

CIRCUIT CELLAR **I N K**®

THE COMPUTER APPLICATIONS JOURNAL

Computer-
Controlled Video
Editing

Scientific
Visualization

RISC For
Embedded
Control

Graphics & Video

SPECIAL SECTION:
Schematic Capture
Software (EECAD)



October/November 1991 — Issue 23

\$3.95 U.S.
\$4.95 Canada



0 7447075349 0

English: The Forgotten Language?

EDITOR'S INK

Ken Davidson

Is the English language really so obscure that people have given up trying to use it **correctly altogether**? Granted, English is one of the more difficult languages to learn due to its endless exceptions to rules. It's usually more important to get your idea across than to get all the parts of the language correct. My frustration stems from my recent observations of some so-called "editors" that can't even catch a spelling mistake let alone bad punctuation or grammar.

We recently started the search for additional staff editors. All that we're asking of these people is that they be able to read a raw manuscript; fix spelling, punctuation, and grammar; and convert any "engineerese" into readable English. As part of the interview process, **we've** been giving each applicant a test consisting of two of the new product releases being used in this issue, and asking them to fix any mistakes they find. In addition to the problems already contained in them, I doctored each release with some specific mistakes that I was interested in. The results have been, frankly, disappointing.

Most of the applicants had some form of English degree and experience in at least technical writing; many had some editing background. Virtually everyone missed one or more blatant spelling mistakes, and the vast majority of them missed many of the planted punctuation, grammar, and usage mistakes. There were even typos in some of the resumes and cover letters we received (we didn't call those people back). Luckily, we had a handful of people that did fairly well, so we should be in good shape when it comes time to make some offers.

I learned a long time ago while corresponding on computerized conferencing systems that, regardless of what someone had to say, if the message looked like hell-full of spelling and punctuation mistakes—that person came across looking less intelligent than he probably was. We see it all the time on the Circuit Cellar BBS, though no one corrects anybody since we are more interested in what people have to say than in how they say it. We also, unfortunately, see it in article manuscripts as well, where it is more important. That's why we need the editor—one who can fix the things that very few of the applicants have been able to fix in the first place!

I'll be the first to admit that there are mistakes between the covers of every issue of *Circuit Cellar Ink*. No matter how many pairs of eyes look at each article, something always slips by. We just received a diatribe from an irate reader pointing out two mistakes in a recent issue. One was in usage, the other in capitalization. I was, of course, embarrassed when I saw them, since they

should have been caught early on. However, since he didn't seem to be able to find anything else wrong, I think two bad words out of the thousands in the issue isn't half bad. He could have a field day with some other publications I see on the newsstand each month.

I don't expect our authors to be able to write prize-winning articles; I certainly can't. I also don't want to discourage anyone from submitting an article because they don't think it's good enough; it can be massaged into something we can all be proud of. What I would like to encourage, though, is that when writing a piece, have someone else look at it to see if it even makes sense. Run it through the spelling checker on your word processor for obvious blunders. The more subtle things can be fixed later.

A FAREWELL

Why are we hiring new editors? The answer is twofold: we need someone to help an already overworked staff, and we recently lost our editor-in-chief, Curt Franklin, to another (noncompeting) publication. Curt came to us three years ago when *Circuit Cellar Ink* looked like something a bunch of engineers had thrown together (because that's what it was). With his guidance, *Circuit Cellar Ink* has developed into a highly regarded journal read and enjoyed both by engineers in some of the finest companies all over the world and by those who simply have a love of tinkering with computers. Curt will be sorely missed, but life goes on, and we plan to continue to bring you a first-class publication destined for the reference shelf after being read cover to cover.



FOUNDER/
EDITORIAL DIRECTOR
Steve Ciarcia

MANAGING EDITOR
Ken Davidson

ENGINEERING STAFF
Jeff *Bachiochi*
Ed *Nisley*

CONTRIBUTING
EDITORS
Tom Cantrill
Chris Ciarcia

NEW PRODUCTS
EDITOR
Harv Weiner

EDITORIAL ASSISTANT
Lucy Tralongo

ART DIRECTOR
Lisa Ferry

PRODUCTION
MANAGER
Mark Vereb

STAFF RESEARCHERS:

Northeast
John Dybowski
Midwest
Jon Elson
Tim McDonough
West Coast
Frank Kuechmann

Circuit Cellar BBS-24 Hrs. 300/
1200/2400 bps, 8 bit, no parity,
1 stop bit. (203) 871-1988;
9600 bps HST (203) 871-0549

All programs and schematics
in *Circuit Cellar INK* have been
carefully reviewed to ensure that
their **performance** is in **accord-**
ance with the specifications de-
scribed, and programs are
posted on the Circuit Cellar BBS
for **electronic transfer** by subscrib-
ers.

Circuit Cellar INK makes no
warranties and assumes no re-
sponsibility or **liability** of any kind
for **errors** in these programs or
schematics or for the conse-
quences of any such errors. Fur-
thermore, because of the possi-
ble **variation** in the **quality** and
condition of **materials** and work-
manship of reader-assembled
projects, *Circuit Cellar INK* **dis-**
claims any **responsibility** for the
safe and proper function of
reader-assembled projects
based upon or from **plans**, de-
scriptions, or information **pub-**
lished in *Circuit Cellar INK*.

Entire contents copyright ©
1991 by Circuit Cellar **incorpo-**
rated. All **rights** reserved. Repro-
duction of this publication in
whole or in part without **written**
consent from *Circuit Cellar inc.* is
prohibited.

Cover Illustration
by Robert Tinney

CIRCUIT CELLAR **I N K**®

THE COMPUTER APPLICATIONS JOURNAL

In This
Issue...

FEATURES

16

A Video Editing Control System-Part 1 *The Hardware*

by *William J. Kressbach*

The key to a successful video editing session is **coordinating** the two tape decks. Part 1 of this controller project describes the hardware end of doing just that.

26

Computer Graphics and the World of Scientific Visualization

by *Chris Ciarcia*

The days of wading through **reams** of printed data are fading fast. Chris Ciarcia shows us the basics of putting those stuffy numbers into motion.

40

Add a Video Display to Your 8031 Microcontroller *Graphics and Color Live Up Any Output*

by *Larry Duarte*

Tired of hard-to-read, cryptic LCD displays? Find out how to easily add video to your next controller project.

46

ISDN (S/T) Interface-Part 2

Design Example of a PC Plug-in Board

by *Steven E. Strauss & P.K. Govind*

We finish up our ISDN Interface description with a sample implementation.

SPECIAL SECTION: Schematic Capture Software (EEDAD)

60

OrCAD Schematic Design Tools

A Working Engineer's Impression

by *Bruce Webb*

Schematic capture software is quickly becoming a necessity in any design engineer's toolbox. Find out what one engineer thinks of *OrCAD* as he uses it in his everyday labors.

64

Schematic Capture with Schema

by *Ken Davidson*

Schema is another schematic package that has been around for a number of years. Managing Editor Ken Davidson has been using the software in his own design work and fills us in on his opinions about it.

DEPARTMENTS

1

Editor's INK
English: The Forgotten language?
by Ken Davkton

4

Reader's INK-Letters to the Editor

10

NEWProductNews

70

Firmware Furnace
(Re-) Starting C
by Ed Nisley

79

From the Bench
Redefining Remote Control
Now You See 'em-Beep-Now You Don't
by Jeff Bachiochi

86

Silicon Update
Nuts About RISC
Go on a low-Fat Acorn Diet
by Tom Cantrell

94

Practical Algorithms
Measuring Subjective Sound levels
by Charles P. Boegli

103

ConnecTime—Excerpts from the Circuit Cellar BBS
Conducted by Ken Davidson

112

Steve's Own INK
The Circuit (Storm) Cellar
by Steve Ciarcia

97

Advertiser's Index

PUBLISHER

Daniel Rodrigues

PUBLISHER'S

ASSISTANT

Susan McGill

CIRCULATION

COORDINATOR

Rose Mansella

CIRCULATION ASSISTANT

Barbara Maleski

CIRCULATION

CONSULTANT

Gregory Spitzfaden

BUSINESS MANAGER

Jeannette Walters

ADVERTISING

COORDINATOR

Dan Gorsky

HAJAR ASSOCIATES NATIONAL ADVERTISING

REPRESENTATIVES

NORTHEAST

Debra Andersen

(617) 769-8950

Fax: (617) 769-8982

MID-ATLANTIC

Barbara Best

(908) 741-7744

Fax: (908) 741-6823

SOUTHEAST

Christa Collins

(305) 966-3939

Fax: (305) 985-8457

MIDWEST

Nanette Traetow

(708) 789-3080

Fax: (708) 789-3082

WEST COAST

Barbara Jones

& Shelley Rainey

(714) 540-3554

Fax: (714) 540-7103

CIRCUIT CELLAR INK (ISSN 0896-8985) is published bimonthly by Circuit Cellar Incorporated, 4 Park Street, Suite 20, Vernon, CT 06066 (203) 875-2751. Second-class postage paid at Vernon, CT and additional offices. One-year (6 issues) subscription rate U.S.A. and possessions \$17.95, Canada/Mexico \$21.95, all other countries \$32.95. All subscription orders payable in U.S. funds only, via international postal money order or check drawn on U.S. bank. Direct subscription orders to Circuit Cellar INK, Subscriptions, P.O. Box 3050-C, Southeastern, PA 19398 or call (215) 630-1914.

POSTMASTER: Please send address changes to Circuit Cellar INK, Circulation Dept., P.O. Box 3050-C, Southeastern, PA 19398.

STANDARDS HOTBEDS

I seldom write a letter like this, but Steve's editorial in the August/September (#22) issue of *CIRCUIT CELLAR INK* is more than I can resist. It says, "Standards happened.. .by committees made up of competitors and large users, each with a particular axe to grind and no strong motivation for real standardization." Later, you remark (almost ruefully) that you have never served on a standards committee.

You should fall to your knees and thank God that He spared you that horrible experience. In my lifetime I've served on two. The most recent was an I.F.A.I. subcommittee of the S.A.E., and I do not exaggerate when I say that the antics of most of the other members almost drove me insane. Things degenerated to the point where I was actually heckled in technical presentations. All because a small minority wanted the S.A.E. to adopt without *modification* a test procedure issued by I.S.O. that could be shown by theory and data to be worthless. Conversations with others who had served on such committees convinced me that they are, almost without exception, hotbeds of political infighting.

Charles Boegli
Blanchester, OH

DEVICE DRIVER NITS

I just received the August/September (#22) issue of *CIRCUIT CELLAR INK* and read Chris Ciarcia's "Using Device Drivers to Change the Rules." I enjoyed it very much. Based upon my knowledge gained from reading such publications as "The MS-DOS Encyclopedia," I think there are a few inaccuracies in his article. These are based upon my familiarity with MS-DOS Version 3.3, however.

In the section "From The Top," you explain how the BIOS ROM searches for extensions and "...marks them with a unique byte sequence which identifies them as ROM." I believe that each ROM starts with a unique "55AA" signature that is verified by the POST. Each such

extension is then called at its entry point just after this signature to perform initialization. The POST can't mark a ROM, nor change any location in an extension it finds.

In addition, the ROM bootstrap does read in the first sector of a disk. In the case of a floppy, this is the actual bootstrap code that attempts to locate and read `IO.SYS` and `MSDOS.SYS`. However, on a hard disk that has multiple partitions, this first sector contains the partition table and another small program that finds the active (bootable) disk partition and then reads the actual bootstrap code from that partition, which is then treated like the first sector of a floppy disk. The partition table processing precedes the actual bootstrap process, but it must happen in order for the real disk bootstrap to occur.

In the section "Tables and Routines," you reference the device header in Figure 2 as being 18 bytes. However, in Listing 1, the 8-byte device name is only initialized to "DRVR- 0 ", or 7 bytes.

In the section "Assembling the Driver," you decided to name your handler "DRIVER(.ASM)," which compiles and links to "DRIVER.SYS", a unique name if I ever heard one! Unfortunately, back in the old days when Microsoft and IBM poured the concrete around MS-DOS, someone had the brainy idea to name the one and only device driver that anyone would ever need to have, for use on hard and floppy disks, as you guessed `it-DRIVER.SYS`, and not some more descriptive name like `HD.SYS` or `FD.SYS`. Now here you are with your own device driver and what do you call it? Later on you refer to "DRVR" which is at least a little more unique.

A little later, you warn us to "...keep a system disk handy" in case the driver code locks the keyboard out. Of course you mean a bootable floppy disk, but some people may not realize this, and this disk needs to have sufficient files on it (like a text editor) to allow you to edit the hard drive's `CONFIG.SYS` and `AUTOEXEC.BAT` files to temporarily remove the commands that loaded in the driver and caused the system to lock up. In addition, if you happen to have a hard drive configuration that requires a device driver to access extended partitions, your safety disk must have this software on it as well. A safer, and more easily recoverable, method is to put your new driver onto a bootable floppy disk, insert that disk into drive A, and reboot the computer. This has several benefits, one of

which being if the system locks up, you only have to open the floppy drive door and reboot the system to bring it back up. In addition, you can configure the floppy disk **such that** it never accesses the hard drive, offering some level of protection in case the driver happens to write data somewhere. A floppy disk is a lot easier to reconstruct and certainly more disposable in case something goes wrong. Granted, your driver does so little, nothing can go wrong, but someone else may write a more sophisticated driver some day.

I may be "picking nits," but such a fine magazine and article should continue their goals of accuracy and perfection. **As** always, keep up the good work.

Bob Meister
Hamden, CT

Chris Ciarcia responds:

I appreciate the effort that Mr. Meister has gone through to ensure that my article on "Device Drivers" continues to maintain the Circuit Cellar tradition of accuracy and perfection, so I will try to respond to his comments as best I can. As can be seen from the article, I am not a professed expert in "writing device drivers." The function of the article was to transfer some of my experience and recent efforts during a development project which involved creating a simple device driver to you the reader. This task was described in the section called "A Learning Experience...."

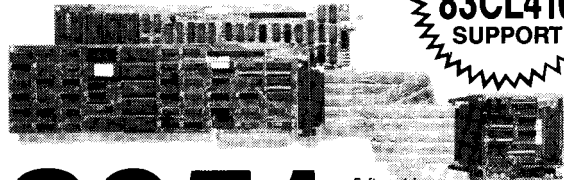
Based on his reading of "The MS-DOS Encyclopedia" and his familiarity with MS-DOS Version 3.3, Mr. Meister points out an error I made in my section called "From the Top," where he proclaims "... you explain how the BIOS ROM searches for extensions and '...marks them with a unique byte sequence which identifies them as ROM' " and then he goes on to say "... POST can't mark a ROM, nor change any location in an extension it finds."

In response, let me begin by saying that we are not marking ROM nor changing locations there. Paraphrasing from the article, "the ROM bootstrap initialization routine sets up some basic parts of the interrupt vector table... and it then initializes the ROM BIOS tables..." He is correct. These extensions are not "marked" during the initialization process. What I tried to say is "these extensions are marked with a unique sequence that identifies them as ROM."

And yes there are several operations going on during the bootstrap phase of the initialization. But within this article, my goal was to 'briefly' overview the boot process to give you, the reader, a sense of the flow of the initialization actions. I was not prepared to discuss every detail. For those of you interested in the details of this operation I refer you to the "DOS Programmer's Reference" by Terry Dettmann, (Que Corp., Carmel, Indiana) where it says:



"The Best 8051 Emulator"



8051

5 ft. cable

SEE EEM 90/91
Pages D 1320-1323

PC based emulators for the 8051 family

8031, 8032, 8051, 8052, 80C152/154/321/451/452/51FA/51GB/515/517/535/537/552/562/652/851, 80532, 83C451/552/652/751/752/851, 8344, 87C451/552/751/752, 8751, 8752, DS5000 + CMOS more.

- PC plug-in boards or RS-232 box.
- Up to 30 MHz real-time emulation.
- Full Source-level Debugger w/complete C-variable support
- 48 bit wide, 16K deep trace, with "source line trace."
- "Bond-out" pods for 8051, 836552, 83C451, 83C652, 83C751, 80C515/80C517, 83C752.

Prices: 32K Emulator 8031 \$1790: 4K Trace \$1495* *US only

CALL OR WRITE FOR FREE DEMO DISK!

Ask about our demo **VIDEO**

NOHAU
CORPORATION

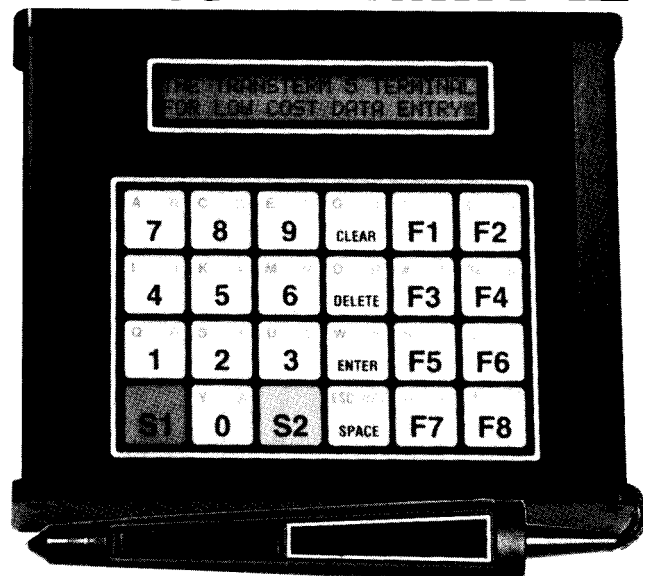
51 E. Campbell Avenue
Campbell, CA 95008
FAX (408) 378-7869
(408) **866-1820**

408-378-2912
Nohau's 24-hour
information center to
receive info via your FAX

Order Service #185

See us at The *Embedded Systems Conference South #316*
and at *Wescon South #460*

\$249. TERMINAL



- Featuring
- Standard RS-232 Serial Asynchronous ASCII Communications
 - 48 Character LCD Display (2 Lines of 24 each)
 - 24 Key Membrane Keyboard with embossed graphics.
 - Ten key numeric array plus 8 programmable function keys.
 - Four-wire multidrop protocol mode.
 - Keyboard selectable SET-UP features-baud rates, parity, etc.
 - Size (5.625" W x 6.9" O x 1.75" H), Weight 1.25 lbs.
 - 5 x 7 Dot Matrix font with underline cursor
 - Displays 96 Character ASCII Set (upper and lower case)
- Options—backlighting for display, M-422 I/O, 20 Ma current loop I/O,

COMPUTERWISE, INC.

302 N. Winchester • Olathe, KS 66062 • 913-829-0600 • 800-255-3739

Order Service #128

"The ROM bootstrap routines now read the **disk bootstrap** code from the first sector (the boot sector) of the boot disk. The bootstrap code is a minimal-services routine responsible for getting the system up and running. The ROM bootstrap routines check all bootable disk drives for the presence of a boot sector on the disk. On a hard disk system with a single floppy disk, the ROM routines first check drive C and then drive A. If no boot sector is found, an IBM PC transfers control to ROM BASIC and starts up a diskless system; PC compatibles prompt you to insert a system disk and wait for you to press a key.

"When a bootstrap record is located, the ROM bootstrap loads it into high memory, away from where DOS itself is loaded. Control is then transferred to the disk bootstrap routine.

"After the bootstrap code has been loaded and has control, it looks back to **the** disk to locate the files `IO.SYS` and the `MSDOS.SYS`."

I felt it was sufficient and appropriate within the article's goal and context to condense the above description into two sentences. If Mr. Meister feels that this was inappropriate, I invite him to submit an article detailing these procedures for us. I personally would enjoy learning more about this aspect of the boot phase.

Now as to the rest of Mr. Meister's comments, I admit that I find myself slightly put off. The fact that I used seven of the available eight bytes for my device driver name seems a "nit point." And "keeping a system disk handy" in

my mind is not a bad idea. Unlike Mr. Meister, I feel **that** most of you are computer literate readers. I believe you all know what I mean. As to whether or not one uses the hard disk or a **floppy** disk environment for driver development, that's entirely up to you. I don't write drivers on a daily basis. I needed to use my Microsoft assembler and its debugger. That was installed on my hard disk, not a **floppy**.

Chris Ciarcia
Los Alamos, NM

We Want To Hear from You!

Write letters of praise, condemnation, or suggestion to the editors of Circuit Cellar INK at:

Circuit Cellar INK Letters to the Editor

4 Park Street
Vernon, CT 06066

Circuit Cellar BBS: 'editor'

EXPRESS CIRCUITS

MANUFACTURERS OF PROTOTYPE PRINTED CIRCUITS FROM YOUR CAD DESIGNS

TURN AROUND TIMES AVAILABLE FROM 24 HRS — 2 WEEKS

Special Support For:

- TANGO. PCB
- TANGO SERIES II
- TANGO PLUS
- PROTEL AUTOTRAX
- PROTEL EASYTRAX
- smARTWORK
- HiWIRE-Plus
- HiWIRE II
- EE DESIGNER I
- EE DESIGNER III
- PADS - PCB
- ALL GERBER FORMATS

- FULL TIME MODEM
- GERBER PHOTO PLOTTING

NEW!

WE CAN NOW WORK FROM YOUR EXISTING ARTWORK BY SCANNING. CALL FOR DETAILS!

Express
—
Circuits

314 Cothren St., P.O. Box 58
Wilkesboro, NC 28697

Quotes:
1-800-426-5396
Phone: (919) 667-2100
Fax: (919) 667-0487

COLOR CRT CONTROLLER ON STD BUS

A color graphics CRT controller, based on Hitachi's HD63484 ACRTC (Advanced CRT Controller) graphics processor, has been announced by Cubit. Using high-level commands, the Model 7050 controller offloads much of the graphics handling load from the main system CPU, leaving it free for other tasks.

The 7050 supports up to three screens with 640 x 480 resolution. It includes 4 Mbits of video RAM and is compatible with VGA, EGA, and monochrome monitors. Sixteen colors from a range of 40% may be selected, and 16 shades of gray can be displayed on monochrome monitors. Both digital and analog monitors are supported through Brooktree's RAMDAC chip. The board is

not CPU dependent and can be used with either 8-bit or 16-bit Intel and Motorola microprocessor-based CPU boards that meet the STD bus specification.

More than 20 graphic drawing commands, including LINE, RECTANGLE, POLYLINE, POLYGON, CIRCLE, ELLIPSE, ARC, ELLIPSE ARC, and PAINT are made available by the ACRTC. Memory management for split screens, zooming, and scrolling are also supported.

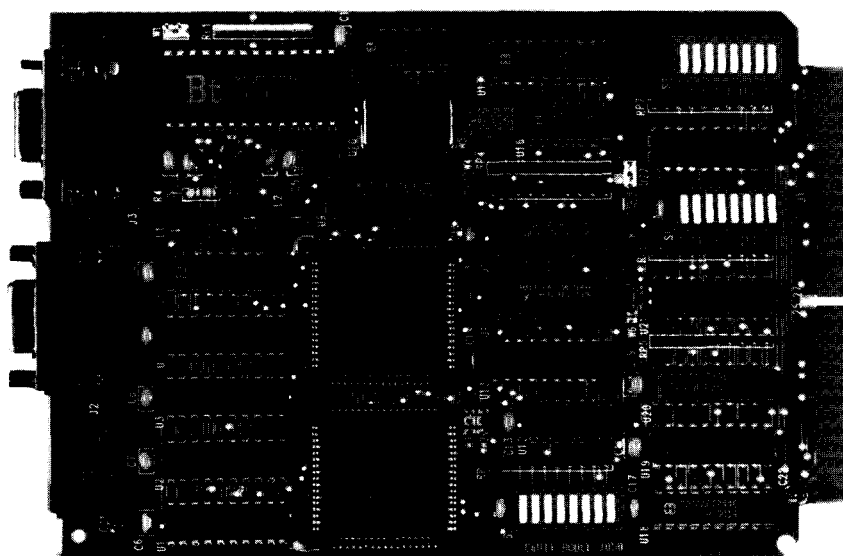
Cubit provides a library of software routines written

in both Borland Turbo C++ and assembly language, and a complete set of alphanumeric characters. The board does not need a DOS operating system since Cubit's library provides the necessary code.

The Model 7050 CRT Controller sells for \$490.00 in single quantities.

Cubit
340 Pioneer Way
Mountain View, CA
94041-1577
(415) 962-8237
Fax: (415) 965-9355

Reader Service #501



MICROMINIATURE CCD CAMERA

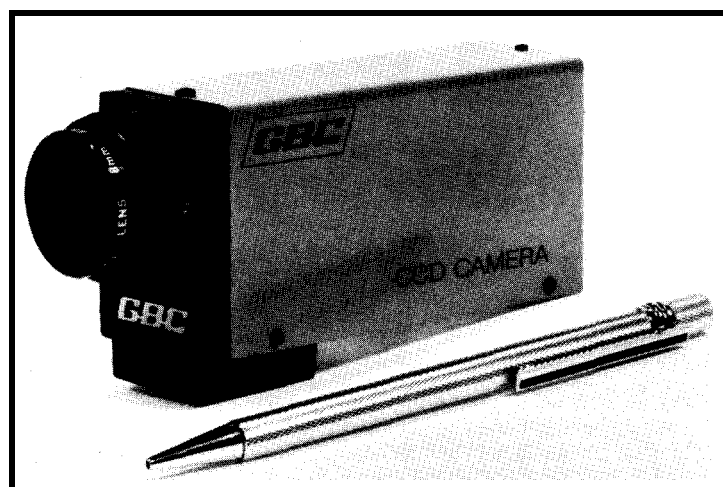
A microminiature solid-state CCD video camera, featuring a unique Microelectronic Shutter has been introduced by CCTV Corporation. The "GBC" CCD300 system allows the Sensor itself to compensate for all light changes, eliminating the need and cost of the traditional autoiris lens. Unlike a tube-type camera, there is no lag, burn in, or image retention.

Measuring 2.38"H x 1.38"W x 4.38"L, the 0.75-pound unit can utilize both "C"- and "CS"-type lenses. It operates from low voltage (7 to 12 volts DC) and comes standard with a 120-volt AC-to-low-voltage DC power module. Full video can be achieved with light levels as low as 2 lux (0.2 footcandles).

Resolution is in excess of 350 lines, both at the center and corners. The CCD300 features adjustable gamma, automatic black level, built-in image enhancer, mirror image reversal, and switchable auto/manual gain. The automatic gain control has a range of 1000 to 1 (four f-stops electronically within the camera). The light compensation is a minimum of 10,000 to 1 electronically and there is no geometric distortion. No price was available at press time.

CCTV Corporation
315 Hudson St. • New York, NY 10013
(212) 989-4433 • Fax: (212) 463-9758

Reader Service #502



NEWPRODUCTNEWS/NEWPRODUCTNEWS

TINY CONTROLLER BOARD INCLUDES DESIGN TOOL FEATURES

An 8051-based single-board controller for data collection, embedded control, and product design applications has been announced by Blue Earth Research. The Micro-440 uses a 12-MHz Intel 83C51FB, which includes advanced features such as high-speed I/O, three 16-bit timer/counters, multiprocessor communications, a Boolean processor, and a watchdog timer. Measuring 1.89" by 2.25", the 6-layer board is manufactured using double-sided surface-mount technology.

Time- and date-based operations are managed by the real-time clock/calendar module. The module features a 12/24-hour format, automatic leap year setting, and interrupt output periods ranging from 1/64 second through 1 hour.

For measuring analog inputs, the on-board 8-bit ADC can convert signals ranging from 0 to 5 volts in less than 40 microseconds. The eight input channels can be programmed for single-ended or differential operation.

Available I/O includes 14 TTL/CMOS-compatible I/O lines, dual RS-232C serial ports with activity LEDs, and a low-power shutdown feature. CPU bus connections are also available for adding memory or other peripherals.

The Micro-440 can be powered from any 6- to 16-volt DC source capable of 75 mA (7 mA in standby). The on-board regulator provides a stable +5 volts ($\pm 1\%$) to internal circuitry and includes a CPU reset feature. An optional 3.6-volt lithium battery maintains RAM data and clock operation for more than 10 years.

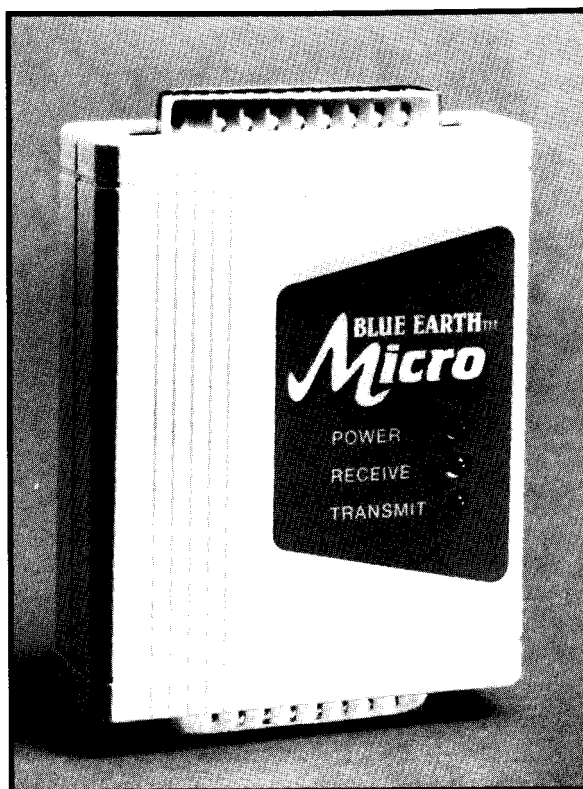
Available evaluation units include the controller board, back-up battery, and two 25-pin D-shell connectors, assembled and installed in a protective plastic housing. A complete system design package includes the Evaluation Unit; Macro Assembler, Symbolic Debugger, and Utility programs; comprehensive manuals (1100+ pages), plug-in type DC power supply, applications development module; and serial interface cable. Also available are Intel's PL/M-51 compiler and Franklin Software's C51 compiler in value-priced packages that include the Micro-440 evaluation unit and accessories.

The Micro-440 sells for \$99, in quantities of 1000. The Evaluation Unit sells for \$199, and the System Design Package is \$379.

Blue Earth Research

310 Belle Ave. • Mankato, MN 56001 • (507) 387-4001 • Fax: (507) 387-4008

Reader Service #503



HIGH-RESOLUTION IMAGE CAPTURE BOARD

Supervision/16, a video image capture system for the IBM PC/AT family of computers, is available from IDEC Inc. The Supervision/16 package consists of the IDEC frame grabber and software for image capture.

The software and hardware is fully compatible with all IBM-style AT-type machines and allows the user to capture video images from any standard RS-170 video source, such as a camera,

video tape, or live broadcast. The image is captured with a resolution of 512 pixels by 488 lines with 256 shades of gray. The resulting picture can be displayed on any VGA monitor in the 320 x 200 x 256 mode. The picture displays as 256 x 200 with 64 shades of gray.

Many super VGAs are supported to allow viewing of the images in 640 x 480 x 256 mode with the image displayed as 512 x 480 with 64 shades of gray. The image can be adjusted for contrast and brightness, stored to and retrieved from disk, and

printed on a laser printer.

With the super VGA display, the 256-gray-level picture rivals black-and-white TV broadcast quality, as the eye can detect no digital artifacts at this level. The choice of display has no bearing on the print quality since the printed image is printed directly from the disk file image, not from the screen as with most screen capture utilities.

The image is captured in 1/30 second and is stored in TIFF or PCX format for direct use by many desktop publishing packages. Pictures

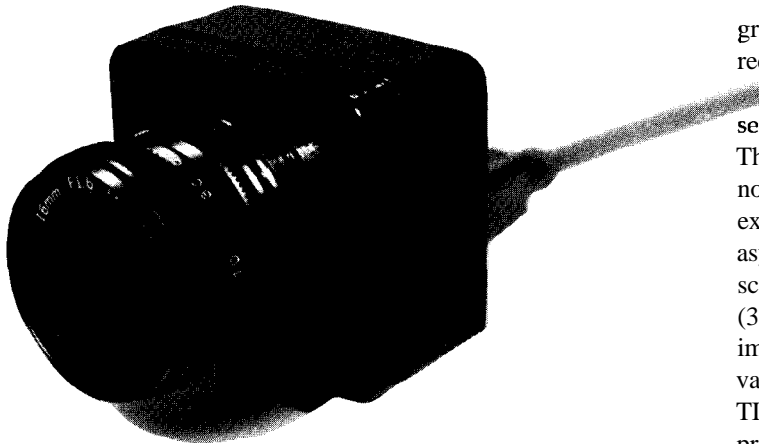
are easily included in such packages without the expense and limitations of a desktop or hand-held scanner.

The Supervision/16 package includes interface card, software on disk, owner's manual, and one-year warranty. The Supervision/16 costs \$369.95 in single quantity.

IDEC, Inc.

1195 Doylestown Pike
Quakertown, PA 18951
(215) 538-2600
Fax: (215) 538-2665

Reader Service #504



HIGH-RESOLUTION ELECTRONIC IMAGER

Electrim Corporation has announced the availability of a high-resolution version of their EDC-1000 solid-state electronic imager. The **EDC-1000HR** gives the user the ability to directly digitize images at up to 754 x 488 pixels. The EDC-1000HR is a compact, digitally controlled, digital output television-like monochrome camera.

The EDC-1000HR is fully compatible with an IBM PC XT/AT or equivalent and does not require a frame grabber or other third-party hardware or software. All popular IBM PC

graphic adapters are supported, but VGA or super VGA is recommended.

The EDC-1000HR uses a frame-transfer CCD image sensor configured into 244 lines with 754 elements in each line. The imager can be operated in either interlaced or noninterlaced mode. Features include: computer-controlled exposure time, frame scanning time, and subarray scanning; asynchronous scanning (external triggering of frame reset and scan); and pixel data collection rates of one megapixel/second (3 to 5 frames per second in live mode). Output from the imager is an 8-bit digital signal corresponding to the quantized value of brightness at serially sampled spatial data points. TIFF and PCX file images can be saved for use by image processing and desktop publishing packages.

The EDC-1000HR digital, asynchronous camera and computer interface card operate entirely under the control of the PC, collecting image data via the PC bus in parallel. Upon a command from the PC, image exposure takes place and 256 gray level image data is read directly into the computer's RAM.

The EDC-1000HR sells for \$850.

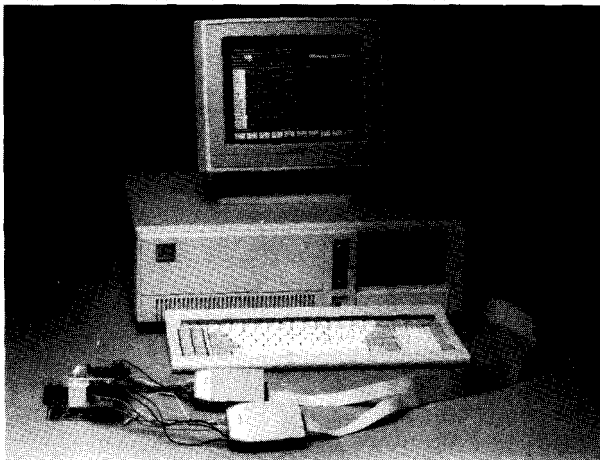
Electrim Corporation

P.O. Box 2074 • Princeton, NJ **08543**

(609) 683-5546 • Fax: (609) 683-5882

Reader Service #505

PC-Based Logic Analyzers



Sophisticated Logic Analysis at Unsophisticated Prices

ID160 (50 MHz) for \$695

*ID161 (100 MHz) for \$895

• 50 MHz or 100 MHz Sampling • 8K Trace Buffer • 32-channel Operation *Multi-Level Triggering *State Pass Counting *Event Timer/Counter *Performance Histograms *Hardcopy Output *Disassembles popular 8-bit micros *and much more !

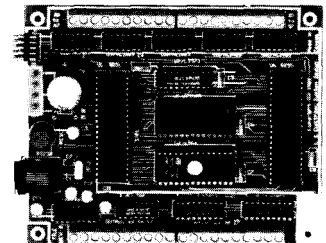
*30 Day Money Back Guarantee



INNOTECH DESIGN, INC.
6910 Oslo Circle, Suite 207
Buena Park, CA 90621
Tel: 714-522-1469 FAX: 714-527-1812

8031 CONTROLLER BOARDS

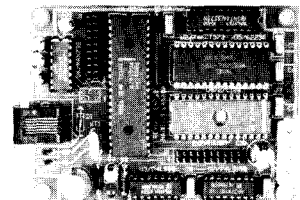
- 11.0592 MHz clock
- 16 K EPROM, 8 K RAM sockets
- 8255 parallel I/O
- ULN2803 outputs, 16 lines
- MC1489 inputs, 20 lines
- MAX232 serial I/O
- Screw terminals
- Support software included
- Monitor & PLC EPROMS & accessories available



DG31 Board (4½ x 5½")

A&T: \$129 — Partial kit: \$70 — PCB Board kit: \$25

- 11.0592 MHz clock
- 8/16 K EPROM socket
- 8/32 K RAM socket
- MAX232 serial I/O
- Two 20-pin expansion headers
- Monitor, application notes & support software included
- Accessories available



EMC32 Board (3" x 4")

A&T: \$80 — Full kit: \$62 — Partial kit: \$36

L.S. ELECTRONIC SYSTEMS DESIGN

2280 Camilla Rd., Mississauga, Ont. L5A 2J8 Canada

Phone/Fax: (416) 277-4893

Terms: Shipping US/Canada \$6. Check or Money Order please.

NEWPRODUCTNEWS/NEWPRODUCTNEWS

MICROCONTROLLER SUPPORTS IN-SYSTEM SOFTWARE UPDATES

A new, miniature microcontroller from Dallas Semiconductor accepts software updates through its serial port without any component removal. That is, software can be upgraded while the board is still in the system, allowing users to build a standard product and configure it with custom software just prior to shipment. Software upgrades can even be downloaded over a telephone line from a remote PC. The DS2340 Flip Stik supports DOS-equivalent operating systems for diskless embedded systems, enabling application development using standard DOS function calls.

The Flip Stik incorporates a V40 microprocessor

(software compatible with the 8088) running at 8 MHz, up to 256K bytes of nonvolatile RAM, and a recently introduced Dallas Semiconductor chip called the DS5340 Softener. This chip crashproofs the microprocessor to safeguard critical data against power failure. A

version of the Flip Stik features a clock/calendar for time stamping and dating data in the nonvolatile RAM.

The Flip Stik is so named because it functions as an expandable microprocessor when plugged in one way, and a single-board microcontroller when

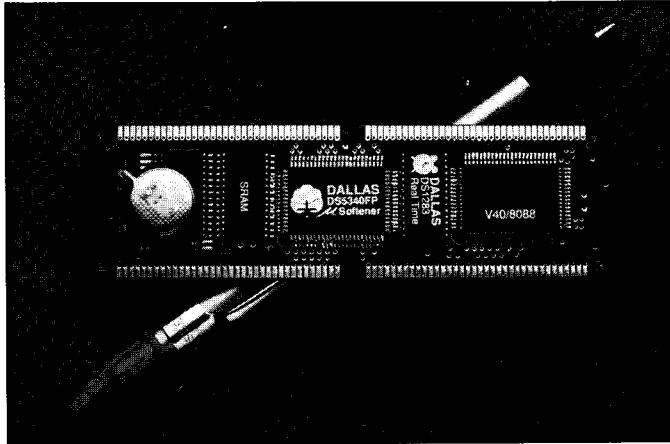
plugged in the other. It measures only 4.25" by 1.25" and conserves valuable PCB space. By using CMOS and lithium technologies, the board typically consumes only about 100 mA.

The Flip Stik provides many functions required in embedded applications. The V40 processor provides a serial port, interrupt controller, timer-counters, and a DMA controller.

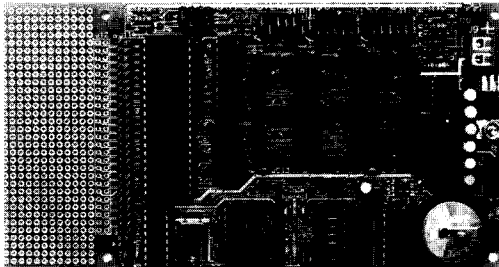
The DS2340 Flip Stik sells for \$54.30 in 1000-piece quantities.

Dallas Semiconductor
4401 South Bellwood Pkwy.
Dallas, TX 75244-3219
(214) 450-0448
Fax: (214) 450-0470

Reader Service #506



EMBEDDED CONTROLLER for Quick Development



The EC-32™ is a versatile 80C32 microcontroller board. It is ideal for quickly developing products, prototypes or test fixtures.

- 80C32 microcontroller (8051 compatible)
- BASIC-52 or MONITOR-52 available
- Program in C, BASIC or assembly language
- 8 to 92K RAM, EPROM or EEPROM
- Breadboard area and expansion bus
- RS-232 port and 12 digital I/O lines
- \$100 for 11 MHz, \$145 for 20 MHz

Iota Systems, Inc.
(702) 831-6302
POB 8987
Incline Village, NV 89450

**20 MHz Board
Available**

Reader Service #158

LASERS



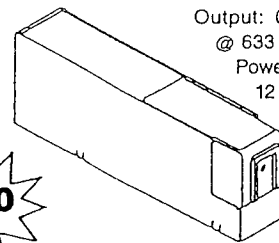
HELIUM NEON LASER MODULE

This Module contains a He-Ne Tube and matching Power Supply, all in one housing! Comes with instructions and a 1 year warranty.

Makes an ideal
Science Fair
Project

Cat. # HNKD-10

Output: 0.5mW continuous
@ 633 nm (Red).
Power Requirements:
12 VDC @ 600mA.



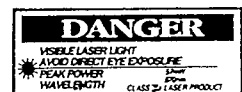
Complies
with C.D.R.H.
regulations

\$59.00

TOLL FREE ORDER # 1-800-722-0392

Phone Hours: Monday - Friday
8:00 A.M. — 6:00 P.M.
Add \$8.00 for S&H. Az Residents add 6.5% tax.

FREE CATALOG
on these and other related items



MEREDITH INSTRUMENTS

P. O. BOX 1724 • 5035 NORTH 55th AVENUE, #5
GLENDALE, AZ 85301
Phone 602-934-9387 FAX 602-934-9482

Reader Service #169

October/November 1991

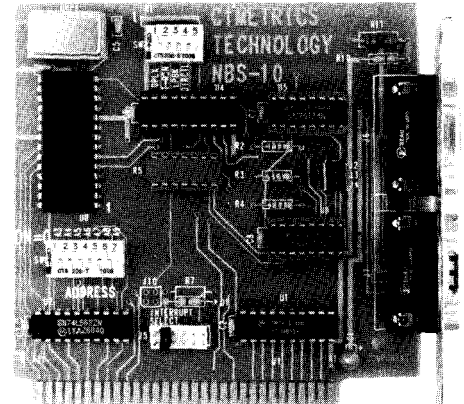
13

9-BIT EMBEDDED CONTROL NETWORKING SYSTEM

An asynchronous 9-bit serial data communications network using the built-in modes in many popular embedded controllers has been announced by Cimetrics Technology. **The system (which consists of the NBS-10 controller and NSP software)** is based on the serial communications modes first used in the Intel 8051 embedded control processor. Industry support of the 9-bit mode has continued to expand and many companies have introduced embedded controllers with this feature. Processors capable of supporting this standard include: Intel 8051 and 8096BH, Motorola's MC68HC05 and MC68HC16Z1, Zilog's 2180,

Hitachi's HD64180 and HD641016, and Texas Instruments' TMS7002/7042. The hardware interface is made possible by the NBS-10 asynchronous serial communication card for the IBM PC/XT/AT or compatible machines. The NBS-10 performs 9-bit asynchronous serial communication in both full- and half-duplex modes, and allows for the construction of networks using either two- or four-wire configurations. It also incorporates several features that aid the designer in networking applications including flexible output switching and bias configurations. The NBS-10 is designed specifically as a master, slave, or development tool for μ -LAN 9-bit

microcontroller networks. The only hardware necessary to transform an embedded controller into a network node is a single RS-485 transceiver and pull-up resistor. A minimum configuration consists of a controller with RS-485 transceiver and no external RAM. Software support is provided by Cimetrics Technology's 9-bit Software Protocol (NSP). This toolkit includes source code to run the network, monitor network activity, and perform data transfers. Complete source code and



documentation are included with the toolkit.

The NBS-10 is priced at \$249 and the 9-bit Software Protocol is priced at \$199.

Cimetrics Technology
120 West State St.
Ithaca, NY 14850
(607) 273-5715
Fax: (607) 273-5712

Reader Service #507

Total control with LMI FORTH™

For Programming Professionals: an expanding family of compatible, high-performance compilers for microcomputers

For Development:
Interactive Forth-83 Interpreter/Compilers for MS-DOS, OS/2, and the 80386

- 16-bit and 32-bit implementations
- Full screen editor and assembler
- Uses standard operating system files
- 500 page manual written in plain English
- Support for graphics, floating point, native code generation

For Applications: Forth-83 Metacompiler

- Unique table-driven multi-pass Forth compiler
- Compiles compact ROMable or disk-based applications
- Excellent error handling
- Produces headerless code, compiles from intermediate states, and performs conditional compilation
- Cross-compiles to 8080, Z-80, 8086, 68000, 6502, 8051, 8096, 1802, 6303, 6809, 68HC11, 34010, V25, RTX-2000
- No license fee or royalty for compiled applications

LMI Laboratory Microsystems Incorporated
Post Office Box 10430, Marina del Rey, CA 90295
Phone Credit Card Orders to: (213) 306-7412
FAX: (213) 301-0761

Reader Service #161

Byte-BOS™ Real-Time Multitasking Operating System

POWERFUL Designed for Embedded Systems, Byte-BOS Real-Time Multitasking Operating System is complete and well tested. With over 80 robust functions, much of the development is done for you.

COMPATIBLE BOS is enhanced and tested with your favorite "C" development systems. Just open the box and start coding your application.

- ✓ Fast Pre-emptive Scheduling
- ✓ Direct Inter-Task Communication
- ✓ Multiple Event Synchronization
- ✓ Timer Services t Auto Sampling
- ✓ Resource Management
- ✓ Asynchronous Communication
- ✓ Fixed Block Memory Management
- ✓ Low Power Management

FAST Written in "C", BOS has an assembly kernel tuned to each supported MCU -- and that includes full on chip serial I/O and timer support.

Intel	8051	80188/86	8096
Hitachi	6301/3	64180	H8XXX
Mitsubishi	37700		
Motorola	6801/3	68HC11	683XX
NEC	All "V" Series		
TI	TMS 320	C2X/C5X	
IBM PC	80X86	DOS Compatible	

PORTABLE BOS code written and tested on the PC transfers directly to any BOS target. Just recompile, link, load, and go.

C Source Code
No Royalty Site License
1 Year Tech Support
Price Only \$1990

Byte-BOS™ 800-788-7288
Integrated Systems, Del Mar CA Outside U.S. 619-755-8836

Reader Service #117

NEWPRODUCTNEWS

INTERACTIVE SCHEMATIC CAPTURE AND DIGITAL SIMULATION SOFTWARE

DesignWorks, from Capilano Computing Systems Ltd., combines the power of Apple Macintosh graphics with a friendly user interface to simplify schematic entry and simulation. **DesignWorks** can write net list, simulation, and graphics files in a variety of standard and user-definable formats to facilitate transfer to other systems. The integrated schematic entry and digital simulation allows detection of design errors before they are wired into hardware.

DesignWorks makes full use of the Macintosh multiwindow environment, allowing any number of circuit files open simultaneously and full Cut/Copy/Paste editing operations between circuits. Multipage schematics are fully supported with page connectors and interactive simulation across pages. Device and signal dragging with fully interactive and orthogonal rubberbanding reduce editing time.

Pull-down menus and an on-screen tool palette provide quick access to all program functions. **Modeless** operation provides immediate access to any program function at any time without moving through hierarchical menus.

Control devices such as switches are active right on the diagram, and can be changed in state to observe their effect on the simulation in progress. Probes and numeric displays can also be placed directly on the schematic to observe signal value changes. Any selected signals can be displayed in the form of a logic-analyzer-style timing diagram. The diagram is updated continuously to reflect design and parameter changes.

DesignWorks has complete symbol drawing and library maintenance capability, allowing the rapid creation of new devices. The "AutoSym" feature will generate a standard rectangular symbol given only a list of input and output pins. For simulation purposes, any circuit can be associated with a symbol to allow the creation of fully functional custom devices. Libraries of these custom devices can be maintained to suit project requirements.

The **DesignWorks** Report module allows full customization of text report formats, eliminating tedious file translation or manual modifications. Report formats include net lists by signal or device, bills of materials, signal lists with simulation event data, and signal and device lists with graphical data.

DesignWorks is compatible with any Macintosh with two megabytes or more of memory. The absolute maximum circuit size is 32,767 devices, although drawing and simulation speed limits circuits to 500 to 5000 devices depending on computer model. **DesignWorks** sells for **\$995.00**.

Capilano Computing Systems Ltd.
1168 Hamilton St., Suite 501
Vancouver, B.C. • Canada **V6B 2S2**
(604) 669-6343 • Fax: **(604) 669-9531**

Reader Service #508

NEW!!
68000, COP800, PIC16Cxx
versions!

µASM™

**Cross Assemblers
for the Macintosh™**

*TEXT EDITOR, CROSS ASSEMBLER, AND COMMUNICATIONS FACILITY IN A COMPLETE INTEGRATED DEVELOPMENT ENVIRONMENT

- MACROS
- CONDITIONAL ASSY
- LOCAUAUTO LABELS
- SYMBOL TABLE CROSS REF
- S OR HEX FILE OUTPUT DOWNLOADS TO MOST EPROM PROGRAMMERS

U.S. **\$149.95**
EACH
PLUS S/H*

AVAILABLE FOR MOST 8-BIT MICROPROCESSORS AND 680001010. CALL OR WRITE FOR TECHNICAL BULLETIN. **30 DAY MONEY BACK GUARANTEE. MC/V/AE.**

* PER SHIPMENT:
\$5 CONTIGUOUS USA
\$10 CANADA AK, HI
\$20 INTERNATIONAL

Micro Dialects, Inc.
DEPT. C, PO BOX 30014
CINCINNATI, OH 45230
(513) 271-9100

Reader Service #173

New! **SUPERPRO™** **UNIVERSAL PROGRAMMER**



- Programs PAL, EPLD, GAL, PEEL, FPL (upto 68 pin PLCC), E(E)PROM, Flash EPROM up to 4 Mbits (40pins), Microcontroller, & Bipolar PROM.
- Tests TTL/CMOS Logic, D/S Memory Device.
- Test Vector verification with screen editor.
- Accepts JEDEC, Intel HEX & Extended, Motorola S1, S2, S3, Tektronix HEX and Binary format.
- High speed parallel interface card into PC/XT/AT/386.
- Pulldown Menu-driven, Library Operating software.
- Fast Device update on user's request.
- **40 pin Gold ZIF** Socket.
- **Lifetime Free Software Updates (BBS).**
- Optional Device Library Generator (SUPERGEN™).
- Includes S/W, Cable, Interface card, and 1 year warranty.

XELTEK

764 San Aleso Ave.
Sunnyvale, CA 94086

Toll Free 1-800-541-1975
Tel: (408) 745-7974
Fax: (408) 745-1401

Reader Service #218

See us at The Embedded Systems Conference Booth #404

October/November 1991

15

FEATURE ARTICLES



page 16

A Video Editing Control System-Part 1



page 26

Computer Graphics and the World of Scientific Visualization



page 40

Add a Video Display to Your 8031 Microcontroller



page 46

ISDN (S/T) Interface—Part 2

A Video Editing Control System

The Hardware

I'm what's known in the video business as a free lance. I produce videos for people who need them—usually industrial customers. Once in a while I get a call from someone who's interested in getting into the video production business. I have to tell them that off-the-shelf consumer camcorders are fine for recording weddings and special events, but if they want to get into more professional productions, the cost of the equipment they need can be overwhelming. Professional editing recorders start at about \$8000 and go up rapidly from there. Even a small system can approach numbers that can make your head spin. But if you can find yourself some used equipment, a soldering iron, and a little imagination, you can set yourself up with an editing system that rivals the ones that broadcasters use.

Before I take you on a tour of my editing controller, let me first fill you in on what it takes to edit video tape. If video editing brings to mind somebody with razor blades and scissors, you're living in the past. We don't even do that with audio tape much anymore. Today, editing is basically a copying process. Picture and/or sound from either a source VCR or a camera is copied onto a destination VCR in the order needed to produce a finished program.

BUILDING A PROGRAM

There are two types of editing: assemble and insert. With assemble editing, you start with a blank tape and build a program by copying segments. You record the first segment,

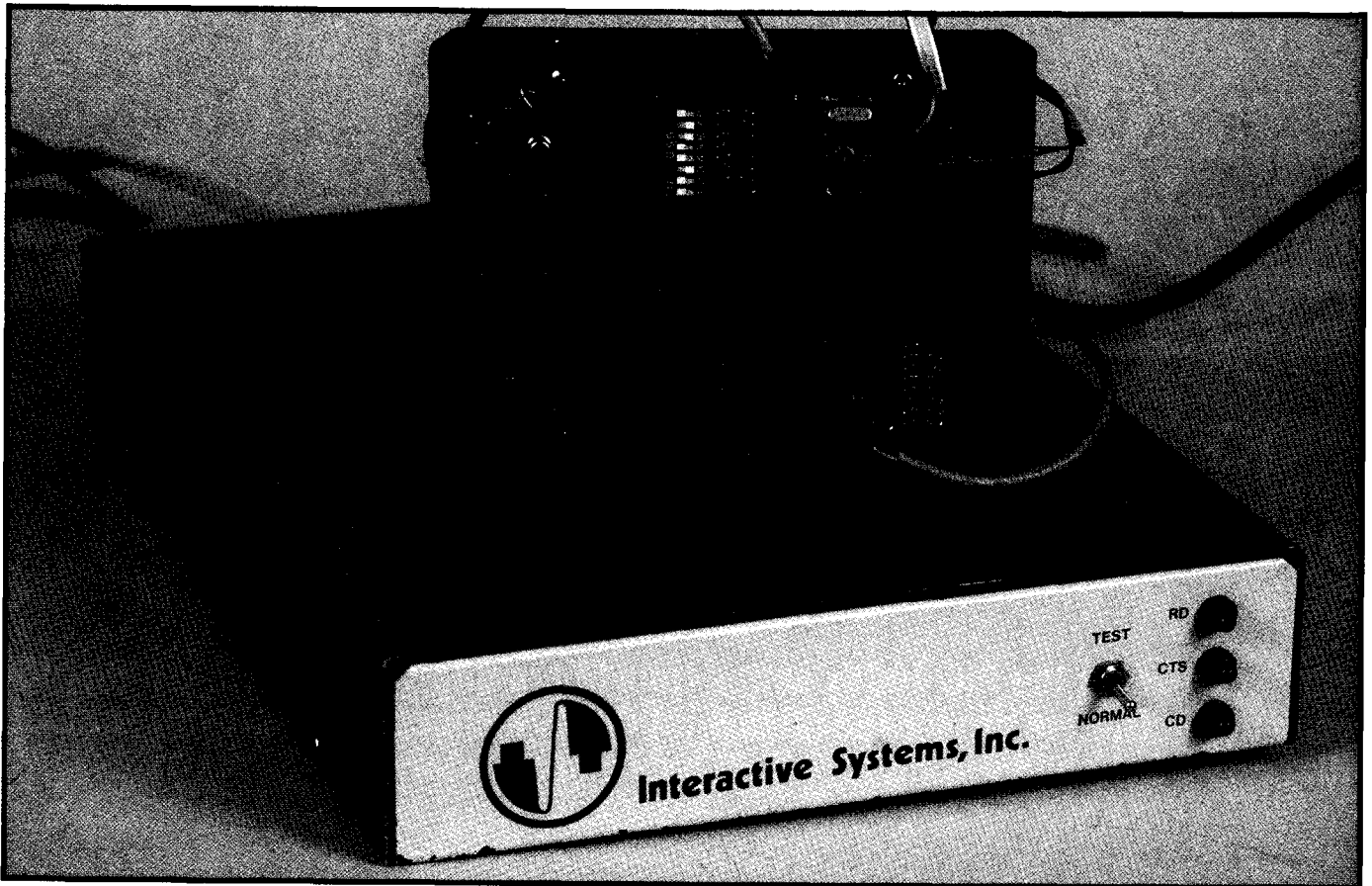
and then add on to it with new segments. You "assemble" by appending additional segments onto the end of each **previous** segment. Because you're assumed to have started with a blank tape, **you** have to record both the video and the audio tracks when doing an edit.

Although assemble editing requires you to add onto the end of a program, you still can go back and change the picture and/or sound by using the second type of edit, called an "insert" edit. When you do an insert edit you have to use a video tape that already has stable video recorded on it. You then insert new segments onto the tape in the required location. Insert editing allows you to choose any combination of audio and/or video. You can "cut-in" and "cut-out" as needed. Most editing is done in the insert mode.

Why can't you just go ahead and use insert editing on a blank tape? Well, the television picture consists of a **series** of lines "painted" on the video screen from left to right and top to bottom. Each frame of video is divided into two fields. One of the fields consists of the odd scan lines and the other the even lines. Each of these fields is completed in 1/60 of a second. This makes for a video frame rate of 30 frames per second. The VCR marks each field by recording a pulse on a separate track called the control (CTL) track. It then uses these pulses as a reference during playback. When an insert edit is made, new picture is recorded but the old control track is left undisturbed. The control track must be continuous or an unstable picture will result.

FEATURE ARTICLE Part 1

William J. Kressbach



If the edit is to be “clean” and glitch free, it must occur in the short time between frames: the vertical interval. The vertical interval is the time when the TV picture has completed one frame but isn’t quite ready to start the next. This means that the vertical interval of the incoming signal must be in the same place at the same time as the signal coming from the tape. Fortunately, the VCR doing the editing is smart enough to do this for you. It adjusts the speed of the drive motor until the vertical sync coming from the tape is synchronous

with the vertical sync from the incoming signal.

It’s easy to see that both the source and destination video must be stable before a glitch-free edit is possible. While the record deck is responsible for ensuring that the “cut in” and the “cut out” edits are clean, it’s still the user’s responsibility to see that the tape is up to speed and in the right place for the edit.

While limited editing can be done manually, the precision necessary to get good edits, every time, where you want them, requires careful control.

EXAMPLE EDIT

Let’s look at an example. Take the simple shot sequence given below of a hammer striking a nail.

1. ms (medium shot) hammer poised to strike nail-hammer swings
2. cu (close up) nail as hammer strikes
3. ms hammer raises

To keep it simple, let’s assume that the source of the picture is a cam-

era. The scene calls for a medium shot of a hammer poised to strike a nail, takes to a closeup as the nail is struck, and then back to a medium shot of the hammer going back up again.

While the scene calls for three shots, it's probably best accomplished in two. It's easier to take one complete shot of the hammer striking the nail and then going back up. After that shot has **been** completed, we'll go back and insert the close-up of the hammer striking the nail. The source is a camera so the first shot is easy to do, but inserting the close-up can be tricky. Hold your finger on the edit button and put the tape in play. When the tape gets to the right place press the edit button, hit the nail, and then press the cut-out button. If you're lucky, you'll get it on the first take. But, suppose you miss the nail or, worse yet, hit your thumb. This means that you'll have to go back and do it again (if you hit your thumb, you'll probably have some audio to fix as well). When you're doing edits manually, it's impossible to hit the edit at exactly the same location as the previous one. The only thing to do is try to start the edit just a little before the last one and end just a little after. It doesn't take very many retakes for the small insert to grow enough in size so that the whole scene will have to be redone. Too much of this and your head will throb as much as your thumb.

HITTING A MOVING TARGET

If the source video is tape, it becomes much more difficult. Now not only do we have two tapes to get up to speed, but we also have to see to it that each picture is in the right place at the right time for the edit. The only way to do this is to rewind both the source and destination machines to a point several seconds before the edit point. This is called a preroll. As each machine reaches this location it's put into pause. When both machines have reached this pause point, they both are started at exactly the same time. Then, if you've done everything right, pressing the edit button at exactly the right time will give us a good edit. Years ago when I first started editing

on open-reel recorders, I would often use a yardstick to measure out the length of tape for the preroll.

Other than avoiding hammers, what can we do to help this situation? Well, to accomplish editing with any kind of precision requires some sort of computerized controller. A number of inexpensive controllers exist that use the pulses on the control track of the VCR as a reference. While this works reasonably well, marking the edit locations is pretty much **hit-and-miss**. Also, shuttling the tape back and forth causes the CTL head to miss pulses. It doesn't take very long for the reading to be off by several frames. This type of controller works well if you don't make any mistakes and can hit the nail on the head every edit.

HITTING THE NAIL ON THE HEAD

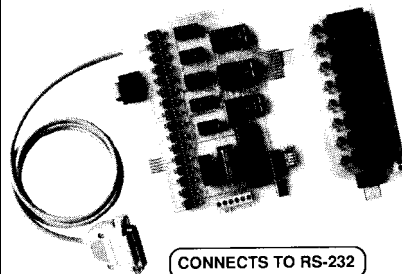
A better method is to design a controller that is locked in some way to the frame rate of the video tape. We can do this by assigning each frame of video a unique number and then recording that number onto one of the audio tracks of the tape.

Let's see how this is done. The CTL signal is available on the remote control jack on the rear panel of the VCR. Dividing the CTL pulse by two (remember, one full frame of video for every two fields) serves as a reference for the assignment of time code. Frames of video are assigned time codes sequentially from the beginning of the tape to the end, with the CTL pulse marking the start of each frame. A time code consists of hours, minutes, seconds, and frames, and looks something like "00:00:00-00." As each frame of video passes by, its time code is derived, converted to an analog signal, then recorded on a spare audio track (note that sometimes time code is encoded into the vertical interval). This is called "time coding" or "stripping a tape." Once a tape is time coded, it becomes a simple matter for a controller to read it back. This gives you the ability to locate any frame on a video tape with precision.

A lot of commercial editing controllers are available, but like most of the equipment in this business, the

RELAY INTERFACE

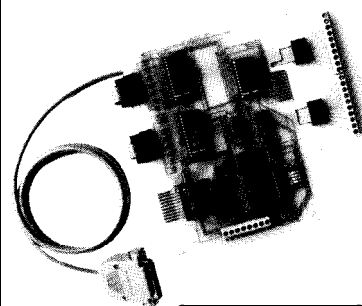
PROVIDES SOFTWARE CONTROL OF RELAYS



CONNECTS TO RS-232

AR-16 RELAY INTERFACE \$ 89.95
Two 8 channel relay output ports are provided for control of up to 16 relays (expandable to 128 relays using EX-16 expansion cards). Each relay output port connects to a relay card or terminal block. A variety of relays cards and relays are stocked, call for more info. RS-422 available (distances to 4,000 feet). PS-8 port selector may be used to control satellite AR-16 interfaces. (up to 16,384 relays)
RD-8 REED RELAY CARD (8 relays) \$ 49.95
RM-8 RELAY CARD (10 amp SPDT 277 VAC) \$ 69.95
EX-16 EXPANSION CARD (16 channel) \$ 59.95

ANALOG TO DIGITAL



CONNECTS TO RS-232

ADC-16 (16 channel) \$ 99.95
Input temperature, voltage, amperage, pressure, energy usage, energy demand, light levels, joystick movement and a wide variety of other types of analog signals. Inputs may be expanded to 32 analog or 126 status inputs using the AD-16 or ST-32 expansion cards, 112 relays may be controlled using EX-16 expansion cards. Analog inputs may be configured for temperature input using the TE-6 temperature input card. RS-422 available PS-8 port selector may be used to connect satellite ADC-16 Interfaces (up to 4,096 analog inputs/16,384 status inputs/14,336 relays, use RS-232 for satellites up to 50 feet or RS-422 for satellites up to 4,000 feet). (terminal block and cable sold separately)
ST-32 STATUS EXPANSION CARD ... \$ 79.95
Input on/off status of relays, switches, HVAC equipment, thermostats, security devices, smoke detectors and other devices. The ST-32 provides 32 status inputs. (opto isolators sold separately)
TE-8 TEMPERATURE INPUT CARD \$ 49.95
Includes 8 solid state temperature sensors. Temperature range is minus 78 to 145 degrees F

- **FULL TECHNICAL SUPPORT**...provided over the telephone by our staff. EACH ORDER INCLUDES A FREE DISK WITH PROGRAMMING EXAMPLES IN BASIC, C AND ASSEMBLY LANGUAGE. A detailed technical reference manual IS also included.
- **HIGH RELIABILITY**...engineered for continuous 24 hour industrial applications. All IC's socketed.
- Use with IBM and compatibles, Tandy, Apple and most other computers with RS-232 or RS-422 ports. All standard baud rates and protocols may be used (50 to 19,200 baud).
- Use our 800 number to order free information packet. TechnicalInformation (614) 464-4470.

24 HOUR ORDER LINE (800) 842-7714
Visa-Mastercard-American Express-COD

ELECTRONIC ENERGY CONTROL, INC.
360 South Fifth Street, Suite 604
Columbus, Ohio 43215

Reader Service # 141

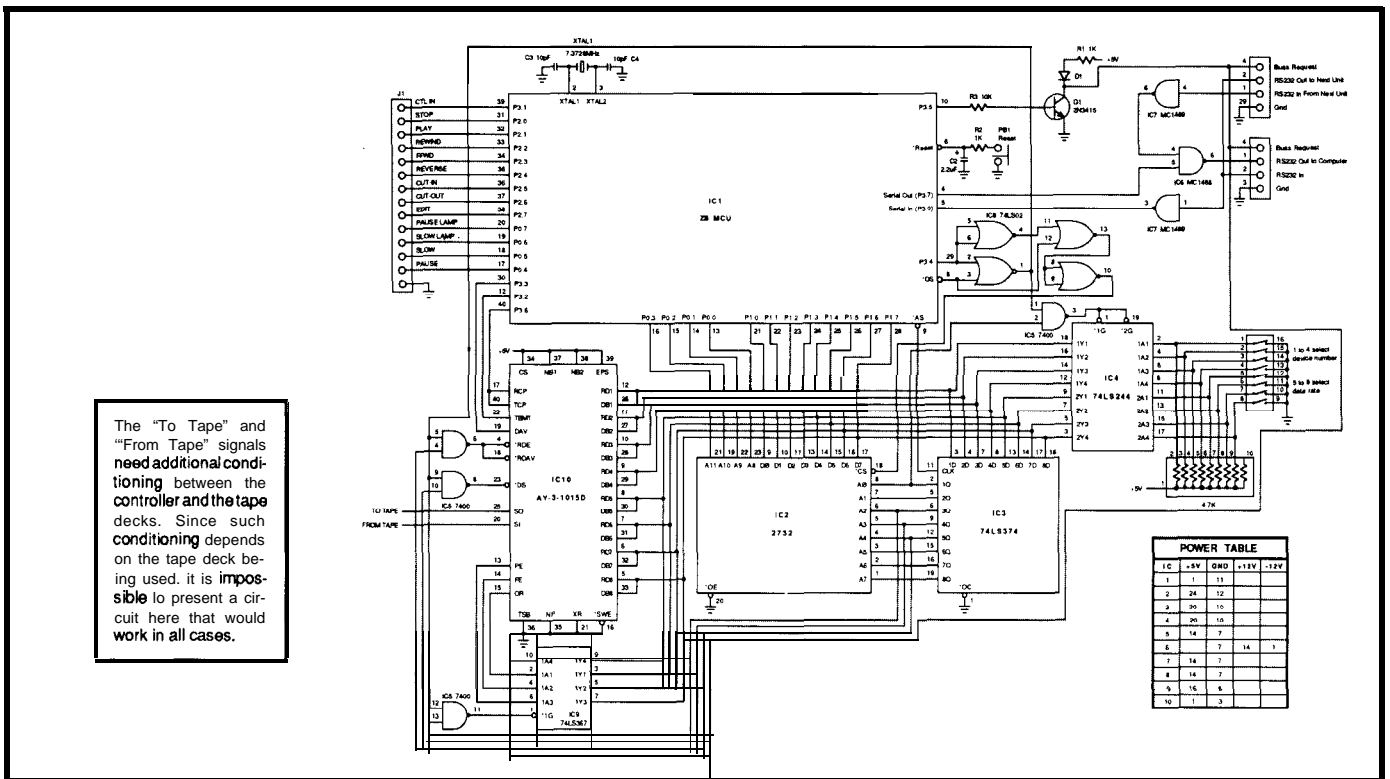


Figure 1—The video editor controller is based on modules that use the venerable Zilog Z8 microcontroller.

cost is prohibitive. For that reason, and because it was kind of fun, I decided to build my own controller.

My early attempts to build a controller using discrete components and a Radio Shack Model III computer met with some success but it had limited portability and required a lot of processor time. Often, a VCR would go into a fast rewind and the computer, reading the control track on an interrupt line, would ignore me until I could stop the VCR manually.

I finally solved the problem of processor time by using a dedicated controller for each VCR. I was already on my third computer and was contemplating my fourth. So, portability also being a factor, communication through a serial port seemed the way to go. The Zilog Z8 controller answered most of those needs. While it only had one serial port, it was available, inexpensive, and I could afford the cross-assembler I needed to program it.

NAILING DOWN A DESIGN

Figure 1 shows the basic design for the controller. It's pretty much standard with one exception: I used a 7402 for address decoding. While this is a bit unconventional, it allowed me to

utilize the extended memory feature of the Z8 giving me extra input ports without adding extra decoding logic. I added a UART for a second serial port, and, of course, I used the machine language version of the Z8.

The time code is recorded at 2400 bps. It begins with a marker byte (A5h), and ends with a checksum. While it's not compatible with industry standard time code, as long as you don't need to use your tapes on a professional reader, it's not a problem.

The controller does most of the leg work. It time codes the tape and then reads it back to you. It keeps track of the current tape position, what mode the VCR is in (rewind, fast forward, etc.), and conveys user commands to the VCR. It can search for and find any location on a tape and then either play the tape or wait for a user command. It can preview an edit; that is, simulate an edit so that the user can see what it will look like. It will then perform the edit and review it for you.

There is one controller module for each of the two editing decks. They are daisy-chained through the serial ports. Each module is addressed separately and listens in on a party line for messages designated for it. The modules share a common line back to the

host as well. There are four lines for communication: three for the serial I/O, and one for bus arbitration. A controller wishing to use the bus pulls the transistor low, signaling the other modules that the bus is in use (more about this later). The control modules are designed to have the capacity of allowing up to 14 modules on one serial port, but I've never tried more than two. I did this to allow for possible use in automated systems where several VCRs might be necessary. All communication is from module to host or from host to module. The modules cannot talk to each other.

Each controller is locked to and is in complete control of the deck to which it is attached. The parallel ports of the Z8 are able to read the status and remotely control all necessary functions of the VCR such as play, rewind, fast forward, cut-in, cut-out, and so on. In general, the same port on the Z8 serves not only to trigger a VCR mode, but to read VCR status as well. For instance, configuring port 2 pin 1 for input reads the status of the VCR's play LED. Configuring the same pin for output and then strobing it low puts the VCR into the play mode. Because of this, the controller is able to follow the VCR status even when op-

erated manually. Figure 2 shows a list of the **commands** to which the module will respond.

The control track pulses are used to follow the tape location when the tape is in rewind, fast forward, or is reading a tape that has not been time coded. When the VCR is in the play mode and the module is reading valid time code from the tape, the software sets a flag telling the counter to ignore the control pulses from the VCR. If, however, the module doesn't get a valid read after three pulses, the module assumes it has lost lock, the flag is

reset, and incrementing continues from the control track.

As it's reading, the module also transmits the current tape location to the host computer. The module first sends an ID code (*nC* hex, where *n* = module number) so that the host will know what is coming and where it is coming from. See Figure 3 for a listing of module response codes. The module then starts with frames and progresses through seconds, minutes, and hours. It ends the transmission with a terminator (**0Dh**). In order to minimize the amount of information

sent, the module will only transmit information that has changed since the last update. Most transmissions are short and consist only of frames.

Serial communication from the module takes place through a ring buffer. It has an "in" and an "out" pointer. The software increments the in pointer as it moves information in. As the information is transmitted, it increments the out pointer. Transmission continues until the in and the out pointers are the same value. The small memory capacity of the **Z8** requires that the buffer size be kept to a minimum. Usually there is plenty of buffer space but, when a VCR is in rewind or fast forward, so much information is being transmitted that the in pointer of the buffer can actually overtake and pass the out pointer and overwrite the outgoing data. While with some systems this might be a problem, in this case so much information is being sent that it's OK to skip some of it when you're in a hurry. After all, it's not really too important for the host computer to know exactly where the VCR is, as long as the module knows.

I had to design a rather complex bus arbitration scheme for two reasons. First, to prevent more than one module from grabbing the bus at one time and second, to prevent a module from capturing the bus and not allowing another to transmit. I wasn't surprised that a module tried to monopolize the bus and that was easy to solve. The module just has to wait several clock cycles before it's allowed to transmit again. But I was amazed at how often two modules would grab the bus at precisely the same time and not know that the other was there. Listing 1 shows the **Z8** machine code for the serial I/O. It first checks to see if the bus is available and, if so, grabs the bus by setting pin 5 of port 3 high. It then waits an assigned number of clock cycles which is based on its module number. After it's waited the required time, it drops the bus line and checks to see if the bus is still available. If another module of a higher number had grabbed the bus at the same time, the bus will show occupied. If the bus is still available, the controller will grab it again and send the data.

<p>MODE 1 COMMANDS</p> <p><i>n0</i>=Stop <i>n1</i>=Play <i>n2</i>=Rew <i>n3</i>=FFwd <i>n4</i>=Toggle Reverse <i>n5</i>=Cut-In (Edit & Cut-In) <i>n6</i>=Cut-Out <i>n7</i>=Set Mode 2</p> <p>MODE 2 COMMANDS</p> <p>18 = Clear Rec, Cut-In, cut-out 19 = Pause 1A = Slow 1 B = Strobe Record 1 C = Hold Record 1 D = Hold Cut-In 1 E = Record/Play</p> <p>1 F = Xmit Hex (BCD) 28 = Xmit ASCII 29 = Xmit Taoe Time 2A = Xmit Tape Time and File Name 2B = Xmit File Name 2C = File Name On 2D = File Name Off 2E = Tape Time and File Name On 2F = Tape Time and File Name Off</p> <p>38 = (Search) Find & Stop 39 = (Search) Find & Play 3A = (Edit) Find, Play, Pause, Postroll 3B = (Edit) Find, Play, Slow, Pause, Postroll 3C = (Edit) Find, Play, Postroll 3D = Cancel Find 3E = Index On 3F = Index Off</p> <p>48 = Insert Master 49 = Insert Preview</p>	<p>4A = Insert Edit 4B = Insert Review 4C = Assemble Master 4D = Assemble Preview 4E = Assemble Edit 4F = Assemble Review</p> <p>58 = Set Baud 150 59 = Set Baud 300 5A = Set Baud 600 5B = Set Baud 1200 5C = Set Baud 2400 5D = Set Baud 4800 5E = Set Baud 9600 5F = Set Baud 19200</p> <p>68 = Device = 0 69 = Device = 1 6A = Device = 2 6B = Device = 3 6C = Device = 4 6D = Device = 5 6E = Device = 6 6F = Device = 7</p> <p>78 = Xmit Frams Off 79 = Xmit Sec Off 7A = Xmit Min Off 7B = Xmit Hours Off 7C = Xmit Frams On 7D = Xmit Sec On 7E = Xmit Min On 7F = Xmit Hours On</p> <p>88 = stop 89 = Play 8A = Rew 8B = FFwd 8C = Hold Record 8D = Cut-In 8E = cut-out 8F = Set Mode 2</p> <p>E8 = Device = 8 E9 = Device = 9 EA = Device = 10 EB = Device = 11 EC = Device = 12 ED = Device = 13 EE = Device = 14 EF = Device = 15</p>	<p>FF = Mode 3 Off On=Mode 3 On</p> <p>When any module is programmed for Mode 3, all other modules are locked out and will not respond to any commands other than a Mode 3 "On" command. This is so that any desired character may be sent to a module without the module accidentally responding to an undesired command. Transmitting a Mode 3 "Reset command (FFh) will return all modules to normal.</p> <p>MODE 3 COMMANDS</p> <p>41 =SetTime 42 = Set Cut-In 43 = Set Cut-Out 44 = Set File Name (Nulls = 20h—space) 45 = Set Edit Parameters: N+0 = Postroll (1 sec) N+1= Preroll Low Limit (10 sec) N+2= Preroll(5 sec)</p> <p>For Set Time, Set Cut-In, and Set Cut-Out: Transmit frames first, followed by seconds, minutes, and hours. Module will receive until FFh received or carriage return (0Dh). If carriage return received, module will fill remainder with nulls (00h).</p> <p>For File Name: Transmit file name up to 12 characters long. Module will receive until FFh or carriage return. If carriage return received, module will fill remainder with spaces.</p>
--	--	--

Figure 2—The list of supported commands is extensive and covers just about any editing task necessary.

I had to allow for three modes of operation. In mode one, the default mode, the **module responds** only to commands addressed to it and ignores those that are not. If the module is put into mode two, it will respond to a second set of global commands. This permits several controllers to respond to the same command simultaneously. Mode three (On hex) does just the opposite. All modules with the exception of the one to which the command is addressed are locked out. This permits sending from the host edit and other information without inadvertently triggering another controller.

HOW IT ALL FITS TOGETHER

Once **you** put it all together, here's how it works. After the user has selected the edit points and is ready for the edit, the host computer first transmits the cut-in and cut-out locations along with other pertinent information to the modules. It then tells each module what task it's to perform (the source module's task is different from the destination module). Then the host instructs the modules to go ahead and do the task. As I said before, editing consists of a preroll, edit, and then a postroll. In basic terms, each VCR is instructed to go to a point exactly five seconds before the edit point, go into the pause mode, and tell the host that it is there. When both modules have signaled that they are ready, the host computer puts them both into play **simultaneously**. When the cut-in point is reached, the destination module puts the VCR into insert mode. When the cut-out location is reached, the module releases insert mode. Finally, the controller does a one-second **postroll** and puts the VCR into stop mode.

While all this may seem pretty straightforward, actually doing it is harder than you think. To get to the first pause point, not only **do you** have to recognize where you are now and where you have to go, you also have to find the best way to get there (play or fast forward, reverse or rewind). You also have to allow the module a little time to establish a valid read from the tape before it reaches the pause location. As the VCR's tape handlers age,

```

n0 = stop
n1 = Play
n2 = Rew
n3 = FFwd
n4 =
n5 = Cut-In (Edit)
n6 = Record (Edit)
n7 = Reverse On
n8 = Reverse Off
n9 = Pause Toggled
nA = Slow Togg'd
nB = Cut-Out (Strobe)
nC = Transmitting Tape Time—
      Transmission starts with frames
      OD—end of transmission
nD = Cut-In (Strobe)
nE = Record (Strobe)
nF = Clear (Clears any issued
      commands)

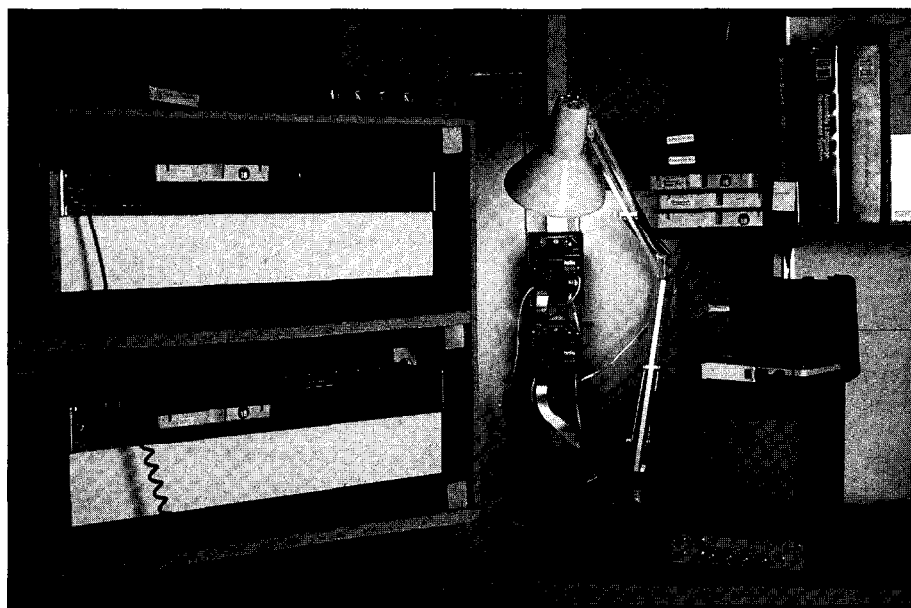
```

Figure J—Response codes close the loop between the host and the remote module. The "n" in each code refers to the target module number.

they get a little sloppy and the tape can sometimes skip over the head that reads the control track. When fast forwarding or rewinding, it's easy to be a few seconds off, especially if a valid read hasn't occurred recently. Another problem the module has to contend with is if the VCR is put into play at a location that the controller thinks is before the pause point, and the first valid read shows that it's actually after the pause point, the module has to recognize that it has missed, and go back and try again.

Finding the best way to get to the pause point is a matter of establishing a window for the controller to shoot for. Figure 4 shows that the top of the window is just above the pause point. If the VCR has to rewind to get to the window, the VCR's momentum will carry it far enough past the pause point for a valid read. On the other hand, if the VCR has to fast forward, the window's low limit has to be low enough to allow for tape momentum plus time for a valid read. The software allows for changing the limits to compensate for differences in VCRs. To further complicate things, if the VCR is in the play mode when the command is received, a second window has to be established. It takes several seconds for the VCR to cycle from play to stop to fast forward, then again from fast forward to stop and back to play. If the module is only a few seconds behind the target point, the module has to determine whether it is quicker to fast forward, or just continue on in the play mode. This also applies if the VCR is beyond the pause point. Is it faster to go into reverse, or stop and rewind?

Once both VCRs have reached the pause point, the host computer allows a few seconds for settling and then



The author's video editing setup. Note the two controller modules mounted on the wall just behind and to the right of the two editing decks. They are connected to the host computer located nearby with ordinary four-conductor telephone wire. The editing decks are mounted on a heavy-duty lazy Susan and may be rotated for easy access to the connections in the rear. The monitor on the right is just for picture balance and has no other function.

```

4580 ;SERIAL OUT SUB
4585 ,
4590 RSOUT PUSH R5
4595     PUSH R6
4600     PUSH R7
4605     TM XMIT, #0E0H ;CK IF JUST ACCESSED
4610     JR NZ,RS11T ;DISALLOW ACCESS UNTIL TIMEOUT
4615     CALL ERRCH ;READ ERROR CHANNEL
4620     AND R5,#10H ;SEE IF BUS AVAILABLE
4625     JR Z,RS1UT ;GO IF NOT
4630     LD R3,#20H ;GRAB BUS
4635     LD R5,6BH ;READ DEVICE #
4640     RCF ;RESET CY
4645     RRC R5 ;ROTATE
4650     DJNZ R5,$ ;WAIT DEVICE # CYCLES
4655     CLR R3 ;RELEASE BUS
4660     LDE R5,@RR6 ;SEE IF STILL AVAILABLE
4665     TM R5,#10H
4670     JR Z,RS1UT ;GO IF NOT
4675     LD R3,#20H ;GRAB BUS
4680 RS01T CP 6FH,6EH ;CK IF DONE
4685     JR NZ,RS0U1
4690     AND 251,#6FH ;RESET INTERRUPT MASK
4695     CLR R3 ;RELEASE BUS
4700     OR XMIT,#0E0H ;SET COUNTER
4705     JR RS0UD ;DONE
4710 RSOU1 LD 240,@6FH ;XMIT BYTE
4715     INC 6FH ;BUMP POINTER
4720     CP 6FH,#60H ;CK IF AT TOP
4725     JR NZ,RS0UD
4730     LD 6FH,#48H ;START OVER
4735 RS0UD POP R7
4740     POP R6
4745     POP R5
4750     IRET
4755 ;
4760 RS1UT TM 3,#20H ;SEE IF BUS FOR THIS MODULE
4765     JR NZ,RS01T
4770 RS11T OR FLAG2,#01H ;SET BUS REQUEST
4775     AND 250,#0EFH ;RESET INTERRUPT REQUEST
4780     JR RS0UD ;TRY LATER
4785 ;
4790 ;$SETRS MOVES DATA INTO RING BUFFER
4792 ;SETS DATA AND INTERRUPT MODE FOR RSOUT
4795 ;PASS DATA IN REGISTER 05H
4800 ;
4805 SETRS LD @6EH,R5 ;MOVE DATA
4810     INC 6EH ;BUMP POINTER
4815     CP 6EH,#60H ;CK FOR TOP
4820     JR NZ,SETR1
4825     LD 6EH,#48H ;RESET
4830 SETR1 TM 251,#10H
4835     JR NZ,SETR2
4840     OR 251,#10H ;SET INTERRUPT MASK
4845     OR 250,#10H ;SET INTERRUPT REQUEST
4850 SETR2 RET
4855 ;

5700 ERRCH CLR R6 ;READ ERROR CHANNEL
5705     LD R7,#02H
5710     LDE R5,@RR6
5715     RET

```

listing 1 — In order to prevent contention, an arbitration scheme is used to seize the bus.

puts both modules in mode 2. The host then releases the pause on both machines simultaneously by sending a 19h code. From here on the modules are on their own. Both VCRs are free running and no attempt is made to see that the timecodes from the two VCRs stay in sync. The built-in ability of the record VCR to lock to the incoming vertical sync does a pretty good job of keeping the two together.

After the preroll, there are three options that the controller might be

called on to do: an insert edit, a preview, or a review. If the user is calling for an edit, when the record VCR reaches the cut-in point, the module pulls the edit pin low (P27), strobes the cut-in pin (P25), and releases the edit pin. When the VCR reaches the cut-out location the controller strobes the cut-out pin (P26).

If the user has called for a preview, the controller simply holds its electronic finger on the cut-in button (holds I²5 low) between the cut-in



Develop Image-based applications with Victor Image Processing

York with any size images

low your applications can support 8-bit color and gray scale images of any size because Victor gives you complete control over conventional, expanded, and extended memory.

Display on Super VGA

Display images on EGA/VGA and super VGA up to 1024 x 768 256 colors.

Load & save PCX/TIFF/GIF

Handle images from any source, or create translation programs between the popular file formats.

Gray scale & color images

Powerful image processing for all images - our software can have features like: zoom, resize, brighten, contrast, sharpen, outline, nearize, matrix conv, colorize, & more.

ScanJet, LaserJet support!

You can have device control for gray scale scanning -- AND print halftones at any size.

Victor supports Microsoft C, QuickC, and TurboC, includes demonstration and prototyping software, and full documentation. Source code available. Victor library, version 2, \$195.

Video Frame Grabbers

Victor Image Processing Library is also available with 256-level gray scale video frame grabbers. Victor supports these digitizers with capture, live video on VGA, and frame averaging.

VICTOR LIBRARY, v 2 \$195
with 256x256 frame grabber \$399
with 512x512 frame grabber .. \$499

Call (314) 962-7833 to order

VISA/MC/COD
CATENARY SYSTEMS
470 BELLEVIEW
ST LOUIS MO 63119
(314) 962-7833

Reader Service #120

The Equipment

The VCRs used in this article are Sony 2860As. I have also used Sony 2850s and tested Sony 5850s with good results. VCRs from other manufacturers should work as long as they are designed to work in an editing system and are equipped with a parallel port. You should be aware there is no compatibility between manufacturers and you'll have to check the manuals carefully for pin assignments and voltage requirements before attempting any interface. Many of the newer VCRs use serial interfaces and may have quite different requirements.

Many of you may want to try using VHS, S-VHS, 8mm, or other small format VCRs. The crucial factor in a successful editing interface is the ability to access the control track pulses. The pulses must be available not only during playback, but also while the tape is in fast forward and rewind. The key to recognizing a suitable VCR is to first check to see if it uses a footage counter or a tape time readout. In the past most consumer VCRs used mechanical footage counters for displaying elapsed tape time. This method is much too inaccurate for editing purposes. But I have noticed lately that most of the higher end consumer VCR and nearly all industrial VCRs now display tape time in minutes and seconds. They seem to maintain good accuracy in both rewind and fast forward and it seems likely that the control track is being used for tape time display. If the manufacturer is using the control track to display elapsed time, it seems likely that the resourceful engineer should be able to find a way to use it as well.

There is a way to determine if a machine is using the control track. First get yourself a blank tape. It must be a tape that has never been recorded on. Record about thirty seconds of video on it, load it into the machine in

question, and play it. The time readout should count while the recorded part of the tape is playing, and then should stop when the blank part of the tape is encountered. Fast-forward the tape a little. The counter should not move. Now, rewind the tape. The counter should remain unchanged until it encounters the part of the tape that has been recorded. Again put the tape in fast-forward. The counter should count until it reaches the blank part of the tape, and then stop counting. A machine that reacts in this manner indicates that it is probably reading from the control track. This makes it a good candidate for computer control. A blank tape has no control track, so if the readout continues to count while blank tape is passing over the heads, then you can assume that the counter is reading something other than the control track and the machine is probably unsuitable for conversion.

Be careful! Each VCR has its own little quirks. For instance, I have a VCR that will not go into record mode from play mode; it has to be in pause first. Be sure that you thoroughly understand the operating characteristics of a VCR to be certain it will fit into an editing system.

Easy remote control of VCRs with infrared remotes could be achieved through the use of an infrared controller similar to the one described by Steve in volume 6 of his "Ciarcia's Circuit Cellar" books.

There are also companies that make computer interface devices for infrared VCRs. You send them your VCR and the company returns it to you with an interface attached directly to the VCR's infrared sensor. The interface is preprogrammed, ready to plug in to your computer.

and the cut-out locations. This makes it look like an edit was performed without actually doing it.

If a review has been called for, the controller just lets the tape play.

Finally, after a short postroll to let the viewer see how well the edit fit, the controller stops the machine and then transmits the exact edit locations to the host computer.

The preroll time, postroll time, and the bottom of the search window are all user selectable. I should say that the preroll time changes to fit certain requirements and VCRs. The preroll time for an edit is five seconds. This allows plenty of time for stable pictures before the edit. That is not quite so important for a preview, and so to save time, the preroll for a preview is only two seconds.

There are three different editing patterns that can be selected. Code



The host computer and monitor are reconnected to the editing decks located nearby. While many operators prefer to use two monitors, I normally use just one. Video is switched between the two VCR sources automatically. The editing station is located on a desk which is backed by a built-in 130-gallon salt wafer aquarium. This helps to make the sometimes stressful task of video editing a bit more relaxing.

3Ah selects the “find, play, pause, edit, and postroll” sequence just described. A second similar pattern, 3Bh, adds a slow mode just before the pause. This is the one I use most since it pauses the tape a little more accurately. Finally, 3Ch has no pause at all. No need to pause if the source is a camera.

You can also have the module find any location on the tape. One version of this (39h) searches and goes

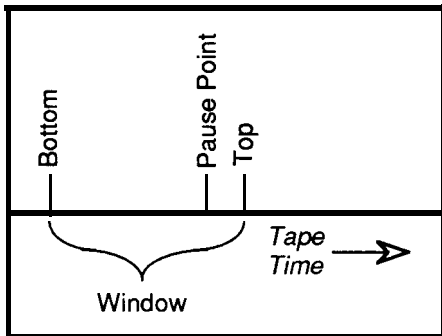


Figure 4-Finding the best way to get to the pause point is to establish a window for the controller to shoot for.

into the play mode when the requested location is found. The other (38h) just stops without going into play. This is in case you're looking for something at the other end of a long tape and want to refill your Coke while it's getting there.

Until I started getting this article ready, I had no idea that the controller had become so complex. I developed it over the course of the nine years that I've been free-lancing. I would usually tinker with it when I had work to do but didn't feel like doing it. That must have happened a lot over the years.

That covers the construction and software for the control modules. In part two, I'll look at software for the host computer and discuss some options that help simplify the entry and retrieval of editing information. ❖

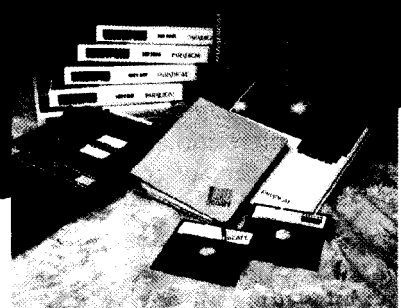
Bill Kressbach holds a Master's degree in Instructional Media and Technology from The University of Toledo. When he's not shooting videos, he does some computer programming and is chief engineer for a college radio station.

IRS

- 401 Very Useful
- 402 Moderately Useful
- 403 Not Useful

REALIZE THE POWER OF PARADIGM

Productivity
 the most popular Intel
 Cereo and NCR
 27 for education
 1987-1990 NCR



PARADIGM

The Model for Programming Productivity

3301 Country Club Road, Suite 2214 • Endwell, NY 13760 • (607) 746-5966 • FAX: (607) 748-5968

FEATURE ARTICLE

Chris Garcia

Computer Graphics and the World of Scientific Visualization

As a result of the advancements in data generation and computer technology over the last few decades, methods of managing and analyzing large volumes of complex data have been largely responsible for a strong interest in scientific visualization as a new computational technology. This visualization technology has its origins within the realms of applied science and engineering where the use of computer display techniques in computer-aided design and engineering analysis have provided an initial driving force for the development of "computer graphics technologies" during the early 1970s and 1980s. Since then the problem of interpreting, understanding, and processing large complex data sets has grown considerably. This need has created a strong desire for effective "realistic" graphics tools which have in turn forced these original display techniques to expand and grow within a variety of computational environments. As such, a whole new set of "visualization" methods has evolved based on the use of computer simulations and computer graphics.

Originally, entertainment applications fueled the fires of computer graphics in the middle 1980s while basic engineering applications became standardized and moved out of the research context into routine practice. And, the simultaneous development of PC technology designed to support advanced graphics display capabilities served as a foundation and principal support to this evolutionary growth process. As a result, the basic visual vocabulary of computer graphics became extended in both breadth

and depth. Graphics applications became more adept at meeting the requirements of applications needing realism, visual richness, and motion (animation). Scientific visualization was born. Now it seems to be the new "catch word" and it has become a major focus of computer graphics in the 1990s.

But this emphasis on visualization technology is an obvious step for the computational world. As numerical simulation continues to become a more accepted (and cost effective) tool for basic scientific research, the need of the scientist to understand and "visualize" the complex nature of their numerical simulations only increases. Fortunately this process has been enriched by advances within the entertainment world. But regardless of the source of advancement, computer graphics technology has grown to become an essential component of this new field of "computational science." As a result, the current rapid development in visualization technology within the sciences and engineering has been only impeded by the growth in the power and sophistication of computer graphics hardware and software systems.

In general, "visualization" can be defined as the use of computer graphics imaging technology as a tool for comprehending data obtained by simulation, computation, or physical measurement. As such, it is built on the integration of techniques "snatched" from older technologies including computer graphics, image processing, computer vision, computer-aided design, geometric modeling, approximation theory, perceptual psychology, and user interface studies. But the most fundamental definition of the visualization technique lies not in its components but rather in its intent. It is important to remember that the purpose of scientific visualization is to gain "insight" into the behavior of some complex process under scrutiny. Individual

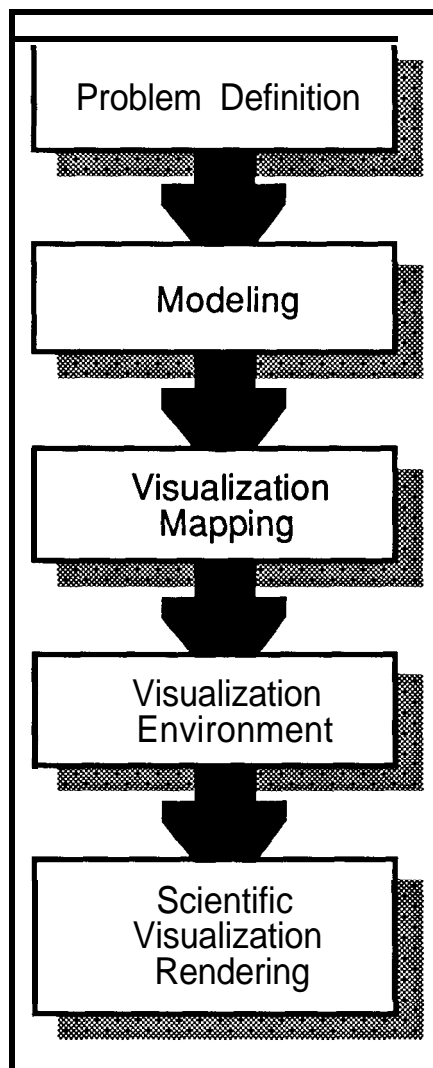


Figure 1 - The scientific visualization process.

numbers (or single components of a large database) are not important. Instead, its effectiveness lies in its ability to rapidly communicate large amounts of information in a format that enhances comprehension and insight.

Historically the concept of visualization predates the computer era. If you think about it, visualization has been a part of the scientific method ever since the Greeks. They employed basic linear graphing techniques in their complex architectural designs. But now we seem to want to handle more than one-dimensional data, and this new "complexity" requires that we be more creative in our rendering technique.

Therefore, in order to explore these visualization concepts on a practical level, I've decided to center this article about a detailed description of how to structure and implement a basic "visualization process" like that flowcharted in Figure 1. I propose that we develop a computational "visual-aid" that will utilize numerically simulated data from the application of Newtonian mechanics to the motion of an object within a gravitational field in order to demonstrate how basic insight into a system's behavior can be gained from this visualization process. In doing this, we will be called upon to apply basic tools from numerical analysis, computer graphics, and animation.

THE VISUALIZATION PROCESS

The best example is one that can be easily recognized and compared to one within our own personal experience. I have therefore chosen to "visualize" a system based on the motion of a ball, falling off a wall and bouncing into a hole. It's a simple system that each of us can visualize from experience. As such, it will be easy to "see" if our numerical modeling and visualization look and act like the real thing. But don't be fooled. A simple example doesn't guarantee a simple computer model. A true representation of this process requires that we numerically solve a second order differential equation, create a visual environment, and animate a time-evolving process.

We will construct this "scientific visualization" along the guidelines shown in the flowchart in Figure 1, where

1. Problem Definition-The basic problem is defined and the nature of the physical system to be visualized is detailed. This process includes the definition of the type of process under study, the limits on its behavior and its environment, and the underlying scientific principles involved.

2. Modeling (*mathematical*)—The system model is chosen. Since this example will involve the application

and the configuration of the display vehicle.

4. Creating the Visualization Environment—The background and each component of the visual display is defined in detail and constructed for use within the "active" visual mode. This step involves the actual coding of each component and the setup of any animation requirements.

5. Rendering—Each component of the "scientific visualization" application is combined to create an integrated environment. As the procedure is run, the mathematical model is used

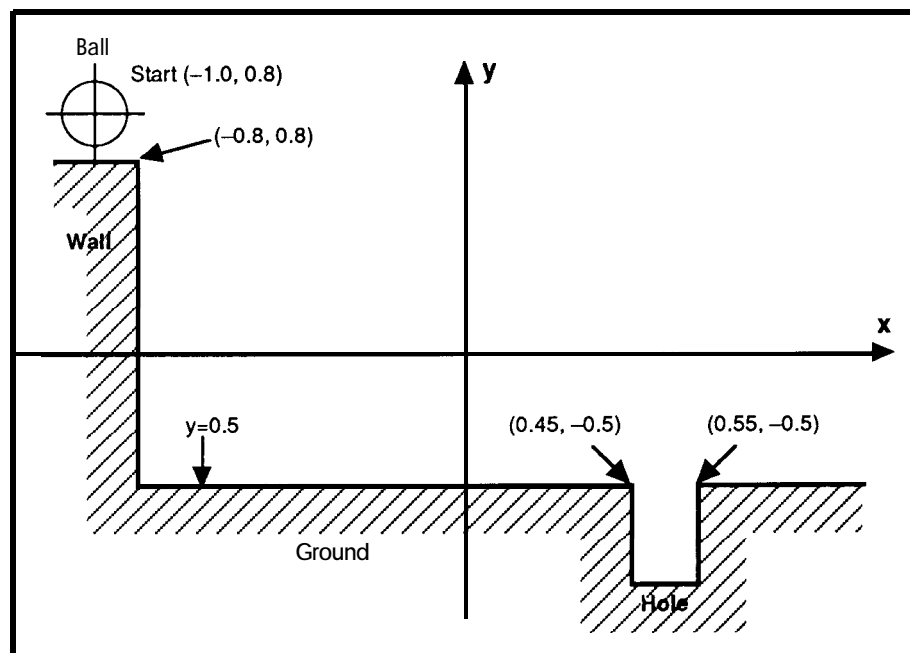


Figure 2—The example used in the article is a simple ball which falls off a wall, bounces, and usually falls into a hole near the right side of the system.

of physical laws of motion, the mathematical technique to be used must be defined and applied in order to derive appropriate equations of motion. These need to be consistent with a form that is easily used within the computer graphics display environment.

3. Visualization Mapping—The nature and extent of the actual "visual" display is defined, that is: What is the viewing field? How many objects will it contain? What form of background is necessary? What video mode and resolution is necessary? Do you use black and white or color? Is the scene animated? In other words, this step in the visualization process requires the definition of all the basic components

to predict the behavior of the system in a step-by-step manner. The visual display is then upgraded in a continuous fashion to reflect the changes in the mathematically derived data.

Of course the specific procedure shown above is not "universal" to all scientific visualization processes. The source of our data could have been empirical or from a closed-form analytic solution instead of our chosen numerically simulated data. And the nature of the problem itself will obviously change with application. However, the techniques and concepts employed are generic across the spectrum of physical science and can be readily applied to many different fields.

Our goal within this article is to go through the flowchart in Figure 1 in a step-by-step fashion while flushing out each component in order to create a practical, working scientific visualization example. But since this or any other example is **highly dependent** on the programming language, graphics library, and **display device used**, I have chosen to develop our example in as “generic” a fashion as possible. For a programming language I have chosen FORTRAN. This is not because C wouldn't be better. It's because I find FORTRAN easier for people to read. They can usually follow the program flow better without becoming programming experts. Most of the time I use C, but this is only because I get better access to system-level functions for enhancing my displays. However, in this application that is not necessary.

For a graphics library I have chosen to use the Microsoft **Graphics Library Routines** contained in the Microsoft FORTRAN V5.0 distribution. Microsoft seems to a fairly universal standard and available to

most programmers. It is not the most ideal graphics set, but it does contain most of the basic graphics utilities needed for addressing and assigning values to individual pixels within the display screen. Its most obvious failing lies in its restriction to the standard VGA modes. The best it can handle with 256 colors is the 320 x 200 display resolution. Of course you can improve the resolution to 640 x 480 if you wish to be restricted to 16 colors, but I usually find this unacceptable. I like maximum “realism” wherever possible. If you are looking for a more comprehensive graphics library to develop your own visualization application under, I recommend the Genus GX Graphics Toolkit.

PROBLEM DEFINITION

The first step in the **example** is the definition of the exact nature of the problem we are going to solve. We will consider the motion of a bouncing ball that first rolls along the top of a wall and then falls off and bounces several times until it falls into a hole in the ground. A drawing of the **appro-**

we will neglect air drag effects in this problem. But because the ball impacts with the ground and loses some energy through that interaction and in the elastic deformation process of the ball's surface, it is known that the vertical velocity is not conserved after each bounce. We can make a good approximation to our own observed real-world action by assuming that

approximately 28% of the total kinetic energy is lost during the ball-ground interaction. This is reflected in a vertical velocity damping of about 15% after each collision with the ground. The vertical velocity (Vv) of the ball after each bounce is therefore approximately equal to the negative value of 85% of the vertical velocity before the bounce.

As a “study” variable in the system, we will allow changes in the input of the magnitude of the constant horizontal velocity Vxo . For our purposes here, this value will range from 0.38 ft/sec (when the ball bounces and then just rolls into the hole) to about 4.7 ft/sec where the ball has sufficient velocity to be “dunked”

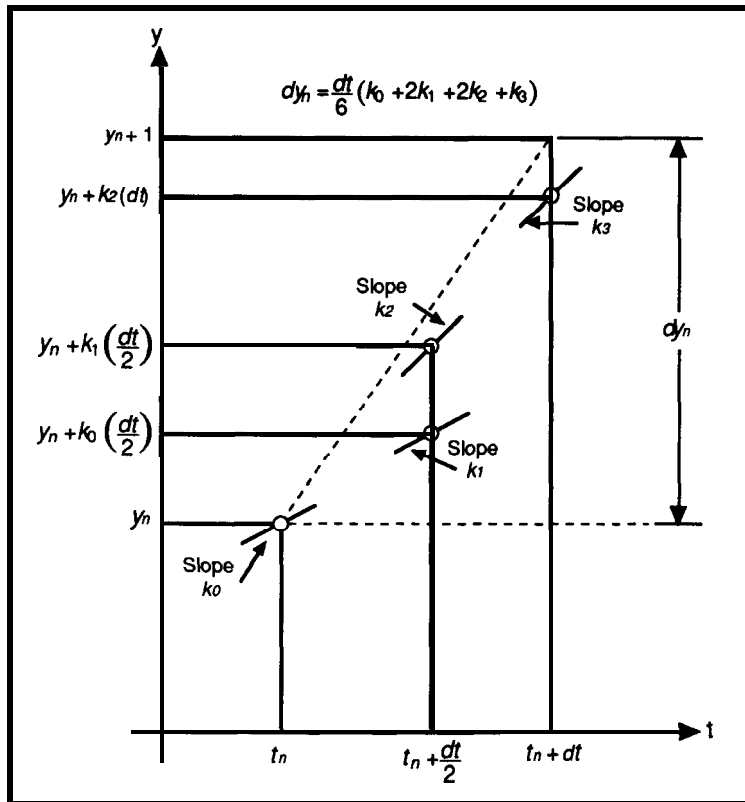


Figure 3—The Runge-Kutta procedure uses a weighted average of slopes, with those in the center receiving twice as much weight as those on the ends.

appropriate geometry and its assigned coordinate system is shown in Figure 2.

The ball will start its motion at the coordinates $(X_0, Y_0) = (-1.0, 0.8)$ and roll horizontally at constant velocity (Vxo) until it reaches the edge of the wall $(-0.8, 0.8)$. It will then fall off the wall and strike the ground 1.3 feet below and bounce upward. The ground itself is defined to be 0.5 feet below the coordinate axis ($y = -0.5$). Each time the ball strikes the ground it will either bounce or, if it falls in the appropriate place, it will fall into the hole and be trapped. This hole is placed 1.25 feet from the base of the wall. It is 0.3 feet deep and 0.10 feet wide.

The ball is assumed to move horizontally with a constant velocity since

into the hole without bouncing. Higher velocities will miss the hole entirely. Several choices of Vxo are possible within this range which will cause the ball to land in the hole after a series of bounces. If the ball misses the hole, we will continue the computation until it reaches a horizontal coordinate location of 0.85 feet ($Xstp$). If it enters the hole, we will allow it to slide downward until it falls to a coordinated depth of $Ystp = -0.8$ feet.

Since we will be studying the “motion” of a “dynamic system,” we will be solving for the coordinates of the ball in a point-to-point fashion during the time of its motion (flight path) from the top of the wall until it stops. The best visual aid for this ex-

ample is therefore one that demonstrates that time evolving change in position. To implement this, we will "animate" that motion by solving for the change in the ball's position every 0.001 seconds (dt).

MODELING

The type of motion defined by the bouncing ball can be described by the application of basic Newtonian mechanics. This results in a second-order differential equation. The best procedure to adopt here is to transform this equation into a system of simultaneous first-order equations which can be easily solved.

The governing differential equations are

$$\frac{dx}{dt} = V_{x0} \quad (1)$$

and

$$\frac{d^2y}{dt^2} = -g \quad (2)$$

Equation 1 indicates that the horizontal velocity is equal to a constant for all times, V_{x0} . Equation 2 expresses the fact that the ball will be within a gravitational force field and subject to a downward acceleration equal to 32.2 ft/sec^2 throughout the motion.

If we let

$$V_v = \frac{dy}{dt}$$

then

$$\frac{dV_v}{dt} = \frac{d^2y}{dt^2}$$

and Equations 1 and 2 may be rewritten in the following system of three first-order equations:

$$\frac{dx}{dt} = V_{x0} \quad (3)$$

$$\frac{dv}{dt} = -32.2 \quad (4)$$

$$\frac{dy}{dt} = V_v \quad (5)$$

The simplest procedure for numerically solving this set of simultaneous equations is through the application of the Runge-Kutta method using Euler's approximation.

RUNGE-KUTTA METHOD

To gain some basic insight into how this technique works, consider a basic first-order differential equation of the form,

$$\frac{dy}{dt} = f(t,y) \quad (6)$$

Now assume that we know the initial condition of the system and it is defined by its starting time and initial y -coordinate location, (t_o, Y_o) . Our goal is to find the new position of the system (y_1) at some short time interval later, $t = t_o + dt$, as defined by Equation 6. And, once we find this new position, we can then find the next (y_2) for $t = t_o + 2dt$, and so on, in a step-by-step manner. Using this approach, the required y - t relation can then be obtained and tabulated.

In the Runge-Kutta method, the basic formula for such a process is

$$Y_{n+1} = y_n + dy_n \quad (7)$$

where

$$dy_n = \frac{1}{6} (k_0 + 2k_1 + 2k_2 + k_3) dt$$

and k_x are called Runge's coefficients. These are defined as,

$$\begin{aligned} k_0 &= f(t_n, y_n) \\ k_1 &= f\left(t_n + \frac{dt}{4}, y_n + \frac{k_0 dt}{4}\right) \\ k_2 &= f\left(t_n + \frac{dt}{2}, y_n + \frac{k_1 dt}{2}\right) \\ k_3 &= f(t_n + dt, y_n + k_2 dt) \end{aligned}$$

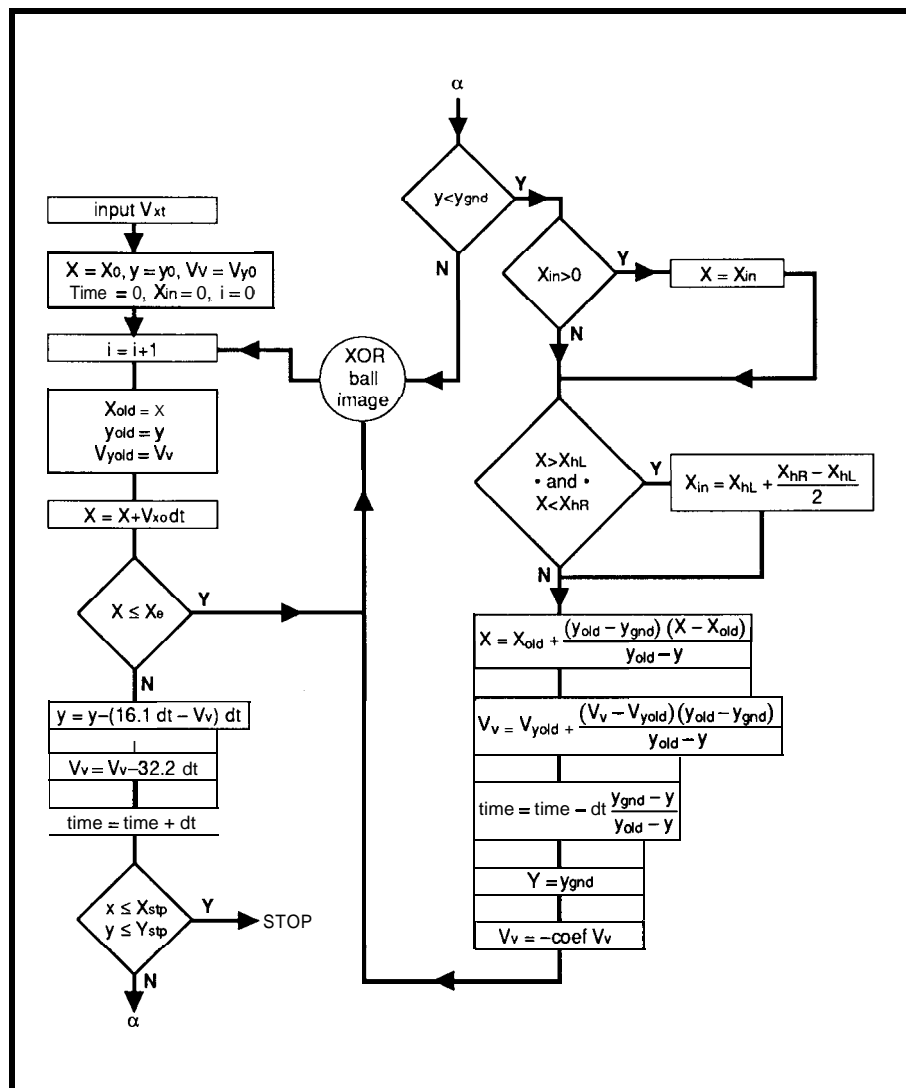


Figure 4—Flowchart for calculation of position of the bouncing ball as a function of the time using the Euler approximation of the Runge-Kutta numerical technique.

All of these coefficients can be derived from a Taylor expansion of Equation 7, but that exercise is not necessary for us to follow through here. Instead, it is much more enlightening to examine the simple geometrical interpretation of these constants as they are shown in Figure 3. All four k -values represent the slopes at various points of the time-changing spatial function. k_0 is the slope at the starting point, k_3 is the slope at the right-hand point whose coordinate is $y_n + k_n dt$, and k_2 and k_1 are the slopes at mid-points whose ordinates are $y_n + 1/2k_0 dt$ and $y_n + 1/2k_3 dt$, respectively.

As can be seen from Equation 7 and Figure 3, the Runge-Kutta procedure uses a weighted average of slopes, with those in the center receiving twice as much weight as those on the ends. For many problems, this extra weighting scheme is not necessary and they can be approximated with sufficient accuracy by a first-order term of the Runge-Kutta method. Here, the basic formula is simply,

$$Y_{n+1} = y_n + k_0 dt \quad (8)$$

This is called the Euler approximation and it depends only on y_n and the slope k_0 at the point (t_n, y_n) .

EQUATIONS OF MOTION

For our simple bouncing ball, Euler's approximation contains a sufficient degree of accuracy for a viable animated visual display. We will therefore apply Equation 8 to our differential Equations 3, 4, and 5. As a result these expressions become:

$$x_{i+1} = x_i + Vx_0(dt) \quad (9)$$

$$Vv_{i+1} = Vv_i - 32.2(dt) \quad (10)$$

$$y_{i+1} = y_i + \frac{1}{2}(Vv_i + Vv_{i+1}) dt \quad (11)$$

If we now substitute the value of Vv_{i+1} of Equations 10 into Equation 11, we obtain a set of equations of the form:

$$x_{i+1} = x_i + Vx_0(dt)$$

$$Vv_{i+1} = Vv_i - 32.2(dt)$$

```

a)
c   The initial velocity, position, time and time step.
c   -----
vvo=.0      ! initial vertical velocity
Xo=-1.0     ! initial x position
Yo=.8       ! initial y position
dT=.001     ! the incremental time step
time=0      ! total time

c   The boundaries and boundary conditions.
c   -----
coef=.85    ! ball-ground damping constant
Ygnd=-.5    ! y location of ground
XhL=.45     ! x location of left-hand-side of the hole
XhR=.55     ! x location of right-hand-side of the hole
Xe=-.8      ! x position where ball falls off the wall
Xin=0.0     ! x coordinate of where ball hits the hole

Xstp=.85    ! the x stop coordinate of the system
Ystp=-.8    ! the y stop coordinate of the system

b)
read(*,*)Vxo
x=Xo
y=Yo
vv=vvo
i=0
time=0
Xin=0.0

c)
100  i=i+1

      ERASE LAST DRAWING OF BALL ! not actual FORTRAN coding

Xold=x
Yold=y
Vyold=Vv

x=x+Vxo*dT

if(x.lt.Xe)goto 200

y=y-(0.5*32.2*dT-Vv)*dT
Vv=Vv-32.2*dT
time=time+dT

d)
if(x.ge.Xstp)goto 500
if(y.le.Ystp)goto 500

if(y.lt.Ygnd)then
  if(Xin.gt.0)x=Xin
  if(x.gt.XhL .and. x.lt.XhR)then
    Xin=XhL+(XhR-XhL)/2
    goto 200
  endif
  x=Xold+((Yold-Ygnd)*(x-Xold))/(Yold-y)
  Vv=Vyold+((Vv-Vyold)*(Yold-Ygnd))/(Yold-y)
  time=time-dT*(Ygnd-y)/(Yold-y)
  y=Ygnd
  Vv=-coef*Vv
endif

200  DRAW THE NEW IMAGE ! not a FORTRAN code statement
      goto 100

500  read(*,*)          ! wait for user input to continue

```

listing 1 -Before starting anything, (a) the initial conditions and (b) initial horizontal velocity must be set. (c) Before the new position can be drawn, the old ball must be removed. (d) The new position is calculated and the ball is redrawn.

$$y_{i+1} = y_i - [16.1(dt) - Vv_i] dt \quad (12)$$

These three above equations can now be applied in a simple computer calculation loop from which the time-dependent coordinate (x,y) of the ball

can be calculated. A sample flowchart of the FORTRAN routine that we will employ for this process is shown in Figure 4.

To implement this process, we must first define all of our initial con-

ditions and decision coordinates. Since a high degree of accuracy is not required, each of these variables can be defined as REAL*4 in our example. The integer counter i is defined as INTEGER*2 since we expect to exceed 127 dt time steps. See Listing 1a.

The initial horizontal velocity is then input and the loop variables (x , y , V_v , i , time, X_{in}) are set to their initial conditions, shown in Listing 1b.

The cycle counter is then incremented by 1 and the basic loop procedure is started. Since this is an "animation sequence," the last drawing of the ball within the display image is erased so that the new image position can be drawn after its new position is determined. However, before this calculation takes place, the current position and velocity corresponding to this last (i th) cycle is stored in temporary storage (X_{old} , Y_{old} , V_{vold}) and the new x position (for cycle $i+1$) is determined. If the ball collides with the ground, these previous (old) values will be used to determine the ball's reflection from that surface. If the ball is still on top of the wall, that is, for x less than x_e , then the ball is drawn at its new horizontal position and the counter is incremented for the next cycle. Otherwise, the $i+1$ cycle y coordinate (y), vertical velocity (v_v), and the total elapsed time is calculated. See Listing 1c.

Once the new position of the ball is determined, it is necessary to make several conditional tests to determine the current status of the calculation sequence. The position is first tested to determine if the ball has reached its end-point boundary conditions. In other words, has it fallen into the hole and reached its bottom ($Y_{stop} = -0.80$) or has it bounced past the hole to the right-hand side of the display screen and reached the x -stopping point ($X_{stop} = 0.85$)? If either of these conditions is met, the calculation and animation sequence is suspended.

If the ball has not reached these stopping boundaries, we must determine if it should bounce or continue in its upward or downward flight path. This is accomplished by testing the value of the y coordinate. If the ball's location is at the surface or slightly

below it (based on the distance traveled in one time step dt), then we must determine if it is falling into the hole or not. This is accomplished by testing to see if the x -position is within the hole's width between $x = 0.45$ to $x = 0.55$. If it is, then the position is reset to the center of the hole (X_{in}) and the ball is drawn at the new position and allowed to fall to the bottom in the next few cycles.

This center coordinate x_{in} is used for multiple purposes. It centers and constrains the motion of the ball as it slides down the hole so that we can simulate the realistic motion of a solid spherical ball sliding down a circular pipe slightly larger than its own diameter. But at the same time it is used as a trigger function. It indicates whether or not the ball has entered the hole. If $x_{in} = 0$ then we know that the ball has yet to reach the hole. If x_{in} is greater than zero then we know it has just hit the hole or is sliding down it.

If the ball is not in the hole and the y coordinate indicates that it has collided with the surface, then the ball must bounce. Under these conditions, the new x -coordinate location and vertical velocity are calculated using the previously stored values from the last cycle. The velocity is then damped by 15% and its direction is inverted to simulate the reflection of the ball off the ground surface. The new position of the ball is then displayed in the animation sequence and the next cycle is initiated, as shown in Listing 1d.

Once the total motion sequence is completed, a new horizontal velocity can then be input and a new test of the ball's motion can be "visualized." How this visualization is accomplished using the simple graphics utilities available within the Microsoft Graphics Utility Library is the subject of the next few sections.

VISUALIZATION MAPPING

The next step in creating a workable "visual" representation of our scientific model is based on the definition of the graphics environment. This display must be chosen carefully since it is highly dependent on the nature of the process being visualized. For our

REALIZE THE POWER OF...



PARADIGM

The Model for
Programming Productivity

3301 Country Club Road
Suite 2214
Endwell, NY 13760
(607) 748-5966
FAX: (607) 748-5968

All trademarks are property of their respective holders.

bouncing ball scenario, we need only consider the animated motion of a single object: the ball. But that ball must move within some referenced environment. This environment, or background image, must be laid out to correspond to the system limits and boundary conditions specified in the previous "modeling" section. And since this is an animated visualization, we must take into account the specific hardware used. This greatly affects the time needed for image modifications. It's obvious that different graphics cards, screen memory management, and processor clocks speeds affect the rate at which a pixel on the screen can be changed. Therefore, before selecting the display mode, the following should be considered:

1. How much visual detail does each object or component of the background require?
2. How many colors or shades of gray are necessary to properly render that detail?
3. If you are animating objects, what constraints on the image refresh

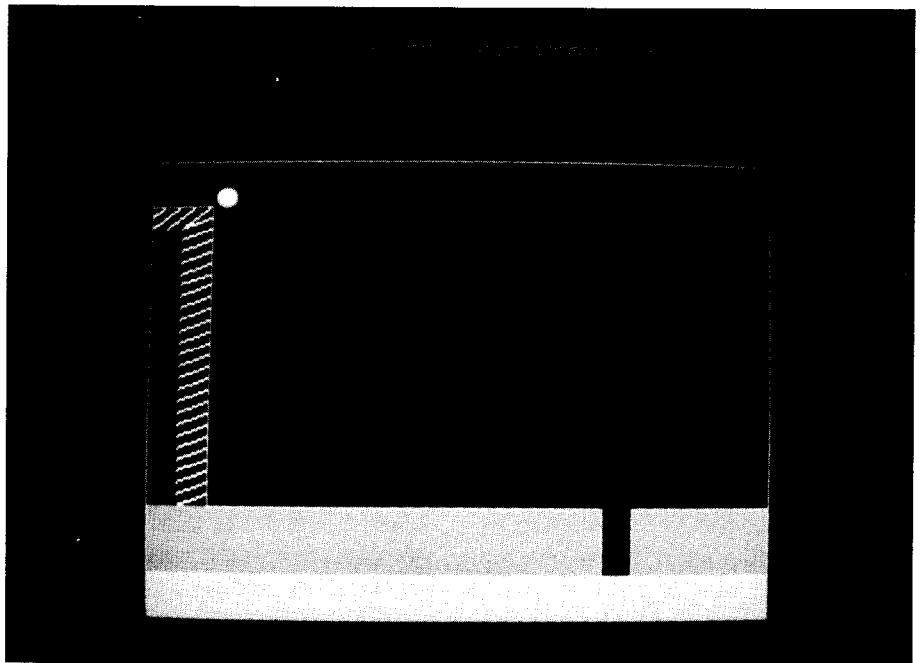


Figure 5a—A two-dimensional textured background is used to add some realism to the simulation.

speed exist due to the size and number of colors of each object (the number of bytes per pixel and the number of total pixels within an object affect perception of the motion since the video refresh rate is not variable)?

4. What constraints do you have placed on your choice of video mode resulting from compatibility and transportability to other PCs?

I'm sure you can think of other important questions that you would

PROCONTROL

Monitor, Simulate, and Control for Your IBM PC

Comprehensive utility features. Easily add into existing or your own programs with a few simple calls!

Takes advantage of ProControl I/O module features, but can be used with other hardware as well.

Rich set of instruments includes: Dial gauges, bar gauges, thermometers, seven-segment displays, strip-charts, annunciators, buttons, alarms, thermocouple linearization, timing, PID control and more!

SCALABILITY and "virtual coordinates" for your application to run unmodified on any display!

Now supports QuickBASIC and Turbo Pascal!

One version supports Turbo C, Microsoft C, QuickC, QuickBASIC and Turbo Pascal.

Hardware Independent

New 3D appearance!

\$249⁰⁰ complete.

Royalty free. Source code available.

Advanced Design Solutions

1920 Moores Mill Road
Atlanta, GA 30318

For a free demo, call

(404) 352-4788

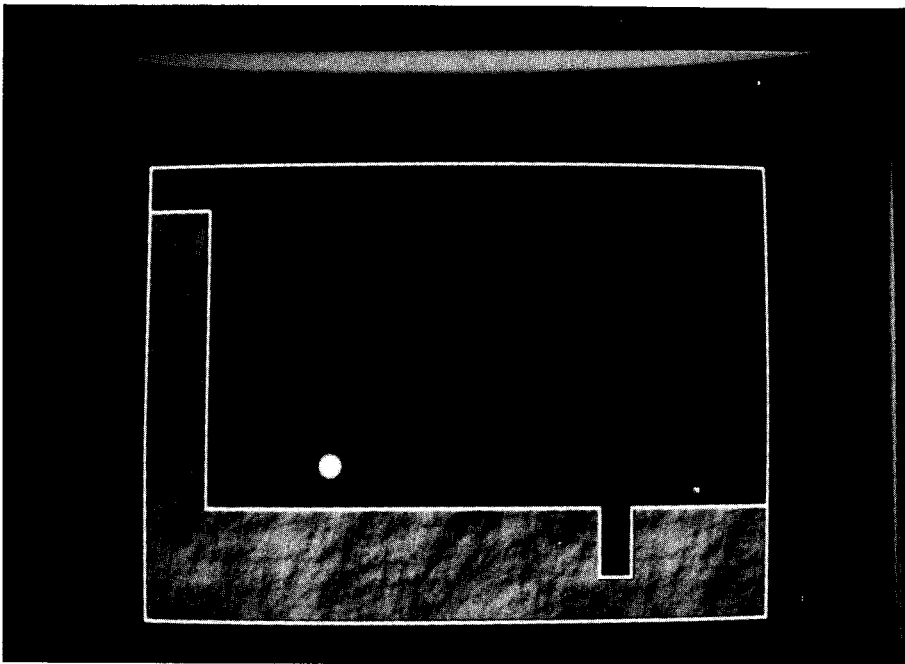


Figure 5b—A block-line background is often used due to its simplicity, but fails to add any realism to the system.

ask during a design review, but I think the four listed above give you an idea of the type of thought that goes into choosing the “visualization map.” For this article I was highly constrained by Question 4. If this example is to be worked with by most of you, I needed to keep my graphics mode to a widely accepted standard. This meant choosing between standard EGA or VGA modes. And since all computer systems with a VGA card support the basic 320 x 200 by 256-color mode, I chose that for this exercise. My choice of the 256-color mode was based on my desire for some realism. I wanted to generate a black-and-white (B/W) display but I didn’t want to be limited to the four shades of gray that the standard 16-color modes of EGA or VGA would allow. By selecting the 256-color mode, I could then construct a B/W table of 64 shades. This would allow me to render the wall and ball with some detail. Also, as long as I kept the size of my ball down, I could render it as a 3-D object instead of a 1-D point or a 2-D filled circle. This at least would add some realism to the visualization.

How you choose this background can make or break the visualization. Perception is a funny thing. It is highly dependent on context, especially where motion is concerned. I therefore chose to create our example back-

ground using a textured surface for the wall and ground. This tended to make the eye see this ball as bouncing on a piece of concrete viewed from face-on, as shown in Figure 5a. The texture lends some 3-D effect to the surface of the wall and ground. This compensates for the “flat-look” experienced when using the more traditional line, solid-fill-rectangle mode of drawing (see Figure 5b).

CREATING THE VISUALIZATION ENVIRONMENT

With the display mode and basic components chosen, each of the visual constructs must be coded in our development language. This is accomplished through the use of a variety of graphics utilities provided within the Microsoft run-time library (GRAPHICS.LIB) which are linked together when the executable is created. However, the steps necessary for this “environment generation” process must be accomplished in specific correlated steps which build on one another to create the desired visual display effect.

Every program using the graphics library must explicitly declare any routine it uses. This means that you are required to reference the interface routines provided in the include files FGRAPH.FI and FGRAPH.FD which

contain all the procedure declarations, structure, and symbolic constant declarations for each of the called graphics functions. These should be the first two lines of code in your program.

SETTING THE VIDEO MODE

The display device utilized by the running computer system must first be polled to determine if it is capable of supporting the chosen video mode. If it is, then the mode is set. Otherwise the execution of the example visualization is terminated.

An example of the FORTRAN code needed to implement this step is shown in Listing 2a. The `getvideoconfig` procedure is used to find the type of video adapter that is installed. This information is returned in the `/videoconf ig/` structure defined in `FGRAPH.FD` and referenced within our code through the `record` declaration. The actual video mode is set by a call to the `setvideomode` utility and passing it the `$MRES 2 5 6COLOR` parameter requesting that the video mode be set to VGA 320 x 200 by 256 color. The integer variable `dummy` is used to return error/success flags. If `setvideomode` sets `dummy` to zero, the hardware requested is not available. This condition is tested. If `dummy` is set to a `nonzero` number, this indicates that the video mode was successfully set.

SETTING THE COLOR PALETTE

The selected VGA video mode enables us to use up to 256 colors at any one time within our scientific visualization display. This grouping of allowed colors is defined as the display palette. If we had chosen some other video mode, we could have palettes containing 2, 4, 8, or 16 colors. Each of the colors in a palette is referenced by its respective “index number.” In the VGA 256-color mode, we can mix varying amounts of the base colors red (R), green (G), and blue (B) to create up to 262,144 (256K) different color combinations. Two-hundred fifty-six of these can in turn be assigned for use within our application’s color palette.

When you create this color combination, you specify a level of intensity (over a range of 0-63) for each of the R, G, and B bases which is coded into a long integer of four bytes (32 bits) for interpretation by the video board.

MSB LSB
 00000000 00BBBBBB 00GGGGGG 00RRRRRR

where B, G, and R represent the bit values for blue, green, and red, respectively. The most-significant bit (MSB) contains all zeros and the two higher bits in each of the next three bytes are also set to zero. For example, to make the brightest red, just set all B and G to 0 and set R to 1.

00000000 00000000 00000000 00111111

In hexadecimal notation, this number equals #0000003F. Each base color, therefore, ranges over the above defined 0 to 63 range. A simple function can be created for mixing colors and encoding the palette's index number. This function has the form shown in Listing 2b.

Here, three integer values for R, G, and B are passed to the function and encoded into a 4-byte variable and then returned to the calling routine.

The best way to visualize this RGB mixing is to view each color as the axis of a three-dimensional coordinate system with the **coordinates** of each point within that system corresponding to some "mixed" color. If you stay on one of the axes, say along R, then G and B will be zero and the intensity of the defined red will vary from black (when R = 0) to the brightest red (when R = 63). From this scheme you can easily see that the shades of gray can be created by combining equal amounts of each color. Therefore, the coordinates for these shades of gray can be seen to fall along a diagonal line that passes through the center of the 3-D color vector space. Under this condition, the RGB value of (0,0,0) is black, a value of (32,32,32) is gray, and (63,63,63) is bright white; the diagonal represents the coordinates of smoothly increasing gray shades from black to white. Due to the discreteness of the

```

a)
integer*2 dummy
record /videoconfig/ myscreen
CALL getvideoconfig( myscreen)

dummy = setvideomode( $MRES256COLOR)
if (dummy.eq.0) then
  print *, 'Error: cannot set graphics mode'
  stop
endif

b)
INTEGER*4 FUNCTION RGB( r, g, b )
INTEGER*4 r, g, b
RGB = ISHL( ISHL( b, 8 ) .OR. g, 8 ) .OR. r
RETURN
END

c)
integer*4 dummy4,  iblue, ired, igreen
integer*4 RGB, tmp

do igreen = 0, 63
  ired = igreen
  iblue = igreen
  tmp = RGB( ired, igreen, iblue )
  dummy4 = remappalette( i, tmp )
  i = i + 1
enddo
  
```

Listing 2—(a) The display device being used must support the proper video mode. (b) A function to mix colors and encode the palette's index number is simple indeed. (c) Sixty-four levels of gray are necessary to add realism.

component colors (0-63), however, the number of gray shades along this diagonal is restricted to 64.

In our example scientific visualization, the 2-D textured background and the 3-D ball are rendered in shades of gray. The actual display would have been faster and simpler to implement if a 16-color palette mode had been chosen, but this limits your perception. It's difficult to distinguish surface texture using only four shades of gray (black, light gray, white, bright white). For that reason, the visualization example used the code in Listing 2c to create a B/W palette with addressable indices ranging from 0 to 63 to define the shades of gray directly.

Here, the remappalette graphics utility was used to load the specified palette index (i) with the appropriately coded gray shade RGB coordinate number.

DEFINING THE COORDINATE SYSTEMS

After the video mode and the color palette were set, the image display coordinate systems were defined and set. This is a two-fold process which was coded and implemented as in Listing 3a.

The current video screen was reset (cleared) using the clearscreen

utility, and all of the specifics which define the current video mode were called and loaded into the getvideoconf ig structure. The "physical coordinates" of the system are returned and set within the variables x1,x2,y1, and y2. They represent the number of pixels in the horizontal or x-axis (=x2-x1+1 pixels = 320) and the vertical or y-axis (=y2-y1+1 = 200). This physical coordinate system has its origin in the upper left-hand corner of your video window. These physical coordinates refer to each pixel directly and, as such, are represented by integer values. These absolute references to a physical location on the viewing window can be used to define the active display area. This is accomplished by calling the setviewport graphics window routine. In our example, I chose to use the entire screen (defined by x1, x2, y1, y2).

Once the physical working area is set, a userdefined coordinate system can be defined using the setwindow utility. This function enables us to overlay the working area with our own coordinate scales and to select the location of the origin. In this specific case, I chose to use the same coordinate scheme (with the origin remaining in the upper left-hand corner). This was done so I could input my

```

a)
integer*2 dummy,x1,y1,x2,y2
double precision wx1,wy1,wx2,wy2
record /videoconfig/ myscreen

call clearscreen ($GCLEARSCREEN)
CALL getvideoconfig( myscreen )

x1 =
y1 = 1
y2 = myscreen.numypixels
x2 = myscreen.numxpixels

call setviewport( x1, y1, x2, y2)

wx1= float(x1)
wy1= float(y1)
wx2= float(x2)
wy2= float(y2)
dummy = setwindow(.FALSE.,wx1,wy1,wx2,wy2)

call fill

```

```

b)
subroutine fill
INCLUDE 'FGRAPH.FD'
integer*1 dat(256,256)
integer*2 dummy,ipx
double precision wx,wy

open(1,file='wall',form='binary',recl=256,status='old')
do i=1,256
  read(1,err=30)(dat(i,j),j=1,256)
enddo
30 close(1,status='KEEP')

do wx=3,318
do wy=4,198
  i=nint2(wx)
  if(i.gt.200)i=i-200+5
  j=nint2(wy)
  ipx = dat(i,j)
  if(ipx.lt.0)ipx=ipx+256
  inc = int4(ipx/4)
  dummy = setcolor(inc)
  dummy = setpixel_w(wx,wy)
enddo
enddo
return
end

```

```

c)
wx1=1
wy1=1.
wx2=1000.
wy2=1000.
dummy = setwindow(.FALSE.,wx1,wy1,wx2,wy2)

```

Listing 3—(a) After the video mode and color palette are set, the image display coordinate system is defined and set. (b) The background texture is read from a file. (c) Once the background is complete, the window coordinates can be reset to values used during the animation.

texture map (contained in the binary image file WALL) and fill the entire screen with its image. The FORTRAN code used to read this texture map has the form shown in Listing 3b.

This texture map was externally created using a Fractional Brownian Motion fractal technique used for modeling surfaces. It is stored as a 256 x 256 binary image. The code in Listing 3b inputs this image and then converts the byte data to integer data so that its dynamic range spans the interval of 0 to 255. This data is then divided by four so that each point in the texture map can be assigned a B/W

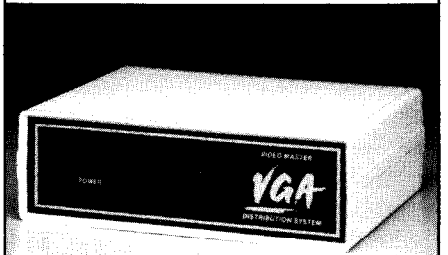
intensity value ranging from 0 to 63. Each of these values corresponds to the gray value defined within the B/W "color" palette. That gray "color" is then set within the video display window. This is accomplished by setting the location of the pixel (within the video window using `setpixel_w`) to the value of the *i,j* indices of the data within the texture array and then setting the pixel color at that location to the referenced palette gray shade (using `setcolor(inc)`). Since the screen is 320 pixels long and the texture map is only 2% points in extent, the horizontal components of the texture map

Video VGA, EGA, CGA

Market Central manufactures video products for sharing, extending, and remoting your Computer's video signal. Our video products are used in classrooms, factories, exhibits; also security installations and trade shows.

VGA KVM

Our newest product, the VGA KVM enables the user to remote the keyboard, (video) monitor and mouse up to 150 feet away from the PC. There is also an optional Local Module that lets you connect another keyboard, monitor and mouse at the PC. The unit has a switch selectable option that sends the display to both monitors simultaneously. Keyboard access is controlled by the local user in a shared system. The unit derives it's power from the PC and there is no external power necessary. Retail are, Host \$250, Remote \$115, Local \$150.



VGA splitter

Our VGA splitter shares your PC video signal (the display on your monitor) with as many as 8 others up to 250 feet away. Retail for the 4 port is \$385, 8 port \$699. This unit is ideal for training and classroom environments.

VGA line driver

Our VGA line driver is used to increase the distance your VGA signal can be sent, up to 250 feet from the PC. Retail is \$199.

Market Central manufactures a variety of interface converters, print buffers, and fiber optic equipment at our plant in Pittsburgh, Pa. For additional information on these or our video products please contact us.



MARKET CENTRAL, INC.
600 N. Main St.
Houston, Pa. 15342
(412) 746-6000
FAX (412) 746-5400

are wrapped around to fill the full 320 horizontal pixels. A 2- to 4-pixel border was also left blank as the background texture map was displayed. This was done to leave room for other background drawing.

Once the background "fill" function is completed, the window coordinates can be reset to values used during the actual animation sequence (shown in Listing 3c). But before the animation is actually initiated, an image map of the three-dimensional ball must be entered and the remainder of the background display needs to be set up.

CREATING AN IMAGE MAP

A three-dimensional rendering of the ball was created externally and stored as a 64 x 64 binary image for use with our visualization code. It was constructed as a simple spheroidal object having a radius of 32 pixels and a surface defined by 64 shades of gray. The size and scaling was chosen to optimize its construction, but it's obvious that the size is unacceptable for our visualization process. The radius of the ball spans 10 percent of our physical coordinate viewing field. Using the "window coordinate" system fixes this problem for us. It basically rescales the ball as it maps it over to the 1 to 1000 coordinate range. This reduces the ball's size to about one-third of its original.

The code in Listing 4a shows how the 'ball' image file was retrieved from a disk file and then displayed on the video screen using the `setcolor` and `setpixel_w` utilities.

It is displayed in the upper right-hand corner of the screen so that it can be captured to an image buffer file for later display at any location within the active viewport. A correction to the horizontal display coordinate was made during the graphical display to adjust for the aspect ratio of the 320 x 200 video window. This correction assures that the mapped ball remains a sphere when it is displayed. After the ball is drawn, the size of buffer needed to contain it is determined and then allocated according to the code in Listing 4b.

```
a) integer*1 bal(64,64)
integer*4 inc, imsize, xwid, ywid
integer*1 buffer[ALLOCATABLE] (:)
real*4 wx, wxx, wy, wyy
integer*2 action(5), error, ipx
DATA action / $GPSET, $GPRESET, $GXOR, $GOR, $GAND /

open(1, file='ball', form='binary', recl=64, status='old')
do i=1,64
  read(1, err=30) (bal(i, j), j=1, 64)
enddo
30 close (1, status='KEEP')

do wx=1., 64.
do wy=1., 64.
  i=nint2(wx)
  j=nint2(wy)
  ipx = bal(i, j)
  if(ipx.lt.0) ipx=ipx+256
  inc = int4( ipx/4)
  wyy=wy
  wxx=wx*(5.5/7.)
  dummy = setcolor( inc )
  dummy = setpixel_w(wxx, wyy)
enddo
enddo
```

```
b) xwid = 64*(5.5/7)
ywid = 64

imsize = (xwid*mymouse.bitsperpixel+7)/8
imsize = 4t imsize*ywid

ALLOCATE ( buffer( imsize ), STAT = error)
if( error.ne.0) then
  dummy = setvideomode( $DEFAULTMODE)
  stop 'error: insufficient memory'
endif
```

```
c) wx1=1.
wx2=64.*(5.5/7)
wy1=1.
wy2=64.

call getimage_w(wx1, wy1, wx2, wy2, buffer)
call putimage_w(wx1, wy1, buffer, action(3))
```

listing 4—(a) The image of the ball is retrieved from disk and displayed. (b) The size of the buffer needed to contain the ball is determined and allocated. (c) The ball is erased by XORing it with itself.

Here the ball's image size (`imsize`) is the number of bytes needed to store the image. This is defined by the bounding rectangle of width of `xwid` and height of `ywid`. The display aspect ratio is included. The image is then captured by a call to the `get image-w` graphics utility and then erased from the screen by writing the image over "onto itself" in a logical XOR fashion using the `put image-w` utility. See Listing 4c.

The `put image_w` function transfers to the screen the image stored in the buffer. It is referenced to the upper left corner of the image as defined in the window coordinate system. The `action` variable defines how the interaction between the stored image and the one already on the display takes place. Use of "`action(3) = $GXOR`" causes the points on the screen

to be inverted wherever a point in the image buffer exists. This behavior is exactly like that of the typical cursor, which, when put against a complex background twice, results in the background being restored. Therefore, employing this same technique, we will be able to "animate" our ball against our background without erasing it or having to rebuild it after each motion increment.

FINISHING THE BACKGROUND

Before that actual animation event can take place, the background needs to be completed. This is accomplished by erasing large rectangular sections of the "texture-filled" screen to leave behind the wall, ground, and the target hole. This is accomplished by setting the active palette color to black

```

a)
inc = 0
dummy = setcolor( inc )
dummy = rectangle_w( SGFILLINTERIOR, 0.,0.,1000.,104.)
dummy = rectangle_w( SGFILLINTERIOR, 98.,100.,1000.,747.)
dummy = rectangle_w( SGFILLINTERIOR, 723.,752.,777.,902.)

```

```

b)
inc = 63
dummy = setcolor( inc )
call moveto w(0.,0.,wxy)
dummy = lineto_w(1000,0)
dummy = lineto_w(1000,1000)
dummy = lineto_w(0,1000)
dummy = lineto_w(0,0)

call moveto_w(0.,100.,wxy)
dummy = lineto_w(100,100)
dummy = lineto_w(100,750)
dummy = lineto_w(725,750)
dummy = lineto_w(725,900)
dummy = lineto_w(775,900)
dummy = lineto_w(775,750)
dummy = lineto_w(1000,750)

```

Listing 5—(a) The background/s completed by erasing the area where the animation is to take place. (b) The final step is to outline the wall, ground, and hole.

and employing the `rectangle_w` function to draw and fill "blank" sections of the screen as shown in Listing 5a.

Once the background structure has been isolated, the active color index is set to bright white and the wall, ground, and hole are outlined using Microsoft's line drawing functions as in Listing 5b.

The background is now completely constructed and the ball is stored within an image buffer for fast video display. The actual "scientific visualization" process can now be activated and rendered.

RENDERING

Once the operator enters the initial horizontal velocity, the motion of the ball is determined according to

procedure defined in the previous Scientific Modeling section. A brief outline of that calculation is shown in Listing 6.

As stated above, the animation is achieved by overwriting the ball in a logical XOR fashion. This is accomplished by the loop shown in Listing 6. The ball is erased at the position it occupied during the last cycle and then drawn at the next position. This is done over and over until the ball's motion is stopped according to the problem boundary conditions. Since the motion calculation utility uses a coordinate system of -1.0 to 1.0 with its origin at the center of the image (the window coordinates range from 1 to 1000), a set of transformation equations ($wx1=$, $wy1=$) between the two systems was employed. This was done deliberately to demonstrate how mul-

```

100  i=i+1

      wx1 = dfloat(((x/0.002)+500)*0.967)
      wy1 = dfloat((500-(y/0.002))-60)

      call putimage_w(wx1,wy1,buffer,action(3))
! erase

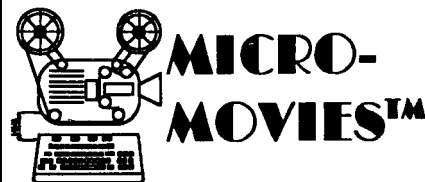
      .
      find new ball position

      and test for boundary conditions

      .
200  call putimage_w(wx1,wy1,buffer,action(3))
! draw
      got0 100

```

listing 6—Once the operator enters the initial horizontal velocity, the motion of the ball is determined and displayed.



COMPUTER ANIMATION FOR PROFESSIONALS

Animate static graphic images from your favorite graphics program, user written program, or post processor in real time!

Superior Tool For:

- Scientific Visualization
CFD • Structural Dynamics • Shock Physics • Fractals • Molecular Biology-Weather Forecasting

- Multi-Media presentations

* Instructional Aid

15 -30 high resolution, full color EGA or VGA frames per second (typical)

LINKABLE routines for user written programs in 'C', FORTRAN, Pascal, and QuickBasic™

Combine SLIDES and ANIMATIONS for seminars, conferences, and classrooms.

ONLY \$199



RAINDROP™

PRINT SCREEN UTILITY

FAST, compact PrtScrn Utility for end users AND developers. Hardcopy as fast as 10 seconds. Average binary size ~ 6 kbyte. 14 video graphic standards. Scale, rotate, colorize and more. 'Call' from user-written programs. Complete 9- and 24-pin dot-matrix, inkjet, and laserjet printers.

COMPLETE LIBRARY
ONLY \$44.95

ECLECTIC SYSTEMS

3106 St. David Ct. • Springfield, VA 22153
(703) 440-0064 FAX (703) 455-6965

For IBM-compatible computers

multiple coordinate systems could be used interactively within a single visualization problem. And, since the ball is not a point, both the x and y window coordinates of the actual mapped ball image had to be offset to account for

its size. This offset was easily accomplished within the transformation equation.

THE VISUALIZATION EXAMPLE

Once all the pieces described above are put together, you end up with a working version of the bouncing-ball scientific visualization model. It would be redundant to list the final code here, but you can download it from the Circuit Cellar BBS.

ONWARD AND UPWARD

By integrating physical simulation with visual simulation we have been able to demonstrate the effectiveness of the scientific visualization process, even with such a simple example as our bouncing ball. This visual approach has made the task of data interpretation a simpler and more straightforward task. Just think of trying to scan lists of (x,y) data in order to determine how the ball would move, and then compare that to the visual

action displayed on your monitor. The insight gained speaks for itself.

But remember, visualization shouldn't be viewed as the end result of a process of scientific analysis. Instead, it should be considered as part of the process itself. Its interaction with the concept of "human perception" makes it more than a simple application of techniques for displaying data. Used with thought, visualization can be used as a paradigm for exploring regions of untapped knowledge. Visualization is not new, but its use as a perceptual tool and by the general scientific community is. ♦

Chris Ciarcia has a Ph.D. in experimental physics and is currently working as a staff physicist at a national lab. He has extensive experience in computer modeling of experimental systems, image processing, and artificial intelligence. Chris is also a principal in Tardis Systems.

IRS

- 404 Very Useful
- 405 Moderately Useful
- 406 Not Useful

REFERENCES

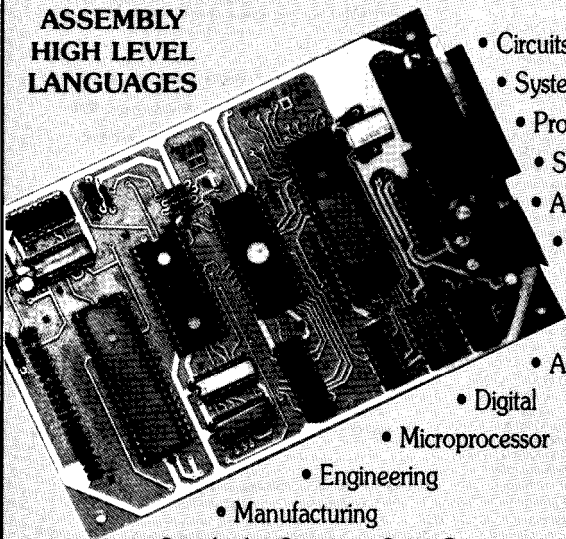
1. Microsoft FORTRAN V5.0, "Advanced Topics," Microsoft Corporation, 16011 NE 36th Way, Box 97017, Redmond, Wash. 98073-9717.
2. J.D. Foley and A. Van Dam, "Fundamentals of interactive Computer Graphics," Addison-Wesley Publishing Co., Reading, Mass., 1982.
3. W.M. Newman and R.F. Sproull, "Principles of Interactive Computer Graphics, 2nd ed.," McGraw-Hill, New York, 1979.
4. "Visualization in Scientific Computing; A Special Report of ACM SIGGRAPH, Computer Graphics, Vol. 21, App E, (1987), pp. E1-E7.
5. A. Smith, 'Volume Graphics and Volume Visualization, A Tutorial,' Pixar Technical Memo 176, Pixar, San Rafael, Calif. (May 28, 1987).

MAGNUM OPUS

CUSTOM ELECTRONIC ENGINEERING

SINGLE BOARD COMPUTERS EPROM PROGRAMMERS

**ASSEMBLY
HIGH LEVEL
LANGUAGES**



- Circuits
- Systems
- Products
- Services
- Assembly
- Repair
- Test
- Analog
- Digital
- Microprocessor
- Engineering
- Manufacturing
- Standard • Custom • Semi-Custom

284 Kennedy St • Iselin, NJ 08830
(908) 283-4925

MS-DOS EPROM PROGRAMMING SYSTEM NEEDS NO INTERNAL CARD

O₂ (24, 28, 32 and 40* pins)

2708, 2758, TMS2716*, 2716
27C16, 2516, 2532*, 2564*
68764*, 68766*, 2732, 2732A
27C32, 2764, 2764A, 27C64
27128, 27128A, 27C128
27256, 27C256, 27512
27C512, 27513*
27011*, 27C011* (1 MEG)
27010, 27C010
27C1000, 27C1001
27C020, 27C2001 (2 MEG)
27C040, 27C4001 (4 MEG)



Shown with power pack & cable stored in case.

EEPROMS

2804, 2816A, 28C16
2817A*, 2864A, 28C64
28256, 28C256, 52B13*
52B33*

MicroControllers

8741A*, 8742*, 8748(H)*
8749(H)* 8751*, 87C51*
8752*, 8753*, 8744*
68705*, TMS7742

* ADAPTER REQUIRED
Diagrams included with manual
Assembled adapters are available

CONNECTS TO YOUR SYSTEM™

PARALLEL PRINTER PORT

FAST EASY TO USE

EXPANDABLE FLEXIBLE DESIGN

READS AND CONVERTS INTO HARDWARE PROTECTED AG NO SOFTWARE INSTALLATION

WORKS WITH ANY DESK OR LAPTOP MACHINE

SUPPORTS CURRENT AND FUTURE DEVICES

FORMATS INCLUDING WORD AND BINARY FILES

SPLIT & DOUBLE WORDS

DETECTIVE AND INCORRECTLY INSERTED DEVICES

PROGRAM NECESSARY - NO SELF CONFIGURING

OR LAPTOP MACHINE

AND FUTURE DEVICES

SPLIT & DOUBLE WORDS

DETECTIVE AND INCORRECTLY INSERTED DEVICES

PROGRAM NECESSARY - NO SELF CONFIGURING

SYSTEM SOFTWARE COMMANDS

<ul style="list-style-type: none"> • PROGRAM EPROM(S) FROM DISK FILE • READ DISK FILE INTO BUFFER • READ EPROM(S) INTO BUFFER 	<ul style="list-style-type: none"> • SAVE EPROM(S) OR BUFFER TO DISK • PROGRAM EPROM(S) FROM BUFFER • COMPARE EPROM(S) WITH BUFFER 	<ul style="list-style-type: none"> • COPY EPROM(S) • VERIFY EPROM ERASED • SELECT BUFFER EDITOR • SELECT DEVICE TYPE • DEVICE CHECKSUM • SET BUFFER (0, 1, 2, 3)
--	---	--

PLUS AN INTEGRATED BUFFER EDITOR WITH 18 BYTE LEVEL COMMANDS

SYSTEM INCLUDES: PROGRAMMING UNIT, POWER PACK, CONNECTING CABLE, OPERATION MANUAL & SOFTWARE **\$289**

SOFTWARE AVAILABLE ON 1/2" OR 5 1/4" DISK. (PLEASE SPECIFY)
CALL ABOUT OPTIONAL ADAPTERS - A SOFT TRAVEL CASE IS AVAILABLE FOR \$10.00

TO ORDER SEND CHECK, MONEY ORDER, WRITE OR CALL:

VISA

ANDRATECH
P.O. BOX 222
MILFORD, OHIO 451.50

(513) 831-9708
FAX (513) 831-7562

MASTER CARD

ADD \$5.00 FOR SHIPPING ADD \$4.00 FOR C.O.D.

WRITE FOR MORE INFORMATION OR CALL AND LISTEN TO OUR "TALKING" DATA SHEET

FEATURE ARTICLE

Larry Duarte

Add a Video Display to Your 8031 Microcontroller

Graphics and Color Liven Up Any Output

In many control applications, it is often necessary to display information of some form for human consumption. Most solutions have centered on one- or two-line LCD displays because of their small size and low cost. However, with the right interface, it is almost as easy and inexpensive to design into your circuit an 80-character by 25-line or 40-character by 25-line video display which uses either monochrome or color. Video displays give you much more room to display text and graphic information, and color liven up any screen.

The design I present here can be used with most basic 8031-based circuits, and requires minimal software and hardware overhead. Since 8031-type circuits have been presented many times in the past within the pages of this magazine, I'm only including those portions of the 8031 circuit necessary to clarify the discussion (see Figure 1a). The video interface could also be adapted to other processors with a few changes.

THE NCR 72681 CGMA

The heart of the video interface is NCR's 72C81 CGMA (Color Graphics and Monochrome Adapter). There are lower priced alternatives, but its features and level of integration made it

our choice. The chip includes full compatibility with IBM CGA and MDA, Hercules, and high-definition CGA. It has an internal 6845 and character-generator ROM, and requires just two RAM chips and a clock to be fully functional. In this article, the 72C81 is configured for monochrome CGA; that gives us the ability to switch between 80-column and 40-column modes, which is very useful for small monitors.

The 72C81 comes in an 84-pin PLCC package. The pinout is shown in the schematic in Figure 1b.

ADDRESS DECODING

The 72C81 was intended for use in IBM-compatible display adapters, and



Adding a video display interface to your microcontroller design is straightforward to do using NCR's 72C81 CGMA chip and allows the use of graphics and color to quickly improve any user interface.

as such uses standard CGA addresses for its I/O ports and video buffer and includes 20 bits of address input. The 8031 does not support separate I/O and memory addressing, but does support two separate 64K areas known as Code space and Data space. In my setup, the system EPROM is mapped in Code space (selected by *PSEN) while RAM and I/O share Data space (selected by *RD and *WR).

"Glue logic" was necessary to convert between the reduced 8031 address space and the much larger 72C81 space. In Figure 1, the 74LS138 breaks up the 8031's Data space into eight blocks of 8K each. The fourth block—24K to 32K—is reserved for I/O addressing. Reads from or writes to this area generate *IOR and *IOW through the 74LS32. The bottom 2K of the I/O block is further broken up by the 74LS154 into 16 blocks of 128 addresses each. A memory map for this sample system is shown in Figure 2.

The 74LS244s are used to move the 72C81 I/O port addresses out of the lower RAM area and adapt the 8031 16-bit address to the 20-bit address bus of the 72C81. They perform the following translation:

	<u>8031</u>
Video Buffer	8000-B000
I/O Ports	63D4-63DC

	<u>73C81</u>
Video Buffer	B8000-BB000
I/O Ports	3D4-3DC

Thus, to display information you only have to write to Data space between addresses 8000h and B000h. To deal with the display's control registers, write to addresses between 63D4h and 63DCh. Dealing with the display's control registers is the **same as** dealing with a standard IBM PC CGA board, so I won't go into that here. There are plenty of excellent references available that describe the CGA in detail.

THE MISSING WAIT STATE

While documentation NCR provides for the 72C81 is very good, it fails to adequately stress the fact that the IORDY output must be hooked up

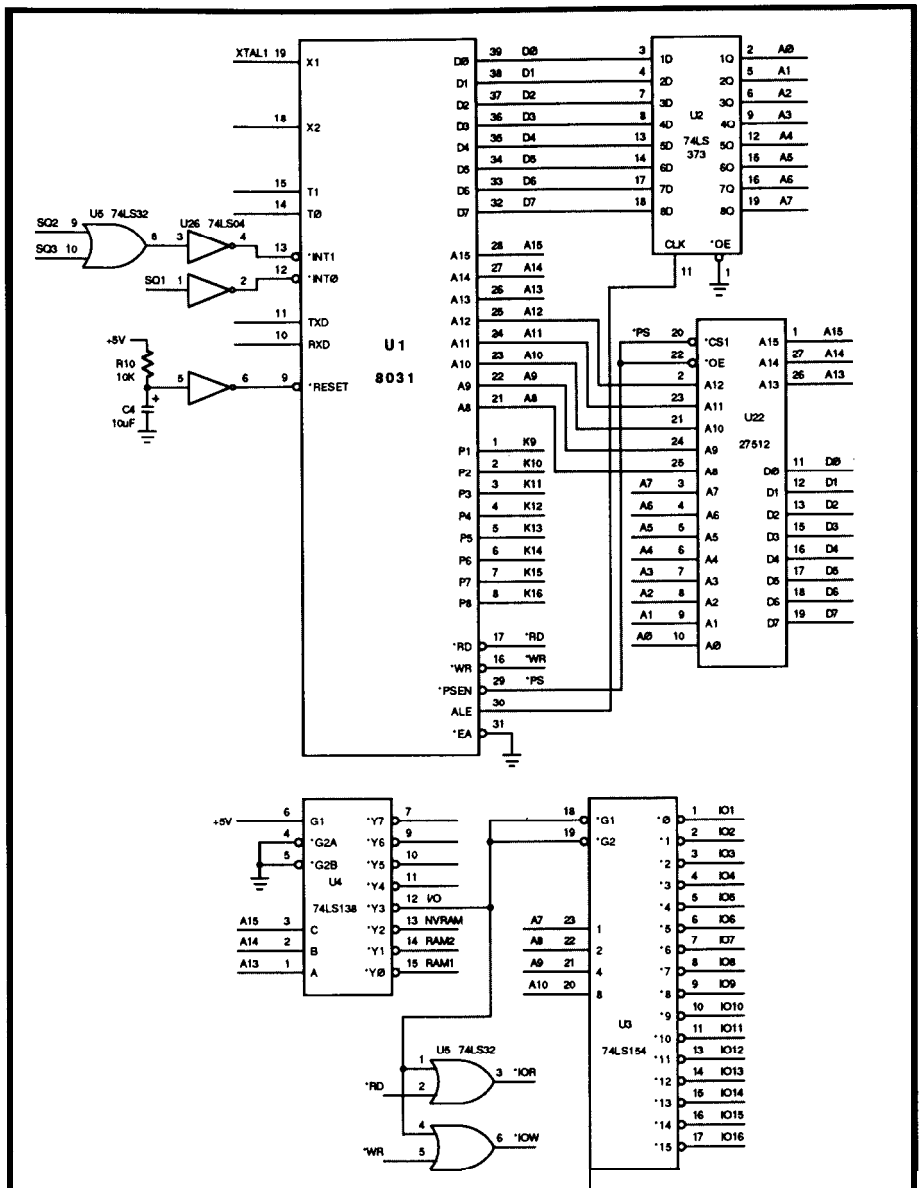


Figure 1 a—Since 8031-type circuits have been presented many times in the past, only those portions of the schematic necessary to clarify the discussion are shown here.

for the chip to operate properly. The 72C81's internal 6845 places a high priority on displaying RAM (so that the CGA's famous "snow" doesn't flurry across the screen) and sometimes has to delay writes to the video buffer. Since the 8031 doesn't have any kind of "wait" or "ready" input, connecting IORDY presents something of a problem. If we ignore IORDY, we'll likely lose information that was sent to the display, but ignored by the 72C81 since it was busy doing other things.

The solution involves using the CMOS 80C51 or 80C31. These chips use static memory for the internal CPU registers, rather than dynamic as in most NMOS processors, allowing us

to stop the processor clock without losing the registers. We also don't use the internal oscillator circuit of the CPU. Instead, a clock is generated externally (in our case using a 7404) and is used to drive a 74LS74 and a 74LS32. While the IORDY line is low, the clock is passed through to the processor. When IORDY line goes high, the clock is shut off, stopping the processor in its tracks until the video processor is ready to continue. The 74LS74 guarantees that turning the clock on and off is synchronized.

SOFTWARE

Listing 1 shows a very simple interface to the video display. Its main

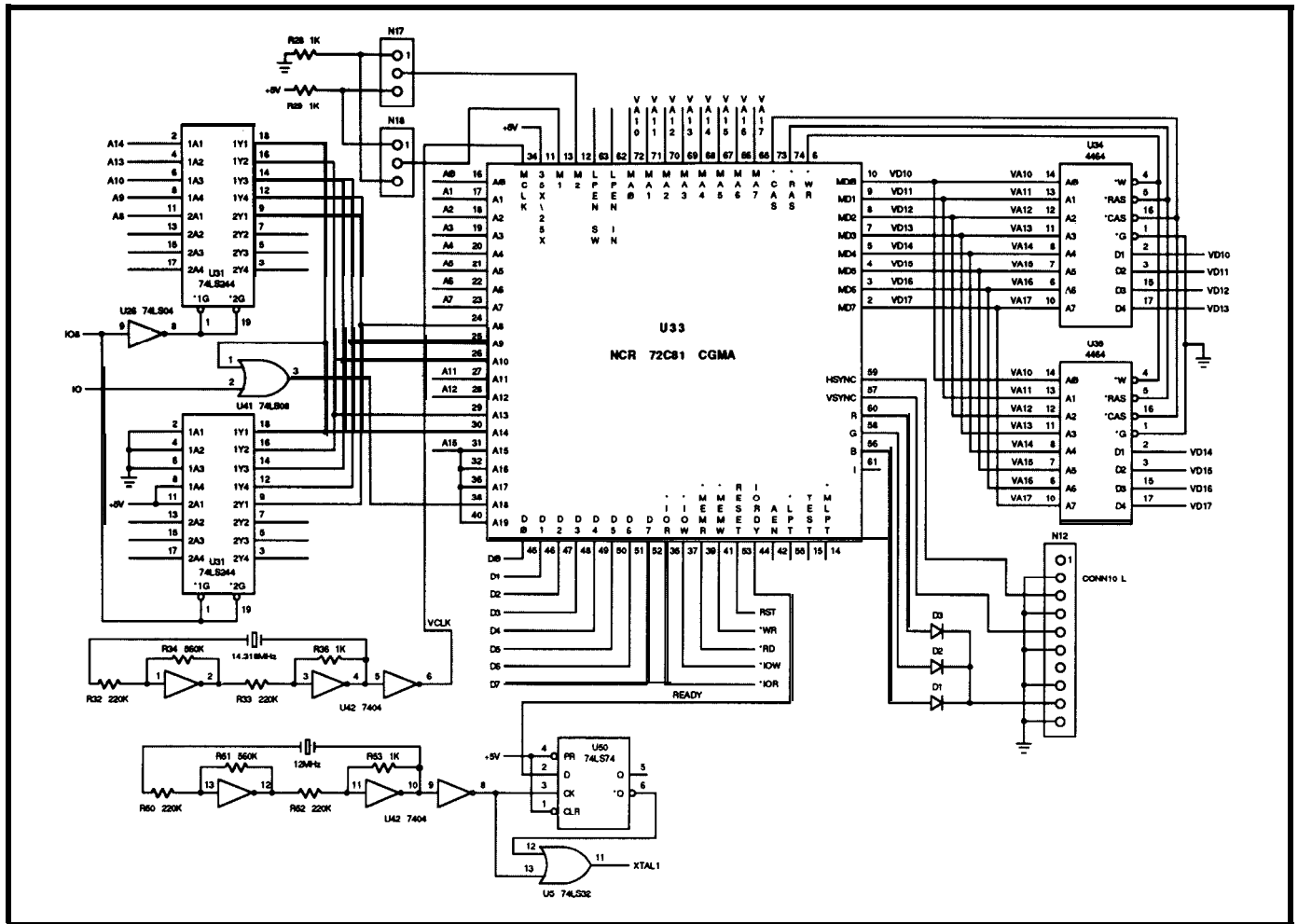


Figure 1 b—The heart of the display interface is the NCR 72C81 CGMA chip. While intended to be used to make graphics boards for IBM PC-compatible computers, it is readily adapted for use by many embedded microcontrollers.

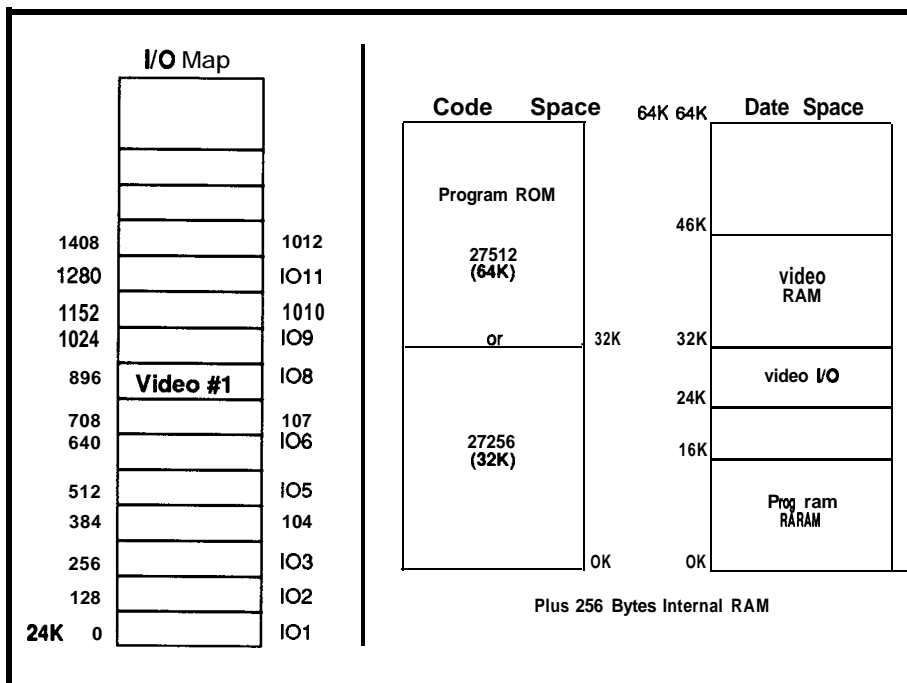


Figure 2—The 8051's address space is broken into two regions: read-only Code space and read/write Data space.

purpose is to initialize the display and write "Hello, world" to the screen. It is written in C and can be compiled to ROMable code using the Franklin C cross-compiler for the 8031 (or anyone else's compiler with some minor modifications to the source). A few additions to this code should make it useful for any application.

CONCLUSION

The circuit board components can be purchased for about \$35.00. For control applications, a small 5-inch to 9-inch monochrome monitor (color frequency, 12-volt supply) can be bought for \$25.00 from surplus houses.

The ability to add a video display will be useful for many 8031 projects. NCR also has a similar chip that supports VGA graphics. The NCR 77C22 supports the standard VGA modes while providing enhanced modes such

```

/* TEST VIDEO PROGRAM */
#include <reg51.h>
/* special address method */
#define XBYTE ((unsigned char *) 0x20000L)

#define VLIREG XBYTE [0x63D4] /* video 6845 address reg. */
#define VLDREG XBYTE [0x63D5] /* video 6845 data register */
#define VLMODE XBYTE [0x63D8] /* video mode register */
#define VLCOLOR XBYTE [0x63D9] /* video color register */
#define VLBUFF 0x8000 /* video frame buffer */

unsigned char let, curx, cury;
unsigned short pos;

movcur() /* MOVE CURSOR TO POSITION X,Y */
{
    unsigned char high, low;
    unsigned short x;

    pos = (unsigned short) (curx + (unsigned short) (cury*80));
    high = (pos / 256);
    low = (pos - (high * 256));
    VLIREG = 0x0E;
    VIDREG = high;
    VLIREG = 0x0F;
    VLDREG = low;

scrn80() /* SET SCREEN TO 80 COL */

    VLMODE = 0x01; VLIREG = 0x00;
    VLDREG = 0x71; VLIREG = 0x01;
    VLDREG = 0x50; VLIREG = 0x02;
    VLDREG = 0x5A; VLIREG = 0x03;
    VLDREG = 0x0A; VLIREG = 0x04;
    VLDREG = 0x1F; VLIREG = 0x05;
    VLDREG = 0x06; VLIREG = 0x06;
    VLDREG = 0x19; VLIREG = 0x07;
    VLDREG = 0x1C; VLIREG = 0x08;
    VLDREG = 0x02; VLIREG = 0x09;
    VLDREG = 0x07; VLIREG = 0x0A;
    VLDREG = 0x06; VLIREG = 0x0B;
    VLDREG = 0x07; VLIREG = 0x0C;
    VLDREG = 0x00; VLIREG = 0x0D;
    VLDREG = 0x00; VLIREG = 0x0E;
    VLDREG = 0x00; VLIREG = 0x0F;
    VLDREG = 0x00;
    VLMODE = 0x29;
    VLCOLOR = 0x00;
    curx = 0;
    cury = 0;
    movcur();

clrscrn()
{
    unsigned char a, b;
    unsigned short c;

    c = 0x8000; /* START OF VIDEO BUFFER */
    for(a = 0; a != 25; at+)
        for(b = 0; b != 80; b++)
        {
            XBYTE [c] = 32; /* SPACE */
            c++;
            XBYTE [c] = 15; /* NORMAL ATTRIBUTE */
            c++;
        }

    currigh() /* MOVE CURSOR RIGHT */
    {
        unsigned char a;
        if (cury == 24 && curx == 79)
            return;
        if (curx == 79)
        {
            curx = 0;
            cury++;
            movcur();
            return;
        }
    }
}

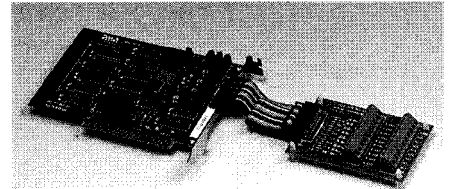
```

(continued)

listing 1-Most of the code necessary to drive the display is used to initialize the controller chip. Once set up, displaying text is straightforward.

Data Acquisition Solutions

Reliable, Affordable and Flexible



PCL-711S 12 Bit A/D + D/A \$295

- 8 single-ended 12-bit analog inputs, 2.5 KHz
- 1 D/A, 16 digital inputs, 16 digital outputs
- Wire terminal board and cable included

PCL-812 A/D + D/A + DIO \$395

- 16 single-ended 12-bit analog inputs, 30 KHz
- A/D with multiple ranges, DMA, interrupt
- 2 D/A, 16 D/I, 16 D/O, counter, pacer
- QBASIC driver, utility software and manual

PCL-818 High Speed A/D \$875

- 16 S.E. or 8 differential 12-bit A/D, 100 KHz
- Programmable ranges, auto channel scanning
- A/D with direct I/O, interrupt or DMA
- 2 D/A, 16 D/I, 16 D/O, counter, pacer
- Supported by most popular DAS software

PCL726 6 Ch. 12-Bit D/A \$495

- 6 12-bit analog outputs, voltage or 4 - 20 mA
- 16 digital inputs and 16 digital outputs, utility

PCL-722 144 Ch. Digital I/O \$345

- 144-channel digital I/O with interrupt
- Emulating 8255 mode 0, buffered circuits for high driving capacity, I/O dir. programmable
- Compatible with OPTO-22 I/O racks

PCL-720 DIO and Counter \$175

- 32 D/I, 32 D/O, 3 programmer counters
- User selectable counter clock source
- Breadboard area for customized circuits

Signal Conditioning and more . . .

- Wiring terminal, isolated D/I, relay output, relay driver, scanner, multiplexer, amplifier
- Interface for thermocouple, RTD, strain gage
- IEEB-488, RS-232/422/485, stepping motor
- Industrial PC chassis, 286/386/486 CPU cards, RAM/ROM disks, PC-bus card cage



Free 120-page reference guide for your system and OEM needs

Industrial & Lab Automation with PCs
ADVANTECH

1310 Tully Rd., Suite 115
 San Jose, CA 95122, USA
 (408) 293-6786, FAX (408) 293-4697
 Reader Service # 107

October/November 199 143

```

    curx++;
    movcur();
}

scrnprt() /* PRINT CHAR TO SCREEN */
{
    unsigned short realpos;
    realpos = ((2 * pos) + 0x8000);
    XBYTE [realpos] = let;
    realpos++;
    XBYTE [realpos] = 15;
    curright();
}

printf(s) /* SIMPLE STRING PRINT FUNCTION */
char *s;
{
    char *p = s;

    movcur();
    while(*p != '\0')
    {
        let = *p;
        scrnprt();
        p++;
    }
}

main()
{
    unsigned short x;

    for(x = 0; x != 10000; x++) /* DELAY FOR UNIT TO
STEADY */
    ;
    scrn80();
    clrscrn();
    printf("Hello, World!");
}

```

listing 1 -continued

as 640 x 480 with 256 colors and 800 x 600 with 16 colors. The only problem is going to be its packaging: a 160-pin surface-mount quad flat pack doesn't lend itself to easy prototyping. For now, and for most control applications, the 72C81 will be quite adequate. ❖

CONTACT

NCR Microelectronics Division
 1635 Aeroplaza Dr.
 Colorado Springs, CO 80916
 (800) 5252252
 (303) 596-5611

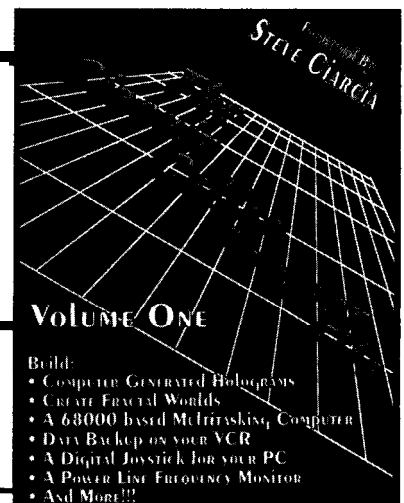
Larry Duarte has designed hardware, firmware, and software for a number of companies, doing applications ranging from cryptographics to point of sale.

IRS

407 Very Useful
 408 Moderately Useful
 409 Not Useful

The Circuit Cellar Project File, Volume 1 has page after page of new or expanded hands-on projects and tutorials. Circuit Cellar INK's editors have chosen a dozen of the top projects from the Circuit Cellar Design Contest, independent submissions, and top-response articles to make a book with something for every interest. You'll get projects about:

- A VCR Data Backup Card
- A Digital Joystick Port for Your PC
- A Power-line Frequency Monitor
- Computer-Generated Holography
- An LCD Namebadge
- A Car-computer Diagnostic Tool
- An LCD Tester
- Fractals
- Front-Door Light Control
- An Electronic Combination Lock
- Building & Debugging WOO Multitasking Computers



topped off with a foreword by Steve Ciarcia!
 All of this is yours for **only \$24.95** (includes domestic delivery)
 Order your copy today!

The Circuit Cellar Project File, Volume 1
 4 Park Street • Vernon, CT 06066
 Tel: (203) 875-2199 • Fax: (203) 872-2204

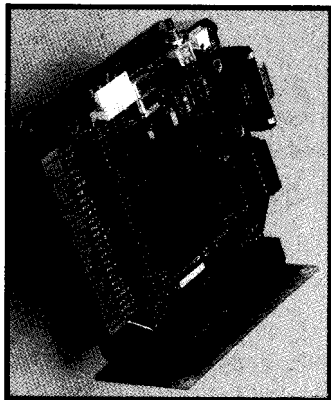
*\$24.95 Visa, Mastercard, Check, or Money Order
 (Add \$2.00 for delivery to Canada or Mexico, \$4.00 for delivery to other non-U.S. addresses)

See this book at The Embedded Systems Conference Booth #8

**NOW
 SHIPPING!**

The MICON-196KC "16-BIT Personal Controller"TM

COMPLETE DIGITAL CONTROL SOLUTION - HARDWARE & SOFTWARE DEVELOPMENT KIT



MICON-196KC "16-BIT PERSONAL CONTROLLER"TM comprises of two parts (kits)

- The HARDWARE DEVELOPMENT KIT contains:**
 - CPU MODULE (3.5"x3.5") with 80C196KC processor (16 MHz) has 8 ADC channels 10Bit resolution, 3 Pulse-Width-Modulated outputs (DAC), one DMA channel, 6 High-Speed Output channels, 4 High-Speed Capture Input channels, 40 Digital I/O ports, one full duplex RS-232 serial port;
 - MEMORY MODULE (3.5"x3") with 64K Memory space RAM and/or EPROM;
 - Z-Connectors BUS MODULE;
 - PROTO MODULE (3.5"x3.5") with a 3-Connectors Bus Module;
 - POWER SUPPLY & 6FT BS-232 COMMUNICATION CABLE
- The SOFTWARE DEVELOPMENT KIT contains:**
 - PC based SYSMON - System Monitor for software development phase;
 - 80C196 Machine Language Assembler;
 - User's Guide with HARDWARE SCHEMATICS and APPLICATION DEMO PROGRAMS.

COMPARE US AGAINST THE COMPETITION

FEATURES:	MICON-196KC	COMPETITION
MODULAR ARCHITECTURE	YES	NO
PROTOTYPING MODULES	YES	NO
COMPACT FOOTPRINT	YES	NO
80C196 Assembler, SYSMON (System Monitor)	INCLUDED	PAY EXTRA
Power Supply & Communication Cable	INCLUDED	PAY EXTRA

THERE IS NO COMPARISON !

The MICON-196KC "16-BIT PERSONAL CONTROLLER"TM sells for: \$245.00

Quantity & students discount. P.O., Visa & MC accepted. Immediate delivery.

MICONA Corporation
MICON Division

1885 Surveyor Ave. Bldg. 102
SIMI VALLEY, CA 93063

TEL: (805) 522-9444
FAX: (805) 522-9779

Reader Service #172

BTK52 BASIC-52 TOOLKIT

The BTK52 is an intelligent front end for program development on the MCS BASIC-52 CPU. It reduces 8052 program development time substantially and can be used with any MCS BASIC-52 based target system. The BTK52 runs on any IBM-PC/XT or compatible.

- Program download from PC host to target
- Program upload from target to PC host
- BASK program renumber utility, with "from," "through," "start," and "increment"
- Full screen program editing
- Single line editing with automatic error line number detection
- Full on-line help facility
- Transparent, adaptive line compression for full input line buffer utilization
- All functions accessible with only one keystroke from the terminal emulator
- \$125

BXC51805118052 BASIC COMPILER

- Fully compatible with code written for MCS BASIC-52 interpreter
- Now with integer, byte and bit extensions for code that runs more than 50 times faster than the MSC BASIC-52 interpreter
- Full floating point support
- In-line assembly language option
- Compile time switch to select 8051/8031 or 8052/8032 CPUs
- Includes Binary Technology's SXA-57 cross-assembler and Hex file manipulation utility
- Compatible with any RAM or ROM memory mapping
- Runs on IBM-PC/XT or compatible
- \$295

603-469-3232 • FAX 603-469-3530



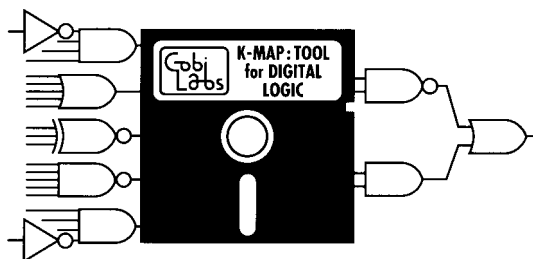
Binary Technology, Inc.

Main Street . P O Box 67 . Menden NH 03770



Reader Service #115

DIGITAL DESIGNERS MINIMIZE YOUR LOGIC



KARNAUGH MAP SOLVER FOR PC'S
Minimum Gates from Your Specs

- Combinational, Sequential
- Mealy & Moore State Machines
- 2 to 14 Inputs, Any Number Outputs
- Sum of Products, Product of Sums
- JK, D, SR, and T Flip Flops
- Expert Reference and Tutorial
- 60 Day Money Back Guarantee
- MUCH MORE

SAVE 40% : **\$110** + \$5 (P&H)

GOBI LABS: (407) 297-0862
BOX 616304, ORLANDO, FL 32861-6304

Reader Service #148

FEATURE ARTICLE Part 2

Steven E. Strauss
& P.K. Govind

ISDN (S/T) Interface

Design Example of a PC Plug-in Board

In Part 1, we discussed the key elements of the ISDN (2B+D) Basic Rate Interface (BRI) standards. Now we present a design example of an integrated voice and data communication circuit which supports the CCITT I.430/ANSI T1.605 BRI standard. The design uses AT&T's family of ISDN devices and includes an interface to an IBM PC host computer. The hardware was built on an XT-style card and installed in a spare expansion slot. The application software contains Layer 2 (I.441/Q.921) and Layer 3 (I.451/Q.931) D channel call management functions conforming to the AT&T 5ESS central office switch.

HARDWARE HIGHLIGHTS

- Line interface support and (2B+D) multiplexing is provided by the T7250B line transceiver for terminal endpoints. D channel operation is handled using the 16-byte internal FIFO buffers.

- B channel voice port support with speakerphone functionality is provided via the T7540 digital telephone CODEC.

- B channel data port support with HDLC or non-HDLC protocols is provided by the T7121 synchronous data formatter which has 64-byte buffers in the transmit and receive paths.

SOFTWARE HIGHLIGHTS

- *Menu-driven operation to exercise each device separately, with access to internal registers.

- Call setup for either voice or data conversation on any B channel.

- Display of Layer 2 and Layer 3 flow control parameters for D channel traffic.

- File transfer exercise on a chosen B channel.

HARDWARE DESIGN

A block diagram of the BRI adapter board is shown in Figure 1. It is subdivided into five sections. We will begin with a quick summary of each section and then move on to a more detailed description of the individual sections.

T7540 are programmed via the PC bus interface.

2. **Data Port.** Data transfer on the B2 channel is supported by the T7121 HDLC formatter. It connects to the PC bus interface and transfers data to and from computer memory. It also connects to the T7250B for serial data exchange on the B2 channel.

3. **2B+D Framer and D channel Signaling Port.** The T7250B provides the user-to-network interface for ISDN connectivity. D channel processing is provided via the embedded HDLC formatter of the T7250B, which has 16-byte FIFO buffers in both transmit and receive directions. This architec-

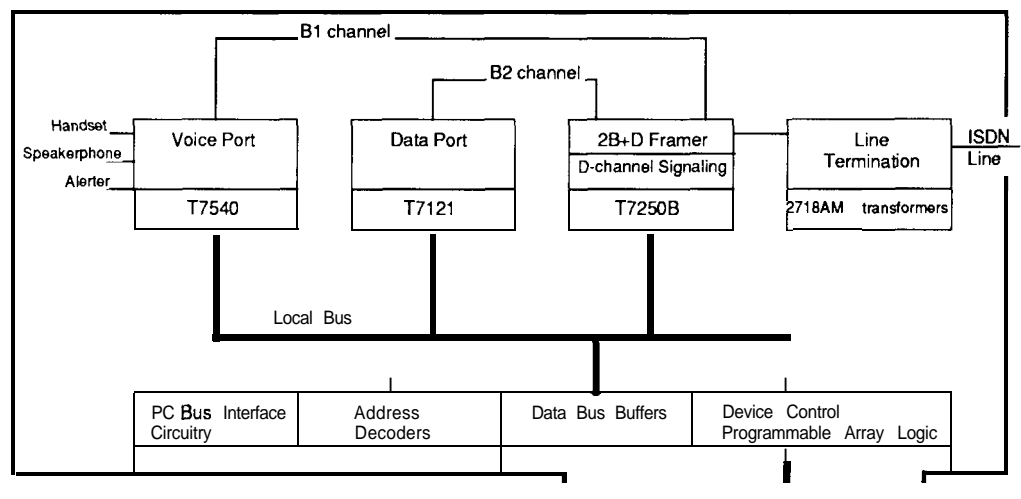


Figure 1—A *simplified* block diagram of the (2B+D) BRI adapter board for the IBM PC.

1. **Voice Port.** Voice access is provided by the T7540 digital telephone CODEC. It connects to the user-provided handset, a speakerphone, and an auxiliary alerter via 4-pin RJ-11 modular jacks. It transfers digitized voice to the T7250B on the B1 channel. The operational parameters of the

structure simplifies the software implementation of LAPD (Link Access Procedure for the D channel), which is used to set up B channel voice and data calls.

4. **Line Termination.** Line transformers, protection circuitry, and resistor circuitry are contained in this

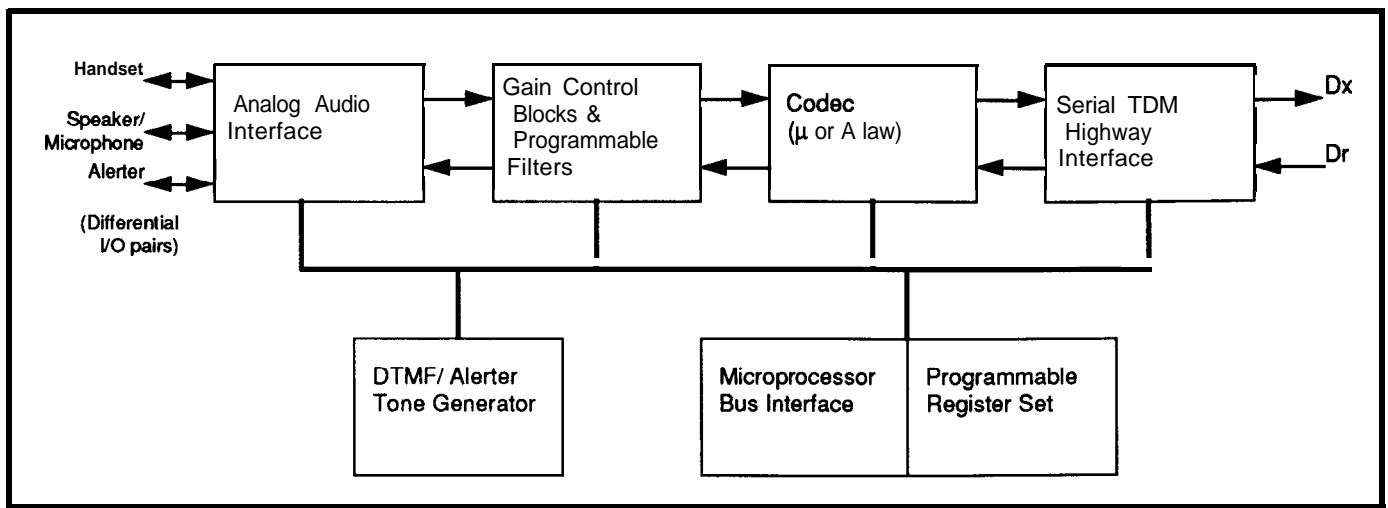


Figure 2-A simplified block diagram of the AT&T T7540 digital telephone CODEC.

section. This circuit is connected to the T7250B line transceiver section. The physical interface for ISDN basic access is provided via an &position RJ-45 modular jack.

5. *PC Bus Interface Circuitry.* The BRI adapter board is mapped into the I/O space of the PC with address locations ranging from 200h to 3FFh. Data bus buffering between the local bus and the PC bus is provided by a bidirectional bus transceiver. Address decoding for chip selects and device control signals are also provided in this section.

VOICE PORT WITH SPEAKERPHONE FUNCTIONALITY

The AT&T T7540 digital telephone CODEC (Figure 2) provides the flexibility needed to support a variety of analog voice ports. Audio functions include a CODEC-filter featuring μ /A-Law companding. Additional voice port functions include programmable touch-tone (DTMF) generation and ringer tone generation. The device also contains a programmable sidetone insertion interface. The T7540 is controlled via an external microprocessor.

The T7540 has a serial interface for digitized voice, a parallel interface for microprocessor control, and three pairs of differential analog audio interfaces to the handset, hands-free speakerphone, and auxiliary equipment such as answering machines or alerters.

The serial PCM interface for digitized voice is a full-duplex serial time-

division-multiplexed bus. This serial highway is programmable and accepts variable data rate clocks ranging in frequency from 64 kHz to 4.096 MHz. The serial highway interface connects directly to most PCM buses. In our design, the PCM interface connects directly to the T7250B ISDN S interface chip. The parallel microprocessor interface of the T7540 is flexible and is interfaced easily with Intel or Motorola microprocessors. In our adapter board design, the PC microprocessor (80x86) controls the transfer of information in and out of the programmable internal registers of the T7540.

The three analog audio interfaces of the T7540 are the handset, hands-free, and auxiliary interfaces. Each of these interfaces consists of a differential input and a differential output. The handset and auxiliary outputs can drive a 300- Ω load directly and are programmable from +0 to -23.25 dB in increments of 0.75 dB. The speaker output can drive a 50- Ω speaker directly and is adjustable over a 69-dB range in 1.5-dB steps.

The T7540 provides the functionality needed to support microprocessor-controlled hands-free speakerphone operation. The microprocessor monitors and controls the system in such a way that the coupling from the speaker to the microphone and poor hybrid matching does not result in oscillations, ringing, or unpleasant echoes.

The T7540 provides signal monitoring of the transmit and receive paths. Received signals may be ex-

amined over the entire channel bandwidth or an 800-Hz second-order high-pass filter can be selected. Speech tends to have a large part of its energy above 800 Hz, whereas room noise has a significant part of its energy below 800 Hz. Selecting the 800-Hz high-pass filter will reduce the need for gain switching due to noise.

To reduce the amount of microprocessor overhead required to monitor the voice signals, maximum-value registers are provided on the T7540 to obtain an envelope of the received and transmitted voice signals. These maximum-value registers retain the highest value of the, received signal since the last register read was performed. The maximum-value method can provide a reasonable representation of the signals for a period of several milliseconds. The time required between register reads depends on the hands-free algorithm developed by the user. Figure 3 shows how the T7540 is wired on the BRI adapter board.

DATA PORT INTERFACE

Data transfer on the ISDN B2 channel is supported by the T7121. It connects the serial communication link carrying High-level Data Link Control (HDLC) bit-synchronous data frames to the host PC. There is an optional transparent mode of operation in which no HDLC processing is performed, allowing the use of other protocols (Figure 4). The T7121 communicates with the PC as an I/O-mapped peripheral and is controlled

via programmable registers. The HDLC transmitter and receiver are each provided with 64 bytes of FIFO storage. The 64-byte buffer depth reduces the number of status polls, thereby reducing the number of interrupts to be processed by the PC.

The T7121 is directly connected to the T7250B in our design (Figure 5). This is made possible by the flexibility of the serial bit transport interface on the T7121. The T7250B produces all clocks and frame strobes necessary for the T7121 to properly function in a TE application.

2B+D FRAMER & D CHANNEL SIGNALING PORT

The network interface for ISDN (2B+D) access is handled by the versatile T7250B. This device conforms to the BRI specifications of the I.430/T1.605 standard, outlined in Part 1. Figure 6 shows how the T7250B is connected in our integrated voice and data application. The B1 channel is connected to the T7540 for voice transport. The B2 channel is connected to the T7121 for data transport. With this hardware arrangement, it is also possible to swap the B1 and B2 channels by programming the T7250B to internally exchange the B1 and B2 octets. The D channel processing does not require an external HDLC formatter since the T7250B formats the D channel information internally as it constructs the BRI (2B+D) frame. Sixteen-byte FIFO buffers allow efficient processing of D channel data by reducing the interrupt overhead for the PC.

A block diagram of the T7250B is given in Figure 7. We will briefly describe the functionally related groups of pins in Figures 6 and 7.

VDDT, TNR, REXT, and TPR belong to the line transmitter. TPR and TNR are the transformer connection points. REXT sets the transmitter output current and VDDT is a dedicated power pin for the line transmitter.

TXB1 and TXB2 are the inputs for the B channel information to be transmitted to the network.

CK6/XTALI and XTALO provide direct connections to a crystal. Alternatively, the CK6/XTALI pin may be

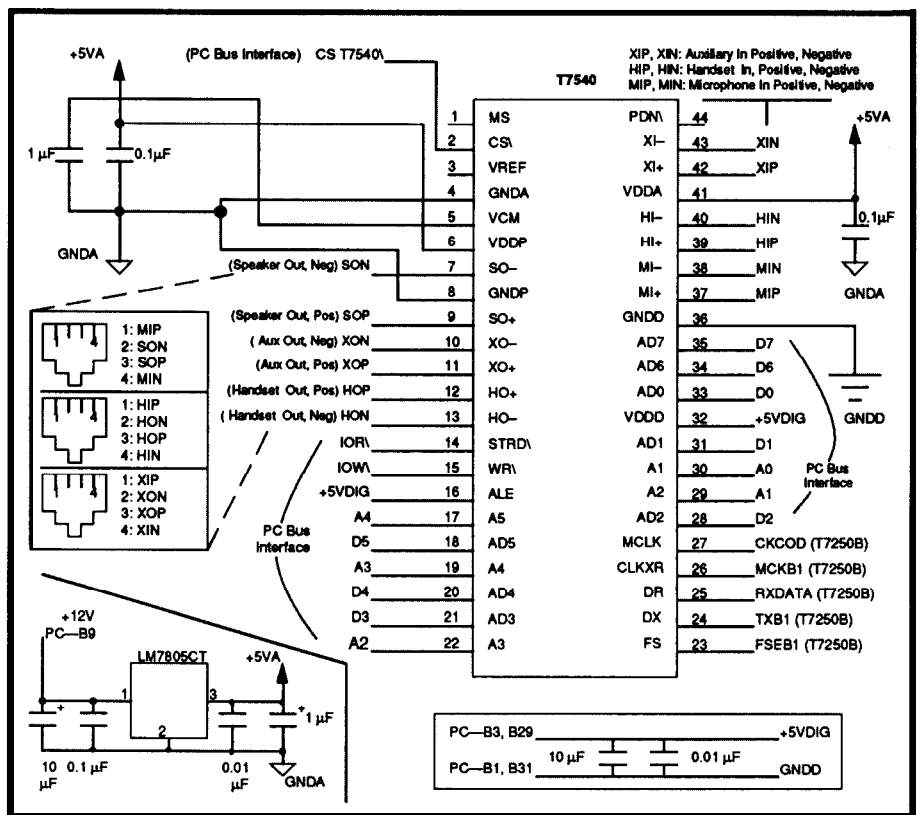


Figure 3—Circuit connections for the T7540 on the BRI adapter board.

driven by an external clock source at 6.144 MHz.

VDD, VSS1, and VSS2 provide power and ground to the device. VSS1 is the ground reference for input buffers and other internal logic. VSS2 is the ground reference for the output buffers. VDD is the power input for all the digital logic.

CS, RD, WR, AO-A3, DO-D7, RESET, and INT form the microprocessor interface (PC bus interface).

VSSR, RPR, RNR, VT, and VDDR are inputs to the line receiver. RPR and RNR are the transformer connections. The VT input allows an external decoupling capacitor to filter noise from the receiver's voltage reference level. Separate power and ground pins for the receiver, VDDR and VSSR, minimize noise problems.

RXDATA contains the 2B+D serial bit stream received from the network. CKCOD, MCKB1, FSEB1,

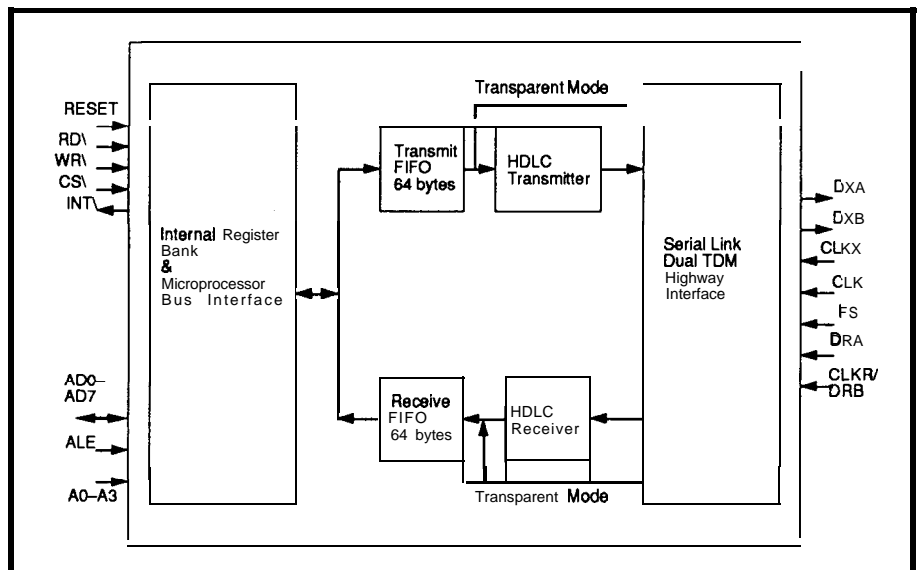


Figure 4-A simplified block diagram of the AT&T T7121 HDLC formatter.

**FAST COMPLETE
ACCURATE
DRAM TEST
DIPs - SIMMs - SIPs**



RAMSTAR

Ins. RESOLUTION

ACCESS SPEED VERIFICATION

80 ns. thru 180 ns. (Std.) \$249.00
45 ns. thru 110 ns. (Fast) \$349.00
4MEG Option Add \$ 89.00

AUTO-LOOP

Continuous Test 6.25 MBits/sec.

ADAPTERS:

SIMM/SIP ADAPTER \$189.00
Tests 64K, 256K, 1 M & 4M Devices
8 or 9 Bit versions.

NEW

NTX ADAPTER \$149.00
Tests 64 Pin Dual-Edge
Laserwriter Type SIMM's.

4 X ADAPTER \$ 89.00
Tests 64K & 256K By 4 Bit Devices

AC ADAPTER \$ 18.00
Regulated +5V @ 1Amp.

**FREE RAMFACTS
DRAM NEWSLETTER**

1-800-RAMSTAR

COMPUTERDOCTORS

9204-B Baltimore Boulevard
College Park, Maryland 20740

MADE IN U.S.A. U.S. PATENT No. 4,965,799

Reader Service #126

CKB2, and FSEB2 provide the programmable clock and strobe signals associated with the B1 and B2 channels. All clock and strobe signals are synchronized to the 2.048-MHz clock, CKCOD, to reduce CODEC noise problems. The RESET pin resets the T7250B and restores all the default values of its internal registers.

The T7250B has two register banks: the foreground bank of 16 registers (R0-R15) and the background bank of nine registers (BR1-BR9). The foreground registers provide most of the chip operation control and status information. The background registers provide additional functionality such as D channel address recognition, autoactivation, and parallel readout of a selected B channel.

An optional serial data transfer mode may be programmed to provide contiguous access to 18-bit (2B+D) information groups. This feature is useful for 144-kbps clear channel applications (e.g., compressed video) where the B and D channels are not channeled into 64-kbps and 16-kbps parcels. In our design, for the purpose of this article, we only use the standard ISDN channel-structured mode of operation.

The background registers are addressed the same way as the foreground registers. Functions provided by the background registers are accessed by setting three bits in R15 of the foreground bank. Even when the background registers are in use, the foreground registers R0 and R15 are always accessible. The foreground registers R1-R14 may be accessed by resetting the three bits of R15.

D channel HDLC operation is handled by a built-in queue manger. The user loads the LAPD bytes into

the 16-byte queue via R3. A transmit-frame-complete tag bit is then set in a control register after writing the last byte of data to the queue. The T7250B automatically handles HDLC framing and transmits the D channel data. Received data is automatically loaded

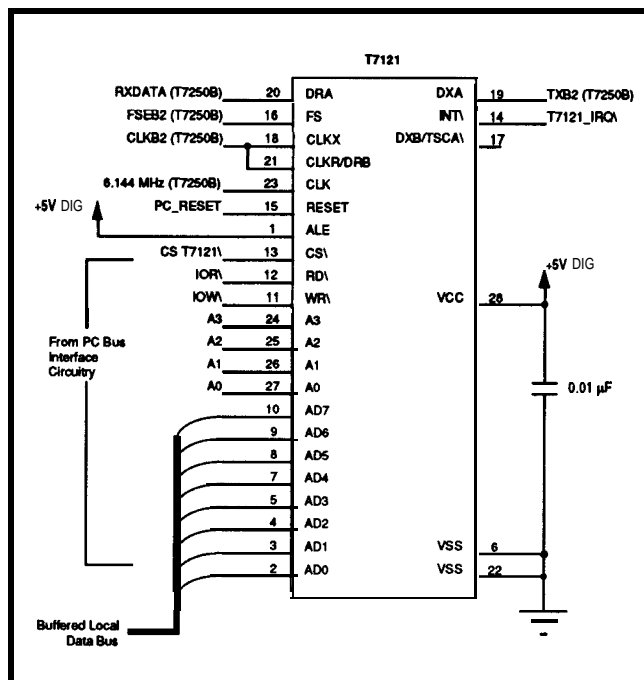


Figure 5— Circuit connections for the T7121 on the BRI adapter board.

into the 16-byte queue after HDLC processing. The queue manager also creates an end-of-frame status byte for each frame and stores it in the queue if an end-of-frame condition has been detected. Status information is also available in a separate register. The transmit and receive FIFOs may be programmed to specific fill levels and the T7250B may be instructed to interrupt the controlling microprocessor. The interrupts are also maskable to allow a polled mode of operation.

LINE TERMINATION

The T7250B meets the I.430/T1.605 line transceiver requirements at the ISDN S/T reference point when a transformer with a 2.5: 1 turn ratio is used. Our design (Figure 8) uses a pair of AT&T 2718AM transformers which have a very 'small leakage inductance and self capacitance. Transformer selection is a crucial element in meeting the line interface imped-

ance requirements of the BRI standard. The transformers also isolate the integrated circuits on the BRI board from line transients.

The resistor network in the transmit path and the resistor network in the receive path are used to meet the electrical requirements of the standard.

Line protection from static discharge transients varies from installation to installation. On our board, we use a simple protection scheme. DI-D4 and D5-D8 are BAT85 Philips Schottky barrier diodes. These diodes

have extremely low capacitance and exhibit a very low forward voltage drop when forward biased. They also have an integrated protection ring to protect against extreme static discharges. The protection circuitry shown here is very minimal, since our board is intended for a laboratory and test environment only.

Jumpers J3 and J4 are used for 100-R terminations of the receive and transmit loop, respectively. These jumpers should be in place when the board is being used in point-to-point configurations. These jumpers should

be removed if the 100-Q termination is included in the premises wiring for passive bus operation.

PC BUS INTERFACE CIRCUITRY

Figure 9 shows the PC bus interface circuitry. Our design uses a total of 96 PC I/O addresses, separated into two distinct blocks of 64 and 32 bytes each. The block of 64 bytes is used to decode the complete I/O space for the T7540 digital telephone CODEC and address its internal registers. The second block of 32 bytes is used to decode the I/O space for the T7250B and the T7121 HDLC data formatter, each of which contains 16 internal registers. Splitting the I/O addresses into two distinct blocks allows easy mapping to the available PC I/O address space. The user must be careful to use an I/O area which is currently not being used by the PC for other peripherals.

A 74LS245 8-bit bidirectional buffer driver (IC1, Figure 9) buffers the PC data bus. Note the PC data bus is brought directly to the buffer input pins (2-9). The buffered PC data bus, referred to simply as DO-D7, is routed to devices on the adapter board.

The enable signal for IC1 is derived from IC6, a Programmable Array Logic (PAL) device. The PAL is programmed to logically AND all the active-low chip select signals for the I/O-mapped peripheral devices located on the adapter board. If the ISDN interface adapter board is being accessed, IC6 outputs an active-low signal to the enable lead of IC1 (pin 19). This action takes IC1 out of the high-impedance tristate mode and enables its output buffer drivers. The data bus (DO-D7) is driven either by the PC or by one of the devices on the board.

The direction of the data flow is determined by the system read pulse, PC_IOR\ . When PC_IOR\ toggles to a logical low, the PC reads data from the selected device on the board. Consequently, data flows from the board to the PC. Otherwise, data flow is from the PC to the board. The default condition for data flow is from the PC

<h1>Cross-Assemblers</h1> <p>from \$50.00</p> <h1>Simulators</h1> <p>from \$100.00</p> <h1>Cross-Disassemblers</h1> <p>from \$100.00</p> <h1>Developer Packages</h1> <p>from \$200.00(a \$50.00 Savings)</p>																					
<h2>Make Programming Easy</h2> <p>Our Macro Cross-assemblers are easy to use. With powerful conditional assembly and unlimited include files.</p>																					
<h2>Get It Debugged--FAST</h2> <p>Don't wait until the hardware is finished. Debug your software with our Simulators.</p>																					
<h2>Recover Lost Source!</h2> <p>Our line of disassemblers can help you re-create the original assembly language source.</p>																					
<h2>Thousands Of Satisfied Customers Worldwide</h2> <p>PseudoCorp has been providing quality solutions for microprocessor problems since 1985.</p>																					
<h3>Processors</h3> <table border="0"> <tr> <td>Intel 8048</td> <td>RCA 1802,05</td> <td>Intel 8051</td> <td>Intel 8096,196kc</td> </tr> <tr> <td>Motorola 6800</td> <td>Motorola 6801</td> <td>Motorola 68HCII</td> <td>Motorola 6805</td> </tr> <tr> <td>Hitachi 6301</td> <td>Motorola 6809</td> <td>MOS Tech 6502</td> <td>WDC 65C02</td> </tr> <tr> <td>Rockwell 65C02</td> <td>Intel 8080,85</td> <td>Zilog Z80</td> <td>NSC 800</td> </tr> <tr> <td>Hitachi HD64180</td> <td>Mot. 68k,8,10</td> <td></td> <td></td> </tr> </table>		Intel 8048	RCA 1802,05	Intel 8051	Intel 8096,196kc	Motorola 6800	Motorola 6801	Motorola 68HCII	Motorola 6805	Hitachi 6301	Motorola 6809	MOS Tech 6502	WDC 65C02	Rockwell 65C02	Intel 8080,85	Zilog Z80	NSC 800	Hitachi HD64180	Mot. 68k,8,10		
Intel 8048	RCA 1802,05	Intel 8051	Intel 8096,196kc																		
Motorola 6800	Motorola 6801	Motorola 68HCII	Motorola 6805																		
Hitachi 6301	Motorola 6809	MOS Tech 6502	WDC 65C02																		
Rockwell 65C02	Intel 8080,85	Zilog Z80	NSC 800																		
Hitachi HD64180	Mot. 68k,8,10																				
<h3>New</h3> <p>Zilog Z8 Zilog Super 8</p> <ul style="list-style-type: none"> All products require an IBM PC or compatible. 																					
<p>For Information Or To Order Call:</p> <h1>PseudoCorp</h1> <p>716 Thimble Shoals Blvd, Suite E Newport News, VA 23606</p> <p>(804) 873-1947 FAX:(804)873-2154</p>																					

to the adapter board. Utilizing this scheme minimizes bus contention problems which can happen in memory- or I/O-mapped systems. IC2, a 74LS244 octal bus driver, buffers the PC address lines PC_A0-PC_A4 and the read and write (IOR\ and IOW\) bus control signals. These signals are used by the I/O-mapped peripheral devices, the T7540, T7121, and the T7250B

Interrupt control functions are provided using a 16L8 PAL (see IC 10, Figure 10). This logic has been included to expand the maskable interrupt ports to three prioritized interrupt ports. In our design, there are two interrupt sources. The interrupt inputs are T7250B_IRQ\ and T7121_IRQ\ . The T7540 does not have an interrupt pin. The interrupt inputs are multiplexed to present an active-high interrupt pulse to the PC via jumper J5.

When IC10 issues an interrupt to the PC, an I/O read qualifier signal,

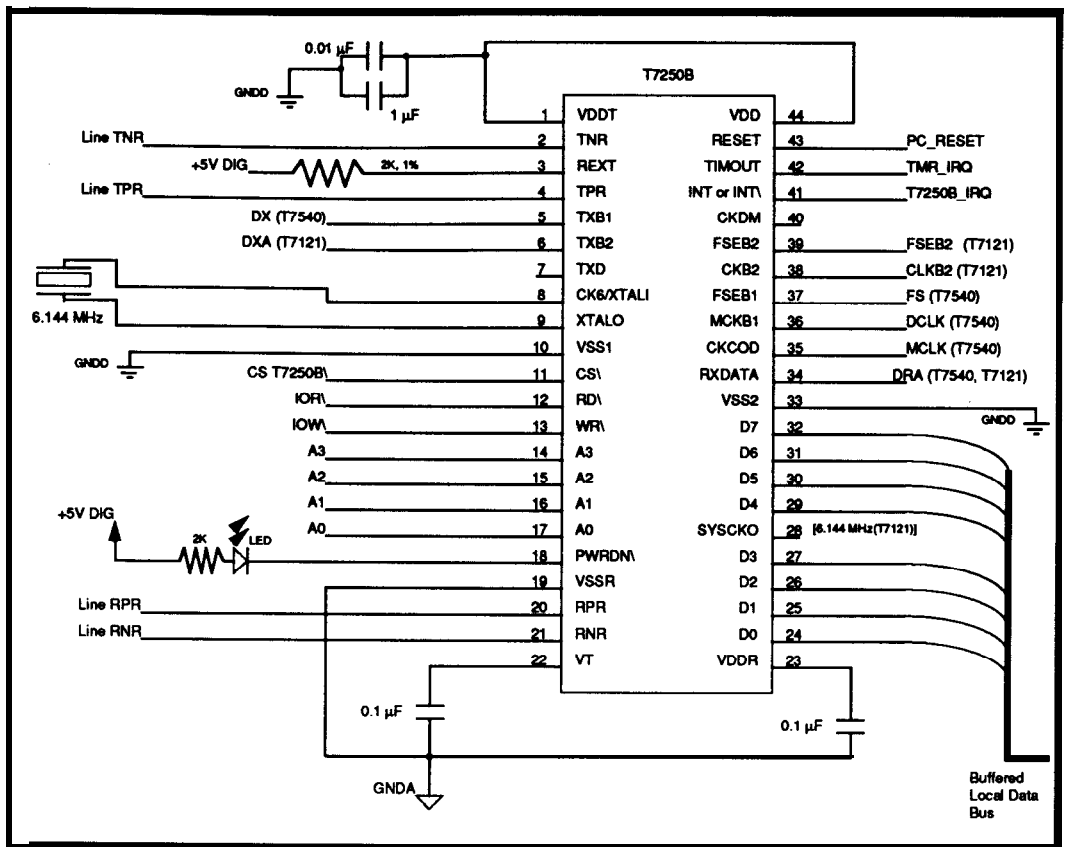


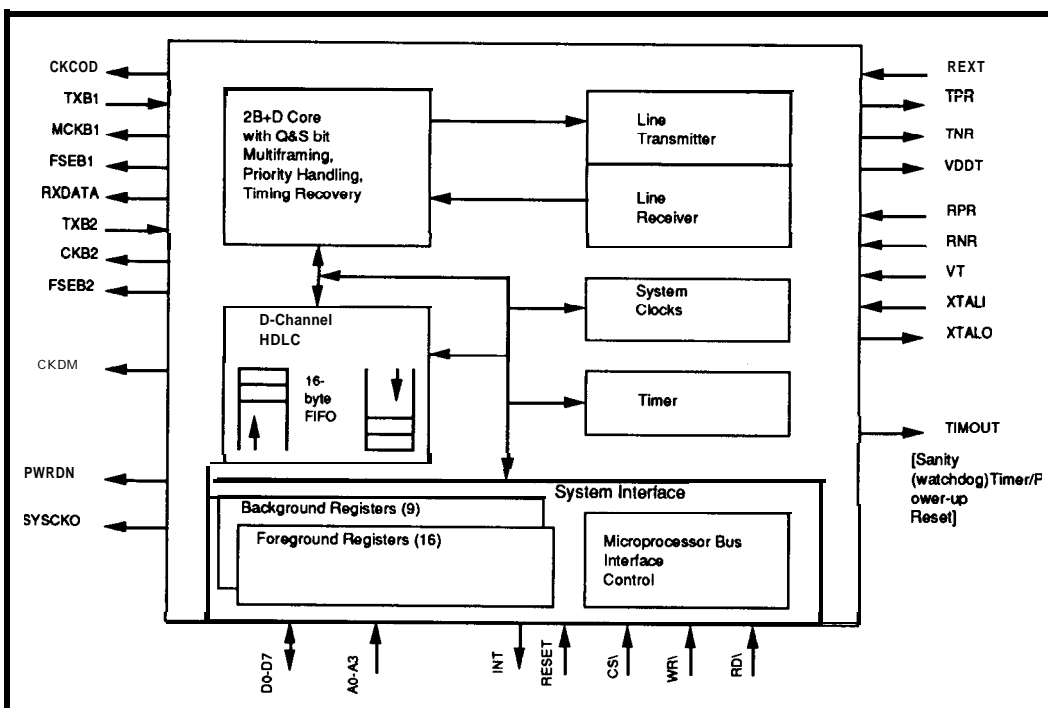
Figure 6-Circuit connections for the T7250B on the BRI adapter board

RD_STATUS\ , is asserted (active low). This causes IC10 to turn on its output pins 102-108 (pins 13-19). These outputs are directly connected to the buffered local data bus leads D6-D0. The PC examines IO4/D4 (T7121_IRQ) or IO3/D5 (T7250B_

IRQ) to identify the source of the interrupt. The lines IO5-IO8/(D3-D0) and IO2/D6 are masked by the interrupt handling software since they are unused in our design. Once the interrupt source has been identified, the interrupt output signal from IC10 clears, allowing further interrupts.

SOFTWARE DESIGN

The modular software is designed to support the call management functions for ISDN Layer 3 as defined in CCITT Recommendation I.451 (Q.931). It also supports Layer 2 flow control functions on the D channel as defined in the Recommendation I.441 (Q.921). The Layer 1 device drivers are specifically designed for ISDN TE applications using the AT&T' devices, the T7250B, the T7121, and the T7540. Layer 2 and Layer 3 software is an



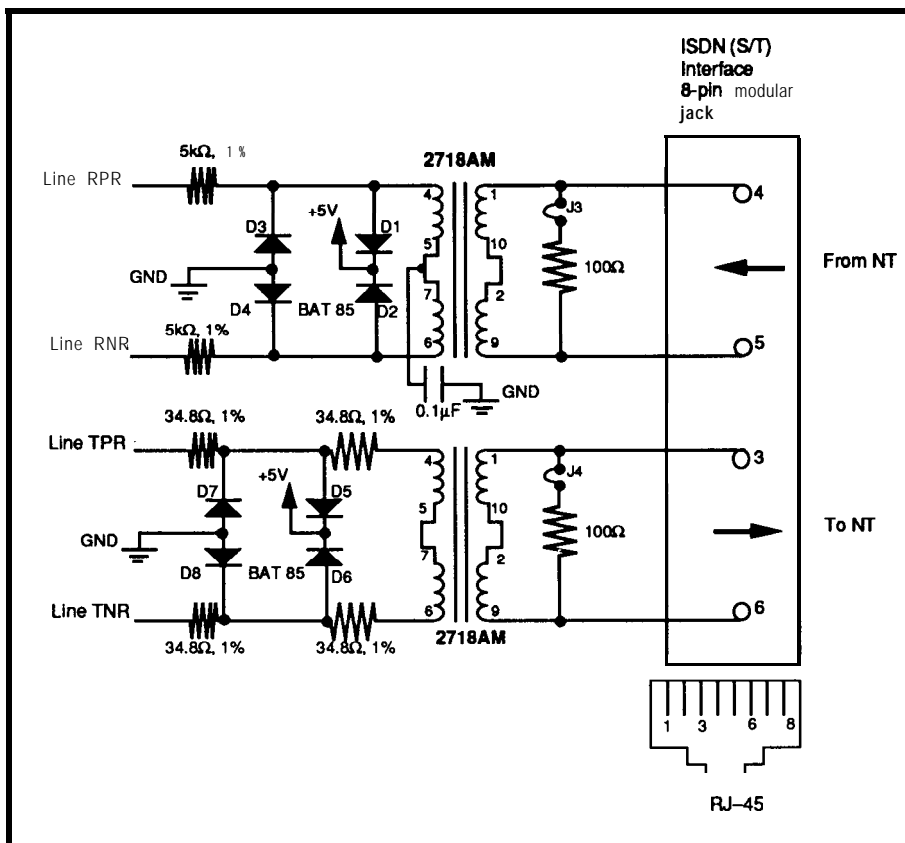


Figure 8—Line termination circuitry for the ISDN BRI board at the (S/T) reference point.

adaptation of the DGM&S ISP188 package designed for the MS-DOS environment of the PC. The general structure of the software modules is shown in Figure 11.

LAYER 1 DEVICE DRIVERS

The device driver modules control and monitor the operation of each device. The following functions are provided for the voice, data, and signaling ports:

- B channel control-T7540, T7121, T7250B
- B channel voice control-T7540 handset gain, filter selection
- B channel data exchange—T7121
- D channel operation-T7250B

The B channel control module allows access to voice port operations provided by the T7540, the data port operations provided by the T7121, and the channel selection provided by the T7250B.

The B channel voice control module provides volume control, DTMF generation, and alerting tones to the handset or the speakerphone connected to the T7540. In addition, bi-

nary ones may be sent to the T7540 to silence the CODEC when the handset is not in use.

The B channel data exchange module supports the functions provided by the T7121 HDLC formatter. Queued packets are transferred via the 64-byte-deep FIFO buffers in the T7121. If needed, the HDLC operation may be bypassed to carry user-defined bit synchronous protocols.

The module for D channel operation contains the following tasks:

- (a) transmit and receive LAPD frames from Layer 2
- (b) activate or detect activation at the (S/T) interface
- (c) deactivate or detect deactivation at the (S/T) interface
- (d) detect or report link error conditions to the ISDN TE state manager

The ISDN INFO state control module supports all activities at the bit transport level of the (S/T) interface by controlling and monitoring the line transceiver functions provided by the T7250B; for example, the transmission of INFO 1, INFO 3 and the reception of INFO 2, INFO 4 using the line transmission registers of the T7250B.

EPROM EMULATORS

An EPROM emulator appears as an EPROM to a target system. Instead of programming EPROMs, you simply download your code to the emulator. In seconds, you see results.

27256 EPROM EMULATOR



Emulates 2764, 27128, & 27255 EPROMs.

Plugs into target EPROM socket and connects to PC parallel port via telephone cable.

Loads Intel, Motorola, hex, and binary files.

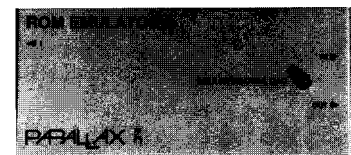
Reset outputs restart target after downloading.

Downloads 32K in 2 sec.
(12 MHz PC AT)

\$199

27010 EPROM EMULATOR

Up to 4 units can be daisy-chained to emulate consecutive EPROMs and to support 18 and 32-bit systems.



Emulates 2754, 27128, 27256, 27512, and 27010 EPROMs.

Plugs into target EPROM socket and connects to PC parallel port via telephone cable.

Reset outputs restart target system.

\$349



(916) 721-8217
Fax (916) 726-1905

Parallax, Inc. • 6200 Desimone Lane, #69A
Citrus Heights, CA 95621

Dealer/OEM Pricing Available

Prices subject to change without notice.
California residents add appropriate sales tax.
Add \$4.00 for UPS ground, \$7.00 for UPS 2nd day,
\$15.00 for UPS next day.

Reader Service #187

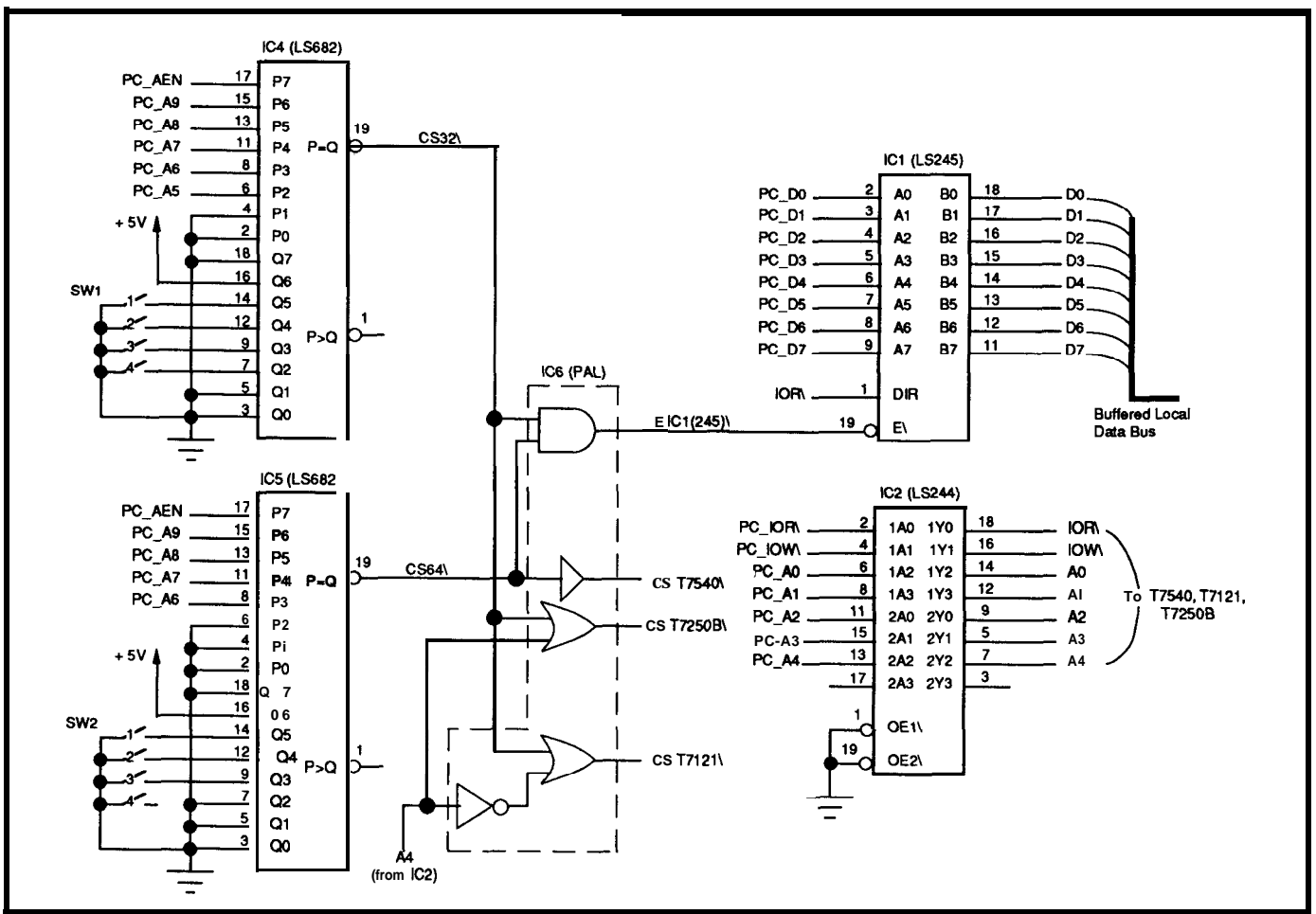


Figure 9 - The PC bus interface for address decoding and data bus buffering on the ISDN BRI board.

LAYER 2 LAPD

The purpose of the LAPD protocol standard (1.441) is to provide the following services:

- *one or more data link connections on the D channel using the address field of each LAPD frame (Figure 12)

- *two types of information transfer: unacknowledged UI (unnumbered information) frames and acknowledged multiple frames using extended (modulo 128) frame numbering

- *establishment of and release of multiple frame data links

- *full implementation of timers T200 (transmit initiation) and T203 (max time without frame exchange)

- *detection and recovery from frame sequence errors

- *display of variables and exceptions associated with established links

The software module for LAPD provides the interface to Layer 3 and Layer 1. It also links to the TE state

manager. Memory management structures allow the transfer of data bytes via the FIFO buffers of the T7250B HDLC data formatter.

LAYER 3 SOFTWARE

The I.451/Q.931 Layer 3 standard documents the procedure to estab-

lish, maintain, and disconnect network connections at the ISDN interface using the resources of Layer 2 and Layer 1. It provides call control procedures via message structures to set up, connect, or release a call on the B channel.

The Layer 3 software module is implemented in subblocks following

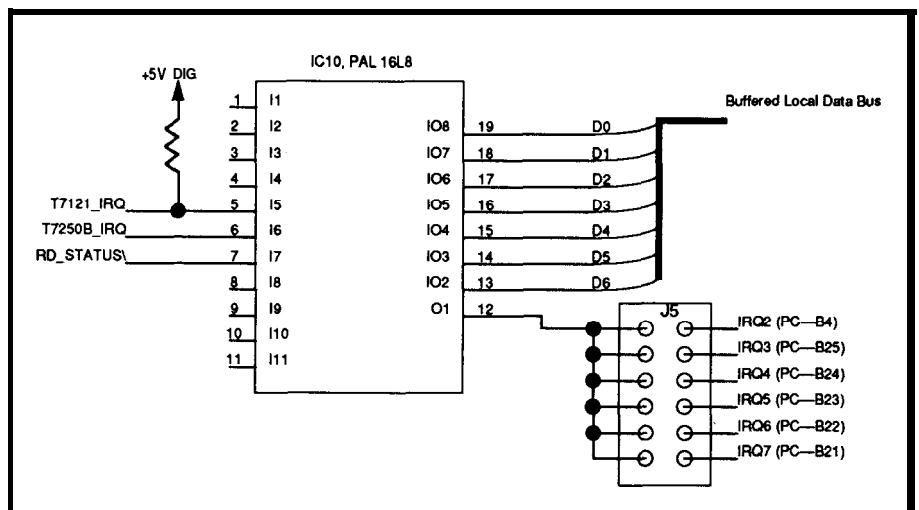


Figure 10 - The interrupt control logic section of the ISDN BRI board.

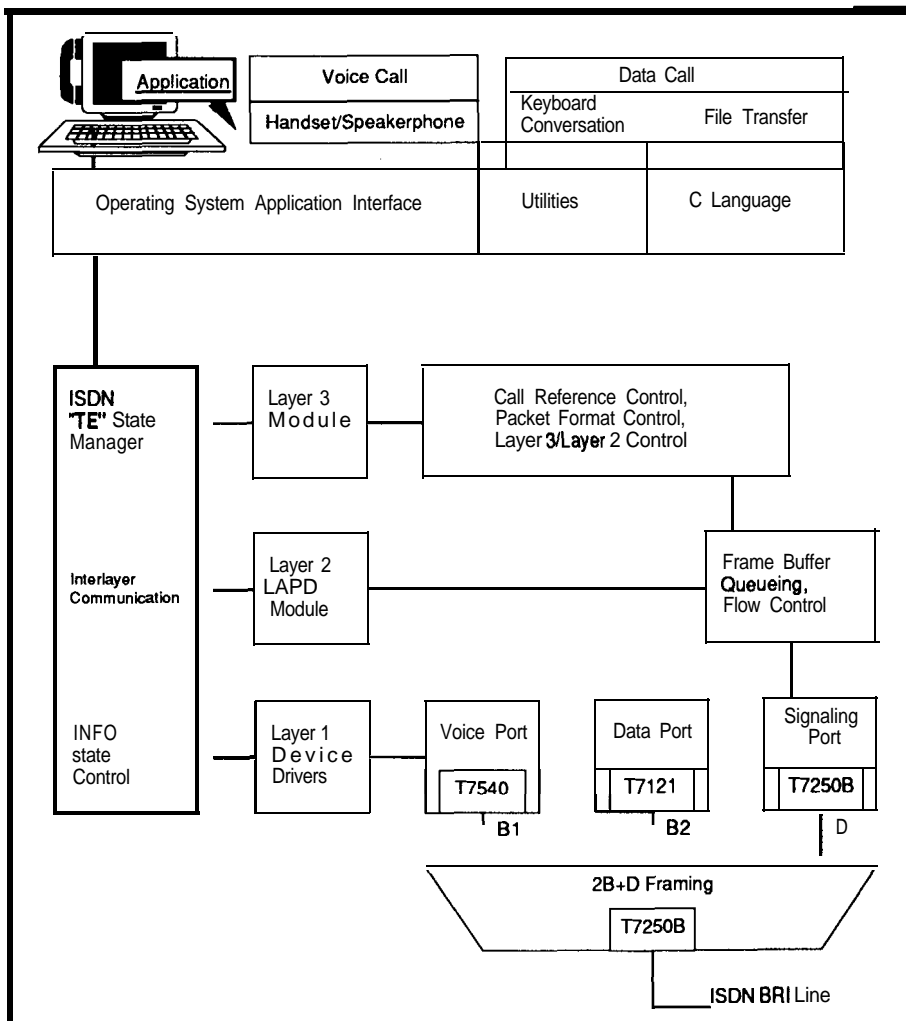


Figure 11 -General structure of the software environment.

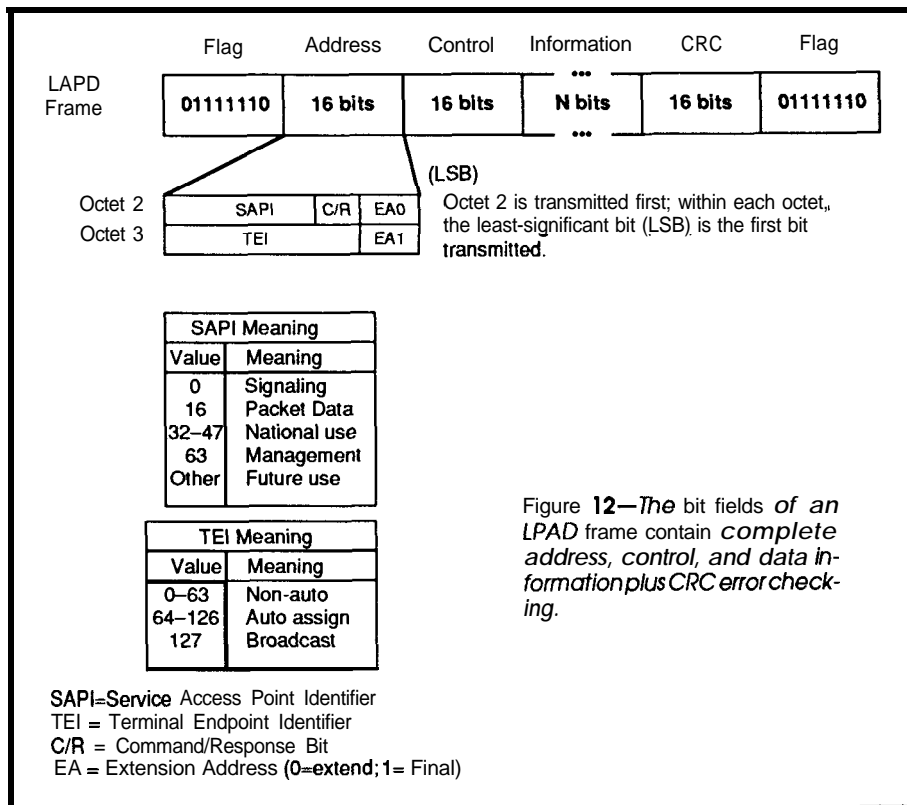
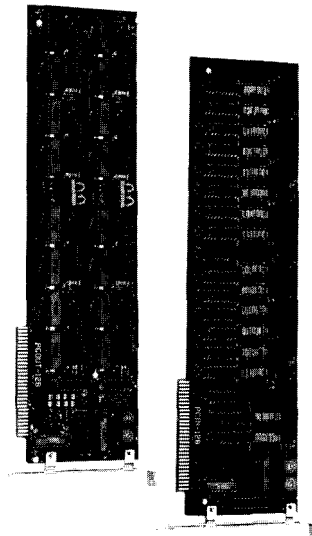


Figure 12—The bit fields of an LPAD frame contain complete address, control, and data information plus CRC error checking.

POWERFUL OUTPUT CAPACITY!



GENEROUS INPUT POTENTIAL:

- ▲ Each input has a built-in pullup, filter, and series resistor protection.
- ▲ Each output can sink a whopping 125 ma. continuous and can be easily paralleled for its drive capacity.
- ▲ Each board has 128 separate I/O points, and signals are paired on the gold dual row headers with a ground wire for easy connections to switches and relays without the need for common bussing of grounds—just peel off a pair on the ribbon cable. **DONE.**
- ▲ The boards' I/O is memory mapped. Powerful and quick instructions can be used beyond simple reads and writes, like move string, exclusive or and even DMA. I/O performs at the speed of system memory without wait states.
- ▲ Rotary switches on board select the fully decoded address and many boards can occupy continuous PC memory space. If outputs are doubled up on a board by using two connectors on a single ribbon for lots of kick on solenoids, address space is cut in half and still continuous.
- ▲ Cost effective at \$495 a board. ▲ Available now. ▲ To Order....
- ▲ ▲ Box 200 ▲ Westview Station ▲ Binghamton, NY 13905 ▲ (607) 729-8273



the protocol control diagrams of the Q.931 standard. Each subblock is treated as an independent task under the control of the application program interface and TE state manager. The functional signaling supported in the Layer 3 module has been tailored for the services offered by the AT&T 5ESS central office switch. Layer 3 software to support new supplementary services or to support other central office switches requires some customization. Such customization is available from third-party vendors such as Link Technology, Holland, Pa., and DGM&S, Mt. Laurel, N.J. Since our software was very coarse and only served to test our design, we feel it would cause more problems than it would solve to release it.

INTEGRATED VOICE/DATA ADAPTER BOARD SUMMARY

The ISDN adapter board design outlined here provides a low-cost, high-performance terminal endpoint. This board allows users to simultaneously access both voice and data services of an ISDN BRI line at the S or T reference point. The PC plug-in board can operate in any IBM-compatible computer using the industry-standard architecture (ISA).

A four-layer printed circuit board with the functionality outlined in this article has been prototyped and tested. The adapter board design provides the audio functionality needed in a digital telephone or an integrated voice-data workstation. Software running on this hardware platform supports not only the Layer 1 "device drivers" but also the CCITT standards I.441(Q.921) and I.451(Q.931) for Layers 2 and 3. The user interface is via window-driven menu options which are very flexible and easy to use. This board allows users to quickly learn about the T7540 digital telephone CODEC providing speakerphone functionality, the T7250B ISDN Basic Rate Interface "S" transceiver, and the T7121 HDLC formatter.

For additional information and documentation on the devices outlined in this article, contact AT&T Microelectronics at (800) 372-2447. ❖

Steven Strauss is a licensed Professional Engineer and a member of the technical staff at AT&T Bell Laboratories in Allentown, Pa., specializing in communications devices. He holds a B.S.E.E. from Pennsylvania State University and an M.S.E.E. from Rensselaer Polytechnic Institute in Troy, N.Y.


P.K. Govind is a distinguished member of the technical staff at AT&T Bell Laboratories in Allentown, Pa. He is an application consultant for communication devices and has extensive experience in product planning, system integration, and product development. Mr. Govind received an M.S. and Ph.D. in physics from the University of Colorado, Boulder, Colo.

REFERENCES

1. T7540 Preliminary Data Sheet, DS89-148SMOS, AT&T Microelectronics, 1991.
2. T7250B Advance Data Sheet, DS89-083SMOS, AT&T Microelectronics, 1989.
3. T7121 Data Sheet, DS90-087SMOS, AT&T Microelectronics, 1990.

IRS

- 4 10 Very Useful
- 4 11 Moderately Useful
- 4 12 Not Useful



AvCase™

C Compilers
Macro Assemblers
Source-Level
Debuggers

**8051
8096
64180
Z80**

**AVOCET
SYSTEMS, INC.**

The Source For Quality Embedded-System Tools

120 Union Street, P.O. Box 490, Rockport, ME 04856
In Maine, or outside U.S., call (207) 236-9035
TUX: 467-210 Avocet CI • FAX: (207) 236-6713

Call Today For Free Catalog 1-800-448-8500

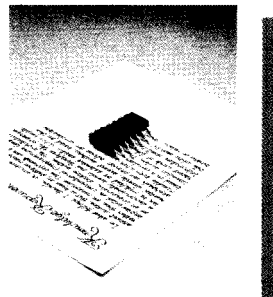
Reader Service #111

Limited Editions

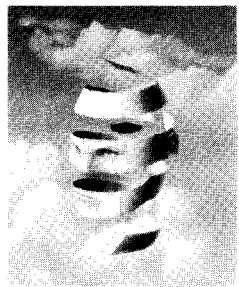
#G \$60 #H \$60



Programmable Hardware



Word Processor



Artificial Intelligence



Technological Breakthrough

#K \$60 #C \$80

Circuit Cellar Ink cover artist Robert Tinney proudly offers these distinctive 16" x 20" Limited Edition Prints. Each is an exquisite reproduction from the pages of *Byte Magazine*, and is part of an edition of only 1000 prints. The museum quality stock is acid free, ensuring brilliance and durability for decades to come. The artist personally inspects, signs and numbers each print, which is accompanied by its own Certificate of Authenticity.

Order your prints beautifully triple-matted and framed! The frames are of the silver metal variety, and mats are chosen to complement the colors of the print(s) you order. Plexiglass only.

The price of each print is shown at left. Order two or more and deduct 15%! Frames are only \$39.50 each. For shipping, add \$5 per order for unframed prints (\$25 overseas); for framed prints, add \$6.50 for one print and \$4.50 for each extra print (ground).

No frames shipped overseas. Full refund if not satisfied. For VISA, MasterCard or AMEX orders call 1-318-826-3003.

ORDER FORM cci

Qty.	#	Title	Amount
_____	_____	_____	\$ _____
_____	_____	_____	\$ _____
_____	_____	_____	\$ _____
If you order two or more, deduct 15% \$			_____
Frames (\$39.50 each) \$			_____
Shipping charges: See above. \$			_____
Total \$			_____

I have enclosed a check or money order to Robert Tinney Graphics. (Must be drawn on a U.S. bank; no foreign collection, please.)

Bill my VISA MasterCard American Express

Card #: _____ Expires: _____

Name: _____

Address: _____

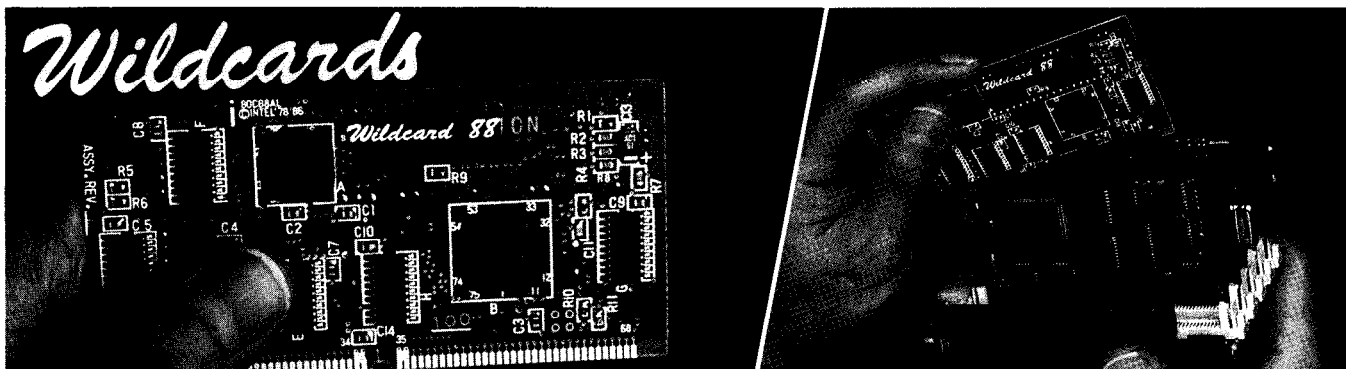
City: _____ State: _____ Zip: _____

Country: _____

Send a brochure showing your other prints.

ROBERT TINNEY GRAPHICS
P.O. Box 778
Washington, LA 7058C

2" x 4" EMBEDDED PC



Microcontroller.

Microcomputer.

"Megatel Wildcards provide PC functionality in a flexible, small format."

Wildcard 88™

- CPU clock to 10 MHz
- Replaces full PC motherboard
- Co-processor and BIOS socket
- DMA, Bus, DRAM, Keyboard controllers

Multi10

- On-board SCSI Host Adapter (supports up to 7 devices)
- Floppy Controller (1.44M, 1.2M)
- 2 RS-232, 1 Parallel, 1 RS-485 multi-protocol serial port

Vid/Mem:

- 640Kb User memory
- Video/Colour LCD controls CGA, Hercules®, IBM® Mono: (runs LCD Panels)

All Wildcards are low power single +5 volt operation.

For information on our representatives please contact our head office at the number below.

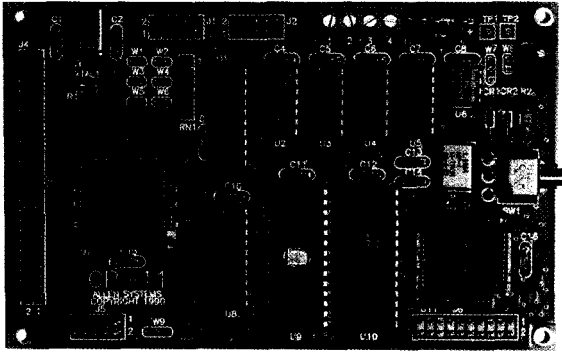
125 Wendell Ave., Weston, Ont. M9N3K9 Fax: (416) 245-6505

(416) 245-3324

Wildcard 88 and Megatel are trademarks of Megatel Computer Corp Hercules is a trademark of Hercules Corp IBM is a trademark of IBM Corp

megatel®

8051, 8096, 68HC11, 68332 SINGLE BOARD COMPUTERS



We feature a series of single board computers for process control applications. Each is available as a bare printed circuit board, or fully assembled and tested. Optional development software is also available. Please contact us to discuss your requirements and receive a literature package covering technical specs and pricing.

ALLEN SYSTEMS

2346 Brandon Road • Columbus, OH 43221
(614) 488-7122

Reader Service #106

NEW
VERSION
AVAILABLE

The Heart of the Matter...

RTXC™

Real-Time Multitasking Executive

. INTEL 80x88/86, 8096/80C196
. MOTOROLA 680x0, 683xx, 68HC11, 68HC16
. INMOS T400, T800 . ANALOG DEVICES 2100

- . Preemptive Scheduling
- . Fixed or Dynamic Priorities
- . Time out on some services
- . Configurable and ROMable
- . Intertask Muncunations
 - Messages
 - Queues
 - Semaphores
- . Memory Management
- . Resource Manager
- . Over 40 Executive Services
- . System Level Debugging Utility
- . System Generation Utility
- . Written in C
- . Source Code Included
- . No Royalties
- . Technical Support
- . Widely Ported
- . Sensible License Agreement
- . Most Popular C Compilers Supported

FREE
DEMO
AVAILABLE

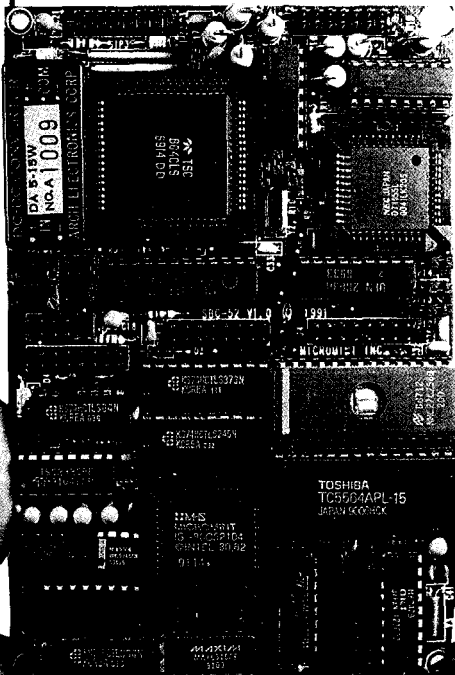
One Time License Fee \$2,995
Discounts for Multiple Licenses/Ports
The only real-time kernel you'll ever need™

A.T. BARRETT & ASSOCIATES
11501 Chimney Rock, Houston, TX 77035
FAX 713/728-1049
Phone 800/525-4302

PC SUPPORT
AVAILABLE

See us at The Embedded Systems Conference Booth #817

Reader Service #109



BRUTE-52 Maximum Power Minimum Space

Micromint's BRUTE-52 is the ultimate compact controller. One look at the list of features will tell you that this full-featured controller has the power to crush your most demanding applications:

- . CMOS 80C52/80C32
- . Three 16-bit counter timers
- . 11.0592 MM System Clock
- . Hardware Watchdog Timer
- . Hardware Clock-calendar
- . Optoisolated Serial Communications
 - RS-232 or RS-485! 300-9600 bps!
- . Optoisolated Serial Printer Port,
 - RS-232! 50-9600 bps
- . 5V-only Operation
 - . Up to 56 Kbytes RAM and/or EPROM
 - . 1 Kbit EEPROM
- . 12-M parallel TTL I/O
- . 8-bits buffered high-voltage, high-current outputs
- . 8-bits optoisolated non-polarized DC inputs
- . 12-bit plus sign analog-to-digital converter
 - 8 channels! 60 Samples/second! 1.2 mV resolution!
- . 12-M digital-to-analog converter
 - 2 channels! 1.2 mV resolution! Selectable ranges!
- . Only 3.5 x 5.3 Inches!
- . Operates at 0-70°C
- . Consumes only 100-200 mA (depending on configuration)
- . Use networked or stand-alone

BRUTE-52 offers you all these features starting at only \$289! (quantity one) When you add in Micromint's renowned quality, services, and support, you won't find a better value in compact control.

To order BRUTE-52, or for more information, contact:

Micromid, Inc.

4 Park Street • Vernon, CT 06066
Phone (203) 871-6170 • FAX (203) 872-2204

See us at The Embedded Systems Conference Booth #819

Reader Service #174



60 OrCAD Schematic Design Tools
A Working Engineer's Impression
by Bruce Webb

64 Schematic Capture with Schema
by Ken Davidson

SPECIAL SECTION

Bruce Webb

OrCAD Schematic Design Tools

A Working Engineer's Impression

Like a lot of people, I stuffed making printed circuits with drafting tape and "rub-on" pads. Not content with such crude techniques, I started investigating what was available to computerize the process. So, five years ago I tried using a general-purpose CAD program.

Then I got a printed circuit layout program (Tango from Accell Technologies). Last year my company bought an autorouter and schematic capture software. Each **upgrade** made the job a little easier, a little faster, and more accurate. The last improvement turned out to be the most profound. Getting the schematic into the computer makes a big difference.

The process of printed circuit layout involves four basic steps:

- 1) Design the circuit (draw schematic)
- 2) Place parts on board
- 3) Route connections (manual and/or auto)
- 4) Check board against schematic

Schematic capture programs perform or assist with three of the four: designing, routing, and checking. The hardest part for me has always been making sure that the printed circuit matches the schematic before I spend the **time and** money fabricating boards. That is where schematic capture makes a big contribution.

ORCAD

OrCAD Schematic Design Tools (SDT) is a low-cost schematic capture program which may be used with other design products as part of a complete printed circuit design package or alone to create standard and easy to read schematic diagrams.

OrCAD SDT is not a single program; it is a collection of programs (or "tools") that provides a way for the user to create, edit, check, and print schematics. It requires a minimum of five megabytes of hard disk space on an IBM-compatible computer with at least EGA graphics, 640K or more of RAM (EMS supported), and a mouse. The results of your efforts may be printed on most popular printers or on an HP-compatible plotter. I use a 16-MHz 80286 machine with 4M bytes of RAM, a super VGA display, and an HP-compatible laser printer. I use TangoPCB and autorouter to create PC board layouts from OrCAD net lists.

The heart of the OrCAD SDT software is a drawing program called **DRAFT**. **DRAFT** starts with a blank drawing sheet that includes a title block where company name, schematic title, revision date, and so on are stored. The title block is not just an area on the drawing; it is a kind of header that makes it easy to keep track of your documents. A pull-down window provides access to the **DRAFT** commands. The operation is intuitive, though some time with the manual may keep the user from confusing commands.

DRAWING FEATURES

Drawing a schematic is as simple as retrieving the parts that make up the design from libraries and connecting them with lines or "wires" on the screen. OrCAD also allows groups of wires to be collected into buses to more easily get the connections from one area of the schematic to another without cluttering. Special connections such as power or ground are made to nodes which are implicitly connected. The process elements are pretty much the same as hand drawing, except the lines stay straight without a ruler.

Large schematics may be divided into sheets with either a flat or hierarchical structure. Connections between sheets are accomplished using module ports. The flat structure has the same level of detail on all sheets. In a hierarchical structure, the detailed lower levels of the drawing are represented as blocks at levels nearer the top. OrCAD SDT allows blocks of schematics which are repeated in the de-

signed to show non-connected pins and then referred to by a block designation. The more complex your schematic, the more you will appreciate the completeness and flexibility of these structures. Fortunately, if your needs are simpler, these options do not get in the way.

LIBRARIES

Schematic versions of electrical components are stored in SDT's extensive libraries of more than 20,000 parts. The components are rectangles or logic shapes with the "pins" represented as lines. Each pin is labeled with its function name. Using library parts relieves the user from the effort of redrawing them, and since the libraries have been debugged by years of use, it is unlikely that you'll encounter any mislabeled or missing pins. The libraries are organized by families such as CMOS, TTL, or analog, and by manufacturer such as Motorola or Intel.

Only the libraries you use need to be installed. There is no point in having OrCAD search through digital components if your design will only use analog parts. Components not available in OrCAD libraries can be made from existing parts or built from scratch. I have created, for example, a complete library of my own of Maxim interface and microprocessor control components that OrCAD didn't include. I also created special RAM/ROM chips for designs where either

an EPROM or static RAM (like 2764 or 6264) can be installed.

One of the hardest skills I have learned in drawing schematics is predicting how much space will be needed between components for wires to run. Moving a part to another area on the schematic can be done so that the wires move with it, but the results are usually messy and have to be redone more

pins wired to a third could be connected together directly. Power and ground lines are generally tied to nodes which are implicitly connected to keep the drawing less cluttered.

The net list provides a way to communicate the schematic connections to a PCB layout program. All of the connections are then routed either by hand or using an autorouter. I pre-

OrCAD boasts 50,000 users and have developed support for them that I found to be reasonable and clear. There has always been someone at the other end when I called and if the person I first reached couldn't help, I was called back within a few hours with the answers.

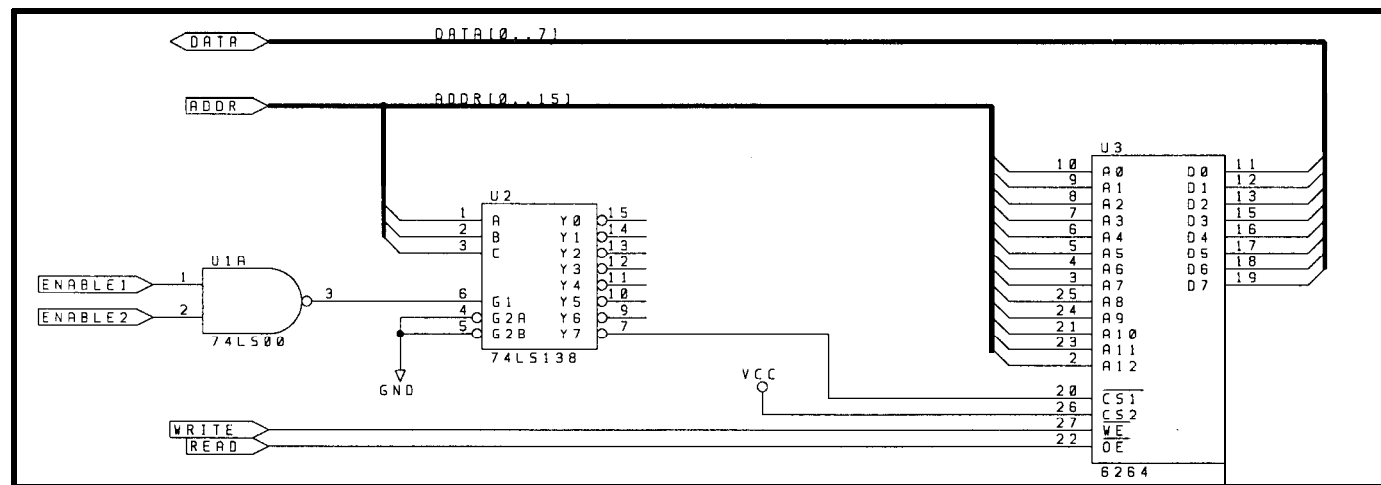
than once. I usually try to complete small blocks of schematic and place them rather than placing individual parts. This is also a good idea when placing parts on the PC layout. You have a schematic. Now what?

A schematic capture program **helps you** to draw an easy-to-read and somewhat standard drawing, but the real magic is what happens once the schematic is complete. The schematic can be used to create a file of the drawing's connections called a net list. The net list is an ASCII listing of all of the wires on the schematic with the endpoints of each being a component pin. The connections are grouped together into nets. The grouping is important because it recognizes that two

fer to route most boards by hand and then check my work using the autorouter. Checking can be thereally hard part to do by hand! When the process is complete, there will be a schematic diagram that exactly matches the printed circuit board and vice versa.

Having an accurate and verified schematic is an extremely valuable tool for someone trying to modify or debug a circuit-ven if that someone is the original designer!

OrCAD SDT can create a number of reports related to the design. They include a parts list, designation list (U1 is an 8031), and so on, that help ensure compatibility between the printed circuit and the schematic.



A typical drawing made with OrCAD might include individual gates, complete chips, and bused signals. This sample was output on a plotter with a line-tipped pen.

SETTING UP

OrCAD has chosen to use an install program for setup and to transmit updates. There is a total of four 1.2M-byte floppy disks on which the software has been archived (compressed). One of the disks marked "INSTALL" is placed in A: drive and the user types the word `INSTALL` from the prompt. The install program asks questions about hardware and library requirements to customize the system. Nothing could be **easier**. The programs are installed in a well-organized way in several subdirectories and changes are made to `CONFIG.SYS` and `AUTOEXEC.BAT` automatically.

The programs come with five manuals including: Installation and Technical Support Guide, OrCADSDT User's Guide, OrCAD ESP Environment User's Guide, Text Editor User's Guide, and OrCAD SDT Reference.

The manuals alone are not particularly helpful if it is the first time using schematic capture software since some of the lingo can be confusing. It

is difficult to distinguish between what might have been problems with the documentation and my limited understanding of the design process and impatience when I started using OrCAD. I needed the software right away, so I began by drawing some simple designs and working things out with the Reference Manual in hand. I don't recommend reading all the manuals before starting. It is better to jump right in and then go back later to learn some of the finer points.

A sample schematic is included with the program and is discussed in the user's guide. Following the tutorial with the manual in hand is very helpful, but cannot cover all of the topics encountered when designing a real schematic for a project.

ORCAD'S LATEST EFFORT

My one major complaint about OrCAD SDT when we first purchased it was that each program of the collection had to be invoked from the DOS command line with **whatever switches**

and file names were appropriate. For example:

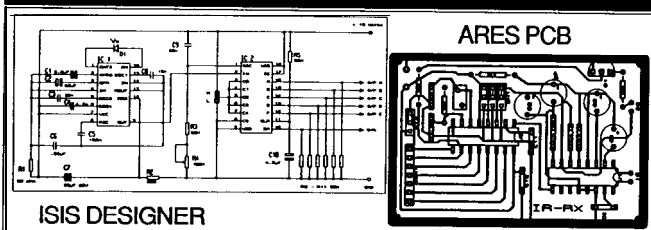
```
NETLIST MYBOARD.SCH
C:\DESIGN\MYBOARD.NET /S
```

is the command to create a net list from a **drawing** called `MYBOARD.SCH` and place the file (`MYBOARD.NET`) in subdirectory `c:\DESIGN`. The `/S` at the end signifies that the file should be written in the format for the Tango PCB program that I use for layout. It required great feats of memorization for those who didn't use it every day, or lots of looking through the manual.

OrCAD has **recently improved** the software (Release IV) to include an integrated windows-style environment that they call ESP. On-screen "buttons" are used to select programs and set up options. The program switches are replaced by setting up a local environment with the switches set as defaults. The result is an intuitive user interface that works well.

The new environment is not perfect, though. The programs are not

HIGH PERFORMANCE EECAD FOR YOUR ENGINEERING NEEDS



Easy to Use PC Software - ICON Based - Mouse Driven
Ultra Fast Performance - Advanced Editing Tools
Output to Printers, Plotters & Lasers

FULLY INTERGRATED SOLUTIONS AT \$1500

ISIS DESIGNER is the Schematic Capture for anyone needing to enter designs. Provides **netlist**, multi-sheet, user configurable partslist & Electrical Rules Check report.

ARES PCB autorouter uses an advanced multi-strategy to achieve very high connection rates & it's fast! 10 copper layers, Design Rule Checker and MORE.

R4 SYSTEMS Inc.
P.O.Box 451
West Hill, Ontario
Canada M1E 4Y9

Free DEMO Package
Write or Call Today

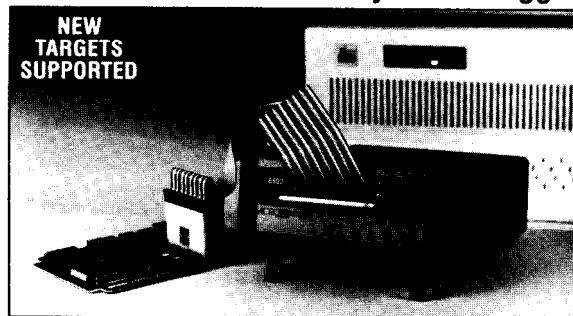
(416) 439-9302

Download DEMO from BBS at 416 289-4554 (2400/8/N/I)

Reader Service 1194

MICROTRACKER™

Real-Time Software Analyzer/Debugger



The MicroTracker™ can significantly reduce the cost of your next real-time product development project. Advanced features speed software development and enhance quality assurance.

FEATURES

- 2K or 8K Trace Memory
 - Interval Timer
 - Performance Analysis
 - RS-232 Interface
 - IBM PC Software
 - Symbolic Disassembly
 - Low Cost from \$1295.
 - Instruction Disassembly for Z80, 8085, 6502, 6802, 6809, 8031/8051, 80188/80186, V40/V50
- Call for Free Brochure

LOGICAL
ADVANCES

52 W. HENDERSON RD.,
COLUMBUS, OHIO 43214
(614) 267-4405

truly integrated and still exist separately. The parameters supplied by the user are correctly inserted when the programs are called, but error handling is sparse and just as cryptic as ever. The dialogue between the environment and DOS is recorded in a file that must be read and interpreted using a text editor. I guess I am a little spoiled by software that diagnoses problems and helps fix them rather than just telling you "file not found" at the point at which the program failed.

SUPPORT

OrCAD's support of their products starts even before you buy them. They have a demo version of all of their products so potential buyers can get a feel for the software and make comparisons before they plunk down a lot of money. They provide a free bulletin board where video and printer drivers and new libraries may be downloaded. Any questions or problems can be answered by a reasonably knowledgeable support staff on the

telephone or via BBS. Updates and fixes are free to registered users. OrCAD publishes a newsletter once a quarter to highlight common misunderstandings, make announcements, and to remind people that they are there if you need help. Support may be extended for an additional charge.

OrCAD boasts 50,000 users and have developed support for them that I found to be reasonable and clear. There has always been someone at the other end when I called and if the person I first reached couldn't help, I was called back within a few hours with the answers.

THE BOTTOM LINE

Schematic capture programs are just about necessary for designing complicated boards to ensure that the board and the schematic match and definitely necessary if an autorouter is included in any part of the equation. OrCAD is affordable and well supported. I've been using the package for about a year, including several

months with Release IV. The few idiosyncrasies in the way it operates are within the limits I am used to and the problems it has solved for me are many. ❖

OrCAD SDT Release IV

Cost: \$595.00

Requirements:

640K RAM
EGA or VGA graphics
Hard Disk
Mouse

OrCAD

3175 N.W. Alcock Dr.
Hillsboro, OR 97124
(503) 6909581

Bruce Webb is a Chemical Engineer turned Electronic Entrepreneur who is a principal in Cottage Resources Corporation.

IRS

4 13 very Useful
4 14 Moderately Useful
415 Not Useful

NOW DOUBLED IN LIBRARY SIZE

DC/CAD

CAD Showdown Results!!

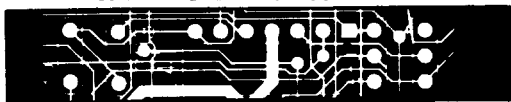
HIGH DENSITY EXPERTS!

Integrates Schematic, PCB Layout & Autorouting

This top-rated CAD out-routed the competition in the 1990 CAD Show-down. DC/CAD displayed its power and flexibility when routing a double-sided board while competing routers used four to six layers. This non-copy protected package with surface mount support includes:

- Multi-strategy 1-mil parts autoplacer
- "1-mil" autorouting with rip up & retry
- ...Thorough layout design checker...
- Full 2-way GERBER and DXF support
- Optional autoground plane support with cross-hatching
- Optional simulation capability & protected mode for 386 users

LEASE PROGRAM & SITE LICENSE AVAILABLE
30 DAY MONEY BACK GUARANTEE



DESIGN
COMPUTATION

Rt. 33, Sberman Square Farmingdale, NJ 07727
(908) 938-6661 • (908) 938-6662 (FAX)

DC/CAD...Innovative, Intelligent
and Integrated Software

CALL TODAY
DC/CAD \$595
(Priced from \$395)

Take It Easy.

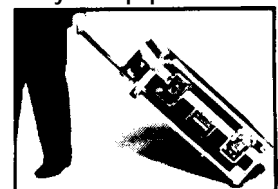
Take it easy on your cargo with a custom Cabbage Case built to the exact dimensions of your equipment.

Take it easy on your back with our extension handle and tilt wheels options.

Take it easy on your wallet. Let Cabbage Cases show you how easy it is to

save money on quality, custom-built road cases that make shipping and traveling with your valuable cargo safer and easier. Prices quoted over the phone.

Call 800-888-2495 today.



CABBAGE CASES, Inc.

1166-C Steelwood Rd.
Columbus, OH 43212

800/888-2495

614/486-2495

FAX/486-2788

egrated and still exist sepa-
The parameters supplied by
er are correctly inserted when
grams are called, but error han-
s sparse and just as cryptic as
he dialogue between the envi-
nt and DOS is recorded in a file
ust be read and interpreted us-
ext editor. I guess I am a little
d by software that diagnoses
ms and helps fix them rather
st telling you "file not found" at
nt at which the program failed.

PORT

CAD's support of their prod-
arts even before you buy them.
have a demo version of all of
roducts so potential buyers can
eel for the software and make
risions before they plunk down
f money. They provide a free
nboard where video and printer
s and new libraries may be
oaded. Any questions or prob-
an be answered by a reasonably
edgeable support staff on the

telephone or via BBS. Updates and
fixes are free to registered users.
OrCAD publishes a newsletter once a
quarter to highlight common misun-
derstandings, make announcements,
and to remind people that they are
there if you need help. Support may
be extended for an additional charge.

OrCAD boasts 50,000 users and
have developed support for them that
I found to be reasonable and clear.
There has always been someone at the
other end when I called and if the
person I first reached couldn't help, I
was called back within a few hours
with the answers.

THE BOTTOM LINE

Schematic capture programs are
just about necessary for designing
complicated boards to ensure that the
board and the schematic match and
definitely necessary if an autorouter is
included in any part of the equation.
OrCAD is affordable and well sup-
ported. I've been using the package
for about a year, including several

months with Release IV. The few idio-
syncrasies in the way it operates are
within the limits I am used to and the
problems it has solved for me are
many. ❖

OrCAD SDT Release IV

Cost: \$595.00

Requirements:

640K RAM
EGA or VGA graphics
Hard Dfsk
Mouse

OrCAD

3175 N.W. Alcock Dr.

Hillsboro, OR 97124

(503) 690-9881

*Bruce Webb is a Chemical Engineer turned
Electronic Entrepreneur who is a principal in
Cottage Resources Corporation.*

IRS

4 13 Very Useful
4 14 Moderately Useful
4 15 Not Useful

NOW DOUBLED IN LIBRARY SIZE

DC/CAD

CAD Showdown Results!!

HIGH DENSITY EXPERTS!

egrates Schematic, PCB Layout & Autorouting
top-rated CAD out-routed the competition in the 1990
Showdown. DC/CAD displayed its power and flexibility
a routing a double-sided board while competing routers
four to six layers. This non-copy protected package with
ce mount support includes:

M-strategy 1-mil parts autoplacer
"nil" autorouting with rip up & retry
rough annotating design rule checker
2-way GERBER and DXF support
tional autoground plane support with cross-hatching
tional simulation capability & protected mode for 386 users

LEASE PROGRAM & SITE LICENSE AVAILABLE
30 DAY MONEY BACK GUARANTEE



DESIGN
COMPUTATION

Rt. 33, Sberman Square Farmingdale, NJ 07727
(908) 938-6661 • (908) 938-6662 (FAX)

DC/CAD...Innovative, Intelligent
and Integrated Software

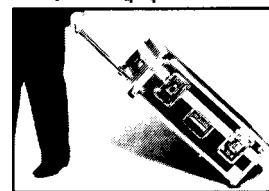
CALL TODAY
DC/CAD \$595
(Priced from \$395)

Take It Easy.

Take it easy on your cargo with a custom Cabbage Case
built to the exact dimensions of your equipment.

Take it easy on your back
with our extension handle
and tilt wheels options.

Take it easy on your
wallet. Let Cabbage Cases
show you how easy it is to



save money on quality, custom-built road cases that make
shipping and traveling with your valuable cargo safer and
easier. Prices quoted over the phone.

Call 800-888-2495 today.

CABBAGE CASES, Inc.

1166-C Steelwood Rd.,
Columbus, OH 43212

800/888-2495
614/486-2495
FAX/486-2788

SPECIAL SECTION

Ken Davidson

Schematic Capture with Schema

So what's wrong with notes on paper napkins? Sure, they can be a little mushy to write on, and you can't fit a whole lot onto one, but they travel real well, and once you're done with them, you can use them to wipe the sweat from your brow.

Steve tells stories of the "good old days" when he could sketch out a circuit on a dinner napkin, bring it down to Ray Long's to have a board laid out (see "Bringing in the Pros" in issue #20 of *CIRCUIT CELLAR INK* for more on Ray's company), and have a working board in hand in short order.

Those were also the days when Circuit Cellar projects tended to be less complex. The PC boards were all laid out by hand using several layers of acetate, rub-ons, tape, and a good, sharp knife; circuit designs were sanity checked by the designer's keen eye and perhaps an associate looking over his shoulder; and the person laying out the PC board had to be able to manually swap gates within a package or catch a bad pin number on the schematic. Invariably, when the prototype PC board came back for first test and didn't work, at least half of the problems could be attributed to layout mistakes.

About six years ago, we started looking for a better way. Reasonably priced schematic capture packages that might be considered for serious professional use were just starting to show upon the market. We also started working on Ray to upgrade his shop, trying to convince him that his productivity could jump markedly if he were to computerize. Up to that point, there wasn't a piece of silicon to be found in his shop.

Automation Inc. was trying a novel idea for the software industry: a free demo disk containing a version of the software that was fully functional except for some key features such as saving or printing. We gave them a call, received the disk, and have been using Schema for all our schematics ever since.

SCHEMATIC CAPTURE SOFTWARE

Before I get into Schema proper, let me go over a few schematic capture basics. There are two key questions to ask when looking at any software: what will the software do for me that I either can't do now, can't do efficiently, or can't do effectively (i.e., how will it save me time?); and is it easy enough to use that I'll continue to use it and not be hindered by it?

For anyone not familiar with schematic capture packages, it is useful to consider the idea that they are to circuit design what word processors are to writing. A good word processor does not make a good writer (as I'm often reminded), but a good writer

can often dramatically improve his productivity by using a good word processor. He is able to shed many of the more mundane and error-prone tasks onto the computer and direct all his energies to actually writing.

Most of the designer's (writer's) time will be spent entering new information and manipulating and modifying it. This stage is where a good user interface is a must. If the designer has to labor at using the computer, it's not going to save him any time and he'll be less likely to use it in the future. All schematic capture packages I've seen use a graphical interface with at least one on-screen menu and support (if not require) the use of a mouse. They allow the designer to manipulate everything associated with a component as a single unit. For example, the outline of the part, the pin numbers, and the pin descriptions are all integrated. Individual components or groups of components may be moved anywhere on the screen and wire connections may be changed at any time.

Humans are the only ones who can look at a design to determine if it's

going to work, but the computer can often help catch the obvious mistakes. A good piece of schematic capture software will do a "design rule check," which is akin to the spelling checker found in most word processors. (The word processor won't tell you whether the critics will like your piece, but it can at least make sure you're using words from the English language.) The design rule check looks for such blunders as multiple drivers connected together, inputs left floating, multiple components with the same reference designator, and labels used in only one place.

Once your design is complete, you may elect to print or plot the finished product. Most schematic packages support at least HPGL pen plotters and dot matrix printers. Most also support a broader range of plotters plus laser printers. Likewise, most word processors have a list of supported printers longer than their list of supported features.

In order to aid in the transfer of your finished design to the next stage, most schematic capture packages also support the generation of a "net list" (which is usually the reason for using the software in the first place). Any connection between two or more components is called a "net." A net list is nothing more than a list of components on the board and a list of connections between them. It is used by

PC board layout software to ensure accurate transfer of the design from the symbolic schematic stage to the physical hardware stage. Similarly, word processors often support numerous file formats to ease the transition from, for example, the author's IBM PC to the publisher's Macintosh.

SCHEMA: THE SCREEN

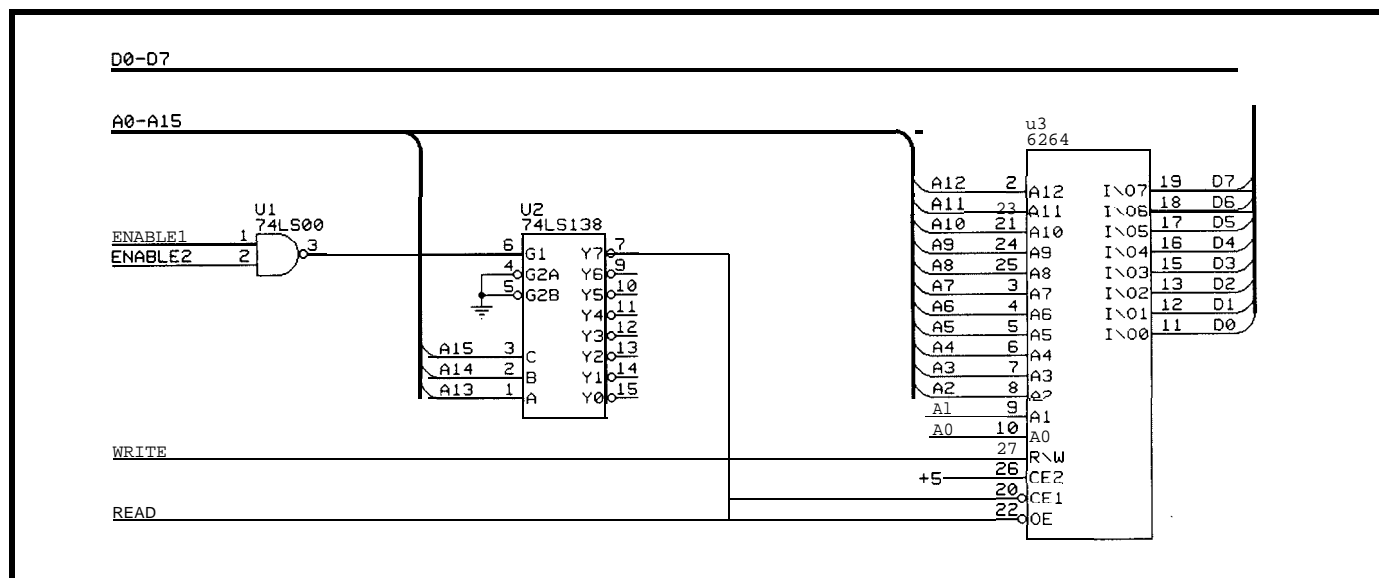
When we received the demo disk and tried out the package, the most striking feature at first glance was its user interface. Most drawing packages allow you to see only a static view of your drawing and **force you** to use scroll bars or awkward keyboard commands to move to other parts of the schematic. With Schema, when **the cursor reaches an edge of the drawing area**, the whole area starts to smoothly and quickly scroll across the schematic, stopping either when the cursor is moved away from the edge or the edge of the schematic is reached. With such a feature, the designer can very easily move from one part of the schematic to another without lifting a finger or moving the cursor very far from the area of interest. It also eliminates the frustratingly slow screen redrawing often encountered with drawing packages. If you want to get a better feel for what the drawing looks like as a whole, a number of zoom levels are supported.

Schema includes full mouse support. While not required, you'd have to be a fool not to use one.

The majority of the screen is consumed by the main drawing area. Down the left side is a menu, the top of which always contains the top-level commands while the bottom changes depending on what command has been selected. At the bottom left of the drawing area is an optional screen coordinate box that is continuously updated with the current location of the cursor. Many different display adapters are supported, including all the major super VGA boards. It was a pleasure to upgrade from a vanilla CGA to an 800 x 600 VGA display. One of the 1024 x 768 boards on the market would provide an even larger window onto the schematic being drawn.

PART DEFINITIONS

The first step in drawing any schematic is defining the components. Definition includes extensive libraries containing all the popular logic families (LS, CMOS, etc.), microprocessors, memory, and, of course, active and passive analog components, so chances are most of the parts you'll want are already defined. For those that aren't, or if you're not happy with the predefined version, you can go to the object editor.



Comparing the same circuit between Schema and OrCAD shows that both programs get the job done, but each has its own style. Schema's output was done on an HP LaserJet printer.

The object editor is integrated into the main drawing program and has an almost identical user interface to the schematic editor. It includes the essentials for drawing boxes, circles, bitmapped images, labels, pin numbers, and so on. You may also edit anything already on the screen.

In order for Schema to be able to do a design rule check, it must know more about the component than simply what its schematic symbol looks

For anyone not familiar with schematic capture packages, it is useful to consider the idea that they are to circuit design what word processors are to writing.

like. For each pin on the component, you must tell Schema what its number is and whether the pin is input, output, bidirectional, tristated, analog, or "don't care." Schema uses the pin function information during postprocessing to make sure all the parts are connected in harmony, and uses the pin number information to generate the net list.

A somewhat confusing aspect of Schema (but powerful at the same time) is the differentiation between "body objects" and "named objects." When we first started using the package, the distinction between the two was vague, but the documentation has been improved over the years. A body object is the graphical symbol used when the component is put on the schematic. Body objects may be nested (and are often called "nested objects"); for example, a simple inverter object may be defined once, then used multiple times when defining body objects representing a 7404, 7406, and 7414. All use the same basic shape, but

may have slightly different labels or additional symbols.

When a component contains several gates within the same package, a each gate must be defined separately. In the above example, you must define each of the six inverters in the 7404 package as separate body objects. Granted, you can define a single gate, then make copies and small changes to do the rest, but it can still be time consuming.

An even bigger nuisance is having to define each style and rotation of a gate separately. To again use our 7404 example, if we want one inverter symbol with the inverter bubble on the output side and one with the bubble on the input side, plus all four rotations of both styles, we must define 48 separate objects to cover all the bases. Luckily, Omation has already done the work for virtually all the popular gates in use, so you may never run into it.

A named object is used to tie together all the body objects associated with a particular component and is used when placing a component on the page. It is a textual description of the component that includes your stock number, a short description of the part, and the name of the body object family. To use the 7404 example once more, the six main body objects making up the package might be called T04-1A, T04-1B, on up to T04-1F, so you include the "T04" family in the description of the "7404" part. The hierarchy helps a great deal when defining a component with several sections, but can be a hassle when defining something like a microprocessor that consists of just one body object.

THE SCHEMATIC

Defining the parts is the boring part of the process. Once done, you can start the actual drawing of the schematic. The schematic editor allows you to place any predefined part on the page, assigning a reference designator and, when necessary, a value to the part. The part requested may be changed at any time, so if you find that you want a different value resistor or a different gate in the same package,

you can make the change without having to delete and re-place the part.

Any pin on a part that has been properly defined has a perpendicular line at one end denoting where the wires are to be attached to the part. Wires also have arrows at each end, so making sure the arrows always touch the perpendicular on the parts' pins is the best way to be sure connections are made properly. Wire arrows and pin perpendiculars may be turned off at any point and are never printed in the final schematic.

Another aid in getting things lined up is a redefinable grid and optional snap to the grid. Leaving the grid and snap on all the time is another good way of assuring that proper connections are made.

When connections must be made between parts that are at opposite ends of the page or on different pages altogether, a wire may terminate at a label. This label becomes the name of the net, and any other wires connected to the same label elsewhere on the drawing are also connected to the same net. To clarify the drawing, groups of labeled connections may be bused together. The schematic editor provides fat and narrow arcs and lines for the creation of buses. The arcs and lines are only cosmetic, however, since it's the labels that determine to which nets the wires are connected.

I find that overuse of buses and labels make the final drawing confusing and difficult to read. I like to make direct connections whenever possible, using buses only for data and address lines. If control and other signals must go from page to page, I always try to bring the individual signals to either the left or the right side of the page. That way, a quick glance down the two sides of the page will tell you if a particular signal is used on that page. When all the interpage signals are brought to buses at random points on the page, you're forced to scan the entire contents of every page to find the signal you're looking for.

Rearranging portions of the schematic is easy. Simply drag a box around the area you want to move, grab the corner, and the whole area moves in unison. When released, any wires that

crossed into or out of the moved area are still connected, though often end up as diagonal lines that must later be squared off and cleaned up. The advantage of this "rubberbanding" effect is that once you make a connection with a wire, that connection is never broken until you delete the wire. It's just one more way that the computer can be used to keep track of the little details while you concentrate on doing the design.

As I mentioned when describing the object editor, support for rotated parts is pretty slim. Recent versions of Schema include a rotate command in the draw and edit menu entries, but the command relies on the existence of rotated versions of the part in one of the libraries. If a rotated version hasn't been defined, one must be defined before a rotation can take place.

Scaling isn't supported at all. Once an object has been defined, its size can't be changed unless you go into the object editor and redefine it. Unless you're trying to fit a D-size drawing onto a B-size page, I don't think you'll miss scaling.

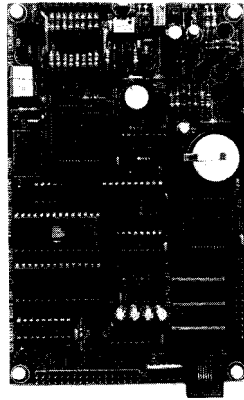
POSTPROCESSING AND PRINTING

Once the drawing is complete, there is a host of postprocessing that may be done to it. While the object and schematic editors described above are integrated into one program, the rest of the Schema package consists of separate programs for each postprocessing task.

As I mentioned before, a design rule check may be done to look for silly drawing mistakes. A list of parts, sections within each part, and reference designators may be produced, as well as a complete bill of materials with your own stock numbers next to each component. The net list may also be generated at this point.

Almost all PC board layout programs use their own net list format. Automation includes a very useful utility that converts their net list format to those used by the major PC board layout programs and systems. Our layout house (Custom Photo) purchased a Calay system in response to our prodding, so the final step we

Ten microwatts.



Less money, less power, less time, less work.

To place an order, or for more information, call:
1-800-GET-DATA (438-3282)
1-602-996-0255/fax

With its automatic power cycling, that's the average power draw of the nanoLINK Controller/Peripheral in a typical application. No extras--it's all built-in and can be enabled with one instruction. Run your application for over two years on a 9V battery, or forever on a thumb-sized solar cell.

And because the nanoLINK Controller/Peripheral was designed to function as a PC peripheral--no programming required--OR as a stand-alone controller, you can use it on the bench, in the field, or both. 8 A/D inputs, 32 I/O lines, 1A power & PWM outputs, and much more.

A full-function BIOS is included--no device drivers to write. A typical data-logging application requires less than a dozen lines of code. And an interactive editor & macro assembler is included--no expensive cross development tools to buy.

And the price? \$395, quantity 1--call today.

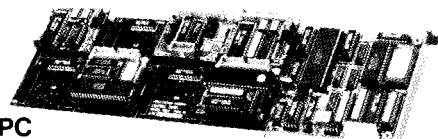
**nano Power
 Devices**

A Division of AirDigital Corporation

Reader Service #104

Position and/or Velocity

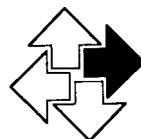
Motion Control



DCX-PC

8-Axis Programmable Motion Controller

- State-of-the-art Digital Multifunction Controller can be configured in minutes using "off-the-shelf" components
- DCX motherboard contains intelligence/ memory/firmware and 8 sockets for "plug-in" Modules
- 1 to 8 "plug-in" DC Servo, Stepper or Special Purpose Control Modules can be mixed/matched on same DCX board
- Install in any PC/XT/AT compatible, or use "stand-alone" with RS232 and/or IEEE-488 interface Modules
- High level interface libraries in "C" and "BASIC," with examples and source code, included



Precision Micro Control

C O R P O R A T I O N

8122 Engineer Road, San Diego, CA 92111

(619) 565-1500 FAX (619) 565-1511

Reader Service #191

always perform before sending the schematic to have a board made is to convert the net list to Calay format. Any potential incompatibilities are flagged during the conversion. (Custom Photo also purchased a copy of Schema so they could draw schematics for customers still using napkins.)

Both printing and plotting are supported to suit the equipment you have on hand. When we started with Schema, we always used a relatively slow plotter to generate output. The results were usually acceptable, but the plotter usually had problems with bitmapped symbols and it often took 15-20 minutes to plot a single page. One advantage of plotting is that multiple colors are supported, so it's possible to, say, separate functional blocks by drawing one block in one color and another block in a second color.

Since it took so long to plot each page, I would always make as many additions, changes, and fixes as possible on the screen before generating new hard copy. The switch to an HP LaserJet changed that methodology, however. With the laser printer,

Schema can output a page in just 2-3 minutes, and I find the quality to be much higher. Since mistakes are more often found on the printed page than on the screen, the printer makes it much more convenient to make check plots as work progresses.

Also included in the suite of programs is a setup utility for configuring the drawing program and individual schematics, a librarian used to maintain libraries of objects and parts, and several utilities used to pass information back and forth between Schema's drawing program and Schema PCB (a separate package that Jeff knows more about than me; see "Working with an Autorouter" in issue #20 of *CIRCUIT CELLAR INK*).

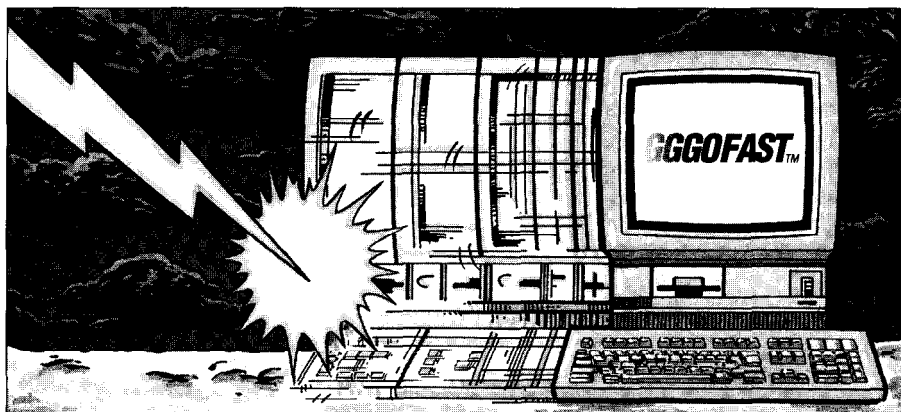
Tying everything together is a text-based menu program that lists all the Schema modules plus most DOS commands and any user-specified programs. Again using a mouse, it is easy to jump around to the various programs. In virtually all cases, program options may be specified on the command line or may be chosen interactively.

THE BOTTOM LINE

Does Schema live up to the two questions I posed earlier? It most certainly does. I can whip out a schematic in a fraction of the time it would take me to do it by hand, a result of Schema's combination of features and user interface. In fact, I'm so comfortable with the package that I'll do design work right on screen and skip the hand-sketching stage altogether. It's easy to draw some ideas, print them out, check them over, perhaps build some test circuits, and make necessary changes, all without lifting a pencil.

Since I've been using Schema for so long, and haven't had any real experience with other schematic packages, it's impossible for me to make any real comparisons between them. While it has its quirks and weaknesses (any piece of software has to have a few), it's always done what any good piece of software should do: act as a tool to help, rather than hinder, the work process.

Should you run out and buy Schema? Only you can make that decision. All I can tell you is I don't think you'll be disappointed. ❖



GOFAST™

Lightning-Fast Floating Point Accelerators

Empower your 80x86 applications with GOFAST accelerators. Fast, reentrant, and ROMable, GOFAST accelerators boost performance and make sure you can embed your application.

Link and go with C: Microsoft®, Borland®, Intel®, MetaWare®, and WATCOM®. Dynamically replace 80x87 coprocessors during execution.

© 1991 US Software Corporation. GOFAST and DynaCOP are trademarks of US Software Corporation. All other trademarks belong to their respective owners.

GOFAST IEEE accelerators are optimized for 8051, 8096, 8086, 80386, i960, 6801, 6809, 68HC11, 68xxx, 8085, Z80, R3000 and more.

Call for your free GOFAST information diskette today: 503-641-8446; FAX 503-644-2413; 800-356-7097.



142 15 NW Science Park Drive
Portland, OR 97229

U S SOFTWARE®

Schema III

cost: \$495

Requirements:

512K RAM

Graphics card (Hercules, CGA, EGA, VGA, super VGA)

Hard Disk

Mouse (optional)

Omaton, Inc.

801 Presidential Dr.

Richardson, TX 75081

(800) 553-9119

(214) 231-5167

Ken Davidson is the *managing* editor and a member of the *Circuit Cellar INK engineering staff*. He holds a B.S. in computer engineering and an M.S. in computer science from Rensselaer Polytechnic Institute.

IRS

416 Very Useful

417 Moderately Useful

418 Not Useful

DEPARTMENTS



page 70

Firmware Furnace



page 79

From the Bench



page 86

Silicon Update



page 94

Practical Algorithms



page 103

ConnectTime

(Re-)Starting C

OK, even I admit it now: C for microcontrollers has arrived. Whether you like it or not, if you don't speak C you won't be a hit at the party. Indeed, a recent RFQ arrived stating "the software will be written in the C programming language" without specifying the CPU. The handwriting is on the wall!

Rest assured that Firmware Furnace won't turn into Yet Another C Column. There are cases where assembly language code is still required (and I will gleefully point them out), but, for the most part, source code listings will be in C rather than assembler. For us, C will serve as a "high-level assembler" rather than a "low-level" language.

During the past two years I used Avocet C for many of my projects, some of which you have seen here. Judging from the BBS traffic, though, the market leaders are Franklin and Archimedes. Any of these three will set you back about two kilobucks, which is a lot of coin to drop in the slot, especially if you are not convinced C is a Good Thing.

Unfortunately, switching between C compilers is not as simple as competing vendors would have you believe, quite apart from the up-front cost. Each compiler accepts a different subset of the C language, the run-time libraries are nearly disjoint, and the assembly language interfaces are utterly bizarre. While "straight C code" will port, your programs won't because they will depend on features unique to your current compiler and assembler.

The good news is that the market has room for more than just the biggies.

A cursory glance through any magazine catering to the firmware trade will reveal several C compilers priced well under the tropopause. For this column I will use the 2500AD compiler, which I bought earlier this year for a specific project that didn't suit Avocet C. It costs \$600 and includes the compiler, assembler, and an assembly language simulator.

While I don't intend to start a review series, either, I will also look at the Micro-C shareware compiler from Dave Dunfield in the next column. If you thought you couldn't afford C, the times they are a-changin' (and for the better, too).

IN THE BEGINNING

The C language makes several assumptions that just aren't true after the CPU emerges from a hardware reset. For example, although uninitialized C variables are supposed to be set to zero, 8051 hardware does not clear either internal or external RAM before executing the instruction at address 0000h. And the hardware has no idea of how to load the proper values into C's initialized variables, either.

Obviously, all variables must be set up before the first line of C code executes, so the code that gets control immediately after a hardware reset **cannot be written in C**. No matter how simple the C code, some assembly is still required!

Every C compiler package includes a startup routine that must be linked with each C program to handle these initializations. Avocet calls it CRT51x, 2500AD refers to C8051xR,

FIRMWARE FURNACE

Ed Nisley

and Micro-C uses 8051RL. In each case, "x" marks the spot for the memory model identifier, which I will discuss later, so you must use the appropriate routine for your situation.

Each company decides what functions should occur in the startup routine. Avocet includes just memory and stack initialization, 2500AD heaves in ring-buffered and interrupt-driven serial handlers, and Micro-C just sets the hardware stack pointer. You must review the contents of the file to make sure that you are not getting too much,

too little, or the wrong kind of initialization for your purposes. Contrary to popular opinion, changing the startup code is not sinful—it can be essential!

MEMORY MODELS

The 8051 architecture defines several different address spaces. Program instructions are burned into EPROM, which is located in Code space and accessed by the *PSEN logic signal. Variables can be in either Internal or External RAM, the latter accessed by

the *RD and *WR signals. I/O ports are memory-mapped in the External data space, so there is no separate I/O address space as there is on 8086 CPUs.

Internal RAM has only 128 bytes (in the 8052 derivatives, 256 bytes) to hold the CPU's working registers, the hardware stack, and 128 directly addressable bit variables. The working registers are not usually accessible from the C code level. The hardware stack may or may not hold C function parameters, but will always hold the function return addresses. Bit address-

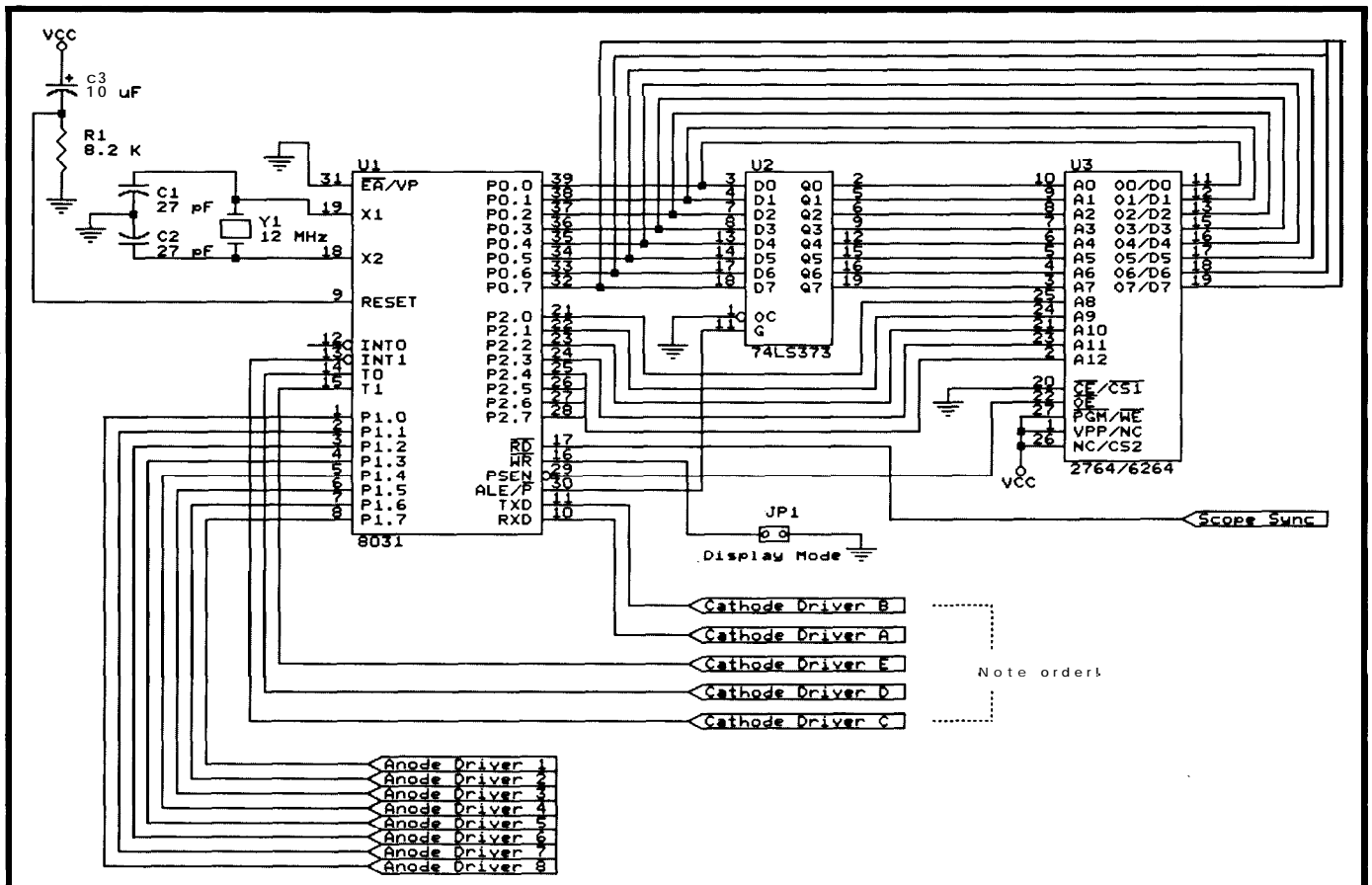


Figure 1—The controller for the sample LED display is a typical 8032-based circuit regular readers should be very familiar with.

ing varies by compiler; some have good support, others none at all, while some have rather clumsy support that isn't worth using.

A further complexity arises when external hardware combines the Code and External RAM address spaces by ORing the *PSEN and *RD signals. The Code space must start at address 0000h because that's where the CPU begins execution, so RAM must start at a higher address (typically 2000h or 8000h) to avoid collisions.

Most 8051 C compilers support at least two memory models, known as Small and Large. The Small model uses only Internal RAM, while Large uses External RAM for variables and the C parameter stack. Of course, the memory model names are not standardized and there are several permutations and combinations available. For example, Micro-C's Medium model corresponds to Avocet's Large, while 2500AD uses the terms Internal and External Mode. Read the manual carefully!

TWINKLE, TWINKLE, LED

There's nothing like a good hardware project to justify some software experimentation and find how things really work. I'll use the 8032 system shown in Figure 1, which will run with the Small memory model because it has no external RAM. Figure 2 shows the "output device," a rectangular array of 40 multiplexed LEDs.

Each of the 40 LEDs is associated with a C variable located in the 8032's Internal RAM. The firmware counts the variable down at a regular rate. When the value hits zero the firmware updates the LED, turning it OFF or ON as needed. The result is a pleasant blinking array that's sure to brighten up any office decor. When you see it in action you may be reminded of the status panels in those old Star Trek sets...the ones before the fake large-screen CRT displays.

A jumper changes the display mode so the LEDs blink briefly when the corresponding timer hits zero. This

mode is more frenetic, suited for those occasional high-caffeine days.

Listing 1 shows BLINKBOX's main loop. There are two key data structures: the Timers array holds the 40 variables that determine when each LED changes state and the AnodeData array, which holds the 40 bits (five bytes) that map each LED. The DataID and BitMask variables convert TimerID into an AnodeData index and bit location.

Although you might think AnodeData should be located in the bit-addressable section of Internal RAM, it turns out that the 8051 instruction set doesn't have a generalized "set bit" instruction (despite what you might conclude from the 8051 App Notes). The bit location is encoded in the instruction, rather than being held in a register, so 120 different instructions are necessary to turn 40 different LEDs on and off or complement their state. It is far simpler to generate a byte index and mask in software and do the bit twiddling "by

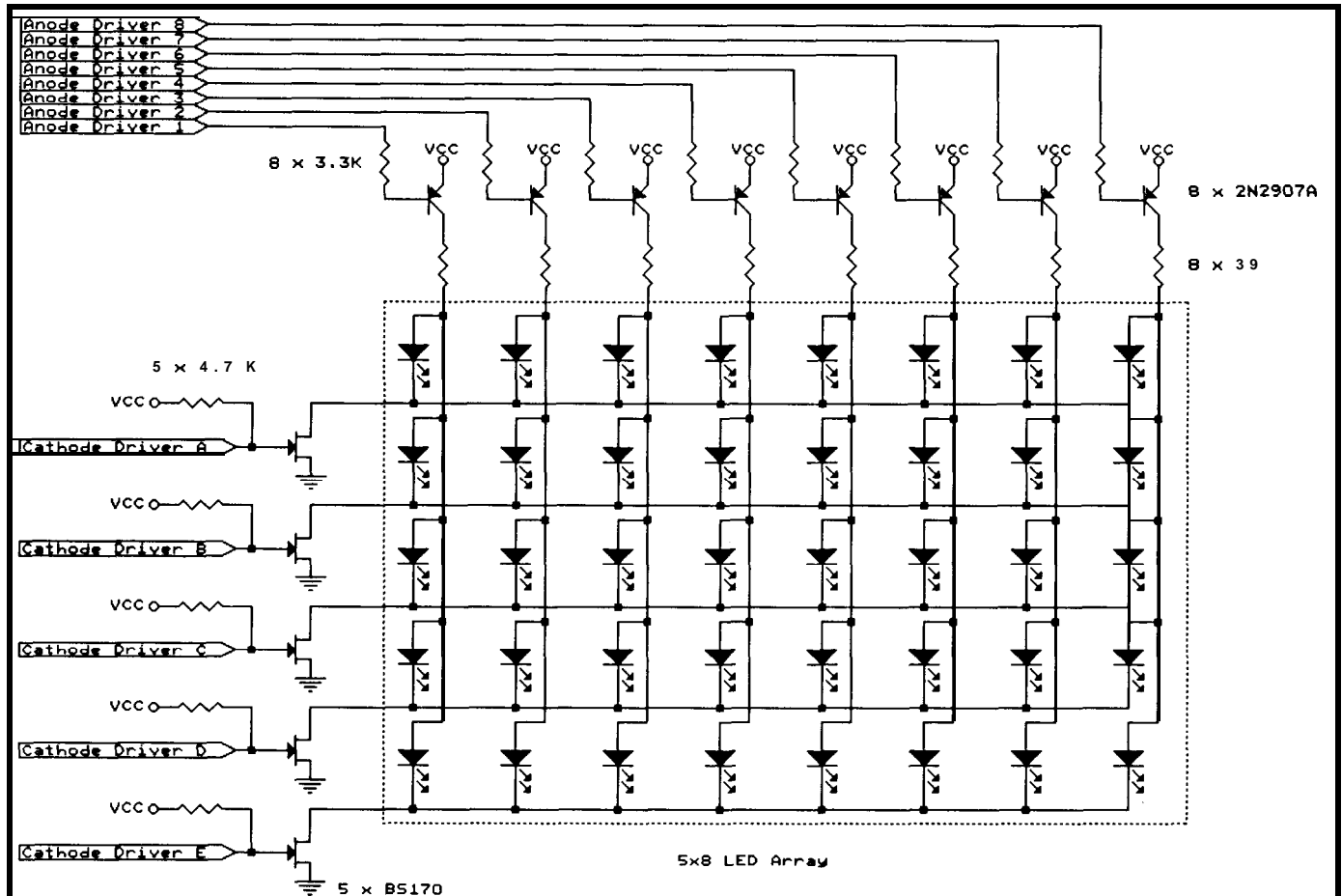


Figure 2—The flashy part of the sample circuit is the array of 40 LEDs arranged in five rows and eight columns.

hand." Situations like this make RISC architectures look good.

MODESEL is the input from the display mode jumper; depending on whether the jumper is installed or not, the LED bits are turned on or complemented when the Timers entries hit zero. In any case, the Timers entries are reloaded with a new (pseudo!) random value and begin timing anew.

We've looked at real random numbers before. There is one pin left over, so you should have no trouble monitoring an outside event to create truly random values. Say, a Geiger counter?

The AnodeData array is in Internal RAM, so changing a bit there has no effect on the LEDs. TmrTick0(), shown in Listing 2, gets control when Timer 0 generates an interrupt. It reloads Timer 0 to generate the next interrupt on schedule, copies an AnodeData entry to the LED anode drivers, and selects the next cathode driver to illuminate the corresponding row of LEDs.

HERE A VAR, THERE A VAR...

Now for the tricky part. Listing 3a shows the C language variable declarations for BLINKBOX.C and Listing 3b is the assembly language code produced by the 2500AD compiler, slightly edited to get rid of blank lines and suchlike. Notice how the variables are spread out over two different address ranges: those near 600 hex are in EPROM, while those near 10h are (or will be) in Internal RAM.

An ordinary variable, such as DataID, is a good starting point. These variables must be initialized to zero before the program begins, which must obviously be handled by the startup code. All we need to know is where the variable is located in Internal RAM (which is 001Bh) in order to plop a zero into it.

Initialized variables are more complex. The initial value must be stored in EPROM (otherwise it would vanish when the power goes off!), but the variable itself must be located in RAM so the program can change it. For example, RandomValue holds the current random number and is initialized to 1. The startup code must somehow

Affordable 8031 Single Board Computers

Control-R Model 1 (now includes MAX232) \$49.95
Our original 8031 SBC. The **Control-R 1** now includes the MAX232 chip to provide serial I/O and has 14 digital I/O lines that can be used to measure or control user designed circuitry. Requires 5vdc and measures 3.0" x 4.0".

Control-R Model 2 (now includes MAX232 & 8K SRAM) \$79.95
An expanded version of the Control-R 1. Now comes fully populated with 8K of SRAM and MAX232 for serial I/O. Expansion is provided by direct access to 8031 ports 1 and 3 as well as data, address, RST, ● INTI, ● WR, ● RD, ● PSEN, ALE and T1. Requires 5vdc and measures 3.5" x 4.5".

Datalog-R Model 1 (with MAX232 & 8K SRAM) \$179.95
The newest member of our SBC line. Features serial I/O, 4 ea. 8K device sockets user configurable as RAM or ROM, expansion connection and a socket that will accept an (optional, \$70.00) 32K byte removable memory card. This SBC is designed for applications requiring removable, non-volatile storage or can be used without the memory card as a conventional SBC. Requires either 5vdc or 12vdc for operation and measures 6.0" x 4.5".

To place an or&r contact:

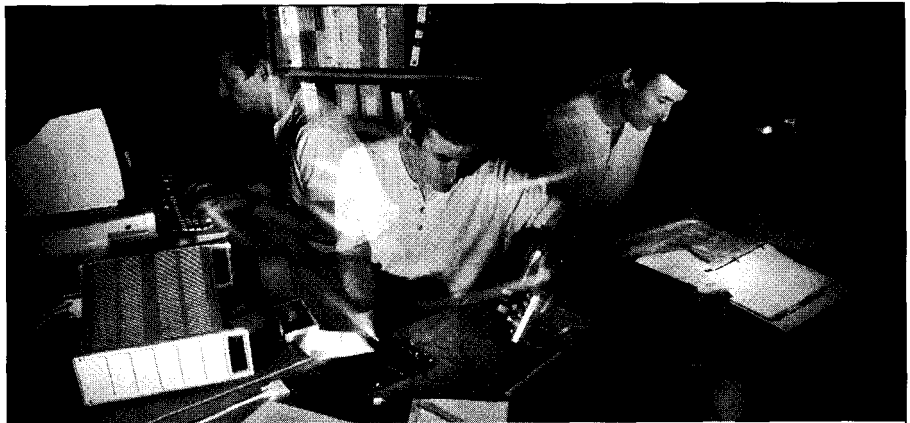
Cottage Resources Corporation

Suite 3-672, 1405 Stevenson Drive

Springfield, Illinois 62703 USA

1-2 17-529-7679 • Visa, Mastercard, or COD Orders accepted

Reader Service #130



MultiTask!™ Executives Accelerate Real-Time Design

In a hurry to develop real-time applications? MultiTask! executives give you a full set of standard system services for most embedded processors.

Source code keeps you in control. And our ProtoTask!™ executive lets you develop code on the PC. No matter which target you choose or when you choose it.

MultiTask! executives support today's most popular microprocessors:

680x0, 68HC11, Z80/Z180/64180, 8051, 80x86/V-Series, 8096/80196 & i960.

Call for a free EasyTask!™ information diskette: (800) 356-7097 or (503) 641-8446.



14215 NW Science Park Drive
Portland, OR 97229

© 1991 US Software Corporation MultiTask!, ProtoTask! and EasyTask! are trademarks of US Software Corporation.

U S SOFTWARE®

Reader Service 6210

See us at The Embedded Systems Conference Booth #701
October/November 1991 73

connect the initial value in EPROM at 065Bh with the variable's true Internal RAM address.

More peculiar are "constant" variables like CathodeBits, which will never be changed during execution. Because the value will never change, the variable can be located in EPROM (at 061Ch) and there is no need to waste precious Internal RAM. However, depending on the compiler and memory model, you may find erst-while constants copied from EPROM to RAM. Some compilers do not support the const keyword, which makes the whole discussion academic.

The BLINKBOX program is simple enough that you could initialize all the variables by name. The right way, however, is to collect all variables of the same type together and treat them as a group. The solution involves (brace yourself!) segments.

SETTING SEGMENTS

Despite the evil reputation segments have gotten in the Intel 8086

```
#define MODESEL SFB.P3_6

while (1) {

    for (TimerID=0; TimerID<NUMTIMERS; TimerID++) {
        DataID = TimerID / 8;          /* locate the bit */
        BitMask = 0x01 << (TimerID % 8);

        Timers[TimerID]-;            /* tick the timer */

        if (MODESEL) {
            if (!Timers[TimerID]){    /* timed out? */
                AnodeData[DataID]^= BitMask; /* flip the bit */
                Timers[TimerID] = GetRandom(MAXTIME) + 1; /* reload */
            }

            else {
                if (Timers[TimerID]){  /* still running? */
                    AnodeData[DataID] &= --BitMask; /* yes, so shut off */
                }
                else {                 /* blink! */
                    AnodeData[DataID] |= BitMask;
                    Timers[TimerID] = GetRandom(MAXTIME) + 1; /* reload */
                }
            }
        }
        TmrWaitScan(SCANDELAY);      /* force delay */
    }
}
```

listing 1 —The 'main loop' of BLINKBOX.C51 decrements the Timers array and flips a bit-mapped LED corresponding to each zero element. The TmrWaitScan function forces a pause while the interrupt handler updates the LED array.

world, the fundamental idea isn't bad at all. A "segment" simply contains a group of similar items, be they functions, initialized variables, constants, or whatever. Perhaps because of the

bad press, 2500AD uses the term "section" to describe their groups.

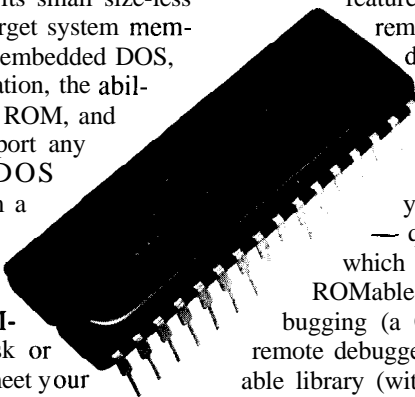
The Data ID variable is in a section called internal_unit_data because that is where the 2500AD com-

Professional Quality

80x86 ROM Development Tools

ROMable DOS, Only \$6 each!

Place your programs in PROM with ROM-DOS. Purchased in quantity, this complete ROMable operating system costs only \$6 per copy. And its small size—less than 32K—helps conserve valuable target system memory. ROM-DOS was designed as an embedded DOS, providing features like power conservation, the ability to run your executables directly in ROM, and full access to built-in devices to support any kind of target system. Plus ROM-DOS provides the power you'd expect from a desktop DOS: a full-featured command processor, extra utilities, batch file support, and more. And now it is compatible with MS-DOS 3.3. ROM-DOS boots and runs from either disk or ROM and can easily be configured to meet your needs. Ask for our free demo disk.



ROM Your MS & Turbo C Code!

Now C_thru_ROM eases ROM development with all versions of Turbo-C, C++, and Microsoft C. New features include support for Turbo Debug and remote debugging via a ROM socket, which does not use a serial port on the target system. C_thru_ROM is a complete ROM development package containing everything you need to work with your choice of compiler and get your application up and running in ROM — quickly. It includes: a full 80x86 locator which outputs in Intel OMF, hex, and binary; ROMable startup code; two choices for remote debugging (a CodeView-like debugger or the Turbo remote debugger); full floating point support; a ROMable library (with printf, malloc, etc.); and much more. Call, write, or fax Datalight today for full details.

For more information Call Today Toll-Free 1-800-221-6630

Datalight

17455 68th Avenue NE, Suite 304, Bothell, WA 98011, USA • (206) 486-8086 • fax (206) 486-0253

```

#define SCOPESYNC SFB.P3_7
#define REFRESH -3333
#define INTOVERHEAD 63

interrupt TmrTick0() {

/*-- reset timer with correction for interrupt setup routine */

SFB.TRO = 0;

THO = (BYTE) ((REFRESH+INTOVERHEAD) >> 8);
TLO = (BYTE) ((REFRESH+INTOVERHEAD) & 0xFF);

SFB.TRO = 1;

/*-- update the display */

SCOPESYNC = 1;

P3 &= -ALLCATHODES;          /* turn off cathode drivers */

if (NUMCATHODES == ++CathodeID) { /* step to next cathode */
  CathodeID = 0;
  Scans++;                    /* indicate a complete scan is done */
}

P1 = ~AnodeData[CathodeID];  /* send this row, inverted */

P3 |= CathodeBits[CathodeID]; /* lights on again */
MODESEL = 1;                 /* ensure input bit is enabled */

SCOPESYNC = 0;
}

```

listing 2—This interrupt handler updates the LEDs with the next row of eight bits from internal RAM.

piler places all variables that don't require initialization (to anything other than zero). If the program consists of several source modules, the linker will combine all the internal `_uninit_data` sections from each file into a single block.

Knowing where that block starts and how long it is, we can write a simple loop to zero the whole section in one shot. We don't need to know the identity of the variables, where

they came from, or what they do for a living. Listing 4 shows the 2500AD startup code for this task; the rather Teutonic naming convention makes it easy to see what's going on: the section starts at internal `_uninit_data_addr` and continues to `_internal_uninit_data_end_addr`, with the total length being just the difference between those two values.

The `internal_const_data` section groups all `const` variables such

```

const BYTE CathodeBits[] = {0x01,0x02,0x08,0x10,0x20};

#define NUMCATHODES (sizeof(CathodeBits)) /* LEDs down display */
BYTE CathodeID;                          /* current cathode selector */

#define NUMANODES 8                       /* LEDs across the display */
BYTE AnodeData[NUMCATHODES];             /* outgoing bitmap data */
BYTE DataID;                             /* index into AnodeData */
BYTE Counter;                            /* utility counter */

BYTE Scans;                              /* incremented for each full scan */

#define NUMTIMERS NUMCATHODES*NUMANODES

WORD Timers[NUMTIMERS];                  /* countdown timers for blinks */
BYTE TimerID;
BYTE BitMask;

unsigned long RandomValue = 1L; /* src of pseudorandom numbers */

const char *CopyRight =
  "Copyright 1991 Circuit Cellar INK/Ed Nisley";

```

listing 3a—This code defines the variables used by BLINKBOX. The `const` keyword identifies 'variables' that will not be changed during the program.

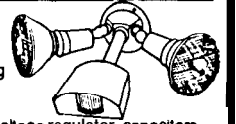
Courteous Service • Discount Prices • Fast Shipping

ALL ELECTRONICS

P.O. Box 567 • Van Nuys, CA 91408

INFRARED SECURITY LIGHT (AS-IS)

Experiment with infrared sensors with these outdoor security lights. Contain lots of interesting components, and IR detector, photoresistor, relay, transformer, IC's, voltage regulator, capacitors, trim pots and other goodies. Returned to the distributor for variety of reasons, we've found that most of them work to some extent. We don't want to test them and would prefer to sell them "as-is" at a greatly reduced price. Mounts to any standard electrical junction box. Infrared sensor detects movement up to 65 feet and turns on lights. Sensor can be adjusted for sensitivity and duration of fighting. The position of the sockets and the infrared sensor can be easily adjusted. Will handle two 150 watt PAR 36 lamps. Suitable for wet locations. Bulbs not included. CAT# IL-101 \$7.50 each



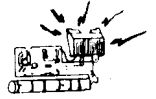
HALL EFFECT LATCH

Sprague # UGN3075LT
Operates on 4.5 - 24 Volts
Can sink 10 ma. With suitable output pull up, can be used directly with bipolar or CMOS logic circuits. Especially suited for electronic commutation in brushless D.C. motors using multiple ring magnets. Very tiny surface mount package 0.175" X 0.09" X 0.06" thick.
CAT # HESW-5 2 for \$1.00
100 for \$45.00 Large quantities available



FLASH ASSEMBLY

This NEW compact flash assembly comes from a U.S. manufacturer of cameras. Unit operates on 3 Vdc and measures 2 1/2" X 1 1/4". Ideal for use as a strobe, warning light or attention get-ter. Complete with instruction on how to wire.
CAT# FSH-1 \$3.75 each 10 for \$35.00



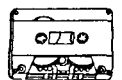
ULTRASONIC CERAMIC MICROPHONE/TRANSDUCER

Panasonic (Matsushita) # EFR RCBK40K54
An ultrasonic microphone consisting of a bimorph type piezoelectric ceramic vibrator. Ideal for burglar alarms, auto door openers, flow rate detectors and remote control systems. Nom. Freq. 40kHz. Max input volts: 20 Volts. 15/16" diameter X 3/8" high. 5/8" long leads.
CAT# UST-1 \$1.00 each



HIGHEST QUALITY METAL C-60 CASSETTES (Erased)

Premium quality metal tape in C-60 cassettes (30 min. per side). One of the finest "brand-name" tapes on the market, in durable, clear plastic transport mechanisms. Recorded and bulk erased, the remrd-protect tabs have been removed and therefore, need to be taped over to re-record. Audiophiles will appreciate the wide dynamic range of this tape. If your cassette deck has a "metal" setting you will hear the difference. A real bargain!
CATX C-60M \$1.25 each. 10 for \$10.00



TOLL FREE ORDER LINES 1-800-826-5432

CHARGE ORDERS to Visa, MasterCard or Discover

TERMS: Minimum order \$10.00 Shipping and handling for the 48 continental U.S.A. \$3.50 per order. All others including AK, HI, PR or Canada must pay full shipping. All orders delivered in CALIFORNIA must include state sales tax (6%, 6 1/4%, 6 1/2%, 7%) Limited Quantities NO C.O.D Prices subject to change without notice.

Call or Write For Our
FREE 60 Page Catalog
(Outside The U.S.A. Send \$2.00 Postage)
ALL ELECTRONICS CORP.
P.O. Box 567 • Van Nuys, CA • 91408

New Low Cost In-Circuit Emulator

The DryICE Plus is a modular emulator designed so you can get maximum flexibility from your emulator purchase. The base unit contains all the hardware necessary to support pods containing many of the most popular members of the

8051 family of embedded control microprocessors. Buy one base unit, and select one or all of the pods you need to do the job at a much reduced cost. You get the same great functionality found in our popular DryICE 8031 emulator plus real-time Execute-to-Breakpoint, Line-by-Line Assembler, and much more. And the price is (almost) unbelievable! (Yes, it works with the Mac, too!)

Base Unit (w/RS-232IF) -- \$299
Available Now!

Available Pods:
8031/32, 80C31/32, 80C154, 80C451,
80C535, 80C552/562, 80C652, 80C51FA,
8751/52, 87C51/52--**\$149** each

16K Trace Buffer option: Avail. 1st Qtr '92
Standard 8031 DryICE -- Still only **\$199**
Enhanced 8031 DryICE -- **\$269**

8051 Simulation

The 8051 **SIM** software package speeds the development of 8051 family programs by allowing execution and debug without a target system. The 8051 **SIMulator** is a screen oriented, menu command driven program doubling as a great learning tool. \$99.

8031/51 Single Board Computer

A fast and inexpensive way to implement an embedded controller. **8031/32** processor, **8+** parallel I/O, up to 2 RS232 serial ports, **+5** volt operation. The development board option allows simple debugging of **8031/51** family programs. **\$99ea**

Call for your custom product needs.
Free Quote - Quick Response

Other products available:

MyGAL - GAL Programmer \$199

FORTH Card - FORTH development card for STD Bus \$279 (OEM-\$199)



HTE

HiTech Equipment Corp
9400 Activity Road
San Diego, CA 92126
(FAX: (619) 530-1458)

(619) 566-892

as CathodeBits in EPROM. The startup code need take no action for these "variables" because the compiler has already created the code to access them.

The initial values of variables such as RandomValue are combined into internal init data in EPROM. The section's Internal RAM starting address is internal data `_init_addr`, but the EPROM address depends on a trick the 2500AD section definitions concatenate the initialized data after the constant data. Other compilers use different methods to determine the addresses, but the final loop looks much the same.

BLINKBOX needs no more Variable initialization because it doesn't use External RAM. The 2500AD compiler comes with startup routines for

programs that use Internal RAM, External RAM, or a combination of the two; pick the one you need.

Some compilers support initialized bit variables, either as ordinary bytes with software bit extraction or directly in 8051 bit RAM. Although the 2500AD compiler includes bit variables that can be coerced into the bit RAM, they cannot be initialized, so your startup code doesn't have that to worry about.

FINAL BEGINNINGS

In addition to setting up the variables, the startup routine must load the stack pointer (SF) with the stack's Internal RAM address. The 8051 hardware stack grows upward from the current SP value; pushing a byte on

```

061C 01      _CathodeBits:      .internal const_data
061D 02      .byte 001h
061E 08      .byte 002h
061F 10      .byte 008h
0620 20      .byte 010h
0620 20      .byte 020h

                                ;unsigned char CathodeID;
0015      _CathodeID:      .internal _uninit_data
                                .ds 1

                                ;unsigned char AnodeData[(sizeof(CathodeBits))];
0016      _AnodeData:      .ds 5*1

                                ;unsigned char DataID;
001B      _DataID:      .ds 1

                                ;*unsigned char Counter;
001C      _Counter:      .ds 1

                                ;*unsigned char Scans;
001D      _Scans:      .ds 1

                                ;unsigned int Timers[(sizeof(CathodeBits))*8];
0621      .i_align
001E      _Timers:      .ds 40*2

                                ;*unsigned char TimerID;
006E      _TimerID:      .ds 1

                                ;*unsigned char BitMask;
006F      _BitMask:      .ds 1

                                ;unsigned long RandomValue = 1;
065B 0000 0001 _RandomValue: .internal _init_data
                                .long 1

0621 43 6F 70 79 72 ___STP0:  .internal const_data
                                .byte "Copyright 1991 "

                                .byte "Circuit Cellar "
0621 69 67 68 74 20
                                .byte "INK/Ed Nisley",0
0621 31 39 39 31 20
                                .word ___STPO
0621 43 69 72 63 75
0621 69 74 20 43 65
0621 6C 6C 61 72 20
0621 49 4E 4B 2F 45
0621 64 20 4E 69 73
0621 6C 65 79 00
064D 0621      _CopyRight:      .word ___STPO

```

listing 3b—The 2500AD compiler produces this assembly language code from Listing 3a. Addresses near 600h are in EPROM and those near 10h are in Internal RAM.

the stack increments the stack pointer and stores the byte. As a result, standard C library functions which assume a "growing down" stack will give the wrong results on an 8051.

Depending on the compiler and memory **model**, the internal stack may be used for function arguments as well as return addresses. However, it seems 8051 C compilers also pass arguments in registers as well as on the stack, and may stack arguments in nonstandard ways. If you plan to write assembly language functions that will accept parameters or return values, make sure you read the documentation on how it works, then write some test code to verify that the **doc** is actually correct.

Just to increase the confusion, stacks in External RAM tend to grow downward. If you write a mixed model program, return addresses **grow** up in Internal RAM and arguments grow down in External RAM. And you thought this was going to be simple, right? You have been warned.. .

The catch with Small model is that miserly 128 bytes of Internal RAM. Expect severe memory problems for any but the tiniest of programs. If your startup routine puts the stack after all the other variables, you can use an 8032 to get 128 more bytes of Internal RAM without having to change your code.

Nearly all 8051 code, including compiled C code, assumes the **working registers are in Bank 0**, which starts at Internal RAM address 00h. The startup code should set the **PSW** to ensure this, as errors resulting from a misplaced register bank are devilishly hard to isolate. For example, library functions often use the direct-address equivalents of the registers: **R1** is Internal RAM location 01h when Bank 0 is active, but it's 19h in Bank 3. Imagine the confusion if you store something into **R1** (at 19h) and then do a `CJNE A, 01h` expecting to test **R1**. The code looks OK, so you can spend a lot of time searching.

```

;-- Clear uninitialized variables to zero
$no_internal_init_data: .equal $
    mov    a, #.low.internal_uninit_data_end_addr
    clr    c
    subb  a, #.low.internal_uninit_data_addr
    mov    r4, a ;save lsb of uninit data size
    mov    a, #.high.internal_uninit_data_end_addr
    subb  a, #.high.internal_uninit_data_addr
    orl   a, r4 ;check for size = 0
    jz    $no_internal_uninit_data
    mov    r0, #.low.internal_uninit_data_addr
    clr    a ;zero uninitialized data area
$internal_uninit_loop: .equal $
    mov   @r0, a
    inc  r0
    djnz r4, $internal_uninit_loop ;dec uninit data size

$no_internal_uninit_data: .equal $

;-- Copy initial values from EPROM to Internal RAM

    mov    a, #.low.internal_init_data_end_addr
    clr    c
    subb  a, #.low.internal_init_data_addr ;init data size
    mov    r4, a ;save lsb of init data size
    mov    a, #.high.internal_init_data_end_addr
    subb  a, #.high.internal_init_data_addr ;init data size
    orl   a, r4
    jz    $no_internal_init_data ;skip init if = 0
    mov    dptr, #internal_const_data_end_addr ;data addr
    mov    r0, #.low.internal_init_data_addr ;runtime addr
$internal_init_loop: .equal $
    clr    a
    movc  a, @a+dptr ;load init data byte
    inc  dptr ;increment source address
    mov  @r0, a ;store init data byte
    inc  r0
    djnz r4, $internal_init_loop ;dec init data size

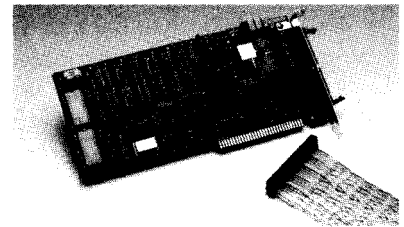
```

Listing 4—This part of the 2500A D startup code clears the uninitialized variables to zero and sets the values of initialized variables. The rather long values starting with 'internal_' correspond to the section names in Listing 3.

Real Time Devices

"Accessing the Analog World"

Quality U.S.-manufactured PC Bus cards and software for single user, OEM, or embedded applications.



AD3700 - \$395

200 kHz THROUGHPUT

- 8 S.E. analog inputs, 12-bit 5 µsec A/D
- FIFO interface & DMA transfer
- Trigger-in and trigger-out; pacer clock
- 4 Conversion modes & channel scan
- 4 Independent timer/counters
- 16 TTL/CMOS digital I/O lines
- Assembler, BASIC, Pascal & C source code

DataModule PRODUCTS

Plug-compatible with Ampro CoreModule
DM402 12-bit 100kHz analog I/O board with trigger, T/C, DMA & 16 DIO lines \$395
DM602 12-bit 4-channel D/A; voltage range select; current loop & DIO control \$289
DM802 24-Line opto 22 compatible 82C55 PPI-based DIO interface \$ 1 4 9

POPULAR XT/AT PRODUCTS

AD1000 8 S.E. 12-bit A/D inputs; 25 kHz throughput; three 8-MHz timer/counters; 24 PPI-based digital I/O lines. \$ 2 7 5
ADA1100 AD1000 with 38 kHz throughput, 2 D/A outputs, and configurable gain \$365
ADA2000 8 Diff./16 S.E. analog inputs; 12-bit 20 µs A/D; 12 or 8 µs A/D optional; two 12-bit D/A outputs; programmable gain: 3 T/Cs; 40 DIO lines from 82C55 PPI \$489
ADA3100 8 Diff./S.E. 12-bit analog inputs; 200 kHz throughput; gain select; FIFO interface & DMA transfer; pacer clock; external trigger; 4 conversion modes, multi-channel scan & channel burst; 4T/Cs; 16 DIO lines; two fast-settling analog outputs \$659
AD510 8 S.E. inputs; 12-bit integrating A/D with programmable gain \$259
ADA900 4 Diff./S.E. inputs; 18-bit V/F type A/D; variable resolution & conversion speed; 16-bit @ 16 Hz; 12-bit D/A, T/C & 16 DIO lines \$410
DA600/DA700 Fast-settling 2/4/6/8 -channel 12-bit D/A; double buffered \$192/\$359
DG24/48/72/96 Digital I/O lines; 82C55 based; optional buffers & line resistors \$110/\$256
TC24 Am9513A System Timing & 82C55 Digital I/O control card \$218
MX32 External analog multiplexer \$198
ATLANTIS/PEGASUS/PEGASUS Acquire Menu-driven, real-time monitoring, control, data acquisition and analysis turn-key software packages \$150/\$290

Call for your Free Catalog!

RTD logo, "Accessing the Analog World", and DataModule are trademarks of Real Time Devices, Inc. AMPRO and CoreModule are registered trademarks of Ampro Computers, Inc. opto 22 is a registered trademark of Opto 22, Inc.

Custom/OEM designs on request!

Real Time Devices, Inc.

State College, PA USA

rted Tel.: 814/234-8087
 FAX: 814/234-5218

Reader Service #196

October/November 1991

77

With all that in place, the startup routine can simply call the `main ()` function to begin the program. Most embedded applications have a "do forever" loop, so `main ()` will never return. If it should, the startup code had probably best branch back to the beginning, reinitialize the variables, and start over.

Of course, your C code begins with more initializations, but the mysterious startup code is finished. It's in your hands now... for better or worse.

SURPLUS SETUPS

The 2500AD startup code is designed to work with their hardware development board and includes serial drivers to support console I/O. Several equate statements define the memory model and other parameters, then a bunch of `#if` statements automatically select the right routines. This works well if you have their hardware and want to use their serial I/O, but neither is true for BLINKBOX and probably won't apply to your situation.

The code in Listing 4 is an excerpt from a severely edited version of the 2500AD startup code that will initialize simple Small (or Internal) model programs that don't use serial I/O. Even if your project doesn't fit that description, looking over my `STARTUP I.A5 1` file will help you sort out what is essential in the standard `C8051x.SRC` files.

The interrupt keyword is a recent addition to the C language which allows you to write interrupt handlers in C rather than assembler. Because an interrupt can occur at any time, the handler must save (and restore!) the CPU's working registers so the interrupted code is not affected. The 2500AD compiler inserts calls to `_interrupt_entry` and `_interrupt_exit` routines around each handler: the logical place for those routines (which must be written in assembler) is the startup code file.

The Avocet compiler, on the other hand, inserts boilerplate `_entry` and `_exit` code around each handler. The 2500AD approach seems better be-

cause you can tailor the routines to your needs. For example, the 2500AD code reserves another work area on the stack under the assumption that your interrupt handler will call the reentrant library routines. BLINKBOX doesn't need this, so I could excise that code and save a considerable amount of stack space.

I cut out the serial handlers because BLINKBOX has no serial I/O functions. In any event, I don't think that code belongs in the startup file: better you should have a separate collection of serial drivers.

READY TO RUN

The BBS files for this column include the `BLINKBOX.C5 1` code, my modified startup code file, and the hex file resulting from the compilation. Even if you don't have the 2500AD compiler, you should be able to use the `STARTUP I.A5 1` file to see what kind of setup you will need when you tinker with your compiler.

[Editor's Note: Software for this article is available from the Circuit Cellar BBS and on Software On Disk #23. See page 105 for downloading and ordering information.]

BLINKBOX requires more than 128 bytes of Internal RAM, so you must use an 8032 instead of an 8031. The obvious savings from making Timers an array of `BYTE` isn't quite enough, but changing the random number to an unsigned `int` might do the trick by eliminating a number of library routines that chew up stack space. Try it with your compiler and see.

Next time, I'll take a look at Dave Dunfield's Micro-C, a shareware C compiler that just might be what you're looking for. Stay synched! ❖

Ed Nisley is a Registered Professional Engineer and a member of the Circuit Cellar INK engineering staff. He specializes in finding innovative solutions to demanding and unusual technical problems.

IRS

4 19 Very Useful
420 Moderately Useful
421 Not Useful

Master PC/XT/AT with NBS-10 or NBS-2 Interface Card

INTEL 8051 HVAC Management

MOTOROLA 68HC11 Data Acquisition

INTEL 8096 Automotive Control

ZILOG Z180 Security Systems

Embedded Controller Networks
 Low Cost - High Performance

Cimetrics Technology's 9-Bit Solution networks popular 8- and 16-Bit microcontrollers by taking advantage of the built-in 9-Bit multiprocessor modes.

9 Benefits of the 9-Bit Solution

- Lowest cost embedded controller networking alternative
- Compatible with your micro-controllers
- Multi-drop Master/Slave RS-485 network
- 250 nodes per network
- High speed (62.5K baud) with low network overhead
- High reliability protocol includes CRC checking
- Low resource requirements no external RAM required
- Simple to use software toolkit
- Applications include data acquisition and distributed control

Cimetrics Technology's 9-Bit Solution includes the NBS-2 (\$165) or NBS-10 (\$249) PC/XT/AT network interface, the NSP (\$750) or Tiny-NSP (\$199) network protocol software toolkit, and network monitor software (\$225).

Cimetrics Technology
 120 West State Street
 Ithaca, New York 14850
 (607) 273-5715 / (607) 273-5712 FAX

See us at The Embedded Systems Conference South #418

Reader Service # 122

Redefining Remote Control

FROM
THE
BENCH

Jeff Bachiochi

Now You See 'em--Beep-Now You Don't

Thank you American Airlines. It is great to be back in the USA. Don't get me wrong, it's not that I didn't like Mexico. It's just a bit difficult without knowing adequate Spanish. "Sí," "no," and "gracias" won't cut it when you need to hold an intelligent conversation. Like, "What do you mean I don't have the proper papers to pass customs? That's my plane!" Believe me, you don't argue with a uniformed guard brandishing a "pistola." I felt like a malformed part on an assembly line: Honk! The quality control inspector checks my specs and presses the reject button. Into the pile of discarded parts I fall, along with the other bodies who can't speak Spanish. I'll never say another bad word about airline counter personnel, the last refuge for a "gringo."

Mexico is a strange combination of high and low technology. Road repair is done with pickaxes, shovels, and wheelbarrows; no heavy equipment. Businessmen, however, wouldn't be caught dead without their hand-held cellular phones. It became obvious by the number of "boops" and "beeps" heard during lunch (from 2 PM to 5 PM) that this was when most deals were formulated. Deals like this one; the one which brought me to Mexico.

HANDS ACROSS THE BORDER

Being hired as a consultant is a lot like playing fireman. One has to know which type of extinguisher will put out the blaze in the shortest period of time, with the least amount of damage. If the fire has not grown too large, you can salvage enough to create success out of failure. The task at hand was not a three-alarm fire, but red tape promised to stoke the blaze.

The biggest failure of this project was a lack of firm specifications. The product had to be continually modified, because the rules changed depending on where the equipment was being used. Hand-wired prototypes would not work consistently in the field. This meant delivery schedules were now approaching impossible.

I want to see specifications before I start any project. Without rules, no one can play the game fairly. Designing data acquisition or any other type of equipment is impossible without complete requirements, I'm not suggesting this product had been designed without any specifications. I'm indicating that these were ideal specifications and only realistic in an ideal environment. The actual environment can play havoc with what you might expect to see. Environmental noise, from **subaudio** through RF, can have an impact on your data. In most cases, simply by looking at the data in the actual environment, you can determine if the environment is adding anything to the raw signal. If it is, you can make appropriate changes in the specifications to eliminate it. It is important that you and your customer agree to the specifications prior to any agreement or, like this company, you may never get out of the design stage.

LET'S SIMPLIFY THINGS

Simplifying a client's design can often result in major cost saving, especially when production quantities are high, and establishes the true value of hiring a consultant. For instance, this client's product used a **laptop** as a serial I/O device permanently attached to the product. Other hardware could be used here to simplify the product and make it more user friendly.

One of the most exciting **products** I've seen lately is the configurable legend switch (see "Silicon Update" in Issue

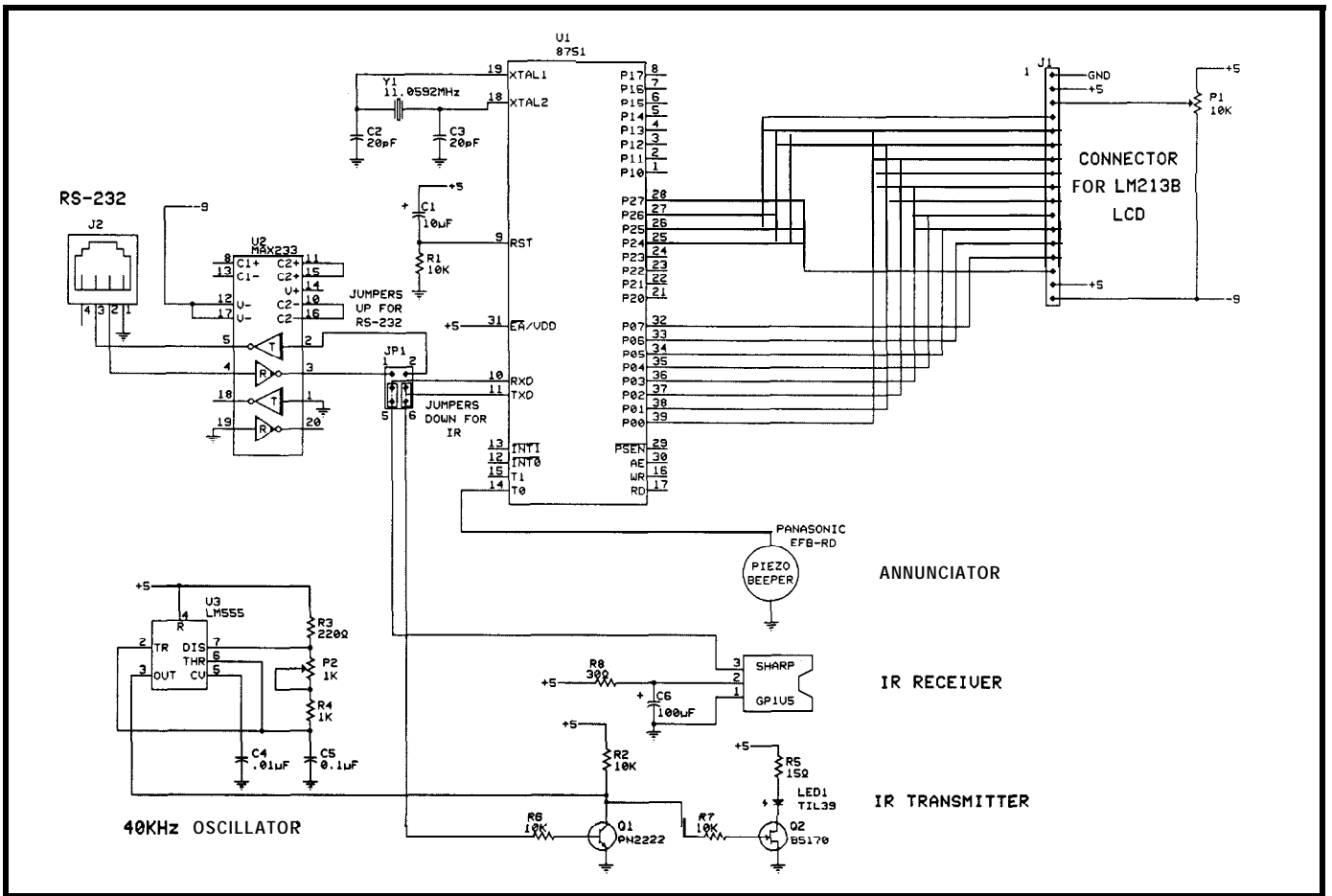


Figure 1 -The complete schematic for the hand-held LCD terminal doesn't include much more than an 8751 controller plus some simple interface circuits.

#20, April/May 1991, of *CIRCUIT CELLAR INK* for more info). These are expensive little buggers, but increase flexibility to the point of payback in many cases. This idea could be used to reduce the number of keys necessary on a product, like the one being developed in Mexico.

Every time I pick up my calculator, my mind goes into a whirl. Every key has a legend printed on it, as well as one above it and one below it. Some of them are printed in white, while others are printed in orange, green, or blue. It has a shift key, a mode key, and arrow keys. It doesn't have a help key. You might have such a calculator somewhere collecting dust because you've lost the manual. Unless I used it every day, I would quickly forget the correct key sequences to do various functions. Powerful, but not user friendly.

A better approach for I/O would be to use a simple LCD display to provide output and redefine the functions of a few keys. You can see this being introduced on some newer test equipment.

HASTA LA VISTA

During the five-hour flight back to Connecticut (thanks again, American), my mind slipped back to thoughts of home control. I couldn't think of a much simpler approach for I/O on my home control system.

I began scratching out bits of circuitry on the back of my drink napkin. My initial thought was to use an RTC52 with an RTC-LCD board, which has a LCD display driven by memory mapped I/O. I realized immediately that it would be overkill. A micro which receives ASCII serial input and directly drives an LCD panel could scan keys and send serial ASCII with few, if any, additional chips. An 8751 would serve well here. That would eliminate an address latch, RAM, and EPROM. With a few glue chips I could map the LCD into I/O space. No, that's not what Ed would do. I remembered his "Firmware Furnace" columns on LCDs and keyboards (see issues #8 and #16). I felt a bit of a challenge here-I'm not much of a software junkie. "Just to keep the glue to a minimum," I kept saying over and over, trying to convince myself that it was necessary.

PALM-SIZED TERMINAL

I've previously written routines for 4 x 20 and 8 x 40 LCD displays. Since the LCD will be mainly displaying menus for my home control system, and key definitions will take at least one line, the small display won't be adequate. This design will use the larger display which has a physical size of 2.5" x 8". See Figure 1 for the complete schematic of the terminal. Only the read and write subrou-

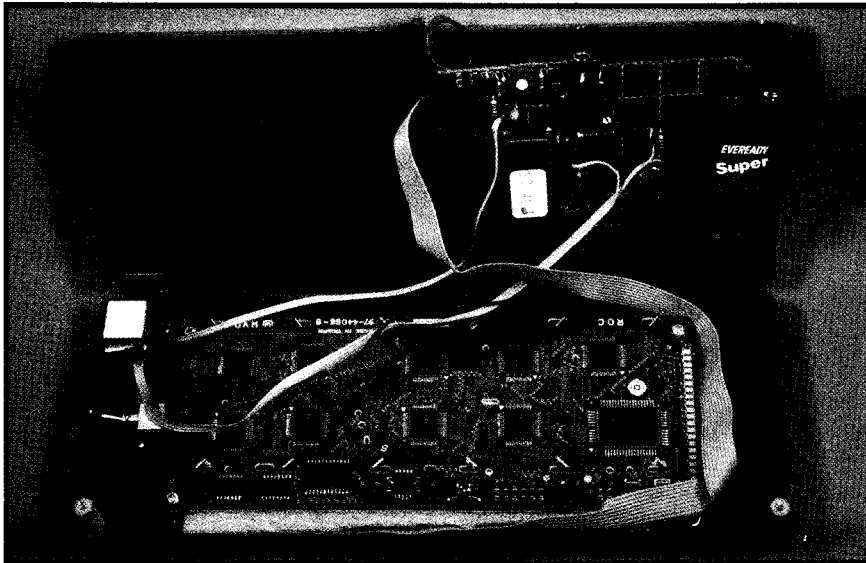


Photo 1 --The terminal consists of an LCD display (on the bottom), the processor board (on the raised portion) and a 9-V battery. Note the IR receiver module near the upper left corner of the display, with the IR transmitter diode next to it.

tines will need any severe tweeking, since these were written for memory-mapped I/O. The code actually ends up being a bit shorter by manipulating the LCD's control lines directly using CLR and SETB.

The display I chose features an HD61830 (an Hitachi dot matrix liquid crystal graphic display controller). The

screen is 340 dots wide by 48 dots high. The controller uses either a dot- or character-addressable mode. The latter uses an internal character set, making simple text display a breeze. The HD61830 controller is used on many LCD displays available today and interfaces easily to most MPUs with either a 4- or 8-bit data bus.

Through the LCD controller's 16 registers, you have control over the display mode, cursor position, display start within memory page, writing and reading a display address, setting or clearing a pixel (in dot mode), and verifying the status of the controller (busy). Once initialized, you needn't be bothered with refresh, character fonts, or display memory. If you have a limited number of I/O lines, the 4-bit interface is handy. However, remember it takes two nibble transfers to pass a byte of data, so there is a bit more work to be done per command.

Eight push buttons fit neatly below the 40-character display. They also form an 8-bit port which works out nicely. I use single-key entries, so the keypress loop is a simple port read to determine whether or not a key has

FLEXIBLE OPERATOR INTERFACE

Throw away your two- or four-line operator interfaces. Why be cryptic and unfriendly when you can make your new or existing machinery or instrument user friendly?

J.B. Designs have produced an intelligent stand-alone graphic controller (IGC) card. This card is designed to meet the requirements of user friendly man-machine interface in industrial systems or portable instruments. The card is like a stand-alone CGA card capable of connecting to existing or new equipment via RS-232/485 channel.

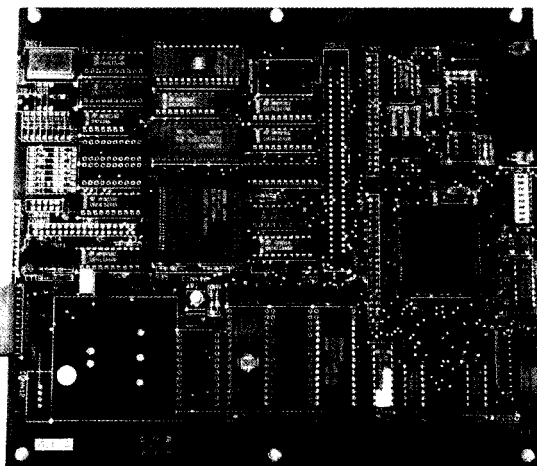
Standard software emulates a substantial subset of the AMPEX 210+ terminal. So you can treat the IGC as a standard VDU in simple applications. Other user-specific interfaces can be written in high-level languages on the PC and transferred to the board for running under a DOS emulator.

Either matrix or IBM-AT keyboards can be used. Touch screen interface is possible.



J.B. DESIGNS & TECHNOLOGY, LTD.

Cirencester, Glos. GL7 2PB, U.K.
Tel: (44) 285 658122 • Fax: (44) 285 655644



TECHNICAL SPECIFICATIONS:

- Based on PC code compatible V25 microprocessor
- Operate locally, or up to 1 km from host
- Drives any graphic panel size from 240 x 128 to 640 x 400 pixel
- CRT or vacuum fluorescent interface optional
- Supports a 6 x 4 matrix keypad in addition to an IBM-AT keyboard
- One RS-232 and one RS-232/485 serial channel
- Supports PC-compatible Time of Day Clock
- Eight-way DIL option switch
- Standard software emulates a VDU
- User-programmable in Microsoft "C" for special applications
- Three 32-pin memory sockets for RAM/EPROMs. One of these sockets contains the driver software for the LCD displays
- Works from a single 5V input power supply (other power input options also available)
- Size 200mm x 220mm
- Board power consumption approximately 200ma. at 5V

been pressed. The port has internal pull-ups and each of the port's eight bits are connected to one of the push-button switches. The other side of each switch is grounded. The port will read an FF hex if no switch is pressed or a low on the particular input bit connected to the pressed (grounded) key. The routine could be rewritten for multiple key entries or even a 4 x 4 keypad.

Let's take a look at a few possibilities using the display and eight keys. Figure 2 shows a simple one-key entry. From this "sub-sub-sub" menu (Living Room/X-10 Control/Lighting), the choices are exit back to the main menu, previous menu (X-10 Control), select a lighting device to control (see more items), or redisplay this screen (it may have been garbled in transmission).

If digits 0-9 are needed, it requires a two-key sequence as shown in Figure 3. If the column with 2, 5, 8, and 0 is chosen, a second screen is displayed eliminating the unselected items. Rotating the selected column eases definitive selection as shown in Figure 4. This can be expanded if necessary to include alphanumeric characters and still only require a two-key sequence (see Figure 5).

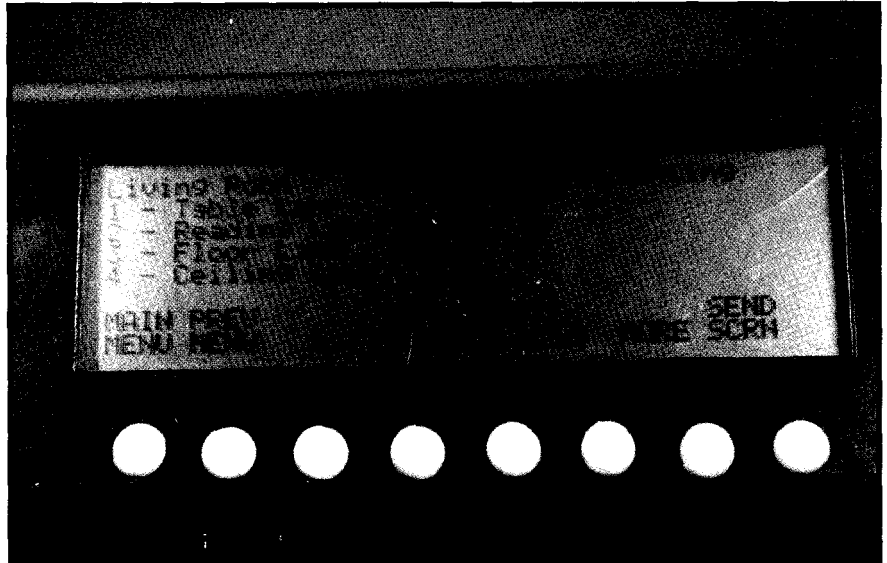


Photo 2—the 8 x 40 display allows the use of very descriptive menu selections and up to eight choices across the bottom of the display.

As you can see, there is a good deal of flexibility in the amount of information and how it is presented. Of course, if you think in binary, any of the ASCII characters can be entered directly by pressing the appropriate keys using seven of your ten fingers. For me and my ten thumbs, this is out of the question (besides the fact that I am acknowledging only single strokes).

X-10 TW523 DEVELOPERS KIT!

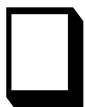
AT LAST !! Now you can TRANSMIT & RECEIVE X-10 COMMANDS under control of YOUR application program. We supply you with the I-O SOURCE CODE. Be up and running in MINUTES !! Write your own CLIPPER/BASIC program. We also have complete application packages. Shipped within 24 hours worldwide. Phone our BBS for complete pricing and details.

\$75

Each Kit contains:

- * X-10 TW523 Transceiver
- * Serial OR Parallel RJ-1 1/DB25 Interface
- * Microsoft/Turbo C I-O Drivers(source)
- * 6 foot cable
- * Technical Reference Manual

DEALER INQUIRIES WELCOME



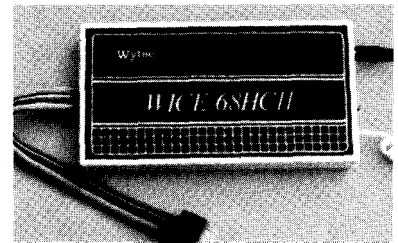
BARAN-HARPER GROUP
LOS ANGELES*TORONTO

VOICE 416-294-6473 BBS 416-471-6776

68HC11

DATA LOGGER

New PC driver converts WICE68HC11 to a real time data logger. Saves target data in a file while running user program.



- PC based user friendly ICE with data logging capability.
- Real time and full speed up to 14MHz clock rates.
- Single chip and expanded modes.
- Data watchwindows for memory, registers & stack.
- On-board 64K emulation RAM maps in 4K blocks.
- 64K real time hardware breakpoints.
- Breaks on address, address range, and memory RD/WR.
- Full symbolic debugging.
- Supports all A.E and D parts.
- 115.2K bps RS-232C link.
- 30 day money back guarantee.

Wytec

WICE 68HC11 emulator \$795.00 Suite 140
52 PLCC to 48 DIP adapter \$55.00 185C E. Lake Street
Call: (708) 894-1440 Bloomingdale, IL 60108

Z8

WICE Z8 emulator \$995
86C08 adapter w/analog comparators \$55

2500AD

We sell all 2500AD software products. Call us for low prices.

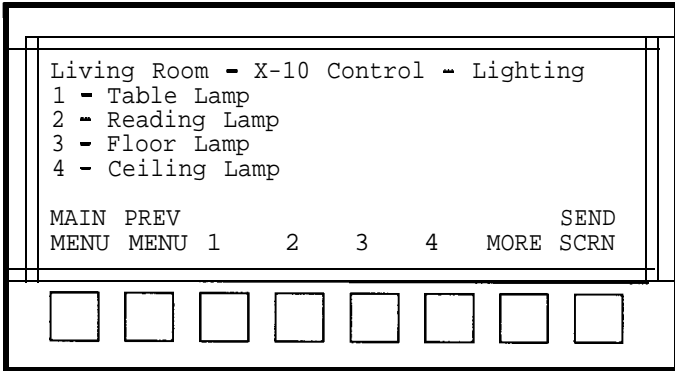


Figure 2—Several levels of menus can be easily traversed using the reconfigurable push buttons.

TALK BACK

If you use a PC, then you probably know when you hear a beep, it probably **means** something just went wrong. If you're using a Macintosh, you might hear a "raspberry" instead of a beep. In any case, this audible sound is enough to get your attention. These indicators give a gentle reminder (except, perhaps, in the case of the raspberry) that you should take a closer look at what you just asked the machine to do.

Similarly, there is a TSR I've seen that can generate a clicking noise whenever a key is pressed on the PC's keyboard. This gives an audible feedback when typing, similar to the old IBM keyboards or an even older typewriter. Not a big deal, but can give a bad typist a bit more confidence. Such feedback is almost mandatory on flat keypads, like the ones used on microwave ovens. You've probably seen the technique used everywhere, from octane selection at the gas station to your bank's ATM.

Piezoelectric devices produce a loud sound output for their size and are therefore a good choice for annunciators. A port pin on the microprocessor serves well as a piezo driver and indicates debounced keystrokes from the eight push buttons. It will also sound at the reception of a bell character (07 hex), from the serial port. This way, the source of the serial transmissions can get my attention or just indicate the end of transmission.

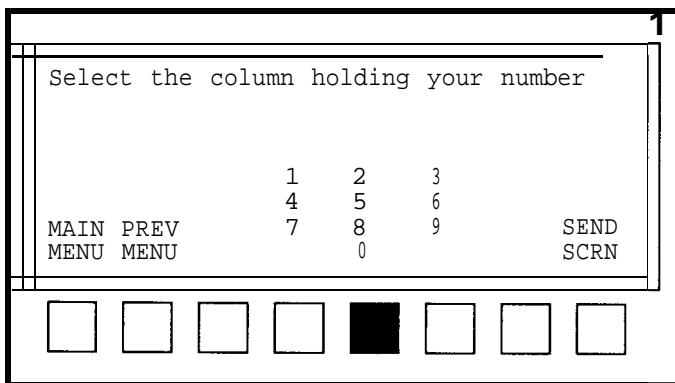


Figure 3—A two-step process is used when selecting from a large list of items. In the first step, the large group is divided down to a much smaller group with the first button press.

ProtoQuick... 8051 or Z8

Prototype boards with complete microprocessor circuits on board

Z8 or 8051 microprocessor circuit - 12 sq. in. prototype area
 Up to 32K EPROM, 8K RAM - 4.5 x 6.5" double-sided PCB, 5v only
 14 I/O lines and decoded DIP switch - Serial port w/ DB25 connector
 Op Sys in EPROM, w/ source code - RAM, I/O control from serial port
 Cross assembler included - CMOS and BASK: options available
 Assembled and tested - Ready for your next project

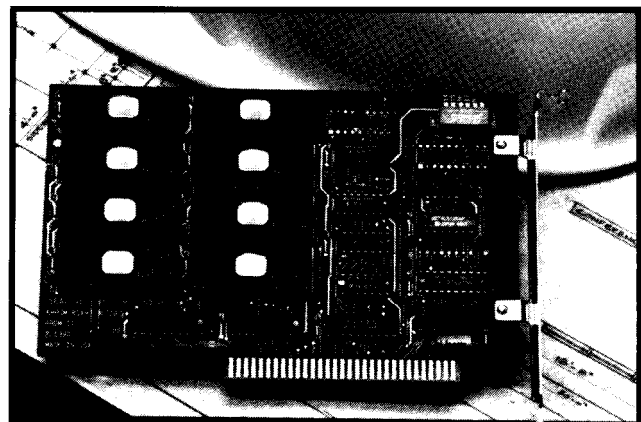
ea.

Software Science

3750 Roundbottom Road
 Cincinnati, OH 45244 USA

Call or write for technical flyer and application notes

Reader Service #203



PROM-III

- PUT DOS AND APPLICATION IN EPROM
- ALLOWS DISKLESS OPERATION
- UP TO 1 MBYTE ROM-DRIVE WITH 16K FOOTPRINT
- PROMKIT SOFTWARE BY ANNA-BOOKS
- FLASH EPROM SUPPORTED
- BATTERY RAM MODULES SUPPORTED
- DELIVERY FROM STOCK

SEALEVEL
 COMMUNICATIONS 81/0

SEALEVEL SYSTEMS INC.
 PO BOX 830
 LIBERTY, SC 29657
 (803) 843-4343

Reader Service #199

```

<ESC><*>      = CLS and HOME CURSOR
<ESC><=><#><#>  = MOVE CURSOR to #(row+20H)
                                     #(column+20H)

```

Figure 6—Two escape sequences are supported by the LCD display routines in the hand-held terminal.

THE SOFT SECTION

The software is written as two parts: the main body and the serial interrupt service routine. **[Editor's Note: Software for this article is available from the Circuit Cellar BBS and on Software On Disk #23. See page 105 for downloading and ordering information.]** The main body includes the initialization code and the key scan loop code. Initialization sets up the serial port (using timer 1), clears the beeper, calls the LCD `UOINIT` routine, and enters the key scan loop. This loop is where we stay as long as a key has not been pressed (or a serial interrupt has occurred). Once a key is pressed, we **debounce** it by pausing and rechecking to make sure it's still good. If it is, the beep routine is called and the appropriate character is placed into the transmit buffer. Once the key is released, execution returns to the key scan loop.

If a serial interrupt—either transmit or receive—occurs, execution branches to the serial interrupt routine. A check is made on the receiver first to give serial input priority over transmitting. Received characters are processed through the LCD routine as part of the serial interrupt. The LCD routine manages the display positioning and filters out two escape sequences (shown in Figure 6) from the serial data.

In addition, the bell character will call the beeper routine. It will normally be used to indicate that the system is awaiting a decision and a **keypress** would be appropriate.

Choosing the 87C51 for its internal serial port, timer, and EPROM, and using an LCD display which doesn't require any housekeeping makes for simple code. Fewer devices also means power savings. Although I could steal the smart power regulator I offered in last issue's column, a simple on/off switch is the simplest and least expensive approach. There is no need for the device to be able to turn itself on, at least none that I've thought of up to now. I already have an alarm clock, smoke alarms, and a door bell. Maybe if I added a **speech** synthesizer to the IR terminal..

On second thought, not now. I just found a package of airline peanuts in my pocket. I think I'll get me a cold one and relax. This one's for you, American! ❖

Jeff Bachiochi (pronounced "BAH-key-AH-key") is an electrical engineer on tk Circuit Cellar INK engineering staff. His background includes product design and manufacturing.

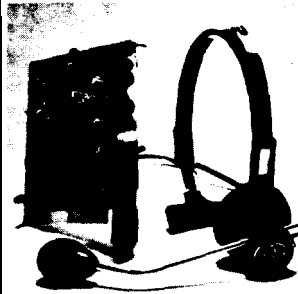
IRS

422 Very Useful
423 Moderately Useful
424 Not Useful

TALK TO YOUR COMPUTER

WITH VOICE MASTER KEY® FOR PCs/COMPATIBLES
VOICE RECOGNITION WITH SPEECH RESPONSE

GIVE A NEW DIMENSION TO PERSONAL COMPUTING The amazing Voice Master Key System adds voice recognition to just about any program or application. Voice command up to 256 keyboard macros from within CAD, DTP, word processing, spread sheet, or game programs. Fully TSR and occupies less than 64K. Instant response time and high recognition accuracy. A real productivity enhancer!



SPEECH RECORDING SOFTWARE Digitally record your own speech, sound, or music to put into your own software programs. Software provides sampling rate variations, graphics-based editing, and data compression utilities. Create software sound files, voice memos, more. Send voice mail through LANs or modem. A superior speech/sound development tool.

INTERACTIVE SPEECH INPUT/OUTPUT Tag your own digitized speech files to voice recognition macros. Provides speech response to your spoken commands -- all from within your application software! Ideal for business, presentation, education, or entertainment programs you currently use.

Augment the system for wireless uses in robotics, factory process controls, home automation, new products, etc. Voice Master Key system does it all!

EVERYTHING INCLUDED Voice Master Key System consists of a plug-in card, durable lightweight microphone headset, software, and manual. Card fits any available slot. External ports consist of mic inputs and volume controlled output sockets. High quality throughout, easy and fun to use.

ONLY \$149.95 COMPLETE

ORDER HO- (503) 3424271 Monday-Friday 8 AM to 5 PM Pacific Time
VISA/MasterCard phone or FAX orders accepted. No CODs. Personal checks subject to 3 week shipping delay. Specify computer type and disk format (3 1/2" or 5 1/4") when ordering. Add \$5 shipping charge for delivery in US & Canada. Foreign inquiries contact Covox for C & F quotes.

30 DAY MONEY BACK GUARANTEE IF NOT COMPLETELY SATISFIED.

CALL OR WRITE FOR FREE PRODUCT CATALOG.



COVOX INC.
675 CONGER ST.
EUGENE, OR 97402

TEL (503) 3424271
FAX: (503) 342-1283

Reader Service #132



ADVANCED COMMUNICATIONS PRODUCTS

SEALEVEL SYSTEMS PROVIDES THE EXACT COMMUNICATION CARDS YOU NEED

PRODUCTS

- 1, 2 OR 4 PORT RS 232 AND RS 422/485 BOARDS
- CURRENT LOOP SERIAL INTERFACES
- HIGH SPEED SYNC (HDLC, SDLC) AND ASYNC WITH DMA
- RS 530 AND V.35 INTERFACE BOARDS
- DIGITAL AND RELAY I/O BOARDS
- DISKLESS EPROM BOARD WITH PROMKIT SOFTWARE BY ANNABOOKS
- NEW LAP TOP ADD ON'S
- DELIVERY FROM STOCK
- MADE IN USA
- SATISFACTION GUARANTEED
- EXCELLENT TECHNICAL SUPPORT

SEALEVEL
COMMUNICATIONS & I/O

SEALEVEL SYSTEMS INC.
PO BOX 830
LIBERTY, SC 29657
(803) 843-4343

Reader Service #200

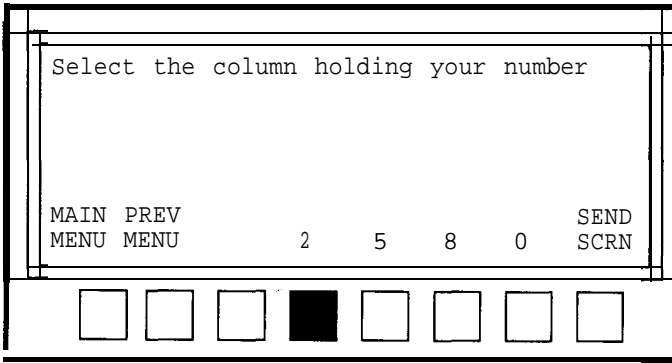


Figure 4—The next step after that shown in Figure 3 is to make a second selection from the smaller group.

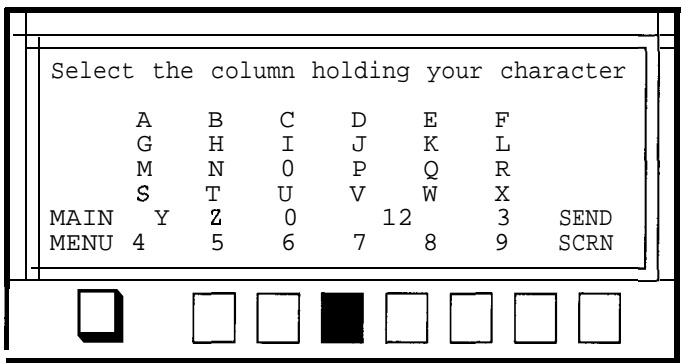


Figure 5—The entire alphabet plus numbers can be encoded using several button presses to select a particular character.

COMMUNICATION IS THE KEY

Communication between the hand-held terminal and the host computer can be done in one of two ways: infrared or hard-wired RS-232. I primarily use the IR interface, which I already described in "From The Bench" in issue #21. To help eliminate transmission and reception errors, I only use 7-bit ASCII characters, even though the display has a full 8-bit character set. Protocol is set by the home control (slave) unit. The slave listens for IR and responds only to the reception of ASCII characters 31h through 38h (ASCII characters 1-8), which are transmitted by the hand-

held master when any of the eight individual keys are pressed. The slave's response is an IR transmission of the appropriate ASCII message, which is displayed by the LCD. With this ping-pong approach, you can find your way quickly through predetermined menus to any number of acquisition and control functions.

The LCD display requires a negative bias voltage to boost the contrast. Since I'm already supporting a standard RS-232 port, I can steal some negative voltage from the MAX233 being used for the RS-232 interface to drive the display (the MAX233 is a MAX232 with internal charge pump capacitors).

The \$595 Solution to 8051 System Development

PDK51



- SIBEC-II microcontroller board (8052AH-BASIC CPU) with 16K RAM
- BASIC interpreter source on disk
- SIBEC-II hardware manual with schematic
- Monitor/debugger ROM plus manual
- Monitor/debugger hardcopy source listing
- UL listed 5V power supply
- Programming power supply adjustable from 4V to 26V
- KERMIT communications program
- SXA51 8051/8052 cross assembler
- BTK52 BASIC programmers toolkit with tutorial
- 160 page BASIC Programming Manual by Systronics
- RS232 cable
- PDK51 tutorial
- Optional aluminum case—\$45, monitor/debugger source—\$25, battery-backed real-time clock—\$39.
- All components are manufactured to exacting standards and warranted for one year.

The PDK51 is a fully integrated hardware, firmware, and software system designed to help you develop your products quickly and cost effectively.

All you need to use the PDK51 is an IBM-PC/XT/AT or compatible. We supply the rest.

PDK51 PLUS includes everything in the PDK51 plus Vers. 3 of our popular BXC518051/8052 BASIC compiler—**\$800.**

Call Now! 603-469-3232 or FAX 603-469-3530

 **Binary Technology, Inc.**
Main Street • PO Box 67 • Meriden, NH 03770

Reader Service #116



CROSS-32 V2.0 META ASSEMBLER

- Table based absolute **macro** cross-assembler using manufacturer's assembly mnemonics.
- Includes manual and MS-DOS assembler disk with tables for ALL of the following processors:

16C5X	64180	6801	8048	H8/300	Z8
	37700	6502	6805	H8/500	280
	50740	65816	6809	TMS320	2180
78C10	COP400	6811	8086	TMS340	Z280
SUPER8	COP800	68000	8096	TMS370	MORE...

- Users can create tables for other processors or ask us, we have many more!
- Generates listing, symbol table and binary, Intel, and Motorola **hexcode**.
- Free worldwide airmail shipping & handling.
- Canadian residents please add 7% GST

US \$199.00 CN \$239.00

UNIVERSAL CROSS-ASSEMBLERS
P.O. Box 6158, Saint John, N.B., E2L 4R6
Canada Voice/Fax: (506) 847-0681




Reader Service #21


```

<ESC><*>      = CLS and HOME CURSOR
<ESC><=><#><#>  = MOVE CURSOR to # (row+20H)
                                   # (column+20H)

```

Figure 6—Two escape sequences are supported by the LCD display routines in the hand-held terminal.

THE SOFT SECTION

The software is written as two parts: the main body and the serial interrupt service routine. *[Editor's Note: Software for this article is available from the Circuit Cellar BBS and on Software On Disk #23. See page 105 for downloading and ordering information.]* The main body includes the initialization code and the key scan loop code. Initialization sets up the serial port (using timer 1), clears the beeper, calls the LCD U01 INIT routine, and enters the key scan loop. This loop is where we stay as long as a key has not been pressed or a serial interrupt has occurred. Once a key is pressed, we debounce it by pausing and rechecking to make sure it's still good. If it is, the beep routine is called and the appropriate character is placed into the transmit buffer. Once the key is released, execution returns to the key scan loop.

If a serial interrupt—either transmit or receive—occurs, execution branches to the serial interrupt routine. A check is made on the receiver first to give serial input priority over transmitting. Received characters are processed through the LCD routine as part of the serial interrupt. The LCD routine manages the display positioning and filters out two escape sequences (shown in Figure 6) from the serial data.

In addition, the bell character will call the beeper routine. It will normally be used to indicate that the system is awaiting a decision and a keypress would be appropriate.

Choosing the 87C51 for its internal serial port, timer, and EPROM, and using an LCD display which doesn't require any housekeeping makes for simple code. Fewer devices also means power savings. Although I could steal the smart power regulator I offered in last issue's column, a simple on/off switch is the simplest and least expensive approach. There is no need for the device to be able to turn itself on, at least none that I've thought of up to now. I already have an alarm clock, smoke alarms, and a doorbell. Maybe if I added a speech synthesizer to the IR terminal...

On second thought, not now. I just found a package of airline peanuts in my pocket. I think I'll get me a cold one and relax. This one's for you, American! ✦

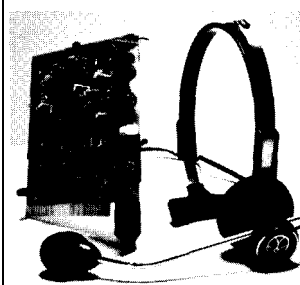
ff Bachiochi (pronounced "BAH-key-AH-key") is an electrical engineer on the Circuit Cellar INK engineering staff. His background includes product design and manufacturing.

422 Very Useful
423 Moderately Useful
424 Not Useful

TALK TO YOUR COMPUTER

WITH VOICE MASTER KEY™ FOR PCs/COMPATIBLES
VOICE RECOGNITION WITH SPEECH RESPONSE

GIVE A NEW DIMENSION TO PERSONAL COMPUTING The amazing Voice Master Key System adds voice recognition to just about any program or application. Voice command up to 256 keyboard macros from within CAD, DTP, word processing, spreadsheet, or game programs. Fully TSR and occupies less than 64K. Instant response time and high recognition accuracy. A real productivity enhancer!



SPEECH RECORDING SOFTWARE
Digitally record your own speech, sound, or music to put into your own software programs. Software provides sampling rate variations, graphics-based editing, and data compression utilities. Create software sound files, voice memos, more. Send voice mail through LANs or modem. A superior speech/sound development tool.

INTERACTIVE SPEECH INPUT/OUTPUT
Tag your own digitized speech files to voice recognition macros. Provides speech response to your spoken commands — all from within your application software! Ideal for business, presentation, education, or entertainment programs you currently use.

Augment the system for wireless uses in robotics, factory process controls, horns automation, new products, etc. Voice Master Key System does it all!

EVERYTHING INCLUDED Voice Master Key System consists of a plug-in card, durable lightweight microphone headset, software, and manual. Card fits any available slot. External ports consist of mic inputs and volume controlled output sockets. High quality throughout, easy and fun to use.

ONLY \$149.95 COMPLETE

ORDER HOTLINE: (503) 342-1271 Monday-Friday 6 AM to 5 PM Pacific Time
VISA/MasterCard phone or FAX orders accepted. No CODs. Personal check: subject to 3 week shipping delay. Specify computer type and disk format (3 1/2" or 5 1/4") when ordering. Add \$5 shipping charge for delivery in USA and Canada. Foreign inquiries contact Covox for C & F quotes.

30 DAY MONEY BACK GUARANTEE IF NOT COMPLETELY SATISFIED.

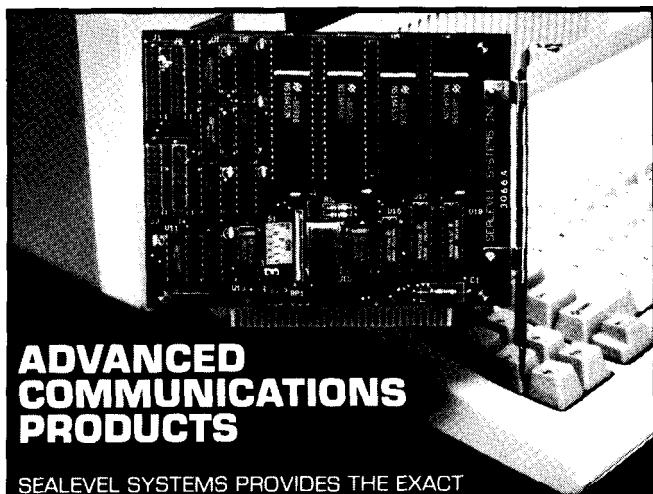
CALL OR WRITE FOR FREE PRODUCT CATALOG.



COVOX INC.
675 CONGER ST.
EUGENE, OR 97402

TEL (503) 342-1271
FAX: (503) 342-1283

Reader Service #132



ADVANCED COMMUNICATIONS PRODUCTS

SEALEVEL SYSTEMS PROVIDES THE EXACT COMMUNICATION CARDS YOU NEED.

PRODUCTS:

- 1, 2 OR 4 PORT RS-232 AND RS-422/485 BOARDS
- CURRENT LOOP SERIAL INTERFACES
- HIGH SPEED SYNC (HDLC, SDLC) AND ASYNC WITH DMA
- RS-530 AND V.35 INTERFACE BOARDS
- DIGITAL AND RELAY I/O BOARDS
- DISKLESS EPROM BOARD WITH PROMKIT SOFTWARE BY ANNABOOKS
- NEW LAP-TOP ADD ONS!
- MADE IN USA
- DELIVERY FROM STOCK
- SATISFACTION GUARANTEED
- EXCELLENT TECHNICAL SUPPORT

SEALEVEL
COMMUNICATIONS & I/O

SEALEVEL SYSTEMS INC.
PO BOX 830
LIBERTY, SC 29657
(803) 843-4343

Reader Service #200

Nuts About RISC

Go on a Low-Fat Acorn Diet

Almost three years ago, in the first issue of *CIRCUIT CELLAR INK*, I wrote short piece entitled "RISC vs. Reality" in which I, with tongue somewhat in cheek, attacked the RISC hype and zealotry then at its most strident.

The main point, admittedly presented with inflammatory rhetoric, was that RISC was more a marketing ploy than a technological "revolution." Essentially, the RISC concept (which I was careful to point out, has some technical merit) had devolved to mean any new chip which isn't an 80x86 or 68xxx.

Since then, having failed to displace low-end PCs (i.e., 80x86- and 68xxx-based systems), many RISC suppliers are pushing (hoping) for an "embedded RISC" market to emerge.

For some reason, whenever I hear the term embedded RISC I can't help but visualize a "UNIX toaster." Built-in multitasking means you can burn more slices at once while the ubiquitous Ethernet port handily connects to your coffee maker. Even the fussiest chef won't tax a 4-gigabyte virtual address space.

Frankly, so far most of the embedded RISC offerings are about as practical as a UNIX toaster. However, there is hope on the horizon as suppliers migrate their existing "CRISCs" (Complex RISCs, or UNIX chips optimized for cranking floating-point loops) towards "Retro-RISCs," which attempt to recapture the simplicity and innocence of the original concept; kind of the micro equivalent of the Mazda Miata.

As suppliers continue to strip unneeded UNIX baggage and cut prices and chip count into the realm of real-world controller feasibility, I imagine I'll have more to say about embedded RISC.

Meanwhile, we can start with one of the oldest RISCs which, thanks to its age, turns out to be a good example of a Retro-RISC. Whether the Acorn CPU (now rechristened the "ARM" for Advanced RISC Machines) itself takes off or not is hard to predict. Nevertheless, I think we can catch a glimpse of tomorrow's popular embedded micros in today's ARM.

HIGH-FAT CHIPS

The problem with most RISCs is that there is nothing reduced about them. In the beginning, the RISC argument was that a simple CPU and a complex compiler could achieve performance similar to the opposite (i.e., complex CPU, dumb compiler) setup. This is meritorious-better to spend more initially for a fancy compiler if the payback is smaller/cheaper CPUs in each and every unit sold.

Somewhere along the line, reflecting various players' marketing agendas, the RISC pitch mutated from one of "good performance at lower price" to "highest performance at any price." Thus, we end up with "reduced" chips like the i860XP with 2.5 million transistors and a \$500+ price tag! Unless you're designing the mother of all toasters, it's not very relevant.

In the quest for performance, traditional RISC features are being jettisoned left and right to the degree that the term **RISC** has become meaningless technically (if in doubt, revert to the marketing definition: RISC = NOT [80x86 OR 68xxx]). Even the feature most identified with RISC—LOAD/STORE architecture (ALU works only on registers, not memory operands)—may be under attack. In a recent review of a new controller chip featuring LOAD/STORE, one pundit noted that "programmers won't ap-

preciate its lack of memory operands for the arithmetic and logical instructions." CISC lives!?"

The bottom line is that complexity is inherently linked to performance, otherwise we'd all be using 1-bit Turing machines.

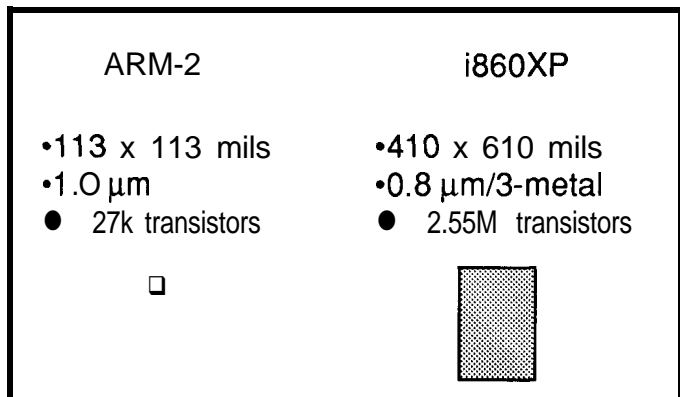


Figure 1—The ARM-2 is only about one-twentieth the die size of the i860XP, even though the latter is built with a denser process.

But isn't the goal of most embedded designs "adequate performance at minimum cost"? If so, the world still needs someone to champion the original RISC concept.

LEAN & MEAN MACHINE

The ARM-2 CPU looks a lot like other 32-bit CPUs on paper: 32-bit registers and ALU, 64-MB physical address space, USER/SUPV protection, and so on. However, there is one big difference: the ARM-2 die size is a measly 113 x 113 mils. Figure 1 puts this in perspective by comparing against the previously mentioned state-of-the-art i860XP. Note how the ARM-2 is only about one-twentieth the die size of the i860XP, even though the latter is built with a denser process (0.8 μm vs. 1.0 μm). Transistor-wise, the ARM-2 is almost exactly 1% the complexity of the i860XP!

Is the i860XP faster than the ARM-2? Of course. Is it 100 times faster? Maybe, especially if your forte is weather simulation or some similar matrix-floating-point-bound application. Is it better for a toaster (or other non-rocket-scientist applications)? No way!

Its tiny die size might lead you to believe the ARM-2 pushes reduction to the frontier of marginal usability as promulgated by true RISC zealots—you know, the folks who would have you believe you don't really need a multiply instruction since it can be reduced to shifts and adds. In fact, the ARM-2 has a multiplier and a number of other "CISCy" features. Since complexity is out of favor, let's call the ARM-2 a "Clever Instruction Set Computer." I'll point out its deviations (most of which I agree with) from "correct" RISC dogma as we move along.

The register set looks about as conventional as can be on the surface: 16 32-bit mostly-general-purpose registers. R13 is used as a software multilevel stack, R14 as a hardware one-level stack (LINK), and R15 is the PC and PSW (using the upper 6 bits of the PC for PSW yields a 26-bit, or 64-MB, address space). The elimination of a conventional dedicated stack pointer and program counter in favor of using regular registers is one example of the minimalist philosophy that characterizes much of the ARM-2 design.

Actually, the register set consists of 27, not 16, registers. As shown in Figure 2, the 27 physical registers are mapped against the 16 logical registers using a register bank approach. However, this isn't the complicated procedure-call/return-oriented register window mechanism promoted by some RISCers, but a simple interrupt-oriented context switch scheme much like the alternate register set on Z80s. Essentially, a portion of the register space is automatically swapped depending on the state—USER, SUPV, IRQ, FIRQ—of the CPU. For example, in response to a fast interrupt request (FIRQ pin), the CPU will automatically switch a portion of the register set (R8–R14). The idea is that the FIRQ handler can (hopefully) find everything it needs in the swapped registers and do its thing without any slow memory context save (push) or restore (pop). This, along with the (mostly) short/fixed latency of instructions, means interrupt response time is very good—typically, less than 1 μs. Contrast this to many UNIX-type RISCs, which seem to have little understanding of real time at all. Just try to figure out the interrupt latency on one of those superscalar/pipelined, compiler-optimized, virtual memory behemoths and you'll see what I mean.

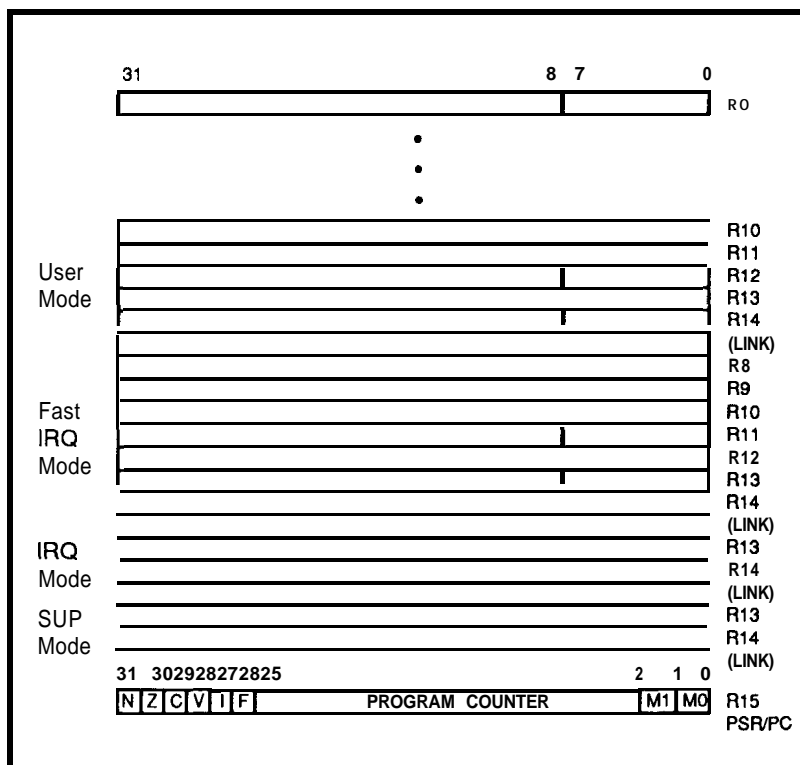


Figure 2—The ARM's 27 physical registers are mapped against 16 logical registers using a register bank approach.

The instruction set is truly **reduced**, consisting of little more than LOAD/STORE to get stuff in/out of registers, the typical ALU ops (Figure 3), and branch. Instructions are fixed in length at 32 bits which won't help code density, but it does keep things simple. The ALU ops typically accept three operands as is usual for a RISC:

```
ADD R0,R1,R2 ; R0=R1+R2
```

Of course, a two-operand instruction can be impersonated by making the destination equal to one of the sources.

Exploiting pipelining, most ALU ops effectively execute in one cycle. Exceptions are Multiply (MUL) and Multiply&Accumulate (MLA), which can take up to 16 clocks. Give the ARM-2 credit for being one of the first to include the Multiply&Accumulate function now in vogue on the newest machines. Loads, stores, and branches are memory bound and thus take a minimum of three clocks and possibly more depending on the speed of the memory subsystem. Load/Store Multiple, which moves multiple registers to/from memory, is much slower depending how many registers are transferred.

Addressing modes (only used with LOAD/STORE; see Figure 4) consist of PC relative (with up to a 12-bit offset), base reg + (12-bit) offset, and base reg + index reg. However, through clever use of pre- and postincrement options, the few basic modes effectively do the work of many more, and getting at memory isn't too hard.

As for data types, the ARM-2 supports 32-bit words and bytes-that's all! No BCD, strings, bit fields, and so forth. Refreshing isn't it?

However, underneath this mild-mannered programmer's model lurk singular features which aren't RISC or CISC or AnyKindOfISC that I know of. They are strange, which is probably why I like them!

First, every instruction features conditional execution depending on the programmer-specified state of the PSW flags (Z, N, etc.). Actually, those of you familiar with microcoding (and other lost arts) will recognize this "skip" function and perhaps even remember what it's useful for.

How often have you faced a programming situation like "If the Z flag is set, store R1, else store R2"? Invariably you end up with something like:

```

BNZ    next
STORE  R1
BRA    continue
next:  STORE  R2
continue:

```

With the ARM-2 skip feature, you can effectively write

```

IF Z   STORE R1
IF NZ  STORE R2

```

Instruction	Function	Operation	Flags Affected
ADC	Add With Carry	$Rd := Rn + \text{Shift}(S2) + C$	N, Z, C, V
ADD	Add	$Rd := Rn + \text{Shift}(S2)$	N, Z, C, V
AND	And	$Rd := Rn \cdot \text{Shift}(S2)$	N, Z, C
BIC	Bit Clear	$Rd := Rn \cdot \sim \text{Shift}(S2)$	N, Z, C
CMN	Compare Negative	$\text{Shift}(S2) + Rn$	N, Z, C, V
CMP	Compare	$Rn - \text{Shift}(S2)$	N, Z, C, V
EOR	Exclusive OR	$Rd := Rn \oplus \text{Shift}(S2)$	N, Z, C
MLA	Multiply With Accumulate	$Rd := Rm \cdot Rs + Rd$	N, Z, C, V
MOV	Move	$Rd := \text{Shift}(S2)$	N, Z, C
MUL	Multiply	$Rd := Rm \cdot Rs$	N, Z, C, V
MVN	Move Negative	$Rd := \sim \text{Shift}(S2)$	N, Z, C
ORR	Inclusive OR	$Rd := Rn + \text{Shift}(S2)$	N, Z, C
RSB	Reverse Subtract	$Rd := \text{Shift}(S2) - Rn$	N, Z, C, V
RSC	Reverse Subtract With Carry	$Rd := \text{Shift}(S2) - Rn - 1 + C$	N, Z, C, V
SBC	Subtract With Carry	$Rd := Rn - \text{Shift}(S2) - 1 + C$	N, Z, C, V
SUB	Subtract	$Rd := Rn - \text{Shift}(S2)$	N, Z, C, V
TEQ	Test For Equality	$Rn \oplus \text{Shift}(S2)$	N, Z, C
TST	Test Masked	$Rn \cdot \text{Shift}(S2)$	N, Z, C

Figure 3—Most of the typical ALU operations are supported by the ARM, but extra functionality is possible

A skip (only one clock) is much faster than a branch and, furthermore, the number of instructions (i.e., code size) is cut in half. A potentially helpful side effect is that the ARM-2 sequence executes in the same time down either path (i.e., whether R1 or R2 is stored).

Another use for the skip feature is replacing sequences of the form:

```

; IF R0=R1 or R2 then branch
CMP R0,R1
BEQ
CMP R0,R2
BEQ

```

with the ARMized

```

CMP R0,R1
IF NZ CMP R0,R2
BEQ

```

Another branch bites the dust!

Indeed, besides conditional execution based on the PSW flags, it turns out that the ARM-2 programmer can also specify whether ALU instructions should set the flags or not. This solves the oft-encountered dilemma of interference during the time between when a condition is evaluated (i.e., flag set) and the result of the evaluation used (e.g., conditional branch). Thus, on the ARM-2 you can do cute things like:

```

; see if R0 and R1 are equal
CMP R0,R1 and set flags
; in any case, set R1 = R1 t R2
ADD R1,R1,R2 but don't set flags
; original CMP flags still valid
BEQ RO_WAS EQ_R1

```

But wait, it gets even better. The processor doesn't have any SHIFT instructions. Well they're not needed

since every ALU instruction is a SHIFT instruction! That's right, backed by a barrel-shifter (i.e., 1- to 32-bit shift in one clock), one source operand (including immediates) can be shifted during any ALU instruction.

So, instead of a SHIFT instruction, you just use a MOVE instruction:

```
MOV RO,RO LSL n (n=1 to 32)
```

Shifting an immediate yields a quick and easy way to load large immediate constants:

```
MOV RO, #3 LSL 31 ; RO=#C000000H
```

Normally, loading large immediate constants on a RISC processor has to be done piecemeal since the number and opcode won't both fit in the fixed 32-bit instruction word.

More interestingly, the shift feature leads to constructs which, though rather bizarre looking, are very fast. For instance, say you want to multiply a register times 45 decimal. Now your average Joe Programmer will probably just use the slow MUL instruction. But, an astute ARM guru will come up with

```
ADD RO,RO,RO LSL 2 ; multiply by 5
ADD RO,RO,RO LSL 3 ; multiply by 9
```

Addressing Mode	Operation	Syntax
PC Relative	EA = PC ± Offset (12 Bits)	LABEL
Base Register Offset With Post-Increment	EA = Rn Rn ± Offset → Rn	[Rn], Off
Base Register Offset With Pre-Increment*	EA = Rn ± Offset (12 Bits) Rn +/- Offset → Rn	[Rn, Off]
Base Register Index With Post-Increment	EA = Rn Rn ± Rm → Rn	[Rn], Rm
Base Register Index With Pre-Increment	EA = Rn ± Rm Rn ± Rm → Rn	[Rn, Rm]

* Effective Address
** Program control of index register update (i.e., Rn may be left unchanged)

Figure 4-Through clever use of pre- and post-increment options, a few basic addressing modes effectively do the work of many more.

It's been said (by me anyway) that RISC means "Religate the Impossible Stuff to the Compiler." In particular, the more esoteric chips call for a whole class of optimizations known as "code scheduling" in which conventionally generated intermediate code is reordered for optimal execution. Many RISCs feature (?) delayed load and/or delayed branch, so the compiler will try to move a useful instruction into the delay slot. The latest chips with multiple execution units call for the sequential intermediate code to be automatically "parallelized," a black art indeed. The problem is that reordering code involves a raft of complicated "dependency analysis" algorithms to ensure the massaged code works as intended.

However, the ARM-2 calls for no such gymnastics since there aren't any delay slots (much less multiple execution units) to worry about. Furthermore, the opportunities afforded by the unique instruction set are easily handled as peephole optimizations-simple text search/replacement of the output code. Thus, the compiler can simply generate a MUL R0, #45 which the optimizer can easily recognize and replace with the faster ADD/SHIFT sequence. Similarly, short forward branches can be optimized into oblivion using the conditional execution feature to skip the formerly branched around code.

YOU GET WHAT YOU PAY FOR

Based on the glowing review so far, you will probably be surprised when I say the ARM-2, as it is today, isn't a good choice for most embedded designs.

The gotcha is that while the basic CPU architecture is neat, the chip implementation has limitations that make system design difficult.

The problems stem from the heritage of ARM-2-the (then Acorn) chip was designed as an alternative to the 80x86 and 68xxx for use in a personal computer supplied to schools by the BBC.

As such, the CPU itself was never designed to work stand-alone, but instead was meant to work in concert with outside memory and I/O control chips. Unfortunately, these bring along a lot of "desktop" baggage—

THE "GREAT DEAL" CATALOG

Protek

- Intel 386-SX microprocessor, at 16 MHz.
- 1 MB RAM.
- 40MB hard drive.
- Five expansion sockets.
- One 3.5" 1.44MB and one 5.25 1.2MB floppy disk drive.
- IBM compatible
- 14" VGA color monitor, res: 640x480, .41mm dot pitch.

386™ -SX16MHz Computer with 40MB Hard Drive and 14" Color VGA Monitor

- 1 parallel and 2 serial ports.
- 101-key keyboard.
- 1 6-bit VGA display adapter res. 800x600.
- 80387 math co-processor slot.
- Includes DR-DOS 5.0.
- Spinnaker Eight-in-One software.
- One Year Mfr. Warranty!
- Factory New!

Mfr. Sugg. Retail: **\$3,469.00**

DAMARK PRICE \$999.99

Item No. B-3537-189036 S/H 549.00

FOR FASTEST SERVICE, CALL TOLL FREE

1-800-729-9000

DAMARK INTERNATIONAL, INC.
7101 Winnetka Ave. N.
P.O. Box 29900
Minneapolis, MN 55429-0900

Please rush me: _____ PROTEK COMPUTER(S)
@ \$999.99 each, plus \$49.00 S/H each.
Item No. B-3537-189036
MN res. add 6.5% sales tax.

DAMARK

Customer Service:
1-800-733-9070

Copyright 1991
DAMARK International, Inc
All rights reserved.

B-3537

Name _____

Address _____

City, State, Zip _____

Check/MO VISA MasterCard Discover

Card No. _____

Exp. Date _____ Ph.# () _____

Signature _____

DELIVERY TO CONTINENTAL UNITED STATES ONLY

video, sound, virtual memory MMU, and so on—that are likely not needed in an embedded design.

Yet, designing an ARM-based system without them isn't easy. The basic problem is the ARM-2 bus interface: there isn't one! Well, of course there are address, data, and control lines (Figure 5), but they need significant conditioning to connect to conventional memory and I/O chips.

For example, the ARM-2 relies on address pipelining and burst transfers to achieve no-wait-state performance with slow DRAMs. If you don't use their outside memory controller (MEMC) chip (with MMU, video support, etc.), a few PALs will be called for unless you can afford a big performance hit. Similarly, without their IOC glue chip, connecting byte-wide static ROM, RAM, and I/O chips isn't easy. Speaking of ROMs, you'll need four of them unless you roll your own "x 8 boot ROM" circuit. Need a wait state? Whip out those old 6800 clock stretch circuits!

I haven't even mentioned the follow-on chip, the ARM-3, since it seems to have just gone farther down the desktop path. While it does have a slightly easier bus interface, the major additions are 4K bytes of cache and a dedicated coprocessor interface. In my opinion, cache is of dubious merit in embedded designs since it blows determinism (how long, exactly, a section of code will take to execute). If the ARM-3 had a way to use the cache as RAM, or even a way to lock key stuff in the cache, I would be more interested. As for the dedicated coprocessor interface, the packaging penalty seems high (I'd rather not deal with 150-160-pin chips) since I probably don't need a coprocessor (which is fortunate, since the catalog doesn't show any for sale). The ARM-3 may be fine for tutoring UK techno-tots, but it's not going in my next toaster design.

BACK TO REALITY

There is hope for the ARM. Recently, control of the architecture has been spun off to a new company, Advanced RISC Machines Ltd., with investment from Acorn (the original designer), VLSI Technology Inc. (chip

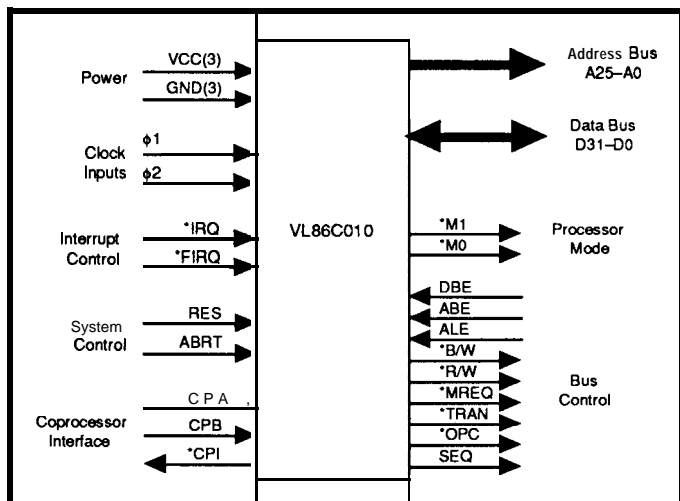
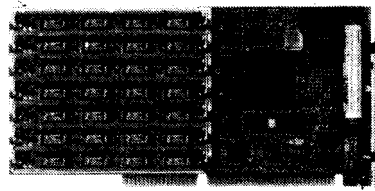


Figure 5—The ARM-2 features the normal address, data, and control lines, but they need significant conditioning to connect to conventional memory and I/O chips.

ROMDISK™



High Performance Multimegabyte Disk Emulators

NEW MODELS / LOWER PRICES

- Floppy Drive and multimegabyte emulators for ISA bus computers
- 180K to 14 MB capacities
- EPROM, Flash or SRAM technologies
- Autobooting, Single or Dual disk emulation under PC or MS DOS
- List prices from \$195

CURTIS, INC.

2837 No. Fairview Ave. • St. Paul, MN 55113

612/631-9512 FAX 612/631-9508

PC DOS is a trademark of IBM; MS DOS is a trademark of Microsoft

Reader Service #134

How many 8051 compilers
promote world peace!



Your firmware development will be more peaceful and easier using the language you already know • BASIC.

• BC151 BASIC cross compiler • Extensive compile-time and run-time error handling options • Integer math with four data types • Co-exist with BASIC-52 or stand-alone • Unlimited phone support, outstanding documentation • Now supports DS500T & C51F Series • With assembler and utilities \$299 • Assembly Language Programmer's Toolkit \$99

Systronix Inc.

754 East Roosevelt Avenue

Salt Lake City, Utah 84105

801-487-7412 FAX: 801-487-3130



Reader Service #204

weighting curves, and investigates some of its effects.

The consequences of this work are of particular interest in the field of noise reduction. Automobile manufacturers, for instance, are interested in reducing noise within the cab of a vehicle. Reducing noise to inaudibility might arguably present a positive danger to the driver by isolating him psychologically from his control of the vehicle. On the other hand, the duration of exposure during long trips argues strongly for reducing cab noise at least to completely innocuous levels. How the use of the linear A-weighted curve causes the expenditure of excessive effort in such an endeavor will become evident later.

SENSITIVITY OF THE "AVERAGE" EAR

A program to calculate the subjective effect of sound must relate subjective loudness levels (phons) to measured intensity levels (dB referred to 10^{-16} watt/cm²) at various frequencies. Nonlinear relationships of this sort maybe handled in two ways: either with a lookup table or an expression that represents the data with acceptable accuracy. Unless lookup tables occupy a large space within the computer memory, multiple interpolations are often necessary in their use. Use of an expression is preferable from the standpoints of speed and economy, and was the approach of choice in this article.

I must admit to some bias in this matter. In 1953 I observed that above 1000 Hz, the Fletcher-Munson curves are almost linear with intensity. This being the case, their inversion to obtain the frequency response of the ear is permissible. The lower peak had a resonance at 3800 Hz with a damping factor of 0.20 [6].

FREQUENCIES BELOW 1000 HZ

Below 1000 Hz, the Fletcher-Munson curves show that the ear's sensitivity is markedly nonlinear with intensity. At 30 Hz, for example, a sine wave tone having an intensity level of 100 dB has a subjective loudness level of 100 phons. Reducing the intensity by less than 40 dB makes the sound virtually inaudible (0 phons). At 100 Hz, the same reduction would not have nearly as much effect; the subjective level would only be reduced to 40 phons.

Extrapolated bass-response Fletcher-Munson curves meet closely at a point corresponding to (6 Hz, 100 dB) with two exceptions: the curves for loudnesses of 110 and 120 phons. These levels, corresponding to the sound generated by hammer blows on a steel plate (or what amounts to the same thing: a rock band) are not of very much interest in sound-reduction work. [Author's Note: Sounds of these intensities are perceived other than by the ears (they are not silenced by closing the ears, and are perceived by some people ordinarily considered deaf). People who are exposed to such sounds often deal with them in unusual ways.] The sensitivity curve for the 10-phon loudness level has a slope of -30 dB/decade, while that for the 70-phon curve is -20 dB/decade.

An analog filter with attenuation of 20 dB/decade is easy to design since it uses a single R-C network. Attenu-

ations less than 20 dB/decade can be realized with somewhat more complex filters [7], but the frequency range over which the characteristic is maintained is restricted, the more so the smaller the attenuation rate. Maintaining an attenuation of, say, 10 dB/decade over a broad range of frequencies usually requires multisection filters.

The A-weighting filter in current use combines two independent 20-dB/decade high-pass sections, one hav-

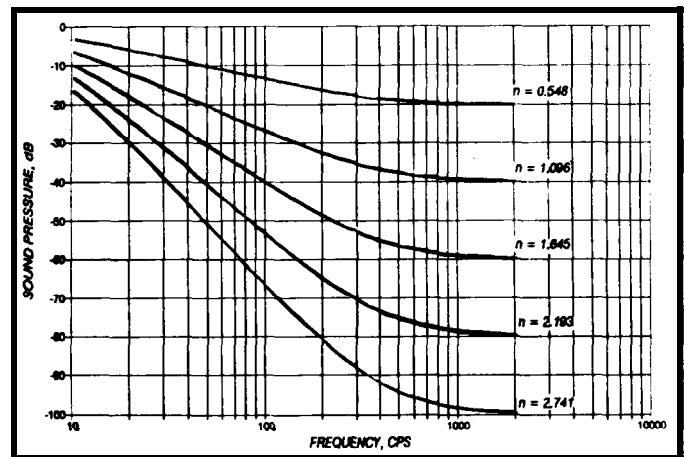


Figure 2—“Partial filter” responses approximate the Fletcher-Munson curves below 1000 Hz.

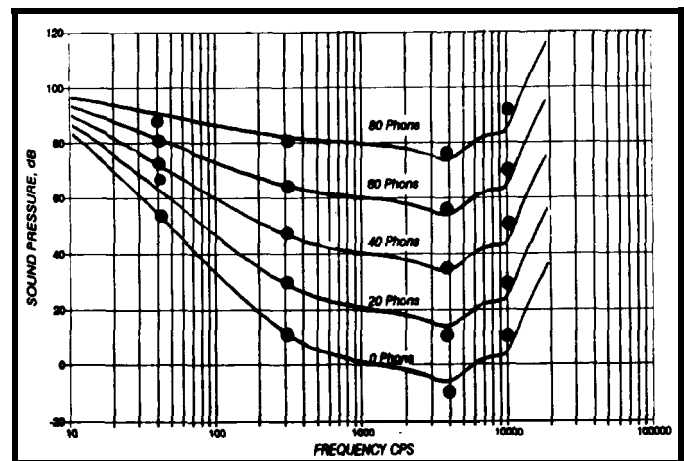


Figure 3—Equations may be used to approximate the Fletcher-Munson curves. Lines are calculated while the dots are from Fletcher-Munson.

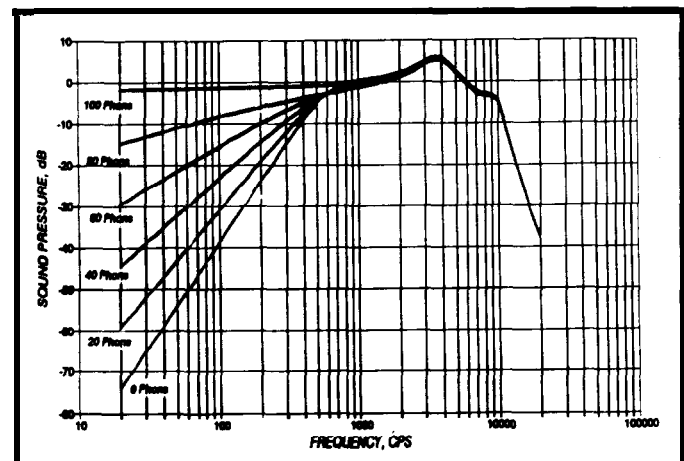


Figure 4—Phon weighting factors. Curves are constant intensity.

foundry), and Apple. The involvement of Apple has inspired speculation; my guess is the ARM core might end up buried in support chips designed for tomorrow's **Macs**.

Meanwhile, ARM Inc. also supplies the usual variety of tools running on the usual platforms: C compiler, simulator, assembler, linker, even (shudder) a UNIX port. While I don't have specific quality info, the fact that the tools have been around a long time is a positive sign.

Anyway, the key point is that the new company is not constrained to serve only one customer (Acorn/BBC) or market (educational PCs). Now, they have the freedom to design chips that truly serve the embedded control market. Manufacturing is expected to open up as well since ARM Inc. intends to license a variety of suppliers to offer standard and custom variants based on the CPU core.

It wouldn't be hard to make a highly integrated version of an ARM microprocessor. This would involve adding the modern bus interface features (chip selects, wait states, DRAM refresh and high-speed access modes, etc.) and typical I/O functions (some timers and a UART or two) found on contemporary competitors. The chip could be further enhanced ("tarted up" as they might say in the jolly old U.K.) with a fancy interrupt controller, low-power operation modes, and similar bells and whistles. A la "My Fair Lady," the once homely ARM can be transformed.

Even better, why not take advantage of the CPU's tiny size to pack on a lot of (EP)ROM and RAM—say 64KB and 4KB respectively. Since the on-chip memory could be

accessed much faster than external memory, performance would be high and scale linearly with clock rate. With all memory on-board, bus interface headaches disappear and low-cost, low pin-count packages are feasible.

Thanks to its excellent efficiency (performance per transistor), the ARM architecture is arguably one of the best candidates to serve as the core of tomorrow's 32-bit "C"ingle Chips. ❖

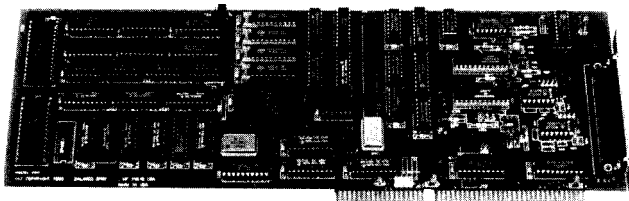
CONTACT

Advanced RISC Machines Ltd.
Park End
Swaffhan **Bulbeck**
Cambridge **CB5 0NA**
England
Phone: 0223 813000
Fax: 0223 812800

Tom Cantrell holds a B.S. in economics and an M.B.A. from UCLA. He owns and operated Microfuture, Inc., and has been in Silicon Valley for ten years working on chip, board, and system design and marketing.

IRS

425 Very Useful
426 Moderately Useful
427 Not Useful

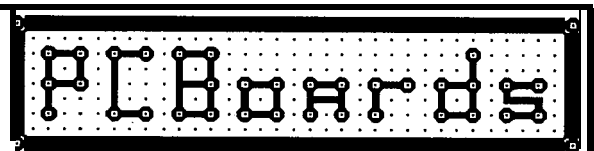


Model 250 for Algorithm Development,
Data Acquisition, Instrumentation. Audio.

- **TMS320C25** DSP at 10 MIPS.
- Up to 192 Kwords RAM.
- Multi-Channel Analog IO - 250K Samples/sec.
- Development Software, including Assembler & Debugger.
- **Applications** Software includes FFT, Signal Display, Data Acquisition & Waveform Editor.
- No Gap Sampling to/from Disk at Very High Rates.
- Supports Multiboard & Standalone (EPROM) Operation.
- From \$1095. Other DSP Products Available.

DALANCO SPRY

89 Westland Avenue
Rochester, N.Y. 14618
(716) 473-3610



P-C-B ARTWORK MADE EASY !
Create and Revise PCB's in a Flash

- * HERC, CGA, EGA, VGA, SUPER-VGA
- * HELP SCREENS
- * EXTREMELY USER FRIENDLY
- * AUTO GROUND PLANES
- * DOT- MATRIX, LASER and PLOTTER ART
- * GERBER and EXCELLON OUTPUT
- * CREATE YOUR OWN FILMS with 1X ART
- * LIBRARIES
- * DOWNLOAD DEMOS from 24 hr. BBS!

REQUIREMENTS: IBM PC or Compatible, 384K RAM
DOS 3.0 or later. IBM compatible printers, HP Laser

PCBoards - layout program 99.00
(PCBoards HP or HI PEN PLOTTER DRIVER 49.00)
PCRoute - auto-router 99.00
SuperCAD - schematic pgm. 99.00
Demo Pkg. - (includes all 3 programs) 10.00

Call or write for more information
PCBoards

2110 14th Ave. South, Birmingham, AL 35205
1-800-473-PCBS / (205)933-1122
BBS / FAX (205) 933-2954

PRACTICAL ALGORITHMS

Charles P. Boegli

Measuring Subjective Sound Levels

Occasionally the merit of an experimental paper stands uncontested for many years. Among such works is that of Fletcher and Munson [1], defining the sensitivity of the human ear to tones of various frequencies. Their curves (Figure 1) were the foundation of all sound evaluation work for more than a half century. They show that the sensitivity of the ear is reasonably linear with intensity above about 1000 Hz. Below that, however, the response is nonlinear with intensity, dropping off more rapidly at lower levels than at midfrequencies.

This effect is familiar to audiophiles. When the level of music reproduction is below that of a live concert, the "bass" control is advanced to restore satisfactory balance. The "treble" control, however, needs little or no adjustment for variations in playback level.

Instruments were soon developed for measuring sound and noise levels. The usual method for determining the subjective noise level at any location was to pass the signal from a calibrated microphone through a weighting filter to convert the flat response of the microphone to the characteristics of the "average" human ear [2]. The subjective sound level was displayed on a meter.

For weighting the contributions of various frequencies, the original recommendation was to use a curve corresponding to the inverse of the sensitivity of the human ear to a level of 40 decibels above a reference of 10^{-16} watt/cm² [3]. This was the "A" weighting filter. The "phon," a unit of loudness level defined as the

"loudness of a sound numerically equal to the intensity level in decibels of a 1000-Hz pure tone which is judged by listeners to be equally loud" [4], was the unit in which loudness levels were expressed.

Obviously no linear filter can reproduce the amplitude sensitivity of the human ear. The proposed B-weighting filter (which used the 70-dB equal loudness contour instead of the 40-dB) recognized this fact. The original suggestion was to use the A-weighting filter if only a single filter were available. In more sophisticated instruments, the A-weighting filter was recommended for measurements up to 55 dB, the B-filter for measurements from 55 to 85 dB, and a flat response for measurements of very loud sounds (85 to 140 dB).

Since the epochal work of Fletcher and Munson, other workers have modified and reinterpreted the curves. The ISO, for instance, wrote a standard [5] that presents curves somewhat different from those of Fletcher and Munson. The difference lies principally in the region below 1000 Hz, where ISO curves are much more linear with level than the

Fletcher-Munson curves.

Examination of the work in sound and noise evaluation that followed on the heels of Fletcher and Munson reveals a continuing tendency toward linear treatment of the ear's response, long after the need for it has passed. This article lays the basis for a computer algorithm that evaluates sound levels with the nonlinear Fletcher-Munson relationships rather than fixed

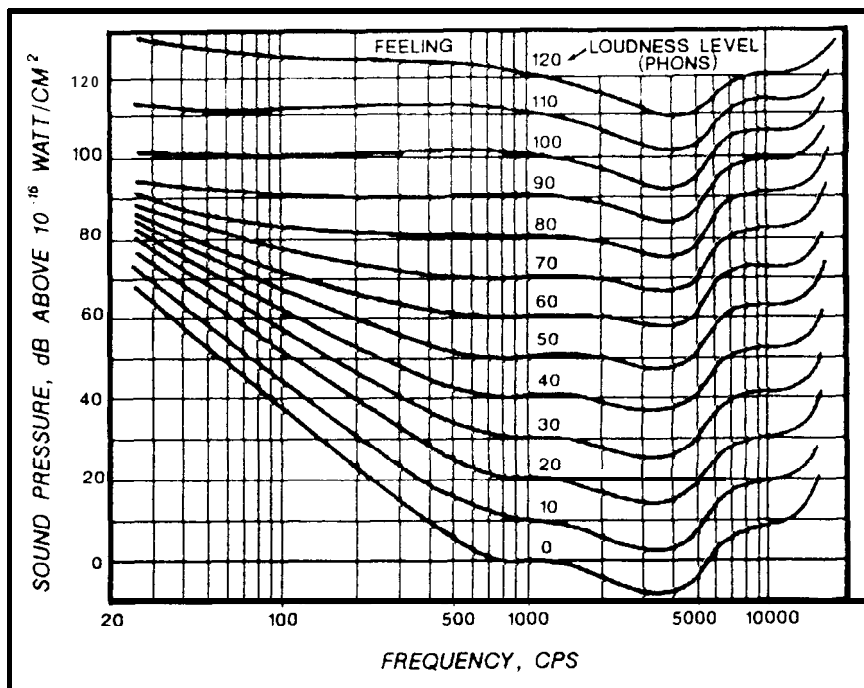


Figure 1 -The original Fletcher-Munson curves were developed in the early 1930s.

curves, and investigates some of its effects. Sequences of this work are of particular interest in the world of noise reduction. Automobile manufacturers are interested in reducing noise within the vehicle. Reducing noise to inaudibility might present a positive danger to the driver by isolating him psychologically from his control of the vehicle. On the other hand, the duration of exposure during long trips is not only for reducing cab noise at least to conspicuous levels. How the use of the linear A-weighting filter causes the expenditure of excessive effort and endeavor will become evident later.

CHARACTERISTICS OF THE "AVERAGE" EAR

Programs are available to calculate the subjective effect of sound pressure levels (phons) to mean loudness levels (dB referred to 10^{-16} watt/cm²) at various frequencies. Nonlinear relationships of this sort are handled in two ways: either with a lookup table or a program that represents the data with acceptable accuracy. The lookup tables occupy a large space within computer memory, multiple interpolations are often required in their use. Use of an expression is preferable for speed and economy, and was the choice in this article.

I must admit to some bias in this matter. In 1953 I published data at above 1000 Hz, the Fletcher-Munson curves are nearly linear with intensity. This being the case, their use to obtain the frequency response of the ear is straightforward. The lower peak had a resonance at 3800 Hz with a Q factor of 0.20 [6].

CHARACTERISTICS BELOW 1000 HZ

Below 1000 Hz, the Fletcher-Munson curves show that sensitivity is markedly nonlinear with intensity. For example, a sine wave tone having an intensity of 100 dB has a subjective loudness level of 100 phons. An intensity of less than 40 dB makes the sound barely audible (0 phons). At 100 Hz, the same reduction in intensity have nearly as much effect; the subjective loudness is only reduced to 40 phons.

Attenuated bass-response Fletcher-Munson curves are available at a point corresponding to (6 Hz, 100 dB) with the same intensity: the curves for loudnesses of 110 and 120 dB levels, corresponding to the sound generated by a hammer blows on a steel plate (or what amounts to the same thing: a rock band) are not of very much interest in noise-reduction work. [Author's Note: Sounds of this nature are perceived other than by the ears (they are perceived by closing the ears, and are perceived by some people as if they were deaf). People who are exposed to such sounds are often perceived in unusual ways.] The sensitivity of the 40-phon loudness level has a slope of -30 dB/decade; that for the 70-phon curve is -20 dB/decade. A weighting filter with attenuation of 20 dB/decade is used since it uses a single R-C network. Attenu-

ations less than 20 dB/decade can be realized with somewhat more complex filters [7], but the frequency range over which the characteristic is maintained is restricted, the more so the smaller the attenuation rate. Maintaining an attenuation of, say, 10 dB/decade over a broad range of frequencies usually requires multisection filters.

The A-weighting filter in current use combines two independent 20-dB/decade high-pass sections, one hav-

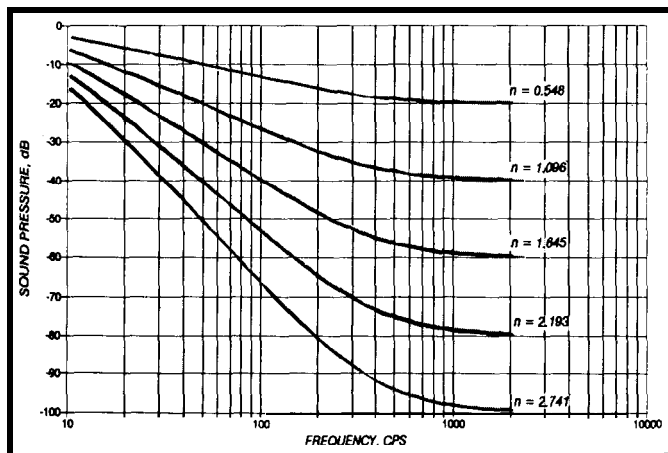


Figure 2—'Partial filter' responses approximate the Fletcher-Munson curves below 1000 Hz.

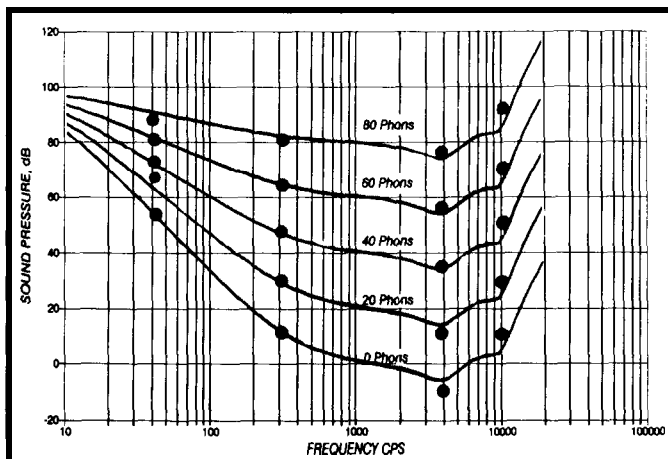


Figure 3—Equations may be used to approximate the Fletcher-Munson curves. Lines are calculated while the dots are from Fletcher-Munson.

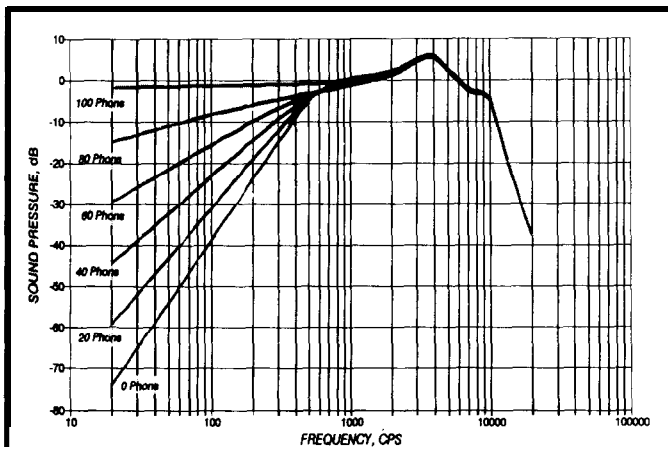


Figure 4—Phon weighting factors. Curves are constant intensity.

ing a corner frequency at about 100 Hz and the other near 500 Hz. Thus, the response shows an attenuation of 6 dB/octave from 500 to 100 Hz, and 12 dB/octave from 100 Hz down. This is the inverse of the 40-phon curve defined by ISO, not Fletcher and Munson. (We do not know whether the filter was designed to match that curve, or the curve to match the filter).

Sensitivity curves may be viewed as the response of the ear to a constant-level input. To simulate that response below 1000 Hz requires filters with broad-band slopes that are not integer multiples of 6 dB/octave. One approach is to introduce a "partial" filter that has no physical existence at this time, but which serves for modeling.

The response of a low-pass filter with a corner frequency of f_0 follows the expression

$$G = \frac{f_0}{\sqrt{f^2 + f_0^2}}$$

or, if the output is expressed in decibels,

$$G \text{ (dB)} = 10 \log(f_0)^2 - 10 \log(f_0^2 + f^2)$$

where f is the frequency. Similarly, a

high-pass filter with a corner frequency of f_1 has a response described by

$$G = \frac{f}{\sqrt{f^2 + f_1^2}}$$

$$G \text{ (dB)} = 10 \log f^2 - 10 \log(f_1^2 + f^2)$$

The frequency response of a cascade of one high- and one

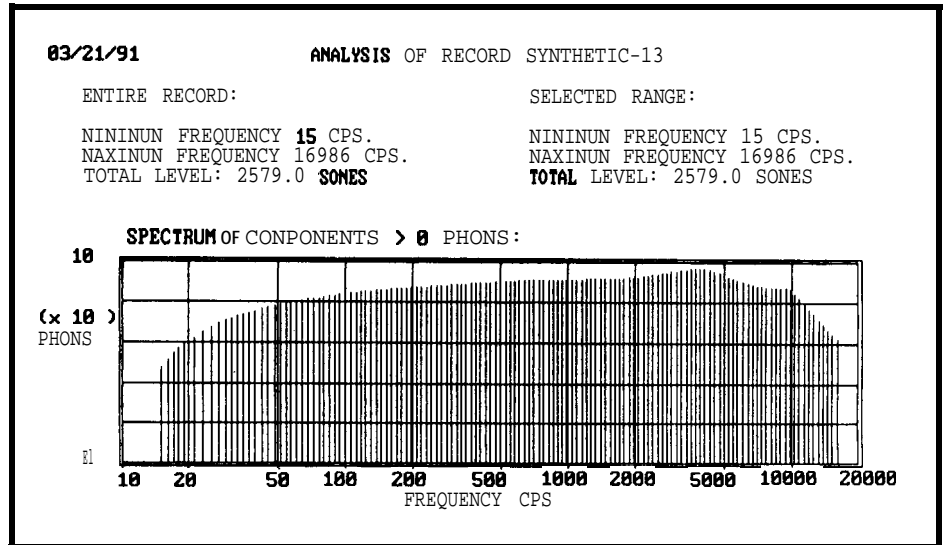


Figure 5—The SOUND.COM program generated a report for a distribution of 90-dB sounds from 15 to 15,000 Hz.

THE \$99.95 EDUCATION

THE PRIMER MICROPROCESSOR TRAINING SYSTEM

- TEACHES:
- INTEL 8085 PROGRAMMING
 - DIGITAL 8 ANALOG INTERFACING
 - PROGRAMMING INTEL PERIPHERALS
 - * MICROCOMPUTER ASSEMBLY & DESIGN
- FEATURES:
- ROM MONITOR OPERATING SYSTEM
 - * 8 DIGIT, 7 SEGMENT, LED DISPLAY
 - 20 KEY KEYPAD
 - * DIGITAL INPUT PORT WITH DIPSWITCH
 - * DIGITAL OUTPUT PORT WITH LEDs
 - ANALOG TO DIGITAL CONVERTER
 - DIGITAL TO ANALOG CONVERTER
 - 14 BIT TIMER/COUNTER WITH SPEAKER
- OPTIONS:
- BASIC OR FORTH LANGUAGES IN ROM
 - * RS232 SERIAL PORT CONNECTS TO PC
 - BATTERY BACKED CLOCK AND RAM

MAC OFFERS A COMPLETE LINE OF MICROPROCESSOR TRAINING SYSTEMS STARTING AT \$99.95 QUANTITY 10 FOR THE PRIMER KIT.

EMAC, inc.

618-529-4525 FAX: 618-457-0110

P.O. BOX 2042 CARBONDALE, IL 62902

HAJAR ASSOCIATES

NATIONAL ADVERTISING SALES REPRESENTATIVES

NORTHEAST

Debra Andersen
49 Walpole Street
Norwood, MA 02062
(617) 769-8950
Fax: (617) 769-8982

MIDWEST

Nanette Traetow
242 East Ogden Avenue,
Suite A
Hinsdale, IL 60521
(708) 789-3080
Fax: (708) 789-3082

MID-ATLANTIC

Barbara Best
569 River Road
Fair Haven, NJ 07704
(908) 741-7744
Fax: (908) 741-6823

WEST COAST

Barbara Jones
& Shelley Rainey
3303 Harbor Blvd.,
Suite G-11
Costa Mesa, CA 92626
(714) 540-3554
Fax: (714) 540-7103

SOUTHEAST

Christa Collins
7640 Farragut Street
Hollywood, FL 33024
(305) 966-3939
Fax: (305) 985-8457

ENTIRE RECORD:

SELECTED RANGE:

MINIMUM FREQUENCY 28 CPS.
 MAXIMUM FREQUENCY 17288 CPS.
 TOTAL LEVEL: 265.8 SONES

MINIMUM FREQUENCY 28 CPS.
 MAXIMUM FREQUENCY 17288 CPS.
 TOTAL LEVEL: 265.8 SONES

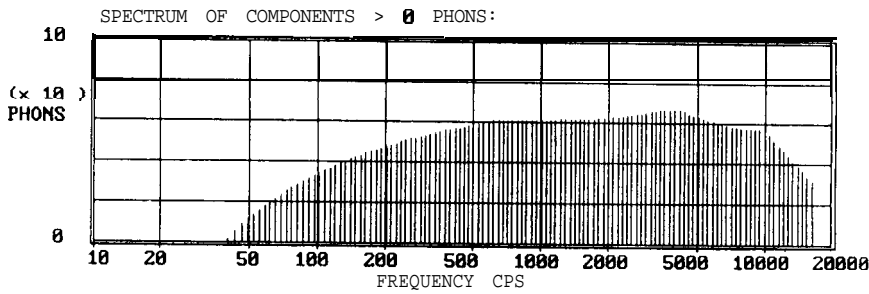


Figure 6—Similar to Figure 6, the software generated a report for a distribution of 60-dB sounds.

The value of n can be found from the fact that at frequencies well above f_1 , the response is $20n \log(f_0/f_1)$ below that at zero frequency. Having set the corner frequencies, we use

$$n = \frac{G_\infty}{\log \frac{f_1}{f_0}} \quad (2)$$

where G_∞ is the gain at a frequency well above f_1 . The response curves of a family of these filters with f_0 at 6 Hz and f_1 at 400 Hz approximate the shapes of the Fletcher-Munson curves below 1000 Hz, as Figure 2 shows.

low-pass filter, both electrically independent, is thus expressed by

$$G \text{ (dB)} = 20 \log\left(\frac{f_0}{f_1}\right) + 10 \log\left(\frac{f_1^2 + f^2}{f_0^2 + f^2}\right)$$

The notion of a partial filter is introduced by a factor n which may have fractional values:

$$G \text{ (dB)} = 20n \log\left(\frac{f_0}{f_1}\right) + 10n \log\left(\frac{f_1^2 + f^2}{f_0^2 + f^2}\right) \quad (1)$$

FREQUENCIES ABOVE 1000 HZ

Above 1000 Hz the response of the ear is much more linear with level than at low frequencies. The similarity of its response to that of a filter with a resonant frequency of 4000 Hz and a damping factor of 0.2 was previously noted [6]. The Fletcher-Munson curves also show a second peak in the vicinity of 14,000 Hz. Attempts to simulate the response with cascaded filters have had limited success, and this by carefully adjusting the damping factors and resonant frequencies. Parallel filters were even less successful. A close approach to the ear's high-frequency response may correspond to the principal resonances of a diaphragm, which will not be attacked here.

The response (by definition, the output for a unit input) of a single simply resonant filter in LaPlacian notation is

$$G(s) = \frac{1}{T^2s^2 + 2ZTs + 1} \quad (3)$$

Because of the separation of the resonant frequencies, the treatment of the two filters as independent, avoiding the complexities of interaction, seems justified. Representing the sensitivity of the ear then involves merely adding the contributions of each resonant filter to the expression for low-frequency response, remembering that the sensitivity (i.e., the input required for a unit output) is the inverse of equation (3).

For the first peak, $T = 3.98 \times 10^{-5}$ and $Z = 0.3$ gave a slightly better fit than the values previously proposed. When these constants are introduced, the phon contribution of the first filter is

$$P = 10 \log\left[\left(3.9062 \times 10^{-15}\right)f^4 - \left(1.025 \times 10^{-7}\right)f^2 + 1\right] \quad (4)$$

A similar expression for the second peak is

$$P = 10 \log\left[\left(1.0000 \times 10^{-16}\right)f^4 - \left(1.910 \times 10^{-8}\right)f^2 + 1\right] \quad (5)$$

The Ciarcia Design Works

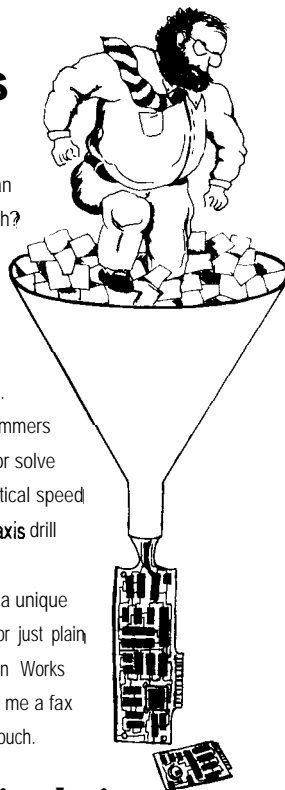
Does your big-company marketing department come up with more ideas than the engineering department can cope with?

Are you a small company that can't afford a full-time engineering staff for once-in-a-while ideas?

Steve Ciarcia and the Ciarcia Design Works staff may have the solution for you.

We have a team of accomplished programmers and engineers ready to design products or solve tricky engineering problems. Need an optical speed sensor, personnel tracking system, or 7-axis drill controller? The team has done it!

Whether you need an on-line solution for a unique problem, a product for a startup venture, or just plain experienced consulting, the Ciarcia Design Works stands ready to work with you. Just send me a fax discussing your problem and we'll be in touch.



Remember...8 **Ciarcia design works!**
Call (203)875-2199 • Fax (203)872-2204

TOTAL SENSITIVITY

The Fletcher-Munson curves are presumably represented by 100 dB plus the sum of expressions (1), (4), and (5). The quality of the fit was checked by plotting these sums at various phon levels with the help of a spreadsheet program. Figure 3 shows the resulting curves, along with points selected from the Fletcher-Munson curves themselves.

The fit is reasonably good in the region below 1000 Hz; an anomaly in the 80-phon curve is also visible in Figure 1. Above this frequency the fit shows the effects of assuming linearity and also the previously mentioned fact that the response at the 14,000-Hz peak is poorly represented by a second resonant filter. I would assume, however, that the determinations made by Fletcher and Munson also showed large variability in the very high- and low-frequency regions because of differences in the limits of audibility of various persons.

Presumably a given subject is incapable of gauging the loudness of tones he cannot hear; thus, all the high-fre-

03/21/91

ANALYSIS OF RECORD SYNTHETIC-11.

ENTIRE RECORD:

MINIMUM FREQUENCY 15 CPS.
MAXIMUM FREQUENCY 145 CPS.
TOTAL LEVEL: 92.0 SONES

SELECTED RANGE:

MINIMUM FREQUENCY 15 CPS.
MAXIMUM FREQUENCY 145 CPS.
TOTAL LEVEL: 92.0 SONES

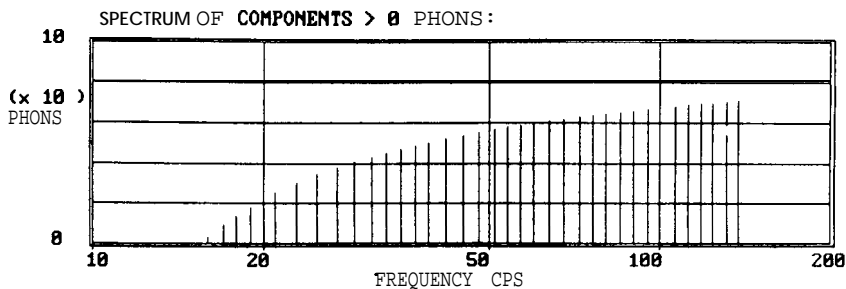


Figure 7-A SOUND-generated report of a recording containing evenly distributed 80-db components from 16 to 100 Hz.

quency curves should pass through a single point at the upper limit of audibility. Within practical limits this means that curves in the high-frequency region should show some tendency toward convergence, whereas the Fletcher-Munson curves are almost parallel. This is not to attack the determination they made, but rather to assess the practical limits to the accuracy of their difficult work. Since most noise-reduction work is concerned with lower frequencies, the inaccuracy of fit should not have a severe effect.

PROFESSIONAL CIRCUIT DESIGN

QICAD

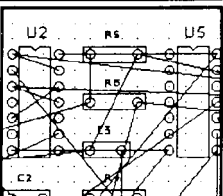
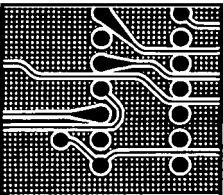
Save time and money!

QICAD is a full-featured printed circuit layout package that gives you everything you need to design circuit boards quickly.

- ON-LINE HELP
- AUTOROUTER
- POWERFUL EDITING
- HPGL/DMPL PLOTS
- GERBER
- POSTSCRIPT
- EXCELLON (DRILL)
- EGA / VGA compatible

\$195.00 complete price

601 South Maine Street
Suite D
Fallon, NV 89406
(702) 423-1653 (702) 423-1654 FAX



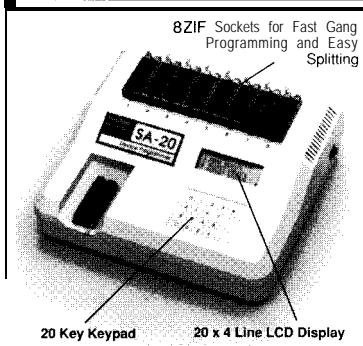
gav

Reader Service # 147

EPROM PROGRAMMERS

Stand-Alone Gang Programmer

\$750⁰⁰



- Completely stand-alone or PC-driven
- Programs E(E)PROMs
- 1 Megabit of DRAM
- User Upgradable to 32 Megabit
- 31.6" ZIF Sockets RS-232, Parallel I/O and Out
- 32K Internal Flash EEPROM for easy firmware upgrades
- Quick Pulse Algorithm (27256 in 5 sec. 1 Megabit in 17 sec.1
- 2 year warranty
- Made in the U.S.A
- Technical support by phone
- Complete manual and schematic
- Siooie Socket Programmer also available. \$550.00
- Split and Shuffle 16& 32 bit
- 100 User Definable Macros. 10 User Definable Configurations
- Intelligent Identifier
- Binary, Intel Hex, and Motorola S
- 2716 to 4 Megabit

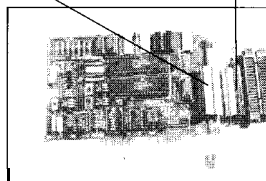
Internal Programmer for PC

\$139⁹⁵

New Intelligent Averaging Algorithm Programs 64A in 10 sec., 256 in 1 min 1 Meg (27010 011) in 2 min 45 sec 2 Meg (27C2001) in 5 min Internal card with external 40 pin ZIF

- Reads, Verifies and programs 2716 32, 32A, 64 64A, 128, 128A 256, 512, 513 010, 011, 301 27C2001, MCM 68764 2532, 4 Megabits
- Automatically sets programming voltage
- Load and save buffer to disk
- Binary, Intel Hex, and Motorola S formats
- Nonpersonality modules rewired
- 1 Year warranty
- 10 days money back guarantee
- Adapters available for 8748, 49 51, 751, 52, 55, TMS 7742, 27210, 57C1024, and memory cards
- Made in U.S.A

2 ft. Cable 40 pin ZIF



EMPDEMO EXE available BBS (916) 972-8042

NEEDHAM'S ELECTRONICS

4539 Orange Grove Ave • Sacramento, CA 95841
(Monday Friday 8 am-5 pm PST)

Call for more information
(916) 924-8037
FAX (916) 972-9960

C. O. D. MasterCard VISA

Reader Service # 182

October/November 1991

99

This **having** been said, it should nevertheless be noted that extremely high-frequency tones, entirely unnoticed by some people, often cause considerable distress to others, and that workers in noise reduction should not overlook their importance.

In Figure 4 the curves derived from the expressions given here have been inverted and adjusted so that they show the weighting factors (in **dB**) at several intensity levels. The weighting is the same for all frequencies above about 1000 Hz; below that, the weighting varies depending on the intensity.

SOUND EVALUATION

In practice, a calibrated microphone may be placed in the location at which the sound is to be measured, and a recording made for an interval of time. Either by direct

digitization of the signal, or by subsequent digitization of an analog recording, this record is converted into one representing the intensity of the sound at the sampling intervals. The sampling speed (samples/second) must be well above the highest frequency to be measured (cycles/second); if the well-known Shannon criterion is used the factor is at least 2.

The record is then transformed (by well-known methods) into one relating intensity to frequency; that is, the frequency spectrum of the sound. Expressions derived in this paper can then be used to convert the sound pressure of each component into phons.

Phons are superficially similar to decibels in appearing to be logarithmic measures and are, in fact, identical in magnitude at 1000 Hz. But the analogy cannot be pushed too far. Sounds below about 0 phons at 1000 Hz are inaudible and have no subjective effect, while sound pressures below 0 **dB** (like voltages) definite physical quantities.

The fact that phon addition does not accurately represent the subjective level of a composite sound is recognized in ISO Recommendation R131, which proposes the use of "sones" as a measurement of subjective sound. The conversion equation [8] is as follows:

$$S = 2^{(P-40)/10}$$

in which *S* represents sones and *P*, phons. Presumably the subjective effect of a composite sound can be found by adding the sone contributions of the individual components. The extensive work that has been done in evaluating composite sound levels, however, appears to indicate that sone summation is not the whole answer.

The program **SOUND.COM** (which operates under **MS-DOS**) computes the phon value of every frequency in an arbitrary frequency spectrum, using the nonlinear equations presented in the last section. It reads a simple ASCII record of up to 20,000 **pairs** of frequency-intensity data points. The frequencies are expressed in cycles per second and the intensities in decibels above 10^{-16} **watt/cm²**; the two numbers are separated by a comma and each

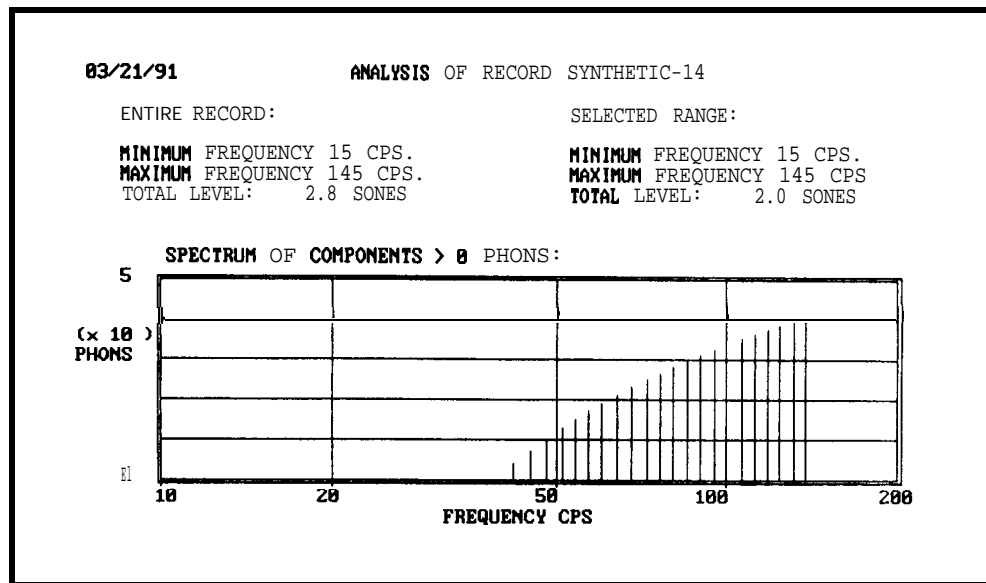


Figure 8—Similar to Figure 7. but with 60-db components.

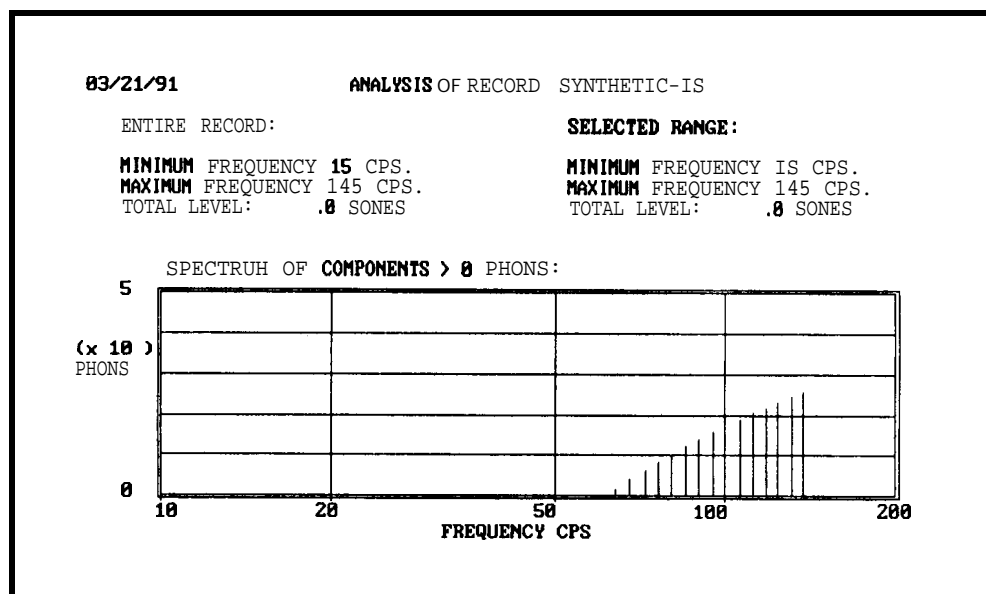


Figure 9—Similar to Figure 7. but with 50-db components.

pair of values is on a separate line in the record. The present version of the `SOUND.COM` program devotes the first four lines to header information. Synthetic records can be generated with a competent word processor or a short program; that is the way they were made for testing this program. [Editor's Note: *Software for this article is available from the Circuit Cellar BBS and on Software On Disk #23. For downloading and ordering information, see page 105.1*

The subjective contribution of each frequency present in a measured spectrum is calculated using the expressions derived in this article, converted into sones, and summed to obtain a quantity presumably proportional to the total subjective sound level. The output of the program is a printed graph showing the phon level of every frequency and intensity in the input record, together with the accumulated sones. `SOUND` also allows the user to examine a smaller range of frequencies more closely, and calculates the sone contribution of that part of the entire range. Figure 5 is the report generated by this program for a distribution of 90-dB sounds from 15 to 15,000 Hz, while Figure 6 is the same for 60-dB sounds. The nonlinear response of the filter is quite evident.

Applying this program to "synthetic" records shows immediately that a linear weighting factor not only wastes effort in reducing low-frequency sounds to which the ear is not sensitive, but fails to assess fully the importance of high-intensity low-frequency sounds. Figure 7 is the analysis printed by `SOUND` of a record containing evenly distributed 80db components from 16 to 100 Hz. Figures 8 and 9 are similar records for identical distributions of 60- and 50-dB components. The poorer low-frequency sensitivity of the ear at lower levels evidently reduces the subjective sound levels more than correspondingly; at the lower

Author's Notes

The fit of the Fletcher-Munson curves presented in this article permits development of a compact algorithm for converting sound pressures into phons, and does away with the need for fixed weighting curves and decisions as to which one to use. A computer program was written to demonstrate its use. Because of the numerous ways to sum contributions of sound components, the program is not proposed as a final solution to the problem of sound evaluation.

What has been shown, however, is that equipment ordinarily available in a competent testing laboratory, combined with a computer program, may do a better job of assessing sound than the many specialized instruments now on the market.

The notion of a "partial" filter will hopefully allow approaching other physiological phenomena than sound. Responses of the human body to stimuli are often nonlinear, which makes devising circuit analogs for them difficult.

A STEP BEYOND.



PROMICE takes ROM emulation a step beyond. It's an affordable, multi-operational development tool with:

- on board intelligence
- modular design
- source level debugging
- future expandability

PROMICE. The Firmware Development System of Tomorrow...

Reader Service #149

BCC52 BASIC-52 COMPUTER/CONTROLLER

The BCC52 Computer/Controller is Micromint's hottest selling stand-alone single-board microcomputer. Its cost-effective architecture needs only a power supply and terminal to become a complete development or end-use system, programmable in BASIC or machine language. The BCC52 uses Micromint's 80C52-BASIC CMOS microprocessor which contains a ROM-resident 8K-byte floating-point BASIC-52 interpreter.

The BCC52 contains sockets for up to 48K bytes of RAM/EPROM, an "intelligent" 2764/128 EPROM programmer, three parallel ports, a serial terminal port with auto baud rate selection, a serial printer port, and is bus-compatible with the full line of BCC-bus expansion boards. BASIC-52's full floating-point BASIC is fast and efficient enough for the most complicated tasks, while its cost-effective design allows it to be considered for many new areas of implementation. It can be used both for development and end-use applications.

PROCESSOR

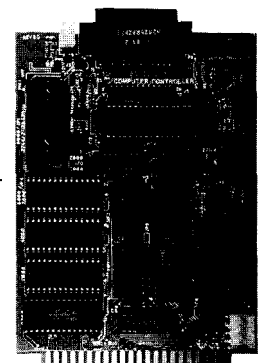
- 80C52-BASIC, 8-bit CMOS microcomputer
- jumper-selectable conversion to 80C31/80C32 functionality
- 8K bytes ROM (full BASIC interpreter)
- 256 bytes RAM
- three 16-bit counter/timers
- 32 I/O lines
- 11 MHz system clock
- 6 interrupts

MEMORY

- expandable to 62K bytes
- five on-board sockets
- up to four 6264 (8Kx8) static RAM
- either 2764 BK 2764 or 128K 27128 EPROM

Input/Output

- console I/O RS-232 serial port
- line printer RS-232 serial port
- three 8-bit programmable TTL-compatible parallel I/O ports using a 8255 PPI
- alternate console RS-422/RS-485



To Order Call
1-800-635-3355
Tel: (203) 871-6170
FAX: (203) 872-2204
TELEX: 643331

		Single Qty.	100 Qty.
BCC52	BASIC-52 Controller Board with 8K RAM	\$189.00	5149.00
BCC52C	Lower-power all-CMOS version of the BCC52	5294.00	5220.00
BCC52I	Full industrial temperature range	5199.00	\$159.00
BCC52CX	CMOS, Expanded BCC52 w/32K RAM	\$259.00	\$189.00

MICROMINT, INC. 4 Park Street, Vernon, CT 06066

Reader Service #11b

See us at The Embedded Systems Conference Booth #819

October/November 1997

101

levels, some low-frequency components become inaudible. Low-frequency sounds being more difficult to attenuate than high, the linear weighting curve can be responsible for **spending** a lot of time reducing the intensities of sounds that are already inaudible! ❖

Charles Boegli is president of **Randen Corporation** in Blanchester, Ohio. **Randen** is a small company offering services in technical computer programming and analog circuit design.

IRS

428 Very Useful
429 Moderately Useful
430 Not Useful

REFERENCES AND NOTES

- (1) Fletcher, H. and Munson, W. A.: 'Loudness, its Definition, Measurement, and Calculation.' J. Acoustical Society of American 5.2 (October 1933) p. 82
- (2) Powertrain Mounting Design Guide. GM Publication, October 1986
- (3) Langford-Smith, F. 'Radiotron Designer's Handbook,' Fourth Edition. Harrison, N. J.: RCA Victor Division, Radio Corporation of America. April 1953 p. 826ff.
- (4) King, A. J. et al: "An Objective Noise-meter Reading in Phons for Sustained Noises," with Special Reference to Engineering Plant. Jour. IEEE88, Part II (1941). p. 163
- (5) ISO Recommendation 226. Within standards organizations like ISO, the development of test methods is usually handled by appointed committees that include representatives of equipment manu-

facturers. I believe the primary interest of those people is in leading the market in developing new testing equipment. Whether the linearization of the Fletcher-Munson curves may have been an accession to the makers of sound-level meters is a matter for conjecture. Certainly, nonlinear responses were difficult to build into the analog circuits in existence when that standard was written.

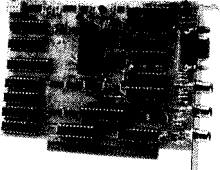
- (6) Boegli, Charles: 'Transient and Frequency Response in Audio Equipment.' Audio Engineering, 38, 2 (February 1954) p. 19ff.
- (7) Boegli, Charles: 'Equalizer Design Chart.' in Markus, J. and Zeluff, V. Electronics for Communication Engineers. New York: McGraw-Hill Book Company, 1952, p. 249.
- (8) ISO Recommendation R 131.

\$495

Video Frame Grabber with Display

- Real Time Capture
- **RS-170A** Video Input
- **RS-170A** Display Output
- Dual Resolution
- 256 Grey Levels
- PC/XT/AT Compatible
- Programmable Input LUT
- Interrupt Capability
- Fast Dual Port Video RAM
- External Trigger Input
- Easy Software Interface
- Complete Documentation

CORTEX-I



Introducing truly affordable, precision image capture Targeted for OEM applications where on board processing and/or color are not required. Superior spatial accuracy, small board size, and price open up many applications. It features dual resolution of either 512 x 484, or four images of 256 x 242, both with 256 grey levels.

Software support includes a menu driven control program, 'C' library with source, TIFF files with LZW compression, and a novel RAM disk emulator which provides DOS access to the board's images. as files!

Simple yet flexible software interface saves development time. For example, all rows and columns are accessible within a single RAM segment, allowing fast searches.

Custom board designs and software services are available. CORTEX-I OEM pricing : Quantities of 100 plus, \$325. Call for brochure and specifications.

IMAGENATION CORPORATION
Specializing in Computer Vision

PO Box 84568, Vancouver, WA 98684.0568
Telephone/FAX (206) 944-9131

PC/XT/AT registered trademark of International Business Machines

NEW!

iceMASTER™

COP8 8051 68HC11



YOUR WINDOW TO EMULATION PRODUCTIVITY

- Easy to learn & use
- Windowed interface -- user configurable
- **FAST! Download -- < 3 sec. typ. at 115KB**
- **Source Level debug**

- A 4K frame trace buffer with advanced searching capabilities.
- Hyperlinked On-line help guides you through the emulation process.
- **iceMASTER** connects easily to Your PC, requires no disassembly, or expansion slots. Works on any PC (DOS or OS/2), MicroChannel or EISA. Even laptops!
- Supports more than 50 different 8051 family derivatives. M68HC11 support will be available early in 1991.
- Try **iceMASTER** risk free! Satisfaction Guaranteed or return for a full refund!*
- **RENTALS AVAILABLE!** Ideal for consultants and researchers!
- Call today for free demo disk and ask about a free 8051 Macro Assembler! **(800) 638-2423**



MetaLink Corporation PO Box 1329 Chandler, Az 85244-1329
Phone (602) 9260797 FAX (602) 926-1198 TELEX 4998050MTLNK
* With 10-day trial period



See us at The Embedded Systems Conference Booth #207

Reader Service 1171

CONNEC- TIME

Conducted by
Ken Davidson

Excerpts from the Circuit Cellar BBS

The Circuit Cellar BBS
300/1200/2400 bps
24 hours/7 days a week
(203) 871-1988
Four Incoming Lines
Vernon, Connecticut

We're real short on space this time, so we'll keep it to just one real thread and a little snippet of one at the end. Event timing always seems to be on people's minds and usually has to be done with as little power as possible. One user who asked a basic question received replies detailing a number of different solutions.

Msg#:24766

From: RICHARD KIMBALL To: ALL USERS

I'm looking for a VERY low-current 5-volt programmable digital timer which can be set for 1-65536 seconds or minutes and which is self-resetting. The application is a remote (outdoors) unattended intervalometer for time-lapse photography. The alarm output must be capable of driving a low-current relay or equivalent device for turning on a shutter-release solenoid.

I've considered numerous microprocessor and microcontroller scenarios and have developed some nice enhancements, but the power requirements are too quick to drain my batteries.

Also, I wonder what effect the seasonal temperature extremes might have on the device. Around here we're talking typically 20 degrees in winter to 8.5 degrees in summer.

Any ideas would be most welcome. If the device exists, I'd love to know where to get it.

Msg#:24815

From: ED NISLEY To: RICHARD KIMBALL

Why digital?

If you want real low current, how about a CMOS 555 timer? It's got good output drive, low standby current, and reasonable temperature stability.

The only catch is that big caps tend to have nasty temperature stability. You could run it at a higher frequency, use a smaller (better) cap, and divide the output by a few orders of magnitude in a fixed digital chain. There goes your current again, but you can use CMOS counters and get most of the benefit. Put a FET driver on the output and you're off and running.

I guess you could make the oscillator fixed and use a programmable digital divider chain, but do you really the precision?

Msg#:24830

From: RICHARD KIMBALL To: ED NISLEY

Thanks for the response. I'm kind of fixated on digital. I've messed with 555s before, even just to generate pulses for a counter, but for this application I want something that either 1) programs similar to a digital watch, or 2) programs via DIP switch or even EPROM (since I have a programmer).

Msg#:24858

From: ERIC BOHLMAN To: RICHARD KIMBALL

If having to set the time in binary isn't a problem, you could use something like the Intersil 7240, which is an **astable** combined with an 8-stage counter (you'd have to chain two of them to get the resolution you want). The counter stages have open-drain outputs, so you can wire-AND them via DIP switches; just tie the common of the switches to the reset line on the chip.

As far as low temperatures are concerned, I think your main problem would be keeping the batteries alive.

Msg#:24876

From: RICHARD KIMBALL To: ERIC BOHLMAN

Thanks, Eric. Your idea is worth looking into. Where can I get the Intersil part and data sheet? I'm pretty sure DIP switches would be OK. Eventually I'd like to refine it with some kind of multi-mode LCD display which could show interval, elapsed time units in current interval, and number of intervals expired. But, hey, I'm just shooting a shutter, right?

Msg#:24898

From: ERIC BOHLMAN To: RICHARD KIMBALL

Most of the small-quantity mail-order distributors carry the 7240. Some of them may be able to supply data sheets; otherwise find your local Intersil distributor (last time I checked, Intersil was part of GE, but that may have changed) and ask for their databook.

Msg#:24895

From: FRANK KUECHMANN To: RICHARD KIMBALL

If you don't mind piecing a few chips together, you could use an **OKI MSM5832RS** clock chip in "interrupt" mode (AO-A3 = 1). D1 would output a 1-Hz signal, D2 a 1/60-Hz. Chain two dual 4-bit binary counters for the 0-65535 unit count. Set the interval desired by driving magnitude comparators with the counters; hex-readout thumbwheel switches set the desired interval. When the mag comps output the "=" level, it resets the counters and triggers something to actuate your shutter. Use 4000-series CMOS at 5 volts and you've got very low drain (essentially leakage current except for the 5832).

Msg#:24928

From: RICHARD KIMBALL To: FRANK KUECHMANN

Where can I learn more about the **OKI MSM5832RS**? Didn't Radio Shack have that in their inventory at one time? Seems like I've seen it somewhere. Maybe Jameco?

Msg#:24941

From: FRANK KUECHMANN To: RICHARD KIMBALL

Places like Digi-Key have had it in the past, but prices are lower at outfits like B.G. Micro in Dallas, Texas. Jameco currently has the Saronix 58321 variant of the 58321 (OKI), but the '321 has muxed address/data lines that complicate things. I have the full OKI MSM5832RS data, plus a lot of experience interfacing the

chip to various devices. Give me an address and I'll send you the data on the chip.

Msg#:24953

From: RICHARD KIMBALL To: FRANK KUECHMANN

Thanks once again, Frank. You know, I've been mulling this thing for well over a year, all the time having Circuit Cellar BBS phone number posted on my PC. I've talked to the few engineer friends I have and ended up basically feeling out of luck. This is GREAT.

Msg#:24977

From: FRANK KUECHMANN To: RICHARD KIMBALL

Some further info: Exar makes at least one "programmable" timer/counter you might look at. Howard Sams publishes a book called "The ICTimer Cookbook" (Jung is the author, I think) with lots of useful info.

Msg#:24915

From: STEVE CIARCIA To: RICHARD KIMBALL

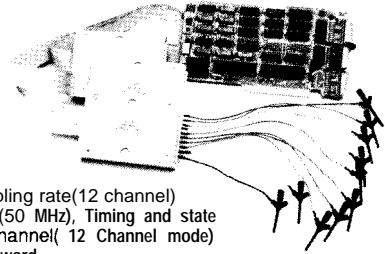
Why not just use a whole RTC31 or RTC52 as a timer. Considering it has an 11-MHz crystal, it can be very accurate. Also, the RTC52 BASIC includes a real-time clock command that can be triggered on interrupt, and could easily give time lapse, interval, history, and so on in just a few program lines.

POWERFUL·AFFORDABLE INSTRUMENTS

100 MHz Logic Analyzer

\$799
LA12100

Price is complete
Pods and Software
included

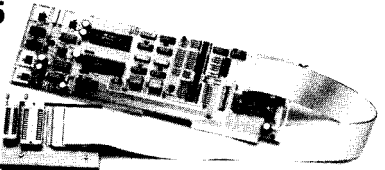


- 100MHz sampling rate (12 channel)
- 24 Channels (50 MHz), Timing and state
- 2K samples/channel (12 Channel mode)
- 24 Bit trigger word
- TTL threshold level
- Internal and External Clocks
- Menu driven software
- FREE software updates on BBS

• More sophisticated units also available

almost **UNIVERSAL PROGRAMMER**

PAL \$475
GAL
EPROM
EEPROM
PROM
87xxx...



- 20 and 24 pin PALs, EPLDs
- 16V8, 20V8, 22V10 GALs
- 2716-27020 EPROMs
- 87xxx MICROS
- EEPROMs (incl. 8 pin serial)
- Byte Split/Merge (16 & 32 bit)
- JEDEC, INTEL HEX, Motorola's files
- Dallas NVS RAM programming
- PC/XT/AT COMPATIBLE
- FREE software updates on BBS

Call- (201) 808-8990
Link Computer Graphics, Inc.
369 Passaic Rd., Suite 100, Fairfield, NJ 07004 FAX (201) 808-8786

IND-286 SBC

AT Compatible DISKLESS SBC Includes DOS in ROM

Complete 16MHz 80C286 Single Board Computer for embedded PC applications features a 4M-byte PROMDISK disk emulator with battery back-up and an MS-DOS 3.3 compatible disk operating system in ROM.

Features Include:

- 4M-byte DRAM
- XI Size Board
- Keyboard Port
- 80287 Socket
- 2 COM, 1 LPT
- WatchDog Timer
- IDE Disk Port
- Floppy Port
- 4M PROMDISK
- 100% PC/AT Compatible
- Optional Video Daughter Bd.

Other Products:

- IND-88 PC/XT Single Board Computers
- PROMDISK III & IV Disk Emulators
- EPROM/RAM Memory Board
- FLASHDISK Driver for Micro Soft FFS
- FlexScan I & II Bar Code Decoders
- Custom PC Compatible Hardware & Software

mcsi micro computer specialists, inc.
2598-g fortune wdy vista, ca 92083
phone: 6 19/598-2 177 fax: 6 19/598-2450

Msg#:24927

From: RICHARD KIMBALL To: STEVE CIARCIA

How about current consumption, Steve? I'm trying to put something together to function in the bush over a several-month period. I'd considered the 8031/52 route, but the specter of battery depletion prevented any serious evaluation.

Msg#:24950

From: STEVE CIARCIA To: RICHARD KIMBALL

Remember, the 80C52-BASIC chip is CMOS. If you remove the 75176 (RS-485 driver) chip, the RTC typically pulls about 15-20 mA. If you reduce the need for RS-232 communication and can remove the MAX232, it drops to about 8-10 mA. Then, if you reduce the crystal frequency, it starts dropping dramatically. I actually got one board down to 2 mA while still running a BASIC program.

Msg#:24954

From: RICHARD KIMBALL To: STEVE CIARCIA

Hey, that ain't so bad! I'm counting my pennies and thinking about how I can short change my six-year-old this Christmas. Thanks again.

One of the biggest advantages of RS422 is its long-distance capability using simple twisted pair wiring, as we see here.

Msg#:25952

From: PHIL ROBERTS To: ALL USERS

Can somebody advise on the spec for cable lengths using RS-485 or RS-422. I have an application which involves mounting a piece of RF hardware as close to its antenna as possible. The unit is presently using a BCC52 to format its nonstandard output to RS 232and feed a computer. I'm looking at needing to drive through

about 150 feet of cable. Any feedback on this would be appreciated.

Msg#:25975

From: KEN DAVIDSON To: PHIL ROBERTS

The RS-422 spec states that you should be able to go up to 4000 feet using 24-gauge wire at data rates under 100 kbps. Your 150 feet shouldn't present any problems (that's the reason why we've been supporting RS-422/485 on all our new processor boards).

The Circuit Cellar BBS runs on a 10-MHz Micromint OEM-286 IBM PC/AT-compatible computer using the multiline version of The Bread Board System (TBBS 2.1M) and currently has four modems connected. We invite you to call and exchange ideas with other Circuit Cellar readers. It is available 24 hours a day and can be reached at (203) 871-1 988. Set your modem for 8 data bits, 1 stop bit, and either 300, 1200, or 2400 bps.

IRS

431 Very Useful
432 Moderately Useful
433 Not Useful

SOFTWARE and BBS AVAILABLE on DISK

Software on Disk

Software for the articles in this issue of Circuit Cellar INK may be downloaded free of charge from the Circuit Cellar BBS For those unable to download files, they are also available on one 360K, 5.25" IBM PC-format disk for only \$12.

Circuit Cellar BBS on Disk

Every month, hundreds of information-filled messages are posted on the Circuit Cellar BBS by people from all walks of life. For those who can't log on as often as they'd like, the text of the public message areas is available on disk in two-month installments. Each installment comes on three 360K, 5.25" IBM PC-format disks and costs just \$15. The installment for this issue of INK (October/November 1991) includes all public messages posted during July and August, 1991.

To order either Software on Disk or Circuit Cellar BBS on Disk, send check or money order to:

Circuit Cellar INK — Software (or BBS) on Disk
P.O. Box 772, Vernon, CT 06066

or use your MasterCard or Visa and call (203) 875-2199. Be sure to specify the issue number of each disk you order. Please add \$3 for shipping outside the U.S.

TIRED OF WAITING FOR THE PROMPT ?

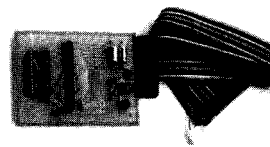
Speed up with a ROM DRIVE! Boots DOS and programs instantly. Also used to replace mechanical drive completely in controllers or diskless workstations. The only perfect protection from viruses. Easy to install half-size card.

MVDISK1 64k....\$95
MVDISK2 360k....200
MVDISK3 1.44m...300

\$95

Quantity discounts!

DOS IN ROM!



\$95 EPROM PROGRAMMER

PLUGS INTO PC
-DD PROGRAMS
-PROGRAM 2754/27010

WORLDS SMALLEST PC !!!

ROBOTS ALARMS RECORDERS DOS

THREE EASY STEPS:
1. Develop on PC
2. Download to SBC
3. Burn into EPROM

\$95

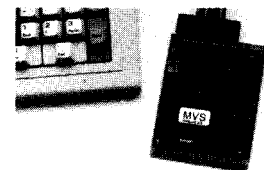
-2 PARALLEL -LCD INTERFACE
-3 SERIAL -KEYBOARD INPUT
-PC TYPE BUS -REAL TIME CLK
-BIOS OPTION -BATTERY OR 5V

FREE SHIPPING IN U.S.

5 YEAR LIMITED WARRANTY

MVS Box 994
Merrimack, NH
(508) 792 9507

8088 SINGLE BOARD COMPUTER



Reader Service #170

There's a funny thing about New Englanders and Hurricanes. For most of the year we sit around fearing that the next hurricane will blow us off the face of the earth, yet at the same time we are thoroughly disappointed when a hurricane happens to choose someone else as the target of opportunity. After all, **why go** through all the preparations and not have the satisfaction of a good story to tell afterwards.

All New Englanders have an indigenous yearning to survive through a particularly devastating hurricane and then narrate, with particular exaggeration, all the details to future generations of friends and relatives. "Boy, you should have been around back in 1954. We had Carol on August 31, Edna on September 11, and then Hazel on October 15! It took six months to dry out, and was the worst..."

Right now I'm sitting in the Circuit Cellar waiting for Hurricane Bob to zero in on me. I'm using a **UPS-backed** computer so I don't have to rewrite this whole thing after the crash. Even with the rather regular power glitches starting to occur, I've still got one TV on a local station and another on the cable Weather Channel. According to the latest reports, Bob's headed straight for the Rhode Island/Connecticut border. Unfortunately, my house is 40 miles from Rhode Island on a hill surrounded by potentially brittle trees.

Actually, I'm probably more prepared than most people to make it through natural disasters, but I equivocate over terminology like "survivalist" in any descriptions. I've already checked the propane-powered backup generator. It's one of those big two-cylinder jobs that's supposed to do the whole house. Given all the electrical goodies around here, however, I have to shed some of the big loads when I start it. It takes care of the essentials, but if it has any problems I can drag the other big Honda generator over from the garage, or use the **PTO** generator on the tractor. Or, I suppose I could attach one of the 200-W **UPSs** to a 12-V battery. You know, now that I think of it, the power has only been out for three hours in the past 11 years. But, there's a hurricane coming.. .

Let's see, I've brought the propane lanterns over, an extra 20-lb tank, checked the chain saw, and mixed an extragallon of two-cycle gas. We have one of those **9600-gallon-per-hour** 4 HP pumps (just in case) and about 100 feet of 2-inch hose, but I'll wait for the flood before dragging that out. Then again, we also have a smaller **1.5-HP** gas pump (1150 GPH), three 1/5-HP electric pumps, and a special 12-V high-volume electric pump too (just in case). Remember, there's a hurricane coming.. .

Let's see, I've pulled out some extra tarps: three 5' x 7', two 9' x 12', a 17' x 29', and a 27' x 39'. I've charged three extra 12-V 100-AH batteries for the pump, eight 12-V **6.5-AH** batteries for the radios and TVs, and made sure the Circuit Cellar automatic **low-voltage** lighting system is ready. There **are** the rechargeable flashlights (7), the regular flashlights (6) and a box of two dozen D-cells, the car cigarette-lighter-type fluorescent lights (3), and two 12-V portable power packs (just in case). After all, there's a hurricane coming.. .

Let's see, the AM/FM radio and a shortwave unit are ready but, wait, what about **a TV**? The two sets operating behind me **are** 115-V units. I don't want to have to start the generator to use them. Better go do an inventory of battery-operated stuff.

I guess I must never throw anything out. I was amazed at what was on the shelves of the Circuit Cellar. There were nine battery-operated TV sets, ranging from 1"-6" B&W and color conventional TVs to the latest 3" and 4" color LCD units. Unfortunately, no one ever discussed hurricanes with the designers or each wouldn't have had such varied operating voltages and unique connector configurations. The lowest-power LCD units had the weirdest connectors and operated at voltages like 9.8 V or 10 V. The largest conventional color TV had a readily available connector and ran on 12 V but would have consumed a whole battery between commercials.

Here was the first obstacle and the storm was only an hour away. Warm up the soldering iron, quick. Do I settle on a 1" 6-V Panasonic unit and get a magnifying glass; whip up a quick 12-V **LM317-based** adjustable regulator with assorted universal output connectors and polarity reversers; swipe one of the marine batteries from the pump and put it on the 12-V color TV anyway; or, just drag over the Honda and leave the projection TV on and forget all this battery crap? Maybe we should do both (just in case). After all, there's a hurricane coming.. .

So here I am sitting among the tarps, lanterns, propane, emergency supplies, TVs and radios, backup this and backup that, waiting for Armageddon. The wind is blowing about 40-50 MPH; the rain is pelting hard against my mostly glass house; the darkness outside has switched on all the outside lights; and the video monitor down in the Circuit Cellar is showing me a new river channel between the house and garage. All I can say after all this is it better be a damn good hurricane.

