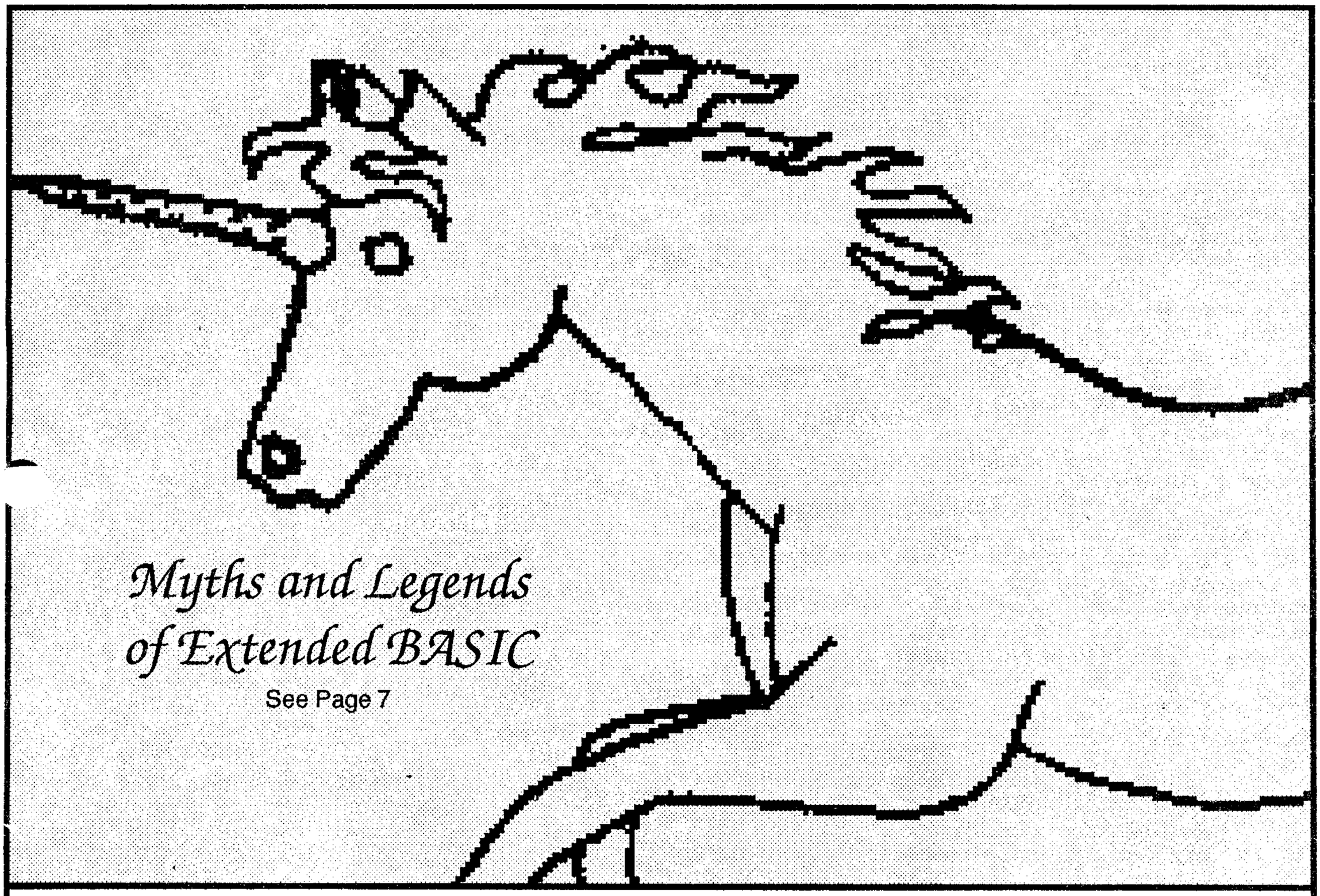
Covering the TI99/4A and the Myarc 9640

MICROpendium

Volume 11 Number 12 | January 1995 | $3.50



Myths and Legends of Extended BASIC

See Page 7

Extended BASIC

Jack Armstrong to the rescue

Hardware project

Power supply upgrade

Geneve

A program to print labels for 3.5-inch disks

Reviews

Reminders and Forget Me Knots, Windows for the TI, Attack of the Creepers, Picture Show

CONTENTS

MICROpendium

MICROpendium (ISSN 10432299) is published monthly for $35 per year by Burns-Koloen Communications Inc., 502 Windsor Rd., Round Rock, TX 78664-7639. Second-class postage paid at Round Rock, Texas. POSTMASTER: Send address changes to MICROpendium, P.O. Box 1343, Round Rock, TX 78680-1343.

No information published in the pages of MICROpendium may be used without permission of the publisher, Burns-Koloen Communications Inc. Only computer user groups that have exchange agreements with MICROpendium may excerpt articles appearing in MICROpendium without prior approval.

While all efforts are directed at providing factual and true information in published articles, the publisher cannot accept responsibility for errors that appear in advertising or text appearing in MICROpendium. The inclusion of brand names in text does not constitute an endorsement of any product by the publisher. Statements published by MICROpendium which reflect erroneously on individuals, products or companies will be corrected upon contacting the publisher.

Unless the author specifies, letters will be treated as unconditionally assigned for publication, copyright purposes and use in any other publication or brochure and are subject to MICROpendium's unrestricted right to edit and comment.

Display advertising deadlines and rates are available upon request.

All correspondence should be mailed to MICROpendium at P.O. Box 1343, Round Rock, TX 78680. We cannot take responsibility for unsolicited manuscripts but will give consideration to anything sent to the above address. Manuscripts will be returned only if a self-addressed stamped envelope is included.

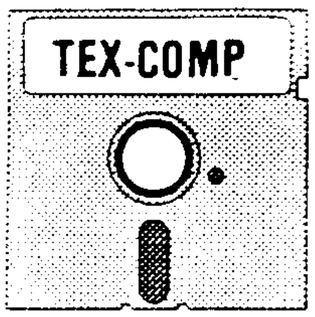
Foreign subscriptions are $40.25 (Mexico); $42.50 (Canada); $40.00, surface mail to other countries; $52 airmail to other countries.

All editions of MICROpendium are mailed from the Round Rock (Texas) Post Office.

Mailing address: P.O. Box 1343, Round Rock, TX 78680.
Telephone & FAX: (512) 255-1512
Delphi TI NET: MICROpendium
GEnie: J.Koloen
Internet E-mail: jkoloen@io.com
John Koloen.....Publisher
Laura Burns.....Editor

*READ THIS

Here are some tips to help you when entering programs from MICROpendium:
1. Most BASIC and Extended BASIC programs are run through Checksum, which places the numbers that follow exclamation points at the end of each program line. Do not enter these numbers or exclamation points. Checksum is available on disk from MICROpendium for $4.
2. Long Extended BASIC lines are entered by inputting until the screen stops accepting characters, pressing Enter, pressing FCTN REDO, cursoring to the end of the line and continuing input.



ANNOUNCING NEW LOW PRICES ON FREEWARE DISKS FROM $2.95 TEX COMP $2.95

TEX-COMP HAS SLASHED PRICES ON THE BIGGEST & BEST COLLECTION OF FREEWARE FOR THE 99/4A. NOW ONLY $2.95 PER DISK WITH A 5 DISK MINIMUM. CHOOSE FROM HUNDREDS OF GREAT PROGRAMS. NO EXTRA CHARGE ON PROGRAMS THAT REQUIRE MORE THAN ONE DISK SIDE. ALL PROGRAMS HAVE BEEN TESTED AND ALMOST ALL HAVE BEEN PROVIDED WITH EXBASIC AUTOLOAD. CHOOSE FROM THE BEST IN GAMES, UTILITIES, GRAPHICS, HOME & BUSINESS AND DISK BACKUPS OF DOZENS OF YOUR FAVORITE MODULES THAT ARE NOW OUT OF PRODUCTION.

GAMES • BUSINESS • GRAPHICS • WORD PROCESSING • UTILITIES • DATABASE • MUSIC • COMMUNICATIONS • HOME

GRAPHICS, MUSIC & ANIMATION
#1 THE SINGING TI VOL. 1 (S)(2)
#4 PRINTART (2)(P)
#5A MUSIC/GRAPHICS
#6 EXBASIC MUSIC (2)
#7 SPACE SHUTTLE MUSIC/GRAPHICS
#9 MONA LISA PRINTOUT (P)
#11 ANIMATED XMAS (WOODSTOCK)
#14 FIGURE STUDIES (P)(PG)
#32 EXBASIC XMAS MUSIC (2)
#41 VIDEO GRAPHS (M)
#52 ANIMATION 99' (2)
#69 PLAYER PIANO/KEYBOARD ANALYSIS
#93 KBGB GIRLIE CALENDAR (4)(P)
#103 SORGAN THE TI ORGAN
#107 STARTREK MUSIC ALBUM
#111 POP MUSIC & GRAPHICS
#114 PANORAMA
#115 GRAPHICS DESIGN SYSTEM
#120 BITMAC (2)(P)
#230 THE SINGING TI VOL. 2 (S)(2)
#231 THE SINGING TI VOL. 3 (S)(2)
#246 THE SINGING TI VOL. 4 (S)(2)
#313 3-D WORLD (CAD for the TI)

COMPUTER UTILITIES, PRINTER UTILITIES & PROGRAMMING LANGUAGES
#3 DUMPIT (E/A)
#15 STAR/EPSON PRINTER DEMO (P)(2)
#16 SIDEWAYS PRINTOUT (P)(2)
#18 TI DIAGNOSTIC (MM)(2)
#20 LOADERS & CATALOGERS
#30 HOUSEHOLD BUDGET PRINTOUT (P)
#35 PROGRAMMING AIDS & UTILITIES I
#42 FUNNELWEB FARM (SHELL UTILITY)
#53 HACKER CRACKER
#55 SCREEN DUMP (P)
#62 DISK MANAGER II (M)
#75 DISK CATALOGER
#76 PROGRAMMING AIDS & UTILITIES II
#78 ARTCON GRAPHIC CONVERSION
#79 DISK MANAGER 1000
#80 BIRDWELL DISK UTILITY (2)
#85 AUTOBOOT UTILITY
#86 COLUMN TEXT III (P)
#87 ARCHIVER III
#89 PROCALC DECIMAL/HEX CONVERTER
#96 STATISC & SORT ROUTINES
#97 MEMORY MANIPULATOR
#101 ENHANCED DISPLAY PACKAGE
#108 FUNLPLUS (PLUS!)
#110 DISK+ AID
#117 UNIVERSAL DISASSEMBLER
#119 RAG LINKER CONVERSION
#127 PIX GRAPHICS UTILITY
#243 OS/99 (GD)
#251 PC TRANSFER TI/IBM (DD)
#253 THE EXPLORER/DM1000
#254 NIBBLER/TURBO
#260 TI FORTH (DISK ONLY- add $8 for manual)
#17 TI FORTH DEMO
#116 TI FORTH TUTORIAL
#5079 FORTH SOURCE CODE
#104 C99 COMPLIER & LIBRARY
#5007 TEACH YOURSELF TI BASIC
#5019 TEACH YOURSELF EX-BASIC
#5067 BEGINNING BASIC TUTOR
#307 GRAPHICS CODE GENERATOR
#3912 SUPER BUGGER
#3913 BETTER BANNERS
#3914 CERTIFICATE 99
#3915 HOROSCOPE MAKER
#3916 GRAPHX+ PRINT SHOPPE
#3917-3720 GRAPHX COMPANIONS 1-4
#3721 MAC FLICK (G)
#3722 PRINTING TO GO (G)
#3723 GRAPHX DINOSAURS (G)

BUSINESS, ACCOUNTING, WORD PROCESSING, DATA BASE HOME OFFICE & HOME
#10 GOTHIC PRINTOUT (P)
#19 TI WRITER/MULTIPLAN UPGRADE
#20 ACCOUNTS RECEIVABLE (2)
#21 DATA BASE DEMO
#23 WILL WRITER
#29 LABEL MAKER I (P)
#36 STRICTLY BUSINESS (2)
#56 SPREAD SHEET
#58 PR BASE (database)
#59 GRAPH MAKER
#74 LABEL MAKER II (P)
#77 MICRodex 99 (database)
#81 HOME ACCOUNTING SYSTEM
#83 HOME APPLICATION PROGRAMS (2)
#90 JET CHECKBOOK MANAGER
#92 HOUSEHOLD INVENTORY
#109 TI WRITER MINI MANUAL
#112 INVOICE PACK
#113 LABEL MAKER III
#129 CASH DRAWER (point of sale)
#130 THE ORGANIZER
#147 CALENDAR-NOTEPAD
#177 HOUSEHOLD BUDGET MANAGEMENT (M)
#30 HBM DATA PRINTOUT (P)
#221 PERSONAL REAL ESTATE (M)
#249 MAPMAKER
#252 99 WRITER II (TI WRITER)(P)
#253 AMA MAILING LIST
#255 TRAK-A-CHECK
#257 DAILY DIARY
#257A EXER-LOG
#23 WILL WRITER
#310 SELF HELP TAX CUT

TELECOMMUNICATIONS (MODEM)
#57 TELCO
#57D TELCO (FOR SYSTEMS WITH DS DRIVES)
#118 FAST TERM

EDUCATION & PERSONAL DEVELOPMENT
#22 ASTROLOGY
#24 ENGINEERING CALCULATIONS (2)
#25 MEDICAL ALERT
#27 KIDS LEARNING I (2)
#31 MORSE CODE TRAINER
#37 LAPD COOKBOOK (2)
#40 ARTIFICIAL INTELLIGENCE (ELIZA)
#54 ASTRONOMY
#66 HEBREW TYPEWRITER
#67 GENEALOGY (2)
#71 KIDS LEARNING II (2)
#95 WEATHER FORECASTER +
#138 FIREHOUSE COOKBOOK
#142 TOUCH TYPING TUTOR (M)
#184 FACE MAKER
#194 ST. NICK/GHOSTLY SPELLING (M)
#195 TINY LOGO
#199 MILLIKEN ADDITION (M)
#200 MILLIKEN DECIMALS (M)
#203 MILLIKEN FRACTIONS (M)
#204 MILLIKEN INTEGERS (M)
#205 MILLIKEN LAWS OF MATH (M)
#211 MIND CHALLENGERS (M)
#212 MINUS MISSION (M)
#213 MILLIKEN PERCENTS (M)
#214 STORY MACHINE (M)
#215 BEGINNING GRAMMAR (M)
#216 METEOR MULTIPLICATION (M)
#217 HANGMAN (M)
#222 MUSICMAKER (M)
#223 PHYSICAL FITNESS (M)
#225 ALIEN ADDITION (M)
#226 ALLIGATOR MIX (M)
#227 DEMOLITION DIVISION (M)
#228 DRAGON MIX (M)
#270 TRIGONOMETRY
#271 CALCULATORS & CONVERSIONS (2)
#272 HIGHER MATH (2)
#273 ASTRONOMY 2
#306 SPEAK & SPELL II (S)(EX)

GAMES
#2 WH. OF FORTUNE, BLKJACK, POKER
#3 TI TREK
#8 LOTTO PICKER
#13 STRIP POKER (PG)
#26 R-RATED NOVELTY GAME
#33 CHECKERS & BACKGAMMON
#34 SOLITAIRE & SCRABBLE
#38 GREAT TI GAMES VOL 1
#39 GREAT TI GAMES VOL 2
#43 BEST OF BRITAIN GAMES VOL 1
#45 BEST OF BRITIAN GAMES VOL 2 (LEGEND OF CARFAX ABBY GRAPHIC-INTERACTIVE ADVENTURE)
#46 SUPER TRIVIA 99
#47 INFOCOM RAPID LOADER
#48 GHOSTMAN (from U.K.)
#49 DEMON DESTROYER (from FRANCE)
#50 OH MUMMY! (HIT from GERMANY)
#51 BERLIN WALL (from CANADA)
#60 FREDDY (HIT from GERMANY)
#61 THE MINE (from GERMANY)
#63 ASTROBLITZ & MAZOG
#64 MAJOR TOM & SPACE STATION PINETA
#65 PERFECT PUSH (HIT)
#69 CHESS (SARGON)
#70 TI RUNNER II (HIT)
#72 CEBERUS (HIT SPACE GAME)
#73 CRYTO (GRAM)
#82 CROSSWORD (PUZZLES)
#84 GALACTIC BATTLE & SPY ADVEN.
#88 AUSSIE GAME COLLECTION VOL 1
#91 THE MAZE OF GROG (WOODSTOCK II)
#94 GREAT TI GAMES VOL 3
#98 DAYS/DOORS OF EDEN (BIBLE ADV)(AM)
#99 GREAT TI GAMES VOL 4
#100 ASSAULT THE CITY (TD)
#102 COLOSSAL CAVES (ADVENTURE)
#105 KINGS CASTLE (M)
#106 QUEST (D&D)
#121 SUPER YAHTZEE & WHEEL II
#122 ADULT ADVENT & GAMES (PG)
#123 GREAT TI GAMES VOL 5 (2)
#124 GREAT TI GAMES VOL 6 (2)
#125 BLACKJACK & POKER (M)
#126 VIDEO CHESS (M)
#128 TETRIS (HIT from RUSSIA)
#131 COMPUTER CRAPS
#132 AMBULANCE (M)
#133 DRIVING DEMON (M)
#134 ROTO-RAIDER (M)
#135 ARTURUS (HIT-ZARGON)
#136 ANT-EATER (M)
#137 CROSS-FIRE (M)
#139 MOONMINE (M)
#140 MASH (M)
#141 MOONSWEEPER (M)
#143 CONGO BONGO (M)
#144 STAR TREK (M)
#145 BUCK ROGERS (M)
#148 KENO & SLOTS
#149 GREAT TI GAMES VOL 7 (with HIT BLOCKBUSTER)
#150 ULTIMATE TRIVIA
#151 JUNGLE HUNT (M)
#152 POLE POSITION (M)
#153 DONKEY KONG (M)
#154 PROTECTOR II (M)
#155 PAC MAN (M)
#156 CENTIPEDE (M)
#157 DEFENDER (M)
#158 SHAMUS (M)
#159 MS. PAC MAN (M)
#160 DIG-DUG (M)
#161 PICNIC PARANOIA (M)
#162 MOON PATROL (M)
#3711 ARCADE SPECIAL (4 GAMES)
#3712 THREE GREAT GAMES
#3713 PRO TENNIS+

INFOCOM ADVENTURE BACKUPS
#163 ZORK I
#164 ZORK II
#165 ZORK III
#166 HITCHIKER'S GUIDE
#167 WITNESS
#168 ENCHANTER
#169 INFIDEL
#170 PLANETFALL
#171 SORCERER
#172 DEADLINE
#173 CUTTHROATS
#174 SUSPENDED
#175 STARCROSS
****
#176 AMAZING (M)
#178 DEMON DESTROYER (M)
#179 POPEYE (M)
#180 QUEBERT (M)
#181 METEOR BELT (M)
#182 BLASTO (M)
#183 CAR WARS (M)
#184 FACE MAKER (M)
#185 SUPER FLY (M)
#186 SPACE BANDITS (M)
#188 KILLER CATERPILLER (M)
#190 BLACK HOLE & SPACE AGRESS. (M)
#191 GREAT TI GAMES VOL 8
#192 GREAT TI GAMES VOL 9
#193 SPY'S DEMISE (HIT) (M)
#194 ST. NICK+ (M)
#196 JOTTO
#197 PRO TENNIS+ (HIT)(M)
#198 TI INVADERS/TOMBSTONE CITY (M)
#202 CONNECT FOUR (M)
#205 HOPPER (M)
#206 TREASURE ISLAND (M)
#206B SLYMOIDS (M)
#207 OTHELLO (M)
#208 FARSEC (M)
#209 SOCCER (M)
#210 SEWERMANIA (M)
#218 HUSTLE/FOOTBALL (M)
#219 CHISOLM TRAIL (M)
#220 ZERO ZAP (M)
#224 ATTACK (M)
#229 4A FLYER (FLIGHT SIM.)(M)
#232 TUNNELS OF DOOM (MOD BACKUP PLUS 2 NEW ADVENTURES)(M)
#233 MS ADVENTURES (3 ADV-EXBASIC)
#248 STRIKE THREE BASEBALL (M)
#248 GREAT TI GAMES VOL 10
#250 BARRAGE/SPOTSHOT
#318 THREE GREAT GAMES
#319 ARCADE SPECIAL
#317 BEANSTALK ADVENTURE
****
SCOTT ADAMS ADVENTURES (USE WITH ADVENTURE MODULE)
#349 TI ADVENTURES 1-13+
#350 TI ADVENTURES 14-16+

P = Printer required
G = Graphx required
S = Speech required
M = Module Backup
MM = Mini Memory req.
E/A = Editor Assem. Exbasic and 32k mem req. for most programs.

Authorized Dealer

TEX COMP

America's Number One TI computer retailer

P.O. Box 33084, Granada Hills, CA 91344

ORDER BY PHONE

(818) 366-6631 24 HOURS A DAY

TERMS: MINIMUM ORDER FOR $2.95 PRICE IS 5 DISKS (REG. 4.95) ADD $4.00 PER ORDER FOR SHIPPING (U.S.)
ALL PRICES ARE FOR CASH/CHECK, ADD 3% FOR CREDIT CARD ORDERS. INCLUDE STREET ADDRESS FOR U.P.S.

COMMENTS

TI users deserve better from OPA

Oasis Pensive Abucators, aka OPA, is an enigma disguised as a computer peripheral company. In the 11 years that I've been involved with the TI I can't think of any company that has toyed with its customers as much as OPA. Toyed is a polite description. I know of a number of OPA customers whose descriptions of the company wouldn't be printed in this magazine.

The enigma that is OPA is that it seemingly has developed reasonably useful products for the TI market. I've seen some of them demonstrated at computer fairs. They seemed to work. Just as reasonably, many TI users paid money to purchase OPA products, most of them through mail order. What went wrong was that OPA didn't always send the products. Worse, the company didn't both getting in touch with its customers to inform them of a delay in delivery. Of course, if they did, they'd have been making the call for several years since many users have been waiting that long for OPA products.

I've heard many explanations of the company's behavior, which is headed by Gary Bowser. Bowser and OPA are seemingly inseparable in the TI marketplace. Most of the explanations resolve themselves around dishonesty. I have no insight into Bowser's thought processes. But I do know that a person who says one thing and does another can't be trusted. I think Bowser probably believes he's doing the right thing, that somehow he's justified in ignoring his customers and keeping their money. That intentions are more important than results. However, I doubt very much whether the scores of TI users who have lost thousands of dollars on OPA products that have never been delivered believe that good intentions are enough. If OPA was an honest company, it would refund all the money that TI users are owed. Just don't hold your breath.

KRYCH ASSEMBLY SERIES NO MORE

Jim Krych reports that his proposed series on assembly language won't be completed. "I am sorry that this will disapoint some people. Time and other constraints have put a damper on this project." Krych was assisted on the project by Bruce Harrison, Brad Snyder, Art Green, Joe Delekto, Chris Bobbitt and Charlie Good.

STILL WAITING FOR SCSI

I'm still waiting for my SCSI card from Bud Mills. As you recall, Bud said at the Chicago TI fair that the Geneve version of the card was ready for shipment. I'd like to know if anyone else received one recently.

—JK

MICROpendium disks, etc.

- ❑ **Series 1994-1995 mailed monthly** (April 1994-March 1995)..... $40.00
- ❑ **Series 1993-1994 mailed monthly** (April 1993-March 1994)..... $25.00
- ❑ **Series 1992-1993** (Apr 1992-Mar 1993, 6 disks).. $25.00
- ❑ **Series 1991-1992** (Apr 1991-Mar 1992, 6 disks).. $25.00
- ❑ **Series 1990-1991** (Apr 1990-Mar 1991, 6 disks) ..$25.00
- ❑ **Series 1989-1990** (Apr 1989-Mar 1991, 6 disks) ..$25.00
- ❑ **Series 1988-1989** (Apr 1988-Mar 1989, 6 disks)...$25.00
- ❑ **110 Subprograms** (Jerry Stern's collection of 110 XB subprograms, 1 disk) $6.00
- ❑ **TI-Forth** (2 disks, req. 32K, E/A, no docs).....$6.00
- ❑ **TI-Forth Docs** (2 disks, D/V80 files) $6.00
- ❑ **1988 updates of TI-Writer, Multiplan & SBUG** (2 disks) $6.00
- ❑ **Disk of programs** from any one issue of MICROpendium between April 1988 and present $4.00
- ❑ **CHECKSUM and CHECK** programs from October 1987 issue (includes docs as D/V 80 file) $4.00

Name _____

Address _____

City _____

State_____ ZIP_____

Texas residents add 7.75% sales tax.. Credit card orders add 5%.
Check box for each item ordered and enter total amount here:

Check/MO Visa M/C
(Circle method of payment)

Credit Card # _____

Exp. Date _____

Signature _____

"FABULOUS"

FEST WEST 95

SAN DIEGO

SATURDAY, FEBRUARY 18, 1995

ADMISSION: $5.00

Children under 15 are FREE when accompanied by an adult.

Raffle tickets are ONLY $1 each. Buy ten (10) raffle tickets & Receive one (1) FREE ADMISSION.

Come buy & visit such All-STAR vendors as:

ComProDiNe, Harrison Software, Inky Dew, Bud Mills Services, Notung Software, Tex-Comp, WHT, & more!

HOURLY DOOR PRIZES!

FEST WEST '95 is being held at the FABULOUS INN.

Room rates are:

One bed	Two beds
$44 per night	$49 per night

DeLuxe room upgrade -- Add $5 per night.

For reservations and related information, please write or call:

FABULOUS INN
2485 Hotel Circle Place
San Diego, CA 92108

In CA, TOLL FREE 1 (800) 647-1903
OR
In U.S., TOLL FREE 1 (800) 824-0950

For more information about TI FEST WEST '95,

Please write or call:

Southern California Computer Group
P.O. BOX 152535
San Diego, CA 92195

S.C.C.G. Data Line at:
(619) 263-9135
300-2400 BPS 8N1
User No.: 25
Password: FEST

FEEDBACK

Brashear looking for a new windmill

One of the TI99 community's oldest friends, loyalists and resident fanatics, Harry Brashear, has apparently moved on to the PC world. This is the Western New York 99er who gave years of service, leadership and editorial talent to his user group, who gave the TI community programs and applications ranging from disk cataloging utilities to home publishing tips, along with countless articles and MICRO-Reviews.

In case you haven't read it in MICROpendium or your favorite user group newsletter, Harry wrote a nice "Open Letter" to the TI community in November '94 that was pretty much an explanation of why he was making the move to the PC world, followed by an invitation to migrate to the PC world with him if the time was right for you, as he explained the time was right for him. His idea seemed to be to flip off Bill Gates (co-owner and co-founder of Microsoft with Paul Allen), and Microsoft Windows with a "Digital Attitude," and adopt Brian Dougherty's Geoworks along the way in protest of the Microsoft "establishment." I'm not sure why he wants to protest Microsoft or Windows, but whatever?

At first Harry's letter seemed to be a bit incongruous, since it was Mr. Brashear himself who scorned Craig Miller and had a running feud with Joe Nuvolini of the Front Range 99ers in Colorado Springs, Colorado, over their move toward the PC world when the Triton Turbo XT vaulted into the limelight in 1987. But after reading the letter again and thinking about it, I have formed the opinion that Harry Brashear needs windmills to fight. In his own way he is a 1980s and 1990s version of Cervantes' Don Quixote, defender of causes and the little man, a man for all seasons, a champion of the antiestablishment, and he's looking for a new windmill to do battle with. So be it! Luck to him. He has added much to the flavor of life among our tiny but awesome group of enthusiasts over the years and I will miss his presence.

In his letter Harry talks about Geos and Geoworks as alternatives to the Microsoft Windows shell over DOS. I happen to be one of those who bought a Triton Turbo XT from Terry Miller back in 1987 and as a result ended up going with PC/GEOS when it appeared in 1990. The Geos software went through versions 1.0, 1.1, 1.2 and Finally Geoworks Pro before it all turned into Geoworks Ensemble, which is the integrated word processor, database, spreadsheet and communications program that Harry Brashear refers to in his "Open Letter." Geoworks Ensemble is even more than what Harry described, though. The program also offers a calendar, schedule manager, an America Online linkup, a notepad, a scrapbook (what Windows calls the Clipboard) and a bunch of other little goodies such as long document names, a Geos version of Solitaire and Tetris and a slick banner maker.

My point in providing PC world information to 99ers is this: if you are thinking of migrating to the PC arena, and have an 80286 or older chip in your machine, or if you want to make the move with the 1960s protest spirit in your heart that Harry Brashear refers to, the Geoworks option is open to you. On the other hand, if you don't want to bother with the "protest" idea and have no "statement" to make to anyone, go the Microsoft Windows route, it's the real world.

In case you *really* want to obtain a copy of Geoworks, you can do so in a couple of different ways. One way is to contact Geoworks, which is the same company that used to be known as Berkeley Softworks when it introduced Geos for the Commodore 64C back in 1986. Geoworks is located at 960 Atlantic Ave., Alameda, CA 94501. Alternately, you can call them direct at (510) 814-1660. Both the address and the phone number are new. Another way that you can obtain the Geoworks software is to contact one of the shareware software distribution houses such as The Software Labs in Seattle and pay a few dollars for the new Geo Publisher shareware program. When you register it for $79.95 you get the whole Geoworks Ensemble package. It is absolutely worth $79.95. You can contact The Software Labs at (206) 869-6802. Ask for item No. 15790 to get Geo Publisher.

On another issue, in case you are wondering what Chris Bobbitt is doing t days, since he no longer has Asgard Soft ware, he is a full time director of manage ment information systems for the Nationa League of Cities, which is a special inter est group of sorts that represents membe towns and cities in Washington, DC. Bob bitt just recently (September 1994) fin ished writing an excellent article on the In ternet for the NLC weekly newspaper tha I had the pleasure of reading. The article was well written and quite informative, which is pretty standard fare for a Chris Bobbitt-authored work.

One final item of interest to owners o First Draft/Final Copy. I called Art Gibson recently and it appears that he has given up on creating any fixes or providing any updates for his First Draft/Final Copy word processor. He never did get it to work right, apparently. I loaned him my extra HFDC so he could fix the hard disk routines in V.2 of the program, but the fixes have not been forthcoming.

Bill Gaskill
Grand Junction, Colora

USVBA explanation

First of all, I'd like to thank Charles Good for his review of my USVBA Volleyball game in the November issue of MICROpendium, and then a few words of explanation. In regards to his comment of the game being too tough, there have been a few joystick-happy youngsters at the Faire in previous years that found it too easy to beat it, so this year the program was entirely rewritten, making it tougher at all levels and adding a fifth level of difficulty. I myself can still beat it once in a while at the lowest levels and have even beaten it once on level 5, but, yes, it definitely is much tougher now.

In previous years the game could not be played on the Geneve, but now, thanks to Tim Tesch's debugging of MDOS, I have been able to include a Geneve version on the disk.

Although the game is on a DSSD disk, it is also available on an SSSD disk. Th complete programs in image format wit the Extended BASIC loader and the documentation are all included. The only

(See Page 7)

The myths and legends of Extended BASIC

By BRUCE HARRISON

Yes, that's your assembly author nosing around in Extended BASIC, as if he hasn't enough to do! Curiosity, however, has caused us to do many foolish things, and this is another case in point. What made us curious was whether things "everybody knows" about Extended BASIC are really true, or whether they fall into some kind of mythology, along with unicorns and such. Our main objective involved the speed of operations in Extended BASIC, and whether certain well-known methods of improving the speed of XB programs were really worthwhile.

FIND THE TOOLS...

A friend of ours used to describe how he went about fixing his aging Land Rover. "First I find the tools, then I fix the tools, then I start the job." That seems to work around our house, too, except when it comes to the TI. Many times on the TI, we have to "First *make* the tools...." (That is, write a new program or routine.)

That was indeed the case here. What we wanted was a method to accurately time operations while they happen, without having to resort to stopwatch methods, which are inadequate in most cases. We knew from some recent assembly projects that the one-sixtieth second interrupt cycle can provide a reliable and fairly accurate time base. The question was, how do we use that while Extended BASIC is going about its business?

Another thing "we all know" is that multiple statement lines execute faster than the same function performed as single statement lines. That turns out to be very nearly a myth.

The answer was use of the "user interrupt" with a small assembly routine or two or three. The idea was to clear a word in memory, then make that word be incremented on each one-sixtieth second interrupt until we told it to stop. We wound up making two such tools, one for measuring time intervals less than 18 minutes, 12 seconds, and another that can go beyond 18 minutes, 12 seconds. This magic number of 18 minutes, 12 seconds is approximately how long it takes for a counter running at 60 increments per second to go from 0 to 65,535, which is the limit for a single word of memory. (To be exact, 65,535 60ths of a second is 18 minutes, 12 and one-quarter seconds.) In any case, two tools were invented before any of the myths could be investigated properly.

MYTH NUMBER ONE

From our first lessons in Extended BASIC, we were taught that it's good practice to keep the names of variables as short as possible. This not only saves memory, but also improves the speed of operation, or at least that's what we always believed. Now, given the "tool," we could test that theory and either prove or debunk the myth. Accordingly, we made up a test program that would simply do FOR-NEXT loops displaying the loop variable, but would use our ACCTIME/O assembly routine to measure execution time.

The program, which you can see in the listings, makes three runs of a FOR-NEXT loop, in which it repeats a DISPLAY AT of the loop variable from 1 to 2000. In the first loop, the variable is named ANY-

(See Page 8)

FEEDBACK

(Continued from Page 6)

things left off the SSSD disk are the DF80 Option 3 files that are unnecessary anyway.

Gene Hitz
Program Innovators
Wauwatosa, Wisconsin

Users groups, unite!

With the TI community getting smaller, it is imperative that those of us who are still supporting the TI99/4A continue to stay in touch with each other. Thus, the reason for this letter and my suggestion. I suggest that MICROpendium list those users groups that are still supporting the TI99/4A (in every issue, along with the users group's address). While this idea may seem like a waste of space, I assure you it isn't. My own users group, the West Penn 99ers, has been in existence for nearly 10 years, and although we exchange newsletters with 40 TI clubs, I know that there are other clubs who haven't even heard of us (or us of them).

Now, more than ever, the lines of communication must be maintained. To get the ball rolling on this idea, here is the correct address for the West Penn 99ers (of which I am president).

West Penn 99ers
c/o Mickey Cendrowski
100 Pine St.
Russellton, PA 15076

Not sure about a listing to be continued every issue — though we could change our minds if the demand is there. In any case, users group representatives are welcome to write letters such as the above with information about their groups. — Ed.

Send your letters and comments to *MICROpendium Feedback*, P.O. Box 1343, Round Rock, TX 78680.

EXTENDED BASIC MYTHS—

(Continued from Page 7)

OLDVARIABLE, which is just one letter short of the maximum length for a variable name. In the second loop, the variable name is just A, the shortest possible length for a variable name. In the third loop, the variable name is again A, but the display is of STR$(A), instead of the numeric itself.

The results of this were interesting. With the long variable name, the 2,000 repeats took 2 minutes and 5 seconds. With the short variable name, the 2,000 repeats took 1 minute and 59 seconds, and the displaying of STR$(A) 2,000 times took 2 minutes and 7 seconds. That proved that our teacher was correct — the shorter variable name did make things go faster, but not by much. Six seconds out of two minutes is only a five percent change, and might not be worth the trouble of reworking a program. The result with STR$ was also surprising, in that it didn't make as dramatic a change as expected. Only eight seconds difference over two minutes, or 6.67 percent. We expected a larger difference, because the STR$ function uses string space in VDP RAM, and that should slow things down.

The length of variable names will, of course, affect the memory use in a program by exactly the number of characters shaved off the name times the number of times it appears. VDP memory (or STACK) will also be saved when the program runs, and this may prove useful.

MYTH NUMBER TWO

Another thing "we all know" is that multiple statement lines execute faster than the same function performed as single statement lines. That turns out to be nearly a myth. It's true, but just barely, and we had to extend our loops to 5,000 repeats to find a difference. Running a loop of 5,000 DISPLAY AT operations takes about five minutes on our TI. In the program TEST2, we ran this twice, with the first being in single statement lines as follows:

```
40 FOR I=1 TO 5000
50 DISPLAY AT(12,12):I
60 NEXT I
```

The second case was done with all of the loop in just one line, like this:

```
110 FOR I=1 TO 5000 :: DISPL
AY AT(12,12):I :: NEXT I
```

Again, results were surprising. The set of three single statement lines executed its 5,000 repeats in 5 minutes and 2 seconds, while the "faster" multiple statement version took an incredible 5 minutes and 1 second. Yes, that's only one second out of five minutes! That's only one-third of one percent! Thus, it's true that multiple statement lines execute faster, but not by much! Certainly not by enough to make it worthwhile to rewrite an old program in hopes of speeding it up by combining lines. There is one saving that may be significant, and that's in memory use. Each separate line that can be combined into another line saves exactly four bytes of memory. (That's because each entry in the Line number table takes four bytes.)

Design of the program you're using can make a big speed difference. This turns out to be very true.

HOW ACCURATE IS THIS?

That, as the saying goes, depends. The accuracy our timer gives depends on the kind of operations XB is performing. For example, if a disk drive is accessed, our "clock" is effectively stopped while the disk access is happening; thus, our clock simply ignores that time. If PRINT is used instead of DISPLAY AT in our example programs, the time measurement is quite inaccurate in absolute terms, because the PRINT function effectively "stops the clock" during screen scrolling, causing our measurement technique to "miss" its counts. Except for PRINT and file access operations, though, the timing is pretty accurate. It would work nicely to time different methods of sorting, for example, provided the sort didn't require PRINT or disk access while sorting.

To get a better handle on our accuracy, we designed the other tool, LONGTIME/O, for measurement of times longer than the magic 18-minute, 12-second limit. Then we ran it with a DISPLAY AT operation for 25,000 repeats. As expec this took slightly over 25 minutes to run. By careful use of a stopwatch, we could measure the error between the digital watch and our internal routine. The difference turned out to be about 6 seconds out of 25 minutes, or about four-tenths of one percent. That is, our routine showed 0.4 percent less time having passed than the digital watch showed. Not bad! In both routines, rounding to the nearest second is performed using the remainder of sixtieths. If the remainder is less than thirty-sixtieths, the seconds are left alone. If it's thirty-sixtieths or greater, the seconds count is incremented.

MYTH NUMBER THREE

Design of the program you're using can make a big speed difference. This turns out to be true. To illustrate that this myth is fact, we took an example program sent us by our dear friend Stephen Shaw back when we were working on our compiler. We found that a redesign of the progra could perform exactly the same job in much less time. The original program, "SPIRAL" was written by Jon Keller of the Bluegrass 99 group, back in April of 1986. In its original form, this filled the screen in spiral fashion, starting at the center and continuing outward until 23 rows by 23 columns were filled with one character. Originally, this ran from character 33 (!) through character 126 (~). That takes a long time indeed to complete, so our test version runs only from character 48 (0) through 57 (9).

We modified the program primarily to minimize the number of math operations required within each FOR-NEXT loop. By making ROW and COL the loop control variables, we were able to avoid having to do math such as ROW=ROW+RD within the loop itself. In general, we "cleaned up" the original to make it somewhat leaner and more efficient. For this article, we added some lines to each version to load and activate our timing routine.

The results are impressive. The origina version, running as modified from (0) through (9), takes 3 minutes and 15 sec-

(See Page 9)

EXTENDED BASIC MYTHS—

(Continued from Page 8)

onds. Our modified version takes 2 minutes and 21 seconds to do the same job. That's about a 38 percent improvement in speed, without using any tricks other than just making Extended BASIC perform more efficiently. Of course you might think that setting up a FOR-NEXT with FOR ROW=ROW TO ROW+N is a trick, but it's perfectly okay, and works as intended. Keep in mind, though, that many hours of programming effort were expended to make this improvement. The revised version even takes less memory to run, provided we delete the REM line that identifies our having modified the program.

Our purpose when we started working on this particular program was to make it perform better after compiling, and there was an enormous difference in running time when compiled, as well.

That's three myths we've clarified, but no legends. (We lied about the legends.) Reminds me of an old joke, whose punch line goes: "An elephant. I lied about the fathers." This article is number one of a series of one. Unless there's untold clamoring for more, this will be just a single standalone Extended BASIC lesson from your assembly person. We hope you can all learn something from it. For the benefit of those who get MICROpendium on disk, we've included the object files and all our test programs in our submission, and asked John and Laura to include those on the distribution disks.

The assembly routines ACCTIME/O and LONGTIME/O are available on a public domain disk from our good friends at the Lima Users Group, and may also be available from your local users group. or a BBS service. By the way, the LONGTIME/O routine does take into account that extra one quarter of a second at the end of each 18 minutes and 12 seconds. It does that by simply advancing the count by fifteen-sixtieths before starting the next cycle of counting. The disk includes instructions, source and object files, all our test programs, and so on. Enjoy!

TEST1

```
1 ! TEST1 - LONG AND SHORT
VARIABLE NAMES
2 ! BY Bruce Harrison
3 ! 01 DECEMBER 1994
4 ! PUBLIC DOMAIN
10 CALL INIT :: CALL LOAD("D
SK1.ACCTIME/O")
20 CALL CLEAR :: ON BREAK NE
XT :: CALL LINK("STRTIM")
30 FOR ANYOLDVARIABLE=1 TO 2
000 :: DISPLAY AT(12,12):ANY
OLDVARIABLE :: NEXT ANYOLDVA
RIABLE
40 GOSUB 150
50 PRINT "PRESS A KEY TO CON
TINUE";:: ON BREAK STOP
60 CALL KEY(0,K,S):: IF S<1
THEN 60 ELSE DISPLAY AT(24,1
):""
70 ON BREAK NEXT :: CALL LIN
K("STRTIM")
80 FOR A=1 TO 2000 :: DISPLA
Y AT(12,12):A :: NEXT A
90 GOSUB 150
100 PRINT "PRESS A KEY TO CO
NTINUE";:: ON BREAK STOP
110 CALL KEY(0,K,S):: IF S<1
 THEN 110 ELSE DISPLAY AT(24
,1):""
120 ON BREAK NEXT :: CALL LI
NK("STRTIM")
130 FOR A=1 TO 2000 :: DISPL
AY AT(12,13):STR$(A):: NEXT
A
140 GOSUB 150 :: ON BREAK ST
OP :: END
150 CALL LINK("STPTIM",M,S)
160 IF S<10 THEN S$="0"&STR$
(S)ELSE S$=STR$(S)
170 PRINT "MINUTES:SECONDS":
STR$(M);":";S$ :: RETURN
```

TEST2

```
1 ! TEST2 - SINGLE AND MULTI
 STATEMENT LINES
2 ! by Bruce Harrison
3 ! 01 DECEMBER 1994
4 ! PUBLIC DOMAIN
10 CALL INIT :: CALL LOAD("D
SK1.ACCTIME/O")
20 CALL CLEAR
30 ON BREAK NEXT :: CALL LIN
K("STRTIM")
40 FOR I=1 TO 5000
50 DISPLAY AT(12,12):I
60 NEXT I
70 GOSUB 120
80 PRINT "PRESS A KEY TO CON
TINUE";:: ON BREAK STOP
90 CALL KEY(0,K,S):: IF S<1
THEN 90 ELSE DISPLAY AT(24,1
):""
100 ON BREAK NEXT :: CALL LI
NK("STRTIM")
110 FOR I=1 TO 5000 :: DISPL
AY AT(12,12):I :: NEXT I ::
GOSUB 120 :: ON BREAK STOP :
: END
120 CALL LINK("STPTIM",M,S)
130 IF S<10 THEN S$="0"&STR$
(S)ELSE S$=STR$(S)
140 PRINT "MINUTES:SECONDS"
150 PRINT STR$(M);":";S$ ::
RETURN
```

TEST3

```
1 ! TEST3 - LONG TIME
2 ! BY Bruce Harrison
3 ! 01 DEC 1994
4 ! PUBLIC DOMAIN
10 CALL INIT :: CALL LOAD("D
SK1.LONGTIME/O")
20 CALL CLEAR :: ON BREAK NE
XT :: CALL LINK("STRTIM")
30 FOR I=1 TO 25000
40 DISPLAY AT(12,12):I
50 NEXT I
60 CALL LINK("STPTIM",M,S)::
 ON BREAK STOP
70 IF S<10 THEN S$="0"&STR$(
S)ELSE S$=STR$(S)
80 PRINT "MINUTES:SECONDS"
90 PRINT STR$(M);":";S$
```

SPIRAL1 ORIGINAL VERSION

```
10 CALL INIT :: CALL LOAD("D
SK1.ACCTIME/O")
20 ON BREAK NEXT :: CALL LIN
K("STRTIM")
30 REM SPIRAL1
40 REM JON KELLER
50 REM BLUEGRASS 99 4/86
60 CALL CLEAR :: CN=48
70 ROW=13 :: COL=17
80 CALL HCHAR(12,16,CN)
90 N=2
100 FOR Z=1 TO 11
```

(See Page 10)

EXTENDED BASIC MYTHS—

```
0001  * PART TWO - SOURCE CODE FILES
0002  * FIRST, THE TIMER FOR SHORTER
0003  * DURATIONS (LESS THAN 18 MINUTES)
0004  * ASSEMBLED, THIS TAKES 150 BYTES IN MEMORY
0005
0006  * ACCTIME/S
0007  * MEASURES ELAPSED TIME FOR EXTENDED BASIC
0008  * START MEASUREMENT WITH CALL LINK("STRTIM")
0009  * STOP MEASUREMENT WITH CALL LINK("STPTIM",M,S)
0010  * WHERE:
0011  *       M IS MINUTES ELAPSED SINCE STRTIM LINK
0012  *       S IS SECONDS AFTER M MINUTES
0013  * PUBLIC DOMAIN SOFTWARE
0014  *
0015  *    CODE BY:  Bruce Harrison
0016  *    PUBLIC DOMAIN
0017  *    28 November 1994
0018  *
0019        DEF  STRTIM,STPTIM ENTRY POINTS
0020  *
0021  * REQUIRED EQUATES
0022  *
0023  GPLWS  EQU  >83E0        GPL WORKSPACE
0024  FAC    EQU  >834A        FLOATING POINT ACCUMULA-
TOR
0025  NUMASG EQU  >2008        NUMERIC ASSIGNMENT
0026  XMLLNK EQU  >2018        XML LINK VECTOR
0027  CIF    EQU  >20          CONVERT INTEGER TO F.P.
0028  *
0029  * MAIN CODE
0030  *
0031  STRTIM CLR  @TIMCNT      CLEAR TIME COUNT
0032         MOV  @INTLOC,@>83C4 SET USER INTERRUPT
0033         LWPI GPLWS        LOAD GPL WORKSPACE
0034         B    @>6A         BACK TO GPL INTERPRETER
0035  STPTIM CLR  @>83C4       CLEAR USER INTERRUPT
0036         LWPI WS           LOAD OUR OWN WORKSPACE
0037         MOV  @TIMCNT,R3   GET TIME IN 60THS
0038         CLR  R2           CLEAR REGISTER 2
0039         DIV  @SIXTY,R2    R2 HAS SECONDS
0040  * AT THIS POINT R3 HAS REMAINDER IN 60THS
0041         MOV  R2,R5        SECONDS INTO R5
0042         CLR  R4           CLEAR R4
0043         DIV  @SIXTY,R4    R4 HAS MINUTES
0044  * AT THIS POINT R5 HAS REMAINDER IN SECONDS
0045  * FOLLOWING SECTION ROUNDS OFF THE RESULTS
0046         CI   R3,30        COMPARE LEFTOVER 60THS
0047         JLT  NORND        IF LESS THAN 30, SKIP
0048         INC  R5           ELSE INCREMENT SECONDS
0049         C    R5,@SIXTY    COMPARE TO 60
0050         JLT  NORND        IF LESS JUMP AHEAD
0051         INC  R4           ELSE INCREMENT MINUTES
0052         CLR  R5           AND CLEAR SECONDS
0053  NORND  CLR  R0           NON-ARRAY VARIABLES
0054         LI   R1,1         FIRST PARAMETER
0055         MOV  R4,@FAC      MOVE MINUTES TO FAC
0056         BLWP @XMLLNK      USE XML
0057         DATA CIF          INTEGER TO F.P.
0058         BLWP @NUMASG      ASSIGN 1ST PARAMETER
0059         INC  R1           NEXT PARAMETER
0060         MOV  R5,@FAC      SECONDS TO FAC
0061         BLWP @XMLLNK      USE XML
0062         DATA CIF          CONVERT
```

(See Page 11)

EXTENDED BASIC PROGRAMS—

(Continued from Page 9)

```
110 RD=-1 :: CD=-1
120 FOR X=1 TO 2
130 FOR Y=1 TO N
140 ROW=ROW+RD
150 CALL HCHAR(ROW,COL,CN)
160 NEXT Y
170 FOR Y=1 TO N
180 COL=COL+CD
190 CALL HCHAR(ROW,COL,CN)
200 NEXT Y
210 RD=1 :: CD=1
220 NEXT X
230 N=N+2 :: COL=COL+1 :: RO
W=ROW+1
240 NEXT Z
250 CN=CN+1 :: IF CN<58 THEN
 70
260 CALL LINK("STPTIM",M,S):
: ON BREAK STOP
270 IF S<10 THEN S$="0"&STR$
(S)ELSE S$=STR$(S)
280 PRINT "MINUTES:SECONDS"
290 PRINT STR$(M);":";S$
```

SPIRAL2
IMPROVED VERSION

```
10 CALL INIT :: CALL LOAD("D
SK1.ACCTIME/O")
20 ON BREAK NEXT :: CALL LIN
K("STRTIM")
30 ! SPIRAL2
40 ! JON KELLER
50 ! BLUEGRASS 99 4/86
60 ! REVISED VERSION BY B. H
ARRISON - 30 NOV 94
70 CALL CLEAR :: FOR CN=48 T
O 57
80 ROW=12 :: COL=16
90 FOR N=0 TO 22 STEP 2
100 FOR ROW=ROW TO ROW+N
110 CALL HCHAR(ROW,COL,CN)
120 NEXT ROW
130 IF N=22 THEN 240
140 FOR COL=COL TO COL+N
150 CALL HCHAR(ROW,COL,CN)
160 NEXT COL
170 FOR ROW=ROW TO ROW-N-1 S
TEP -1
180 CALL HCHAR(ROW,COL,CN)
190 NEXT ROW
200 FOR COL=COL TO COL-N-1 S
TEP -1
210 CALL HCHAR(ROW,COL,CN)
220 NEXT COL
230 NEXT N
240 NEXT CN
250 CALL LINK("STPTIM",M,S)
: ON BREAK STOP
260 IF S<10 THEN S$="0"&STR$
(S)ELSE S$=STR$(S)
270 PRINT "MINUTES:SECONDS"
```

EXTENDED BASIC MYTHS—

(Continued from Page 10)

```
0063        BLWP @NUMASG      ASSIGN 2ND PARAMETER
0064        LWPI GPLWS        LOAD GPL WS
0065        B    @>6A         EXIT TO GPL INT.
0066  *
0067  * THE INTERRUPT ROUTINE IS THIS:
0068  *
0069  USRINT INC  @TIMCNT      INCREMENT TIME COUNT
0070        RT                THEN RETURN
0071  *
0072  * DATA SECTION
0073  *
0074  WS     BSS  32           OUR OWN WORKSPACE
0075  TIMCNT DATA 0            TIME COUNT WORD
0076  INTLOC DATA USRINT       INTERRUPT LOCATION
0077  SIXTY  DATA 60           60 AS A WORD
0078        END
0079
0080  NEXT, THE TIMER FOR LONGER DURATIONS
0081  (ANY AMOUNT YOU LIKE)
0082  ASSEMBLED, THIS TAKES 198 BYTES IN MEMORY
0083
0084  * LONGTIME/S
0085  * FOR MEASURING TIMES LONGER THAN 18 MINUTES
0086  * MEASURES ELAPSED TIME FOR EXTENDED BASIC
0087  * START MEASUREMENT WITH CALL LINK("STRTIM")
0088  * STOP MEASUREMENT WITH CALL LINK("STPTIM",M,S)
0089  * WHERE:
0090  *       M IS MINUTES ELAPSED SINCE STRTIM LINK
0091  *       S IS SECONDS AFTER M MINUTES
0092  * PUBLIC DOMAIN SOFTWARE
0093  *
0094  *    CODE BY:  Bruce Harrison
0095  *    PUBLIC DOMAIN
0096  *    30 November 1994
0097  *
0098        DEF  STRTIM,STPTIM ENTRY POINTS
0099  *
0100  * REQUIRED EQUATES
0101  *
0102  GPLWS  EQU  >83E0        GPL WORKSPACE
0103  FAC    EQU  >834A        FLOATING POINT ACCUMULA-
TOR
0104  NUMASG EQU  >2008        NUMERIC ASSIGNMENT
0105  XMLLNK EQU  >2018        XML LINK VECTOR
0106  CIF    EQU  >20          CONVERT INTEGER TO F.P.
0107  *
0108  * MAIN CODE
0109  *
0110  STRTIM CLR  @TIMCNT      CLEAR TIME COUNT
0111        CLR  @CARRY       CLEAR THE CARRY WORD
0112        MOV  @INTLOC,@>83C4 SET USER INTERRUPT
0113        LWPI GPLWS        LOAD GPL WORKSPACE
0114        B    @>6A         BACK TO GPL INTERPRETER
0115  STPTIM CLR  @>83C4       CLEAR USER INTERRUPT
0116        LWPI WS           LOAD OUR OWN WORKSPACE
0117        MOV  @TIMCNT,R3   GET TIME IN 60THS
0118        CLR  R2           CLEAR REGISTER 2
0119        DIV  @SIXTY,R2    R2 HAS SECONDS
0120  * AT THIS POINT R3 HAS REMAINDER IN 60THS
0121        MOV  R2,R5        SECONDS INTO R5
0122        CLR  R4           CLEAR R4
0123        DIV  @SIXTY,R4    R4 HAS MINUTES
0124  * AT THIS POINT R5 HAS REMAINDER IN SECONDS
0125        MOV  @CARRY,R7    GET CARRY NUMBER IN R7
0126        JEQ  CHKRND       IF ZERO, SKIP AHEAD
0127        MOV  R7,R9        ELSE COPY INTO R9
0128        MPY  @CMINS,R7    MULTIPLY R7 BY 18
0129        A    R8,R4        ADD TO MINUTES
0130        MPY  @CSECS,R9    MULTIPLY R9 BY 12
0131        A    R10,R5       ADD TO SECONDS
0132  CHKRND CI   R3,30        REMAINING 60THS
0133        JLT  CHKSEC       IF <30, SKIP
0134        INC  R5           ELSE ROUND UP SECONDS
0135  CHKSEC C    R5,@SIXTY    COMPARE SECONDS TO 60
0136        JLT  NORND        IF LESS, SKIP AHEAD
0137        INC  R4           ELSE INCREMENT MINUTES
0138        AI   R5,-60       SUBTRACT 60 FROM SECONDS
0139        JGT  CHKSEC       IF ABOVE ZERO, CHECK AGAIN
0140  NORND  CLR  R0          NON-ARRAY VARIABLES
0141        LI   R1,1         FIRST PARAMETER
0142        MOV  R4,@FAC      MOVE MINUTES TO FAC
0143        BLWP @XMLLNK      USE XML
0144        DATA CIF          INTEGER TO F.P.
0145        BLWP @NUMASG      ASSIGN 1ST PARAMETER
0146        INC  R1           NEXT PARAMETER
0147        MOV  R5,@FAC      SECONDS TO FAC
0148        BLWP @XMLLNK      USE XML
0149        DATA CIF          CONVERT
0150        BLWP @NUMASG      ASSIGN 2ND PARAMETER
0151        LWPI GPLWS        LOAD GPL WS
0152        B    @>6A         EXIT TO GPL INT.
0153  *
0154  * THE INTERRUPT ROUTINE IS THIS:
0155  *
0156  USRINT INC  @TIMCNT      INCREMENT TIME COUNT
0157        JNE  INTEX        IF NOT ZERO, EXIT
0158        INC  @CARRY       ELSE INCREMENT CARRY
0159        A    @HEXF,@TIMCNT ADD ROLLOVER CORRECTION
0160  INTEX  RT                THEN RETURN
0161  *
0162  * DATA SECTION
0163  *
0164  WS     BSS  32           OUR OWN WORKSPACE
0165  TIMCNT DATA 0            TIME COUNT WORD
0166  INTLOC DATA USRINT       INTERRUPT LOCATION
0167  SIXTY  DATA 60           60 AS A WORD
0168  CARRY  DATA 0            CARRY FOR TIMES >18 MIN 12
SEC
0169  CMINS  DATA 18           MINUTES FOR EACH CARRY
COUNT
0170  CSECS  DATA 12           SECONDS FOR EACH CARRY
COUNT
0171  HEXF   DATA >F           1/4 SECOND IN 60THS
0172        END
```

1995 is a good year to attend a TI fair

Jack Armstrong to the rescue!

(The following article is reprinted from the December 1994 St. Louis Computer Bridge.*)*

By HAROLD C. HOYT JR.

Charlotte has been hooked on Hardy Boy mysteries. She has read more than 250 of them. After I downloaded the public library list of Hardy Boy mysteries and printed it out, Charlotte systematically read and checked off every one they had. If that wasn't enough, she went on a search and found a bunch more from other sources.

Her interest in the Hardy Boys seems to be finally petering out, like her earlier interest in the Teen Age Mutant Ninja Turtles, but now she shows an interest in Goosebumps, a series of scary stories well suited for her age group. She seems to have developed a permanent interest in detective and spy stories. The other day, she was filling up piles of papers with substitution codes for encoding and decoding secret messages like the Jack Armstrong secret decoder ring we had as kids.

You saved three boxes from — ? Was it Wheaties? No, some other cereal, and mailed it in with some small change and waited what seemed like forever for it to come in the mail. With the ring, you could decode the secret message broadcast at the end of each weeknight program. Each message had something to do with the continuing adventures of Jack Armstrong. There were a whole bunch of adventure programs on radio aimed for kids. The Green Hornet, Mr. District Attorney, Inner Sanctum, Mr. Keen, Tracer of Lost Persons, The Shadow sponsored by Blue Coal.

When viewing the junk for kids on TV, my selective memory says, "Boy, radio was better." As an adult, I've listened to some nostalgic recordings and will have to amend my colored childhood memories to state that *some* of radio was better. I was a grownup and cried when NBC radio introduced their new format "Monitor radio" and all the series were gone, canceled, vaporized. We were usually in the car going somewhere when The Shadow came on. In one hour, the crime was committed, and solved by Lamont Cranston and his beautiful fiancee, Margo Lane. Who knows what evil lurks in the hearts of men? The Shadow knows! Heh, heh, heh.

Many of the TI community have had a hand at encryption codes and it's good, clean fun. Dr. Roy Tamashiro and Tigercub Jim Peterson pop into mind. You can make a substitution code that is a real dog to break if you don't have a 99/4A just using the pseudo-random pattern generator in XBASIC. What makes it miserable is that the patterns don't repeat for a long time so what you use for the letter E, for instance, keeps changing. With a long message and straight kid stuff substitution, just count which character occurs the most often, and that is probably E, since the letter E is used more often than any other in the English language. Did you notice that the phrases are getting more difficult in Wheel of Fortune? Guessing an E for a vowel is no longer safe.

Without some clue, there are 26!-1 possible ways to rearrange letters for substitution. You can have any letter A through Z for the first letter that corresponds to your A that is to be encoded. Each of these sub groups is in turn divided into 25 sub-sub-groups. If A is the first letter, then any of 25 letters, B through Z, can be second. A third layer of 24 letters fans out from each of these 25 subgroups. A total of 26! combinations is possible. The -1 comes from excluding the arrangement where none of the letters are changed. It gets messier if you change some of the coded symbols to lowercase. You could mix in a few numbers with letters and interchange a letter randomly for some of the space characters so your adversary doesn't have a clue how long a word is. That ought to confuse the bad guys. It's been a while, but I finally got a little of my XBASIC back. How quickly you lose it if you don't use it!

CODE

```
1 !SAVE DSK1.CODE !046
100 !Prog H. Hoyt Jr.12/7/94
 !228
110 !Letter substitution cod
es for Charlotte !171
120 DIM X$(26)!165
130 DATA ZYXWVUTSRQPONMLKJIH
GFEDCBA !084
135 DATA ZYXWVUTSRQPONMLKJIH
GFEDCBA !084
140 READ X$(0):: FOR I=1 TO
26 :: X$(I)=SEG$(X$(0),I,1):
: NEXT I :: CALL CLEAR !097
145 R,C=3 !002
150 CALL KEY(3,K,S):: IF S<1
 THEN 150 !223
160 IF K=32 OR(K>64)*(K<91)T
HEN 170 ELSE 150 !147
170 CALL HCHAR(R,C,K):: IF K
>64 THEN CALL HCHAR(R+12,C,A
SC(X$(K-64)))ELSE CALL HCHAR
(R+12,C,32)!231
180 C=C+1 :: IF C>29 THEN R=
R+1 :: C=3 !137
190 IF R>10 THEN C,R=3 :: C
LL CLEAR !198
200 GOTO 150 !229
```

The above XBASIC program allows you to type in text starting on Row 3, Column 3, filling up about a half-screen page before clearing itself and starting over. As written, you will have to type FCTN4 to break out. It would be easy to rewrite line 190 to display CONTINUE?(Y/n) for a clean exit. As the teachers used to say, "Let's leave some of the fun for the reader."

The above program was first written and tested using the Swartz v5.01 TIEMULATOR! on a 486 and then tested again on a real TI99/4A. A Terminate and Stay Resident (TSR) PRINDIR v8.02 program was used to intercept and redirect a request to LIST"PIO" from inside the XBASIC simulation of the working CODE program. The listing was redirected to a DOS text file which was then EDITed to remove a few kinks due to the way the 4A lists programs. All programs are listed with a maximum of 80 characters to a next line, even if it cuts a command in half. The text was further EDITed to make the

(See Page 13)

JACK ARMSTRONG TO THE RESCUE —

(Continued from Page 12)

maximum line length 28 characters, which is the way it is displayed on screen.

Type in the program, it really isn't very long. Here is how it works: Line 120 reserves 27 spaces for X$. Remember, 0 is a number, too! Line 130 has 26 ASCII characters read in as one string into X$(0) into X$(I) one character at a time. One of Charlotte's choices of substitution was to reverse the letters. A becomes Z, B becomes Y, etc.

Line 135 is a duplicate of line 130. After a while you learn some time-saving tips to make programming in XBASIC easier. Edit line 130 all you want. To get back to where you started from, simply type 135 FCTN-X (down arrow) and then Enter, FCTN-8 (redo) to put you back at the beginning of the line and type over 135 changing it to 130 and then Enter and your line 130 DATA is restored. If you are going to use funny stuff for letters, such as leading spaces, enclose the 26-character DATA on line 130 in quotes.

How the program runs:

Type RUN (Enter). The screen clears. Type anything to be coded. The input appears at the top of the screen and the coded output appears at the bottom half of the screen. In the simple case of A through Z being reversed, simply typing the coded message back in will decode it. Unfortunately, in most cases, you will have to activate a complementary DATA statement to decode the message. Let's comment out 135 by making line 135 read:

135 !DATA... (This is for comparing)->abcdefghijklmnopqrstuvwxyz

A new DATA line 136 DATA AYBWCUDSEQFOGMHKIZXVTRPJLN is installed. Line 137 DATA ACEGIKMOQXPYNZLWJVHUFTDSBR will decode any transposition created by line 136. Leave all the DATA lines in the program in pairs. Line 137 would be activated to decode lines encoded by line 136. With line 136 active, typing an A will return an A, so the first DATA character of line 137 is A. Typing a B with line 136 produces a Y as code, so to get your B back when typing a Y, you need to have a B in the 25th position of the DATA in line 137.

With a little more work, we could make the program automatically code and decode and display both from a single DATA key. For now, we will settle for a letter transposition coder that works because this newsletter is getting close to deadline.

THE ART OF ASSEMBLY — PART 43

Bitmap explained

By BRUCE HARRISON

Many years ago, your author found a book in the Navy Department's library called *Television Really Explained*. This turned out to be a British publication. It was intended to explain the operation of a television system in terms for the most rank beginner. One line from that book sticks in the mind to this day. It describes the motion of the electron beam in the picture tube as "moving across the screen at amazing speed." Nicely said! No need to understand microseconds, reflex scanning systems, or anything technical like that. Just "amazing speed." We could sum up this whole series of articles just that simply: Assembly language executes at "amazing speed."

LAST MONTH'S SIDEBAR

Now's the time to get out last month's issue. We'll wait.

Okay, now find the page with our Sidebar 42 on it.

Here we go, ready or not. Down to the label SINWAV is mostly pretty standard stuff, including a menu setup to offer a selection of actions, and an exit routine to get out of the program. Just above the label EXIT is the first thing that's different from the usual. Before branching out to the selected item, the program has a BL @SETBM. That uses a subroutine to set all the required conditions for the bitmap mode of operations. From there on, each operation takes place in bitmap. Let's now skip down the page a bit to label SETBM, in file BITSUBS. This, with the auxiliary subroutine CPDT, is all that's needed to take your TI from its normal graphics mode into a cleared bitmap screen.

SETBM begins with a few "write to register" operations, which set up the VDP tables to locations compatible with the bitmap mode. The screen image table is set to >1800, the pattern descriptor table to 0, the color table to >2000, and the sprite descriptor table and sprite attribute list are both set to >3800. (This program does not use sprites, so the tables are set there just to insure we don't get any undesired sprites by accident.) Once that's done, the table areas are filled with the appropriate bytes to create a blank screen, black-on-white color combination and to fill the screen image table with bytes from 0 through 255, repeated three times. The screen image table, then, is left alone at all times, while the pattern table, with three sets of 255 eight-byte character patterns, determines what's seen on the screen. Things get written to the screen (and colorized) when you write individual bits to the pattern descriptor table and bytes to the color table.

THE SINE WAVE

Now that we've used SETBM to put us in bitmap mode, let's go back into the main part of our code, at label SINWAV. As the name implies, this will plot a sine wave on the screen as single pixel dots. It starts by pointing R12 at the large block of data labeled SINDAT. Next it sets R9 to >C000, which will make the dots plotted appear in dark green. We complicated things a little here by making the upper half of the sine wave plot in green, and the lower half in red. In the data that R12 is pointing to, each pair of words gives the X (horizontal) and Y (vertical) values for one spot on the sine curve. Thus, when we MOV *R12+,R7, we've taken the X value into R7. This value is expressed in dot-row co

(See Page 14)

THE ART OF ASSEMBLY—

(Continued from Page 13)

ordinates. We add 20 just to move the sine wave in a bit from the screen edge. Now we get the vertical coordinate into R8 by MOV *R12+,R8. At this point we check the value in R12 to see whether we're still in the positive half-cycle. If we are, we skip ahead to label POSWV, otherwise we change the color nybble in R9 to >6000, so the points will be plotted in dark red. Now the number in R8 is set up for a zero-based vertical coordinate, but what we want is to have things centered around dot-row 96, or halfway down the screen. Thus we take the number in R8 and NEG it, then add our offset of 96, and we're ready to place one pixel on the screen.

To do that, we use the subroutine PLOT. PLOT takes the numbers in R7 and R8, and calculates from them the exact bit that represents the pixel at those coordinates. It uses Modulo math processes to accomplish this, and the source code is pretty well annotated. The methods used here to calculate Modulos are short cut methods that take full advantage of the number base in use, and are not general purpose Modulo routines. Let's say we wanted to calculate 10 Modulo 3. The general purpose way to do this would be as follows:

```
LI  R3,10
CLR R2
LI  R1,3
DIV R1,R2
```

After this DIV, R3 would contain 10 modulo 3, which happens to equal 1, since that's the remainder after dividing 10 by 3. The quotient 3 would be left in R2, but that's of no importance. This same thing could be done in BASIC, by taking this kind of steps: A=3 :: B=10 :: M=(B/A-INT(B/A))*A, M would then contain the answer 1. (On some computers, M would contain .9999999... instead of 1, but you get the idea, or at least we hope you do.)

Dividing, however, takes time, so the subroutine PLOT uses shortcuts that take advantage of the fact that it's MODULO 8 that we're after, thus doing its calculations more quickly. Once it has done its Modulo math, the routine reads one byte from VDP, sets just the correct bit in that byte, than writes the byte back to VDP. This places one dot at the correct spot on the screen in the dot-row, dot-column coordinate system. The last part of PLOT checks to see if R9 is non-zero, which means that a color has been specified. It then checks the appropriate byte in the color table by adding >2000 to R0 and doing a VSBR. If the color specified by R9 has already been set in that byte, it's left alone. Otherwise the routine sets the color from the left nybble of R9 into the foreground nybble, and writes that byte back into the color table in VDP.

SINWAV then just keeps repeating all of this until it runs out of data, and by then the sine curve is all plotted on the screen. Now we put in a visible X-axis by plotting a line with the code at LINELP. This puts 255 dots across dot-row 96. The next trick is to print the legends on the screen. That uses the subroutines at BITSTR and CHAR.

BITSTR is similar in concept to our old routine DISSTR for displaying strings in the standard graphics and text modes. Before calling BITSTR, we set R8 to the desired graphics row, R7 to the desired graphics column, R9 to the color scheme desired and R12 to the address of the string to be displayed. As in the case of VSBW-STR, the string begins with a length byte, followed by the text content of the string. BITSTR starts by stashing the R11 return address in R15, then saving the starting column number in R13. Next it gets the length of the string into R4 and, if that's zero, jumps to the exit. Assuming the length is at least one byte, that number is right justified in R4 to serve as a character count for the loop at BITST0. That loop gets a byte from the string into R6, right justifies that byte, then uses the subroutine CHAR to display it on the bitmap screen. BITST0 continues this process, incrementing the column in R7 on each pass, until the count in R4 becomes zero.

The subroutine CHAR is somewhat strange, in that it writes the eight bytes of the character pattern into the pattern table, rather than the character itself into the screen image table. Back at the early part of this program, we grabbed all the existing character definitions for the graphics mode from VDP >800 into memory at location CHRTBL. We did that precisely for the use that CHAR will make of that data at CHRTBL. CHAR starts by getting the desired row from R8 into R0. Remember that these are not Dot-Rows, but the normal 1-24 graphics row numbers. Char decrements the row count in R0 to zero-base the number. Next, it loads R2 with 8, which we'll need later. Now the number in R0 gets multiplied by 32, so it represents a graphics screen position. The desired graphics column from R7 is added, then R0 is DEC'd again to zero-base the column. Since the character definition table is in eight-byte chunks, we multiply the number now in R0 by eight, so it points to the correct part of the pattern definition ta Now to get the character data, we take the desired character number from R6 into R1, multiply that by eight to index into our CHRTBL block in memory, then add the address of the start of the saved character definitions. (AI R1,CHRTBL)

At this point, R0 points to the correct place in the character definition table in VDP, R1 points to the pattern for the desired character in CHRTBL, and R2 contains 8, the number of bytes needed to define one character. BLWP @VMBW makes that desired character appear at the desired row and column on the screen. Now R9 is checked, and the desired color scheme for this character is written to the color table if needed. By this method, we could in theory write each character in a string in a different color scheme. (The present subroutine BITSTR doesn't allow that, but the creative programmer can see how to do it.)

The ability to write strings of characters at various places on the screen with different color schemes, as this program does, is an important difference between bitmap and the normal graphics mode. That, and the ability to write selectively to any pixel on the screen, makes bitmap mode a powerful tool. The one significant drawback is that almost all the VDP RAM space gets eaten up. In this case, that's no problem, but if your program needs VDP space for file buffers and such, then the crunch can get serious.

When the sine wave is all plotted and all the legends are printed, it executes a BL @KEY, which just puts the computer into a key-sensing loop until any key has been pressed. Given a keypress uses the subroutine CPDT (Clear Pattern Descriptor Table) to clear the bitmap screen, then uses SETGM (Set Graphics Mode)

(See Page 15)

THE ART OF ASSEMBLY—

(Continued from Page 14)

put the VDP back into its "normal" graphics mode. This routine starts by re-setting all the VDP registers except #0 to their normal values, then puts back the color table and character definition table data that was saved at the start of the program. Finally, with everything in place, it clears VDP register 0, to put the VDP back in graphics mode, and returns to the main code. After BL @SETGM, the SINWAV exits by branching back to label MENU.

The other two "main" routines, called SPIRAL and BOXES, work in a similar fashion, except that no curves are plotted, and all lines drawn are straight horizontal and vertical. They use the same subroutines to do these jobs, so we'll spare you the description of all that, and leave analysis of SPIRAL and BOXES as an exercise for the serious student.

Now is the time to mention one small limitation to the capability of bitmap. One can individually set pixels on and off, but the color table entries of eight bits each affect eight pixels in the pattern table. Thus, if we set a color byte for a spot in the image, a total of eight pixels including that one will have this same color scheme.

Let's try to illustrate this. Imagine that we are looking at a single-byte section of the pattern descriptor table and the corresponding byte in the color table:

Pattern Table location >0508 - bits 00101100
Color Table location >2508 - byte >4F

With this setup, each of the ones in the eight bits at >0508 will result in a dark blue pixel on the screen, while each of the zeros in those eight bits will be a white spot on the screen. Thus this one byte in the color table affects eight adjacent pixels in the image we see, so when we set a color scheme for any of the pixels at >0508 by writing a color byte to >2508, the background and foreground colors we've set apply to all eight of those pixels. This means that, while we can turn individual pixels on or off one at a time, we can only set their colors in groups of eight, not by individual pixel.

After getting our feet wet in bitmap with the program for last month's column, we tried getting even more adventurous, and created a drawing program. In future columns, we'll pass along some more things learned from that effort, including "UNPLOT," to erase a single pixel from the bitmap screen, and an algorithm to create straight lines from any dot-row, dot-column position to any other dot-row, dot-column position. The drawing program itself has been released to public domain, and is available from the Lima Users Group. (Disk 928 in their library, which includes all the source code and sample pictures.) That program is really just a toy, but we have used it to make drawings for our children's science fair projects, one of which won a blue ribbon.

Next month we'll cover two topics: The first will be how to deal with those prompts the assembler issues, and then the much more difficult topic of doing a CALL FILES from assembly. Good luck with your own experiments.

Hardware project

Power supply upgrade

By STEVE EGGERS

It is assumed that you know how to use a soldering iron, multimeter and have a basic understanding of electronics to perform this upgrade. *I will not be held liable if anything should go wrong doing this project! You do this at your own risk!*

TOOLS/ITEMS NEEDED:

- Soldering iron
- Solder
- Desoldering tool or desoldering wick (I used this!)
- Electrical tape
- Masking tape
- Phillips screwdriver
- Socket set
- Circuit board posts (get at local computer store)
- Silicon cement (RTV)

I used a power supply I had laying around in a mini-tower case, 200-Watt power supply. This is adequate for the Geneve and other cards.

Step 1: Open the lid on the TI Peripheral Expansion Box and remove and disconnect the disk drive from the disk controller card and power cable. Remove all cards from PEB.

Step 2: Remove the eight screws from the card cage at the inside bottom of the PEB. The cage should come out.

Step 3: Remove all visible screws on the PEB (14 total) except the 2 holding the lid retaining clips. The case should slide forward and come off. Put to the side.

Step 4: The power supply should be visible. Remove the transformer, fan and circuit board and bracket. Remove the clip plugs on the circuit card. Unplug all wires leading to the AC plug/fuse and to switch and discard. Leave the ground (top) wire on the power receptacle.

Step 5: Now the power supply area should be empty. Remove the screw posts where the transformer sat. (I accomplished this by getting a pair of pliers and wiggling the posts back and forth till I could push them out the bottom of the box.)

Step 6: With the new power supply, *IMPORTANT: Ensure you make note how the power receptacle is wired, as well as the power switch!* If there is a 110/220 Volt switch, desolder the wires at the 110 end and connect (solder) them together. Desolder all wires connected to the power receptacles. Discard the power output receptacle. You should have a switch cable with four wires in it, black, white, brown and blue.

Remove the fan. Remove the power supply circuit card.

Step 7: Install new fan into the PEB using the TI hardware. It should be an exact match.

Step 8: Install foot pegs to the new power supply circuit card.

Step 9: Take the switch cable and solder the positive wire (white) to the left ter-

(See Page 16)

POWER SUPPLY—

(Continued from Page 15)

minal post on the AC power receptacle in the PEB (as you look at it in the power supply area). Do the same with the negative wire (black) to the right side of AC receptacle.

Step 10: Find a place for the new P/S circuit board to sit without covering any screw holes for the PEB case. Once you find a spot, put four globs of silicon cement and set in the foot pegs on the P/S. *Let dry!*

Step 11: While glue is drying, desolder the green, yellow, black and brown wires off the expansion bus card.

Step 12: Once glue is dry, find the power plug on the power supply itself. It should be a plug with a white and black wire coming out of it. Take the power switch cable and solder the *brown* wire to the *black* wire and solder the *white* wire to the *blue* wire.

Step 13: Connect the fan to the circuit board.

Step 14: Connect the four leads on the switch cable to the power switch. It goes as follows as you look at the back of switch (rocker type):

BLUE BROWN
WHITE BLACK

These steps are performed with power on! Use caution!

Step 15: Separate the equipment cables (P8 & P9) from the disk drive cables. Connect the power cord to the PEB and turn on power. *NOTE: Fan may or may not be running. IBM power supplies have a loading circuit in them.* (I connected an old disk drive to a power cable to provide a load for the circuit.)

Step 16: With power on, take a multimeter set to DC Volts and in the equipment cables (P8 & P9), find +5 Volts, +12 Volts, -12 Volts and ground. Once you find these, turn off power and take masking tape and mark wires accordingly.

Step 17: Cut these wires from the plug, strip insulation back and connect to the Expansion Bus Card as follows:

+5 to green (GRN) wire
+12 to brown (BRN) wire
-12 to yellow (YLW) wire
Ground to black (BLK) wire

After soldering, ensure that the length coming through the bottom of the bus card is adequate and not touching the bottom case. Cut to desired length!

Step 18: Route disk drive cables to drive bay. (I routed only two cables.)

Step 19: Coil up unused cables and place out of the way in the power supply area.

Step 20: Put PE box back together.

Now the fun part. *All* expansion cards have to be modified. All regulators need to be bypassed. Put a jumper on the two outer legs. *Be sure the wire does not touch the center pin!* I used telephone wire for this, since it is a small-gauge wire and also insulated.

All five regulators on the Geneve card need to be jumpered, as well as any other cards.

That's it! Any questions, call me, Steve Eggers, at (915) 695-4604.

Myarc Advanced BASIC

A program to print labels for 3.5-inch disks

By JIM UZZELL
©1995 DDI Software

Now that we have the capability of using 3.5-inch 1.44-megabyte floppy drives, which I am currently using, I decided I needed a way to create a disk label, so I wrote the following program.

This program will probably not work for everyone, because of the diverse features of printers.

To help determine if you need to modify the program, the following line numbers are explained:

Line 260 — CHR$(15)=Condensed; CHR$(27);CHR$(65);CHR$(6)=Line spacing 6/72

Line 270 — CHR$(27);CHR$(71)=Bold on; CHR$(72)=Bold off; CHR$(9)=9/72

Line 280 — CHR$(27);CHR$(83);CHR$(0)=Superscript; CHR$(6)=6/72; CHR$(27);CHR$(88);CHR$(0);CHR$(47)=Margins (0=Left 47=Right)

Trim the listing on the dotted line, then attach using paper stick glue. This program will automatically select a short form or long form label. The short form will print a maximum of 84 files on the front. The long form will be selected if there are more than 84 files, up to a maximum of 112. The long form should be creased by placing it on a disk and folding over to the back before applying glue.

DISKLABEL

```
100 CALL GRAPHICS(2,2)
110 F$="| " :: G$="|" :: C$=
"########## " :: DIM B$(128)
,Z$(32),B(32,32)
120 E$="--------------------
--------------------------"
130 H$="|_____________DDI SO
FTWARE_(C)1994_____________|"
140 GOSUB 430
150 XT=84 :: DISPLAY AT(23,1
)BEEP :"WHICH DRIVE i.e DSK1
.":
160 ACCEPT AT(24,1):D$
170 OPEN #1:D$,INPUT ,RELATI
VE,INTERNAL
180 INPUT #1:A$,Z1,Z2,Z3
190 I=I+1
200 INPUT #1:B$(I)
210 IF LEN(B$(I))=0 OR EOF(1
) THEN 230
220 GOTO 190
230 I=I-1
240 CLOSE #1 :: IF I>84 THEN
 XT=112
250 OPEN #1:"PIO"
260 PRINT #1:CHR$(15);CHR$(
7);CHR$(65);CHR$(6);E$
270 PRINT #1:CHR$(15);F$;CHR
```

(See Page 17)

LABELS—

(Continued from Page 16)

```
(27);CHR$(71);"VOL ";A$;TAB
(36);DATE$ ;CHR$(27);CHR$(72
);TAB(52);G$;CHR$(27);CHR$(6
5);CHR$(9)
280 PRINT #1:CHR$(27);CHR$(8
3);CHR$(0);CHR$(27);CHR$(65)
;CHR$(6);CHR$(27);CHR$(88);C
HR$(0);CHR$(47);
290 PRINT #1:F$;"Free: ";Z3;
"    Used: ";Z2-Z3;TAB(57);G
$
300 FOR X=1 TO XT
305 IF XT>84 AND X=13 THEN P
RINT #1:F$;TAB(47);G$ :: PRI
NT #1:F$;TAB(47);G$
310 IF X=1 OR X=5 OR X=9 OR
X=13 OR X=17 OR X=21 OR X=25
 OR X=29 OR X=33 OR X=37 OR
X=41 THEN PRINT #1:F$; :: GO
TO 330
320 IF X=45 OR X=49 OR X=53
OR X=57 OR X=61 OR X=65 OR X
=69 OR X=73 OR X=77 OR X=81
THEN PRINT #1:F$;
325 IF X=85 OR X=89 OR X=93
OR X=97 OR X=101 OR X=105 OR
 X=109 THEN PRINT #1:F$;
330 PRINT #1,USING C$:B$(X);
340 IF X=4 OR X=8 OR X=12 OR
 X=16 OR X=20 OR X=24 OR X=2
8 OR X=32 OR X=36 OR X=40 OR
 X=44 THEN PRINT #1:G$ :: GO
TO 360
350 IF X=48 OR X=52 OR X=56
OR X=60 OR X=64 OR X=68 OR X
=72 OR X=76 OR X=80 OR X=84
THEN PRINT #1:G$
355 IF X=88 OR X=92 OR X=96
OR X=100 OR X=104 OR X=108 O
R X=112 THEN PRINT #1:G$
360 NEXT X
370 PRINT #1:H$
380 PRINT #1:CHR$(27);"@" ::
 CLOSE #1
390 DISPLAY AT(23,1):"ANOTHE
R DISK  Y/N" :: DISPLAY AT(2
4,1):RPT$(" ",40)
400 CALL KEY(0,K,S) :: IF S<
1 THEN 400
410 IF K=78 OR K=110 THEN 42
 ELSE I=0 :: CALL MEMSET(B$
(),"") :: GOTO 150
420 CALL RESETPLT :: CLS ::
END
430 CALL PALETTE(1,7,7,7) ::
 CALL PALETTE(13,8,3,7) :: C
ALL PALETTE(9,7,5,4) ::CALL
PALETTE(11,3,8,8)
440 !Converted by ASM2MYB fr
om DDI SOFTWARE
450 CALL INIT
460 CALL LOAD(-8352,71,69,84
,32,32,32,36,248)
470 CALL LOAD(8194,39,96,223
,96)
480 CALL LOAD(9460,0,6,0,0,2
00,11,36,246,2,224,240,0,2,4
,0,0,2,0,0,14,192,96)
490 CALL LOAD(9482,37,94,192
,160,37,88,193,96,37,92,193,
160,37,90,2,7,37,96,208,247,
11,131)
500 CALL LOAD(9504,208,247,4
4,32,36,244,16,1,208,247,6,1
95,11,67,44,32,36,244,5,129,
129,65)
510 CALL LOAD(9526,19,7,11,1
95,44,32,36,244,5,129,129,65
,19,1,16,241,192,96,37,94,5,
130)
520 CALL LOAD(9548,129,130,1
9,1,16,235,194,224,36,246,4,
91,0,88,0,120,0,138,0,106,25
5,255)
530 CALL LOAD(9570,255,255,2
55,255,255,255,255,255,255,2
55,255,255,255,255,255,255,2
55,255,255,255,255,255)
540 CALL LOAD(9592,255,255,2
55,255,255,255,255,255,255,2
55,255,255,255,255,255,255,2
55,255,255,255,255,255)
550 CALL LOAD(9614,255,255,2
55,241,30,153,153,153,153,15
3,153,153,153,153,153,153,22
5,31,255,241,30,153)
560 CALL LOAD(9636,153,153,1
53,153,153,153,153,153,153,1
53,225,31,238,225,30,153,153
,153,153,153,153,153)
570 CALL LOAD(9658,153,153,1
53,153,225,31,238,225,30,153
,153,153,153,153,153,153,153
,153,153,153,225,31)
580 CALL LOAD(9680,238,17,17
,17,17,17,17,17,17,17,17,17,
17,17,17,17,238,31,255,255,2
55,255)
590 CALL LOAD(9702,255,255,2
55,255,255,255,255,255,255,2
41,238,30,17,31,241,31,255,2
55,17,255,255,255)
600 CALL LOAD(9724,255,255,2
55,241,238,30,17,241,255,255
,255,255,17,255,255,255,255,
255,255,241,238,30)
610 CALL LOAD(9746,17,241,24
1,31,17,31,17,241,255,255,25
5,255,255,241,238,30,17,241,
241,31,17,255)
620 CALL LOAD(9768,17,31,255
,255,255,255,255,241,238,30,
17,241,241,31,255,31,17,241,
255,255,255,255)
630 CALL LOAD(9790,255,241,2
38,30,17,31,241,31,17,31,17,
241,255,31,31,31,255,241,238
,30,255,255)
640 CALL LOAD(9812,255,255,2
55,255,255,255,255,255,255,2
55,255,241,238,30,102,255,25
5,102,255,246,102,255)
650 CALL LOAD(9834,102,102,2
46,111,255,241,238,30,102,25
5,246,111,111,102,255,111,10
2,255,246,111,255,241)
660 CALL LOAD(9856,238,30,10
2,255,246,111,111,102,255,11
1,102,255,246,111,255,241,23
8,30,102,255,246,111)
670 CALL LOAD(9878,111,102,2
55,111,102,255,246,111,255,2
41,238,30,102,255,246,111,11
1,102,255,111,102,255)
680 CALL LOAD(9900,246,111,2
55,241,238,30,102,255,246,11
1,111,102,255,111,102,255,24
6,111,255,241,238,30)
690 CALL LOAD(9922,102,255,2
46,102,111,102,102,111,102,1
11,246,111,255,241,238,30,10
2,255,246,111,111,102)
700 CALL LOAD(9944,255,111,1
02,255,246,111,255,241,238,3
0,102,255,246,111,111,102,25
5,111,102,255,246,111)
710 CALL LOAD(9966,255,241,2
38,30,102,255,246,111,111,10
2,255,111,102,255,246,111,25
5,241,238,30,102,255)
720 CALL LOAD(9988,246,111,1
11,102,255,111,102,255,246,1
11,255,241,238,30,230,102,24
```

(See Page 18)

READER TO READER

❑ Bill Gaskill, 2310 Cypress Court, Grand Junction, CO 81506, writes:

I would like to compile a name and address database of 99/4A and 9640 users, vendors and other supporters. I currently have such a database that contains more than 1,300 entries. It is in TI-Base format for use with the Mailing List Manager application that Texaments used to sell, but I am in the process of keying the entries into Larry Tippet's MAILROOM format and may also consider another format such as Extended Software's (Jim Schwaller and Forrest Morrow) NAME-IT data base format since that program is now in the public domain. I'll be releasing the MLM and MAILROOM versions of this data base into the fairware distribution channel some time in 1995. The creation and release of a NAME-IT version is dependent upon my receiving a copy of the docs and a working copy of the NAME-IT program from some 99er out there reading this.

I would also like to find a copy of the RMJ Mailing List program that was written by Joe Nuvolini of the Front Range 99ers back in the mid-'80s. If anyone has a copy of that program I wish you would send it to me. It was always my favorite until TI-Base came along, then I kind of lost it.

If you would like your name and address included in the effort, along with a phone number and maybe your primary user group name, drop me a line with the information and I will include it in the data base.

Also, ever since buying my first Munchmobile cartridge, the programming for Munchmobile being provided by SNK, I've wondered who they are, where they are located and why Munchmobile was the only game they produced for the 99/4A. Well, I ran across an ad for SNK the other day, in a 1986 *Compute!* magazine, that had an 800 phone number and the address, "Los Angeles, CA." Although it was not the kind of information I was hoping for, it was a start. When I called the 800-253-6665 number, though, I got a recording that told me the number had been disconnected and that no other information was "attached" to the number. If anyone knows more about SNK, I would really appreciate hearing from you.

❑ Olden Warren, 4016 Weber Way, Lexington, KY 40514, (606) 223-4599, writes:

Help! I have an old TI standalone disk controller which will read from disks but will not write to them. I also have a Triton Super Extended BASIC cartridge on the same system that will not show up on the program menu when it is plugged into the cartridge port. I fear that both problems may have resulted from the same cause because they both stopped working at the same time. Please contact me if you can help me figure out what to do next.

❑ Bruce Harrison has shared his answer to Jerome A. Noel's query in the December 1994 issue:

No, you can't directly take SIZE in Extended BASIC, but you can use CALL PEEK to find out how much "STACK" is free. It's not a perfect way to do things, but it can save you from the "OUT OF MEMORY" error.

The key is to peek at location >831A, which in Extended BASIC is done by CALL PEEK(-31974,A,B) and then let A=A*256+B. This will give you a number related to the free STACK space. A little test program you can use in XB to try out this concept follows. Of course in your own program you wouldn't want to keep printing the number from -31974 on the screen, and you'd use your own variables in place of A and B.

The number to compare in line 50 has been set for this particular case. To be absolutely safe, you'd use a higher number, such as, perhaps, 2700, but here we've used a lower number than that because we know our strings are less than the maximum length.

When this runs, you'll see the number keep decreasing until it gets to somewhere around 2500, then XB will pause, and the next number will be higher. That happens because XB is using up stack space with dummy strings for the STR$ function, and during the pause it gets rid of those dummy strings.

This program will end when #309 of the array A$ is about to be assigned. Line 90 will then print the "final" amount of stack reported and the number of the array index I.

```
10 DIM A$(450)!192
20 FOR I=0 TO 449 !169
30 CALL PEEK(-31974,A,B)!123
40 A=A*256+B !110
50 IF A<2517 THEN 90 !242
60 PRINT A !221
70 A$(I)="HERE'S ANOTHER NUM
BER"&STR$(I)!182
80 NEXT I !223
90 PRINT A:I :: END !232
```

Reader to Reader is a column to put TI and Geneve users in contact with other users. Address questions to *Reader to Reader, c/o MICROpendium, P.O. Box 1343, Round Rock, TX 78680.* We encourage those who answer the questions to forward us a copy of the reply to share with readers.

LABEL—

(Continued from Page 17)

```
6,111,111,246,102,255)
730 CALL LOAD(10010,102,102,
255,102,111,241,238,30,239,2
55,255,255,255,255,255,255,2
55,255,255,255,255,241)
740 CALL LOAD(10032,238,30,2
38,238,238,238,238,238,238,2
38,238,238,255,255,255,255,241,2
38,17,17,17,17,17)
750 CALL LOAD(10054,17,17,17
,17,17,17,17,17,17,17,238,23
8,238,238,238,238,238,238,23
8,238,238,238)
760 CALL LOAD(10076,238,238,
239,255,165,90)
770 CALL LINK("GET")
780 RETURN
```

Extended BASIC

ONECHECK appeals to checkers and solitaire players

By LUCIE DORAIS
©1995 L. Dorais

ONECHECK is yet another solitaire game. Not a card game this time, but a one-player checker game, adapted from a program by David H. Ahl. It was included in his first *Basic Computer Games Collection*, 1978. His version was to be played on a teletype, with the computer printing a new version of the board after each move. Using Extended BASIC allows for graphics and a two-dimensional array, which greatly simplifies the task of both programming and playing! Sprites enhance the game.

The rules are displayed by lines 370-440. There is only one important rule to remember: Only diagonal moves are allowed, as in the real game of checkers, however nice it would be to be allowed otherwise. The sprites put on by line 410 are a red square to enhance the "FROM" square of the board and a green one to enhance the "TO" square. They are used in the instruction screen to highlight the example given, and will be used in the program itself to help you make wiser moves.

The screen checkerboard is an 8x8 grid (64 squares), each square using four screen characters. Lines 140-150 define the two types of squares used on the board, and line 160 the two 2-line patterns used to draw it at the beginning of the game (as in the picture). All characters are part of set 10.

The checkerboard is quickly displayed by line 180, which also initializes the move counter JUMP. The nested FOR-NEXT loops do two things: Put the row and column numbers at top and left of the board, and initialize the status of each square (the ST 2-DIM array) as full (=1). Since there are 16 empty squares at the center when the game starts. The next loop (line 210) resets their status to empty (=0).

THE GAME

The player enters a move, Tex validates it and, if okay, moves the checker FROM square one TO square two. The only valid move is in a diagonal, JUMPing over another checker, which will be removed in the process. The target square has to be empty. You can make only one jump per move, hence the title Ahl gave to his game. Some planning ahead is obviously required. According to the author, "It is easy to remove 30 to 39 checkers, a challenge to remove 40 to 44, and a substantial feat to remove 45 to 47," out of a total of 48. So far, I have succeeded in removing only 39 at the most.

There is only one important rule to remember: Only diagonal moves are allowed, as in the real game of checkers, however nice it would be to be allowed otherwise.

Line 230 displays the prompts and hides the sprites before you enter a new move. For each move, Tex will ask you the FROM and TO coordinates. It will keep the FROM coordinates in DR and DC (Departure Row/Column) and the TO coordinates in AR and AC (Arrival Row/Col.). The SUB ENTER accepts your input and checks for the proper format, which must be a letter (A-H) followed by a digit (1-8). Two other inputs are also allowed: When asked for the FROM square, you can also enter a "Q" to give up; when asked for the TO square, you can enter an "R" to redo the FROM input. Line 490 checks for these two letters at the proper accept row (Z=21 for Q, Z=22 for R), using row number R as a flag to return to the CALLs in lines 240-250.

Line 510 does the actual format checking, sending you back to ACCEPT if things are not okay. Otherwise, line 520 transforms the first character, a letter, into row number "R" (1-8), while column number "C" is simply the numeric value of the second input character. With that information, Tex can highlight the square you have chosen in red (FROM, sprite No. 1, passed as the fourth parameter to the SUB) or green (TO, sprite No. 2).

Back to the main program, Tex still has to check the validity of your move. Since you can jump only over one checker, and always in a diagonal, the difference between the FROM and TO row or column numbers should always be two (line 260). Otherwise, you must GOTO 290 for the appropriate "Illegal Move" message. This also removes the second sprite from the visible portion of the screen, since most often you will want to remember where you started from when you redo a wrong move.

The next thing to do is to find the array coordinates of the jumped checker. Since you must jump two rows and columns, the jumped square will always be located one row below and one column right of the FROM or TO square, depending in which direction you are moving. So Tex takes the MINimum value of the FROM and TO rows (DR,AR) and the MINimum value of the FROM and TO columns (DC,AC), and adds one to them (line 270).

The last thing to check is whether the FROM and the JUMPed squares contain a checker (STatus=1) and is the TO one empty (STatus=0). See line 280; if not, "Illegal Move...."

If the move is valid, Tex can redraw the three affected squares (lines 300-310, empty the FROM and JUMPed ones and put a checker in the TO one; SUB SQ first adjusts the array row and column to the corresponding screen ones) and reset the STatus of these squares accordingly. The JUMP total is incremented, and this completes one move.

(See Page 20)

ONECHECK—

(Continued from Page 19)

When you cannot jump anymore, you enter "Q" at the FROM prompt and Tex will GOTO line 330 to tell you how many jumps you have made and how many checkers remain on the board, before asking you if you wish to play again.

ONECHECK

```
100 REM ** ONECHECK ** L.Dor
ais/Ottawa UG/Jan. 92 (adapt
ed from David Ahl) !143
110 DIM ST(8,8):: GOSUB 370
!152
120 CALL HIDE :: CALL CLEAR
:: CALL MAGNIFY(2):: CALL CO
LOR(10,2,16)!204
130 GOTO 140 :: AC,AR,C,DC,D
R,JC,JR,JUMP,L1$,L2$,R,S$ ::
 CALL KEY :: CALL CHAR :: CA
LL SPRITE :: !@P- !088
140 CALL CHAR(104,"FF8080838
78F9F9F9F9F8F87838080FFFF010
1C1E1F1F9F9F9F9F1E1C10101FF"
)! checker in square !236
150 CALL CHAR(108,"FF"&RPT$(
"80",14)&"FFFF"&RPT$("01",14
)&"FF")! empty square !182
160 S$="      " :: L1$=S$&RP
T$("hj",8)&S$&S$&RPT$("ik",8
):: L2$=S$&"hjhjlnlnlnlnhjhj
"&S$&S$&"ikikmomomomoikik" !
105
170 ! ** init game ** !249
180 JUMP=0 :: DISPLAY AT(3,1
):L1$:L1$:L2$:L2$:L2$:L2$:L1
$:L1$ ! board !124
190 FOR R=1 TO 8 :: CALL HCH
AR(R*2+2,8,64+R):: CALL HCHA
R(2,R*2+8,48+R)!188
200 FOR C=1 TO 8 :: ST(R,C)=
1 :: NEXT C :: NEXT R ! chec
ker status !208
210 FOR R=3 TO 6 :: FOR C=3
TO 6 :: ST(R,C)=0 :: NEXT C
:: NEXT R ! empty status !24
3
220 ! ** enter/validate jump
 ** !254
230 DISPLAY AT(21,8):"FROM:"
;TAB(23);"(Q)UIT":TAB(10);"T
O:";TAB(23);"(R)EDO" :: CALL
 HIDE !033
240 CALL ENTER(21,DR,DC,1)::
 IF DR=0 THEN 330 ! from (qu
it) !084
250 CALL ENTER(22,AR,AC,2)::
 IF AR=0 THEN 230 ! to (redo
) !254
260 IF ABS(DR-AR)<>2 OR ABS(
DC-AC)<>2 THEN 290 ! skip 2
rows/col? !057
270 JR=MIN(DR,AR)+1 :: JC=MI
N(DC,AC)+1 ! find jumped che
cker coordinates !106
280 IF ST(DR,DC)=1 AND ST(AR
,AC)=0 AND ST(JR,JC)=1 THEN
300 ! all status ok? !082
290 CALL MSG(S$&" ILLEGAL MO
VE..."):: CALL LOCATE(#2,193
,1):: GOTO 240 !080
300 ST(DR,DC)=0 :: CALL SQ(D
R,DC,108):: ST(JR,JC)=0 :: C
ALL SQ(JR,JC,108)!146
310 ST(AR,AC)=1 :: CALL SQ(A
R,AC,104):: JUMP=JUMP+1 :: G
OTO 230 ! redraw squares !22
7
320 ! ** quit ** !078
330 DISPLAY AT(21,6):48-JUMP
;"PIECES REMAIN":S$&"AFTER";
JUMP;"JUMPS.": :S$&"PLAY AGA
IN? (Y/N)" !225
340 CALL KEY(0,AC,R):: IF R=
0 OR(AC<>89 AND AC<>78)THEN
340 !044
350 IF AC=89 THEN CALL CLEAR
 :: GOTO 180 ELSE END !019
360 ! ** instructions ** !19
2
370 CALL CHAR(35,"FF8199BDBD
9981FF",64,"FF818181818181FF
"):: DISPLAY AT(1,10)ERASE A
LL:"ONE CHECK" !162
380 DISPLAY AT(3,1):" 123456
78  48 checkers are  A######
##  placed as shown  B######
##  on a standard" !210
390 DISPLAY AT(6,1):"C##@@@@
##  checkerboard.":"D##@@@@@#
#":"E###@@@@##  Remove as man
y" !139
400 DISPLAY AT(9,1):"F##@@@@
##  as possible,":"G########
  by making only":"H#########
  DIAGONAL JUMPS." !174
410 CALL SPRITE(#1,64,9,25,2
5,#2,64,3,41,41)! highlight
FROM(red)/TO(green) !126
420 DISPLAY AT(13,1):"Use
e A-H and 1-8 coord-  inates
 at the FROM and TO  prompt
s (FROM: A1 TO: C3)." !164
430 DISPLAY AT(17,1):"When y
ou have no possible  jumps
remaining, enter 'Q'":"at th
e FROM prompt." !124
440 DISPLAY AT(21,1):"To red
o the input, enter 'R'at the
 TO prompt.": :"  PRESS ANY
 KEY TO START" !140
450 CALL KEY(0,AC,R):: IF R=
0 THEN 450 ELSE RETURN !071
460 !@P+ ** u-d subs ** !041
470 SUB ENTER(Z,R,C,SP)!004
480 ACCEPT AT(Z,14)SIZE(-2)V
ALIDATE("ABCDEFGH12345678RQ"
):E$ !031
490 IF Z-20=POS("QR",E$,1)TH
EN R=0 :: SUBEXIT !231
500 A$=SEG$(E$,1,1):: B$=SEG
$(E$,2,1)!218
510 IF A$<"A" OR B$>"8" OR L
EN(E$)<2 THEN CALL MSG("
  FORMAT: A3,C5..."):: GOTO
480 !048
520 R=ASC(A$)-64 :: C=VAL(B$
):: CALL LOCATE(#SP,16*R+1,1
6*C+49):: SUBEND !086
530 SUB SQ(R,C,K):: R=2*R+1
:: C=2*C+7 ! square screen l
ocation !177
540 CALL HCHAR(R,C,K):: CALL
 HCHAR(R+1,C,K+1):: CALL HCH
AR(R,C+1,K+2):: CALL HCHAR(R
+1,C+1,K+3):: SUBEND !220
550 SUB MSG(A$):: DISPLAY AT
(24,1)BEEP:A$ :: FOR D=1 TO
500 :: NEXT D :: DISPLAY AT(
24,1):"" :: SUBEND !006
560 SUB HIDE :: CALL LOCATE(
#1,193,1,#2,193,1):: SUBEND
!245
```

Support our advertisers!

TI books

A brief, annotated bibliography of books relating to the TI99/4A

(From the personal library of BARRY A. TRAVER)

(This article appeared in the disk newsletter of the 9T9 User Group of Islington, Ontario. As bookstores seldom carry TI99/4A related material any more, we suggest checking with the publisher or with TI99/4A dealers as to the availability of any book mentioned.). — Ed.)

Books marked with an asterisk are classified as "highly recommended."

ASSEMBLY LANGUAGE FOR THE TI99/4A

*Lottrup, Peter M.L. *Beginner's Guide to Assembly Language on the TI-99/4A. Compute!* Books, 1985. Although oriented toward Mini-Memory, this book is excellent for beginners, with very clear explanations and many short but useful program examples.

*McComic, Ira. *Learning TI 99/4A Home Computer Assembly Language Programming*. Prentice-Hall, 1984. A good book for beginners who have the Editor/Assembler but no previous experience in assembly language.

*Molesworth, Ralph. *Introduction to Assembly Language for the TI Home Computer*. Steve Davis Publishing, 1983. Primarily for use with the Editor/Assembler, but also can be used with Mini-Memory. Moves faster and further than the Mc-Comic book.

*Morley, M.S. *Fundamentals of TI-99/4A Assembly Language*. TAB Books, 1984. A good book for those who have the Mini-Memory Cartridge but not the Editor/Assembler.

BASIC PROGRAMS AND PROGRAMMING FOR THE TI99/4A

Ahl, David H. *The Texas Instruments Home Computer Idea Book*. Creative Computing Press, 1983. "Includes 50 Ready-to-Run Educational Programs," but most of them seem to be written in minimal BASIC and make no use of the special features of the TI99/4A.

*Carlson, Edward H. *Kids and the TI99/4A*. DATAMOST, 1982. This book is truly "not just for kids," but one of the *best* introductions to learning how to program in TI BASIC.

Casciato, Carol Ann, and Don Horsfall. *TI-99/4A: 24 BASIC Programs*. Howard W. Sams, 1983. Available with optional program cassette. Games, finances, home management, personal records, and utilities are included, all in TI BASIC.

*Compute!'s *TI Collection: Volume One*. A worthwhile collection of "over 30 TI99/4A games, applications, utilities, and tutorials — most never before published," including a word processor, a database management system, an electronic spreadsheet, several games, helpful programming tricks, and a super graphics program called "SuperFont."

Creative Programming for Young Minds...on the TI-99/4A. Creative Programming, 1982-1983. Several volumes in series. Hands-on instruction in TI BASIC (plus some small later reference to TI Extended BASIC). This series — like Carlson's book — is "not just for kids."

*Davis, Steve, ed. *Programs for the TI Home Computer*. Steve Davis Publishing, 1983. Four dozen programs that *do* make use of the special features of the TI99/4A. Most of the programs require only TI BASIC and a cassette system, though some make use of TI Extended BASIC, disk system, memory expansion, or Terminal Emulator 2 and speech synthesizer.

D'Ignazio, Fred. *TI in Wonderland*. Hayden Book Company, 1984. Has "21 programs for learning and fun," intended for youngsters, by the popular author of Katie and the Computer.

D'Ignazio, Fred. *The TI Playground*. Hayden Book Company, 1984. Contains "23 programs for learning and fun," intended for young children.

Dusthimer, Dave and Ted Buchholz. *The Tool Kit Series: TI-99/4A Edition*. Howard W. Sams, 1984. Brief 5- to 15-line subroutines — dealing with color, sound and music, graphics, animation, and computation — that can be combined to form the basis of educational programs and computer games.

Engel, . *Stimulating Simulations for the TI-99/4A*. Hayden Book Company, 1984. Eleven "simulation game programs" in TI BASIC, two in TI Extended BASIC, adapted from a popular book first published in 1977.

*Flynn, Brian. *33 Programs for the TI-99/4A. Compute!* Books, 1984. Although this book contains a few games, including

(See Page 22)

1995 TI FAIRS

FEBRUARY

Fest West '95, Feb. 18, Fabulous Inn, San Diego, California. Contact Southern California Computer Group, P.O. Box 152535, San Diego,CA 92195, or call the SCCG BBS, (619) 263-9135, User No. 25, password FEST

APRIL

Lima Multi Users Group Conference, April 29, Reed Hall, Ohio State University at Lima. Contact Lima Users Group, P.O. Box 647, Venedocia OH 45894, or call Charles Good (evenings) at (419) 667-3131 or Internet cgood@lima.ohio-state.edu.

SEPTEMBER

10th International TI-Meeting, Sept. 22-24, Wohlfahrtsgebäude der Wiener E-Werke (Welfare Building of the Vienna Electricity Board), Wachaustr. 28, A-1020 Vienna, Austria. For information write Kurt Radowisch, TI- and Geneve User Group Vienna, Fugbachgasse 18/17, A-1020 Vienna, Austria.

This TI event listing is a permanent feature of MICROpendium. User groups and others planning events for TI/Geneve users may send information for inclusion in this standing column. Send information to MICROpendium Fairs, P.O. Box 1343, Round Rock, TX 78680.

TI BOOKS —

(Continued from Page 21)

a version of "Chomp" called "Vanilla Cookie," it is primarily concerned with mathematically oriented programs, including money management and business programs, curve-fitting routines, matrix manipulations, statistics and numerical analysis, all in Extended BASIC.

*Flynn, Christopher. *Extended BASIC Home Applications on the TI-99/4A. Compute!* Books, 1984. An excellent book containing data file management utilities, bar graph programs, an electronic card file, an appointment calendar and two electronic spreadsheets. Flynn's programs always allow data to be saved on either tape or disk.

*Grillo, John P., and others. *Data and File Management for the TI-99/4A*. Wm. C. Brown Publishers, 1984. "Includes 48 programs to give the more advanced user techniques for information management." All programs are in TI Extended BASIC, and many make use of disk. Topics included: pointers, sorting, strings, linear and linked lists, sequential access files, direct access files, trees, and inverted files.

Grillo, John P., and others. *Introduction to Graphics for the TI-99/4A*. Wm. C. Brown, 1984. Includes 38 programs in TI Extended BASIC, some making use of disk, BUT note this comment by the authors: "In this book, we have limited our discussion to low-resolution graphics only. We do not discuss the color, sound, joystick, and lightpen features of this fine machine. We hope to cover these topics in a subsequent book."

Herold, Raymond J. *TI-99/4A Sound and Graphics*. A fairly good guide to sound, graphics and speech synthesis on the TI99/4A (including coverage of TI's text-to-speech diskette). Of the games, "Alphabet Invasion" and "Slot Machine" are done quite well.

Holtz, Frederick. *Using & Programming the TI-99/4A Including Ready-to-Run Programs*. TAB Books, 1983. Although this book is widely distributed, many chapters are either too elementary or too advanced to benefit the aver TI99/4A owner.

Inman, Don, and others. *Introduction to TI BASIC*. Hayden Book Company, 1980. A straightforward textbook on TI BASIC that does not go very far beyond the two manuals supplied with the TI99/4A.

Knight, Timothy Orr. *TI-99/4A Graphics and Sounds*. Howard W. Sams, 1984. Available with optional program cassette. Contains 37 sample (and simple) TI BASIC programs, originally written for the Commodore 64, most of which are rather trivial in nature.

Knight, Timothy Orr, and Darren LaBatt. *TI-99/4A BASIC Programs*. Howard W. Sams, 1984. Available with optional program cassette. Although these 30 TI BASIC programs were also originally written for the Commodore 64, they are more substantial than those contained in the other book by Knight.

Kreutner, Donald C. *TI-99/4A Favorite Programs Explained*. Que Corporation,

(See Page 23)

CECURE™

UPGRADES

Includes parts and labor

EXTRA 64K VIDEO-ON SOCKETED MEMORY	$25
EXTRA 64K VIDEO-NON SOCKETED MEMORY	$35
RESET SWITCH	$10
CLOCK AND BATTERY HOLDER	$7.50
HARD DRIVE BYPASS for 4A	$15

Eliminates HARD DRIVE time out, when HARD DRIVE not connected

HFDC 250 KB upgrade to 500KB $20

NOTE: This is used for 1.44 MEG FLOPPY and tape back-up QIC 40/80

PFM

PFM * Programmable Flash Memory * PFM

UPDATE YOUR OWN BOOT BIOS WITH PFM

PFM BOOT BIOS VER.#2.0 SYSTEM/SYS $75

PFM+ WITH 128K FLASH DISK VER. 3.0 $125

MEMORY

SALE • SALE • SALE•SALE• SALE

128K X 8 LP 80ns MEMORY HITACHI $17.95

Look again...$17.95

Quantities limited to 8 per customer.

SORRY, NO DEALER SALES.

S&T SOFTWARE

*** CYA * 9640 * CYA**

CYA $15

Configure Your "SYSTEM/SYS" Attributes without AUTOEXEC!

* ASSIGN and REMAP your Drives. * Customize your PROMPT.
* Change your Default Screen COLORS and Default Drive.
* Customize your PATH. * TEST your Joysticks and Mouse

TEAC 720K/1.44M 3.5" MODEL FD 235	$44.95
5.25 MOUNTING KIT	$4.95

VOICE 414 679-4343 BBS 414 442-9669

MASTERCARD or VISA ORDERS USE OUR TOLL FREE ORDER ONLY PHONE NUMBER:

1-800-959-9640 ORDERS ONLY!

CECURE ELECTRONICS INC.

Francis J. Bubenik of LITI99ers dies

The Long Island TI99ers sent MICROpendium the following obituary notice for Francis J. Bubenik Jr., born in 1937.

With deep regret we inform the TI99/4A community of the death of Francis J. ("Frank") Bubenik on Nov. 23, 1994, following a brief illness.

Frank had been a most important member of the Long Island TI99er Users Group since its foundation, and played such an important role through the years that the group could observe its 11th anniversary as a users group in April of 1994.

From the beginning, his was always an energetic input. Frank served first as a newsletter contributor, was elected president of the group in 1987 and subsequently added the functions of secretary and editor during '87-'89. He then concentrated on the editorship of the *Long Island Sound* right up to the latest monthly issue.

His newsletter ultimately became the sole source of local TI reporting on the island, and a fund of news, fair notices and much educational material for the readership. He worked constantly to improve the format and appeal of his publication and was attracting other off-island readers. Frank attended many sectional fairs and kept his group informed on what was happening in the TI world. His contributions and his energy were the dominating influence in keeping a TI group going, despite the gradual loss of those members who moved on to more sophisticated computer systems. To the very end Frank remained an ardent devotee of the TI system. He became a one-man force in keeping a group together. He will be missed by all who caught some of his enthusiasm and dedication.

The QB 99ers have also sent in a note that Bubenik, though member of their group for less than a year, was "actively involved" and "always willing to help out a member." He was also a coordinator of the TICOFF computer fair in Roselle, New Jersey.

TI BOOKS—

(Continued from Page 22)

83. Has 40 practical and entertaining programs in TI BASIC, with explanations.

*Loreto, Remo A., ed. *The TI-99/4A in Bits and Bytes*. Remo A. Loreto, 1983. A hodgepodge collection, but one containing within it a number of worthwhile programs (some in Extended BASIC) and programming hints.

Peckham, Herbert D. *Programming BASIC with the TI Home Computer*. McGraw-Hill Book Company, 1979. Another straightforward textbook on TI BASIC, going a bit further than Inman's book.

Regena, C. *BASIC Programs for Small Computers. Compute!* Publications, 1984. Although this book contains "things to do in 4K or less" for other computers (notably the Vic-20 and TRS-80), it also contains programs in TI BASIC for the TI99/4A.

Regena, C. *Programmer's Reference Guide to the TI-99/4A. Compute!* Publications, 1983. Not so much a reference guide as an instruction manual on how to program in TI BASIC, this book contains 48 programs by popular columnist Cheryl Whitelaw (or "Regena" of *99'er, Compute!* and MICROpendium fame).

Rugg, Tom, and others. *32 BASIC Programs for the TI-99/4A*. dilithium Press, 1984. Programs include applications, education, games, graphics display, and mathematics; 30 programs in TI BASIC, 2 in TI Extended BASIC. (The programs can be ordered on disk or cassette.)

Sanders, William B. *The Elementary TI-99/4A*. DATAMOST, 1983. Contains useful chapters on "Data and Text Files" and "You and Your Printer," topics usually ignored in similar books.

Schechter, Gil M. *TI-99/4A: 51 Fun and Educational Programs*. Howard W. Sams, 1983. Available with optional program cassette. All programs are in TI BASIC, and all are probably 4K or less in size.

Schreiber, Linda M. and Allen R. *The Last Word on the TI-99/4A*. TAB Books, 1984. Has "55 practical and entertaining programs, all written in TI Extended BASIC," perhaps the best of which are "Battleship" and "Towers Game." (Programs are available on tape.)

*Sternberg, Charles D. *TI BASIC Computer Programs for the Home*. Hayden Book Company, 1984. Programs include automobile, conversion, home finances, kitchen helpmates, list, tutorial and others, and each program is documented with description, symbol table, and output sample. The book is an adaptation for the TI99/4A of Sternberg's *BASIC Computer Programs for the Home*; now if only someone will do an adaptation of his excellent two volumes on *BASIC Computer Programs for Business*!

Turner, Len. *101 Programming Tips & Tricks for the Texas Instruments TI-99/4A Home Computer*. ARCsoft Publications, 1983. An unimpressive book carried in many bookstores.

Turner, Len. *36 Texas Instruments TI-99/4A Programs for Home, School & Office*. ARCsoft, 1983. Many other books on this list contain a much better selection of programs in TI BASIC.

*Winter, Mary Jean. *Computer Playground on the TI 99/4A*. A colorful collection of TI BASIC computer activities intended for children in grades 2 through 6. Adapted for the TI-99/4A by Marcia Carrozzo.

*Wyatt, Allen. *BASIC Tricks for the TI-99/4A*. Howard W. Sams, 1984. Available with optional program cassette. A good collection of 28 useful subroutines dealing with selective input, rounding, dollars and cents, report formatting, time and dates, upper and lower cases, sorting and menus.

*Zaks, Rodnay. *Your First TI 99/4A Program*. Like anything done by Zaks, this book is clearly written and well done. It is, however, as the title indicates, a book for those who are just beginning to learn "the basics of BASIC."

GAMES IN TI BASIC OR TI EXTENDED BASIC

Holtz, Frederick. *TI-99/4A Game Programs*. TAB Books, 1983. Has 32 "games, puzzles, and brain teasers" in TI BASIC, with explanations.

*Ingalls, Robert P. *TI Games for Kids. Compute!* Publications, 1984. An excellent collection of 32 educational game programs in TI BASIC for children ages 2 to 17.

McEvoy, Seth. *Creating Arcade Games on the TI-99/4A. Compute!* Publications, 1984. With the exception of one chapter devoted to TI Extended BASIC, this book tells "how to" write arcade games in TI BASIC, and includes eight finished games.

*Mullish, Henry, and Don Kruger. *Zappers: Having Fun Programming and Playing 23 Games for the TI-99/4A*. Simon & Schuster, 1984. Many favorites in TI BASIC, including "Blackjack," "Hangman," "Hidden Word Search," "Othello" ("Flip-a-Disk"), "Simon," and "Tic Tac Toe."

*Regena, C. *TI Games. Compute!* Publications, 1983. About 30 games for the TI99/4A, mostly in TI BASIC, but including 7 in TI Extended BASIC, including the excellent "Mystery Spell" and "Mosaic Puzzle."

Renko, Hal, and Sam Edwards. *Terrific Games for the TI 99/4A*. Addison-Wesley Publishing Company, 1983. A mixed bag of 30-some unusual game programs from The Netherlands in TI BASIC and TI Extended BASIC.

*Singer, Scott L., and Tony E. Bartels. *Games TIs Play*. DATAMOST, 1983. Has 32 TI BASIC game programs; based on the book *Games Apples Play* by Mark James Capella and Michael D. Weinstock. (Programs are available on disk.)

*Ton, Khoa, and Quyen Ton. *Entertainment Games in TI BASIC and Extended BASIC*. Howard W. Sams, 1983. Available with optional program cassette. One of the *best* program collections available; "Frogger"-lookalike "HomeBound" is excellent. Book also contains a few non-game programs, e.g., "Address Inventory" and "Auto Sprite Editor."

LOGO PROGRAMS AND PROGRAMMING FOR THE TI99/4A

*Abelson, Harold. *TI LOGO*. McGraw-Hill Book Company, 1984. If you have TI LOGO II, you already have this excellent book, but if you have TI LOGO (I), get it!

Bearden, Donna. *1, 2, 3, My Computer & Me*. Prentice-Hall, 1983. Though not just for the TI, this "LOGO funbook for kids" contains an appendix on "editing features for Apple LOGO, MIT LOGO, and TI LOGO."

*Conlan, Jim, and Don Inman. *Sprites, A Turtle, and TI LOGO*. Prentice-Hall, 1984. "A friendly, playful introduction to the TI LOGO computer language," very well done.

Dry Gulch

A shoot-em-up that doesn't shoot straight

The following program, Dry Gulch, was written by Jim Peterson. Peterson died last winter.

Dry Gulch is an Extended BASIC game in which you try to shoot an opponent with a richochetting bullet. Instructions are included with the program.

DRY GULCH

```
1 DATA 23,23,20,20,4,16,8,16
,8,16,4,20,8,23,8,23,4,21,8,
21,8,21,4,25,8,28,8,28 !099
2 DIM NN(36)!198
10 GOTO 100 !179
11 X1,X2,X3,X4,Q$,JS,SKIP$,S
KIP,@$,F,I,NN(),SET,MEL,M1,M
2,K,ST,A,RR,CC,G1,G2,G3,G4,G
G,AR,AC,BR,BC,T,X,XR,XC,G,V,
U,H,S5,S1,S2,N,CZ,ML,NI,B,Z,
C$,C,J !060
30 CALL CLEAR :: CALL CHAR :
: CALL SCREEN :: CALL GCHAR
:: CALL COLOR :: CALL VCHAR
:: CALL SOUND :: CALL KEY ::
 CALL HCHAR :: CALL MAGNIFY
:: CALL SPRITE :: CALL JOYST
 !127
40 !@P- !064
100 CALL CLEAR :: CALL CHAR(
94,"3C4299A1A199423C"):: CAL
L SCREEN(11):: DISPLAY AT(2,
11):"DRY GULCH": : :"TCX-105
3 ^ Tigercub Software" !085
110 DISPLAY AT(8,2):"EVENA
BUSHWHACKIN'": :" SHEEP-HERD
IN'HOSSTHIEF": :" WOULDN'T
 BE SO LOW-DOWN AS": :" TO R
USTLE A THREE DOLLAR": :" PR
OGRAM!" !208
120 CALL GCHAR(8,8,X1):: CAL
L GCHAR(8,10,X2):: CALL GCHA
R(10,17,X3):: CALL GCHAR(10,
22,X4)!028
130 DISPLAY AT(20,1):"Will y
ou use joysticks? Y" :: ACCE
PT AT(20,25)VALIDATE("YNyn")
SIZE(-1)BEEP:Q$ :: IF Q$="Y"
 THEN 150 !150
140 IF Q$="y" THEN 160 ELSE
170 !005
150 DISPLAY AT(22,1):"UNLOCK
 ALPHA LOCK!" !036
160 JS=1 !085
170 DISPLAY AT(20,1):"Want t
o skip instructions? Y" :: A
CCEPT AT(20,28)VALIDATE("YNy
n")SIZE(-1):SKIP$ :: IF (SKI
P$<>"Y")AND(SKIP$<>"y")THEN
190 !087
180 SKIP=1 !239
190 !DRY GULCH programmed by
 Jim Peterson 11/82, rev. 9/
83, 10/84,XB version 6/85 !2
49
200 REM COPYRIGHT 1982 Tiger
cub Software TC-53  156 Coll
ingwood Ave., Columbus Ohio
43213 !220
210 REM REPRODUCTION FOR RES
ALE PROHIBITED. DELETION OF
COPYRIGHT NOTICE PROHIBITED.
     !047
220 @$=CHR$(18)&CHR$(16)&CHR
$(13)&CHR$(15)&CHR$(2)&CHR$(
12)&CHR$(11)&CHR$(4)&CHR$(14
)!127
230 F=110 :: FOR I=0 TO 36 :
: NN(I)=INT(F*1.059463094^I+
.5):: NEXT I !145
240 CALL CLEAR :: FOR SET=1
TO 12 :: CALL COLOR(SET,2,11
):: NEXT SET !182
250 CALL VCHAR(1,31,1,96)::
CALL SCREEN(X1-139):: IF SKI
P=1 THEN 380 !119
260 DISPLAY AT(2,9):"DRY GUL
CH": :"  Marshal Wyatt Urp a
nd bad-":" man Mild Bill Hic
cup are" !112
270 DISPLAY AT(6,1):" having
 a shoot-out in Dry":" Gulch
. Their bullets rico-":" che
t off the buildings and" !20
3
280 DISPLAY AT(9,1):" they a
re apt to shoot":" themselve
s in the back.": :"  Marshal
 Urp is the good":" guy in t
he white hat, of" :: IF JS<>
1 THEN 310 !014
290 DISPLAY AT(14,1):" cours
e. His player can move":" hi
m up and down, and shoot":"
up, forward and down, with"
!109
300 DISPLAY AT(17,1):" the #
1 joystick. ": :"  Mild Bill
's player can":" move and sh
oot with the #2":" joystick.
" :: GOTO 330 !138
310 DISPLAY AT(14,1):" cours
e. His player can move":" hi
m left and right with the":"
 > and / keys and shoot with
" !235
320 DISPLAY AT(17,1):" the P
, L and < keys.": :"  Mild B
ill's player can":" move him
 with the Q and Z" :: DISPL
Y AT(21,1):" keys, shoot wit
h the W, S":" and X keys." !
024
330 DISPLAY AT(24,1):"  Pull
 any trigger to start" !029
340 !!131
350 RESTORE 1 :: FOR MEL=1 T
O 13 :: READ M1,M2 !170
360 CALL SOUND(400,NN(M1),5,
NN(M2),5):: NEXT MEL !140
370 CALL KEY(0,K,ST):: IF ST
<1 THEN 370 !097
380 CALL CHAR(128,"FF8080808
0808080",129,"FF",130,"FF010
10101010101",131,RPT$("80",8
),132,"0",133,RPT$("01",8))!
229
390 CALL CHAR(134,"808080808
08080FF",135,"000000000000000
FF",136,"0101010101010FF",1
37,"FF818181818181FF",138,RP
T$("F",16))!037
400 CALL CHAR(60,"C381FF5A7E
3C3C3C",61,"00000024",62,"10
145C5070101010")!245
410 CALL VCHAR(1,1,32,720)::
 RANDOMIZE :: CALL COLOR(11,
```

(See Page 25)

DRY GULCH—

(Continued from Page 24)

```
2,11,12,16,11,13,2,16,14,2,1
6):: CALL HCHAR(1,1,138,64)!
132
420 CALL HCHAR(24,1,138,32):
: CALL VCHAR(1,30,138,120)::
 FOR A=1 TO 5 !158
430 RR=(INT(9*RND)+2)*2 :: C
C=(INT(9*RND)+4)*(SQR(X4-7)-
10)!149
440 CALL GCHAR(RR,CC,G1):: C
ALL GCHAR(RR,CC+2,G2):: CALL
 GCHAR(RR+2,CC,G3):: CALL GC
HAR(RR+2,CC+2,G4):: IF (G1=3
2)+(G2=32)+(G3=32)+(G4=32)<>
-4 THEN 430 !137
450 DISPLAY AT(RR,CC-2):CHR$
(128)&CHR$(129)&CHR$(130);::
 DISPLAY AT(RR+1,CC-2):CHR$(
131)&CHR$(132)&CHR$(133);!00
2
460 DISPLAY AT(RR+2,CC-2):CH
R$(134)&CHR$(135)&CHR$(136);
!035
470 CC=(INT(9*RND)+4)*2 :: C
 L GCHAR(CC,CC,GG):: IF GG<
>32 THEN 470 !085
480 DISPLAY AT(CC-2,CC-2):CH
R$(137);:: NEXT A !163
490 AR=10 :: AC=4 :: BR=14 :
: BC=28 :: CALL CHAR(37,"000
00000343C3CFF"):: CALL CHAR(
127,"0"):: CALL CHAR(119,"0"
):: CALL HCHAR(AR,AC,119)::
CALL HCHAR(BR,BC,127)!030
500 CALL MAGNIFY(2):: CALL S
PRITE(#1,37,2,AR*8-15,AC*8-1
5):: CALL SPRITE(#2,37,16,BR
*8-15,BC*8-8)!053
510 CALL SPRITE(#3,60,16,170
,20,#4,61,2,170,20,#5,62,4,1
2,210)!204
520 T=0 !011
530 X=X+1+(X=2)*2 :: IF JS=1
 THEN 1310 !029
540 CALL KEY(X,K,ST):: IF ST
=0 THEN 530 :: IF K<1 THEN 8
90 !095
550 ON POS(@$,CHR$(K),1)+1 G
OTO 530,560,610,630,580,660,
0,980,720,1070 !038
560 CALL HCHAR(AR,AC,32):: A
R=AR-2 :: IF AR>2 THEN 600 !
051
570 AR=AR+2 :: GOTO 600 !203
580 CALL HCHAR(AR,AC,32):: A
R=AR+2 :: IF AR<24 THEN 600
!102
590 AR=AR+X3-X1 !123
600 CALL HCHAR(AR,AC,119)::
CALL SPRITE(#1,37,2,AR*8-15,
AC*8-15):: GOTO 520 !161610
CALL HCHAR(BR,BC,32):: BR=BR
-2 :: IF BR>2 THEN 650 !106
620 BR=BR+2 :: GOTO 650 !255
630 CALL HCHAR(BR,BC,32):: B
R=BR+2 :: IF BR<24 THEN 650
!157
640 BR=BR-2 !163
650 CALL HCHAR(BR,BC,127)::
CALL SPRITE(#2,37,16,BR*8-15
,BC*8-8):: GOTO 520 !172660
XR=AR :: XC=AC :: GOSUB 1300
 !069
670 CALL GCHAR(XR,XC+2,G)::
IF G<>32 THEN 880 !209
680 XC=XC+2 :: CALL HCHAR(XR
,XC,46):: CALL HCHAR(XR,XC,3
2):: GOTO 670 !088
690 XR=BR :: XC=BC :: GOSUB
1300 !071
700 CALL GCHAR(XR,XC-2,G)::
IF G<>32 THEN 880 !210
710 XC=XC-2 :: CALL HCHAR(XR
,XC,46):: CALL HCHAR(XR,XC,3
2):: GOTO 700 !119
720 XR=AR :: XC=AC :: GOSUB
1300 !069
730 CALL GCHAR(XR-2,XC+2,G):
: IF G<>32 THEN 760 !021
740 XR=XR-2 :: XC=XC+2 :: GO
SUB 750 :: GOTO 730 !112
750 CALL HCHAR(XR,XC,46):: C
ALL HCHAR(XR,XC,32):: RETURN
 !191
760 IF G>128 THEN 790 !131
770 IF G=127 THEN 1160 !243
780 IF G=119 THEN 1210 !038
790 GOSUB 800 :: GOTO 820 !1
23
800 T=T+1 :: IF T=6 THEN 520
 !187
810 RETURN !136
820 CALL GCHAR(XR,XC+2,G)::
IF G<>32 THEN 840 !169
830 XC=XC+2 :: GOSUB 750 ::
GOTO 900 !202
840 IF G<128 THEN 770 !110
850 GOSUB 800 :: CALL GCHAR(
XR-2,XC,G):: IF G<>32 THEN 8
80 !199
860 XR=XR-2 :: GOSUB 750 !14
5
870 GOTO 990 !048
880 IF (G=119)+(G=127)THEN 7
70 ELSE 520 !198
890 XR=AR :: XC=AC :: GOSUB
1300 !069
900 CALL GCHAR(XR+2,XC+2,G):
: IF G<>32 THEN 920 !181
910 XR=XR+2 :: XC=XC+2 :: GO
SUB 750 :: GOTO 900 !026
920 IF G<128 THEN 770 !110
930 GOSUB 800 :: CALL GCHAR(
XR,XC+2,G):: IF G<>32 THEN 9
50 !012
940 XC=XC+2 :: GOSUB 750 ::
GOTO 730 !031
950 IF G<128 THEN 770 !110
960 GOSUB 800 :: CALL GCHAR(
XR+2,XC,G):: IF G<>32 THEN 8
80 !198
970 XR=XR+2 :: GOSUB 750 ::
GOTO 1080 !157
980 XR=BR :: XC=BC :: GOSUB
1300 !071
990 CALL GCHAR(XR-2,XC-2,G):
: IF G<>32 THEN 1010 !017
1000 XR=XR-2 :: XC=XC-2 :: G
OSUB 750 :: GOTO 990 !118
1010 IF G<128 THEN 770 !110
1020 GOSUB 800 :: CALL GCHAR
(XR,XC-2,G):: IF G<>32 THEN
1040 !104
1030 XC=XC-2 :: GOSUB 750 ::
 GOTO 1080 !128
1040 IF G<128 THEN 770 !110
1050 GOSUB 800 :: CALL GCHAR
(XR-2,XC,G):: IF G<>32 THEN
880 !199
1060 XR=XR-2 :: GOSUB 750 ::
 GOTO 730 !062
1070 XR=BR :: XC=BC :: GOSUB
 1300 !071
1080 CALL GCHAR(XR+2,XC-2,G)
:: IF G<>32 THEN 1100 !107
1090 XR=XR+2 :: XC=XC-2 :: G
OSUB 750 :: GOTO 1080 !208
1100 IF G<128 THEN 770 !110
1110 GOSUB 800 :: CALL GCHAR
(XR,XC-2,G):: IF G<>32 THEN
1130 !194
1120 XC=XC-2 :: GOSUB 750 ::
 GOTO 990 !037
```

(See Page 26)

DRY GULCH—

(Continued from Page 25)

```
1130 IF G<128 THEN 770 !110
1140 GOSUB 800 :: CALL GCHAR
(XR+2,XC,G):: IF G<>32 THEN
880 !198
1150 XR=XR+2 :: GOSUB 750 ::
 GOTO 900 !232
1160 CALL VCHAR(1,3,32,672)!
243
1170 DISPLAY AT(12,1):" MARS
HAL URP HAS BEEN SHOT!" :: F
OR T=880 TO 110 STEP -40 ::
CALL SOUND(-99,T,0):: NEXT T
 :: FOR V=0 TO 30 STEP 2 ::
CALL SOUND(-99,110,V):: NEXT
 V :: GOSUB 1280 :: U=U+1 !0
39
1180 IF U+H=5 THEN 1240 !243
1190 DISPLAY AT(24,1):" PULL
 ANY TRIGGER" !002
1200 CALL KEY(5,K,S5):: CALL
 KEY(1,K,S1):: CALL KEY(2,K,
S2):: IF S5+S1+S2=0 THEN 120
0 ELSE 410 !183
1210 CALL VCHAR(1,3,32,672)!
243
1220 DISPLAY AT(12,1):" MILD
 BILL HAS BEEN SHOT!" :: FOR
 N=220 TO 110 STEP -10 :: CA
LL SOUND(-99,20000,30,20000,
30,N*3.75,30,-4,0):: NEXT N
:: GOSUB 1340 :: H=H+1 :: IF
 U+H=5 THEN 1240 !042
1230 GOTO 1190 !249
1240 DISPLAY AT(15,1):U;"BUL
LET "&SEG$("HOLES",1,4-(U<>1
))&" IN THE": :" MARSHAL": :
 :H;"BULLET "&SEG$("HOLES",1
,4-(H<>1))&" IN THE": :" BAD
MAN" !040
1250 DISPLAY AT(24,1):"ANY T
RIGGER FOR NEW GAME" !215
1260 CALL KEY(5,K,S5):: CALL
 KEY(1,K,S1):: CALL KEY(2,K,
S2):: IF S1+S2+S5=0 THEN 125
0 !004
1270 U=0 :: H=0 :: CZ=0 :: G
OTO 410 !207
1280 DATA 4,6,8,4,9,8,6,4 !1
13
1290 RESTORE 1280 :: FOR ML=
1 TO 8 :: READ NI :: CALL SO
UND(250,NN(NI),0):: NEXT M
:: RETURN !033
1300 CALL SOUND(50,-5,0):: C
ALL SOUND(100,-6,0):: RETURN
 !022
1310 CALL JOYST(X,A,B)!177
1320 IF (A=0)*(B=0)THEN 530
!159
1330 Z=(((A+3*B)/4)+5)+ABS(X
>1)*9 :: C$=STR$(C)&" " :: O
N Z GOTO 530,580,890,530,530
,660,530,560,720,1070,630,53
0,690,530,530,980,610,530 !0
28
1340 DATA 22,22,24,22,19,19,
15,19,19 !229
1350 RESTORE 1340 :: FOR ML=
1 TO 9 :: READ NI :: FOR J=1
 TO 2 :: CALL SOUND(-99,NN(N
I),0):: CALL SOUND(-99,NN(NI
)*1.02,2):: NEXT J :: NEXT M
L :: RETURN !073
```

MICRO-REVIEWS

Reminders and Forget Me Knots, Windows for the TI, Attack of the Creepers, Picture Show

By CHARLES GOOD

REMINDERS and FORGET ME KNOTS
By Alfred Malcolm

I guess MICROpendium's publisher, John Koloen, and I both recognize new and interesting 99/4A software when we see it. We both received these programs independently at about the same time. While John was preparing to publish the BASIC listings and the author's doc file in the November 1994 issue of his magazine I was busy writing the following review. What follows, therefore, is a MICRO-Review of software already published in MICROpendium.

Imagine this. Every time you turn on your 99/4A computer and select Extended BASIC you are immediately presented with a full screen of monthly reminders showing all the important birthdays, holidays, appointments, TI user group meeting dates, etc. for the current month. Press the "any key" and you get a screen showing all the important events for the current day of the month. Press the "any key" again and you get your usual startup menu (BOOT or Funnelweb) of commonly used TI software immediately available on your RAMdisk or boot disk. If you are in a big hurry you can press FCTN/9 at the first (monthly reminders) screen to bypass the daily reminder and go directly to your application menu. In either case, every time you boot up your computer, the 99/4A will never let you forget!

"Reminders" and "Forget Me Knots" are the two Extended BASIC programs that allow this to happen. You enter your monthly and daily notes into the computer using Forget Me Knots. These daily and monthly notes are saved as separa[illegible] DV/80 files. If you change the name of the Reminders program to LOAD, Reminders will read your notes each time you turn on

(See Page 27)

MICRO-REVIEWS —

(Continued from Page 26)

your computer. To work this magic exactly as I described in the preceding paragraph all you need is this software and a computer clock such as PGRAM clock, Myarc HFDC clock, MBP card, or Triple Tech card. It also helps to have a RAMdisk or hard drive so that REMINDERS will always automatically be there right after XB is selected.

You can use Forget Me Knots as a standalone application even if you don't have a computer clock or RAMdisk. From within Forget Me Knots you can read screens of previously saved monthly and daily reminders by manually entering the month, day, and year. Forget Me Knots allows you to spice up screen displays of your text notes with nice borders other graphic patterns. The results look very professional.

The only similar 99/4A software I am familiar with is Remind Me. If you have Remind Me daily reminder files, these can be converted to Forget Me Knot format from within Forget Me Knot. For a standalone appointment calendar application I prefer Remind Me over Forget Me Knots. Remind Me provides an on-screen calendar display of the current (or any) month, something that Forget Me Knots doesn't do. However, Remind Me cannot be programmed to run another application. When you exit Remind Me you are back at the TI color bar screen. The unique feature of Reminders and Forget Me Knots is the automatic display of time-specific reminders at the beginning of each session with your 99/4A.

Reminders and Forget Me Knots are public domain. To obtain your copy you can either type in the BASIC listings from the November 1994 issue of MICROpendium, or send the author a disk and paid return mailer and a donation for his user group, or send me $1 and I will then send it to you on an SSSD disk. The author requests that if you find the program useful you send a donation to his user group, the British Columbia UG, whose address is given in the software documentation.

WINDOWS FOR THE TI
By John Bull

No, this isn't the Microsoft graphical interface found on mamy IBM compatible computers. It is assembly software that allows the XB programmer to place multiple text windows on screen without destroying the appearance of the rest of the screen. When you remove the window the original screen, which was stored in memory, is restored.

You can have up to six of these text windows and their underlying screen stored in memory. The text windows may all be on screen at once overlapping each other, or you can display the windows a few at a time. Call links let you determine the size and position of each window. If you have ever seen the Funnelweb configure utility you know what I am talking about in terms of overlapping text windows appearing and disappearing from the screen.

The idea of text windows in Extended BASIC is not new. This can be done in 80 columns (if you have an 80-column card or a Geneve) using Alexander Hulpke's X80. Windows can be done on a 40-column 99/4A system using The Missing Link. The nice thing about John Bull's software is that, unlike TML, it is free. Windows For the TI is public domain and comes on a SSSD disk. To get the software send $1 to either John Bull or me and we will mail it to you. Your money pays for the SSSD disk and postage.

ATTACK OF THE CREEPERS
By Ian J. Howle

In his letter to me Ian writes: "I have just finished writing this game program in TMS9900 assembly language. It has taken a little over a year to develop. I would like to put this in the public domain so that everybody can enjoy it....Could you put this in your library and distribute it to anyone who is interested."

This is a "make your man climb ladders from the bottom up through several screens to the top of the maze" game. The game comes as EA5 files with an optional Extended BASIC loader. I don't usually get excited about arcade games with fast action but little strategy. I must admit, however, that this game has held my attention for several days now. This is mainly due to the very complicated joystick action required to finish the game. The scenario is this: The creepers have attacked your mineral mine and completely surpassed your defenses. You have to escape from the bottom of the mine to the surface where a spaceship awaits. Along the way you have to make sure that you don't come in contact with creepers (they look like spiders) as they move through the mine and that you don't fall off one of the many narrow ledges in the mine.

You need a really good responsive joystick to succeed in this game. The official TI "wired remote controllers" really aren't good enough. I have had best luck with a Prostick. This game has the most complicated sequence of joystick movements I have ever encountered in a TI game. Of course if you move the stick up/down your man moves up/down the ladders and if you move the stick left/right your man walks along the ledges in those directions, and if you press the fire button he shoots his gun in the direction he faces. But there are more possible types of man moves, most of which require a particular sequence of stick and fire-button actions. Your man can be made to move left/right at double speed to run away from a creeper and then turn around and fire. He can slide under obstacles and come to a rapid stop. He can also jump left or right either short distances or long distances. Jumps are used to get up and down from one ledge to another in the mine.

Part of the fun is figuring out the complete repertoire of movements your man can make and the exact sequence of joystick movements need for each type of movement. Without giving away any significant secrets I can tell you that two of these man movements require a specific sequence of three joystick actions. First you do this with the joystick, then do that with the joystick, then do the third thing with the joystick at which point your man will make his move. If you mess up on any one of the three joystick actions either the wrong thing happens or nothing happens.

Once you have figured out how to get your man to do this and that it is still a challenge to predetermine which action to take in a given situation. There are two levels of difficulty, and the easier of the two usually

(See Page 28)

MICROREVIEWS—

(Continued from Page 27)

gives you plenty of time to think about what to do next. Should you jump short or long to get across a gap to the next ledge? Should you go down in order to later go up? A fair amount of strategy is required. The harder level of difficulty gives you less time to think because you have many more creepers to contend with. I haven't managed to get very far under these harder conditions. Nevertheless, I find this a highly entertaining arcade game.

Send me $1 and I'll send you the game on a SSSD disk. Include an extra $1 and I will send Ian's source code on a second DSSD disk.

PICTURE SHOW
By Bruce Harrison

Here is another of Bruce's simple and useful public domain utilities. It is designed for people who do not own TI-Artist. My user group software library has many disks full of TI-Artist pictures. How do you view these pictures? Well, you can load each picture into MaxRLE, but you have to know and type in the picture file name. Or you can use one of several "slide show" programs to display all the pictures on the disk, but such programs don't tell you the file name of the pictures you are viewing and you only see each picture for a few seconds. Or, you can use Picture Show, the most full-featured TIA picture display utility anywhere.

Picture Show is an EA5 assembly language program with an XB loader. Once loaded, it asks you for a drive number (1-9 or A-Z) and then displays on screen the file names of all TI Artist pictures (names ending in "_P") on that disk. Other types of files on the same disk are ignored. Use the arrow keys to move the cursor next to a picture name, press Enter to mark the name. You can mark any or all pictures manually by pressing a key to go on to the next picture, or automatically using a time interval you specify. Selected time intervals can be between less than a second up to 5 minutes per picture. Pictures can be included in a sequence up to three times, and you can have a picture sequence automatically repeated as many times as you specify. You can at any time get back to your list of pictures or specify a new drive to bring up a new list of pictures. Talk about versatility and multiple options!

There are North American and European versions of Picture Show, each on a separate DSSD disk filled with a nice selection of sample full color TIA pictures. Accuracy of the automatic timing feature is based on whether you have 60-cycle or 50-cycle electricity, hence the need for two versions. I'll send you either version if you send me $1 ($2 for both). Each version comes with Bruce's user-friendly instructions.

ACCESS:

Alfred Malcolm (Reminders and Forget Me Knots): 216 10th Ave., New Westminster, British Columbia, Canada V3L-2B2.

John Bull (Windows for the TI): 409 Blue Valley Lane, Knoxville, TN 37922. Phone (615) 694-4750.

Ian J. Howle (Attack of the Creepers), 3707 SW Southern St., Seattle, WA 98126. Phone (206) 938-4065.

Bruce Harrison (Picture Show): 5705 40th Place, Hyattsville, MD, 20781. Phone (301) 277-3467

Charles Good (your humble reviewer and sender of $1 software): P.O. Box 647, Venedocia OH 45894. Phone (419) 667-3131. Internet cgood@lima.ohio-state.edu

NEWSBYTES

HCS has new address

The Historical Computing Society has a new address: 2962 Park St. No. 1, Jacksonville, FL 32205.

The group publishes a newsletter regarding older computers, including the TI99/4A.

Bluegrass 99ers change address

The new mailing address for the Bluegrass 99 Computer Inc. is c/o Olden Warren, 4016 Weber Way, Lexington, KY 40514.

Cactus Patch BBS area code changes

Tom Wills, SysOp of the Southwest Ninety-Niners Cactus Patch BBS, reports that the area code to access the BBS is changing. Effective March 19, 1995, all of Arizona outside of Maricopa County will be in the 520 area code. The current phone number is 602-290-6277. As of March 19, the number will be 520-290-6277.

BUGS&BYTES

Emulator legit?

Recent posts on the Internet have indicated that Edward Schwartz has worked out the necessary legalities with Texas Instruments regarding use of proprietary materials in his emulator. Here's hoping we will see it back on the boards again soon.

Everywhere an emulator

Mike Brent of the Ottawa Users Group is reportedly working on yet another TI emulator — this one is for the Amiga.

Send Newsbytes to MICROpendium Newsbytes, P.O. Box 1343, Round Rock, TX 78680.

USER NOTES

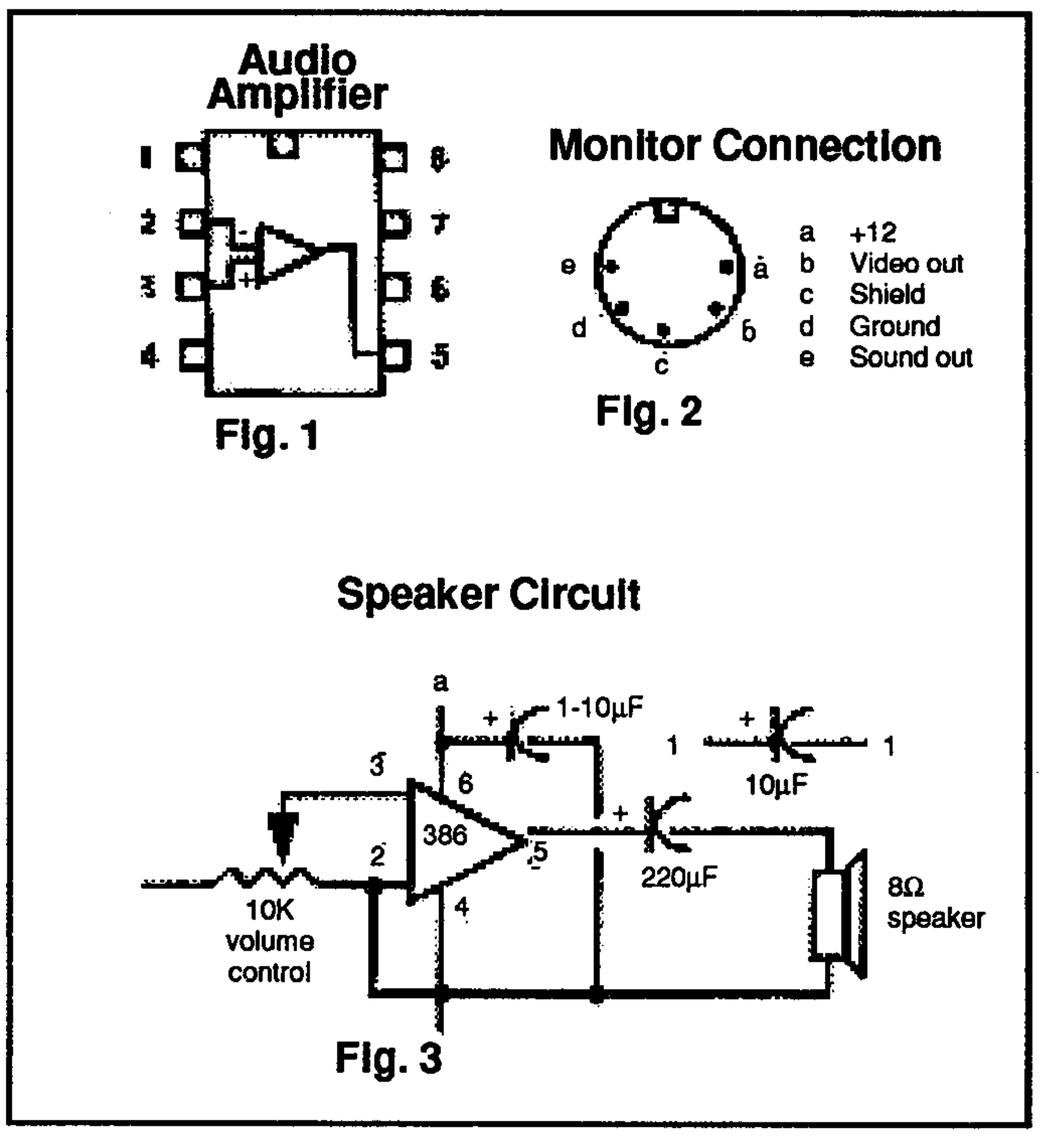
Building and installing a sound circuit

The following was written by Olden Warren of the Bluegrass 99 Computer Society. It appeared in the group's newsletter, Bytemonger. Since it involves hardware, the reader is entirely responsible for the outcome of the project.

My first monitor was an Amdek 300 monochrome that I acquired from work. I was thrilled to have it at the time because I was using a black and white TV set that I had during my college years. The monitor made my system look more computer-like and I didn't have to answer the question 'Why do you still have a black and white TV set in your room?' anymore. The only problem was that the monitor had no speaker, so I could hear sound. This was fine for many programs, but a major bummer when trying to play games.

Being an engineer by training I figured I could remedy this situation; and I did, with a little help from Radio Shack. I cam across a series of books called *Engineer's Mini-Notebook* by Forest M. Mims III, one of which was subtitled "Op Amp IC Circuits." Radio Shack sells these books for about $1 to $2. I also bought another book in this series called "Digital Logic Circuits," which was a tremendous help to me in other projects. The books are entirely handwritten on graph paper and, despite the title, can be understood by non-engineers. You will need to develop a little proficiency at soldering, but with a few hours of practice you can learn enough to put together basic circuits like the one below.

Basically, what I needed to do was to take the sound input from the TI, amplify it, and run it through a speaker so it could be heard. This is no different than what your ordinary stereo system does at home; its sound input is a CD player, cassette player or radio receiver; it amplifies the sound using an audio amplifier and then runs it through a network of speakers. The circuit below uses a 386 audio amplifier IC (Fig. 1) to perform the job. For those unfamiliar with how to read the pins on an integrated circuit, if you hold it in your hand with the pins facing down and you can read the marking on the chip, pin No. 1 is the one in the lower left corner.



The TI monitor connection (Fig. 2) provides the sound input as well as power and ground, so the circuit can be powered by the TI. The circuit (Fig. 3) is from the Mims books, but I made two adjustments. I added a capacitor on the power lead to clean it up and a second capacitor between No. 1 and No. 8 because it increases the gain (amplification) by a factor of 10.

There are a few enhancements that can be implemented to make assembly easier, and the finished product more attractive. First, you may want to use a socket for your IC so that you don't have to solder directly on the pins of the IC and risk damaging it. Second, I added a DPST (double-pole single-throw) switch across my power and sound inputs, so I could turn the speaker on and off. Third, make sure that the volume control resistor pot is located in a spot where it can be adjusted while it is on.

Finally, putting it in an attractive container makes it look more professional. I used a small, old FM radio that already had a speaker in it for my project. It worked out beautifully.

Sending TI-Base files to another computer

Harlan Warden of Melbourne, Iowa, has this tip:

Recent articles on sending TI-Base files to another computer included the instruction to increment the number of lines per page to a value higher than the number of records to avoid page breaks in the transmitted file. A simpler — and certainly more elegant — method is to set the lines-per-page value to zero. This turns off the page-break function for all file print operations.

TI vs. Pentium?

We found this in the December 1994 issue of Wordplay, the newsletter of the Portland Users of Ninety-Nines (PUNN):

By the way, did you know that your TI is more accurate on floating point math operations than the flawed Pentium chips? Yup, it's true. Pentium has error at 9 decimal places. The TI, while displaying 10 digits, actually keeps precision to 13 or 14 places internally. Displaying only 10 digits *usually* will eliminate cumulative round-off errors. However, *all* computers have limitations that can under the right circumstances give an incorrect answer. Try X=2/3 - 1/3 - 1/3. This will give a non-zero answer — i.e. -14, indicating a problem in the 14th place (on a TI99).

A cheap fix for loose HRD chips

This item was written by Harry Guenther and appeared in the newsletter of the

(See Page 30)

USER NOTES

(Continued from Page 29)

Long Island 99'ers.

Some owners of Horizon RAMdisks have experienced periods of less than perfect performance from their units. In my own case, my 6-year-old unit has had some ups and downs, but thanks to the excellent service of Bud Mills, a major improvement in reliability was obtained three years ago.

Since that service my board has done beautifully, except for two instances. Each of those occasions was characterized by completely unexpected malfunctioning after weeks of excellent service. The Funnelweb editor or certain Extended BASIC programs might refuse to fire up, or the Disk Review program might lack "windows" or have them only partially framed. Worse yet, none of the usual procedures to get the HRD to stabilize made any difference.

In each case a relatively direct approach did the job. I first ran the MEGTEST on the 32x8 memory chips. By chance, in each case, chip No. 6 was indicated to be in error on a couple of the test numbers. Since the computer had worked so well during its prior use, I doubted that the chip was bad.

Consequently, I removed the HRD from the Peripheral Expansion Box and carefully but firmly and evenly pressed down on the indicated chip. I then replaced the board.

On firing up, all indications were that the system was once again under control. Successful re-installation of RAMdisk Operating System 8.14 and all the other files on the partitioned HRD followed. Again, for a 4 to 6 month period the HRD worked without failure.

I submit this note for others to consider when troubleshooting erratic behavior in their own HRD.

Several factors, such as board removal, changes in temperature or humidity, etc., might in some cases lead to a slight shifting of a chip in its socket. The MEGTEST may indicate a slightly uncertain socket setting which, if you are as lucky as I have been, can be easily reinforced. The drawback is the inconvenience of losing the installed ROS and menu, but if you have kept a floppy copy it's no big deal, and the frustrations are over. You have also avoided the task and irritation of removing the batteries.

BUY - SELL - TRADE
HARDWARE - SOFTWARE
TI Buyers Guide $2
WANTED; *Disk Manager, MBX, Yahatzee, X-Basic, TI Recorders, Geneve, Hard Disk Controller, RS-232 Cards & Rare Items!*
Dual ½ HT Kit complete with cables, screws, drill template & instructions $59
PE Box Complete $149
Corcomp DD Cont $180
Star NX-1001 Printer $175
2400 Baud Modem $79
TI-99/4A Console $45
Rare - Myarc Hard & Floppy Controller with 10 Meg Hard Disk & Case $350
800-471-1600
(Nationwide & Canada ORDERS ONLY)
414-672-1600 Local/tech Support
Huge Genuine TI Inventory Since 1982
Competition Computer
2219 S Muskego Milwaukee WI 53215 Fax 414-672-8977
Open Daily 9-5 Sat 10-3
Bankcards, Discover, Checks & UPS/COD

Using a 40-column display may reduce size of TI-Writer files

The following item appeared in several user group newsletters. It was written by George Brandt.

Many times I have wondered what TI-Writer would be like if there was only a single window to look at. If you have, too, then this is for you.

Let's begin by asking ourselves how we use TI-Writer. If composing documents on the screen is typical, then it is a distinct advantage to not have to either scroll between three screens or to have to print the document to see what was written in previous sentence or paragraph.

Scrolling makes a 40-column screen seem one-third the size of an 80-column screen, which we all know is not true, but it does make reading difficult. Printing is easier, but adds extra steps and chops down a few trees in the process.

Well, there is another alternative. The secret is to set the left margin at zero and the right margin at 40 in the editor. The next requirement is that the line numbers are turned off (FCTN-0). This gives you a full 40-column screen and full wordwrap capabilities we appreciate so much.

To test the space requirements on the diskette, the following test was run:

Three files were created with the same data (all "X"s). The first file had the margin set at 21 and 60, with 10 paragraphs of six lines (240 characters) of data. The second file had the margins at one and 80 with 10 paragraphs of three lines (240 characters) of data. Comparing the sectors used, the first file required 19 sectors while the second used only 12 sectors.

This represents a 50 percent loss of capacity on each floppy and would raise a question about the end justifying the means. The light bulb came on. The only reason the margins were set to 21 and 60 was to get a full 40 columns of data on the screen. In truth, the only purpose was to eliminate the line numbers on the left side of the screen.

Hence the third file was created using margins at one and 40, and with line numbers turned off. With 10 paragraphs of six (240 characters) lines of data I was ready to use ShowDiretory to tell me the answer.

Surprise, surprise! The sectors used were not 19, or even 13. It required 12 sectors! The same as with a full line of 80 columns. If this fact surprises you as much as it did me, you may wish to set up a test to verify my results. In the meantime, I plan to do all of my composing on a 40-column screen, and then use the formatter to adjust the printed page back to a full 8 columns.

So, who says you can't get anything for

(See Page 31)

USER NOTES CLASSIFIEDS

(Continued from Page 30)

free? You now have a choice of 40- or 80-column mode without feeling guilty about requiring more disk space.

Using 132-column output with TI-Writer

This item was written by Andy Frueh of the Lima User Group. We saw it in the Kawartha Kronicle, the newsletter of the Kawartha 99ers, Peterborough, Ontario.

If you want to type in 132 columns using condensed mode, be sure line 0001 reads:

.LM 1;RM 132;FI;AD

Then start typing on line 0002. When finished, use PF then C DSK1.filename as the device. This removes control characters and the tab line. Enter the formatter and print out as usual.

MICROpendium pays $10 for items sent in by readers that are used in this column. Send your tips, hints and programs to MICROpendium User Notes, P.O. Box 1343, Round Rock, TX 78680; or email them to us at Delphi, GEnie or Internet (account numbers are listed on page 2).

ATTEND A TI FAIR IN 1995

FOR SALE

TI99/4A

TI software & hardware. Joy Electronics Inc., P.O. Box 542526, Dallas, TX 75354-2526. 214-243-5371. Call or write for free catalogue. v12n5

PROGRAM INNOVATORS SOFTWARE SALE

TOUCHDOWN: NFL Football Predictor $10.00 WALLSTREET: Investment Package $30.00 USVBA Power Volleyball (ML-Game) $10.00 Klingon Invaders, Destroy Klaatu, Desert Rat (Machine Language Games) $10.00 Cutthroat Cribbage, TItris, Cockroaches, Snomobile, Martian Missiles, $10.00 Multiple Orders 10% discount 4122 Glenway, Wauwatosa WI 53222-1116

SERVICES

PROGRAM DEBUGGING

Custom program debugging and conversion of repetitive lines into subprogram by a programmer with experience in XBASIC, TML and FORTH. Also experienced in programming the TI 300, 400, 500 programmable controllers. Don Steffen, Phone 503-873-4217 10082 Silverton Road, Silverton, Oregon 97381.

SERVICES

WANTED TO BUY

WANTED SOFTWARE & HARDWARE

1. Mechatronics GRAM-Karte 128K/512K with software. 2. J&KH Videotitle/all softwares. 3. SST Expanded BASIC Compiler with high resolution and Pre SST Prg. 4. Comand DOS TI99/4A by Monty Schmidt. 5. Mega RAM from Atronic. 6. John Guion TI Controller modification. 7. Maximem modification & need disk library and have most if need for back up. 8. Windows 99 for Turbo-Pasc 99 & Tool-Box. 9. Beginning A/L by D&D Publishing. 10. BASIC Compiler 99. 11. Expanded Graphics for Ext. BASIC. 12. John T. Dow A/L book & software. 13. Softies A/L tutorials and softwares. 14. Home Computer 99er 1-5 & Journal 3 & 4 & 5 & 6. 15. Asgard AEMS or AMS card with software & schematic. 15. Asgard screen scroll package Ext. BASIC. Oscar A. Ros, 14007 Hubbard St., Sylmar, CA 91342, 818-362-6387.

LOOK

What a deal!

Prices slashed on MICROpendium Classifieds!

Classified ads are now available for 10 cents per word,
a reduction of more than 50 percent.

If you want to sell or buy something,advertise it in MICROpendium Classifieds.
Simply write your ad on a separate sheet of paper, count the word
(a phone number counts as one word) and send it, along with payment, to:

MICROpendium Classifieds
P.O. Box 1343
Round Rock, TX 78680.

MasterCard VISA

The ONLY monthly devoted to the TI99/4A

Subscription Fees

❑ 12 issues, USA, **$35** ❑ 12 issues, Mexico, **$40.25**

❑ 12 issues, Canada **$42.50** ❑ 12 issues, other countries surface mail, **$40.00**

❑ 12 issues, other countries, air mail, **$52.00**

Outside U.S., pay via postal or international money order or credit card; personal checks from non-U.S. banks will be returned.

Address Changes

Subscribers who move may have the delivery of their most recent issue(s) delayed unless MICROpendium is notified six weeks in advance of address changes. Please include your old address as it appears on your mailing label when making an address change.

Check each item ordered (or list on separate page) and enter total amount here:_____

Check/MO ❑ MasterCard ❑ VISA ❑ (check one)

Card No. _____

Expiration Date _____

(Minimum credit card order is $9)

Signature _____

(Required on credit card orders.)

No sales tax on magazine subscriptions. Texas residents add 7.75% sales tax on other items, including back issues and disk subscriptions. Credit card orders add 5%.

Mail to: MICROpendium, P.O. Box 1343, Round Rock, TX 78680

Name _____

Address _____

City _____

State _____ **ZIP** _____

The set of numbers at the top of your mailing label indicates the cover date of the last issue of your subscription.

Disks, Etc.

❑ **Back Issues**, $3.50 each. List issues:_____

_____

No price breaks on sets of back issues. Free shipping USA. Add 30 cents, single issues to Canada/Mexico. Other foreign shipping 50 cents single issue surface, $1.50 airmail. Write for foreign shipping on multiple copies.

OUT OF STOCK: Vols. 1, No. 1-2; Vol. 2, No. 1

❑ **MICROpendium Index** (2 SSSD disks, 1984-1992), Extended BASIC required $6.00

❑ **MICROpendium Index II** (9 SSSD disks — 1 for each year — 1984-1992), XB required $30.00

❑ **MICROpendium Index II with MICROdex 99** (11 SSSD disks), XB required.....$35.00

❑ **MICROdex 99** (for use with MP Index II, 2 SSSD disks), XB required $10.00

❑ **MICROpendium Index II** annual disks ordered separately (1 disk per year, 1984-1992); each $6.00

MICROdex 99, by Bill Gaskill, is a collection of programs that allow users of MP Index II to modify their index entries, as well as add entries. MICROdex 99 supports many other functions, including file merging, deletion of purged records, record counting and file browsing.

GENEVE DISKS (SSSD unless specified)

❑ **MDOS 2.0** (req. DSSD or larger (for floppy & hard drive systems) $4.00

❑ **GPL 1.5** $4.00

❑ **Myarc Disk Manager 1.50**.....$4.00

❑ **Myarc BASIC 3.0**.....$4.00

❑ **MY-Word V1.21** $4.00

❑ **Menu 80** (specify floppy or hard disk versions(s); includes SETCOLR, SHOWCOLOR, FIND, XUTILS, REMIND $4.00

GENEVE PUBLIC DOMAIN DISKS

These disks consists of public domain programs available from bulletin boards. If ordering DSDD, specify whether Myarc or CorComp.

	SSSD	DSSD	DSDD
❑ **Series 1**	$9.00	$7.00	$5.00
❑ **Series 2**	$9.00	$7.00	$5.00
❑ **Series 3**	$9.00	$7.00	$5.00
❑ **Series 4**	$9.00	$7.00	$5.00
❑ **Series 5**	$9.00	$7.00	$5.00
❑ **Series 6**	$9.00	$7.00	$5.00

SECOND CLASS

A T EXPIRES 1/95
CHARLES GOOD
P.O. BOX 647
VENEDOCIA OH 45894