7. DEC CURSOR=ON LIT '0EH';
8. DEC MESS1 (*) BYTE DATA (0F3H,'SET',0F3H);
9. DEC MESS2 (*) BYTE DATA (0F3H,'WH',0F3H);
10. DEC MESS3 (*) BYTE DATA ('EXIT');
11. DEC MESS4 (*) BYTE DATA ('NO OF CHANS:');
12. DEC MESS5 (*) BYTE DATA ('SCAN RATE:');
13. DEC MESS6 (*) BYTE DATA ('NO OF SCAN:');
14. DEC MESS7 (*) BYTE DATA ('SET CHAN. TO BE');
15. DEC MESS8 (*) BYTE DATA ('DISPLAYED:');
16. DEC MESS9 (*) BYTE DATA ('SET TRIGGER:');
17. DEC MESS10 (*) BYTE DATA ('ALARM:');
18. DEC MESS11 (*) BYTE DATA ('EXIT',0F3H);
19. DEC MESS12 (*) BYTE DATA ('LOG STARTS AT:');
20. DEC MESS13 (*) BYTE DATA ('NO ');
21. DEC MESS14 (*) BYTE DATA ('YES ');
22. DEC MESS15 (*) BYTE DATA ('X-TRIG OVERRIDE?');
23. DEC (FLAG1,KEY) BYTE EXTERNAL;
24. DEC CHAR (KEY) BYTE EXTERNAL;
25. DEC (POSITION) BYTE EXTERNAL;
26. DEC (TRIGGER) BYTE PUBLIC AT (2010H); /* M=ALARM,X=EXTERNAL,I=MANUAL */
27. DEC NOOF#CHANS BYTE PUBLIC AT (2011H);
28. DEC SCANRATE BYTE PUBLIC AT (2012H); /* 15m=500ms,14=250ms,13=125ms etc */
29. DEC NOOF#SCANS ADDRESS PUBLIC AT (2013H);
30. DEC DISP BYTE PUBLIC AT (2015H); /* CHANNEL TO BE DISPLAYED */
31. DEC BIN#DATA ADDRESS EXTERNAL;
32. DEC RTC#TIME:6 BYTE EXTERNAL;
33. DEC RST#S5#FLAG BYTE EXTERNAL;
34. DEC REG#B BYTE EXTERNAL;
35. INST#OUT: PROCEDURE (INST) EXTERNAL;
36. DEC INST BYTE;
37. END INST#OUT;
38. C#OUT: PROCEDURE (CHAR) EXTERNAL;
39. DEC CHAR BYTE;
40. END C#OUT;
42 1 TEXT#OUT: PROCEDURE(P4,SIZE) EXTERNAL;
43 2 DEC P4 ADDRESS;
44 2 DEC SIZE BYTE;
45 2 END TEXT#OUT;
46 1 BIN#TO#ASCII: PROCEDURE(READING,SIZE) EXTERNAL;
47 2 DEC SIZE BYTE;
48 2 DEC READING ADDRESS;
49 2 END BIN#TO#ASCII;
50 1 ASCII#TO#BIN: PROCEDURE(ASCII#ADR,SIZE) EXTERNAL;
51 2 DEC ASCII#ADR ADDRESS;
52 2 DEC SIZE BYTE;
53 2 END ASCII#TO#BIN;
54 1 KEY#ENTRY: PROCEDURE EXTERNAL;
55 2 END KEY#ENTRY;
56 1 KEY#IN: PROCEDURE EXTERNAL;
57 2 END KEY#IN;
58 1 SET#DATE#TIME: PROCEDURE(A) EXTERNAL;
59 2 DEC A BYTE;
60 2 END SET#DATE#TIME;
61 1 LOG#PARAM#2: PROCEDURE;
62 2 CALL TEXT#OUT(.CHAR,2);
63 2 CALL INST#OUT((POSITION-2) OR 80H);
64 2 CALL KEY#IN;
65 2 END LOG#PARAM#2;
66 1 LOG#PARAM#4: PROCEDURE;
67 2 CALL TEXT#OUT(.CHAR,4);
68 2 CALL INST#OUT((POSITION-4) OR 80H);
69 2 CALL KEY#IN;
70 2 END LOG#PARAM#4;
71 1 MENU: PROCEDURE(P1,P2,P3) EXTERNAL;
72 2 DEC (P1,P2,P3)ADDRESS;
73 2 END MENU;
74 1 RTC#DATA: PROCEDURE(A) EXTERNAL;
75 2 DEC A BYTE;
76 2 END RTC#DATA;
77 1 LOG#SCAN: PROCEDURE EXTERNAL;
78 2 END LOG#SCAN;
79 1 ALARM#DISP: PROCEDURE; /* DISPLAY ALARM TIME */
80 2 CALL INST#OUT(CLEAR#DISP);
81 2 CALL TEXT#OUT(.NES#18,LEN(NES#18)); /* LOG STARTS AT */
82 2 CALL RTC#DATA(1);
83 2 CALL INST#OUT(HOME#2);
84 2 CALL TEXT#OUT(.RTC#TIME(0),2);
85 2 CALL COUT('1');
86 2 CALL TEXT#OUT(.RTC#TIME(2),2);
CALL INST#OUT((POSITION +7) OR 80H);
CALL TEXT#OUT(MES#4,4);  /* EXIT */
END ALARM#DISP;

X#OVERRIDE:PROCEDURE;
CALL INST#OUT(CLEAR#DISP);
CALL TEXT#OUT(MES#26,LEN(MES#26));  /* X-TRIG OVERRIDE */
CALL MENU(MES#24,MES#25,MES#6);  /* NO YES EXIT */
END X#OVERRIDE;

LOG#TRIG:PROCEDURE PUBLIC;
KEY=OFFH;  /* CLEAR KEY */
IF TRIG#TYPE = 'A' THEN  /* ALARM TRIGGER */
DO;
CALL ALARM#DISP;
DO WHILE KEY<>2;  /* EXIT */
ENABLE;
DO WHILE (FLAG1=0) AND (RST#55=0);  /* WAIT FOR KEY OR ALARM INTERRUPT */
ELSE;
CALL LOG#SCAN;
ENDIF;

IF TRIG#TYPE = 'X' THEN  /* EXTERNAL TRIGGER */
DO;
CALL X#OVERRIDE;
DO WHILE KEY<>3;
FLAG1=0;
CALL KEY#ENTRY;
END;
DO CASE KEY;
CALL LOG#SCAN;  /* YES */
CALL LOG#SCAN;  /* EXIT */
CALL LOG#SCAN;  /* EXTERNAL START */
ENDIF;

IF TRIG#TYPE = 'M' THEN  /* MANUAL TRIGGER */
CALL LOG#SCAN;
ENDIF;

CHAN#DISP#SELECT: PROCEDURE PUBLIC;
CALL INST#OUT(CLEAR#DISP);
CALL INST#OUT(CURSOR#ON);
CALL TEXT#OUT(MES#11,LEN(MES#11));  /* SET CHAN, TO BE */
CALL INST#OUT(NAME#2);
CALL TEXT#OUT(MES#12,LEN(MES#12));  /* DISPLAYED */
CALL CR#OUT('');
CALL CR#OUT('
');
CALL BINTO#ASCII(DISP,2);
CALL LOG#Para#2;
CALL ASCII#TOBIN(.CHAR,2);
DISP=BIN#Data;
END CHAN#DISP#SELECT;

SET#LOG#PARAMETERS: PROCEDURE PUBLIC;
DEC 1 BYTE;
DO I=0 TO 2;
CALL INST#OUT(CLEAR#DISP);
CALL INST#OUT(8SH);
CALL TEXT#OUT(.MES#1,LEN(MES#1)); /* SET */
CALL INST#OUT(HOME#2);
DO CASE 1;
DO;
CALL TEXT#OUT(.MES#8,LEN(MES#8)); /* NO OF CHANS */
CALL BIN#TO#ASCII(NO#OF#CHANS,2);
CALL LOG#PAR#H#2;
CALL ASCII#TO#BIN(.CHAR,2);
NO#OF#CHANS#B#DATA;
END;
DO;
CALL TEXT#OUT(.MES#9,LEN(MES#9)); /* SCAN RATE */
CALL BIN#TO#ASCII(SCAN#RATE,2);
CALL LOG#PAR#H#2;
CALL ASCII#TO#BIN(.CHAR,2);
SCAN#RATE#B#DATA;
END;
DO;
CALL TEXT#OUT(.MES#10,LEN(MES#10)); /* NO OF SCANS */
CALL BIN#TO#ASCII(NO#OF#SCANS,4);
CALL LOG#PAR#H#4;
CALL ASCII#TO#BIN(.CHAR,4);
NO#OF#SCANS#B#DATA;
END;
END;
END CASE (KEY);
DO;
TRI#TYPE=('A'); /* ALARM TRIGGER */
CALL SET#DATE#TIME(1);
RST#5#5#FLAG=0;
REG#B=2H;
* ENABLE ALARM *
END;
TRI#TYPE=('X'); /* EXTERNAL TRIGGER */
TRI#TYPE=('M'); /* MANUAL TRIGGER */
END;
END SET#LOG#PARAMETERS;
END SET#LOG#PARAMETERS#MODULE;
MODULE INFORMATION:

CODE AREA SIZE = 0325H 905D
VARIABLE AREA SIZE = 0001H 1D
MAXIMUM STACK SIZE = 0006H 4D
215 LINES READ
0 PROGRAM ERROR(S)

END OF PL/M-80 COMPILATION
**PL/M-80 Compiler**

ISIS-II PL/M-80 V3.1 Compilation of Module IEEE488 MODULE
OBJECT MODULE PLACED IN :F1:IEEE.OBJ
COMPILER INVOKED BY: PLM80 :F1:IEEE.SRC DEBUG PRINT(:F1:IEEE.LST) DATE(14/01/85) PAGELENGTH(59)

```plaintext
/*
** IEEE 488 MODULE
** TALK/LISTEN MODE
*/

1 IEEE#488#MODULE:
   DO:
   2 1 DECLARE DEC LITERALLY 'DECLARE';
   3 1 IEEE#488#INIT: PROCEDURE PUBLIC;
   4 1 2 OUTPUT(63H)=0C2H; /* INIT 8255 */
   5 1 2 OUTPUT(62H)=01H; /* INIT 74LS245 VIA PORT C */
   6 1 END IEEE#488#INIT;
   7 1 IEEE#488#TEST: PROCEDURE PUBLIC; /* OUTPUTS FROM 0 TO 255 CONSECUTIVELY */
   8 1 2 DEC I BYTE;
   9 1 2 DO I=0 TO 255H;
   10 1 3 OUTPUT (60H)=NOT I; /* TO IEEE 488 BUS */
   11 1 END;
   12 1 END IEEE#488#TEST;

   END IEEE#488#MODULE;

MODULE INFORMATION:

   CODE AREA SIZE = 0020H 40D
   VARIABLE AREA SIZE = 0001H 1D
   MAXIMUM STACK SIZE = 0002H 2D
   21 LINES READ
   0 PROGRAM ERROR(S)

END OF PL/M-80 COMPILATION
```
**Data Acquisition & Logging Module**

/ * Displays Bipolar Output in mVOLTS (+/- 5V Full Scale) / *

DECLARE DEC LITERALLY 'DECLARE';
DECLARE LITERAL 'LITERALLY';
DECLARE DISP LIT '01H';
DECLARE RETURN LIT '02H';
DECLARE CURSOR OFF LIT '0CH';
DECLARE HOME LIT '00H';
DECLARE 16 BIT BYTE;
DECLARE TEMP ADDRESS;
DECLARE TEMP ADDRESS;
DECLARE DATA(32) BYTE; /* Temp storage of logged data for one scan (Hi Byte) */
DECLARE DATA(32) BYTE; /* (Lo Byte) */
DECLARE ADDR ADDRESS;
DECLARE CHAR(16) BYTE EXTERNAL; /* Channel counter for mux */
DECLARE CHAN BYTE;
DECLARE NOF#SCANS ADDRESS EXTERNAL;
DECLARE NOF#CHARS BYTE EXTERNAL;
DECLARE SCAN RATE BYTE EXTERNAL;
DECLARE BLANK (*) BYTE DATA(' ');'
DECLARE MESS* (*) BYTE DATA('EXIT');
DECLARE MESS* (*) BYTE DATA('ENTRY');
DECLARE MESS* (*) BYTE DATA('7EH');
DECLARE MESS* (*) BYTE DATA('CHAN,H');
DECLARE MESS* (*) BYTE DATA(' ');'
DECLARE FLG# BYTE EXTERNAL;
DECLARE RST#5#FLG Byte EXTERNAL;
DECLARE DISP BYTE EXTERNAL;
DECLARE KEY BYTE EXTERNAL;
DECLARE POSITION BYTE EXTERNAL;
DECLARE RE# BYTE EXTERNAL;
DECLARE RE# BYTE EXTERNAL;
DECLARE BUFFER (16384) BYTE PUBLIC AT (8000H);
DECLARE BUFS#POINT ADDRESS;
DECLARE TAPE#MODE#FLG BYTE EXTERNAL;
DECLARE IEEE#MODE#FLG BYTE EXTERNAL;
DECLARE TRIG# TYPE BYTE EXTERNAL;

INST#OUT: PROCEDURE (INSTRUCTION) EXTERNAL;

DE# INSTRUCTION BYTE;

END INST#OUT;

C#OUT: PROCEDURE (CHAR) EXTERNAL;

DEC CHAR BYTE;

END C#OUT;
44 1  TEXT*OUT: PROCEDURE(P4,SIZE)EXTERNAL;
45 2  DEC P4 ADDRESS;
46 2  DEC SIZE BYTE;
47 2  END TEXT*OUT;
48 1  MENU: PROCEDURE(P1,P2,P3)EXTERNAL;
49 2  DEC (P1,P2,P3)ADDRESS;
50 2  END MENU;
51 1  BINTOASCII:PROCEDURE (READING,SIZE) EXTERNAL;
52 2  DEC SIZE BYTE;
53 2  DEC READING ADDRESS;
54 2  END BINTOASCII;
55 1  CHANN*DISP*SELECT:PROCEDURE EXTERNAL;
56 2  END CHANN*DISP*SELECT;
57 1  TAPE*BLOCK*DUMP:PROCEDURE EXTERNAL;
58 2  END TAPE*BLOCK*DUMP:
59 1  RTC*DATA:PROCEDURE (A) EXTERNAL;
60 2  DEC A BYTE;
61 2  END RTC*DATA;
62 1  DAC0*INIT: PROCEDURE PUBLIC;
63 2  OUTPUT(57H)=80H; /* INIT. DATA ACQUISITION BOARD 8255 */
64 2  END DAC0*INIT;
65 1  MUX*$35: PROCEDURE;
66 2  DEC A BYTE;
67 2  CHAN = CHAN+1; /* INC. CHANNEL PLUS 35 MICRO SEC DELAY*/
68 2  OUTPUT(55H)=CHAN;
69 2  A=A+1; /* DUMMY TASK */
70 2  A=A+1; /* */
71 2  END MUX*$35;
72 1  LOG*DISP:PROCEDURE (DISP,TEST);
73 2  DEC (DISP,TEST) BYTE;
74 2  CALL INST*OUT(CLEAR*DISP);
75 2  CALL INST*OUT(CURSOR*OFF);
76 2  CALL TEXT*OUT(MES$23,LENGTH(MES$23)); /* CHAN. */
77 2  CALL BINTOASCII(DISp,2);
78 2  CALL TEXT*OUT(CHAP,2); /* DO CASE TEST; */
79 2  CALL MENU(.BLANK,.BLANK,.MES$6); /* CALL MENU(.BLANK,.MES$16,.MES$6); /* EXIT */
80 3  CALL MENU(.BLANK,.MES$6); /* ENTRY EXIT */
81 3  END;
82 3  END LOG*DISP;
83 1  LOG*DATA#NORMALIZE:PROCEDURE;
84 2  CALL INST*OUT(089H);
85 2  TEMP$HID = (((DATA$HI(DISP-1) AND 0Fh)*1000)/8)*5;
86 2  TEMP$HI = (((SHR(DATA$HI(DISP-1),4))*1000)/8)*5;
87 2  TEMP$LO = (((SHR(DATA$LO(DISP-1),4))*1000)/8)*5;
88 2  ADD$DATA = TEMP$HI + SHR(TEMP$HI,4) + SHR(TEMP$LO,8); /* ADD$DATA RANGE 0-10000 mV */
89 2  END LOG*DISP;
DISABLE;
91 2 IF (SHL(LOW(TEMP#MID),4) + LOW(TEMP#LO)) > 80H THEN
92 2 ADDR#DATA = ADDR#DATA + 1; /* ROUND UP LSB */
93 2 IF IEEE#MODE#FLAG=1 THEN /* IF IEEE MODE SELECTED THEN O/P TO IEEE 488 BUS */
94 2 DO;
95 3 ENABLE;
96 3 B=0H;
97 3 DO WHILE B <> 20H; /* WAIT FOR NRFD=1 AND DRB=0 FROM IEEE BUS */
98 4 B=(INPUT(06H) AND 028H); /* INDICATING IEEE LISTENER READY FOR DATA */
99 4 IF KEY=2 THEN B=20H;
100 4 END;
101 3 OUTPUT(60H)=NOT(HIGH(ADDR#DATA)); /* OUTPUT HI-BYTE TO IEEE */
102 3 B=0H;
103 3 DO WHILE B <> 20H; /* WAIT FOR NRFD=1 AND DRB=0 FROM IEEE BUS */
104 4 B=(INPUT(06H) AND 028H);
105 4 IF KEY=2 THEN B=20H;
106 4 END;
107 3 OUTPUT(60H)=NOT(LOW(ADDR#DATA)); /* OUTPUT LO-BYTE TO IEEE */
108 3 DISABLE;
109 3 END;
110 2 IF ADDR#DATA >= 5000 THEN /* 0/P LOGGED VALUE (BI-POLAR) OF SELECTED */
111 2 DO;
112 3 CALL C#OUT('!'); /* CHANNEL (IN mV, PLUS SIGN) TO DISPLAY */
113 3 ADDR#DATA = ADDR#DATA - 5000;
114 3 END;
115 2 ELSE
116 2 DO;
117 3 CALL C#OUT('-');
118 3 ADDR#DATA = 5000 - ADDR#DATA;
119 3 END;
120 2 CALL BIN#TO#ASCII(ADDR#DATA,4);
121 2 ENABLE;
122 2 CALL TEXT#OUT(.CHAR,4);
123 2 CALL C#OUT('m');
124 2 CALL C#OUT('v');
125 2 END C#OUT('u');
126 2 END LOG#SCAN#NORMALIZE;
127 1 LOG#SCAN:PROCEDURE PUBLIC:
128 2 KEY=0FH;
129 2 BUFF#POINT=0;
130 2 OUTPUT(55H)=01;
131 2 CHW#=0;
132 2 CALL LOG#DISP(DISP,0);
133 2 REGA# = SCAN#RATE AND 0FH;
134 2 DISABLE;
135 2 REGB = 42H;
136 2 CALL RTC#DATA(0);
137 2 DO N=0 TO (HOR#OF#CHWS - 1);
138 3 IF KEY<>2 THEN
139 3 DO;
140 4 RST#MS#FLAG=0;
141 4 DO WHILE RST#MS#FLAG=0;
142 5 ENABLE;
143 5 END;
144 4 DISABLE;
145 4 DO J=0 TO (HOR#OF#CHWS - 1):
146 5  OUTPUT (50H)=0;
147 5  CALL MUX(35);
148 5  DATA#HI(1) = INPUT(50H);
149 5  DATA#LO(1) = INPUT(51H);
150 5  BUFFER(BUFF#POINT)=DATA#HI(1);
151 5  BUFFER(BUFF#POINT+1)=DATA#LO(1);
152 5  BUFF#POINT=BUFF#POINT+2;
153 5  END;
154 4  CHAN = 0;
155 4  OUTPUT(55H)=0;
156 4  ENABLE;
157 4  CALL LOG#DATA#NORMALIZE;
158 4  END;
159 3  END;
160 2  REG#B = 02H;
161 2  DISABLE;
162 2  IF (TAPE#MODE#FLAG=0) AND (KEY=2) THEN
163 2  CALL TAPE#BLOCK#DUMP;
164 2  RST#55#FLAG=0;
165 2  IF TRIG#TYPE='A' THEN
166 2  REG#B=22H;
167 2  END LOG#SCAN;
168 1  D#AC#TEST:PROCEDURE PUBLIC:
169 2  KEY,FLAG#0=
170 2  OUTPUT(55H)=DISP-1;
171 2  CALL LOG#DISP(DIS-1);
172 2  DO WHILE KEY=2;
173 3  DISABLE;
174 3  OUTPUT(50H)=0;
175 3  CALL MUX(35);
176 3  ENABLE;
177 2  OUTPUT(55H)=DISP-1;
178 3  DATA#HI(DIS-1)=INPUT(50H);
179 3  DATA#LO(DIS-1)=INPUT(51H);
180 3  CALL LOG#DATA#NORMALIZE;
181 3  DISABLE;
182 3  IF FLAG#0=0FEH THEN
183 3  DO:
184 4  IF KEY=1 THEN CALL CHAN#DISP#SELECT; /* ENTER NEW CHANNEL TO BE DISPLAYED (DISP) */
185 4  IF KEY=#12 THEN DISP=DISP-1; /* DECREMENT DISP */
186 4  IF KEY=#15 THEN DISP=DISP+1; /* INCREMENT DISP */
187 4  OUTPUT(55H)=DISP-1;
188 4  CALL LOG#DISP(DIS-1);
189 4  FLAG#0=0;
190 3  END;
191 2  END D#AC#TEST;
192 1  END D#AC#MODULE;

MODULE INFORMATION:
CODE AREA SIZE = 0380H 909D
TAPENMODUKE: 00:
  01 DECLARE DEC LITERALLY 'DECLARE';
  02 DEC LIT LITERALLY 'LITERALLY';
  03 DEC PORTA LIT '044H';
  04 DEC PORTB LIT '045H';
  05 DEC PORTC LIT '046H';
  06 DEC PORTCON REG LIT '047H';
  07 DEC IDLE LIT '0FFH';
  08 DEC CE1 LIT '0FDH';
  09 DEC CE2 LIT '0FBH';
  10 DEC CE3 LIT '0FH';
  11 DEC CE4 LIT '0EFH';
  12 DEC CSE LIT '0DFH';
  13 DEC CSE LIT '0BFH';
  14 DEC CSE LIT '0FH';
  15 DEC CSE LIT '07FH';
  16 DEC WRITEMODE LIT '08H';
  17 DEC FUD LIT '001H';
  18 DEC REV LIT '002H';
  19 DEC SEARCHSPEED LIT '010H';
  20 DEC IPS#00 LIT '030H';
  21 DEC IPS#120 LIT '004H';
  22 DEC I ADDRESS;
  23 DEC (N,K,T*,DATA) BYTE;
  24 DEC BUFFER (16384)BYTE EXTERNAL;
  25 DEC BUFF#SIZE ADDRESS;
  26 DEC BUFF#POINT ADDRESS;
  27 DEC SYNC LIT '0AAH';
  28 DEC (KEY,FLAG1) BYTE EXTERNAL;
  29 DEC CLEAR#DISP LIT '01H';
  30 DEC CURSOR#OFF LIT '0CH';
  31 DEC CURSOR#ON LIT '0EH';
  32 DEC CHAR(16) BYTE EXTERNAL;
  33 DEC HOME#2 LIT '0CH';
  34 DEC POSITION BYTE EXTERNAL;
  35 DEC LAST#RECORD BYTE;
  36 DEC RECORD BYTE AT (2016H);
  37 DEC B#DATA ADDRESS EXTERNAL;
  38 DEC BLANK (*) BYTE DATA (' ');
  39 DEC MES#1 (*) BYTE DATA ('0F3H', 'SET', '0F3H');
  40 DEC MES#2 (*) BYTE DATA ('EXIT');
  41 DEC MES#16 (*) BYTE DATA ('EXIT');
  42 DEC MES#35 (*) BYTE DATA ('NEXT RECORD = ');
  43 DEC MES#36 (*) BYTE DATA ('READ ');
  44 DEC MES#37 (*) BYTE DATA ('SET RECORD TO BE');
91 2 \textbf{OUTPUT(FRINTC} = INST; /\* CONTROL LINE PASSED TO "INST" */
92 2 \textbf{TIDATA} = INPUT(PORTB); 93 2 \textbf{OUTPUT(FRINTC} = IDLE;
94 2 \textbf{END TAPESTROBEINH};
95 1 \textbf{DELAY:PROCEDURE(HIL_SEC);} /\* DELAYS (1ms \* VALUE OF HIL_SEC) */
96 2 DEC HIL_SEC ADDRESS;
97 2 \textbf{DO I = 0 TO HIL_SEC;}
98 2 CALL TIME(?); /\* 1 mSEC */
99 3 \textbf{END;}
100 2 \textbf{END DELAY;}
101 1 \textbf{PORTAOUT:PROCEDURE(CONTROLL,OUTDATA);} /\* O/P DATA TO USART */
102 2 DEC (CONTROLL,OUTDATA) BYTE;
103 2 \textbf{OUTPUT(PORTA} = OUTDATA;
104 2 \textbf{CALL TAPESTROBEOUT(CONTROLL);}
105 2 \textbf{END PORTAOUT;}
106 1 \textbf{TEST:PROCEDURE;}
107 2 \textbf{CALL TAPESTROBEIN(CE6);} /\* READ USRT STATUS */
108 2 \textbf{DO WHILE (TIDATA AND 00H)=00H;}
109 3 \textbf{CALL TAPESTROBEIN(CE6);}
110 3 \textbf{END;}
111 2 \textbf{END TEST;}
112 1 \textbf{READ:PROCEDURE;}
113 2 \textbf{CALL TAPESTROBEIN(CE6);} /\* READ USRT STATUS */
114 2 \textbf{DO WHILE (TIDATA AND 00H)=00H;}
115 2 \textbf{CALL TAPESTROBEIN(CE6);}
116 3 \textbf{END;}
117 2 \textbf{END READ;}
118 1 \textbf{TAPERRORMES:PROCEDURE;} /\* O/P MESSAGE DEPENDING ON ERROR TYPE */
119 2 \textbf{CALL INOUT(CLEARDISP);}
120 2 IF (TIDATA AND 00H)=00H THEN
121 2 DO:
122 3 \textbf{CALL TEXTOUT(MES\#39,LENGTH(MES\#39));} /\* LOAD CASSETTE */
123 3 \textbf{CALL MENU(MES\#1,BLANK,BLANK);} /\* SET */
124 3 \textbf{RETURN;}
125 3 \textbf{END;}
126 2 IF (TIDATA AND 00H)=20H THEN
127 2 DO:
128 3 \textbf{CALL TEXTOUT(MES\#40,LENGTH(MES\#40));} /\* END OF TAPE */
129 3 \textbf{RETURN;}
130 3 \textbf{END;}
131 2 IF (TIDATA AND 01H)=01H THEN
132 2 \textbf{CALL TEXTOUT(MES\#41,LENGTH(MES\#41));} /\* WRITE PROTECTED */
133 2 \textbf{END TAPERRORMES;}
134 1 \textbf{RENUM:PROCEDURE;}
135 2 \textbf{CALL TAPESTROBEOUT(CE4);} /\* RESET EOT */
136 2 \textbf{CALL TAPESTROBEIN(CE5);} /\* RESET OPTICAL */
137 2 \textbf{CALL PORTAOUT(CE1,SEARCHSPEED OR REV);}
138 2 \textbf{DO WHILE (TIDATA AND 00H)=00H;}
139 3 \textbf{TAPE LOADED, NO EOT? */
140 3 \textbf{CALL TAPESTROBEIN(CE5);} /\* TRANSPOT STATUS */
141 3 \textbf{END;}
CALL PORT#4&OUT(CE1,00H); /* STOP */
CALL DELAY(360); /* UNIT 360mSEC */
END REWIND;

POSITION=TAPE: PROCEDURE;
* POSITION TAPE AT BEGINNING OF GAP */
DO I = 1 TO K;
DO WHILE (T#DATA AND 0A2H) = 02H;
CALL TPE#STROBE=IN(CE5); /* DO WHILE GAP PRESENT */
END;
DO WHILE (T#DATA AND 0A2H) = 00H;
CALL TPE#STROBE=IN(CE5); /* DO WHILE DATA PRESENT */
END;
END POSITION=TAPE;

GET#OFF#LEADER: PROCEDURE(READ#FLAG);
DEC READ#FLAG BYTE;
CALL TPE#STROBE#OUT(CE4); /* RESET EOT */
CALL TPE#STROBE#IN(CE5); /* RESET OPTICAL */
DO CASE READ#FLAG:
CALL PORT#4&OUT(CE1,(SEARCH#SPEED OR FWD OR WRITE#MODE))/* START TAPE TRANSPORT FORWARD */
CALL PORT#4&OUT(CE1,FUD); /* CLEAR LEADER PRESENT? */
CALL TPE#STROBE#IN(CE5); /* TRANSPORT STATUS */
END;
DO CASE READ#FLAG;
CALL PORT#4&OUT(CE1,00H); /* STOP */
DO;
CALL POSITION=TAPE;
CALL PORT#4&OUT(CE1,00H); /* STOP */
END;
END GET#OFF#LEADER;

RECORD#DISP: PROCEDURE;
CALL INST#OUT(CLEAR#DISP);
CALL TEXT#OUT(.MES$35,LENGTH(MES$35)); /* NEXT RECORD= */
DISABLE;
CALL BIN#TO#ASCII(RECORD,2);
ENABLE;
CALL TEXT#OUT(.CHAR,2);
CALL MENU(.MES$35,.MES$16,.MES$6); /* READ ENTRY EXIT */
END RECORD#DISP;

TAPE#BLOCK#DUMP: PROCEDURE PUBLIC;
IF RECORD=0 THEN
BUFF#SIZE=0FFH;
ELSE
BUFF#SIZE=NO#OF#SCANS * NO#OF#CHANS *2;
CALL TPE#STROBE#OUT(CE4); /* RESET EOT & USRT */
CALL TPE#STROBE#IN(CE5); /* RESET OPTICAL */
CALL TPE#STROBE#IN(CE5); /* TRANSPORT STATUS */
BUFF#POINT=0;
152 2 IF (T#DATA AND 04H)=00H THEN /* TAPE ERROR? */
153 2 CALL PORT#OUT(CE1,(FWD OR WRITE#MODE));
155 3 ELSE
156 3 DO:
157 3 CALL PORT#OUT(CE1,0); /* STOP */
158 3 TAPEERROR$FLAG=OFFH;
159 3 RETURN;
160 3 END:
161 2 CALL DELAY(50);
162 2 CALL TMT;
163 2 CALL PORT#OUT(CE3,00H);
167 2 CALL TMT;
168 2 CALL PORT#OUT(CE3,SYNC);
169 2 CALL TMT;
170 2 DO I=0 TO 5:
171 3 CALL PORT#OUT(CE3,RTC$DATE(I)); /* O/P DATE */
172 3 CALL TMT;
173 2 END:
174 2 DO I=0 TO 5:
175 3 CALL PORT#OUT(CE3,RTC$TIME(I)); /* O/P TIME */
176 3 CALL TMT;
177 2 END:
178 2 CALL PORT#OUT(CE3,SCAN$DATE);
179 2 CALL TMT;
180 2 CALL PORT#OUT(CE3,NO$OF$CHANNELS);
181 2 CALL TMT;
182 2 CALL PORT#OUT(CE3,HIGH$NO$OF$SCANS); /* O/P NUMBER OF SCANS (HI BYTE) */
183 2 CALL TMT;
184 2 CALL PORT#OUT(CE3,LOW$NO$OF$SCANS); /* O/P NUMBER OF SCANS (LO BYTE) */
221 2 CALL TMT;
222 2 DO I=0 TO BUFSIZE;
223 3 CALL PORT#OUT(CE3,BUFFER(I)); /* DUMP LOGGED DATA TO TAPE FROM BUFFER */
224 3 CALL TAPE$STROBE$IN(CE5);
225 3 IF (T#DATA AND 04H)=00H THEN /* TAPE ERROR? */
226 3 CALL TMT;
229 3 ELSE
230 3 DO:
231 4 CALL PORT#OUT(CE1,0); /* STOP */
232 4 TAPEERROR$FLAG=OFFH;
233 4 RETURN;
234 4 END;
235 2 CALL PORT#OUT(CE3,SYNC);
236 2 CALL TMT;
237 2 CALL PORT#OUT(CE3,00H);
238 2 CALL TAPE$STROBE$IN(CE5); /* TRANSPORT STATUS */
239 2 DO WHILE (T#DATA AND 04H)=00H;
240 2 CALL TAPE$STROBE$IN(CE5); /* TRANSPORT STATUS */
241 2 CALL DELAY(1000); /* WAIT FOR GAP */
242 2 CALL DELAY(500); /* TRANSPORT STATUS */
243 2 CALL DELAY(50); /* WAIT 50mS */
244 2 CALL PORT#OUT(CE1,00H); /* STOP */
245 2 CALL DELAY(360); /* WAIT 360mSEC */
246 2 CALL PORT#OUT(CE1,00H); /* WRITE MODE OFF */
247 2 RECORD$=RECORD$+1;
249 2 END TAPE$BLOCK$DUMP;
TAPEFILEREAD: PROCEDURE;

DISABLE;

CALL TAPE$STROBE$OUT(C4);  /* RESET EOT & USRT */
CALL TAPE$STROBE$INC(E5);  /* RESET OPTICAL */
CALL TAPE$STROBE$INC(E5);

IF (T$DATA AND 0A0H)=00H THEN  /* TAPE ERROR? */
    DO:
    IF RECORD <= LAST$RECORD THEN  /* POSITION TAPE AT GAP IMMEDIATELY */
        DO:
            K=(LAST$RECORD - RECORD)+1;
            CALL PORT$OUT(CE1,REV);
            CALL POSITION$TAPE;
            DO WHILE (T$DATA AND 0A2H)=02H;
                CALL TAPE$STROBE$INC(E5);  /* DO WHILE GAP PRESENT */
            END;
            CALL DELAY(50);
            CALL PORT$OUT(CE1,OH);
            CALL DELAY(360);
            CALL PORT$OUT(CE1,FWD);
            DO WHILE (T$DATA AND 0A2H) = 00H;
            CALL TAPE$STROBE$INC(E5);  /* DO WHILE DATA PRESENT */
            END;
        END;
    ELSE
        DO:
            K=RECORD -(LAST$RECORD+1);
            CALL PORT$OUT(CE1,OH);
            IF K > 0 THEN
                CALL PORT$OUT(CE1,FWD);
            CALL POSITION$TAPE;
            END;
        END;
        ELSE
            DO:
                CALL PORT$OUT(CE1,OH);
                TAPE$ERROR$FLAG=0FFH;
                RETURN;
            END;
            CALL TAPE$STROBE$OUT(C4);  /* STOP */
            CALL PORT$OUT(CE2,SYNC);
            CALL RDA;
            CALL TAPE$STROBE$INC(E7);
            DO I = 0 TO 15;
            CALL RDA;
            CALL TAPE$STROBE$INC(E7);
            BUFFER(I) = T$DATA;
            END;
            BUFF$POINT=16;
            NO$OF$CHANS=BUFFER(13);
            NO$OF$SCANS=BUFFER(14);
            NO$OF$SCANS=SHL(NO$OF$SCANS,8) + BUFFER(15);  /* FORM 16 BIT WORD */
            DO I = 0 TO NO$OF$SCANS-1;
                DO I = 0 TO SHL(NO$OF$CHANS,1)-1;
                    CALL RDA;
                    CALL TAPE$STROBE$INC(E7);
                    BUFFER(BUFF$POINT) = T$DATA;
            END;
            CALL TAPE$STROBE$INC(E5);  /* TRANSPORT STATUS */
299 4 IF (TIA*TAPE AND 0A0H)=00H THEN /* TAPE ERROR */
300 4 BUFF#POINT=BUF#POINT+1;
301 4 ELSE
302 5 DO;
303 5 CALL PORT#OUT(CE1,0);
304 5 TAPEERROR#FLAG=0FFH;
305 5 RETURN;
306 5 END;
307 5 END;
308 2 CALL TAPE#STROBE#IN(C5);
309 2 DO WHILE (TIA*TAPE AND 0A2H)=00H; /* WAIT FOR GAP */
310 2 CALL TAPE#STROBE#IN(C5); /* TRANSPORT STATUS */
311 2 END;
312 2 CALL DELAY(50);
313 2 CALL PORT#OUT(CE1,00H); /* STOP */
314 2 CALL DELAY(360);
315 2 LAST#RECORD=RECORD;
316 2 RECORD=RECORD+1;
317 2 CALL RECORD#DISP;
318 2 ENABLE;
319 2 END TAPE#BLOCK#READ;

320 1 RECORD#SELECT: PROCEDURE;
321 2 CALL INST#OUT(CLEAR#DISP);
322 2 CALL INST#OUT(CURSOR#ON);
323 2 CALL TEXT#OUT(MES#37,LENGTH(MES#37)); /* SET RECORD TO BE */
324 2 CALL INST#OUT(HOME#2);
325 2 CALL TEXT#OUT(MES#38,LENGTH(MES#38)); /* READ # */
326 2 CALL BIN#TO#ASCII(RECORD,2);
327 2 CALL TEXT#OUT(CAR,2);
328 2 CALL INST#OUT(POSITION-2) OR 00H);
329 2 CALL #IN;
330 2 CALL ASCII#TO#BIN(CAR,2);
331 2 RECORD=BIN#DATA;
332 2 CALL INST#OUT(CURSOR#OFF);
333 2 END RECORD#SELECT;

334 1 TAPE#READ#INIT: PROCEDURE;
335 2 CALL TAPE#STROBE#IN(C5);
336 2 IF (TIA*TAPE AND 080H)=00H THEN /* TAPE LOADED */
337 2 DO;
338 3 TAPEERROR#FLAG=0FFH;
339 3 RETURN;
340 3 END;
341 2 ELSE;
342 2 CALL INST#OUT(CLEAR#DISP);
343 2 CALL TEXT#OUT(MES#42,LENGTH(MES#42)); /* TAPE INITIALIZE */
344 2 CALL INST#OUT(HOME#2);
345 2 CALL TEXT#OUT(MES#43,LENGTH(MES#43)); /* IN PROGRESS */
346 2 CALL REWIND;
347 2 =1;
348 2 CALL GET#OFF#LEADER(1);
349 2 LAST#RECORD=01H;
350 2 RECORD=01H;
351 2 END TAPE#READ#INIT;
BUFF$TO$IEEE$DUMP: PROCEDURE;
353 2  DEC TEMP$BYTE;
354 3  DO I = 0 TO 15;
355 4  OUTPUT$(40H)=NOT$(BUFFER(I));  /* DUMP PREAMBLE TO IEEE 488 BUS */
356 5  END;
357 6  TEMP=0;
358 7  DO WHILE TEMP$(20H);
359 8  TEMP=$(INPUT$(61H) AND 28H);  /* WAIT FOR IEEE 488 BUS = READY FOR DATA */
360 9  END;
361 10  BUFF$POINT=16;
362 11  DO I=0 TO NO$OF$SCANS-1;
363 12  DO I = 0 TO HIM$(NO$OF$SCANS,1)-1;
364 13  OUTPUT$(40H)=NOT$(BUFFER(BUFF$POINT));  /* DUMP DATA FROM BUFFER TO IEEE 488 BUS */
365 14  BUFF$POINT=BUFF$POINT+1;  /* INVERT DUE TO IEEE 488 NEGATIVE LOGIC */
366 15  END;
367 16  END;
368 17  END BUFF$TO$IEEE$DUMP;
369 1  TAPE$PLAY: PROCEDURE PUBLIC;  /* TAPE REPLAY MODE */
370 2  CALL TAPE$READ$INIT;
371 3  KEY=0;
372 4  FLAG=0;
373 5  CALL RECORD$DISP;
374 6  DO WHILE (KEY$(2) AND (TAPE$ERROR$FLAG=0));  /* DO WHILE NOT EXIT OR TAPE ERROR */
375 7  CALL KEY$ENTRY;
376 8  IF KEY=0 THEN
377 9  DO:
378 10  CALL TAPE$BLOCK$READ;  /* READ BLOCK OF DATA */
379 11  IF IEEE$MODE$FLAG=1 THEN
380 12  CALL BUFF$TO$IEEE$DUMP;
381 13  END;
382 14  ELSE
383 15  DO:
384 16  IF KEY=1 THEN CALL RECORD$SELECT;  /* SELECT RECORD */
385 17  IF KEY=12 THEN RECORD=RECORD-1;  /* DECREMENT RECORD */
386 18  IF KEY=15 THEN RECORD=RECORD+1;  /* INCREMENT RECORD */
387 19  CALL RECORD$DISP;
388 20  END;
389 21  END;
390 22  TAPE$MODE$FLAG=0;  /*RETURN TO RECORD MODE */
391 23  END TAPE$PLAY;
392 1  TAPE$WRITE$INIT: PROCEDURE;
393 2  CALL INST$OUT(CLEAR$DISP);
394 3  CALL TEXT$OUT(HES$42,LENGTH(HES$42));  /* TAPE INITIALLY */
395 4  CALL INST$OUT(HOME$2);
396 5  CALL TEXT$OUT(HES$43,LENGTH(HES$43));  /* IN PROGRESS */
397 6  CALL P.Handle;
398 7  CALL GET$OFF$LEADER(0);
399 8  RECORD=0;
400 9  DO CALL TAPE$BLOCK$DUMP;  /* INITIAL DUMMY DUMP */
401 10  END TAPE$WRITE$INIT;
402 1  TAPE$ERROR: PROCEDURE PUBLIC;
403 2  CALL TAPE$ERROR$HES;
<table>
<thead>
<tr>
<th>Line Number</th>
<th>Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>END PROGRAM</td>
</tr>
<tr>
<td>2</td>
<td>END</td>
</tr>
<tr>
<td>3</td>
<td>CALL DELAY (1)</td>
</tr>
<tr>
<td>4</td>
<td>BEGIN</td>
</tr>
<tr>
<td>5</td>
<td>CALL TPEM1LTEL1</td>
</tr>
<tr>
<td>6</td>
<td>TPEM1=0</td>
</tr>
<tr>
<td>7</td>
<td>IF (TPEM1 AND 80H) THEN</td>
</tr>
<tr>
<td>8</td>
<td>DISP1</td>
</tr>
<tr>
<td>9</td>
<td>CALL TPEM1PDES (C5)</td>
</tr>
<tr>
<td>10</td>
<td>DO WHILE (KEY=0) AND (TPEM1 ERROR=OFF)</td>
</tr>
<tr>
<td>11</td>
<td>KEYP1</td>
</tr>
<tr>
<td>12</td>
<td>CALL TPEM1PDES</td>
</tr>
<tr>
<td>13</td>
<td>END</td>
</tr>
<tr>
<td>14</td>
<td>END TPEM1PDES</td>
</tr>
</tbody>
</table>

END OF PL/M-80 COMPILER

PL/M-80 COMPILER
APPENDIX B

Software Version V1.3
Program listings

Note: Only those modules which differ from Version V1.2 are included.
45 1 DEC MES#38 (*) BYTE DATA ('READ: # ');  
46 1 DEC MES#39 (*) BYTE DATA ('LOAD CASSETTE!');  
47 1 DEC MES#40 (*) BYTE DATA ('END OF TAPE!');  
48 1 DEC MES#41 (*) BYTE DATA ('WRITE PROTECTED!');  
49 1 DEC MES#42 (*) BYTE DATA ('READ INITIALIZE');  
50 1 DEC MES#43 (*) BYTE DATA ('IN PROGRESS');  
51 1 DEC MES#44 (*) BYTE DATA ('TAPE BLOCK READ');  
52 1 DEC MES#45 (*) BYTE DATA ('TAPE BLOCK WRITE');  
53 1 DEC MES#46 (*) BYTE DATA ('IEEE TRANSFER');  
54 1 DEC MES#47 (*) BYTE DATA ('WRITE INITIALIZE');  
55 1 DEC RTC#DATE(6) BYTE EXTERNAL;  
56 1 DEC RTC#TIME(6) BYTE EXTERNAL;  
57 1 DEC NO#OF#SCANS ADDRESS EXTERNAL;  
58 1 DEC NO#OF#SCANS BYTE EXTERNAL;  
59 1 DEC SCAN#BYTE BYTE EXTERNAL;  
60 1 DEC TAPE#ERROR#FLAG BYTE PUBLIC;  
61 1 DEC TAPE#MODE#FLAG BYTE EXTERNAL;  
62 1 DEC IEEE#MODE#FLAG BYTE EXTERNAL;  
63 1 INST#OUT; PROCEDURE(INSTRUCTION) EXTERNAL;  
64 2 DEC INSTRUCTION BYTE;  
65 2 END INST#OUT;  
66 1 TEXT#OUT; PROCEDURE(P4,SIZE) EXTERNAL;  
67 2 DEC SIZE BYTE;  
68 2 DEC P4 ADDRESS;  
69 2 END TEXT#OUT;  
70 1 BIN#TO#ASCII; PROCEDURE(READING,SIZE) EXTERNAL;  
71 2 DEC SIZE BYTE;  
72 2 DEC READING ADDRESS;  
73 2 END BIN#TO#ASCII;  
74 1 ASCII#TO#BIN; PROCEDURE(ASCII#ADR,SIZE) EXTERNAL;  
75 2 DEC SIZE BYTE;  
76 2 DEC ASCII#ADR ADDRESS;  
77 2 END ASCII#TO#BIN;  
78 1 MENU; PROCEDURE(P1,P2,P3) EXTERNAL;  
79 2 DEC (P1,P2,P3) ADDRESS;  
80 2 END MENU;  
81 1 KEY#IN; PROCEDURE EXTERNAL;  
82 2 END KEY#IN;  
83 1 KEY#ENTRY; PROCEDURE EXTERNAL;  
84 2 END KEY#ENTRY;  
85 1 TAPE#INIT; PROCEDURE PUBLIC;  
86 2 OUTPUT (47H)=092H;  
87 2 END TAPE#INIT;  
88 1 TAPE#STROBE#OUT; PROCEDURE(INST);  
89 2 DEC INST BYTE;  
90 2 OUTPUT(PORT#A) = INST;  
91 2 OUTPUT(PORT#C) = IDLE;  
/* INIT INE TAPE BOARD 8255 */  
/* STROBES CONTROL LINE PASSED TO "INST" */  
/* FROM HI TO LO TO HI */
92 2 END TAPE$STROBE$OUT;

93 1 TAPE$STROBE$IN; PROCEDURE(INST); /* STROBE$ DATA INTO T$DATA */
94 2 DEC INST BYTE;
95 2 OUTPUT(PORT$C) = INST;
96 2 T$DATA = INPUT(PORT$B);
97 2 OUTPUT(PORT$C) = IDLE;
98 2 END TAPE$STROBE$IN;

99 1 DELAY;PROCEDURE(MIL$SEC);
100 2 DEC MIL$SEC ADDRESS;
101 2 DO I = 0 TO MIL$SEC;
102 3 CALL TIME(7);/* ~1 mSEC*/
103 3 END;
104 2 END DELAY;

105 1 PORT$#0$OUT;PROCEDURE(CONTROLL,OUT$DATA); /* 0/$ DATA TO USART */
106 2 DEC (CONTROLL,OUT$DATA) BYTE;
107 2 OUTPUT (PORT$A) = OUT$DATA;
108 2 CALL TAPE$STROBE$OUT(CONTROLL);
109 2 END PORT$#0$OUT;

110 1 TBMT;PROCEDURE;
111 2 CALL TAPE$STROBE$IN(CE6);
112 2 DO WHILE (T$DATA AND 01H)=00H;
113 3 CALL TAPE$STROBE$IN(CE6);
114 3 END;
115 2 END TBMT;

116 1 RD$4;PROCEDURE;
117 2 CALL TAPE$STROBE$IN(CE6);
118 2 DO WHILE (T$DATA AND 080H)=00H;
119 3 CALL TAPE$STROBE$IN(CE6);
120 3 END;
121 2 END RD$4;

122 1 TAPE$ERROR$HES; PROCEDURE; /* 0/$ MESSAGE DEPENDING ON ERROR TYPE */
123 2 CALL INST$OUT(CLEAR$DISP);
124 2 IF (T$DATA AND 80H)=80H THEN DO;
125 3 CALL TEXT$OUT(.HES$39,LENGTH(HES$39)); /* LOAD CASSETTE! */
126 3 CALL MENU(.HES$1,.BLANK,.BLANK);
127 3 RETURN;
128 3 END;
129 2 IF (T$DATA AND 20H)=20H THEN DO;
130 3 CALL TEXT$OUT(.HES$40,LENGTH(HES$40)); /* END OF TAPE! */
131 3 RETURN;
132 3 END;
133 2 IF (T$DATA AND 01H)=01H THEN DO;
134 3 CALL TEXT$OUT(.HES$41,LENGTH(HES$41)); /* WRITE PROTECTED! */
135 3 RETURN;
136 3 END;
137 2 END TAPE$ERROR$HES;

138 1 REWIND; PROCEDURE;
139 2 CALL TAPE$STROBE$OUT(CF4);
140 2 CALL TAPE$STROBE$IN(CE5); /* RESET EOT */
141 2 CALL TAPE$STROBE$IN(CE5); /* RESET OPTICAL */
CALL PORT#1#OUT(CE1,(SEARCH#SPEED OR REV));
DO WHILE (T#DATA AND 0A0H) = 00H; /* TAPE LOADED, NO EOT? */
CALL TAPE#STROBE#IN(CE5); /* TRANSPORT STATUS */
END;
CALL PORT#1#OUT(CE1,00H); /* STOP */
CALL DELAY(360); /* WAIT 360mSEC */
END REMIND:

POSITION#TAPE: PROCEDURE;
DO I = 1 TO K;
DO WHILE (T#DATA AND 0A2H) = 02H;
CALL TAPE#STROBE#IN(CE5); /* DO WHILE GAP PRESENT */
END;
DO WHILE (T#DATA AND 0A2H) = 00H;
CALL TAPE#STROBE#IN(CE5); /* DO WHILE DATA PRESENT */
END;
END REMIND:
END POSITION#TAPE;

GET#OFF#LEADER: PROCEDURE(READ#FLAG);
DEC READ#FLAG BYTE;
CALL TAPE#STROBE#OUT(CE4); /* RESET EOT */
CALL TAPE#STROBE#IN(CE5); /* RESET OPTICAL */
DO CASE READ#FLAG;
CALL PORT#1#OUT(CE1, (SEARCH#SPEED OR FWD OR WRITE#MODE)); /* START TAPE TRANSPORT FORWARD*/
CALL PORT#1#OUT(CE1,FWD); /* START TAPE TRANSPORT FORWARD*/
END;
DO WHILE (T#DATA AND 0E0H) = 00H; /* CLEAR LEADER PRESENT? */
CALL TAPE#STROBE#IN(CE5); /* TRANSPORT STATUS */
END;
DO CASE READ#FLAG;
CALL POR#1#OUT(CE1,08H); /* STOP */
DO;
CALL POSITION#TAPE;
CALL PORT#1#OUT(CE1,00H); /* STOP */
CALL DELAY(365); /* WAIT 360mSEC */
CALL PORT#1#OUT(CE1,00H); /* WRITE MODE OFF */
END GET#OFF#LEADER:

RECORD#DISP: PROCEDURE;
CALL INST#OUT(CLEAR#DISP); CALL TEXT#OUT(H#35,LENGTH(H#35)); /* NEXT RECORD= */
DISABLE;
CALL BIN#TO#ASCII(RECORD,2); ENABLE;
CALL TEXT#OUT(CH#35,CH#35); /* READ ENTRY EXIT */
END RECORD#DISP;

TAPE#BLOCK#DISP: PROCEDURE PUBLIC;
IF RECORD=0 THEN
BUFF#SIZE=OFFH;
ELSE
BUFF#SIZE=NO#OF#SCANS * NO#OF#CHANS *2;
CALL INST#OUT(CLEAR*DISP);
CALL TEXT#OUT(HES#45,LENGTH(HES#45)); /* TAPE BLOCK WRITE */
CALL INST#OUT(HOME#2);
CALL TEXT#OUT(HES#43,LENGTH(HES#43)); /* IN PROGRESS */
CALL TAPE#STROBE#OUT(CE4); /* RESET EOT & UST */
CALL TAPE#STROBE#IN(CE5); /* RESET OPTICAL */
CALL TAPE#STROBE#IN(CE5); /* TRANSPORT STATUS */
BUFF#POINT=0;
IF (TDATA AND 0AH)=00H THEN /* TAPE ERROR? */
CALL PORT#A#OUT(CE1,(FWD OR WRITE#MODE));
ELSE
DO;
CALL PORT#A#OUT(CE1,0); /* STOP */
TPE#ERROR#FLAG=OFFH;
RETURN;
END;
CALL DELAY(50);
CALL TB#T;
/* WAIT TILL TRANS BUFF = EMTY */
CALL PORT#A#OUT(CE3,0OH);
CALL TB#T;
CALL PORT#A#OUT(CE3,SYNC);
CALL TB#T;
DO I=0 TO 5;
CALL PORT#A#OUT(CE3,RTC#DATE(I)); /* O/P DATE */
CALL TB#T;
END;
DO I=0 TO 5;
CALL PORT#A#OUT(CE3,RTC#TIME(I)); /* O/P TIME */
CALL TB#T;
END;
CALL PORT#A#OUT(CE3,SCAN#RATE); /* O/P SCAN RATE */
CALL TB#T;
CALL PORT#A#OUT(CE3,H#0#OF#CHANS); /* O/P NUMBER OF CHANNELS */
CALL TB#T;
CALL PORT#A#OUT(CE3,H#0#OF#SCANS); /* O/P NUMBER OF SCANS (HI BYTE) */
CALL TB#T;
CALL PORT#A#OUT(CE3,L#0#OF#SCANS); /* O/P NUMBER OF SCANS (LO BYTE) */
CALL TB#T;
DO I=0 TO BUFF#SIZE;
CALL PORT#A#OUT(CE3,BUFFER(I));
CALL TAPE#STROBE#IN(CE5); /* TRANSPORT STATUS */
IF (TDATA AND 0AH)=00H THEN /* TAPE ERROR? */
CALL TB#T;
ELSE
DO;
CALL PORT#A#OUT(CE1,0); /* STOP */
TPE#ERROR#FLAG=OFFH;
RETURN;
END;
END;
CALL PORT#A#OUT(CE3,SYNC); /* SYNC WORD */
CALL TB#T;
CALL PORT#A#OUT(CE3,0OH);
CALL TAPE#STROBE#IN(CE5); /* TRANSPORT STATUS */
DO WHILE (TDATA AND 0AH)=00H;
CALL TAPE#STROBE#IN(CE5); /* TRANSPORT STATUS */
246 3 END;
247 2 CALL DELAY(50); /* WAIT 50mS */
248 2 CALL PORT#OUT(CE1,80H); /* STOP */
249 2 CALL DELAY(360); /* WAIT 360mSEC */
250 2 CALL PORT#OUT(CE1,00H); /* WRITE MODE OFF */
251 2 RECORD=RECORD+1;
252 2 END TAPe#BLOCK#DUMP;

TAPE#BLOCK#READ: PROCEDURE;

DISABLE;
253 2 CALL INST#OUT(CLEAR#DISP);
254 2 CALL TEXT#OUT(.MES#44,LENGTH(MES#44)); /* TAPE BLOCK READ */
255 2 CALL INST#OUT(HOME#2);
256 2 CALL TEXT#OUT(.MES#43,LENGTH(MES#43)); /* IN PROGRESS */
257 2 CALL TAPE#STROBE#OUT(CE4); /* RESET EOT & USRT */
258 2 CALL TAPE#STROBE#IN(CE5); /* RESET OPTICAL */
259 2 CALL TAPE#STROBE#IN(CE5); /* TRANSPORT STATUS */
260 2 IF (#DATA AND 0A0H)=00H THEN /* TAPE ERROR? */
261 2 THEN DO;
262 3 IF RECORD < LAST#RECORD THEN /* POSITION TAPE AT GAP IMMEDIATELY */
263 3 THEN DO;
264 4 K=(LAST#RECORD - RECORD)+1;
265 4 CALL PORT#OUT(CE1,REV);
266 4 CALL POSITION#TAPE;
267 4 DO WHILE (#DATA AND 0A2H)=02H;
268 4 CALL TAPE#STROBE#IN(CE5); /* DO WHILE GAP PRESENT */
269 4 END;
270 4 CALL DELAY(50);
271 4 CALL PORT#OUT(CE1,0H);
272 4 CALL DELAY(360);
273 4 CALL PORT#OUT(CE1,FUD); /* DO WHILE DATA PRESENT */
274 4 CALL TAPE#STROBE#IN(CE5);
275 4 END;
276 4 ELSE DO;
277 4 K=RECORD -(LAST#RECORD+1);
278 4 CALL PORT#OUT(CE1,FUD);
279 4 IF K > 0 THEN CALL POSITION#TAPE;
280 4 END;
281 3 ELSE DO;
282 3 CALL PORT#OUT(CE1,0); /* STOP */
283 3 TAPE#ERROR#FLAG=0FFH;
284 3 RETURN;
285 3 END;
286 3 END;
287 2 DO;
288 3 CALL PORT#OUT(CE1,0);
289 3 TAPE#ERROR#FLAG=0FFH;
290 3 RETURN;
291 3 END;
292 2 CALL TAPE#STROBE#OUT(CE4); /* RESET USRT (START SYNC WORD SEARCH) */
293 2 CALL PORT#OUT(CE2,SYNC); /* SET SYNC WORD */
294 2 CALL RD4;
295 2 CALL TAPE#STROBE#IN(CE7); /* READ PREAMBLE */
296 2 DO I = 0 TO 15;
297 3 CALL RD4;
298 3 CALL TAPE#STROBE#IN(CE7);
BUFFER(1) = T#DATA;
300  3  END;
301  2  BUFF#POINT=16;
302  2  NO#OF#CHANS=BUFFER(13);
303  2  NO#OF#SCANS=BUFFER(14);
304  2  NO#OF#SCANS=SHL(NO#OF#SCANS,8)+BUFFER(15); /* FORM 16 BIT WORD */
305  2  DO N=0 TO NO#OF#SCANS-1;
306  3  DO I = 0 TO SHL(NO#OF#CHANS,1)-1; /* (NO#OF#CHANS * 2) - 1 */
307  4  CALL RDA; /* DATA FOR EACH CHAN = 2 BYTES (12 BITS) */
308  4  CALL TAPE#STROBE#IN(CE7); /* TRANSFER DATA FROM TAPE TO BUFFER */
309  4  BUFFER(BUFF#POINT) = T#DATA;
310  4  CALL TAPE#STROBE#IN(CE5); /* TRANSPORT STATUS */
311  4  IF (T#DATA AND 0A0H)=00H THEN /* TAPE ERROR? */
312  4  BUFF#POINT=BUFF#POINT+1;
313  4  ELSE DO;
314  5  CALL PORT#A#OUT(CE1,0); /* STOP */
315  5  TAPE#ERROR#FLAG=0FFH;
316  5  RETURN;
317  5  END;
318  4  END;
319  3  END;
320  2  CALL TAPE#STROBE#IN(CE5); /* TRANSPORT STATUS */
321  2  DO WHILE (T#DATA AND 0A2H)=00H; /* WAIT FOR GAP */
322  3  CALL TAPE#STROBE#IN(CE5); /* TRANSPORT STATUS */
323  3  END:
324  2  CALL DELAY(50); /* STOP */
325  2  CALL PORT#A#OUT(CE1,00H); /* WAIT 360mSEC */
326  2  CALL DELAY(360); /*wait */
327  2  LAST#RECORD=RECORD; /* UPDATE LAST#RECORD */
328  2  RECORD=RECORD+1;
329  2  CALL RECORD#DISP;
330  2  EN#ABLE;
331  2  END TAPE#BLOCK#READ;
332  1  RECORD#SELECT; PROCEDURE; /* SELECT DESIRED RECORD (1-99) */
333  2  CALL INST#OUT(CLEAR#DISP);
334  2  CALL INST#OUT(CURSOR#ON);
335  2  CALL TEXT#OUT(MES#37,LENGTH(MES#37)); /* SET RECORD TO BE */
336  2  CALL INST#OUT(HOME#2);
337  2  CALL TEXT#OUT(MES#38,LENGTH(MES#38)); /* READ */
338  2  CALL BIN#TO#ASCII(RECORD,2);
339  2  CALL TEXT#OUT(CHR,2);
340  2  CALL INST#OUT((POSITION-2) OR 80H);
341  2  CALL KEY#IN;
342  2  CALL ASCII#TO#BIN(CHR,2);
343  2  RECORD=BIN#DATA;
344  2  CALL INST#OUT(CURSOR#OFF);
345  2  END RECORD#SELECT;
346  1  TAPE#READ#INIT; PROCEDURE; /* TRANSPORT STATUS */
347  2  CALL TAPE#STROBE#IN(CE5); /* TAPE LOADED? */
348  2  IF (T#DATA AND 80H)=00H THEN DO;
349  3  TAPE#ERROR#FLAG=0FFH;
350  3  RETURN;
408 2 TAPE#MODE#FLAG=0;
409 2 END TAPE#PLAY;

410 1 TAPE#WRITE#INIT: PROCEDURE;
411 2 CALL INST#OUT(CLEAR#DISP);
412 2 CALL TEXT#OUT(MES#47,LENGTH(MES#47)); /* WRITE INITIALIZE */
413 2 CALL INST#OUT(HOME#2);
414 2 CALL TEXT#OUT(MES#43,LENGTH(MES#43)); /* IN PROGRESS */
415 2 CALL REWIND;
416 2 CALL GET#OFF#LEADER(0);
417 2 RECORD=0;
418 2 CALL TAPE#BLOCK#DUMP; /* INITIAL DUMMY DUMP */
419 2 END TAPE#WRITE#INIT;

420 1 TAPE#ERROR:PROCEDURE PUBLIC;
421 2 CALL TAPE#ERROR#INES;
422 2 DO WHILE (T#DATA AND 80H)=0;
423 3 CALL TAPE#STROBE#IN(CE5); /* DO WHILE CASSETTE LOADED */
424 3 END;
425 2 CALL TAPE#ERROR#INES;
426 2 KEY=9;
427 2 DO WHILE (KEY)=0) AND (TAPE#ERROR#FLAG=OFFH);
428 3 CALL TAPE#STROBE#IN(CE5); /* TRANSPORT STATUS */
429 3 DISABLE;
430 3 IF (T#DATA AND 80H)=0 THEN /* NEW CASSETTE LOADED? */
431 3 DO;
432 4 TAPE#ERROR#FLAG=0;
433 4 CALL TAPE#WRITE#INIT;
434 4 END;
435 3 ENABLE;
436 3 CALL DELAY(1);
437 3 END;
438 2 END TAPE#ERROR;

439 1 END TAPE#MODULE;

MODULE INFORMATION:

CODE AREA SIZE = 0894H 2196D
VARIBLE AREA SIZE = 0013H 1D
MAXIMUM STACK SIZE = 0008H 8D
477 LINES READ
0 PROGRAM ERROR(S)

END OF PL/M-80 COMPILATION
DATA ACQUISITION & LOGGING MODULE

DISPLAYS BIPOLAR OUTPUT IN mVOLTS (+/- 5V FULL SCALE)

DECLARE DEC LITERALLY "DECLERE";
DEC LIT LITERALLY "LITERALLY";
DEC CLEAR&DISP LIT '01H';
DEC RETURN&HOME LIT '02H';
DEC CURSOR&OFF LIT '0CH';
DEC HOME&2 LIT '0CH';
DEC (1,8,N) BYTE;
DEC TEMP&10 ADDRESS;
DEC TEMP&10 ADDRESS;
DEC TEMP&11 ADDRESS;
DEC DATA&32H (32) BYTE; /* TEMP STORAGE OF LOGGED DATA FOR ONE SCAN (HI BYTE) */
DEC DATA&30H (32) BYTE; /* (LO BYTE) */
DEC ADDR DATA ADDRESS;
DEC CHAR&16 BYTE EXTERNAL;
DEC CHAN BYTE;
DEC NOOF &SCANS ADDRESS EXTERNAL;
DEC NOOF &CHANS BYTE EXTERNAL;
DEC SCAN RATE BYTE EXTERNAL;
DEC BLANK (*) BYTE DATA('');
DEC MESS&6 (*) BYTE DATA('EXIT');
DEC MESS&16 (*) BYTE DATA('ENTRY');
DEC MESS&19 (*) BYTE DATA('7F',',','7EH');
DEC MESS&23 (*) BYTE DATA('CHAN.');
DEC MESS&24 (*) BYTE DATA('()');
DEC FLAG&1 BYTE EXTERNAL;
DEC RST&55#FLAG BYTE EXTERNAL;
DEC DISP BYTE EXTERNAL;
DEC KEY BYTE EXTERNAL;
DEC POSITION BYTE EXTERNAL;
DEC REG&4 BYTE EXTERNAL;
DEC REG&2 BYTE EXTERNAL;
DEC BUFFER (16384) BYTE PUBLIC AT (8000H);
DEC BUFFER&POINTER ADDRESS;
DEC TAPE&MODE&FLAG BYTE EXTERNAL;
DEC IEE&MODE&FLAG BYTE EXTERNAL;
DEC TRIG&TYPE BYTE EXTERNAL;

INST#OUT: PROCEDURE (INSTRUCTION) EXTERNAL;
DEC INSTRUCTION BYTE;
END INST#OUT;

C#OUT: PROCEDURE (CHAR) EXTERNAL;
DEC CHAR BYTE;
END C#OUT;
TEXT@OUT:  PROCEDURE(P4,SIZE)EXTERNAL;
DEC P4 ADDRESS;
DEC SIZE BYTE;
END TEXT@OUT;

MENU: PROCEDURE(P1,P2,P3)EXTERNAL;
DEC (P1,P2,P3)ADDRESS;
END MENU;

BIN@TO@ASCII:PROCEDURE (READING,SIZE) EXTERNAL;
DEC SIZE BYTE;
DEC READING ADDRESS;
END BIN@TO@ASCII;

CHAN@D@ISP$SELECT:PROCEDURE EXTERNAL;
END CHAN@D@ISP$SELECT;

TAPE@B@LOCK$D@UMP:PROCEDURE EXTERNAL;
END TAPE@B@LOCK$D@UMP;

RTC@DATA:PROCEDURE (A) EXTERNAL;
DEC A BYTE;
END RTC@DATA;

D@AC@INIT: PROCEDURE PUBLIC;
OUTPUT(57H)=80H; /* INIT. DATA ACQUISITION BOARD 8255 */
END D@AC@INIT;

MUX$35: PROCEDURE;
DEC A BYTE;
CHAN = CHAN+1; /* INC. CHANNEL COUNTER */
OUTPUT(55H)=CHAN;
A=A+1;
/* DUMMY TASK */
A=A+1;
END MUX$35;

LOG@DISP:PROCEDURE (DISP,TEST) BYTE;
DEC (DISP,TEST) BYTE;
CALL INST@OUT(CLEAR@DISP);
CALL INST@OUT(CURSOR@OFF);
CALL TEXT@OUT(MES$23,LENGTH(MES$23)); /* CHAN.H */
CALL BIN@TO@ASCII(DISP,2);
CALL TEXT@OUT(.CHAR,2);
DO CASE TEST;
CALL MENU(.BLANK,.BLANK,MES$6);
CALL MENU(.BLANK,.MES$16,.MES$16);
END;
END LOG@DISP;

LOG@DATA@NORMALIZE:PROCEDURE;
CALL INST@OUT(089H);
TEMP@HID = (((Dat@H@HID(DISP-1)) AND 0FH)×1000)×1000;
TEMP@HI = (((Dat@H@HLO(DISP-1),4))×1000)×1000;
TEMP@LO = (((SHR(Dat@H@HLO(DISP-1),4))×1000)×1000;
ADD@DATA = TEMP@HI + SHR(TEMP@HID,4) + SHR(TEMP@LO,3);
/* ADD@DATA RANGE 0-10000 m */
DISABLE;
IF (SHL(LOU(TEMP#10),4) + LOU(TEMP#10)) > 50H THEN
    AD#DATA = AD#DATA + 1; /* ROUND UP LSB */
END;
IF IEEE#MODE#FLAG = 1 THEN /* IF IEEE MODE SELECTED THEN 0/P TO IEEE 488 BUS */
DO:
B = 0H;
DO WHILE B <> 20H;
    B = INPUT(01H) AND 02H; /* WAIT FOR NRFD=1 AND DRB=0 FROM IEEE BUS */
    IF KEY = 2 THEN B = 20H;
END;
OUTPUT(60H) = NOT(HIGH(AD#DATA)); /* OUTPUT HI-BYTE TO IEEE */
B = 0H;
DO WHILE B <> 20H;
    B = INPUT(01H) AND 02H; /* WAIT FOR NRFD=1 AND DRB=0 FROM IEEE BUS */
    IF KEY = 2 THEN B = 20H;
END;
OUTPUT(60H) = NOT(LOW(AD#DATA)); /* OUTPUT LO-BYTE TO IEEE */
DISABLE;
END;
IF AD#DATA = 5000 THEN /* 0/P LOGGED VALUE (BI-POLAR) OF SELECTED */
    DO:
    CALL C#OUT('+'); /* CHANNEL (IN mvPLUS SIGN) TO DISPLAY */
    AD#DATA = AD#DATA - 5000;
END;
ELSE
    DO:
    CALL C#OUT('-');
    AD#DATA = 5000 - AD#DATA;
END;
CALL BIN#TO#ASCII(AD#DATA,4);
ENABLE;
CALL TEXT#OUT('.CHAR,4);
CALL C#OUT('m');
CALL C#OUT('v');
END LOG#DATA#NORMALIZE;
DATA#QUALIFY: PROCEDURE;
DEC THRESHOLD LIT '61'; /* APPROX 3mG OR 150mV */
DEC (MAX,MIV,FILT#DATA) ADDRESS;
BUFF#POINT = (DISP-1); /* INIT BUFF POINT */
AD#DATA = (BUFFER(BUFF#POINT)*16) + SHR(BUFFER(BUFF#POINT+1),4); /* RIGHT JUSTIFY DATA */
MIN,MIV,FILT#DATA = AD#DATA; /* INIT VARIABLES */
DO I = 0 TO (NOOF#SCANS - 1);
    AD#DATA = (BUFFER(BUFF#POINT)*16) + SHR(BUFFER(BUFF#POINT+1),4); /* RIGHT JUSTIFY DATA */
    FILT#DATA = (AD#DATA * 3 / 10) + (FILT#DATA * 7 / 10); /* R/C FILTER CALC */
    BUFF#POINT = BUFF#POINT + (NOOF#CHANs * 2); /* INC. BUFF POINT */
    IF Filt#DATA > MAX THEN MAX = Filt#DATA;
    IF Filt#DATA < MIN THEN MIN = Filt#DATA;
END;
IF MAX - MIN > THRESHOLD THEN /* IF THRESHOLD EXCEEDED THEN DATA CAN BE DUMPED */
CALL TAPE#BLOCK#DUMP;
END DATA#QUALIFY;
LOG#SCAN: PROCEDURE PUBLIC;
KEY = 0FH;
BUFF*POINT=0;
OUTPUT(55H)=0;
CHAN=0;
CALL LOGICDISP(DISP,0);
REG# = SCAN RATE AND OFFH;
DISABLE;
REG# = 42H;
CALL RTCDATA(0);
DO NO=0 TO (NOOFSCANS-1):  
IF KEY<>2 THEN DO:  
RST55#FLAG=0;
DO WHILE RST55#FLAG=0;
ENABLE;
DISABLE;
DO NO=0 TO (NOOFCHANS-1):
OUTPUT (50H)=0;
CALL MUX#35;
DATA#H(1) = INPUT(50H);
DATA#L(1) = INPUT(51H);
BUFFER(BUFF*POINT)=DATA#H(1);
BUFFER(BUFF*POINT+1)=DATA#L(1);
BUFF*POINT=BUFF*POINT+2;
END:
CHAN = 0;
OUTPUT(55H)=0;
ENABLE;
CALL LOGICDATA#NORMALIZE;
END:
END;
REG# = 02H;
DISABLE;
IF (TAPE#MODE#FLAG=0) AND (KEY<>2) THEN  
CALL DATA#QUALIFY;
RST55#FLAG=0;
IF TRIG#TYPE='A' THEN  
REG# = 22H;
END LOGICSCAN:

DISACOSTEST:PROCEDURE PUBLIC:
KEY,FLAG#0;
OUTBUFFER(55H)=DISP-1;
CALL LOGICDISP(DISP,1);
DO WHILE KEY<>2:
DISABLE;
OUTPUT(50H)=0;
CALL MUX#35;
ENABLE;
OUTPUT(55H)=DISP-1;
DATA#H(DIS-1)=INPUT(50H);
DATA#L(DIS-1)=INPUT(51H);
CALL LOGICDATA#NORMALIZE;
DISABLE;
IF FLAG#0=OFFH THEN DO:
202 4 IF KEY=1 THEN CALL Chan##Disp##Select; /* ENTER NEW CHANNEL TO BE DISPLAYED (Disp) */
204 4 IF KEY=12 THEN Disp=Disp-1; /* DECREMENT Disp */
206 4 IF KEY=15 THEN Disp=Disp+1; /* INCREMENT Disp */
208 4 OUTPUT(55H)=Disp-1; /* O/P NEW CHANNEL TO MUX */
209 4 CALL LOG##Disp##Disp(1);
210 4 FLG1=0;
211 4 END;
212 3 END;
213 2 END D##ACO##TEST;
214 1 END D##ACO##MODULE;

MODULE INFORMATION:

CODE AREA SIZE = 0460H 1128D
VARIABLE AREA SIZE = 0057H 07D
MAXIMUM STACK SIZE = 0008H 8D
229 LINES READ
0 PROGRAM ERROR(S)

END OF PL/M-80 COMPILATION
APPENDIX C

Software Version V1.4
Program listings

Note: Only those modules which differ from Version V1.2 are included.
### Data Logger Main Module

```plaintext
/*

DATA LOGGER MAIN MODULE

*/

1 DATA#LOGGER:
2    DO:
3      DECLARE DEC LITERALLY 'DECLARE';
4      DEC LIT LITERALLY 'LITERALLY';
5      DEC LEN LIT 'LENGTH';
6      DEC CLEAR#DISP LIT '01H';
7      DEC RETURN#HOME LIT '02H';
8      DEC HOME#2 LIT '0C0H';
9      DEC CURSOR#ON LIT '0E0H';
10     DEC CURSOR#OFF LIT '0CH';
11     DEC (POSITION) BYTE EXTERNAL;
12     DEC RST##55##FLAG BYTE EXTERNAL;
13     KEY##INIT: PROCEDURE EXTERNAL;
14     END KEY##INIT;
15     RTC##INIT: PROCEDURE EXTERNAL;
16     END RTC##INIT;
17     LCD##INIT: PROCEDURE EXTERNAL;
18     END LCD##INIT;
19     TAPE##INIT: PROCEDURE EXTERNAL;
20     END TAPE##INIT;
21     D##ACO##INIT: PROCEDURE EXTERNAL;
22     END D##ACO##INIT;
23     LOG##SCAN: PROCEDURE EXTERNAL;
24     END LOG##SCAN;

/* MAIN PROGRAM */

25    CALL LCD##INIT;
26    CALL KEY##INIT;
27    CALL RTC##INIT;
28    CALL TAPE##INIT;
29    CALL D##ACO##INIT;
30    CALL TAPE##ERROR##FLAG=0;
31    CALL RST##55##FLAG=0;
32    DO WHILE 1;
33    CALL LOG##SCAN;
34    END;
35    END DATA#LOGGER;
*/
```
MODULE INFORMATION:

CODE AREA SIZE = 0024H  36D
VARIABLE AREA SIZE = 0000H  0D
MAXIMUM STACK SIZE = 0002H  2D
50 LINES READ  
0 PROGRAM ERROR(S)

END OF FL/M-80 COMPILATION
REAL-TIME CLOCK MODULE

DECLARE CLEAR#DISP LITERALLY '01H';
DECLARE DISP#CURSOR#ON LITERALLY '0EH';
DECLARE BLINK LITERALLY '0F8';
DECLARE HOME#2 LITERALLY '0CH';
DECLARE MESS#14(*) BYTE DATA ('ALARM');
DECLARE DATE#MENU (**) BYTE DATA ('SET DATE DDDMMYY');
DECLARE TIME#MENU (**) BYTE DATA ('SET TIME HHMMSS');
DECLARE (FLAG1,KEY) BYTE EXTERNAL;
DECLARE RTC#DATE(6) BYTE PUBLIC;
DECLARE RTC#TIME(6) BYTE PUBLIC;
DECLARE (IN) BYTE;
DECLARE CHAR (16) BYTE EXTERNAL;
DECLARE POSITION BYTE EXTERNAL;
DECLARE TEMP BYTE;
DECLARE SECONDS BYTE AT (2000H);
DECLARE MINUTES BYTE AT (2002H);
DECLARE HOURS BYTE AT (2004H);
DECLARE DAY OF WEEK BYTE AT (2006H);
DECLARE DAY OF MONTH BYTE AT (2007H);
DECLARE MONTH BYTE PUBLIC AT (2008H);
DECLARE YEAR BYTE AT (2009H);
DECLARE SECONDS#ALARM BYTE AT (2011H);
DECLARE MINUTES#ALARM BYTE AT (2031H);
DECLARE HOURS#ALARM BYTE AT (2055H);
DECLARE REG#1 BYTE PUBLIC AT (200AH);
DECLARE REG#2 BYTE PUBLIC AT (200BH);
DECLARE REG#3 BYTE AT (200CH);
DECLARE REG#4 BYTE AT (200DH);
DECLARE RST#55#FLAG BYTE PUBLIC; /* CLOCK INTERRUPT FLAG */
DECLARE SCAN#RATE BYTE EXTERNAL;

INST#OUT: PROCEDURE (INST) EXTERNAL;
DECLARE INST BYTE;
END INST#OUT;

$#MASK: PROCEDURE (MASK) EXTERNAL;
DECLARE MASK BYTE;
END $#MASK;

TEXT#OUT: PROCEDURE (P4,SIZE) EXTERNAL;
DECLARE P4 ADDRESS;
DECLARE SIZE BYTE;
END TEXT#OUT;
KEY*IN; PROCEDURE EXTERNAL;
END KEY*IN;

SET#LOG#PARAMETERS; PROCEDURE EXTERNAL;
END SET#LOG#PARAMETERS;

RST#55: PROCEDURE PUBLIC;
DECLARE C BYTE;
C = REG#C;
RST#55#FLAG=0FFH;
ENABLE;
END RST#55;

RTC#INIT: PROCEDURE PUBLIC;
DECLARE JUMP#55 BYTE AT (02CH);
DECLARE VECTOR#55 ADDRESS AT (02DH);
JUMP#55 = 0CH;
VECTOR#55 = RST#55;
REG#A = SCAN#RATE AND 0FH;
REG#B = 02H;
CALL S#MASK(01AH); /* SET INTERRUPT MASK (ENABLE 7.5 & 5.5,RESET 7.5 F/F) */

END RTC#INIT;

RTC#INIT#DATA: PROCEDURE;
DO I=0 TO 4 BY 2;
TEMP = (CHR(I)+1) AND 0FH OR SHL((CHR(I) AND 0FH),4); /* CONVERT 2 ASCII CHARs */
/* TO PACKED BCD */

DO CASE (I+N);
DAY#OF#MONTH = TEMP;
HOURS = TEMP;
MONTH = TEMP;
MINUTES = TEMP;
YEAR = TEMP;
SECONDS = TEMP;
END;
END RTC#INIT#DATA;

SET#DATE#TIME: PROCEDURE PUBLIC;
REG#B = 100000010B;
REG#A = 00H;
DO N = 0 TO I;
IF YEAR=56H THEN N=0;
DO CASE N;
DO I=0 TO 5;
CHR(I) = RTC#DATE(I);
END;
DO I=0 TO 5;
CHR(I) = RTC#TIME(I);
END;
END;
CALL INST#OUT(CLEAR#DISP);
CALL INST#OUT(84H);
DO CASE N;
CALL TEXT#OUT(.DATE#MENU,?);
CALL TEXT#OUT(.TIME#MENU,?);
END;
END;
94 3 CALL INST#OUT(HOME$+5); /* MOVE CURSOR TO LINE 2 POSITION 5 */
95 3 DO CASE N;
96 4 CALL TEXT#OUT(.DATE$MENU($),6); /* D0$MY */
97 4 CALL TEXT#OUT(.TIME$MENU($),6); /* HH:MM */
98 4 END;
99 3 CALL INST#OUT((POSITION-6)OR 80H); /* MOVE CURSOR -6 PLACES */
100 3 CALL INST#OUT(BLINK$); /* BLINK ON */
101 3 CALL KEY$IN;
102 3 CALL RTC#INIT$DATA;
103 3 IF YEAR=56H THEN /* TEST FOR PASSWORD(****56) */
104 3 CALL SET#LOG$PARAMETERS;
105 3 END;
106 2 REG$B = 00000010B; /* GET OUT OF CLOCK SET MODE */
107 2 CALL INST#OUT(DISP$CURSOR$ON$); /* BLINK OFF */
108 2 CALL INST#OUT(CLEAR$DISP$);
109 2 END SET$DATE$TIME;
110 1 RTC#DATA$PROCEDURE PUBLIC;
111 2 DECLARE (I,N) BYTE;
112 2 DECLARE ASCII (6) BYTE;
113 2 DIS$ABLE;
114 2 DO WHILE (REG$A AND 80H)<<0;
115 3 END;
116 2 DO N =0 TO 1;
117 3 DO I =0 TO 4 BY 2;
118 4 DO CASE (I+N);
119 5 TEMP = DAY$OF$MONTH;
120 5 TEMP = DAY$OF$MONTH;
121 5 TEMP = HOUR$;
122 5 TEMP = MINUTES;
123 5 TEMP = YEAR$;
124 5 TEMP = SECONDS;
125 5 END;
126 4 ASCII(I) = SHR(TEMP,4) OR 30H; /* CONVERTS PACKED BCD TO 2 ASCII CHARACTERS */
127 4 ASCII(I+1) = (TEMP AND 0FH) OR 30H; /* */
128 4 END;
129 3 END CASE N;
130 4 DO I = 0 TO 5;
131 5 RTC$DATE(I) = ASCII(I);
132 5 END;
133 4 DO I = 0 TO 5;
134 5 RTC$TIME(I) = ASCII(I);
135 5 END;
136 4 END;
137 3 END;
138 2 END$ENABLE;
139 2 END RTC$DATA;
140 1 END REAL$TIME$CLOCK$MODULE;

MODULE INFORMATION:

CODE AREA SIZE = 0348H 840D
/*
 *  SET LOG PARAMETER MODULE
 */

1 SET#LOG#PARAMETER#MODULE:
  DO:
  2 1 DECLARE DEC LITERALLY 'DECLARE';
  3 1 DEC LIT LITERALLY 'LITERALLY';
  4 1 DEC LEN LIT 'LENGTH';
  5 1 DEC CLEAR#DISP LIT '01H';
  6 1 DEC RETURN#HOME LIT '02H';
  7 1 DEC HOME#2 LIT '060H';
  8 1 DEC CURSOR#LIT LIT '0EH';
  9 1 DEC MES#11(*) BYTE DATA ('SET CHAN. TO BE');
 10 1 DEC MES#12(*) BYTE DATA ('DISPLAYED:');
 11 1 DEC MES#18(*) BYTE DATA ('SET THRESHOLD:');
 12 1 DEC MES#24(*) BYTE DATA ('mV');
 13 1 DEC MES#25(*) BYTE DATA ('SCANS');
 14 1 DEC MES#26(*) BYTE DATA ('SET PRE-TRIGGER:');
 15 1 DEC CHAR (16) BYTE EXTERNAL;
 16 1 DEC POSITION BYTE EXTERNAL;
 17 1 DEC PRE#TRIG BYTE PUBLIC AT (2010H); /* MAX 255 SCANS */
 18 1 DEC NO#OF#CHANS BYTE PUBLIC AT (2011H); /* MAX 32 CHANNELS */
 19 1 DEC SCAN#RATE BYTE PUBLIC AT (2012H); /* 15=500ms,14=250ms,13=125ms etc */
 20 1 DEC NO#OF#SCANS ADDRESS PUBLIC AT (2013H); /* MAX 64K SCANS */
 21 1 DEC DIS#P BYTE PUBLIC AT (2015H); /* CHANNEL TO BE DISPLAYED */
 22 1 DEC THRESHOLD ADDRESS PUBLIC AT (2016H); /* EI-POLAR TRIGGER LEVEL IN mV */
 23 1 DEC BIN#DATA ADDRESS EXTERNAL;

24 1 INST#OUT: PROCEDURE (INST) EXTERNAL;
 25 2 DEC INST BYTE;
 26 2 END INST#OUT;

27 1 C#OUT: PROCEDURE (CHAR) EXTERNAL;
 28 2 DEC CHAR BYTE;
 29 2 END C#OUT;

30 1 TEXT#OUT: PROCEDURE(P4,SIZE) EXTERNAL;
 31 2 DEC P4 ADDRESS;
 32 2 DEC SIZE BYTE;
 33 2 END TEXT#OUT;

34 1 BINT#TO#ASCII: PROCEDURE(READING,SIZE) EXTERNAL;
 35 2 DEC SIZE BYTE;
 36 2 DEC READING ADDRESS;
 37 2 END BINT#TO#ASCII;

38 1 ASCII#TO#BIN: PROCEDURE(ASCII#ADR,SIZE) EXTERNAL;
 39 2 DEC ASCII#ADR ADDRESS;
 40 2 DEC SIZE BYTE;
41 2   END ASCII10BIN;
42 1   KEY*ENTRY; PROCEDURE EXTERNAL;
43 2   END KEY*ENTRY;
44 1   KEY*IN; PROCEDURE EXTERNAL;
45 2   END KEY*IN;
46 1   LOG*PARAM2; PROCEDURE;
47 2   CALL TEXT*OUT(.CHAR,2);
48 2   CALL INST*OUT((POSITION-2) OR 80H);
49 2   CALL KEY*IN;
50 2   END LOG*PARAM2;
51 1   LOG*PARAM4; PROCEDURE;
52 2   CALL TEXT*OUT(.CHAR,4);
53 2   CALL INST*OUT((POSITION-4) OR 80H);
54 2   CALL KEY*IN;
55 2   END LOG*PARAM4;
56 1   MENU: PROCEDURE(P1,P2,P3) EXTERNAL;
57 2   DEC (P1,P2,P3)ADDRESS;
58 2   END MENU;
59 1   CHW*DISP*SELECT: PROCEDURE PUBLIC;
60 2   CALL INST*OUT(CLEAR*DISP);
61 2   CALL INST*OUT(CURSOR*OFF);
62 2   CALL TEXT*OUT(.MES$11,LEN(MES$11)); /* SET CHANNEL TO BE */
63 2   CALL INST*OUT(HOME$2);
64 2   CALL TEXT*OUT(.MES$12,LEN(MES$12)); /* DISPLAYED */
65 2   CALL C$OUT(' /');
66 2   CALL C$OUT('#');
67 2   CALL BIN$TOASCII(DISP,2);
68 2   CALL LOG*PARAM$2;
69 2   CALL ASCII10BIN(.CHAR,2);
70 2   DISP=BIN$DATA;
71 2   END CHW*DISP*SELECT;
72 1   SET*LOG*PARAMETERS: PROCEDURE PUBLIC;
73 2   CALL INST*OUT(CLEAR*DISP);
74 2   CALL INST*OUT(RETURN*HOME);
75 2   CALL TEXT*OUT(.MES$18,LEN(MES$18)); /* SET THRESHOLD */
76 2   CALL INST*OUT(OCAN);
77 2   CALL TEXT*OUT(.MES$24,LEN(MES$24)); /* [mV] */
78 2   CALL INST*OUT(OCSH);
79 2   CALL BIN$TOASCII(THRESHOLD$4);
80 2   CALL LOG*PARAM$4;
81 2   CALL ASCII10BIN (.CHAR,4);
82 2   THRESHOLD$4=(BINS$DATA); /* BIPOLAR TRIGGER LEVEL IN mV */
83 2   CALL INST*OUT(CLEAR*DISP);
84 2   CALL INST*OUT(RETURN*HOME);
85 2   CALL TEXT*OUT(.MES$26,LEN(MES$26)); /* SET PRE-TRIGGER */
86 2   CALL INST*OUT(OCAN);
87 2   CALL TEXT*OUT(.MES$25,LEN(MES$25)); /* SCANS */
88 2   CALL INST*OUT(OCSH);
CALL BIN\$TO\$ASCII(PRE\$TRIG,2);

CALL LOG\$PARAM\$2;

CALL ASCII\$TO\$BIN\$(CHAR,2);

PRE\$TRIG=BIN\$DATA;

CALL CHANNEL\$DISPLAY\$SELECT;

/\* SET CHANNEL TO BE DISPLAYED */

END SET\$LOG\$PARAMETERS;

END SET\$LOG\$PARAMETER\$MODULE;

MODULE INFORMATION:

CODE AREA SIZE = 0138H  3150

VARIABLE AREA SIZE = 0000H 0D

MAXIMUM STACK SIZE = 0000H  60

115 LINES READ
0 PROGRAM ERROR(S)

END OF PL/I\$80 COMPILATION
/* Data Acquisition & Logging Module */
/* Displays bipolar output in mVOLTS (+/- 5V Full Scale) */

D*A*C#MODULE:

DO:

1 DEC LIT LITERALLY 'DECLARE';
2 DEC LIT LITERALLY 'LITERALLY';
3 DEC CLEAR#DISP LIT '01H';
4 DEC RETURN#HOME LIT '02H';
5 DEC CURSOR#OFF LIT '0CH';
6 DEC HOME#2 LIT '0OH';
7 DEC (1,B,N,T) BYTE;
8 DEC TEMP#LO ADDRESS;
9 DEC TEMP#MO ADDRESS;
10 DEC TEMP#HI ADDRESS;
11 DEC TEMP#HI ADDRESS;
12 DEC DATA#H(32) BYTE; /* TEMP STORAGE OF LOGGED DATA FOR ONE SCAN (HI BYTE) */
13 DEC DATA#L(32) BYTE; /* (LO BYTE) */
14 DEC AD#DATA ADDRESS;
15 DEC CHAR(16) BYTE EXTERNAL;
16 DEC CHAN BYTE;
17 DEC NO#OF#SCANS ADDRESS EXTERNAL; /* CHANNEL COUNTER FOR MIX */
18 DEC NO#OF#CHANS BYTE EXTERNAL;
19 DEC SCAN#RATE BYTE EXTERNAL;
20 DEC BLANK (*) BYTE DATA(' ');
21 DEC MESS1 (*) BYTE DATA(OF3H,'SET',OF3H);
22 DEC MESS2 (*) BYTE DATA('TEST');
23 DEC MESS3 (*) BYTE DATA('EXIT');
24 DEC MESS4 (*) BYTE DATA('ENTRY');
25 DEC MESS5 (*) BYTE DATA('7EH');
26 DEC MESS6 (*) BYTE DATA('LOG-T');
27 DEC MESS7 (*) BYTE DATA('CHAN,M');
28 DEC MESS8 (*) BYTE DATA(' ');/*
29 DEC FLAGO BYTE EXTERNAL;
30 DEC RST#5#FLAG BYTE EXTERNAL;
31 DEC DISP BYTE EXTERNAL;
32 DEC KEY BYTE EXTERNAL;
33 DEC POSITION BYTE EXTERNAL;
34 DEC REG#A BYTE EXTERNAL;
35 DEC REG#B BYTE EXTERNAL;
36 DEC BUFFER (16384) BYTE PUBLIC AT (8000H);
37 DEC BUFF#POINT ADDRESS PUBLIC;
38 DEC BUFF#SIZE ADDRESS PUBLIC;
39 DEC THRESHOLD ADDRESS EXTERNAL; /* TRIGGER THRESHOLD IN mV */
40 DEC FILT#DATA ADDRESS;
41 DEC LAST#BUFF#POINT ADDRESS;
42 DEC DATES#MES (*) BYTE DATA('JANFEBMARAPRMAYJUNJULAOUGSEPOTHOUDEC');
43 DEC RTC#DATE (6) BYTE EXTERNAL;
44 DEC RTC#TIME (6) BYTE EXTERNAL;
45 DEC MOUTH#BYTE EXTERNAL;
DEC (DISP,TEST) BYTE;
CALL INST#OUT(CLEAR#DISP);
CALL INST#OUT(CURSOR#OFF);
CALL TEXT#OUT(HES#23,LENGTH(HES#23)); /* CHAN, # */
CALL BIN#TO#ASCII(DISP,2);
CALL TEXT#OUT(CHAR,2);
DO CASE TEST;
CALL MENU(.BLANK,.BLANK,HES#6); /* EXIT */
CALL MENU(.BLANK,HES#16,HES#6); /* ENTRY EXIT */
END;
END LOG#DISP;

LOG#DATA#NORMALIZE;PROCEDURE;
CALL INST#OUT(09H);
TEMP#HID = (((DATA#H(DISP-1) AND 0FH)*1000)/8)*5; /* NORMALIZE DATA FOR 10V I/P */
TEMP#HI = (((SHR(DATA#H(DISP-1),4))*1000)/8)*5;
TEMP#LO = (((SHR(DAT#A#L(DISP-1),4))*1000)/8)*5;
ADD#DATA = TEMP#HI + SHR(TEMP#HID,4) + SHR(TEMP#LO,8); /* ADD#DATA RANGE 0-10000 mV */
DISABLE;
IF (SHL(Low(TEMP#HID),4) + LOW(TEMP#LO)) >80H THEN
ADD#DATA = ADD#DATA + 1; /* ROUND UP LSB */
IF ADD#DATA = 5000 THEN /* 0/P LOGGED VALUE (BI-POLAR) OF SELECTED */
DO;
CALL C#OUT('+');
ADD#DATA = ADD#DATA - 5000;
END;
ELSE
DO;
CALL C#OUT ('-');
ADD#DATA = 5000 - ADD#DATA;
END;
CALL BIN#TO#ASCII(ADD#DATA,4);
ENABLE;
CALL TEXT#OUT(CHAR,4);
CALL C#OUT ('m');
CALL C#OUT ('/');
CALL C#OUT ('v');
END LOG#DATA#NORMALIZE;

D#ACT#TEST;PROCEDURE PUBLIC;
REG#B = 02H; /* DISABLE PIE (FROM RTC) */
KEY#FLG=0;
OUTPUT(55H)=DISP-1; /* O/P SELECTED CHANNEL TO MUX */
DO WHILE KEY<2;
DISABLE;
OUTPUT(5OH)=0;
CALL MUX#35;
ENABLE;
OUTPUT(55H)=DISP-1; /* O/P SELECTED CHANNEL TO MUX */
DATA#HI(DISP-1)=INPUT(50H); /* I/P TWO BYTES OF DATA */
DATA#LO(DISP-1)=INPUT(51H); /* FROM A/D CONVERTER */
CALL LOG#DATA#NORMALIZE;
DISABLE;
IF FLG#0 = 0FFH THEN
DO;
IF KEY=1 THEN CALL CHAN#DISP#SELECT; /* ENTER NEW CHANNEL TO BE DISPLAYED (DISP) */
END;
IF KEY=12 THEN DISP=DISP-1; /* DECREMENT DISP */
IF KEY=15 THEN DISP=DISP+1; /* INCREMENT DISP */
OUTPUT(S5H)=DISP-1; /* O/P NEW CHANNEL TO MUX */
CALL LOG*DISP(DISPl);
FLAGl=0;
END;
END D*AC0*TEST;

THRESH*CHECK: PROCEDURE;
TEMP*HI=LAST*BUFF*POINT + ((DISP-1)*2); /* POSITION DATA POINTER */
AD*DATA = (BUFFER(TEMP*HI))•16)+SHR(BUFFER(TEMP*HI+1),4); /* RIGHT JUSTIFY DATA */
IF FILT*DATA=FLAG=OH THEN
DO;
FILT*DATA=AD*DATAl
FILTER*DATA=FLAG=OFFH;
END;
ELSE
DO;
TEMP*LO=(FILT*DATA<>9); /* FILTER CALC. [Y=X*0.01 + Y*0.99 ] */
FILT*DATA=((AD*DATAlO)+TEMP*LO+(TEMP*LO / 10))/10; /* % MAX Rounding ERROR */
IF FILT*DATA = AD*DATA THEN
DO;
IF (FILT*DATA=AD*DATA)T THEN
DO;
KEY=1;
RETURN;
END;
END;
ELSE
IF (AD*DATA=FILT*DATA)T THEN
DO;
KEY=1;
RETURN;
END;
END;
END THRESH*CHECK;

TRIG*SCAN:PROCEDURE;
KEY=0FH;
CALL LOG*DISP(DISPl,0);
DO HwO TO (NO*OF*CHWNS - 1) - FRE*TRIGl;
IF KEY<>2 THEN
DO;
RST*S5#FLAG=0;
DO WHILE RST*S5#FLAG=0;
EN*BLE;
END;
DISABLE;
DO l=0 TO (NO*OF*CHWNS - 1);
OUTPUT(SOH)=0;
CALL MUX*35;
DATA#HI(l) = INPUT(SOH);
DATA#LO(l) = INPUT(SIH);
BUFFER(BUFF*POINT)=DATA#HI(l); /* STORE TWO BYTES OF DATA */
BUFFER(BUFF#POINT+1)=DATA#LO(1);  /* IN "BUFFER", */
BUFF#POINT=BUFF#POINT+2;  /* INCREMENT BUFFER POINTER */
END;
IF BUFF#POINT =BUFF#SIZE THEN
BUFF#POINT=0;
CHAN = 0;
OUTPUT(S5H)=0;
ENABLE;
CALL LOG#DATA#NORMALIZE;
END;
END;
REG#B = 02H;
DISABLE;
IF KEY <2 THEN
CALL TAPE#BLOCK#DUMP;
RST$S5$FLAG=0;
END TRIG#SCAN;

DISP#DATE#TIME: PROCEDURE;
DEC TEMP BYTE;
CALL INST#OUT(RETURN#HOME);
CALL RTC#DATA(0);
DO I=0 TO 1;
CALL C#OUT (RTC#DATE(I)));  /* O/P DAY OF MONTH */
END;
CALL C#OUT (' ');  /*
TEMP=MES(1);  /* CONVERT MONTH NUMBER TO */
IF TEMP>09H THEN
TEMP=TEMP-6;  /* A 3 CHARACTER WORD */
TEMP=(TEMP+3)-3;
DO I=0 TO 2;
CALL C#OUT (DATE#HES(I+TEMP));  /* O/P MONTH WORD */
END;
CALL C#OUT ('.');  /*
CALL C#OUT (' ');  /*
DO I=0 TO 5;
IF (I=2) OR (I=4) THEN
DO;
CALL C#OUT ('>');  /*
CALL C#OUT (RTC#TIME(I)));  /* O/P TIME */
END:
ELSE
CALL C#OUT (RTC#TIME(I));  /*
END:
END DISP#DATE#TIME;

LOG#SCAN:PROCEDURE PUBLIC;
IF TAPE#ERROR$FLAG > 0 THEN
CALL TAPE#ERROR';
CALL MENU#HES(1-HES#22-HES#3);  /* SET LOG-T TEST */
FILTER#DATA#FLAG=0H;
NO#OF#SCANS = 240;
NO#OF#CHANS = 27;
SCAN#RATE = 15;
BUFF#SIZE = (NO#OF#SCANS*NO#OF#CHANS*2);
T = (THRESHOLD*4)/10;  /* CONVERTS mV TO bits[250mV=100bits] */
250 2 KEY=OFH;
251 2 BUFF#POINT=0;
252 2 LAST#BUFF#POINT=0;
253 2 OUTPUT(S5H)=0;
254 2 CHAN=0;
255 2 REG#A = SCAN#RATE AND OFH;
256 2 DISABLE;
257 2 REG#B = 42H;
258 2 DO WHILE KEY >2;
259 3 RST#55#FLAG=0;
260 3 DO WHILE RST#55#FLAG=0;
261 4 ENABLE;
262 4 END;
263 3 DISABLE;
264 3 DO I=0 TO (NO#OF#CHANS - 1);
265 4 OUTPUT(S5H)=0;
266 4 CALL MUX#35;
267 4 DATA#HI(I) = INPUT(S5H);
268 4 DATA#LO(I) = INPUT(S5H);
269 4 BUFFER(BUFF#POINT)=DATA#HI(I);
270 4 BUFFER(BUFF#POINT)=DATA#LO(I);
271 4 BUFF#POINT=BUFF#POINT+2;
272 4 END;
273 3 IF BUFF#POINT = BUFF#SIZE THEN
274 3 BUFF#POINT=0;
275 3 CHAN = 1;
276 3 OUTPUT(S5H)=0;
277 3 CALL THRESH#CHECK;
278 3 IF KEY =) 1 THEN
279 3 CALL DISP#DATE#TIME;
280 3 END;
281 2 DO CASE (KEY);
282 3 CALL SET#DATE#TIME;
283 3 CALL TRIG#SCAN;
284 3 CALL DIS#ACO#TEST;
285 3 END;
286 2 END LOG#SCAN;
287 1 END DIS#ACO#MODULE;

MODULE INFORMATION:

CODE AREA SIZE = 061AH 1562D
VARIABLE AREA SIZE = 005BH 91D
MAXIMUM STACK SIZE = 000AH 10D
313 LINES READ
0 PROGRAM ERROR(S)

END OF PL/M-80 COMPILATION
PL/M-80 COMPILER

ISIS-II PL/M-80 V3.1 COMPILATION OF MODULE TAPEMODULE
OBJECT MODULE PLACED IN :F1:TAPEMOD.OBJ
COMPILER INVOKED BY: PLMSO :F1:TAPEMOD.SRC DEBUG PRINT(/F1:TAPEMOD.LST) DATE(22/08/85) PAGELength(59)

="/************ MFE 450 TAPE MODULE ************="

TAPEMODULE:
DO:
  1 DECLARE DEC LITERALLY 'DECLARE';
  2 DEC LIT LITERALLY 'LITERALLY';
  3 DEC PORT#A LIT '044H';
  4 DEC PORT#B LIT '045H';
  5 DEC PORT#C LIT '046H';
  6 DEC PORT#CON#REG LIT '047H';
  7 DEC IDLE LIT '0FFH';
  8 DEC CE1 LIT '0FDH';
  9 DEC CE2 LIT '0FBH';
 10 DEC CE3 LIT '0F7H';
 11 DEC CE4 LIT '0FFH';
 12 DEC CE5 LIT '0FH';
 13 DEC CE6 LIT '0FH';
 14 DEC CE7 LIT '0FH';
 15 DEC WRITEMODE LIT '08H';
 16 DEC FWD LIT '01H';
 17 DEC REV LIT '02H';
 18 DEC SEARCH#SPEED LIT '010H';
 19 DEC IPS#80 LIT '030H';
 20 DEC IPS#120 LIT '004H';
 21 DEC I ADDRESS;
 22 DEC N ADDRESS;
 23 DEC (J,K,T#DATA) BYTE;
 24 DEC BUFFER (16384) BYTE EXTERNAL;
 25 DEC SIZE ADDRESS EXTERNAL;
 26 DEC BUFFERPOINT ADDRESS EXTERNAL;
 27 DEC SYNC LIT '0AAH';
 28 DEC (KEY,FLAG) BYTE EXTERNAL;
 29 DEC CLEAR#DISP LIT '01H';
 30 DEC CURSOR#OFF LIT '0CH';
 31 DEC CURSOR#ON LIT '0EH';
 32 DEC C#H#16 BYTE EXTERNAL;
 33 DEC HOME#2 LIT '00CH';
 34 DEC POSITION BYTE EXTERNAL;
 35 DEC LASTRECORD BYTE;
 36 DEC RECORD BYTE AT (2016H); /
 37 DEC BIN#DATA ADDRESS EXTERNAL;
 38 DEC BLANK (*) BYTE DATA ('');
 39 DEC MESS#1 (*) BYTE DATA (OF3H, SET, 0F3H);
 40 DEC MESS#25 (*) BYTE DATA ('RECORD # ');
 41 DEC MESS#39 (*) BYTE DATA ('LOAD CASSETTE!');
 42 DEC MESS#40 (*) BYTE DATA (' END OF TAPE!');
 43 DEC MESS#41 (*) BYTE DATA ('WRITE PROTECTED!');
 44 DEC MESS#43 (*) BYTE DATA (' IN PROGRESS');
45 1 DEC MES#45 (*) BYTE DATA ('TAPE BLOCK WRITE');
46 1 DEC MES#47 (*) BYTE DATA ('WRITE INITIALIZE');
47 1 DEC RTC#DATE(*) BYTE EXTERNAL;
48 1 DEC RTC#TIME(*) BYTE EXTERNAL;
49 1 DEC NO#OFSCANS ADDRESS EXTERNAL;
50 1 DEC NO#OFCHANS BYTE EXTERNAL;
51 1 DEC SCAN#RATE BYTE EXTERNAL;
52 1 DEC TAPERROR#FLAG BYTE PUBLIC;
53 1 DEC PRE#TRIG BYTE EXTERNAL; /* PRE-TRIGGER VALUE IN SCANS */
54 1 INST#OUT: PROCEDURE(INSTRUCTION) EXTERNAL;
55 2 DEC INSTRUCTION BYTE;
56 2 END INST#OUT;
57 1 TEXT#OUT: PROCEDURE(P4,SIZE) EXTERNAL;
58 2 DEC SIZE BYTE;
59 2 DEC P4 ADDRESS;
60 2 END TEXT#OUT;
61 1 BIN#TO#ASCII: PROCEDURE(READING,SIZE) EXTERNAL;
62 2 DEC SIZE BYTE;
63 2 DEC READING ADDRESS;
64 2 END BIN#TO#ASCII;
65 1 MENU: PROCEDURE(P1,P2,P3) EXTERNAL;
66 2 DEC (P1,P2,P3) ADDRESS;
67 2 END MENU;
68 1 TAPE#INIT: PROCEDURE PUBLIC;
69 2 OUTPUT (47H)=82H; /* INIT NFE TAPE BOARD 8255 */
70 2 END TAPE#INIT;
71 1 TAPE#STROBE#OUT: PROCEDURE(INST); /* STROBES CONTROL LINE PASSED TO "INST" */
72 2 DEC INST BYTE;
73 2 OUTPUT(PORT#C) = INST; /* FROM HI TO LO TO HI */
74 2 OUTPUT(PORT#C) = IDLE;
75 2 END TAPE#STROBE#OUT;
76 1 TAPE#STROBE#IN: PROCEDURE(INST); /* STROBES DATA INTO T#DATA */
77 2 DEC INST BYTE;
78 2 OUTPUT(PORT#C) = INST;
79 2 T#DATA = INPUT(PORT#D);
80 2 OUTPUT(PORT#C) = IDLE;
81 2 END TAPE#STROBE#IN;
82 1 DELAY:PROCEDURE(MIL#SEC);
83 2 DEC MIl#SEC ADDRESS;
84 2 DO i = 0 TO MIl#SEC;
85 3 CALL TIME(7); /* 1 mSEC*/
86 3 END;
87 2 END DELAY;
88 1 PORT#A#OUT:PROCEDURE(CONTROLL,OUT#DATA); /* O/P DATA TO USAR */
89 2 DEC (CONTROLL,OUT#DATA) BYTE;
90 2 OUTPUT (PORT#A) = OUT#DATA;
91 2 CALL TAPE#STROBE#OUT(CONTROL);
END PORT#OUT;

TEMPT:PROCEDURE;
CALL TAPE#STROBE#IN(C6);
/* READ USRT STATUS */
DO WHILE (#DATA AND 01H)=00H;
/* WAIT FOR TRANSMIT BUFFER TO BE EMTY */
CALL TAPE#STROBE#IN(C6);
END;
END TEMPT;

TAPE#ERROR#MES: PROCEDURE;
/* O/P MESSAGE DEPENDING ON ERROR TYPE */
CALL INST#OUT(CLEAR#DISP);
IF (#DATA AND 80H)=80H THEN
DO:
CALL TEXT#OUT(#MES$39,LENGTH(#MES$39));
/* LOAD CASSETTE! */
CALL MENU(#MES$1,.BLANK,.BLANK);
/* SET */
RETURN;
END;
IF (#DATA AND 20H)=20H THEN
DO:
CALL TEXT#OUT(#MES$40,LENGTH(#MES$40));
/* END OF TAPE! */
RETURN;
END;
IF (#DATA AND 01H)=01H THEN
CALL TEXT#OUT(#MES$41,LENGTH(#MES$41));
/* WRITE PROTECTED! */
END TAPE#ERROR#MES;

REWIND: PROCEDURE;
CALL TAPE#STROBE#OUT(C4);
/* RESET EOT */
CALL TAPE#STROBE#IN(C5);
/* RESET OPTICAL */
CALL PORT#OUT(CE1, SEARCH#SPEED OR REV));
DO WHILE (#DATA AND 00H) = 00H;
/* TAPE LOADED,NO EOT? */
CALL TAPE#STROBE#IN(C5);
/* TRANSPORT STATUS */
END;
CALL PORT#OUT(CE1,00H);
/* STOP */
CALL DELAY(360);
/* WAIT 360mSEC */
END REWIND;

GET#OFF#LEADER: PROCEDURE;
CALL TAPE#STROBE#OUT(C4);
/* RESET EOT */
CALL TAPE#STROBE#IN(C5);
/* RESET OPTICAL */
CALL PORT#OUT(CE1, SEARCH#SPEED OR FWD OR WRITE#MODE))/
/* START TAPE TRANSPORT FORWARD */
DO WHILE (#DATA AND 00H) = 00H;
/* CLEAR LEADER PRESENT? */
CALL TAPE#STROBE#IN(C5);
/* TRANSPORT STATUS */
END;
CALL PORT#OUT(CE1,08H);
/* STOP */
CALL DELAY(360);
/* WAIT 360mSEC*/
CALL PORT#OUT(CE1,00H);
/* WRITE MODE OFF */
END GET#OFF#LEADER;

TAPE#BLOCK#DUMP: PROCEDURE PUBLIC;
IF RECORD=0 THEN
BUFF#SIZE=OFFH;
CALL INST#OUT(CLEAR#DISP);
CALL TEXT#OUT(#MES$45,LENGTH(#MES$45));
/* TAPE BLOCK WRITE */
CALL INST#OUT(0C2H);
CALL TEXT#OUT(#MES$35,LENGTH(#MES$35));
/* RECORD */
196 3 CALL TAPE$STROBE$IN(CE5); /* TRANSPORT STATUS */
197 3 END;
199 2 CALL DELAY(50); /* WAIT 50mS */
199 2 CALL PORT#A$OUT(CE1, 00H); /* STOP */
200 2 CALL DELAY(360); /* WAIT 360mSEC */
201 2 CALL PORT#A$OUT(CE1, 00H); /* WRITE MODE OFF */
202 2 RECORD=RECORD+1;
203 2 END TAPE$BLOCK$DUMP;
204 1 TAPE$WRITE$INIT: PROCEDURE PUBLIC;
205 2 CALL INST$OUT(CLEAR$DISP);
206 2 CALL TEXT$OUT(.MES#47, LENGTH(MES#47)); /* WRITE INITIALIZE */
207 2 CALL INST$OUT(HOME$2);
208 2 CALL TEXT$OUT(.MES#43, LENGTH(MES#43)); /* IN PROGRESS */
209 2 CALL REWIND;
210 2 CALL GET#$OFF#LEADER;
211 2 RECORD=0;
212 2 CALL TAPE$BLOCK$DUMP; /* INITIAL DUMMY DUMP */
213 2 END TAPE$WRITE$INIT;
214 1 TAPE$ERROR$:PROCEDURE PUBLIC;
215 2 CALL TAPE$ERROR$MES;
216 2 DO WHILE (TIDATA AND 80H)=0;
217 3 CALL TAPE$STROBE$IN(CE5); /* DO WHILE CASSETTE LOADED */
218 3 END;
219 2 CALL TAPE$ERROR$MES;
220 2 KEY=9;
221 2 DO WHILE (KEY>0) AND (TAPE$ERROR$FLAG=OFFH);
222 3 CALL TAPE$STROBE$IN(CE5); /* TRANSPORT STATUS */
223 3 DISABLE;
224 3 IF (TIDATA AND 80H)=0 THEN /* NEW CASSETTE LOADED? */
225 3 DO;
226 4 TAPE$ERROR$FLAG=0;
227 4 CALL TAPE$WRITE$INIT;
228 4 END;
229 3 ENABLE;
230 3 CALL DELAY(1);
231 3 END;
232 2 END TAPE$ERROR;
233 1 END TAPE$MODULE;

MODULE INFORMATION:

CODE AREA SIZE = 03EBH  1000D
VARIABLE AREA SIZE = 0000H  13D
MAXIMUM STACK SIZE = 0000H  8D
256 LINES READ
0 PROGRAM ERROR(S)

END OF PL/M-80 COMPILATION
APPENDIX D

DATA SHEETS

1. Keyboard encoder 74C922
2. Densitron Dot Matrix Display LMA IC24-DC
3. Real-Time Clock MC146818
4. 12-bit A/D Converter AD 574A
5. Sample and Hold AD582
6. 16 Channel Multiplexer AD7506
7. GPIB (IEEE-488) Circuit 96LS488
Typical Applications

Multiplex 7 Segment to Straight BCD

Memory Expansion from 7 Segment Outputs

General Description

These CMOS key encoders provide all the necessary logic to fully encode an array of SPST switches. The keyboard scan can be implemented by either an external clock or external capacitor. These encoders also have on-chip pull-up devices which permit switches with up to 50 kΩ on resistance to be used. No diodes in the switch array are needed to eliminate ghost switches. The internal debounce circuit needs only a single external capacitor and can be defeated by omitting the capacitor. A Data Available output goes to a high level when a valid keyboard entry has been made. The Data Available output returns to a low level when the entered key is released, even if another key is depressed. The Data Available will return high to indicate acceptance of the new key after a normal debounce period; this key roll over is provided between any two switches.

An internal register remembers the last key pressed even after the key is released. The TRI-STATE® outputs provide for easy expansion and bus operation and are LP5TL compatible.

Features

- 50 kΩ maximum switch on resistance
- On/off chip clock
- On chip row pull-up devices
- 2 key roll over
- Keybounce elimination with single capacitor
- Last key register at outputs
- TRI-STATE outputs LP5TL compatible
- Wide supply range 3V to 15V
- Low power consumption

Connection Diagrams
**Absolute Maximum Ratings (Note 1)**

- Voltage at Any Pin: $V_{CC} = 0.3V$ to $V_{CC} + 0.3V$
- Operating Temperature Range: $-55^\circ C$ to $125^\circ C$
- Package: 32-lead plastic dual in-line packaging
- Lead Temperature (Soldering 10 seconds): $300^\circ C$

**DC Electrical Characteristics**

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>PARAMETER</th>
<th>CONDITIONS</th>
<th>MIN</th>
<th>TYP</th>
<th>MAX</th>
<th>UNITS</th>
</tr>
</thead>
<tbody>
<tr>
<td>$V_{IL}$</td>
<td>Input Low Voltage</td>
<td>$V_{CC} - 5V$</td>
<td>-1.75</td>
<td>-1.35</td>
<td>mA</td>
<td></td>
</tr>
<tr>
<td>$V_{IL}$</td>
<td>Input Low Voltage</td>
<td>$V_{CC} = 15V$</td>
<td>26</td>
<td>60</td>
<td>ns</td>
<td></td>
</tr>
</tbody>
</table>

- **Input Capacitance**: Any Input, (Note 2) 5 to 7.5 pF
- **Output Capacitance**: Any Output, (Note 2) 10 pF

**Switching Time Waveforms**

- **T1, T2**: RC, $T3 = 0.7RC$ where $R = 10k$ and $C$ is external capacitor at K/BM input.

**Notes**:
- Absolute Maximum Ratings are those values beyond which the safety of the device cannot be guaranteed. Except for Operating Temperature Range, they are not meant to imply that the device should be operated at these limits. The table of Electrical Characteristics provides conditions for actual device operation.
- Capacitance is guaranteed by periodic testing.
**Block Diagram**

**Truth Table**

<table>
<thead>
<tr>
<th>INPUT</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTPUT</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

*Omit for MM54C922/MM74C922*

**Typical Performance Characteristics**

**Typical ON vs VOUT at Any Y Output**

**Typical ION vs VOUT at Any X Output**

**Typical Performance Characteristics (Cont'd.)**

**Typical Debounce Period vs TCHSM**

**Typical Applications**

**Synchronous Handshake (MM74C922)**

**Synchronous Data Entry On Bus (MM74C922)**

**Asynchronous Data Entry On Bus (MM74C922)**

Note 2: The keyboard may be synchronously scanned by omitting the meter at one sec., and driving one directly if the system clock rate is lower than 10 kHz.
MM54C932/MM74C932 Phase Comparator

General Description
The MM54C932/MM74C932 consists of two independent output phase comparator circuits. The two phase comparators have a common signal input and a common comparator input. The signal input can be directly coupled for a large voltage signal, or capacitively coupled to the self-biasing amplifier at the signal input for a small voltage signal.

Phase comparator I, an exclusive OR gate, provides a digital error signal (phase comp. I out) and maintains 90° phase shifts at the VCO center frequency. Between signal input and comparator input (both at 50% duty cycle), it may lock onto the signal input frequencies that are close to harmonics of the VCO center frequency.

Phase comparator II is an edge-controlled digital memory network. It provides a digital error signal (phase comp. II out) and lock in signal (phase pulse) to indicate a locked condition and maintains a 0° phase shift between signal input and comparator input.

Features
- Wide supply voltage range
- Convenient mini-DIP package
- TRI-STATE® phase comparator output (comparator II)
- 200 mV input voltage (signal to sensitivity (typical))

Block and Connection Diagrams
FEATURES:
- SIX MODELS — 16X1, 16X2, 24X1, 24X2, 40X1, 40X2 ARE AVAILABLE
- 5V, 2mA, SINGLE POWER SUPPLY
- EASY READABILITY, DOT MATRIX CHARACTERS WITH CURSOR
- SUNLIGHT AND NIGHT READABILITY WITH E-L BACKLIGHT
- WIDE, ADJUSTABLE VIEWING ANGLE
- EXTREMELY SMALL AND LIGHT WEIGHT
- TTL AND 5V CMOS COMPATIBLE
- INTERFACE DIRECTLY TO ANY 4 OR 8-BIT DATA BUS
- "POWERFUL" INSTRUCTIONS SAVE LINES OF CODING
- DISPLAY "BLANK," "WINK" OR "WINK LIMITED AREA"

- FULLY ASCII COMPATIBLE
- 192-CHARACTER GENERATOR ROM
  (96 Alphanumerics/Symbols, 64 Kata Kana, 32 Euro/Greek/Symbols)
- 8-USER PROGRAMMABLE RAM LOCATIONS FOR CUSTOM SYMBOLS
- 80-CHARACTER MEMORY ALLOWS EASY "SCROLLING" OR GENERAL PURPOSE STORAGE
- CURSOR "WINK," "REVERSE VIDEO," OR OFF
- AUTOMATIC DISPLAY SHIFT—SIMPLE TRANSMISSION OF "STRINGS"
- CAN SCROLL LEFT, RIGHT OR ALTERNATE WITH ENTIRE LINE REPLACEMENT
- 5X7 OR 5X10 FORMATS, DISPLAYS "TRUE" LOWER CASE DESCENDERS
APPLICATIONS
• TELECOMMUNICATIONS
• MEDICAL INSTRUMENTS
• HAND-HELD TERMINALS
• ELECTRONIC TYPEWRITERS
• POS TERMINALS
• TEST INSTRUMENTS
• LIGHT METERS
• CHEMICAL ANALYZERS
• WORD PROCESSORS
• NAVIGATION EQUIPMENT

TYPICAL MODULE BLOCK DIAGRAM

These intelligent, compact modules are complete display systems with minimum external requirements. Their cost effectiveness, versatility and simplicity now make information display easy.

The modules' low power is due to an exceptional, on-board, flat-pack CMOS microprocessor and driver. Operation is this simple: connect to any 4 or 8-bit data bus (or a single memory chip), switch one control line, and apply a single, non-critical “enable” pulse. Instantly the desired character appears on the LCD. Shifting is automatic.

The modules' compactness is impressive. The 16-character display requires just 3 3/8 x 1 x 1/2 inch. Mounting is simple and only 14 connections join the display. Electro-luminescent backlighting, integral to the display, is a low cost option. Holding frame is brushed stainless steel.

The display is virtually burden-free to the host processor. Internal registers store up to 80 characters and all display update and refresh is internal. Software is greatly eased by powerful, single step instructions which eliminate many lines of conventional graphics code.

Featured are a single, low power 5V supply, compatibility with TTL or 5V CMOS, and direct interface to 4 or 8-bit processors. A 192-character font provides 32 special codes, abbreviations and symbols. A further enhancement is an 8-character, fully user programmable, custom character capability. A uniquely customized, field-changeable, specialized graphic representation is now possible.

The simplicity and convenience of the module’s operation means that now, for the first time, only the designer's imagination will be challenged. Display applications are truly limitless.

Available upon request are a series of broad based application notes to speed your system implementation. Our staff of experienced engineers are eager to assist and provide comprehensive technical support. We welcome your call.

GENERAL DESCRIPTION:

ORDERING INFORMATION

<table>
<thead>
<tr>
<th>Characters per Line X Number of Lines</th>
<th>Module with Reflective Polarizer without EL Lamp</th>
<th>Module with Translfective Polarizer and EL Lamp</th>
<th>(EL Lamp Part Number : see page 15)</th>
<th>Applicable DC/AC Inverter : see page 16</th>
</tr>
</thead>
<tbody>
<tr>
<td>16 x 1</td>
<td>LMA1C16-DC</td>
<td>LMB1C16-DCL</td>
<td>(ELS 16)</td>
<td>DAS 5V</td>
</tr>
<tr>
<td>24 x 1</td>
<td>LMA1C24-DC</td>
<td>LMB1C24-DCL</td>
<td>(ELS 24)</td>
<td></td>
</tr>
<tr>
<td>40 x 1</td>
<td>LMA1C40-DC</td>
<td>LMB1C40-DCL</td>
<td>(ELS 40)</td>
<td></td>
</tr>
<tr>
<td>16 x 2</td>
<td>LMA2C16-DC</td>
<td>LMB2C16-DCL</td>
<td>(ELS 16)</td>
<td></td>
</tr>
<tr>
<td>24 x 2</td>
<td>LMA2C24-DC</td>
<td>LMB2C24-DCL</td>
<td>(ELS 24)</td>
<td></td>
</tr>
<tr>
<td>40 x 2</td>
<td>LMA2C40-DC</td>
<td>LMB2C40-DCL</td>
<td>(ELS 40)</td>
<td></td>
</tr>
</tbody>
</table>
MECHANICAL DIMENSIONS

PCB PADS FOR "E-L" LAMP LEADS AND CONNECTIONS

ALL DIMENSIONS IN MM
TOL ± 0.5MM ON 2 PLACE DIMENSIONS TO THE LEFT OF DECIMAL

TYPICAL MODULE MOUNTING

<table>
<thead>
<tr>
<th>PART NO.</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>J</th>
<th>K</th>
<th>L</th>
<th>M</th>
</tr>
</thead>
<tbody>
<tr>
<td>LMA(B)1C16-DC</td>
<td>85.0</td>
<td>80.0</td>
<td>2.5</td>
<td>73.5</td>
<td>59.4</td>
<td>63.5</td>
<td>2.05</td>
<td>13.25</td>
<td>36.0</td>
<td>31.0</td>
<td>2.5</td>
<td>28.5</td>
</tr>
<tr>
<td>LMA(B)1C24-DC</td>
<td>118.0</td>
<td>113.0</td>
<td>2.5</td>
<td>108.5</td>
<td>89.4</td>
<td>93.5</td>
<td>2.05</td>
<td>13.25</td>
<td>36.0</td>
<td>31.0</td>
<td>2.5</td>
<td>28.5</td>
</tr>
<tr>
<td>LMA(B)1C40-DC</td>
<td>182.0</td>
<td>175.0</td>
<td>3.5</td>
<td>163.5</td>
<td>149.4</td>
<td>154.5</td>
<td>2.85</td>
<td>16.30</td>
<td>33.5</td>
<td>26.5</td>
<td>3.5</td>
<td>32.5</td>
</tr>
<tr>
<td>LMA(B)2C16-DC</td>
<td>85.0</td>
<td>80.0</td>
<td>2.5</td>
<td>73.5</td>
<td>59.4</td>
<td>63.5</td>
<td>2.05</td>
<td>13.25</td>
<td>36.0</td>
<td>31.0</td>
<td>2.5</td>
<td>28.5</td>
</tr>
<tr>
<td>LMA(B)2C24-DC</td>
<td>118.0</td>
<td>113.0</td>
<td>2.5</td>
<td>108.5</td>
<td>89.4</td>
<td>93.5</td>
<td>2.05</td>
<td>13.25</td>
<td>36.0</td>
<td>31.0</td>
<td>2.5</td>
<td>28.5</td>
</tr>
<tr>
<td>LMA(B)2C40-DC</td>
<td>182.0</td>
<td>175.0</td>
<td>3.5</td>
<td>163.5</td>
<td>149.4</td>
<td>154.5</td>
<td>2.85</td>
<td>16.30</td>
<td>33.5</td>
<td>26.5</td>
<td>3.5</td>
<td>32.5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PART NO.</th>
<th>N</th>
<th>P</th>
<th>Q</th>
<th>R</th>
<th>S</th>
<th>T</th>
<th>U</th>
<th>V</th>
<th>W</th>
<th>X</th>
<th>Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>LMA(B)1C16-DC</td>
<td>7.88</td>
<td>8.7</td>
<td>13.65</td>
<td>15.8</td>
<td>10.1</td>
<td>9.3</td>
<td>1.4</td>
<td>4.7</td>
<td>2.5</td>
<td>1.6</td>
<td>20.0</td>
</tr>
<tr>
<td>LMA(B)1C24-DC</td>
<td>7.88</td>
<td>8.7</td>
<td>13.65</td>
<td>15.8</td>
<td>10.1</td>
<td>9.3</td>
<td>1.4</td>
<td>4.7</td>
<td>2.5</td>
<td>1.6</td>
<td>20.0</td>
</tr>
<tr>
<td>LMA(B)1C40-DC</td>
<td>7.00</td>
<td>8.7</td>
<td>13.25</td>
<td>15.8</td>
<td>8.5</td>
<td>9.3</td>
<td>1.4</td>
<td>4.7</td>
<td>3.5</td>
<td>1.6</td>
<td>22.5</td>
</tr>
<tr>
<td>LMA(B)2C16-DC</td>
<td>7.88</td>
<td>12.3</td>
<td>11.85</td>
<td>15.8</td>
<td>10.1</td>
<td>9.3</td>
<td>1.4</td>
<td>4.7</td>
<td>2.5</td>
<td>1.6</td>
<td>20.0</td>
</tr>
<tr>
<td>LMA(B)2C24-DC</td>
<td>7.88</td>
<td>12.3</td>
<td>11.85</td>
<td>15.8</td>
<td>10.1</td>
<td>9.3</td>
<td>1.4</td>
<td>4.7</td>
<td>2.5</td>
<td>1.6</td>
<td>20.0</td>
</tr>
<tr>
<td>LMA(B)2C40-DC</td>
<td>7.00</td>
<td>12.3</td>
<td>10.60</td>
<td>15.8</td>
<td>8.5</td>
<td>9.3</td>
<td>1.4</td>
<td>4.7</td>
<td>3.5</td>
<td>1.6</td>
<td>22.5</td>
</tr>
</tbody>
</table>
### ELECTRO-OPTICAL CHARACTERISTICS \( V_{DD} = 5.0 \pm 0.25V, \ T_a = 25^\circ C \)

<table>
<thead>
<tr>
<th>ITEM</th>
<th>SYMBOL</th>
<th>CONDITION</th>
<th>MIN.</th>
<th>TYP.</th>
<th>MAX.</th>
<th>UNIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input &quot;High&quot; Voltage</td>
<td>( V_{IH} )</td>
<td>( V_{CC} )</td>
<td>2.2</td>
<td>—</td>
<td>—</td>
<td>V</td>
</tr>
<tr>
<td>Input &quot;Low&quot; Voltage</td>
<td>( V_{IL} )</td>
<td>—</td>
<td>—</td>
<td>0.6</td>
<td>—</td>
<td>V</td>
</tr>
<tr>
<td>Output &quot;High&quot; Voltage</td>
<td>( V_{OH} )</td>
<td>( I_{OH} = 0.205mA )</td>
<td>2.4</td>
<td>—</td>
<td>—</td>
<td>V</td>
</tr>
<tr>
<td>Output &quot;Low&quot; Voltage</td>
<td>( V_{OL} )</td>
<td>( I_{OL} = 1.6mA )</td>
<td>—</td>
<td>0.4</td>
<td>—</td>
<td>V</td>
</tr>
<tr>
<td>Power Supply Current</td>
<td>( I_{DD} )</td>
<td>( V_{DD} = 5.0V )</td>
<td>0.5</td>
<td>2.0</td>
<td>—</td>
<td>mA</td>
</tr>
<tr>
<td>1 Line Module Drive Voltage</td>
<td>( V_{DD} \cdot V_{EE} )</td>
<td>( T_a = 0^\circ C )</td>
<td>4.2</td>
<td>4.3</td>
<td>4.4</td>
<td>V</td>
</tr>
<tr>
<td>5x7 Font + Cursor</td>
<td>( V_{DD} \cdot V_{EE} )</td>
<td>( T_a = 25^\circ C )</td>
<td>3.8</td>
<td>3.9</td>
<td>4.0</td>
<td>V</td>
</tr>
<tr>
<td>1 Line Module Drive Voltage</td>
<td>( V_{DD} \cdot V_{EE} )</td>
<td>( T_a = 50^\circ C )</td>
<td>3.4</td>
<td>3.5</td>
<td>3.6</td>
<td>V</td>
</tr>
<tr>
<td>1 Line Module Drive Voltage</td>
<td>( V_{DD} \cdot V_{EE} )</td>
<td>( T_a = 0^\circ C )</td>
<td>4.5</td>
<td>4.6</td>
<td>4.7</td>
<td>V</td>
</tr>
<tr>
<td>5x10 Font + Cursor</td>
<td>( V_{DD} \cdot V_{EE} )</td>
<td>( T_a = 25^\circ C )</td>
<td>4.1</td>
<td>4.2</td>
<td>4.3</td>
<td>V</td>
</tr>
<tr>
<td>1 Line Module Drive Voltage</td>
<td>( V_{DD} \cdot V_{EE} )</td>
<td>( T_a = 50^\circ C )</td>
<td>3.6</td>
<td>3.7</td>
<td>3.8</td>
<td>V</td>
</tr>
<tr>
<td>2 Line Module Drive Voltage</td>
<td>( V_{DD} \cdot V_{EE} )</td>
<td>( T_a = 0^\circ C )</td>
<td>4.8</td>
<td>4.9</td>
<td>5.0</td>
<td>V</td>
</tr>
<tr>
<td>1 Line Module Drive Voltage</td>
<td>( V_{DD} \cdot V_{EE} )</td>
<td>( T_a = 25^\circ C )</td>
<td>4.3</td>
<td>4.4</td>
<td>4.5</td>
<td>V</td>
</tr>
<tr>
<td>2 Line Module Drive Voltage</td>
<td>( V_{DD} \cdot V_{EE} )</td>
<td>( T_a = 50^\circ C )</td>
<td>3.8</td>
<td>3.9</td>
<td>4.0</td>
<td>V</td>
</tr>
<tr>
<td>Viewing Angle ( \alpha_1 - \alpha_2 )</td>
<td>( K = \frac{\alpha_1}{\alpha_2} )</td>
<td>( K = 20^\circ )</td>
<td>20</td>
<td>—</td>
<td>—</td>
<td>deg.</td>
</tr>
<tr>
<td>Contrast Ratio ( K )</td>
<td>( K = \frac{\alpha_1}{\alpha_2} )</td>
<td>( e = 0^\circ )</td>
<td>3</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>Rise Time ( t_r )</td>
<td>( \alpha = 20^\circ )</td>
<td>—</td>
<td>150</td>
<td>250</td>
<td>ms</td>
<td></td>
</tr>
<tr>
<td>Fall Time ( t_f )</td>
<td>( \alpha = 20^\circ )</td>
<td>—</td>
<td>150</td>
<td>250</td>
<td>ms</td>
<td></td>
</tr>
</tbody>
</table>

### POWER SUPPLY

![Diagram](image)

**PIN FUNCTION**

<table>
<thead>
<tr>
<th>PIN NAME</th>
<th>I/O</th>
<th>FUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>( V_{SS} )</td>
<td>I/O</td>
<td>Ground; 0V</td>
</tr>
<tr>
<td>( V_{DD} )</td>
<td>+5V</td>
<td>Power supply for LC driving</td>
</tr>
<tr>
<td>( V_{EE} )</td>
<td></td>
<td>Signal to select registers</td>
</tr>
<tr>
<td></td>
<td></td>
<td>“0”: Instruction register (for write)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Busy flag: address counter (for read)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>“1”: Data register (for read and write)</td>
</tr>
<tr>
<td>RS</td>
<td>I</td>
<td>Signal to select read (R) and write (W)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>“0”: write MPU ( \rightarrow ) LCD Module</td>
</tr>
<tr>
<td></td>
<td></td>
<td>“1”: read MPU ( \leftarrow ) LCD Module</td>
</tr>
<tr>
<td>R/W</td>
<td>I</td>
<td>Operation start signal for data read or write</td>
</tr>
<tr>
<td>E</td>
<td>I</td>
<td>Data bus of lower order 4 lines having bidirectional tri-state.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Used for data transfer between the MPU and the module.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>These four are not used during 4-bit operation</td>
</tr>
<tr>
<td>( DB0 )</td>
<td>I/O</td>
<td>Data bus of higher order 4 lines having bidirectional tri-state.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Used for data transfer between the MPU and the module.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>DB7 can be used as a BUSY flag.</td>
</tr>
</tbody>
</table>

**NOTE:** In the module, the data can be sent in either 4-bit 2 sequence operation or 8-bit single-operation so that it can interface to both 4 and 8 bit MPU’s.

1. When interface data is 4 bits long, data is transferred using only lines \( DB4 - DB7 \) and \( DB0 - DB3 \) are not used.
   Data transfer between the module and the MPU is complete when the 4-bit data is transferred twice. Data of the higher order 4 bits (the contents of \( DB4 - DB7 \) when the interface data is 8 bits long) is transferred first, and the lower order 4 bits (the contents of \( DB0 - DB3 \) when the interface data is 8 bits long) follows.

2. When the interface data is 8 bits long, data is transferred using all 8 data lines of \( DB0 - DB7 \).
ABSOLUTE MAXIMUM RATINGS

<table>
<thead>
<tr>
<th>ITEM</th>
<th>SYMBOL</th>
<th>MIN.</th>
<th>MAX.</th>
<th>UNIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logic circuit power supply voltage</td>
<td>$V_{DD} - V_{SS}$</td>
<td>0</td>
<td>7.0</td>
<td>V</td>
</tr>
<tr>
<td>LC driver circuit supply voltage</td>
<td>$V_{DD} - V_{EE}$</td>
<td>0</td>
<td>13.5</td>
<td>V</td>
</tr>
<tr>
<td>Input voltage</td>
<td>$V_I$</td>
<td>$V_{SS}$</td>
<td>$V_{DD}$</td>
<td></td>
</tr>
<tr>
<td>Operating temperature</td>
<td>$t_a$</td>
<td>0</td>
<td>50</td>
<td>°C</td>
</tr>
<tr>
<td>Storage temperature</td>
<td>$t_{stg}$</td>
<td>$-20$</td>
<td>70</td>
<td>°C</td>
</tr>
</tbody>
</table>

PIN CONNECTION

<table>
<thead>
<tr>
<th>PIN NO.</th>
<th>SIGNAL</th>
<th>LEVEL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$V_{SS}$</td>
<td>GROUND</td>
</tr>
<tr>
<td>2</td>
<td>$V_{DD}$</td>
<td>+5.0V</td>
</tr>
<tr>
<td>3</td>
<td>$V_{EE}$</td>
<td>$\approx +0.7V$</td>
</tr>
<tr>
<td>4</td>
<td>RS</td>
<td>L = INST, H = CHAR</td>
</tr>
<tr>
<td>5</td>
<td>R/W</td>
<td>L = W, H = R</td>
</tr>
<tr>
<td>6</td>
<td>E</td>
<td>LATCH ON FALL</td>
</tr>
<tr>
<td>7</td>
<td>DB0</td>
<td>POSITIVE</td>
</tr>
<tr>
<td>8</td>
<td>DB1</td>
<td>LOGIC</td>
</tr>
<tr>
<td>9</td>
<td>DB2</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>DB3</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>DB4</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>DB5</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>DB6</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>DB7</td>
<td></td>
</tr>
</tbody>
</table>

Display Pattern

Definition of optical response

Definition of contrast ‘K’

Definition of angle $\phi$ and $\theta$
<table>
<thead>
<tr>
<th>UPPER 4 BIT HEXADECIMAL</th>
<th>0</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>00P</td>
<td>:p</td>
<td>-</td>
<td>3</td>
<td>e</td>
<td>a</td>
<td>p</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0010</td>
<td>11A</td>
<td>w</td>
<td>q</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0011</td>
<td>2B</td>
<td>R</td>
<td>r</td>
<td>e</td>
<td>p</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0100</td>
<td>3C</td>
<td>S</td>
<td>s</td>
<td>e</td>
<td>t</td>
<td>e</td>
<td>s</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0101</td>
<td>4D</td>
<td>T</td>
<td>d</td>
<td>t</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0110</td>
<td>5E</td>
<td>u</td>
<td>e</td>
<td>u</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0111</td>
<td>6F</td>
<td>V</td>
<td>v</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1000</td>
<td>7G</td>
<td>W</td>
<td>w</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1001</td>
<td>8H</td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1010</td>
<td>9I</td>
<td>Y</td>
<td>y</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1011</td>
<td>:J</td>
<td>z</td>
<td>z</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1100</td>
<td>:K</td>
<td>K</td>
<td>k</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1101</td>
<td>:L</td>
<td>I</td>
<td>I</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1110</td>
<td>:M</td>
<td>i</td>
<td>m</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1111</td>
<td>:N</td>
<td>o</td>
<td>o</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FFFF</td>
<td>:O</td>
<td>o</td>
<td>o</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### TIMING CHARACTERISTICS

#### WRITE OPERATION

<table>
<thead>
<tr>
<th>ITEM</th>
<th>SYMBOL</th>
<th>MIN.</th>
<th>TYP.</th>
<th>MAX.</th>
<th>UNIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Cycle Time</td>
<td>$t_{EycE}$</td>
<td>1.0</td>
<td></td>
<td></td>
<td>$\mu s$</td>
</tr>
<tr>
<td>Enable Pulse Width</td>
<td>$PW_E$</td>
<td>450</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Enable Rise/Fail Time</td>
<td>$t_{ER}, t_{EF}$</td>
<td></td>
<td></td>
<td>25</td>
<td>ns</td>
</tr>
<tr>
<td>Address Set-up Time</td>
<td>$t_{AS}$</td>
<td>140</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Address Hold Time</td>
<td>$t_{AH}$</td>
<td></td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Data Start-up Time</td>
<td>$t_{DBSW}$</td>
<td>195</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Data Hold Time</td>
<td>$t_{DH}$</td>
<td>10</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
</tbody>
</table>

#### READ OPERATION

<table>
<thead>
<tr>
<th>ITEM</th>
<th>SYMBOL</th>
<th>MIN.</th>
<th>TYP.</th>
<th>MAX.</th>
<th>UNIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Cycle Time</td>
<td>$t_{EycE}$</td>
<td>1.0</td>
<td></td>
<td></td>
<td>$\mu s$</td>
</tr>
<tr>
<td>Enable Pulse Width</td>
<td>$PW_E$</td>
<td>450</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Enable Rise/Fail Time</td>
<td>$t_{ER}, t_{EF}$</td>
<td></td>
<td></td>
<td>25</td>
<td>ns</td>
</tr>
<tr>
<td>Address Set-up Time</td>
<td>$t_{AS}$</td>
<td>140</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Address Hold Time</td>
<td>$t_{AH}$</td>
<td></td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Data Delay Time</td>
<td>$t_{DD}$</td>
<td></td>
<td></td>
<td>320</td>
<td>ns</td>
</tr>
<tr>
<td>Data Hold Time</td>
<td>$t_{DH}$</td>
<td>20</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
</tbody>
</table>

#### RESET FUNCTION

The module automatically performs initialization (reset) when power is turned on (using internal reset circuit). The following instructions are executed in initialization:

1. **Clear display**
   - The busy flag is kept in the busy state ($BF = 1$) until initialization ends. The time is 15ms.

2. **Function set**
   - $DL = 1$: 8 bits wide interface data
   - $N = 0$: 1-line display
   - $F = 0$: 5 x 7 dot character font

3. **Display ON/OFF control**
   - $D = 0$: Display OFF
   - $C = 0$: Cursor OFF
   - $B = 0$: Blink OFF

4. **Entry mode set**
   - $I/O = 1$: +1 (increment)
   - $S = 0$: No shift

5. **DD RAM is selected**

Because initialization may not be performed completely depending on the rise time of the power supply when it is turned on, pay attention to the following time relationship.

$t_{OFF}$ stipulates the time of power OFF for power supply instantaneous dip or when power supply repeats ON and OFF.

---

**NOTE:** When the above power supply condition is not satisfied, the internal reset circuit does not operate normally. In this case, perform the needed initialization by sending instructions from the MPU after turning the power ON. When initialization by the internal reset circuit is not performed normally, whether the interface data is 4 bits long or 8 bits long is not clear. Therefore designate 8-bit data length by sending the function set instruction twice and then perform the required initialization.

**RS R/W DB7 DB6 DB5 DB4 DB3 DB2 DB1 DB0**

0 0 0 0 0 1 1 · · · ·
0 0 0 0 1 1 · · · ·

When this instruction is sent, the module enters the 8-bit data length mode without fail.
### INSTRUCTIONS

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Code</th>
<th>Description</th>
<th>Execution Time (when ( f_{op} ) or ( f_{osc} ) is 250KHz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clear Display</td>
<td>0 0 0 0 0 0 0 0 0 1</td>
<td>Clears all display and returns the cursor to the home position (Address 0)</td>
<td>82 ( \mu )s \sim 1.64ms</td>
</tr>
<tr>
<td>Return Home</td>
<td>0 0 0 0 0 0 0 0 1</td>
<td>Returns the cursor to the home position (Address 0). Also returns the display being shifted to the original position. DD RAM contents remain unchanged.</td>
<td>40 ( \mu )s \sim 1.6ms</td>
</tr>
<tr>
<td>Entry Mode Set</td>
<td>0 0 0 0 0 0 0 0 0 1</td>
<td>Sets the cursor move direction and specifies or not to shift the display. These operations are performed during data write and read.</td>
<td>40 ( \mu )s</td>
</tr>
<tr>
<td>Display ON/OFF Control</td>
<td>0 0 0 0 0 0 0 1 D C B</td>
<td>Sets ON/OFF of all display (D), Cursor ON/OFF (C), and blink of cursor position character (B).</td>
<td>40 ( \mu )s</td>
</tr>
<tr>
<td>Cursor or Display Shift</td>
<td>0 0 0 0 0 0 0 0 1</td>
<td>Moves the cursor and shifts the display without changing DD RAM contents.</td>
<td>40 ( \mu )s</td>
</tr>
<tr>
<td>Function Set</td>
<td>0 0 0 0 0 0 0 0 1</td>
<td>Sets interface data length (DL), number of display lines (L), and character font (F).</td>
<td>40 ( \mu )s</td>
</tr>
<tr>
<td>Set CG RAM Address</td>
<td>0 0 0 0 1</td>
<td>Sets the CG RAM address. CG RAM data is sent and received after this setting.</td>
<td>40 ( \mu )s</td>
</tr>
<tr>
<td>Set DD RAM Address</td>
<td>0 0 1</td>
<td>Sets the DD RAM address. DD RAM data is sent and received after this setting.</td>
<td>40 ( \mu )s</td>
</tr>
<tr>
<td>Read Busy Flag &amp; Address</td>
<td>0 1 BF</td>
<td>Reads Busy flag (BF) invoking internal operation is being performed and reads address counter contents.</td>
<td>40 ( \mu )s</td>
</tr>
<tr>
<td>Write Data to CG or DD RAM</td>
<td>1 0</td>
<td>Writes data into DD RAM or CG RAM.</td>
<td>40 ( \mu )s</td>
</tr>
<tr>
<td>Read Data to CG or DD RAM</td>
<td>1 1</td>
<td>Reads data from DD RAM or CG RAM.</td>
<td>40 ( \mu )s</td>
</tr>
</tbody>
</table>

### MODULE BLOCK DIAGRAMS

**16X1 AND 16X2**

**24X1 AND 24X2**

**40X1 AND 40X2**

**N = 16 FOR TWO LINE DISPLAYS**

**N = 11 FOR SINGLE LINE DISPLAY**

*Don't care*
INSTRUCTION DESCRIPTION

**OUTLINE**
Two registers of the HD44780, the Instruction Register (IR) and the Data Register (DR) only can be controlled by MPU directly. Control information is temporarily stored in these registers, prior to internal operation start, to allow interface to various types of MPUs which operate in different speeds from HD44780 internal operation or to allow interface to peripheral control ICs. The HD44780 internal operation is determined by signals sent from the MPU, these signals including register selection signals (RS), read/write signals (R/W) and data bus signals (DBa - DB), are called instructions in this paragraph. Table on page 10 shows the instructions and the execution time of the instructions. Details are explained in the following sections. The instructions can be divided into the following 4 types:

1. Instructions that designate the HD44780 functions such as display format, data length, etc.
2. Instructions that give internal RAM addresses.
3. Instructions that perform data transfer with internal RAM.
4. Other instructions

In the normal use, instructions of category (3), which sends display data, is used most frequently. However, since the HD44780 internal RAM addresses are configured to be automatically incremented or decremented by +1 after each data write, MPU program load is lessened. Especially, display shift is performed concurrently with display data write, and this enables the user to develop systems with minimum time and maximum efficiency of programming. When an instruction is being executed (during internal operation), the busy flag DB is active high. This must be monitored when high speed operation is planned (≈50 KHz).

**CLEAR DISPLAY**

<table>
<thead>
<tr>
<th>Code</th>
<th>RS</th>
<th>R/W</th>
<th>DB0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Writes space code "20" (hexadecimal) into all the DD RAM addresses. The cursor returns to Address 0 (AD = "80") and display, if it has been shifted, returns to the original position. In other words, display disappears and the cursor goes to the left edge of the display (the first line if 2 lines are displayed).

**RETURN HOME**

<table>
<thead>
<tr>
<th>Code</th>
<th>RS</th>
<th>R/W</th>
<th>DB0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 0 0 0 0 0</td>
<td>0 0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Returns the cursor to Address 0 (AD = "80") and display, if it has been shifted, to the original position. The DD RAM contents remain unchanged.

**ENTRY MODE SET**

<table>
<thead>
<tr>
<th>Code</th>
<th>RS</th>
<th>R/W</th>
<th>DB0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 0 0 0 1</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

I/D: Increments (I/D = 1) or decrements (I/D = 0) the DD RAM address by one upon writing into or reading from the DD RAM a character code. The cursor moves to the right when incremented by one. The same applies to writing and reading of CG RAM.

S: Shifts the entire display to either the right or the left when S is 1; to the left when I/D = 1 and to the right when I/D = 0. Therefore, the cursor looks as if it stood still with the display only moved. Display is not shifted when reading from the DD RAM. Display is not shifted when S = 0.

**DISPLAY ON/OFF CONTROL**

<table>
<thead>
<tr>
<th>Code</th>
<th>RS</th>
<th>R/W</th>
<th>DB0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 0 0 1 0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

D: Display is turned ON when D = 1 and OFF when D = 0. When display is turned off due to D = 0, the display data remains in the DD RAM and it can be displayed immediately by setting D = 1.

C: The cursor is displayed when C = 1 and not displayed when C = 0. Even if the cursor disappears, function of I/D, etc. does not change during display data write. The cursor is displayed using 5 dots in the 8th lines when the 5x 7 dot character font is selected and in the 11th line when 5x 10 dot character font is selected.

B: The character residing at the cursor position blinks when B = 1. The blink is done by switching between all the black dots and display characters at 0.4 second interval. The cursor and the blink can be set concurrently.

**CURSOR OR DISPLAY SHIFT**

<table>
<thead>
<tr>
<th>Code</th>
<th>RS</th>
<th>R/W</th>
<th>DB0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 0 0 1</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Shifts the cursor position or display to the right and the left without writing or reading the display data. This function is used for correction or search of display.

S/C R/L

| Shifts the cursor position to the left. (AC is decremented by one.) |
| 0 0 |

| Shifts the cursor position to the right. (AC is incremented by one.) |
| 0 1 |

| Shifts the entire display to the left. The cursor follows the display shift. |
| 1 0 |

| Shifts the entire display to the right. The cursor follows the display shift. |
| 1 1 |
• FUNCTION SET

<table>
<thead>
<tr>
<th>Code</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>1</th>
<th>DL</th>
<th>N</th>
<th>F</th>
<th>0</th>
</tr>
</thead>
</table>

*Don't Care*

DL: Sets interface data length. Data is sent or received in 8 bit length (DB7 ~ DB0) when DL = 1 and 4 bit length (DB3 ~ DB0) when DL = 0. When 4 bit length is selected, data must be sent or received twice.

N: Sets number of display lines.
F: Sets character font.

<table>
<thead>
<tr>
<th>No. of Display Lines</th>
<th>Character Font</th>
<th>Duty Factor</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>5x7 dots</td>
<td>1/8</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>5x10 dots</td>
<td>1/11</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>5x7 dots</td>
<td>1/16</td>
<td></td>
</tr>
</tbody>
</table>

*Don't Care*

• SET CG RAM ADDRESS

<table>
<thead>
<tr>
<th>Code</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>1</th>
<th>AA</th>
<th>AA</th>
<th>AA</th>
</tr>
</thead>
</table>

Sets the CG RAM address in a binary number of AAAAAA to the address counter, and data is written or read from the MPU related to the CG RAM after this.

• SET DD RAM ADDRESS

<table>
<thead>
<tr>
<th>Code</th>
<th>0</th>
<th>0</th>
<th>1</th>
<th>AA</th>
<th>AA</th>
<th>AA</th>
</tr>
</thead>
</table>

Sets the DD RAM address in a binary number of AAAAAAAA to the address counter, and data is written or read from the MPU related to the DD RAM after this. However, when N = 0 (1-line display), AAAAAAA is "00" ~ "4F" (hexadecimal). When N = 1 (2-line display), AAAAAAAA is "00" ~ "27" (hexadecimal) for the first line, and "40" ~ "67" (hexadecimal) for the second line.

• WRITE DATA TO CG OR DD RAM

<table>
<thead>
<tr>
<th>Code</th>
<th>1</th>
<th>0</th>
<th>DD</th>
<th>DD</th>
<th>DD</th>
<th>DD</th>
<th>DD</th>
</tr>
</thead>
</table>

Writes binary 8 bit data DDDDDDDD to the CG or the DD RAM. Whether the CG or the DD RAM is to be written is determined by the previous designation (CG RAM address setting or DD RAM address setting). After write, the address is automatically incremented or decremented by one according to entry mode. Display shift also follows the entry mode.

• READ DATA FROM CG OR DD RAM

<table>
<thead>
<tr>
<th>Code</th>
<th>1</th>
<th>1</th>
<th>DD</th>
<th>DD</th>
<th>DD</th>
<th>DD</th>
<th>DD</th>
</tr>
</thead>
</table>

Reads binary 8 bit data DDDDDDDD from the CG or the DD RAM. Whether the CG RAM or the DD RAM is to be read is determined by the previous designation. Prior to inputting this read instruction, either the CG RAM address set instruction or the DD RAM address set instruction must be executed. If it is not done, the first read data becomes invalid, and data of the next address is read normally from the second read. After read, the address is automatically incremented or decremented by one according to the entry mode. However, display shift is not performed regardless of entry mode types.

• READ BUSY FLAG AND ADDRESS

<table>
<thead>
<tr>
<th>Code</th>
<th>0</th>
<th>1</th>
<th>BF</th>
<th>AA</th>
<th>AA</th>
<th>AA</th>
<th>AA</th>
</tr>
</thead>
</table>

Reads Busy Flag (BF) that indicates the system is internally operating on an instruction received before. When BF = 1, it indicates that internal operation is going on and the next instruction is not accepted until BF is set to "0." Check the BF status before the next write operation. At the same time, this value of the address counter expressed in a binary number of AAAAAA. The address counter is used by both of CG and DD RAM address, and its value is determined by the previous instruction. Address contents are those of CG RAM and DD RAM previously shown.
### RELATION BETWEEN CG RAM ADDRESSES AND CHARACTER CODES (DD RAM) AND CHARACTER PATTERNS (CG RAM DATA)

#### FOR 5 × 7 DOT CHARACTER PATTERNS

<table>
<thead>
<tr>
<th>Character Codes (DD RAM Data)</th>
<th>CG RAM Address</th>
<th>Character Patterns (CG RAM Data)</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 6 5 4 3 2 1 0</td>
<td>5 4 3 2 1 0</td>
<td></td>
</tr>
<tr>
<td>Higher Order Bits</td>
<td>Higher Order Bits</td>
<td></td>
</tr>
<tr>
<td>Lower Order Bits</td>
<td>Lower Order Bits</td>
<td></td>
</tr>
<tr>
<td>0 0 0 0 0 • 0 0 0</td>
<td>0 0 0 0</td>
<td></td>
</tr>
<tr>
<td>Character Pattern Example (1)</td>
<td>Cursor Position</td>
<td></td>
</tr>
<tr>
<td>0 0 1 0 1 1</td>
<td>0 0 1 0 1</td>
<td></td>
</tr>
<tr>
<td>Character Pattern Example (2)</td>
<td>Cursor Position</td>
<td></td>
</tr>
<tr>
<td>0 0 0 0 0 • 0 0 1</td>
<td>0 0 0 0 0</td>
<td></td>
</tr>
</tbody>
</table>

Note 1: Character code bits 0 – 2 correspond to CG RAM address bits 3 – 5 (3 bits: 8 types).
Note 2: CG RAM address bits 0 – 2 designate character pattern line position. The 8th line is the cursor position and display is performed in logical OR with cursor.
Note 3: Character pattern row positions correspond to CG RAM data bits 0 – 4, as shown in the figure (Bit 4 being at the left end). Since CG RAM data bits 5 – 7 are not used for display, they can be used as general data RAM.
Note 4: CG RAM character patterns are selected when character code bits 4 – 7 are all “0.” However, since character code bit 3 is a don’t care bit, “R” display in the character pattern example, for example, are selected by character code “00” (hexadecimal) or “08” (hexadecimal).
Note 5: “1” for CG RAM data corresponds to selection for display and “0” for non-selection.

#### FOR 5 × 10 DOT CHARACTER PATTERNS

<table>
<thead>
<tr>
<th>Character Codes (DD RAM Data)</th>
<th>CG RAM Address</th>
<th>Character Patterns (CG RAM Data)</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 6 5 4 3 2 1 0</td>
<td>5 4 3 2 1 0</td>
<td></td>
</tr>
<tr>
<td>Higher Order Bits</td>
<td>Higher Order Bits</td>
<td></td>
</tr>
<tr>
<td>Lower Order Bits</td>
<td>Lower Order Bits</td>
<td></td>
</tr>
<tr>
<td>0 0 0 0 0 • 0 0 0</td>
<td>0 0 0 0</td>
<td></td>
</tr>
<tr>
<td>Character Pattern Example</td>
<td>Cursor Position</td>
<td></td>
</tr>
<tr>
<td>0 0 1 0 1 1</td>
<td>0 0 1 0 1</td>
<td></td>
</tr>
<tr>
<td>Character Pattern Example</td>
<td>Cursor Position</td>
<td></td>
</tr>
<tr>
<td>0 0 0 0 0 • 0 0 1</td>
<td>0 0 0 0 0</td>
<td></td>
</tr>
</tbody>
</table>

Note 1: Character code bits 1, 2 correspond to CG RAM address bits 4, 5 (2 bits: 4 types).
Note 2: CG RAM address bits 0 – 3 designate character pattern line position. The 11th line is the cursor position and display is performed in logical OR with cursor.
Note 3: Character pattern row positions are the same as the positions for 5 × 7 dot character patterns. Since the 12th – 16th lines are not used for display, they can be used as general data RAM.
Note 4: CG RAM character patterns are selected when character code bits 4 – 7 are all “0.” However, since character code bit 3 and 4 are don’t care bits, “P” display in the character pattern example, for example, are selected by character code “00,” “01,” “08” and “09” (hexadecimal).
Note 5: “1” for CG RAM data corresponds to selection for display and “0” for non-selection.
INTERFACING WITH MICROPROCESSORS

Write instruction execution (RS=0)
- Module executes code instruction.

Write data execution (RS=0)
- Write DDRAM or CGRAM data into Module.

Read instruction execution (RS=1)
- Read out DDRAM or CGRAM data from Module.

Note: In this flowchart, Read and Write instructions correspond with external RAM reference instruction or I/O instruction.

INTERFACE WITH HMCS 40

INTERFACE WITH 8080A

INTERFACE WITH 68B00/PIA

INTERFACE WITH 6800

TIMING CHART

- LCD module is treated as I/O or RAM device.
- As Ao is connected to RS, Therefore Ao's level controls the instruction register or the data register.
- The address of the module as I/O device in the address area of the 6800 are:
  - Write function: #"FXX0" (R/W = 0)
  - Write CGRAM/DDRAM data: #"FXX1" (R/W = 0)
  - Read Busy flag/address: #"FXX0" (R/W = 1)
  - Read CGRAM/DDRAM data: #"FXX1" (R/W = 1)

Note: The lead bit "F" may be 'C', 'D', or 'E'

- 68B00
  - The 68B00 is a high speed version of the 6800. The 02 signal is too narrow to properly drive the LCD module. A simple interface is the insertion of the 6821 PIA between the CPU and module. This widens the pulse sufficiently.
INTERFACE WITH 8048 (8049)

- The module is treated as external RAM.
- As the 8048 doesn't have an equivalent signal to R/W, this signal is made from RD and WR.
- 'NAND' of RD and WR is regarded as 'E' signal. Set-up-time between the leading edge of R/W and the leading edge of 'E' signal should be min. 140nsec, where the leading edge of R/W should precede that of 'E' signal. To keep this set-up-time (min. 140nsec.), the value of 'C' and 'R' should be determined.
- In addressing of the module as external RAM in the address area of 8048, the address on data bus (DB0~DB7) are latched by ALE signal at the execution of external RAM reference instruction (MOVXA, @ Rr: MOVX @ Rr, A). In this case, Ao corresponds to RS of the module, and 'AND' of A4~A7 and 'OR' signal of RD, WR has chip select function.

Write function
Write CGRAM/DDR data  # 'F0' (R/W = 0)
Read busy flag/address  # 'F1' (R/W = 0)
Read CGRAM/DDR data   # 'F0' (R/W = 1)

INTERFACE WITH 8085A

- LCD module is treated as I/O or RAM device.
- RD, WR signal of the 8085A are min. 400nsec. and insufficient to meet the needs of the 'E' signal of the module, min. 450nsec. RD, WR pulse width should be widened with providing TWAIT cycle.
- The address of the module as I/O device in the address area of the 8085A are:

Write function  # 'EX' (R/W = 0)
Write CGRAM/DDR data  # 'FX' (R/W = 0)
Read busy flag/address  # 'EX' (R/W = 1)
Read CGRAM/DDR data   # 'FX' (R/W = 1)
The module is treated as I/O device.

In I/O cycle, 1 WAIT STATE (TW) is automatically inserted, and OR signal (E) of RD, WR can be sufficient min. 450nsec.

When the module is treated as RAM device, 1 WAIT STATE (TW) should be provided with WAIT signal.

The address of the module as I/O device in the address area of the Z-80 are:

- Write function: # 'EX' (R/W = 0)
- Write CGRAM/DDRAM data: # 'FX' (R/W = 0)
- Read busy flag/address: # 'EX' (R/W = 1)
- Read CGRA/DDRAM data: # 'FX' (R/W = 1)
Ordering Number:
ELS16 for LMB1C16-DC, LMB2C16-DC
ELS24 for LMB1C24-DC, LMB2C24-DC
ELS40 for LMB1C40-DC, LMB2C40-DC

E-L backlighting is provided as a standard option. The blue-green lamp has 50cd/m² (14fL) of initial brightness and a half-life of 7,500 hours under a constant voltage/frequency condition of 100V/400Hz. Constant power DC to AC inverters are commercially available which will extend the half-life. The lamp color can be shifted to green, yellow or light red by the addition of a filter. However, once the lamp is put behind the LCD display, the brightness observed by the viewer is reduced to approximately 15% of what the lamp emits resulting in about 7.5cd/m² (2fL) which is more than sufficient for normal night time viewing and may need to be reduced for almost completely dark ambients.
This inverter is a compact encapsulated module used to generate the AC drive signal necessary to excite the E-L lamp. It should be operated from your unregulated DC line due to generated switching noise and will draw relatively low input currents.

**MAXIMUM RATINGS:**
- Input Voltage: 6.0 VDC
- Storage Temp.: -20 to +70°C

**OPERATING RANGE:**
- Input Voltage: 2.5 to 5.5 VDC
- Ambient Temp.: -10 to +60°C

**TYPICAL OPERATING CONDITIONS:**
- Input Voltage: 5.0 VDC

<table>
<thead>
<tr>
<th>Lamp Load</th>
<th>Input Current</th>
<th>Output Voltage</th>
<th>Output Freq.</th>
<th>Lamp Brightness</th>
</tr>
</thead>
<tbody>
<tr>
<td>16x1 or 2</td>
<td>29 mA</td>
<td>84 V</td>
<td>525 Hz</td>
<td>74 (2&quot;)</td>
</tr>
<tr>
<td>24x1 or 2</td>
<td>31 mA</td>
<td>91 V</td>
<td>500 Hz</td>
<td>53 (15&quot;)</td>
</tr>
<tr>
<td>40x1 or 2</td>
<td>35 mA</td>
<td>82 V</td>
<td>415 Hz</td>
<td>45 (13&quot;)</td>
</tr>
</tbody>
</table>

Overvoltage Input Current: 60 MA Max

**OPERATION OF INVERTER FROM LARGER INPUT VOLTAGES WITH VARIABLE LAMP DIMMING**

**DIMENSIONAL OUTLINE AND PINOUT:**
The HD146818 is a ICMC5608 peripheral CMOS device which combines three unique features: a complete time-of-day clock with alarm and one hundred calendar, a programmable periodic interrupt and square-wave generator, and 50 bytes of low-power static RAM.

This device includes HD6801, HD6301 multiplexed bus interface circuit and RO55's multiplexed bus interface as well, so it can be directly connected to HD8081, HD6301 and RO55.

The Real Time Clock plus RAM has two distinct uses. First, it is designed as battery powered CMOS part including all the common battery backed-up functions such as RAM, time, and calendar. Secondly, the HD146818 may be used with a CMOS microprocessor to relieve the software of timekeeping workload and to extend the available RAM of an MPU such as the HD6301.

- **FEATURES**
  - Time of Day Clock and Calendar
  - Counts Seconds, Minutes, and Hours of the Day
  - Counts Days of Week, Date, Month, and Year
  - Binary or BCD Representation of Time, Calendar, and Alarm
  - 12 or 24 Hour Clock with AM and PM in 12 Hour Mode
  - Automatic End of Month Recognition
  - Automatic Leap Year Compensation
  - Interfaced with Software as 64 RAM Locations
  - 14 Bytes of Clock and Control Register
  - 50 Bytes of General Purpose RAM
  - Three Interrupts are Separately Software Maskable and Testable
    - Time of Day Alarm, Once per Second to Once per Day
    - Periodic Rates from 30.5ps to 500ms
    - End of Clock Update Cycle
  - Programmable Square Wave Output Signal
  - Three Time Base Input Options
    - 4.194304 MHz
    - 1.048576 MHz
    - 32.768 kHz
  - Clock Output May be used as Microprocessor Clock Input
  - At Time Base Frequency < 4 or = 1
  - Multiplexed Bus Interface Circuit of HD6801, HD6301 and RO55
  - Low-Power, High-Speed, High-Density CMOS
  - Battery Backed-up Operation
  - Motorola MC146818 Compatible

### PIN ARRANGEMENT

The Flat Pack product is under development.

### ABSOLUTE MAXIMUM RATINGS

<table>
<thead>
<tr>
<th>Item</th>
<th>Symbol</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supply Voltage</td>
<td>VCC</td>
<td>4.5</td>
<td>V</td>
</tr>
<tr>
<td>Input Voltage</td>
<td>VIL</td>
<td>-0.3</td>
<td>V</td>
</tr>
<tr>
<td>Operating Temperature</td>
<td>Toper</td>
<td>0</td>
<td>°C</td>
</tr>
<tr>
<td>Storage Temperature</td>
<td>Tstg</td>
<td>-55</td>
<td>°C</td>
</tr>
</tbody>
</table>

* With respect to VCC

---

**RECOMMENDED OPERATING CONDITIONS**

<table>
<thead>
<tr>
<th>Item</th>
<th>Symbol</th>
<th>min</th>
<th>typ</th>
<th>max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supply Voltage</td>
<td>VCC</td>
<td>4.5</td>
<td>5.0</td>
<td>5.25</td>
<td>V</td>
</tr>
<tr>
<td>Input Voltage</td>
<td>VIL</td>
<td>-0.3</td>
<td>-</td>
<td>0.7</td>
<td>V</td>
</tr>
<tr>
<td>Operating Temperature</td>
<td>Toper</td>
<td>0</td>
<td>25</td>
<td>70</td>
<td>°C</td>
</tr>
</tbody>
</table>

* With respect to VCC

**ELECTRICAL CHARACTERISTICS**

**DC CHARACTERISTICS**

<table>
<thead>
<tr>
<th>Input &quot;High&quot; Voltage</th>
<th>ADX, AD1, CE, AS, R/W, DS, CKFS, PS</th>
<th>VIL</th>
<th>VCC-7.0</th>
<th>VCC</th>
<th>V</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input &quot;Low&quot; Voltage</td>
<td>ADX, AD1, CE, AS, R/W, DS, CKFS, PS</td>
<td>VIL</td>
<td>-0.3</td>
<td>0.7</td>
<td>V</td>
</tr>
<tr>
<td>Input Leakage Current</td>
<td>ADX, AD1, CE, AS, R/W, DS, CKFS, PS</td>
<td>IL =</td>
<td>2.5</td>
<td>µA</td>
<td></td>
</tr>
<tr>
<td>Three-state (off state) Input Current</td>
<td>ADX, AD1</td>
<td>ILSS</td>
<td>-</td>
<td>10</td>
<td>µA</td>
</tr>
<tr>
<td>Output Leakage Current</td>
<td>ADX, AD1</td>
<td>ICOH</td>
<td>-</td>
<td>10</td>
<td>µA</td>
</tr>
<tr>
<td>Output &quot;High&quot; Voltage</td>
<td>SOW, CKOUT</td>
<td>VOH</td>
<td>IOH = 1.6 mA</td>
<td>4.1</td>
<td>V</td>
</tr>
<tr>
<td>Input Leakage Current</td>
<td>ADX, AD1</td>
<td>IC</td>
<td>IINF = 10 µA</td>
<td>VCC-0.1</td>
<td>V</td>
</tr>
<tr>
<td>Output &quot;Low&quot; Voltage</td>
<td>CKOUT</td>
<td>VOL</td>
<td>IOL = 1.6 mA</td>
<td>0.5</td>
<td>V</td>
</tr>
<tr>
<td>Input Capacitance</td>
<td>ADX, AD1</td>
<td>CVIH</td>
<td>VCC = 0V</td>
<td>12.5</td>
<td>pF</td>
</tr>
<tr>
<td>Output Capacitance</td>
<td>SOW, CKOUT, IRO</td>
<td>CVIL</td>
<td>Ta = 25°C</td>
<td>12.5</td>
<td>pF</td>
</tr>
<tr>
<td>Supply Current (MPU Read/Write operating)</td>
<td>Crystal Oscillation</td>
<td>fosc = 4 MHz</td>
<td>fosc = 4 MHz</td>
<td>10</td>
<td>mA</td>
</tr>
<tr>
<td>Supply Current (MPU not operating)</td>
<td>Crystal Oscillation</td>
<td>fosc = 4 MHz</td>
<td>fosc = 4 MHz</td>
<td>7</td>
<td>mA</td>
</tr>
<tr>
<td>Supply Current (MPU Read/Write operating)</td>
<td>Crystal Oscillation</td>
<td>fosc = 4 MHz</td>
<td>fosc = 4 MHz</td>
<td>5</td>
<td>mA</td>
</tr>
<tr>
<td>Supply Current (MPU not operating)</td>
<td>Crystal Oscillation</td>
<td>fosc = 4 MHz</td>
<td>fosc = 4 MHz</td>
<td>2</td>
<td>mA</td>
</tr>
</tbody>
</table>

* Supply current of HD146818 is defined as the value when the time base frequency to be used is programmed into Register A. When power is turned on, these bits are undefined, so there is a case that current more than this shown specification may flow.

**Notes**

1. Permanent LSI damage may occur if maximum ratings are exceeded. Normal operation should be under recommended operating condition. If these conditions are exceeded, it could affect reliability of LSI.

2. Supply current of HD146818 is defined as the value when the time base frequency to be used is programmed into Register A. When power is turned on, these bits are undefined, so there is a case that current more than the above specification may flow.

Please note that this text is a direct transcription from the image and may contain errors or omissions. It is intended for educational purposes only and should not be used as the sole source of information.
**AC CHARACTERISTICS** \( V_{CC} = 4.5 \text{ to } 5.25 \text{V}, V_{SS} = 0 \text{V}, T_a = 0 \text{ to } +70^\circ \text{C}, \text{unless otherwise noted.} \)

**BUS TIMING**

<table>
<thead>
<tr>
<th>Item</th>
<th>Symbol</th>
<th>min</th>
<th>typ</th>
<th>max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cycle Time</td>
<td>( t_{CC} )</td>
<td>953</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Pulse Width, AS/ALE “High”</td>
<td>( t_{PW} )</td>
<td>100</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>AS Rise Time</td>
<td>( t_{AS} )</td>
<td></td>
<td>30</td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>AS Fall Time</td>
<td>( t_{AF} )</td>
<td></td>
<td>30</td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Delay Time DS/E to AS/ALE Rise</td>
<td>( t_{AD} )</td>
<td>40</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>DS Rise Time</td>
<td>( t_{DS} )</td>
<td></td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>DS Fall Time</td>
<td>( t_{DF} )</td>
<td></td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Pulse Width, DS/E Low or RD/WR “High”</td>
<td>( t_{PWLO} )</td>
<td>325</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Pulse Width, DS/E High or RD/WR “Low”</td>
<td>( t_{PWHL} )</td>
<td>300</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Delay Time, AS/ALE to DS/E Rise</td>
<td>( t_{ADSR} )</td>
<td>90</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Address Setup Time (R/W)</td>
<td>( t_{AS} )</td>
<td>15</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Address Setup Time (CE)</td>
<td>( t_{AS} )</td>
<td>55</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Address Hold Time (R/W, CE)</td>
<td>( t_{AH} )</td>
<td>10</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Muxed Address Valid Time to AS/ALE Fall</td>
<td>( t_{MAD} )</td>
<td>50</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Muxed Address Hold Time</td>
<td>( t_{MADH} )</td>
<td>20</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Peripheral Data Setup Time</td>
<td>( t_{DS} )</td>
<td>195</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Write Data Hold Time</td>
<td>( t_{DH} )</td>
<td>0</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Peripheral Output Data Delay Time From DS/E or RD</td>
<td>( t_{PD} )</td>
<td>220</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Read Data Hold Time</td>
<td>( t_{DR} )</td>
<td>100</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
</tbody>
</table>

**CONTROL SIGNAL TIMING**

<table>
<thead>
<tr>
<th>Item</th>
<th>Symbol</th>
<th>min</th>
<th>typ</th>
<th>max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oscillator Startup ( 1 \text{ MHz, 4 MHz, 32 kHz} )</td>
<td>( t_{RC} )</td>
<td>100</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>Reset Pulse Width</td>
<td>( t_{PW} )</td>
<td>5.0</td>
<td></td>
<td></td>
<td>ps</td>
</tr>
<tr>
<td>Reset Delay Time</td>
<td>( t_{PD} )</td>
<td>5.0</td>
<td></td>
<td></td>
<td>ps</td>
</tr>
<tr>
<td>Power Sense Pulse Width</td>
<td>( t_{PSW} )</td>
<td>5.0</td>
<td></td>
<td></td>
<td>ps</td>
</tr>
<tr>
<td>Power Sense Delay Time</td>
<td>( t_{PSD} )</td>
<td>5.0</td>
<td></td>
<td></td>
<td>ps</td>
</tr>
<tr>
<td>SRQ Release from DS</td>
<td>( t_{SRQ} )</td>
<td>2.0</td>
<td></td>
<td></td>
<td>ps</td>
</tr>
<tr>
<td>SRQ Release from RES</td>
<td>( t_{SRQ} )</td>
<td>2.0</td>
<td></td>
<td></td>
<td>ps</td>
</tr>
<tr>
<td>VRT Bit Delay</td>
<td>( t_{VRTD} )</td>
<td>2.0</td>
<td></td>
<td></td>
<td>ps</td>
</tr>
</tbody>
</table>

**Notes:**
- \( V_{HL} = V_{ON} = V_{CC} - 2.0 \text{V} \)
- \( V_{IL} = 0.7 \text{V} \)
- \( V_{OH} = 0.5 \text{V} \)

![Figure 1: Bus Read Timing (6801 Family)](image-url)
**Battery Backed-up Operation**

**Definition of Battery Backed-Up Operation**

1. **Active Functions:**
   - Clock function
   - Data bus read/write operation

2. **Inactive Functions:**
   - RAM data retention
   - RES, IRQ, CKFS, CKOUT, PS, SOW functions

**Battery Backed-Up Electrical Characteristics**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Test Condition</th>
<th>min</th>
<th>typ</th>
<th>max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supply Voltage</td>
<td>VCC</td>
<td>C.C. = 3.0V SOW: disable CKOUT: fosc (No load)</td>
<td>4MHz</td>
<td>1MHz</td>
<td>32kHz</td>
<td>4.5V V</td>
</tr>
<tr>
<td>Supply Current</td>
<td>ICC</td>
<td>VCC = 3.0V SOW: disable CKOUT: fosc (No load)</td>
<td>600 pA</td>
<td>350 pA</td>
<td>100 pA</td>
<td>pA</td>
</tr>
<tr>
<td>Crystal Oscillation</td>
<td>Iosc</td>
<td>1Hz</td>
<td>0</td>
<td>-</td>
<td>-</td>
<td>nA</td>
</tr>
<tr>
<td>External Clock</td>
<td>ICLC</td>
<td>VCC = 3.0V SOW: disable CKOUT: fosc (No load)</td>
<td>32kHz</td>
<td>18kHz</td>
<td>30 pA</td>
<td>pA</td>
</tr>
<tr>
<td>Battery Back-Up Transit Setup Time</td>
<td>tCE</td>
<td>= Fig. 9</td>
<td>0</td>
<td>-</td>
<td>-</td>
<td>ns</td>
</tr>
<tr>
<td>Operation Recovery Time</td>
<td>tR</td>
<td>Vpp</td>
<td>300</td>
<td>-</td>
<td>-</td>
<td>pA</td>
</tr>
<tr>
<td>Supply Voltage Fall Time</td>
<td>tV</td>
<td>Vpp</td>
<td>300</td>
<td>-</td>
<td>-</td>
<td>pA</td>
</tr>
<tr>
<td>Supply Voltage Rise Time</td>
<td>tV</td>
<td>Vpp</td>
<td>300</td>
<td>-</td>
<td>-</td>
<td>pA</td>
</tr>
<tr>
<td>Input &quot;High&quot; Voltage</td>
<td>VIL</td>
<td>VCC = 2.7V - 3.5V</td>
<td>CE, PS CKFS</td>
<td>0.7 x VCC</td>
<td>-</td>
<td>V</td>
</tr>
<tr>
<td>Input &quot;Low&quot; Voltage</td>
<td>VIL</td>
<td>VCC = 3.5V - 4.5V</td>
<td>RES</td>
<td>0.8 x VCC</td>
<td>-</td>
<td>V</td>
</tr>
<tr>
<td>Input &quot;High&quot; Voltage</td>
<td>VOIH</td>
<td>0.8 x VCC</td>
<td>CKFS, PS</td>
<td>0</td>
<td>-</td>
<td>V</td>
</tr>
<tr>
<td>Output &quot;High&quot; Voltage</td>
<td>VOIL</td>
<td>0</td>
<td>CKFS, PS</td>
<td>0.8 x VCC</td>
<td>-</td>
<td>V</td>
</tr>
</tbody>
</table>

*The time base frequency to be used needs to be chosen in Register A.*
**CRYSTAL OSCILLATION CIRCUIT**

The on-chip oscillator is designed for a parallel resonant crystal at 4.194304 MHz or 1.048576 MHz or 32.768 kHz frequencies. The crystal connections are shown in Figure 11.

---

**Table 1 Oscillator Circuit Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>fosc 4.194304 MHz</th>
<th>1.048576 MHz</th>
<th>32.768 kHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rs</td>
<td>150 kΩ</td>
<td>150 kΩ</td>
<td>5.6 kΩ</td>
</tr>
<tr>
<td>Rf</td>
<td>22 pF</td>
<td>33 pF</td>
<td>15 pF</td>
</tr>
<tr>
<td>Cm</td>
<td>22 pF</td>
<td>33 pF</td>
<td>33 pF</td>
</tr>
<tr>
<td>Cc</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cl</td>
<td>80 (1 max)</td>
<td>700 (1 max)</td>
<td>40 (1 max)</td>
</tr>
</tbody>
</table>

**NOTE FOR BOARD DESIGN OF THE OSCILLATION CIRCUIT**

In designing the board, the following notes should be taken when the crystal oscillator is used:

1. Crystal oscillator, load capacity Cm, Cc, Cl and Rs, Rf must be placed near the LSI as much as possible.
2. Normal oscillation may be disturbed when external noise is induced to pin 2 and 3.
3. Pin 2 signal line should be wired apart from pin 4 signal line as much as possible. Don't wire them in parallel, normal oscillation may be disturbed when this signal is fed back to OSC.
4. A signal line or a power source line must not cross or near the oscillation circuit line as shown in the right figure to prevent the induction from these lines and perform correct oscillation. The resistance among OSC1, OSC2, or other pins should be over 10MΩ.

The following design must be avoided.

---

**INTERFACE CIRCUIT FOR HD6801, HD6301 AND 8085 PROCESSOR**

HD146818 has a new interface circuit which permits the HD146818 to be directly interfaced with many type of multi-plexed bus microprocessor such as HD8081, HD6301 and 8085 etc.

---

**ADDRESS MAP**

Figure 14 shows the address map of the HD146818. The memory consists of 50 general purpose RAM bytes, 10 RAM bits which normally contain the time, calendar, and alarm data, and four control and status bytes. All 64 bytes are lderly readable and writable by the processor program except Registers C and D which are read only. Bit 7 of Register A and the second byte are also read only. Bit 7, of the second byte, always reads “0”. The contents of the four control and status registers are described in the Register section.

---

**Figure 13 Functional Diagram of the Bus Control Circuit**

Figure 13 shows the bus control circuit. This circuit automatically selects the processor type by using AS/ALE to latch the state of DS/RD pin. Since DS is always “Low” and RD is always “High” during AS/ALE, the latch automatically indicates which processor type is connected.

---

**Figure 14 Address Map**

The processor program obtains time and calendar information by reading the appropriate locations. The program may initialize the time, calendar, and alarm by writing to these RAM locations. The contents of the 10 time, calendar, and alarm byte may be either binary or binary-coded decimal (BCD).

Before initializing the internal registers, the SET bit in Register B should be set to a “1” to prevent time/calendar updates from occurring. The program initiates the 10 locations in the selected format (binary or BCD), then indicates the format in the data mode (DM) bit of Register B. All 10 time, calendar, and alarm bytes must use the same data mode, either binary or BCD. The SET bit may now be cleared to allow updates. Once initialized the real-time clock makes all updates in the selected data mode. The data mode cannot be changed without reinitializing the 10 data bytes.
Table 2 shows the binary and BCD formats of the 10 time, calendar, and alarm locations. The 24:12 bit in Register B establishes whether the hour locations represent 1-12 or 0-23. The 24:12 bit cannot be changed without reinitializing the hour locations. When the 12-hour format is selected the high-order bit of the hours byte represents PM when it is a "1." The time, calendar, and alarm bytes are not always accessible by the processor program. Once per-second the the 10 bytes are switched to the update logic to be advanced by one second, and to check and to change the alarm conditions. If any of the 10 bytes are read at this time, the data outputs are unfilled. The update lockout time is 248 µs at the 4.194304 MHz and 1.048576 MHz time bases and 194 µs for the 32.768 kHz time base. The Update Cycle section shows how to accommodate the update cycle in the processor program.

The three alarm bytes may be used in two ways. When the program inserts an alarm time in the appropriate hour, minutes, and seconds alarm locations, the alarm interrupt initiated at the specified time each day if the alarm enable is "1." The alternate usage is to insert a "don't care" state, one or more of three alarm bytes. The "don't care" code any hexadecimal byte from 00 to FF. That is, the two non-significant bits of each byte, when set to "1," create a "don't care" situation. An alarm interrupt each hour is created with a "don't care" code in the hour alarm location. Similarly, alarm is generated every minute with "don't care" code in the hour and minutes alarm bytes. The "don't care" code in all three alarm bytes create an interrupt every second.

Table 2 Time, Calendar, and Alarm Data Modes

<table>
<thead>
<tr>
<th>Address Location</th>
<th>Function</th>
<th>Decim lage</th>
<th>Binary Data Mode</th>
<th>BCD Data Mode</th>
<th>Example*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seconds</td>
<td>0 - 59</td>
<td>$00 - $38</td>
<td>$00 - $59</td>
<td>15</td>
<td>21</td>
</tr>
<tr>
<td>Seconds Alarm</td>
<td>0 - 59</td>
<td>$00 - $38</td>
<td>$00 - $59</td>
<td>15</td>
<td>21</td>
</tr>
<tr>
<td>Minutes</td>
<td>0 - 59</td>
<td>$00 - $38</td>
<td>$00 - $59</td>
<td>3A</td>
<td>58</td>
</tr>
<tr>
<td>Minutes Alarm</td>
<td>0 - 59</td>
<td>$00 - $38</td>
<td>$00 - $59</td>
<td>3A</td>
<td>58</td>
</tr>
<tr>
<td>Hours (12 Hour Mode)</td>
<td>1 - 12</td>
<td>$01 - $12 (AM) and $01 - $12 (PM)</td>
<td>$01 - $92 (PM)</td>
<td>05</td>
<td>05</td>
</tr>
<tr>
<td>Hours (24 Hour Mode)</td>
<td>0 - 23</td>
<td>$00 - $17</td>
<td>$00 - $23</td>
<td>05</td>
<td>05</td>
</tr>
<tr>
<td>Hours Alarm (12 Hour Mode)</td>
<td>1 - 12</td>
<td>$01 - $12 (AM) and $01 - $12 (PM)</td>
<td>$01 - $92 (PM)</td>
<td>05</td>
<td>05</td>
</tr>
<tr>
<td>Hours Alarm (24 Hour Mode)</td>
<td>0 - 23</td>
<td>$00 - $17</td>
<td>$00 - $23</td>
<td>05</td>
<td>05</td>
</tr>
<tr>
<td>Day of the Week</td>
<td>Sunday = 1</td>
<td>$01 - S7</td>
<td>$01 - S7</td>
<td>05</td>
<td>05</td>
</tr>
<tr>
<td>Day of the Month</td>
<td>1 - 31</td>
<td>$01 - S7</td>
<td>$01 - S7</td>
<td>0F</td>
<td>1b</td>
</tr>
<tr>
<td>Month</td>
<td>1 - 12</td>
<td>$01 - $12</td>
<td>$01 - $12</td>
<td>02</td>
<td>02</td>
</tr>
<tr>
<td>Year</td>
<td>0 - 99*</td>
<td>$00 - $63</td>
<td>$00 - $99</td>
<td>4F</td>
<td>79</td>
</tr>
</tbody>
</table>

* Example: 5:58:21 Thursday 15th February 1979
* Set the lower two digits of year in AD. If this number is multiple of 4, update applied to leap year it exceeded.

- Static CMOS RAM
The 50 general purpose RAM bytes are not dedicated within the HD146818. They are not by the processor program, and are fully available during the update cycle.
Within the static CMOS CMOS memory must be battery backup, very frequently there is in other non-volatile data that must be retained when main power is removed. The 50 user RAM bytes serve the need for low-power CMOS battery-backed storage, and extend the RAM available to the program.
When further CMOS RAM is needed, additional HD146818s may be included in the system. The time/calendar functions may be disabled by holding the dividers, in Register A, in the reset state by setting the SET bit in Register B or by removing the oscillator. Holding [1] the dividers in reset prevents interrupts or SWO output from operating while setting the SET bit allows these functions to operate. With the dividers clear, the available interrupt inputs are the IRQ pin in Register A, Registers C and D, and the high-order bit of the seconds byte cannot be effectively used as general purpose RAM.

- Interrupts
The RTC plus RAM includes three separate fully autonomous sources of time information. The three sources may be programmed to occur at rates from once per second to once-a-day. The periodic interrupts may be selected for any half-second to 30.517 µs. The updated-period interrupt may be use to initiate a program that is a time-pulse output in the 50:9 pin. If a program is selected by Register A, the square-wave frequency is set as shown in Table 4. The SQW pulse width can be varied between 15 and 60% of the frequency. The square-wave output is much of the power and can be used for timing or counting applications. For example, it can be used as a frequency standard or for external use, or a frequency synthesizer, or be used to control one or more audio tones under program control.

- IROF in Register C is a "1" whenever the INT pin is being driven.

The processor program can determine that the RTC initiated interrupt by reading Register A. C a"1" in bit 7 (IROF bit) indicates that one of more interrupts have been initiated by the processor program. The act of reading Register C clears all the then-active flag bits, plus the IRQQ bit. When the processor finds IROF set, it should look at each of the individual flag bits in the base register which have the corresponding interrupt mask bits set and service each interrupt which is set. Again, more than one interrupt flag bit may be set.

- Divider Stages
The HD146818 has 22 binary-divider stages following the time base as shown in Figure 10. The output of the divider is a 1 Hz signal to the update-cycle logic. The dividers are controlled by three divider bits (DV2, DV1, and DV0) in Register A.

- Divider Control
The divider-control bits have three uses, as shown in Table 3. Three usable operating time bases may be selected (4.194 304 MHz, 1.048576 MHz, or 32.768 kHz). The divider chain may be held reset, which allows precision setting of the time base. When the divider is changed from reset to an operating time base, the first update cycle is one second later. The divider-control bits are also used to facilitate testing the HD146818.

- Interrupt
The periodic interrupt is enabled by the IRQ pin to be triggered from once every 500 µs to once every 30.517 µs. The periodic interrupt is separate from the alarm interrupt which may be output from once-per-second to once-per-day.
Table 4 shows that the periodic interrupt rate is selected with the same Register A bits which select the square-wave frequency. Changing one also changes the other. Each function may be separately enabled so that a program could switch between the two features or one both. The 50:9 pin is enabled by the SWQE bit. Similarly the periodic interrupt is enabled by the PIP bit in Register B. Periodic interrupt is usable by practically all real-time systems. It can be used to scan for all forms of input from contact closures to avoid receive bits or thy. It can be used in multi-plexing displays or with software circuits to measure inputs, create output intervals, or await the next needed software function.
Table 4  Periodic Interrupt Rate and Square Wave Output Frequency

<table>
<thead>
<tr>
<th>Rate Select Control Register 1</th>
<th>Periodic Interrupt Rate (Hz)</th>
<th>SOW Output Frequency</th>
<th>Periodic Interrupt Rate (Hz)</th>
<th>SOW Output Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>RS3</td>
<td></td>
<td></td>
<td>RS1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>None</td>
<td>None</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>30.517 Hz</td>
<td>3.90625 ms</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>61.035 Hz</td>
<td>7.8125 ms</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>122.070 Hz</td>
<td>15.625 ms</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>244.141 Hz</td>
<td>23.125 ms</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>488.28 Hz</td>
<td>46.875 ms</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>976.56 Hz</td>
<td>95.625 ms</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1953.125 Hz</td>
<td>190.625 ms</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3.90625 Hz</td>
<td>256 Hz</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>7.8125 Hz</td>
<td>512 Hz</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>15.625 Hz</td>
<td>256 Hz</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>31.25 Hz</td>
<td>512 Hz</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>62.5 Hz</td>
<td>256 Hz</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>125 Hz</td>
<td>256 Hz</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
<td>500 Hz</td>
<td>256 Hz</td>
</tr>
</tbody>
</table>

- Initialization of the Time and the Start Sequence
  The first update of the time occurs about 500mS later after the SET bit of control register B is set. Keep following in mind when initializing and adjusting the time.

- Procedure of time initialization
  (1) Set the SET bit of control register B. (SET = “1”)
  (2) Set “1” into all the DVO, 1, 2 bits of control register A.
  (3) Set the time and calendar to each RAM.
  (4) Set the frequency in use into DVO, 1 and DVO.
  (5) Reset the SET bit. (SET = “0”)

Figure 15  Time Initialization and the First Update

Restriction on Time-of-day and Calendar Initialization

There is a case in HD46618 (RTC) that update is not executed correctly if time of day and calendar shown below are initialized. Therefore, initialize the RTC without using time of day shown below.

- UPDATE CYCLE

The HD46618 executes an update cycle once per second assuming one of the proper time base is in place, the status is not clear, and the SET bit in Register B is clear. The SET bit in the “1” state permits the program to initialize the time if calendar bytes by stopping an existing update and prevent a new one from occurring.

The primary function of the update cycle is to increment the second byte, check for overflow, increment the minute byte when appropriate and so forth through the year-byte. The update cycle also adjusts each byte to the corresponding time byte and issues an alarm if a match or if a "don't care" code (111100000) is present in all three positions.

With a 4.194304 MHz or 1.048576 MHz time base the update cycle takes 248 µs while a 32.768 kHz time base update cycle takes 1984 µs. During the update cycle, the time, calendar, and alarm bytes are not accessible by the processor program. The HD46618 protects the program from reading transitional data. This protection is provided by switching the time, calendar, and alarm portion of the RAM off the microprocessor bus during the entire update cycle. If the processor reads these RAM locations before the update is complete the output will be undefined. The update in progress (UIF) status bit is set during the interval.

A program which randomly accesses the time and date information finds data unavailable statistically once every 4302 attempts. Three methods of accommodating nonavailability during update are usable by the program. In discussing the three methods it is assumed that at random point users program are able to call a subroutine to obtain the time of day.

The first method of avoiding the update cycle updates the microprocessor. If enabled, an interrupt occurs after each update cycle which indicates that 999 mS are available to read valid time and date information. During this time a display could be updated or the information could be transmitted to continuously available RAM. Before leaving the interrupt service routine, the IRQ bit in Register C should be cleared.

The second method uses the update-in-progress bit (UIF) in Register A to determine if the update cycle is in progress or not. The UIF bit will pulse once-per-second. Statistically, the UIF bit will indicate that time and date information is unavailable once even every 2322 updates. After the UIF bit goes "1", the update cycle begins 244 µs later. Therefore, if a "0" is read on the UIF bit, the micro has at least 244 µS before the time/calendar data may be invalid. The user should avoid interrupt service routines that would cause the time needed to read valid time/calendar data to exceed 244 µS.

The third method uses a periodic interrupt to determine if an update cycle is in progress. The UIF bit in Register A is set "1" between the setting of the PF bit in Register C (see Figure 16) Periodic interrupts that occur at a rate of greater than 31µS + 1µS allow valid time and date information to be read at each occurrence of the periodic interrupt. The reads should be completed within (tp1 + 2) + 1µS to insure that data is not read during the update cycle.

- POWER-DOWN CONSIDERATIONS

In most systems, the HD46618 must continue to keep time when system power is removed. In such systems, a conversion from system power to an alternate power supply, usually a battery, must be made. During the transition from system to battery power, the designer of a battery backed-up RTC system must protect data integrity, minimize power consumption, and ensure hardware reliability according to the specification described in the section regarding Battery Backed-up operation.

The chip enable (CE) pin controls all bus inputs (R/W, DS, AS, AD0 ~ AD10). CE, when negated, drives all undefined modification of the RTC data by the bus. CE also reduces power consumption by reducing the number of transitions seen internally.

Power consumption may be further reduced by removing resistive and capacitive loads from the clock out (CKOUT) pin and the squarewave (SOW) pin.

During and after the power source conversion, the Vpp maximum specification must never be exceeded. Failure to meet the Vpp maximum specification can cause a virtual SCR to appear which may result in excessive current drain and destruction of the part.

Figure 16  Update-Ended and Periodic Interrupt Relationship
The on-chip oscillator is designed for a parallel resonant crystal at 4.193404 MHz or 1.048576 MHz or 32.768 kHz frequencies. The crystal connections are shown in Figure 11.

- **CKOUT - Clock Out (Output)**
  The CKOUT pin is an output at the time-base frequency divided by 1 or 4. A major use for CKOUT is as the input clock to the microprocessor; thereby saving the cost of a crystal oscillator. The frequency of CKOUT depends upon the time-base frequency and the state of the CKFS pins as shown in Table 5.

- **CKFS - Clock Out Frequency Select (Input)**
  The CKFS pin is an output at the time-base frequency divided by 1 or 4. CKFS tied to VCC causes CKOUT to be at the same frequency as the time base at the OSC, pin. When CKFS is at VCC, CKOUT is the OSC, time-base frequency divided by four. Table 5 summarizes the effect of CKFS.

![Figure 17: External Time Base Connection](image)

The clock output frequencies are as follows:

<table>
<thead>
<tr>
<th>Time Base (USC,)] Frequency</th>
<th>Clock Frequency Select Pin</th>
<th>Clock Frequency Output Pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.193404 MHz</td>
<td>&quot;High&quot;</td>
<td>4.193404 MHz</td>
</tr>
<tr>
<td>4.194504 MHz</td>
<td>&quot;Low&quot;</td>
<td>1.048576 MHz</td>
</tr>
<tr>
<td>1.048576 MHz</td>
<td>&quot;High&quot;</td>
<td>1.048576 MHz</td>
</tr>
<tr>
<td>32.768 kHz</td>
<td>&quot;High&quot;</td>
<td>32.768 kHz</td>
</tr>
<tr>
<td>32.768 kHz</td>
<td>&quot;Low&quot;</td>
<td>8.192 kHz</td>
</tr>
</tbody>
</table>

- **SOW - Square Wave (Output)**
  The SOW pin can output a signal one of 15 of the 22 internal divider stages. The frequency and output enable of the SOW may be altered by programming Register A, as shown in Table 4. The SOW signal may be turned on and off using a bit in Register B.

- **AD1 - AD0 - Multiplexed Bidirectional Address/Data Bus**
  Multiplexed bus processors save pins by generating the address during the first portion of the bus cycle and using the same pins during the second portion for data. Address-then-data multiplexing does not slow the access time of the HD46188 since the bus reversal from address to data is occurring during the internal RAM access time.

- **RESET - Reset (Input)**
  The RES pin does not affect the clock, calendar, or RAM functions. Use of the RES pin in the "High" state may be an advantage of using the RTC as a clock source in applications where the system clock is not used, or is used to maintain some system clock. Use of the RES pin in the "High" state does not affect the calendar. The calendar is independent of the system clock. The system clock is used to update the calendar. The calendar never changes in the "High" state unless the system clock is present. Use of the RES pin in the "High" state does not affect the programming of the calendar. The calendar is updated by the system clock. Use of the RES pin in the "High" state does not affect the programming of the calendar. The calendar is updated by the system clock.
cycle is in progress or will soon begin. When UIP is a "0", the update cycle is not in progress and will not be for at least 244 ps (for all time bases). This is detailed in Table 6. The time, calendar, and alarm information in RAM is fully available to the program when the UIP bit is zero; it is not in transition. The UIP bit is a read-only bit, and is not affected by Reset. Writing the SET bit in Register B to a "1" inhibits any update cycle and then clears the UIP status bit.

**DV2, DV1, DVO** Three bits are used to permit the program to select various conditions of the 22-stage divider chain. The divider selection bits identify which of the three time-base frequencies in use. Table 3 shows that time bases of 4 194304 MlZ, 1 048576 MlZ, and 32 768 KHz may be used. The divider selection bits are also used to reset the divider chain. When the time calendar is first initialized, the processor program may reset the divider to the precise time stored in the RAM. When the divider reset is removed the first update cycle begins half a second later. These three read/write bits are never modified by the RTC and are not affected by RES.

**RS3, RS2, RS1, RS0** The four rate selection bits select one of 15 taps on the 22-stage divider, or disable the divider output. The tap selected may be used to generate an output square wave (SQW pin) and/or a periodic interrupt. The program may do one of the following: 1) enable the interrupt with the PI1 bit, 2) enable the SQW output pin with the SQWE bit, 3) enable both at the same time at the rate, or 4) enable neither. Table 4 lists the periodic rate frequencies and the square-wave frequencies that may be chosen with the RS bits. These four bits are read/write bits which are not affected by RES and are never changed by the RTC.

**Register B (D00)**

<table>
<thead>
<tr>
<th>MB</th>
<th>Read/Write</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>b7</td>
<td>b6</td>
<td>b5</td>
</tr>
</tbody>
</table>

**Register D (D50)**

<table>
<thead>
<tr>
<th>MB</th>
<th>Read Only</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>b7</td>
<td>b6</td>
<td>b5</td>
</tr>
</tbody>
</table>

VBT - The valid RAM and time (VBT) bit indicates the condition of the contents of the RAM, provided the power sense (PS) pin is satisfactorily connected. A "0" appears in the VBT bit when the power sense pin is "Low." The processor program can set the VBT bit when the time and calendar are initialized to indicate that the RAM and time are valid. The VBT is a read-only bit which is not modified by the RES pin.

**Register C (D5C)**

<table>
<thead>
<tr>
<th>MB</th>
<th>Read Only</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRQF</td>
<td>PF</td>
<td>AF</td>
</tr>
</tbody>
</table>

**Register E (D60)**

<table>
<thead>
<tr>
<th>MB</th>
<th>Read Only</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>b7</td>
<td>b6</td>
<td>b5</td>
</tr>
</tbody>
</table>

**NOTE FOR USE** Input Signals which is not necessary for men's application, should be used fixed to "High" or "Low" level. This it applicable to the following signal pin. CXTS, PS.
HD146818

RESTRICTION ON HD146818 USAGE (1)
The daylight saving function can not be performed on the HD146818P (X type). So do not use this function for the system design.

< Type number > HD146818P (X type . . . .) Marked as follows - X or RX

< Restriction on usage > Please set "0" to DSE bit (Daylight Saving Enable bit) on initializing the control register B. DSE = "1" is prohibited.

RESTRICTION ON HD146818 USAGE (2)
Access to HD146818 needs to be performed under following conditions.
(i) Chip enable (CE) must be asserted to active "Low" level only when MPU performs read/write operation from/into internal RAM (Time and Calendar RAM, Control register, User RAM).
(ii) User RAM and control register must be accessed in less than 1/4 frequency shown below.
(Example: After one access, non-access cycles more than three cycles are necessary to be inserted.)

[Example 1]
Address
4 Cycles (min)
CE
Access to HD146818

[Example 2]
Address
8 Cycles
CE
Access to HD146818 (Two Continuous Accesses)

As shown in the above [example 2], when HD146818 is accessed continuously, continuous access must not be executed over fifty times.

(iii) The application that User RAM is used for program area should be avoided. (Inhibit continuous access.)
(iv) Minimize the noise by inserting noise bypass condenser between power supply and ground pin (VCC - VSS).
(Insert noise bypass condenser as near HD146818 as possible.)

606

HD6318, HD63A18
RTC (Real Time Clock Plus RAM)

ADVANCE INFORMATION

HD6318P, HD63A18P

The HD6318 is a 1MCS6800 peripheral CMOS device which combines three unique features: a complete time-of-day clock with alarm and one hundredfold calendar, a programmable periodic interrupt and square-wave generator, and 50 bytes of Low-power static RAM.

This device includes HD6801, HD6301 multiplexed bus interface circuit and 8085's multiplexed bus interface as well, so it can be directly connected to HD6801, HD6301 and 8085.

The Real Time Clock plus RAM has two distinct uses. First, it is designed as battery powered CMOS part including all the common battery backed-up functions such as RAM, time, and calendar. Secondly, the HD6318 may be used with a CMOS microprocessor to relieve the software of timekeeping workload and to extend the available RAM of an MPU such as the HD6301.

- FEATURES
- Revisited product of HD146818 in both Function and Characteristics
- Compatible with HD146818 and Motorola MC146818
- Time-of-Day Clock and Calendar
  - Counts Seconds, Minutes, and Hours of the Day
  - Counts Days of Week, Date, Month, and Year
  - Binary or BCD Representation of Time, Calendar, and Alarm
- 12- or 24 Hour Clock with AM and PM in 12 Hour Mode
- Automatic End of Month Recognition
- Automatic Leap Year Compensation
- Interfaced with Software as 64 RAM Locations
  - 14 Bytes of Clock and Control Register
  - 50 Bytes of General Purpose RAM
- Three Interrupts are Separately Software Maskable and Testable
  - Time of Day Alarm, Once per Second to Once per Day
  - Periodic Rates from 30.5 ps to 500 ms
  - End-of-Check Update Cycle
- Programmable Square Wave Output Signal
- Three Time Base Input Options
  - 4.194304 MHz
  - 1.048576 MHz
  - 32.768 kHz
- Clock Output May be used as Microprocessor Clock Input
  - 8 MHz
  - 4 MHz
  - 2 MHz
- Multiplexed Bus Interface Circuit of HD6801, HD6301 and 8085
- Low Power, High Speed, High-Density CMOS
- Battery Backed-up Operation

- PIN ARRANGEMENT

- TYPE OF PRODUCTS

<table>
<thead>
<tr>
<th>Type No.</th>
<th>Bus Timing</th>
</tr>
</thead>
<tbody>
<tr>
<td>HD6318</td>
<td>1.0 MHz</td>
</tr>
<tr>
<td>HD63A18</td>
<td>1.5 MHz</td>
</tr>
</tbody>
</table>

(Top View)

607
Fast, Complete 12-Bit A/D Converter with Microprocessor Interface

FEATURES

- Complete 12-Bit A/D Converter with Reference and Clock
- Full 8- or 16-Bit Microprocessor Bus Interface
- Guaranteed Linearity Over Temperature
  - 0 to +70°C - AD574AJ, AK, AL
  - -55°C to +125°C - AD574AS, AT, AU
- No Missing Codes Over Temperature
  - Fast Successive Approximation Conversion - 25μs
  - Buried Zener Reference for Long-Term Stability and Low Gain T.C. 10ppm/°C max AD574AL
  - 12.5ppm/°C max AD574AU
- Low Profile 28-Pin Ceramic DIP
- Low Power: 390mW
- Low Cost

PRODUCT DESCRIPTION

The AD574A is a complete 12-bit successive-approximation analog-to-digital converter with 3-state output buffer circuitry for direct interface to an 8-, 12-, or 16-bit microprocessor bus. The AD574A design is implemented with two LSI chips each containing both analog and digital circuitry, resulting in the maximum performance and flexibility at the lowest cost.

One chip is the high performance AD565A 12-bit DAC and voltage reference. It contains the high speed current output switching circuitry, laser-trimmed thin film resistor network, low T.C. buried zener reference and the precision input scaling and bipolar offset resistors. This chip is laser-trimmed at the wafer stage (LWT) to adjust ladder network linearity, voltage reference tolerance and temperature coefficient, and the calibration accuracy of input scaling and bipolar offset resistors.

The second chip uses the proven LCI (linear-compatible integrated injection logic) process to provide the low-power 2-level successive-approximation register, converter control circuitry, clock, bus interface, and the high performance latching comparator. The precision, low-drift comparator is adjusted for initial input offset error at the wafer stage by the "zener-zap" technique which trims the comparator input stage to 1/10 LSB typical error. This form of trimming, while cumbersome for complex ladder networks, is an attractive alternative to thin film resistor trimming for a simple offset adjustment and eliminates the need for thin film processing for this portion of the circuitry.

The AD574A is available in six different grades. The AD574AJ, AK, and AL grades are specified for operation over the 0 to +70°C temperature range. The AD574AS, AT, and AU are specified for the -55°C to +125°C range. All grades are packaged in a low-profile, 0.600 inch wide, 28-pin hermetically-sealed ceramic DIP.

PRODUCT HIGHLIGHTS

1. The AD574A interfaces to most popular microprocessors with an 8-, 12-, or 16-bit bus without external buffers or peripheral interface controllers. Multiple-mode three-state output buffers connect directly to the data bus while the read and convert commands are taken from the control bus. The 12-bits of output data can be read either as one 12-bit word or as two 8-bit bytes (one with 8 data bits, the other with 4 data bits and 4 trailing zeros).

2. The precision, laser-trimmed scaling and bipolar offset resistors provide four calibrated ranges, 0 to +10 and 0 to +20 volts unipolar, or -5 to +5 and -10 to +10 volts bipolar. Typical bipolar offset and full scale calibration of ±0.1% can be trimmed to zero with one external component each.

3. The internal buried zener reference is trimmed to 10.00 volts with 1% maximum error and 15ppm/°C typical T.C. The reference is available externally and can drive up to 1.5mA beyond that required for the reference and bipolar offset resistors.

4. The two-chip construction renders the AD574A inherently more reliable than hybrid multi-chip designs. All three military grades have guaranteed linearity error over the full -55°C to +125°C and are especially recommended for high performance needs in harsh environments. These units are available processed to MIL-STD-883B, Level B.
SPECIFICATIONS (typical @+2°C with \( V_{CC} = +15\text{V} \) or +12V, \( V_{LOGIC} = +5\text{V}, \ V_{EE} = -15\text{V} \) or -12V, unless otherwise specified)

DC AND TRANSFER ACCURACY SPECIFICATIONS

<table>
<thead>
<tr>
<th>MODEL</th>
<th>AD574AJ</th>
<th>AD574AK</th>
<th>AD574AL</th>
<th>UNITS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>RESOLUTION (max)</strong></td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>Bits</td>
</tr>
<tr>
<td><strong>LINEARITY ERROR</strong></td>
<td>±1</td>
<td>±1/2</td>
<td>±1/2</td>
<td>LSB</td>
</tr>
<tr>
<td>( 25^°C ) (max)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( T_{min} ) to ( T_{max} ) (max)</td>
<td>±1</td>
<td>±1/2</td>
<td>±1/2</td>
<td>LSB</td>
</tr>
<tr>
<td><strong>DIFFERENTIAL LINEARITY ERROR</strong> (Minimum resolution for which no missing codes are guaranteed)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( 25^°C )</td>
<td>11</td>
<td>12</td>
<td>12</td>
<td>Bits</td>
</tr>
<tr>
<td>( T_{min} ) to ( T_{max} )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>UNIPOLAR OFFSET (max) (Adjustable to zero)</strong></td>
<td>±2</td>
<td>±2</td>
<td>±2</td>
<td>LSB</td>
</tr>
<tr>
<td><strong>BIPOLAR OFFSET (max) (Adjustable to zero)</strong></td>
<td>±10</td>
<td>±24</td>
<td>±24</td>
<td>LSB</td>
</tr>
<tr>
<td><strong>FULL SCALE CALIBRATION ERROR</strong> (with fixed 50Ω resistor from REF OUT to REF IN)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(Adjustable to zero) ( 25^°C ) (max)</td>
<td>0.25</td>
<td>0.25</td>
<td>0.25</td>
<td>% of F.S.</td>
</tr>
<tr>
<td>( T_{min} ) to ( T_{max} ) (Without Initial Adjustment)</td>
<td>0.47</td>
<td>0.37</td>
<td>0.30</td>
<td>% of F.S.</td>
</tr>
<tr>
<td>( T_{min} ) to ( T_{max} ) (With Initial Adjustment)</td>
<td>0.22</td>
<td>0.12</td>
<td>0.05</td>
<td>% of F.S.</td>
</tr>
<tr>
<td><strong>TEMPERATURE RANGE</strong></td>
<td>0 to +70</td>
<td></td>
<td></td>
<td>°C</td>
</tr>
<tr>
<td><strong>TEMPERATURE COEFFICIENTS (Using internal reference)</strong> Guaranteed max change</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( T_{min} ) to ( T_{max} )</td>
<td>±2</td>
<td>±1</td>
<td>±1</td>
<td>LSB</td>
</tr>
<tr>
<td>Unipolar Offset</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(10)</td>
<td>(5)</td>
<td>(5)</td>
<td>(ppm/°C)</td>
<td></td>
</tr>
<tr>
<td>Bipolar Offset</td>
<td>±2</td>
<td>±1</td>
<td>±1</td>
<td>LSB</td>
</tr>
<tr>
<td>(10)</td>
<td>(5)</td>
<td>(5)</td>
<td>(ppm/°C)</td>
<td></td>
</tr>
<tr>
<td>Full Scale Calibration</td>
<td>±9</td>
<td>±5</td>
<td>±2</td>
<td>LSB</td>
</tr>
<tr>
<td>(50)</td>
<td>(27)</td>
<td>(10)</td>
<td>(ppm/°C)</td>
<td></td>
</tr>
<tr>
<td><strong>POWER SUPPLY REJECTION</strong> Max change in Full Scale Calibration</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+13.5 V ( \leq V_{CC} \leq +16.5 \text{V} ) or +11.4 V ( \leq V_{CC} \leq +12.6 \text{V} )</td>
<td>±2</td>
<td>±1</td>
<td>±1</td>
<td>LSB</td>
</tr>
<tr>
<td>+4.5 V ( \leq V_{LOGIC} \leq +5.5 \text{V} )</td>
<td>±1/2</td>
<td>±1/2</td>
<td>±1/2</td>
<td>LSB</td>
</tr>
<tr>
<td>-16.5 V ( \leq V_{EE} \leq -13.5 \text{V} ) or -12.6 V ( \leq V_{EE} \leq -11.4 \text{V} )</td>
<td>±2</td>
<td>±1</td>
<td>±1</td>
<td>LSB</td>
</tr>
<tr>
<td><strong>ANALOG INPUTS</strong> Input Ranges</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bipolar</td>
<td>-5 to +5</td>
<td></td>
<td></td>
<td>Volts</td>
</tr>
<tr>
<td>Unipolar</td>
<td>-10 to +10</td>
<td></td>
<td></td>
<td>Volts</td>
</tr>
<tr>
<td><strong>Input Impedance</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 Volt Span</td>
<td>0 to +10</td>
<td></td>
<td></td>
<td>Volts</td>
</tr>
<tr>
<td>20 Volt Span</td>
<td>0 to +20</td>
<td></td>
<td></td>
<td>Volts</td>
</tr>
<tr>
<td><strong>POWER SUPPLIES</strong> Operating Range</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( V_{LOGIC} )</td>
<td>+4.5 to +5.5</td>
<td></td>
<td></td>
<td>Volts</td>
</tr>
<tr>
<td>( V_{CC} )</td>
<td>+11.4 to +16.5</td>
<td></td>
<td></td>
<td>Volts</td>
</tr>
<tr>
<td>( V_{EE} )</td>
<td>-11.4 to -16.5</td>
<td></td>
<td></td>
<td>Volts</td>
</tr>
<tr>
<td>Operating Current</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( I_{LOGIC} )</td>
<td>30 typ, 40 max</td>
<td></td>
<td></td>
<td>mA</td>
</tr>
<tr>
<td>( I_C )</td>
<td>2 typ, 5 max</td>
<td></td>
<td></td>
<td>mA</td>
</tr>
<tr>
<td>( V_{EE} )</td>
<td>18 typ, 30 max</td>
<td></td>
<td></td>
<td>mA</td>
</tr>
<tr>
<td><strong>POWER DISSIPATION</strong></td>
<td>390 typ, 725 max</td>
<td></td>
<td></td>
<td>mW</td>
</tr>
<tr>
<td><strong>INTERNAL REFERENCE VOLTAGE</strong> Output Current (available for external loads) (External load should not change during conversion)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10.00 ±0.1 (max)</td>
<td>1.5 max</td>
<td></td>
<td></td>
<td>Volts</td>
</tr>
</tbody>
</table>
| NOTES

1 The reference should be buffered for operation on ±12V supplies.
Specifications subject to change without notice.
## DC AND TRANSFER ACCURACY SPECIFICATIONS

<table>
<thead>
<tr>
<th>MODEL</th>
<th>AD574AS</th>
<th>AD574AT</th>
<th>AD574AU</th>
<th>UNITS</th>
</tr>
</thead>
<tbody>
<tr>
<td>RESOLUTION (max)</td>
<td>12</td>
<td>12</td>
<td>12</td>
<td>Bits</td>
</tr>
<tr>
<td>LINEARITY ERROR</td>
<td>±1</td>
<td>±1/2</td>
<td>±1/2</td>
<td>LSB</td>
</tr>
<tr>
<td>25°C (max)</td>
<td>±1/2</td>
<td>±1/2</td>
<td>±1/2</td>
<td>LSB</td>
</tr>
<tr>
<td>-25°C to +85°C (max)</td>
<td>±1</td>
<td>±1/2</td>
<td>±1/2</td>
<td>LSB</td>
</tr>
<tr>
<td>-55°C to +125°C (max)</td>
<td>±1</td>
<td>±1</td>
<td>±1</td>
<td>LSB</td>
</tr>
</tbody>
</table>

### DIFFERENTIAL LINEARITY ERROR

(Minimum resolution for which no missing codes are guaranteed)

<table>
<thead>
<tr>
<th>25°C</th>
<th>11</th>
<th>12</th>
<th>12</th>
<th>Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tmin to Tmax</td>
<td>±2</td>
<td>±2</td>
<td>±2</td>
<td>LSB</td>
</tr>
</tbody>
</table>

### UNIPOLAR OFFSET (max) (Adjustable to zero)

<table>
<thead>
<tr>
<th>11</th>
<th>12</th>
<th>12</th>
<th>Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>±10</td>
<td>±4</td>
<td>±4</td>
<td>LSB</td>
</tr>
</tbody>
</table>

### FULL SCALE CALIBRATION ERROR

(with fixed 50Ω resistor from REF IN to REF OUT)

<table>
<thead>
<tr>
<th>0.25</th>
<th>0.25</th>
<th>0.25</th>
<th>% of F.S.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tmin to Tmax (Without Initial Adjustment)</td>
<td>±1/4</td>
<td>±2</td>
<td>±2 LSB</td>
</tr>
<tr>
<td>(With Initial Adjustment)</td>
<td>0.25</td>
<td>0.25</td>
<td>0.12</td>
</tr>
</tbody>
</table>

### TEMPERATURE RANGE

-55 to +125°C

### TEMPERATURE COEFFICIENTS (using internal reference)

Guaranteed max change

<table>
<thead>
<tr>
<th>tmin to Tmax</th>
<th>±2</th>
<th>±1</th>
<th>±1</th>
<th>LSB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unipolar Offset</td>
<td>(5)</td>
<td>(2.5)</td>
<td>(2.5)</td>
<td>ppm/°C</td>
</tr>
<tr>
<td>Bipolar Offset</td>
<td>±4</td>
<td>±2</td>
<td>±1</td>
<td>LSB</td>
</tr>
<tr>
<td>(10)</td>
<td>(5)</td>
<td>(2.5)</td>
<td>(ppm/°C)</td>
<td></td>
</tr>
<tr>
<td>Full Scale Calibration</td>
<td>±20</td>
<td>±10</td>
<td>±5</td>
<td>LSB</td>
</tr>
<tr>
<td>(50)</td>
<td>(25)</td>
<td>(12.5)</td>
<td>(ppm/°C)</td>
<td></td>
</tr>
</tbody>
</table>

### POWER SUPPLY REJECTION

Max change in Full Scale Calibration

| ±1.35V < VCC < ±16.5V or ±11.4V < VCC < ±12.6V | ±2 | ±1 | ±1 | LSB |
| +4.5V < VLOGIC < 5.5V | ±1/2 | ±1/2 | ±1/2 | LSB |
| -16.35V < VEE < -13.5V or -12.6V < VEE < -11.4V | ±2 | ±1 | ±1 | LSB |

### ANALOG INPUTS

<table>
<thead>
<tr>
<th>Input Ranges</th>
<th>Calibration Offset</th>
<th>Volts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bipolar</td>
<td>-5 to +5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-10 to +10</td>
<td></td>
</tr>
<tr>
<td>Unipolar</td>
<td>0 to +10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>0 to +20</td>
<td></td>
</tr>
</tbody>
</table>

| Input Impedance | 5k (3k min., 7k max) | Ω |
| 10 Volt Span    |                      |   |
| 20 Volt Span    |                      |   |

### POWER SUPPLIES

<table>
<thead>
<tr>
<th>Operating Range</th>
<th>Calibration Offset</th>
<th>Volts</th>
</tr>
</thead>
<tbody>
<tr>
<td>VLOGIC</td>
<td>+4.5 to +5.5</td>
<td></td>
</tr>
<tr>
<td>VCC</td>
<td>+11.4 to +16.5</td>
<td></td>
</tr>
<tr>
<td>VEE</td>
<td>-11.4 to -16.5</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operating Current</th>
<th>Calibration Offset</th>
<th>mA</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILOGIC</td>
<td>30 typ. 40 max</td>
<td></td>
</tr>
<tr>
<td>ICC</td>
<td>2 typ. 5 max</td>
<td></td>
</tr>
<tr>
<td>VEE</td>
<td>18 typ. 30 max</td>
<td></td>
</tr>
</tbody>
</table>

### POWER DISSIPATION

390 typ., 725 max | mW |

### INTERNAL REFERENCE VOLTAGE

<table>
<thead>
<tr>
<th>Output Current (available for external loads)</th>
<th>Calibration Offset</th>
<th>Volts</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.00 ±0.1 (max)</td>
<td>1.5max</td>
<td>mA</td>
</tr>
</tbody>
</table>

### NOTES

1 The reference should be buffered for operation on ±12V supplies.

Specifications subject to change without notice.
DIGITAL CHARACTERISTICS\(^1\) (All grades, \(T_{\text{min}} - T_{\text{max}}\))

<table>
<thead>
<tr>
<th>Logic Inputs ((\text{CE, CS, R, A}))</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voltages</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Logic &quot;1&quot;</td>
<td>(-2.0) V</td>
<td>(+5.5) V</td>
<td></td>
</tr>
<tr>
<td>Logic &quot;0&quot;</td>
<td>(-0.5) V</td>
<td>(+0.8) V</td>
<td></td>
</tr>
<tr>
<td>Current</td>
<td>(-50) (\mu)A</td>
<td>(+50) (\mu)A</td>
<td></td>
</tr>
<tr>
<td>Capacitance</td>
<td>5pF</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Logic Outputs (DB11-DB0, STS)

| Logic "0" | \(+0.4\) \(\mu\)A | \(+1.6\) mA |
| Logic "1" | \(2.4\) \(\mu\)A |     |

Leakage (When in high-Z state)

| \(-40\) \(\mu\)A | \(+40\) \(\mu\)A |

Capacitance

| 5pF |     |     |

\(^1\)Detailed Timing Specifications appear in the Digital Interface Section.

\(^2\)12 \(S\) Input is not TTL-compatible and must be hard-wired to \(\text{I}_{\text{IN}}\) or DIGITAL COMMON.

---

**ABSOLUTE MAXIMUM RATINGS**

(Specifications apply to all grades, except where noted)

- \(V_{\text{CC}}\) to Digital Common: 0 to \(+16.5\) V
- \(V_{\text{EE}}\) to Digital Common: 0 to \(+16.5\) V
- \(V_{\text{LOGIC}}\) to Digital Common: 0 to \(+7\) V
- Analog Common to Digital Common: \(\pm 1\) V
- Control Inputs (CE, CS, A, 12, R) to Digital Common: \(-0.5\) V to \(V_{\text{LOGIC}}\) + \(0.5\) V
- Analog Inputs (REF IN, BIP OFF, 10V\(_{\text{IN}}\)) to Analog Common: \(\pm 16.5\) V
- 20V\(_{\text{IN}}\) to Analog Common: \(\pm 24\) V
- REF OUT: Indefinite short to common
- Power Dissipation: 1000mW
- Lead Temperature, Soldering: 300°C, 10 sec.
- Storage Temperature: \(-65°C\) to \(+150°C\)
- Thermal Resistance, \(\theta_{JA}\): 60°C/W

**AD574A ORDERING GUIDE**

<table>
<thead>
<tr>
<th>Model</th>
<th>Temp. Range</th>
<th>Linearity Error Max ((T_{\text{min}}) to (T_{\text{max}}))</th>
<th>Resolution No Missing Codes Max ((T_{\text{min}}) to (T_{\text{max}}))</th>
<th>Max Full Scale T.C. (ppm/°C)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AD574AJD</td>
<td>0 to +70°C</td>
<td>1 LSB</td>
<td>11 Bits</td>
<td>50.0</td>
</tr>
<tr>
<td>AD574AKD</td>
<td>0 to +70°C</td>
<td>1/2 LSB</td>
<td>12 Bits</td>
<td>27.0</td>
</tr>
<tr>
<td>AD574ALD</td>
<td>0 to +70°C</td>
<td>1/2 LSB</td>
<td>12 Bits</td>
<td>10.0</td>
</tr>
<tr>
<td>AD574ASD</td>
<td>-55°C to +125°C</td>
<td>1 LSB</td>
<td>11 Bits</td>
<td>50.0</td>
</tr>
<tr>
<td>AD574ASD/883B</td>
<td>-55°C to +125°C</td>
<td>1 LSB</td>
<td>11 Bits</td>
<td>50.0</td>
</tr>
<tr>
<td>AD574ATD</td>
<td>-55°C to +125°C</td>
<td>1 LSB</td>
<td>12 Bits</td>
<td>25.0</td>
</tr>
<tr>
<td>AD574ATD/883B</td>
<td>-55°C to +125°C</td>
<td>1 LSB</td>
<td>12 Bits</td>
<td>25.0</td>
</tr>
<tr>
<td>AD574AUD</td>
<td>-55°C to +125°C</td>
<td>1 LSB</td>
<td>12 Bits</td>
<td>12.5</td>
</tr>
<tr>
<td>AD574AUD/883B</td>
<td>-55°C to +125°C</td>
<td>1 LSB</td>
<td>12 Bits</td>
<td>12.5</td>
</tr>
</tbody>
</table>
THE AD574A OFFERS GUARANTEED MAXIMUM LINEARITY ERROR OVER THE FULL OPERATING TEMPERATURE RANGE

DEFINITIONS OF SPECIFICATIONS

LINEARITY ERROR
Linearity error refers to the deviation of each individual code from a line drawn from "zero" through "full scale". The point used as "zero" occurs 1/2:LSB (1.22mV for 10 volt span) before the first code transition (all zeros to only the LSB "on"). "Full scale" is defined as a level 1/2:LSB beyond the last code transition (to all ones). The deviation of a code from the true straight line is measured from the middle of each particular code.

The AD574AK, AL, AT, and AU grades are guaranteed for maximum nonlinearity of ±1/2:LSB. For these grades, this means that an analog value which falls exactly in the center of a given code width will result in the correct digital output code. Values nearer the upper or lower transition of the code width may produce the next upper or lower digital output code. The AD574AJ and AS grades are guaranteed to ±1LSB max error. For these grades, an analog value which falls within a given code width will result in either the correct code for that region or either adjacent one.

Note that the linearity error is not user-adjustable.

DIFFERENTIAL LINEARITY ERROR (NO MISSING CODES)
A specification which guarantees no missing codes requires that every code combination appear in a monotonic increasing sequence as the analog input level is increased. Thus every code must have a finite width. For the AD574AK, AL, AT, and AU grades, which guarantee no missing codes to 12-bit resolution, all 4096 codes must be present over the entire operating temperature ranges. The AD574AJ and AS grades guarantee no missing codes to 11-bit resolution over temperature; this means that all code combinations of the upper 11 bits must be present; in practice very few of the 12-bit codes are missing.

UNIPOLAR OFFSET
The first transition should occur at a level 1/2:LSB above analog common. Unipolar offset is defined as the deviation of the actual transition from that point. This offset can be adjusted as discussed on the following two pages. The unipolar offset temperature coefficient specifies the maximum change of the transition point over temperature, with or without external adjustment.

BIPOLAR OFFSET
Similarly, in the bipolar mode, the major carry transition (0111 1111 1111 to 1000 0000 0000) should occur for an analog value 1/2LSB below analog common. The bipolar offset error and temperature coefficient specify the initial deviation and maximum change in the error over temperature.

QUANTIZATION UNCERTAINTY
Analog-to-digital converters exhibit an inherent quantization uncertainty of ±1/2:LSB. This uncertainty is a fundamental characteristic of the quantization process and cannot be reduced for a converter of given resolution.

LEFT-JUSTIFIED DATA
The data format used in the AD574A is left-justified. This means that the data represents the analog input as a fraction of full-scale, ranging from 0 to ±1/2:LSB. This implies a binary point to the left of the MSB.

FULL SCALE CALIBRATION ERROR
The last transition (from 1111 1111 1111 to 1111 1111 1111) should occur for an analog value 1 1/2LSB below the nominal full scale (9.9963 volts for 10.000 volts full scale). The full scale calibration error is the deviation of the actual level at the last transition from the ideal level. This error, which is typically 0.05 to 0.1% of full scale, can be trimmed out as shown in Figures 3 and 4. The full scale calibration error over temperature is given with and without the initial error trimmed out. The temperature coefficients for each grade indicate the maximum change in the full scale gain from the initial value using the internal 10 volt reference.

TEMPERATURE COEFFICIENTS
The temperature coefficients for full-scale calibration, unipolar offset, and bipolar offset specify the maximum change from the initial (25°C) value to the value at T_{min} or T_{max}.

POWER SUPPLY REJECTION
The standard specifications for the AD574A assume use of ±5.00 and ±15.00 or ±12.00 volt supplies. The only effect of power supply error on the performance of the device will be a small change in the full scale calibration. This will result in a linear change in all lower order codes. The specifications show the maximum change in calibration from the initial value with the supplies at the various limits.

CODE WIDTH
A fundamental quantity for A/D converter specifications is the code width. This is defined as the range of analog input values for which a given digital output code will occur. The nominal value of a code width is equivalent to 1 least significant bit (LSB) of the full scale range or 2.44mV out of 10 volts for a 12-bit ADC.
CIRCUIT OPERATION

The AD574A is a complete 12-bit A/D converter which requires no external components to provide the complete successive-approximation analog-to-digital conversion function. A block diagram of the AD574A is shown in Figure 1. The device consists of two chips, one containing the precision 12-bit DAC with voltage reference, the other containing the comparator, successive-approximation register, clock, output buffers and control circuitry.

When the control section is commanded to initiate a conversion (as described later), it then enables the clock and resets the successive-approximation register (SAR) to all zeros. Once a conversion cycle has begun, it cannot be stopped or re-started and data is not available from the output buffers. The SAR, timed by the clock, will then sequence through the conversion cycle and return an end-of-convert flag to the control section. The control section will then disable the clock, bring the output status flag low, and enable control functions to allow data read functions by external command.

During the conversion cycle, the internal 12-bit current output DAC is sequenced by the SAR from the most-significant-bit (MSB) to least-significant-bit (LSB) to provide an output current which accurately balances the input signal current through the 5kΩ (or 10kΩ) input resistor. The comparator determines whether the addition of each successively-weighted bit current causes the DAC current sum to be greater or less than the input current; if the sum is less, the bit is left on; if more, the bit is turned off. After testing all the bits, the SAR contains a 12-bit binary code which accurately represents the input signal to within ±1/2 LSB.

The temperature-compensated buried Zener reference provides the primary voltage reference to the DAC and guarantees excellent stability with both time and temperature. The reference is trimmed to 10.00 volts ± 1%; it can supply up to 1.5mA to an external load in addition to that required to drive the reference input resistor (0.5mA) and bipolar offset resistor (1mA) when the AD574A is powered from ±15 volts supplies. If the AD574A is used with ±12 volts supplies, or if external current must be supplied over the full temperature range, an external buffer amplifier is recommended. Any external load on the AD574A reference must remain constant during conversion. The thin film application resistors are trimmed to match the full scale output current of the DAC. There are two 5kΩ input scaling resistors to allow either a 10 volt or 20 volt span. The 10kΩ bipolar offset resistor is grounded for unipolar operation or connected to the 10 volt reference for bipolar operation.

DRIVING THE AD574A ANALOG INPUT

The AD574A is a successive-approximation type analog-to-digital converter. During the conversion cycle, the ADC input current is modulated by the DAC test current at approximately a 500kHz rate. Thus it is important to recognize that the signal source driving the AD574A must be capable of holding a constant output voltage under dynamically-changing load conditions.

This can be accomplished either by using a wideband op amp or by placing a discrete-transistor or integrated buffer inside the amplifier's feedback loop.

SUPPLY DECOUPLING AND LAYOUT CONSIDERATIONS

It is critically important that the AD574A power supplies be filtered, well-regulated, and free from high frequency noise. Use of noisy supplies will cause unstable output codes to be generated. Switching power supplies are not recommended for circuits attempting to achieve 12-bit accuracy unless great care is used in filtering any switching spikes present in the output. Remember that a few millivolts of noise represents several counts of error in a 12-bit ADC.

Decoupling capacitors should be used on all power supply pins; the +5V supply decoupling capacitor should be connected directly from pin 1 to pin 15 (digital common) and the +VCC and -VEE pins should be decoupled directly to analog common (pin 9). A suitable decoupling capacitor is a 47µF tantalum type in parallel with a 0.1µF disc ceramic type.

Circuit layout should attempt to locate the AD574A, associated analog input circuitry, and interconnections as far as possible from logic circuitry. For this reason, the use of wire-wrap circuit construction is not recommended. Careful printed-circuit construction is preferred.

Figure 1. Block Diagram of AD574A 12-Bit A-to-D Converter

![Figure 1](block_diagram.png)

Figure 2. Op Amp – AD574A Interface

![Figure 2](op_amp_diagram.png)
UNIPOLAR RANGE CONNECTIONS FOR THE AD574A
The AD574A contains all the active components required to perform a complete 12-bit A/D conversion. Thus, for most situations, all that is necessary is connection of the power supplies (-5, -12 -15 and -12 -15 volts), the analog input, and the conversion initiation command, as discussed on the next page. Analog input connections and calibration are easily accomplished; the unipolar operating mode is shown in Figure 1.

Analog input connections and calibration are casually accomplished; zero offset error and ±0.2% (IOLSB) max full scale error.

All of the thin film application resistors of the ADS74A are trimmed for absolute calibration. Therefore, in many applications, no calibration trimming will be required. This absolute accuracy for each grade is given in the specification tables. For example, if no trims are used, the AD574AK guarantees ±2LSB max zero offset error and ±0.25% (10LSB) max full scale error. (Typical full scale error is ±2LSB.) If the offset trim is not required, pin 12 can be connected directly to pin 9; the two resistors and trimmer for pin 12 are then not needed. If the full scale trim is not needed, a 50Ω ±1% metal film resistor should be connected between pin 8 and pin 10.

The analog input is connected between pin 13 and pin 9 for a 0 to +10V input range, between 14 and pin 9 for a 0 to +20V input range. The AD574A easily accommodates an input signal beyond the supplies. For the 10 volt span input, the LSB has a nominal value of 2.44mV, for the 20 volt span, 4.88mV. If a 10.24V range is desired (nominal 2.5mV/bit), the gain trimmer (R2) should be replaced by a 50Ω resistor, and a 200Ω trimmer inserted in series with the analog input to pin 13 (for a full scale range of 20.48V (5mV/bit), use a 500Ω trimmer into pin 14). The gain trim described below is now done with these trimmers. The nominal input impedance into pin 13 is 5kΩ, and 10kΩ into pin 14.

UNIPOLAR CALIBRATION
The AD574A is intended to have a nominal ½LSB offset so that the exact analog input for a given code will be in the middle of that code (halfway between the transitions to the codes above and below it). Thus, when properly calibrated, the first transition (from 0000 0000 0000 to 0000 0000 0001) will occur for an input level of +1/2LSB (1.22mV for 10V range). If pin 12 is connected to pin 9, the unit typically will behave in this manner, within specifications. If the offset trim (R1) is used, it should be trimmed as above, although a different offset can be set for a particular system requirement. This circuit will give approximately ±15mV of offset trim range.

The full scale trim is done by applying a signal 1 1/2LSB below the nominal full scale (9.9963 for a 10V range). Trim R2 to give the last transition (1111 1111 1110 to 1111 1111 1111).

BIPOLAR OPERATION
The connections for bipolar ranges are shown in Figure 4. Again, as for the unipolar ranges, if the offset and gain specifications are sufficient, one or both of the trimmers shown can be replaced by a 50Ω ±1% fixed resistor. The analog input is applied as for the unipolar ranges. Bipolar calibration is similar to unipolar calibration. First, a signal ½LSB above negative full scale (−4.9988V for the ±5V range) is applied and R1 is trimmed to give the first transition (0000 0000 0000 to 0000 0000 0001). Then a signal 1½LSB below positive full scale (+4.9963V for the ±5V range) is applied and R2 trimmed to give the last transition (1111 1111 1110 to 1111 1111 1111).

GROUNDING CONSIDERATIONS
The analog common at pin 9 is the ground reference point for the internal reference and is thus the "high quality" ground for the AD574A; it should be connected directly to the analog reference point of the system. In order to achieve all of the high accuracy performance available from the AD574A in an environment of high digital noise content, it is required that the analog and digital commons be connected together at the package. In some situations, the digital common at pin 15 can be connected to the most convenient ground reference point; analog power return is preferred.
CONVERSION START/DATA READ CONTROL LOGIC

The AD574A contains on-chip logic to provide conversion initiation and data read operations from signals commonly available in microprocessor systems. Figure 5 shows the internal logic circuitry of the AD574A.

TIMING

The AD574A is easily interfaced to a wide variety of microprocessors and other digital systems. Discussion of the timing requirements of the AD574A control signals will provide the system designer with useful insight into the operation of the device.

Figure 6 shows a complete timing diagram for the AD574A convert start operation. R/C should be low before both CE and CS are asserted; if R/C is high, a read operation will momentarily occur, possibly resulting in system bus contention. Either CE or CS may be used to initiate a conversion. As shown in Figure 6,
times. This speed improvement simplifies the interface to faster microprocessors. During data read operations, access time is measured from the point where CE and R杞 both are high, assuming CS is already low. If CS is used to enable the device, access time is extended by 100ns.

In the 8-bit bus interface mode, the input wired to DIGITAL COMMON, the address bit, A10, must be stable at least 150ns prior to CE going high and must remain stable during the entire read cycle. If A10 is allowed to change, damage to the AD574A output buffers may result.

**READ TIMING – FULL CONTROL MODE**

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Parameter</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>TA2</td>
<td>Access Time from CE</td>
<td>210</td>
<td>250</td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>TD2</td>
<td>Data Valid after CE Low</td>
<td>25</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>TD3</td>
<td>Output Float Delay</td>
<td>110</td>
<td>150</td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>TCRE</td>
<td>R杞 to CE Setup</td>
<td>150</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>TCSR</td>
<td>A10 to CE Setup</td>
<td>150</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>TCSR</td>
<td>CS Valid After CE Low</td>
<td>50</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>TCSR</td>
<td>R杞 High After CE Low</td>
<td>0</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
<tr>
<td>TCSR</td>
<td>A10 Valid After CE low</td>
<td>50</td>
<td></td>
<td></td>
<td>ns</td>
</tr>
</tbody>
</table>

1. TA2 is measured with the load circuit of Figure 8 and defined as the time required for an output to cross 0.4V or 2.4V.

2. TD3 is defined as the time required for the data lines to change 0.5V when loaded with the circuit of Figure 9.

**STAND-ALONE OPERATION**

The AD574A can be used in a “stand-alone” mode, which is useful in systems with dedicated input ports available and thus not requiring full bus interface capability.

In this mode, CE and 128 are wired high, CS and A10 are wired low, and conversion is controlled by R杞. The three-state buffers are enabled when R杞 is high and a conversion starts when R杞 goes low. This gives rise to two possible control signals—a high pulse or a low pulse. Operation with a low pulse is shown in Figure 10. In this case, the outputs are forced into the high-impedance state in response to the falling edge of R杞 and return to valid logic levels after the conversion cycle is completed. The STS line goes high 500ns after R杞 goes low and returns low 300ns after data is valid.

If conversion is initiated by a high pulse as shown in Figure 11, the data lines are enabled during the time when R杞 is high. The falling edge of R杞 starts the next conversion and the data lines return to three-state (and remain three-state) until the next high pulse of R杞.

**INTERFACING THE AD574A TO MICROPROCESSORS**

The control logic of the AD574A makes direct connection to most microprocessor system buses possible. While it is impossible to describe the details of the interface connections for every microprocessor type, several representative examples will be described here.

**GENERAL A/D CONVERTER INTERFACE CONSIDERATIONS**

Analog-to-digital converters, like any I/O device, may be interfaced to microprocessors by several methods. These methods include (but are not limited to) direct memory access, isolated or accumulator I/O, and memory-mapped I/O. Direct memory access (DMA) is the fastest, since conversions occur automatically and data updates into memory are transparent to the processor. DMA logic is very processor-dependent and makes use of dedicated specialized hardware.

Memory-mapped and accumulator I/O are more often used and somewhat easier to understand. Memory-mapped I/O assigns the I/O device to one or more locations in the memory space of the microprocessor. This technique has the advantage that the full range of memory reference instructions may be used to operate on the data. The potential disadvantages include limiting the memory space available for program and data memory, somewhat more complex address decoding and more difficult isolation of device select pulses for system debugging. Many processors offer only memory-mapped I/O.

Accumulator I/O uses a set of control signals which are distinct and different from the memory control signals. These control signals, combined with the address bus, serve to define a totally
A typical A/D converter interface routine involves several operations. First, a write to the ADC address initiates a conversion. The processor must then wait for the conversion cycle to complete, since most integrated circuit ADCs take longer than one instruction cycle to complete a conversion. Valid data can, of course, only be read after the conversion is complete. The AD574A provides an output signal (STS) which indicates when a conversion is in progress. This signal can be polled by the processor by reading it through an external three-state buffer (or other input port). The STS signal can also be used to generate an interrupt upon completion of conversion, if the system timing requirements are critical (bear in mind that the maximum conversion time of the AD574A is only 35 microseconds) and the processor has other tasks to perform during the ADC conversion cycle. Another possible time-out method is to assume that the ADC will take 35 microseconds to convert, and insert a sufficient number of “do-nothing” instructions to ensure that 35 microseconds of processor time is consumed.

Once it is established that the converter is done with its cycle, the data can be read. In the case of an ADC of 8-bit resolution (or less), a single data read operation is sufficient. In the case of converters with more data bits than are available on the bus, a choice of data formats is required, and multiple read operations are needed. The AD574A includes internal logic to permit direct interface to 8-bit or 16-bit data buses, selected by connection of the 12/8 input. In 16-bit bus applications (12/8 high) the data lines (DB11 through DB0) may be connected to either the 12 most significant or 12 least significant bits of the data bus. The remaining four bits should be masked in software. The interface to an 8-bit data bus (12/8 low) is done in a left-justified format. The even address (A0 low) contains the 8 MSBs (DB11 through DB4). The odd address (A0 high) contains the 4 LSBs (DB3) through DBO) in the upper half of the byte, followed by four trailing zeroes, thus eliminating bit masking instructions.

This control structure is directly compatible with the AD574A. The R/W line can be used for R/C, the active-low decoded base address (the AD574A occupies two memory locations) is applied to CS, and A2 is used for CE. The least-significant address line ties to the AD574A A0 input.

In this interface, the processor can write to one address (A0 low) to start a full 12-bit conversion or another address (A0 high) to start a short 8-bit conversion. The contents of the data bus are meaningless during these writes. After sufficient time has passed for the conversion to complete, the processor can read the data in the two memory locations occupied by the AD574A. The even location (A0 low) contains the eight MSBs and the odd location contains the four LSBs and four trailing zeroes.

The AD574A may be used directly with 6800 series processors running at clock speeds up to 1.5 MHz.

This control structure is directly compatible with the AD574A. The R/W line can be used for R/C, the active-low decoded base address (the AD574A occupies two memory locations) is applied to CS, and A2 is used for CE. The least-significant address line ties to the AD574A A0 input.

In this interface, the processor can write to one address (A0 low) to start a full 12-bit conversion or another address (A0 high) to start a short 8-bit conversion. The contents of the data bus are meaningless during these writes. After sufficient time has passed for the conversion to complete, the processor can read the data in the two memory locations occupied by the AD574A. The even location (A0 low) contains the eight MSBs and the odd location contains the four LSBs and four trailing zeroes.

The AD574A may be used directly with 6800 series processors running at clock speeds up to 1.5 MHz.

The AD574A may be used directly with 6800 series processors running at clock speeds up to 1.5 MHz.

The 8085A microprocessor uses a multiplexed address/data bus. At the beginning of a machine cycle, this bus contains the low byte of the address being exercised. The ALE output signal is available to strobe a latch to hold the low address byte. For the rest of the machine cycle, this bus carries data to or from the CPU.

The 8085A can use either accumulator I/O or memory-mapping for I/O devices. The system RD and WR are gated with 10:0 to provide distinct I/O read and write signals and memory read and write signals. The control signals required for the AD574A are easily derived from the 8085A control bus. CS is taken from an address decoder on the high-order address bits. R/C can be taken from WR (either I/O write or memory write), A0 is tied to the LSB of the address bus, and CE is taken from the output of a NAND gate driven from RD and WR. All bus access and float delay requirements are met for direct bus interface for 8085A clock rates up to 3 MHz.
In 8085A systems running at high clock frequencies some external circuitry is required. First, the AD574A delay from CE going low to the data lines going into three-state will cause a bus conflict when the 8085A sends out the low byte of the next instruction address. This conflict will occur if the AD574A data outputs are tied directly to the 8085A bus. In systems where bus transceivers (e.g., 74 LS245, 8286, etc.) are used to separate the address and data lines, the conflict is eliminated. The transceivers are disabled at the end of the read cycle and thus isolate the AD574A from the 8085A bus. Since most systems incorporate such buffers, this does not add to system complexity.

A second consideration when interfacing to higher speed 8085A systems is the width of the convert start pulse. The WR pulse from a 5MHz 8085A is only guaranteed to be 230 nanoseconds wide and is thus not long enough to initiate a conversion. There are two solutions to this problem. One possibility is to use a dual D-type flip-flop connected as shown in Figure 15 to insert a single wait state in read and write operations directed towards the AD574A. Another solution is to substitute the earlier-occurring S1 and S0 outputs from 8085A for RD and WR in the circuit of Figure 14 to generate the required control signals. It is important that bus transceivers be employed if S1 and S0 are used for control signals since these signals remain active longer than RD and WR, enabling the AD574A output buffers in read operations for too long, causing potential bus conflicts.

Figure 15. Wait State Generator for 5MHz 8085A Interface

Z-80 System Interface

The Z-80 series of 8-bit microprocessors, like the 8085A, offers both memory-mapped and accumulator I/O capability. While the 8085A only includes two instructions for accumulator I/O (IN and OUT), the Z-80 I/O instruction set is considerably more extensive.

The control signals available on the Z-80 include MREQ, IORQ, RD, and WR. The RD and WR signals indicate direction of data flow while MREQ and IORQ determine whether the read or write cycle in progress is a memory or I/O cycle. During I/O reads and writes, only 8 address lines are active (as in the 8085A). An interesting feature of the Z-80 is that I/O read and write cycles are automatically extended by one clock cycle (one wait state is inserted) and are thus slower. The Z-80 control signal connections to the AD574A are identical to the 8085A connections.

The AD574A can be interfaced to Z-80 series processors with clock speeds up to 2.5MHz in the memory address space using the MWR and MRD signals. At higher clock rates (4 and 6MHz), the memory write pulse is not wide enough to properly start a conversion. The extra wait state added during I/O write operations will extend this pulse to a suitable width at clock rates up to 6MHz so that accumulator I/O is possible.

INTERFACING THE AD574A TO THE APPLE II COMPUTER

The AD574A can be used to provide a low-cost precision analog input port for the Apple II microcomputer without the need for additional power supplies or extensive digital interface logic. The AD574A can be mounted on a hobby card designed to plug into an Apple II I/O slot.

Hardware

All required supply voltages and control signals are available on the Apple's peripheral connectors. Each connector contains, on pin 41, a DEVICE SELECT output which is active when the address bus holds a hexadecimal address between COn0 and COnF, where n is equal to the slot number plus 8. This signal can be connected to pin 3 (CS) of the AD574A. The 40 clock on pin 40 of the peripheral connector can be used for the AD574A CE input (pin 6). The AD574A R/W input (pin 5) can be driven directly by the R/W output available on peripheral connector pin 18. Pin 2 of the peripheral connector, A0, connects directly to the AD574A pin 4. The connections between the peripheral connector and the AD574A are shown in Figure 16.

Figure 16. AD574A Connections to Apple II Peripheral Connector

The Apple II represents a relatively hostile electrical environment to the AD574A. The high frequency clocks radiate a large amount of noise which can be inadvertently coupled into analog signal lines. Furthermore, the switching power supply in the Apple is noisy, and this noise will often pollute the analog signals. It is possible, however, by judicious bypassing, decoupling, and ground management, to achieve a data acquisition system with only occasional flicker. A suggested grounding and decoupling scheme is shown in Figure 17.

It is recommended that any signal preamplification used in such a system be physically located outside the Apple cabinet. A full-scale signal range is less susceptible to electromagnetically coupled interference than a smaller signal range would be. Thus, the preferred method is to deliver a buffered, high-level signal to the AD574A through a shielded cable. The ±5V or ±10V
range is suggested. Full-scale and offset trims, if desired, are performed as shown on page 7.

When data is read, it can be read only 8 bits at a time, as explained on page 10. The sample subroutine below starting at location $4000 performs the control for the AD574A and returns the result in RAM locations $0300 and $0301.

4000 A9 02 LDA #02
4002 8D A0 C0 STA $C0A0
4005 20 A8 FC JSR $FCA8
4008 AD A0 C0 LDA $C0A0
400B 8D 00 03 STA $0300
400E AD A1 C0 LDA $C0A1
4012 8D 01 03 STA $0301
4015 60 RTS

Software
In this discussion, the AD574A is assumed to be located in I/O slot 2 of the Apple II and be the only device in that slot. The AD574A thus occupies the sixteen locations from $C0A0 through $C0AF, even though only two locations are actually required.

It is possible to operate the AD574A from either machine language or a high-level language. In machine language, the converter is started by writing data to either $C0A0 or $C0A1, using a STA instruction. Writing to $C0A0 will start a full 12-bit conversion cycle; writing to $C0A1 starts an 8-bit cycle. Accumulator contents are unimportant during convert start operations. It is then necessary to wait for the AD574A to finish converting before attempting to read the data. This can be accomplished by loading the accumulator with the value 02 and calling the WAIT subroutine located at $FCA8 in the Apple Monitor.

Programs written in Applesoft Basic can also operate the AD574A. Conversion is started by POKEing into location 49312 decimal for a 12-bit conversion (or location 49313 for an 8-bit conversion). Basic executes slowly enough that no delay routines are needed. The output of the AD574A is read by PEEKing into those locations. In order to compute the actual analog voltage, it is necessary to establish the proper weighting for the two bytes read. The Basic subroutine shown in Figure 19 will accomplish this arithmetic. This routine assumes a ±5V analog signal range and returns the value of actual analog signal voltage in the variable V.

100 POKE (49312)
110 A = PEEK (49312): B = PEEK (49313): C = 256
120 A = (A + B/C)/C
130 V = A*10 - 5
140 RETURN

Figure 19. Applesoft II Basic Subroutine for AD574A Control
The T/H amplifier slew rate determines the maximum frequency tracking rate and part of the settling time when sampling pulses and square waves. The feedthrough from input to output while in the hold mode should be less than 1LSB. The amplitude of 1LSB of the companion A/D converter for a given input range will vary from 60mV for a 14-bit A/D using a ±10V input range to 4.8mV for a 12-bit A/D using a ±20V input range. The hold mode droop rate should produce less than 1LSB of droop in the output during the conversion time of the A/D converter. For 60mV/LSB, as noted in the example above, for a 50mV 14-bit A/D converter, the maximum droop rate will be 60mV/50μs or 12μV/μs during the 50μs conversion period.

The linearity error should be less than 1LSB over the transfer function, as set by the resolution of the A/D converter. The T/H acquisition time, T/H settling time along with the conversion time of the A/D converter determines the highest sampling rate. This in turn will determine the highest input signal frequency that can be sampled at twice a cycle per the Nyquist criteria. The pedestal shift due to input signal changes should either be linear, to be seen as a gain error, or negligible as with the feedthrough spec. The temperature coefficients for drift should be low enough such that full accuracy is maintained over some minimum temperature range. The droop rate and pedestal will shift more over temperature about 1.7°C (±15°F). For medical and industrial users, these shifts will only appear above the highest temperatures their equipment will ever expect to experience. Most precision instrumentation is installed only in human inhabitable work spaces or in controlled enclosures if the area has a hostile environment.

Minimal thermal tail effects are another requirement of high resolution applications. The self-heating errors induced by the changing current levels in the output stages of T/H amps may cause more than 1LSB of error due to thermal tail effects. The performance of a typical AD389 in contrast to a typical 12-bit T/H circuit is shown in Figures 11a and 11b. The test circuit is shown in Figure 8.

OFFSET ADJUST TRIM

In most data acquisition systems only one offset adjustment is made. In many cases it is the offset adjust of the M/C that is used to cancel all other accumulated system offsets. The offset or pedestal of the AD389 can be nulled by means of Skil potentiometer between pins 7, 9, and 11. If the offset of the AD389 is not adjusted, then connect pins 7 and 9 to pin 14, the negative supply. Otherwise the high impedance of the null pin together with parasitic capacitances can cause tail effects.

T/H REQUIREMENTS FOR HIGH RESOLUTION APPLICATIONS

The characteristics required for high resolution track-and-hold amplifiers are low feedthrough, low pedestal shifts with changes of input signal or temperature, high linearity, low temperature coefficients, and minimal droop rate.

For sampling a 20kHz signal to 14 bit and 16 bits for example, the following specs are required:

<table>
<thead>
<tr>
<th>Spec</th>
<th>14 Bit</th>
<th>16 Bit</th>
<th>AD389KD Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aperture Jitter (max)</td>
<td>2.4</td>
<td>0.6</td>
<td>0.4 μs</td>
</tr>
<tr>
<td>Feedthrough (1.5kHz)</td>
<td>86.3</td>
<td>96.3</td>
<td>86 dB</td>
</tr>
<tr>
<td>Dropout (1.5kHz max)</td>
<td>0.2</td>
<td>0.1</td>
<td>μV/μs</td>
</tr>
<tr>
<td>Dropout (15kHz max)</td>
<td>0.2</td>
<td>0.1</td>
<td>μV/μs</td>
</tr>
<tr>
<td>Acquisition Time (max)</td>
<td>10</td>
<td>10</td>
<td>5 μs</td>
</tr>
<tr>
<td>Feedthrough (1kHz)</td>
<td>84.1</td>
<td>96.3</td>
<td>86 dB</td>
</tr>
<tr>
<td>Gain Temperature Coefficient (max)</td>
<td>6.1</td>
<td>1.5</td>
<td>2.0 μV/°C</td>
</tr>
<tr>
<td>Thermal Tail Error within 50μA offset Hold</td>
<td>±0.0004</td>
<td>±0.0015</td>
<td>±0.003 NFSR</td>
</tr>
</tbody>
</table>

Table II. T/H Amplifier Requirements vs. AD389 Specs

Aperture Jitter will affect exactly when the switch closes, even though the T/H control line is driven by a very precise clock. All high speed sampled data systems are very dependent on low aperture jitter for digitizing high frequency signals for spectrum analysis and accurate signal reconstruction.

PRODUCT DESCRIPTION

The AD582 is a low cost integrated circuit sample and hold amplifier consisting of a high performance operational amplifier, a low leakage analog switch and a JFET integrating amplifier— all fabricated on a single monolithic chip. An external holding capacitor, connected to the device, completes the sample and hold function.

With the analog switch closed, the AD582 functions like a standard op amp; any feedback network may be connected around the device to control gain and frequency response. With the switch open, the capacitor holds the output at its last level, regardless of input voltage.

Typical applications for the AD582 include sampled data systems, D/A deglitchers, analog de-multiplexers, auto null systems, strobed measurement systems and A/D speed enhancement.

The device is available in two versions: the "K" specified for operation over the 0 to +70°C commercial temperature range and the "S" specified for the extended temperature range, -55°C to +125°C. All versions may be obtained in either the hermetic sealed, TO-100 can or the TO-116 DIP.

PRODUCT HIGHLIGHTS

1. The specially designed input stage presents a high impedance to the signal source in both sample and hold modes (up to ±12V). Even with signal levels up to ±1V, no undesirable signal inversion, peaking or loss of hold voltage occurs.

2. The AD582 may be connected in any standard op amp configuration to control gain or frequency response and provide signal inversion, etc.

3. The AD582 offers a high, sample-to-hold current ratio: 107. The ratio of the available charging current to the holding leakage current is often used as a figure of merit for a sample and hold circuit.

4. The AD582 has a typical charge transfer less than 2pC. A low charge transfer produces less offset error and permits the use of smaller hold capacitors for faster signal acquisition.

5. The AD582 provides separate analog and digital grounds, thus improving the device’s immunity to ground and switching transients.
**SPECIFICATIONS**

(typical @ +25°C, V_{GS} = ±15V and C_{H} = 1000pF, A = +1 unless otherwise specified)

<table>
<thead>
<tr>
<th>MODEL</th>
<th>AD582K</th>
<th>AD582S</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAMPLE/HOLD CHARACTERISTICS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Acquisition Time, 10V Step to ±0.1%</td>
<td>6ns</td>
<td></td>
</tr>
<tr>
<td>Acquisition Time, 10V Step to ±0.01%</td>
<td>25ns</td>
<td></td>
</tr>
<tr>
<td>Acquisition Time, 10V to -10V</td>
<td>250ns</td>
<td></td>
</tr>
<tr>
<td>Aperature Time, 20V p to Input, Hold 10</td>
<td>200ns</td>
<td></td>
</tr>
<tr>
<td>Aperature Jitter, 20V p Input, Hold 10</td>
<td>10ns</td>
<td></td>
</tr>
<tr>
<td>Settling Time, 20V p Input, Hold 10</td>
<td>0.5ns</td>
<td></td>
</tr>
<tr>
<td>Dropout Current, Settling Stage, 10VDC</td>
<td>100nA max</td>
<td></td>
</tr>
<tr>
<td>Dropout Current, T_{A} = 25°C</td>
<td>1μA</td>
<td>150nA max</td>
</tr>
<tr>
<td>Charge Transfer</td>
<td>5pC (μA, 1μA typ)</td>
<td>5pC (μA, 1μA typ)</td>
</tr>
<tr>
<td>Sample to Hold Offset</td>
<td>0.5mV</td>
<td></td>
</tr>
<tr>
<td>Feedthrough Capacitance</td>
<td>20pF p</td>
<td>100pF Input</td>
</tr>
</tbody>
</table>

**TRANSFER CHARACTERISTICS**

| Vpp | 25ns (50k typ) |
| Vpp | 600ns (1MHz typ) |
| | 1.5MHz |
| Vpp | 70kHz |
| Vpp | 100pF p, C_{H} = 100pF |
| Vpp | 10nA |
| | 12Ω |
| Vpp | 10V p, p, R_{L} = 2k |
| | 150mA |

**ANALOG INPUT CHARACTERISTICS**

| Offset Voltage | 4mV max (10k typ) |
| OffsVoltage | 4mV max (1M typ) |
| Bias Current | 50μA max (1kA typ) |
| | 100μA max (10kA typ) |
| | 1nA |
| | 2μA |
| | 30mA |
| | 10V |
| | 5V |

**DIGITAL INPUT CHARACTERISTICS**

| Logic Input Voltage | +25V |
| Hold Mode | T_{H,Hold} = T_{H,Logic} - 0.8V max |
| Sample Mode | T_{H,Sample} = T_{H,Logic} - 0.8V max |
| | 5μA |
| | 1nA |
| | 2μA |
| | 1.2V |
| | 10V |
| | 5V |

**POWER SUPPLY CHARACTERISTICS**

| Operating Voltage Range | 15V to 20V |
| Supply Current | 4.5mA max (5mA typ) |
| Power Supply Rejection | 60mV (±5V) |

**TEMPERATURE RANGE**

| Specified Performance | 0°C to +70°C |
| Operating | -35°C to +125°C |
| Storage | -40°C to +125°C |
| Lead Temperature (soldering 15sec) | +200°C |

**PACKAGE OPTIONS**

- "S" Package: TO 100
- "D" Package: TO 116 (SSP)

**NOTES**

* Specifications were at 10MHz.
* No section 10 for package outline information.

---

**APPLYING THE AD582**

Both the inverting and non-inverting inputs are brought out to allow op amp type versatility in connecting and using the AD582. Figure 1 shows the basic non-inverting unity gain connection requiring only an external hold capacitor and the usual power supply bypass capacitors. An offset null pot can be added for more critical applications.

**Figure 1. Sample and Hold with A = +1**

Applying the AD582

The hold capacitor, C_{H}, should be a high quality polystyrene (for temperatures below +85°C) or Teflon type with low dielectric absorption. For high speed, limited accuracy applications, capacitors as small as 100pF may be used. Larger values are required for accuracies of 12 bits and above in order to minimize feedthrough, sample to hold offset and dropout errors (see Figure 6). Care should be taken in the circuit layout to minimize coupling between the hold capacitor and the digital or signal inputs.

In the hold mode, the output voltage will follow any change in the V_{GS} supply. Consequently, this supply should be well regulated and filtered.

Raising the +5V Logic input anywhere between -5V to +0.8V with respect to the -5V Logic will set the sample mode. The hold mode will result from any bias between +2.0V and +5V (V_{GS} = 3V). The sample and hold modes will be controlled differentially with the absolute voltage at either logic input ranging from -V_{GS} to within 3V of +V_{GS} (V_{GS} = 3V). Figure 3 illustrates some examples of the flexibility of this feature.

**Figure 3A. Standard Logic Connection**

**Figure 3B. Inverted Logic Sense Connection**

**Figure 3C. High Threshold Logic Connection**

---

**SAMPLE/TRACK HOLD AMPLIFIERS**

VOL. 14-24 SAMPLE/TRACK HOLD AMPLIFIERS
Figure 4. Pictorial Showing Various S/H Characteristics

Aperture Time is the time required after the "hold" command until the switch is truly open and produces a delay in the effective sample timing. Figure 5 is a plot giving the maximum frequency at which the AD582 can sample an input with a given accuracy (lower curve).

Aperture Jitter is the uncertainty in Aperture Time. If the Aperture Time is "tuned out" by advancing the sample-to-hold command 200ns with respect to the input signal, the Aperture Jitter now determines the maximum sampling frequency (upper curve of Figure 5).

Acquisition Time is the time required by the device to reach its final value within a given error band after the sample command has been given. This includes switch delay time, slew time and settling time for a given output voltage change.

Drop is the change in the output voltage from the "hold" value as a result of device leakage. In the AD582, drop can be in either the positive or negative direction. Drop rate may be calculated from drop current using the following formula:

\[ \text{Drop} \times \text{Vin} = \frac{\Delta V}{\Delta t} (\text{Volts/sec}) \times \text{Drop} (\text{mA}) \]

(See also Figure 6.)

Feedthrough is that component of the output which follows the input signal after the switch is open. As a percentage of the input, feedthrough is determined as the ratio of the feedthrough capacitance to the hold capacitance (Cf/H). Charge Transfer is the charge transferred to the holding capacitor from the interelectrode capacitance of the switch when the unit is switched to the hold mode. The charge transfer generates a sample-to-hold offset where:

\[ \text{S/H Offset} (V) = \frac{\text{Charge} (pC)}{C_f(pF)} \]

(See also Figure 6.)

Sample-to-Hold Offset is that component of D.C. offset independent of Cf (see Figure 6). This offset may be nullled using a null pot, however, the offset will then appear during the sampling mode.
III. Peak-to-Peak Signal Handling Capability

The electrical specifications of the IH6108 are guaranteed for ±14V signals, but the specifications have very minor changes for ±14V signals. The notable changes are slightly lower ratios and slightly higher leakages.

FEATURES
- Pin compatible with DG506, HI-506 & AD7506
- Ultra Low Leakage — IDISABLE ≤ 100μA
- ±11 analog signal range
- 700 ohms over full signal and temperature range
- Break-before-make switching
- TTL and CMOS compatible Address control
- Binary Address control (4 Address Inputs control 16 channels)
- Two for submultiplexing to facilitate expandability
- Power supply quiescent current less than 100μA
- No SCR latchup

GENERAL DESCRIPTION

The IH6116 is a CMOS monolithic, one of 16 multiplexer. The part is a plug in replacement for the DG506. Four line binary decoding is used so that the 16 channels can be controlled by 4 Address inputs; additionally a fifth input is provided to use as system enable. When the Enable input is high (VH) the channels are sequenced by the 4 line Address inputs, and when low (VL), all channels are off. The 4 line inputs are controlled by TTL logic or CMOS logic elements with a "0" corresponding to any voltage less than 0.8V and a "1" corresponding to any voltage greater than 3.0V. Note that the Enable input must be taken to 5V to enable the system and less than 0.8V to disable the system.

FUNCTIONAL DIAGRAM

DECODE TRUTH TABLE

<table>
<thead>
<tr>
<th>A3</th>
<th>A2</th>
<th>A1</th>
<th>A0</th>
<th>EN</th>
<th>ON SWITCH</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>HOME</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>8</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>13</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>14</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>15</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>16</td>
</tr>
</tbody>
</table>

Logic "1" = Vdd ≥ 3.0V  VDD ≥ 4.5V
Logic "0" = Vss ≤ 0.8V

PIN CONFIGURATION

ORDERING INFORMATION

<table>
<thead>
<tr>
<th>PART NUMBER</th>
<th>TEMPERATURE RANGE</th>
<th>PACKAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>IH6116MJI</td>
<td>-55°C to +125°C</td>
<td>28 pin CERDIP</td>
</tr>
<tr>
<td>IH6116GJI</td>
<td>0°C to 70°C</td>
<td>28 pin CERDIP</td>
</tr>
<tr>
<td>IH6116CFI</td>
<td>0°C to 70°C</td>
<td>28 pin Plastic DIP</td>
</tr>
</tbody>
</table>
### IH6116

**ABSOLUTE MAXIMUM RATINGS**

- V\(_\text{in}, \text{EN} \to \text{Ground} \) \(-15\) V to 15 V
- V\(_\text{S}, \text{Vs} \to \text{Vs} \) \(-0.32\) V
- V\(_\text{S}, \text{Vss} \to \text{Vss} \) \(-0.32\) V
- V\(_\text{S}, \text{Vss} \to \text{Ground} \) \(-16\) V
- V\(_\text{S}, \text{Ground} \) \(-16\) V
- Current (Any Terminal) \(30\) mA

*Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions above those contained in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

**ELECTRICAL CHARACTERISTICS**

<table>
<thead>
<tr>
<th>CHARACTERISTIC</th>
<th>MEASURED TERMINAL</th>
<th>TESTS</th>
<th>TYPE</th>
<th>M SUFFIX</th>
<th>C SUFFIX</th>
<th>LIMITS</th>
<th>UNIT</th>
</tr>
</thead>
<tbody>
<tr>
<td>DC Input Test</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>V(_\text{S})</td>
<td>0.1 V</td>
<td>0.5 V</td>
<td>0.2</td>
<td>0.5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>V(_\text{S})</td>
<td>0.1 V</td>
<td>0.5 V</td>
<td>0.2</td>
<td>0.5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>V(_\text{S})</td>
<td>0.1 V</td>
<td>0.5 V</td>
<td>0.2</td>
<td>0.5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>V(_\text{S})</td>
<td>0.1 V</td>
<td>0.5 V</td>
<td>0.2</td>
<td>0.5</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**IH6116 APPLICATIONS**

1. Out of 32 channel multiplexer using 2 IH6116s

**DECODE TRUTH TABLE**

<table>
<thead>
<tr>
<th>(A_0)</th>
<th>(A_1)</th>
<th>(A_2)</th>
<th>(A_3)</th>
<th>(A_4)</th>
<th>(ON) SWITCH</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>S1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>S2</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>S3</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>S4</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>S5</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>S6</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>S7</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>S8</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>S9</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>S10</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>S11</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>S12</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>S13</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>S14</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>S15</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>S16</td>
</tr>
</tbody>
</table>

**Figure 1**

*3-100*

**Figure 2**

**Figure 3**

**Figure 4**

*3-101*
IH6116

IH6116 APPLICATIONS (Continued)

II. 1 out of 32 channel multiplexer using 2 IH6116's; using an IH5041 for submultiplexing

* TTL gate must have
  pullup resistor to +5V to drive EN inputs

<table>
<thead>
<tr>
<th>DECODE TRUTH TABLE</th>
<th>ON SWITCH</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0 0 0 0 0 S1</td>
<td></td>
</tr>
<tr>
<td>0 0 0 0 0 0 1 S2</td>
<td></td>
</tr>
<tr>
<td>0 0 0 0 1 0 S3</td>
<td></td>
</tr>
<tr>
<td>0 0 0 1 0 0 S4</td>
<td></td>
</tr>
<tr>
<td>0 0 1 0 0 0 S5</td>
<td></td>
</tr>
<tr>
<td>0 1 0 0 0 0 S6</td>
<td></td>
</tr>
<tr>
<td>0 0 1 1 1 1 S7</td>
<td></td>
</tr>
<tr>
<td>0 1 1 1 1 1 S8</td>
<td></td>
</tr>
<tr>
<td>0 0 0 0 1 1 S9</td>
<td></td>
</tr>
<tr>
<td>0 0 0 1 1 1 S10</td>
<td></td>
</tr>
<tr>
<td>0 0 1 0 1 1 S11</td>
<td></td>
</tr>
<tr>
<td>0 1 0 1 1 1 S12</td>
<td></td>
</tr>
<tr>
<td>0 1 0 0 1 1 S13</td>
<td></td>
</tr>
<tr>
<td>0 1 1 1 1 1 S14</td>
<td></td>
</tr>
<tr>
<td>0 1 1 1 1 1 S15</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DECODE TRUTH TABLE</th>
<th>ON SWITCH</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 0 0 0 0 0 0 S17</td>
<td></td>
</tr>
<tr>
<td>1 0 0 0 1 0 S18</td>
<td></td>
</tr>
<tr>
<td>1 0 0 1 1 1 S19</td>
<td></td>
</tr>
<tr>
<td>1 0 1 0 0 0 S20</td>
<td></td>
</tr>
<tr>
<td>1 0 1 0 1 0 S21</td>
<td></td>
</tr>
<tr>
<td>1 0 1 1 0 0 S22</td>
<td></td>
</tr>
<tr>
<td>1 1 0 0 1 0 S23</td>
<td></td>
</tr>
<tr>
<td>1 1 0 1 0 0 S24</td>
<td></td>
</tr>
<tr>
<td>1 1 0 0 1 0 S25</td>
<td></td>
</tr>
<tr>
<td>1 1 1 0 0 0 S26</td>
<td></td>
</tr>
<tr>
<td>1 1 1 0 1 1 S27</td>
<td></td>
</tr>
<tr>
<td>1 1 1 1 0 1 S28</td>
<td></td>
</tr>
<tr>
<td>1 1 1 1 1 0 S29</td>
<td></td>
</tr>
<tr>
<td>1 1 1 1 1 1 S30</td>
<td></td>
</tr>
<tr>
<td>1 1 1 1 0 1 S31</td>
<td></td>
</tr>
<tr>
<td>1 1 1 1 1 0 S32</td>
<td></td>
</tr>
</tbody>
</table>

Figure 5

IV. General note on expandability of IH8116

The IH8116 is a two tier multiplexer, where sixteen input channels are routed to a common output in blocks of 4. Each block of 4 input channels is routed to one common output channel, and thus the submultiplexed system looks like 4 blocks of 4 inputs routed to 4 different outputs with the 4 outputs tied together. Thus 20 switches are needed to handle the 16 channels of information. The advantage of this is lower output capacity and leakage that would be possible using a system with all 16 channels tied to one common output. Also the expandability into 32, 64, 128, etc. is facilitated. Figures 4, 5, and 6 show how the IH8116 is expanded.
IH6116

Figure 4 shows a 1 of 32 multiplexer, using 2 IH6116s. Since the 6116 is itself a 2 tier MUX, the system as shown is basically a 2 tier system. The four output channels of each 6116 are tied together so that 8 channels are tied to the Vout common point. Since only one channel of information is on at a time, the common output will consist of 7 OFF channels and 1 ON channel. Thus the output leakage will correspond to 7 ION(ON) and 1 ION(0), or about 1.0 nA of typical leakage at room temperature. Thruput speed will be typically 0.8µs for ION and 0.5µs for I0N. Thruput channel resistance will be in the 500Ω area.

Figure 5 shows the 1 of 32 MUX of Figure 4, with a third tier of submultiplexing added to further reduce leakage and output capacity. The IH5041 has typical ON resistances of 500Ω max. in 75Ω so it only increases thruput channel resistance from the 500 ohm of Figure 4 to about 550 ohms for Figure 5. Thruput channel speed is a little slower by about 0.5µs for both ON and OFF time, and output leakage is about 0.2 nA.

Figure 6 shows a 1 of 64 MUX using 3 tier MUXing similar to Figure 5. The Intersil IH5053 is used to get the third tier of MUXing. The Vout point will see 3 OFF channels and 1 ON channel at any one time, so that the typical leakages will be about 0.4 nA. Thruput channel resistance will be in the 550 ohm area with thruput switching speeds about 1.3µs for ON time and 0.8µs for OFF time. The IH5053 was chosen as the third tier of the MUX because it will switch the same AC signals as the IH6116 typically plus and minus 15V and uses break before make switching. Also power supply quiescent currents are on the order of 1-2µA so that no excessive system power is generated. Note that the logic of the 5053 is such that it can be tied directly to the ENABLE input (as shown in the figures) with no extra logic being required.

V. Enable input strobing levels

The enable input (EN) acts as an enabling or disabling pin for the IH6116 when used as a 16 channel MUX. However, when expanding the MUX to more than 16 channels, the EN pin acts as another address input. Figures 4 and 5 show the EN pin used as the A4 input.

For the system to function properly the EN input pin 18 must go to 5V ±15% for the high state and to +0.8V for the low state. When using TTL logic, a pull-up resistor of 1kΩ or less should be used to pull the output voltage up to 5V. When using CMOS logic, the high state goes to the power supply so no pull-up is required. If used on high voltage logic supplies, EN should be at least 0.7V below V+ at all times. See IH6108 data sheet for details.

APPLICATION NOTES

Further information may be found in:
A003 "Understanding and Applying the Analog Switch," by Dave Fullagar
A006 "A New CMOS Analog Gate Technology," by Dave Fullagar
A20A "A Cookbook Approach to High Speed Data Acquisition and Microprocessor Interfacing," by Ed Slegier
P009 "Reduce CMOS Multiplexer Troubles Through Proper Device Selection," by Dick Wilkenen

NOTE: This multiplexer does not require external resistors and/or diodes to eliminate what is commonly known as a latch up or SCR action. Because of this fact, the ION(ON) of the switch is maintained at specified values.

---

IH6201

Dual CMOS Driver/Voltage Translator

When used in conjunction with the Intersil IH401 family of Varfets, the combination makes a complete solid state switch capable of switching signals up to 22Vpp and up to 20MHz in frequency. This switch is a "break-before-make" type i.e. for time < t(KN), the combination has typical tON ≈ 80ns and typ. tOFF ≈ 200ns for signals up to 20Vpp in amplitude.

A TTL "1" input strobe will force the "0" driver output up to V+ level, the "0" output will be driven down to the "1" level. When the TTL input goes to "0", the "0" output goes to "1", thus "0" and "1" are 180° out of phase with each other. These complementary outputs can be used to create a wide variety of functions such as SPDT and DPDT switches, etc.; alternatively the complementary outputs can be used to drive an N and P channel Mosfet, to make a complete Mosfet analog gate.

The driver typically uses +5V and ±15V power supplies; however a wide range of V+ and V- is possible, however V+ > 5V is necessary for the driver to work properly.
The 96LS488 is a TTL LSI circuit containing all of the logic necessary to interface Talk, Listen and Talk/Listen type instruments and system components in accordance with the IEEE-488 standard for programmable instrumentation. All outputs that drive the IEEE-488 bus are guaranteed to sink 48 mA, and all bus inputs have Schmitt triggers and bus terminating networks. All pins that interface to the instrument logic are LSTTL compatible.

The 96LS488 has programming inputs that determine whether it is to be a talker, listener or both, single or dual address, high or low speed, etc., according to the instrument and system requirements. It operates with a minimum of external support logic and readily interfaces with most microprocessors. It operates from a single 5.0 V supply and a 10 MHz single phase clock and is capable of operating the bus handshake at the full 1 MHz data rate. It offers a variety of handshaking and status connections to the instrument logic for versatility and ease of design.

- SINGLE 5.0 V SUPPLY
- COMPLETE SOURCE AND ACCEPTOR HANDSHAKE LOGIC
- SAME OR SEPARATE TALK AND LISTEN ADDRESS
- SECONDARY ADDRESS CAPABILITY
- TALK ONLY OR LISTEN ONLY CAPABILITY
- SOURCE HANDSHAKE DELAY PROGRAMMABLE FOR LOW OR HIGH SPEED
- SERIAL POLL CAPABILITY
- PARALLEL POLL CAPABILITY
- SYNC TRIGGER AND DEVICE CLEAR OUTPUTS
- IMPLEMENTS REMOTE/LOCAL FUNCTION
- ON-CHIP CLOCK OSCILLATOR
- SERVICE REQUEST INTERRUPT FACILITY
- ALL BUS I/O SIGNALS COMPLY WITH THE IEEE-488 (1980) AND IEC-625-1 INPUT THRESHOLD, TERMINATION AND OUTPUT SPECIFICATIONS
- ALL INSTRUMENT INTERFACE SIGNALS ARE LSTTL COMPATIBLE
- GPIB PINS PRESENT NO ELECTRICAL LOAD WHEN DEVICE IS POWERED OFF
## 96LS488

### Input Loading/Fan-Out

<table>
<thead>
<tr>
<th>Pin Names</th>
<th>Description</th>
<th>96LS (U.L.)** HIGH/LOW</th>
</tr>
</thead>
<tbody>
<tr>
<td>A&lt;sub&gt;1&lt;/sub&gt;–A&lt;sub&gt;5&lt;/sub&gt;</td>
<td>Address Inputs</td>
<td>0.5/0.25</td>
</tr>
<tr>
<td>ATN</td>
<td>Attention Input (Active LOW)</td>
<td>T’/1.4</td>
</tr>
<tr>
<td>CP</td>
<td>Clock Input (Active Falling Edge)</td>
<td>0.5/0.25</td>
</tr>
<tr>
<td>IF/C</td>
<td>Interface Clear Input (Active LOW)</td>
<td>T’/1.4</td>
</tr>
<tr>
<td>IST</td>
<td>Instrument Status Input</td>
<td>0.5/0.25</td>
</tr>
<tr>
<td>M&lt;sub&gt;0&lt;/sub&gt;–M&lt;sub&gt;3&lt;/sub&gt;</td>
<td>Mode Control Inputs</td>
<td>0.5/0.25</td>
</tr>
<tr>
<td>MR</td>
<td>Master Reset Input (Active LOW)</td>
<td>0.5/0.25</td>
</tr>
<tr>
<td>REN</td>
<td>Remote Enable Input (Active LOW)</td>
<td>T’/1.4</td>
</tr>
<tr>
<td>RSV</td>
<td>Request Service Input (Active LOW)</td>
<td>0.5/0.25</td>
</tr>
<tr>
<td>RTL</td>
<td>Return to Local Input (Active LOW)</td>
<td>0.5/0.25</td>
</tr>
<tr>
<td>RXRDY</td>
<td>Receiver Ready Input (Active HIGH)</td>
<td>0.5/0.25</td>
</tr>
<tr>
<td>STRDY</td>
<td>Status Ready Input (Active HIGH)</td>
<td>0.5/0.25</td>
</tr>
<tr>
<td>TXRDY</td>
<td>Transmitter Ready Input (Active HIGH)</td>
<td>0.5/0.25</td>
</tr>
<tr>
<td>DAV</td>
<td>Data Valid Input (Active LOW)</td>
<td>T’/1.4</td>
</tr>
<tr>
<td></td>
<td>as Output (Active LOW)</td>
<td>130/30</td>
</tr>
<tr>
<td>D&lt;sub&gt;O&lt;/sub&gt;&lt;sub&gt;1&lt;/sub&gt;–D&lt;sub&gt;O&lt;/sub&gt;&lt;sub&gt;7&lt;/sub&gt;</td>
<td>Data Inputs</td>
<td>T’/1.4</td>
</tr>
<tr>
<td></td>
<td>as Outputs (Active LOW)</td>
<td>T’/30</td>
</tr>
<tr>
<td>D&lt;sub&gt;O&lt;/sub&gt;&lt;sub&gt;8&lt;/sub&gt;</td>
<td>Data Output (Active LOW)</td>
<td>T’/30</td>
</tr>
<tr>
<td>EOI</td>
<td>End or Identify Input (Active LOW)</td>
<td>T’/1.4</td>
</tr>
<tr>
<td>NDAC</td>
<td>Not Data Accepted Input (Active LOW)</td>
<td>T’/1.4</td>
</tr>
<tr>
<td></td>
<td>as Output (Active LOW)</td>
<td>T’/30</td>
</tr>
<tr>
<td>NRFD</td>
<td>Not Ready for Data Input</td>
<td>T’/1.4</td>
</tr>
<tr>
<td></td>
<td>as Output (Active LOW)</td>
<td>T’/30</td>
</tr>
<tr>
<td>ASEL</td>
<td>Address Select Output</td>
<td>10/5.0 (2.5)</td>
</tr>
<tr>
<td>CLR</td>
<td>Device Clear Output (Active LOW)</td>
<td>10/5.0 (2.5)</td>
</tr>
<tr>
<td>D&lt;sub&gt;S/E&lt;/sub&gt;</td>
<td>Data/Status Output (Active LOW)</td>
<td>10/5.0 (2.5)</td>
</tr>
<tr>
<td></td>
<td>End-of-String Output (Active HIGH)</td>
<td>10/5.0 (2.5)</td>
</tr>
<tr>
<td>DRB</td>
<td>Bus Drive Enable Output (Active LOW)</td>
<td>10/5.0 (2.5)</td>
</tr>
<tr>
<td>LAD</td>
<td>Listen Address Status Output (Active LOW)</td>
<td>10/5.0 (2.5)</td>
</tr>
<tr>
<td>ROS</td>
<td>Requested Service Status Output (Active LOW)</td>
<td>130/30</td>
</tr>
<tr>
<td>RXST</td>
<td>Receiver Strobe Output (Active HIGH)</td>
<td>10/5.0 (2.5)</td>
</tr>
<tr>
<td>RL/L</td>
<td>Remote/Local Output</td>
<td>10/5.0 (2.5)</td>
</tr>
<tr>
<td>SRQ</td>
<td>Service Request Output (Active LOW)</td>
<td>T’/30</td>
</tr>
<tr>
<td>STST</td>
<td>Status Strobe Output (Active HIGH)</td>
<td>10/5.0 (2.5)</td>
</tr>
<tr>
<td>TAD</td>
<td>Talk Address Status Output (Active LOW)</td>
<td>10/5.0 (2.5)</td>
</tr>
<tr>
<td>TXST</td>
<td>Transmitter Strobe Output (Active HIGH)</td>
<td>10/5.0 (2.5)</td>
</tr>
<tr>
<td>TRIG</td>
<td>Device Trigger Output (Active LOW)</td>
<td>10/5.0 (2.5)</td>
</tr>
<tr>
<td>XTAL</td>
<td>Crystal Output</td>
<td>10/5.0 (2.5)</td>
</tr>
</tbody>
</table>

* T = Resistive Termination per IEEE-488 Standard.

**Unit Load (U.L.) definitions

LOW State: 1.6 mA = 1 U.L.
HIGH State: 40 µA = 1 U.L.

Where two sets of output LOW loading factors are shown, the one in parentheses applies over the Military V<sub>CC</sub> and temperature ranges.
GPIB Protocol

A full description and specification of the GPIB system is published in the IEEE document 'IEEE Standard Interface for Programmable Instrumentation' IEEE Std 488-1978 and is used as the reference in this description.

The standard is a 16-wire interface that can transmit byte serial data at rates of up to 1 megabyte/second. Using this standard up to 15 individual devices (instruments or system components) may be interconnected in a star or linear network, with a maximum cable length of 20 m and automatically controlled or programmed. Data may be exchanged between instruments or between a controller and instruments. The use of a bus extender or a controller which can handle a number of separate instrument buses allows more than 15 instruments to be used in a system:

- **Talkers**—These instruments can only transmit data (when addressed), e.g., a timer or counter.
- **Listeners**—These instruments can only receive information, e.g., a programmable power supply or a printer.
- **Talker/Listeners**—These instruments can receive data or functional instructions and later transmit data, e.g., a programmable DVM or multichannel aid converter.
- **Controller**—A device that is able to generate control instructions for instruments on the bus, e.g., a mini-computer or programmable calculator.

The 96LS488 is designed for use in any of the first three types of devices.

The 16-wire bus is organized as 3 functional groups (Figure 1). The 8-line Data bus, DO1-DO7, DO8, is used to transfer commands in bit parallel/byte serial form from Talkers to Listeners. The 3-line Data Byte Transfer Control bus, NRFD, NDAC and DAV, implements a data handshake which ensures that information transfer proceeds as fast as the device will allow but no faster than the slowest device currently addressed as active (Figure 2). The 5-line General Interface Management bus ATN, REN, EOI, IFC and SRQ is principally used by the Controller.

In its simplest configuration the GPIB can consist of only two instruments, a Talker and a Listener; in its most complex configuration up to 961 instruments could be controlled by one or more mini-computers. A bus controller dictates the role of each device by making the ATN line LOW and sending Talk and/or Listen Addresses on the bus data lines. Those devices with matching addresses are activated accordingly. Device addresses are set by switches or PC board jumpers. In single address mode each device has a 5-bit address allowing up to 31 different addresses (one code is used as an unaddress command). In extended address mode each device has a 10-bit address, allowing up to 961 different addresses and the Controller must send two bytes in order to activate a device.

In the configuration shown in Figure 1 a sequence to initiate a data transfer from device A to device D would proceed as follows:

- Controller sends ATN (attention) command; whenever the ATN line goes LOW, devices using the Data bus immediately stop all operations. The Controller keeps the ATN line LOW during the remainder of this sequence.
- Controller sends UNL (unlisten) command; this instruction disables any devices that are in Listen mode.
- Controller sends Talk Address command to device A; this instruction puts device A into Talk mode and disables any other devices that had been in that mode.
- Controller sends Listen Address command to device D, putting it in Listen mode.

When the Controller stops sending ATN, the bus will be released for data transfer functions and device A will begin transmitting to device D.

The SRQ line allows any device to interrupt the Controller and request service. The Controller can identify the interrupting devices by conducting a Serial Poll. To do this it issues an Unlisten (UNL) command followed by a Serial Poll Enable (SPE) command and then the Talk Address of each device in turn. The interrupting device will optionally drive DO7 LOW. Alternatively the Controller can conduct a Parallel Poll by making both EOI and ATN LOW. Devices will then return one bit of status on a DIO line previously assigned via a Parallel Poll Enable (PPE) command.

The REN line allows the Controller to put all Listen addressed instruments into remote control mode, while the IFC line allows the Controller to initialize the system.
Fig. 2 Source and Acceptor Handshake Logic

Talker

Start

Set DAV HIGH

Are Both NRFD & NDAC Sensed HIGH?

Yes

Error Condition

No

Put or Change Data on Data Lines

Is NRFD Sensed HIGH?

No

Set DAV LOW

Yes

Data is valid and may now be accepted

Is NDAC Sensed HIGH?

No

Set DAV HIGH

Yes

More Data?

No

End

Yes

Is NRFD Sensed LOW?

No

Set NRFD HIGH

Is DAV Sensed LOW?

Yes

Accept Data Byte

Set NRFD LOW

Set NDAC HIGH

Has DAV Gone HIGH Yet?

No

Data is not to be considered valid after this time

Yes

Set NDAC LOW
Functions implemented by the 96LS488

**Acceptor Handshake (AH)**
Controls the acquisition of addresses, interface commands and data bytes from the bus. The AH passes data bytes to the instrument logic via a 2-wire handshake (RXST, RXRDY) and a status output (LAD).

**Source Handshake (SH)**
Controls the passing of data bytes and status information from the instrument to the bus. The SH communicates with the instrument via a 2-wire handshake (TXST, TXRDY) and a status output (TAD). It has two operating speeds, selectable via the M0-M3 inputs. Low speed is used with open collector data drivers and gives a settling delay of 2.0 μs. High speed is used with 3-state bus drivers and gives a settling delay of 1.1 μs for the first byte sent and 0.5 μs for the subsequent bytes (clock frequency 10 MHz).

**Listener, Extended Listener (L or LE)**
Either the single or extended address L function can be selected by the M0-M3 inputs. The 96LS488 flags the listener addressed status via the LAD output. The listen address is defined by the A1-A5 inputs and, where the extended address feature is used, the ASEL output controls an external multiplexer to select the primary or secondary address as required.

**Talker, Extended Talker (T OR TE)**
Either the single or extended address T function can be selected. The T function employs the M0-M3 and A1-A5 inputs in the same manner as the L function. The 96LS488 flags the talker addressed status via the TAD output. The T function also incorporates the Serial Poll Function.

**Talker/Listener, Extended Talker/Listener**
For instruments with both talk and listen capabilities, the 96LS488 implements both the T and L functions. In this mode, the "Untalk if My Listen Address" and "Unlisten if My Talk Address" feature is standard. Where the single address mode is selected, different Talk and Listen addresses can be used. In the extended address mode, the Talk and Listen addresses must be identical.
Device Trigger, Device Clear (DT, DC)
These functions generate a pulse on the Trigger or Clear output upon receipt of the relevant bus command.

Remote Local (RL)
The complete RL function is implemented, including the Local Lock-Out feature.

Talk Only, Listen Only
The 96LS488 can operate in either of these two modes via the M0-M3 input code selection.

Service Request (SR)
The 96LS488 initiates an SR on receipt of the RSV input and returns its status on the RQS line when serial polled. The RQS output can be used to drive the DIO7 bus line directly.

Parallel Poll (PP)
The Controller assigns, via the PPE command, one of the eight data lines for use as the Parallel Poll Response output. When the Controller issues the IDY command, the 96LS488 compares the state of the IST input with the state defined by the last PPE command. If the comparison is True, the previously assigned OIO line is driven LOW by the 96LS488.

Pin Functions

488 Bus Signals
All bus inputs have Schmitt trigger buffers, and all bus outputs can sink 48 mA. Each bus signal is terminated with a resistive load and meets the dc load characteristics specified in section 3.5.3 of the IEEE Std 488 – 1978 specification.

DIO1–DIO7 (Data Input/Output)—These are used as inputs to receive addresses and interface commands. They are used as outputs, along with DIO8 to provide Parallel Poll response.

ATN (Attention)—This is an input from the GPIB Controller. When ATN is LOW the 96LS488 interprets the data on DIO1–DIO7 lines as commands or addresses. If the 96LS488 is interrupted by ATN while sending data, it will relinquish control of the Data and Management lines within 200 ns.

DAV (Data Valid)—A bidirectional signal with a 3-state output driver, DAV is part of the handshake system and is driven LOW by current talker when a valid data byte, command or address is on the GPIB. DAV is treated as an input when the 96LS488 is addressed to Listen, or is receiving ATN. It is an output when the 96LS488 is addressed to Talk and ATN is HIGH.

NRFD (Not Ready for Data)—A bidirectional signal with an open-collector output driver, NRFD is part of the handshake system and is driven LOW to indicate that an instrument is not ready to receive data. The 96LS488 drives NRFD HIGH when addressed as a Listener and the instrument is ready to accept a data byte or when ATN is LOW and the 96LS488 is ready to accept an address or interface command. NRFD is treated as an input when the 96LS488 is addressed to Talk.

NDAC (Not Data Accepted)—A bidirectional signal with an open-collector output driver, NDAC is part of the handshake system and is pulled LOW to indicate that a device has not yet accepted a data byte. NDAC is treated as an input when the 96LS488 is addressed to Talk. It is an output and is driven LOW when the 96LS488 is addressed to Listen and the instrument has not accepted a data byte or when receiving ATN and the 96LS488 has not accepted an address or interface command.

ROS (Requested Service)—A 48 mA 3-state output, enabled during a Serial Poll response and driven LOW if the 96LS488 initiated an SRQ, ROS can be directly connected to DIO7 in applications where it is the only status information to be sent.

EOI (End or Identify)—This is an input from the GPIB Controller used to elicit a Parallel Poll response, or from the current active talker to indicate the End-of-string (END) message.

REN (Remote Enable)—This is an input driven by the Controller. The Controller drives it LOW when it needs to remotely program an instrument.

IFC (Interface Clear)—This is an input from the Controller, driven LOW to clear the interface logic. (See Figure 18.)

Instrument Interface and Auxiliary Pins

Instrument Logic Signals—All instrument logic signals are standard low-power Schottky compatible.

CP (10 MHz Clock)—Used to clock internal-state flip-flops and is divided down internally to generate the SH data settling delay. All output changes are synchronous with the negative clock edge. The CP input can be driven by an external oscillator or used in conjunction with XTAL output, as an RC or a crystal oscillator. (See Figure 6.)
XTAL (Crystal)—Used to connect a crystal or external RC timing components for the on-chip oscillator. (See Figure 6.)

MR (Active-LOW Master Reset)—Initializes all internal latches and is completely asynchronous. After a reset, all outputs to the GPIB are passive HIGH and ROS is in the high-Z state: R/L, TRIG, CLR, DRB, ASEL, TAD and LAD are HIGH; D/S/E, RXST, STST and TXST are LOW.

Mo–M3 (Mode Control Inputs)—Define 1 of 14 possible operating modes for the 96LS488. Mo–M3 are HIGH-true inputs.

A1–A5 (Device Address Inputs)—Define the instrument address and originate from switches, PC jumpers or software-loaded register. Where different Talk and Listen addresses are required or when the secondary address feature is used, these inputs must be externally multiplexed, using the ASEL output to control the multiplexer. A1–A5 are HIGH-true inputs (H = 1, L = 0) and thus have the opposite polarity of the DIO1–DIO5 addresses.

ASEL (Address Select Output)—Selects, via an external multiplexer, the Talk/Listen or primary/secondary address input, depending on the operating mode selected; LOW for Talk or primary address; HIGH for Listen or secondary address. (See Figure 11.)

LAD, TAD (Address Status Outputs)—Indicate the Listen-Address or Talk-Address status respectively. They are also activated in the Talk-Only and Listen-Only modes. The outputs are active-LOW to facilitate driving LED indicator lamps. (See Figures 7, 10.)

RXST (Receiver Strobe Output)—Forms part of the handshake logic to pass data bytes to the instrument. When addressed to Listen, the 96LS488 takes RXST HIGH when a valid data byte is on the bus and holds it HIGH until the instrument signals (via the RXRDY input) that it has processed the byte. RXST may be inverted and connected to RXRDY, in which case the 96LS488 will receive data bytes from the bus at a data rate determined solely by the bus handshake. (See Figure 9.)

STST (Status Strobe Output)—Forms part of the handshake logic to pass a status byte from the instrument to the bus during a Serial Poll sequence. It operates in conjunction with the STRDY input in the same way as the TXST and TXRDY signals. STST may be inverted and connected to STRDY, in which case the status byte will be repeated as long as the 96LS488 is addressed to Talk. (See Figure 16.)

TXST (Transmit Strobe Output)—Forms part of the handshake logic controlling the passing of data from the instrument to the bus. When the 96LS488 is addressed to Talk, TXRDY is driven HIGH when the instrument has a data byte to send and LOW to acknowledge receipt of a data byte. (See Figure 8.)

STRDY (Status Ready Input)—Forms part of the handshake logic controlling the passing of a status byte to the bus during a Serial Poll sequence. It operates in a similar fashion to TXRDY. (See Figure 9.)

RSV (Request Service Input)—Is pulled LOW by the instrument to request service and initiate an SRQ interrupt to the controller. This interrupt will be cleared if RSV goes HIGH before it is serviced, but once the SRQ is serviced, RSV must, after exiting SPAS, go HIGH then LOW to initiate another service request. (See Figure 16.)

CLR (Clear Output)—Issues a negative pulse when the 96LS488 receives a Device Clear (DC) command, or when it is addressed to Listen and receives a Selected Device Clear. The CLR Output will stay LOW during Accept Data State (ACDS) or until ATN goes HIGH. (See Figure 13.)

TXRDY (Transmitter Ready Input)—Forms part of the handshake logic controlling the passing of data from the instrument to the bus. When the 96LS488 is addressed to Talk, TXRDY is driven HIGH when the instrument has a data byte to send and LOW to acknowledge that the byte has been accepted by the bus. (See Figure 9.)

DRB (Drive Bus Output)—Taken LOW to enable an external data bus driver when the 96LS488 is addressed to Talk and is in the Talker Active State. DRB will go LOW one clock period after ATN goes HIGH, and will go HIGH asynchronously within 200 ns (typically 70 ns) after ATN goes LOW. (See Figures 9, 10, 11.) DRB can also be used to tell the instrument logic to fetch the first byte.
RQS (Requested Service Output)—A 48 mA 3-state output, enabled during a Serial Poll response and driven LOW if the 96LS488 initiated an SRQ. RQS can be directly connected to DIO7 in applications where it is the only status information to be sent. (See Figures 4, 5, 16.)

DIS/E (Data/Status or END Output)—Valid during the Talk Addressed state and indicates to the instrument logic whether the information to be sent via the bus is to be data or status (LOW for data, HIGH for status). A status byte is sent only in response to a Serial Poll, and, in this case, DIS/E may be used to control a multiplexer to select data or status as the source to the bus data drivers. (See Figures 4, 5, 16.) Valid during the Listener Active State (LACS) to indicate that the current talker is sending the END message; a HIGH output indicates that the END message is true.

R/L (Remote/Local Output)—Goes LOW when the Controller puts the instrument into Remote mode via the REN command. (See Figures 14, 15.)

RTL (Return to Local Input)—Taken LOW to request return of the instrument to local control. RTL will set R/L HIGH unless the Controller has put the 96LS488 into Local Lock-out state. (See Figure 14.)

IST (Instrument Status Input)—Used by the Parallel Poll logic, IST is compared with the logic state defined by DIO4 during the last PPE command. If IST is in the defined state, the 96LS488 will make an affirmative response to the next IDY message by making the assigned DIO line LOW. Note that IST is a HIGH-true input while DIO4 is LOW-true. (See Figure 17.)

Operating Modes

The 96LS488 has 14 operating modes, defined by a 4-bit input code M0-M3 which would normally be selected by switches or PC board jumpers.

Table 1 defines the input codes and operating modes.

<table>
<thead>
<tr>
<th>Mode Inputs</th>
<th>Operating Mode</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>M0 M1 M2 M3</td>
<td>Off Line</td>
<td>The device cannot take part in any GPIB operations</td>
</tr>
<tr>
<td>L L L L</td>
<td>TON (LOW Speed)</td>
<td>The device goes directly to the talk addressed state and can source data to the bus</td>
</tr>
<tr>
<td>L L H L</td>
<td>LON</td>
<td>The device goes directly to the listen addressed state and can receive data from the bus</td>
</tr>
<tr>
<td>L L H H</td>
<td>TON (HIGH Speed)</td>
<td>As for TON (LOW Speed)</td>
</tr>
<tr>
<td>L H L L</td>
<td>T (LOW Speed)</td>
<td>Talker Only, single address mode</td>
</tr>
<tr>
<td>L H L H</td>
<td>TE (LOW Speed)</td>
<td>Talker Only, extended address mode</td>
</tr>
<tr>
<td>L H H L</td>
<td>T (HIGH Speed)</td>
<td>Talker Only, single address mode</td>
</tr>
<tr>
<td>L H H H</td>
<td>TE (HIGH Speed)</td>
<td>Talker Only, extended address mode</td>
</tr>
<tr>
<td>H L L L</td>
<td>L</td>
<td>Listener Only, single address mode</td>
</tr>
<tr>
<td>H L L H</td>
<td>LE</td>
<td>Listener Only, extended address mode</td>
</tr>
<tr>
<td>H H L L</td>
<td>T/L (LOW Speed)</td>
<td>Talker/Listener, dual address mode</td>
</tr>
<tr>
<td>H H L H</td>
<td>TE/LE (LOW Speed)</td>
<td>Talker/Listener, extended address mode</td>
</tr>
<tr>
<td>H H H L</td>
<td>T/L (HIGH Speed)</td>
<td>Talker/Listener, dual address mode</td>
</tr>
<tr>
<td>H H H H</td>
<td>TE/LE (HIGH Speed)</td>
<td>Talker/Listener, extended address mode</td>
</tr>
</tbody>
</table>

Notes
1. The LOW speed talker option is selected where open-collector data drivers are used. The delay from putting valid data on the GPIB to DAV going true is 2.0 µs. The HIGH speed option is selected where 3-state drivers are used. The settling delay (data to DAV) is 1.1 µs for the first byte sent after a LOW to HIGH transition of ATN and 500 ns for subsequent bytes.
2. For dual address Talker/Listener modes the Talk and Listen addresses can be different.
Addressing Modes

Where extended addressing or different Talk and Listen addresses are required, the address codes must be externally multiplexed, using ASEL. (See Figure 3.) In the extended address modes, ASEL is LOW for the primary address and HIGH for the secondary address. (See Figure 11.) In the dual address modes, ASEL is HIGH for the Listen address and LOW for the Talk address.

In single address mode the 96LS488 will go into the addressed state on receipt of the primary address. In extended address mode it will go to the addressed state on receipt of its secondary address if, and only if, it has received its primary address. If a device is addressed to Talk and receives its Listen address it will un-address as a Talker and go to Listener addressed state, and vice versa. A Talker Addressed device will un-address if it receives a non-matching talk address. The 96LS488 indicates its address status on the TAD, LAD and D/S/E outputs. (See Table 3.)

Fig. 3 Address Multiplexer

<table>
<thead>
<tr>
<th>DIO8</th>
<th>DIO7</th>
<th>DIO6</th>
<th>DIO5</th>
<th>DIO4</th>
<th>DIO3</th>
<th>DIO2</th>
<th>DIO1</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>H</td>
<td>L</td>
<td>A5</td>
<td>A4</td>
<td>A3</td>
<td>A2</td>
<td>A1</td>
</tr>
<tr>
<td>X</td>
<td>H</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>X</td>
</tr>
<tr>
<td>X</td>
<td>L</td>
<td>H</td>
<td>A5</td>
<td>A4</td>
<td>A3</td>
<td>A2</td>
<td>A1</td>
</tr>
<tr>
<td>X</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>X</td>
</tr>
</tbody>
</table>

Table 2

Status Codes

<table>
<thead>
<tr>
<th>TAD</th>
<th>LAD</th>
<th>D/S/E</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>H</td>
<td>L</td>
<td>Off Line</td>
</tr>
<tr>
<td>H</td>
<td>L</td>
<td>L</td>
<td>Addressed to Listen (LADS)</td>
</tr>
<tr>
<td>L</td>
<td>H</td>
<td>L</td>
<td>Addressed to Talk (TADS)</td>
</tr>
<tr>
<td>L</td>
<td>H</td>
<td>H</td>
<td>Serial Poll Mode (SPM)</td>
</tr>
<tr>
<td>H</td>
<td>L</td>
<td>H</td>
<td>Receiving END Message (LACS)</td>
</tr>
</tbody>
</table>
Status Response

In Serial Poll Active State (SPAS) the instrument is requested to return a status byte, via the usual handshake, to the Controller. Seven bits are defined by the instrument. Bit 7 denotes the Request Service Status (ROS) and is provided by the 96LS488 on the ROS output. If the instrument provides no status, other than ROS, then the ROS output can drive the bus directly (Figure 4).

If the instrument provides status information this can be multiplexed to the bus using D/S/E (Figure 5). After the bus handshake, the instrument can send a second status byte by making STRDY LOW then HIGH again, which starts the handshake. This sequence can be repeated to send additional bytes, as long as ATN remains HIGH. (Figure 16.)

Fig. 4 Status Bit Driven Direct by ROS
### Clock Input

The CP and XTAL inputs allow the 96LS488 either to accept external clock pulses or to generate its own clock. (See Figure 6.)

(a) An external clock can drive CP. The XTAL pin may be used as an inverted buffered version of the external clock.

(b) The Schmitt-trigger buffer-inverter between the CP and XTAL pins can be used as a relaxation oscillator by connecting an RC network to provide feedback. The frequency of oscillation is variable up to 10 MHz. XTAL may be used to clock external circuits provided it is buffered.

(c) The CP and XTAL pins will form a stable crystal oscillator by connecting a 10 MHz crystal and an RC network to provide positive feedback. XTAL may be used to clock external circuits provided it is buffered.
Timing Sequences

A 10 MHz clock frequency is recommended to give the correct Source Handshake delays. The 96LS488 can be clocked at a slower rate if the GPIB is not running at its maximum data rate of 1 M byte. The lowest clock frequency allowable is dependent on the GPIB speed.

Regardless of clock frequency, the 96LS488 will respond to ATN within 200 ns by disabling NRFD, NDAC and DAV while forcing DRB HIGH to disable the data drivers. In Parallel Poll sequences the relevant DIO line will be enabled or disabled within 200 ns of an IDY transition.

Since the internal logic of the 96LS488 is synchronous with the CP input, while in general all inputs are asynchronous, the precise timing of all responses to external signals is subject to a maximum uncertainty equal to one clock period. This is illustrated in the following timing diagrams, where some delays are defined as $t_{cp} + t_x$ (i.e., clock period + propagation delay).

Listen Address Sequence (See Figure 7)

1. Controller takes ATN line LOW followed by
2. putting the Listen address on the GPIB.
3. Within $(t_{cp} + t_{PHL})$, the 96LS488 takes NRFD and NDAC LOW.

and a $t_{cp}$ later takes NRFD HIGH, indicating that the 96LS488 is ready for data.

e. After a delay TCDAV(L) (determined by the Controller logic) the Controller takes DAV LOW.

f. Within $(t_{cp} + t_{PHL})$, the 96LS488 takes NRFD LOW and

g. 3 clock periods later NDAC goes HIGH, indicating that the 96LS488 has accepted the data,

h. followed by LAD indicating listen addressed status

i. After a Controller dependent delay, TCDAV(H), the DAV line goes HIGH and

j. within $(t_{cp} + t_{PHL})$, the 96LS488 takes NDAC LOW.

k. A $t_{cp}$ later the 96LS488 allows NRFD to go HIGH. NRFD stays HIGH until the Controller takes ATN HIGH, unless a further command is sent over the GPIB when a similar handshake sequence takes place.

l. ATN goes HIGH followed by

m. NRFD going LOW within $(t_{cp} + t_{PHL})$. This occurs if the instrument is not ready to receive data (RXRDY LOW). If RXRDY is HIGH, NRFD will stay HIGH allowing a data transfer to take place.

Fig. 7 Timing Diagram for Listen Address Sequence

Note: ATN, DIO Bus, & DAV driven by controller.
NRFD, NDAC Driven by 96LS488.
Data Transfer from Bus to Listener (See Figure 8)
Assuming the instrument logic responds to RXST within one clock period. (See Figure 8a.)

a. The instrument signals it is ready to receive a byte by taking RXRDY HIGH (keeping RXRDY LOW constitutes an "NRFD hold").
b. Provided the 96LS488 is in the Listen Addressed State (LADS), NRFD is taken HIGH within (tcp + tPHL)
c. When the current Talker sees the NRFD line HIGH it takes DAV LOW after a settling delay TIDAV(L).
d. The 96LS488 takes RXST HIGH within (tcp + tPLH) to inform the instrument that the GPIB data is valid and takes NRFD LOW

e. Assuming the instrument responds by pulsing RXRDY LOW within one clock period (tpRX < tcp) then RXST will remain HIGH only for one clock
f. At the same time as RXST returns LOW, NDAC is taken HIGH to inform the Talker that data has been accepted
g. After a delay TTDAV(H) determined by the Talker the DAV line goes HIGH
h. Within (tcp + tPLH) NDAC goes LOW
i. Followed by NRFD going HIGH one clock later

Assuming the instrument logic is slow and requires more than one clock period to process a data byte. (See Figure 8b.)

The timing sequence is identical to Figure 8a until RXST goes HIGH

a. After RXST goes HIGH the instrument delays tpRX (>1 clock cycle) before taking RXRDY LOW
b. RXST will remain HIGH and NDAC will remain LOW during this period, causing the bus data to be maintained valid
c. RXST goes LOW, and NDAC goes HIGH within (tcp + tPHL) of RXRDY
d. After a delay TTDAV(H) the Talker takes DAV HIGH
e. Within (tcp + tPLH) NDAC goes LOW
f. Assuming the instrument is not ready (i.e., is holding RXRDY LOW) the 96LS488 holds NRFD LOW preventing another data transfer from starting
g. Within (tcp + tPLH) of RXRDY going HIGH, NRFD goes HIGH and the next data transfer cycle can start

Fig. 8 Timing Diagram, Data Transfer from Bus to Listener

In applications where the bus data can be latched by the RXST rising edge and handshaking is not necessary, RXRDY can be driven by RXST via an inverter.
Data Transfer from Talker to GPIB (See Figure 9)

a. ATN goes HIGH after completion of a Talk address sequence; TAD (not shown) is already LOW.

b. Within \( (t_{cp} + t_{PHL}) \) DRB goes LOW to enable the bus drivers.

c. At a time determined by the instrument logic, TXRDY goes HIGH.

d. If NRFD is already HIGH, the 96LS488 drives DAV LOW after delay \( T_1 \) (11 \( \times \) \( t_{cp} \) + \( t_{PHL} \) in high or 20 \( \times \) \( t_{cp} \) + \( t_{PHL} \) in low speed). If NRFD is LOW, DAV will stay HIGH until NRFD goes HIGH (assuming \( T_1 \) has expired). The DAV LOW period corresponds to the Source Transfer State (STRS).

e. The Listener(s) respond by eventually taking NDAC HIGH.

f. Within \( (t_{cp} + t_{PLH}) \) 96LS488 takes DAV HIGH and TXST HIGH to inform the instrument that the data has been accepted and a new byte can be presented.

g. The instrument takes TXRDY LOW and holds it there until it has provided a new byte, h or \( h' \). The minimum time TXRDY must be LOW is \( t_{PLW} \).

h. If TXRDY pulses LOW within \( t_{cp} \) of TXST, TXST(H) will be a minimum of \( t_{cp} \) wide. Otherwise TXST goes LOW within \( (t_{cp} + t_{PHL}) \) of TXRDY, \( j \). After the first byte is sent, \( T_1 \) drops from 11 \( \times \) \( t_{cp} \) to 5 \( \times \) \( t_{cp} \) in high speed mode.

i. If TXRDY is HIGH before NRFD, DAV goes LOW within \( T_2 \) of NRFD going HIGH (5 \( \times \) \( t_{cp} \) in high speed or 20 \( \times \) \( t_{cp} \) in low speed)

If TXRDY does not go HIGH until \( h' \) (after NRFD goes HIGH) DAV goes LOW \( T_3 \) later, \( i' \). \( (T_3 = 6 \times t_{cp} \) in HIGH and 21 \( \times \) \( t_{cp} \) in low speed).

If the Talk sequence is interrupted by ATN, while the instrument is generating a new byte, (between \( f \) and \( h \), DRB will go HIGH within 200 ns and the DAV line will be relinquished. DRB will return LOW and the sequence will continue within \( (t_{cp} + t_{PLH}) \) of ATN going HIGH. If the 96LS488 is in high speed mode the first data byte sent will have a delay \( T_1 = 11 \times t_{cp} \).

Note

In order to get the source handshake delays specified in IEEE Std 488 - 1978, \( t_{cp} \) must be 100 ns (\( f_{clock} = 10 \) MHz).

---

Fig. 9 Timing Diagram Data Transfer from Talker to Bus

---

Notes

NRFD, NDAC are driven by the current listener(s) DAV is driven by 96LS488.

\[ T_1 = 11 \times t_{cp} + t_{PHL} \] in high speed, \( 20 \times t_{cp} + t_{PHL} \) in low speed

\[ T_2 = 5 \times t_{cp} + t_{PHL} \] in high speed, \( 20 \times t_{cp} + t_{PHL} \) in low speed

\[ T_3 = 6 \times t_{cp} + t_{PHL} \] in high speed, \( 21 \times t_{cp} + t_{PHL} \) in low speed
Talk Address Sequence (See Figure 10)
This is similar to the Listen address sequence.

Fig. 10 Talk Address Sequence

Fig. 11 Timing Diagram for Secondary Address Sequence
Fig. 12 Timing Diagram for 96LS488 Receiving Bus Commands

**Diagram Description:**
- **ATN:**
  - Transition times: $t_{cp}$ and $t_{PHL}$
- **NRFD:**
  - Transition times: $t_{cp}$ and $t_{PHL}$
- **DAV:**
  - Transition times: $4 \times t_{cp}$ and $t_{PHL}$
- **NDAC:**
  - Transition times: $4 \times t_{cp}$ and $t_{PHL}$

**Note:**
Internal state changes do not necessarily change any 96LS488 outputs.

Fig. 13 Timing Diagram for Device Clear and Device Trigger Commands

**Diagram Description:**
- **ATN:**
  - Transition times: $t_{cp}$ and $t_{PHL}$
- **NRFD:**
  - Transition times: $2 \times t_{cp}$ and $t_{PHL}$
- **DAV:**
  - Transition times: $t_{cp}$ and $t_{PHL}$
- **NDAC:**
  - Transition times: $4 \times t_{cp}$ and $t_{PHL}$
- **TRIGCLR:**
  - Transition times: $2 \times t_{cp}$ and $t_{PHL}$

**Note:**
Internal state changes do not necessarily change any 96LS488 outputs.
Fig. 14 Timing Diagram for Remote/Local Logic  
(Starting in LOCAL State)

DIO BUS: Valid Listen Address or LLO Command

REN

ATN

NRFD

DAV

NDAC

RL

RL

Note:
If LLO has been sent, RTL will not cause RL to change.

Fig. 15 Timing Diagram for Remote/Local Logic  
(Starting in REMOTE State)
Serial Poll Sequence (See Figure 16)
a. The Controller has sent the Unlisten (UNL) and Serial Poll Enable (SPE) commands, generally in response to a LOW signal on SRQ, and places a Talk address on the GPIB. The 96LS488 is sending the Service Request (SRQ) message, which was caused by the instrument logic making RSV LOW.
b. The 96LS488 allows NRFD to float passive HIGH to initiate normal handshake routine.
c. The Controller forces DAV LOW. Since ATN is also LOW, the 96LS488 receives the GPIB information as the message My Talk Address (MTA)
d. NRFD is active, acknowledging that the device is receiving a data byte

e. The 96LS488 enters the Talker Addressed State at time (t_{CP} + t_{PHL}) after DAV was forced LOW
f. D/S/E goes HIGH at time (4 \times t_{CP} + t_{PZH}) after DAV was forced LOW. This indicates that the 96LS488 is in the Serial Poll Mode.
g. NDAC is allowed to float passive HIGH, indicating that the command data byte has been received
h. The Controller takes DAV HIGH
i. NDAC is pulled LOW, showing that the 96LS488 is ready for a new handshake cycle
j. The Controller allows the bus to float

k. The Controller releases ATN. Because the 96LS488 is in the Serial Poll Mode, it now enters the Serial Poll Active State (SPAS), which prevails until the Controller makes ATN LOW again.
l. DSB goes active LOW, allowing the instrument to place its status byte on the bus. DSB goes LOW at time (t_{CP} + t_{PZL}) after ATN goes HIGH
m. When the 96LS488 enters SPAS, SRQ goes HIGH at time (t_{PLH} after ATN goes HIGH
n. The instrument indicates that a status byte is ready by taking STRDY HIGH. This may occur earlier than shown, without affecting any of the foregoing
o. DSB enables the 3-state output RQS when in SPAS. RQS goes LOW at time (t_{CP} + t_{PLH}) after ATN is released
p. The Controller has taken NRFD HIGH, acknowledging that it is ready for the status byte
q. The 96LS488 takes DAV LOW after the time interval T_s, which starts either at the rising edge of STRDY or the falling edge of DSB, whichever occurs later. The DAV LOW period corresponds to the Source Transfer State (STRS).
r. The Controller takes NRFD LOW
s. The instrument may release RSV at any time after the 96LS488 enters SPAS
t. The Controller takes NDAC HIGH, acknowledging that it has received the status byte
u. The 96LS488 releases DAV at time (t_{CP} + t_{PHL}) after NDAC goes HIGH
v. STS goes HIGH at time (t_{CP} + t_{PZH}) after NDAC goes HIGH. This tells the instrument that the status byte has been accepted. The instrument takes STRDY LOW to indicate that the data is no longer valid and to allow STS to go LOW again one t_{CP} after STRDY goes LOW
w. The Controller takes NDAC LOW once more after DAV goes HIGH
x. The data on the bus is no longer valid. If the ATN line remains HIGH, the instrument can provide another status byte by making STRDY HIGH to indicate valid data and to start the handshake
y. At the completion of the Serial Poll of this instrument, the Controller assumes control of the bus by forcing ATN LOW
z. DSB goes HIGH at time t_{PZH} after ATN goes LOW. This places the bus drivers of this instrument in the high-impedance (3-state output) or off (open-collector outputs) state
aa. Since DSB is no longer valid, the RQS output reverts to its high-impedance state
bb. NRFD goes HIGH, indicating that devices are ready to receive a command from the bus
c. The Controller forces DAV LOW to show that it has placed a control byte on the bus
dd. NRFD goes LOW to acknowledge DAV
e. NDAC goes HIGH when devices all acknowledge acceptance of the command byte
ff. The 96LS488 has received the Other Talk Address (OTA) command and reverts to its unaddressed state. TAD goes HIGH at time (4 \times t_{CP} + t_{PLH}) after DAV went LOW
gg. DAV is set HIGH by the Controller
hh. Because the 96LS488 has been unaddressed it is no longer in the Serial Poll Active State (SPAS). The D/S/E output goes LOW at time (4 \times t_{CP} + t_{PZH}) after DAV went HIGH. The 96LS488 is still in the Serial Poll Mode State (SPMS), however, and will return to SPAS (D/S/E HIGH, STS and STRDY valid) if subsequently addressed to talk. The 96LS488 enters the Serial Poll Idle State (SPIS) when the Controller either issues the Serial Poll Disable (SPD) command or makes IFC LOW.
Parallel Poll Sequence (See Figure 17)

a. The Controller sends the instrument’s listen address.
b. Then the Controller issues the Parallel Poll Configure command; this enables the 96LS488 to receive a subsequent PPE command.
c. The Controller issues the Parallel Poll Enable command. Bits D1–D3 of the command byte determine which data output (DIOj) will be valid when the IDY command is sent. Bit D4 of the command is compared with IST (Instrument Status) during the IDY command.
d. The Controller issues the Unlisten command. The 96LS488 will now not respond to further PPE commands, allowing the Controller to configure other instruments.
e. The Instrument Status bit (IST) is set by the instrument before an IDENTIFY command is received. IST must be in a stable state when IDY is received so the setup and hold times must be observed (tS IST and tH IST).
f. During the IDY command the Controller releases the data lines DIO1–DIO7, DIO8. The assigned data line DIOj will be taken active LOW by the 96LS488 if IST compares with bit D4 of the PPE command.
g. The IDY message is received (IDY = EOI • ATN). At this time the Instrument Status bit IST is latched in the 96LS488, and the output data DIOj is true if IST compares with bit D4 of the PPE command. DIOj is LOW if D4 was LOW and IST was HIGH, or if D4 was HIGH and IST was LOW.
h. The 96LS488 enables data bit DIOj after a delay tPHL from the time EOI goes active LOW. DIOj remains valid while IDY is active. Note that this is an asynchronous message sent and is not governed by the handshake protocol.

i. IST may be altered not less than time t1 IST after IDY is active. Because IST is latched by IDY it will not affect the status byte sent during this IDY routine.

j. IDY is false and the 96LS488 stops sending a status byte. Outputs DIO1-DIO7, DOs again float passive HIGH.

k. The status bit DIOj floats passive HIGH at a time not greater than tPLH after IDY goes FALSE.

l. The Controller can now place information on the data bus.

m. Once the 96LS488 has been configured via steps a-c, the Controller can examine IST at any time by issuing the IDY command. To change the D1-D4 assignment of a particular 96LS488, the Controller must address it to listen, issue the PPC command, then the Parallel Poll Disable (PPD) command (which clears the D1-D4 latches), then the PPE command with the revised D1-D4 assignment. The D1-D4 assignment will also be cleared by the universal Parallel Poll Unconfigure (PPU) command or by MR, but not by IFC. PPU, MR or the MLA/PPC/PPD sequence puts the 96LS488 in the Parallel Poll Idle State (PPIS) and it will not respond to IDY until it is subsequently reconfigured.

Fig. 17 Timing Diagram for Parallel Poll Sequence
Fig. 18 IFC Timing Diagram

Absolute Maximum Ratings
(above which the useful life may be impaired)

- Storage Temperature: -65°C to +150°C
- Temperature (Ambient) Under Bias: -55°C to +125°C
- Vcc Pin Potential to Ground Pin: -0.5 V to +7.0 V
- *Input Voltage (dc): -0.5 V to +15 V
- *Input Current (dc): -30 mA to +5.0 mA
- Voltage Applied to Outputs (Output HIGH): -0.5 V to +5.5 V
- Output Current (dc) (Output LOW): +50 mA

*Either Input Voltage limit or Input Current limit is sufficient to protect the inputs.
### DC Characteristics

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Parameter</th>
<th>Min</th>
<th>Max</th>
<th>Unit</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$V_{IH}$</td>
<td>Input HIGH Voltage All Inputs</td>
<td>2.0</td>
<td></td>
<td>V</td>
<td>Recognized as a HIGH Signal Over Recommended $V_{CC}$ and $T_A$ Range</td>
</tr>
<tr>
<td>$V_{IL}$</td>
<td>Input LOW Voltage All Inputs</td>
<td>DM</td>
<td>0.7</td>
<td>0.8</td>
<td>V</td>
</tr>
<tr>
<td>$V_{T^+}-V_{T^-}$</td>
<td>Hysteresis Voltage</td>
<td></td>
<td>0.4</td>
<td></td>
<td>V</td>
</tr>
<tr>
<td>$V_{CD}$</td>
<td>Input Clamp Diode Voltage</td>
<td></td>
<td>1.2</td>
<td></td>
<td>V</td>
</tr>
<tr>
<td></td>
<td>Output HIGH Voltage RQS, DAV</td>
<td></td>
<td></td>
<td></td>
<td>V</td>
</tr>
<tr>
<td>$V_{OH}$</td>
<td>$DIO_1 - DIO_7, DIO_{IL}$, SRQ, NRFD, NDAC</td>
<td>2.5</td>
<td></td>
<td>3.7</td>
<td>V</td>
</tr>
<tr>
<td></td>
<td>$\bar{R}/L, \bar{D}/S/E, RXST, TXST$, STST, CLR, TRIG, DRB, ASEL, XTAL, LAD, TAD</td>
<td>2.5</td>
<td></td>
<td></td>
<td>V</td>
</tr>
<tr>
<td>$V_{OL}$</td>
<td>Output LOW Voltage $DIO_1 - DIO_7, DIO_{IL}$, SRQ, RQS, NRFD, NDAC, DAV</td>
<td></td>
<td></td>
<td></td>
<td>V</td>
</tr>
<tr>
<td></td>
<td>All Other Outputs DM, DC</td>
<td></td>
<td></td>
<td></td>
<td>V</td>
</tr>
<tr>
<td></td>
<td>DC</td>
<td></td>
<td></td>
<td></td>
<td>V</td>
</tr>
<tr>
<td>$I_{IH}$</td>
<td>Input HIGH Current 1.0 U.L.</td>
<td>0.5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>n U.L.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>n(40)</td>
<td>20</td>
<td>40</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\mu A$</td>
<td>Max</td>
</tr>
<tr>
<td>$I_{IL}$</td>
<td>Input LOW Current 0.25 U.L.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>n U.L.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>n(−1.6)</td>
<td>0.4</td>
<td>0.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>mA</td>
<td>Max</td>
</tr>
<tr>
<td>$I_{POPS}$</td>
<td>Leakage into GPIB Pins in Powered-off State</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\mu A$</td>
<td>100</td>
</tr>
<tr>
<td>$I_{OZH}$</td>
<td>3-State Output OFF Current HIGH</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\mu A$</td>
<td>40</td>
</tr>
<tr>
<td>$I_{OZL}$</td>
<td>3-State Output OFF Current LOW</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>$\mu A$</td>
<td>−40</td>
</tr>
<tr>
<td>$I_{CC}$</td>
<td>Power Supply Current</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>mA</td>
<td>250</td>
</tr>
</tbody>
</table>

*Typical Value for $I_{CC}$ is 180 mA.*
### AC Characteristics \( V_{CC} = 5.0 \, V, \, T_A = +25^\circ C \)

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Parameter</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Unit</th>
<th>Fig. No.</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f_{\text{max}} )</td>
<td>Maximum Clock Frequency</td>
<td>12</td>
<td>15</td>
<td></td>
<td>MHz</td>
<td></td>
<td>For correct timing for SH or AH, ( CP ) must be 10 MHz.</td>
</tr>
<tr>
<td>( t_{\text{PLH}} )</td>
<td>Propagation Delay</td>
<td>150</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>7, 8, 10, 11, 12, 13, 14</td>
<td>( C_L = 300 , pF, , R_L = 200 , \Omega )</td>
</tr>
<tr>
<td>( t_{\text{PLH}} )</td>
<td>Propagation Delay ( CP ) to ( NRFD )</td>
<td>90</td>
<td></td>
<td></td>
<td>ns</td>
<td></td>
<td>( C_L = 300 , pF, , R_L = 200 , \Omega )</td>
</tr>
<tr>
<td>( t_{\text{PLH}} )</td>
<td>Propagation Delay ( ATN ) to ( NRFD )</td>
<td>150</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>11</td>
<td>( C_L = 300 , pF, , R_L = 200 , \Omega )</td>
</tr>
<tr>
<td>( t_{\text{PLH}} )</td>
<td>Propagation Delay ( CP ) to ( NDAC )</td>
<td>150</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>7, 8, 10, 11, 12, 13, 14</td>
<td>( C_L = 300 , pF, , R_L = 200 , \Omega )</td>
</tr>
<tr>
<td>( t_{\text{PLH}} )</td>
<td>Propagation Delay ( ATN ) to ( NDAC )</td>
<td>90</td>
<td></td>
<td></td>
<td>ns</td>
<td></td>
<td>( C_L = 300 , pF, , R_L = 200 , \Omega )</td>
</tr>
<tr>
<td>( t_{\text{PLH}} )</td>
<td>Propagation Delay ( CP ) to ( DAV )</td>
<td>100</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>9</td>
<td>( C_L = 300 , pF, , R_L = 200 , \Omega )</td>
</tr>
<tr>
<td>( t_{\text{PLZ}} )</td>
<td>Output Disable Time ( CP ) to ( DAV )</td>
<td>100</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>9</td>
<td>( C_L = 300 , pF, , R_L = 200 , \Omega )</td>
</tr>
<tr>
<td>( t_{\text{PHZ}} )</td>
<td>Output Enable Time ( ATN ) to ( DAV )</td>
<td>100</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td></td>
<td>( C_L = 300 , pF, , R_L = 200 , \Omega )</td>
</tr>
<tr>
<td>( t_{\text{PLZ}} )</td>
<td>Output Disable Time ( CP ) to ( DAV )</td>
<td>100</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td></td>
<td>( C_L = 300 , pF, , R_L = 200 , \Omega )</td>
</tr>
<tr>
<td>( t_{\text{PHZ}} )</td>
<td>Propagation Delay ( IFC ) to ( NRFD ) or ( NDAC )</td>
<td>180</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>18</td>
<td>( C_L = 300 , pF, , R_L = 200 , \Omega )</td>
</tr>
<tr>
<td>( t_{\text{PHL}} )</td>
<td>Propagation Delay ( IFC ) to ( RXST )</td>
<td>160</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>18</td>
<td>( C_L = 15 , pF )</td>
</tr>
<tr>
<td>( t_{\text{PHL}} )</td>
<td>Propagation Delay, ( IFC ) to ( D/S/E )</td>
<td>100</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>18</td>
<td>( C_L = 15 , pF )</td>
</tr>
<tr>
<td>( t_{\text{PHL}} )</td>
<td>Propagation Delay ( CP ) to ( LAD ) or ( TAD )</td>
<td>150</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>7, 8, 10</td>
<td>( C_L = 15 , pF )</td>
</tr>
<tr>
<td>( t_{\text{PHL}} )</td>
<td>Propagation Delay ( Mo - M_3 ) to ( LAD ) or ( TAD )</td>
<td>60</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td></td>
<td>( C_L = 15 , pF )</td>
</tr>
<tr>
<td>( t_{\text{PHL}} )</td>
<td>Propagation Delay ( CP ) to ( RXST ), ( TXST ) or ( STST )</td>
<td>75</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>8, 9</td>
<td>( C_L = 15 , pF )</td>
</tr>
<tr>
<td>( t_{\text{PHL}} )</td>
<td>Propagation Delay ( CP ) to ( DRB )</td>
<td>85</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>9, 11</td>
<td>( C_L = 15 , pF )</td>
</tr>
<tr>
<td>( t_{\text{PHL}} )</td>
<td>Propagation Delay ( ATN ) to ( DRB )</td>
<td>70</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>9</td>
<td>( C_L = 15 , pF )</td>
</tr>
<tr>
<td>( t_{\text{PHL}} )</td>
<td>Propagation Delay ( CP ) to ( TRIG ) or ( CLR )</td>
<td>130</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>13</td>
<td>( C_L = 15 , pF )</td>
</tr>
<tr>
<td>( t_{\text{PHL}} )</td>
<td>Propagation Delay ( DAV ) to ( ASEL )</td>
<td>25</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>11</td>
<td>( C_L = 15 , pF )</td>
</tr>
</tbody>
</table>
### AC Characteristics

\( V_{CC} = 5.0 \, V, \, T_A = +25^\circ C \) (Cont'd)

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Parameter</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Unit</th>
<th>Fig. No.</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>( t_{PHL} )</td>
<td>Propagation Delay, MR to RXST, TXST, STST, D/S/E</td>
<td>100</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>6</td>
<td>( C_L = 15 , pF )</td>
</tr>
<tr>
<td>( t_{PLH} )</td>
<td>Propagation Delay MR to LAD, TAD or ( \bar{R}/L )</td>
<td>50</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>14, 15</td>
<td>( C_L = 15 , pF )</td>
</tr>
<tr>
<td>( t_{PLH} )</td>
<td>Propagation Delay, MR to NRFD, NDAC, DAV, ( \overline{DIO_1-DIO_7}, \overline{DO_b} ) or SRO</td>
<td>100</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td></td>
<td>( C_L = 300 , pF, , R_L = 200 , \Omega )</td>
</tr>
<tr>
<td>( t_{PLH} )</td>
<td>Propagation Delay ( \overline{CP} ) to XTAL</td>
<td>15</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>6</td>
<td>( C_L = 15 , pF, , \overline{CP} ) driven from ext. oscillator</td>
</tr>
<tr>
<td>( t_{PHL} )</td>
<td>Propagation Delay ( \overline{CP} ) to ( \bar{R}/L )</td>
<td>150</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>14, 15</td>
<td>( C_L = 15 , pF )</td>
</tr>
<tr>
<td>( t_{PLH} )</td>
<td>Propagation Delay, ( R_T L ) to ( \bar{R}/L )</td>
<td>150</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>14</td>
<td>( C_L = 15 , pF )</td>
</tr>
<tr>
<td>( t_{PHL} )</td>
<td>Propagation Delay, ( R_N ) to ( \bar{R}/L )</td>
<td>45</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>14</td>
<td>( C_L = 15 , pF )</td>
</tr>
<tr>
<td>( t_{PLH} )</td>
<td>Propagation Delay, ( \overline{CP} ) to ( \overline{D/S/E} )</td>
<td>140</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>16</td>
<td>( C_L = 15 , pF )</td>
</tr>
<tr>
<td>( t_{PHL} )</td>
<td>Propagation Delay, ( \overline{RSV} ) to ( \overline{SFQ} )</td>
<td>30</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td></td>
<td>( C_L = 300 , pF, , R_L = 200 , \Omega )</td>
</tr>
<tr>
<td>( t_{PLH} )</td>
<td>Propagation Delay, ( A_{TN} ) to ( \overline{SFQ} )</td>
<td>150</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>16</td>
<td>( C_L = 300 , pF, , R_L = 200 , \Omega )</td>
</tr>
<tr>
<td>( t_{PZH} )</td>
<td>Output Enable Time, ( \overline{CP} ) to ( R_Q S )</td>
<td>100</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>16</td>
<td>( C_L = 300 , pF, , R_L = 200 , \Omega )</td>
</tr>
<tr>
<td>( t_{PLZ} )</td>
<td>Output Disable Time ( A_{TN} ) to ( R_Q S )</td>
<td>85</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>16</td>
<td>( C_L = 300 , pF, , R_L = 200 , \Omega )</td>
</tr>
<tr>
<td>( t_{PHL} )</td>
<td>Propagation Delay ( \overline{EOI} ) to ( \overline{DIO_1-DIO_7}, \overline{DO_b} )</td>
<td>110</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td>17</td>
<td>( C_L = 300 , pF, , R_L = 200 , \Omega )</td>
</tr>
<tr>
<td>( t_{PHL} )</td>
<td>Propagation Delay ( \overline{EOI} ) to ( \overline{D/S/E} )</td>
<td>30</td>
<td>ns</td>
<td></td>
<td>ns</td>
<td></td>
<td>( C_L = 15 , pF )</td>
</tr>
</tbody>
</table>
### AC Operating Requirements  \( V_{CC} = 5.0 \, V, \, T_A = +25^\circ C \)

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Parameter</th>
<th>Min</th>
<th>Typ</th>
<th>Max</th>
<th>Unit</th>
<th>Fig. No.</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>( t_s ) (L)</td>
<td>Setup Time, HIGH or LOW IST to EOI</td>
<td>20</td>
<td></td>
<td></td>
<td>ns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( t_s ) (H)</td>
<td>Setup Time, HIGH or LOW IST to EOI</td>
<td>20</td>
<td></td>
<td></td>
<td>ns</td>
<td></td>
<td>17</td>
</tr>
<tr>
<td>( t_H ) (L)</td>
<td>Hold Time, HIGH or LOW IST to EOI</td>
<td>5.0</td>
<td></td>
<td></td>
<td>ns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( t_H ) (H)</td>
<td>Hold Time, HIGH or LOW IST to EOI</td>
<td>5.0</td>
<td></td>
<td></td>
<td>ns</td>
<td></td>
<td>17</td>
</tr>
<tr>
<td>( t_s ) (L)</td>
<td>Setup Time, RSV to ATN</td>
<td></td>
<td>-50</td>
<td></td>
<td>ns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( t_H ) (L)</td>
<td>Hold Time, RSV to ATN</td>
<td></td>
<td>-50</td>
<td></td>
<td>ns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( t_s ) (H)</td>
<td>Setup Time, HIGH or LOW ( \overline{DIO_1} \rightarrow DIO_7 ) to DAV</td>
<td>100</td>
<td></td>
<td></td>
<td>ns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( t_s ) (L)</td>
<td>Setup Time, HIGH or LOW ( \overline{DIO_1} \rightarrow DIO_7 ) to DAV</td>
<td>100</td>
<td></td>
<td></td>
<td>ns</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>( t_H ) (H)</td>
<td>Hold Time, HIGH or LOW ( \overline{DIO_1} \rightarrow DIO_7 ) to DAV</td>
<td></td>
<td>0</td>
<td></td>
<td>ns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( t_H ) (L)</td>
<td>Hold Time, HIGH or LOW ( \overline{DIO_1} \rightarrow DIO_7 ) to DAV</td>
<td></td>
<td>0</td>
<td></td>
<td>ns</td>
<td></td>
<td>11</td>
</tr>
<tr>
<td>( t_m ) (H)</td>
<td>Delay in external address mux from ASEL to valid address</td>
<td></td>
<td>100</td>
<td></td>
<td>ns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( t_m ) (L)</td>
<td>Delay in external address mux from ASEL to valid address</td>
<td></td>
<td>100</td>
<td></td>
<td>ns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( t_w ) (H)</td>
<td>CP Pulse Width</td>
<td>30</td>
<td></td>
<td></td>
<td>ns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( t_w ) (L)</td>
<td>CP Pulse Width</td>
<td>30</td>
<td></td>
<td></td>
<td>ns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( t_w ) (L)</td>
<td>Pulse Width ( \overline{RXRDY}, \overline{TXRDY} ) or ( \overline{STRDY} )</td>
<td>20</td>
<td></td>
<td></td>
<td>ns</td>
<td></td>
<td>8, 9, 16</td>
</tr>
<tr>
<td>( t_w ) (L)</td>
<td>( MR ) Pulse Width</td>
<td>50</td>
<td></td>
<td></td>
<td>ns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( t_w ) (L)</td>
<td>( RTL ) Pulse Width</td>
<td>60</td>
<td></td>
<td></td>
<td>ns</td>
<td></td>
<td>14</td>
</tr>
</tbody>
</table>
96LS488

Ordering Information

<table>
<thead>
<tr>
<th>Order Code</th>
<th>Package</th>
<th>Temperature</th>
</tr>
</thead>
<tbody>
<tr>
<td>96LS488DC</td>
<td>Ceramic DIP</td>
<td>Commercial</td>
</tr>
<tr>
<td>96LS488DM</td>
<td>Ceramic DIP</td>
<td>Military</td>
</tr>
</tbody>
</table>

*C = 0°C to +70°C
M = -55°C to +125°C

Package Outlines

48-Pin Side-brazed Package

Notes
All dimensions in inches bold and millimeters in parentheses
Pin material is nickel, gold-plated Kovar or Alloy 42
Cap is Kovar or Alloy 42
Base is ceramic
Cavity size is .310 x .310 (7.874 x 7.874)
Package weight is 7.7 grams
APPENDIX E

DATA PROCESSING BY HP85 COMPUTER
The IMT General Purpose Data Logger was designed to be able to output data, via the IEEE-488 port, to an external computer for data processing or display purposes.

In this instance the HP85 desk top computer was chosen because it is small, portable and has an integral CRT Display, Printer and Data Cassette as well as an optional IEEE-488 (HPIB) interface.

Two programs have been written (in Basic) which are controlled by menu prompted function keys:

Test (Autost):

This program is automatically loaded and run if the program tape is inserted before the HP85 is switched on. The program displays the following "header" for 3 seconds:

```
* ***************************************************** *
* I M T DATA LOGGER                                  *
* ***************************************************** *
* #PROGRAM NAME: TEST(Autost) *                      *
* #BY: MIKE GARDENER                                 *
* #DESCRIPTION: THIS PROGRAM TESTS*                  *
* #THE IMT LOGGER/HP85 IEEE 488 *                    *
* #INTERFACE OR ALLOWS THE HP85 *                    *
* #TO PLOT REAL TIME DATA [SCOPE]*                   *
* #DATE: 02/01/85                                    *
* ***************************************************** *
```

and then the select option menu and description
SELECT OPTION!

TEST: HP85/DATA LOGGER IEEE-488 INTERCONNECTION TEST

SCOPE: DISPLAYS PLOT OF REAL TIME DATA

READ: LOADS 'IMTLOG' PROGRAM (DATA READ & PLOT)

-------------

TEST  SCOPE  READ

At this point the operator should ensure that the Data Logger is connected to the HP85 HPIB interface cable and is powered up.

TEST: [K1]

When selected the CRT will display:

IEEE TEST MODE
WAITING FOR DATA!

The Data Logger IEEE-488 port test procedure (7.5) should now be followed which will result in either

IEEE TEST PASS
being displayed for 2 seconds before reverting to the "Select Option" display.

OR

This display is accompanied by a beep sound and both the message and beep are repeated until the operator aborts the program.

If the test fails then the Data Logger / HP85 interconnection is suspect. Locate and rectify the cause and repeat TEST1 to make sure that it is now correct.

SCOPE: [K2]

This routine allows a pre-selected Data Logger channel to be displayed on the HP85 CRT in real-time. The axes and labels are initially set up and then

WAITING FOR DATA!

is displayed.

If a log is now initiated or the Data Acquisition Test procedure (7.4) is followed then the real-time data will be plotted onto the CRT.
Note: 1. This program is limited to a maximum scan rate of 125 msec and will only display the first 240 points.

2. The Data Logger IEEE MODE must be ON.

3. In the Data Acquisition Test mode the scan rate is governed by the speed at which the HP85 can receive and process the data i.e. scan rate is maximum for this configuration.

READ: [K4]

This option will load and run the second program, "IMTLOG", and display the following header for 3 seconds.

******************************************************************************
* I.M.T DATA LOGGER *
******************************************************************************

*PROGRAM NAME:IMTLOG *
*BY:MIKE GARDENER *

*DESCRIPTION:THIS PROGRAM READS* AND PLOTS DATA TRANSFERED FROM* THE I.M.T. DATA LOGGER RECORD */ BY RECORD: <240 SCANS MAX.> */ */ *DATE:03/01/85 */ 
******************************************************************************
and then

```
SELECT OPTION!

*HD-CPY: HARD COPY OF PLOT

PLOT: PLOTS DATA VS TIME FOR A SINGLE (SELECTABLE) CHANNEL

TXFR-D: TRANSFERS DATA FROM IMT LOGGER TO HP85

T-DUMP: DUMPS DATA TO HP TAPE

T-READ: READS DATA FROM HP TAPE

---------------------------------------------------

HD-CPY PLOT T-DUMP TXFR-D
```

This is an option select description and menu for five new functions:

TXFR-D: [K4]

This function allows data to be transferred from the Data Logger to the HP85.

WAITING FOR DATA FROM LOGGER

is displayed until data is received.

The Data Logger DATA PLAYBACK procedure (7.8) with IEEE Mode ON, should now be followed.

After selecting the record to be read the [READ] key will initiate the Data Logger cassette read and IEEE data transfer. The HP85 will indicate the transfer status by displaying

TRANSFER OF DATA IN PROGRESS

Press [CONT] TO CONTINUE
Otherwise

TRANSFER OF DATA COMPLETE

is displayed for 2 seconds before the returning to the "Select Option" display.

PLOT: [K2]

This routine allows a channel to be selected (limits 1 - 27) when

SELECT CHANNEL TO BE PLOTTED:
1 TO 27

is displayed. The desired number should be keyed in and then [END LINE]. The program now plots this channel's data onto the CRT.
After the plot is complete the program returns to the "Select Option" display.

**HD-CPY [K1]**

Allows a CRT dump to the integral HP85 printer of the last PLOT.

**T-DUMP [K3]**

Allows the data transferred by TXFR-D to be dumped to the HP85 data tape. It first prompts:

```
INSERT HP DATA TAPE, PRESS [CONT]
```

and then

```
ENTER FILE NAME
```

at which point a 6 character (max) file name should be entered. After the data is recorded the program returns to the "Select Option" display.

**T-READ [K7]**

Reads back data from a HP85 data tape which has previously been recorded by T-DUMP.

```
INSERT HP DATA TAPE, PRESS [CONT]
```

is displayed and then

```
ENTER FILE NAME.
```

The file name used in T-DUMP must be entered which allows the data to be read back into the HP85 memory.

The program then returns to the "Select Option" display.
HP85 COMPUTER
PROGRAM LISTINGS
10 DISP "**********************************************************************
20 DISP "# I.M.T DATA LOGGER #
30 DISP "**********************************************************************
40 DISP "#" 
50 DISP "#PROGRAM NAME:TEST(Autost) #
60 DISP "#" 
70 DISP "#BY:MKE GARDENER #
80 DISP "#" 
90 DISP "#DESCRIPTION:THIS PROGRAM TESTS#" 
100 DISP "#THE IMT LOGGER/HP85 IEEE 488 #
110 DISP "#INTERFACE OR ALLOWS THE HP85 #
120 DISP "#TO PLOT REAL TIME DATA (SCOPE)" 
130 DISP "#" 
140 DISP "#DATE:02/01/85 #
150 DISP "**********************************************************************
160 WAIT 3000
170 OPTION BASE 1
180 ON KEY 1,"TEST" GOTO 530
190 ON KEY 2,"SCOPE" GOTO 330
200 ON KEY 4,"READ" GOTO 760
210 CLEAR
220 KEY LABEL
225 DISP "SELECT OPTION!"
230 DISP "**********************************************************************
240 DISP "#TEST: HP85/DATA LOGGER IEEE-488 INTERCONNECTION TEST#"
250 DISP "#SCOPE: DISPLAYS PLOT OF REAL TIME DATA#"
260 DISP "#READ: LOADS 'IMTLOG' PROGRAM (DATA READ & PLOT)#"
270 DISP "**********************************************************************
280 GOTO 290
290 GOTO 290
300 : "************************************************************************
310 : # SCOPE #
320 : "************************************************************************
330 CLEAR
340 SCALE -30,250,-500,10500
350 YAXIS 5000,10,0,240
360 YAXIS 0,1000,0,10000
370 MOVE -30,10000 @ LABEL "5v"
380 MOVE -20,5000 @ LABEL "0v"
390 MOVE -30,0 @ LABEL "-5v"
400 CLEAR
410 DISP "WAITING FOR DATA!"
420 ENTER 710 USING ",,W" ; A
430 MOVE 0,A
440 FOR I=1 TO 239
450 ENTER 710 USING ",,W" ; A
460 DRAW I,A
470 NEXT I
480 WAIT 2000
490 GOTO 210
500 : "************************************************************************
510 : # IEEE TEST #
520 : "************************************************************************
530 DIM AH$(264)
540 IBUFFER A$ 
550 A,B=0
560 CLEAR
570 DISP "IEEE TEST MODE"
580 DISP & DISP
590 DISP " WAITING FOR DATA!"
600 BEEP
610 TRANSFER 710 TO A# FHS
620 CLEAR
630 DISP A#
640 FOR I=1 TO 256
650 A=NUM(A#(I))+A
660 B=B+1-1
670 NEXT I
680 BEEP
690 DISP & DISP
700 IF A=B THEN DISP "IEEE TEST PASS" WAIT 2000 GOTO 210
710 IF A>B THEN DISP "IEEE TEST FAILURE!!" BEEP
720 GOTO 710
730 ! ****************************************
740 ! READ I
750 ! ****************************************
760 CHAIN "IMTLOG"
770 END
PROGRAM NAME: IMTLOG

By: MIKE GARDEMER

DISPATCH

PROGRAM "IMTLOG"

DATE: 03/01/85

DISPATCH "HD-CPY: HARD COPY OF PLOT"

DISPATCH "TXFR-D: TRANSFERS DATA FROM IMT LOGGER TO HP85"

DISPATCH "T-READ: READS DATA FROM HP TAPE"

DISPATCH "T-DUMP: DUMPS DATA TO HP TAPE"

DISPATCH "PLOT: PLOTS DATA vs TIME FOR A SINGLE (SELECTABLE) CHANNEL"

SELECT OPTION!

DISPATCH "HD-CPY: HARD COPY OF PLOT"

DISPATCH "TXFR-D: TRANSFERS DATA FROM IMT LOGGER TO HP85"

DISPATCH "T-READ: READS DATA FROM HP TAPE"

DISPATCH "T-DUMP: DUMPS DATA TO HP TAPE"

DISPATCH "PLOT: PLOTS DATA vs TIME FOR A SINGLE (SELECTABLE) CHANNEL"

DISPATCH "HD-CPY: HARD COPY OF PLOT"

DISPATCH "TXFR-D: TRANSFERS DATA FROM IMT LOGGER TO HP85"

DISPATCH "T-READ: READS DATA FROM HP TAPE"

DISPATCH "T-DUMP: DUMPS DATA TO HP TAPE"

DISPATCH "PLOT: PLOTS DATA vs TIME FOR A SINGLE (SELECTABLE) CHANNEL"

DISPATCH "HD-CPY: HARD COPY OF PLOT"

DISPATCH "TXFR-D: TRANSFERS DATA FROM IMT LOGGER TO HP85"

DISPATCH "T-READ: READS DATA FROM HP TAPE"

DISPATCH "T-DUMP: DUMPS DATA TO HP TAPE"

DISPATCH "PLOT: PLOTS DATA vs TIME FOR A SINGLE (SELECTABLE) CHANNEL"

DISPATCH "HD-CPY: HARD COPY OF PLOT"

DISPATCH "TXFR-D: TRANSFERS DATA FROM IMT LOGGER TO HP85"

DISPATCH "T-READ: READS DATA FROM HP TAPE"

DISPATCH "T-DUMP: DUMPS DATA TO HP TAPE"

DISPATCH "PLOT: PLOTS DATA vs TIME FOR A SINGLE (SELECTABLE) CHANNEL"

DISPATCH "HD-CPY: HARD COPY OF PLOT"

DISPATCH "TXFR-D: TRANSFERS DATA FROM IMT LOGGER TO HP85"

DISPATCH "T-READ: READS DATA FROM HP TAPE"

DISPATCH "T-DUMP: DUMPS DATA TO HP TAPE"

DISPATCH "PLOT: PLOTS DATA vs TIME FOR A SINGLE (SELECTABLE) CHANNEL"

DISPATCH "HD-CPY: HARD COPY OF PLOT"

DISPATCH "TXFR-D: TRANSFERS DATA FROM IMT LOGGER TO HP85"

DISPATCH "T-READ: READS DATA FROM HP TAPE"

DISPATCH "T-DUMP: DUMPS DATA TO HP TAPE"

DISPATCH "PLOT: PLOTS DATA vs TIME FOR A SINGLE (SELECTABLE) CHANNEL"

DISPATCH "HD-CPY: HARD COPY OF PLOT"

DISPATCH "TXFR-D: TRANSFERS DATA FROM IMT LOGGER TO HP85"

DISPATCH "T-READ: READS DATA FROM HP TAPE"

DISPATCH "T-DUMP: DUMPS DATA TO HP TAPE"

DISPATCH "PLOT: PLOTS DATA vs TIME FOR A SINGLE (SELECTABLE) CHANNEL"

DISPATCH "HD-CPY: HARD COPY OF PLOT"

DISPATCH "TXFR-D: TRANSFERS DATA FROM IMT LOGGER TO HP85"

DISPATCH "T-READ: READS DATA FROM HP TAPE"

DISPATCH "T-DUMP: DUMPS DATA TO HP TAPE"

DISPATCH "PLOT: PLOTS DATA vs TIME FOR A SINGLE (SELECTABLE) CHANNEL"

DISPATCH "HD-CPY: HARD COPY OF PLOT"

DISPATCH "TXFR-D: TRANSFERS DATA FROM IMT LOGGER TO HP85"

DISPATCH "T-READ: READS DATA FROM HP TAPE"

DISPATCH "T-DUMP: DUMPS DATA TO HP TAPE"

DISPATCH "PLOT: PLOTS DATA vs TIME FOR A SINGLE (SELECTABLE) CHANNEL"
600 XAXIS 5000.1,0.240
610 YAXIS 0.1000,0.1000
620 MOVE -30.9750 @ LABEL "+5v"
630 MOVE -20.5000 @ LABEL "0v"
640 MOVE -30.000 @ LABEL "-5v"
650 MOVE -20.1050 @ LABEL "DATE:.,D$11.21,"/".D$11.21,"/".D$5.61
660 MOVE 120.10500 @ LABEL "TIME:.,T$11.21,"/".T$11.21,"/".T$5.61
670 MOVE -30.750 @ LABEL "SCANS:.",S
680 MOVE 70. -7500 @ LABEL "RATE:.",R
690 MOVE 150. -7500 @ LABEL "CHAN.1:"/"D1
700 MOVE 0,0
710 B2=DH2-1
720 FOR 1=1 TO 5
730 A=NUM(B$1(B2))#256+NUM(B$1(B2)+1))
740 A=INT(A/16#10000/4095)
750 DRAW 1,A
760 B2=B2+C$2
770 NEXT I
790 WAIT 2000 @ GOTO 230
800 CLEAR @ DISP "TAPE READ ERROR!!"
810 PAUSE
820 DISP @ DISP "PRESS [CONT] TO CONTINUE" @ GOTO 230
830 GRAPH 230
840 ! ****************************
850 !   HARD COPY   
860 ! ****************************
870 GRAPH 2 COPY @ GOTO 230
880 ! ****************************
890 !   HP TAPE DUMP   
900 ! ****************************
910 CLEAR @ DISP "INSERT HP DATA TAPE,PRESS [CONT]"
920 PAUSE
930 DISP "ENTER FILE NAME"
940 INPUT X$1.6]
950 CREATE X$.1.13056
960 ASSIGN X TO X$
970 PRINT# 1 : A$1.B$
980 ASSIGN# 1 TO X$1
990 GOTO 230
1000 ! ****************************
1010 !   HP TAPE READ   
1020 ! ****************************
1030 CLEAR @ DISP "INSERT HP DATA TAPE,PRESS [CONT]"
1040 PAUSE
1050 DISP "ENTER FILE NAME"
1060 INPUT X$[1.6]
1070 ASSIGN# 1 TO X$
1080 READ# 1 : A$1.B$
1090 ASSIGN# 1 TO X$
1100 D#=A$1.6]
1110 T#=A$17.123
1120 R=NUM(A$113])
1130 C=NUM(A$143])
1140 S=NUM(A$1153))#256+NUM(A$116])
1150 IF C<27 THEN GOTO 800
1160 GOTO 230
1170 END
APPENDIX F

TAPE HEADER and DATA FORMAT
TAPE HEADER & DATA FORMAT.

|----------------------|----------------------|---------------------|----------------------|-------------------------|

| No of SCANS [low byte] | CHAN #1 DATA SCAN #1 [high byte] | CHAN #1 DATA SCAN #1 [low byte] | CHAN #2 DATA SCAN #1 [high byte] | CHAN #2 DATA SCAN #1 [low byte] |

| CHAN #X DATA SCAN #1 [low byte] | CHAN #1 DATA SCAN #2 [high byte] | CHAN #1 DATA SCAN #2 [low byte] | CHAN #X DATA SCAN #Y [low byte] |

NOTE: X is determined by No of CHANS selected. Y is determined by No of SCANS selected.

DATE = 6 ASCII characters in DDMMYY format.
TIME = 6 ASCII characters in HHMMSS format.
SCAN RATE = 1 binary byte [limits 1-15]
No of CHANS = 1 binary byte [limits 1-32]
No of SCANS = 2 binary bytes [limits 1-9999]

DATA [high byte] = 8 msb's
DATA [low byte] = 4 lsb's & 4 trailing zeros
Component Lists, Component Overlays
and Schematic Diagrams
## 8085 C.P.U. Board

### Parts List

<table>
<thead>
<tr>
<th>Component</th>
<th>Value</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>10k</td>
<td>1/4 Watt 5%</td>
</tr>
<tr>
<td>R2</td>
<td>5k6</td>
<td>1/4 Watt 5%</td>
</tr>
<tr>
<td>R3</td>
<td>10M</td>
<td>1/4 Watt 5%</td>
</tr>
<tr>
<td>R4</td>
<td>1k</td>
<td>1/4 Watt 5%</td>
</tr>
<tr>
<td>R5</td>
<td>3k3</td>
<td>1/4 Watt 5%</td>
</tr>
<tr>
<td>R6</td>
<td>15k</td>
<td>1/4 Watt 5%</td>
</tr>
<tr>
<td>VR1</td>
<td>5k</td>
<td>20 Turn</td>
</tr>
<tr>
<td>VR1</td>
<td>15x4k7</td>
<td>Type 1615</td>
</tr>
<tr>
<td>C1</td>
<td>0,1μF</td>
<td>Tantalum</td>
</tr>
<tr>
<td>C2</td>
<td>22pF</td>
<td>Ceramic</td>
</tr>
<tr>
<td>C3</td>
<td>22pF</td>
<td>&quot;</td>
</tr>
<tr>
<td>C4</td>
<td>5nF</td>
<td>&quot;</td>
</tr>
<tr>
<td>C5-17</td>
<td>0,01μF</td>
<td>&quot;</td>
</tr>
<tr>
<td>D1</td>
<td>IN4148</td>
<td></td>
</tr>
<tr>
<td>D2,3</td>
<td>OA91</td>
<td></td>
</tr>
<tr>
<td>D4</td>
<td>3V Zenner</td>
<td>400mW</td>
</tr>
<tr>
<td>XTAL</td>
<td>4.194304 MHz</td>
<td></td>
</tr>
<tr>
<td>Ni Cad</td>
<td>3,6V 100mA hr</td>
<td></td>
</tr>
<tr>
<td>P2,3</td>
<td>4x7 Molex 0,1&quot; Wafer</td>
<td></td>
</tr>
<tr>
<td>P4</td>
<td>2x13 Molex 0,1&quot; Wafer</td>
<td></td>
</tr>
<tr>
<td>P1</td>
<td>DIN 41612 64 Pole Connector (A+C, 90°)</td>
<td></td>
</tr>
<tr>
<td>IC</td>
<td>Part Number</td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>IC1,5</td>
<td>74LS244</td>
<td></td>
</tr>
<tr>
<td>IC2</td>
<td>74LS14</td>
<td></td>
</tr>
<tr>
<td>IC3</td>
<td>8085</td>
<td></td>
</tr>
<tr>
<td>IC4</td>
<td>74LS74</td>
<td></td>
</tr>
<tr>
<td>IC6</td>
<td>76LS245</td>
<td></td>
</tr>
<tr>
<td>IC7,8</td>
<td>74LS373</td>
<td></td>
</tr>
<tr>
<td>IC9</td>
<td>74LS42</td>
<td></td>
</tr>
<tr>
<td>IC10</td>
<td>2716 or 2732 EPROM</td>
<td></td>
</tr>
<tr>
<td>IC11</td>
<td>6116 CMOS RAM</td>
<td></td>
</tr>
<tr>
<td>IC12</td>
<td>8205</td>
<td></td>
</tr>
<tr>
<td>IC13,14</td>
<td>74LS00</td>
<td></td>
</tr>
<tr>
<td>IC15</td>
<td>146818</td>
<td></td>
</tr>
<tr>
<td>IC16</td>
<td>8155</td>
<td></td>
</tr>
<tr>
<td>IC17</td>
<td>74LS02</td>
<td></td>
</tr>
<tr>
<td>IC18</td>
<td>LM3130</td>
<td></td>
</tr>
<tr>
<td>IC19</td>
<td>MC4066</td>
<td></td>
</tr>
</tbody>
</table>
### Parts List

<table>
<thead>
<tr>
<th>Part</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>4k7 0.25Watt 5%</td>
</tr>
<tr>
<td>R2</td>
<td>68k 0.25Watt 5%</td>
</tr>
<tr>
<td>R3</td>
<td>330k 0.25Watt 5%</td>
</tr>
<tr>
<td>R4</td>
<td>1k 0.25Watt 5%</td>
</tr>
<tr>
<td>R5</td>
<td>820k 0.25Watt 5%</td>
</tr>
<tr>
<td>C1</td>
<td>15μF 16V Tantalum</td>
</tr>
<tr>
<td>C2</td>
<td>0.1μF polycarbonate</td>
</tr>
<tr>
<td>C3</td>
<td>1.0μF 16V Tantalum</td>
</tr>
<tr>
<td>C4</td>
<td>0.47μF polycarbonate</td>
</tr>
<tr>
<td>C5</td>
<td>680pF ceramic</td>
</tr>
<tr>
<td>C6</td>
<td>0.047μF polycarbonate</td>
</tr>
<tr>
<td>T1</td>
<td>2N2222</td>
</tr>
<tr>
<td>S1</td>
<td>1x8 Molex socket</td>
</tr>
<tr>
<td>P1</td>
<td>2x7 Molex 0.1&quot; Wafer</td>
</tr>
<tr>
<td>8Ω</td>
<td>Mini Speaker (Kobishi)</td>
</tr>
<tr>
<td>IC1</td>
<td>74C922</td>
</tr>
<tr>
<td>IC2</td>
<td>7555 (CMOS)</td>
</tr>
<tr>
<td>4x4</td>
<td>Matrix Keyboard</td>
</tr>
<tr>
<td>DRAWN</td>
<td>GETEKEN</td>
</tr>
<tr>
<td>-------</td>
<td>---------</td>
</tr>
<tr>
<td>CHECKED</td>
<td>TITEL</td>
</tr>
<tr>
<td>NAGESIEN</td>
<td></td>
</tr>
<tr>
<td>APPROVED</td>
<td></td>
</tr>
<tr>
<td>GOEDGEKEUR</td>
<td></td>
</tr>
<tr>
<td>SCALE</td>
<td>DWG NO</td>
</tr>
<tr>
<td>SKAAL</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Diagram**: Electronic circuit diagram showing components such as IC1, C1, C2, C6, P1, R1, R2, R3, R5, and a speaker.
32 k memory addressable to 1 megabyte
- Interlaced RAM and EPROM in 2 k intervals
- Switch selectable page address
- Switch selectable 16 or 20-bit address option
- Switch selectable card address 0–32 k or 32–64 k
- Write protect switches for each of 4 k blocks
- Low or high data byte option for 8- and 16-bit CPUs
- Single 5 V supply

Circuit description*

All the bus signals used on the card are buffered and place a maximum of one low-power Schottky TTL load on the buses.

Page address selection
1 megabyte in 64 k pages
The most significant four bits of address (AB16 to AB19) go to one of the inputs of four exclusive OR-gates (IC22). The switches G, H, I and K select one of 16 64 k pages of memory. When AB16 to AB19 contain the address selected on the switches, the open collector outputs of the XOR-gates will all be high. The wired AND of all these outputs goes to switch E. This switch allows the selection of 16 or 20 bits of addressable memory. The pull-up resistor R6 will maintain a logic high when the switch is open. This line is the page-enabling line and goes to pins 10 of IC16, 9, 12, 13 of IC17 and 6 of IC19. Switches G, H, I and K correspond to AB16, AB17, AB18 and AB19 respectively. When switch E is open the page switches are irrelevant.

Card address selection
32 k at 0–32 k or 32–64 k
The card address is selected by switch F on pin 13 of IC21. When the switch is closed the output of IC21 pin 11 is low when AB15 is low; when switch F is open pin 11 will be low when AB15 is high. This line partly enables ICs 19 and 20.

Chip select lines
One of 16 2 k chips
The AB14 line, when low, enables pin 4 of IC19 and, when high, pin 6 of IC20. The 4 to 16 line decoder, ICs 19 and 20, decode address buses 11 to 14 to one of 16 active low chip select lines. IC0 is the lowest and IC15 is the highest addressable memory device.

Memory address
0 to 2 k
Address buses 0 to 10 go through buffers directly to the address pins of each of the 16 memories in the array.

Memory devices
The 32 k memory array may consist of a random mixture of RAM and EPROM. The EPROMs are of Intel type 2716 or TI 2716. The RAM may be Hitachi type 6116, Toshiba 5516 or OKI 2128.

Memory read
The output enable pins of the memory devices will go active low when the card is correctly addressed during a memory read cycle. The active low memory read line is buffered and inverted by IC16 pin 5; pins 3 and 4 are held high by the output of IC17 pin 6. The read line is qualified by the card address selection. This line (pin 8 IC16) controls the direction of the bidirectional data buffer.

Memory write
The active low memory write line (MW) is buffered and inverted by IC17 pin 1; pin 2 is held high by pin 6 of IC17. The memory write signal is again inverted by IC18. Each output of the four NAND gates is connected to an 8 k block of memory. The outputs are controlled by four dual in line switches, accessible at the card panel. These switches A to D are the memory write protect switches. When open, the switches allow the memory write line on VppWR (pin 21 of the memory devices) to become active low during a memory write cycle. In all other cases Vpp will remain at a logical high level.

The specifications are that Vpp must be low prior to Vpp going low. Somehow the data sheets led to the popular belief that Vpp may never be low. For the unbelievers, the write protect switch may be used — writing to an EPROM with Vpp being low is not harmful to the device.

Data buffers
The data may be placed on the low or high byte of the data buses by installing a 74LS245 in either socket IC25 or IC26. When IC25 is installed the data will appear on DB 0 to 7, and when IC26 is installed the data will go to DB 8 to 15. If both ICs are installed, identical 8-bit data will appear on the low and high bytes. The direction of the data path is always from the bus to the memory card, only the memory read operation will direct the data towards the data bus. The bidirectional data buffer will go into "tri-state" when either MEMDIS is low or the page address is incorrect.

Using two 32 k memory modules with the same address, but one with a low data byte buffer and the other with a high data byte buffer, the combination will form a 16 bit memory of 32 k.

*The circuit diagram is overleaf

Options summary
A, B, C and D switches
Write protect for RAM in 8 k blocks
- closed = write protected.

E switch
AB16—AB19 disable
- open = 16-bit address
- closed = 20-bit address.

F switch
Card address selection
closed = 0 to 32 k
- open = 32 to 64 k.

G, H, I and K switches
Page address selection
G = AB16
H = AB17
I = AB18
K = AB19

High data byte
Install only IC26 in buffer sockets.

Low data byte
Install only IC25 in buffer sockets.

Hexadecimal chip address table

<table>
<thead>
<tr>
<th>Hex</th>
<th>Decimal</th>
<th>Chip address</th>
</tr>
</thead>
<tbody>
<tr>
<td>IC0</td>
<td>0</td>
<td>0 to 7FF</td>
</tr>
<tr>
<td>IC1</td>
<td>1</td>
<td>800 to FFF</td>
</tr>
<tr>
<td>IC2</td>
<td>2</td>
<td>1000 to 17FF</td>
</tr>
<tr>
<td>IC3</td>
<td>3</td>
<td>1800 to 1FFF</td>
</tr>
<tr>
<td>IC4</td>
<td>4</td>
<td>2000 to 27FF</td>
</tr>
<tr>
<td>IC5</td>
<td>5</td>
<td>2800 to 2FFF</td>
</tr>
<tr>
<td>IC6</td>
<td>6</td>
<td>3000 to 37FF</td>
</tr>
<tr>
<td>IC7</td>
<td>7</td>
<td>3800 to 38FF</td>
</tr>
<tr>
<td>IC8</td>
<td>8</td>
<td>4000 to 47FF</td>
</tr>
<tr>
<td>IC9</td>
<td>9</td>
<td>4800 to 4FFF</td>
</tr>
<tr>
<td>IC10</td>
<td>10</td>
<td>5000 to 57FF</td>
</tr>
<tr>
<td>IC11</td>
<td>11</td>
<td>5800 to 5FFF</td>
</tr>
<tr>
<td>IC12</td>
<td>12</td>
<td>6000 to 67FF</td>
</tr>
<tr>
<td>IC13</td>
<td>13</td>
<td>6800 to 6FFF</td>
</tr>
<tr>
<td>IC14</td>
<td>14</td>
<td>7000 to 77FF</td>
</tr>
<tr>
<td>IC15</td>
<td>15</td>
<td>7700 to 7FFF</td>
</tr>
</tbody>
</table>

Component list

<table>
<thead>
<tr>
<th>Component</th>
<th>Type</th>
<th>Memory devices</th>
</tr>
</thead>
<tbody>
<tr>
<td>IC0 to 15</td>
<td>Memory devices</td>
<td></td>
</tr>
<tr>
<td>EPROM</td>
<td>Intel 2716, TI 2516</td>
<td></td>
</tr>
<tr>
<td>RAM</td>
<td>Hitachi 6116, Toshiba 5516, OKI 2128</td>
<td></td>
</tr>
<tr>
<td>IC 16</td>
<td>74LS10</td>
<td></td>
</tr>
<tr>
<td>IC 17, 18</td>
<td>74LS00</td>
<td></td>
</tr>
<tr>
<td>IC 19, 20</td>
<td>74LS138</td>
<td></td>
</tr>
<tr>
<td>IC 21</td>
<td>74LS86</td>
<td></td>
</tr>
<tr>
<td>IC 22</td>
<td>74LS136</td>
<td></td>
</tr>
<tr>
<td>IC 23, 24</td>
<td>74LS244</td>
<td></td>
</tr>
<tr>
<td>IC 25, 26</td>
<td>74LS245</td>
<td></td>
</tr>
<tr>
<td>R1 to R5</td>
<td>2.2 kΩ All resistors are 0.25 W</td>
<td></td>
</tr>
<tr>
<td>R7 to R10</td>
<td>2.2 kΩ</td>
<td></td>
</tr>
<tr>
<td>All decoupling capacitors 22 μF</td>
<td>16 V</td>
<td></td>
</tr>
<tr>
<td>Switches</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A to D</td>
<td>4 DIL make/break</td>
<td></td>
</tr>
<tr>
<td>E, F</td>
<td>2 DIL make/break</td>
<td></td>
</tr>
<tr>
<td>G to K</td>
<td>4 DIL make/break</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>R1</td>
<td>56Ω</td>
<td>1/4 Watt 5%</td>
</tr>
<tr>
<td>R2</td>
<td>1k</td>
<td>1/4 Watt 5%</td>
</tr>
<tr>
<td>R3,4</td>
<td>10k</td>
<td>1/4 Watt 1%</td>
</tr>
<tr>
<td>VR1</td>
<td>5k</td>
<td>20 Turn</td>
</tr>
<tr>
<td>C1,5,7</td>
<td>0,1μF Polycarbonate</td>
<td></td>
</tr>
<tr>
<td>C8</td>
<td>10μF Ceramic</td>
<td></td>
</tr>
<tr>
<td>C2,4,6</td>
<td>470μF 25V Electrolytic</td>
<td></td>
</tr>
<tr>
<td>C3,8,10</td>
<td>10μF 35V Tantalum</td>
<td></td>
</tr>
<tr>
<td>D1,2</td>
<td>IN4006</td>
<td></td>
</tr>
<tr>
<td>BR1,2</td>
<td>1A</td>
<td></td>
</tr>
<tr>
<td>T1</td>
<td>2x9,5V @ 290 mA secondary, 220V primary. Type 4892.</td>
<td></td>
</tr>
<tr>
<td>T2</td>
<td>2x17V @ 85 mA secondary, 220V primary. Type 4295.</td>
<td></td>
</tr>
<tr>
<td>IC1</td>
<td>LM 301</td>
<td></td>
</tr>
<tr>
<td>IC2</td>
<td>LM 7915 CT</td>
<td></td>
</tr>
<tr>
<td>IC3</td>
<td>LM 340T15 or LM 7815 CT</td>
<td></td>
</tr>
<tr>
<td>IC4</td>
<td>LM 7905 CT.</td>
<td></td>
</tr>
</tbody>
</table>
POWER SUPPLY INTERCONNECTION BLOCK DIAGRAM

A1 C1 A32 C32
+5V 0V

+5V 0V

-5V

OLV 15 PSU

DATA LOGGER

PSU BOARD

CONNECTED TO BACK PLANE AT A/D BOARD

VME TAPE DRIVE VIA P3 ON MFE TAPE BOARD

A GND

+15V -15V

0V

220 V

250mA

100mA

IMD-D034-0014-08
**IEEE 488 Interface Board**

**Parts List**

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>2k7</td>
<td>1/4 Watt 5%</td>
</tr>
<tr>
<td>R2,6</td>
<td>10k</td>
<td>1/4 Watt 5%</td>
</tr>
<tr>
<td>R3-5</td>
<td>270k</td>
<td>1/4 Watt 5%</td>
</tr>
<tr>
<td>C1-3</td>
<td>0,01μF polycarbonate</td>
<td></td>
</tr>
<tr>
<td>C4,5</td>
<td>33pF ceramic</td>
<td></td>
</tr>
<tr>
<td>C6</td>
<td>0,01μF polycarbonate</td>
<td></td>
</tr>
<tr>
<td>C7,15</td>
<td>10μF 35V Tantulum</td>
<td></td>
</tr>
<tr>
<td>C8-14</td>
<td>0,01μF polycarbonate</td>
<td></td>
</tr>
<tr>
<td>P1</td>
<td>2x13 Molex 0,1&quot; Wafer</td>
<td></td>
</tr>
<tr>
<td>P2</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>P3</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>P4</td>
<td>DIN 41612 64 Pole connector (A+C, 90°)</td>
<td></td>
</tr>
<tr>
<td>IR1</td>
<td>15x1k Type 1615</td>
<td></td>
</tr>
<tr>
<td>IR2</td>
<td>15x4k7 Type 1615</td>
<td></td>
</tr>
<tr>
<td>XTAL</td>
<td>10,000 MHz</td>
<td></td>
</tr>
<tr>
<td>SW1,3</td>
<td>8 Pole D.I.L. switch</td>
<td></td>
</tr>
<tr>
<td>SW2</td>
<td>4 Pole D.I.L. switch</td>
<td></td>
</tr>
<tr>
<td>IC1</td>
<td>96LS488</td>
<td></td>
</tr>
<tr>
<td>IC2</td>
<td>74LS04</td>
<td></td>
</tr>
<tr>
<td>IC3,4,10</td>
<td>74LS245</td>
<td></td>
</tr>
<tr>
<td>IC5,6</td>
<td>74LS157</td>
<td></td>
</tr>
<tr>
<td>IC7,8</td>
<td>74LS02</td>
<td></td>
</tr>
<tr>
<td>IC9</td>
<td>8255</td>
<td></td>
</tr>
<tr>
<td>IC11</td>
<td>DMB136</td>
<td></td>
</tr>
<tr>
<td>IC12</td>
<td>74LS241</td>
<td></td>
</tr>
<tr>
<td>IC13</td>
<td>74LS00</td>
<td></td>
</tr>
</tbody>
</table>
### IEEE Address

<table>
<thead>
<tr>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>A5</td>
<td>A4</td>
<td>A3</td>
<td>A2</td>
<td>A1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Set for addr. 10 ON = '1'

### Auto Handshake

<table>
<thead>
<tr>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>STRDY</td>
<td>TXRDY</td>
<td>RXRDY</td>
</tr>
</tbody>
</table>

### IEEE Mode Select

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>M3</td>
<td>M2</td>
<td>M1</td>
<td>M0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Set for talker/listner mode ON = '1'

### SA Bus Address

<table>
<thead>
<tr>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>NC</td>
<td>NC</td>
<td>A7</td>
<td>A6</td>
<td>A5</td>
<td>A4</td>
<td>A3</td>
<td>A2</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Set for 60H ON = '0'

---

**Title:** IEEE 488 Switch Settings

**Drg. No.:**

**File Name:** D 034

**Drgn. For:**

**Drgn. By:** M Gardener

**Date:** 08-10-86

**Scale:**

**Sheet:** 1 of 1

---

**IMT**
MFE Tape Interface Board
Parts List

R1  27k \( \frac{1}{4} \) Watt 5%
R2  300Ω \( \frac{1}{4} \) Watt 5%

IR1, 2  15 x 4k7 Type 1615

C1  10\( \mu \)F 16V Tantalum
C2-6  0.01\( \mu \)F Ceramic

T1  2N2907

P1  DIN 41612 64 Pole connector (A+C, 80°)
P2  2x20 Molex 0.1" Wafer
P3  10 Pole Stocko Pin Housing. Horizontal mounting.

S1  8 Pole D.I.L. switch

DC to DC convertor (5V to -12V)  CB 3811

1C1  74LS241
1C2  DM 8136
1C3  74LS00
1C4  8255
1C5  74LS245
SA BUS ADDRESS

<table>
<thead>
<tr>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>NC</td>
<td>NC</td>
<td>A7</td>
<td>A6</td>
<td>A5</td>
<td>A4</td>
<td>A3</td>
<td>A2</td>
</tr>
<tr>
<td>X</td>
<td>X</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

SET FOR 44H
ON = '0'

TITLE: MFE TAPE INTERFACE
SWITCH SETTINGS

DRG. NO.: D 034
DRN. FOR: M. GARDENER
DATE 08-10-86

FILE NAME: D 034
SCALE: SHEET 1 OF 1
### 32 Channel Data Acquisition Board

#### Parts List

<table>
<thead>
<tr>
<th>Part</th>
<th>Description</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1,2</td>
<td>1k 1/4 Watt 5%</td>
<td></td>
</tr>
<tr>
<td>VR1,2</td>
<td>100Ω 20 turn</td>
<td></td>
</tr>
<tr>
<td>VR3</td>
<td>10k 20 turn</td>
<td></td>
</tr>
<tr>
<td>IR1</td>
<td>15x4k7 Type 1615</td>
<td></td>
</tr>
<tr>
<td>C1</td>
<td>10μF Tantalum</td>
<td></td>
</tr>
<tr>
<td>C2</td>
<td>1000pF Polystyrene</td>
<td></td>
</tr>
<tr>
<td>C3,4</td>
<td>0,1μF Polycarbonate</td>
<td></td>
</tr>
<tr>
<td>C5-10</td>
<td>0,01μF Ceramic</td>
<td></td>
</tr>
</tbody>
</table>

- **P1**: DIN 41612 64 Pole connector (A+C, 90°)
- **P2**: Molex 0,1" Wafer
- **P3**: 34 way, 3m, Ribbon cable connector
- **S1**: 8 Pole D.I.L. switch

- **IC1**: 74LS241
- **IC2**: DM1836
- **IC3,4**: 74LS00
- **IC5**: AD574
- **IC6**: 8255
- **IC7**: 74LS245
- **IC8**: AD582
- **IC9,10**: AD5760 or 1H6116
SA BUS ADDRESS

<table>
<thead>
<tr>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>NC</td>
<td>NC</td>
<td>A7</td>
<td>A6</td>
<td>A5</td>
<td>A4</td>
<td>A3</td>
<td>A2</td>
</tr>
</tbody>
</table>

**SET FOR 50H**

**ON = '0'**
DATA ACQUISITION BOARD
ANALOG SUPPLY & DECOUPLING CIRCUIT

DRAWN M. GARDENER
GETEKEN 11/84

CHECKED NAGESIEN
APPROVED GOEDGEKEUR

SCALE SKAAL

DRAWN M. GARDENER
GETEKEN 11/84

CHECKED NAGESIEN
APPROVED GOEDGEKEUR

SCALE SKAAL

IMD-0034-0005-08

IMT