

CIRCUIT CELLAR

I N K[®]

THE COMPUTER APPLICATIONS JOURNAL

June 1993 — Issue #35

COMMUNICATIONS

ACCESS.bus Design Tips

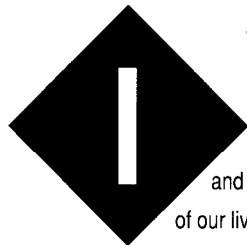
High-speed Modem
Basics

Designing With IR LEDs

Embedded Interrupts
on the '386SX

Component Selection
Issues





We Don't Talk Anymore

Last month, I wrote a bit on our "information age" and the constant data collection going on in all facets of our lives. Information as a commodity is worthless without a way to distribute it, though. Enter communications.

Infrared (optical, both free space and fiber), RF, and even power line are rapidly improving as viable communication media in both the home and the workplace as they struggle to handle the ever increasing amounts of information that is being passed around each day.

Witness also the continued growth of the Circuit Cellar BBS and other on-line services as communication media. Ever since we started the BBS over seven years ago, it has continued to serve as a premiere forum for people to exchange thoughts and ideas on computers, electronics, programming, and just about anything else you can think of.

On a much larger scale, the Internet continues explosive growth as it starts to grapple with increased commercial usage. Such "information highways" (as the White House puts it) are going to be vitally important in the future to continue the free exchange of information among the general public.

I alluded earlier to the increased use of infrared as a general-purpose communication medium. In our first article, we take a look at some of the design issues surrounding the use of IRLEDs and what kinds of distances and data rates you can expect from a given setup.

Next, the ACCESS.bus promises to clean up the clutter of cables that seems to grow out of the back of virtually any desktop PC-compatible system. Our second feature article with extended sidebar gives some background and history of ACCESS.bus and shows how to design with it.

The modem is one of the key components in the explosion of information exchange forums. Constant improvements have allowed modems to keep up with the increasing demand for faster and more reliable data transmissions. If you feel like you've been left behind by the technology over the past few years, our third feature article should catch you right up.

In our columns this month, Ed adds interrupt support to the embedded '386SX system; Jeff illustrates how component selection and PC board layout can make or break a circuit; Tom takes a trip down memory lane as he looks at a new technology that pulls core memory back from the grave; John continues his I²C exploration with some working hardware; and Russ digs out some communications-related patents.

THE COMPUTER APPLICATIONS JOURNAL

FOUNDER/EDITORIAL DIRECTOR
Steve Ciarcia

EDITOR-IN-CHIEF
Ken Davidson

TECHNICAL EDITOR
Michael Swarlzenruber

ASSOCIATE EDITOR
Robert Rojas

ENGINEERING STAFF
Jeff Bachiochi & Ed Nisley

WEST COAST EDITOR
Tom Cantrell

CONTRIBUTING EDITORS
John Dybowski & Russ Reiss

NEW PRODUCTS EDITOR
Harv Weiner

ART DIRECTOR
Lisa Ferry

GRAPHIC ARTIST
Joseph Quinlan

CONTRIBUTORS:
Jon Elson
Tim McDonough
Frank Kuechmann
Pellervo Kaskinen

Cover Illustration by Bob Schuchman
PRINTED IN THE UNITED STATES

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

CIRCUIT CELLAR INK, THE COMPUTER APPLICATIONS JOURNAL (ISSN 0896-8985) is published monthly by Circuit Cellar Incorporated, 4 Park Street, Suite 20, Vernon, CT 06066 (203) 675-2751. Second class postage paid at Vernon, CT and additional offices. One-year (12 issues) subscription rate U.S. and possessions \$21.95, Canada/Mexico \$31.95, all other countries \$49.95. All subscription orders payable in U.S. funds only, via international postal money order or check drawn on U.S. bank. Direct subscription orders and subscription related questions to The Computer Applications Journal Subscriptions, P.O. Box 7694, Riverton, NJ 08077 or call (609) 786-0409. POSTMASTER, Please send address changes to The Computer Applications Journal, Circulation Dept. P.O. Box 7694, Riverton, NJ 06077

HAJAR ASSOCIATES NATIONAL ADVERTISING REPRESENTATIVES

NORTHEAST
Debra Andersen
(617) 769-8950
Fax: (617) 769-8982

SOUTHEAST
Christa Collins
(305) 966-3939
Fax: (305) 9858457

WEST COAST
Barbara Jones
& Shelley Rainey
(714) 540-3554
Fax: (714) 540-7103

MID-ATLANTIC
Barbara Best
(908) 741-7744
Fax: (908) 741-6823





MIDWEST
Nanette Traetow
(708) 789-3080
Fax: (708) 789-3082

Circuit Cellar BBS—24 Hrs 300/1200/2400/9600/14.4k bps, 8 bits, no parity, 1 stop bit, (203) 871-1988; 2400/9600 bps Courier HST, (203) 871-0549

All programs and schematics in *Circuit Cellar INK* have been carefully reviewed to ensure their performance is in accordance with the specifications described, and programs are posted on the Circuit Cellar BBS for electronic transfer by subscribers.

Circuit Cellar INK makes no warranties and assumes no responsibility or liability of any kind for errors in these programs or schematics or for the consequences of any such errors. Furthermore, because of possible variation in the quality and condition of materials and workmanship of reader-assembled projects, *Circuit Cellar INK* disclaims any responsibility for the safe and proper function of reader-assembled projects based upon or from plans, descriptions, or information published in *Circuit Cellar INK*.

Entire contents copyright © 1993 by Circuit Cellar Incorporated. All rights reserved. Reproduction of this publication in whole or in part without written consent from Circuit Cellar Inc. is prohibited.

- 1 4** Long-range Infrared Communications
by Larry Foltzer
- 2 6** Embedded Control Using ACCESS.bus
by David Wyland
Featuring: A PC-to-ACCESS.bus Interface Card
by Robert Clemens and Tom Stockebrand
- 3 8** High-speed Modem Basics: Standards and Theory
by Michael Swartzendruber
- 5 0**  Firmware Furnace
After This Brief Interruption: IRQs and
INTs for the '386SX
Ed Nisley
- 6 0**  From the Bench
Component Selection, Inspection, Rejection
Jeff Bachiochi
- 6 4**  Silicon Update
The Ultimate RAM?
The Quest for Core Continues
Tom Cantrell
- 70**  Embedded Techniques
Putting PC Through Its Paces
John Dybowski

INSIDE ISSUE 35

- 2** Editor's INK
Ken Davidson
We Don't Talk
Anymore
- 6** Reader's INK
Letters to the Editor
- 8** New Product News
edited by Harv Weiner
- 78** Patent Talk
Russ Reiss

- ConnectTime 85**
Excerpts from
the Circuit Cellar BBS
conducted by
Ken Davidson
- Steve's Own INK 96**
Steve Ciarcia
Eat at Joe's
- Advertiser's Index 81**

READER'S INK

INTERRUPTS ARE IN THE EYE OF THE BEHOLDER

"Given the choice between an interrupt-driven system or a polled one, I would opt for the interrupt system any way I could get it. The event handling is cleaner and much more well defined.. ." James Grundell, "Add Interrupt Support to Polled Parallel Ports," *The Computer Applications Journal*, March 1993.

Would that it were the whole story. There are two big problems with interrupt-driven systems. The first is that polling almost always offers better performance than interrupts. The second is that interrupt-driven systems are very difficult to test adequately.

The first consideration-performance-applies when CPU limits are being pushed. Polling usually offers better performance because interrupts almost always require saving and restoring more state information than does a polling loop. Interrupts are fine for handling a 9600-bps data line on an IBM PC/XT, but if you try to push the same line to 100 kbps, you need to go to a polling loop. It is easy to convert polling software to interrupt-driven logic, but not vice versa. If a system is designed interrupt driven and an attempt is made later to push the hardware limits, a major software rewrite may be needed.

The second problem is not one of testing the interrupt itself. That's easily done. The problem is that interrupt-driven systems are almost always indeterminate in the sense that no given set of test stimuli are 100% controllable or reproducible. Interrupts alter logic sequencing in a random fashion and introduce some truly fascinating bugs. Interrupt-driven systems, especially large systems, are prone to have transient, unreproducible problems. They usually are not tested for the worst case, because nobody knows what the worst case is and were it known, nobody would know how to create it.

So you shouldn't use interrupts? Of course you should-where they are appropriate. But you should understand that interrupts are not an unmixed blessing and that you may pay a price in performance and/or reliability when you opt to use them.

Donald Kenney
Canton, Mich.

SOME CHILLING THOUGHTS

I enjoyed Philip C. Pilgrim's article "Build a Single-chip Video Wind Gauge" in your March 1993 issue, and it was obvious why the Circuit Cellar staff picked it as

the esign contest winner. But what really caught my interest was his mention of wind chill in his closing comments. For a long time, I've wondered about this notion. How was it determined, anyway? Certainly it wasn't as "easy" as relative humidity, which is hard enough, but at least has a precise definition.

By a strange coincidence, I just came across a "formula" for wind chill published by a local TV station:

$$WC = 91.4 - [(91.4 - T)(0.478 + 0.301 \sqrt{V} - 0.02V)]$$

where V is in miles per hour and T is in degrees Fahrenheit. A phone call to the station resulted in prerecorded messages, so I have no idea about typos and so forth.

No doubt Mr. Pilgrim could tackle such a formula in assembly language, but it might foul up his video timing! Possibly the PIC 16C71, which he alluded to, would do the job.

My question, addressed to any *Computer Applications Journal* readers who would know, is: Is wind chill based on any meteorological theory or (as I suspect, since the dimensions don't match) is it just an empirical formula concocted to fit a table which is based on a subjective feeling? How do we measure "how cold it feels outside"?

Dana Romero
Salt Lake City, Utah

"V" STANDS FOR VARIABLE

I would like to compliment you on the magazine; I have been an avid reader of *Circuit Cellar INK* since its introduction. The articles are interesting and usually of high quality and accuracy. However, the article in the April issue about CVSD by Jeff Schmoyer had one fundamental inaccuracy: what Jeff described was delta modulation, not CVSD.

CVSD is a derivative of delta modulation where the step size varies in a continuous fashion as a result of recent history of the data. This is where the "continuously variable slope" part of the name comes from.

As was noted in the article, one problem with delta modulation is slope overload if the input is too high in amplitude at high frequencies, while preserving signal-to-noise ratio at low levels. The step size gradually reduces if these overload conditions do not exist.

Motorola has a good explanation of their implementation in their "Telecommunications Data Book." They vary the size of the step by feeding the overload informa-

READER'S INK

tion into a first-order low-pass filter. The demodulator uses the output of this filter as the step size. The modulator uses an identical mechanism.

CVSD can achieve a dynamic range of 40-50 db for voice while running at a 32-kbps data rate. One megabit of memory (such as an EPROM) will hold about 30 seconds worth of speech.

The software listing in the article modeled a delta modulator with a perfect integrator (a typical hardware implementation would use a "leaky" integrator made from a first-order low-pass RC filter). The software to correctly convert PCM to CVSD will be significantly more complex. It is closely related to the techniques used in many CD players with single-bit D/A converters. In this case, a DSP converts the 16-bit PCM data from the CD into a single-bit data stream by modeling the third- or fourth-order filter used in the bit-stream demodulator.

Kevin White
Los Gatos, Calif.

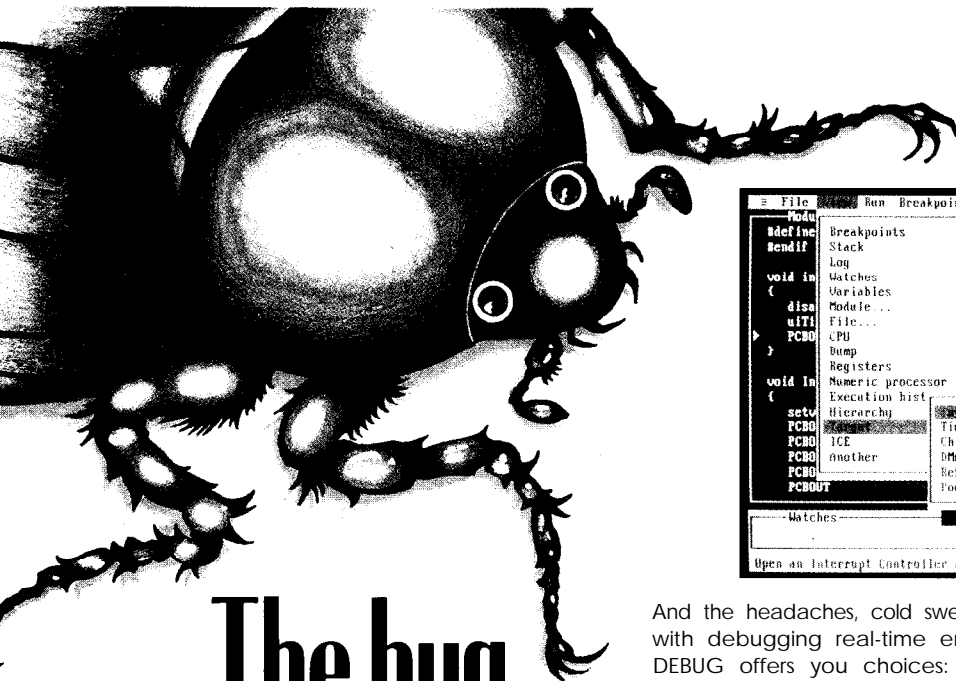
CORRECTION

In the February, 1993 issue (#31) in Steve Ciarcia's "Temperature Monitoring" article, Figure 5 on page 40 doesn't quite match the caption or the description in the text of the article. To match the diagram to the caption and article text, the offset and gain stages must be reversed; that is, the gain stage must come first, followed by the offset stage. Alternatively, you may use the diagram as published if you set the offset to 0.2 volts.

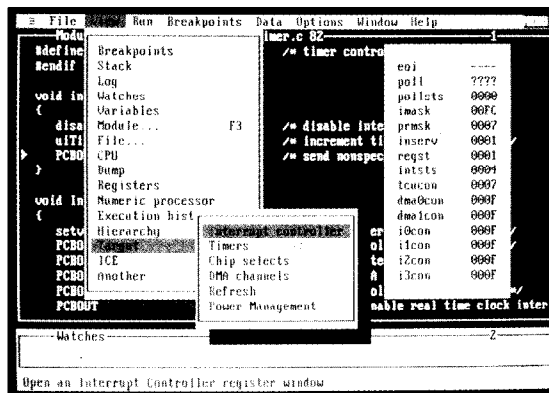
We Want to Hear from You

We encourage our readers to write letters of praise, condemnation, or suggestion to the editors of the Computer Applications Journal. Send them to:

The Computer Applications Journal
letters to the Editor
4 Park Street
Vernon, CT 06066



The bug stops here



NOW SHIPPING!
VERSION 3.0
C++ Templates, Clipboard,
Breakpoint Groups & More!

And the headaches, cold sweats and other symptoms associated with debugging real-time embedded applications. Paradigm DEBUG offers you choices:

- Intel or NEC microprocessors
- Remote target or in-circuit emulator support
- C, C++ and assembler debugging
- Borland, Microsoft and Intel compatibility.

Kickstart your embedded system with the only debugger family to have it all. Give us a call for Paradigm DEBUG ...before it's too late!

800-537-5043

PARADIGM DEBUG

Proven Solutions for Embedded C/C++ Developers

PARADIGM: (607) 748-5966
FAX: (607) 748-5968

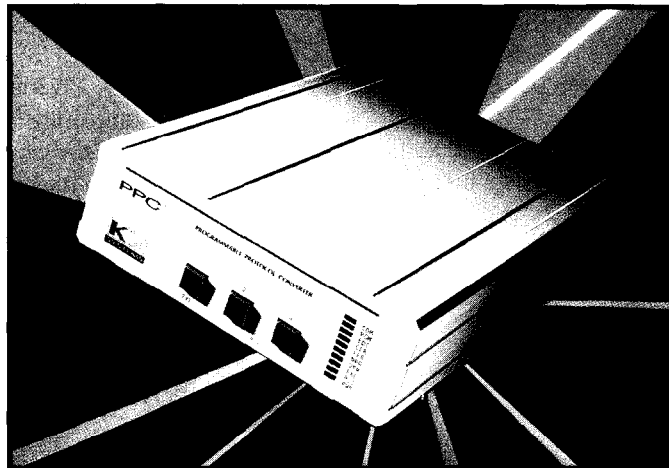
NEW PRODUCT NEWS

Edited by Harv Weiner

PROGRAMMABLE PROTOCOL CONVERTER

The PPC **Programmable Protocol Converter** from the Saelig Company makes it easy for two incompatible pieces of computerized equipment to talk to each other. Converting between differing data formats and replacing words or strings is easily accomplished with a four-line program.

The PPC is a unique device for translating incoming data streams to new formats "on the fly," and at high speeds. Applications such as custom file converters, device emulators, and



CNC language modification can be rapidly achieved, using easy-to-use BASIC, or full-featured Pascal.

The PPC is connected to a PC and then loaded with a Pascal or BASIC program. The learning curve

for using this device is virtually zero. For the most common PPC applications, the program is surprisingly small.

Ready-made routines such as CAD/CAM device drivers are supplied.

Programs are stored in EEPROM on the PPC so they can be easily modified. The base-configured device comes with two RS-232 ports (four optional). Other options for the PPC include a real-time clock, 16-bit analog I/O, and memory cards.

The PPC Programmable Protocol Converter sells for \$699.

The Saelig Company
1193 Moseley Rd.
Victor, NY 14564
(716) 4253753
Fax: (716) 425-3835

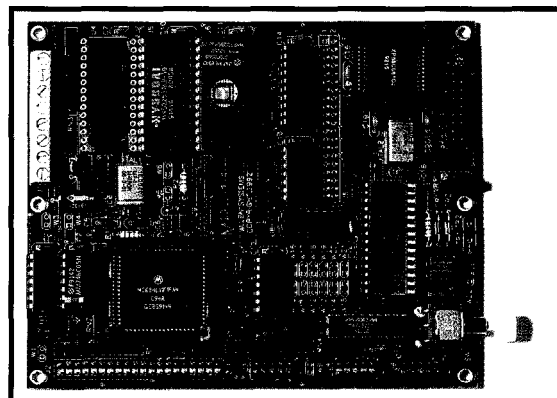
#500

68HC11-BASED SINGLE-BOARD COMPUTER

A single-board computer, designed for process control applications and based on the Motorola 68HC11F1 microcontroller, has been announced by Allen Systems. The **MP-11** is ideal for robotics and CPU-intensive, process control applications and provides a prototyping base to expedite 68HC 11 F 1 development.

The MP-11 contains a number of features that distinguish it from other cards based on the 68HC11. Included in the features are: 16-MHz operation, power and ground planes for noise minimization, a processor supervisory circuit, an optional DS1286 clock/calendar, an optional 8-kbyte EEPROM, and an expansion connector that can support an optional analog (includes A/D and D/A) daughter board, or custom circuitry designed by the user.

The MP-11 supports up to 3 1.5 kbytes of EPROM and 27 kbytes of static RAM. A socket is provided for an



optional 8-kbyte EEPROM. Up to three 8-bit parallel ports are available on the chip, depending on resource usage. In addition, the 88C681 DUART provides one 8-bit output port and one 7-bit input port. Three UARTs, two of which are RS-232C buffered and brought out to header connectors, are available on the board. An optional analog daughter board, providing four channels of 12-bit A/D conversion and two channels of 12-

bit D/A conversion is also available. The MP-11 measures 4.5 inches by 5.5 inches and requires 5 volts DC at a maximum current of 125 mA.

The MP-11 is available as a bare board with a User's Manual for \$100, or assembled and tested for \$300. The EEPROM and DS1286 clock/calendar option costs \$50.

Allen Systems
2346 Brandon Rd.
Columbus, OH 43221
Voice/fax: (614) 488-7122

#501

NEW PRODUCT NEWS

IN-CIRCUIT EMULATOR

Two low-cost, nonintrusive, in-circuit emulators for Microchip's PIC16C5x series and PIC16C718-bit RISC microcontrollers have been announced by Advanced Transdata Corporation. The ICE-16C5x and ICE-16C71 emulators provide an interactive development environment for debugging PIC 16Cxx applications. They run on any IBM PC (or compatible), including laptop and notebook computers. These emulators interface with the PC through the parallel printer port. The emulator designs use the PIC 16Cxx microcontrollers for true hardware emulation, supporting RTCC, WDT, all I/O ports and special function registers. The host PC simulates the execution of all PIC instructions.

The windowed development environment provides separate windows for examining source code, program memory, data file registers, watched variables, processor status, program counter, and stacks. Each window can be sized, moved, added, or removed to customize the debugging environment to the user's taste. Hot keys, on-line context-sensitive indexed help, and complete mouse support all add to the development system's ease of use.

Source level debugging and full symbolic debugging are available on the TASM16 and TASM71 cross-assemblers, which are included with the ICE package. The units provide comprehensive emulation controls and, as the user single steps execution, each piece of updated information is highlighted for easy reference.

Both emulators provide a 4-kbyte (1 kbyte deep by 32 bits wide) trace buffer that captures ICE trace data and records program flow in real time. Eight software breakpoints and two hardware-break triggers can be set to break on any address or external signal.

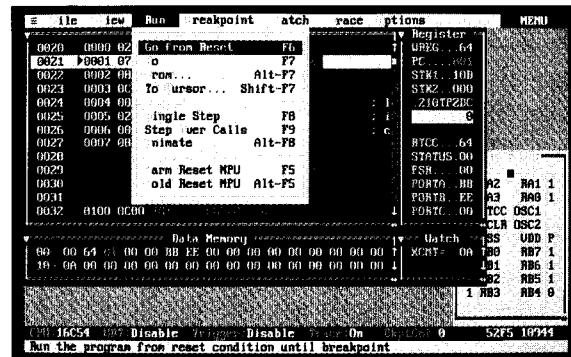
Each ICE for the PIC microcontrollers consists of a compact, portable emulator unit (measuring 4.75" x 2" x 1"), its respective cross-assembler, simulator software, emulator cables, a trigger source input cable with probe clips, parallel extension cable, and power adapter.

The ICE-16C5x sells for \$395 and the ICE-16C71 sells for \$445.

Advanced Transdata Corporation

14330 Midway Rd., Ste. 104 • Dallas, TX 75244 • (214) 980-2960 • Fax: (214) 980-2937

#502

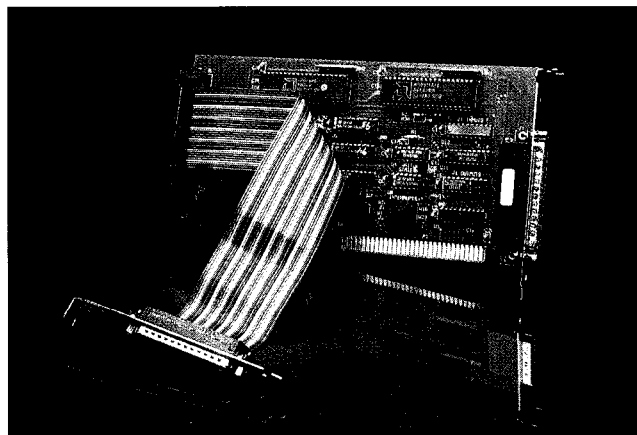


PC/AT COUNTER-TIMER BOARDS

Analogic Corporation has announced the CTRTM Series of counter-timer boards for PC/AT and compatible computers. The boards feature either five- or ten-channel general-purpose 16-bit counters and are ideal for applications in event counting, frequency synthesis, coincidence alarms, or complex pulse generation.

The CTRTM boards are both register and connector compatible with the industry standard, and offer a 1- or 5-MHz internal clock for greater flexibility. A variety of internal frequency sources and outputs can be selected as inputs for individual counters. Each counter can be gated in hardware, or by software, and can be programmed to count up or down, in either binary or BCD. In addition, the counters may be connected together by software to form a 32-, 48-, or 80-bit counter.

The CTRTM-05 five channel Counter Timer sells for \$225 and the ten channel CTRTM-10 sells for \$390.



Analogic Corporation
360 Audubon Road • Wakefield, MA 01880
(508) 977-3000 • Fax: (617) 245-1274

#503

NEW PRODUCT NEWS

RADIO MODEM BOARD SET

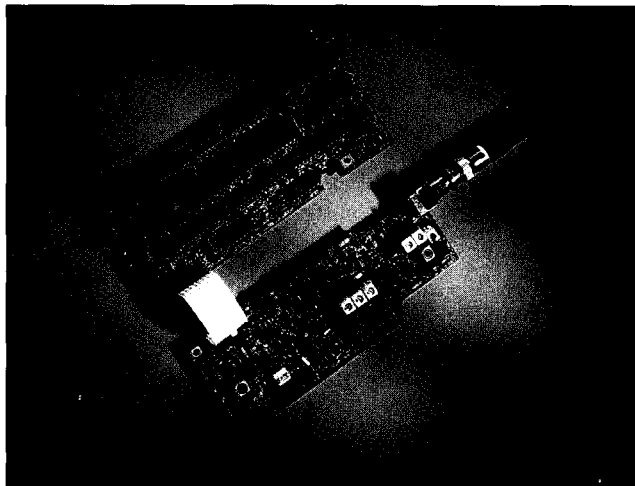
A new radio-transmission modem for OEM use in computers and peripherals has been announced by Monicor Electronics Corp. The **System 200** is a board set that completely eliminates the interconnecting cables between installed systems.

The System 200 consists of two PC boards; one of them being a specially designed digital transmitter/receiver and the other a high-performance modem. The two boards are linked to the DTE by a flexible cable that carries only low-frequency signals and direct current. The power requirement for the System 200 is 7.5 volts DC at 250 mA maximum, and can be supplied via the RS-232 interface connector or by NiCd batteries on the modem board.

Any output level from 2 W to 1 mW can be specified for the transmitter. A 2-W output can provide line-of-sight connectivity for two miles or more. At 250 mW, signals are useful over a million square feet of enclosed warehouse space. At 1 mW, connectivity radius is about ten feet, with no interference with other nearby radio transmissions. Options up to 10 W (and higher) are available.

System 200 comes with Monicor's highly efficient TurboLink 2.0 Operating System that can accommodate up to 48 terminal nodes. This operating system resides in an EEPROM on the modem board, or users can install their own operating system. TurboLink 2.0 supports the X.30 protocol for intelligent modem networking with DEC and IBM hosts, personal computers, and preexisting LANs.

The communications link (DTE to DCE) uses an asynchronous serial RS-232 protocol. The radio protocol



is based on a scan sequence/collision detection system. The data is encoded in Frequency Shift Keyed (FSK) Manchester II format. The data packet size is variable from 16 to 128 characters.

The transceiver operates using narrow band FM and is factory set between 450 and 470 MHz. The base station uses a quarter-wave whip antenna and peripheral nodes use a rugged heliflex antenna. The receiver sensitivity is specified at 0.5 μ V/12 dB SINAD or better.

The System 200 Radio Modem Board Set is priced at \$465 in OEM quantities.

Monicor Electronics Corporation
2964 NW 60th St. . Fort Lauderdale, FL 33309
(305) 979-1907 . Fax: (305) 979-2611

#504

ON-LINE MAGAZINE INDEX

R&D Publications has released their **On-Line Magazine Index** (1988-1992) for articles published in the *C Users Journal* and *Windows/DOS Developer's Journal* during the years 1988 through 1992.

The Index allows searches by author, title, and keyword. All articles, columns,

editorials, reviews, product user reports, and readers' letters with technical content are indexed. The Index also includes information about the C User's Group Library, new releases, bug fixes, and updates.

The Index provides the professional developer with a quick way of identifying and retrieving information from the two magazines on

such topics as C, C++, Windows, and DOS. The Index also allows grouping and printing of searched records.

The Index was compiled by Stephen Bach, edited by Bernard Williams, and programmed by Kenji Hino.

The On-Line Magazine Index 1988-1992 sells for \$29.95 and is available on

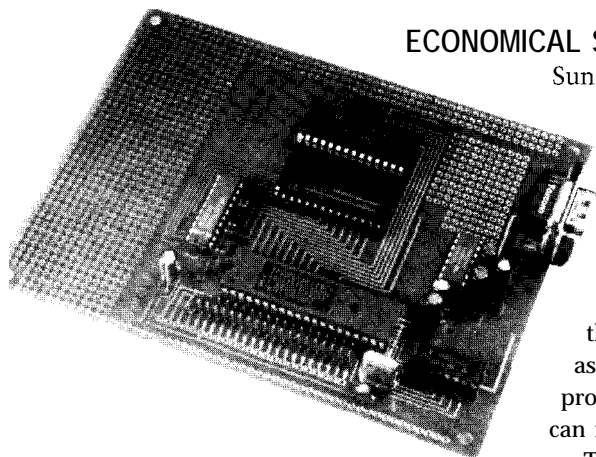
3.5" or 5.25" diskettes. Windows 3.1 is required.

R&D Publications
1601 W. 23rd St., Ste. 200
Lawrence, KS 66046
(913) 841-1631
Fax: (913) 841-2624

#505

NEW PRODUCT NEWS

ECONOMICAL SINGLE-BOARD COMPUTER



Suncoast Technologies introduces the **70691RAM**, an economical single-board computer based on the SOS 1 microcontroller chip. The unit features 8 kbytes RAM, a standard RS-232 interface, and a large prototyping area.

The 70691RAM is compatible with compiled BASIC-52 programs such as the code produced by Binary Technology's BXC5 1. This allows development of 805 1 assembly programs in BASIC using an 8052AH BASIC development system. Once the BASIC program is created, it can be transformed into an assembly language file using a BASIC compiler. The compiled program is fully compatible with the 70691RAM computer and can run on the inexpensive 805 1 microcontroller.

The 11.059-MHz clock allows the 70691RAM to be programmed or any baud rate from 300 to 9600. High address decoding is provided in eight increments of 2 kbytes, and the address decoder lines are brought out to a 2x20-pin header. Also included on the header are all major 805 1 signal and control lines.

The 70691RAM board measures 4.5 inches by 6 inches and requires only a five-volt DC power supply. A 9-pin D-type connector terminates the RS-232 line and an EPROM socket is provided for custom programs.

The 70691RAM single-board computer sells for \$50. A CMOS version, featuring the 80C31/51 chip, sells for \$56.

Suncoast Technologies

P.O. Box 5835 • Spring Hill, FL 34606 • Voice/fax: (904) 596-7599

#506

MOVE OVER INTEL MICROMINT SOURCES 80C52 CMOS BASIC CHIP

Micromint has a more efficient software-compatible successor to the power-hungry Intel 8052AH-BASIC chip. The 80C52-BASIC chip was designed for industrial use and operates beyond the limits of standard commercial-grade chips. Micromint's 80C52-BASIC chip is guaranteed to operate flawlessly at DC to 12 MHz over the entire industrial temperature range (-40°C to +85°C). Available in 40-pin DIP or PLCC

80C52-BASIC chip	\$25.00
OEM 100-Qty. Price	\$14.50
BASIC-52 Prog. manual	\$15.00

MICROMINT, INC.

4 PARK ST., VERNON, CT 06066

TO ORDER CALL

1-800-635-3355



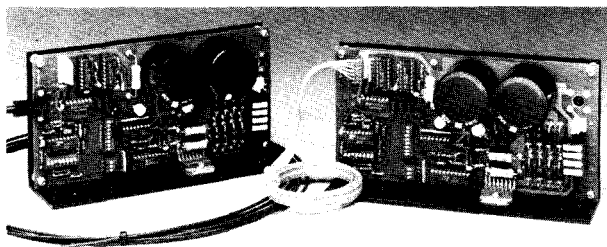
STEP MOTOR DRIVER

Buy our **7006-DB** Chopper Drive for \$130
& we'll throw in the motor* for \$15

- *High-speed pulse-width-modulated drive for motors to 20ksteps/sec.
- All power supply components are on-board (except transformer).
- *Simply connect to 24VAC (range is 12-24VAC or 18-40VDC).
- *Automatic Current-Reduction mode (adjustable).
- 128 dip-switch selectable currents to **2A/phase** (conservatively rated).
- *Half-step and full-step modes. Enable/disable function.
- *On-board oscillator for stand-alone mode (using external dpdt switch).
- Clock and Direction inputs. Led's indicate motion and power.
- Optional OPTICAL FIBER control. (The only one in the business).

*50 oz-in, 3700 steps/sec. size 23, 1.8°/step.

Offer ends June 30th, 1993



Ask for our FREE Catalog

American Scientific Instrument Corp

PO Box 651

Smithtown, NY 11787

(516) 361-9499 Tel

(516) 265-6241 Fax



NEW PRODUCT NEWS

SOFTWARE ANALYZER

General Software has announced **CodeProbe**, a software analyzer for DOS developers. Designed as a companion tool to debuggers and profilers, this new product enables the developer to monitor a program running at full speed and capture system events such as hardware, DOS, and BIOS interrupts. It can also capture user-defined events triggered by the user's code calling a special trace function from C or assembly language. After the events are captured, the developer can display the trace in summary and fully decoded forms, with each event time-stamped to sub-millisecond resolution.

CodeProbe installs directly on any DOS-based PC, AT, or 386/486 based machine, and runs concurrently with the software under test. During event capture mode, CodeProbe's full-screen display shows event traffic by type (DOS, BIOS, user, or other) with real-time bar graphs, giving the developer a good feel for the activities being generated by the software under test. CodeProbe can also be used to analyze operating systems and network operating systems to determine which DOS and BIOS functions are used along with relevant timing information.

The software works by storing each event in a "ring buffer" maintained in a reserved area of memory. Each event is time-stamped with a special query of the PC's hardware timer chip. By reading the timer registers directly, 0.838-microsecond resolution is possible.

The product is based on the same idea employed in network protocol analyzers to capture "live" traffic from the network and display the trace of captured packets on the screen. Unlike profilers, the software analyzer actually records sequences of events and shows the elapsed time between them. This enables the developer to take a "micro" rather than "macro" view of the system.

CodeProbe is priced at \$350.

General Software, inc.

P.O. Box 2571 . Redmond, WA 98073 • (206) 391-4285 . Fax: (206) 557-0736

#507

ECAL Universal Assembly Language Development System

- ▶ Alternative to Real-Time Emulator
- ▶ Support for 8051, 8096, and 186
- ▶ Support for 170+ additional processors
- ▶ User control of syntax and instructions
- ▶ Extremely fast assembly-2 Kbytes/sec
- ▶ Integrated split-screen editor or command-line assembly supported
- ▶ Integrated linker/loader
- ▶ Instruction trace and I/O windows
- ▶ Monitor and RS-232 com. windows
- ▶ Single micro processor versions available
- ▶ Optional EPROM emulator and programmer
- ▶ Source-level debugger

Ordering Information

- 05-0200-010 ECAL OAS
- 05-0200-020 ECAL with EPROM Emulator
- 05-0200-XXX ECAL Single Processor

Product Information

ECAL is a complete assembly-language development system that provides all the tools needed to assemble, link, load, run, and debug your project for over 170 processors. By using user-editable control files, the ECAL macroassembler in its full configuration can handle 4-, 8-, 16-, or 32-bit microprocessors with unsurpassed speed and consistency.

Using the familiar DOS-based text windows, you can edit, assemble, set breakpoints, trace execution, watch registers and I/O, and communicate with your target's serial port in separate closable windows. If you prefer to use other tools, with a few keystrokes, ECAL will incorporate your previous work into its consistent and intuitive environment.

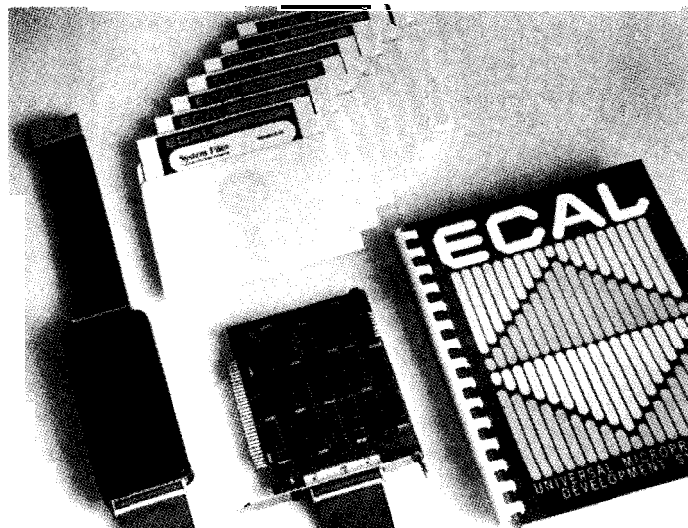
The free ECAL evaluation program features all of the ECAL tools for all of the supported micros, giving you a true sampling of ECAL development cycle (source and object length limited).

VAIL Silicon Tools sells and supports ECAL and can bundle ECAL with additional hardware and software to satisfy your need for economical project development tools.

Vail
Silicon Tools

Contact

Vail Silicon Tools
692-A S. Military Trail
Deerfield Beach, FL 33442
Tel: (305) 570-5580
Fax: (305) 428-1811

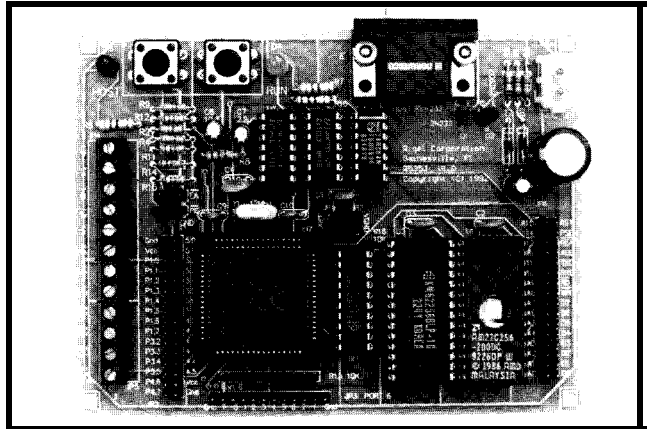


NEW PRODUCT NEWS

TRAINING SYSTEM FOR 8031 FAMILY

Rigel Corporation introduces a low-cost training system for 8031 microcontrollers. The system consists of a **R-535J Prototyping Board**, **READS** (Rigel's Embedded Applications Development System), and sample programs. The R-535J accepts the 80C535 microcontroller in the 68-pin PLCC. The board has terminal blocks connected to digital I/O ports, with 28 I/O ports available. System signals are available at two 32-pin headers. The R-535J has a monitor EPROM and 32 kbytes of SRAM. A two-way reset allows programs to be placed in low memory, giving access to all interrupt vectors.

The R-535J/READS system allows writing, assembling, downloading, debugging, and running applications software in MCS-51 language. READS has an editor, a cross-assembler, and provides development board communications in a menu-driven environment. Debug functions include: break points, single-stepping, source-level debugging, and inspecting/modifying external memory, internal registers, and special function registers. READS includes a comprehensive on-line help system. All functions are accessible by hot-keys.



This system is ideally suited for home study, or as an embedded controller.

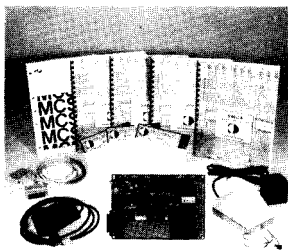
The R-535J/READS package sells for \$150. A kit is available for \$130.

Rigel Corporation
P.O. Box 90040 . Gainesville, FL 32607
Voice/fax: (904) 373-4629

#508

The \$595 Solution to 8051 System Development

PDK51



- SIBIC-II microcontroller board (8052AH-BASIC CPU) with 16K RAM
- BASIC interpreter source on disk
- SIBIC-II hardware manual with schematic
- Monitor/debugger ROM plus manual
- Monitor/debugger hardware source listing
- UL listed 5W power supply
- Programming power supply adjustable from 4V to 26V
- K1KMH1 communications program
- SXA51 8051/8052 cross assembler
- BTK52 BASIC programmers toolkit with tutorial
- 160 page BASIC Programming Manual by Systronix
- 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 BXC51 8051/8052 BASIC compiler—\$800.

Call Now! 508-369-9556 or FAX 508-369-9549



Binary Technology, Inc.

P.O. Box 541 . Carlisle, MA 01741



#106

Fast Code Relief!



- Supports 8051 and Dallas DS5000 families
- Program in easy industrial BASIC (similar to GW-BASIC)
- Artificial Intelligence engine assists you with setup tasks
- Ring buffered interrupt driven serial I/O
- True time-based and external interrupts
- Supports unlimited in-line assembly code
- Your program can coexist with BASIC-52
- Same day shipping and 30 day money-back guarantee
- Competent technical support

"A potent development combination..." EDN January 20, 1992

Systronix Inc.

555 South 300 East
Salt Lake City, Utah 84111

TEL: 801-534-1017
FAX: 801-534-1019

#107

FEAT'URES

14

Long-range Infrared Communications

26

Embedded Control Using ACCESS.bus

38

High-speed Modem Basics: Standards and Theory

FEATURE ARTICLE

Larry Foltzer

Long-range Infrared Communications

IR communications are useful for more than just controlling your TV. High data rates and long distances are possible with off-the-shelf devices as long as you know the proper design techniques.



ircuit **Cellar**
INK has presented many articles over the years that dealt with the subject of free-space infrared communication. Moreover, recent articles about the HCS II have used free-space optical links for remote control and people tracking applications (*The Computer Applications Journal*, April/May 1992, issue #26). Common to all of these applications are a relatively slow transmission speed and short-distance operations. As a long-time veteran in the field of optical fiber transmission technology (19 years, and then some), I was curious about exploring the limits of what could be done with free-space links using low-cost infrared LEDs, PIN photodiode detectors, and small lenses. What I discovered was that transmission rates approaching 100 kbps and transmission distances in excess of a thousand feet are possible using high-speed modulation techniques rather than the "standard 40-kHz" remote control system designs.

Interestingly enough, I found that one can increase the transmission bandwidth considerably with virtually no decrease in link range. Depending upon which modulation technique you use, you can even reduce transmitter requirements. The only penalty you

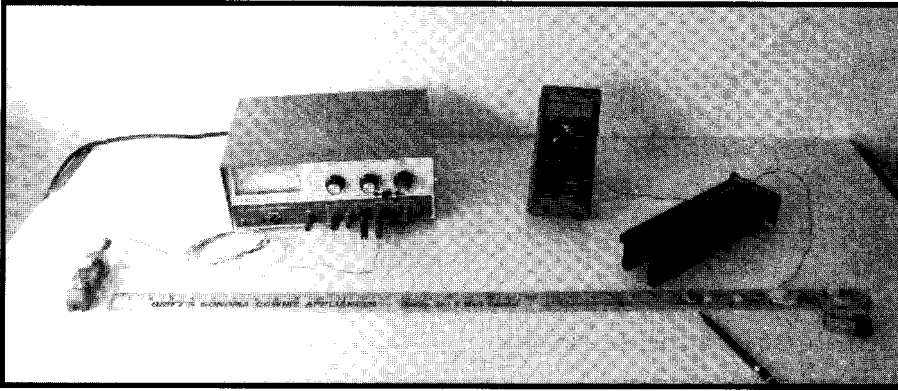


Photo 1—The radiant intensity measurement detector assembly (at the pencil point) is mounted on a yard stick so the distance to the LED can be easily measured. The black foam shield behind the detector straddles it to reduce ambient light interference during measurement.

pay for this performance increase is that the circuit designs become discrete implementations rather than single-chip solutions.

In this article, I will demonstrate how to characterize an LED and detector pair for potential application in high-speed, line-of-sight data transmission links. The results of the characterization will then allow you to make a first-order prediction of the range you should expect from a particular optical system design. I begin by discussing some of the fundamental and interdependent characteristics of infrared LEDs. Following that, I describe the so-called “radiant intensity” parameter that is often absent from manufacturers’ data sheets, and yet is absolutely crucial to system range prediction. I will show you how to construct a simple apparatus that will enable you to characterize LEDs for potential use in data transmission systems.

Next, I present the design of a high-speed transmitter and receiver that you can incorporate in your own applications. If you use some of the faster LEDs, you can expect transmission rates up to about one megabit per second. Finally, I conclude by presenting some sample calculations showing the range of performance one might expect from optical links, with and without lenses, based on the measurements described here.

LED BASICS

The most important characteristics of an LED for use in transmission links are its emission wavelength (color), switching speed, and radiant

intensity. You can measure the last two of these parameters yourself, but in all likelihood, you will not have the ability to measure the emission wavelength in your home laboratory, since the equipment needed to do this is rather expensive. This is of little consequence, since the infrared LEDs you have in your junk box most likely emit energy in one of the following three wavelength regions: 880 nm, 900 nm, or 940 nm.

The 900-nm LEDs are typically zinc-doped gallium arsenide (GaAs) devices and represent the oldest LED technology. While it is true that they have the lowest output power, their response time is quite fast. The turn-on (rise) and turn-off (fall) times of these components are typically less than 50 and often less than 10 nanoseconds. Chances are you don’t have any of these, but if you do, you may be able to tell by measuring the rise and fall times of their output signal. You will be able to use them for short-distance or wide-bandwidth applications.

LEDs that emit energy in the 930-nm to 950-nm region represent second-generation devices. Chances are good that you have some of these in your junk box. These components are made of silicon-doped GaAs. Silicon doping increases power output by making the LED transparent to its own emissions, but this gain comes at the expense of speed. The response time of these devices (rise and fall) is typically on the order of a microsecond or so, limiting them to applications that require less than 350 kHz of bandwidth. This technology, while old,

survives today because it yields very high power devices that are well matched to the response characteristic of filtered photodetectors. These devices are frequently used in remote control transmitters.

The 880-nm emitters are made of gallium-aluminum arsenide (GaAlAs). They exhibit high-power output and have greater speed than their 940-nm counterparts. In fact, the output power of these devices—relative to the 940-nm LEDs—is so high that it completely compensates for the lower response of typical filtered detectors at the 880-nm wavelength of excitation as compared to the output of the same detector when excited by energy in the 940-nm region. The speed of these devices typically falls in the range of 100300 ns. If you intend to purchase an LED for an application that requires less than 1-MHz bandwidth, I suggest you consider the 880-nm devices.

RADIANT INTENSITY

The most important parameter for determining the range capability of a free-space optical link is the Radiant Intensity (RI) of the transmitter’s LED. The RI of a source gives us a convenient way of calculating the flux density of a beam at some arbitrary distance from the source, and therefore the power that a receiving aperture can potentially collect at that distance. The RI of a source is expressed in the units of watts per steradian (W/sr), which inherently refers to the way in which a cone-shaped beam of light diverges as it propagates through space. The steradian is the unit of measure of a cone’s solid angle since the cone is a three-dimensional figure.

Be careful not to confuse the steradian and the radian. The one-dimensional [planar] included angle of a one-steradian cone is *not equal* to one radian!

THE STERADIAN

The easiest way to understand what a steradian is is to start with something familiar, for example the surface area of a sphere. The surface area of a sphere is determined from the following well-known relationship:

8051 SBC AT A NEW LOW PRICE

We are proud to offer our standard **8031SBC-10** Single Board Computer at a new, low price - just \$79 per unit or as low as \$49 each for quantity purchases. An 8031 with two JEDEC sockets, one RS232, 5V regulator, expansion connector. Optional second serial port, **80C31** or 32.

Controller 80C552

At \$149, our **552SBC-10** has the price and features you need right now! It's an 8051 core processor with an eight channel, 10-bit A/D, two PWM outputs, capture/compare registers, one RS232, four JEDEC memory sockets, and more digital I/O. And we didn't stop there! You can add options like two more RS232/422/485 ports, 24 more digital I/O ports, Real-Time Clock, EEPROM, and battery-backup for clock and RAM right on board. Start with the Development board; it has all the peripherals plus a debug monitor for only \$349. Download and debug your code right on the SBC, then move to the OEM board above for your production needs. We also do custom design work - call for our reasonable prices.

New 8051 Family Emulator Support

Our **DryICE** Plus product has been expanded to include support for the Siemens **80C537**. The base emulation unit is still only \$299, with the **80C537** pod priced at \$199. Other 8051 family processors supported are 8031132, **80C31/32**, 8751/52, 87C51/52, **80C154**, **80C451**, **80C535**, **80C552/562**, **80C652**, and **80C51FA,B,C**. Each of these pods is priced at \$149. Where else can you get an emulator with this much power and flexibility for only \$448 - complete?

Our original stand-alone 8031 ICE is still priced at \$199. Though not as flexible as the DryICE Plus, it offers excellent price/performance for learning or the occasional job need.

Call for your custom product needs.
Free Quote - Quick Response



HiTech Equipment Corp
9400 Activity Road
San Diego, CA 92126
[FAX: (619) 530-1458]

(619) 566-1892

$$A_{\text{sph}} = 4\pi r \quad (1)$$

where r is the radius of the sphere.

To help visualize what a 1-steradian cone looks like, draw a circle on the surface of a sphere whose radius is 0.271 times the diameter of the sphere. The spherical or convex surface area enclosed within that circle is then equal to the radius of the sphere squared. Equation 1 indicates 4π of them make up the total surface area of the sphere. The 1-steradian cone is formed by drawing lines of intersection from the center of the sphere to the endpoints of the diameter of the circle that bounds the encircled area on the surface of the sphere. The aspect ratio of the 1-steradian cone remains the same regardless of the radius (distance) of the cone.

$$SR = \frac{A}{r^2} \quad (2)$$

In Equation 2, A is the spherical surface area at the top of a cone, r is the radius of the cone, and SR is expressed in steradians. Note that when A is equal to the radius squared, $SR = 1$.

Solid Angle (steradians)	Total Angle (milliradians)	Angle (degrees)
1	1144	65.54
0.1	357	20.47
0.01	113	6.47
0.001	36	2.04
0.0001	11.3	0.65
0.00001	3.6	0.21
0.000001	1.2	0.07

Table 1—Relationship between 3D/solid angle (steradian measure) and 2D/planar angle.

The mathematical relationship between the solid angle of a cone (steradians) and the total included planar angle (degrees or radians) is given below. Table 1 shows the corresponding values for steradian measure and planar angle measure.

$$\theta = 2 \times \cos^{-1} \left(1 - \left(\frac{SR}{2\pi} \right) \right) \quad (3)$$

In Equation 3, SR is the solid angle of the cone in steradians, and θ is the total included angle of the cone in degrees or radians.

RI MEASUREMENT APPARATUS

The measurement of a source's RI can be done with a simple setup like

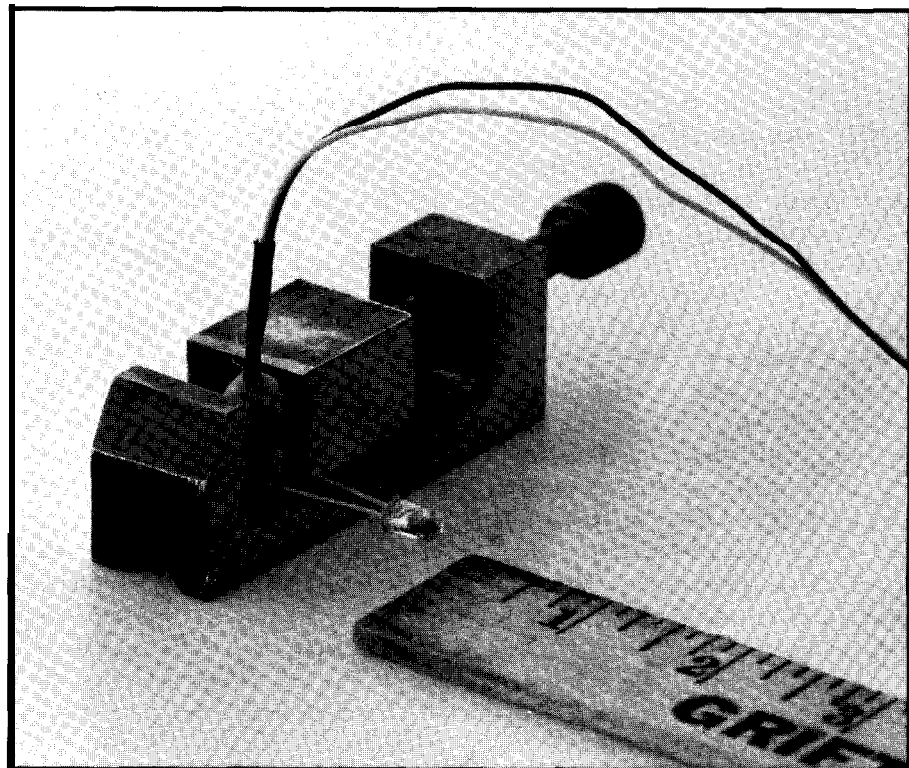


Photo 2—An LED under test is shown held in a socket in a vice that is positioned at a convenient distance from the detector. You may need to adjust the pitch and yaw of the LED to peak the reading. LEDs with strong integral lenses are especially sensitive to angular misalignment.

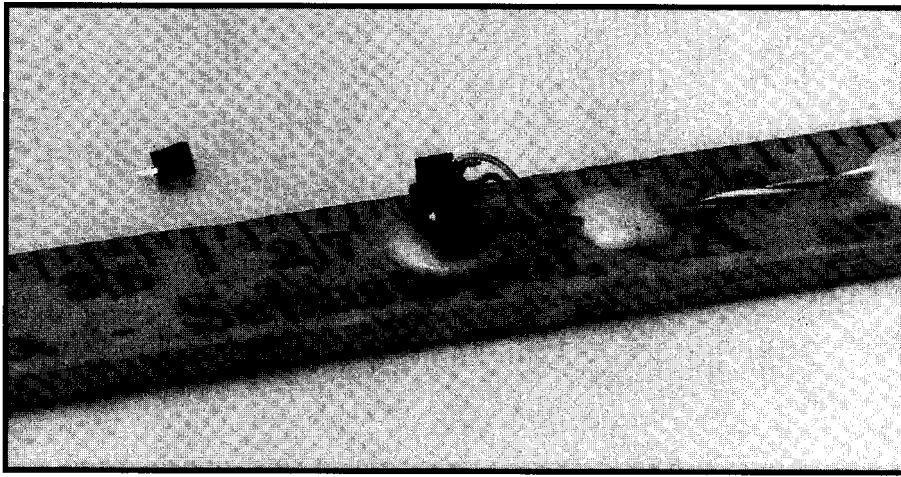


Photo 3—Hot glue was used to attach the detector mounting arrangement to a yard stick.

that shown in Photo 1. The LED on the left side of the fixture (see Photo 2) is plugged into an 8-pin DIP socket that is clamped in a small machinist's vice for stability and mobility. The detector is also mounted in an 8-pin DIP socket (at the position pointed to by the pencil in Photo 1). An ambient light shield, fabricated from black conductive foam, is shown behind the detector. The shield is placed over the detector, straddling the yard stick, when taking measurements. Photo 3 shows a close-up view of the detector mounting arrangement. Note the liberal use of hot glue, which greatly facilitated the rapid construction of the measurement apparatus.

The distance between the source and detector is selected to keep the subtended angles between the source and detector as small as is practical. However, the distance should not be made so large that the received signal has to compete with background light and/or the detector's dark current, which would compromise the accuracy of the measurement.

Before I built the apparatus shown in Photo 1, I conducted a survey of the performance I could expect from the various kinds of infrared LEDs I previously discussed. My research indicated that the RI could vary between 4 mW/sr and 160 mW/sr when driven at 100 mA, depending on the device's emission wavelength and the focusing power of the LED's integral lens structure. I then converted the RI numbers to a parameter that I call the "Radiant Photoelectron

Intensity" (RPI), which is the product of an LED's RI and the response of the BP104 detector I used to make the measurement. RPI is expressed in terms of amperes per steradian (A/sr).

$$\begin{aligned} RPI_{\min} &= 0.56 \text{ A/W} \times 4 \text{ mW/sr} \\ &= 2.24 \text{ mA/sr} \end{aligned}$$

RPI_{\min} corresponds to the corrected response of the BP104 at 880 nm.

$$\begin{aligned} RPI_{\max} &= 0.7 \text{ A/W} \times 160 \text{ mW/sr} \\ &= 112 \text{ mA/sr} \end{aligned}$$

RPI_{\max} corresponds to the corrected response of the BP104 at 950 nm.

The reason I made this conversion is because the measurement apparatus measures the photocurrent directly. I can use this figure to calculate the received power indirectly using

Solid Angle (steradians)	Distance (inches)
0.001	2.74
0.0001	8.66
0.00001	27.39
0.000001	86.61

Table 2—In order to assist in making correct measurements with the BP104, a table of various measurement distances and their corresponding solid angles is useful.

assumed values for the detector's response. Using RPI will ultimately allow me to determine the potential of an LED/detector system more accurately, as long as I use these same components in the targeted application. In addition, using RPI rather than RI avoids having to constantly make

the conversion to and from the RI units, which may be of academic interest, but are of lesser interest in the electronic domain when analyzing signal current levels.

I selected the Siemens BP104 detector for use in my RPI measurement apparatus. The typical dark current (I_{dark}) rating of this detector when reverse biased with voltages less than 10 V is about 2 nA. I arbitrarily decided that I would need at least a signal-to-noise ratio (S/N) of 5 to achieve good measurement accuracy.

$$I_{\text{signal}} = \frac{S}{N} \times I_{\text{dark}} \quad (4)$$

Now I can find the minimum solid angle that the receiver aperture must make with the source to obtain the required signal level using the expression below:

$$RSR_{\min} = \frac{I_{\text{signal}}}{RPI} \quad (5)$$

Equation 6 is then applied to determine the distance R_{\max} that can be supported between the source and detector. This optical range equation must be used with caution, however, since it does not take into account other potential sources of signal attenuation (such as rain, fog, or smoke) in the optical path and additive noise due to extraneous background light in the field of view of the detector. Long-distance links may also suffer attenuation due to beam wander caused by thermally induced refractive effects along the optical path.

$$R_{\max} = \sqrt{\frac{A}{RSR}} \quad (6)$$

A is the effective, or active, area of the receiver.

Now I have to consider what the measurement configuration would have to be in order to measure an LED with the lowest anticipated RPI (2.2 mA/sr). I use Equation 5 for this calculation. The result of this operation is shown below:

$$\begin{aligned} RSR &= \frac{5 \times 2.0 \text{ nA}}{2.2 \text{ mA/sr}} \\ &= 4.55 \text{ } \mu\text{sr} \end{aligned}$$

Since we are using a BP104, the receiver aperture or active area is

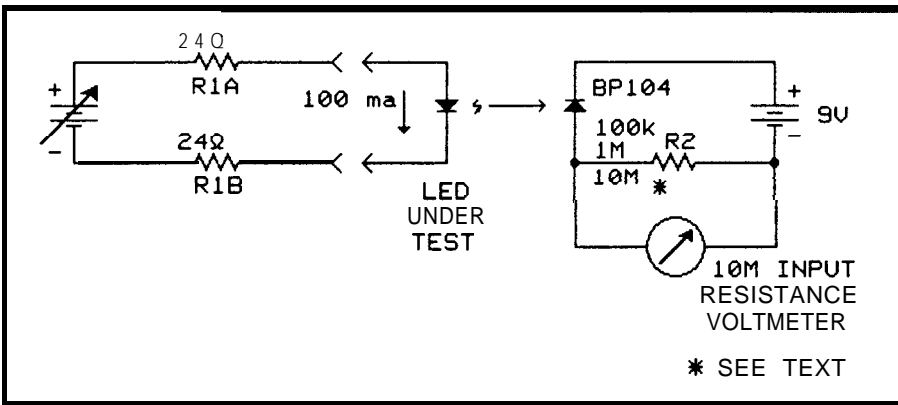


Figure 1--The measurement circuit has such a simple current source that you must make sure the LED has warmed up and the current is steady before taking any measurements. Be sure to take into account the input resistance of the meter, particularly for large R2.

(based on the physical dimensions of the device):

$$A = (2.2 \text{ mm})^2 = 4.84 \text{ mm}^2$$

From Equation 6, I determine that the distance between the BP104 detector and the LED under test must be about 41 inches. However, to make it easy to calculate RPI values, I use source-to-detector distances that create solid angles related to powers of ten. For the 2.2-mA/sr device, I use a distance corresponding to 10 μsr.

For the BP104 detector, the 10-μsr measurement range is calculated from Equation (6). This calculation is shown below:

$$R = \sqrt{\frac{A}{\text{RSR}}} = \sqrt{\frac{(2.2 \text{ mm})^2}{1 \times 10^{-5}}} = 695.7 \text{ mm}$$

For $R = 27.39''$, the total angle subtended by the detector is (from Table 1) 3.6 milliradians or 0.21°.

The light that the detector will see from a source 27.4" away is the light contained in a 10-μsr cone. To convert the signal current measured from the detector to an RPI number, use:

$$\text{RPI} = \frac{I_{\text{sig}} - I_{\text{dark}}}{Q} \quad (7)$$

In this Equation, I_{sig} is the total current with the LED turned on, I_{dark} is the detector dark current measured with the LED turned off, and Q is the solid angle of the measurement setup.

Table 2 lists various measurement distances and their corresponding solid angles when used with a BP104 detector or another device with an equal active area.

The schematic of the RI/RPI measurement apparatus is shown in Figure 1. The LED drive circuit consists of a fixed resistor in series with the LED under test and a variable voltage power supply. I drive the LEDs

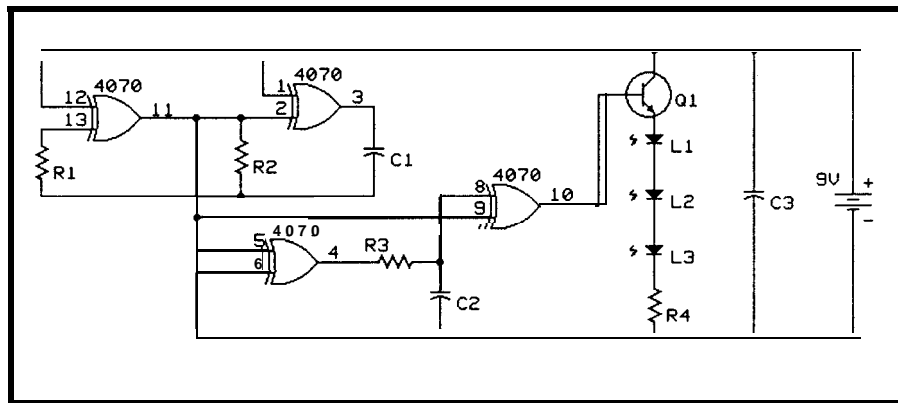
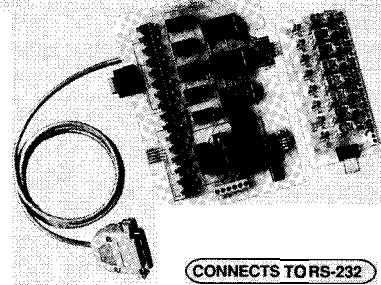


Figure 2-A low-duty-factor, high-speed pulse transmitter may be built with off-the-shelf parts. The loop consisting of Q1, L1-3, R4, and C3 must be kept short for good pulse fidelity.

RELAY INTERFACE

PROVIDES SOFTWARE CONTROL OF RELAYS

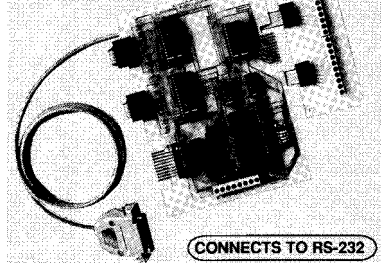


CONNECTS TO RS-232

AR-16 RELAY INTERFACE.....\$69.95
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 relay cards and relays are stocked. Call for more info. RS-422 available (distances to 4,000 feet). PS-4 port selector may be used to control satellite AR-16 interfaces. (up to 16; 8 relays)
RD-8 REED RELAY CARD (8 relays, 10 VA).....\$49.95
RH-8 RELAY CARD (10 amp SPDT 277 VAC).....\$69.95
EX-16 RELAY EXPANSION CARD (16 channel).....\$59.95

ANALOG TO DIGITAL

8.10 & 12 BIT RESOLUTION



CONNECTS TO RS-232

ADC-16 A/D CONVERTER (16 channel, 8 bit).....\$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 conversion. RS-422 available. PS-4 port selector may be used to connect satellite ADC-16 interfaces (up to 4,096 analog inputs/16,384 status inputs and 14,336 relays). Call for info on 10 & 12 bit converters. (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 including keypads and binary coded outputs. Provides 32 status inputs (optoisolators sold separately).
TE-6 TEMPERATURE INPUT CONVERSION.....\$49.95
Includes 8 temperature sensors & terminal block. Temperature range is minus 40 to 145 degrees F.
PS-4 FORT SELECTOR (4 channels RS-422).....\$79.95
Converts an RS-232 port into 4 selectable RS-422 ports. TOUCH TONE DECODER and other serial interfacing products available. Call for free information packet

- 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. Detailed technical reference manual is also included.
- HIGH RELIABILITY...engineered for continuous 24 hour industrial applications. All ICs socketed.
- Use with IBM and compatibles, Tandy, Apple, Mac 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. Technical Information (614) 464-4470.

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

International & Domestic FAX (614) 464-9656
Use for information, technical support & orders
ELECTRONIC ENERGY CONTROL, INC.
380 South Fifth Street, Suite 604
Columbus, Ohio 43215

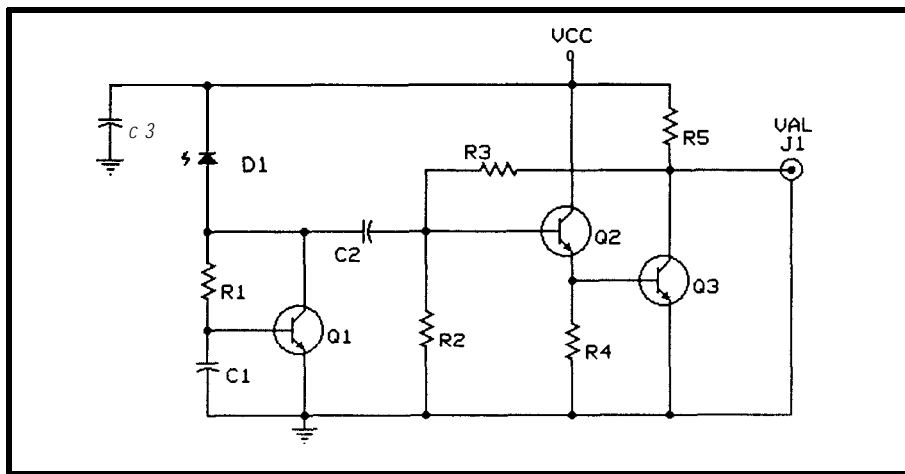


Figure 3—The high-speed receiver uses a Siemens BP 104 at its core. The BP 104 incorporates a built-in absorption filter to block out a large portion of the visible region of the optical spectrum.

at a current level of about 100 mA for the measurement, which is consistent with the levels used by the LED manufacturers. For R1 to have a value of 48 ohms [two 24-ohm, 1/4-watt resistors in series), you will need a power supply capable of supplying between 6 and 8 volts.

Figure 1 also shows the circuit used for the receiver of the measurement system. The detector, D1 (BP104), is reverse biased by a 9-volt battery to operate the detector in the reversed-biased, or photoconductive mode. Resistor R2 serves a dual role in the circuit. It limits the detector current, protecting the diode from damage in case the bias voltage is applied incorrectly. In addition, R2 develops a signal voltage that is proportional to the photocurrent generated when the detector is exposed to light energy.

The value of R2 should be selected to restrict the signal voltage to less than 1 volt for a 9 volt bias supply, or roughly 10% of the reverse-bias potential to assure photodiode linearity. The value of R2 is best determined experimentally. However, I have found that having just three values available for R2 (100k ohms, 1.0M ohms, and 10M ohms) is sufficient to cover all LEDs when making measurements at distances that correspond to solid angles between 10 and 100 micro-

steradians. R2's actual value in the circuit is the parallel equivalent of the resistance of R2 and the input resistance of the voltmeter used to make the measurement.

SOME SIMPLE LINK DESIGN EXAMPLES

Figure 2 is a schematic of a transmitter you can build from readily available parts to explore the capability of free-space transmission links. A photograph of the transmitter is shown in Photo 4. The output of this transmitter is a continuous series of 5- μ s pulses occurring at a 1-kHz rate. The transmitter may be used to drive one or three series-connected LEDs. When used with three Siemens SFH484-2

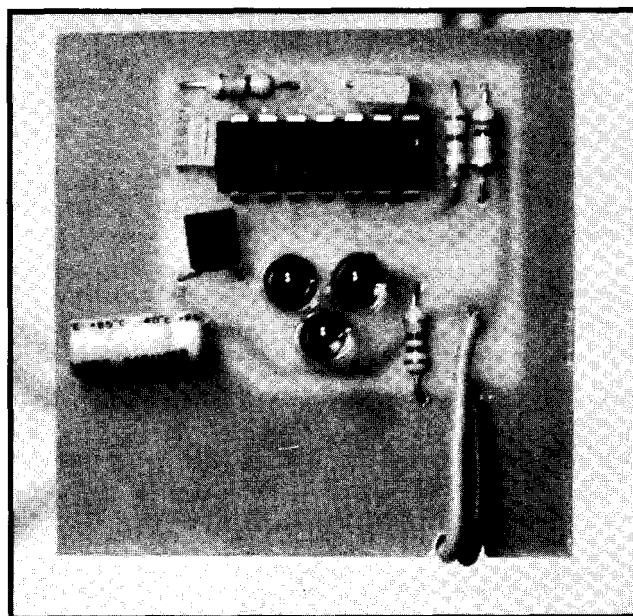


Photo 4—The transmitter lays out neatly on a small, single-sided PC board. Note the cluster of three LEDs in the center. Power is supplied by a single 9-V battery.

LEDs and driven at a peak current of 200 mA, the RPI of the transmitter is approximately 0.75 A/sr. Alternatively, the transmitter may use one SFH484-2 LED and a lens to increase the effective RI [and RPI) of the LED. When used with an Edmund Scientific double-convex lens (50-mm diameter, 150-mm focal length, P/N A37,794 @ \$4.25 each) I have obtained RPIs in the range from 3 to 6.2 A/sr from the same 200-mA peak drive current. The penalty one must pay for the increased RI obtained with the lens is that greater precision will be required to align the transmitter to the receiver. This is due to the fact that the beam divergence of the transmitter drops from about 10° (without external optics), to less than 2° when used with the Edmund Scientific lens.

Figure 3 shows the schematic of a high-speed receiver you can build to observe the pulses from the transmitter of Photo 4. The receiver is shown in Photo 5. The detector is a Siemens BP104 that incorporates a built-in absorption filter to block out a large portion of the visible region of the optical spectrum. The active area of the detector is 2.2 mm by 2.2 mm, and may be used without a lens to observe the three-LED (also without a lens) transmitter at distances of about 20 feet. Photo 6 shows an oscilloscope trace of the output of the receiver when illuminated by the three-LED transmitter from 22 feet. Coupling the detector to a simple lens like that shown in Photo 7 will extend the range to more than 100 feet.

To examine what transmission distances are practical for small optical systems, let's make some calculations using some of the source and detector configurations I mentioned above. For these calculations, I'll conservatively require that the received signal level at the output of the receiver be 50 mV peak-to-peak. For the receiver of Figure 3, 50 mV at the receiver output corresponds to a photocurrent

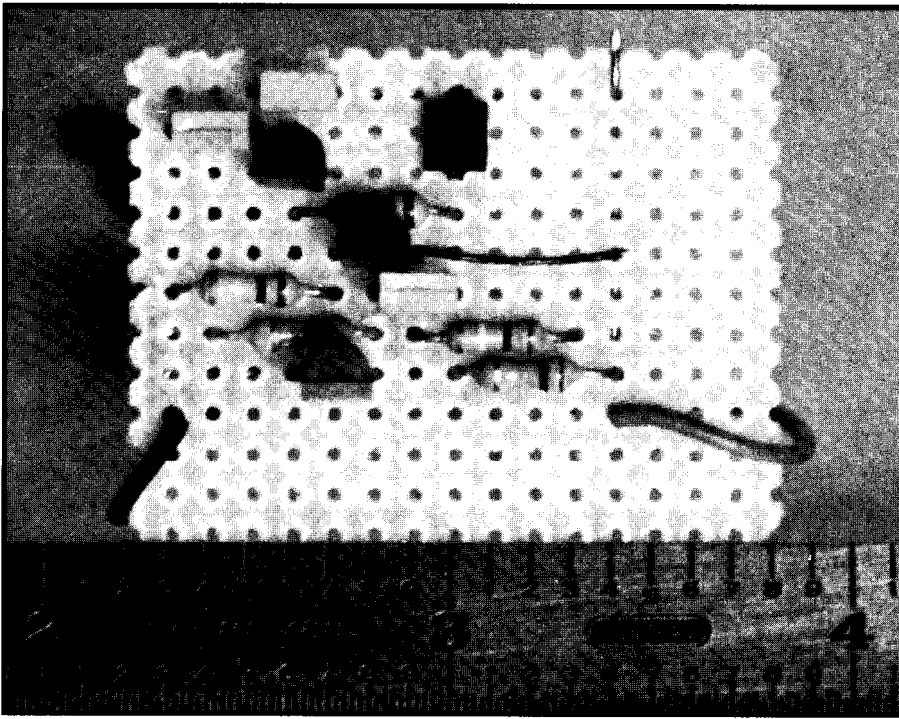


Photo 5-The receiver board is about the same size as the transmitter and, again, could be laid out on a single-sided board. Also like the transmitter, the receiver uses a single 9-V battery.

of 25 nA. Using Equation 5, we determine the minimum solid angles needed for two LED and detector

arrangements. The first configuration uses three LEDs and an unaided BP104 detector. The second uses the 2"

diameter Edmund lenses on both the transmitter and receiver.

$$\begin{aligned} \text{RSR1 (unaided, 3 LED)} &= \frac{25 \text{ nA}}{0.75 \text{ A/sr}} \\ &= 33 \text{ nsr} \end{aligned}$$

$$\begin{aligned} \text{RSR2 (single LED and lens)} &= \frac{25 \text{ nA}}{6.2 \text{ A/sr}} \\ &= 4 \text{ nsr} \end{aligned}$$

Now I can compute the distances that these transmitters are capable of by using Equation 6, while being careful to use consistent units.

$$\begin{aligned} R1_{\text{max}} &= \sqrt{\frac{A}{\text{RSR1}}} \\ &= 40 \text{ feet} \end{aligned}$$

$$\begin{aligned} R2_{\text{max}} &= \sqrt{\frac{A}{\text{RSR2}}} \\ &= 2335 \text{ feet} \end{aligned}$$

Obtaining long transmission distances with line-of-sight systems is a nontrivial task requiring a high degree of mechanical precision. In long-distance applications, you may need to use telescopes that are

NOW SHIPPING!

The New
HCS II
Version 2.0

Display Time, Date & Variables on LCD-Link or Printer

Data Logging

16-bit, 4-function Integer Math

More Powerful Comparisons

Test Month, Day, Year & Day of Week

Faster

More Variables, Labels & Timers

Plus Lots More...

An upgrade consisting of a new EPROM, disk, and manual costs just \$60. Order yours today!

Circuit Cellar HCS

Circuit Cellar, Inc.
4 Park Street, Suite 12
Vernon, CT 06066
Tel: (203) 875-2751
Fax: (203) 872-2204

#112

μLAN

The 9-Bit Solution is the Answer to your Embedded Network Needs

The Cimetrics Technology 9-Bit Solution is a complete microcontroller network (μLAN) that supports the 8051, 68HC11, 80186, and many other popular processors. The 9-Bit Solution takes full advantage of multiprocessor modes built into microcontroller serial ports. Our flexible software and hardware allow developers to create powerful, yet inexpensive master/slave multidrop embedded controller networks.

- Up to 250 nodes
- 16-bit CRC error checking with sequence numbers
- Low network overhead and low resource requirements
- RS-485 interface card for the PC
- Complete source code included
- Comprehensive documentation

PC/XT/AT
8051
8096
80C186EB/EC
68HC11
68HC16
Z180

CIMETRICS TECHNOLOGY

120 West State Street, Ithaca, NY 14850
TEL: (607) 273-5715 FAX: (607) 273-5712

#113

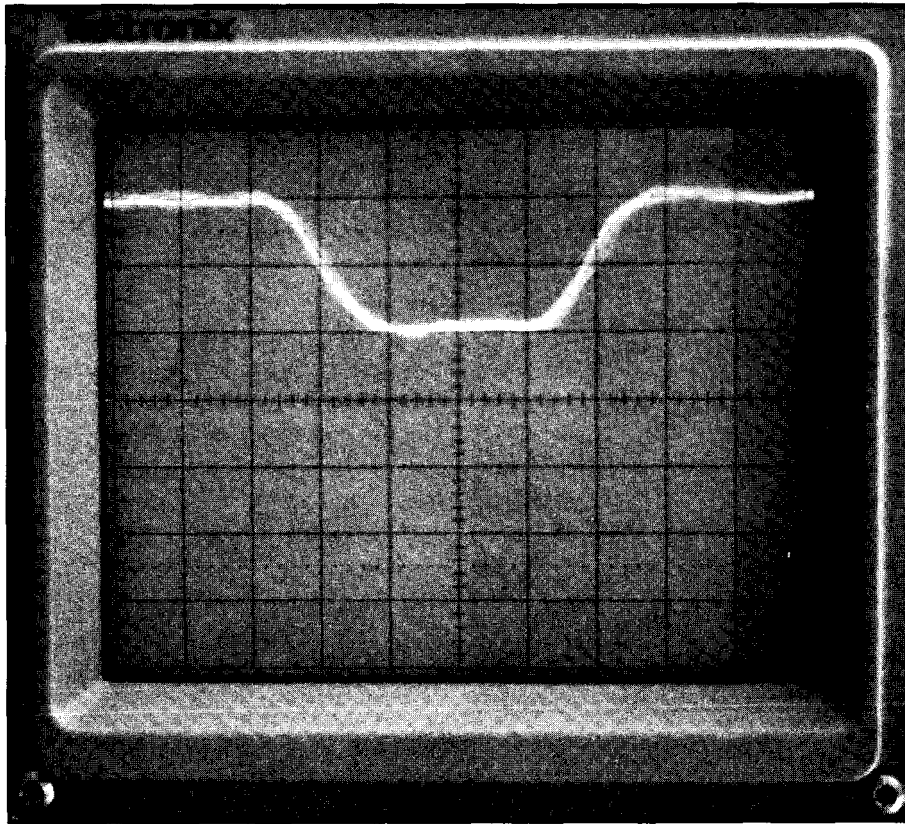


Photo 6—When illuminated by the three-LED transmitter from 22 feet, the receiver puts out a very clean pulse.

mounted on, and coaligned with, the terminal equipment in order to sight the terminal equipment pairs. In these applications, the terminal equipment will have to be mounted to a platform that has two degrees of angular adjustment freedom. Optimum performance also requires accurate placement of the source and detector relative to the optical axis and focal plane of the lens system.

APPLICATIONS

What are some practical free-space optical link designs based on the design concepts described here? Well, would you consider a simple break-beam sensor that you can use to monitor the perimeter around your home? Or how about a free-space RS-232 data link between computers? Or maybe even an optical sensor that can be used as a position sensor in a

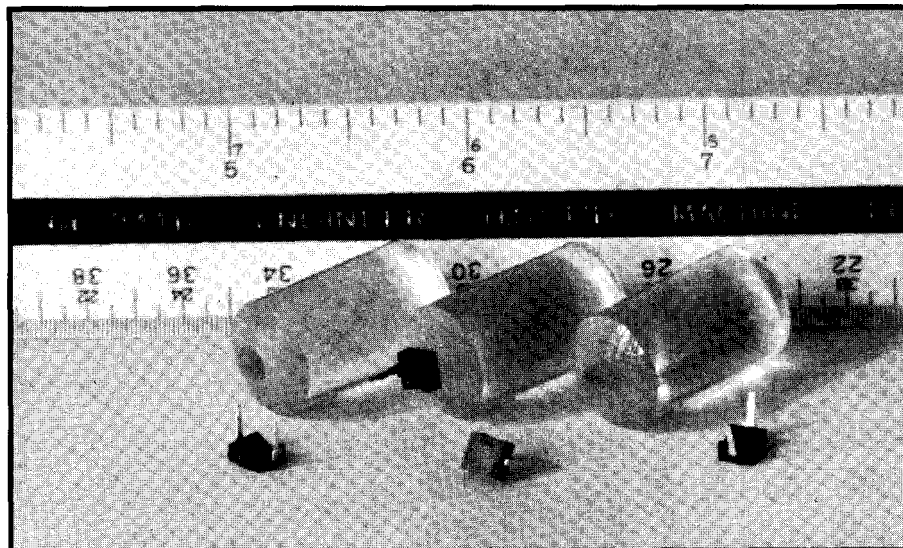


Photo 7—Coupling the detector to a simple lens will extend the range of the transmitter/receiver pair to 100 feet.

motion control servo system? These are but a few of the many ways that free space optical systems can be put to good use. I have given you the tools to design reliable links, and now it is up to you to apply them in your particular environment. □

Larry Foltzer has over 19 years experience in optical fiber communication technology. He is head of the Lightwave Development Group at DSC Commuincations, Optilink Access Products Division, where he is responsible for the development of SONET optical interfaces for the division's products.

SOURCES

Most of the components required to build the transmitter and receiver described in this article can be found at local stores or through the major mail order distributors. For lenses, contact:

Edmund Scientific Co.
101 E. Gloucester Pike
Barrington, NJ 08007-1380
(609) 573-6250
Fax: (609) 573-6295

The following components are available from the author:

SFH484-2	\$1.50 each
BP104	\$2.00 each
Lens	\$5.00 each
Transmitter PCB [no LEDs]	\$8.00 each
Receiver kit with PCB (no BP-104)	\$8.50 each

U.S. residents include \$5.00 for shipping and handling. California residents include 7.5% sales tax. Please allow 3 weeks for delivery. Send check or money order to:

L. Foltzer
P.O. Box 488
Occidental, CA 95465

IRS

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

Embedded Control Using ACCESS.bus

Macintosh users have long been used to plugging multiple peripherals together with a single kind of cabling system using Apple's Desktop Bus. Now, ACCESS.bus promises to clean up the cable clutter for PCs, too.

FEATURE ARTICLE

David Wyland

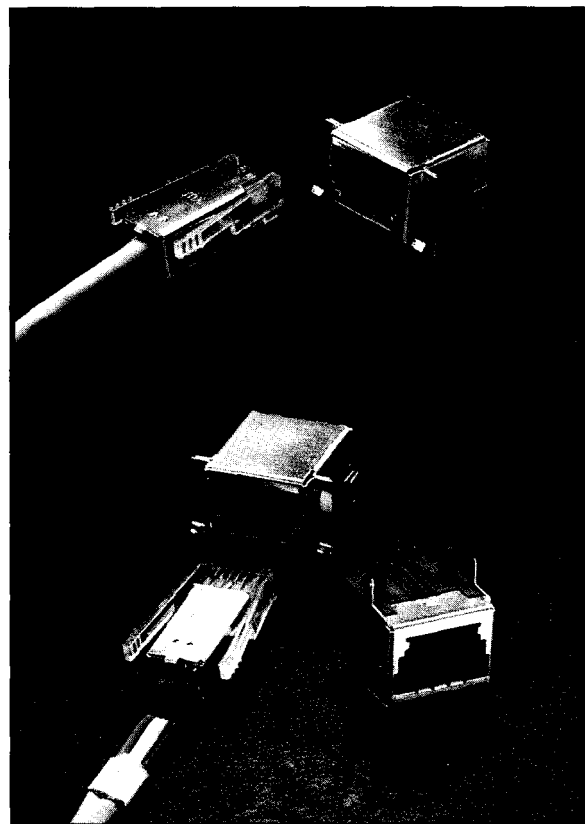
Wouldn't you love to have a perfect serial interface for general-purpose instrumentation and control that works every time you plug it together? One with all the aggravations associated with RS-232 eliminated? Could you appreciate an interface ten times as fast as 9600 bps, but automatically slows down if it needs to? Could you find some use for the real estate saved by using TTL signal levels instead of ± 12 volts (so you don't need space or power for level converters)? Would you like to simplify your system interconnects by using a multidrop bus connection! Would you like a cabling solution where you can have one kind of cable for all situations? Would you object to never wiring a null modem again, or never again pondering if CTS/RTS/DTR/DSR are wired to the wrong pins? Would you like this interface to be an openly defined industry standard, supported by lots of devices and by several large companies?

This wonderful, utopian bus actually exists! The ACCESS.bus meets the standards I outlined in my "wish list." It is a serial bus, but there are no "settings" to concern yourself with. The source (Master) sends data to the destination (Slave) in 8-bit bytes with an acknowledge at the end

of each byte. The ACCESS.bus transfers data at up to 100 kbps (400 kbps in the future). It features automatic slowdown by either the sending or receiving device as required. It uses TTL signal levels (0 and +5 volts) for data transmission. It is a multidrop bus using modified modular-phone cable with a simple 4-wire connection. ACCESS.bus devices can also be *hot plugged*, meaning that it lets you safely add or remove devices from the ACCESS.bus while it is running. It doesn't even have addresses to set or DIP switches to fiddle with! The CPU automatically assigns addresses at reset or when a device is powered up.

Best of all, the ACCESS.bus is on its way to being an industry standard for PCs. The ACCESS.bus is a result of the work by DEC and Philips/Signetics to create a desktop bus for the PC. The goal of the desktop bus is to simplify the cabling to keyboards, mice, graphical tablets, lightpens, and so forth.

ACCESS.bus is a software overlay on the ubiquitous I²C bus standard, so it can easily leverage off the momentum already established for that standard. The I²C bus has over 150 chips available that support it. There



are at least eight 8051 microcontroller chip derivatives available with I²C bus UARTs on them. There is also an I²C UART—the PCD8584—that you can lash to the processor of your choice.

ACCESS.BUS BASICS

The ACCESS.bus is a serial bus that uses a four-wire interconnection standard. The cable called out in the standard is a modified modular-phone cable (see Figure 1) and is shielded to minimize RFI. The cable contains two signals—serial clock and serial data. The other two wires are power and ground. The power line supplies +5 volts at up to 1 amp for powering small devices such as mice, keyboards, and so forth, directly from the cable.

The ACCESS.bus is a half-duplex bus and uses a multidrop protocol where each device on the bus has a unique address. Up to 124 devices can share the bus, with addresses 00h, 50h, and 6Eh reserved. The maximum bus capacitance of 400 pF restricts the number of devices and constrains the cable length to 8 meters. This capacitance limit ensures a rise time of less than 1 μs when termination resistances of less than 2.5k ohms are used. You connect devices to the bus in parallel.

The Serial Data line (SDA) carries data transmissions which are clocked into the receiving device by the Serial Clock line (SCL). If the receiving device needs more time, it holds down the SCL line until it is ready for more data. The specification limits this hold time to 2 ms. The sending device holds the data bit unchanged while the clock is high except for start and stop conditions. The sending device initiates a message by changing the

SDA line from high to low while SCL is high, and terminates the message by taking the SDA line low to high while SCL is high.

Since any device can send a message to the CPU at any time, collisions are possible. This happens when multiple devices see the bus in a “not busy” state and start sending a message simultaneously. The devices do collision detection to sense such events. Each one checks to see that the data on the bus is the same as the data it is putting on the bus. In case of a collision, the two devices will eventually try to send different data bits. Since the bus is open drain, the one sending a low level wins, so the device trying to send a high level detects a bus error, stops, and retries its transmission later. Plugging a new device on the bus might corrupt a message in progress, but the same collision detection mechanism will also sense such a corruption. Collision detection coupled with the open-drain nature of the bus are the key features of the standard that allow hot plugging.

The ACCESS.bus moves data in 8-bit bytes similar to RS-232. However, byte transfers over the bus use the I²C bus protocol. I²C defines byte transfers as follows: The transmitter sends the most-significant bit first, followed by an acknowledge bit supplied by the receiver (see Figure 2). Messages begin with a Start condition and end with a Stop condition. This differs from RS-232 that has Start and Stop bits for each byte. The ACCESS.bus provides a standard that organizes groups of bytes into messages, defines how to assign addresses to slave devices, and defines all messages as writes—from CPU to slave or from slave to CPU.

The ACCESS.bus messages vary in length from 1 to 127 bytes. Figure 3 shows the message protocol. Each message consists of a destination address, a source address, a byte count, a control/data flag bit, the message with 1-127 bytes of data, and a checksum. The I²C UART transfers each byte automatically, and each byte receives an acknowledge from the destination device. The I²C UART hardware handles message initiation, byte acknowledge, speed control, and message termination.

If the Control/Data flag in the byte count field is a 1, the message is a command. The operation code is the first byte immediately after the byte count. Opcodes in the range of 00h through 7Fh are available for general use. For instance, I use opcode 10h as a Read Request command to an I/O device. The ACCESS.bus reserves opcodes from 80h through FFh for control functions. These control functions and their reserved codes include: Reset (F0h), Identification Request (F1h), Assign Address (F2h), Attention (E0h), Identification Reply (E1h), and Interface Error (E3h).

AN ACCESS BUS SYSTEM

ACCESS.bus systems are simple to design and work with. You have only two signal wires and one power wire to deal with. The I²C bus is an open-drain pull-down bus with a single pair of pull-up termination resistors, which are typically installed at the CPU end. You determine the resistor value from the maximum-rated drive current (3 mA) for the I²C drivers. The resistor value also determines the data transmission speed since the resistor current charges the line capacitance. A 2k-ohm resistor works well, as shown in Figure 4. An optional 100-ohm resistor in series with each driver helps kill noise. The +5 V is provided through a fuse or a current-limited regulator. A current-limited regulator has the advantage of automatic recovery with no fuse to replace after a failure. The choice of, resistor size and fuse method is about all the hardware design you must do.

A good way to explore the ACCESS.bus is to use it in a system.

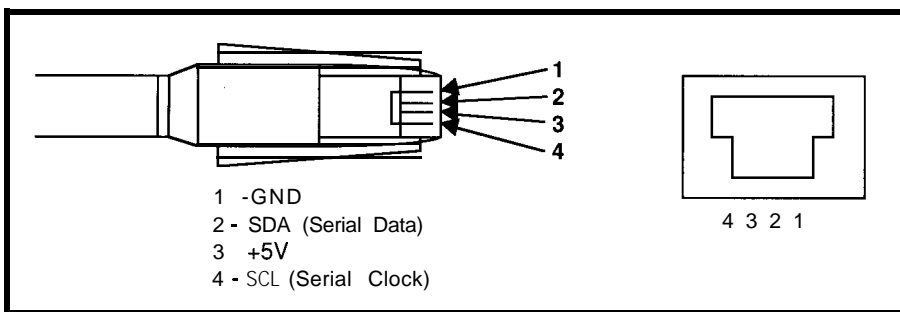
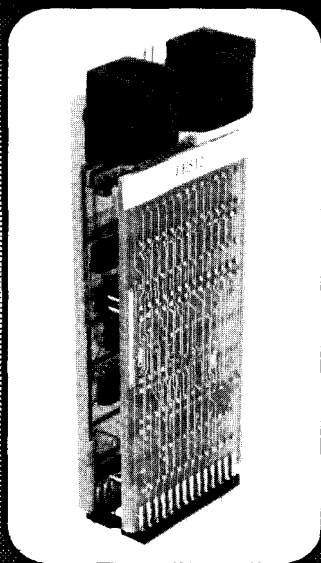


Figure 1—Based on PC, ACCESS.bus uses a simple four-wire interface that provides not only a data channel, but also power to peripherals. The proposed modular connector locks in place and eliminates orientation confusion.

EPROM EMULATORS



\$129

AND UP

- Downloads through standard PC compatible printer port
- Includes RJ-11/12 adaptor and 7' modular cable
- Generates RESET and /RESET
- Accepts Binary, Intel HEX, Intel Extended HEX and 'S' files
- Unique Vertical design minimizes mechanical interference and eliminates the possibility of noise cross-talk or transmission-line effects from the use of a cable
- Multiple devices (except EE256) can be daisy-chained for 16/32 bit or multi-bank target systems

EE256 (2764-27256)	\$129
EE512 (2764-27512)	\$159
EE1M (2764-27010)	\$199

**MORE ADVANCED MODELS
AVAILABLE (TO 4Mb)!**

**10 day MONEY-BACK GUARANTEE
90 day repair/replace warranty**

Technical Solutions

PO BOX 462101
Garland, TX 75046-2101

Call or FAX for ENGINEERING SPECS.
Voice or FAX: (214) 272-9392

Listing 1-The main 80C652 CPU message transfer code handles the master side of the ACCESS.bus interface.

:Subroutine to send a message from CPU to I/O device

SEND:

Enter with slave address, length, pointer to bytes of data
Set to Master mode, send Start bit by writing 68h to Control reg
Check for valid start
Send I/O address using SBYTE
Send CPU address = 6Eh using SBYTE
Send length using SBYTE
Send data bytes using SBYTE
Calc checksum = XOR of all bytes from slave address through last data byte
Send checksum using SBYTE
Send stop bit
Clear from Master mode, set to slave mode (default):
Set Assert Acknowledge by writing 04h to Control register
Exit

:Subroutine to send one byte

SBYTE:

Write byte to data register
Wait for acknowledge from Status reg.: can be interrupt response
Exit

:Error routines

ARB:

On Send arbitration error, send stop bit, clear from master mode and restart at SEND.

NAK:

On Send not acknowledge, send stop bit, clear from master mode and restart at SEND.

TIMO:

On Timeout, exit with error code

:Subroutine to read a byte from I/O device to CPU

READ:

Set slave address
Set message length = 81h = 1 byte with command bit = 1
Set command byte = 10h (Read Request)
Call SEND to send Read Request message
Call RECV to receive message
Exit

:Subroutine to receive a message from I/O device to CPU

RECV:

Note: CPU in slave mode as default
Receive CPU Address = 6Eh
Receive master (sender) address
Receive message length
Receive data bytes
Receive checksum but don't acknowledge yet
Verify checksum
Send acknowledge if checksum OK, not if not OK
Receive STOP condition
Exit

:Subroutine to receive one byte

RBYTE:

Wait for Interrupt
Exit; Return byte on interrupt, set interrupt

Listing 2—The message transfer code for the 87C751 I/O Controller handles the remote data acquisition and control.

```
;Subroutine to send a message from I/O device to CPU
SEND:
Enter with slave address, length, pointer to bytes of data
Set Master mode: Write 50h to Config = req bus master, 100 kbps
Send I/O address using SADDR
Send CPU address = 6Eh using SBYTE
Send length using SBYTE
Send data bytes using SBYTE
Calc. checksum = XOR of bytes from slave addr through last byte
Send checksum using SBYTE
Send stop bit
Set to Slave mode: Write 90h to Config. Reg. (default mode)
Exit
```

```
;Subroutine to request bus mastership and send address
SADDR:
Wait for ATN bit in Control Register, go to SAERR if error
Go to send byte routine
```

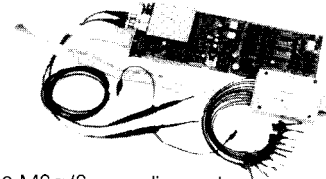
```
SBYTE:
Set bit counter to 8
Write MSB to Data Register
Rotate left for next bit
Wait for bit sent: wait for ATN in Control Register
Decrement bit count and loop back to SBYTE+1 if not zero
Set to receive mode to receive ack: Send A0h to Control Reg
Wait for ATN
Exit; Return acknowledge status
```

```
;Error routines
SAERR:
On Send arbitration error, send stop bit, clear from master
mode and restart at SEND.
NAK:
On Send not acknowledge, send stop bit, clear from mastermode
and restart at SEND.
TIMO:
On Timeout, exit with error code
```

```
;Subroutine to receive a message from CPU to I/O Device
RECV:
Enter in Slave mode (This is the default mode)
Receive slave address: call RDACK
Receive CPU Address = 6Eh
Receive message length
Receive data bytes
Receive checksum but don't acknowledge yet
Verify checksum
Send ack if checksum OK and slave addr compares; otherwise not
Receive STOP condition
Check for Read Req. command = command with Operation code 10h
If Read Req.t, get byte of input data from Port 1 and call SEND
Exit
```

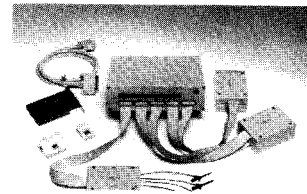
```
;Subroutine to receive one byte
RBYTE:
Set bit counter to 7, clear accumulator
Wait for bit
Get bit, clear ATN
Rotate to LSB
Decrement bit count and loop back to RBYTE+1
Wait for last bit
Get bit, don't clear ATN
Rotate to LSB
Send acknowledge
Wait for ATN
Check for errors
Exit
```

200MSa/s Digital Oscilloscope



- 200 MSa/s sampling rate
 - PC-BASED INSTRUMENT
 - 2 Analog channels
 - 8 Digital channels (8ch. logic analyzer)
 - 125MHz Single shot Bandwidth
 - 4K samples/channel (analog & digital)
- \$1599 -DSO-28100** Price is Complete
\$1999 -DSO-28200 Pods and Software included

400 MHz Logic Analyzer




- up to 128 channels
- up to 400 MHz
- 16K samples/channel
- Variable threshold
- 8 External clocks
- 16 level triggering

\$799 - LA12100 (100 MHz, 24 Ch) Price is Complete
\$1299 - LA32200 (200 MHz, 32 Ch) Pods and Software
\$1899 - LA32400 (400 MHz, 32 Ch) included

Universal Programmer

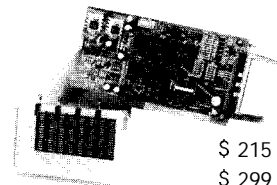
PAL
GAL
EPROM
EEPROM
FLASH
MICRO



\$475

5ns PALs
4 MEG EPROM (8 & 16 bit)
22V10 & 26CV12 GALs
Free software updates on BBS

Gang Programmer



\$ 215 (4 Socket)
\$ 299 (8 Socket)

Up to 1 MEG EPROMS

Call (201) 808-8990
Link Computer Graphics, Inc.
369 Passaic Ave Suite 100, Fairfield NJ 07004 fax 808-8786

#116

I'll make a simple system with one CPU and three I/O ports. Each port has 8 bits of digital input and 8 bits of digital output. Figure 4 shows a block diagram of my system. I use four microcontroller chips: an 83C652 as the primary CPU and three 87C751s as I/O device controllers. These chips are variants of 8051 microcontroller chips with I²C UARTs. I use the microcontrollers to implement the ACCESS.bus communication protocol over their I²C connections. The 83C652 is a 40-pin device with external EPROM and RAM. These features make it useful as the central CPU. The 87C751s are 20-pin, 300-mil components with internal EPROMs, the I²C interface, and two 8-bit bidirectional I/O ports. These devices will serve as the I/O device controllers in my prototype.

The three 87C751s each provide peripheral device control. The I²C interface uses two of the three bits on Port 0; Port 1 receives the 8 bits of data input; and Port 3 supplies the 8 bits of data output. Note that both ports are bidirectional. If you need more data I/O, you can use Port 1 as a bidirectional 8-bit data bus, and use Port 3 as an 8-bit address bus. This allows up to 256

bytes of data I/O from this single device.

As you can see, this is a simple system in terms of hardware design. Unlike RS-232, there are no level converters, no baud rate configuration switches, and the connector pinout is simple and fixed for all devices. The bus connection method simplifies cabling and means there is only one I²C UART at the host end rather than one for each I/O device. This also means you can add I/O devices to the system without adding hardware to the CPU.

The bus connection method is possible because messages can be more than one byte long. In ACCESS.bus, there is a multibyte message between Start and Stop codes that contains the addresses of the source and destination of the message. This allows several devices to share the same bus.

The hardware design of our ACCESS.bus system is simple: the software makes it work. The software converts data into ACCESS.bus messages for transfers between the CPU and I/O devices, and sets up the addresses of the devices when they power up.

ACCESS.BUS PROGRAMMING

There are two areas to the software design for an ACCESS.bus system: the set of routines that send data to and receive data from the remote devices, and the initialization code that assigns the soft addresses to the devices when the system powers up. Initialization sequences are also required when a device is plugged onto a bus that is already running. Let's look at the operating routines first. The CPU sends messages to an I/O device, and I/O devices send messages to the CPU. The message protocol is the same in both of these cases. Any I/O device can send a message to the CPU at any time. For example, a keyboard sends key data to the CPU whenever you press a key. The sending device is always the master and the receiving device is always the slave.

My 4-MPU system can write bytes from the CPU to the output port of a selected I/O device, and it can read bytes from the input port of a selected I/O device. To write a byte from the CPU to another device, the CPU must send a message with the appropriate address. In this case, the CPU sends a data message to the desired I/O device address. The I/O device receives this data and writes it to its output port.

For the CPU to read a byte, it sends a command message called Read Request to the I/O device, and it responds with a data message containing the data. The Read Request command is a single-byte command message which is the opcode. I use a user-definable opcode (10h) for the Read Request.

CPU PROGRAM

The I²C controls in the 83C652 and the 87C751 are different. The 83C652 contains a full I²C UART. The 87C751 has a less-capable interface, so the program does the serialize, de-serialize, and timing-control functions.

Listing 1 shows the 83C652 routines for message transfer. The CPU writes a byte to the output port of an I/O device by sending a Read Request command (10h) to the I/O device. In turn, the I/O device responds by sending a one-byte message with data to the CPU.

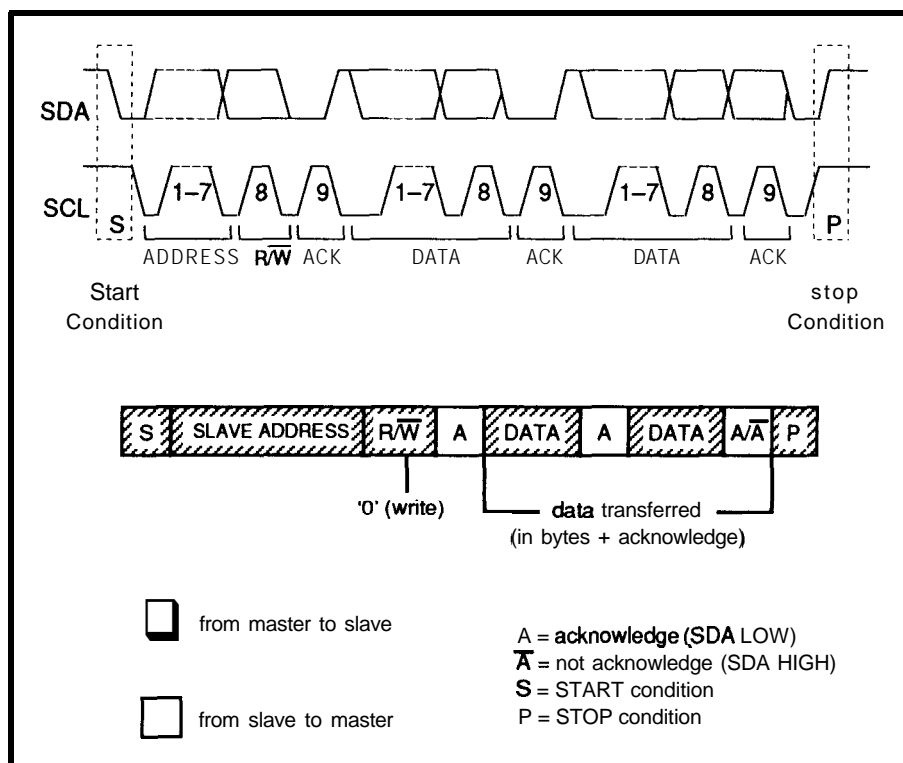


Figure 2—In I²C, serial data (SDA) is sampled when the serial clock (SCL) is high. The basic packet of information consists of a destination address, a R/W flag, data, and start/stop framing. A simple ACK/NACK status is sent back by the destination device.

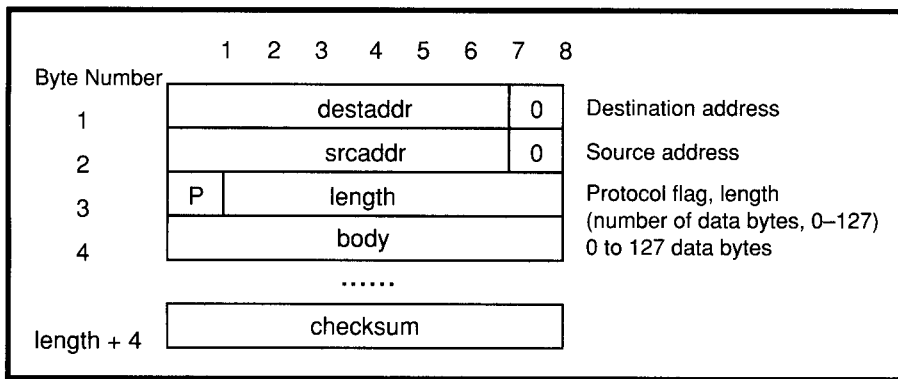


Figure 3—The standard ACCESS.bus message packet is based on I²C, but includes additional information. The packet consists of a destination address, source address, length, data bytes, and a simple checksum.

The 83C652 I²C interface has four 8-bit registers: Control, Status, Slave Address, and Data. The Control register controls I²C communication. It enables the I²C UART, sets the maximum bit rate, sends Start and Stop states, sends byte acknowledge messages, and enables an 8051 interrupt for status changes. The Status register provides a 5-bit status code indicating function completion or error. The Slave Address register provides the response address for messages that are sent to the CPU. The protocol sets the CPU Slave address to 50h. The Data register sends and receives 8-bit bytes of data.

I/O DEVICE PROGRAM

Listing 2 shows the message transfer program for the 87C751. When the CPU writes a byte to the I/O device, it sends a one-byte data message. The I/O device program receives this message from the CPU and writes it to Port 3. When the I/O device receives the Read Request from the CPU, it reads the data on the input port and sends a message to the CPU.

The 87C751 I²C interface has four registers: Configuration, Control, Status, and Data. The Configuration register defines whether the I²C interface is in Master or Slave mode. It also holds the timer value for the maximum baud rate. The Control register sends the Start and Stop conditions to the bus and indicates attention and data ready conditions. The Status register indicates the status of Start and Stop conditions. The Data register provides one bit of storage.

The 87C751 program provides data serialize and deserialize functions, slave address recognition, and individual bit timing. These routines complicate the program but reduces the cost of the silicon in the system. It is also practical since the 87C751 is used as a simple I/O controller and the program hasn't much to do except tend the I²C interface and pass the data to the digital I/O port the '751 serves.

SOFTENING UP THE ADDRESSES

An address is assigned to each I/O device by the CPU. The I/O device stores the address and uses it when

We're Small, We're Powerful, And We're Cheaper.

In fact, you'll get the best product for about half the price. If you're interested in getting the most out of your project, put the most into it. For the least amount of money. Call us today for complete data sheets, CPU options, prices and availability.

MMT-EXP Board, \$78.00

LCD Display Interface, Keypad Interface, Parallel Printer Interface, 4x6 Dimension (EURO Card)

Options: 12 Bid D/A, 12 Bit A/D, Serial I/O Interface, 8 Channel Analog Multiplexor, LCD Display, Keypad.
Ideal for any MMT Single Board Computer.

MMT-196 6MHz, \$199.00

2 Serial Ports, 48 Bits I/O, 64K (RAM/ROM), A/D C, 5 Counter/ Timers, Intel 80196 Processor, Watchdog Timer, 4x6 Dimensions (EURO Card).

MMT-HC 11 2MHz, \$178.00.

2 Serial Ports, 40 Bits I/O, 64K (RAM/ROM), A/D C, 4 Counter/Timers, Motorola 68HC11 Processor, Watchdog Timer, 4x6 Dimensions (EURO Card).

MMT-Z 180 6MHz, \$159.00.

2 Serial Ports, 40 Bits I/O, 1M (RAM/ROM), A/D C option, 5 Counter/ Timers, Zilog Z180 Processor, Watchdog Timer, 4x4 Dimension.

MMT-188EB 16MHz, \$239.00.

2 Serial Ports, 40 Bits I/O, 1M (RAM/ROM), A/D C option, 3 Counter/ Timers, Intel 80188EB Processor, Watchdog Timer, 4x4 Dimension.

Custom Work Welcome. Call or fax for complete data sheets

2308 East Sixth Street, Brookings,
SD 57006, Phone (605) 697-8521,
Fax (605) 697-8109



WE'RE SMALL BUT WE'RE POWERFUL

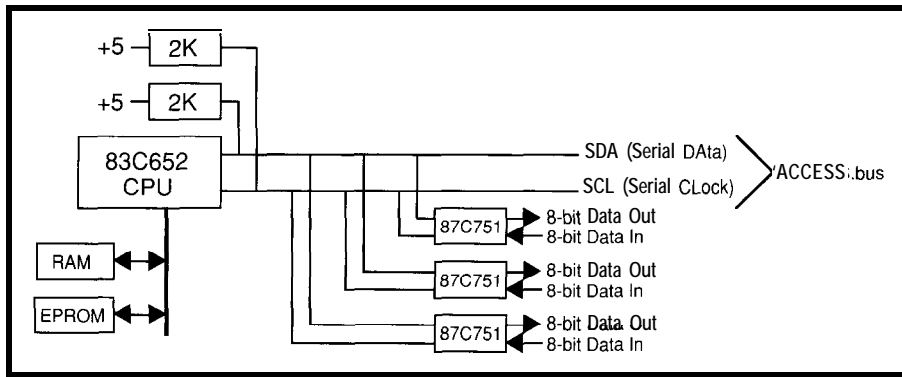


Figure 4—Many 87C751-based I/O ports may be added to the 83C652-based ACCESS.bus system with just a single pair of wires.

communicating with the CPU. When the CPU sends a message to the I/O device, this address is part of the message. If the destination address matches the address stored in its register, the I/O device accepts the message; otherwise it ignores the message. When the I/O device sends a message to the CPU, the message includes the source address so the CPU can tell who sent it. The I/O device address is also called the *slave address*. The CPU has a fixed slave address of 50H. The CPU assigns each device on the bus a unique I/O address as part of the initialization sequence.

The CPU also assigns an I/O address to each device plugged into the bus while the bus is running. The device plugged into the bus notifies the host of its existence, and the host assigns it a unique address.

Each device sends an Attention command to the CPU asking for an address assignment when the device powers up or is reset. The Attention routine in the CPU receives the command and sends an Identification Request command to the default I/O device slave address, 6Eh. The device responds with an Identification Reply command. This consists of a 29-byte ID string including device type, model, and so forth, plus a unique 32-bit number, typically a random number. The random number lets the CPU distinguish between identical devices.

The CPU records this data, picks the next available soft address and sends it with an Assign Address command to the device. The CPU sends a copy of the identification string with the new device address to ensure that the correct device receives

it. As a final precaution, the I/O device sends a Reset command to its own address in case another identical device had an identical 32-bit random number and was assigned the same slave address. In this case, the arbitration ensures that only one device sends a message at a time. One device sends its reset command first, and the other device receives it before it can issue its own reset command.

The sequence for soft address initialization of the CPU is summarized below and is shown in pseudocode in Listing 3. The initialization of each I/O device is the complement of this, and is shown in Listing 4.

- CPU broadcasts a reset command and all devices revert to default address: 6Eh.
- Each device sends an Attention message, informing the CPU of its existence.

- CPU sends an ID Request command to the default slave device address: 6Eh.
- All devices try to respond with their ID data containing a unique 32-bit number.
- I²C bus arbitration causes the messages to be received one at a time.
- CPU records the ID data and random ID number for each device.
- CPU sends an Assign Address command with its new slave address to each device.
- Each device sends a reset to its assigned address to solve any duplicate device problems.

The Attention routine in the host that does the soft address assignment automatically is the same one that handles hot plugging. When you plug a new device into the bus, it powers up in a reset state and issues an Attention command. The CPU responds with an Identification Request, and slave address assignment proceeds like at system power up.

One of the side benefits of automatic address assignment is the CPU generates a record of all devices currently active on the bus. Each time you add a new device to the bus, the CPU automatically updates this table. The CPU does not automatically update the table when you remove a device, however. If you want this

Listing 3—Each device on the ACCESS.bus is assigned an address when it's connected or when the bus is reset. The CPU initiates the process.

```

;Setup Code at Reset

RESET:
  Write C9H to Control Register to enable I2C, interrupt, 100 Kbaud
  Write 6Eh to Slave Address register
  Loop to send a Reset command to each I/O device address, 01h
    through FEh
  Exit to main program loop

;Interrupt routine to service Attention Command from slave at 6Eh
  to CPU at 50h

ATTN:
  Send Identification Request command to 6Eh (default slave address)
  Wait for Identification Reply. If no response in 40 ms, exit
  Put identification string in device table. Select next available
  slave address.
  Send Assign Address command with ID string and new slave address
  Exit

```

A PC-to-ACCESSbus Interface Card

by Robert Clemens and Tom Stockebrand

The card and software described here are for a first-generation interface that we supplied to DEC and Signetics as a means of quickly getting an ACCESS.bus interface in a PC. ACCESS.bus is a handy tool for solving the problem of not enough hardware ports. It also allows for adding or removing peripherals while the system is running (hot plugging). For example, you may want to navigate through Windows using a mouse while in one application, but use a digitizing tablet or track ball in others. You can have them all available to plug in in any combination and still have serial ports available.

BACKGROUND

A few years ago, Ken Olsen at DEC asked our group to solve the problem of hooking a bunch of desktop peripherals together without the rat's nest of wiring that is usually needed for this task. Our A/D group had

adopted the Philips I²C bus as the means to build our display system prototypes. We were exploring the idea of extending it to the world outside a PC board. Since it is a fairly fast (100 kbps) serial bus that can be daisy chained and hot plugged, it was very attractive. It turned out that the workstation group had been thinking of the same thing, so the project was born.

Our group in Albuquerque worked on the hardware, the Display Systems Group back in Westford, Mass., did the software architecture, and Robert did the software design and construction. We soon joined forces with Signetics (now Philips) and they have taken over the work (along with the ACCESS.bus Industry Group) of getting the ACCESS.bus supported and encouraged.

The hardware problem was that the specs for the I²C bus required a 1-us maximum rise time. This limits the capacitive load to 300 pF with the specified 3k-ohm pull-ups. This is not very much head room if any significant amount of cable is to be driven. The hardware solution that we hit upon was to drive the bus with current sources rather than with resistors. That way the rise time specification could be met with a load as high as 800 pF. Peter Sichel and the design team at DEC worked

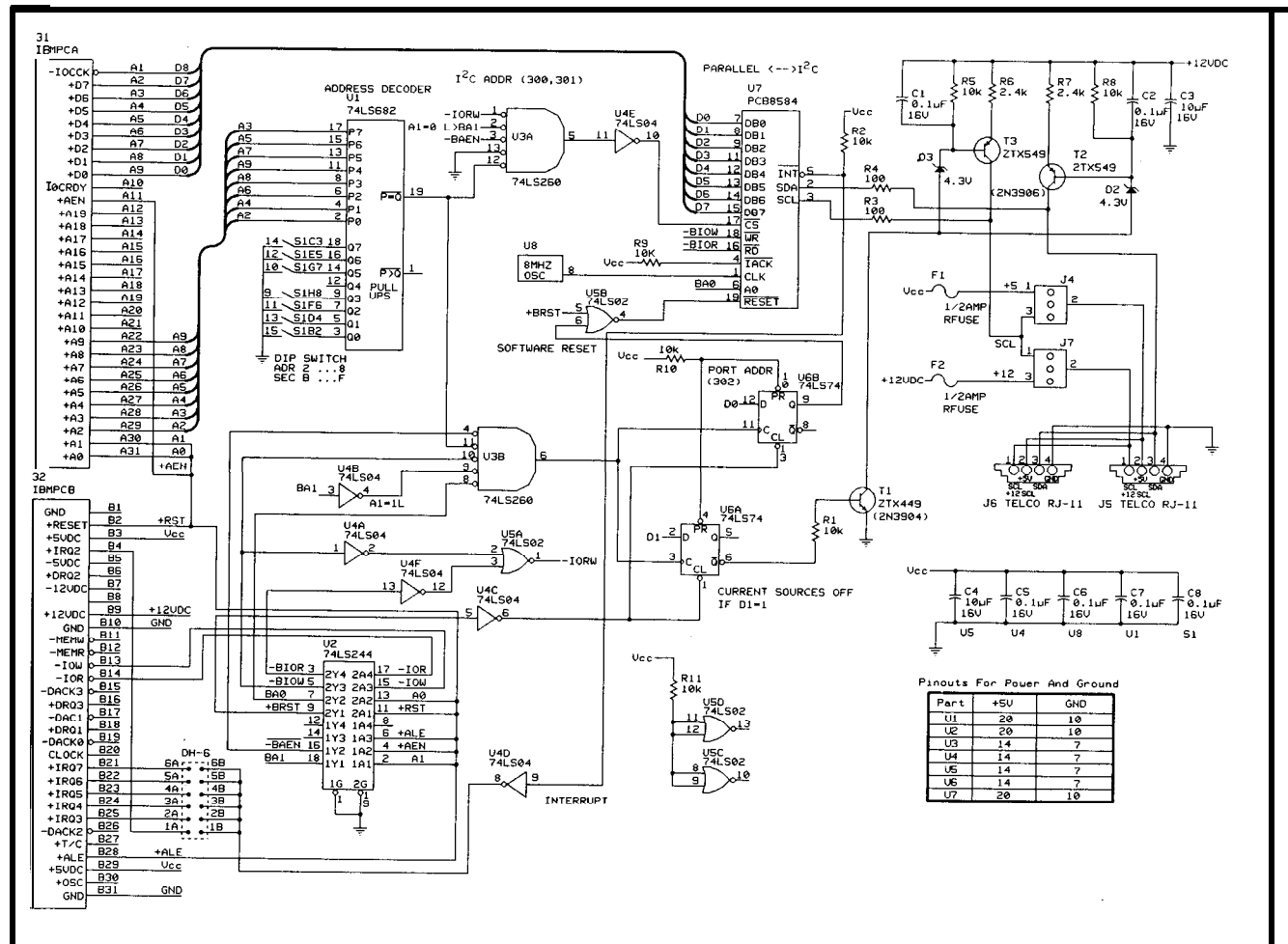


Figure 1—The core of the PC-to-ACCESSbus interface board is a Philips PCDB584 PC serial-to-parallel converter chip.

out a fairly spare software protocol, and other details were worked out such as standardizing on a plug and a low-capacitance cable.

At this time, the ability to put two formal hosts on the bus isn't in the spec. It is possible, though (more information on this is available from Robert). In fact, we think providing a means of hooking two or more CPUs together is one key to success for the concept. Therefore, there is a requirement to be able to shut off the current sources in all but one of them so as to limit the current that has to be sunk by the PC hardware. The board described here has current limiting using PTC resistors, but no direction limiting. A revised version should provide current direction limiting or just disable the local power supplied to the bus when the current sources are turned off.

HARDWARE DESCRIPTION

Figure I contains the schematic of the PC-to-ACCESS.bus interface card. The core chip on the board is a serial-to-parallel converter for the I²C bus made by Philips called the PCD8584. It is very similar to a UART but provides for the particular sophistication needed by the I²C protocol. It is driven with a standard PC address decode structure. There are current sources on the card for powering the bus and a means to turn them off by setting a port bit and also to do a software reset.

In this first-generation board, we limited the current that would run down the power supply wire by means of PTC resistors. They work, but are slow and expensive. A better way is to provide the +5 V with a regulator chip that is current limited to 0.5 A, adding fuses "just in case." The latest ACCESS.bus specification does not

require the +12-V option as provided in this design. The detailed specs are outlined later in this article.

SOFTWARE DESCRIPTION

I use a base address of 300h as an example in this discussion. An IORead from address 300h gets the status of the I²C interface. An IOWrite to address 300h issues a command. A Read from 301h gets the I²C data into the PC and a Write to 301h puts data out to the I²C bus. An IOWrite to address 302h loads the port bits. Setting the LSB (bit zero of address 302h) to 1 causes a software reset (it is cleared with a hardware reset and must be cleared again after a software reset). Setting bit one of port 302h turns the current sources OFF.

The duties of a driver or application that wants to talk to an ACCESS.bus network are the address configuration and management of devices during power-up of the host, recognition and configuration of new devices as they are plugged in, and management of messages to and from ACCESS.bus devices and PC-based applications. Management of the devices is the same in implementations for all platforms. The handling of messages to and from host-based applications are platform specific.

As mentioned in main article, in order to maintain a basic working ACCESS.bus system, a table of devices must be built in software that keeps track of a device's current running status, its I²C address, ID string, capabilities string, and a current pointer into the capabilities string. Also, a global variable for bus status must be maintained because a forced reset of the bus causes certain time-dependent actions to occur, and it is possible for a misbehaved device to create a bus error that can affect the state of the other devices on the bus.

Listing I shows some basic program structures and condition states that are useful when writing the software.

The last components needed are the routines that parse the message stream, install the appropriate hardware I/O and interrupt routing, and a link into the platform's timebase to drive time-dependent activity.

A Turbo Pascal source file is available on the Circuit Cellar BBS that works with this hardware design to implement a simple ACCESS.bus system for an IBM-compatible PC. This system allows viewing messages from devices, sending a message to a device, and starting/managing a reset sequence.

Listing I--There are several data structures and condition code definitions that are helpful when writing ACCESS.bus support code.

Recommended device data structure:

```
AB_DEVICE = record
    status      : integer;
    address     : integer;
    ID          : string[30];
    Capabilities : string[MAXCAPABILITIESLEN];
    CapOffset   : word;
end;
```

Recommend global bus conditions:

```
BusReset      Software is starting a reset sequence
BusAssignAddress Software is assigning addresses to devices
BusConfigured Software can begin normal operation
BusError      Error has occurred, attempt to reset bus
```

Recommended device conditions for each device are:

```
DevReset      Device is being reset or no device is
               at this address
DevWait       Device is busy, do not send message
DevConfirm    Confirm device is still connected
DevConfigured Device is ready for normal operation
DevError      Device has a problem
```

DETAILED HARDWARE INSTRUCTIONS

JUMPERS

There are three headers on the board, each with a single shorting jumper. One is to select one of the six Interrupt Request Levels (IRQ 2-7). The other pair is to set the unit to provide either 12 volts or 5 volts to the RJ-11-style output connectors. There is also one DIP switch to be used in selecting the address group to which the card will respond.

IRQ

The IRQ header has six pairs of pins. The upper row of pins is connected to the output of the -INT line from the PCB 8584 serial-to-parallel converter chip through an inverter. The lower row of pins is connected to IRQ 2-7 in order across the plug from left to right. In order to use the Interface card, one IRQ should be selected by moving the jumper clip to the appropriate horizontal position and using it to jump the upper to the lower pin at that point.

POWER TO THE LOADS

Just behind the output sockets on the card are a pair of three-pin headers. A jumper on the upper one selects whether +5 V or clock (SCL) is applied to pin 2 of the output socket. A shorting block on the lower header determines whether +12 V or SCL is applied to pin one (the bottom pin) of the output sockets. The output sockets are wired in parallel. The two voltages are supplied through positive temperature coefficient resistors, used as resetting fuses, which have a resistance of 0.5 ohm and will allow 0.5 A to pass before starting to heat up and limiting the current that can be supplied externally.

For 5-V operation, the jumper clip on each header should short the center pin to the upper pin. For 12-V operation, they should each short the center to the lower pin on the header. If the jumpers are in the wrong position (either connecting the power to the wrong line or supplying both pins 1 and 2 with power, or neither) no harm will be done, but the system will not work.

ADDRESS SELECTION

The only DIP switch on the board is for selecting the port address for the interface card, modulo 4. The switch position labeled "1" is unused. The seven switches marked 2-8 enable the corresponding lines for address comparison with the corresponding address lines on the PC bus. For example, if the base address for the card is to be 300 (the usual case) then all switches should be closed, grounding their respective inputs to the address comparator, except number 8 which will then be pulled high internally. Internally, the address comparator's "9" input is held high at all times. This is because address bit 9 must be a one for all port accesses on a PC. Since

address bits 0-1 are not fed to the comparator, its output will be true for addresses 300 through 303. The low address (e.g., 300) is for reading or writing data to the ACCESS.bus. The next one (e.g., 301) reads or writes the command/status register and the third (e.g., 302) writes from PC to the two data port register bits for doing a software reset (DO) or turning off the current sources (D1).

The lowest address in the available space is 200 [all switches closed or "On"] since this corresponds to bit 9=1 alone. The highest address is obtained with all switches open yielding 3FCh as the base address. This address range can be shown as 01000000xx to 111111 lxx in binary, where xx are the low-order bits used to select the port addresses 0-3 as described above.

JUST THE BEGINNING

As ACCESS.bus catches on and makes your desktop a lot less cluttered, a board such as the one we describe here can be your ticket to exploring the possibilities ACCESS.bus opens up. Have fun.

Robert Clemens has done extensive programming for MIDI interfaces including sound editors for the PC. He also does embedded application programming in 8051-type microcontrollers using I²C and ACCESS.bus protocols. He is available at MediaMultiTech, P.O. Box 426, Durango, CO 81302, (303) 247-4726.

Tom Stockebrand spent 28 years working for DEC, mostly in the capacity of product design engineering and management for peripherals such as tape drives, communications interfaces, and displays. He did the original DECTape design, and his group did the original DEC VT50 series of terminals. In the 1980s, he ran an A/D group developing very high resolution displays at DEC's Albuquerque facility. He now has a small engineering consultancy called LGK Corp. which tries to provide a one-stop garage shop for getting breadboards and prototypes designed and built.

Listing 4—ACCESS.bus Pseudocode for 87C751/O Controller for Soft Address Initialization

```

RESET:
Set to Slave mode: Write 90h to Config. Reg. (default mode)
Write 00h to Control Register
Set 6Eh as default slave address
Send Attention command to CPU at 50h
Wait for Identification Request command from CPU
Send Identification Reply command to CPU with 32-bit random ID #
Wait for Assign Address command.
Check Id Reply against ID string. If match, accept new slave addr.
Send Reset command to new slave address.
If arbitration error, wait for N+50 usec, where N is 8 LSb s of
32-bit random number
If reset received at new slave address while waiting, start over
at RESET.
Exit to main program loop
    
```

feature, you must add an additional background scan routine.

WHERE CAN I GET MORE INFORMATION?

Information on the ACCESS.bus is available from the ACCESS.bus Industry Group. They will supply you with the latest detailed specification of the bus. Another good source for both

ACCESS.bus and I²C bus information is the Philips Semiconductor Data Handbook for 80C51-based 8-bit Microcontrollers. It contains information on both buses as well as the I²C chips that can be used with them. Computer Access Technology provides boards and software for ACCESS.bus. You can quickly set up a system using a PC as a host with their boards.

David Wyland specializes in Scheduled Inventions, combining his background in analog and digital system architecture to run his consulting group. Contact him at The Wyland Group, 15213 Bowden Ct., Morgan Hill, CA 95037, (408) 778-3860.

CONTACT

ACCESS.bus Industry Group
415-l 12 N. Mary Ave., Ste. 265
Sunnyvale, CA 94086
(408) 991-3517

Philips/Signetics Semiconductor
8 11 E. Arques Ave.
Sunnyvale, CA 94086
(408) 991-3445

Computer Access Technology
949 Hillsboro Ave.
Sunnyvale, CA 94087
(408) 732-8910

IRS

404 Very Useful
405 Moderately Useful
406 Not Useful

INTERFACE

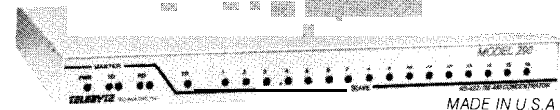
Model 285 . SUPERVERTER™



\$ 1 4 8

A unique interface converter, user selectable, connects RS-232 to RS-485 or RS-422. Converts to full duplex, 4 wire, or half duplex, 2 wire.

- Intelligent Control of RS-485 TD/RD
- TD and RD LED's
- DTE/DCE Compatible
- Rates up to 64 KBPS



Model 290 . RS-422/RS-485 CONCENTRATOR-WIRING HUB \$725

Increase system reliability while breaking the RS-485 limitation. Expand to 496 modes, sixteen ports, individually settable to RS-422 or RS-485. The built-in interface converter conditions RS-232 master port to RS-422/RS-485 signals for slave ports. Half or full duplex operation.

- Expand to 496 nodes
- Isolation between ports
- 16 Port Capacity
- Surge Protected

TELEBYTE TECHNOLOGY, INC.

270 E. Pulaski Road
Greenlawn, NY 11740
TEL: (516) 423-3232
FAX: (516) 385-8184
1-(800) 835-3298

CONNECT • COMMUNICATE • SUPPORT

Project Parts

Firmware Flyers (prices shown are postpaid in US/ with any parts order)	
8051 Firmware Debugging Techniques (65 pages w/ 3.5" diskette)	\$18 /15
Introduction to the 8051 Instruction Set (46 pages)	\$10 / 8
8051 Instruction Reference Card	\$3 / 2
8255 Cheat Sheet Reference Card	\$3 / 2
Embedded 386SX Project Parts (INK 33, 34, 35)	45 / 0
LD273 dual IR LED (bright, wide beam)	2.10
UGN3503U Ratiometric linear Hall Effect sensor	2.25
IR3C02A laser diode controller (±5, for LT022MCLN9705P laser)	2.30
IR3C07 laser diode controller (+5V, for LT022PDLN9705 laser)	2.85
PH302 fast IR photodiode	3.10
GP1U52Y 40 kHz IR receiver (side-looking)	4.00
ULN3751Z Power op amp (±13V to ±13V supply, 3.5 A output)	4.60
IS1U60 38 kHz IR receiver	4.80
IDN2993B Dual full H-bridge bipolar stepper motor driver	5.45
IRSAMPLE parts (PH302, LM311, etc. See MCIR Link article, INK 29)	5.70
Excellent IR filter (opaque to visible light, 35 mm slide mount)	6.75
MC145030 IR encoder/decoder	6.75
DS1232 Micromonitor watchdog	6.75
UCN5841A Serial-input, 8 latched 500 mA sink drivers	6.90
UCN5895A Serial-input, 8 latched 250 mA source drivers	7.50
UCN5804B Unipolar stepper motor translator/driver	8.00
CS212 S-ART I/O network/security monitor	8.10
DS2400 Silicon Serial Number (2 chips)	8.40
DS1210 NVRAM power controller	9.75
DS1231 Power monitor	9.90
IR I/O: IS1U60, LD273, CD4047, 2N2907, red LED, schematic (IR-Link)	10.40
MAX691 Power supervisor	11.85
DS1202 Serial clock/calendar	12.00
75T204 DTMF decoder	12.00
IL300 linear optoisolator	12.70
MAX233 self-contained -5V powered RS-232 interface	14.25
M18880 DTMF encoder/decoder (bus interface)	18.90
PL513 Send-only X10 power line interface	20.00
MAX134 digital multimeter	33.00
MAX252 optically isolated +5V powered RS-232 interface	64.00

UPS Ground/2nd day/Next day \$6.50/9/18 to 48 US states. COD add \$4.50. Canada \$6 via USPS Air Small Packet, no CODs. Check, MO, or COD only, no credit cards. POs add \$50, but call first. CT residents add 6% sales tax. Quantity discounts start at five parts. Data Sheets included.

Call/write/fax for seriously tempting catalog...

Pure Unobtainium

► Your unusual parts source ◄

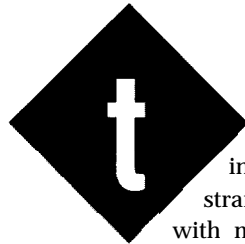
89 Burbank Road • Tolland, CT 06084-2416
FAX/voice (203) 870-9304

High-Speed Modem Basics: Standards and Theory

What do all those modem "V." numbers mean and what do they do? What about all those file transfer protocols available on BBSs? In this first part, Michael tries to shed some light and introduces his Gemini modem.

FEATURE ARTICLE

Michael Swartzendruber



The computer industry has a strange relationship with modem technologies. On the one hand, we have become so used to modems being around that we don't give the technology the respect it deserves. Similarly, we are always looking to get the most horsepower and bandwidth out of these tools, and the market is always trying to get the industry to push the envelope. So far, they have been successful in this pursuit; the latest modem standard-V.FAST-is being designed to enable throughputs of up to 38.4 kilobits/second (before compression of data) over a dial-up line. If you think about it, if the industry doubles the V.FAST specified speed, you are talking about ISDN rates or the same bandwidth as one fractional T1 channel. These rates over the dial-up network would have been un-dreamed of as little as a decade ago. But the modernization of the central office switches and new silicon technologies have come together to

make today's data-speed demons a reality.

In this first part of a two-part series, I will explore modern modem standards, define some commonly bantered about terms, and look closely at the state-of-the-art. In the second installment, I will build a modem that I have called the Gemini, which uses an impressive array of cutting edge technologies.

OVERVIEW OF THE GEMINI PROJECT

The Gemini modem project is composed of chip technology primarily from Exar Corporation. It also employs technology from a hybrid DAA manufacturer: Cermetek. Together, these two companies' products form the "core" of the Gemini modem.

What do these vendors bring to the project? In a nutshell, the Gemini modem gets its capabilities from the technologies these companies provide. The Gemini inherits the following set of features from them: the Gemini is a Bell 103/V.22/V.22bis/MNP2-5/V.42bis modem capable of sustained data throughputs up to 9.6 kilobits/second. It also supports industry de facto standard command sets and includes extensions to allow configuration of the modem's MNP protocol features. The Gemini modem contains an EIA V.24 DTE serial interface for connection to the host PC system. The DAA (described next month) is designed to be compliant with FCC Part 68, UL-1459, and Canadian DOC standards. The DAA used in the Gemini is approved for use in Europe.

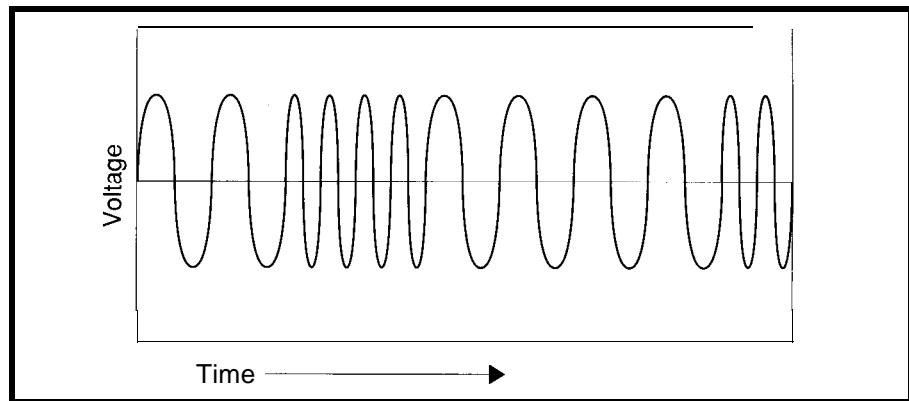


Figure 1—In FSK modulation, the data bits are encoded to the frequency appropriate to the value of the bit. A mark (logical one) is 2225 Hz and a space (logical zero) is 2035 Hz. The Bell 103 modulation standard uses 300 baud with one bit per baud.

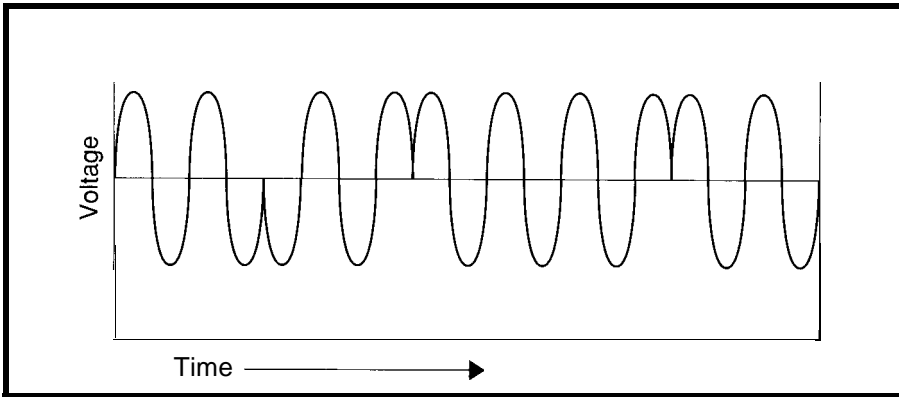


Figure 2—In DPSK modulation, the receiver uses the phase difference between baud frames to recover the bits encoded in the carrier wave. This technique is used in the V.22 and Bell 212 standards. Each standard specifies 600 baud with 2 bits encoded per baud, giving a throughput of 1200 bps.

First two bits in quadbit (2400 bit/s) or dibit values (1200 bit/s)	Phase quadrant change	
00	1 → 2 3 → 4 4 → 1	90°
01	1 → 1 3 → 3 4 → 4	0°
11	1 → 4 2 → 1 3 → 2 4 → 3	270°
10	1 → 3 2 → 4 3 → 1 4 → 2	180°

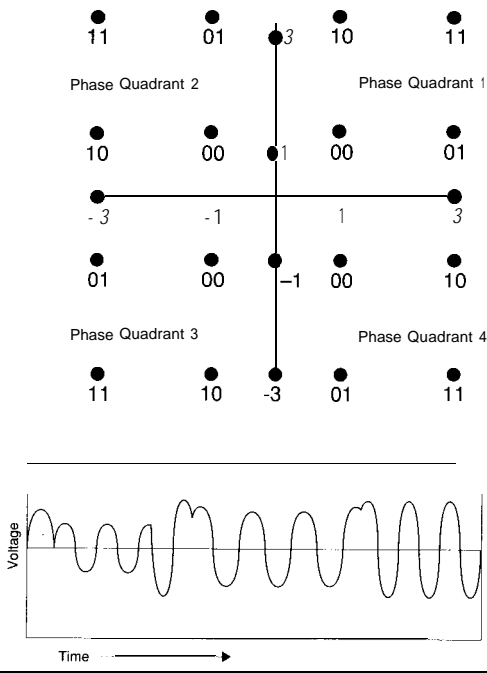


Figure 3—In quadrature amplitude modulation (QAM), the first dibit is used to determine the phase change between baud frames (top). For each of the four possible values of this dibit, there are relative and absolute phase changes possible. The signal constellation (middle) shows how the value of the second dibit determines the location of the signal in the signal space. In an example of the modulated signal stream (bottom), both the phase and the amplitude of the wave are affected by this modulation technique.

WHAT'S ALL THIS V STUFF?

To the uninitiated, these standards may be next to meaningless. This section describes some of the meaning of the standards mentioned above. Understanding what these standards mean can be very beneficial in many ways.

THE LOWEST LAYERS

Modems compliant with the Bell 103 data modulation scheme support a base data rate of 300 baud and 300 bits per second (one bit per baud), and use Frequency Shift Keying (FSK). One frequency (2225 Hz) is assigned as a logic one (mark) and another frequency (2025 Hz) is assigned as being a logic zero (space). The Gemini modem does not normally transmit data at this speed, but if it were commanded to, it could. See Figure 1 for a pictorial description of the Bell 103 modulation standard.

When a modem is described as being compliant with CCITT (Consultative Committee International Telegraphy Telephony—an international standards setting institution that operates out of the United Nations) standard specification V.22, it is a modem that can engage another modem using a full-duplex DPSK (Differential Phase Shift Keying) modulation scheme.

DPSK works by comparing the phase of the carrier wave being received during the current baud frame with the phase of the carrier from the previous baud frame. The phase change detection circuitry used during demodulation of successive baud frames is where the word “differential” applies. In this modulation scheme, the carrier's baud rate is 600 baud, with two bits per baud encoded on the carrier frequency resulting in a basic throughput of 1200 bits per second.

The method used to encode two bits per baud is one wherein a *dibit* is examined and found to have one of four possible states. Each of these states is assigned a phase shift of 0°, +90°, +180°, or +270°. For instance if the dibit with the value 11 (binary) is received at the modulator, it is assigned the fourth state. The baud

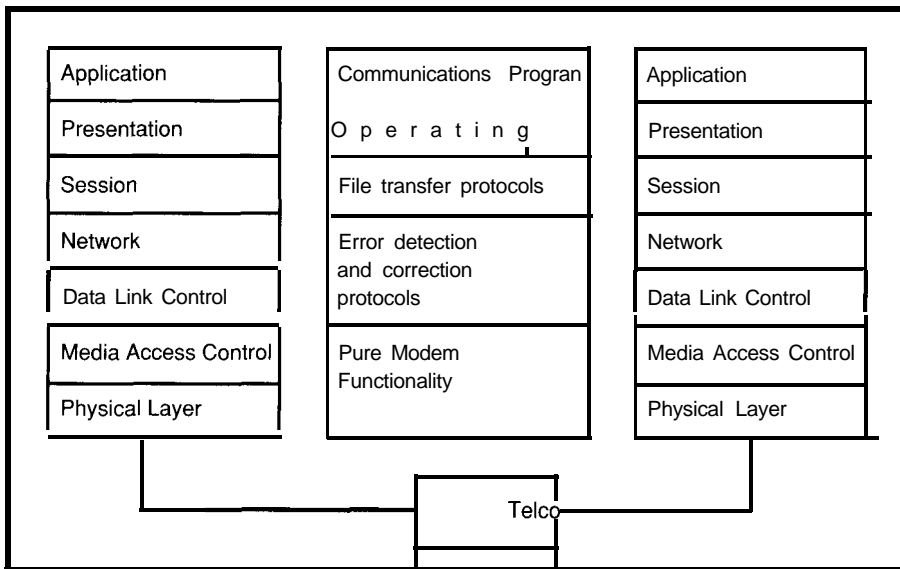


Figure 4—Labor is divided between the various hardware, protocols, and software components of a typical data exchange using modems compared against the OSI seven-layer stack.

frame associated with this dibit will be transmitted with a frequency shift 270° in the positive direction in relation to the frequency waveform of the previous baud frame. The degree of shift between the current baud frame and the previous baud frame is used at the demodulation end of the data circuit to derive the two bits encoded in this baud frame: the modem detects the amount of phase shift between the current baud frame and the previously received baud frame.

Since this standard describes a full-duplex protocol, the Gemini modem can simultaneously send and receive data with any modem that complies with the V.22 standard. See Figure 2 for an illustration of the DPSK technique.

The next data modulation standard that the Gemini modem supports is V.22bis—another CCITT standard. V.22bis describes a QAM (Quadrature Amplitude Modulation) data encoding technique. In this modulation scheme, four bits are encoded per baud. The baud rate of the carrier is 600 baud. In this encoding scheme, the four bits are encoded onto the carrier wave as follows: two bits are used to modulate the phase of consecutive baud frames as described in the V.22 standard; the other two bits are used to determine the signaling element of the new quadrant of the baud frame. Modulating 4 bits per baud on a 600-baud carrier results in a throughput of 2400

bits per second. For a visual description of this modulation method, see Figure 3.

The number of transitions that the carrier wave makes in a second is the baud rate of the modem. As I described above, up to four bits may be encoded on a single baud, which means a modem operating at 600 baud may have actual data throughputs of 1200 or 2400 bits per second. So with higher-speed modems, it is wrong to state that the modem transmits at “9600 baud.” It is correct to state that the modem is transmitting data at 9600 bits per second.

HIGHER PROTOCOL LEVELS

The Gemini modem also supports MNP levels 2-5, V.42, and V.42bis. I'll explain each of these data protocol standards in turn. Note that these data protocols operate in addition to the modulation standards I described above. For those of you familiar with the OSI seven-layer-stack protocol

model, the modulation standards I described are equivalent to the physical layer (including partial attributes of the MAC and DLC models), while the standards that I will explain now are roughly equivalent to the transport or network layer. The communications program running on the host computer is assigned the Presentation and Application layers. The session layer is shared between the modem and the communications program. See Figure 4 for a drawing that illustrates the OSI stack in relation to a typical modem data exchange.

I'll explain the MNP levels first, followed by the V.42 standards.

MNP is a data communications protocol that ensures the integrity of the data being transferred between modems. It uses HDLC (High-level Data Link Control) packet framing and LAPM (Link Access Protocol Modem) data protocols. HDLC is a synchronous data transfer protocol, referenced to a signal clock that is provided by one of the two modems. HDLC frames are stuffed with a negotiated number of octets (sets of 8 bits) into an “envelope.” A single collection of octets wrapped in delimiting symbols is called a frame.

The HDLC data frames are delimited at the start and end with a special character called a *flag*. The flag is a reserved character that generally will not occur in the data stream. If it does, the sending modem toggles one or more of the bits of the octet to create a certain “other” character. This “other” character is recognized at the receiving end as a “modified false flag” octet and the receiving modem reverses the bit operations the sending modem performed. The outcome of

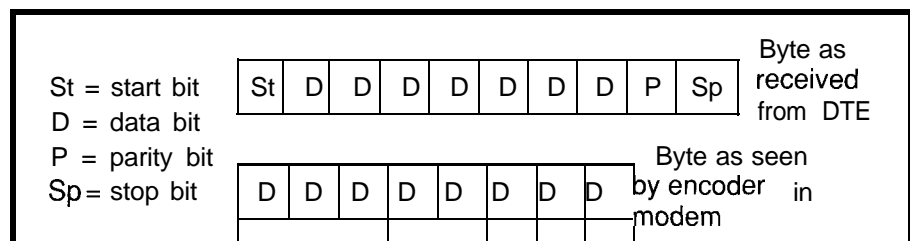


Figure 5—Since there are several error detection schemes in operation during a data exchange between two modems, the start, stop, and parity bits can be removed from the data sent by the transmitting DTE. The receiving DCE reinserts these bits if they are required at the receiving DTE. As a result, only seven out of ten bits have to be sent over the telco link, so the extra bandwidth may be used for data.

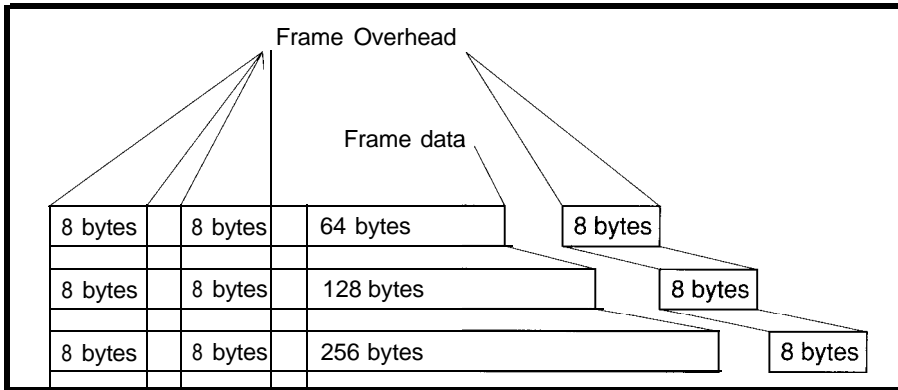


Figure 6—Since there is a certain amount of framing overhead associated with data packets that remains constant regardless of the size of the data portion of the packet, one way to increase the bandwidth efficiency of the analog channel is to increase the size of the data portion. The net effect is more bandwidth is used to transmit data in the channel. In this simple example, when the data portion of the data packet is increased from 64 to 256 bytes, the ratio of frame overhead to data transmitted is greatly improved.

this operation is the recreation of the original octet.

At the end of the frame, the sending modem adds a CRC (Cyclic Redundancy Check). The receiving modem calculates a CRC on the data as it receives and decodes the frame. If

the CRCs match then the receiving modem accepts the data frame by sending an ACK; otherwise, it sends a NACK (Negative ACK), which prompts the sending modem to resend the packet. The CRC is calculated by a modulo-2 polynomial division algorithm

OTHER COMMON AND EMERGING STANDARDS

There are four MNP standards that are not implemented in the Gemini that you may benefit from knowing about. The first one is MNP level 6, which defines link negotiation. It works by each modem trying to connect at its lowest speed (modems that include this protocol typically have a lowest speed of 2400 bps). Each modem will try to upshift the line speed through this negotiation. The protocol will disengage when one or the other modem fails to respond to the other modem's upshift request. At this point, they will both use the last successful upshift request as the line speed.

The next MNP protocol is MNP level 7. This protocol is another compression protocol that achieves a compression ratio of around 3: 1. MNP 7 compression is based on an adapted Huffman encoding technique with a predictive Markhov algorithm.

MNP level 9 is a protocol that increases the bandwidth of the analog channel by "piggyback ACKs." This technique attaches ACKs to data packets instead of transmitting a dedicated ACK packet. The NACK of MNP 9 modems is also more intelligent. It informs the sender of precisely which messages were corrupted so the sender only has to send those packets as opposed to a set of packets. This is especially helpful with streaming protocols.

The final MNP standard is MNP level 10. This interesting protocol allows modems to upshift transmission speed in the case where line conditions caused the modems to downshift during a transmission. Whereas before, when modems downshifted due to line conditions they would keep transmitting at the lower speed for the duration of the call. Now, they can negotiate to upshift again to the higher level of throughput if line conditions allow it.

While you're on the phone.. I thought I should discuss some current standards that are not implemented in the Gemini modem as well as some emerging standards that I am sure we will all begin to hear more about.

on the bits of information contained in the data frame.

The sending modem must receive an ACK or a NACK between each frame. When the sending modem receives an ACK, it sends the next frame of data. When it receives a NACK, it retransmits the previous data frame until it either receives an ACK on the frame or the retransmission limit is reached, at which point the connection is automatically terminated. Retransmissions lower effective throughput of the link, but the data is received error free.

Some newer modems will do one of two things when they detect excessive retransmits: downshift to the next lower speed or automatically disconnect. Automatic disconnects may seem like a drastic action, but if the link is bad enough to cause a disconnect triggered by retransmissions, it is probably better to disconnect and try another connection. Modems that supports midsession downshifting are pretty common; some will sense improving line conditions and will try to upshift again.

MNP level 2 describes an asynchronous data frame transfer with octet-oriented data formatting.

MNP level 3 is the first level where a gain in data throughput is possible. MNP 3 allows asynchronous transfer of data between the DTE (PC) and the DCE (modem). The modem strips off the start and stop bits from each byte received before it transmits the data. The receiving modem reinserts the stripped bits as it relays the data to the DTE. The link between the modems is synchronous, and 1 to 64 octets are packaged into each data frame. The MNP 3 protocol allows throughput to increase to 108% of the DTE port data rate because of the bit stripping performed on every byte. See Figure 5 for a description of the bit stripping methods.

MNP level 4 is an extension of MNP level 3. The frame is expanded up to 256 octets per frame. The modem can adjust the size of the packet according to the changing qualities of the phone line. Another way that MNP 4 increases frame

IMAGENATION...

VIDEO FRAME GRABBERS

- ◆ Simplicity
- ◆ Functionality
- ◆ Affordability
- ◆ Accuracy

Cortex-I

\$495⁰⁰

- ◆ Real-Time Capture
- ◆ Half Slot XT/AT
- 512x484x8Bit
- ◆ RS-170/CCIR
- ◆ External Trigger

Cortex-STD

\$595⁰⁰

- ◆ Dual Video Input
- ◆ Opt. XMS Mapped
- ◆ Low Power Options
- ◆ STD-80 or 32 Bus
- ◆ External Trigger

Vidmux-4

\$99⁰⁰

- ◆ 4 to 1 MUX
- ◆ Half Slot XT/AT

Includes Software...

- ◆ C Library & Source
- ◆ Image Capture Utility
- ◆ Tiff Utilities
- ◆ "Image" Drive Ram Disk Emulation

STARTER PAK...

Cortex-Pak

\$1395⁰⁰

- ◆ NEC/TI-23EX Camera With Lens
- ◆ 9" Video Monitor
- ◆ Frame Grabber
- ◆ Software & Cables

OEM PRICING AVAILABLE

IMAGENATION CORP.

P.O. Box 84568
Vancouver, WA 98684

PH/FX (206) 944-9 13 1

One of the established standards that the Gemini does not use is the V.32 and the V.32bis modulation standard. It's not because I didn't want to add the support; Exar's chip set that supports this standard is not quite ready yet. But, no discussion of protocols would be complete without it.

The V.32 standard defines a higher baud rate than the V.22 and V.22bis standard. Where V.22 defines a 600-baud transmission rate, V.32 defines a base transmission rate of 2400 baud. In V.32, the data is encoded into the carrier using two optional techniques. One of these techniques, called the 16-point signal structure with nonredundant coding, looks virtually identical to the QAM methods associated with V.22bis.

The other encoding technique is a little more interesting. Called **trellis-coded modulation**, this encoding technique uses *quadbit data objects* (nybbles to some of us) as inputs to the modulation function. The quadbit is then broken in half in the customary way. One of the dibits is run through a differential encoder, where the two bits are substituted with another pair of bits. This substitution is based on the current value of the dibit and the previous value of the same dibit. The dibit that is generated by the differential encoder is then input into a **convolutional encoder**. The convolutional encoder generates a bit called the **redundant** bit. The redundant bit and the differentially encoded dibit (passed through this encoder unharmed) are output from the encoder.

Next these three bits are recombined with the other dibit of the original quadbit. Since we now have five bits to encode onto the carrier, the signal space is expanded to 32 points. What this does is place more distinct signal points in each of the phase quadrants. The trellis-coded modulation technique is more immune to line-induced signal errors since the redundant bit acts as a sort of "parity bit." Figure 7 shows the signal constellation for the two modulation techniques used in V.32 or V.32bis.

Another interesting fact about the V.32bis standard is the decode function in the modem is able to perform more error correction on the signal because of the "parity bit" that is encoded in the signal. This is because there are strictly defined transitions allowed between successive baud frames, and only a subset of these can be produced by the sender. When the receiving modem sees a suspicious baud frame, it is allowed to "guess" at the appropriate decode for that baud frame based on the previous states. The receiving modem has more information to reconstruct its guess, therefore it has more chances at making a correct "guess."

AND IF THAT WEREN'T ENOUGH ALREADY

There are two new standards about to hit the streets. You may have heard about them already, but if you haven't they are the V.terbo and the V.FAST standards. These are the next up-and-coming, rising young stars on the modem horizon. They are the next evolutionary step beyond V.32. I'll share with you what I have been able to find out so far.

V.terbo is an evolution of everything that's been applied up to now. It uses DPSK modulation and TCM. However, it is a stepwise refinement over V.32bis because it can encode up to 8 bits of data per baud. V.terbo modems can run at a base speed of 14.4 kbps, 16.8 kbps, or 19.2 kbps. When these modems are running at 14.4 kbps, they use the signal constellation defined by V.32bis. When they are running at 16.8 kbps, they use the signal constellation shown in Figure 8. And when they are running at a full-speed 19.2 kbps, then they use the signal constellation shown in Figure 9. If the modem is running at full speed, it is pumping data at a throughput of up to 19,200 bits per second **before any** compression is performed on the data. Now if you apply the V.42bis compression algorithm on top of this base throughput, you are talking about 76,800 bits per second. Pretty darn

fast by most people's standards.

V.terbo modems also use a technique called *nonlinear encoding* at the transmitter (and *nonlinear decoding* at the receiver). This encoding method is used to keep the power level of each baud frame at a consistent level for each signal in the signal space. This encoding technique offers another very powerful benefit. It allows the analog signal to survive the vagaries of ADPCM that is applied in the telco network. ADPCM is used to digitize signals from the local loop (which is mostly analog) for transmission in the all-digital telco network. ADPCM can be too slow for some of these high-speed signals, so nonlinear encoding is used to reduce the negative effects of ADPCM. These modems also use preemptive amplification of the signals that are at the upper end of the frequency band of the local loop so it is more robust.

Believe it not, V.terbo is a fallback position taken by many key players in the modem marketplace who had some trouble implementing V.FAST. V.FAST outlined a base throughput of 28.8 kbps over a dial-up line. If you play the four-to-one game on this one, you come up with the staggering figure of 115,200 bits per second! Fractional T1 from a modem!?! Until the local loops are completely digital, some observers think that 76 kbps is as fast you can get the circuit to go, no matter what you hang off of it. The V.FAST standard is still in its early evolutionary stages and mundane things such as modem negotiations are still being hammered out. It will be some time before these modems become commonplace (let alone affordable). Another interesting trick these modems will probably have to perform is "line probing," where the modem will "sound out" the line to determine the quality of the connection before it will begin to transmit, so it can take best advantage of the bandwidth and line conditions available on that virtual circuit.

There is an insidious lurking problem associated with such a dense signal space, and that's line noise. When signals become closer together detecting them and decoding them becomes increasingly problematic. That's the very reason some cautious spokesmen say that even though these devices may perform well in the labs, don't expect it in the real world until the infrastructure in the telco is modernized with equipment that can transmit these high-speed signals cleanly.

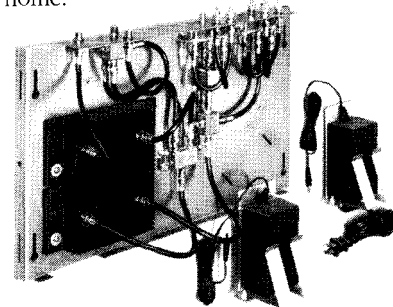
AND FOR THE TRAILBLAZERS AMONG US

Another interesting modulation scheme used by Telebit Corporation is a spread spectrum modulation technique called PEP (a couple of other companies may have adopted this idea, although their implementations are not necessarily compatible). PEP stands for Packetized Ensemble Protocol. This interesting (albeit proprietary) method of data encoding and modulation works as follows: The analog channel of the voice circuit is divided into 511 distinct frequency regions. Each of these regions is assigned a particular carrier frequency. Data from the DTE is encoded onto more than one carrier simultaneously with up to seven bits of data. This creates a sort of parallel transmission of the data over the phone circuit, if you will.

Another interesting trick that PEP performs is in regards to its response to error detection. When the modem senses an error increase, it responds by not using the carriers nearest the passband edges of the voice circuit and concentrates more information nearer the center frequencies of the voice circuit. This trick takes greatest advantage of the bandwidth inherent in the voice circuit since error-causing noise is most problematic near the edges of the spectrum of the voice circuit. The PEP protocol has been logged in at 33,000 bps before compression, and, of course, Telebit offers compression to boot.

Home Runs For MultiRoom Video. Great Hits. No Errors.

It's the unseen hero. The ChannelPlus® 3100 Coaxial Cable Panel distributes great pictures to up to 8 TVs. Just home run TVs, modulators and incoming cable (CATV, off-air antenna) to the panel and you're nearly home.



ChannelPlus® Coaxial Cable Panel delivers excellent picture quality to up to 8 TVs, features all needed components to amplify and balance signals, minimizes installation work.

All amplifiers, splitters, combiners, taps and tilt compensators are mounted on a base plate. Power supplies and brackets are also included. Just connect everything and you're done. Adjustments are minimal, and all inputs and outputs are clearly marked for error-free set-up.

Complete Video System

The 9020 Multi-Room Video System includes the 3100 Coaxial Cable Panel, a model 4336 modulator (3 inputs) for inserting up to three "in-house" generated channels, and all hardware required, including wall plates and connectors. Just add coax and J-boxes. Installation is a snap.

For more complete information, call Multiplex Technology, Inc. (714) 996-4100 or fax us at (714) 996-4900. For technical assistance, call 1-800-999-5225.

Channel
PERFORMANCE MULTI-ROOM VIDEO
#1 22

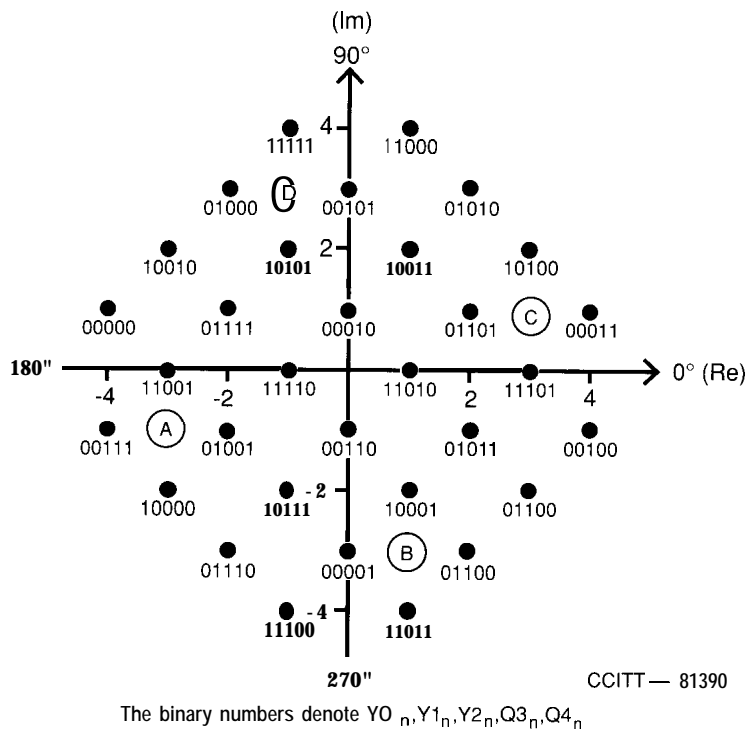
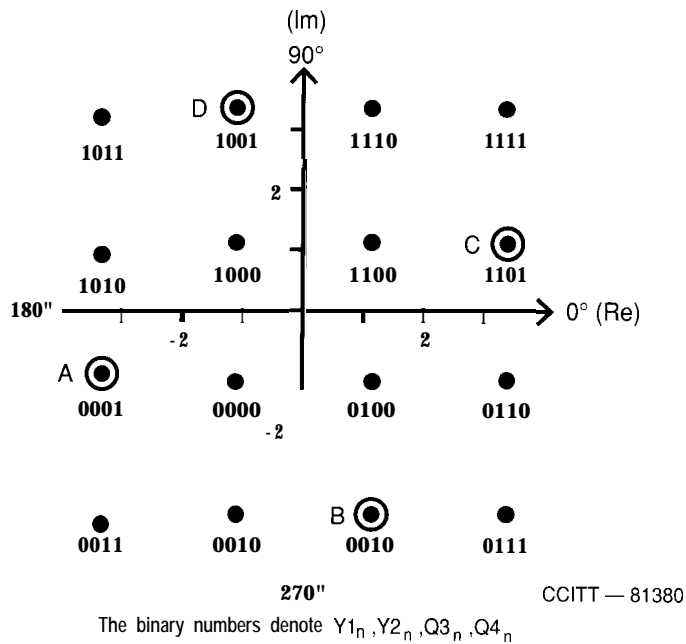


Figure 7—Note the striking similarities between the signal constellations for V.22bis (Figure 3) and V.32 (top). V.42bis (bottom) is very similar except it employs a 32-point signal space. The signal points outlined with circles are those used when the modems are in negotiation or have fallen back to 4800 bps.

efficiency is that certain redundant control information is not transmitted with each packet. MNP 4 can increase throughput to 120% of the nominal data rate because less data framing overhead is required for each character transmitted. The result is a 2400-bps modem can have an actual throughput of about 2900 bps. Figure 6 shows how this works.

MNP level 5 performs real-time data compression on the byte stream. MNP 5 can increase the effective throughput of data up to 200% of the nominal data rate. This throughput gain is the combined gains of the bit-stripping techniques and the reduction of data bits that are transmitted due to the compression algorithms associated with MNP 5.

You may have noticed that MNP modems communicate synchronously and asynchronously. The link between the modems is synchronous, while the link between the PC and the modem is asynchronous. The synchronous link doesn't require start and stop bits for each byte being transmitted, so they are stripped off before being transmitted through the modem link. Therefore, only 8 bits are sent for every 10 bits the modem receives from the PC, which gives a possible maximum performance increase of 20%. However, some of this increase is eaten up by framing bits and CRCs, so the actual increase is closer to 8%. MNP 4 packages more octets into a single frame, therefore boosting its throughput performance in comparison to level 3.

CCITT's V.42 is implemented as an alternative data transmission error control mechanism that is practically identical to those employed in the LAPM implemented with MNP/HDLC.

V.42bis is a technique that can provide up to 4:1 compression. It is based on British Telecom Lempel-Ziv-Welch compression algorithm. It is superior to MNP 5 because the compression algorithm has the ability to compress the data to a greater level than MNP 5 is able to do.

The V.42bis compression algorithm builds compression tables based on the data input to the algorithm rather than the static, blind substitution table as in MNP 5. The substitution table employed by V.42bis is refreshed periodically to keep the compression table "accurate," therefore it is an "adaptive" algorithm. Table updates occur when the algorithm senses data expansion (instead of compression) is beginning to occur. Invalid table entries are deleted using the least recently used process of elimination. Some implementations of V.42bis will turn the compression algorithm off if expansion is sensed.

CONNECTIONS

Like most modems available today, the Gemini modem connects to the DTE through a serial port. The serial port on today's garden-variety

modem contains the following signals on its serial port: TXD (transmit data) on pin 2, RXD (receive data) on pin 3, RTS (request to send) on pin 4, CTS (clear to send) on pin 5, DSR (data set ready) on pin 6, Signal Ground on pin 7, CD (carrier detect) on pin 8, and DTR (data terminal ready) on pin 20. This configuration of signals is the "standard serial connection" employed by most commercial modems today and will work with virtually all communications software even though it is not a full implementation of the EIA V.24 standard.

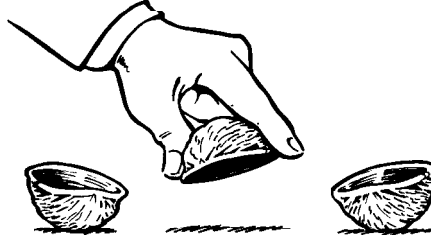
It is easy to imagine a situation where the link speed between the modems differs from the link speed between the PC and the modem. In this scenario, two features are required: data buffering and flow control. Most modems support RTS/CTS and DTR/CTS hardware flow control schemes, or will also operate with software (XON/XOFF) flow control. However, XON/OFF flow control is less useful when compared with hardware flow control, since characters in the data stream may be mistakenly interpreted as flow control commands. If XON/XOFF flow control is required, then XON/XOFF with command pass through is the most reliable method for synchronous links.

There are also standards in place that regulate any device that connects to the telephone network. The telco connections of modems must be designed to be compliant with FCC Part 68, UL 1459, and Canadian DOC. In order for the DAA [Data Access Arrangement-the term used to describe the interface between the modem and the telco) to be compliant with FCC Part 68, it must perform to the following standards:

- 1) it must provide a bidirectional line interface to the telco network on tip and ring
- 2) it must meet FCC requirements for RFI suppression
- 3) it must provide 10 megohms of DC isolation between tip and ring when the modem is on hook
- 4) it must provide 1500 volts of isolation and protection between the telco and the modem

No hidden charges!

One reason to say yes to KADAK's new RISC Kernel.



KADAK proudly announces AMX 3000 for the MIPS R3000 and compatible RISC processors. As with our other real-time multitasking kernels, AMX 86, AMX 386 and AMX 68000, you:

- pay no royalties
- pay the same price for your site license, no matter how many employees you have at one site or how many applications you produce
- pay nothing extra for Source Code.

KADAK has been leading the way in no-hidden-charges site licenses since 1980.

AMX 3000 That's one of the reasons why developers count on KADAK for the best in products, documentation and technical support.

For a **free** Demo Disk and your copy of our excellent AMX product description, contact us today. Phone: (604) 734-2776 Fax: (604) 734-8114

Count on KADAK.



KADAK Products Ltd. Setting real-time standards since 1978, 206-1847 West Broadway, Vancouver, BC, Canada, V6J1Y5

AMX is a trademark of KADAK Products Ltd. All trademarked names are the property of their respective owners.

#123

AVOCET SYSTEMS, INC.

<p>Intel 8051 8048 8085 8096</p> <p>Hitachi 64180 H8</p> <p>Western 65816</p> <p>Rockwell 6500 6502</p> <p>RCA 1802</p>	<div style="border: 1px solid black; padding: 10px; margin: 0 auto;"> <p>C and Pascal Compilers</p> <p>Simulators</p> <p>Assemblers</p> <p>In-Circuit Emulators</p> <p>EPROM Programmers</p> </div>	<p>Motorola 680x0 68300 68HC11 6800 6801 6802 6803 6804 6805 6809</p> <p>Zilog Z8 Z80 Z180</p> <p>TI 32010 32020</p>
--	--	---

Source For Quality Embedded Systems

1000 Main Street, P.O. Box 490, Rockport, ME 04856

For orders outside U.S., call (207) 236-9055

TELEX: 467710 Avocet CL • FAX: (207) 236-6713

Call today: 1-800-448-8500

#124

5) it must provide out-of-band frequency suppression.

While this is not an extreme or stringent list of requirements, they are necessary to protect the telephone network from outside devices.

Some modem designs implement the DAA with discrete components such as optoisolators and relays. This is often the most economical solution from a component standpoint, but has the problem of being uncertified. Since the Gemini modem may be built by a variety of different persons with a wide range of skill levels, I used a single-component DAA that is pretested to meet or exceed each of the requirements.

FILE TRANSFER PROTOCOLS

On top of the lower-layer data protocols, there is another set of standards that define data exchanges. These protocols fit quite nicely into the Session layer of the OSI seven-layer network model. The communication software running on the DTE is responsible for implementing these protocols, and is also responsible for the data packetizing and depacketizing at this level.

This short list is by no means complete, but it should give you an idea of what is going on behind the screen when you select one of these protocols.

File transfer protocols are conventions that define data-block size, data transfer mode (simplex, duplex, full duplex), handshaking, and error detection and correction.

First up is ASCII. This is barely a protocol at all since it has no provision for error detection. It only has one protocol provision: it can implement a crude form of flow control with XON/

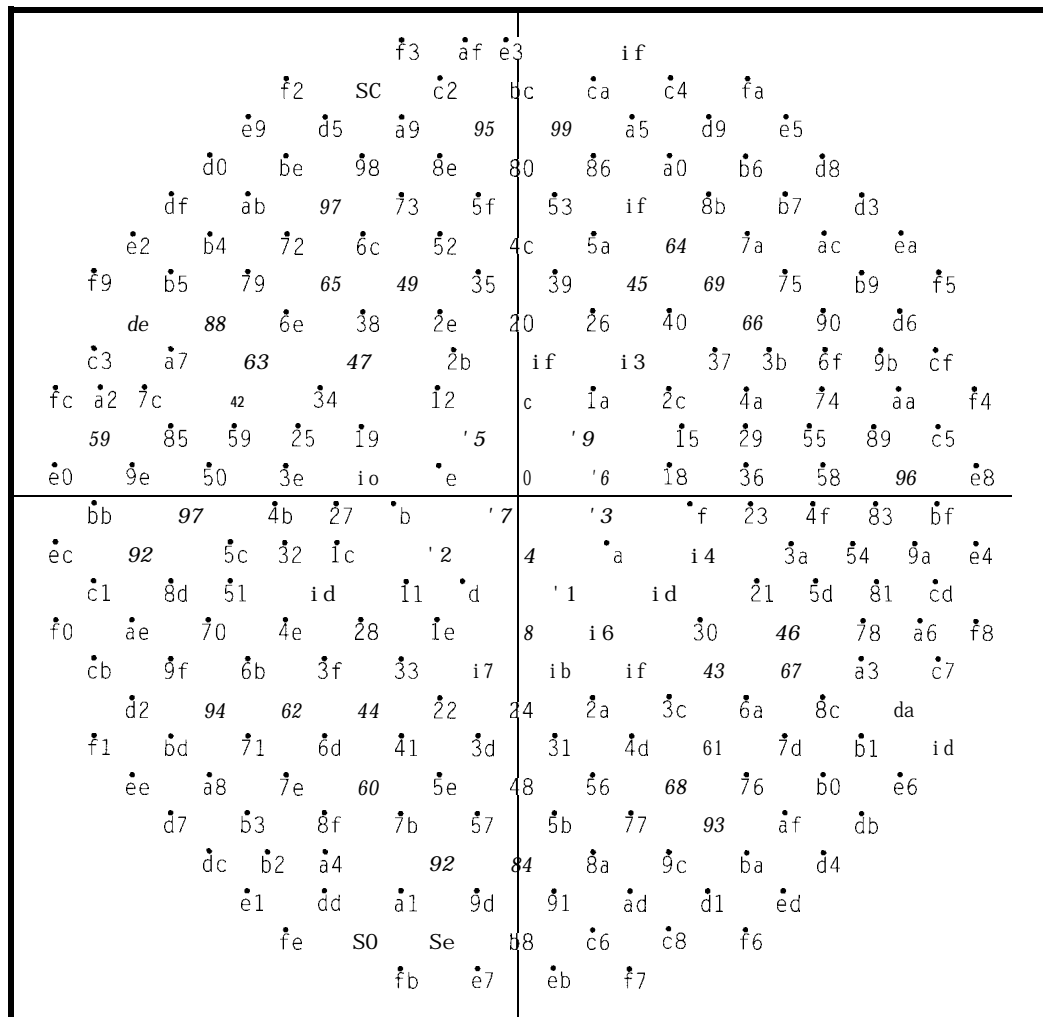


Figure 8—The signal constellation for a V.Terbo modem operating at 16,800 bps starts to get complicated

XOFF characters in the data stream. This “protocol” is not very useful and should only be used for raw text transfers.

Second up to bat is Kermit. This is a packet-oriented protocol that can use 7- or 8-bit characters, which is especially useful if data is being transferred between systems where one of the systems uses 7-bit characters and the other uses 8-bit characters. The protocol is able to perform this trick by manipulating the eighth bit using a technique known as “prefixing.”

Kermit is a protocol that continues to evolve. Newer versions of the protocol support such advanced features as compression and sliding windows. Sliding windows Kermit is a full-duplex protocol and it sends continuous packets of data while it simultaneously receives responses. One interesting feature of Kermit is that it supports a user interface called

Kermit Server that accepts high-level commands from the user.

Next up is XMODEM. This protocol is a half-duplex, block-oriented protocol based on a 128-byte data block. The original XMODEM uses simple 8-bit checksums for error detection, while almost all modern implementations use a CRC. Initial negotiation works as follows: The receiving DTE will make three attempts to contact the calling DTE to set up the CRC protocol. If this fails, the receiving DTE will fall back to using a checksum for error detection. Some implementations of this protocol are not entirely compatible with each other when negotiating the error detection method. When this happens, one of the DTEs will be “speaking CRC” and the other will be “speaking checksum.” In this situation, a number of CRC errors will be reported just before the communications

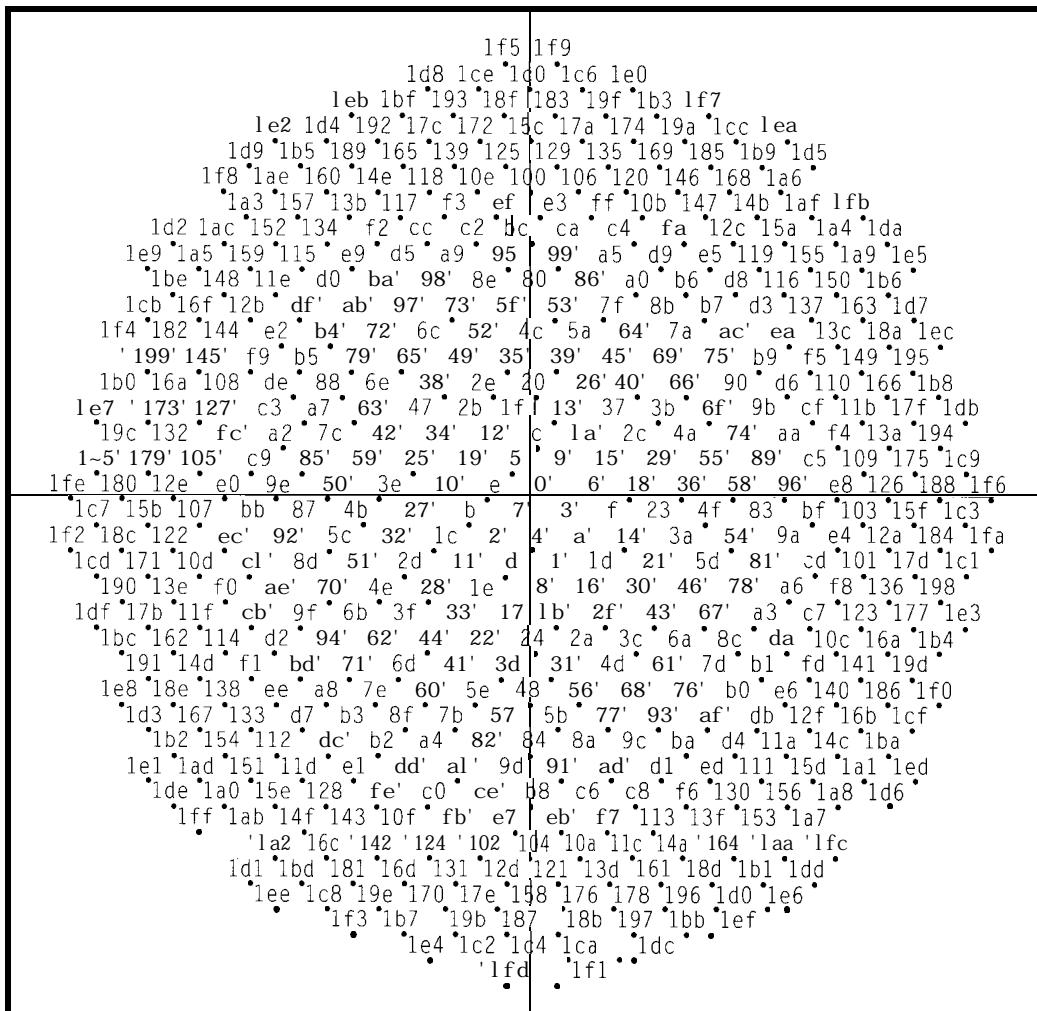


Figure 9—The signal constellation for a V.Terbo modem operating at 19,200 bps is downright scary.

program breaks the connection. XMODEM has some pretty tight timing constraints and may not work between systems if one of them is time-sharing between many users.

The next protocol is called YMODEM. YMODEM uses 1024-byte packets. Because of its relationship to XMODEM, it is sometimes known as 1K XMODEM. A derivative of the YMODEM protocol is called YMODEM Batch. This protocol adds the capability to transfer multiple files with a single command. YMODEM Batch accepts wildcard characters (such as * or ?) from the user when specifying files for transfer.

Another derivative of the YMODEM protocol is YMODEM-G. This protocol uses the same data packetizing methods of YMODEM with one very important difference: it does not use CRCs. The CRC is set to zero. This protocol was developed for

use in data circuits where the hardware performs error detection and recovery. Another difference particular to the YMODEM-G protocol is that it is a streaming protocol, meaning it will send data continuously until it is told to stop. Another YMODEM variant that works very well with high-speed error-correcting modems is YMODEM-G Batch, which is the same as YMODEM-G except it allows batch file transfers.

Yet another variant of the ever-popular XMODEM protocol is WXMDEM. This protocol implements sliding windows on XMODEM to give it full-duplex capabilities. This protocol can also send up to four consecutive packets before it will wait for an ACK or a NACK from the receiver, which makes this protocol particularly efficient. The protocol is especially adept at handling data network flow control, which makes it

useful for use on data networks.

Sealink is a close relative to WXMDEM. Sealink is another XMODEM derivative that uses sliding windows. It was designed for application over packet switched networks or when the data must take hops through satellite signal relay stations. This protocol uses 128-byte packets and can send up to six consecutive packets before it will wait for an ACK or a NACK.

As I said, this list is by no means complete, as each release of communications software typically includes these protocols as well as other newer variants of the protocols described here. Some new packages also include proprietary protocols that have unique features such as inbound virus scanning, automatic, on-the-fly, zipping and unzipping of files, and so forth.

CONCLUSIONS

The Gemini modem project implements some of the latest modem technology in the marketplace today. In order to understand how the project fits in the modem marketplace, I thought an explanation was necessary in relation to what modems really do when a manufacturer says its modem is compliant with <<fill in appropriate catch phrase here>>. Next month, I'll get to the hardware. □

Michael Swartzendruber is an engineer with experience in network and communications design and Windows and Macintosh programming. He is also a Technical Editor for the Computer Applications Journal.

- 407 Very Useful
- 408 Moderately Useful
- 409 Not Useful

DEPARTMENTS

50 Firmware Furnace

60 From the Bench

64 Silicon Update

70 Embedded Techniques

78 Patent Talk

85 ConneCTime

After This Brief Interruption: IRQs and INTs for the '386SX

Interrupts are tricky at best even for the most experienced embedded programmer. The original designers of the PC didn't make the task any easier. Ed does his best to try to clear the muddied water.

FIRMWARE FURNACE

Ed Nisley



Stop me if this hasn't happened to you.

When the clothes dryer buzzed I decided to take a break. The laundry room light went nova, so I detoured to the garage for a new bulb. While passing through the workshop I pocketed the outdoor pole lamp widget I'd fixed the previous evening and perched the laundry basket on the workbench.

I punched the garage door opener, found a bulb, walked down the driveway to install the widget, then retrieved the day's mailbox treasures. Mary leaned out the door to tell me the call was for me, so, passing the mail to her, I snagged the phone.

Several hours later Mary stuck her head in the office and asked "Why is the garage door open and what have you done with the laundry?"

We pros call this a "blown stack" although, to be fair, there are other interpretations....

The Firmware Development Board has enough hardware that we can investigate something that often goes unmentioned: what really happens when a hardware interrupt occurs? I'll concentrate on real mode and leave protected mode for a later column.

Our first task is to nail down the nomenclature. You have probably read about interrupt handlers, exceptions, traps, faults, IRQs, Ints, and vectors, but often the definitions are either vague or just plain wrong. Let's start with the bare silicon.

THE INSIDE STORY

Intel 80x86 CPUs handle interrupts from several sources, such as:

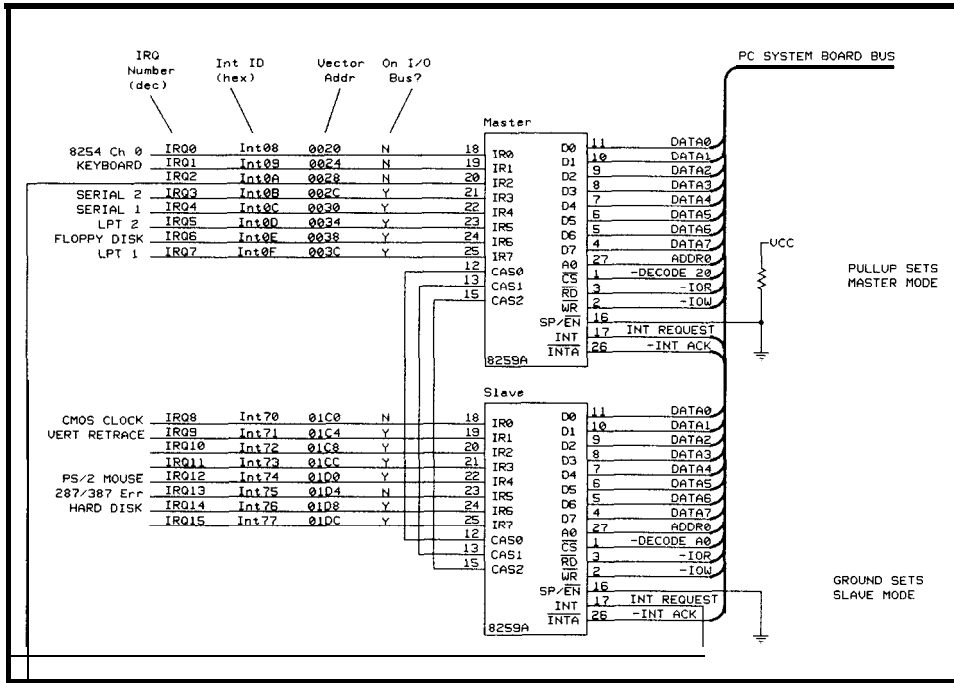


Figure 1—The interrupt circuitry boils down to a pair of 8259 Programmable Interrupt Controllers. Shown here are the 8259 Interrupt Request (IRQ) lines, the CPU Interrupt (INT) lines, and the vectors for each. The slave cascades into the master's IRQ2, so IRQ8 through IRQ15 have higher priorities than IRQ3 through IRQ7. The BIOS redirects IRQ9 to IRQ2 to maintain compatibility with older PCs; the pin that used to be IRQ2 on PCs became IRQ9 on ATs.

external events, instructions, and the dreaded divide-by-zero error. Fortunately, the CPU uses the same basic mechanism to cope with all these events.

External interrupts are caused by hardware outside the CPU that activates the INTR pin. The CPU hardware activates the INTA (Interrupt Acknowledge) pin and reads a single byte from the data bus to identify the interrupt. This mechanism can funnel up to 256 external interrupts through one pin, although PCs have only 15 external interrupts. The CPU ignores INTR when the Interrupt Enable bit (a.k.a., the Interrupt Flag or IF) in the Flags register is zero.

Unlike INTR, the NMI pin causes a nonmaskable interrupt that cannot be ignored. There is only one nonmaskable interrupt because the CPU does not read an ID byte from the data bus: NMI is hardwired as Int 02. The PC does have a way to shut off the NMI input, but it involves circuitry entirely external to the CPU.

Software interrupts occur when the CPU executes one of a class of instructions devoted specifically to causing them. The instructions have mnemonics like Int 10h, INTO, B O U N D, and so forth. These interrupts cannot be ignored, and, unlike hard-

ware interrupts, are entirely predictable: whenever the CPU executes the instruction, a software interrupt ensues.

Exceptions are interrupts caused by errors or conditions within the CPU. They are generally data-dependent, so a given instruction may not cause an exception every time, but they can be reproduced if you set all the hardware to the same state. Exceptions cannot be ignored.

Each interrupt is identified by a number from 00 through FF, which is called its Interrupt ID (or type, or just plain number). Interrupt IDs are generally shown in hex notation, although some sources use the decimal equivalents.

However, one reference managed to convert a hex Interrupt ID into decimal, then listed the decimal value as hex. Moral of the story: you need more than one book to cross-check things that seem out of place.

An interrupt's *raison d'être* (pardon my French) is diverting the CPU's attention from its current task and setting it to work on something else. The code associated with each interrupt is called the "interrupt handler" or "interrupt service routine" (ISR). Interrupt handlers are short routines that cope with whatever caused the interrupt and return to the interrupted task.

The link between an interrupt and its handler is the "interrupt vector" location holding the handler's address. The vector's address is the Interrupt ID multiplied by four: Int 08h uses the vector at address 0020h. The collection of all 256 vectors occupies 1024 bytes of

storage and is referred to as the Interrupt Vector Table.

Although "everyone knows" that the IVT starts with the Int 0 vector at address 0000:0000, the L I D T instruction (available on 80286 and later CPUs) sets the table's starting address and maximum length. The default is 0000:0000 and 1024 bytes. While operating systems may need to relocate the IVT, ordinary programs have little need for such shenanigans.

The next piece of the puzzle involves returning from the interrupt handler. In principle, the interrupted program should resume execution as though the handler never got control apart from the time required to complete the handler.

For all external and software interrupts, the CPU pushes the Flags register, the CS register, and the address of the next instruction. If the interrupt was caused by the INTR pin, the CPU clears the Interrupt Flag to suppress all further external interrupts. Although NMI interrupts do not clear IF, the CPU ignores the INTR pin until the NMI handler ends with an I RET instruction.

Exception interrupts come in three flavors: faults, traps, and aborts. You need to spend some time with the references to understand the differences, but, in real mode, you'll see only the first two unless you're

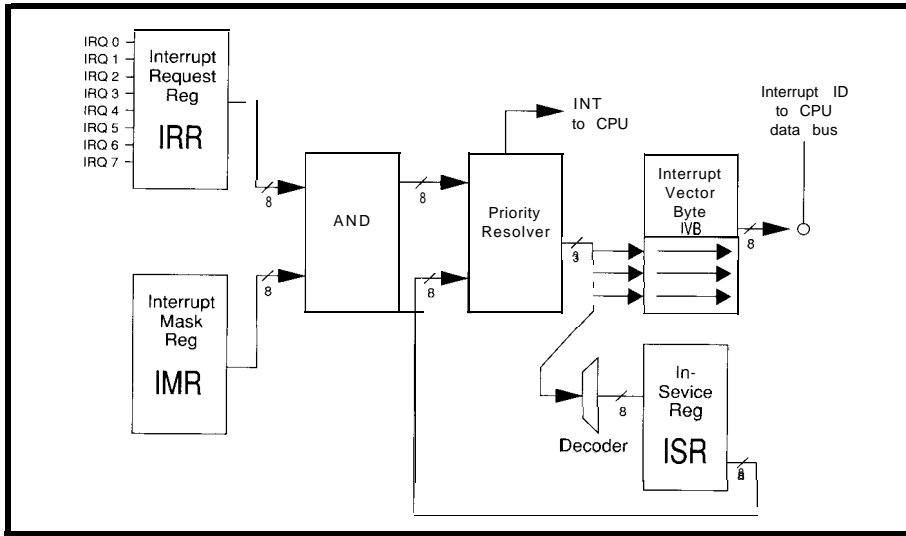


Figure 2— The 8259 Programmable Interrupt Controller has a variety of modes and settings, this diagram shows it set up for normal PC operation. A rising edge on an IRQ line sets an IRR bit. If that bit is not masked off by the IMR, the priority resolver decides if it has a higher priority than any of the bits set in the ISR. If so, it activates INT. When the CPU acks the interrupt, the 8259 combines the IVB with the IRQ number producing an Interrupt ID and turns on the appropriate ISR bit. An EOI command from the CPU resets the ISR and IRR bits for the next IRQ edge.

exceedingly unlucky. For our purposes, faults push the address of the *current* instruction (the one that failed) and traps push the address of the *next* instruction, just like external and software interrupts.

In principle, you can “fix up” the condition that caused a fault and reexecute the failed instruction successfully, although this can be difficult to pull off in actual practice. A trap, on the other hand, is over and done with by the time your handler gets control, so you should continue with the next instruction.

Finally, the **IRET** instruction restores the CPU’s Flags, CS, and IP registers from the stack, which is exactly what we need regardless of what caused the interrupt. The stacked Flags register restores IF, which, if this was an external interrupt, will re-enable further external interrupts.

To summarize: when the CPU detects an interrupt, it pushes the Flags, CS, and (usually incremented) IP registers on the stack, computes the interrupt vector address from the Interrupt ID (which may be supplied by external hardware, internal hardwiring, or the instruction), fetches the starting address of the interrupt handler, and transfers control to it.

When the handler is done, an **IRET** instruction restores the registers and the CPU picks up where it left off.

THE REST OF THE STORY

The Intel 8259 Programmable Interrupt Controller is the key to

understanding PC interrupts. As with any Intel peripheral with “programmable” in its name, the 8259 is a maze of modes, options, and control bits. I’ll concentrate on how it behaves in a PC and leave the rest for an evening of data book spelunking.

As shown in Figure 1, the system board has a pair of 8259s in tandem to provide 15 external interrupts. Of course, the separate chips have long since been subsumed into an LSI package along with all the other CPU support circuitry, but the PC compatibility barnacles dictate how the hardware must behave.

Word of warning: if you’re contemplating

anything at all out of the ordinary, remember that the 8259 data sheet does not apply to the LSI chips inside your PC! You need the real specs on the actual LSI marvel you’re using, as modes or functions unused in “normal” PC applications may not work correctly. Give them a go, but don’t be surprised if it doesn’t work quite right.

Each 8259 is an I/O device with two internal addresses selected by the A0 address line. Unlike the 8254 circuitry in the last column, you cannot use 16-bit I/O operations with the 8259. The master 8259 is at addresses 0020 and 0021, while the slave is at 00A0 and 00A1.

For what it’s worth, the terms “master” and “slave” seem to have fallen out of favor lately. I’ll continue to use them here because that’s how the 8259 doc is worded. “Primary” and “secondary” may be more politically correct.

Int	CPU	BIOS	Hardware
00	Divide By Zero-Error		
01	Step/Debug		
02	NMI pin		
03	Breakpoint		
04	INT0 (Overflow)		
05	Bound Check	Print Screen	
06	Invalid Op code		
07	x87 Not Available		
08	Double Exception		IRQ0
09	287 Seg Overrun		IRQ1
0A	Invalid TSS *		IRQ2/IRQ9
0B	Segment not present		IRQ3
0C	Stack fault		IRQ4
0D	General protection		IRQ5
0E	Page fault		IRQ6
0F	Reserved		IRQ7
10	x87 Error (-ERR pin)	Video functions	
11	Alignment check *	System info functions	
12	Reserved	Get memory size	
13	Reserved	Disk I/O functions	
14	Reserved	Serial I/O functions	
15	Reserved	System functions	
16	Reserved	Keyboard functions	
17	Reserved	Printer functions	
18	Reserved	Cassette BASIC (!)	
19	Reserved	Bootstrap loader	
1A	Reserved	Time functions	
1B	Reserved	Ctrl-Break handler	
1C	Reserved	System timertick	
1D	Reserved	Video param tables	
1E	Reserved	Diskette param tables	
1F	Reserved	8x8 graphic font	

Figure 3— Interrupt IDs between 00 and 1F were to be reserved for CPU exceptions, but the BIOS interrupt hardware took several of them. CPU exceptions marked with an asterisk occur only in protected mode, which means BIOS conflicts are irrelevant and conflicts can be avoided.

Listing 1—Interrupt handlers are written in assembly, or C in some cases. This handler *turns on* a parallel port bit, records the timer value in an array, sends an *EOI* to the 8259, and turns off the parallel port bit. The *INTERRUPT* macro is the key to using a standard C function as an interrupt handler.

```

/* Timer 0 hardware interrupt handler          */
/* Assumes interrupt on master controller      */
INTERRUPT(Timer0Handler) HandlerT0() {

    outp(SYNC_ADDR,inp(SYNC_ADDR) | 0x01);

    if (!(Response[0][RESP_SET]++){
        Response[0][RESP_NOW] = ReadTimer(0,I8254_BASE);
        /*fetch times*/

    outp(I8259A,NS_EOI);

    outp(SYNC_ADDR,inp(SYNC_ADDR) & ~0x01);

    return;
}

```

The BIOS initializes the 8259s for normal PC interrupts, which is all we need right now. Figure 2 is an approxi-

mate diagram of an 8259 minus all the control logic and special case after the BIOS is finished setting it up.

Listing 2—This macro saves the CPU registers and the *?temp* scratch variable before calling an interrupt handler. The macro restores everything before returning. The *_SPACE_* macro prevents the preprocessor from gluing the *?temp* string onto the *MOV* instruction.

```

#define INT_PROLOGUE 30          /* size of prologue code */

#define INT_ENTRY(fn) (&fn-INT_PROLOGUE) /* start of prologue */

#define _SPACE_

#define INTERRUPT(fn) asm {\
fn
    PUSH AX \
    PUSH BX \
    PUSH CX \
    PUSH DX \
    PUSH SI \
    PUSH DI \
    PUSH ES \
    PUSH DS \
    PUSH CS \
    POP  DS \
    MOV  AX,?temp \
    PUSH AX \
    CALL fn+INT_PROLOGUE \
    POP  AX \
    MOV  _SPACE_?temp,AX \
    POP  DS \
    POP  ES \
    POP  DI \
    POP  SI \
    POP  DX \
    POP  cx \
    POP  BX \
    POP  AX \
    IRET \
} \

#undef _SPACE_

```

Each 8259 has eight Interrupt Request inputs called IRQ0 through IRQ7, but the slave's IRQ inputs are called IRQ8 through IRQ 15. By convention, IRQs are numbered in decimal. The numbers are simply labels and have no mystical significance; as you'll see later, the slave's IRQ numbers are particularly inauspicious.

A rising edge on any IRQ input constitutes an interrupt request, so the 8259 turns on the corresponding Interrupt Request Register bit. You can force the 8259 to ignore any combination of IRQ inputs by turning on the appropriate Interrupt Mask Register bits (a 1 bit *disables* the interrupt).

The 8259 then activates the INT output, which raises the CPU's INTR input, which in turn triggers a hardware interrupt as described above if the CPU's Interrupt Flag is set (here, a 1 bit *enables* the interrupt). The CPU blips the INTA pulse once to tell the 8259 to resolve the highest priority interrupt.

If two or more IRR bits are on simultaneously, the 8259 figures out which one is "highest priority" and ignores the rest. The priority rules can be complex and may change on the fly, but in PCs the rule is "lowest numbered IRQ wins."

A considerable amount of time may elapse between an IRR bit going active and the CPU responding. If additional IRR bits become active, the highest priority interrupt at the time of the first INTA pulse is recognized. The other IRR bits remain active and will cause additional interrupts as they become the highest priority inputs.

In any event, the winning IRR bit turns on the corresponding In-Service Register (ISR, not to be confounded with the Interrupt Service Routine described above) bit to indicate that the CPU has acknowledged the interrupt request. Several ISR bits can be active at one time, each indicating that a lower-priority interrupt has been interrupted by a higher-priority interrupt.

So far, so good!

The CPU blips the INTA line a second time to tell the 8259 to put the

Interrupt ID on the data bus. This is not a standard I/O operation because the -IOR and -IOW lines remain inactive. Because the INTA and INTR lines do not appear on the PC's I/O Expansion bus, you cannot put an 8259 on an I/O card to get more vectored interrupts.

The Interrupt ID is a single byte made up of five high-order bits from the 8259's Interrupt Vector Byte register and three low-order bits identifying the winning ISR bit. The BIOS loads a different value into each 8259's IVB register during the power-up sequence: the master gets 0x08 and the slave gets 0x70.

Once the CPU reads the Interrupt ID, it proceeds as I described earlier: pushes registers, turns off IF, converts the ID to a RAM address, fetches the interrupt vector, and starts the interrupt handler.

Meanwhile, the 8259 can accept new interrupts on any IRQ inputs that don't have active IRR bits. If an IRQ goes active, the 8259 compares its priority to the highest ISR bit and recognizes only higher-priority interrupts by turning on the INT output. The CPU is free to respond or ignore its INTR input depending on whether the Interrupt Flag is set or not.

The interrupt handler routine must write an End Of Interrupt command to the 8259 to clear the ISR

Listing 3—If NEST_1 is defined this handler will send EOI commands to the 8259 and turn on the CPU's IF, allowing other interrupts to gain control during this handler. If the timer produced another IRQ before this code returns, the CPU pushes the registers and starts executing the handler from the top.

```

/* Timer 1 hardware interrupt handler          */
/* Assumes interrupt on secondary controller */

INTERRUPT(Timer1Handler) HandlerT1() {

    outp(SYNC_ADDR,inp(SYNC_ADDR) | 0x02);

#ifdef NEST_1
    outp(I8259B,NS_EOI);           /* tell slave we are done */
    outp(I8259A,NS_EOI);           /* tell master we are done */
    enable();                       /* & allow other interrupts */
#endif

    if (!(Response[1][RESP_SET]++){
        Response[1][RESP_NOW]=ReadTimer(1,I8254_BASE); /* fetch times */
    }

#ifdef NEST_1
    outp(I8259B,NS_EOI);           /* tell slave we are done */
    outp(I8259A,NS_EOI);           /* tell master we are done */
#endif

    outp(SYNC_ADDR,inp(SYNC_ADDR) & ~0x02);

    return;
}

```

bit and reenables lower-priority interrupts. The BIOS sets things up so that what's called a nonspecific EOI will reset the highest-priority ISR bit. You can also issue a specific EOI to reset a different bit, but this isn't usually desirable.

The nonspecific EOI command is 0x20. Yes, folks, that's identical to

both the master 8259's base I/O address and the first address of its interrupt vectors. This is why named constants are such a good idea... you cannot tell what 0x20 is or will do just by staring at it.

Interrupts on the slave 8259 work slightly different. As you can see from Figure 1, the slave's INT output connects to the master's IRQ2 input. A slave IRQ activates its INT output (subject to priority resolution),

which triggers a master IRQ2 (again, after resolving priorities). When the CPU responds to the master's INT, the slave provides the Interrupt ID.

In this case there are two ISR bits active: one in the slave 8259 identifying the actual interrupt and ISR bit 2 in the master 8259. The interrupt handler (forgive me for not calling it an ISR) must send an EOI command to each 8259 to clear both ISR bits.

COLLISION ALARM!

The value of the master 8259's Interrupt Vector Byte register is probably the second-worst idea in the whole PC kingdom. It's a classic case of what happens when you ignore what's printed in the data books...read and heed!

As I described above, exception interrupts are generated by events inside the CPU. Intel reserved Interrupt IDs 00 through 1F for those exceptions, but the 8086/8 did not use all 32 IDs (nor does the 80486, for that matter). For whatever reason, IBM's engineers wrote the BIOS code to use many of those reserved interrupts, with the inevitable result that later

Interrupt Handler Timing Exerciser
Firmware Furnace #35 — Ed Nisley
(c) 1993 Computer Applications Journal

Timers are not synchronized
Initial alarm time set successfully
Preloading response array... done
Setting interrupt vectors... done
Starting timers... running
Enabling interrupts... active

Interrupt response times in 139 ns ticks:

Timer	Current	Minimum	Average	Maximum
0	102	96	97	545
1	114	108	108	597
2	40	35	35	524

Max stack used: 00C8
Press any key to clear min/max values

Figure 4—The interrupt handlers record the elapsed time between the IRQ signal and the 8254 timer-latch command. The program summarizes those values displaying how different handlers respond. Displaying the response time table requires an ANSI terminal emulator.

CPUs collided with existing PC usage.

My guess is that IBM used the "reserved" interrupts because they didn't want to use any more RAM than necessary. Remember that this design dates back to the days when 16K of RAM was standard and 64K was large. Packing the interrupts into the lower part of the table left a big gap for other stuff near the top. As the saying goes, it seemed like a good idea at the time.

Figure 3 summarizes the conflicts between the CPU, BIOS, and hardware interrupt IDs. Many of the CPU exceptions occur only in protected mode where the BIOS is irrelevant, but the remainder pose a thorny problem because the conflicts are hard-coded into BIOS and DOS.

However, you can resolve the hardware conflicts by writing a different Interrupt Vector Byte into the master 8259. Although this is generally done only by protected-mode operating systems, we embedded systems types may find it a handy trick. The standard alternate seems to be 0x50, so if you're going to be nonstandard you may as well pick the standard method.

I'll leave these machinations as exercises for the reader, because there are entirely too many details in this column already. In any event, you should peruse the data books before trying anything fancy.

THREE TIMERS TICKING

Having stunned the subject, let's back up and run it over. The demo program for this column uses the Firmware Development Board's 8254 timer to produce three external interrupts at known rates and report on the handler response times. As before, the code is written in Micro-C and can be loaded with either the diskette boot routine or MON86's serial HEX transfer command.

The program puts all three 8254 timers into Mode 2 to produce a blip when the count reaches zero. I picked a 5-ms period for the timers to simplify scope sync: 200 traces per second is easy on the eyes and allows plenty of time for the handlers to finish what they're doing.



WIN AN EE08
FREE!
SEE DETAILS AT
RIGHT

TO REGISTER FOR THE FREE EE08, CALL US AND MENTION THIS AD. OR FAX/MAIL YOUR NAME, ADDRESS AND PHONE NUMBER TO US BY 6/30/93. THE WINNER WILL BE SELECTED AT RANDOM ON 7/15/93 AND WILL RECEIVE AN EE08 WITH 1Mb SRAM, A \$279 VALUE.

FLEXIBLE EPROM EMULATORS

- 10 DAY MONEY-BACK GUARANTEE, 90 DAY REPAIR/REPLACE WARRANTY
- BOTH TARGET AND HOST CAN READ AND WRITE TO EMULATOR
- SOURCE AND OBJECT CODE SUPPLIED FOR DIRECT R/W
- BUILT-IN ARBITOR ALLOWS REAL-TIME R/W ACCESS
- FIELD UP-GRADABLE SRAM (TO 4Mb)
- ACCESS TIMES DOWN TO 50ns

EE08 WITHOUT SRAM other models start at \$129

\$259

Voice or FAX (214) 272-9392

CALL OR FAX TODAY FOR MORE INFORMATION ON THE EE08 AND OUR COMPLETE LINE OF EPROM EMULATORS!

Technical Solutions

**PO BOX 462101
GARLAND, TX 75046-2101**

#125



Melts the Price of 8051 ICE

Only \$851 for **iceMASTER-PE**

The world's most innovative emulator for members of the 8051 family is incredibly affordable. Metalink's unique Advanced Emulation Technology (AET, patent pending) delivers the best possible emulator value for engineers, consultants and students.

AET is a revolutionary design architecture that provides more features with 75% fewer components, smaller board space and lower cost. Emulator and probe electronics are integrated in a single package only 3" by 4".

MetaLink also delivers leading-edge customer service, including a 30 day money back guarantee, 10 day trial for qualified customers, rental plans and free technical support.

- Up to 40 MHz Operation
- 64K Emulation Code Memory
- 64K External Data Memory
- 128K Hardware Breakpoints
- 16K Trace Memory
- Transparent Trace (View Trace While Executing)
- Real Time & Nonintrusive
- Symbolic & Source level Debug
- Built-In Self-Test

- SUPPORTS 8031/8032'S
- SUPPORTS 8XC751/8XC752'S
- Windowed User Interface
- Serial link to Any PC
- Motro Cross Assembler



Call today for FREE DEMO DISK!

(800) METAICE (800) 638-2423

MetaLink Corporation P.O. Box 1329 Chandler, Az 85244-1329
Phone: (602) 9260797 FAX: (602) 926-1198

#126

interrupts off because the CPU shuts them off when it accepts the interrupt, so the other handlers don't have a chance.

Listing 3 shows the IRQIO handler. Compiling with NEST_1 defined causes the code to send E0Is and enable interrupts immediately after setting the parallel port bit. Photo 2 shows the result: HandlerT1() is interrupted by HandlerT2(). Even though IRQ15 has a lower priority than IRQIO, the E0Is tell the 8259 that the IRQIO is finished.

An implication of this is that a second IRQIO interrupt would start another copy of HandlerT1()! As far as the 8259 is concerned, HandlerT1() is history because it sent an E0I command to shut off that In-Service Bit. In our case, the handler is finished long before the next IRQIO pulse, but it's something to bear in mind if you have fast interrupts and slow handlers.

The source code this month includes the routines that I wrapped around the BIOS interrupt handlers to activate the parallel port bits for those pictures. This trick can be handy even for DOS programmers. Well, low-level DOS programmers like you folks, anyway....

DIGITAL READOUT

Even if you don't have a scope you can still experiment with interrupt handlers using this month's code. Each of the handlers reads back the current value of its 8254 timer channel and stores it in a global array. Once each second, the main line code calculates the minimum, maximum, and average values of the times for each channel and displays the results as shown in Figure 4.

Recall that 8254 timers in Mode 2 count down to zero, generate an output blip [which we wired to an I/O bus IRQ line), then reload the count value and continue to tick. The difference between the maximum and current timer values is just the number of ticks since the reload, which is precisely the elapsed time since INTR was activated.

The minimum and maximum values shown in Figure 4 give you an

idea of how fast your handler can possibly respond to an interrupt and how long it may take under less-than-ideal conditions. These figures change as the program continues to run.

The handler triggered by IRQ15 is obviously faster than the one on IRQ5. Listing 4 shows the reason: it's written in assembly language inside a standard Micro-C function. Unlike the other two handlers, this one doesn't have to save and restore all the CPU registers, so it gets started faster. In round numbers, it's three times faster than the competition...but at least that much harder to write and understand, too.

The main line code monitors the total stack used by the routines, and you can see the value change as the program runs. This is particularly noticeable with the nested interrupts, as the combinations use an unpredictable amount of stack space.

Unlike my cerebral stack, it's essentially impossible to blow the stack in this program because Micro-C's setup code puts the stack at the far

end of the 64K segment. It's worthwhile to check the stack occasionally, but this is a big change from the 8031's cramped quarters!

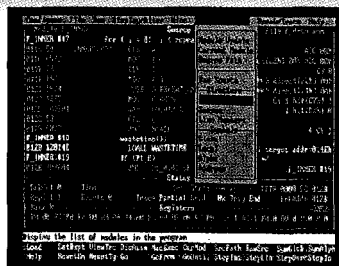
There are a few other tricks buried in the code, but this should get you started. I recommend spending some time with interrupt handlers so you know what your system is capable of... and what it can't do no matter how good you are!

RELEASE NOTES

You've probably already noticed the bug in the Serinit() function presented in the March issue: I didn't save and restore the BP register. Normally, that error clobbers the calling routine's local variables and makes itself readily apparent, but the main() function didn't have any locals! A revised version is already on the BBS.

I've recoded the support routines in assembler and moved them into the firmdev.asm library file. Use SLIB to update TINY.LIB on your system.

8051 68HC11 COP8 68HC05



iceMASTER™ Improved User Interface Features

Rental And 1 O-Day Trials Available

- **iceMASTER** delivers **productivity**: easy to learn, easy to use and fast!
- **Hyperlinked On-line help** guides you through the emulation process.
- **iceMASTER is FAST!** The 115.2K baud serial link keeps typical download times to under 3 seconds using a standard COMM port!
- Broad support of derivative devices.
- Flexible user interface: you can completely configure the windows for size, content, location and color.
- **iceMASTER** is convenient! It connects easily to your PC, requires no disassembly, nor does it take up any expansion slots. It works on any PC (DOS or OS/2), Micro Channel or EISA. Even Laptops!
- Supports source level debug (C and PL/M) and source level trace. 4K trace buffer with advanced searching and filtering capabilities.
- Now virtual memory and mouse support!



Call today for FREE DEMO DISK!
Call today to ask about FREE 8051 Micro Assembler!
(800) METAICE (800) 638-2423



Metalink Corporation P.O. Box 1329 Chandler, Az 85244-1329 Phone: (602) 9260797 FAX: (602) 926-1198 TELEX: 4998050MTLNC

Listing 4- This interrupt handler is written in assembly language to reduce the overhead required to save and restore the CPU state. The result is a much faster handler, but it's much harder to understand.

```

/* Timer 2 hardware interrupt handler. Assumes interrupt on */
/* secondary controller. CAUTION -all constants are hard */
/* coded in here and registers are saved manually. Micro-C */
/* inserts PUSH BP and MOV BP,SP before the first line */
/* of code. */

HandlerT2() {
asm {
    PUSH AX          save bystanders
    PUSH DX
*
    MOV AL,#$80      latch Timer 2
    MOV DX,#$030
    OUT DX,AL
*
    MOV DX,#$0378    flag startup
    IN AL,DX
    OR AL,#$04
    OUT DX,AL
*
    MOV AX,CS        set up data segment addressing
    MOV DS,AX
*
    PUSH SI          save more bystanders
    PUSH DS
*
    MOV DX,#$030     read the latched LSB
    IN AL,DX
    JMP <Punt4
Punt4
*
    MOV AH,AL
    IN AL,DX          and the MSB
    XCHG AH,AL       rearrange them
    MOV DX,AX        save for later
*
    MOV SI,#Response+(2*7*2) aim at our slice of Response
*
    MOV AX,4[SI]     fetch _SET element
    AND AX,AX        previous entry processed
    JNZ Done         nonzero says skip this one
*
    MOV 2[SI],DX     stash it away
    INC ASWORD(4[SI]) account for this sample
*
Done    MOV AL,$20    send the EOIs
    OUT $20,AL      ... to master
    OUT $A0,AL      and slave
*
    MOV DX,$0378    flag shutdown
    IN AL,DX
    AND AL,$FB
    OUT DX,AL
*
    POP DS          restore bystanders
    POP SI
    POP DX
    POP AX
    POP BP          saved at function entry
*
    IRET           preempt normal return
}
}

```

The BBS files for this issue include all the batch files I used to create the code for these columns. You'll need to tweak them for your system, but that's all part of the learning experience, right? Drop a note on the BBS if you have troubles; somebody else has probably already flattened them for you.

Dave Dunfield tells me he's putting together a diskette of DOS-less Micro-C routines, including a simple way to put the boot loader on a diskette, interrupt-driven serial I/O, windowing functions, and keyboard and other drivers. It sounds handy for embedded PC projects.

Next month we get to something that's generated a lot of interest on the BBS: adding memory to the Firmware Development Board. I'd planned a simple battery-backed RAM, but folks on the BBS convinced me to cover EEPROMs and EPROMs as well. It'll take a few columns, but you'll like the results! 📁

Ed Nisley, as Nisley Micro Engineering, makes small computers do amazing things. He's also a member of the Computer Applications Journal's engineering staff. You may reach him on CompuServe at 74065,1363 or through the Circuit Cellar BBS.

SOFTWARE

Software for this article is available from the Circuit Cellar BBS and on Software On Disk for this issue. Please see the end of "ConnecTime" in this issue for downloading and ordering information.

CONTACT

Dunfield Development Systems
P.O. Box 31044
Nepean, Ontario K2B8S8
Canada
(613) 256-5620

IRS

410 Very Useful
411 Moderately Useful
412 Not Useful

Component Selection, Inspection, Rejection

FROM THE BENCH

Jeff Bachiochi



opping by the supermarket is always an unpredictable experience.

Through the years I have become rather chummy with a few of the local supermarket's shopping carts. They remind me of the seven dwarfs. There's Shaky, Bumpy, Topsy, Bull, Rattles, Squeaky, and Rusty. You may have had the opportunity to urge one of these fellows into service yourself. Or, if you've done much shopping during rush hour's clamor of clattering carts, you may have learned to appreciate the masking quality of (elevator) music.

As I meander down the aisles looking for the few items on my shopping list, I can't help but think about psychology of manufacturers. We can no longer trust manufacturers to price the largest quantities of product at the least per-unit price. They've caught on that most shoppers will grab the large size container, assuming it to be the best deal without bothering to compare the unit price.

Shopping is becoming more complicated. Take milk, for instance. There is whole milk, 2%, 1%, skim, evaporated, condensed, and nonfat dry milk. Knowing you need milk just isn't good enough anymore. That's how it is with some circuit components—you have to pick and choose based on the application.

THREE EXTERNAL COMPONENTS

As I demonstrated last month using the MAX639, simple step-down switching regulators can improve battery life. Now I'd like to discuss choosing components and circuit layout. Let's see how the three external components (inductor, diode, and capacitor) will affect the operation of this regulator.

A capacitor is a capacitor, right? Not necessarily. A connection is a connection, right? Think again. Both component selection and PC board layout can be critical in making a circuit perform properly.

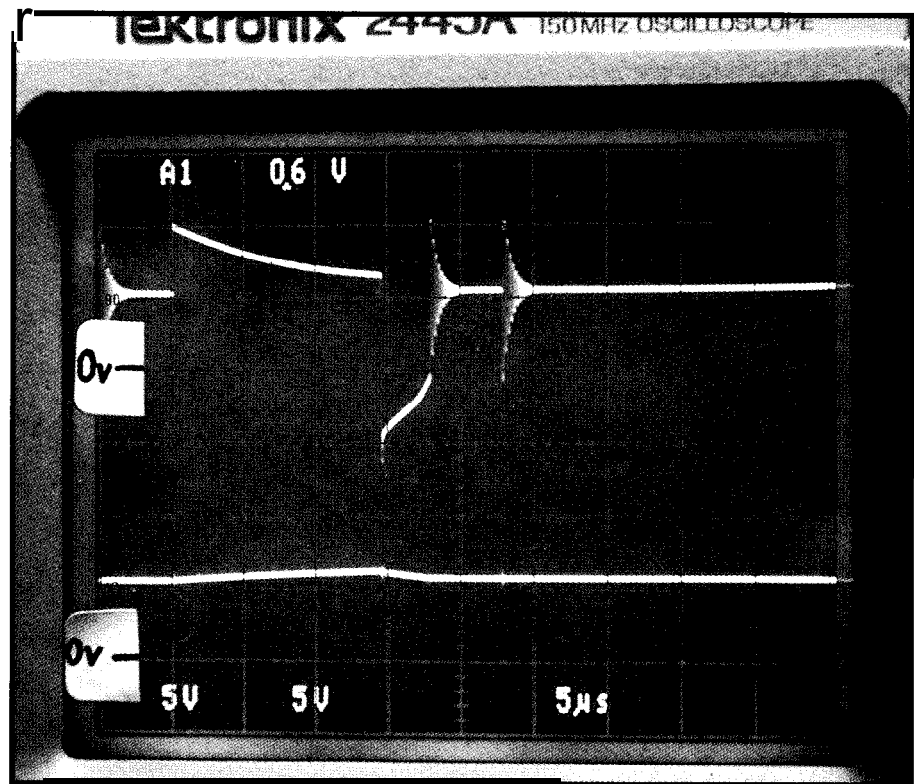


Photo 1—In the switching regulator circuit, you must select an inductor that can provide the peak current without going into saturation.

INDUCTOR SELECTION

The inductor is the most important component in the switching regulator circuit. It is accountable for the storage and conveyance of the excess voltage, which is the difference between V_{in} and V_{out} . In the MAX639's circuit, the inductor is charged when an internal FET acting like a switch between V_{in} and the inductor is turned on, and it discharges into the capacitor (and the load) when the FET is turned off. A gated internal oscillator drives the FET at a frequency based on V_{in} . An internal comparator which observes V_{out} starts and stops the oscillator as necessary to keep V_{out} within specification. The regulator's efficiency, though based on many components, is primarily affected by the inductor. Four factors must be considered for ensuring proper inductor selection: peak current rating, inductance value, series resistance, and physical size.

Do not choose RF chokes or air-core inductors; they don't have high peak-current ratings. Ferrite bobbins, toroids, and pot cores all work well, however the bobbin inductors are the smallest in size. Peak current must be limited to 600 mA for the MAX639, which is an $I_{L,}$ about one half of I_{peak} or 300 mA for an ideal circuit. Resistive and diode losses account for a lower maximum output current. The inductor chosen must be able to provide the peak current without going into saturation, as shown in Photo 1.

The minimum inductor value can be calculated using the formula $L=50/I_{peak}$, but the inductor used must not be less than 100 μ H. Higher inductor values will increase the efficiency of the switcher, but then it may not be able to supply peak current throughout the total V_{in} range.

Series resistance (DC resistance) is an inverse function of the wire size used to wind the coil. The larger the wire size, the smaller the series resistance. A value of 0.5 ohms (or less) is good, and generally speaking, the smaller the better. The resistance value of the inductor is added to the internal FET's on resistance, along with the coils inductance value, and the sum is what determines the real

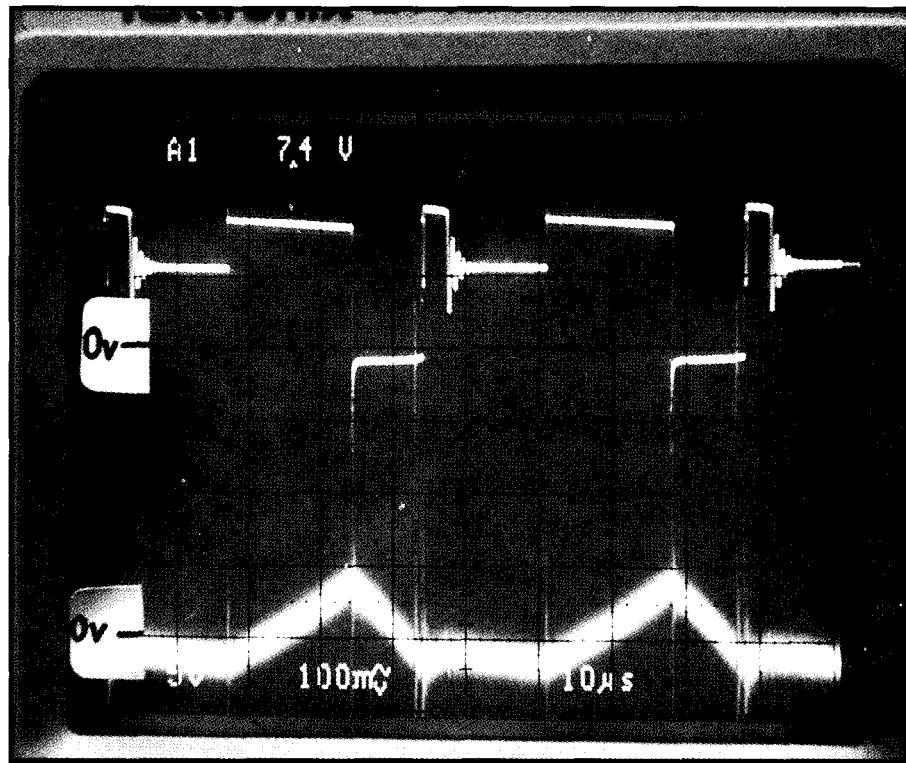


Photo 2—When the diodes used in the switching power supply circuit are too slow, they create heavy losses in the transference of energy.

turn-on time. If this LR time constant is greater than that of the oscillator's t_{on} , maximum current cannot be transferred.

The size of the inductor is determined by the materials in its core, its

inductance, and its wire size. The physical size grows with increasing peak current, decreasing series resistance, and/or increasing inductance. Shielded coils are available to cut down on radiated EMI.

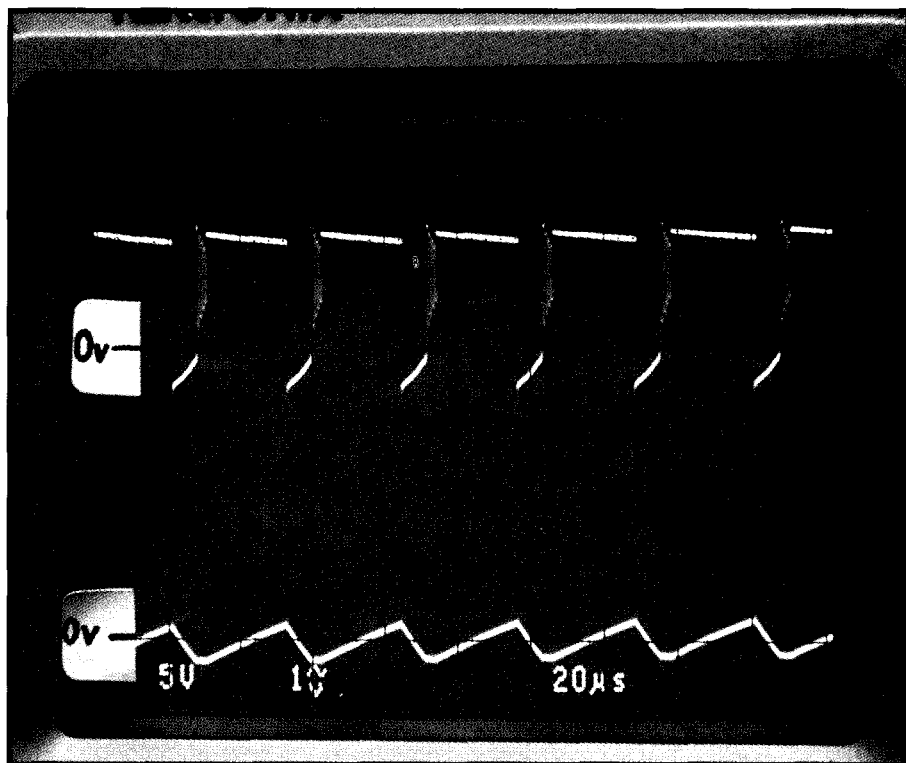


Photo 3—When selecting a capacitor for the switching power supply circuit, the lower the equivalent series resistance (ESR), the better. When the ESR is high, the output ripple goes up.

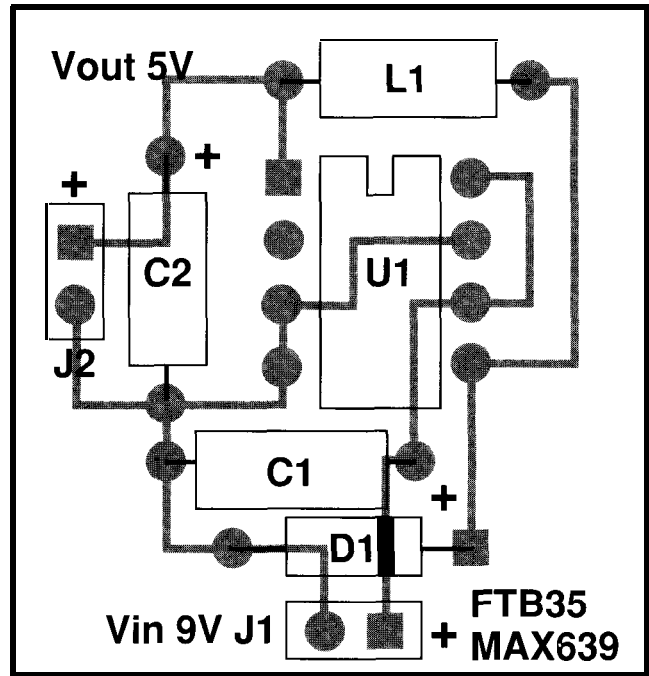
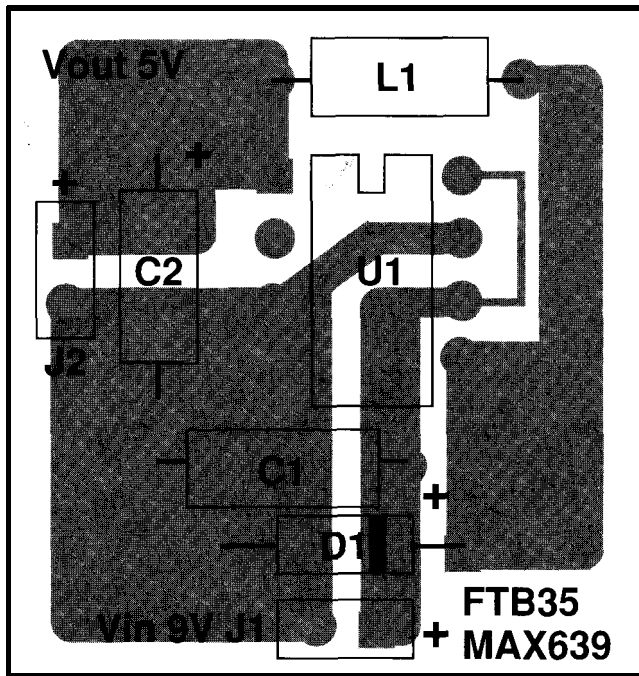


Figure 1—Printed circuit board layout is every bit as important as component selection. By using planes whenever possible and fat traces for high-current paths, you can minimize both radiated and power supply noise. A well-laid-out board (left) uses lots of copper between component pins. A poorly laid-out board (right) uses narrow traces and daisy chains for connections that will ultimately carry a lot of current.

DIODE SELECTION

The trick here is selecting a diode that will allow as much of the coils' stored energies as possible to be transferred into the capacitor. This implies the loss due to the diode drop must be minimized. The 1N5817 Schottky diode can handle the necessary peak currents, and it also has a minimal drop (0.45 volts). When designing for smaller currents, a 1N4148 can be used. Other diodes like 1N4000 series are too slow and will create heavy losses in the transference of energies, as seen in Photo 2.

CAPACITOR SELECTION

The output capacitor is used in a low-pass filter arrangement to reject the high-frequency switching of the PFM (pulse frequency modulation) regulator. Using PFM has a slight advantage over PWM (pulse width modulation) in that it can shut off entirely, which means saving quiescent current in micro power applications. The most critical specification in selecting the appropriate capacitor is the equivalent series resistance (ESR). The ESR is responsible for a capacitor's inability to completely reject all output voltage ripple. Aluminum electrolytics have greatest

change in ESR over temperature and should be avoided when the minimum operating temperatures will be below 0°C.

ESR represents all the energy losses of the capacitor including lead resistance, termination losses, dissipation in the dielectric material, and foil resistance. The energy loss results in internal heating (which negatively affects component life). High ESR caps also exhibit higher impedance, which leads to higher ripple current.

The ESR is given in many parts catalogs for electrolytics. These values range from many ohms to fractions of an ohm. Tantalums have much lower ESRs than aluminum electrolytics. Photo 3 shows the effect of high ESR on output ripple.

CHOOSING THE CORRECT PATH

The best component choices won't help a poor circuit layout. The size and placement of circuit traces play an important role in reducing radiated noise. Keeping the resistance in high-current paths to a minimum, will reduce ground bounce.

Take the time to analyze the circuit and determine the high-current paths even before attempting parts placement. The high current paths

must be kept as short as possible.

Ground bounce can occur when high currents flow through nonzero ohm PCB traces. This creates a difference in potential between points where the components are connected to the ground plane. To minimize ground bounce, pick a logical point—usually the ground pin of the regulator—and connect all component grounds to it individually as opposed to daisy chaining the ground trace all over the glass.

Place the input capacitor as close to the IC as possible. If V_{out} is to be something other than 5 volts (MAX639 is preset for 5 volts without the need of feedback), then the resistor-divider feedback network must be kept as close to the VFB input as possible. See Figure 1 for contrasting layout techniques.

SHOP AROUND

Knowing how each component is used in a circuit is essential to choosing the right one. Choosing the right component sometimes takes more information than a supplier has presented for a given part. When information is missing, ask questions; if the questions can't be answered, call the manufacturer. Most manufactur-

er's representatives will fax you data sheets immediately. Don't forget to ask the IC manufacturer for available application notes based on the part you are interested in. Often the app note will give actual part numbers for individual discretres. No sense reinventing the wheel. □

Jeff Bachiochi (pronounced "BAH-key-AH-key") is an electrical engineer on the Computer Applications Journal's engineering staff. His background includes product design and manufacturing.

SOURCES

MAX639:

Maxim Integrated Products (408) 737-7600

Other Switching Regulators:

Linear Technology (408) 432-1900
National Semiconductor (408) 721-5000
Raytheon (415) 968-9211
Texas Instruments (714) 660-1200
Teledyne (415) 968-9241

SMT Inductors:

CoilTronics (305) 781-8900
Sumida Electric (708) 956-0666
Toko (708) 297-0070

Inductors:

Caddell-Burns (516) 746-2310
Renco (516) 586-5566
Toko (708) 297-0070
Wilco (317) 293-9300

SMT Capacitors:

Elna (714) 761-8600
Kemet (803) 963-6300
Matsuo Electronics (714) 969-2491
NEC (415) 960-6000
Panasonic (201) 392-4818
Thomson Passive Components (818) 887-1010

IRS

413 Very Useful
414 Moderately Useful
415 Not Useful

What is C_thru_ROM?

ROM Your Borland or Microsoft C/C++ Code.

C_thru_ROM is the complete ROM development software tool kit. It lets you run Microsoft and Borland C and C++ programs on an embedded 80x86 CPU without using DOS or a BIOS.

C_thru_ROM saves you money. There are no DOS or BIOS royalties to pay for your embedded systems.

C_thru_ROM is complete! It includes the following and much more:

- *Supports Borland's Turbo Debugger.
- *Remote Code View style source level debugger.
- ROMable startup code brings CPU up from cold boot.
- ROMable library in source code.
- *Flexible 80x86 Locator.

COMPLETE PACKAGE ONLY \$435. 30-DAY MONEY BACK GUARANTEE.

Datalight®

107 N. OLYMPIC, SUITE 201 • ARLINGTON, WA 98223 • (206) 435-8086 • FAX: (206) 435-0253

#128

Deep in the Hundred Acre Wood...

Our engineers produce fine software and hardware designs that are hand-hewn with the cutting edge of modern technology.

If you've been told "It can't be done..."
Get a second opinion!

1-800-245-2885



HUNDRED ACRE CONSULTING
5301 Longley Lane, Suite D-144
Reno NV 89511-1805
Tel: +1-702-829-9700
Fax: +1-702-829-9926
Email: info@pooh.com

#129

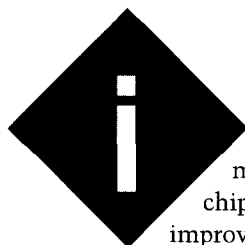
The Ultimate RAM?

The Quest for Core Continues

The more things change, the more they stay the same. Just when you thought core memory had been relegated to the museum, along comes a new idea based on an old concept.

SILICON UPDATE

Tom Cantrell



In most respects, modern micro chips are a gigantic improvement from the dinosaur-like computers of yesteryear. However, you may be surprised to learn that today's silicon whippersnappers still can't match all the tricks of their venerable elders.

Younger readers may hear and use terms like "core memory" or "core dump" without realizing that, before the advent of semiconductor ICs, memory was actually constructed of tiny magnetic "cores" which were intricately threaded with addressing and sense wires as shown in Figure 1.

While core can't match the speed,

bit density, and price of semiconductor memory, it has one big advantage common to all magnetic media—the fact that they are nonvolatile.

Since the invention of solid-state memory, much effort has been expended trying to reinvent the nonvolatility of core. First came ROM, then EPROMs, and more recently EEPROM and Flash Memory, not to mention a variety of battery-backed SRAM schemes. Yet, each of these approaches suffers from compromises that cramp designers' style to one degree or another.

Founded in 1984, Ramtron has been struggling to marry IC technology with "core-like magnetic structures." Now, they are finally starting to deliver the aptly named "Ferroelectric RAM" or FRAM.

CRYSTAL POWER

The Ramtron concept is disarmingly simple in principle. The idea is to sandwich a ferroelectric crystalline layer between the usual transistor and metal layers of a conventional IC. As shown in Figure 2, the key is that the crystal, once polarized by an applied electric field, retains a stable alignment. Note that since this is a "ferro-

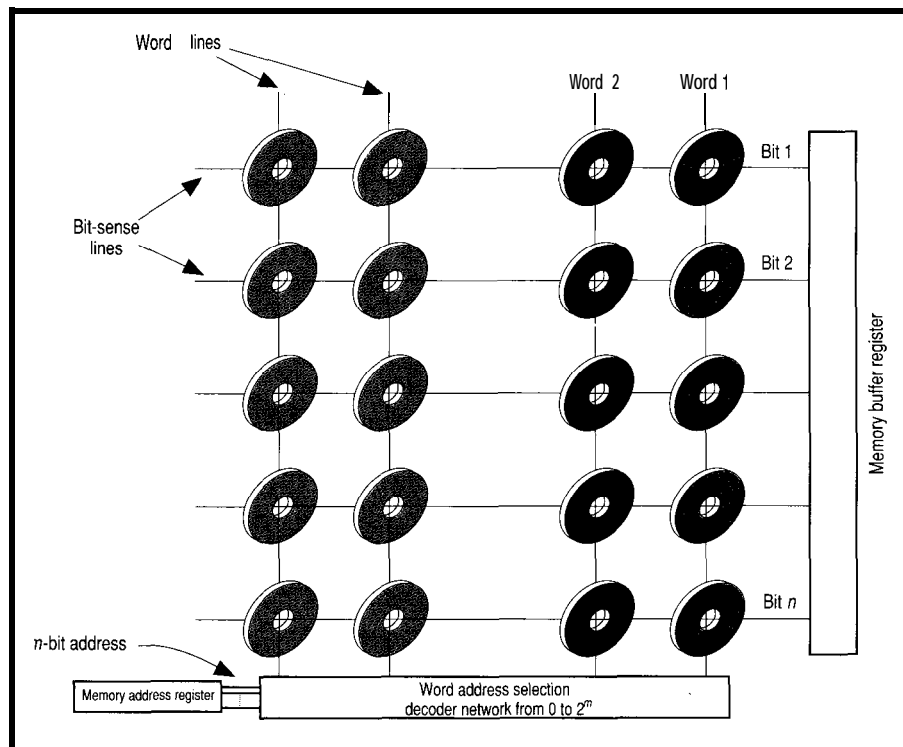


Figure 1—Core memory is actually constructed of tiny magnetic "cores" which are intricately threaded with addressing and sense wires. Once the center of any mainframe computer, core isn't found very often any more.

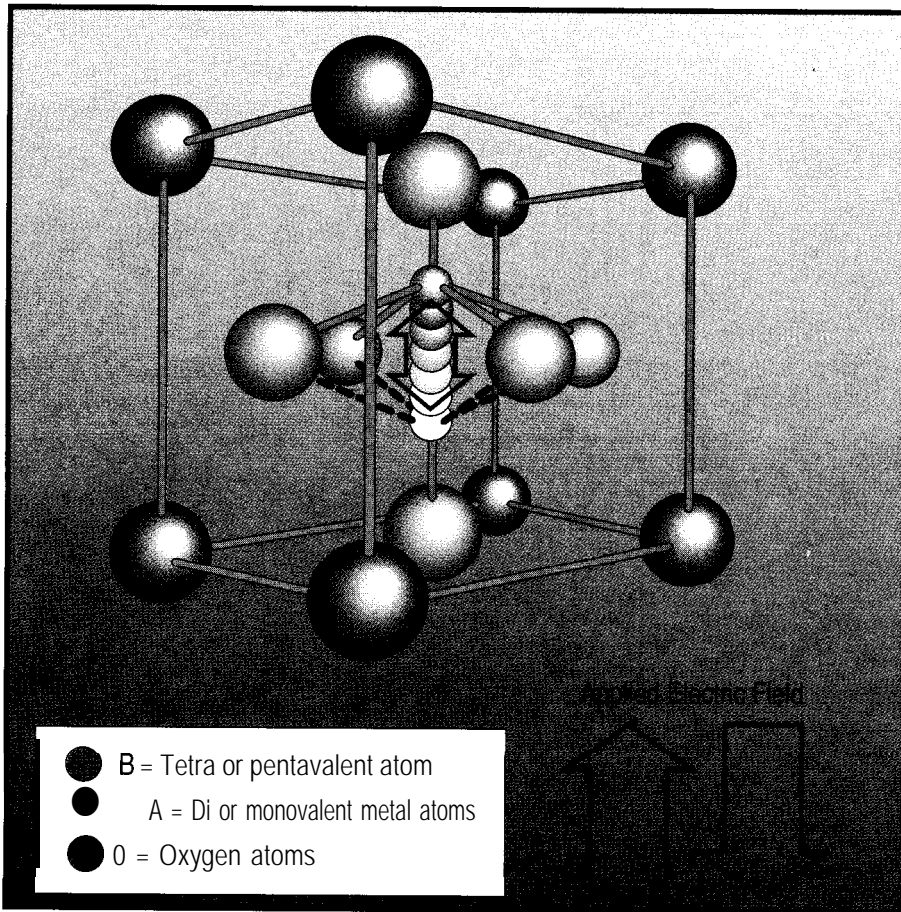


Figure 2-In a ferroelectric RAM, or FRAM, a ferroelectric crystalline layer is sandwiched between the usual transistor and metal layers of a conventional IC. Once polarized by an applied electric field, the crystal retains a stable alignment. Since the FRAM is a ferroelectric, and not a ferromagnetic, device, it isn't sensitive to stray magnetic fields.

electric" (not "ferromagnetic") effect, FRAMs, unlike a disk, aren't sensitive to stray magnetic fields.

Though it sounds easy, Ramtron could tell you that actually building a working FRAM has been a rather torturous process. Ramtron is now shipping 4-kbit FRAM chips, after 8 years of experimental devices, and tweaking the process. Later this year, 64-kbit devices are expected to become available.

First-generation FRAM memory cells consist of two "Ferro" elements polarized in opposite directions (Figure 3) and read via a differential-sense amplifier that minimizes common mode variations. Ramtron hopes to perfect a single-element memory cell in the future.

Like DRAMs (and core as well), FRAM data access is destructive in that the common charge applied to the two "Ferro" elements is opposite to

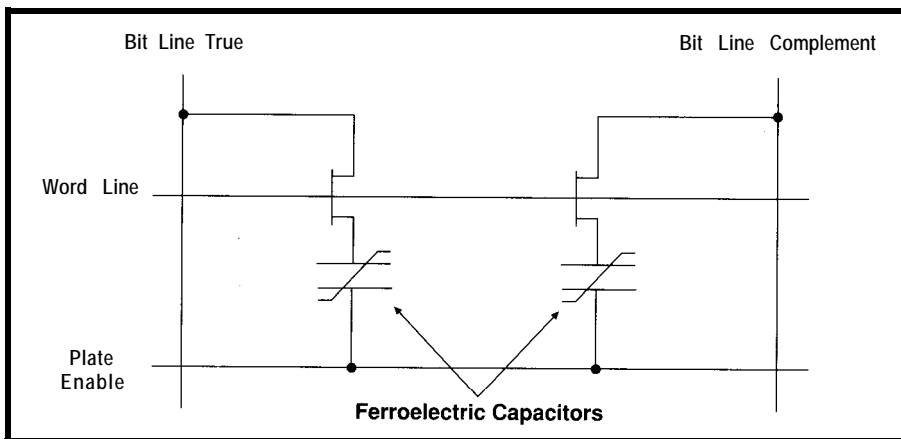


Figure 3—First-generation FRAM memory cells consist of two "ferro" elements polarized in opposite directions and read via a differential-sense amplifier that minimizes common mode variations. A single-element memory cell may be perfected in the near future.

one of them. So, FRAMs also require a "precharge" cycle to rewrite the contents. The impact on system design is that the FRAM "cycle" time (500 ns) is longer than the "access" time (250 ns).

SPEED/PIN TRADEOFF

The FRAM is offered in both serial (FM24C04) and parallel (FM1208S) bus versions (Figure 4), both of which contain a 512-byte storage array. The choice allows the designer to reduce pin count and size at the expense of access time.

The FM24C04, like many serial EEPROMs, uses a bidirectional, two-wire, clocked, serial protocol on SDA and SCL. The WP (Write Protect) pin disables writes to the top 256 bytes for the ultimate in protection. Taking advantage of the multidrop capabilities of the protocol, the address lines (A1 and A2) allow up to four devices to be daisy chained.

To write a byte, the clocked, serial protocol (Figure 5) requires shifting an 8-bit slave address (MSB first), followed by an 8-bit word address, and eight bits of data. Variations of the protocol support byte and block reads and writes. With clock (SCL) frequency limited to 100 kHz maximum, a full-chip write takes 47 ms or about 100 us per byte. Though this might seem like a leisurely pace, keep in mind that this is nearly 10 times faster than EEPROMs, which suffer from that technology's Achilles heel—slow write times.

If speed is important, the parallel FM1208S is the ticket. With its SRAM-like interface, the 250-ns access and 500-ns cycle time capabilities of the "Ferro" cell are fully exploited.

Notice I said "SRAM-like." On the '1208S, /CE is really more like an ALE (Address Latch Enable) or AS (Address Strobe) signal in that the

address inputs are internally latched by the device. The good news is this is quite helpful if connecting to a processor that uses a multiplexed bus, because the address latch required for an SRAM can be eliminated. On the other hand, it does require that addresses are valid before the /CE signal becomes active, which may not be the case for an SRAM socket—especially those that use the scheme in which /CE is permanently grounded.

Remember that SRAMs don't suffer from the access/cycle time differential that FRAM devices do, which might require an extra wait state for the latter. However, it is only a problem for "back-to-back" accesses. Often, the software design or the application guarantees a couple of hundred nanoseconds between accesses, thus eliminating precharge time as a constraint.

you certainly don't have to worry about power consumption if you are replacing an SRAM with an FRAM. Active power consumption is less than

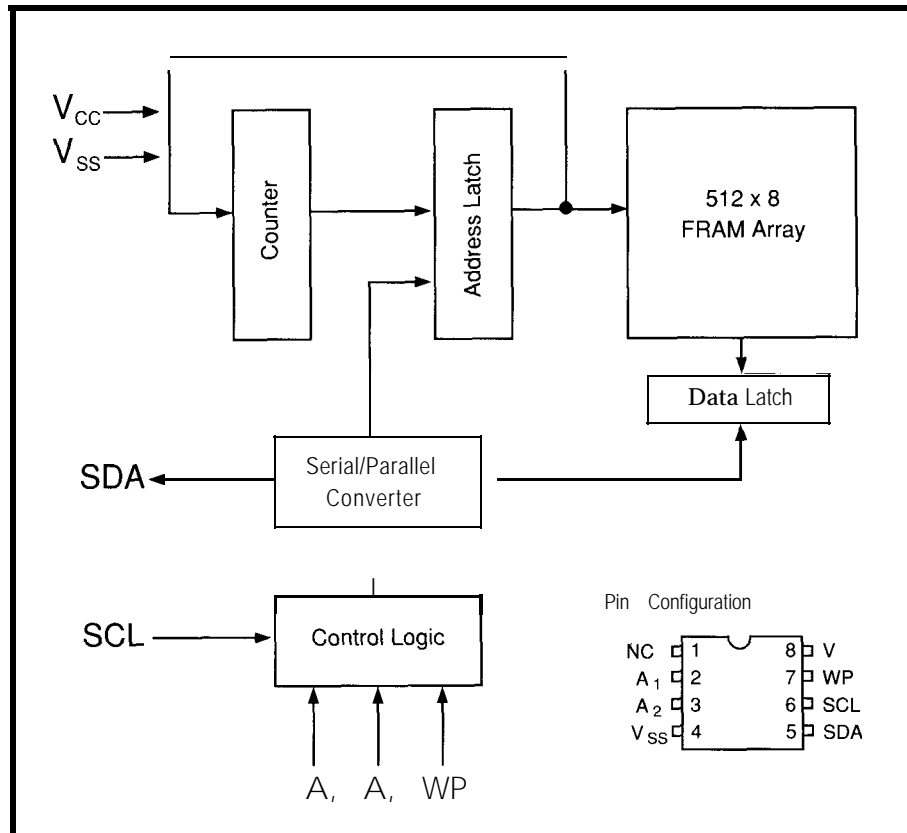


Figure 4a—The serial-based FM24C04 uses the same SCL and SDA lines used by many serial EEPROMs. Two address lines allow up to four parts to be daisy-chained together.

RTC-HC 1 1

PROCESSOR BOARD

Offering an exceptional value in a single-board embedded controller, Micromint's RTC-HC1 1 combines all of the most-asked-for features into a compact 3.5" x 4.5" package at a reasonable price. Featuring the popular Motorola MC68HC118-bit microcontroller, the RTC-HC1 1 gives you up to 21 lines of TTL-compatible I/O; an 8-bit, 8-channel analog-to-digital converter; two serial ports; a real-time clock/calendar with battery backup; 512 bytes of nonvolatile EEPROM; and up to 64K of on-board RAM or EPROM, 32K of which can be battery backed.

Software development can be done directly on the RTC-HC1 1 target system using BASIC-1 1, an extremely efficient integer BASIC interpreter with dedicated keywords for I/O port, ND converter, timer, interrupts, and EEPROM support. In addition, a flexible configuration system allows a BASIC program to be saved in the on-board, battery-backed static RAM, and then automatically executed on power-up. Micromint also offers several hardware and software options for the RTC-HC1 1 including the full line of RTC-series expansion boards as well as an assembler, ROM monitor, and a C language cross-compiler.

Additional features include:

- Asynchronous serial port with full-duplex RS-232 and half-duplex RS-485 drivers
- 1-MHz synchronous serial port
- CPU watchdog security
- 5-volt-only operation
- RTC stacking expansion bus

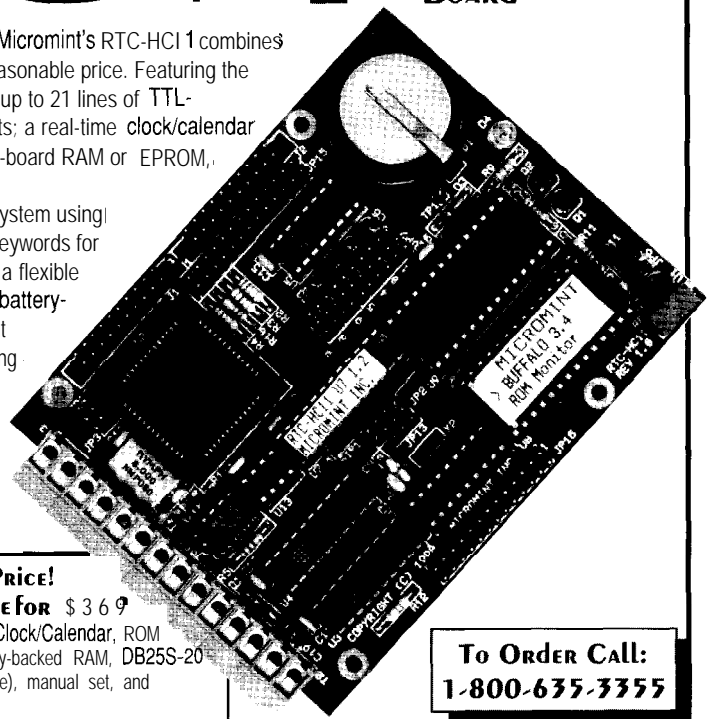
Special Development System Price!

RTCHC11-DEV A \$437 value for \$369

Board w/8-bit ADC, EEPROM, 8K RAM, Clock/Calendar, ROM monitor, BASIC-11 in EPROM, 32K battery-backed RAM, DB25S-20 serial cable, utilities diskette (PC compatible), manual set, and HCTerm software.

Other configurations starting at \$239

**To Order Call:
1-800-635-3355**



MICROMINT, INC. 4 Park Street, Vernon, CT 06066. (203) 871-6170. Fax (203) 872-2204
in Europe: (44) 0285658122 • in Canada: (514) 336-9426 • in Australia: (02) 888-6401. Distributor Inquiries Invited!

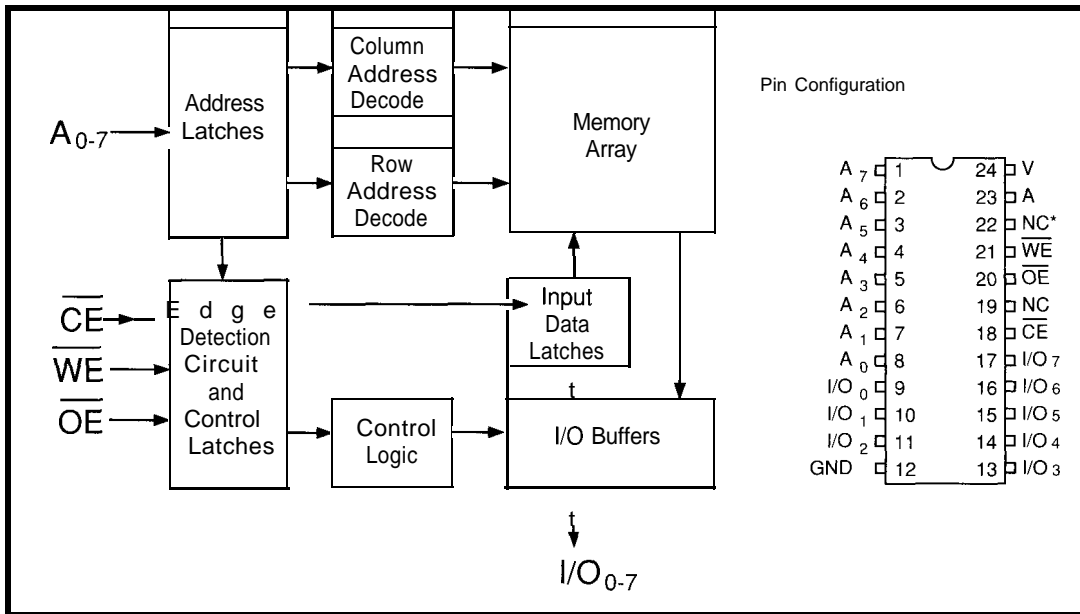


Figure 4b—The parallel-bus FM1208S contains the same 512-byte storage array as the FM24C04, but offers an SRAM-like interface with 250-ns access and 500-ns cycle times.

most SRAMs and, not surprisingly, standby power is almost negligible at 100 μ A for the '1208S and only 25 μ A for the '24C04.

CHOOSE YOUR POISON

So, does the FRAM lay claim to title of the ultimate RAM? Does it do

everything as well as the venerable core? Unfortunately, the answer is no.

The FRAM rightfully touts superiority to EEPROMs in both write-cycle speed, and "endurance." Endurance refers to the fact that EEPROMs can typically only take ten thousand to a million writes before the silicon

hopes it can, and expects to, boost FRAM endurance in the future.

To make matters worse, unlike an EEPROM, the FRAM endurance limit applies to both reads and writes. This limits the applicability of the FRAM for general-purpose code storage since even a lowly 8-bit micro-fetching

equivalent of semi-charge tunneling causing oxide layer breakdown-renders them feeble-minded. However, "superiority" doesn't mean "perfection" and the FRAM suffers from its own endurance limit-of 100 million cycles. Ramtron

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
- ALL GERBER FORMATS

- FULL TIME MODEM
- GERBER PHOTO PLOTTING

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

Express
Circuits

1150 Foster Street • P.O. Box 58
Industrial Park Road
Wilkesboro, NC 28697

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

COMPARING NONVOLATILE ALTERNATIVES

ROM

Though featuring high density and low power, ROM is suffering in this era of complex (and thus buggy) software and "run-it-up-the-flagpole" products. Both of these design philosophies increase the likelihood of having to eat a bunch of useless ROMs-and their mask charges. ROM is still appropriate for "stable data" applications such as in character generators or dictionaries used in high-volume, consumer products.

EPROM

EPROM remains the workhorse of nonvolatility. Besides its high density and low-power, aggressive competition among vendors is leading to ever-improving speeds and an undisputed price per bit leadership position. Windowed EPROMs, due to package expense and the difficulty in erasing, are increasingly relegated to lab work. Meanwhile, OTP (One Time Program) EPROMs in low-cost plastic packages are largely replacing ROMs by offering the escape hatch to last second changes with little extra cost.

EEPROM

Great hopes surround EEPROM up to and including the ultimate replacement of disk drives. In addition to the well-known limitations of slow write times and endurance limits, these devices need high power supplies with output voltages that are only required for the EEPROM programming procedure. The cost of these programming supplies shouldn't be overlooked, especially considering that they are unused for most of the time. In principle, EEPROMs could replace EPROMs since the cell-size is similar and, at least in the case of bulk-erase (rather than block- or byte-erase), little extra logic is required. However, continuing production difficulties (the process is apparently quite tricky) are keeping the price per bit comparatively high, thus prohibiting any such crossover in the near term.

BATTERY-BACKED SRAM

The clear winner in terms of accessibility with fast access/cycle time and symmetrical read/writes, and no endurance limits to boot. Downsides to SRAM include circuit-board real estate requirements, and a higher cost

per bit at low densities, where the battery cost becomes dominant. Also, don't forget that the battery will ultimately die-what will your equipment do then and where will you be?

If your design needs a real-time clock, then the battery becomes a given, favoring the use of SRAM. Indeed, Dallas Semiconductor and Epson offer units that integrate RTC, SRAM, power control, and the battery in one module which serves to ease the space requirements.

DRAM

The high-frequency and high-current refresh requirements of traditional DRAMs ruled them out for nonvolatile applications. But now, the big DRAM suppliers are introducing specialized chips tailored for battery operation with low-current self-refresh capability. Hitachi offers 512Kx8 and 256Kx16 devices that automatically refresh themselves whenever RAS is held low for more than 100 μ s and need only 200 μ A to stay alive.

SHADOWS/HYBRIDS

These refer to devices that combine multiple technologies to achieve the best of both worlds. For instance, Simtek offers a 16-kbit "NV SRAM" that combines 2Kx8 each of SRAM and EEPROM. The result is the speed and accessibility of SRAM during normal operation, with EEPROM called into play across power cycles.

FRAM

The main "gotcha" is read-cycle endurance limits, limiting FRAM to "data-only" use. Otherwise, for low density applications, the FRAM has advantages over both EEPROM (fast write, low write power, write endurance) and battery-backed SRAM (size, cost and life span). Note that FRAM specs a data life of 10 years, but it isn't very problematic since it is 10 years after the last write, not 10 years total.

ONE LAST NOTE

A special caution is in order when using writable technologies-remember that they are writable! Make sure your design can tolerate power transients or software crashes, lest unintended writes lead to IC amnesia.

perhaps a million opcodes per second-would kill the FRAM in under two minutes.

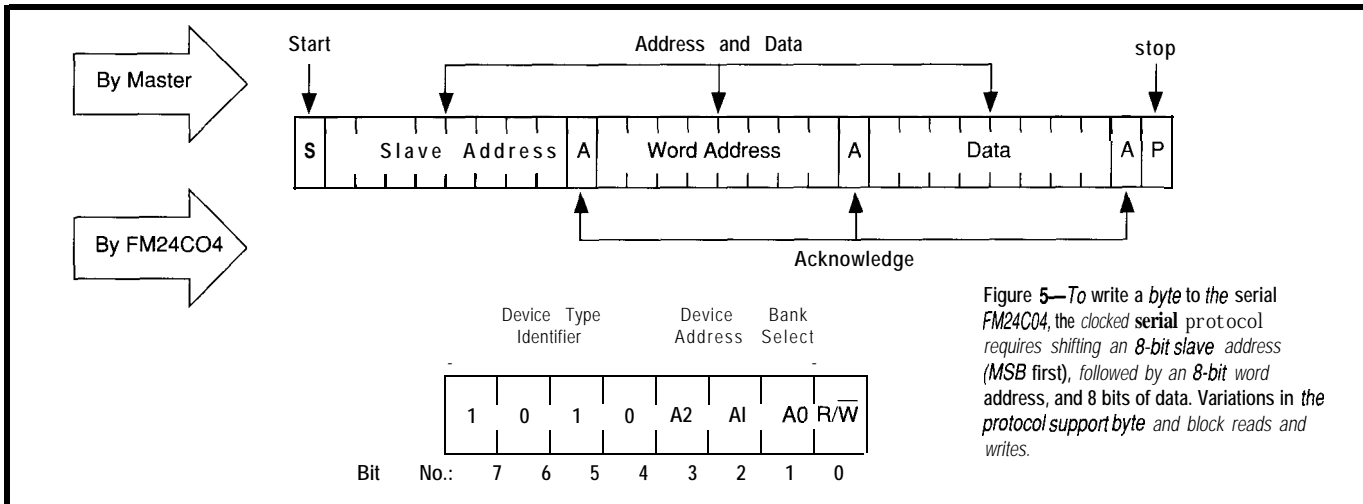
Despite these flaws, you shouldn't dismiss FRAM out-of-hand as the next "bubble memory" (a core pretender from 10 years ago that never quite got off the ground) or yet another "technology of tomorrow." In fact, compar-

ing FRAM to other "nonvolatile" solutions (see the sidebar), it is especially well-suited for applications requiring fast writes and low power.

One example might be the emerging "RF ID tags" which are designed as a replacement for bar codes. Interestingly, one application driving this technology development is

"cattle ID." This becomes understandable when considering the prospect of convincing an irate beast to hold still for repeated prodding with a bar code wand. Instead, the RF tag can be read with a radio receiver that is safely out of kicking range.

The key point is the tag doesn't even need a battery, since the RF generated



by the reader not only carries the data, but can also power the tag! FRAM is a good fit for this application, thanks to lower read/write power than EEPROM and, of course, no need for the size and expense of a battery for SRAMs.

In the "back to the future" quest for core-like ICs, the FRAM is another step, though certainly not the last, in the right direction. Even at the relatively high introductory price of

\$2.80 (1000) for a 4-kbit device, the FRAM is worthy of consideration in certain applications. □

CONTACT

Ramtron International Corp.
1850 Ramtron Dr.
Colorado Springs, CO 80921
(719) 481-7000
Fax: (719) 481-9170

Tom Cantrell has been an engineer in Silicon Valley for more than ten years working on chip, board, and systems design and marketing. He can be reached at (510) 657-0264 or by fax at (510) 657-5441.

IRS

416 Very Useful
417 Moderately Useful
418 Not Useful

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 ICC-bus expansion boards: BASIC-523 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 an 8K 2764 or 16K 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 6255 PPI
 - Alternate console RS-422/RS-485

To Order Call
1-800-635-3355

Tel: (203) 871-6170

Fax: (203) 872-2204

		Single Qty.	100 Qty.
BCC52	BASIC-52 Controller Board with 8K RAM	\$189.00	\$149.00
BCC52C	Lower-power all-CMOS version of the BCC52	\$199.00	\$159.00
BCC52I	Full industrial temperature range -40° to +85° C	\$294.00	\$220.00
BCC52CX	CMOS, Expanded BCC52 w/32K RAM	\$259.00	\$159.00

MICROMINT, INC. 4 Park Street, Vernon, CT 06066

FINALLY

A Universal Tool for the Firmware Developer

More flexible than any Microprocessor ICE
PROMICE doesn't use a microprocessor pod and handles any word size (8, 16, ..., 2048 bits).

More versatile than intrusive ROM based debuggers
PROMICE doesn't use your target system's I/O resources yet allows you to debug firmware contained in ROM.

More powerful than simple ROM emulators
PROMICE provides complete, source level debugging and full memory control.

PROMICE ... The Universal Firmware Development Tool.

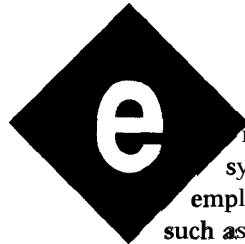
Grammar Engine Inc.
921 Eastwind Dr.
Westerville, OH 43081
Tel: (614) 899-7878
Fax: (614) 899-7888



Putting I²C Through Its Paces

EMBEDDED TECHNIQUES

John Dybowski



Embedded systems often will employ subsystems such as real-time clocks, analog-to-digital converters, digital-to-analog converters, nonvolatile memories, and digital I/O lines. Each of these subsystems are also available with serial I²C interfaces.

By making use of serial connections, you can attach functional blocks to your processor by using just a couple of wires. This simplified connection method allows denser designs since fewer traces are required for each peripheral, which allows more function per square inch, thus lowering system cost.

BUILDING BLOCKS

The National Semiconductor NM24C09 is an 8192-bit serial memory organized in four pages. Each page consists of 256 bytes. A pin is provided that inhibits the ability to program the upper half of the device, effectively transforming a portion of the memory into a ROM. This "virtual ROM" feature can be useful for storing permanent information that must not be altered during normal operations. This implementation of write protection is a hardware function that is enabled by tying the write protect pin to V_{cc} . It cannot be defeated by any software method.

Data retention for the NM24C09 is specified at greater than 40 years and the advertised endurance of 100,000 write cycles permits its use as general-purpose read/write storage in many applications. The device has an active current requirement of 2 mA with a standby current of 60 μ A, which

means you could use it in portable battery-operated systems. Of course, the NM24C09 has all the amenities that you'd expect from this type of device such as self-timed write cycles (with a typical write time of 5 ms), a random read/write capability, and a page write mode.

As with all I²C components, there's nothing to hooking it up. You connect the I²C interface lines to the serial bus in the conventional manner, strap the write-protect pin to the desired state, and tie the address select pins suitably. Of the three address pins, A2 alone is used to set the address to which the chip will respond. Address lines A0 and A1 don't take part in the chip's address selection, and the data sheet says they must be grounded for proper operation. These two address bits are used internally to select one of four available memory pages. Although this may not seem like a big deal, it does have other implications. The RTC chip I selected has the same base address. If these two devices are used together, then only the RTC and a single NM24C09 can coexist on the I²C bus.

If you have a need for an I²C real-time clock/calendar, the PCF8583 from Signetics performs well. Combining a time-keeping function block with flexible alarm capability and 256 bytes of RAM, this device should fit the bill for many requirements. The clock operating voltage and RAM retention voltage for this chip is specified from 1 to 6 volts. At 1 volt, the typical current drain is stated as 2 μ A assuming a clock frequency of 0 Hz. The drain is 200 μ A with a 1 -kHz clock when the RTC is operating in the event counter mode with an external clock. The maximum allowable high-level voltage on any I/O pin for this device is 0.8 volts over V_{dd} . You can use cheap silicon diodes to isolate the power pin from the power supplies. What's really nice is that you can use a single NiCd cell as the backup battery.

The PCF8583 includes several programmable alarms: The list includes a date alarm, a daily alarm, a weekly alarm, or a timer alarm. Each of them may be programmed by setting the alarm control register. The

Does an interface board with A/D and D/A conversion, EEPROM, real-time clock, and 8 bits of I/O mean lots of processor connections? Not with I²C. Following up on last month's introduction, John builds some I²C hardware.

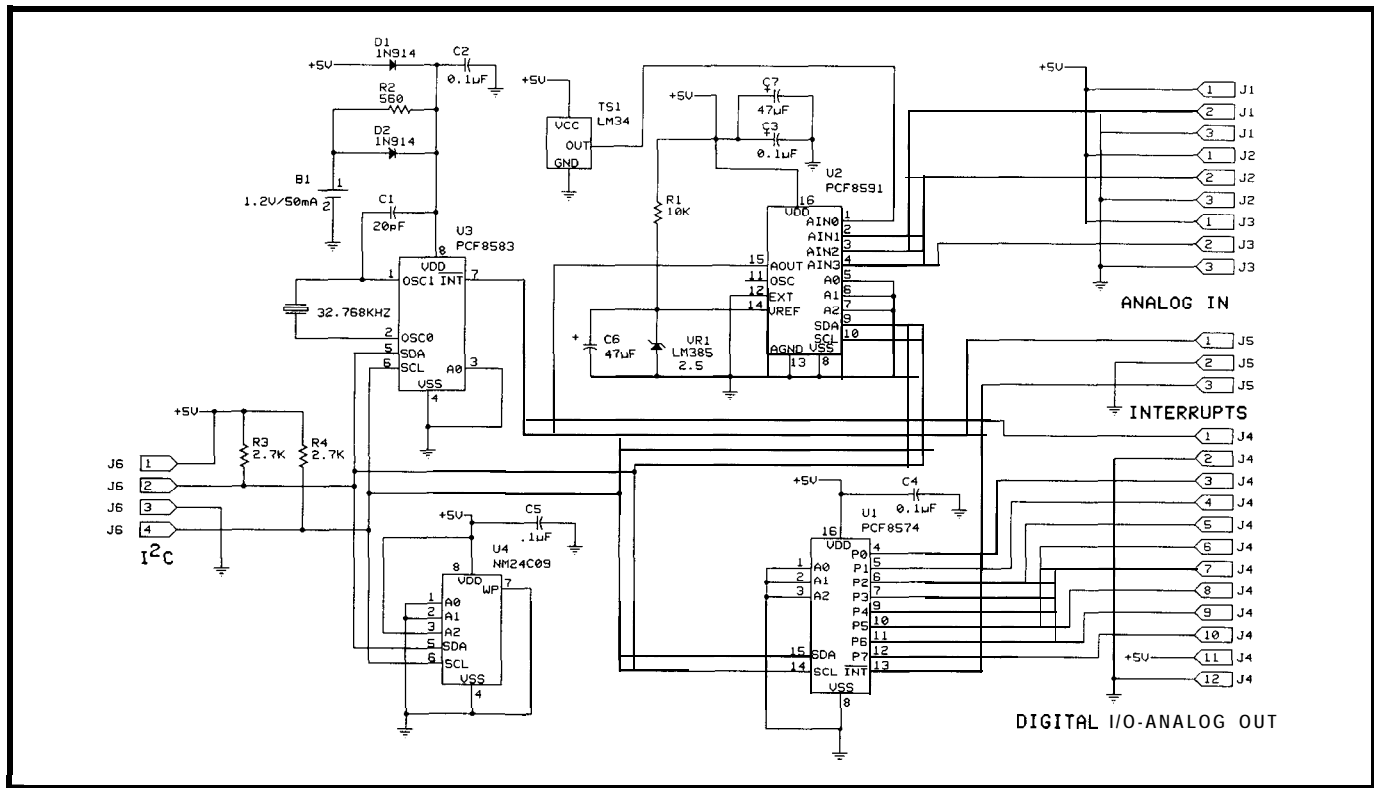


Figure 1—All four PC chips, along with the LM34 integrated Fahrenheit temperature sensor, fit on a one-by-three-inch prototype circuit board.

timer register can be programmed to count hundredths of a second, seconds, minutes, hours, or days. Someone finally got smart and made a timepiece with some useful interrupt generation capability!

You can program the PCF8583 to work with a 32.768-kHz clock, a 50-Hz clock, or to operate in an event counter mode. The event counter mode is used to count pulses at the

oscillator input. Up to six digits of event data can be stored on chip.

Capture latches ease the load of the processor attached to the PCF8574 and lessen the hazards that are usually associated with directly reading a running counter. When one of the counters is read, the contents of all the counters are strobed into a set of capture registers at the beginning of that read cycle. The PCF8583 auto-

matically guarantees that all the counts acquired during a read operation are correct. The capture latches are available in the event counter mode and in the clock mode.

No matter how much I/O you have, you can always use a little more. The Signetics PCF8574 is an I²C peripheral that functions as an 8-bit remote I/O expander. This device features a wide power supply range and low power requirements. The PCF8574 offers eight quasi-bidirectional I/O lines that can source 400 μ A and sink 20 mA. The minuscule source capability is what you'd expect from a quasi-bidirectional port structure. However, with the substantial current sinking capacity, you can directly drive LEDs or other fairly heavy loads without requiring external buffers.

The PCF8574 has an interesting feature that can be used to reduce the CPU burden when monitoring inputs. An interrupt signal is generated by any rising or falling edge at any pins on the port that are currently configured as being in input mode. The interrupt condition is cleared when data on the port returns to the original settings, or when data is written to or read from the port that generated the interrupt.

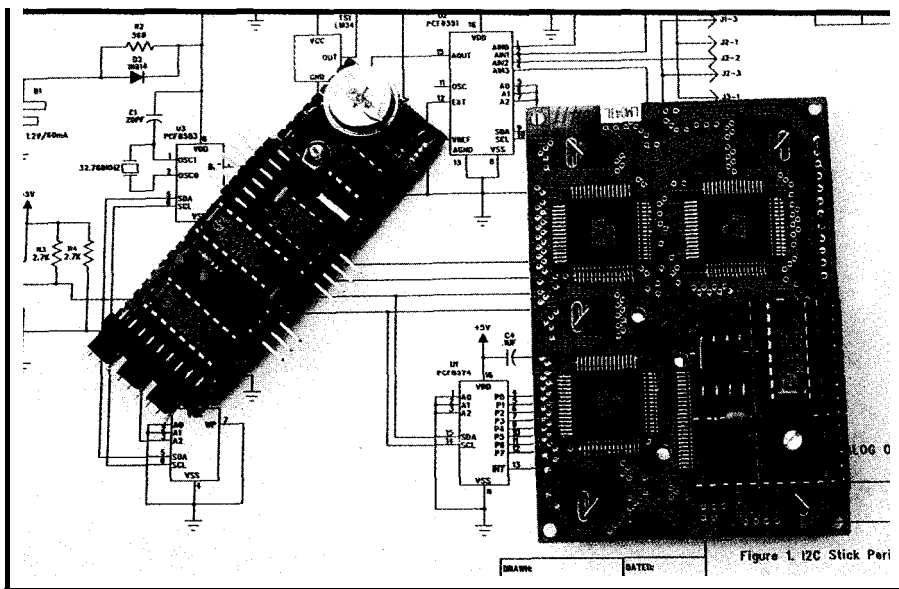


Photo 1--The PC prototype board and the LCD display can be interfaced to just about any processor with just four wires: power, ground, clock, and data.

SAVE

53%

SUBSCRIBETODAY TO

THE COMPUTER APPLICATIONS JOURNAL

12 ISSUES FOR ONLY

\$21.95*

WRITTEN
BY ENGINEERS
FOR ENGINEERS!

HANDS-ON
HARDWARE PROJECTS

ADVANCED
APPLICATIONS

TECHNOLOGY
TUTORIALS

NO VAPORWARE!

TO TAKE ADVANTAGE OF
ALL THIS TECHNOLOGY,
JUST FILL OUT THE
SUBSCRIPTION CARD ON
PAGE 16 OF THIS ISSUE
AND

FAST FAX

YOUR ORDER!

(203) 872-2204

OR MAIL TO:

THE COMPUTER
APPLICATIONS JOURNAL
P.O. Box 7694
RIVERTON, NJ 08077-8794

* Price good in U.S. only. Canada/Mexico \$31.95,
all other foreign \$49.95. U.S. funds drawn on U.S.
banks only.

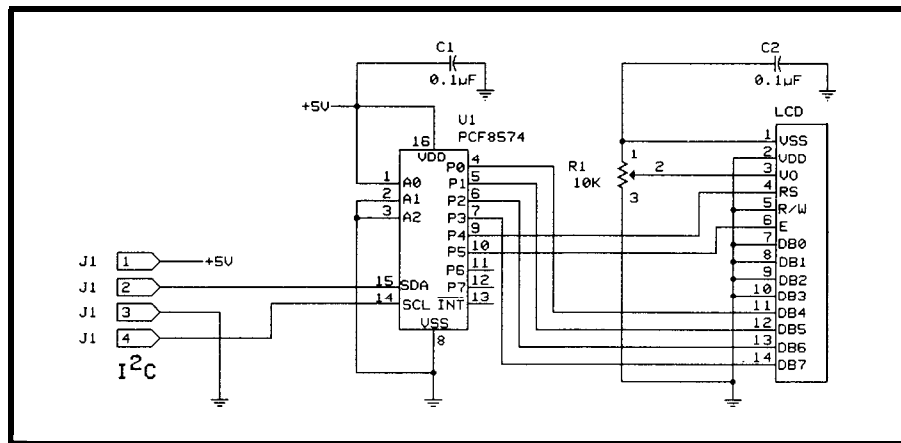


Figure 2—The PCF8574 I/O expander makes an idea/ interface between an LCD display and the PC bus.

Although full-featured PC micro-controllers are available, you could use the PCF8574 to interface processors to the I²C bus. This would work as long as you were content to provide these processors with slave-mode only capabilities.

Rounding out the usual control-oriented peripheral set, the PCF8591 provides a 4-channel, 8-bit ADC along with a single DAC channel. This multifunction chip features a wide operating voltage range and very low power consumption.

The PCF8591 provides the standard fare of features such as track and hold, auto-increment channel selection, and three selectable address

bits that you'd come to expect from serial peripherals of this class.

Analog-to-digital conversion is performed using the standard successive approximation method. Using an external voltage reference, the A/D conversion can be set up to operate using four single-ended inputs, three differential inputs, two single-ended inputs with one differential input, or two differential inputs. Each of these conversion options is software selectable. The conversion method used can be reconfigured on-the-fly. The clock for the converter's internal operations can be derived directly from the I²C bus signaling or from an external source. This option is pin selectable.

Listing 1—The LCD support code includes low-level PC routines.

```

;Driver for 4 x16 LC Display
;Global Entry Points

PUBLIC LCD_INIT
PUBLIC LCD-CLEAR
PUBLIC SET-CURSOR
PUBLIC DISP_BYTE
PUBLIC DISP_IRAM
PUBLIC DISP_XRAM
PUBLIC DISP_PROM

;External References

EXTERN Xmit_I2C_Byte

;Defined I/O

LCD_PORT EQU 42H
DEN EQU ACC. 5
DRS EQU ACC. 4

;Internal Data

LCD-CURSOR RSEG DATA DS 1

```

(continued)

Listing 1 - continued

```
:Assemble Into Code Segment
      RSEG   CODE
:WRITE To LCD COMMAND REGISTER, INPUT: CHARACTER TO WRITE IN ACC
LCD_COMM_WR: PUSH   ACC
              ANL   A, #0F0H
              SWAP  A
              CLR   DRS
              SETB  DEN
              CALL  LCD-OUT
              CLR   DEN
              CALL  CD_OUT
```

(continued)

An Unbiased Survey for DOS Developers.

No!

I don't want to find out how I can save a lot of money using ROM-DOS 5 instead of MS-DOS[®] in our 80x86 product line. I don't care if ROM-DOS 5 is incompatible with MS-DOS 5 but costs much less. I like spending much more than I have to. It makes me feel like a philanthropist and besides Microsoft[®] probably needs the money more than I do anyway.

Yes, I want to know the facts about ROM-DOS 5.

Please send me information and a free bootable demo disk to try with my software.

In the U. S. A. Call Toll Free 1-800-221-6630
or fax this coupon to (206) 435-0253.

Name _____

Company _____

Address _____

City _____ State _____ Zip _____

Phone _____ Fax _____

307 N. OLYMPIC, SUITE 201 • ARLINGTON, WA 98223 USA • (206) 435-8086 • FAX: (206) 435-0253

Datalight[®]

MS-DOS and Microsoft are registered trademarks of Microsoft Corporation

Surprisingly, the combination of A/D and D/A functionality is absent from many of the converter chips on the I²C market. The PCF8591 does a sensible thing and uses the same DAC for both purposes. In order to release the DAC for an A/D conversion cycle, a unity-gain amplifier is equipped with a track-and-hold circuit. This circuit serves to hold the analog output voltage while the A/D cycle is executing. This amplifier may be switched off using an internal control bit when the analog output is not needed.

I²C ON A STICK

Easy as it is to wire up these functions as needed, I decided to combine these elements for use as a component in a larger system. Such an approach not only lends itself to fast prototyping, but can be put to use when constructing small devices that require these basic functions. Figure 1 shows the circuitry of this function block. Photo 1 shows the wired circuit board (along with the serial LCD interface that I'll describe shortly). The small number of interface pins makes for an easy hookup to just about any single-board computer.

Aside from the parts that I've already described, and their necessary support components, I tied an LM34 integrated Fahrenheit temperature sensor to channel 0 of the ADC.

Interfacing this to your favorite processor is a lesson in simplicity. Power, ground, and the two I²C interface signals are brought out on a 4-pin header; plug it into your controller board and you're ready to go live. A second 3-pin header handles the two interrupt signals along with an extra ground. These lines connect to the PCF8574 port expander and the PCF8583 real-time clock. Note that the RTC's interrupt line operates even when the main logic power supply is shut off. You can use this signal as a wake up call for external power control circuitry. Add a small processor and you have a miniature, battery-operated data logger.

SERIAL LCDS

The I²C display driver I wrote performs the usual functions that

would be required of an LCD: initialize and clear the display, position the cursor, display a byte, and display a string.

This program is basically a variation of a standard driver that I've modified numerous times to run on different hardware. The modifications related to I²C are straightforward.

The key to reliable LCD operation is in the initialization routine. Since the HD44780 LSI powers up in its default 8-bit interface state, the first thing that must be done is to put it into 4-bit mode. However, you'll notice that the code puts the LSI into 8-bit mode three times! This step puts the display into a known state.

Following this step, the LSI is put into the 4-bit mode of operation. What follows is the standard sequence that sets up the operational parameters such as turning the display on, turning the cursor off, setting the input mode, and so forth. Finally, the Clear function is invoked and the routine terminates. Looking at the code, you'll see that I track the cursor position by

Listing I-continued

```

POP      ACC
ANL      A, #0FH
CLR      DRS
SETB     DEN
CALL     LCD-OUT
CLR      DEN
CALL     LCD_OUT
RET

```

;WRITE TO LCD DATA REGISTER, INPUT: CHARACTER TO WRITE IN ACC

```

LCD_DATA_WR: PUSH  ACC
ANL      A, #0F0H
SWAP     A
SETB     DRS
SETB     DEN
CALL     LCD-OUT
CLR      DEN
CALL     LCD-OUT
POP      ACC
ANL      A, #0FH
SETB     DRS
SETB     DEN
CALL     LCD-OUT
CLR      DEN
CALL     LCD_OUT
RET

```

;SYNCHRONIZE WITH LCD AND TRACK CURSOR POSITION

(continued)



An Official Entry Form must accompany all entries. To receive an Official Entry Form and a complete set of contest rules,

write or call:

CIRCUIT CELLAR DESIGN CONTEST

Circuit Cellar Design Contest

4 Park St., Ste. 90 • Vernon, CT 06066

(203) 875-2199 (ask for Rose)

Fax: (203) 872-2204

All entries must be received by September 17, 1993. Prizes include \$500 for first, \$200 for second, \$100 for third, and \$50 honorable mentions.

Listing 1-continued

```

LCD-WAIT:  PUSH  ACC
           MDV  A,LCD_CURSOR
           CJNE A,#16,LW1      ;AT END OF 1ST LINE?
           MDV  A,#64+10000000B
           CALL LCD_COMM_WR
           SJMP LW3
LW1:      CJNE A,#32,LW2      ;AT END OF 2ND LINE?
           MDV  A,#16+10000000B
           CALL LCD_COMM_WR
           SJMP LW3
LW2:      CJNE A,#48,LW3      ;AT END OF 3RD LINE
           MDV  A,#80+10000000B
           CALL LCD_COMM_WR
LW3:      POP   ACC
           RET

;SET LCD CURSOR POSITION, INPUT: ACC CONTAINS CURSOR POSITION
SET-CURSOR: MDV  LCD_CURSOR,A
            CJNE A,#48,$+3
            JC   SC1
            ANL  A,#0FH
            ADD  A,#80+10000000B ;LINE 4
            CALL LCD_COMM_WR
            SJMP SC4
SC1:      CJNE A,#32,$+3
            JC   SC2
            ANL  A,#0FH
            ADD  A,#16+10000000B ;LINE 3
            CALL LCD_COMM_WR
    
```

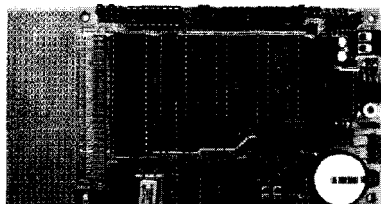
(continued)

using a RAM variable. Calling the `CLEAR` routine is necessary in order to initialize this variable properly.

Two intermediate-level support routines are provided to write to the command register and to the data register. These routines are the link between the higher-level functions and the low-level I²C interface. The interface routine simply saves several important registers on the stack, positions the data to write in the B register, and stuffs the I/O expander's port address in the accumulator before calling the I²C byte-level driver. On exit, the PUS Hed registers are restored.

When using a bidirectional interface, you can pick up the cursor address by reading the status register. You can use this cursor information to place data onto the panel sequentially. Since this interface drives the LCD as an output-only device, I don't have access to the status register, which is why I keep track of the cursor position in RAM. `LCD_WAIT` looks at this information and repositions the cursor to make things come out right.

8051 EMBEDDED CONTROLLERS With Lots of Extras!



We offer a full line of low cost 80C32 embedded controllers and software tools which are ideal for developing products, test fixtures and prototypes.

Features Include:

- Low power CMOS design
- Up to 60K of codespace and up to 60K of data space
- 5 to 15 volt operation
- Small form factor (3.5" * 6.5") with prototyping area
- System diskette includes application notes
- Start at \$100

Available Options:

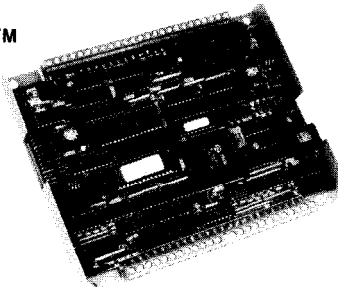
- Multifunction Board adds A/D, 24 I/O lines and more!
- BASIC-520r Monitor/Debugger in EPROM
- C Compiler \$100 or BASIC Compiler for \$300

Iota Systems, inc.

POB 8987 *Incline Village, NV 89452
PH: 702-831-6302 • FAX: 702 831-4629

C-PLC™

\$289!



- New C Programmable miniature controller
- Seven 10-bit analog inputs
- Seven digital inputs
- 10-bit DAC: voltage or current output
- Twelve digital/relay driver outputs
- RS-232/RS-485 serial ports
- Enclosure with LCD/Keypad available
- Expansion bus for additional, low cost I/O
- Easy to use Dynamic C™ development software only \$195!

Z-World Engineering

1724 Picasso Ave., Davis, CA 95616

(916) 757-3737 Fax: (916) 753-5141

24 hr. Information Service: (916) 753-0618

(Call from your fax and request data sheet #24)

I take a similar approach with the SET_CURSOR routine. Here, the input is a number that starts at zero and counts up to the last character position on the display. SET_CURSOR handles the translation from this standard notation to what the LCD understands.

Finally, DISP_BYTE, DISP_IRAM, DISP_XRAM, and DISP_PROM handle display operations for bytes and strings. With all the support functions in place, these operations should be self explanatory.

Figure 2 is the schematic for the serial LCD interface. Listing 1 is the serial LCD driver program.

THE EXTENDED I²C

The I²C bus capacitance limitation of 400 pF confines line lengths to a few meters, which places significant restrictions on many applications. The predicament of line buffering is tricky since the whole concept of the I²C bus centers on the bidirectional/open-drain characteristics of the interface lines. An answer comes in the form of an I²C bus extender circuit from Signetics.

The 82B715 is a bipolar integrated circuit that retains all the operating modes and features of the I²C bus. The practical separation distance between I²C components is increased by buffering both the SDA and SCL lines.

The 82B715 provides a bidirectional impedance transformation by using dual bidirectional unity-voltage-gain buffers with an effective current gain of ten. This means that ten times the current of whatever is flowing into the I²C bus side flows out of the buffered side. It follows that the buffered side will be able to drive capacitive loads up to ten times the unbuffered limit while preserving the bidirectional, open-drain (or open-collector in this case) characteristics of the SDA and SCL lines.

Since these buffers retain the qualities of an I²C device, a system can be constructed that is designed around these extenders. Alternatively, extended sub-buses can be added to existing I²C implementations.

Recognizing that this is a bipolar circuit helps explain why its operating

Listing 1 - continued

```

SC2:      SJMP   SC4
          CJNE  A,#16,$+3
          JC    SC3
          ANL  A,#0FH
          ADD  A,#64+10000000B ;LINE 2
          CALL LCD_COMM_WR
          SJMP SC4
sc3:      ADD  A,#00+10000000B ;LINE 1
          CALL LCD_COMM_WR
sc4:      RET

;CLEAR LCD
;
LCD_CLEAR: MOV   A,#1
          CALL  LCD_COMM_WR
          MOV   LCD_CURSOR,#0
          MOV   RO,#5
          CALL  DELAY
          RET

;DISPLAY A CHARACTER. INPUT: ACC CONTAINS CHARACTER

DISP_BYTE: CALL  LCD_WAIT
          CLR   ACC
          CALL  LCD_DATA_WR
          INC   LCD_CURSOR
          RET

;DISPLAY IRAM DATA, INPUT: R1 POINTS TO DATA
; RO CONTAINS BYTE COUNT
;
DISP_IRAM: MOV   A,@R1
          CALL  DISP_BYTE
          INC   R1
          DJNZ RO,DISP_IRAM
          RET

;DISPLAY XRAM DATA, INPUT: DPTR POINTS TO DATA
; RO CONTAINS BYTE COUNT

DISP_XRAM: MOVX  A,@DPTR
          CALL  DISP_BYTE
          INC   DPTR
          DJNZ RO,DISP_XRAM
          RET

;DISPLAY PROM DATA, INPUT: DPTR POINTS TO DATA
; RO CONTAINS BYTE COUNT
;
DISP_PROM: CLR   A
          MOVC  A,@A+DPTR
          CALL  DISP_BYTE
          INC   DPTR
          JNZ  RO,DISP_PROM
          RET

;INITIALIZE LCD
;
LCD_INIT: MOV   RO,#20
          CALL  DELAY ;INITIAL DELAY
          MOV   A,#0011B ;PUT LSI IN KNOWN STATE
          SETB DEN
          CALL  LCD_OUT
          CLR  DEN
          CALL  LCD_OUT
          MOV  RO,#5

```

(continued)

Listing 1-continued

```

CALL    DELAY
MOV     A,#0011B           ;AGAIN
SETB   DEN
CALL   LCD_OUT
CLR     DEN

CALL   LCD-OUT
MOV    RO,#5
CALL   DELAY
MOV    A,#0011B           ;AGAIN
SETB   DEN
CALL   LCD-OUT
CLR     DEN
CALL   LCD_OUT

MOV    RO,#5
CALL   DELAY
MOV    A,#0010B           ;4-BIT MODE
SETB   DEN
CALL   LCD-OUT
CLR     DEN
CALL   LCD-OUT
MOV    RO,#5

CALL   DELAY
MOV    A,#00101100B       ;4-BIT, Z-LINE, 4X7 MATRIX
CALL   LCD_COMM_WR
MOV    RO,#5
CALL   DELAY
MOV    A,#00001100B       ;DISPLAY ON, CURSOR OFF
CALL   LCD_COMM_WR
MOV    RO,#5

CALL   DELAY
MOV    A,#00000110B       ;AUTO INC SHIFT RIGHT
CALL   LCD_COMM_WR
MOV    RO,#5
CALL   DELAY
CALL   LCD-CLEAR         ;CLEAR DISPLAY
RET

```

```

;OUTPUT A BYTE TO THE I2C LCD PORT
;INPUT: ACC CONTAINS BYTE TO OUTPUT
VALUE OF ACC IS RETAINED ON EXIT

```

```

LCD_OUT:  PUSH  ACC
          PUSH  0
          PUSH  1
          MOV   B,A
          MOV   A,#LCD_PORT
          CALL  Xmit_I2C_Byte
          POP   1
          POP   0
          POP   ACC
          RET

```

```

;GENERAL DELAY ROUTINE
;INPUT: RO CONTAINS DELAY IN MSEC'S
;
DELAY:   MOV   R1,#2
DELAY1:  MOV   R2,#0F8H
          DJNZ R2,
          DJNZ R1,DELAY1
          DJNZ R0,DELAY
          RET

```

END

voltage is specified at 4.5 to 5 volts with a supply current of 16 mA (when V_{cc} is 5 V). The minimum sink capability on the I²C side is the typical 3 mA rating, where the buffered side can handle 30 mA.

As in standard I²C systems, pull-up resistors are required for the logic-high levels. If the buffer is permanently connected to the system, the pull-ups would be configured on the buffered bus with none on the unbuffered side. If the buffer were connected to an existing system, the buffered bus pull-ups would act in parallel with the unbuffered pull-ups.

WRAPPING UP

By now you should see the utility and flexibility of the I²C bus. For many embedded applications, even the 100-kbps throughput can prove to be entirely adequate. If it's not good enough, then you can increase the data rate using the faster versions of these parts that are now available. With the new bus extender IC, the limitation on line length becomes much less of an obstacle. Of course, there will be those malcontents (like me) who won't be happy until they can get that extender chip in CMOS!

John Dybowski is an engineer involved in the design and manufacture of hardware and software for industrial data collection and communications equipment.

CONTACT

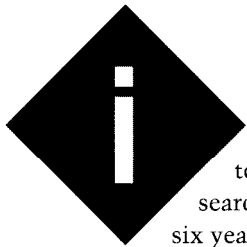
National Semiconductor Corp.
2900 Semiconductor Dr.
P.O. Box 58090
Santa Clara, CA 95052-8090
(408) 721-5000

Signetics Corp.
811 Arques Ave.
P.O. Box 3409
Sunnyvale, CA 94088-3409
(408) 721-7700

IRS

419 Very Useful
420 Moderately Useful
421 Not Useful

PATENT TALK

 by Russ Reiss

In keeping with this month's special topic theme of Communications, I searched the patent database for the past six years for pertinent patents relating to both computers and communications. It's always interesting, and sometimes surprising, to see what turns up as a result of a keyword search. As you'll see from the variety of abstracts selected for presentation here, computer communication covers a wide range of topics!

It is interesting to note that the database contains over 250 patents issued just since 1990 on spread spectrum related topics. This is truly a fast-developing area. However, I specifically eliminated all but one spread spectrum patent from this month's list, for I plan to devote an entire column to that important topic in the near future.

Computer communication at perhaps the most intrinsic level is presented in the first abstract of a patent from Ramtech Inc. Their "CPU socket" concept proposes that the actual IC socket for the CPU will contain an embedded optical communication mechanism in future computers. This approach will overcome a host of problems associated with metallic-contact-type interconnection to a CPU.

Certainly, the complexity of CPUs increases yearly, along with the number of data, address and control lines, as well as their speed. A high-speed optical link sounds like a straightforward, trouble-free, high-bandwidth solution to the interconnect problem. The inventors even go a step beyond communication by proposing that power for the CPU also be provided photovoltaically across the socket.

Another interesting patent that uses an optical link to cross a unique physical barrier is Northern Telecom's infrared hookswitch presented in Abstract 2. While the hookswitch function of a telephone is improved only slightly through the use of an interrupted optical switch versus a mechanical one, the presence of light emitting and detecting diodes at the surface of the enclosure suggested to the inventors other uses. The abstract does not spell out the specific uses planned for the optical link to the CPU embedded within their telephone, but certainly programmability by field personnel comes to mind. However, it seems like an ideal means for a user to connect either a high-speed data or fax modem to a pay-phone system. This type of connection would overcome many shortcomings of acoustic or magnetic coupled modems.


One discussion that appeared recently on the Circuit Cellar BBS centered around communication and display of pricing information throughout supermarket shelves. Abstract 3 presents a fairly recent patent that discloses the

Patent Number	4,953,930
issue Date	1990 09 04
Inventor(s)	Ramsey, Bernard; Christy, Dean A.; Beverly, Richard S.; Wucher, Jerome M.
State/Country	VA
Assignee	Ramtech, Inc.
US References	3,418,533 3,536,404 4,356,395 4,737,778 4,532,532 4,553,813 4,597,631 4,695,120 4,696,536 4,703,471 4,732,446
US Class	350/96.11 357140
Int. Class	G02B6/12
Title	CPU socket supporting socket-to-socket optical communications
Abstract	A communication socket hereinafter referred to as the "CPU socket" and its preferred methods of integration into conventional electronic circuits are described. The CPU socket advantageously uses hybrid devices embedded within a conventional integrated circuit (IC) socket. Circuit connectivity is maintained via photon transmission without the need of conventional metallic connections. Typical problems associated with metallic traces, circuit board geometries, packing densities, and parasitic limitations of conventional PCBs are eliminated. The CPU socket emulates all of the physical aspects of PCB metallic etched traces via a photon mechanism. The CPU socket supports system networking functions normally available only in large "intra" and "inter" computer communications facilities, and reduces system networking design down to socket/PCB level. Additionally, the CPU socket generates the power necessary for operation as well as that needed by hosted ICs via photovoltaic devices contained within the CPU socket. The voltaic devices may be driven by any natural or artificial photon source of sufficient intensity to power the socket and piggyback IC.




PATENT TALK

Patent Number	4,847,900
Issue Date	198907 11
Inventor(s)	Wakim, Michael J.
State/Country	CAX
Assignee	Northern Telecom Limited
US References	4,203,006 4592,069
US Class	3791424 3791443
Int. Class	H04M 1106 H04M 11/00
Title	Infrared hookswitch for a telephone
Abstract	An optical hookswitch assembly useable on a telephone set as an optical communications port is disclosed. The hookswitch assembly is composed of light emitting and detecting diodes so disposed in the telephone so as to detect the presence of a handset. A processor connectable to the telephone circuitry and the light emitting and detecting diodes is provided such that the telephone circuitry is activated by the processor when the light emitting and detecting diodes fail to detect the presence of a handset. The light emitting and detecting diodes are useable as an optical communications port for accessing the processor by allowing an external computer to communicate with the processor via an optical coupler.



Patent Number	4,937,586
Issue Date	1990 06 26
Inventor(s)	Stevens, John K.; Waterhouse, Paul I.
State/Country	CAX
US References	2,250,370 2,823,382 3,049,711 3,683,389 4,155,091
US Class	3431702 3431742 3431788
Int. Class	HOI Q7/00
Title	Radio broadcast communication systems with multiple loop antennas
Abstract	The invention comprises a low-power broadcast system that is applicable especially to the so-called "electronic shelf" for retail stores, wherein the shelf edge carries price-displaying modules that can be addressed and controlled from a central computer operated station. The system also permits the modules to broadcast back to the central station to confirm safe receipt of data and to give information as to stock levels, etc. A broadcast system avoids the need for wiring so that location changes are facilitated. To overcome the extremely noisy environment and to conserve power consumption, and hence battery life, the system employs a low-frequency (132 kHz) reference carrier transmitted by the base station in discrete segmented packages, each of which frames a base data word transmitted by the base station and a corresponding module data word transmitted by the module a fixed period after the end of the base word; the base receiver then has precise time information for receipt of the module word and can "look" for it among the noise. The carrier received by the module is divided and the lower frequency used to demodulate the information-carrying transmission from the base station of the same frequency, avoiding the need for a phase locked loop detector; this lower frequency is also used for the module transmission. The module employs an air-cored loop antenna coil for the lower frequency and a ferrite-cored loop antenna for the higher reference frequency, while the store antenna is segmented for selection of the group of modules to be addressed; the antenna contacts the metal shelving to provide electromagnetic coupling thereto. Each module contains a microprocessor which controls the operation. Each module has "concealed" buttons which can be enabled and used to insert data to be transmitted therefrom. A charging circuit can be used as the power source employing the received RF carrier energy.



PATENT TALK

details of one specific means for achieving this goal. Their approach uses a low-radio-frequency, bidirectional broadcast technique with interesting timing synchronization based on their 132-kHz carrier. Interestingly, my search uncovered patents 4,821,291 and 4,879,756 by the same inventors and with the *identical* abstract. I fail to see what is gained by receiving multiple patents on the same device, or why the practice is permitted by the Patent Office.

Abstract 4 is the one spread spectrum patent I let through the door this month because of its potential importance and the novelty of its design in so many areas. If it can truly provide the bidirectional communication link to 75,000 subscribers that it promises, it could have great impact on the future of interactive TV. The system combines the use of existing synchronization pulses within a TV system, a novel approach to communication based on radar principles, and spread spectrum to pull off this feat. Interestingly, it works both for fixed cable installations as well as for mobile RF-based stations.

The next two abstracts demonstrate how computers and communications might impact everyday life in the future by using the telephone line in novel ways. The system in Abstract 5 appears to make use of the power of a central computer to perform text-to-speech conversion.

Low-error-rate and high-quality speech still requires a very powerful computer beyond the means of most individuals. This approach uses a conventional fax (most likely via a board inserted in a PC) to send the text graphically to the computer over phone lines (of course, it's an AT&T patent!). The (time-shared, super) computer provides OCR of the received image and conversion to speech, which goes back to the user over the same phone lines. Interesting service that places AT&T as both a computer and communication provider. It also has interesting connections to multimedia computing!

Abstract 6 improves on existing radio pager capability by using the computer to control a cross-point link via phone lines between the calling party and the subscriber who calls back in response to the page. Should the paging party have hung up already, the computer attempts to establish the connection by calling back the pager.

Finally, the Sundstrand patent in Abstract 7 appears to be a very useful system for pilots. It combines the power and portability of a small computer with both telephone and radio links in order to obtain, select, and update flight plans. As presented, the system requires that the floppy disk containing the flight plan generated on the ground be entered into the aircraft navigational system. I envision the

Patent Number	4,750,036
Issue Date	1988 06 07
Inventor(s)	Martinez, Louis
State/Country	CA
Assignee	Radio Telcom & Technology, Inc.
US References	3,529,081 4,074,199 4,155,039 4,177,405 4,208,630 4,231,114 4,513,415 4,538,174 4,578,815 4,591,906 4,622,694
US Class	3581147 358/142 358/84
Int. Class	H04N 7/093 H04N 7/08 H04N 7100
Title	Interactive television and data transmission system
Abstract	A spread spectrum system provides bidirectional digital communication on a vacant television (TV) channel for simultaneous use by more than 75,000 subscribers using time and frequency division multiplex signals locked to horizontal and vertical sync pulses of an adjacent channel Host TV station. The system, whose operation is analogous to a radar system, comprises: (1) the Host TV station to send down-link sync and data pulses to subscribers during the horizontal blanking interval (HBI), (2) subscriber "transponders" which detect those signals and transmits up-link "echo" data pulses only during the HBI to eliminate interference to TV viewers, and (3) a central receiver which also uses the host TV sync pulses to trigger range gates to detect the up-link data pulses. In a preferred embodiment, the central receiver employs directional antennas to determine direction to transponders and to define angular sectors partitioning the service area into pie-link "cells" which permit frequency reuse in noncontiguous sectors (like cellular radio). The system thus operates like a radar to measure elapsed time between receipt of TV sync pulses and receipt of transponder response pulses and measures bearing to transponders to thereby determine the location of fixed or mobile subscribers as well as provide data links to them. Transponders may share user's existing TV antenna or may operate on cable TV and could be packaged as "RF modems" for personal computers, as transceivers for mobile or portable use, or they may be integrated with a TV receiver to provide "interactive television."

4

PATENT TALK

possibility of that step being eliminated in a small private airplane. In that case, optional auxiliary inputs to the same portable computer could provide connection to the navigational instrumentation. But, in either case, updating the flight plan and weather information in the portable computer via the VHF radio while en route seems quite feasible and useful. This would achieve most of the functions proposed without the need for a different, built-in aircraft computer. □

Russ Reiss holds a Ph.D. in EE/CS and has been active in electronics for over 25 years as industry consultant, designer, college professor, entrepreneur, and company president. Using microprocessors since their inception, he has incorporated them into scores of custom devices and new products. He may be reached on the Circuit Cellar BBS or on CompuServe as 70054,1663.

SOURCE


Patent abstracts appearing in this column are from the Automated Patent Searching (APS) database from:


MicroPatent
25 Science Park
New Haven, CT 065 11
(203) 786-5500 or (800) 648-6787

MicroPatent databases include the abstract-only APS version; FullText, which contains the entire patent without drawings; PatentImages, for the complete patent listing including drawings; and other specialized databases for just chemical, computer, or European patents.

IRS

422 Very Useful 423 Moderately Useful 424 Not Useful

Patent Number	5,091,931	
Issue Date	1992 02 25	
Inventor(s)	Milewski, Allen E.	
State/Country	NJ	
Assignee	AT&T Bell Laboratories	
US References	2,500,630 2,615,992 3,059,064 3,114,980 3,704,345 4,278,838 4,685,135 43908,867 4,996,707	
US Class	379/100 381152	
Int. Class	H04M 11/00	
Title	Facsimile-to-speech system	
Abstract	Written material is read at low cost by a computer-based system which is designed to receive via a telephone line a facsimile of the written material submitted by a system user. Once the facsimile is received, the system performs an optical character recognition (OCR) process thereon. The text thus identified by the OCR process is converted to intelligent speech using a speech synthesizer. The synthesized speech is communicated back to the system user either over the already established telephone connection or in a subsequent call.	

Patent Number	5,151,929	
Issue Date	1992 09 29	
Inventor(s)	Wolf, Sherman	
State/Country	NH	
Assignee	Wolf, Sherman	
US References	3,627,955 4,263,480 4,313,035 4,575,592 4,680,785	
US Class	379157 379/67 379/201	
Int. Class	H04M 11/00	
Title	Computer-controlled radio-paging and telephone communication using recorded voice messages	
Abstract	A method of and apparatus for notifying a remote subscriber of a caller's attempted communication by a communication system accepting and recognizing a call for a subscriber, paging the subscriber, and connecting the original caller to the subscriber's telephone line when the subscriber calls the system in answer to the page. Other features include recording a caller message for the subscriber if the subscriber calls back to the system after the original caller has disconnected.	

PATENT TALK



Patent Number 4642,775
 Issue Date 198702 10

Inventor(s) Cline, J.; Wilson, James A.; Feher, Stanley H.;
 Ward, George D.

State/Country CA
 Assignee Sundstrand Data Control, Inc.

US References 3,781,530 3,786,505 3,805,261 4,086,632 4,103,300 4,144,571 4,179,693 4,212,067 4,220,994
 4,224,669 4,253,150 4,312,041 4,340,936 4,360,876 4,413,322 4,428,052 4,454,510 4,495,580
 4,521,857

US Class 3641443 3641420 3641444
 Int. Class G06F 15150

Title Airborne flight planning and information system

Abstract A flight planning system for obtaining flight plans and/or weather information is provided with a portable computer having a display unit, keyboard, memory, built-in modem, and built-in disk drive that can be connected via telephone lines to a ground-based data center. The basic flight planning data and/or weather request data is input in response to menu-driven prompts and reviewed on the display by the pilot. The portable computer is then connected to the data center which generates a series of optimized flight plans and provides desired weather information. After the desired flight plan and/or weather information has been selected by the pilot, it is loaded onto a floppy disk in the disk drive. The aircraft is provided with a data transfer unit which accepts the floppy disk and downloads the flight plan and requested weather information into the on-board computerized navigation system. In addition, the aircraft is provided with a VHF radio system for in-flight communication with the data center so that the flight plan and/or weather information can be updated.

The Ciarcia Design Works

Does your big-company marketing

Steve Ciarcia and the Ciarcia Design Works staff may have the solution.

department come up with more ideas

We have a team of accomplished programmers and engineers ready to

than the engineering department can

design products or solve tricky engineering problems. Whether you

cope with? Are you a small company

need an on-line solution for a unique problem. a product for a startup

that can't afford a full-time engineer-

venture, or just experienced consulting, the Ciarcia Design Works is

ing staff for once-in-a-while designs?

ready to work with you. Just fax me your problem and we'll be in touch.

Remember...a Ciarcia design works!

Call (203) 875-2199 Fax (203) 875-8786

REMOTE POWER CARD!

3 VERSIONS:

RESET

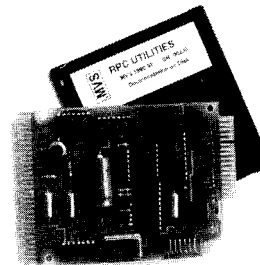
WATCHDOG RESETS PC IF
 IT HANGS FOR HARDWARE
 OR SOFTWARE REASONS

PHONE

TURN ON PC WITH PHONE,
 SHARE VOICE / MODEM LINE,
 CONTROL AC APPLIANCES

TIMER

WAKEUP OR SHUTDOWN PC,
 LATE NITE BACKUP / MODEM,
 CONTROL AC APPLIANCES



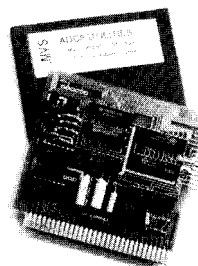
95\$ 27\$
 OEM

ALL MVS CARDS INCLUDE ASM, BASIC, AND C SOURCE FOR PC OR SB

8 CHAN ADC

DATA ACQUISITION, SERVO CTL, AUDIO
 8-BIT RESOLUTION 22KHZ SAMPLE RATE
 SHARP CUTOFF ANTI-ALIAS FILTER
 CREATE STEREO BLASTER(VOX) FILES

95\$



2 CHAN DAC

VOICE MAIL, MUSIC, ALARMS, CTL VOLT
 8-BIT RESOLUTION 44KHZ SAMPLE RATE
 PLAYS MONO / STEREO BLASTER FILES
 FUNCTIONS AS DIGITAL ATTENUATOR TOO

75\$



MVS BOX 850
 MERRIMACK, NH
 (508) 742-4507

5 YEAR LIMITED WARRANTY
 P I N G I N U S A

CONNECT TIME

conducted by Ken Davidson

The Circuit Cellar BBS
300/1 200/2400/9600/1 4.4k bps
24 hours/7 days a week
(203) 871-1988—Four incoming lines
Vernon, Connecticut

In the first thread this month, we discuss what's involved in doing some very precise measurements of high-frequency current flow. It's not as easy as you might think.

The next thread is about a fascinating solid-state cooling/heating device that has many applications where size or other constraints prohibit traditional heating or cooling devices.

Next, we have a somewhat spirited debate over when C will do the job or when assembly language must be called to the task.

Finally, we take a quick look at sensors for measuring hydrocarbons and carbon monoxide.

High-frequency current measurement

Msg#:10328

From: ALIN LAVARIERE To: ALL USERS

Does anyone know how to measure current through a load when the frequency is one megahertz? I have a 500-watt linear amplifier driving a bank of light bulbs as a load to prevent mismatch for what I'm trying to do. I am trying to measure leakage current in the 100-milliamp range through a test fixture for electrosurgical devices. The voltage is 800 volts peak-to-peak, or 285 volts RMS. The voltage is taken from across the light bulb load bank, and applied to the device under test. I need high accuracy—1 %—so a thermocouple-type ammeter won't do. I am using a Fluke DVM with an RF demodulator probe. I have a 10-ohm resistor in series with the device under test, so I measure the voltage across it and calculate current with Ohm's Law ($I = E / R$).

My problem is that I get high-voltage readings across the resistor—40 volts—but the 1-watt resistor is cool (calculations suggest 4 amps flowing!). This output power seems difficult to work with.

If you can help by suggesting anything, I will be delighted!

Msg#:11101

From: PELLERVO KASKINEN To: ALIN LAVARIERE

You want 1% accuracy of current readings at 1 MHz? You don't want to push the state of the art, do you? ;-) The fact is, NIST (formerly NBS) did not used to certify any instruments to much better than that! So, let's take a little closer look at what is going on.

The first issue to make sure of in your measurements is that you have as purely resistive a shunt as possible. Wire wounds are a definite no-no. The carbon film resistors probably don't do as well as carbon composition, either. But even the best resistors have more than zero length, which translates into more than zero inductance.

Assuming you have a good carbon composition or a bulk metal resistor, you then want to make sure your measuring system does not pick up anything else than the actual signal generated by the resistor. Here we are talking about common-mode signals (your 285 V supply). Any meter has a certain, limited ability to reject the common-mode disturbance. It typically decays by increasing frequency. May start at 60, 80, maybe even 120 dB at 60 Hz, but then goes down at, say, 20 dB per decade of frequency. If you had 80 dB at the beginning, at 1 MHz you have next to nothing. Discouraging?

We have some common-sense remedies. The first and foremost is to try to break the common-mode loop with appropriate use of differential measuring techniques. You could make a 1:1 turn-ratio transformer from a ferrite toroid and then use coaxial cable to bring the signal to the RF probe. At that point, make sure the probe ground lead is used rather than a separate lead and it is connected to the shield of your cable plus to your safety ground (you do have one, don't you?) at the same point. This ought to bring the common-mode signal you feed to your RF probe down to tolerable levels if your transformer is made with a good physical separation of the primary and the secondary windings. The less capacitance you produce there the better.

Now, after all this, a couple of practical suggestions. You might rent a Tektronix A6302 current probe with AM503 amplifier. That is how the other people handle similar situations. Check with places like Electra-Rent, U.S. Instrument Rentals, or Leasa-Metric. But do not expect anybody to guarantee a 1% accuracy. The other thing to do is to work your measuring arrangement into such a form that you have inside a Faraday cage the necessary supply, a table that insulates your target from the ground, and then a single cable through your shunt to ground from the target. That way your shunt is not floating up at the 285-V potentials and you can use your RF probe properly. What I mean is, the shunt is actually soldered or welded to the cage at

CONNECT TIME

one end and you use that as the sole ground point for your RF probe. You also bring the ground side of your signal source as close as possible to the same ground point. This is the second way the FCC compliance measurements are performed.

No matter how good your results appear to be, they are not worth much until you can reproduce the same results in an independent way. You might, for instance, try to use a known capacitance as the sole leakage path through to the shunt and see if the measured current matches the calculated one. And to eliminate some of the antenna effects, you probably want to make this "calibration" twice, with different capacitor values.

Solid-state cooling devices

Msg#:11452

From: CHRIS GATES To: ALL USERS

Has anyone out there used a solid-state cooling module? I believe these are nothing more than large thermocouples, but beyond that I have no knowledge about them. I am looking for a source for them as well as any technical tidbits (such as power consumption) that I can get.

Msg#:11478

From: TOM MAIER To: CHRIS GATES

I think I know what you are talking about. They go by several names. They are not thermocouples. One name is "Peltier Crystal," named after the person who discovered the effect. It is a crystal sandwiched between plates of metal. When you apply a current through the sandwich, one side of the thing gets hot, and the other side gets cold. Reversing the current direction reverses which side is cooled or heated. Neato, huh? You can kludge these things onto anything you wanted to cool and or heat. I have used them for cooling laser diodes and I have seen them advertised for cooling ICs.

Another name is "thermoelectric cooler," "TE cooler," "solid-state cooler," and a bunch other names. I recently saw a six-pack cooler made by Coleman that had TE crystals mounted in the thing and you can run it off your cigarette lighter.

Msg#:11486

From: DAVE TWEED To: CHRIS GATES

Actually, they're called "Peltier Junctions" and there's no crystal involved. In fact, they are simply many thermocouples in series, physically arranged so that all the "cold" junctions are on one side of a plate and all the "hot" junctions are on the other. You typically run a few amps

through one to get the heat-pump effect. In essence, you are forcing the thermocouple effect to "run in reverse" by applying a current. If you keep one side hot and one side cool, you can actually extract power from one of these.

Recently, they have been used extensively to stabilize the temperature of solid-state laser diodes in laser printers-if you can find a junked mechanism, maybe you can get the cooler out of it. I know you can buy functional laser assemblies (laser diode + cooler + lens) on the surplus market for about the same price as a small HeNe tube.

Msg#:11492

From: FRANK KUECHMANN To: CHRIS GATES

I think you're probably referring to solid-state heat pumps based on the Peltier effect. Peltier discovered in 1834 that a current passing through the junction of two dissimilar conductors either cools or heats the junction, depending on the direction of the current. The degree of heating/cooling is directly proportional to the current.

Commercial thermoelectric heat pumps are essentially arrays of P-doped and N-doped bismuth telluride in series electrically and in parallel thermally. In an open circuit, each P-N pair acts as a simple thermocouple that produces a voltage proportional to a temperature gradient across it.

If you connect that P-N pair to a DC voltage, it absorbs heat (cools) at one end and emits heat at the other. If you solder a bunch of these P-N pairs to copper strips, then add ceramic (electrically insulating) face plates, you have a typical thermoelectric heat pump module.

Thus they're groups of small thermocouples rather than single large ones. Power consumption is relatively high, and it must be low-ripple DC.

Two manufacturers are Melcor (Trenton, N.J.) and Cambion (Midland Ross, Electronic Connector Division, Midland-Ross Corporation, One Alewife Pl., Cambridge, MA 02140; order through distributors like IPI, 2601 S. Garnsey St., Santa Ana, CA 92707). Minimum orders typically \$100-\$150. They can occasionally be found in singles (\$15-\$30) at outfits like M.P. Jones and American Science & Surplus (nee JerryCo).

Msg#:11504

From: P. EDWARD BECKER To: CHRIS GATES

I have one right in front of me, mounted to a *large* heat sink (the cold side gets cold, but the hot side gets REAL hot). My boss had some surplus electronics in his garage and was going to throw it away. He brought in a box full, and I found this thing. After reading the tag, I realized what it must be, took it to the lab, and hooked it up to a power supply. IT REALLY SUCKS JUICE, but it gets real cold/hot. Hey, does anyone know if it is possible to burn one of these up? I've been putting a resistor in series with it.

CONNECT TIME

Msg#:11798

From: FRANK KUECHMANN To: P. EDWARD BECKER

I don't know whether it qualifies as "burning one up," but you can melt the solder that holds the heat pump together if you run too much current through it. "Too much current" varies with the part, but the small units I've worked with are limited to 2.5 amps and 3.75 volts.

Inadequate heat sinking can also cause meltdown. If you have run yours any amount of time without problems, you're probably safe with your current resistor.

Msg#:12016

From: FRANK KUECHMANN To: CHRIS GATES

Cat #22627, \$25 from American Science & Surplus, P.O. 48838, Niles, IL 60714-0838, (708) 4758440, fax (708) 864-1589.

C versus assembler

Msg#:11567

From: VU NGUYEN To: ALL USERS

Help! I'm working on a project that implements a Siemens SAB80C537 microcontroller (high-performance version of 8051). I need serial interrupt routines that let me use BOTH built-in serial ports of the controller at rates of greater than 9600 bps. I use Franklin's C51 compiler version 5 (their latest). I would appreciate any suggestion/hints.

Msg#:11620

From: ED NISLEY To: VU NGUYEN

My dipstick test says that you'll probably need to code those serial interface handlers in assembler rather than C. It goes a little something like this: at 9600 bps you will get two interrupts (one on each channel) every 1000 μ s. If you have lots of computations to do, you want to spend less than half your time in the serial handlers, which limits you to a path length of less than 250 μ s per interrupt.

Typical 8051 code (mine, at least) runs around 1.25-1.5 cycles per instruction, so each handler weighs in at less than 200 instructions. Typical C code (mine, at least) runs around 12-15 instructions per line, so you've got maybe 16 lines of code.. .and that's a tough row to hoe!

I think you can probably adapt the interrupt handlers I've done for a variety of projects over the years. Take a look though your collection of INK back issues and see if anything strikes your fancy; the PL-Link project last year is probably a good starting point. You might be able to share the code between the two ports, but, given the 8051's peculiar bit addressing, it might be easier to just replicate the grubby parts with a few changes and be done with it.

Msg#:11787

From: JIM WHITE To: ED NISLEY

Actually, carefully written Franklin C51 C code produces optimum 8051 assembly code. The tricks revolve around using the memory space "hint" keywords for allocating variables and specifying what memory space a pointer points to. Also, careful arrangement of the procedure call tree, if any, so that the register allocation algorithm keeps all or most of your variables in registers. Rearranging expressions and avoiding expressions that require intermediate storage (i.e., prefer ++x over x++ in value contexts) also shaves cycles. The effort required is only somewhat less (if any) than writing assembly language directly, but the results are understandable by more people, and portable as well (with judicious use of macros and conditional compilation).

As for some actual serial interrupt service code, I can't distribute any of mine, but I recall seeing some on a BBS very recently. Either it was here in one of the "miscellaneous" areas or it was on Franklin's BBS (408) 296-8060.

Msg#:11844

From: ED NISLEY To: JIM WHITE

Mmmmm.. .minor quibble: writing "optimum" assembler code in C requires that you have intimate knowledge of how the compiler works, how the 8051 works, and how the two fit together. While you can twiddle the code so this version of the compiler does precisely what you want, and the result does look pretty much like C, I pity the poor guy who has to make "one little change" in that whole edifice!

Given that most interrupt handlers are short and to the point (yes, I've written some exceptions to that rule, too), would it not make more sense to write that code in a short and to-the-point assembler routine?

Methinks anyone working on 8051 code who =cannot= understand that kind of code is in deep yogurt on other grounds.

Msg#:11858

From: JIM WHITE To: ED NISLEY

I readily acknowledge that coaxing the C compiler to give the code you want requires all the same knowledge needed for mixed C and assembly language programming, plus something about code generation by compilers.

The benefit is in reducing the amount of assembly language, which is less portable by far than C code. Much of the code which requires this sort of tweaking is communications code and timing code. It is very nice not to have to maintain multiple versions of communications code for multiple platforms. C code tweaked for the 8051 often generates nice code for other CPUs as well. I don't agree

CONNECT TIME

that the same knowledge is required to understand the code as is required to create it. If that were true, I would have much more trouble in teaching new programmers. Example is a great teacher.

None of this is to say there aren't times when dropping to assembly language is necessary and/or desirable. But I do say that situations where additional speed is needed rarely *require* the use of assembly language.

Msg#:11865

From: DAVE TWEED To: JIM WHITE

I completely agree with your points. I have been using the Franklin C compiler for a medium-large project (about 25K bytes of executable), and staying in C wherever possible is extremely valuable. Besides, I find that the source-level techniques that cause the compiler to emit good code are, for the most part, good techniques to use in C coding anyway. There are some specific gotchas (like avoid using more than three arguments to a function, and stay away from library routines that use "generic" pointers) that wouldn't apply to a more general-purpose architecture, but don't really obfuscate the kind of code you run on an 8051 in most cases. In the remaining cases, judicious use of `#define` and `#ifdef` can help document the tweaks in a reasonably portable way.

Msg#:11977

From: ED NISLEY To: JIM WHITE

Mmmm.. .OK, I'll yield the point with one quibble.

I think it's dangerous to assert that optimized code for one CPU is pretty good on another. It may be true for a given class of CPU (8-bit with lousy index operations, for example), but is certainly not true in general.

As a case in point, the old (and I hope obsolete) Avocet C library had a bunch of routines that were "optimized" for 16-bit CPUs. They generated =terrible= 8051 code, in part because of the optimizations, and I found that the only way to get decent performance was to do 'em in assembler. That may not be true with a better code generator, of course.

Methinks we're in violent agreement on one topic: you =really= can't take anything for granted!

Msg#:12065

From: JIM WHITE To: ED NISLEY

I agree that we agree.

I got started on this because of an comment that code for handling moderately high serial interrupt rates would require assembly language. My modest point was that such was not necessarily the case.

The situations in which one is forced into assembly are many.

A very recent one for me is I tried to implement a

Fletcher's checksum routine in Franklin C5.1. After a modest amount of tweaking, I got code that was several times larger than the equivalent assembly code, the reason being that the compiler would not recognize the "trick" needed for the good assembly code. But that is not the reason I coded the routine in assembly. I could almost accept the overhead, but I could not accept the *incorrect* code the compiler generated. The Franklin C51V3.20 has numerous bugs related to the promotion of `unsigned char` to `int` for all sorts of expressions.

Msg#:12101

From: ED NISLEY To: JIM WHITE

Don't get me started on code-generation bugs!

Well, just one story.. .one version of Microsoft C (back around 5.0 or so) had =real= code-generation problems. I wound up with code that compiled correctly with debugging turned on and failed with debugging turned off. Talk about tearing your hair out!

Ever since then I've been a big fan of looking at the assembler output just to see what's going on.. .but they also had problems where the "assembler output" file didn't match the actual code in the "object file" that got linked into the program.

I wasn't the only one with such problems.. .

Msg#:12132

From: JIM WHITE To: ED NISLEY

Ditto.

This sort of thing could go on indefinitely.

Last year, using the Microsoft linker from MSC 5.1, the '386 BIOS I was writing for a pen-top machine suddenly quit working. Add some code, it goes out to lunch, take it out it works. It was blowing up long before it could reach the new code. Much tearing of hair. Finally (after going back to initial bootstrap debug mode) I discovered the absolute references to the BIOS's segment were all OOOO! When some table size crossed some threshold, the linker quit fixing up those references. That one cost me four hours.

Switched to Borland TLINK, no more problems.

Hydrocarbon and carbon monoxide sensors

Msg#:11518

From: RUSS REISS To: ALL USERS

Can anyone give me information on how the hydrocarbon (HC) and carbon monoxide (CO) sensors like those used to measure auto emissions work? What principle? Where can they be purchased? What to watch out for in applying them? Thanks!

CONNECT TIME

Msg#:12165

From: BOB WEINBERG To: RUSS REISS

CO sensors have long been made using the tin oxide "pellister" sensor. This is heated ceramic bead having a tin oxide coating. In the presence of levels of C, there is an exothermic reaction which occurs on the sensor and temperature increases. I believe this is the idea behind many sensors which are similar, using other materials.

There are also chemioptical sensors; these react more like the body itself does when exposed to CO. It is a dosage-type detector, and it also is reversible. The sensor is maybe 0.25" diameter and 0.1" thick, an amber natural color. When exposed to concentrations of CO -say 75 to 400 PPM-it darkens the longer it is exposed. The sensor is usually configured to darken to a threshold of 200 PPM for 1 hour, a standard being considered a caution level by the ANSI Z21 working group (I've attended these meetings).

There is lots of information available, and commercially available stuff ranging from \$5 things to stick on your refrigerator as a monitor to \$50 things which are like smoke detectors but alarm on CO levels. One manufacturer is Quantum in San Diego, California. There are others.

We invite you call the Circuit Cellar BBS and exchange messages and files with other Circuit Cellar readers. It is available 24 hours a day and may be reached at (203) 871-1988. Set your modem for 8 data bits, 1 stop bit, no parity, and 300, 2200, 2400, 9600, or 14.4k bps.

Software for the articles in this and past issues of *The Computer Applications Journal* may be downloaded from the Circuit Cellar BBS free of charge. For those unable to download files, the software is also available on one 360K IBM PC-format disk for only \$12.

To order Software on Disk, send check or money order to: The Computer Applications Journal, Software On Disk, P.O. Box 772, Vernon, CT 06066, or use your VISA or Mastercard 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.

IRS

425 Very Useful 426 Moderately Useful 427 Not Useful

How Can You Catch Nasty Race Conditions That Take All Night to Happen Without Waiting All Night?

CodeProbe.™

The new high-performance software analyzer that captures, time-stamps, and records software and hardware interrupts, DOS calls, BIOS interrupts, and user-defined events in real-time for analysis of race conditions, interrupt activity, and service times. CodeProbe gives you the hard facts you need to fix the big one that stands between system test and shipping your product.

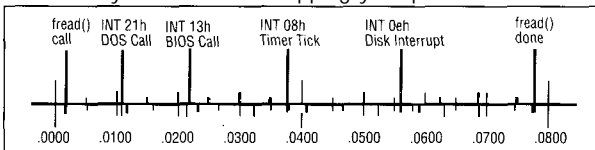


Figure 1. Detailed timestamping of C library fread() function call.

If CodeProbe can break-down a library function call into its components (above), imagine how you'll see context switches, device interrupts, and other asynchronous code

Call today for free technical specifications!



GENERAL SOFTWARE™

Tel 206.39 1.4285

Fax 206557.0736

BBS 206.557.4BBS

P.O. Box 2571, Redmond, WA 98073

copyright(C) 1993 General Software, Inc. All rights reserved. General Software, the GS logo, and CodeProbe OR trademarks of General Software

Cross-Development Tools

from \$50.00

Cross Assemblers

- Extensive arithmetic and logical operations
- Powerful macro substitution capability
- Unlimited include file capability
- Selectable Intel hex or Motorola hex object file format

Simulators

- Ten user-definable screens
- Unlimited breakpoints and memory mapping
- Trace file to record simulator session

Disassemblers

- Automatic substitution of defined label names for all jumps and branches
- Automatic insertion of supplied comments and expressions

Broad range of processor specific tools - Intel, Motorola, Zilog, RCA, Rockwell ..

All products require an IBM PC or compatible, MS DOS 2.1 or greater

Same day shipment VISA, MasterCard, American Express, and COD

Unlimited technical support Thousands of satisfied customers worldwide

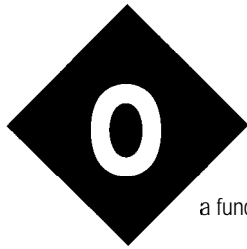
PseudoCorp

716 Thimble Shoals Blvd.
Newport News, VA 236C.5

(804) 873-1947

FAX:(804) 873-2154

BBS (804) 873-4838



Eat at Joe's

One of the first things we do as human beings is learn to communicate. That first scream for attention plays a fundamental function in our survival. Imagine for a moment what it would be like if you could no longer communicate or you were unable to understand others trying to communicate with you. RADIOACTIVE AREA and EAT AT JOE'S would have about the same significance.

The purpose and decided benefit of communicating is that it allows us to interact and, to a limited extent, exercise control over our world. Oral or written exchange is our normal method for influencing or directing the activities of others around us. Typing a command sequence or pushing an activation button is the method we employ to interact with machines. Either way, such tasks involve diverse skills.

Using such communication skills, we continually update our knowledge and understanding of the world around us so that we can better compete in it. Of course, decisions are made based on our current understanding of a situation, which in turn is related to the magnitude, freshness, and credibility of the information we have absorbed in relation to that topic. The quality of our decisions is directly related to the quality of our information. Our ability to communicate that intelligence dictates the character and complexion of the exchange.

In an information-driven age, suitable communication demands superior skills and tools for us to advance beyond the ability to simply exchange ideas. Keeping up on our understanding and the application of these tools demonstrates how effectively we will communicate in the new electronic world.

It has been speculated that the power brokers of the future will be primarily those who control the flow of information or those who understand the information and its implications. To participate in this evolving information society, we have to continually modernize the way we capture and absorb information. We have to find better ways to collect raw knowledge while at the same time improve the filtering algorithms which reduce this massive data overload into useful intelligence. Quite a task indeed.

All of these ideas are food for thought and I claim no solutions or pious prophecies. I sit here pondering the future in the presence of today's telephone message cassette, magazine on disk, and a pile of electronic as well as printed mail. Within my view are the fax machine, high-speed modem, '486 computer with mounds of application software, a cellular phone, an electronic memo pad, an electronic address directory, a radio subcarrier broadcast decoder, and an automatic telephone with a dozen telephone lines. No longer executive toys to signify accomplishment, somehow this paraphernalia has become elevated into tools for survival.

A handwritten signature in black ink, appearing to read 'Steve', is located at the bottom right of the page.