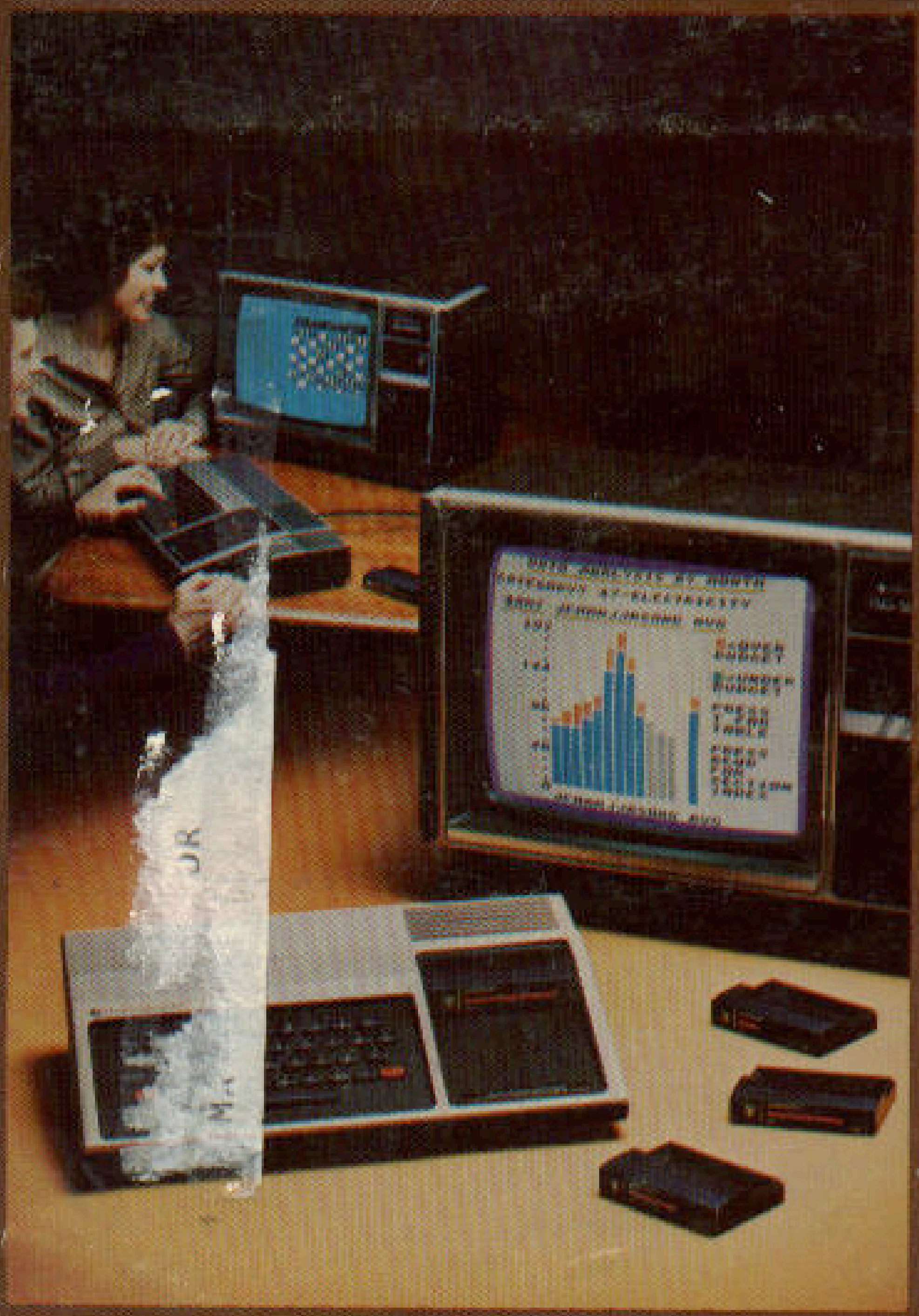
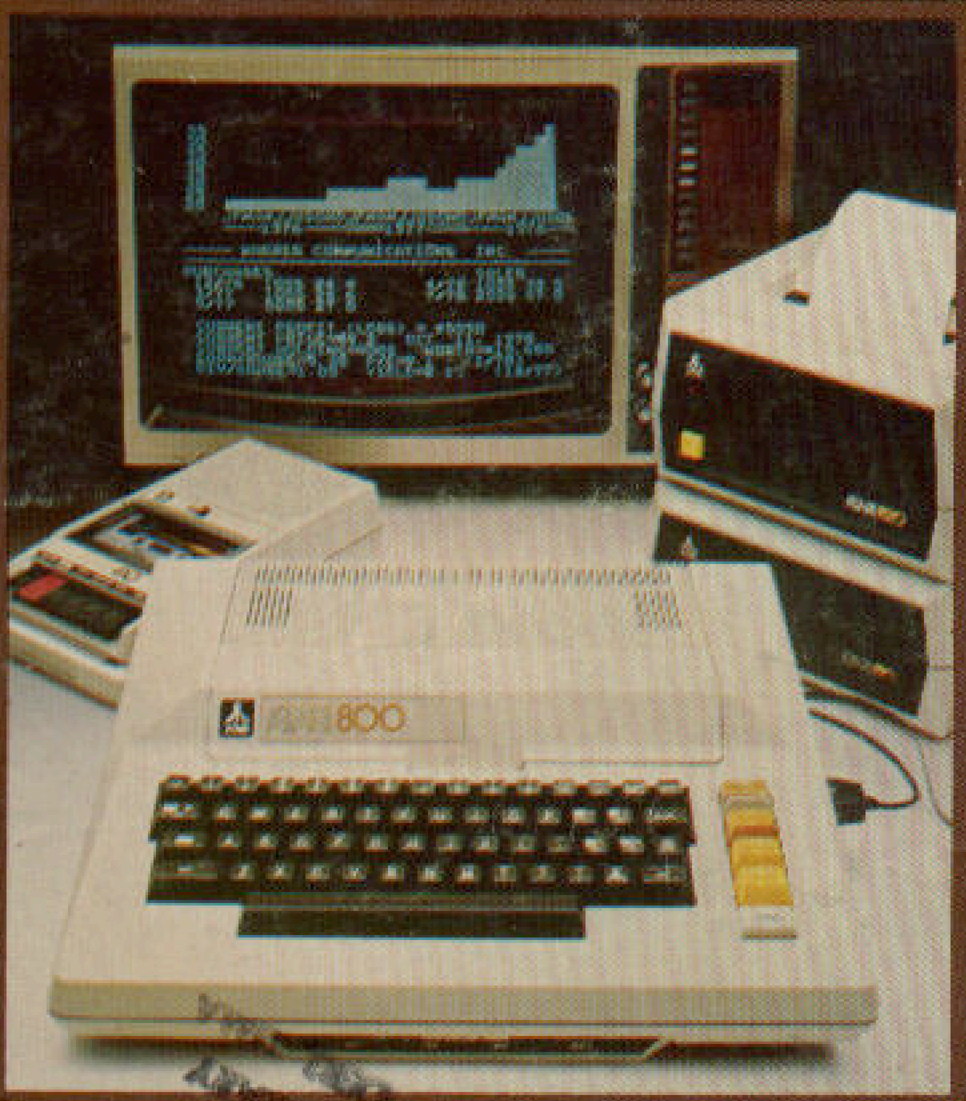
# INTERFACE AGE™

COMPUTING FOR HOME AND BUSINESS APPLICATIONS

VOLUME 4, ISSUE 8 AUGUST/SEPTEMBER 1979 $2.00
CANADA/MEXICO $2.50 INTERNATIONAL $4.50

## TEXAS INSTRUMENTS, ATARI, MATTEL USHER IN THE NEW AGE OF HOME COMPUTERS





## NTS
mini series
## UNIT 6

## MICRO AIDS FOR THE HANDICAPPED

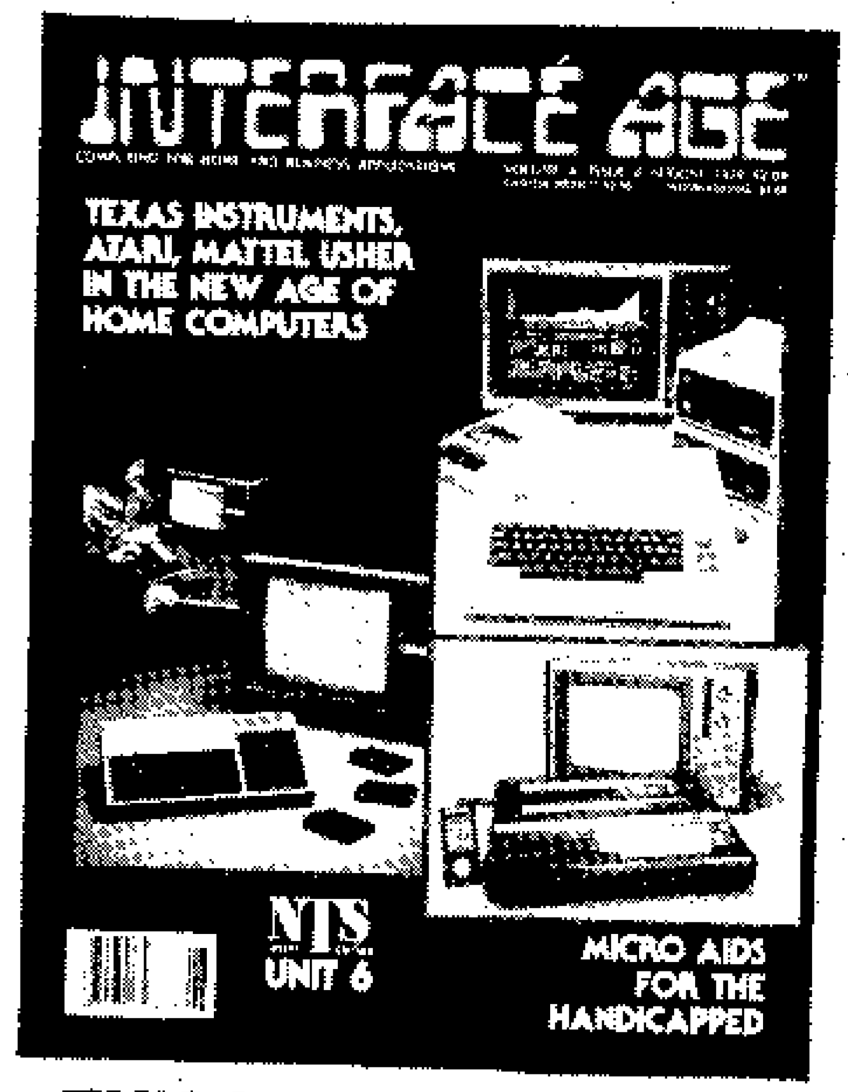
08
0 74470 65683

VOL. 4, ISSUE 8
SINCE DECEMBER 1975
AUGUST/SEPTEMBER 1979

# INTERFACE AGE™

## COMPUTING FOR HOME AND BUSINESS APPLICATIONS






### THIS MONTH'S COVER
Cover photos courtesy of Texas Instruments, Atari and Mattel Electronics.

16704 Marquardt Avenue, Cerritos, CA 90701
(213) 926-9544

**PUBLISHERS**
PUBLISHER **ROBERT S. JONES**
EXECUTIVE PUBLISHER **NANCY A. JONES**

**ADMINISTRATION**
PUBLICATION DIRECTOR **MIKE ANTICH**
PUBLICATION ASSISTANT **DORIS RIOPEL**
ACCOUNTING SUPERVISOR **KAY SOTO**
ACCOUNTING ASSISTANTS **MARY ANN LOWER, SHIRLEY MAZENKO**
CIRCULATION SECRETARY **JETTA ANDREONI**
CIRCULATION ASSISTANTS **CHARLOTTE SEVEDGE, DOTTIE SMITH**

**EDITORIAL**
EDITOR-IN-CHIEF **CARL WARREN**
SENIOR EDITOR SOUTHEASTERN REGION **BILL TURNER**
EDITOR **TERRY COSTLOW**
ASSOCIATE EDITORS **JIM SCHREIER, HENRY DAVIS**
NORTHWESTERN REGIONAL EDITOR **ADAM OSBORNE, PhD**
NORTHEASTERN REGIONAL EDITOR **ROGER C. GARRETT**
SYSTEMS EDITOR **TOM FOX**
SOFTWARE EDITOR **ALAN R. MILLER, PhD**
CONTRIBUTING EDITORS **R. H. DISTLER, R. W. BEMER**
STAFF REPORTER **BETSY GILBERT**
EDITORIAL ASSISTANT **KATHY NEHIRA**
**Editorial Correspondence**
Direct all correspondence to the appropriate editor at INTERFACE AGE Magazine, P.O. Box 1234, Cerritos, CA 90701.

**PRODUCTION**
PRODUCTION MANAGER **DAVE ANDERSON**
ASSISTANT PRODUCTION MANAGER **SHELLEY WRIGHT**
ART DIRECTOR **FINO ORTIZ**
ARTIST **SAMANTHA LEE**
TYPOGRAPHER **MELODY A. MARTENS**

**ADVERTISING**
NEW ENGLAND REGION **DICK GREEN**
7 Lincoln St., Wakefield, MA 01880 (617) 245-9105
EASTERN REGION **PAUL CONFREY**
20 Community Pl., Morristown, NJ 07960 (201) 267-3032
MIDWEST REGION **AL GRAVENHORST & STEVE SKINNER**
5901 N. Cicero Ave., Chicago, IL 60646 (312) 545-8621
WESTERN REGION **BRUCE BERKEY & ZACH BOVINETTE**
P.O. Box 4566, Pasadena, CA 91106 (213) 795-7002
COMPUTER RETAIL STORES NATIONWIDE
CALL (213) 795-7002 (COLLECT)

**INTERFACE AGE EUROPE**
DIRECTOR EUROPEAN OPERATIONS **H.L. GROHMANN**
Birkenweg 43 8051 Unterzolling West Germany
Telephone 8167-366

**FOREIGN CIRCULATION**
JAPAN CIRCULATION **KAZUHIKO NISHI**
ASCII Publishing, 305 HITORIO, 5-6-4 Minami Aoyama, Minato-ku, Tokyo 107 Japan
Telephone (03) 407-4910
UNITED KINGDOM CIRCULATION **VINCENT COEN**
L.P. Enterprise, 313 Kingston Road, Ilford, Essex, England IG1IPJ
Telephone 01-553-1001
FRANCE CIRCULATION **ROLAND HESSE**
Euro Computer Shop Paris 92, Rue Saint-Lazare, 75009 Paris, France
Telephone Paris 281-29-03 or 281 29 16, Telex REPTC 270339 F
WESTERN CANADA CIRCULATION **BRIAN J.J. WIEBE**
Kitronic, 25236 26th Avenue RR5, Aldergrove, B.C. VOX 1A0
Telephone (604) 856-2301
EASTERN CANADA CIRCULATION **LIZ JANEK**
RS 232 Ltd., 186 Queen Street West, Toronto, Ontario M5V 1Z1
Telephone (416) 598-0288
AUSTRALIA CIRCULATION **R. J. HOESS**
Electronic Concepts Pty. Ltd., 52-58 Clarence Street, Sydney NSW 2000
Telephone 29 3753
INDONESIA, SINGAPORE, MALAYSIA CIRCULATION **LEE MILES**
The Computer Centre, Pte. Ltd., 5366, Woh Hup Complex, Beach Rd. Singapore 7
Telephone 293-2630

MEMBER OF THE WESTERN PUBLICATIONS ASSOCIATION
AMERICAN SOCIETY OF BUSINESS PRESS EDITORS

INTERFACE AGE Magazine, published monthly by McPheters, Wolfe & Jones, 16704 Marquardt Ave., Cerritos, CA 90701. Subscription rates: U.S. $18.00, Canada/Mexico $20.00, all other countries $28.00. Make checks payable in U.S. funds drawn on a U.S. bank. Opinions expressed in by-lined articles do not necessarily reflect the opinion of this magazine or the publisher. Mention of products by trade name in editorial material or advertisements contained herein in no way constitutes endorsement of the product or products by this magazine or the publisher. Circulation Department, (213) 926-9540.

INTERFACE AGE Magazine is catalogued in the Library of Congress, Classification No. QA75.5.155. ISSN Publication No. 0147-2992. Membership in Audit Bureau of Circulations applied for.
POSTMASTER: Please send change of address form 3579 and undelivered copies to INTERFACE AGE Magazine, 16704 Marquardt Ave., Cerritos, CA 90701. Second-class postage paid at Artesia, California 90701 and at additional mailing offices.

# THE TI-99/4

# HOME COMPUTER

## Texas Instruments' Answer to Tom Swift's Wonderful Magic Machine

By Carl Warren

Imagine the idea of going to your local computer store or electronics shop and purchasing a box that contains a computer. One which the salesman tells you doesn't require any special knowledge or sophisticated housing or even the requirement that you own a soldering iron.

Sound impossible? Maybe three years ago or even as recent as late '77 or early '78, the idea of buying a computer like an appliance was more than most computerists could possibly comprehend. The time has now come for that to happen, with the introduction of the long-awaited computer from Texas Instruments, the TI 99/4.

The tiny computer is housed in an eye-pleasing case that contains the keyboard, a slot for the special ROM program packages and a cassette interface. The rest of the system is made up of a 13-inch color monitor designed specifically for the 99/4 system, not just a modified television set.

The basic system also includes built-in, full floating point 13-digit expanded BASIC which meets current ANSI standards. An equation calculator is also part of the system software.

When the packing box is first opened, the new owner is immediately exposed to three important documents designed to enhance the use of the system.

The first document is a "read me first," written and illustrated in such a manner to make setting up the computer as goof proof as possible.

Next is found a book prepared by Bob Albrecht to help the novice computer user, *Beginners Guide to BASIC*. This book, like all Albrecht books, begins at the start of what you need to know and carries you on through the fascinating world of BASIC programming.

To round out the basic literature included with the machine is the "User's Reference Guide," which contains further advanced details on the TI BASIC, plus information on servicing and warranty — or answers to all those questions you didn't know to ask but need to.

TI has provided a tremendous amount of useful documentation with the basic machine, but they have not stopped there. Working with Herbert Peckham and McGraw-Hill, a more advanced BASIC programming book has been prepared. It will be available later this year.

### INSIDE THE 99/4

The insides of the 99/4 are probably the most exciting aspect of the machine. Based around the powerful 9900 MPU, the system has available 16K of user accessible RAM and 26K ROM in the basic system.

The 99/4 is currently the only unit on the market that is fully shielded to prevent radiation. Also, to provide the clarity in the video display and to avoid possible FCC restrictions, TI elected to make the display monitor part of the total package.

The real secret of the system lies in the solid state software modules. These paks enhance the basic ROM and RAM that are inherent to the system by providing expansion up to 72K, and also serving as the container for high level programs.

Another built-in function of the tiny system supplies music and sound effects. They are generated by the handling of the output of the system. This of course means that some exciting games or applications can be created by the user, or supplied by TI on the software modules that bring the screen to life.

Speaking of bringing the screen to life, the system provides 16-color graphics capability, which gives the illusion of covering the entire gray scale and color spectrum. Due to the engineering that went into the graphics of the system, and the use of interrupts and DMA type processing, several functions appear to take place at the same time. Consequently, it is not unusual to hear the machine playing music and see several animated characters on the screen at the same time.

## MORE ON THE SOFTWARE

Texas Instruments has developed a method of distributing consumer-oriented software on solid state modules. That is software designed to be used by anyone with or without an understanding of computers.

**It is not unusual to hear the machine playing music and see several animated characters on the screen at the same time.**

Each module was designed after careful research into the game or working application to ensure that the computer would correctly handle user response and in all cases give the correct answers or perform the next function. Each module contains anywhere from one to several ROM chips that have been microprogrammed with the given application.

The solid state modules are not designed to be programmed by the user but are to provide them with high quality reliable software. The modules are planned to sell in the $20 to $70 price range, depending upon the application. To aid dealers who want to carry the system, TI offers a phone assistance service at (214) 234-0692.

According to company officials, the solid state approach was taken to give users value and reliability for their program purchases. Currently 15 modules are available from TI on applications ranging from football to statistics, with more planned for release early in 1980.

Because the TI 99/4 is designed not only to fit the average man's pocketbook — selling in the $1,000 range — but also provide computer power to non-computer people, TI teamed with one of the major game companies so all members of the family could use it. Milton Bradley has gone to great lengths to provide high level support for the machine in the form of games.

The games from Milton Bradley are adaptations of their famous tried and true board games of: YAHTZEE, HANGMAN, and CONNECT FOUR. They have also added a new game called ZEROZAP, which would never work well on a board but is exciting on a computer.

All the Milton Bradley games take advantage of the graphics and sound abilities of the 99/4 and as a result that spirit of excitement is further increased while playing the games.

Although Milton Bradley is only planning to market these games for the next several months, they are ready to release other exciting modules as the machine finds its way into the family circle.

Milton Bradley and Texas Instruments both view the 99/4 as the obvious extension of man's instinct to find new and different ways of entertainment. Because of this, both companies have and are expending a great deal of effort into the research needed to design popular software modules.

## MORE ON THE HORIZON

There is even more to the 99/4 than high resolution graphics and exciting software packages. There is a voice. With TI's experience in the realm of speech synthesis, evidence Speak and Spell, it was only natural that their entry to the home computer market would have this ability.

This talking ability is based on the tone generation ability of the basic machine, plus the use of an add-on synthesizer that will have plug-in vocabulary modules. The idea is to


carry the human engineering of the system to the nth degree. Using the speech function of the system, a user can write a program that will not only display a question but ask it at the same time, eliciting both a visual and verbal response.

The possibilities based around these functions alone are endless and could quite conceivably open up a market for enterprising entrepreneurs who wish to service some sector of the handicapped populace.

To round out the system, TI is also planning a later introduction of an RS232 adaptor, disks and a printer. The remote control paddles are available now as system options.

## MAKING IT WORK

The TI 99/4 has to be one of the most simplistic computers ever invented. All the user does is turn it on, insert a software module and tap a key on the keyboard. The machine responds with quick visual displays, or with music. In a few months voice will be added.

The user is led through every function that is available, both in the running application, and also through the supplied system documentation and module documents.

What makes the TI 99/4 really unique is that it was designed for functional purposes by some of the world's best electronic engineers, then designed for human operation by everyday folks.□

### Table 1. TI-99/4 Specifications

**DISPLAY**

13" Color monitor
16 Colors
5' Video/sound cable

**CONSOLE**

**CPU:** 9900 Family, 16-bit microprocessor, plus 256-byte scratchpad RAM.
**Memory:** Total combined memory capacity: 72K bytes. Internal ROM memory supplied: 256K bytes. External ROM memory: (Solid State Software™ Command Modules) Up to 30K bytes each. RAM memory supplied: 16K bytes.
**Keyboard:** 40-key staggered Qwerty, full travel. Overlay for second functions.
**Sound:** 5 octaves, 3 simultaneous tones plus noise generator. From 110 Hz to beyond 40,000 Hz.
**Power:** 110V, 60Hz, 20W. Wall mounted console transformer, UL listed 8' power cord.
**I/O:** Composite video and audio output for monitor. Interface for up to 2 audio cassettes. 44-pin peripheral connector — up to 3 peripherals attached to system. System memory and address signals available at peripheral connector. Mini-earphone jack. Remote control interface.
**Built-in Software:** 14K byte BASIC interpreter. Internal Graphics Language interpreter, not user accessible. Equation calculator. Internal 4.4K byte monitor, not user accessible.
**Size:** 25.9 x 38.1 x 7.1 cm (10.2 x 15.0 x 2.5 in.)
**Weight:** Less than 2.3 kg. (5 lbs.)

**CONSOLE TECHNOLOGY (Detail Description)**

**CPU Chip (NMOS):** TMS9900 16-bit microprocessor. Minicomputer instruction set including hardware multiply and divide. Architecture with 16 general registers. Can address up to 64K bytes of memory. 4 interrupt lines.
**Video Display Processor Chip (NMOS):** Controls display memory and generates composite video signal. 24 lines of 32 characters with 8x8 dot resolution. Provides sixteen colors: white, gray, magenta, light yellow, yellow, light red, medium red, dark red, cyan, light blue, blue, light green, medium green, dark green, black, transparent. Provides 32 sets of 8 characters each with different foreground/background colors. Addresses up to 16K bytes of RAM for CPU or display.
**Sound Controller Chip ($I^2L$):** 3 voices with 5 octave musical resolution. 15 bit programmable noise source. 100 mW audio drive with 30 db control in 2db steps.
**Solid State Software Command Modules.** Up to 30K bytes PMOS ROM. Up to 8K bytes NMOS ROM. Simple plug-in module.

**PERIPHERALS**

**Remote Controls:** 2 controllers. Cable: twin cable with single connector, 4' long. 8 direction control — up, down, left, right and diagonals. Fire button. Controls designed for use with either hand.
**Solid State Speech™ Synthesizer.** Uses Speak 'N Spell™ technology. Approx. 250 words in unit. Plug-in vocabulary expansion. Hundreds of words can be added. Interfaces via I/O port.
**RS-232 Interface:** Provides interface to RS-232 type computer peripherals such as printer, plotters, digitizers, D/A converters, terminals, and data tablets. Dual port, two standard DB-25 connectors.
**Recorder Interface:** 1 or 2 standard audio cassette units (not included) interface through a cassette interface. Cable: twin cable with single connector, 2' long. Each record recorded twice for 600 baud effective recording rate. Checksum added to end of each double recorded record. Operator interaction fully prompted. Provides mass non-volatile program/data memory storage.

**Three Month Limited Warranty:** On TI console, monitor and Command Module hardware.

# A Versatile Memory Test for the TMS9900

By Tom Morris

One of the many necessary pieces of software for a computing system is the memory test. This software, along with other diagnostics, provides the means of locating a faulty component within the computer system.

Memory problems may manifest themselves in many ways. An addressing error may go unnoticed for an extended period of time, and when it finally makes itself known, may be interpreted as a software *bug*. A random bit dropped or added may produce the same results, or produce a program that *crashes* for no apparent reason. These problems may go unnoticed or lived with until too late, when a valuable data file is accidentally destroyed or partially overwritten.

Program flow can be seen in the flowchart (Figure 1). Startup initializes the necessary constants and reads in the range of memory to be placed under test (steps 1-4). Step 5 presets the memory under test to zero. Step 6 is performed for each location tested and allows the program to be interrupted so the operator may regain control of the system without having to do a restart.

Step 7 validates that the location under test is still zero, a failure here would indicate an addressing problem. If an error is encountered, the error reporting subroutine is called and the necessary information is given to the operator (ADDRESS, CONTENTS, and what the location SHOULD BE).

Steps 9-11 perform a walking bit on test, again all errors are reported as they occur. The remainder of the tests are performed in a similar manner by steps 13-16, 17-19 and 20-23.

When all of the specified memory has been tested, the message 'END OF PASS XXXX' is printed on the console signifying the end of a successful test. If, appended to this message is ', TOTAL ERRORS = YYYY', then at some time during the test errors have been encountered. This feature was added to allow a terminal to be taken off-line, and at a later time returned to on-line. This procedure may be used to save paper and still know if any errors occurred during the test sequence.

The object portion of Listing 1 was loaded and executed beginning at location #5000 (# indicates HEX).

The first request is to enter the start and end address of the memory to be placed under test. The program then cycles through all of the tests and announces a pass completion (Listing 2).

In my system, the memory from #6000 to #7FFF can be write protected, this feature was used to test the memory diagnostic.

Lines 6-17 of Listing 2 show the type of error reporting used, as the memory is occasionally placed in a protected state. Finally a 'BREAK' is issued to regain program control. Again a block of memory is tested. This time the terminal is taken off-line for an extended period of time. When it is finally returned to an on-line state, it can be seen that 604 passes without an error have been made.

This program, while being compact, provides a thorough workout of the memory under test and should provide enough clues to pinpoint the more common memory problems. □

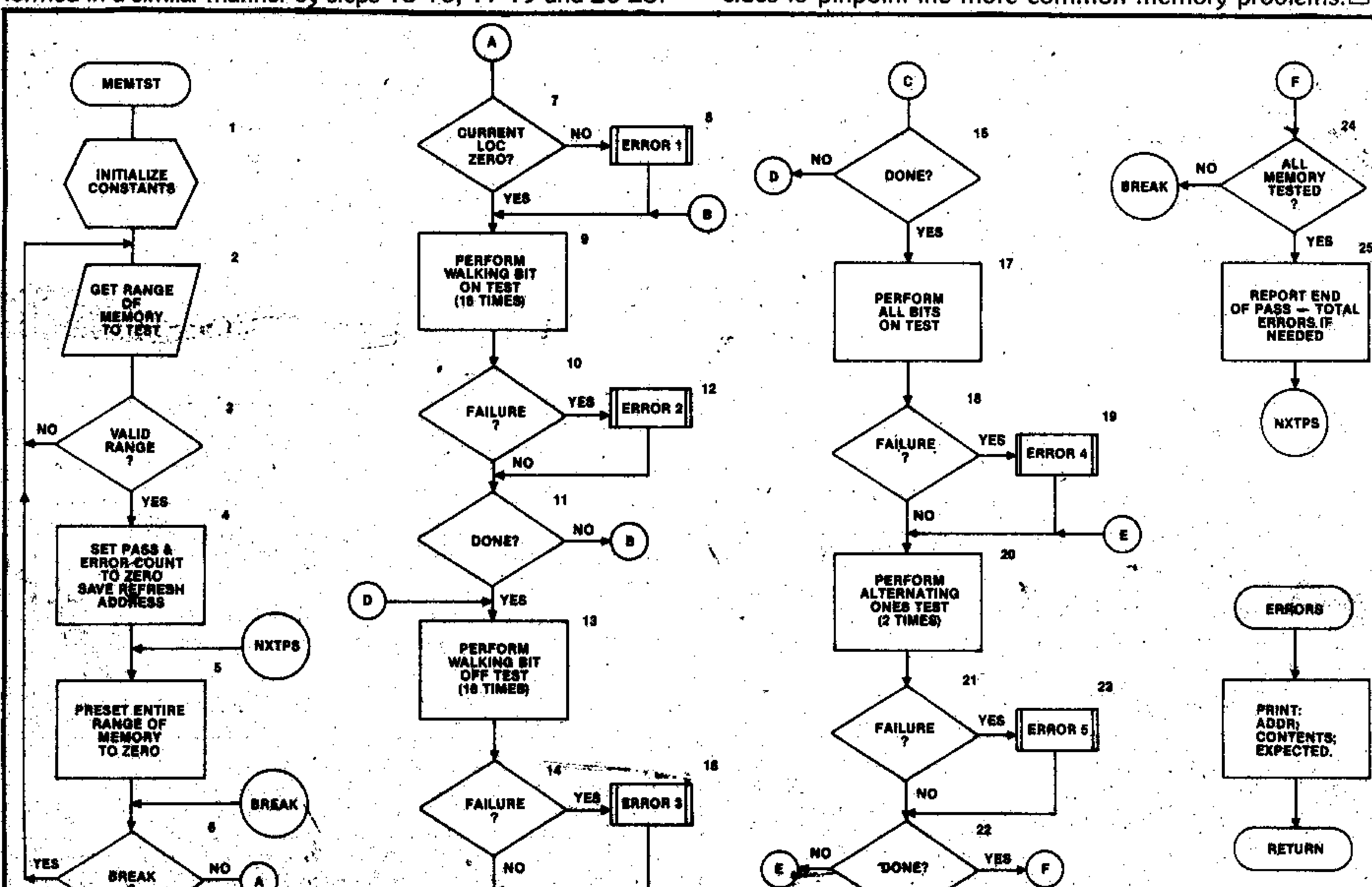



Figure 1. Flowchart

# LISTING 1

```
                   TITL 'MODIFIED COOKS MEMORY TEST FOR THE TMS9900'
0000          MEM1DT IDT
              *
              *   THIS PROGRAM PERFORMS A MODIFIED
              *   VERSION OF COOK'S MEMORY TEST
              *   THIS VERSION WRITTEN BY:
              *       TOM G. MORRIS
              *       861 ST. MARY AVE
              *       SAN LEANDRO, CA 94577
              *
              *   DEFINE THE ENTRY POINT AND REGISTERS
              *
0000                 RORG *
                     DEF  MEMTST,ENDTST
0000                 DREG
              *
              *   DEFINE THE EXTERNAL LINKAGE
              *   TO THE TECHNICO MONITOR (REV. 2)
              *
FE18          RDNUM  EQU  >FE18          ;READ HEX PARAMETERS
FDBA          TYPEWD EQU  >FDBA          ;PRINT R5 AS HEX NUMBER
              *
              *   DEFINE OPERATIONAL EQUIVALENCES
              *
0000          PNTR   EQU  0              ;SCRATCH POINTER
0001          CA     EQU  1              ;CURRENT ADDRESS
0002          EA     EQU  2              ;ENDING ADDRESS
0003          SA     EQU  3              ;STARTING ADDRESS
0006          PASS   EQU  6              ;PASS COUNTER
0007          SCR    EQU  7              ;SCRATCH REGISTER
0008          TEMP   EQU  8              ;TEMP COMPARISON
0009          ERRS   EQU  9              ;TOTAL # OF ERRORS
000A          TEN    EQU  10             ;BCD CONVERSION DIVISOR
000C          CRUBAS EQU  12             ;CRU BASE REGISTER
FFFE          WDMSK  EQU  >FFFE          ;WORD BOUNDARY MASK
0000          TTYI   EQU  0              ;TTY INPUT BIT
0000          MYWS   EQU  0              ;USE INT. VECTOR LOC. FOR WS
              *
              *   ENTRY TO THE PROGRAM
              *
0000          MEMTST EQU  $
0000 0300 0000       LIMI 0              ;DISABLE INTERRUPTS
0004 02E0 0000       LWPI MYWS           ;SET WORKSPACE POINTER
0008 04CC            CLR  CRUBAS         ;PRESET CRU BASE TO ZERO
000A 020A 000A       LI   TEN,TEN        ;PRESET DIVISOR FOR PASS COUNT
000E          RESTRT EQU  $
000E 0200 013E       LI   PNTR,MESS0     ;RESTARTS ARE MADE HERE
0012 06A0 012A       BL   @TYPE          ;PRINT SIGN ON MESSAGE
0016 0205 0004       LI   R5,4           ;READ TWO HEX NUMBERS
001A 06A0 FE18       BL   @RDNUM
001E 0241 FFFE       ANDI CA,WDMSK       ;FORCE ADDRESSES TO
0022 0242 FFFE       ANDI EA,WDMSK       ;A WORD BOUNDARY
0026 8081            C    CA,EA          ;VERIFY ADDRESSES
0028 1BF2            JH   RESTRT         ;RESTART IF BAD VALUE
002A 04C6            CLR  PASS           ;SET PASS COUNT TO ZERO
002C 04C9            CLR  ERRS           ;SET ERROR COUNT TO ZERO
002E C0C1            MOV  CA,SA          ;SAVE REFRESH ADDRESS
0030          NXTPS  EQU  $
0030 C043            MOV  SA,CA          ;REFRESH STARTING ADDRESS
0032          CLEAR  EQU  $
0032 04D1            CLR  *CA            ;PRESET MEMORY TO ZERO
0034 05C1            INCT CA             ;BUMP ADDRESS
0036 1302            JEQ  CLEAR1         ;HANDLE WRAP-AROUND
0038 8081            C    CA,EA          ;TEST FOR ALL OF MEMORY
003A 12FB            JLE  CLEAR          ;NO, CLEAR SOME MORE
003C          CLEAR1 EQU  $
003C C043            MOV  SA,CA          ;REFRESH THE CURRENT
                                          ADDRESS
              *
              *   BREAK: CHECK TO SEE IF THE TERMINAL
              *          BREAK KEY IS DEPRESSED.
              *          IF SO, RESTART THE PROGRAM
              *
003E          BREAK  EQU  $
003E 1F00            TB   TTYI           ;CHECK FOR A BREAK
0040 1303            JEQ  TEST1          ;NO, CONTINUE
0042          BREAK1 EQU  $
0042 1F00            TB   TTYI           ;YES, WAIT FOR IT TO
0044 16FE            JNE  BREAK1          END
0046 10E3            JMP  RESTRT         ;GO DO A RESTART
              *
              *   TEST1: VERIFIES THAT THE CURRENT
              *          LOCATION IS STILL ZERO
              *          IF IT IS NOT ZERO, THEN
              *          THERE IS MOST LIKELY A
              *          MEMORY ADDRESSING ERROR
              *
0048          TEST1  EQU  $
0048 04C7            CLR  SCR            ;COMPARE AGAINST ZERO
004A C211            MOV  *CA,TEMP       ;GET RESULT
004C 8207            C    SCR,TEMP
004E 1302            JEQ  TEST2
0050 06A0 00D0       BL   @ERROR1        ;REPORT THE ERROR
              *
              *   TEST2: PERFORMS A WALKING
              *          BIT TEST (SINGLE BIT ON)
              *
0054          TEST2  EQU  $
0054 0207 8000       LI   SCR,>8000      ;SET MSB
0058          TEST2A EQU  $
0058 C447            MOV  SCR,*CA        ;STORE THE PATTERN
005A C211            MOV  *CA,TEMP       ;GET IT BACK
005C 8207            C    SCR,TEMP       ;DID IT TAKE ?
005E 1302            JEQ  TEST2B
0060 06A0 00D6       BL   @ERROR2        ;NO, REPORT IT
0064          TEST2B EQU  $
0064 0B17            SRC  SCR,1          ;ROTATE THE PATTERN
0066 17F8            JNC  TEST2A         ;TEST NEXT BIT IF NO
                                          CARRY
              *
              *   TEST3: PERFORMS A WALKING
              *          BIT TEST (SINGLE BIT OFF)
              *
0068          TEST3  EQU  $
0068 0207 7FFF       LI   SCR,>7FFF      ;ALL BITS BUT MSB
006C          TEST3A EQU  $
006C C447            MOV  SCR,*CA        ;STORE THE PATTERN
006E C211            MOV  *CA,TEMP       ;GET IT BACK
0070 8207            C    SCR,TEMP       ;DID IT TAKE ?
0072 1302            JEQ  TEST3B
0074 06A0 00D6       BL   @ERROR3        ;NO, REPORT IT
0078          TEST3B EQU  $
0078 0B17            SRC  SCR,1          ;GET THE NEW PATTERN
007A 18F8            JOC  TEST3A         ;TEST AGAIN IF CARRY
              *
              *   TEST4: SET ALL BITS TO ONE
              *
007C          TEST4  EQU  $
007C 0707            SETO SCR            ;SET COMPARE TO ONES
007E C447            MOV  SCR,*CA        ;STORE IT
0080 C211            MOV  *CA,TEMP       ;GET IT BACK
```

```
0082 8207             C     SCR,TEMP         ;ARE THEY THE SAME
0084 1302             JEQ   TEST5            ;YES, CONTINUE
0086 06A0 00DC        BL    @ERROR4          ;NO, REPORT THE ERROR
                *
                *  TEST5: CHECK ALTERNATING ONES PATTERN
                *
008A            TEST5  EQU  $
008A 0207 AAAA        LI    SCR,>AAAA        ;GET PATTERN
008E            TEST5A EQU  $
008E C447             MOV   SCR,*CA          ;STORE THE PATTERN
0090 C211             MOV   *CA,TEMP         ;GET IT BACK
0092 8207             C     SCR,TEMP         ;DO THEY AGREE ?
0094 1302             JEQ   TEST5B
0096 06A0 00E2        BL    @ERROR5          ;NO, REPORT THE ERROR
009A            TEST5B EQU  $
009A 0B17             SRC   SCR,1            ;GET NEW PATTERN
009C 17F8             JNC   TEST5A           ;DONE IF CARRY
                *
                *  AT THIS POINT, INCREMENT THE CURRENT
                *  ADDRESS AND CHECK FOR END OF PASS.
                *  IF SO, ANNOUNCE IT, ELSE TEST NEXT WORD
                *
009E            CONT   EQU  $
009E 05C1             INCT  CA               ;WORD ADDRESSING
00A0 1302             JEQ   ENDPS            ;HANDLE THE WRAP-AROUND CASE
00A2 8081             C     CA,EA            ;TEST FOR END OF PASS
00A4 12CC             JLE   BREAK            ;GO TEST NEXT WORD
00A6            ENDPS  EQU  $
00A6 0586             INC   PASS             ;PASS COMPLETED
00A8 0200 01A7        LI    PNTR,EOP         ;ANNOUNCE IT
00AC 06A0 012A        BL    @TYPE
00B0 C3C6             MOV   PASS,R15         ;GET PASS COUNT
00B2 06A0 0112        BL    @BCD             ;CONVERT PASS TO BCD
00B6 06A0 FDBA        BL    @TYPEWD          ;PRINT NUMBER OF PASSES
00BA C3C9             MOV   ERRS,R15         ;GET NUMBER OF ERRORS
00BC 13B9             JEQ   NXTPS            ;NONE, FORGET IT
00BE 0200 01B4        LI    PNTR,TOTERR      ;GET TOTAL ERRORS PNTR
00C2 06A0 0132        BL    @TYPE2           ;PRINT MSG
00C6 06A0 0112        BL    @BCD             ;CONVERT ERRORS TO BCD
00CA 06A0 FDBA        BL    @TYPEWD          ;PRINT IT
00CE 10B0             JMP   NXTPS
                *
                *  THIS IS THE ERROR REPORTING SECTION.
                *  MULTIPLE ENTRIES ARE;
                *  ERROR1: ALL ZEROES FAILURE
                *  ERROR2: WALKING BIT ON FAILURE
                *  ERROR3: WALKING BIT OFF FAILURE
                *  ERROR4: ALL ONES FAILURE
                *  ERROR5: ALTERNATING ONES FAILURE
                *
00D0            ERROR1 EQU  $
00D0 0200 015A        LI    PNTR,ERR1        ;GET ERROR MSG
00D4 1008             JMP   RETURN           ;JOIN COMMON LOGIC
00D6            ERROR2 EQU  $
00D6            ERROR3 EQU  $
00D6 0200 0168        LI    PNTR,ERR23       ;SAME ERROR MSG FOR BOTH
00DA 1005             JMP   RETURN           ;JOIN COMMON LOGIC
00DC            ERROR4 EQU  $
00DC 0200 0176        LI    PNTR,ERR4
00E0 1002             JMP   RETURN           ;JOIN COMMON LOGIC
00E2            ERROR5 EQU  $
00E2 0200 0184        LI    PNTR,ERR5        ;GET ERR MSG
00E6            RETURN EQU  $
00E6 C34B             MOV   R11,R13          ;SAVE RETURN ADDRESS
00E8 0589             INC   ERRS             ;UPDATE ERROR COUNT
```

```
                *
0168 524F 5441  ERR23  TEXT 'ROTATING AT: '
016C 5449 4E47
0170 2041 543A
0174 20
0175 00                BYTE 0
                *
0176 414C 4C20  ERR4   TEXT 'ALL ONES AT: '
017A 4F4E 4553
017E 2041 543A
0182 20
0183 00                BYTE 0
                *
0184 414C 5420  ERR5   TEXT 'ALT ONES AT: '
0188 4F4E 4553
018C 2041 543A
0190 20
0191 00                BYTE 0
                *
0192 2C20 4953  IS     TEXT ', IS: '
0196 3A20
0198 00                BYTE 0
                *
0199 2C20 5348  SB     TEXT ', SHOULD BE: '
019D 4F55 4C44
01A1 2042 453A
01A5 20
01A6 00                BYTE 0
                *
01A7 454E 4420  EOP    TEXT 'END OF PASS '
01AB 4F46 2050
01AF 4153 5320
01B3 00                BYTE 0
                *
01B4 2C20 544F  TOTERR TEXT ', TOTAL ERRORS = '
01B8 5441 4C20
01BC 4552 524F
01C0 5253 203D
01C4 20
01C5 00                BYTE 0
                *
01C6 0D         RET    BYTE >0D                  ;RETURN
01C7 0A         LINF   BYTE >0A                  ;LINE FEED
                *
01C7            ENDTST EQU  $-1
01C8                   END
```

```
 0112 BCD      003E BREAK    0042 BREAK1   0001 CA       0032 CLEAR
 003C CLEAR1  *009E CONT     000C CRUBAS   0002 EA       00A6 ENDPS
 013C ENDT    *01C7 ENDTST   01A7 EOP      015A ERR1     0168 ERR23
 0176 ERR4     0184 ERR5     00D0 ERROR1   00D6 ERROR2   00D6 ERROR3
 00DC ERROR4   00E2 ERROR5   0009 ERRS     0192 IS       01C7 LINF
*0000 MEMIDT  *0000 MEMTST   013E MESS0    0000 MYWS     0118 NEXTD
 0030 NXTPS    0006 PASS     0000 PNTR    *0000 R0      *0001 R1
*000A R10      000B R11     *000C R12      000D R13      000E R14
 000F R15     *0002 R2      *0003 R3       0004 R4       0005 R5
*0006 R6      *0007 R7      *0008 R8      *0009 R9       FE18 RDNUM
 000E RESTRT   01C6 RET      00E6 RETURN   0003 SA       0199 SB
 0007 SCR      0008 TEMP     000A TEN      0048 TEST1    0054 TEST2
 0058 TEST2A   0064 TEST2B  *0068 TEST3    006C TEST3A   0078 TEST3B
*007C TEST4    008A TEST5    008E TEST5A   009A TEST5B   01B4 TOTERR
 0000 TTYI     012A TYPE     0132 TYPE2    FDBA TYPEWD   FFFE WDMSK
```

```
00EA 06A0 012A        BL    @TYPE          ;PRINT ERROR MSG
00EE C141             MOV   CA,R5          ;PRINT ADDRESS OF ERROR
00F0 06A0 FDBA        BL    @TYPEWD
00F4 0200 0192        LI    PNTR,IS        ;PRINT WHAT IT IS
00F8 06A0 0132        BL    @TYPE2
00FC C148             MOV   TEMP,R5
00FE 06A0 FDBA        BL    @TYPEWD
0102 0200 0199        LI    PNTR,SB        ;PRINT WHAT IT SHOULD BE
0106 06A0 0132        BL    @TYPE2
010A C147             MOV   SCR,R5
010C 06A0 FDBA        BL    @TYPEWD
0110 045D             B     *R13           ;RETURN TO CALLER
                *
                *  CONVERT THE BINARY VALUE IN R15
                *  TO A BCD VALUE IN R5
                *
0112            BCD   EQU   $
0112 04C5             CLR   R5             ;CLEAR SUM
0114 0204 0004        LI    R4,4           ;NUMBER OF DIGITS
0118            NEXTD EQU   $
0118 04CE             CLR   R14            ;CLEAR MSWD
011A 3F8A             DIV   TEN,R14        ;DIVIDE R14,15 BY TEN
011C 0945             SRL   R5,4           ;MAKE ROOM FOR DIGIT
011E 0ACF             SLA   R15,12         ;POSITION NEW DIGIT
0120 A14F             A     R15,R5         ;ADD IN DIGIT
0122 C3CE             MOV   R14,R15        ;GET NEW DIVIDEND
0124 0604             DEC   R4             ;CHECK FOR FOUR DIGITS
0126 16F8             JNE   NEXTD          ;NO, DO NEXT DIGIT
0128 045B             B     *R11           ;RETURN
                *
                *  TYPE: PRINT THE STRING POINTED
                *        TO BY (PNTR) ON A NEW LINE.
                *        TERMINATE ON NULL
                *
                *  TYPE2: DON'T ISSUE A NEW LINE
                *
012A            TYPE  EQU   $
012A 2CA0 01C6        OUT   @RET           ;ISSUE NEW LINE
012E 2CA0 01C7        OUT   @LINF
0132            TYPE2 EQU   $
0132 D410             MOVB  *PNTR,*PNTR    ;TEST FOR NULL
0134 1303             JEQ   ENDT
0136 2C90             OUT   *PNTR          ;NO, PRINT THE CHAR
0138 0580             INC   PNTR           ;UPDATE POINTER
013A 10FB             JMP   TYPE2          ;PRINT NEXT CHAR
013C            ENDT  EQU   $
013C 045B             B     *R11           ;RETURN TO CALLER
                *
                *  STRING STORAGE AREA
                *
013E 5354 4152  MESS0 TEXT  'START,END ADDRESS TO TEST ?'
0142 542C 454E
0146 4420 4144
014A 4452 4553
014E 5320 544F
0152 2054 4553
0156 5420 3F
0159 00               BYTE  0
                *
015A 4E4F 4E2D  ERR1  TEXT  'NON-ZERO AT: '
015E 5A45 524F
0162 2041 543A
0166 20
0167 00               BYTE  0
```

```
OUTPUT READY?
00000MEMIDT
          A0000A0000B0300B0000B02E0B0000B04CCB020AB000AB0200C013E7F141F
B06A0C012AB0205B0004B06A0BFE18B0241BFFFEB0242BFFFEB0081B1BF2B04C67F117F
B04C9BC0C1BC043B04D1B05C1B1302B8081B12FBBC043B1F00B1303B1F00B16FE7F159F
B10E3B04C7BC211B8207B1302B06A0C00D0B0207B8000BC447BC211B8207B13027F1BDF
B06A0C00D6B0B17B17F8B0207B7FFFBC447BC211B8207B1302B06A0C00D6B0B177F14FF
B18F8B0707BC447BC211B8207B1302B06A0C00DCB0207BAAAABC447BC211B82077F154F
B1302B06A0C00E2B0B17B17F8B05C1B1302B8081B12CCB0586B0200C01A7B06A07F19BF
C012ABC3C6B06A0C0112B06A0BFDBABC3C9B13B9B0200C01B4B06A0C0132B06A07F14BF
C0112B06A0BFDBAB10B0B0200C015AB1008B0200C0168B1005B0200C0176B10027F1E4F
B0200C0184BC34BB0589B06A0C012ABC141B06A0BFDBAB0200C0192B06A0C01327F187F
BC148B06A0BFDBAB0200C0199B06A0C0132BC147B06A0BFDBAB045DB04C5B02047F135F
B0004B04CEB3F8AB0945B0ACFBA14FBC3CEB0604B16F8B045BB2CA0C01C6B2CA07F0E6F
C01C7BD410B1303B2C90B0580B10FBB045BB5354B4152B542CB454EB4420B41447F190F
B4452B4553B5320B544FB2054B4553B5420B3F00B4E4FB4E2DB5A45B524FB20417F179F
B543AB2000B524FB5441B5449B4E47B2041B543AB2000B414CB4C20B4F4EB45537F180F
B2041B543AB2000B414CB5420B4F4EB4553B2041B543AB2000B2C20B4953B3A207F1B6F
B002CB2053B484FB554CB4420B4245B3A20B0045B4E44B204FB4620B5041B53537F1AEF
B2000B2C20B544FB5441B4C20B4552B524FB5253B203DB2000B0D0A501C7ENDTST7F0EEF
50000MEMTST001C8        7FAEEF
:

EDIT/ASM/LOAD?
```

## LISTING 2

```
?G5000

START,END ADDRESS TO TEST ?5700,7FFE
END OF PASS 0001
END OF PASS 0002
END OF PASS 0003
END OF PASS 0004
ROTATING AT: 6BF2, IS: BFFF, SHOULD BE: DFFF
END OF PASS 0005, TOTAL ERRORS = 0001
ROTATING AT: 6B24, IS: 1000, SHOULD BE: 0800
ROTATING AT: 6E66, IS: DFFF, SHOULD BE: EFFF
ROTATING AT: 70A4, IS: 0040, SHOULD BE: 0020
ROTATING AT: 735E, IS: 1000, SHOULD BE: 0800
ROTATING AT: 763A, IS: 0400, SHOULD BE: 0200
ROTATING AT: 7FDA, IS: 0004, SHOULD BE: 0002
END OF PASS 0006, TOTAL ERRORS = 0007
END OF PASS 0007, TOTAL ERRORS = 0007
END OF PASS 0008, TOTAL ERRORS = 0007
END OF PASS 0009, TOTAL ERRORS = 0007
END OF PASS 0010, TOTAL ERRORS = 0007
START,END ADDRESS TO TEST ?4000,4FFE
END OF PASS 0001
END OF PASS 0002
END OF PASS 0003
END OF PASS 0004
END OF PASS 0005
END OF PASS 0006
END OF PASS 0007
END OF PASS 0008
END OF PASS 0612
END OF PASS 0613
END OF PASS 0614
START,END ADDRESS TO TEST ?
?
```