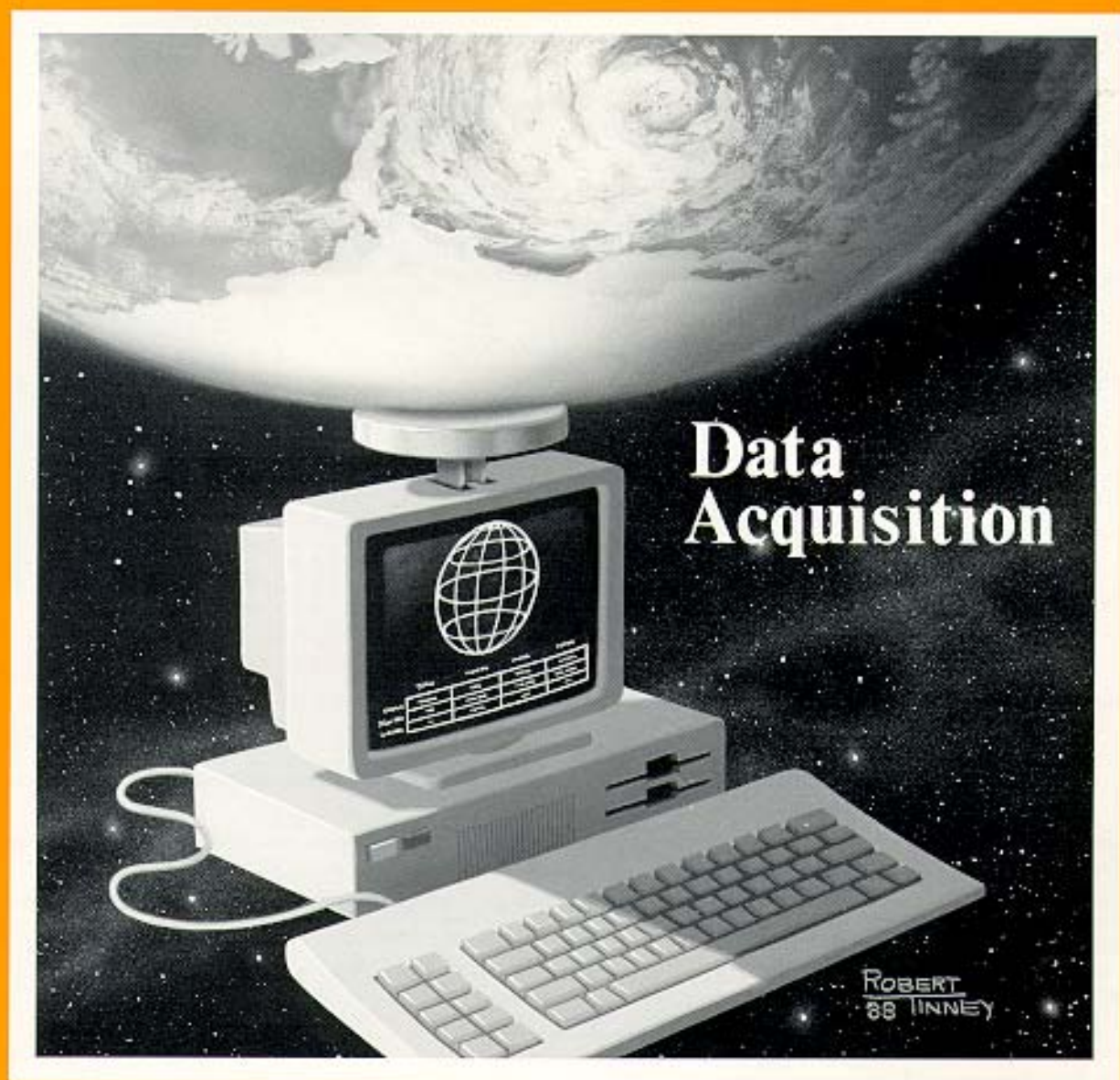


CIRCUIT CELLAR

Ctrl I N K

THE COMPUTER APPLICATIONS JOURNAL



November/December 1988 - Vol. 1, No. 6

\$3.95

CIRCUIT CELLAR INK®

Ctrl

PUBLISHER
Daniel Rodrigues

EDITORIAL

Steve Ciarcia

EDITOR-in-CHIEF
Curtis Franklin, Jr.

EDITORIAL ASSISTANT
Rose Mansella

TECHNICAL EDITORS
Ken Davidson
Jeff Bachiochi

CONTRIBUTING EDITORS
Thomas Cantrell
Edward Nisley

CONSULTING EDITOR
Harv Weiner

CIRCULATION DIRECTOR
Jeannette Dojan

CIRCULATION CONSULTANT
Gregory Spitzfaden

PRODUCTION MANAGER
Tricia Dzedzinski

John Hayes

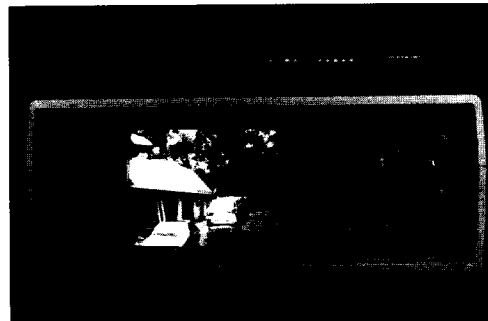
STAFF RESEARCHERS

Northeast
Eric Albert
William Curlew
Richard Sawyer
Robert Stek
Midwest
John Elson
Tim McDonough
West Coast
Frank Kuechmann
Mark Voorhees

FEATURES

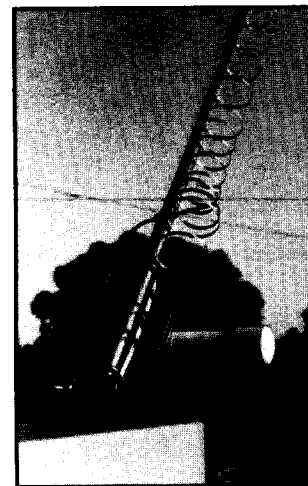
- 6 ROVER: Remotely Operated Video-based Electronic Reconnaissance -- Part 2**
The Software
by Steve Ciarcia & Ken Davidson

Software completes the picture of this portable remote video system built from off-the-shelf parts.



- 20 The Satellite Home Weather Center -- Part 6**
Adding Serial and Parallel Ports to the Peripheral Controller
by Mark Voorhees

Standard and custom I/O for interfacing two Heathkit weather centers to the peripheral controller.



DEPARTMENTS

Editor's Ink

An Active and Growing Industry
by Curtis Franklin, Jr. ————— 2

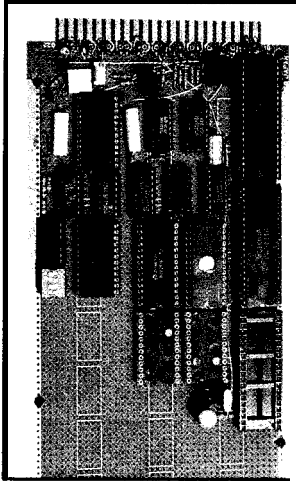
Reader's Ink - Letters to the Editor ————— 3

Visible Ink - Letters to the CCINK Research Staff ————— 16

From the Bench

Fiber Optics
Your link to the future
Conducted by Jeff Bachiochi ————— 28

THE COMPUTER APPLICATIONS JOURNAL



45 Build a Remote Analog Data Logger -- Part 1 A Simple 6809-Based Data Acquisition System

by R. W. Meister

Low component count and simple construction mark the hardware portion of this highly accurate **analog-to-digital** conversion system.

36 ImageWise/PC -- The Digitizing Continues -- Part 1 Video Basics

by Ed Nisley



A higher digitizing rate and more functionality introduced with the PC-bus (ISA) version of one of Steve Ciarcia's most popular projects--the **ImageWise** Video Digitizer.

Excerpts from the Circuit Cellar BBS

Conducted by Ken Davidson

DDT-51 Revealed

by Ed Nisley

A Gathering of Eagles

by Steve Ciarcia

Cover Illustration by Robert Tinney

Circuit Cellar BBS - 24 Hrs.
300/1200/2400 bps, 8 bits, no parity, 1 stop bit, 203-871-1988

The schematics provided in Circuit Cellar INK are drawn using Schema from Omaton Inc. All **programs** and schematics in Circuit Cellar INK have been carefully reviewed to ensure that 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 the possible variation in the quality and condition of materials and workmanship of reader-resembled 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.

CIRCUIT CELLAR INK (ISSN 0896-8985) is published bimonthly by Circuit Cellar Incorporated, 4 Park Street, Suite 20, Vernon, CT 06066 (203-875-2751). **Second-class** postage paid at Vernon, CT and additional offices. **One** year (6 issues) charter **sub**scription rate U.S.A. and possessions \$14.96, **Canada** \$17.96, all other **countries** \$26.96. All subscription **order**s payable in U.S. funds only, **via** international postal money order or check drawn on U.S. bank. Direct subscription orders to Circuit Cellar INK Subscriptions, 12 Depot Sq. Peterborough, NH 03458-9909 or call (203) 876-2199. **POSTMASTER: Please** send address changes to Circuit Cellar INK, Circulation Dept., 12 Depot Square, Peterborough, NH 03458-9909.

Entire contents copyright 1988 by Circuit **Cellar** Incorporated. All rights reserved. Reproductions of **this** publication in whole or in part without written consent from Circuit Cellar Inc. **is** prohibited.

Ctrl

EDITOR'S I N K

An Active and Growing Industry

In October, I went to a small computer show that had a lot of exciting displays and conference sessions. **BUSCON '88** was held at the Javits Convention Center in New York, and featured exhibits from most of the big players in industrial control and general-purpose buses. There were companies exhibiting **Multibus I & II**, VME bus, STD and STE buses, **NuBus** and more, and it was good to see the rate of development for all of these architectures. Of course, being 1988, most of the talk centered on 16- and 32-bit buses, but there is still a lot of vitality in the mature 8-bit bus families.

If you read the general computer press, it's easy to get the impression that desktop platforms for word processing and spreadsheets are the only computers being built. The reality is that, while PCs and their kin are important, there's more happening in single-board and dedicated control applications than meets the eye. From machine vision and robotics to process control and data acquisition, we are part of a dynamic industry, an industry that develops many of the concepts that the desktop people pick up down the road. Right now I'm wondering whether I'll be as impressed by the products at **COMDEX** as I was by what I saw at **BUSCON**.

The products I saw ran from the inevitable (a PC-clone on a VME bus board) to the amazing (a **Multi-bus II** disk controller that uses an 80386 for its on-board intelligence). I talked to a lot of people, and you'll be seeing the results of some of those conversations in the months to come. One of the most important things coming your way will be the series on the various buses themselves. Engineers tend to get religious about their favorite bus, so we will try to give you the facts about the bus, its characteristics, advantages, limitations, and design goals, without getting stuck in the "My bus is better than your bus" debate.

into the Future of Circuit Cellar INK

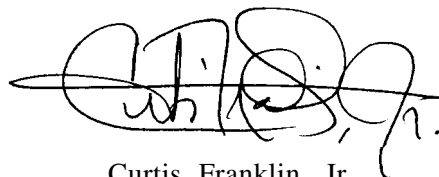
In the world of publishing, there are large magazines that serve a diverse audience, and there are smaller magazines that speak to the particular needs of a special audience. Circuit Cellar INK is, thankfully, one of the latter. When we put together an issue of this magazine, we concentrate on how each article will appeal to one group of people: Those who design, build, and program working computer applications.

In the first year of publication, we have concentrated on articles that closely follow the format laid out by Steve Ciarcia in his popular "Ciarcia's Circuit Cellar" series in **BYTE Magazine**. Looking ahead to year number two, we will still have buildable application projects at our core, but we will be adding articles that we believe will help you broaden your applications horizons and deepen your understanding of current technology.

Starting in Issue 7, we will have some "software only" articles. The first will show you how to write a real-time operating system for a single-board computer. Later in 1989, we are planning to begin tutorial articles on various bus designs and the numerous serial communications standards. We will also have articles describing important new processors and controllers as they are introduced. The common thread through all of these articles will be information aimed at letting you take what you read and apply it to your applications.

This has been a great year for Circuit Cellar INK. We've grown from a first issue that started life as a newsletter to a magazine that is rapidly winning respect throughout the industry as the source for honest, genuine computer applications information. Since I came on board in late July, I have heard a lot of you tell me how glad you are that there is a Circuit Cellar INK, and I've gotten good suggestions on what you want to see in "your" magazine. We depend on your direct input for ideas and inspiration.

Take a minute and drop me a line or give me a call. I really want to hear from you.



Curtis Franklin, Jr.
Editor-in-Chief

CM

READER'S | INK

Letters to the Editor

[Editor's Note: A few weeks ago we asked folks on the Circuit Cellar BBS to tell us what they would like to see in upcoming issues of Circuit Cellar INK. Here are a few of their answers.]

Regarding articles on kit building, I have built many projects from kits, primarily Heath and CCI. I am not interested in things along the lines of "Building the **Heathkit** Something-or-other." I would be interested in articles that tell how to take a commercially available kit and modify it to do something else or that shows how it works. I subscribed to the **Heathkit** "Kit Builders Journal*" for a while in the hope that it might contain some inside info from Heath. Sadly, it never appeared to have much in the way of technical articles. Short pieces on kits that are available from obscure manufacturers would be valuable (such as the PT-68K from Peripheral Technology). I will always be interested in projects that offer some form of kit for those of us that do not have the time or equipment to build from scratch.

Articles that I would like to see include:

- D/A for the BCC bus
- Motion control -- Stepper and servo motors
- Optical Encoders -- Feedback for motion control
- Software for data acquisition and control (Z-Transforms boggle my mind)
- How to use **RS-422/485** (SN75176 Transceiver)
- Building a logic analyzer (with parts sources)
- Applications ideas (like ROVER)

I enjoy reading Circuit Cellar INK and realize that not every article will interest me. I hope that INK will get ads from companies that can furnish the **hard-to-find** parts. I think that this type of advertising will help ensure the long-term survival of INK.

William Giles

Being an amateur kit builder, I would like very much to see some articles about kit building. I realize that most of INK's readers are more advanced than I am, but I really need to know more about what I am doing. I would also like to see some product reviews, such as EPROM programmers and test equipment. As far as what type of projects I'd like to see, almost anything goes! Although I don't build every project, I have on many occasions taken part of the project and incorporated it into other things. To be honest, I read INK to learn about electronics. I'll be going back to college soon and hopefully by 1993 I'll have a BSEE degree. Until then, INK is my teacher.

Brian Joseph

I would very much like to see stuff on design and fabrication tools. My company has spent several thousand dollars on PCB-CAD tools for the PC to find that most of them are junk. The industry trade magazines usually just republish press releases and so are of no use. What would be really useful would be an examination of design tools available to the serious experimenter/small company. I seem to see either articles about >\$100,000 systems or rehashed press releases.

John Dearmond

I am interested in IEEE-488 (GPIB). I don't know much about it technically, that is, what it would take to build a project from scratch for PC-compatibles. [I would like to use it as] a skeleton for managing multiple processes (e.g., an optical reader controlled from the PC console, with real-time counter updating, etc.). Thanks, and keep the Circuit Cellar **INKs** coming in the mail!

Robert Schuh

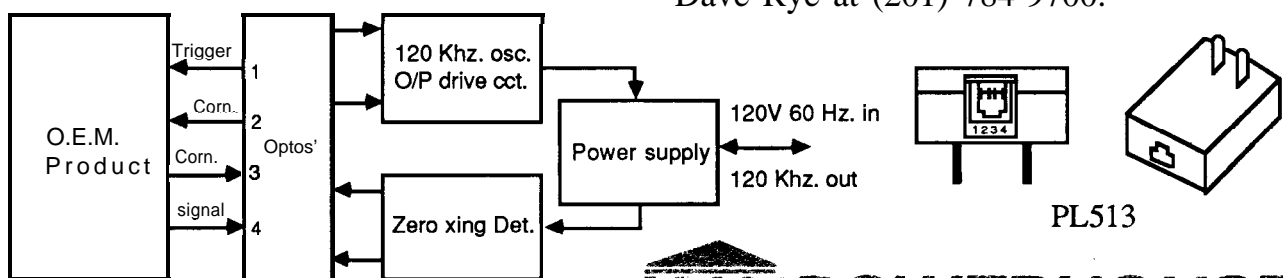
If you're interested in Home Control talk to the World Leader!

X-10 has been in the Home Control business since 1978 (formerly marketing under the brand name **BSR SYSTEM X-10**). We manufacture versions of the system under private labels as well as under the **X-10 POWERHOUSE** name. We have developed and manufactured O.E.M. versions of the system for many major corporations, and there are also compatible systems which use our protocol and transmit to X-10 modules manufactured and supplied by us.

Our capabilities range from supplying "standard product" in any color with any name on it, to development of completely new systems which can take advantage of the large installed base of X-70 modules.

The latest addition to the X-10 line is the Power Line Interface, Model # PL513. This enables any O.E.M. product to couple an X-10 signal to the AC power line without having to bring 120V anywhere near the O.E.M. product. There will also be a 2-Way version available later this year which will allow an O.E.M. to transmit and receive X-10 signals.

The X-10 POWERHOUSE system is considered by many to be the "De Facto" standard for Power Line Carrier Transmission. So if you would like to add to your product - any product, the capability of controlling electrical devices over existing house wiring, or receiving signals from remote sensors, please call: Dave Rye at (201) 784-9700.



PL5 13 Block Diagram

X-10 POWERHOUSE
NUMBER ONE IN HOME CONTROL

X-10 (USA) Inc., 185A LeGrand Ave., Northvale, NJ 07647

Announcing

CIRCUIT CELLAR INK

DESIGN CONTEST



FAME!!!

FORTUNE!!!

FUN??:

We've decided that it's high time we found out just what kind of hardware designers the Circuit Cellar INK readers are. To find out, and to give you a chance to work the kinks out of your mental muscles this winter, we're sponsoring the First Circuit Cellar INK Design Contest. This is your chance to see one of your designs published in the pages of Circuit Cellar INK without having to write an article.

For this contest, the emphasis is on embedded controllers. You can design and build anything at all for the contest, as long as it is based on a commercially available controller chip (8052, 8096, 6811, 8742, etc.). The judges will be looking for utility, creativity, professionalism, and elegance in the designs.

To enter the contest, send for a complete set of rules and an official entry blank. Send a SASE to:

Circuit Cellar INK
Design Contest
P.O. Box 772
 Vernon, CT 06066

All entries must be received by midnight on May 1, 1989. An individual may enter more than once, but each entry must be separate, and must be accompanied by a separate official entry form. Winners will be announced in the July/August issue of Circuit Cellar INK.

PRIZES??:

PRIZES!!!

PRIZES!!!

We couldn't hold a contest without prizes, now could we? First prize is worth \$500, Second prize is worth \$200, and Third prize is worth \$100. In addition, the judges may award Honorable Mention prizes worth \$50 and a 1-year subscription (or extension of an existing subscription) to Circuit Cellar INK.

TWO CONTESTS IN ONE!!!

In addition to the general awards mentioned above, we will award a set of prizes to the projects that feature the most cost-effective design. This criteria will give special consideration to those who use simple, low-cost controllers, such as the 8031, 6809, and Z8 in their designs. The prizes for the Most Cost-effective category are exactly the same as those for the general category.

1. All designs submitted must be the property of the person submitting them.
2. Each entry must consist of an Official Entry Form, complete legible schematics for the project, complete documentation for the project, and three (3) photographs of the completed project. All schematics, documentation, and photographs become the property of Circuit Cellar INK and cannot be returned. The actual design and the project itself remain the property of the person submitting them.
3. If your entry is chosen as a finalist, you will ship the completed project to Circuit Cellar INK, where it will be examined, verified, and photographed before being returned to you. All responsibility for shipping and insurance while in transit are the entrant's.
4. A First Prize (**\$500**), Second Prize (**\$200**), and Third Prize (**\$100**) will be awarded in the Open Category and in the Cost-Effective Category. In addition, the Judges may award an unspecified number of Honorable Mention Prizes (**\$50** and a 1-year subscription to Circuit Cellar INK) if the quality of entries makes this necessary.
5. Judging will be on the basis of utility, creativity, professionalism, elegance, and any other criteria the judges may choose to apply. The decision of the judges will, in all cases, be deemed final.
6. Employees of Circuit Cellar INK and their immediate families are not eligible.

ROVER: Remotely Operated Video-based Electronic Reconnaissance

“**A**nd in the continuing saga of ‘As the Garage Turns,’ our hero can be found patiently watching out the front window as the last of the garage’s foundation is poured ...”

It’s true. Mr. Wright finally came to his senses and issued the proper permits. The garage is coming along nicely, thanks in part to ROVER.

It’s time to talk about just what makes ROVER tick. But for those who may have missed the last issue of INK and the action-packed tale of how ROVER earned its keep, let me fill in some details.

ROVER stands for Remotely Operated Video-based Electronic Reconnaissance, and consists of two parts: the digitizer/transmitter and the receiver/display. On the digitizer end is a video camera with a motorized zoom lens, mounted on a motorized pan and tilt mechanism. The video from the camera is digitized by an **ImageWise** serial video digitizer and is sent to a BCC180 computer/controller. Also under control of the BCC180 are all the motorized gadgets (the zoom lens and the pan and tilt). Finally, the **BCC180** is connected to an optoisolated input board for reading alarm status and a **9600**-bps modem for talking to the outside world.

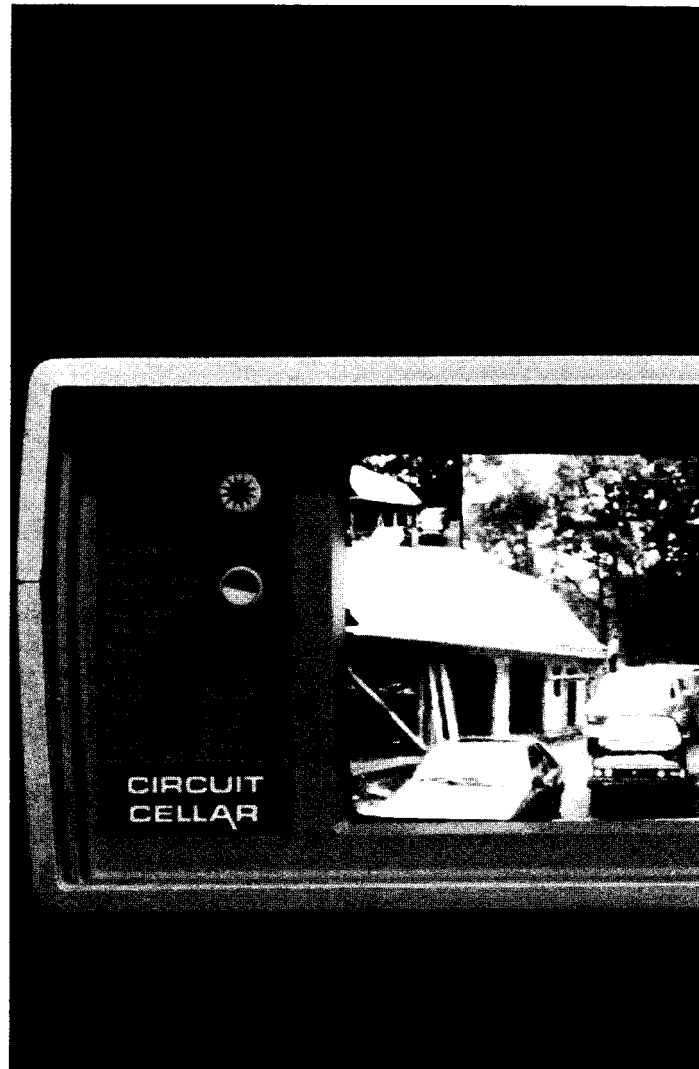
On the display end (which is built into a portable-computer case) is another **9600**-bps modem and BCC 180 computer. Connected to the BCC180 is an **ImageWise**

receiver/display board, capable of displaying a digitized 64-gray-level video image. When a switch is pressed on the display unit, the BCC180 directs the modem to place a phone call to the digitizer. Once connected, the digitizer sends video images from the camera to the receiver/display, where they are displayed on a video screen. Other switches on the display unit allow remote control of the zoom, pan, tilt, and other devices at the house. Status about the HCS and alarm systems sent from the house is displayed on lights on the receiver unit.

One of the nice parts about the system is that I can carry the display unit with me, call home from anywhere there is a telephone, and get instant video pictures of the whole house perimeter plus status information.

Off-the-shelf Boards

As I explained in part one, the vast majority of ROVER consists of off-the-shelf boards that can be readily purchased or built from plans



presented in Ciarcia’s Circuit Cellar articles from BYTE Magazine. What little additional hardware was added can be classified as “bells and whistles.”

Figure 1 shows how the relays on the digitizer end are wired. Each of the eight relays on the

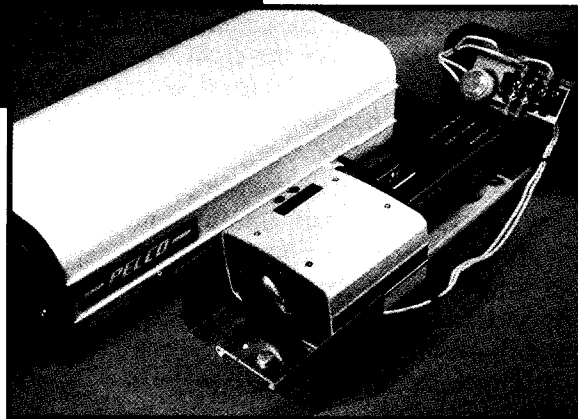
Part 2

The Software

by Steve Ciarcia & Ken Davidson



Photo 1 -- ROVER lets you make a single data call to receive images from, and control the autoiris/zoom/tilt/pan video camera (right) using the compact display unit (above).



BCC40R board is connected to one bit in an 8-bit byte. By sending a single byte to the BCC40R board, any or all of the relays on the board may be turned on or off.

The camera pan control has two separate wires for pan left and pan right. We could very easily just connect pan left to one relay and pan right to another relay, but what happens if a glitch in the software turns both relays on at once? Instead, the pan left and pan right wires are connected through a pair of relays wired such that both can't be activated at the same time. Panning left or right is still as easy as setting one bit or another, but if both bits are set at the same time, nothing will happen and no damage will be done.

The same is true of the tilt controls. Tilting up or down is as easy as setting one of two bits, and if both bits become set at the same time, nothing will happen.

Zooming is a bit easier since a single control wire is used, with the polarity of the voltage applied to that wire determining the direction of the zoom.

Finally, two relays

are used for on/off control of a VCR and the laser.

The extra circuitry added on the display end is more involved. Figure 2 shows how the switch inputs and LED outputs are wired to the BCC180's six parallel I/O ports. Since everything is in one box, protection from input transients is less important than on the rather vulnerable digitizing unit, so the switches are connected directly to the TTL inputs. Likewise, the LEDs are connected (through drivers) right to the TTL outputs. Active-low logic is used throughout. That is, when a switch on an input is open, a pull-up resistor keeps the input at a high level. When the switch is closed, the input is grounded, causing a low (active) level. Likewise, when an output bit is high, the LED connected to it is off. When the output goes low, the LED turns on.

If you want to build ROVER without doing all the extra hand wiring, the system can be built with a fixed camera (no pan, tilt, or zoom) and without status indicators using off-the-shelf devices and little else.

The real secret to making ROVER a reality, however, is the software.

Software for ROVER

While small by today's programming standards (the total size of the assembled code for both ends is only about 2K), listing the source code for the ROVER software would take over 30 pages. Instead, we'll use flowcharts to show you where the code is going.

[Editor's Note: All software mentioned in this article is available from the Circuit Cellar BBS or on the Software Disk for this issue. For information on downloading and ordering, see page 35.]

Since off-the-shelf boards are used in the ROVER setup, very little prototype hardware design was needed (just the connections for the switches and LEDs). The functionality of the whole setup really centers around the software. Even though the BCC180 has a multitasking BASIC compiler available for it, the only way to write tight, efficient, real-time code is in assembly language, so all the ROVER control software is written in HD64180 (280) assembly language. For ease of explanation, the task is divided into two parts: the digitizer/transmitter end and the receiver/display end.

The Digitizer/Transmitter

At the digitizer end, the event sequence is very straightforward. Once the telephone connection is established (more on that later), the BCC180 receives a picture request from the modem. It passes that request on to the **ImageWise** digitizer board which digitizes the video signal coming from the camera on the roof. At that point, the BCC180 just passes the information straight through to the modem.

Why, then, do we need a host computer at all? Because I chose to incorporate two-way communication of status and control parameters in addition to one-way picture transmission in the design of ROVER, a computer is necessary to sift the data stream and channel it correctly. Every 200 milliseconds, the display unit sends two bytes containing front-panel switch information to the digitizer end. That information must be passed on to the relay output boards for use in moving the camera around, zooming in and out, and so on.

Also every 200 ms, the digitizer end reads its optoisolated inputs, which contain alarm status information, and sends them to the

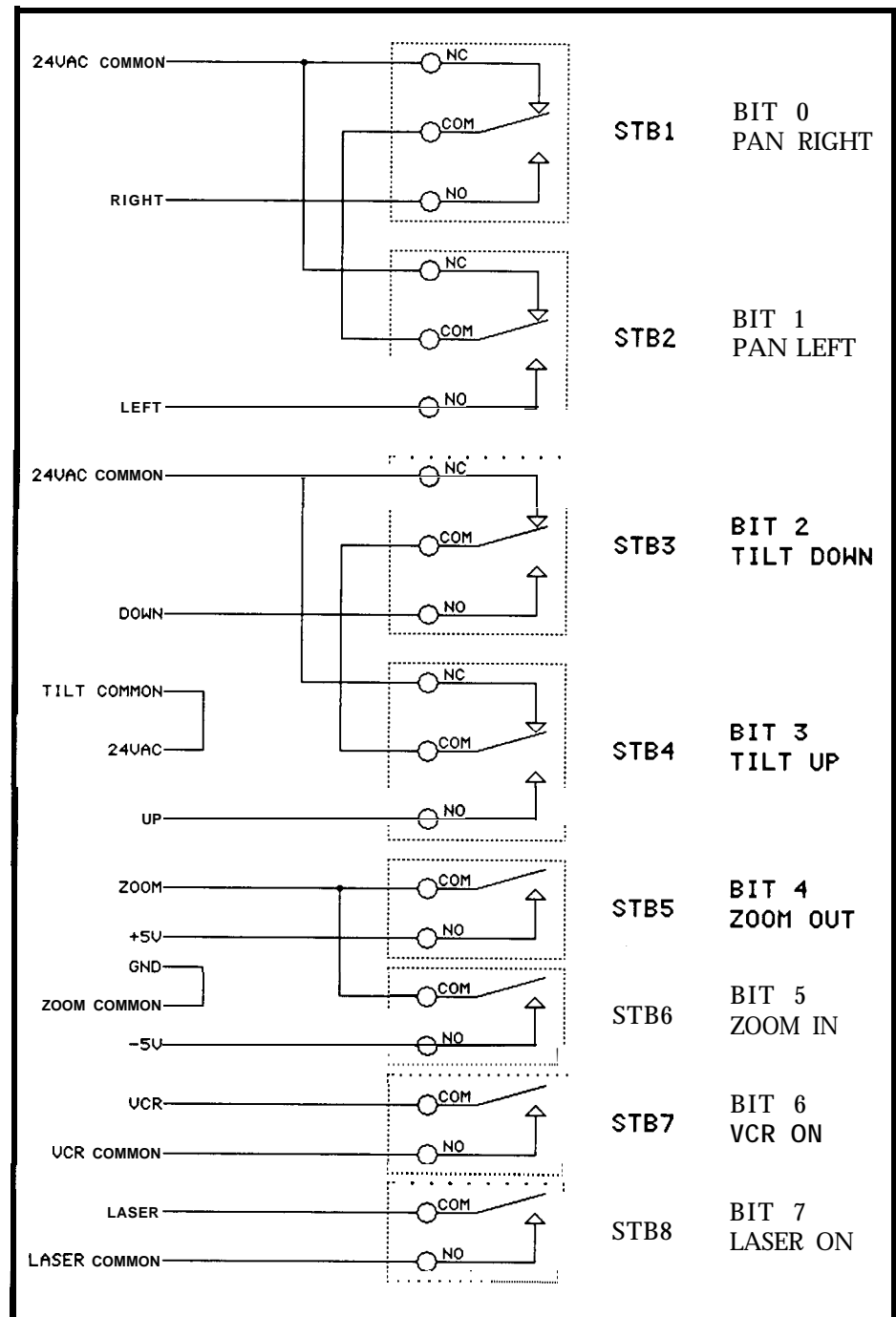


Figure 1 -- Wiring for the eight relays of the BCC40R board. Since each is controlled by a single bit, all eight can be controlled with each byte sent to the board.

display end. The host computers in handle any real-time program-ROVER take care of adding and stripping information to and from the serial stream so the **ImageWise** boards think they are talking directly to each other.

Interrupts are the best way to

resources waiting for an event to happen, ROVER's processor relies heavily on interrupt processing to maximize efficiency.

The HD64 180's on-board serial

THE INTERCHANGE™

Bi-directional Data Migration Facility for IBM PS/2, AT, PC, PORTABLE and Compatibles

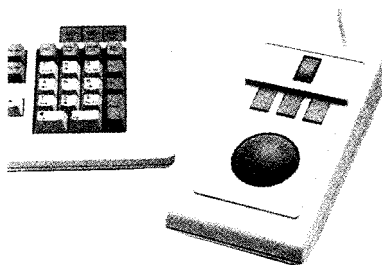


Features:

- *Parallel port to parallel port.
 - *Economical method of file transfer.
 - *Bi-Directional file transfer easily achieved.
 - *Supports all PS/2 systems (Models 30, 50, 60, and 80).
 - *Supports IBM PC, XT, AT, Portable and 100% compatibles.
 - *Supports 3 1/2 inch and 5 1/4 inch disk transfers.
 - *Supports hard disk transfers.
 - *Supports RAMdisk file transfers.
 - *The SMT 3 Year Warranty.
- ONLY \$39.95

FastTrap™

The pointing device of the future is here!



- *Two and three axis pointing capability.
- *High resolution trackball for X and Y axis input.
- *High resolution fingerwheel for Z axis input.
- *Use with IBM® PC's, XT's, AT's and compatibles.
- *Three input buttons.
- *Full hardware emulation of Microsoft® Mouse.
- *Standard RS-232 serial interface.
- *Includes graphics drivers and menu generator.
- *Easy installation.
- *1 year warranty.
- *Made in U.S.A.

ONLY \$149.00



LTS/C Corp.
167 North Limestone Street
Lexington, Kentucky 40507
Tel: (606) 233-4156

Orders (800) 872 - 7279
Data (606) 252-8968 [3/12/2400 8-N-11
VISA, Mastercard, Discover Card,
TeleCheck

Should the checksum of the information byte disagree with the received checksum, all the information is thrown away (the last thing we need is a run-away pan or tilt!). Since the reception of the information packet is interrupt driven, it is again handled automatically and in the background.

The final task is sending status information to the display. We'd like to send it every 200 ms, regardless of what else is going on. The HD64180 has two on-board programmable timers that can be set up to generate an interrupt on fixed intervals of time. Since the timers can't be slowed down enough to generate an interrupt every 200 ms, a 100-ms heartbeat interrupt is established, but information is only sent on every other interrupt. The timer service routine in Figure 3c reads the optoisolated inputs and formats them in the same way as the information packet described above. A header byte is sent, followed by the information byte, then the checksum.

We now have a conflict. The information packet and the digitized picture data both want to be sent out the same port at the same time. We rely on two different interrupt service routines for the information, but only one can talk to the port. To solve the dilemma, we set a flag in the timer interrupt routine which tells the serial transmit interrupt routine that there is an information packet to be sent. Interrupts are then **re-enabled**. The next time a serial transmit interrupt is received, the service routine first checks to see if an information packet is ready. If so, it checks to see what portion must be sent next (header byte, information, or checksum) and sends the appropriate byte. After the last byte of the packet has been sent, the flag is cleared so transmission of picture data can resume. It turns out that sending the information packets every 200 ms only adds about 1.5% to the overall transmission time.

So what's left for the foreground

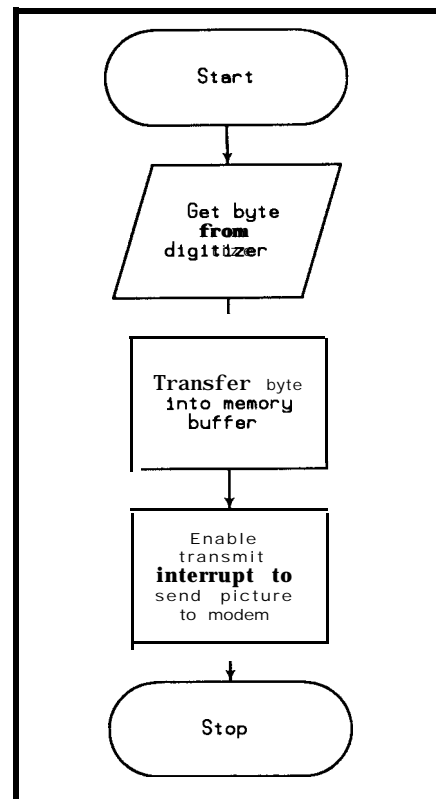


Figure 3a -- Most of the programming for ROVER involves generating and servicing interrupts. The routine for the serial receive interrupt takes care of enabling the transmit interrupt.

code diagrammed in Figure 3d to do? Since everything is interrupt driven, not much. After it sends out the picture request to the digitizer, the foreground code sits back and relaxes, checking the serial transmit interrupt enable bit to know when the picture transmission has finished. (Recall that the transmit interrupt is automatically disabled when the picture has been completely sent.) The code then loops back to the top and waits for another picture request from the modem.

The Receiver/Display

The activities on the digitizer and display ends are very similar. Once we determine the resolution

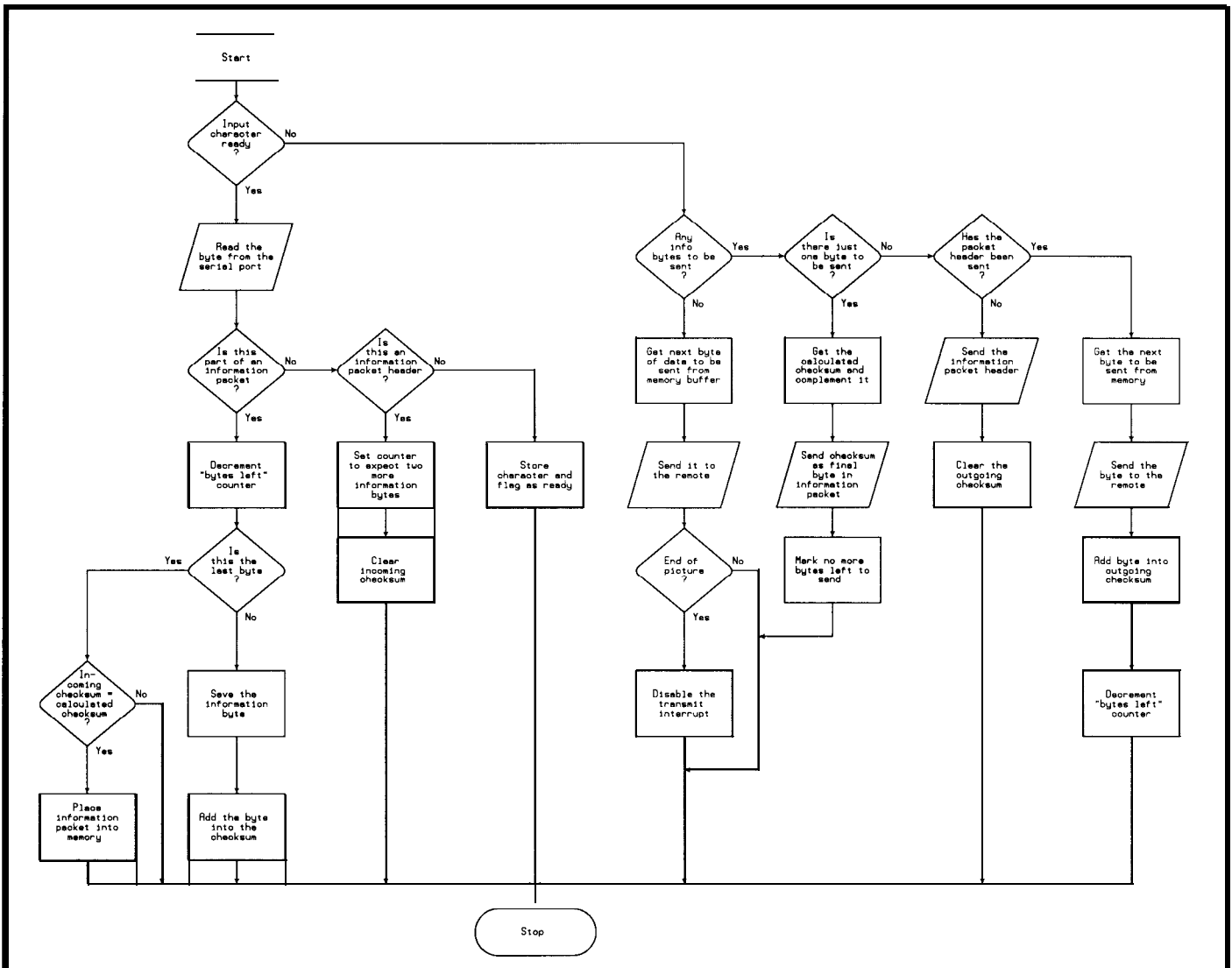


Figure 3b -- The interrupt service routine for the modem port handles information sent from the switches on the front panel of the display unit and also empties the local picture buffer.

of the picture we want, a request is sent to the modem. Once data starts coming in, it is passed through to the ImageWise receiver/display board. Information packets are sent to the digitizer every 200 ms, and information packets received are displayed on the front-panel LEDs.

One of the basic premises of the digitizer end is different than the receiver end, though. On the digitizer end, the buffer is filled by the digitizer faster than it is emptied by the modem, so we can use inter-

rupts to perform both filling and emptying. On the display end, however, communication with the modem is at 9600 bps and communication with the display board is at 19200. In this configuration, the buffer can potentially be emptied faster than it is filled, so it doesn't make sense to use interrupts for both operations. Instead, we fill the buffer using an interrupt (serial receive interrupts are always easier to use than transmit interrupts) and empty the buffer in the foreground code.

After the picture request has been

sent, a buffer input pointer and a buffer output pointer are set up. The buffer input pointer is the address within the buffer of the last byte received. When a byte is received, the pointer is incremented and the byte is placed in the buffer. Likewise, the buffer output pointer is the address of the next byte to be sent to the receiver/display board. The foreground code is shown in Figure 4a (I'll describe the telephone functions in a moment).

Next, the serial receive inter-

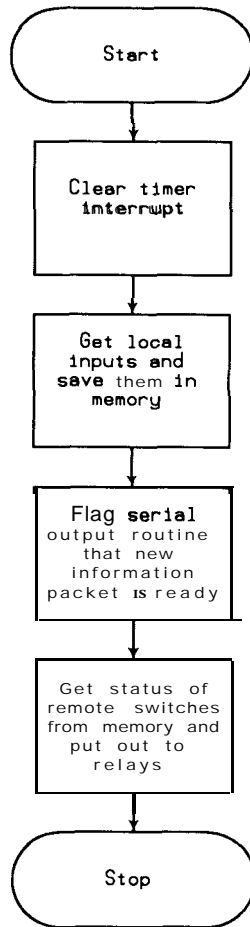


Figure 3c -- This timer service routine reads the optoisolated inputs and sends data to the display unit every 200 ms.

interrupt is enabled. The foreground code then starts comparing the buffer output pointer to the buffer input pointer. As long as the two are the same, there isn't any new information to be sent to the display board. Once a serial receive interrupt has been received and a new character is in the buffer; the input pointer will be greater than the output pointer, so the foreground code picks up the new character, sends it to the display board, and increments its pointer. This process is repeated until the "end of picture" character is sensed. We then jump back to the loop to request another picture.

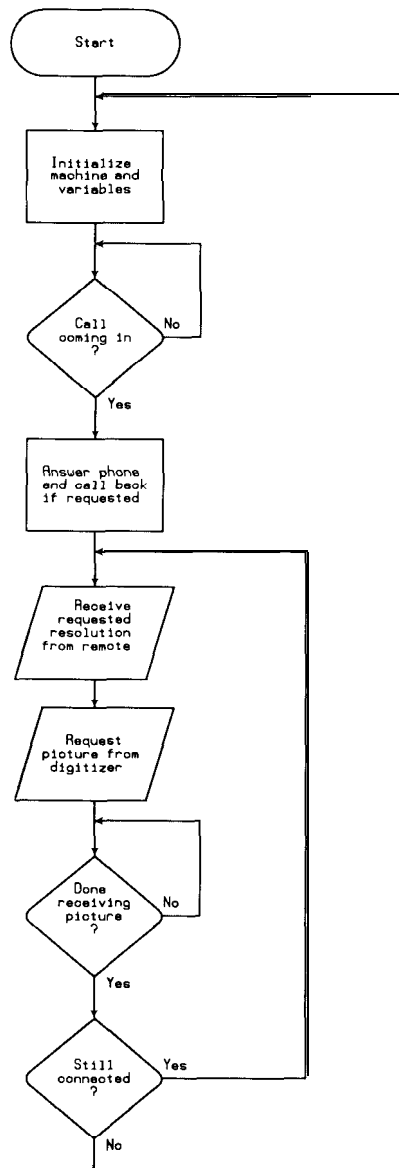


Figure 3d -- The foreground code for the digitizer/transmitter does little more than send a picture request to the digitizer and then wait for the transmission to end.

Just like the code on the digitizer end, the interrupt service routine for the serial input (Figure 4b) checks for the header byte of an information packet. Upon sensing one, it strips the information from the incoming data stream before it can make it to the picture buffer and stores the information for use in updating the

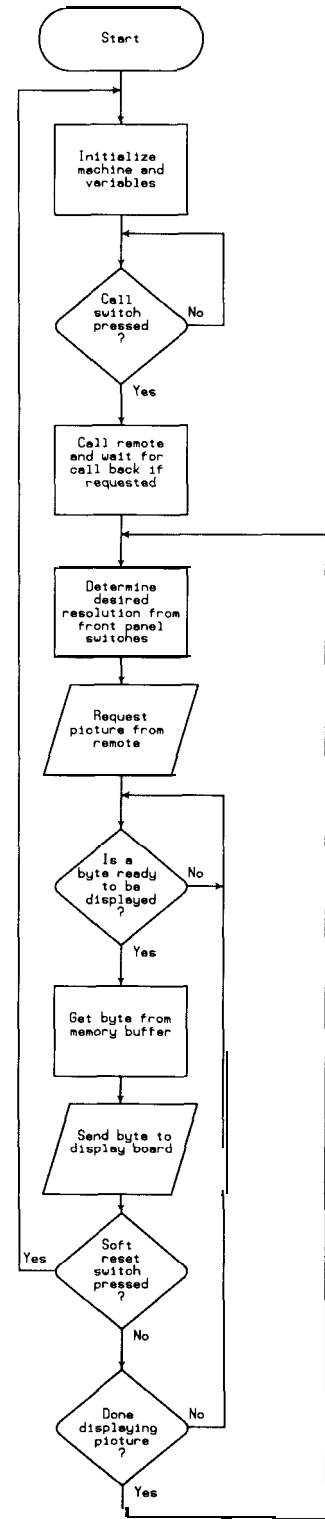


Figure 4a -- The foreground code for the receiver/display is much busier than that for the digitizer/transmitter since it has to empty and keep track of the image input buffer.

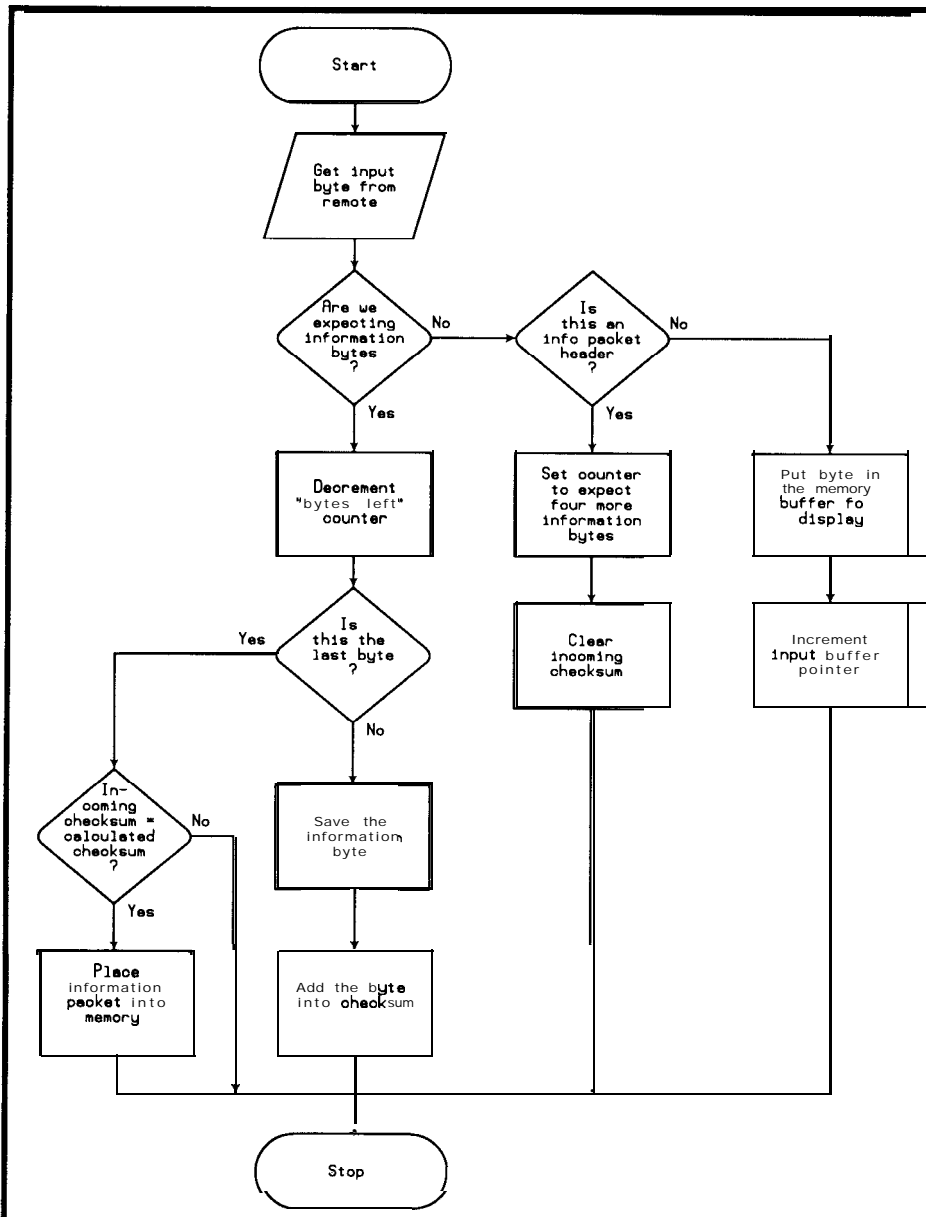


Figure 4b -- The serial input routine for the receiver is similar to the transmitter routine shown in Figure 3a.

front-panel LEDs.

Also like the code on the digitizer end is the heartbeat timer interrupt used to send out information packets (Figure 4c). Since we don't have to worry about adding the information packet to any other outgoing data, the timer interrupt service routine can send the information to the serial port itself without having to flag another routine to do the operation for it.

Multiple Display Resolutions

ROVER's display has the added burden of determining what resolution to request and how to display the resulting picture.

Beginning with version 1.2 of the ImageWise receiver/display selects which resolution combination to use. A look-up table is used by the support code to determine how to handle each switch setting. We also incorporated an auto-

low-resolution picture immediately followed by a high-resolution picture without the need to change any switches on the display board. We can use this feature to fool the display board into displaying more than one picture on the screen.

If we tell the display board that a high-resolution picture is coming, but instead send it a low-resolution picture, the picture will be displayed in the upper left corner of the screen, taking up just one sixteenth of the screen. If we tell the display board that a medium-resolution picture is coming, then send it a low-resolution picture, the picture will be displayed in the upper left quarter of the screen.

Such a technique yields several useful combinations of resolutions. We can, for instance, request ten consecutive low-resolution pictures and display them on a medium-resolution screen. Each picture takes about five seconds to send and is displayed in the upper left quarter of the screen. After ten pictures, we can request a high-resolution picture and display it on a high-resolution screen. This takes about 50 seconds and fills the whole screen.

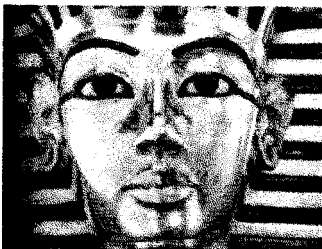
When we start requesting low-resolution pictures again, three-fourths of the high-resolution picture remains on the screen until we request another high-resolution picture, and low-resolution pictures are constantly updated in one corner of the screen. Using this sequence, it's possible to monitor an area with relatively fast low-resolution pictures, then get an occasional high-resolution version of the scene.

There is a front-panel rotary switch on the ROVER receiver that selects which resolution combination to use. A look-up table is used by the support code to determine how to handle each switch setting.

We also incorporated an auto-

IMAGE PROCESSING FOR THE IMAGEWISE VIDEO DIGITIZER

Introducing ZIP, software for ImageWise control, image processing, and outstanding display of video images on EGA/VGA



KINGTUT displayed at 640 x 460 on EEGA

Superior EGA/VGA displays

- 3 levels of zoom
- Color/gray level displays
- 64 level ordered dithers
- Minimum error techniques
- Halftones and duotones

Process single + multiple images

- Math and logic functions
- Matrix convolution
- Histo equalization/linearization
- Square aspect ratio
- Pixellation, and more

Supports ImageWise digitizer

- Transmitter and receiver
- Use 1 or 2 serial ports
- Process 3 images at a time
- Combine images

Saves images for desktop publishing Saves in PCX and MAC file formats

ZIP price: \$79 plus \$2 s/h
Missouri residents add 5.8%. check/VISA/MC

HOGWARE COMPANY
470 BELLEVIEW
ST LOUIS MO 63119
(314) 962 7833

• * call for information **

EGA/VGA SCREEN DISPLAY OF IMAGEWISE VIDEO IMAGES

* requires 384K RAM, MSDOS 2 or higher. Display requires EGA, EEGA, or VGA
ZIP trademark of HOGWARE. IMAGEWISE trademark of Circuit Cellar Inc.

matic mode and a manual mode. In the automatic mode, pictures are requested continuously and displayed in the requested resolution combination. In manual mode, once a picture has finished being scanned onto the display, no new pictures are requested until the "Sample" switch is pressed. At that time, a picture at the set resolution is requested and displayed.

The Telephone Connection

One thing not discussed yet is how the phone connection is established. Ken said that this task was almost as cumbersome as everything else in the project put together. I felt bad but explained that "it was the price of success."

The front-panel switches include a center-off momentary-contact toggle switch labeled "Call A" and "Call B." When the switch is thrown to "Call A," the display unit tells the modem to pick up the phone, dial a number, and establish a connection with the remote digitizer unit. While this sounds easy on the surface, what happens when the remote doesn't answer, is busy, or there isn't any dial tone? In order to do the implementation correctly, all kinds of error control needs to be incorporated. In the present software, any of the above errors result in a system reset.

In order to accommodate more than one phone number, the camera movement switches are read at the same time as the "Call" switch is thrown. Using the Pan, Tilt, and Zoom switches, a total of seven phone numbers can be stored in memory.

The "Call B" switch invokes a slightly different sequence of events. In actual use, I make a connection from my office to the house and leave it in place for hours on end. We eventually realized that the phone line that I was using was billed on a per-minute basis, making 8-hour phone calls very expensive (someone

down in finance gave me a call!). The phones at my house, on the other hand, are billed flat rate no matter how long a call lasts. When the "Call B" switch is thrown, the display unit places a call to the digitizer unit, sends it a call-back phone number, and hangs up. The digitizer immediately calls the receiver back at the number it was given. This way a short, 10-second

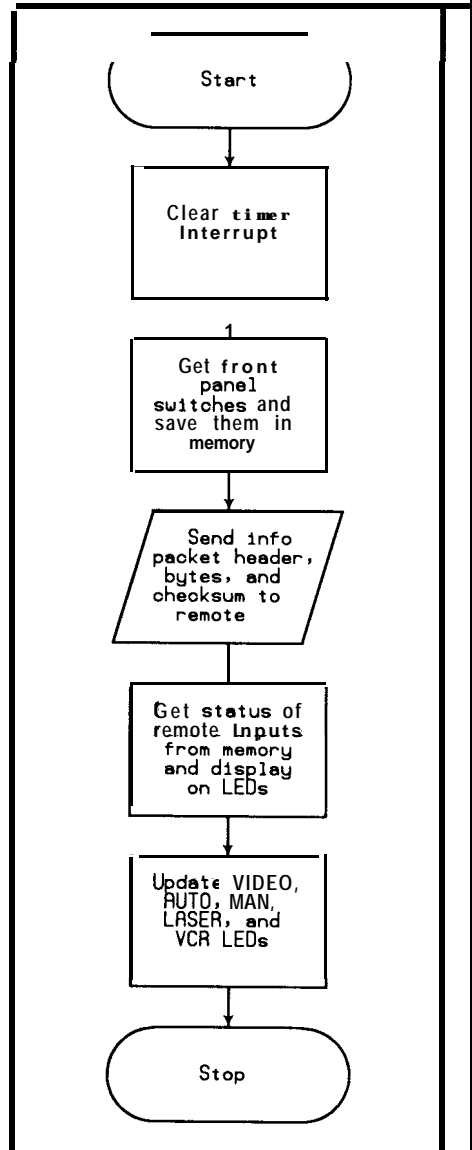


Figure 4c -- The timer interrupt for the receiver is even simpler than the routine shown in Figure 3c, since it doesn't have to add any additional information packets to the data stream.

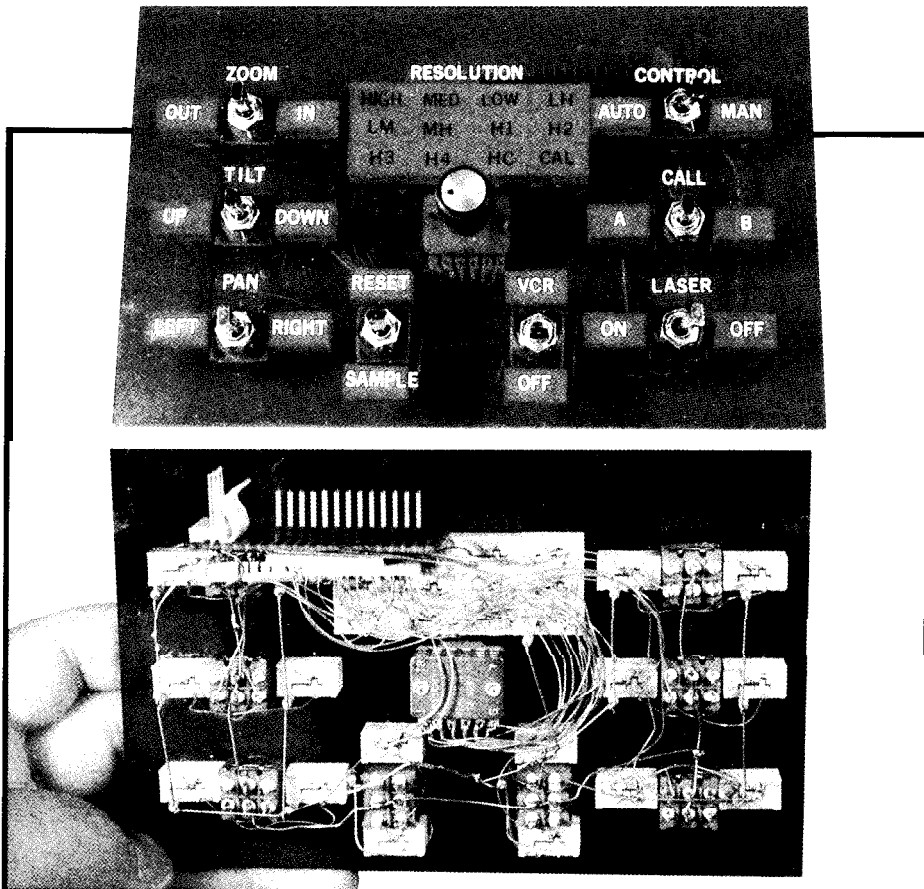


Photo 2 -- The control panel of the receiver/display writ sends camera positioning digitizing commands to the digitizer/transmitter, and has status lights that are updated by the transmitter.

phone call can initiate the 8-hour connection and the connection will be billed at a flat rate.

An Open Door

That just about covers ROVER. You really have to see him in action to appreciate him. The reaction of most people when shown ROVER is "Neat!" There just isn't anything on the commercial market that does quite what ROVER does.

ROVER also opens the doors to some new project ideas. We still have that laser mounted on top of the camera, controlled by a switch on the display unit. Perhaps a laser-guided tracking and targeting system will be the next installment of the ROVER Series. ■

The complete source code for ROVER is available on disk (see page 35) or can be downloaded from the Circuit Cellar BBS. For a reprint of the Image-Wise Serial Video Digitizer articles presented by Steve Ciarcia in May-June '87 BYTE, order DT/DR Reprint. Send \$3 for postage and handling.

Add \$10 for the two ImageWise assembly and user's manuals.

For a reprint of the BCC180 Multi-tasking Computer/Controller articles presented by Steve Ciarcia in Jan.-Mar. '88 BYTE, order BCC180 Reprint. Send \$4 for postage and handling.

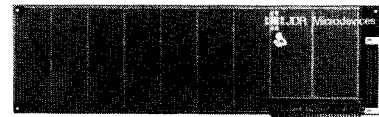
Add \$6 for the BCC180 assembly and user's manual.

Send all manual and reprint requests (mail order only, please) to:

Circuit Cellar INK
Reprints
P.O. Box 772
Vernon, CT 06066

JDR Microdevices

- 30 day money back guarantee
- 1 year warranty on all products
- Toll-free technical support



Wirewrap prototype cards

All are FR4 epoxy glass laminate with gold-plated edgecard fingers, plated holes and silkscreened legends. +5V and ground planes. Mounting brackets and complete documentation included.

S-bit card **for XT** **\$27.95**

All 62 bus pins are labelled on both sides of the card to simplify wire wrapping and soldering.
JDR-PR1

8-bit w/decode for XT **\$29.95**

Decode support for a wide range of applications! Use the decode and buffering circuitry as designed, or modify it to suit your application. Test points and components are labeled on both sides.
JDR-PR2

16-bit card for A J **\$34.95**

Programmable Logic Devices speed up card design process. Pre-wired buffers, address decoding.
JDR-PR10
JDR-PR10-PK Parts Kit (Req. for decoding) \$12.95

16-bit for PS/2 **\$49.95**

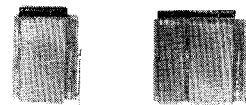
I/O decoding for Microchannel! With the optional parts kit, this card has the decode circuitry to design Microchannel interface cards. Logic supports software programmability, card enable/disable & more software programmable I/O address programmable I/O channel ready delay
JDR-PR16

JDR-PR16-PK Parts Kit (decode circuitry) \$15.95
JDR-16V 16-bit Prototype card w/Video \$39.95
(Extended edge connectors for video applications)

32-bit card for PS/2 **\$69.95**

Microchannel compatible! Full-length card routes power and ground signals throughout the board for quiet operation—ideal for use in a high speed bus. With power & bypass capacitor installation sites.
JDR-PR32

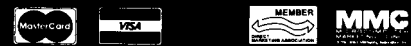
Order toll free 800-538-5000



JDR'S extender cards

For prototype debugging, testing & trouble-shooting. FR4 epoxy glass laminate PC board. Ground plane for quiet operation. Gold plated edge fingers, labelled trace lines. Solder-masked to prevent shorting.

- EXT-8088 For XT System \$29.95
- EXT-80286 For AT System \$39.95
- EXT-32 32-bit card for Microchannel \$99.95
- EXT-16 16-bit card for Microchannel \$69.95



110 KNOWLES DRIVE, LOS GATOS, CA 95030
LOCAL (408) 866-6200 FAX (408) 378-8927 TELEX 171-110
RETAIL STORE: 1256 S. BASCOM AVE., SAN JOSE, CA
HOURS: MON.-FRI. 9-7. SAT. 9-5. SUN. 12-4 (408) 947-8881

Minimum order \$10. Add \$2.50 shipping/handling for UPS ground \$3.50 UPS air. Orders over \$10 and foreign orders may require additional shipping charges. Please contact our Sales Dept. CA residents include applicable sales tax. Prices subject to change without notice. We are not responsible for typographical errors. We reserve the right to limit quantities and to substitute manufacturer. All merchandise subject to prior sales. A full copy of our terms is available upon request. Items pictured may only be representative.

Ctrl

VISIBLE INK *Answers; Clear and Simple*

Letters to the INK Research Staff

Of Disk Drives and Sine Waves

I have a ZAP computer that I built using Steve Ciarcia's book "Building Your Own Z80 Computer." I also have a Radio Shack Model III computer with one disk drive. I was planning to buy a Shugart disk drive, model 410. From what I have gathered so far, it is a single-sided/single-density drive. Is this drive compatible with the Model III? If it is, how should I strap it to identify it as drive 1? I already have a working drive as drive 0.

I need a circuit to produce a sine wave output. I will need a frequency bandwidth of 10 Hz to 50 Hz with an adjustable RMS value from 20 mV to 800 mV. The distortion should be below 0.1%. I know that this is asking a lot from a circuit, but I'm confident in Circuit Cellar INK's ability to give me an answer. By the way, the signal from this circuit will be connected to geophones used in the geophysical oil industry.

Thanks in advance for any help you can give me. Keep up the good work on the articles and projects. They are the one thing that keeps me in the learning mode while having fun.

Sam Maldonado, Jr.
Missouri City, TX

Inquiries at Radio Shack indicate that the Shugart 410 should be compatible with your system.

Drive selection is usually accomplished using jumpers or shorting blocks (available from sources such as JDR Microdevices) on the drive's analog board. The "drive select" jumpers will usually be labeled DSO, DS1, DS2, and DS3. First, be sure that your current drive is jumpered for DSO; it may be jumpered so that it's selected by any active DS line. Then jumper the new drive for DS1.

Other jumpers, with labels like MM, MS, HM, and HS should initially be set the same as similar jumpers on your current drive, but you may have to change them for reliable operation. Getting drives jumpered properly is something of a black art. Patience and systematic experimentation usually yield success.

While there are a few monolithic waveform generator ICs available (the EXAR 2206C is one example), I

know of none with the distortion spec as low as you require. The 2206C offers a typical maximum distortion of 0.5% if carefully adjusted, however, and it might meet your needs if used with carefully selected precision components. Jameco Electronics (1355 Shoreway Rd., Belmont, CA 94002) carries an inexpensive function generator kit utilizing the 2206C as its central component. This kit would enable you to easily conduct experiments to determine the IC's suitability for your application.

-- **INK Research Staff**

Reconfiguring CONFIG.SYS

Please give me an education on CONFIG.SYS file device drivers. I am experiencing some confusion about device drivers, specifically how they are loaded and removed from memory.

Recently this has become an increasingly frustrating problem. I am using an 80386 IBM clone for desktop publishing. This system has a 19" NEC Monograph display, an HP LaserJet Series II printer, and an HP ScanJet scanner. The software we are trying to use is Xerox's Ventura Publisher and Z-Soft's Publisher's Paintbrush.

This is my situation: The HP scanner requires a device driver called SJDRIVER.SYS to be loaded with a DEVICE= command in CONFIG.SYS. I understand this driver is sitting in memory all the time, even when I am using programs that do not address the scanner. This was an offensive situation, but not catastrophic until I installed Publisher's Paintbrush. It requires two drivers: one for the HP scanner (HPSCAN.SYS) and one for the extended memory manager (EXMM.SYS).

Now I have three device drivers automatically loaded into memory when I boot up my computer. Guess what? Ventura Publisher no longer has enough memory to operate!

Why do these drivers have to sit in memory all the time? Isn't it possible to load and unload them with a batch file when I am using a program that needs them? I asked HP and Z-soft technical support this question -- they said no. The only solution they could propose is to have a separate boot disk with a CONFIG.SYS file

that contains all these drivers. I would have to boot the computer from different disks for Publisher's Paintbrush and Ventura. This is terribly inconvenient.

Any alternatives you could provide me would be greatly appreciated. Thank you.

Herald S. Herrington

Jerome, AZ

Device drivers of the type loaded by CONFIG.SYS are a convenient way to fit the software needed to drive special-purpose hardware into the minimum amount of memory. Once loaded, however, they are there to stay. The reason they can't be changed while the computer is up is because under normal conditions they can only be loaded by CONFIG.SYS, which can only be run at boot time. If you could unload one after DOS is loaded you wouldn't free the memory, anyway, because it would not be contiguous with the normal DOS program memory. It would just leave a hole in memory.

There are at least a couple of ways to make the reboot process less aggravating than two or more boot disks. One method I have used in a similar situation is to make two AUTOEXEC files, one named AUTOEXEC.BAT and one named AUTOEXEC.ALT, and similarly, two CONFIG files named with SYS and ALT extensions. The file extensions are switched by another batch file named RECONFIG.BAT. If the computer does not have the correct set of drivers installed you just enter the RECONFIG command and the filename extensions are swapped and the machine reboots. A variation on this can be used to reconfigure the machine for any number of different applications. We'll show you how to do this below.

Probably the most convenient method for your purposes is to incorporate the reconfiguring process into the batch files that load Ventura Publisher and Publisher's Paintbrush. If we assume that you want some way to return to DOS, you will need four CONFIG files and four AUTOEXEC files.

The first pair of AUTOEXEC and CONFIG files will be named AUTOEXEC.BAT and CONFIG.SYS. These will be set up to boot the system as you normally would for general use with all the PATH, PROMPT, etc. you now have. The computer will boot and stop at the DOS prompt.

The second pair will be named AUTOEXEC.DOS and CONFIG.DOS, and will be identical to the first pair. The reason for this should become obvious in just a couple of paragraphs.

Now, to set up for Ventura Publisher, find the VP.BAT file that loads Ventura Publisher. This file is made by Ventura Publisher during installation, and has commands that are needed. Rename this to VENPUB.BAT. Next, make three batch files as follows:

1. An AUTOEXEC replacement named AUTOEXEC.VP, containing at the least:

```
REM AUTOEXEC.VP
```

```
PATH=C:\;C:\VENTURA;...
```

```
CALL VENPUB
```

```
REM The following commands are executed
```

```
REM on exit from Ventura Publisher
```

```
COPY AUTOEXEC.DOS AUTOEXEC.BAT
```

```
COPY CONFIG.DOS CONFIG.SYS
```

```
REM end
```

2. A CONFIG replacement named CONFIG.VP containing at least:


```
FILES=20
BUFFERS=20 [or whatever number you need]
DEVICE= [whatever drivers you need for Ventura J
```
3. A new version of VP.BAT as follows:


```
COPY C:\AUTOEXEC.VP C:\AUTOEXEC.BAT
COPY C:\CONFIG.VP C:\CONFIG.SYS
WARMBOOT
```

This completes the setup needed to automatically configure the system to run Ventura Publisher. Now, let's walk through a bootup and loading of VP.

When the computer first boots up it calls CONFIG.SYS and AUTOEXEC.BAT. These set up the computer in whatever default configuration you define. Now when you enter the usual VP command to load Ventura Publisher, DOS begins execution of the new VP.BAT. The first thing this does is change AUTOEXEC.BAT and CONFIG.SYS to the ones containing the drivers needed for that program. The two COPY commands take care of this. Next, WARMBOOT is executed, which reboots the computer running the new CONFIG and AUTOEXEC files, and calls the old Ventura loader VENPUB.BAT (remember that we renamed this from VP.BAT). This loads Ventura in the normal way, but when you exit from Ventura the next two copy commands will be executed and the default DOS configuration will be restored. Another WARMBOOT at the end would go ahead and reboot the computer in the original configuration.

You will, of course, need a similar setup for Publisher's Paintbrush and any other programs that need special drivers.

Now for WARMBOOT. This is a little program that you make with DEBUG. If DEBUG didn't come with your DOS, try to find DOS version 2.1, and use the DEBUG with that. To create WARMBOOT:

1. Load DEBUG. At the DOS prompt enter the command DEBUG WARMBOOT.COM. You will see the message "file not found" followed by a "-", which is the DEBUG prompt.
2. Enter the following data exactly as shown here. We show what you will be entering in lower case, and DEBUG's response in CAPS. <enter> means that you should press the Enter key.

```
-a100
XXXX:0100 mov ax,1234<enter>
XXXX:0103 mov bx, 40<enter>
XXXX:0106 mov es,bx<enter>
XXXX:0108 es:<enter>
XXXX:0109 mov [72],ax<enter>
XXXX:010C jmp ffff:0000<enter>
XXXX:0111 ret<enter>
XXXX:0112 <enter>
-rcx<enter>
cx 0000
:112<enter>
-w<enter>
```

WRITING 0112 BYTES

Now, to check your program, enter the following "u" command and **DEBUG** will list the program. If you don't get the correct response, go back to the "a100" command and try again. You should see:

```
-u ZOO 112<enter>
XXXX:0100 MOV AX,1234
XXXX:0103 MOV BX,0040
XXXX:0106 MOV ES,BX
XXXX:0108 ES:
XXXX:0109 MOV [0072],AX
```

```
XXXX:010C JMP FFFF:0000
XXXX:0111 RET
```

If all is well, you can enter "q" at the **DEBUG** prompt to exit **DEBUG**. Now, typing **WARMBOOT** should reboot exactly as if you had hit the <ctrl><alt> key combination.

-- **INK Research Staff**

In Visible Ink, the Circuit Cellar Research Staff answers microcomputing questions from the readership. The representative questions are published each month as space permits. Send your inquiries to:

INK Research Staff
c/o Circuit Cellar INK
Box 772
Vernon, CT 06066

All letters and photos become the property of CCINK and cannot be returned.

68000 POWER

HARDWARE

Consider the **PT68K-2** for general computing, control, education, experimenters. 512 or **1024K** no-wait-state RAM, 4 serial and 2 parallel ports, clock/calendar, battery-backed static RAM, floppy disk controller, debugger and Basic in ROM. Advanced 68000 system fits into an XT/clone cabinet, its six XT-compatible slots accept color or monochrome video boards, hard disk controller, keyboard. Expandable in stages, starting with the Basic Kit (8 MHz, w/o DRAM, FDC, DOS) at **\$200**; 10 MHz Kit (with **512K**, FDC, ports, SK*DOS) \$575; Assembled board (12.5 MHz, 1meg, SK*DOS) \$899. OS-9, other systems available.

PERIPHERAL TECHNOLOGY

1480 Terrell Mill Rd. #870C
Marietta GA 30067
(404) 984-0742

SOFTWARE

Consider **SK*DOS** for general computing, control, education, experimenters. Full DOS documentation plus on-line help, multiple directories, user-installable device drivers, RAM disk, disk cache, floppy or hard drives to 64 megabytes each, I/O redirection, time/date stamping, batch files, read and write MS-DOS disk files. Includes powerful utilities such as copy-by-date, undelete, show differences between files, prompted delete, text file browse, editor, assembler, Basic, more. Included with **PT68K-2** computer (left), **\$140** for other systems. Compilers, editors, cross- & disassemblers, communication programs, other software available.

STAR-K SOFTWARE SYSTEMS CORP.

P. O. Box 209C
Mt. Kisco NY 10549
(914) 241-0287 / Fax (914) 2418607

MICROMINT'S Gold Standard in Single Board Computers and Industrial Controllers

BCC180 — \$395.00
Multitasking Controller



The BCC180, only 4.5" x 8.5" uses the same 64180 CMOS Z80 instruction compatible processor as Micromint's SB180 and SB180FX single board computers. Configured primarily for process control w/ 384K of memory, 6 parallel I/O ports, console serial port, RS-232/422/485 selectable auxiliary serial port, and an interrupt driven ROM-resident multitasking BASIC-180 compiler, the BCC180 uses the same 44-pin I/O expansion bus as Micromint's BCC52 controller board.

- CMOS HD64180, 9.216MHz 8-bit CPU, 68-pin PLCC package
- Up to 384K bytes total memory on-board
- 128K of either static RAM (62256) or EPROM (27256)
- 256K dynamic RAM SIMM
- Full-function 8K ROM monitor included
- Console RS-232 serial port with auto baud rate select to 19,200 baud
- Peripheral serial port, 1920-19,200 baud, selectable RS-232, RS-422, or RS-485
- 48 bits bidirectional parallel I/O
- 64K I/O available through the BCC bus edge connector

- Dual 22-pin (0.156") edge connector
- Compatible with all Micromint BCC-series I/O expansion boards
- Two 26-pin headers for six bidirectional parallel ports
- 25-pin DB-25 for RS-232 console I/O
- 20-pin header for RS-232 auxiliary serial port
- 4 screw terminals for RS-422/485 connection

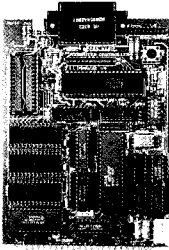
BCC180-1 20 9MHz assembled and fully socketed BCC180 Computer/Controller with 32K bytes of static RAM, ROM Monitor, BASIC-180 development software and user's manual \$395.00

For Additional 256K DRAM add \$100.00

BCC180-1 100 Quantity w/ 32K RAM w/o ROM Monitor \$209.00

BCC52 — \$189.00
BASIC-52 Computer/Controller

The BCC52 Computer/Controller is Micromint's hottest selling standalone 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 Micromint's new 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" 256/128 EPROM programmer, 3 parallel ports, a serial terminal port with auto baud rate selection, a serial printer port, and it is bus compatible with the full line of BCC-bus expansion boards. The BCC52 bridges the gap between expensive programmable controllers and hard-to-justify price sensitive control applications. BASIC-52's full floating point BASIC is fast and efficient enough for the most complicated tasks, while its cost effective design allows it to be considered for many new areas of implementation. It can be used both for development and end-use applications.

Since the BASIC-52 is bus oriented, it supports the following Micromint expansion boards in any of Micromint's card cages with optional power supplies:

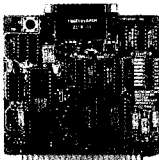
- BCC22 Smart terminal board
- BCC25 LCD display board
- BCC33 3 port I/O expansion board
- BCC40D 8-Channel optoisolated I/O expansion board
- BCC40R 8-Channel relay output board
- BCC53 Memory and 6 port I/O expansion board
- BCC113 8-bit and BCC30 12-bit A/D converter boards
- BCC118 Dual channel serial I/O board

BCC52 BASIC 52 Controller Board \$189.00
BCC-SYST.5 "52 PAK" Starter System includes: BCC52,ROM A&B UTIL.,CC01,MB08,UPS10 \$449.00

BCC52 OEM 100 Quantity Price -- \$149.00

BCC52C Lower power all CMOS version of the BCC52 \$199.00
NOTE: The BCC52 series is available in Industrial Temperature Range, fully tested at temperature. Prices start as low as \$294.00 in single quantities. Be sure to call for a quote on your specific Industrial OEM requirements.

BCC11 — \$139.00
Z8 BASIC Computer



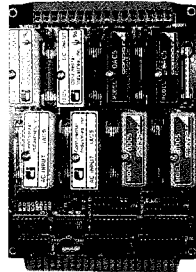
- Uses Z8 single chip microcomputer
- On-board tiny BASIC interpreter
- 2 on-board parallel ports & serial port
- 6 interrupts (4 external)
- Just connect a terminal and write control programs in BASIC
- 6K bytes of RAM or EPROM memory on-board
- Baud rates 110-9600 bps
- Data and address bus available for 56K memory and I/O expansion
- Consumes only 1.5 watts at +5, +12, and -12V

BCC11 BASIC System Controller \$139.00

OEM 100 Quantity Price \$89.00

• Now Available in Industrial Temperature Range

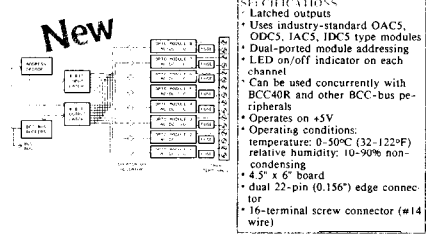
BCC40D — \$139.00
8-Channel Optoisolated I/O Expansion Board



The Micromint BCC40D is an 8-channel optoisolated input/output expansion board designed for use with Micromint's family of BCC-bus Computer/Controllers. Using industry standard optoisolated I/O modules, the BCC40D provides on/off control and input monitoring of eight 115-230VAC or 5-48VDC devices used in data acquisition and control applications.

Up to 16 BCC40D boards can be used together in a single system to provide a total of 128 input and output channels. Individual channels can be read or updated by reading from or writing to a single I/O address. The BCC40D can be directly controlled from BASIC or it can function completely in the background under an application program using high-speed interrupt-driven ROM C firmware. This firmware sets aside a table in memory which reflects the status, setpoints, and change-of-state flags of the I/O modules. Interaction among programs within a multiboard BCC40x system merely consists of reading or setting these memory table values.

Each optoisolated channel is fused and has screw contacts for direct connection to the controlling device and/or the power source. Both input or output modules, and AC or DC functions can be intermixed on the same BCC40D board.



BCC40D-1 Without modules \$139.00
BCC40D-4 With 4 modules \$189.00
BCC40D-8 With 8 modules \$229.00

- NEW**
- Latched outputs
 - Uses industry-standard OACS, ODCS, IACS, IDC3 type modules
 - Dual-ported module addressing
 - LED on/off indicator on each channel
 - Can be used concurrently with BCC40R and other BCC-bus peripherals
 - Operates on +5V
 - Operating conditions: temperature: 0-50°C (32-122°F) relative humidity: 10-90% non-condensing
 - 4.5" x 6" board
 - dual 22-pin (0.156") edge connector
 - 16-terminal screw connector (#14 wire)



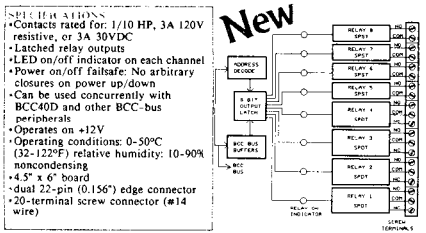
BCC40R — \$169.00
8-Channel Relay Output Board

The Micromint BCC40R is an 8-channel relay output expansion board designed for use with Micromint's family of BCC-bus Computer/Controllers. Using efficient mechanical relays, the BCC40R provides contact-closure on/off control of eight AC- or DC-powered devices for data acquisition and control applications.

Up to 16 BCC40R boards can be used together in a single system to provide a total of 128 relay output channels. The relays are controlled by writing to a board-specific I/O address. The relays on a BCC40R board can be controlled either as a set of eight relays at a single I/O address or individual relays at eight separate I/O addresses.

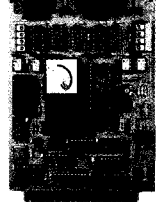
The BCC40R can be directly controlled from BASIC or it can function completely in the background under an application program using high speed interrupt-driven ROM C firmware. This firmware sets aside a table in memory which reflects the status and setpoints of the relays. Interaction among programs within a multiboard BCC40x system merely consists of reading or setting these memory table values.

The eight relay outputs have screw contacts for direct connection to the controlling device and/or the power source. Four of the relays have single-pole-double-throw (SPDT) output connections and four relays have single-pole-single-throw (SPST) output connections.



BCC40R 8-Channel relay output board \$169.00
OEM 100 Quantity price \$124.00

BCC18 — \$175.00
Dual-Serial Board



The BCC18 is a general-purpose dual-serial-port interface board for use with the Micromint BCC-bus. Optional support software is available for the BCC52 and BCC180 computer/controllers.

The BCC18 Serial Board contains two serial interfaces. Each interface can be either a 110/300/1200-bps modem, or a hard-wired RS-232C/RS-422/RS-485 interface. The modem interface uses a Xecom XE1201/XE1203 MOSART (MODEM Synchronous/Asynchronous Receiver/Transmitter), capable of 110, 300, or 1200 bps communication and compatible with Bell 103 and Bell 212A standards. The hard-wired serial interface uses an industry-standard 8251A USART (Universal Synchronous/Asynchronous Receiver/Transmitter), capable of supporting asynchronous serial communications at speeds up to 19.2 kbps and synchronous serial communications at speeds up to 64k bps.

The BCC18 can be configured with two MOSARTs, two 8251As, or one of each. Up to 16 BCC18s can be used in a single system (for a total of 32 serial ports).

- MOSART**
Connects directly to any phone line
DTMF or pulse dialing
DTMF reception and decoding
Call progress monitoring
Priority generation/checking
- 8251A**
Full-duplex, double-buffered transmitter and receiver
Fully programmable with several speed and character modes
Error detection for parity, overrun, and framing
False start bit detection, automatic break detect and handling
Supports binary
Bit rate is software programmable using an on-board 8251 counter/timer
RS-232C, RS-422, and RS-485 supported
- SOFTWARE**
MOSART and 8251A use the same program interface, so most software will work with both
Optional support software for the BCC52 and BCC180 is available that makes it easy to access and use most of the features of the MOSART and 8251A from within user-written programs

BCC18S OEM configuration fixed dual 8251 RS-232 only \$175.00 serial port board

BCC18S OEM 100 Quantity Price \$134.00

BCC18U-1	8251/8251 Dual RS-232/485 serial	\$209.00
BCC18U-2	8251/1201 Modem and serial port	\$359.00
BCC18U-3	1201/1201 Modem /Modem board	\$499.00
BCC52/18	BCC52 serial port utilities software	\$75.00

BCC-BUS
Expansion Products

	Single Qty. Price
BCC08	Single Channel UART Serial Board \$149.00
BCC13	8-bit, 8 Channel A/D Board \$129.00
BCC40	12-bit, 16 Channel A/D Board \$197.00
BCC22	25-line, 80 Character Terminal Board \$249.00
BCC25	LCD Board (4x20) or (8x40) \$159.00
BCC55	Fully Decoded & Buffered Prototyping Board \$ 79.00
BCCXX	X-10 Power Line Controller \$ 59.00
BCC53	Multifunction Expansion Board \$160.00
MB04	4 Slot Mini-Motherboard \$ 69.00
MB08	8 Slot Full Motherboard \$ 85.00

REPRESENTATIVES
ACCESS TECHNOLOGIES
1408 Richmond Dr
Placentia, Ca. 92670 Tel:(714) 996-3917

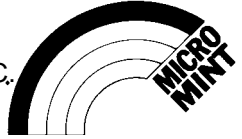
MICROFUTURE
4104 Comac Terrace
Fremont, Ca. 94538 Tel:(415) 657-0264

DISTRIBUTOR
J.B. DESIGNS & TECHNOLOGIES LTD.
15 Market Place
Cirencester, Glos. GL7 2PB
England
Tel: 0285-68122 Fax: 0285-68859

To order call
1-800-635-3355

TEL: (203)871-6170
TELEX: 643331
FAX : (203) 872-2204

MICROMINT INC.
4 PARK STREET
VERNON, CT. 06066



The Home Satellite Weather Center

by Mark Voorhees

Part 6

Adding Serial and Parallel Ports to the Peripheral Controller

We've covered most of the hardware for our 68000-based Peripheral Processor in the last few installments. This time, we'll look at the serial and parallel ports needed for communication with our instrument units, and we'll discuss the operating firmware of the Peripheral Processor. I'll also give you information about the RF gear we'll be using for reception of the WEFAX satellite images.

First, though, I'd like to take a short survey. As I said in the first issue of Circuit Cellar INK, I feel that it's important for you to provide input as to what you'd like to see the system accomplish. I've received a few letters requesting some additional features, and those are either under investigation or already being integrated into the system (and will be subjects of future columns). This time, though, I need a bit of specific information. The question of the day, therefore, is:

What other instrument packages need to be supported by the Peripheral Processor?

As I stated early in the series, the Heath ID-4001 and ID-5001 Weather Computers will be the initial units supported. Some of you, however, may have found units that serve your instrumentation needs better and want to have those units supported. I would ask that you drop me a short note, including manufacturer, model,

and, if possible, protocol information (from your user's manual) so that I can investigate writing support software for your unit. I'll provide a summary of the survey, along with descriptions of the submitted products, in a future issue.

You can drop me a line either on the Circuit Cellar BBS, the Drig BBS, CompuServe (70566,777), or by mail:

Mark Voorhees
P.O. Box 27476
Phoenix, AZ 85061-7476

The Instrumentation Data Ports

The last board of the basic version of the Peripheral Processor provides the serial and parallel ports used to communicate with the instrumentation package. The serial port is RS-232 compatible; the parallel port is, however, special, since it is actually four 8-bit TTL-compatible parallel ports in a single connector. This is necessary to allow support of the somewhat unusual Heath ID-4001, using an interface circuit which I'll show in a future article. This interface circuit ties to their "computer interface," which we'll discuss at length later in this issue.

The circuit for this board is shown in Figure 1, and is pretty straightforward. The parallel ports are dual MC6821 Peripheral Interface Adapters, which will be used as inputs in the standard configuration, to monitor information from an ID-4001 and/or other accessory instru-

ments. Serial communications to a "smart instrument," such as an ID-5001 Weather Computer, are handled by an MC6850 Asynchronous Communications Interface Adapter, which has on-board



hardware data-rate selection.

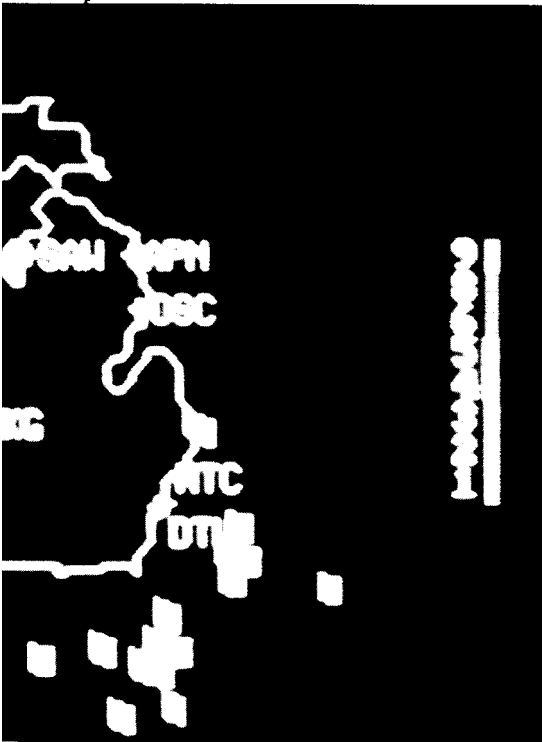
Peripheral Processor Firmware

Let's first outline what we expect of the Peripheral Processor in the way of performance, then discuss the approach taken to perform each function.

The initial responsibilities of the unit are:

1. Perform diagnostic checks of memory at power-up.
2. Check for valid configuration information (held in battery-backed SRAM).
3. Display abbreviated status report on front panel.
4. Monitor front-panel switches.
5. Monitor host port for commands.
6. Process commands from host.
7. Transfer data to host.
8. Request and receive data from instruments at configuration-selected intervals.
9. Handle timing of WEFAX sampling and processing of WEFAX images.
10. Transfer incoming WEFAX data to memory.
11. Control internal WEFAX receiver.

Other duties may be added as we develop accessories to enhance



the unit, so we will perform these duties as quickly as possible to allow for the additional processing time we'll require for future expansion.

Our basic approach will be to use hardware interrupt functions and support devices to minimize continuous demands on the processor. We've already given the proc-

essor back some time by using a DRAM controller to handle refresh timing and synchronization of data to the DRAM. You'll remember that we are also using a multifunction peripheral IC on the main board, and it has two as-yet unused timers available.

[Editor's Note: See page 35 for information on ordering back issues of Circuit Cellar INK.]

As for interrupts, we will set the priorities in relation to the availability of the data in real time. I think you'll easily understand the reasons behind this structure:

Default: Front-panel processing. This is the routine which is least important in real time and the one that will be executed when no interrupts are being received. We will also monitor the external status byte in this routine, looking for an AC power loss.

Level 1: Host communication. This is also a real-time, independent process since the host communication can be delayed while other interrupts are serviced.

Level 2: Reserved for future needs.

Level 3: Reserved for future needs.

Level 4: WEFAX receiver control. Under normal circumstances, this will be a set-and-forget type of process, but it needs to be allowed a reasonable priority in case it requires attention.

Level 5: Reserved for future needs.

Level 6: Instrument communications. We want our data samples to be timed as precisely as possible, but a short delay (on the order of microseconds) won't appreciably affect the data received, even in a worst-case condition. This function is somewhat interactive with the instrumentation equipment (the processor either waits for a condition to be met, or "asks" the instrument for information and

waits for a reply), so the process can be interrupted for a level 7 process without harm.

Level 7: WEFAX processes. We obviously cannot control the timing of data being received from the satellite; to attempt to do so would probably cause a picture to be mistimed and garbled, so these are our highest-priority processes.

Now, let's look at the individual processes in detail.

At Power-Up

We need to define the **power-up** condition. For our purposes, power up will be that time when AC power is first applied, except when the unit has been performing under full-power battery back-up. This condition would indicate that no valid data exists in DRAM, and will allow us to test all DRAM for memory errors.

We have provided a sense input in hardware to allow the processor to know when AC power is lost. We will set up an SRAM memory location to allow the processor to save this status bit (and the other bits of that byte, saved for future use) when AC power fails.

At power-on reset or a **front-panel** manual reset, the processor will look at the status byte in memory and abort the memory test if the status indicates that AC power failed, but battery took over (if not, the processor wouldn't have been able to save the status byte). The processor will then store the current status byte and continue to the next routine.

If the status byte is clear when checked, the processor will assume that current DRAM is invalid, and perform a memory test.

The memory test will consist of three passes:

1. Zeros will be written to all addresses in DRAM. The processor will read every location and define the top of DRAM as the first

location that fails. This location will be displayed as "TOP xxxxxH" on the front panel if all other tests succeed.

2. A "checkerboard" (10101010101010) pattern will be written to all locations and verified.

3. A descending pattern will be written to all locations in sequence and verified.

If the tests succeed, the processor will display "MEMORY OK" and "TOP xxxxxH" on the front panel for a preset time, then continue to the next routine.

Configuration Check

The processor now checks two locations in memory which will contain a preset value if a configuration block exists.

If the values at these locations are incorrect, then the processor will assume that no configuration exists and will display "NO CONFIG" on the front panel. It will then initialize the host port and wait for the configuration to be sent by the host. Upon reception of a configuration block and the setting of values in the check locations, the processor will

continue this routine.

If the tested values are correct, the configuration will be assumed to be valid. Accordingly, the processor will initialize the instrument ports, the WEFAX receiver, and the WEFAX processor before completing initialization, enabling interrupts, and beginning normal processing. The front panel will display "xxx% MEM FREE," indicating the amount of assigned instrument data space available, and "x WEFAX AVAIL," indicating the number of WEFAX images available in memory for download.

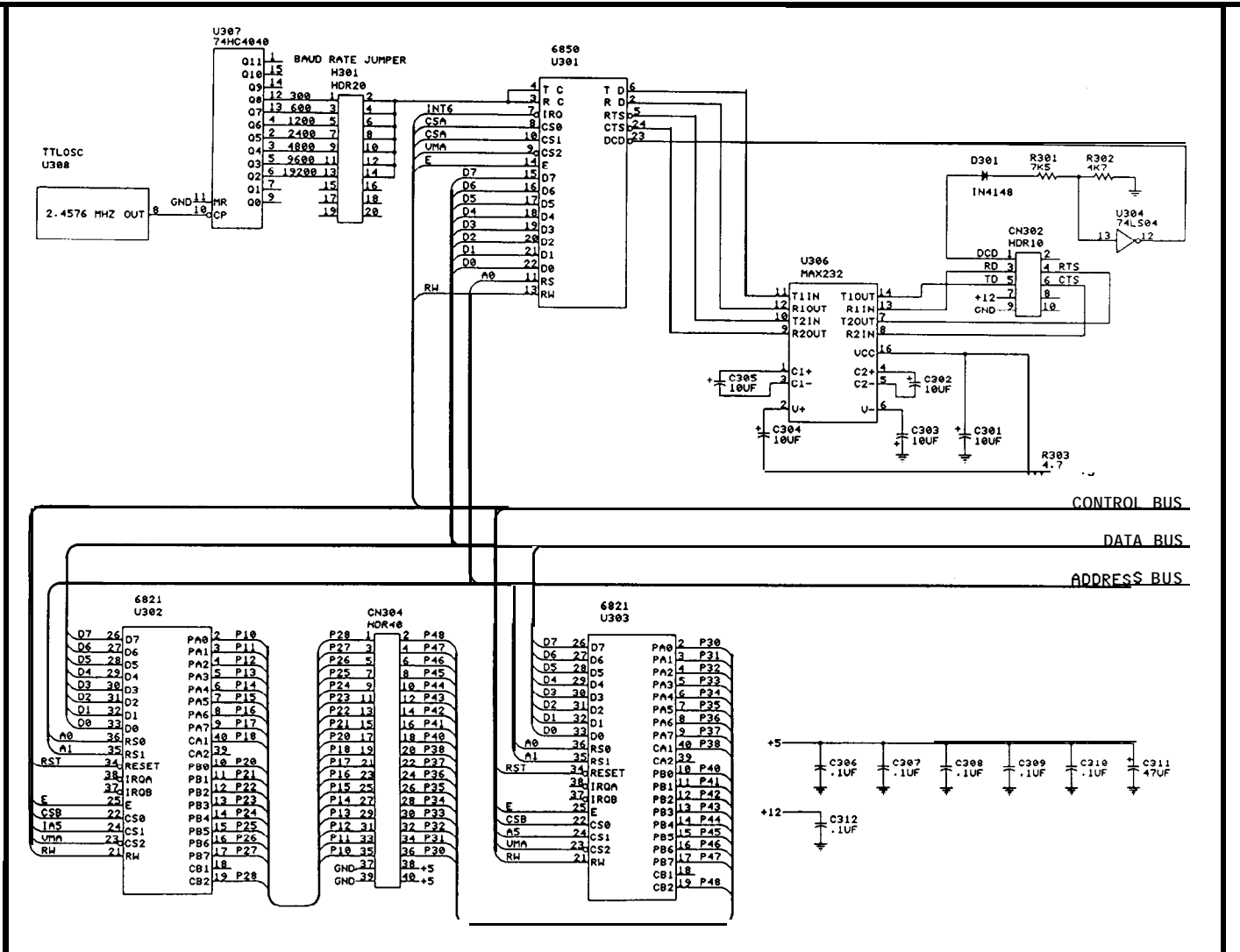


Figure 1 -- The serial/parallel board of the Peripheral Processor features a single RS-232 port and four 8-bit TTL parallel ports in a single connector.

Initialization Routines

Depending on the configuration requirements, the processor will now initialize the host port (if not already initialized), the instrument port(s), the WEFAX processor, and the WEFAX receiver (and any enhancements we might add later).

Interrupts will be enabled as the normal processing begins.

Front-Panel Routines

As I mentioned earlier, this series of routines operates when interrupts are not being processed. These routines sense the front-panel switches and write information to the front-panel display.

Initially, we'll only use the first two sensed switches on the front panel. The fifth switch is a direct system reset, so we have two sensed switches available for future use.

Switch 1 will cause the front panel to display two additional pieces of information: "DATA xx:xx," indicating the time that the last instrument sample was taken, and

reception time of the latest WEFAX image.

The processor will always process incoming WEFAX, to the maximum available memory limits. Thus, if four images can be stored in memory at a time, they will be saved in sequence so that the latest four images are available. Switch 2 will allow for the WEFAX processing to be suspended; for instance, if you know that a particular image is sent at a given time and you want to make sure that it's available for download (but don't have time to download it right away), pressing

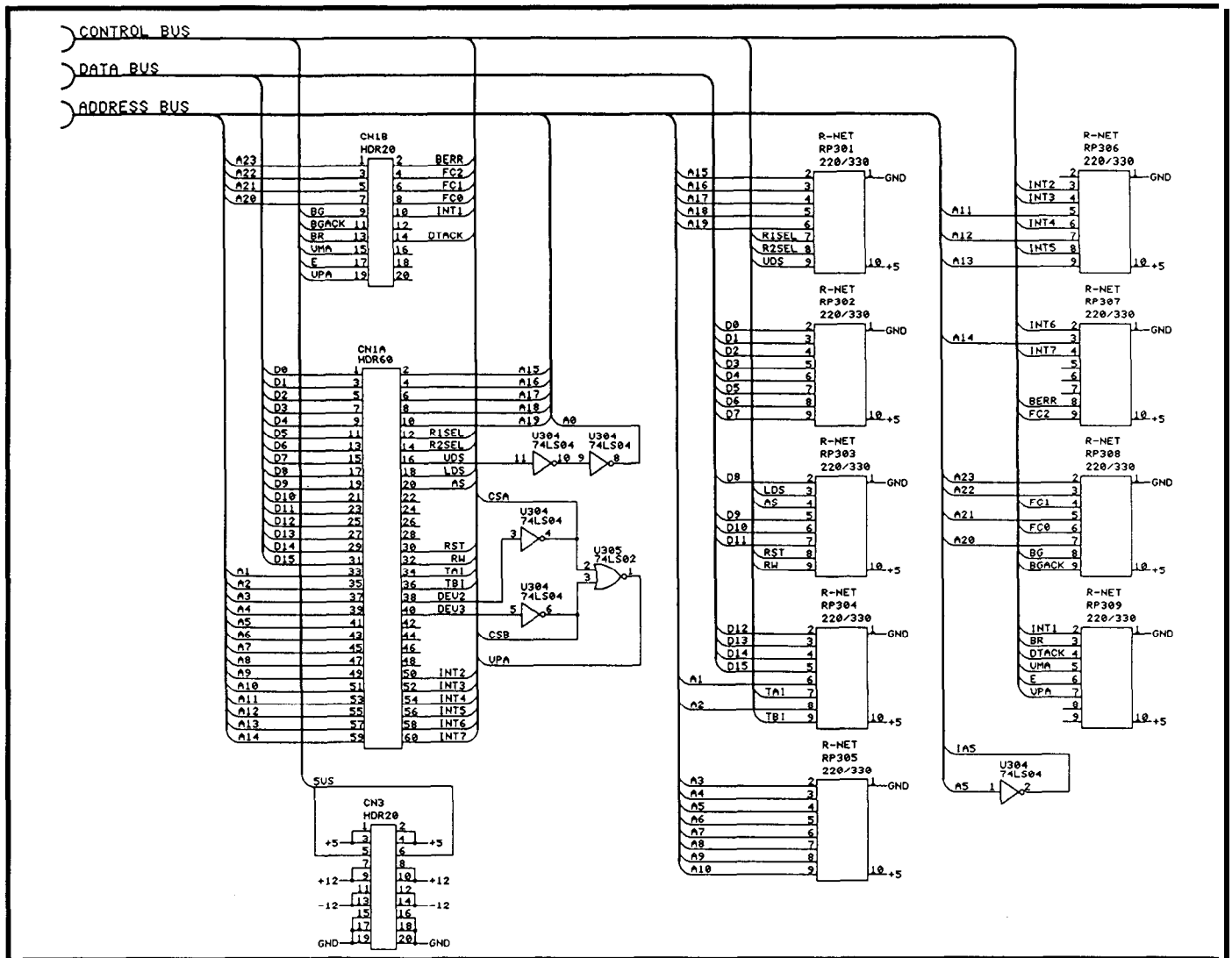


Figure 1 (continued)

switch 2 will “freeze” the **WEFAX** function until a download is completed or the switch is pressed again. When this option is used, the normal front-panel display will show “**WEFAX HOLD**” to indicate the suspension of operation.

Host Communications

The processor will accept and act on several command strings from the host computer. They are:

#LC<Configuration Block><CR>

Load the configuration block data to local memory to define operation. The configuration block consists of instrument type, frequency of instrument data sampling, **WEFAX** receiver information, and frequency of **WEFAX** sampling (defining the image’s width and height parameters). This block of data will be redefined as necessary for any future enhancements.

#CL<CR> --

Clear DRAM memory. This is a forced clear of memory after a download or as required.

#DI<CR> --

Download instrument data from memory. This will cause all sampled instrument data to be downloaded to the host. The host will hold the data until it is processed and integrated into the instrument database.

#DW<CR> --

Download **WEFAX** images. This will cause the **WEFAX** data to be downloaded to the host for processing into display images, which can then be saved as screen pictures.

#TP<CR> --

Force performance of power-up memory test, regardless of memory validity. Used for diagnostic purposes.

#IS<CR> --

Invalid data sent by processor. The processor will resend the last data string.

Command strings will also be fed from the processor to the host to define the data being transmitted:

#CC<CR> --

Confirm command reception. Used to confirm the reception of a

command string from the host, this string is sent after command validation.

#CI<CR> --

Sent to host to identify reception of an invalid command string. The host will resend the command.

#ID<Character Count> <Instrument Data> <CR> <CR> --

Informs host that instrument data follows; also gives count of characters in data string. The host will expect to receive <Character Count> characters followed by the two <CR>s.

#WD<Character Count><WEFAX Image Data><CR><CR> --

Informs host that **WEFAX** data follows. The host will expect to receive the defined quantity of characters followed by the two <CR>s.

#CC<CR> --

Confirms that the clearing of memory has been completed.

Instrument Processes

We have to address the needs of both of our initial instrument packages in the instrument processes.

The ID-5001 Weather Computer uses a command/response structure to transfer the information for our data sample, so we have to trigger a series of “questions” at sample time. The commands are:

ATLC<CR> --

Linefeed clear. Causes only the <CR> character to be sent following data.

ATRB<CR> --

Downloads barometer reading.

ATRHO<CR> --

Downloads outdoor humidity reading.

0001	--	Temperature 10s digit
0010	--	Barometric pressure 10s digit
0011	--	Barometric pressure 1/10ths digit
0100	--	Wind speed 10s digit
0101	--	Time 10s of hours digit
0110	--	Time 10s of minutes digit
0111	--	Time 10s of seconds digit
1000	--	Temperature sign and overflow digit
1001	--	Temperature 1s digit
1010	--	Barometric pressure 1s digit
1011	--	Barometric pressure 1/10ths digit
1100	--	Wind speed 1s digit
1101	--	Time 1 s of hours digit
1110	--	Time 1s of minutes digit
1111	--	time 1s of seconds digit

Table 1 -- Data on the first four pins of the ID-4001 computer port, shown here, acts as a switch to determine the type of data transmitted on pins 5-12 of the port.

ATRR<CR> --

Downloads current rainfall measurement.

ATRT<CR> --

Downloads current time.

ATRTO<CR> --

Downloads current outdoor temperature.

ATRWA<CR> --

Downloads average wind speed and direction.

ATRWG<CR> --

Downloads peak wind gust speed and direction.

ATVT:<CR> --

Sets time separator as “:”.

ATXCA<CR> --

Defeats the “auto transmit” function which sends updated information at the time of each change. We don't need this function since we will sample at the rate specified by the configuration sent to the Peripheral Processor.

Following the gathering of the data, the processor will place it into the proper format and save it in memory.

The ID-4001 Weather Computer is quite different from the ID-5001. The ID-5001 processes raw data into a precise format before transmitting it to the host computer. The ID-4001 shifts the data formatting task to the host.

The ID-4001 “computer port” outputs information as if it is driving a remote set of seven-segment displays. Nineteen of the 25 pins of the interface connector are used: The first four select which of the sixteen display digits' information is present on pins 5-12. The format of data on the first four pins is shown in Table 1.

We will monitor pins 18 and 19 for status of the data (these pins are

front-panel switch bus and rear-panel switch bus, respectively). The data strobe is on pin 20 and will be used to strobe data into our ports.

The wind direction is handled separately as four bits on pins 13-16 which signify sixteen positions on the compass rose (22.5-degree increments).

The processor will handle these inputs as latched data from our inter-

that we investigate the WEFAX concept more closely, I'll hold the discussion of these functions until next time. I want to use the remainder of our time to introduce the RF hardware which we'll use as the “front end” of our WEFAX system.

The WEFAX accessories themselves will be discussed after we cover the subjects of the host soft-

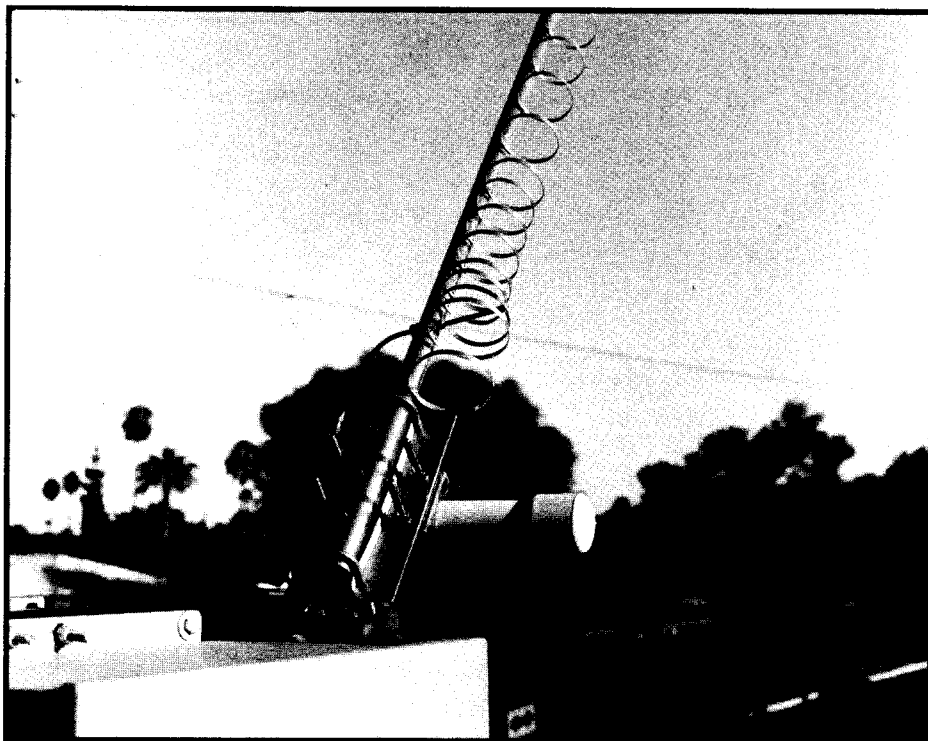


Figure 2 - Author's 1691-MHz Loop Yagi antenna from Spectrum International.

face circuit and make it available at sample time. Thus, it will monitor the digit select lines, convert the seven-segment data to ASCII, and save the digit in its proper order. This operation will repeat until the entire sample is completed, at which time any remaining formatting is done and the sample stored in memory.

Obviously, any additional instruments supported will require additional routines, but the same basic form will be followed.

Since the WEFAX routines are really a separate subject and require

ware and interfacing for the Heath Relative Humidity instrument and Heath Rain Gauge (both of which can be used as accessory instruments to the ID-4001 Weather Computer but do not have “computer interfaces”) in the next two installments. Since the RF units can take 30 to 60 days to arrive after you order them, I wanted to give you something of a head start.

The construction of high-gain, low-noise RF hardware is a very precise science, and I have gained great respect for the technology through my professional use of sat-

elite LNAs, downconverters, and related products. I believe that the design and construction of hardware such as this, with the intent that a reader/experimenter will construct a working model meeting the high requirements of satellite reception, is beyond the scope of our goals. I'm certain that total success in this project series is as important to you as it is to me.

for a nominal +20 dBi gain at the 1691 -MHz WEFAX frequency. The unit comes unassembled (you'll need a "pop-rivet" tool) and without mount. Figure 2 shows my installation of this antenna; when we discuss the WEFAX system in detail, I'll include drawings of the mount and housing design in the figure. If you find the need for greater gain, multiple Yagi antennas can be combined

with an accessory 2- or 4-beam harness.

The Spectrum MMG- 1691 preamp is next. Both the preamp and the downconverter are enclosed in the weatherproof housing shown in Figure 3a, with placement shown in Figure 3b. The MMG-1691 offers a typical power gain of 15 dB with a noise figure of 1.2 dB. This unit is designed with GASFETs to minimize noise, and packaged in a die-cast case.

The signal is converted to the 137.5-MHz IF frequency by Spectrum's model MMK- 1691/ 137.5 downconverter, which also provides a nominal 25-dB gain with a 3.5-dB noise figure. The output of this will go to the Peripheral Processor's integral receiver or other suitable receiver, and then to the WEFAX processor, both of which we will discuss in future issues.

You will want to contact Spectrum International as to current pricing and delivery time since both of the electronic modules are made in England. Disregarding what I thought was a very long delivery date when I ordered (Spec-

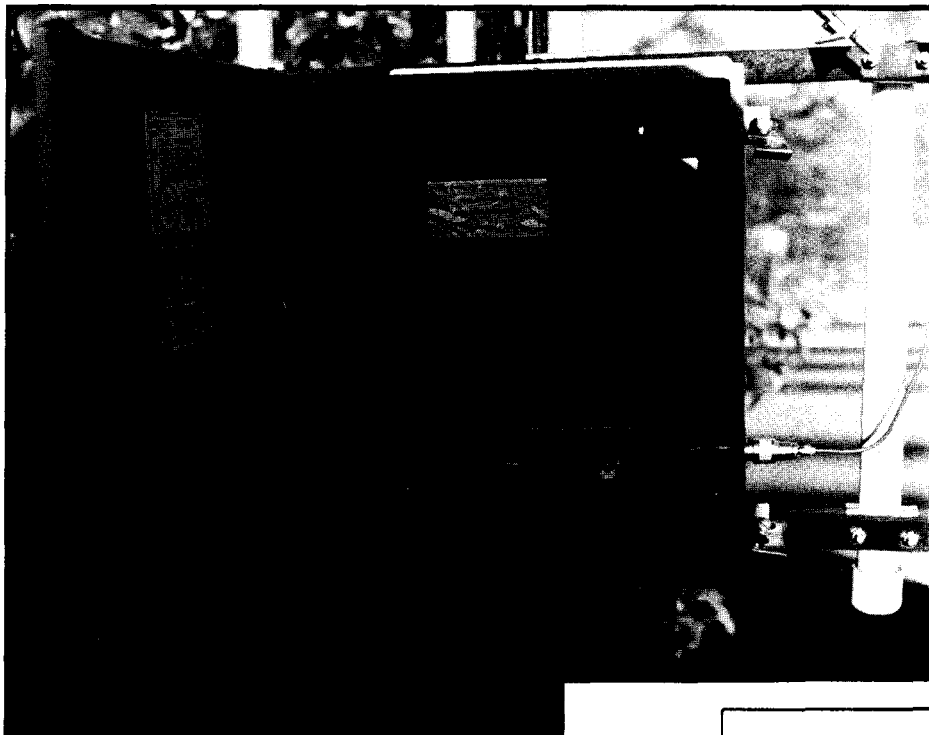


Figure 3a -- Inside look at author's weatherproof housing for WEFAX RF preamp and downconverter.

Hence, there are three devices which I will recommend that you purchase: the WEFAX antenna, a low-noise preamplifier, and a WEFAX downconverter.

After looking at several manufacturers' products, I determined that the units sold by Spectrum International Inc. (P.O. Box 1084, Concord, MA 01742, (617) 263-2145) would easily fit our requirements.

The antenna is a Yagi design, model 1691-LY, which is designed

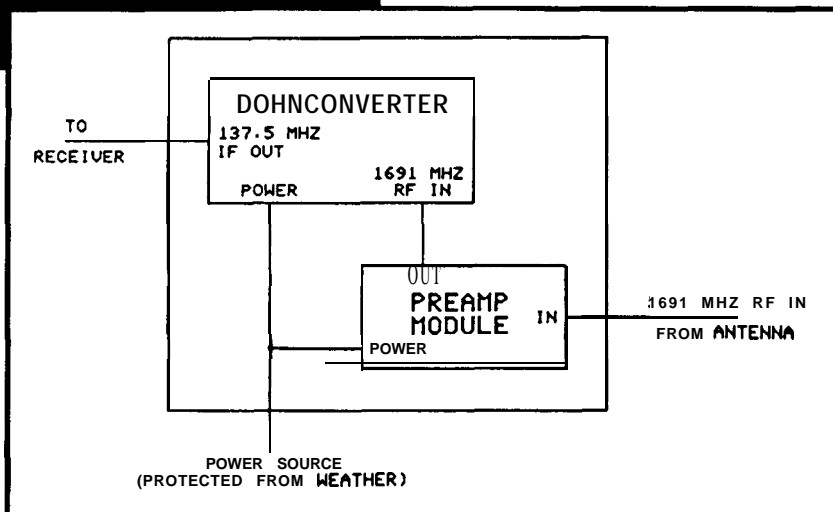



Figure 3b -- Layout of mast-mounted RF hardware for WEFAX. Details of the installation and setup will be presented in a future issue.

trum was out of stock on the downconverter), I have been very impressed with the quality of the products and the helpful attitude of Spectrum's personnel.

We've covered a lot of ground this time around, and there's more to come in our next installment: we'll finish our discussion of the Peripheral Processor firmware, talk about initial hookup and testing, and cover interfacing to the Heath ID-4001. Until then, remember to drop me a note if you have input as to additional instrument packages which the Peripheral Processor might support. 

I am making kits available for each portion of the Home Weather Center system. Each kit, unless otherwise noted, consists of a PC board and all devices and parts (except SRAM or DRAM devices) to construct the standard design. Pricing for the kits are as follows (all include shipping charges):

- 88000 Main Processor Board. \$319.00
- Front-Panel Board. \$88.00
- 1 Meg x 16 Memory Board. \$189.00
- Power Supply (quantities limited). ... \$40.00
- Cabinet (quantities limited) \$44.00

I will be making individual components available for those with "well-stocked" parts cabinets. Send me a stamped, self-addressed business-size envelope for a complete listing with prices.

Mail orders to:

Mark Voorhees
P.O. Box 27476
Phoenix, AZ 85061-7476

Include check or money order if ordering kits. I regret that, at this time, I am unable to accept credit cards.

Allow 4 to 6 weeks for shipment since most of this material will be shipped UPS.

Software for this project is available on the Software Disk for this issue and on the Circuit Cellar BBS. For ordering and downloading information, see page 36.

BECOME A CIRCUIT CELLAR PROJECT BUILDER!

Circuit Cellar Inc. kits are a proven vehicle for accomplishing a very special goal. With well designed circuits, pretested key components, documentation, and a knowledgeable support team you can have the thrill of making something you built yourself actually work! This is a CCI project! Call (203) 875-2751 to order your kit or for information.

IMAGEWISE - Serial Digital Imaging System

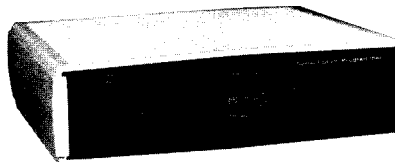


unretouched photos

The Circuit Cellar ImageWise Serial Digital Imaging System was designed to function intelligently as a stand-alone digitizer or as an integral component of a complete tele-imaging system.

- DT01-Full ImageWise Transmitter Full kit.....\$249.00
- DR01-Full ImageWise Receiver Full kit.....\$249.00
- Both Units purchased together..... \$489.00
- DT01-Exp ImageWise Transmitter Exp. kit..... \$99.00
- DR01 Exp ImageWise Receiver Exp. kit.....\$99.00
- Both Units purchased together.....\$179.00
- Case & power supply for either unit..... 549.00

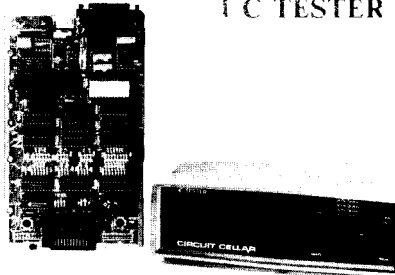
SERIAL EPROM PROGRAMMER



The Serial EPROM Programmer provides a fast and efficient way of programming, verifying and copying a large variety of EPROM types. Supports 27x16 thru 27x512.

- SEP27/2 Serial EPROM Programmer complete kit..... \$199.00
- SEP27/1 Serial EPROM Programmer Exp. kit..... \$89.00
- Power Supply..... \$19.00

IC TESTER



The IC Tester has the ability to identify unmarked ICs as well as designate specific pin failures of hundreds of 74xx00 logic chips.

- ICT01-EXP IC Tester Experimenters kit..... \$99.00
- ICT01-FULL IC Tester complete kit..... \$179.00
- ICT02 Complete kit with enclosure..... \$349.00
- 2x20 LCD for ICT01..... \$32.00

BCC180 - Multi-Tasking Computer



The BCC180 is a 9 MHz single board computer with 384K, 6 parallel ports, and 3 serial ports onboard. Multi-tasking BASIC-180 runs 32 simultaneous tasks.

- BCC180-Kit- 20..... \$295.00

FROM THE BENCH

Conducted by Jeff Bachiochi

Fiber Optics

Your Link to the Future

When you make a long-distance telephone call, chances are your call will leave the copper lines at some point and travel through a fiber-optic data link. As advertised by many long-distance services, copper wire and even microwave equipment is being replaced by fiber-optic equipment. Your conversations will be crisp and clear thanks to this substitution.

Fiber optics is the process of transmitting and receiving infrared or visible light frequencies through a low-loss glass or plastic fiber. A transmitting laser or LED generates a serial bit stream of on and off transitions which are akin to the lows and highs of an RS-232 transmission.

Transmission speed through copper wire (balanced and unbalanced transmissions) is inversely proportional to distance. As seen in Tables 1 and 2, unbalanced (RS-423/RS-232) transmissions of 19200 bps are limited to a maximum of about 100 feet, where balanced (RS-422/RS-485) transmissions of 19200 bps can go to about

4000 feet (300 feet at 1 Mbps).

Data rates greater than 1 Mbps are possible using fiber optics. Transmission distances are limited by the material and preparation of the fiber material being used. Plastic fiber has a practical limit of about 200 feet. Glass fiber, because of lower loss, is capable of thousands of feet.

Higher data rates and distances are not the only advantages to using fiber optics. Electrical isolation between the communicating equipment can prevent the "domino effect" should lightning or some other calamity hit part of the system. The non-current-carrying transmissions of fiber are safe for use in a spark-free environment. Fiber-optic transmissions do not pollute the environment with radiated RFI (no visits from the FCC) and are immune to the RFI and EMI radiated by other equipment.

Radiant Communications Corporation has pre-packaged the electronics necessary to interface RS-232 to fiber-optic cable. Two-way asynchronous commu-

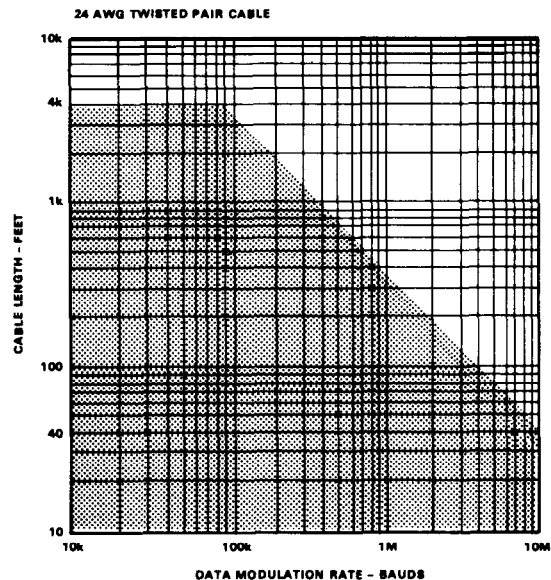
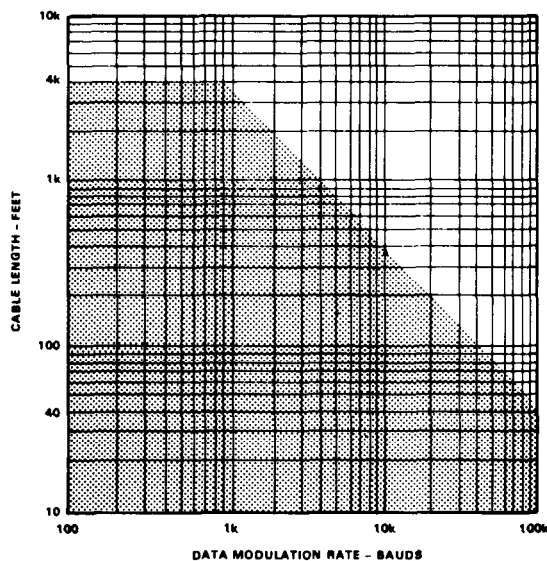


Table 1 and 2 -- These tables clearly show the limiting relationship between transmission speed and maximum cable length in copper wire.

nication at speeds of 0-19200 bps plus six control/handshake lines are all accomplished over a duplex fiber-optic cable. Packaging in a DB-25 housing makes it easy to attach to existing equipment. Cost is \$85.00 for each converter (two are necessary) plus \$549.00 for 500 feet of terminated glass fiber.

If you can't justify \$700 just to experiment a bit, don't despair! T&B (Thomas & Betts Corporation) has a fiber-optic PCB Data Link Kit for \$32.00. The kit contains a PCB-mount transmitter housing, a PCB-mount receiver housing, plus three meters of plastic fiber-optic cable and self-terminating cable plugs. Although two kits are necessary for full-duplex communication, you can communicate in one direction with only one kit.

I've mounted a transmitter/receiver pair on a Radio Shack 44-pin edge-card prototyping board (Figure 1) for connection to the Micromint BCC bus. TTL-level serial in and out are available on the BCC bus, so no level shifters are necessary for microprocessor inter-

face. A drive transistor, Q1, is necessary to provide the 30-mA maximum allowable drive current for the LED transmitter. The receiver has a Schmitt-trigger output and can drive one standard TTL load. Five volts for the transmitter/receiver circuit is supplied by the BCC bus.

A second transmitter/receiver pair is mounted on a 2" by 3" piece of perf board (Figure 1). The TTL signals are level-shifted with a MAX232 (see From the Bench, Sept/Oct CCINK) to provide an interface to an RS-232 communication port with only a +5V power supply. Rather than borrowing 5V from the host computer through an unused pin on the DB-25 connector, an external 8V power supply is used (which was procured inexpensively from a surplus house). Note the 5V regulator on the perf board. Except for the transmitter/receiver housings, everything could fit within a DB-25 housing.

In a harsh industrial environment, most Machine Control Systems are mounted within a Nema-type

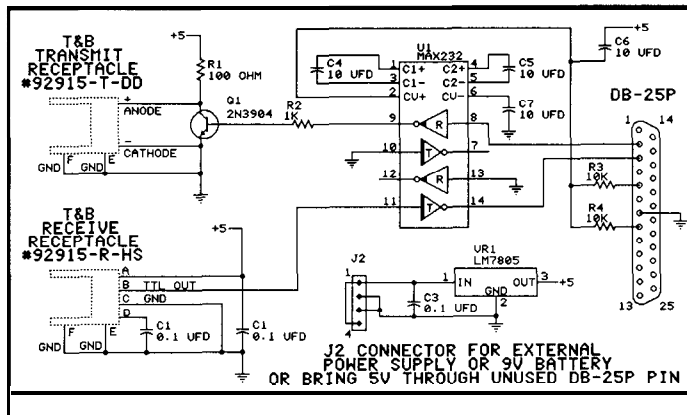
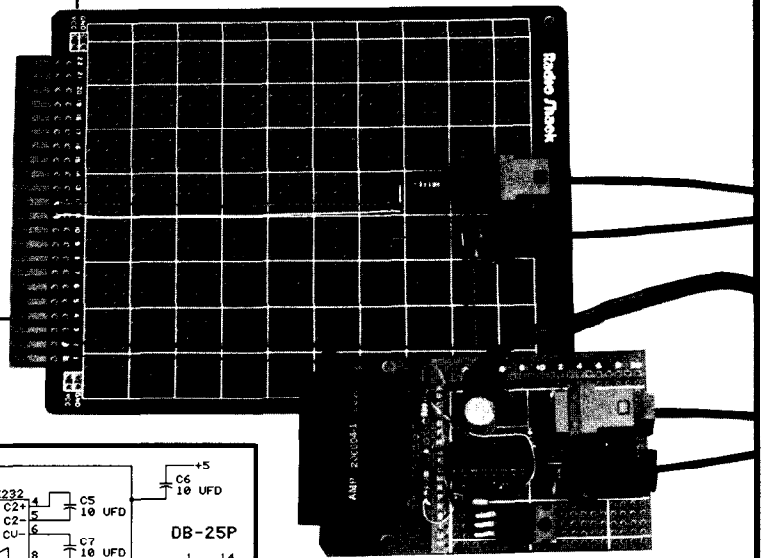
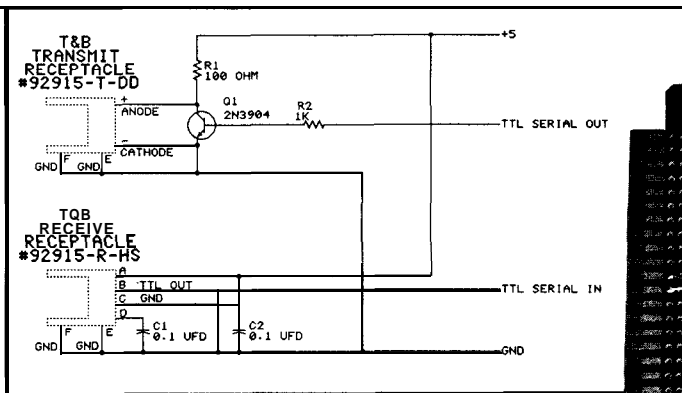
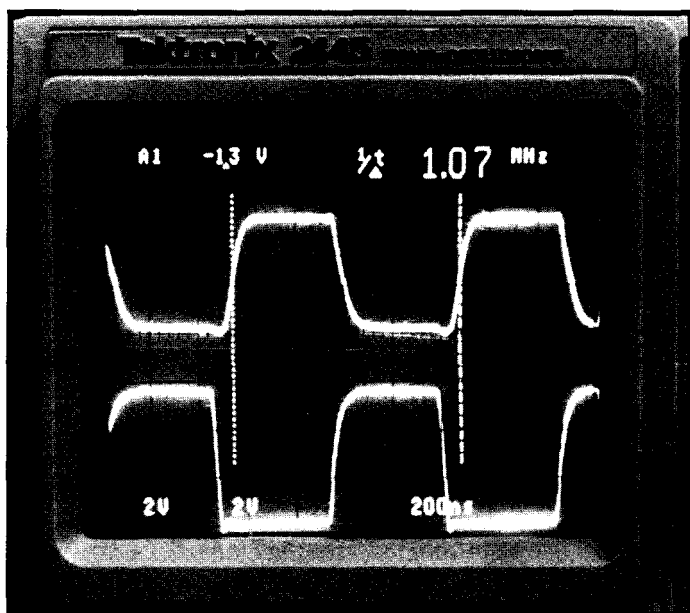


Figure 1 -- One end of the prototyping fiber optic link is mounted on a bus prototyping board. in this case a BCC Bus board (top), while the other end of the link connects to any RS-232 port (bottom).




enclosure on the factory floor. This keeps out dirt, but EMI/RFI radiated from the factory equipment can be induced on the RS-232 ribbon or twisted-pair cable. Fiber-optic cable cures the induced-noise problem, so allows a console I/O device to be located quite a distance from the microcontroller.

A previous application involved interfacing to and monitoring a Numeric Control Machine. The heart of this system was Micromint's BCC52. The link to console I/O, normally an RS-232 ribbon cable, was replaced by fiber optics. The fiber-optic/TTL interface made ± 12 volts unnecessary since the BCC52 can operate on just +5 volts if the RS-232 level shifters are not used. The control program, in this case, was transferred to the BCC52 and run without ever visiting the factory floor (T&B offers a 100-foot roll of plastic cable for \$38). An IBM PC was used as the console I/O device. This allowed storage of control programs on diskette and also provided a good program-editing environment.

Figure 2 shows oscilloscope traces of the fiber-optic transmitter (channel 1) and receiver (channel 2). The transmitter is driven by a signal generator at over 1 MHz (the limit of this generator). Using such a high frequency demonstrates the bandwidth and reveals the potential of a fiber-optic system.

Let's face facts: for small distances, nothing beats the simplicity and affordability of a prefabricated wire cable to interface serial equipment. But as the distances

get longer, the data rates become higher, the environment gets harsher, or the price of copper goes up, the clear choice is fiber optics.

Fiber optics: helping us close the gap between the present and the future. 

Radiant Communications Corporation
P.O. Box 335
470 Ridgedale Ave. E.
Hanover, NJ 07936
(201) 386-1643

Thomas & Betts Corporation
920 Route 202
Raritan, NJ 08869
(201) 685-1600

American National Standards Institute, Inc.
1430 Broadway
New York, NY 10018

Technical Aspects of Data Communication
John E. McNamara
Digital Press ISBN 0-932376-18-5

Understanding Telephone Electronics
John L. Fike & George E. Friend
Texas Instruments Learning Center
Radio Shack 62-1388

Innovations like these help to make today's technology more cost effective, reliable, and easier to use. Please share your favorite ideas, chips, and circuits with others.

We will pay \$25 for any **From the Bench** accepted for publication. All submissions should be typed, double-spaced, and include neatly drawn schematics or Schema configuration, library, and page files.

Include a stamped, self-addressed envelope large enough to hold everything if you wish the materials that have not been accepted to be returned.

Submit to: **From the Bench, c/o Circuit Cellar INK, Box 772, Vernon, CT 06066.**

CONNECTIME *Excerpts from the Circuit Cellar BBS*

THE CIRCUIT CELLAR BBS
 300/1200/2400 bps
 24 hours/7 days a week
 (203) 871-1988 -- 4 incoming lines
 Vernon, Connecticut

Conducted by Ken Davidson

The message base of the Circuit Cellar BBS is now available on disk. See page 35 for details.

We've just finished compiling some statistics for the Circuit Cellar BBS for the month of August 1988, so I'd thought I'd start off this month by relating a few of them. We had a total of 4,089 calls, which averages out to 132 per day. Some days we receive over 150 calls. Of those calls, 58% were at 1200 bps, 36% at 2400 bps, and 6% at 300 bps. Distribution of the calls throughout the day is pretty even, with the lowest number coming in between 4 A.M. and 7 P.M. (Eastern time). The most number of calls come in just after lunchtime (1 A.M. to 2 P.M., in the late afternoon (4 P.M. to 5 P.M., and just before midnight (11 P.M. to 12 mid). It appears that many people tend to take long lunches, get bored just before it's time to go home, or check in just before going to bed.

As for messages, we had a total of 1,116 messages entered during the month of August, which averages to 36 per day. Again, I've seen much higher numbers on individual days (up to 50 in one day).

Even though we emphasize the message areas rather than the file areas, file transfers are still quite heavy. Most of the files downloaded were related to the DDT-51 article, which first appeared in the August issue of BYTE. We had a total of 4698 downloads, which averages 151 per day. Even with a limit of 200K per person per day, people appear to have no problem getting what they want.

I also want to tell you about a service that has been around for a while, but wasn't of any use to CCBBS users until now. It is called PC Pursuit, and is run by Telenet Communications Corp. For a flat rate of \$25 per month, users of personal computers with modems can make unlimited data calls to any of approximately 30 cities around the U.S. during off-peak hours (evenings and weekends). Hartford (about 15 minutes from Vernon and a local call) has just been added to the list of outdial cities for PC Pursuit. Now, with just a call to your nearest Telenet node (which is usually a local call), you can spend as much time on the Circuit Cellar BBS as you want for just \$25 per month (the usual 1-hour-per-call and 2-hour-per-day limits still apply, however).

More information about PC Pursuit is available on the Circuit Cellar BBS, or you can call Telenet's PCP information BBS at (800) 835-3001 or (703) 689-2987

(in Virginia). If the BBS is always busy or you'd rather do things by voice, Telenet's registration and information number is (800) 368-4215 or (703) 689-5700. For you international BBSers, sorry, but PCP is only available to those within the U.S.

Let's start off this month's messages with a discussion of automatic identification badges (combined with ROVER, even more shades of Big Brother).

Msg#: 7554 *GENERAL*
 From: KEN HOWELL
 To: ALL USERS
 Subj: INTERROGATED BADGES

This is a little out of the ordinary for this board, but here goes. I have a cat that takes special delight in using my neighbor's pet door. I'd like to make a "pet badge" that would respond to a radio or other (magnetic?) "interrogation" and respond with a ID signal. I'd think with the recent availability of surface-mount components, one could come up with a badge that is of reasonable size (1/4 inch thick and about 1 inch diameter). I've seen something similar advertised by The Sharper Image, but they want about \$200 for it, and it looks like just a magnet and a hall-effect sensor.

I'm just looking for ideas. Any thoughts?

--Ken

Msg#: 7677 *GENERAL*
 From: BOB PADDOCK
 To: KEN HOWELL
 Subj: REPLY TO MSG # 7554 (INTERROGATED BADGES)

Dallas Semiconductor has announced a product line that does what you want. They market it as a way to transmit info from tractor trailers to a central dispatch as the truck goes through the main gate. They have a transmission range of about 6 feet.

Msg#: 7659 *GENERAL*
 From: KEN HOWELL
 To: BOB PADDOCK
 Subj: REPLY TO MSG # 7577 (INTERROGATED BADGES)

Thanks a lot. Will look into it.

--Ken

Msg#: 7628 *GENERAL*
From: MARK LAMPKIN
To: BOB PADDOCK
Subj: REPLY TO **MSG#** 7577 (INTERROGATED BADGES)

Bob, that is some interesting information on Dallas Semiconductor. We have a 32K x 8 tag that goes to >1000mm, and I'm doing the same thing with a large furniture manufacturer, but I'm transferring the whole bill of lading plus the truck's stats.

...Mark

Msg#: 7666 *GENERAL*
From: MARK LAMPKIN
To: KEN HOWELL
Subj: RF IDENTIFICATION

Ken, it's interesting that you should bring up this subject. My line of work is in that specific area. The way to do it is not with magnetics. A simple AM transmitter, or better yet FM, could do the job for you perfectly. Locate a schematic for a one- or two-transistor FM transmitter and build a small tag for the cat. This transmitter would send out a simple tone. You receive the tone and drive a PLL like the NE567. The best way is for the cat to have a tag that is a receiver and that responds to the transmitter mounted on the door. Lower current draw on the tag supply (i.e., lithium battery) and a more secure system.

...Mark

Msg#: 7572 *GENERAL*
From: KEN HOWELL
To: MARK LAMPKIN
Subj: REPLY TO **MSG#** 7566 (RF IDENTIFICATION)

Mark -- Sounds great. I have a number of the Forrest Mims booklets on simple transistor RF circuits. These do seem a bit cumbersome for use on a cat's name tag, but I will look them over again. Is the circuit you propose one in which the tag is "asleep" until interrogated? This circuit could have all sorts of robotic applications.

--Ken

Msg#: 7622 *GENERAL*
From: MARK LAMPKIN
To: KEN HOWELL
Subj: REPLY TO **MSG#** 7572 (RF IDENTIFICATION)

Ken, the way we do it for industrial use is to have the tag sleep until it is present in the field AND it receives a valid query. This is very easy to accomplish with an all-CMOS circuit. Of course, for the commercial products we use extensive custom integration.

...Mark

Msg#: 7626 *GENERAL*
From: MARK LAMPKIN
To: KEN HOWELL
Subj: REPLY TO **MSG#** 7572 (RF IDENTIFICATION)

Ken, another by-the-by. Don't think in transistors only; CMOS

inverters at a lower frequency can do the same thing. Just use a longer antenna in a flat coil configuration, and, if you can, increase the effective size with a little ferrite.

...Mark

Msg#: 7664 *GENERAL*
From: STEVE CIARCIA
To: MARK LAMPKIN
Subj: REPLY TO **MSG#** 7566 (RF IDENTIFICATION)

Interestingly, I am also working on a tag system, primarily to be used as ID badges and for tracking people within a building. So far we have built two infrared units. One sends a coded pulse every 20 seconds and the other sends a coded pulse whenever it receives an interrogation pulse. In both cases, each room has an infrared receiver on the wall that communicates with a central tracking unit.

--Steve

Msg#: 7623 *GENERAL*
From: MARK LAMPKIN
To: STEVE CIARCIA
Subj: REPLY TO **MSG#** 7604 (RF IDENTIFICATION)

Steve, you might want to check out Texas Instruments' Factory Integration Division. They have an IR communication link for global communications on the factory floor. As far as RF, we have some extremely large installations in Japan for people-type usage. Interestingly enough, we are the #1 system in the Japanese market, however we are designed and built in Scotts Valley, Cal.

...Mark

Microprocessors with on-board BASIC are always popular among designers of dedicated control systems. National Semiconductor and Zilog are manufacturers of two such chips. The next few message threads discuss the fate of the two chips and show that the popularity of the Zilog Z8671 (Z8) has diminished little since Steve first wrote about it in 1981.

Msg#: 7124 *GENERAL*
From: MARK LAMPKIN
To: ALL USERS
Subj: NATIONAL 8073

Has anyone built a system with the National Semiconductor 8073? It is a CPU with internal BASIC in ROM, like the 8052, but that's where the similarity ends. There was a company advertising in both EDN and the old Kilobaud Microcomputing that offered a single-board system. They were testing a new XMOS process from National. The standard 4.0-MHz 8073 was running 20 MHz. Needless to say, a very fast BASIC interpreter. If anyone has any knowledge of this, can you share it?

...Mark

Msg#: 7195 *GENERAL*
From: BOB PADDOCK
To: MARK LAMPKIN
Subj: REPLY TO MSG# 7124 (NATIONAL 8073)

The last I heard National had discontinued the 8073. Best to check with your local National rep before you start designing one into a board.

Msg#: 7256 *GENERAL*
From: MARK LAMPKIN
To: BOB PADDOCK
Subj: REPLY TO MSG# 7195 (NATIONAL 8073)

Bob, I had heard the same thing about two years ago. The news came to me from an engineer at Digi-Key. They market a single-board computer based on the 8073 and at the time were worried. I don't know if they are still selling the board, but there are a number of discount houses still selling the chip. I'm not designing a product with it, but still have eight or nine sitting around.

...Mark

Msg#: 7250 *GENERAL*
From: STEVE CIARCIA
To: BOB PADDOCK
Subj: REPLY TO MSG# 7195 (NATIONAL 8073)

There is an even better story that goes along with the 8073. National says (I got this from some people high up in marketing) that the 8073 was never a successful product because some guy (yours truly :-)) published an article using the **Zilog Z8671 (Z8)** just about the time the 8073 was introduced that caused everyone to ignore the 8073. For six years all they ever did was give away samples. There was never any significant production orders for the 8073.

--Steve

P.S. Interestingly, I contacted National BEFORE the Zilog about using the 8073 in a project. They ignored me (of course, this was back in 1980)! When I contacted Zilog, they welcomed me with open arms (the Z8 marketing manager was a Circuit Cellar reader). The rest is history.

Msg#: 7269 *GENERAL*
From: BOB PADDOCK
To: STEVE CIARCIA
Subj: REPLY TO MSG# 7250 (NATIONAL 8073)

National never seems to do very well with their CPUs (e.g., SC/MP, SC/MP-II(8060), 8073, and probably others). They do seem to do well with the I-bit COP stuff, though. Acolleague of mine here was evaluating CPUs and decided the 32k family was the highest-performance machine going (at the time), but all we got from National were broken promises about availability of their CMOS 32k stuff, so we went with the 63000 (later the 68HCOOO) from Hitachi/Motorola. We'll never even consider another National CPU after our experience, which is too bad because they do have some great parts. It's just too bad you can't get them when you need them.

Msg#: 7135 *GENERAL*
From: SILAS ANTONE
To: ALL USERS
Subj: Z8 AUTO APPLICATION

In 1987, I purchased a **Mazda RX7**, and lately I've been consid-

ering some modifications. By interfacing to the dealer options (power windows, power door locks, power antenna, retractable lights, and sunroof), the possibilities are endless!!!

After endless hours of flipping through Steve's articles in BYTE, I decided on the 28671 (Jul, Aug '81). Cost and the lack of a development system were the main reason for the decision. JDR carries the Z8 but not the matching quasistatic RAM (26132). Also, I don't have a source for documentation. JDR offers a few pages out of the Zilog manual for \$3.50, which doesn't include any timing information! Does Zilog offer a PIA for the Z8 family?

One of my many concerns is electrical noise from the RX7's four ignition coils. I took a look at the car's computer and it is enclosed in a metal case. Would the power supply require special filtering?

If anyone could offer any assistance it would be more than welcome.

Msg#: 7184 *GENERAL*
From: JEFF BACHIOCHI
To: SILAS ANTONE
Subj: REPLY TO MSG# 7135 (Z8 AUTO APPLICATION)

Silas,

The Zilog 28671 is still a neat chip even though it's been around a while. You don't need the quasistatic RAMs from the original article; standard 2K/8K/32K static RAMs are fine.

Development systems for it are a bit slim, but I know of two. Cross-assemblers for the Z8 are available through Micromint from Allen Ashley and from Micro Resources. You'll notice from all the talk on the board that the Intel 8052 (80C52 from Micromint) is used in many projects. It is slightly easier to use than the Z8. Every processor has its project, and the Z8 is well-suited for many!

As far as enclosing the processor in a metal box, you won't get away without shielding the whole computer from the EM1 produced by the ignition and charging systems. Noise will also try to enter the computer through the power supply. The 12V must be filtered to remove the noise riding on it.

This is a tough environment to work in, and is just one more example of how computers will control our future (be it good or bad).

--Jeff

Msg#: 7563 *GENERAL*
From: PETER SANDERS
To: STEVE CIARCIA AND ALL
Subj: CROSS-ASSEMBLER FOR Z8

Is there a cross-assembler available as a public domain program, and if so is it available for the IBM PC or clones? Thank you.

Peter Sanders
 Perth, W. Australia

Msg#: 7597 *GENERAL*
From: KEN DAVIDSON
To: PETER SANDERS
Subj: REPLY TOMSG# 7563 (CROSS-ASSEMBLER FOR Z8)

The only Z8 cross-assemblers that I'm aware of are commercial. One from Micro Resources is available from Micromint on an IBM PC disk for \$75. If someone does have a public domain cross-assembler, we'd be glad to post it.

-- Ken

We'll conclude this month with a discussion of how one might add numerous CGA boards to a single IBM PC so that one central processor can be used to display different status information in several locations at once.

Msg#: 7412 *GENERAL*
From: PETER SANDERS
To: STEVE CIARCIA
Subj: VIDEO CONTROL

Hello Steve,

Thank you for all the great articles you've written for BYTE. Your articles and solutions to other people's problems have prompted me to request your advice.

The company I work for has a need to control output to multiple videodisplays. CGA displays are preferred. I know that IBM PCs and clones can normally support up to two (different) displays. However, I need to control a minimum of three, and maybe as many as eight. This is in conjunction with the PC's normal operator display.

Of the displays, only one needs to be accessed (and updated) at a time. The possibility of using terminals has been discussed, and so far proves to be the only solution.

What I would like to be able to do is as follows:

- retain a standard PC display for the PC itself, PLUS use three or more additional CGA cards within a single PC
- select one CGA card
- update the screen
- deselect the screen
- wait for input from the keyboard or **comm** port
- repeat the above process

Effectively what this means is, by using Microsoft **QB4** and Hammerly Computer Services' PRO-BAS routines, write the update data to a dynamic array, select a display, do a block move to the standard screen RAM, deselect a display, perform more data updates in dynamic arrays, and repeat the process as needed to the display desired.

As you probably **realize**, each display has to maintain the last display screen (sent to that display) until the next update.

My preferred solution would be to use standard CGA cards modified to suit.

The next solution would be to have a card with multiple blocks of "screen RAM" each with its own video output.

The last solution would be terminals. Your **TermMite** video terminal is a serious contender for the last solution.

I hope you can be of assistance. Thank you in advance for any advice you can provide.

Peter Sanders

Msg#: 7447 *GENERAL*
From: STEVE CIARCIA
To: PETER SANDERS
Subj: REPLY TO **MSG#** 7412 (VIDEO CONTROL)

I've forwarded your letter to Ed Nisley, our resident PC guru. While I could offer a suggestion, he can give you solutions.

BTW, I'm no longer in BYTE after December. You'll have to start reading Circuit Cellar INK if you want the same quality material.

--Steve

Msg#: 7562 *GENERAL*
From: PETER SANDERS
To: STEVE CIARCIA
Subj: REPLY TO **MSG#** 7447 (VIDEO CONTROL)

Thank you for your assistance. I forgot to mention in my last message that I'm calling your BBS from Perth in Western Australia. I assume I should call back in the near future for some info from your PC guru.

Peter Sanders

Msg#: 7692 *GENERAL*
From: STEVE CIARCIA
To: PETER SANDERS
Subj: REPLY TO **MSG#** 7562 (VIDEO CONTROL)

Call again anytime. There are a few others on here from Australia, too.

--Steve

Msg#: 7637 *GENERAL*
From: ED NISLEY
To: PETER SANDERS
Subj: REPLY TO **MSG#** 7412 (VIDEO CONTROL)

Well, it can be done. . . but you've got to really WANT to do it because the whole thing is a little grisly . . .

First of all, you'll need an expander chassis to hold all those CGA cards. I'd strongly suggest that you use a commercial box rather than trying to roll your own.

Next, you'll need to hack the cards to allow you to selectively enable them under program control. Given the variety of cards available nowadays, there's no way to be too specific about this, but here's how it would work on the original CGA:

Essentially, you have to add a pair of gates to the memory and I/O address decoders, each with one input coming from the original logic and the other from an enable signal. When the enable is inactive, the card simply doesn't respond. When it's active, you get normal responses. It's your responsibility to make CERTAIN that only one card is enabled at a time, because two cards responding to the same memory or I/O read will give garbage on the bus.

Current clone cards probably use LSI decoders, so you may have some problems figuring out what to change. It may not be possible to modify some cards; they may have everything buried inside one undocumented LSI chip.

Failing all else, you can add the enable gates to the memory and I/O control lines (**-IOR, -IOW, -MEMR, -MEMW**) so that the decoding proceeds normally but the actual read and write signals are blocked. This will work with any card, but may mess up the signal timing a little. You may need 74F or 74S gates instead of 74LS; there's no way to tell without some experimenting.

The enables can come from an LS138 decoder "piggybacked" on a parallel port card, with outputs driving the new CGA enable gates. One of the **CGAs** will be selected by the data pattern written by the BIOS power-on tests, so that's the default CGA display during booting. After that, you can switch them around as needed. You may need two parallel port cards, one in the base unit and the other in the expansion unit, but that shouldn't be a problem -- I don't think running enable wires between the units is a good idea.

Selecting cards is then simply a matter of writing the appropriate binary value to the parallel port; if you've put the decoder on the low-order 3 bits you can select the cards by number. You could use the port bits directly, but then you'd be unable to GUARANTEE that only one card was enabled at once; the BIOS would probably enable a few of them during the power-on tests long before your code gets control.

You should probably include a Monochrome Display/Printer Adapter (or a Hercules clone) to act as the "normal" PC display for the **system** unit. This won't give you CGA graphics, but output directed to the MDA will always show up regardless of how scrambled your code gets!

Remember that the CGA displays can't be too far away from the cards; the TTL video signals were not designed for long cables. You may need low-capacitance wires or even commercial "boosters" if you're planning to wire up a building with remote displays. Make a few tests with a standard CGA before you commit to anything expensive.

Sounds like fun!

Msg#: 7768 *GENERAL*

From: PETER SANDERS

To: ED NISLEY

Subj: REPLY TO **MSG#** 7637 (VIDEO CONTROL)

Thank you, Ed, for your assistance. I haven't read the full text of your message at this time since time is money, however when I have read it and tried it out I will get back to you.

Regards

Peter

The Circuit Cellar BBS runs on a 10-MHz Micromint OEM-286 IBM PC/AT-compatible computer using the multiline version of The Bread Board System (TBBS 2.0M) and currently has four modems connected. We invite you to call and exchange ideas with other Circuit Cellar readers. It is available 24 hours a day and can be reached at (203) 871-1988. Set your modem for 8 data bits, 1 stop bit, and either 300, 1200, or 2400 bps.

SOFTWARE and BBS AVAILABLE on DISK

Software on Disk

Software for the **articles** in this **issue** of Circuit Cellar INK may be downloaded free of charge from the Circuit Cellar BBS. For those unable to download the files, they are also available on one **360K**, 5.25" IBM PC-format disk for only \$12.

Circuit Cellar BBS on Disk

Every month, hundreds of information-filled messages are posted on the Circuit Cellar BBS by people from all walks of life. For those who can't logon as often as they'd like, the text of the public **message areas** is available on disk in two-month installments. Each installment **comes** on two **360K**, 5.25" IBM PC-format disks and costs just \$15. The installment for this issue of INK (November/December 1988) includes all public messages posted during September and October, 1988.

To order either Software on **Disk** or Circuit Cellar BBS on Disk, send check or money order to:

Circuit Cellar INK
Software (or BBS) on Disk
P.O. Box 772
Vernon, CT 06066

or use your **MasterCard** or Visa and call (203) 875-2199. Be **sure** to specify the **issue** number of each disk you order.

THE TECHNOLOGY RESOURCE of the 90's



BACK ISSUES AVAILABLE!

Issue #1 - Premier -- Inside the Box Still Counts

- IR Motion-Controlled Video Camera Multiplexer
- Image Processing: Video Hand Scanner/Identifier
- RGBI-to-NTSC Converter
- Keyboard Scanning

Issue #2 - Techies

- Neighborhood Strategic Defense Initiative -- Tutorial Application on Ballistic Physics of Plastic Soda Bottles
- Home Satellite Weather Center
- Multichannel Temperature Logger
- Digitizing Infrared Signals

Issue #3 - Control Magic

- X- 10 Power-Line Command and Communication Transmitter
- Home Weather Center -- Weather Databases
- Software Emulation of Full-Duplex Serial Channels
- Video Signal Timing and Real-Time Interrupt Control

Issue #4 - Stepper Motors

- Stepper-Motor-Controlled Scanning Sonar Sensing
- 68000-Based Single-Board Peripheral Controller
- Stepper-Motor-Controlled Robot Arm
- Really Using the IBM PC Joystick Port

Issue #5 - Remote Video Surveillance

- ROVER: Remotely Operated Video-based Electronic Reconnaissance
- Focus on the MC68000 Peripheral Controller
- 10-Hz/8-bit Digitizing Board for the IBM/PC
- Precision Pulses -- Carrier-Current Transmission Timing

Send \$4.00 per issue (includes first class, shipping & handling) in check or money order to: Circuit Cellar INK, P.O. Box 772, Vernon, CT 06066. VISA and MasterCard accepted, call (203) 875-2199

Imagewise/PC -- The Digitizing Continues

Part 1 Video Basics

by Ed Nisley

Two years ago when we designed the **ImageWise** video digitizer system we wanted it to work with any personal computer, not just IBM PCs. An RS-232 serial interface was the simplest way to satisfy that requirement, so we traded off transmission time for generality. Even though we used run-length data compression to speed the data through the line, it seemed many people wanted faster transmission than a serial link could ever achieve.

The new **ImageWise/PC** is a video digitizer designed to capture and display images with 256 gray levels from NTSC, PAL, or SECAM video sources. It plugs into an IBM PC I/O slot and communicates directly with the CPU using I/O-mapped data transfers. Depending on the PC's bus speed and storage capacity, it can capture several fields per second.

The images can be saved in Publisher's Paintbrush gray scale format, using either 16 or 256 gray levels. These files are directly compatible with a wide variety of desktop publishing software, as well as other programs that use PCX files. They can be saved as 16-color files for use with EGA displays, too!

In this article I will describe the **ImageWise/PC's** overall design.

Later articles will cover the hardware and software needed to make it work.

It is always a good idea to start at the beginning, so I'll start by describing the video signal itself. With that in mind it will be easier to understand how the **ImageWise/PC** acquires and recreates an image. The **ImageWise/PC** can handle a variety of video standards so there's more to the subject than meets the eye!

There are three main video standards in use throughout the world. The NTSC (National Television Standards Committee) standard is used in the USA, Japan, some of the Pacific, and some of South America. SECAM (the French acronym for Sequential Color with Memory) is found in France, the USSR, and Eastern Europe. Much of the rest of the world uses PAL (Phase Alternation Line-rate) video. If you are familiar with computer standards you won't be surprised to find that there are variants and flavors of the "standards" in use throughout the world.

All of these standards specify color video transmission, but we need just the synchronizing pulses and video intensity. A filter in the **ImageWise/PC** video amplifier strips out the color information so that color and monochrome video sources

work equally well. This turns out to be more important than we thought, because many people use camcorders rather than stand-alone cameras -- and all camcorders produce color signals.

As you probably know, the image you see on the TV screen is produced by an electron beam projected from the back of the tube. That beam scans the faceplate from top to bottom along rows from left to right. Figure 1 shows a simplified sketch of the scanning pattern



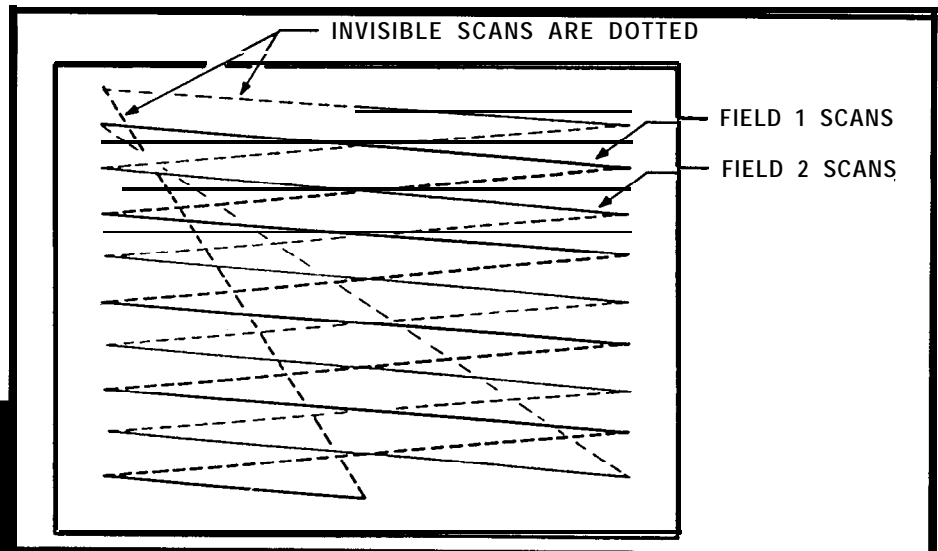


Figure 1 -- This is a simplified scan pattern for interlaced video. Since the beam is turned on and off mid-scan, only the top and bottom lines are visible to the viewer.

for a single video frame of two interlaced fields. All of the lines in the first field are completed before the second field begins. Notice that only half of the top and bottom lines are visible because the electron beam is turned off and on in mid-scan.

Each video standard has two critical parameters: the amount of time it takes to display a single frame, and the number of lines within that frame. NTSC video displays 30 frames of 525 lines each second, so a single line takes 63.5 μ s.

Both PAL and SECAM display 25 frames of 625 lines each second, so the line period is 64 μ s. The frame rate chosen by a country is generally close to its AC power line frequency, which helped to reduce "crawling" interference back in the early days of TV transmission.

The ImageWise/PC system uses composite video, which simply means that the timing and intensity information are encoded as voltage levels on a single wire. Photo 1 shows a picture taken from an oscilloscope measuring a standard NTSC compos-

ite video signal. The upper trace shows one of the two fields that make up a frame, while the lower trace is a magnified slice of the upper one, detailing a few lines near the middle of the field.

Photo 2 shows the lines near the end of the field. You can see that the final line is only half as long as the preceding lines, which marks the first video field. The first field (in NTSC video) ends with that half line, so the remaining sync pulses are in the second field. SECAM and PAL begin line numbering with the start of the field sync pulse, just to keep you on your toes.

Figure 2 is an overview of the sync pulses in a complete NTSC video frame. Notice that the vertical sync pulses in the field sync interval are lined up, but the visible lines in each field are offset by half a line time. This causes the interlacing shown in Figure 1.

There are 244 visible lines in Figure 2, but some video sources may produce fewer lines by increasing the number of blanked lines following the field sync. The ImageWise/PC accepts the maximum number of visible lines, so it



ImageWise/PC lets you display a stored digital image "over" a live analog image.

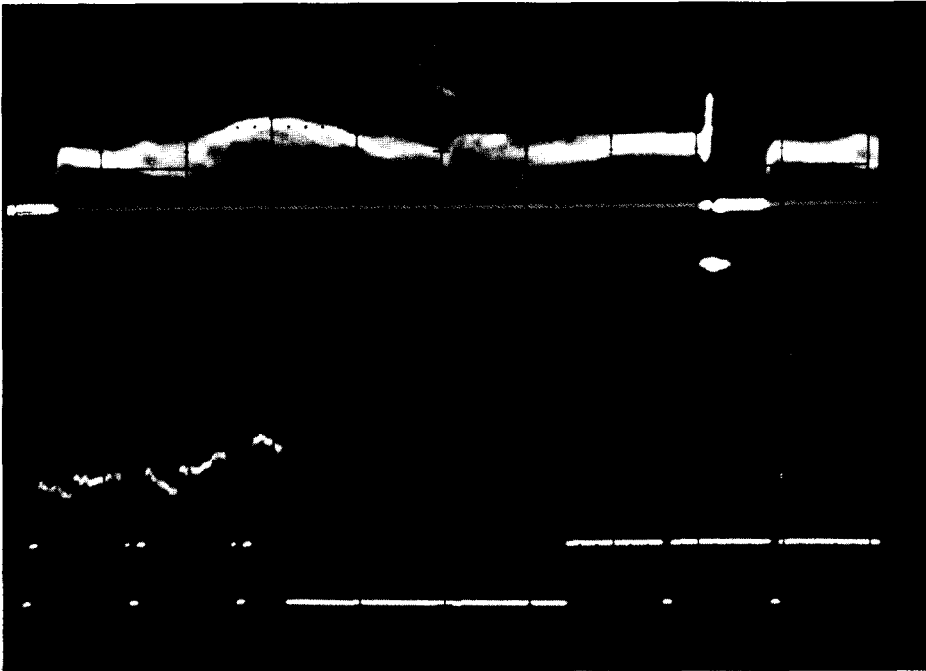


Photo 1 -- NTSC video displays 30 frames of 525 lines each second. The upper trace shows one of the two fields that makes up a single frame.

should work with any source. In any case there must be precisely 525 lines in each frame, with 262.5 lines in each field -- no more and no less!

TV 102 -- Voltages

With synchronizing under control, the next step is to examine a single line of video as shown in Figure 3. The most negative points of a composite video signal are the tips of the sync pulses and the most positive points are the brightest whites in the image. If the sync tips are set to zero volts, then the brightest white will be about 1 volt, although color modulation can push the signal slightly above that.

Mercifully, all three standards specify roughly 1 Vp-p video signals so the ImageWise/PC need not accept or produce a wide variety of amplitudes. Some cameras take liberties with the signal levels as well as the timings, though, so some adjustability is built into both the ADC and DAC circuitry.

There are three critical voltages shown in Figure 3: white, black, and blanking. The first two are easy enough to understand; when the voltage is at the white level the electron

beam is turned on completely and the screen appears white at that point. Similarly, the black level corresponds to the blackest areas in the image. Remember that all voltages are referred to the sync tips at ground or zero volts.

Blanking simply ensures that the beam is turned off during horizontal and vertical retraces. Although it is "blacker than black," if you turn your monitor's contrast and intensity controls up all the way you can see it. The gray border around the visible image is the blanked part of the video signal that is normally not visible.

Many video sources have AC-coupled outputs, so the ImageWise/PC must clamp the sync tips to ground. This ensures that the rest of the circuitry will find the other voltages at the expected levels. The ImageWise/PC video outputs are DC coupled.

You might wonder where the color information is hidden if the image brightness is controlled by the video signal voltage. It turns out that color is encoded by an AC

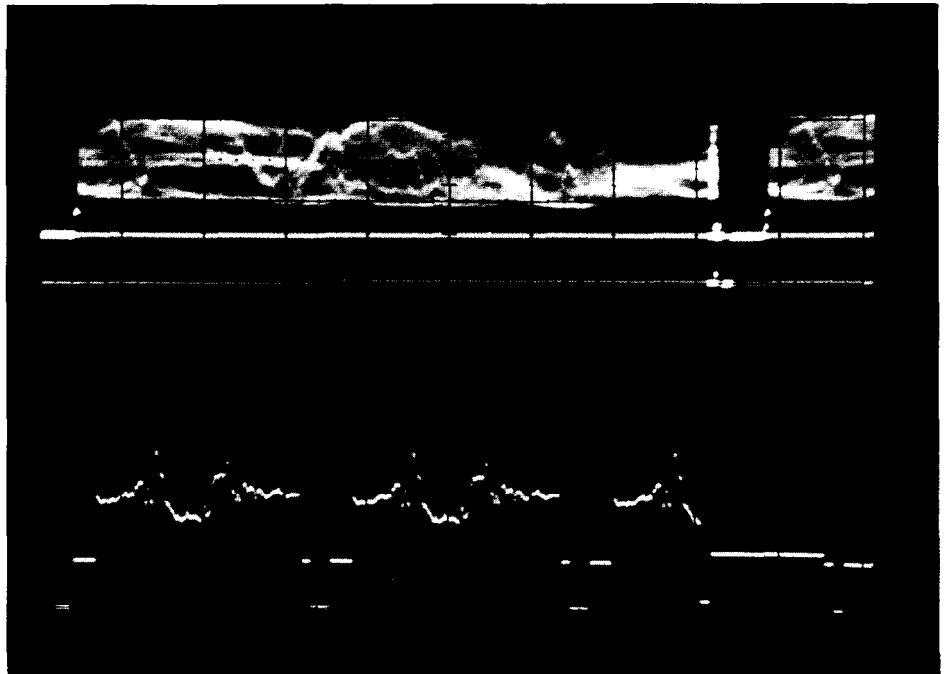


Photo 2 -- The end of the first field in an NTSC from. All remaining sync pulses will come with the second field.

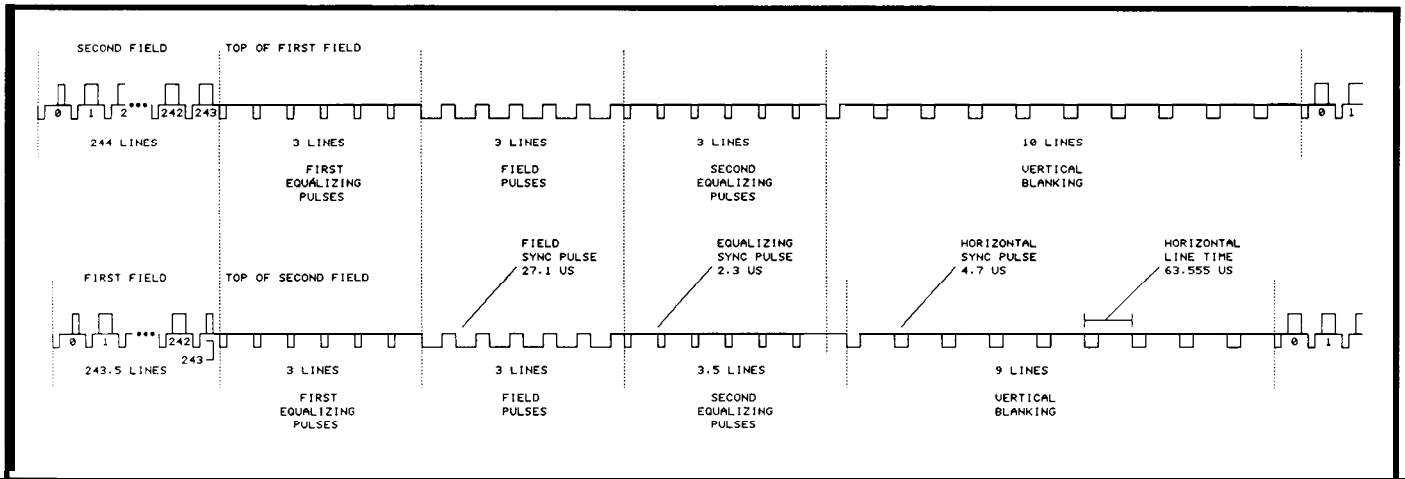


Figure 2 -- The sync pulses for a complete frame of NTSC video. Offsetting the visible lines while keeping vertical sync pulses aligned gives the interlacing shown in Figure 1.

signal superimposed on the intensity level. The filter I mentioned earlier removes that AC signal and passes only the average intensity level.

As you might expect, the three systems use three different frequencies to encode the color information. NTSC uses 3.58 MHz, while most PAL and SECAM systems use 4.2 or 4.4 MHz. Rather than design a complex double-notch filter, we just change the value of an inductor to select the frequency; you can choose an ImageWise/PC for either NTSC or PAL/SECAM. The filter is largely irrelevant if the signal source does not include color information, so either inductor (or none at all) will work for monochrome sources.

The Big Picture

Figure 4 shows an overall block diagram of the ImageWise/PC system. The circuitry can be divided into five major sections: an 8-bit flash ADC, a 64K-byte RAM buffer, an 8-bit DAC, the 8031 controller, and the PC bus interface. The new unit has several features that simply weren't possible in the older design, starting with faster data transfer.

The 64K-byte RAM buffer is

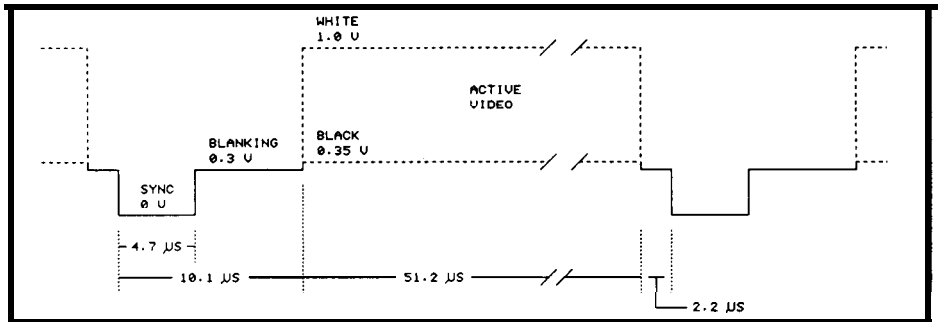


Figure 3 -- The levels of a single line of NTSC video. Lowest active levels are deepest black, while highest levels produce the brightest whites.

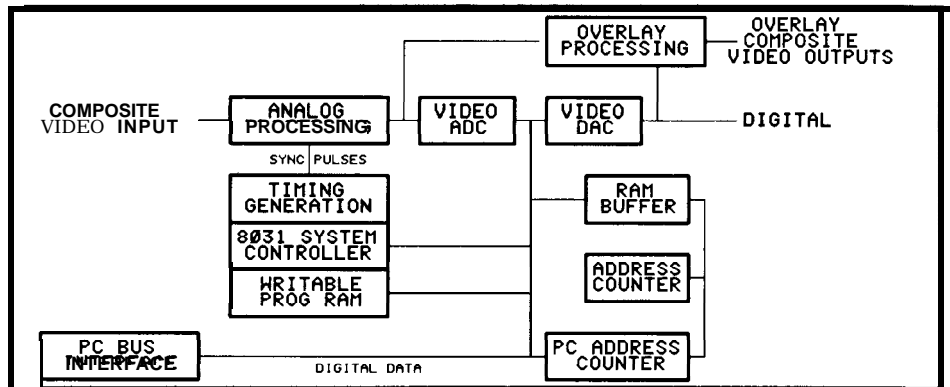


Figure 4 -- The five major sections of the ImageWise/PC are an 8-bit flash ADC, a 64K-byte RAM buffer, an 8-bit DAC, the 8031 controller, and the PC bus interface.

shared by the ADC and DAC circuits so the digitized images are immediately visible. In fact, the ImageWise/PC can show "real-time" video by digitizing every frame, then freeze a single frame by simply halting the

conversion. The PC can set up continuous digitizing until an alarm signal occurs; the ImageWise/PC will save the image at the time of the alarm.

Although it would be nice to be

8051 C COMPILER

SOURCE DEBUGGER

**Call today for a
FREE
technical bulletin**

**MICRO COMPUTER CONTROL
P.O. Box 275
Hopewell, NJ 08525 USA
Telex 9102404881 MICRO UQ**

**FOR IMMEDIATE ACTION
CALL:
(609) 466-1751**

able to digitize and save images in real time, the PC's data transfer rate simply isn't up to the task. Because the RAM buffer holds one field, the PC would have to read all 64K bytes in less than 1/60 second to capture every other field. That **simple**-sounding task requires a transfer rate of about 4 megabytes/second, far beyond the specs for ordinary PCs.

Unlike many image capture boards, the ImageWise/PC is accessed using I/O instructions rather than memory instructions. We chose to do this because memory addresses are a scarce commodity in PCs -- there aren't many 64K "holes" available in a fully-packed PC. In particular, a PC with an EGA, hard disk, and EMS memory to save all those images doesn't have any addresses left over!

Using I/O operations to transfer the images doesn't reduce the data transfer rate as much as you'd expect. The 80286 and 80386 processors include single I/O instructions that transfer data from a port into memory, increment the memory pointer, and loop until a count is satisfied. Older 8088 CPUs require a **three**-instruction loop, but the bottom line is that you can capture between two and ten images per second depending on your hardware and coding cleverness.

There is room for 256 lines of 256 bytes each in the 64K video buffer. One of the lines is reserved for internal use, so a video field can contain at most 255 lines. An NTSC video signal has about 244 active lines in each field, which fits nicely. Unfortunately-; PAL and SECAM signals have about 280 active lines, so the ImageWise/PC cannot capture a complete field.

It turns out, however, that most monitors have a lot of both horizontal and vertical overscan. This means that the top, bottom, and sides of the image aren't visible because the electron beam is off the screen during that part of the video signal. The

ImageWise/PC takes advantage of this **overscan** by centering the 255 available lines in the middle of the 280 active lines. While not ideal, it works out well enough in practice. The supporting software saves 255 lines to disk regardless of the video standard so that the files have a standard format; in NTSC mode the extra lines are filled with binary zeros.

Hands-off Twiddling

It would be nice if every camera in the world produced the same voltages and every monitor used the same inputs, but that's not the way it works. The ImageWise/PC uses a control DAC to set four control voltages to adapt the circuitry to varying input and output requirements. Because the adjustments are under computer control, you don't have to open your PC's case to adjust the ImageWise/PC.

The flash ADC that digitizes the input video requires two reference inputs to define the black and white voltage levels. Ordinarily a pair of trim pots set these two levels, but the ImageWise/PC uses two control DAC outputs instead.

The ADC converts the range of voltages between the black and white references into 256 binary numbers between 0 and 255. Generally 0 corresponds to the blackest region in the images and 255 to the whitest white, but there are some applications where you are more interested in a particular band of grays. A PC program can analyze the content of an image and set the reference voltages on the fly.

Incidentally, the reason the analog-to-digital converter is called a "flash ADC" is that it converts the video input into a digital output in one step. The chip contains 256 analog comparators that test the input against a fraction of the reference voltage range. A logic network converts the 256

comparator outputs into an 8-bit digital code. The CA3318 ADC is an enhanced version of the CA3306 converter used in the serial **ImageWise**, which produced 64 gray levels using 6 bits.

On the other end of the video chain, the ImageWise/PC uses an 8-bit video DAC to create the output image. A third control DAC output sets the video DAC's reference current, so the overall video brightness is also under program control. The video DAC can drive either one or two terminations (75-ohm loads) and the control DAC makes it a simple matter to adjust the output for the right level.

The video DAC is the same TML1852 we used in the original **ImageWise**. In that design the firmware shifted the six bits from the CA3306 ADC left by two bits so the output voltage covered the full dynamic range.

While the control DAC reduces the number of trim pots on the ImageWise/PC board, you must still select the video termination and board address using jumpers. There didn't seem to be much justification for changing any of those under program control.

The board can store the 8031 firmware in either RAM or EPROM, with either 8K- or 32K-byte chips. The PC can load different programs into the RAM version, while the EPROM is cheaper for fixed applications. Although the 8031 can access the video buffer, you will find it faster to transfer the data into the PC's memory for "heavy duty" signal processing.

Using RAM for program storage is particularly helpful while you are developing a new 8031 program. It eliminates much of the need for a development system, although debugging is still a challenge. The ImageWise/PC isn't set up to work with the DDT-51 debugger, but it should be reasonably

simple to make the adaptations: I've left the INT1 and T1 pins free for those of you who are up to the challenge.

Getting the Picture

Although the flash ADC converts the analog video input into digital values, it must be turned on and off at the right moments so the top of the scene winds up at the start of the RAM. The 8031 microcontroller handles all of the sync timing and image registration, assisted by a little logic to handle the high-speed details.

The NTSC sync pulses shown in Figure 2 show an idealized picture. In principle, the ImageWise/PC has a simple task: determine when the field sync starts, count out the right number of pulses, and turn on the ADC at the start of the first visible line. It's not quite that easy, of course.

Because the two fields are in-

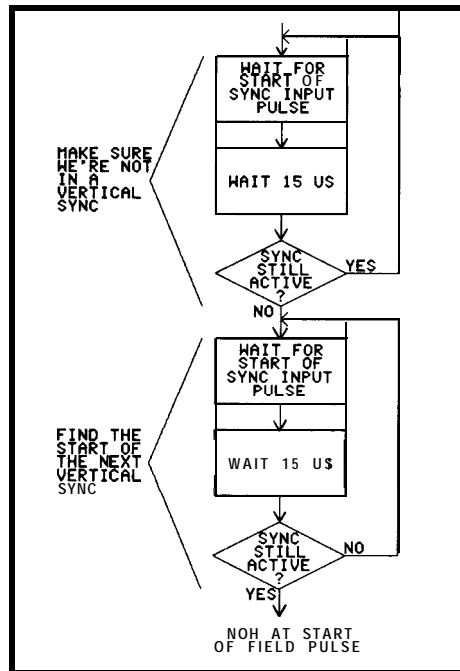


Figure 5 -- *The firmware takes advantage of the fact that the vertical sync pulses are longer than horizontal sync pulses to find the beginning of each frame.*

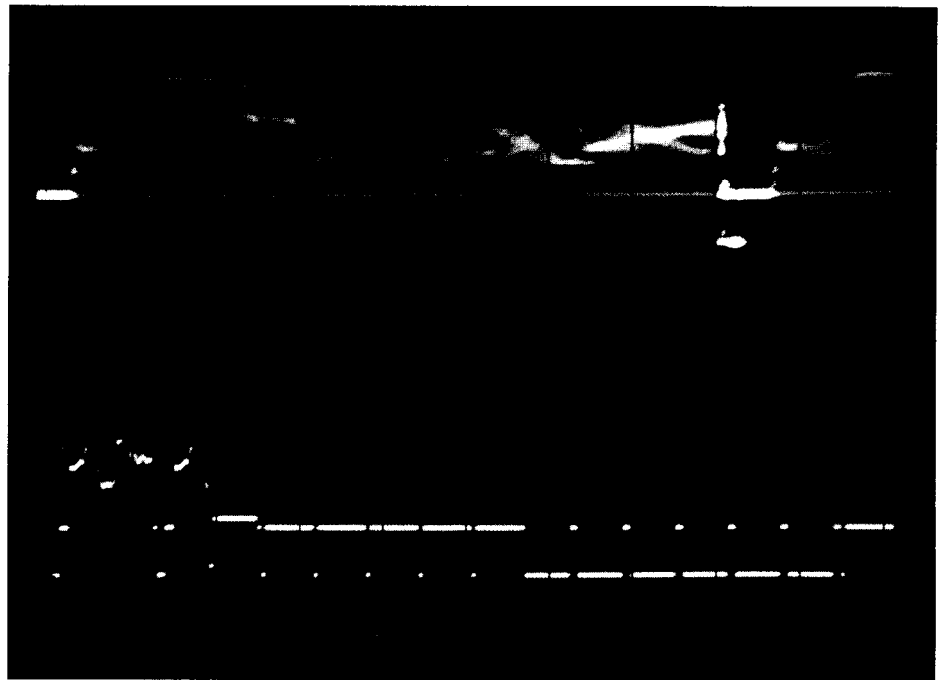
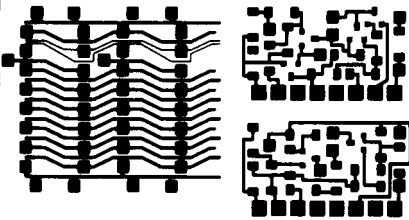


Photo 3 -- *Some Video cameras take liberties with the NTSC standard. This photo is similar to Photo 2, but was taken from a camera that omits equalizing pulses, blanking before the field sync, and some vertical blanking.*

DROEGE

DESIGN ROBOT FOR
THE ORIGINATION OF
EXACTING GRAPHIC
ENGINEERING

A MANUAL PRINTED
CIRCUIT CAD/CAM
SYSTEM



MULTI-LEVEL SYMBOL CONCEPT.
MEMORY LAYOUT ABOVE IS A
SYMBOL MADE FROM TWO CHIP
SYMBOLS WITH ADDED BUS WIRE
CHIP SYMBOLS ARE MADE FROM
MULTIPLE PAD SYMBOLS.

BASIC \$10.00 POSTPAID:

CGA 3 COLORS 12 LAYERS
64 BY 64 INCH WORKSPACE
ANY GRID TO 0.001 INCH
RUNS ON ANY PC COMPATIBLE
15 LINE WIDTHS
DOT MATRIX OUTPUT
200KB DOCUMENTATION ON
TWO DISKS
16 WORK AREAS SAVEABLE
STITCH BETWEEN LAYERS
ZOOM AND PAN TO ANY SCALE

ADVANCED \$100.00 POSTPAID :

ABOVE FEATURES
HIGHER RESOLUTION 15 COLORS
LARGE R JOBS
MOUSE
ADVANCED EDITING
PRINTS 0 MANUAL

ENVIRONMENTAL OPTICS CORP.
P.O. BOX 296 BATAVIA, IL 60510
SEND BASIC ADVANCED INFORMATION
CHARGE VISA MC PAYMENT ENCLOSED
CHARGE NUMBER _____ EXP _____

NAME _____

ADDRESS _____

CITY _____

STATE _____ ZIP _____

TELEPHONE ORDERS EVENINGS (312) 879-2949

THIS AD WAS COMPOSED WITH DROEGE AND
PLOTTED ON OUR PHOTO PLOTTER. WE CAN MAKE
PHOTO PLOTS FOR YOU FROM PROGRAM OUTPUT.

terlaced in a single frame, the ImageWise/PC must digitize the same field every time to prevent vertical jitter. This jitter would be most evident in the "real-time" mode I mentioned above, where the digital image echoes the analog input. Therefore, the 8031 must be able to distinguish between the two fields.

Although there are 262.5 video lines in each field of the frame, there are 272 sync pulses in the first field and 271 in the second (count them carefully!). Those sync inputs generate hardware interrupts for the 8031, so counting sync pulses means counting interrupts.

The first step is to locate the field sync pulses in the input video. This turns out to be straightforward because the vertical sync pulses within the field syncs are longer than horizontal or equalizing syncs. The 8031 waits for about 15 microseconds after the start of a sync pulse; if the input is still active (low), it must be a vertical sync. To locate the first vertical sync, the firmware identifies a horizontal sync and then waits for the next vertical. Figure 5 shows a flowchart of this process.

Determining which field is which requires one complete field time, or 1/30 second. During that time the 8031 is so busy that it can't stand any interrupts, so the video output must be disabled. The end result is a pair of numbers that represent the number of sync pulses in each field.

The next step is to wait until the first field comes around again. This may be immediate if the count started at the top of the first field, or it may require waiting through the second field. In the worst case synchronizing to the external video can require just under two complete frames, or 1/15 second.

In any event, once the top of the first field occurs again, the code counts out the appropriate number of sync pulses to account for vertical blanking, and enables the ADC at the start of the first visible line. Whether

the video source is actually producing video on that line is irrelevant because the ADC simply records whatever is coming in. The firmware counts out the active lines and turns off the ADC at the end of the last one.

That's how the system works with standard NTSC sync. Unfortunately, some camera manufacturers reduce their costs by taking liberties with the standard. They rely on the fact that most monitors don't need absolutely clean sync pulses to produce an acceptable picture. Photo 3 is similar to Photo 2, but shows a camera that omits all of the equalizing pulses, doesn't bother with blanking the signal before the field sync, and has fewer lines of vertical blanking than expected.

While analog systems can cope with this sort of nonsense fairly easily, the all-digital ImageWise/PC has to take special precautions. It's obvious that counting syncs and delaying by predetermined counts will work only if the incoming video matches your expectations. The ImageWise/PC firmware includes some tests that help it to cope with oddball sync signals, but it can't handle everything.

For example, one camera we tested didn't lock the horizontal and vertical sync signals together! The horizontal syncs "walked" past the vertical sync, so that our system of using horizontal pulses to count out delays didn't work. We think that camera was defective, but it certainly was a surprise!

Most modern cameras use digital sync generators rather than analog circuits, so they produce nearly perfect sync pulses. If you have trouble with your camera or video source it will be worthwhile to check it out using an oscilloscope. If the vertical interval looks more like Photo 3 than Photo 2, you know where your troubles are coming from ...

The flash ADC and the RAM address counters are driven by a 5-MHz clock signal. Every 200 ns a new pel is written into the RAM and the address counters step to the next address. After 256 writes, the counters automatically pause, wait for the next sync pulse, and start after the horizontal blanking interval. The blanking time is controlled digitally by the firmware, so different cameras can be handled without adjusting a trim pot.

In the original ImageWise design, the high-order byte of the video address came from an 8031 output port, which meant that the

input during acquisition. In effect, the ADC and DAC form a rather expensive piece of wire during that time!

Showing Off

After the conversion is complete, the RAM buffer contains the digital equivalent of the analog video field. The firmware disables the ADC and the DAC then accepts and converts data read from the RAM. That data produces the digital image as a "frozen" version of the analog input.

The TML1852 DAC is designed for video applications and produces

this article shows a live "analog" scene with a stored "digital" overlay inserted into it.

The overlay is controlled by a comparator circuit that matches video intensity against a reference level supplied by the control DAC's fourth output. Whenever the intensity exceeds the reference, the video output is switched to the stored digital image. Conversely, when the intensity is below the reference, the video output is switched to the analog input.

A DG202 quad analog switch handles the video overlay switching. Because the IC has four

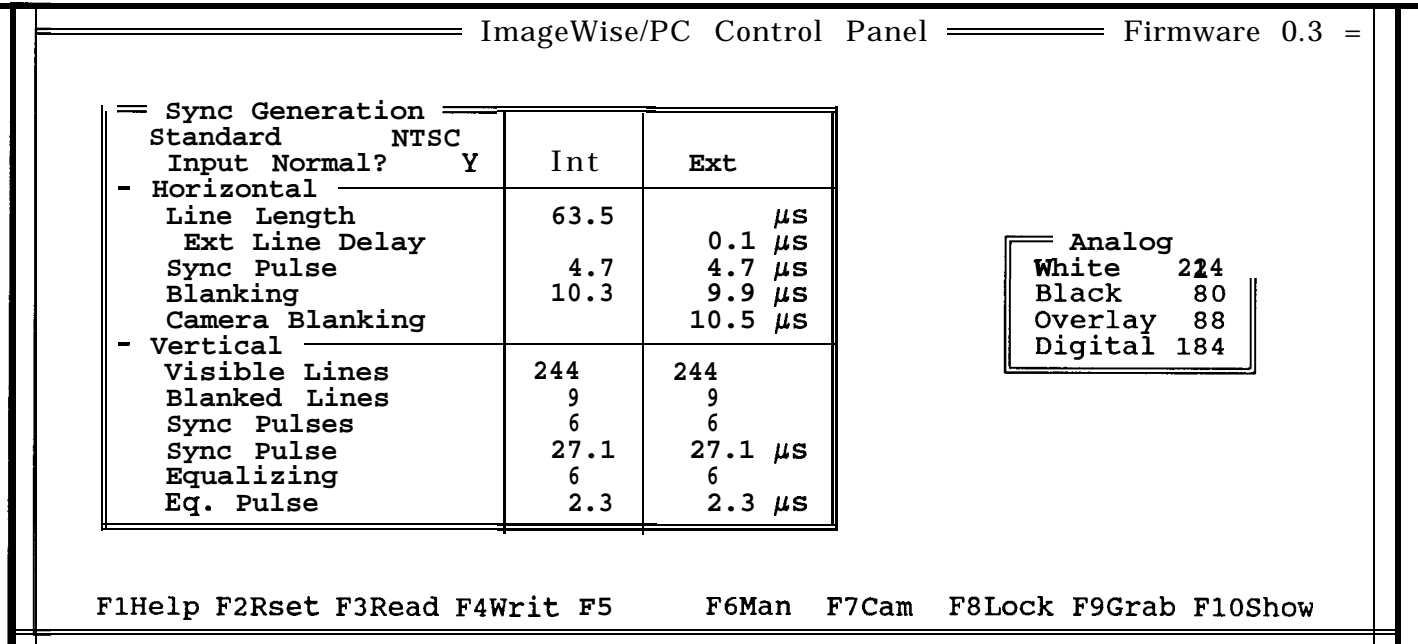


Figure 6 -- SETIMWPC allows the user to set levels to adjust for a broad variety of input types and qualities. This screen from the software shows the levels that are software controlled.

firmware had to update that address immediately after the sync pulse. The new design uses hardware for both bytes so the timing isn't quite as critical. The firmware counts interrupts and disables the hardware counters after the last valid line.

Because the ADC and DAC share a common video RAM buffer and use similar control signals, the digital output echoes the analog

correct sync and blanking voltage levels as well as the full range of 256 gray levels. The sync and blanking inputs are driven by control signals from the 8031 and the RAM address counters, which I'll describe in the next issue.

Because the ImageWise/PC card includes both video input and output circuitry, it was easy enough to add an overlay capability that merges the two images. The opening photo of

switches, I added the ability to route either the digital or analog video signal to the comparator. This allows you to control the location of the overlay using the intensity of either the digital or analog signal.

The DG202 switch has built-in "break-before-make" action to avoid shorting two inputs together. This produces a short gray dot during the switching interval when

both inputs are turned off. I could have used a faster switch or more complex circuitry to reduce the border, but the overall effect isn't offensive. While not up to broadcast standards, it suffices for simple special effects and titling.

When overlays aren't needed, the overlay output normally echoes the analog input. You can also adjust the overlay switching level so that the overlay is active all the time, in which case the overlay output echoes the digital signal.

The output of the overlay switch passes through a buffer amplifier to drive the overlay output. If your application doesn't need overlays you can omit a handful of parts to reduce the cost of the board.

Setting Up Exercises

One of the requirements of the new ImageWise/PC design was to control nearly all of the timing and voltage settings using firmware rather than trim pots. While this gives a great deal of knob twiddling freedom, it does pose a problem: where should all the parameters be stored?

The standard ImageWise/PC board has the 8031 program stored in EPROM, so when you turn on the power, the program begins running immediately. That program sets the hardware for normal NTSC input and output, which should match the majority of cameras and monitors. PAL and SECAM versions are available on special order, though, if NTSC isn't your cup of tea.

However, the default settings may not match your requirements. The ImageWise/PC board comes with a PC program on diskette that presents a full-screen display of all the setup constants. You can adjust these until the results on the video monitor are perfect, then create a disk file holding all your defaults.

Figure 6 shows the display created by SETIMWPC.EXE.

Once you have created and tested the configuration file using SETIMWPC, you can use LOADIMW.EXE to read it and send it to the board without manual intervention. If your ImageWise/PC uses RAM program storage, LOADIMW can also read an Intel hex file and load the 8031 program. You can insert this command into your AUTOEXEC.BAT file to automatically set up your ImageWise/PC board whenever you boot your PC.

Back when we wrote the original ImageWise GRAB program, there was no standard file format for 64-gray-scale images, so we used a simple binary file. Now that computer displays have finally caught up with analog monitors (by using analog inputs!), several paint programs can handle pictures with 256 gray levels. The new GRAB program can store images in a variety of formats, one of which should suit your needs.

The simplest format is raw binary, which holds 255 lines of 256 bytes each. It turns out that there is little benefit in run-length compression on a 256-gray-level picture because each pel is slightly different from its neighbors, so raw binary format files always require 65280 bytes.

The new version of ZSoft's Publisher's Paintbrush can support 256 gray levels on some displays, so GRAB can create a PCX file directly. There are several choices for picture size to suit either VGA displays in 320x200 mode or 8514 displays in 640x480 mode.

GRAB can also store the images in the PC Paintbrush 16-gray-level format for use with VGAs in 640x480 mode or EGAs in 640x350 mode. EGAs will show the grays as 16 different colors, but at least the image is in a standard format that you can play with!

Finally, GRAB can store the images as 64-gray-level files in the

old ImageWise format, so you can interchange files between the systems. Of course, you won't get the full tonal range that the new hardware can produce, but the original ImageWise was pretty good, too!

The new SHOW performs the opposite translation, reading a PC disk file and writing the image into the ImageWise/PC's video buffer. The program can handle any of the file formats. It can also display the images as an overlay, which gives you the opportunity to create overlays using a paint program and insert them into live analog images.

Next Time

Well, I've given you an overview of what we've made. In the next article, I'll begin talking about specific details of the hardware and software that make up ImageWise/PC.

Reference

If you are doing any work with video at all, you need a copy of the *Television Engineering Handbook* (ISBN 0-07-004779-0). It is edited by Blair Benson and published by McGraw-Hill Book Company. Whatever you need to know about TV is in there somewhere! ■

WRITE FOR INK!

Writing technical articles may not make you rich and famous but it might be just the incentive to finish that 100-MIPS computer you started last summer. Or, if your expertise is software, perhaps it's time you presented your talents to the world.

Unlike most narrowly specialized publications, Circuit Cellar INK's charter is to cover a wide variety of hardware and software technology and ideas.

Send your project outline to:

Curtis Franklin, Jr.
Circuit Cellar INK
P.O. Box 772
Vernon, CT 06066

or contact him on the Circuit Cellar BBS at (203) 871-1988.

Build A Remote Analog Data Logger

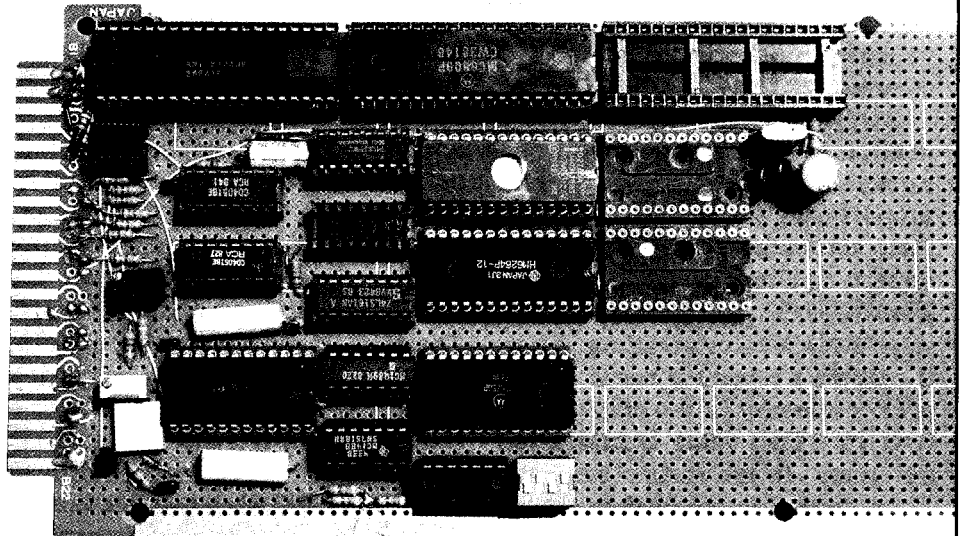
Part 1

A Simple, 6809-Based Data Acquisition System

by R.W. Meister

At one of the local computer club meetings, talk turned to projects that would be useful, inexpensive, simple, and marketable to experimenters. After listening to several rounds of suggestions, I thought about a stand-alone device that could measure several voltages and present the values to a computer or hard-copy terminal for logging purposes. Since I specifically wanted a device to take meter readings on an FM broadcast transmitter where I was working at the time, it would have to be accurate enough to meet FCC regulations. These units typically provide metering connections for remote control units that can measure ± 2 volts DC within a few percent. I also wanted to measure the temperature of the transmitter room, and the inlet and outlet air temperatures of the transmitter itself. With the addition of several LM335 temperature sensor devices, I thought that my idea would be able to fit the bill.

There were several \$250-and-up A/D devices already on the market. Inspectioning their advertisements showed that they typically read only one channel, and that their prime function was to continuously transmit the sampled voltage on a serial communications line. What I really wanted required a little more intelligence and the ability to handle more than one input without any external hard-



ware. The resulting project meets these requirements with ease, and allows me to measure temperature and actually see the value in degrees rather than in some voltage equivalent.

I discussed my project with Steve Ciarcia who was kind enough to provide me with a copy of an article ("Try an 8-Channel DVM Cocktail") from the December 1977 issue of BYTE Magazine on a similar project that used a Motorola MC14433 3-1/2 digit analog-to-digital chip. The entire project used six integrated circuits. I decided to use the same converter chip and make the box much more intelligent, because I wanted to use a dumb terminal with it rather than a desktop computer.

This is a two-part article. This part discusses the project's general design and hardware. The second part will look at the software.

Hardware & Software Design Goals

The original design specifications for this device were, for the most part, inherent in the MC14433: 1-millivolt resolution, ± 1 count of

least-significant digit accuracy, range from -1.999 to +1.999 volts DC, high input impedance, and minimal support components. Getting from 8 to 16 channels was as simple as connecting the output of two eight-channel multiplexers. I needed the box to provide 300- and 4800-bps communications, and I wanted it to support several other rates.

Other features were implemented in software, which I wrote in C. I decided to use C because I'm familiar with it and it deals with hardware on a level that approaches assembly code. In C, it was relatively easy to program features like XON/XOFF flow control, half- and full-duplex communications with odd, even, mark, or space parity, and automatically sending a linefeed after every carriage return. Part 2 of this article will explain the software in full detail.

Dual-Slope Analog-to-Digital Conversion

Since the purpose of this project is analog-to-digital conversion,

some explanation of how the process takes place is in order. The MC14433 performs a "ratiometric" analog-to-digital conversion, where the input voltage is measured as a ratio of a reference voltage. Therefore, a full-scale voltage of 1.999 volts requires a reference voltage of 2.000 volts. In industry terms, the MC14433 is a dual-slope ramp converter, since a capacitor is first charged at a rate proportional to the input voltage, then discharged by a reference voltage applied to the converter. The MC14433 goes through several steps when it takes a reading. First, it charges a capacitor based on DC offsets inherent in the chip's components. This value is subtracted out later. In the second step, the input voltage is sampled to charge the capacitor at a linear rate. This charging takes place over a fixed time that is determined by the MC14433's basic clock rate. Then, the input switches to the reference voltage which now attempts to discharge this same capacitor. The time the capacitor takes to reach its original voltage is measured in clock counts. The discharge time is directly proportional to the original input voltage. Since the same components are used to both charge and discharge the capacitor, the input voltage is directly equal to the ratio of the time periods multiplied by the reference voltage. This time is then presented as some number of counts, up to 1999. In between each reading, the chip reads the voltage at analog ground and saves this value which is then subtracted from the charged capacitor during the discharging time. Together

with the charge that was obtained in the first step, the A/D chip can automatically cancel out any internal offset voltages. This auto-zeroing makes for a much more accurate, self-compensating device.

The MC14433 chip does something a bit peculiar when the input voltage changes polarity. Understand, that since 16 channels of an unrelated voltage can be measured, when polarity changes are inevitable. When polarity is reversed, the charging step described above actually discharges the capacitor, so that the discharging step immediately stops. The chip presents a reading of zero, inverts the input signal, and takes another stab at the conversion. This time, it should be successful and the erroneous zero value has to be thrown away. The software handles this by waiting for a second reading to come in after a zero value. If this reading is also zero, then a true value of zero volts is displayed. Of course, if an input signal continues to shift polarity on successive readings, the chip can never return a valid reading, so the software will return a value of zero volts, which is appropriate as the DC value of an AC signal. If the voltage was of the correct polarity but the discharging phase of the cycle never brings the capacitor back down to zero, then the voltage must have

exceeded the ± 2000 -count range of the device and an over-range condition occurs. This normally doesn't cause any damage to the device, since its input pin can go from +5 to -5 volts. Due to the ratiometric operation, if the charge time exceeds the discharge time, the input voltage must be higher than the reference voltage, so the chip can only report an over-range condition.

The specifications for the MC14433 indicate an accuracy of $\pm 0.05\%$ of reading, ± 1 count, an input impedance greater than 1000 megohms, and up to 25 conversions per second. In this project, the display shows millivolts directly, and is typically accurate to within 2 millivolts. With the device clock set near 400 kHz, and each conversion taking 16,400 clock cycles, the chip can take about 24 readings per second. With some channels requiring two readings (due to polarity changes), the effective sampling rate is typically 12-16 readings per second. The software can keep up with this rate and output 16 channels' readings only if the terminal is operating at 4800 bps. Any slower and data could start accumulating in the output buffer, causing readings to be occasionally missed.

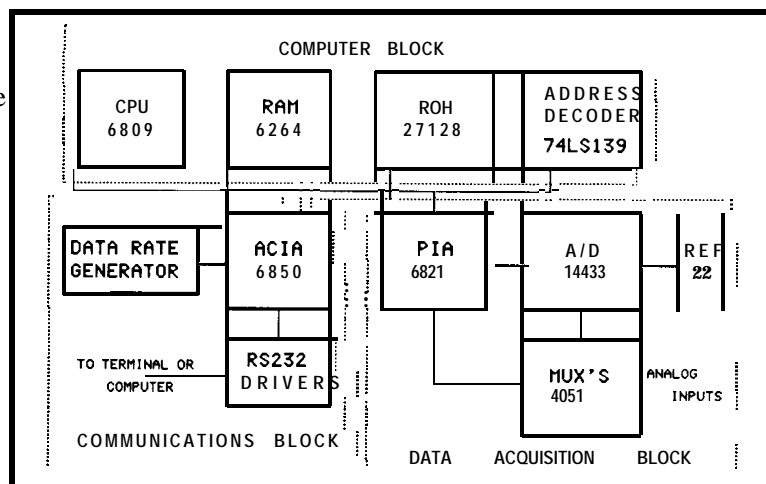


Figure 1 -- The hardware design is broken into four modular blocks. The power supply block, described in the text, is not shown here.

Building Block Design

The A/D Serial Box consists of building blocks, shown in Figure 1. The computer block consists of the M6809 CPU, some RAM for the program's variables, some EPROM to hold the program, and an address decoder for the various devices. The communications block consists of an M6850 ACIA (Asynchronous Communications Inter-

face Adaptor), some RS-232 transceivers, and a shift register and divider chip to provide data transfer rates for the ACIA. The data acquisition block consists of an M6821 PIA (Peripheral Interface Adapter), the MC14433 A/D chip, two eight-line multiplexers, and the reference voltage power supplies. A power supply block, not shown, supplies regulated positive and negative 5 and 12 volts.

Computer Block

The M6809 CPU (U1) is an 8-bit microprocessor that can address 64K of memory. With a 4.0-MHz crystal, the computer has a basic clock time of one microsecond, and most instructions are completed

within three clock cycles. Internally, there are two 8-bit accumulators, two 16-bit indexing registers, two stack pointers, a program counter, and a condition code register. Many of the current microprocessors have similar architectural features. PB1 is a switch that forces the CPU to reset.

As you can see in Figure 2, the A/D box utilizes two of the CPU's four possible interrupt inputs. The RAM I used (U2) was a 6264 8K-byte by 8-bit static device. The program actually uses only a fraction of this space (approximately 1600 bytes), but the chip was on hand and is a very common component. I used a 27128 16K-byte by 8-bit EPROM (U3). Like the RAM, the EPROM is only partially used. In fact, during the debugging phases of the project, the

entire program and data area fit into the RAM so changes could be made and new versions downloaded without burning the program into an EPROM every time.

The address decoder (U6) simply takes the two highest address lines and creates four individual select lines: one for the RAM, one for the ACIA, one for the PIA, and one for the EPROM. This divides the 64KB address space into four equal parts of 16KB each. The 6264 RAM chip actually doesn't care whether the computer is addressing the 0-8KB range or the 8-16KB range. The 27128 EPROM does fully decode its 16KB address space. The ACIA and PIA will be decoded many, many times within their 16KB address space, but since

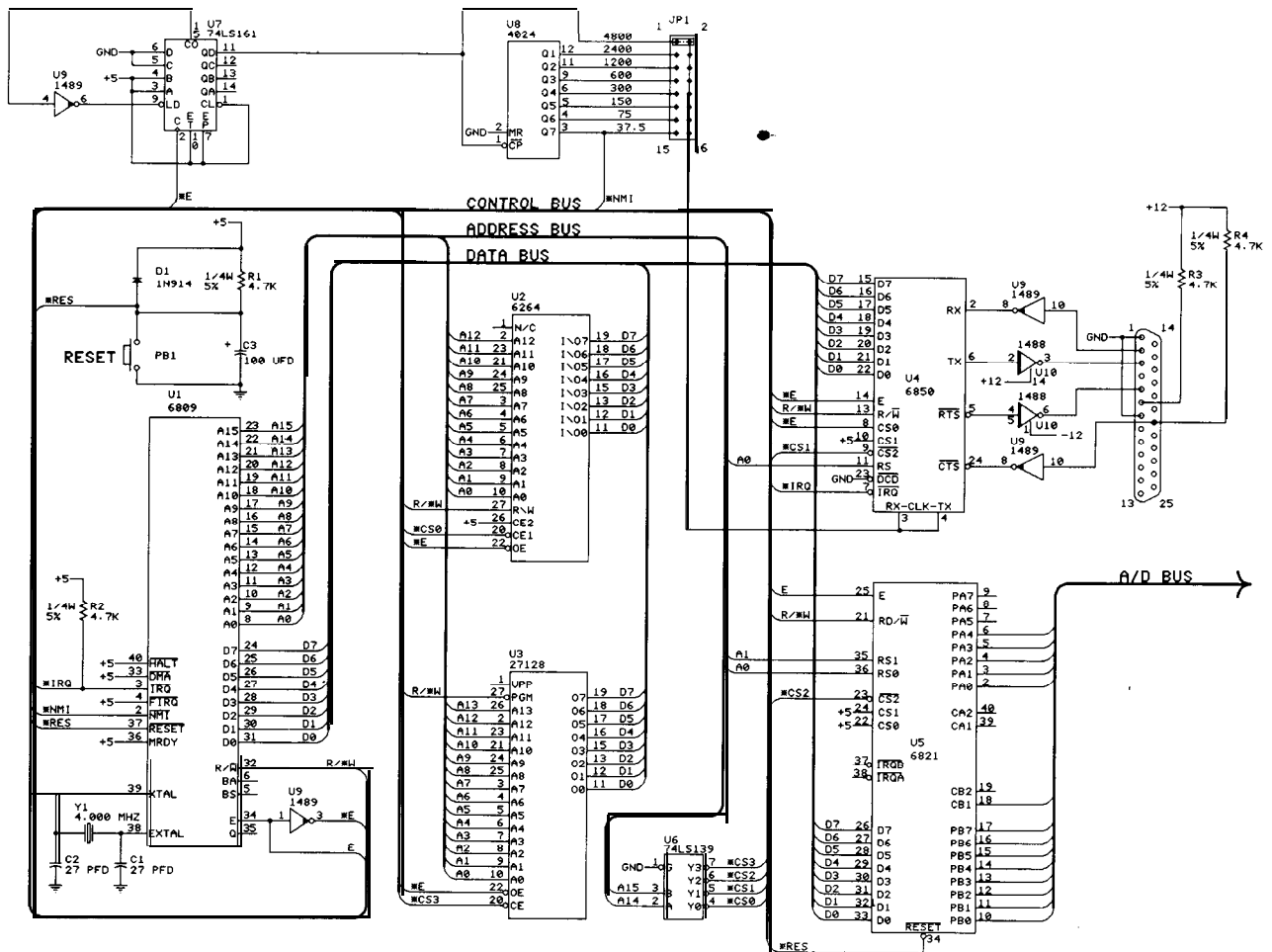


Figure 2 -- The CPU and Communications blocks of the design. The 6264 RAM and 27128 EPROM are more than sufficient for this design, but their ready availability and inexpensive price justifies their use.

CAN YOUR SCOPE DO THIS?



- Display 8 Channels of Logic Signals
- Store 16 K of Logic Events
- Capture Pre & Post Trigger Logic Events
- Trigger From Any Combination of 8 Qualifiers
- Capture Data Using Internal or External Clock

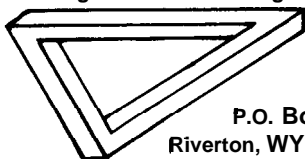
The A-96 can make your scope do this & more for only \$569.00

TO GET ALL THE FACTS ABOUT THE A-96 CALL
BECKTRON

AT 1-800-592-4141

and we will rush you all the details about the New A-96

BECKTRON
Honing The Technical Edge



P.O. Box 528
Riverton, WY 82501

Also Ask For Our 1989
Logic Test Equipment Catalog

the program only references one distinct address, this is not a problem. This "incomplete" addressing is often done when address space is not a concern and component costs need to be kept to a minimum.

Communications Block

The M6850 serial interface chip (U4) provides the functionality of a common UART (Universal Asynchronous Receiver/Transmitter) chip, accepting serial data from a terminal and converting it to parallel data for the CPU, and transmitting parallel data from the CPU serially to the terminal. The program configures the ACIA for 8 data bits with 1 stop bit. These devices typically require a clock rate that is 16 times the desired data rate, so for 4800 bps, the clock should be 76.8 kHz. Fortunately, most of the other data rates are submultiples of 4800 bps (i.e., 2400, 1200, 300). The data rate generator takes the 1-MHz 'E' signal that comes out of the M6809 CPU chip and feeds it into the 74LS161 counter (U7). The 74LS161 can count upward from 0 to 15, then continue back at 0, but in this application it is configured so that it resets itself to 3 rather than 0. It only gets to count 13 times before resetting, hence its output is 1/13th of its input (1 MHz) signal. It is a synchronous counter, meaning that all of the logic gets clocked at the same time, and no glitches or short unwanted-pulse-type signals come out of it. The resulting 76.923-kHz signal is only 0.16% faster than the desired 76.8-kHz signal, well within the $\pm 2.0\%$ data rate error that most UARTs can handle.

The highest data rate available with this hardware is 4800 bps, with slower data rates provided by the 4024 7-bit divider chip (U8). It generates 1/2, 1/4, 1/8, etc., signals from the incoming 4800-bps clock signal. In addition to providing several user-selectable communications rates, the

slowest data rate (37.5 bps) provides a 601-Hz real-time clock that is fed to one of the CPU's interrupt inputs. The software counts 601 of these interrupts to measure 1 second and keeps track of minutes and hours based on this signal. Although the clock runs 0.16% slow, it takes many hours of running before the time is off by a minute or more. As shown in Figure 2, this 601-Hz signal is connected to the CPU'S "nonmaskable" interrupt input, which is always enabled. This means that no matter what the CPU is doing, this signal has the ability to interrupt the running program to keep time. The ACIA, when it has received a character from its serial input, also generates an interrupt, but this one is able to be "masked"* or postponed by the CPU if the program desires. In actuality, this interrupt is never ignored, and comes in on its own signal line purely for simplicity.

Several of the communications lines are handled by industry-standard RS-232C driver and receiver chips (U9 & U10). These properly convert TTL-level signals to and from the ACIA to ± 12 -volt signals that drive the terminal. These chips can be replaced to handle RS-422 or RS-423 interface levels.

Data Acquisition Block

The third building block, shown in Figure 3, consists of the PIA, A/D converter, input multiplexers, and reference power supplies. The PIA (U5) selects which channel and which input multiplexer will be connected to the A/D converter. It also presents the digit and value signals from the A/D converter to the CPU and monitors the EOC (End Of Conversion) signal that tells the program when the A/D converter has taken a reading and its value is available. The A/D chip (U11) continuously outputs digit information (1st, 2nd,

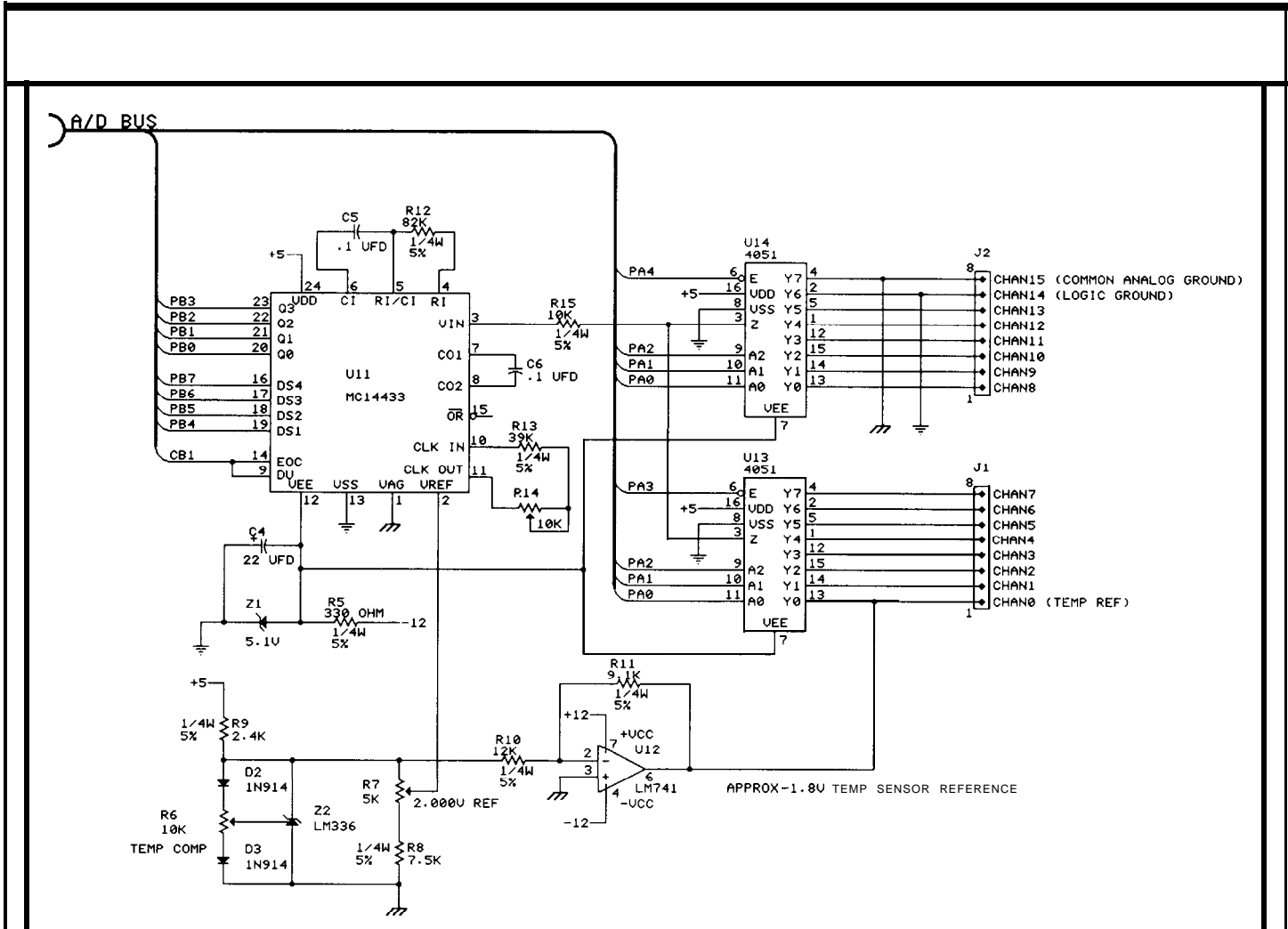


Figure 3 -- The Input Block -- The $-1.8V$ temperature sensor reference allows the MC14433 to deal with voltages above the 2.000V A/D reference.

3rd, 4th) on each of its DS lines, while the binary value of that digit is presented on the Q lines. Each digit but the first can have values from 0 to 9. The first digit can either be 0 or 1 (which is why this chip is referred to as a 3-1/2 digit device), and two of the other bits are used to indicate negative polarity and over-range. It is the responsibility of the program to separate these digit signals into a meaningful 4-digit value between -1.999 and $+1.999$ volts. The A/D chip actually provides a reading that is a ratio of the input voltage to its reference voltage. By setting the reference voltage to the highest reading (1999 counts), the chip directly outputs the input voltage as

1 count per millivolt. The LM336 voltage reference (Z2) provides a very stable 2.49-volt source. This is made available through an adjustment that sets the A/D chip's reference input to 2.000 volts DC.

There are other voltage reference components available that provide the required 2.000 volts directly, but we used those we had on hand. The basic full-scale accuracy of the A/D box depends on how accurately this 2.000-volt reference is set. If it is set to 1.000 volt, for example, and you apply 0.500 volts to its input, the device will read 1/2 scale (1000 counts), or 1.000 volt. This could be adjusted by the software, of course, but one of the basic requirements of the project was direct reading of R7, however, is very critical as it

provides this when its reference is set to 2.000 volts. The LM741 operational amplifier (U12), which I'll describe in more detail a little later, provides a source of approximately -1.8 volts as an offset for the LM335 temperature sensors.

There are three adjustments that can be made on the A/D box. The basic clock frequency of the MC14433 chip can be set to 400 kHz with R14 -- this is not critical. The adjustment on the LM336 2.49-volt regulator is really a fine-tuning adjustment of R6 and provides the best temperature regulation when set to 2.49 volts. The 2.000-volt reference adjustment, R7, however, is very critical as it

affects the accuracy of the entire A/D chip. This must be set with a very accurate digital voltmeter (DVM). Since, the A/D box will only be as accurate as the voltmeter that it's calibrated with.

Power Supply Block

You'll need to supply the box with $+5V @ 1/2$ amp, and $\pm 12V @ 50-100$ mA. For each of these, a simple three-terminal voltage regulator provides ample power, and is probably the easiest design solution. The MC14433 requires a $-5V$ supply, and I used a single zener diode as the regulator.

Taking your Temperature

One of the major functions of the A/D box was to report temperature in degrees. LM335 temperature sensors are simple, inexpensive, two-lead devices that provide a voltage equal to 10 millivolts per Kelvin. A third lead is present that can be used for accurate temperature trimming. The Kelvin temperature scale starts at absolute zero (-459 degrees Fahrenheit, -273 degrees Celsius). Since simple relationships exist to convert between all three temperature scales, the A/D software only has to perform some simple arithmetic operations to get into the desired scale. Unfortunately, room temperature (68 degrees Fahrenheit, 20 degrees Celsius) is equal to 293 Kelvin. At 10 millivolts per degree, this device will output 2.93 volts at room temperature. As the useful range of the A/D box is ± 2 volts, we have a slight problem. Remember the LM741 that provided -1.8 volts? It is used as the common reference for all of the LM335 temperature sensors, effectively offsetting all readings by 1.8 volts. So, our 2.93 volts, referenced to ground, is now down to 1.13 volts when referenced to the

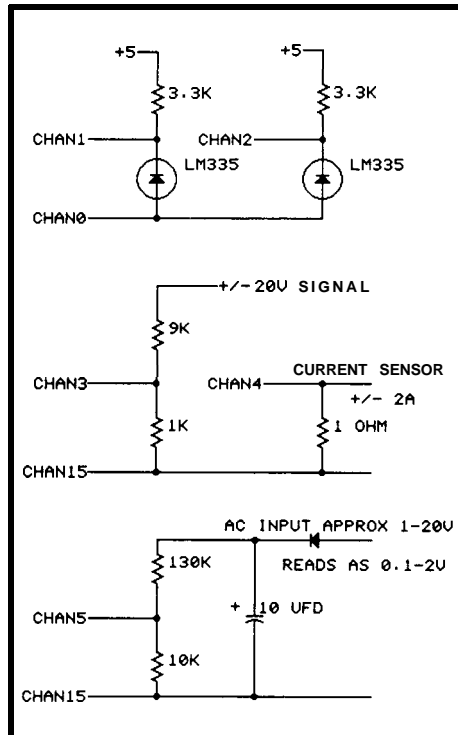


Figure 4 -- A variety of sensors may be attached to the box for measuring various types of data. At the top is a circuit for temperature measurement. The other two circuits are for current measurement.

-1.8 -volt supply. Voilà! A very respectable and in-range reading, but who knows if that -1.8 volt reference is accurate? Well, the A/D box is wired to read that voltage on channel 0. If any channel is configured for temperature conversion, channel 0 is read first to get the offset voltage that is subtracted from the actual channel with the temperature sensor. A conversion only takes a few milliseconds, and neither the reference nor temperature is likely to change between readings, so the A/D box automatically self-adjusts for temperature readings. By the way, the -1.8 -volt reference value is not critical as long as it is within the acceptable range for the A/D chip and brings the temperature sensor voltages down to this same range. I'll be saying more on this topic in Part 2 of this article.

Signal Connections

While there are 16 channels available, channel 0 is hard wired as the temperature sensor reference, channel 14 is connected to a logic ground point, and channel 15 is connected to the analog ground, which is the ground reference for all other input signals. When you build this box, it is very important to keep the two grounds separate and interconnect them at one point only within the A/D box, since they are not at exactly the same potential.

The channel-1 through channel-13 inputs may be used to measure any desired signal that is within the ± 2 -volt range of the device. If higher voltages must be read, then simple 10:1 divider networks can be built to attenuate the input signal. Channel 0 is the temperature sensor reference signal of approximately -1.8 volts. All sensors should have their negative leads connected to this point. Their positive leads should connect to any available channel and must also be connected to a positive voltage source through a pull-up resistor that will provide about 1 milliamperes of current. Figure 4 shows some typical connections for measuring voltages and temperatures.

Well, that about does it for the hardware. In Part 2, I'll describe the C language software that controls the device and performs the necessary conversions on the input.

Circuit Cellar Books

Circuit Cellar INK authors often refer to previous Circuit Cellar articles. These past articles are available in book form from Circuit Cellar Inc., 4 Park St., Suite 12, Vernon, CT 06066.

Ciarcia's Circuit Cellar Volume I covers articles in BYTE from September 1977 through November 1978. Volume II covers, December 1978 through June 1980. Volume III covers July 1980 through December 1981. Volume IV covers January 1982 through June 1983. Volume V covers July 1983 through December 1984. Volume VI covers January 1985 through June 1986.

FIRMWARE FURNACE

DDT-51 Revealed

by Ed Nisley



If your next project should use an 8031 but you can't quite justify the cost of a full-blown development system, the DDT-51 system covered in the August and September 1988 BYTE Circuit Cellars should push you over the edge. Now you can start writing Real Firmware!

For those of you who haven't heard yet, DDT-51 allows you to run and debug 8031 programs using an IBM PC as a host. By simply replacing the EPROM in the target system (the 8031 gadget you're developing) with a RAM, DDT-51 can download a program from disk in seconds; there's no need to burn an EPROM to find your latest bug. Even better, the DDT-51 hardware and software can set 8031 breakpoints, single step through your code, and display the contents of (nearly) all the 8031's internal registers and ports.

Files describing DDT-51 construction tips and schematic corrections are available on the Circuit Cellar BBS. You should consult these files before building the hardware, because there are some tricks that you need to know before melting the solder. Pay particular attention to the suggestions concerning bypassing and cable layout, because intermittent bugs are exceedingly hard to track down.

[Editor's Note: See page 35 for information on logging on to the

Circuit Cellar BBS. The schematic corrections also appear in issue #5 of Circuit Cellar INK. A reprint of the original Circuit Cellar DDT-51 article is available from Circuit Cellar INK. Send \$3.00 to DDT-51 Reprint, Circuit Cellar INK, 4 Park St., Vernon, CT 06066.1

Hardware Basics

DDT-51 requires some hardware and software resources in the target system. Fortunately, the list isn't long: one interrupt input pin, one output pin, a few bytes of program code, some stack space, and room in External Data Memory for 2K bytes of Debug RAM. The Debug RAM is provided on the DDT-51, so your system won't need any special hardware.

I'll restrict this column to the basic configuration provided on the BBS, which uses INT1 for the interrupt input, T1 for the output bit, and decodes the Debug RAM at 8000H for 2000H (8K) bytes. You can change most of these if they aren't appropriate.

Because DDT-51 is not a true ICE system it cannot help you find hardware wiring errors. Your hardware must be able to load and run 8031 code before you can debug your firmware! By the simple fact of not working correctly DDT-51 can often reveal hardware problems, but this isn't quite what we had in mind. A better starting point is the PC code in

TESTER3 1 .ARC, which exercises the target system hardware without using the 8031.

[Editor's Note: All software mentioned in this article is available on the Software Disk for this issue, and on the Circuit Cellar BBS. See page 35 for ordering and downloading information.]

TESTER.COM will exercise (nearly) all of the hardware in both the DDT-51 and target systems needed to run the debugger code. It will report on any problems it finds, but you're responsible for tracking down the precise causes. Your system must be able to run TESTER correctly before you can start debugging your own code (and other hardware) using DEBUG3 1.

Software Basics

The DDT-51 code comes in two chunks: DEBUG31 for the PC and KERNEL for the 8031. DEBUG3 1 is written in Turbo Pascal (Version 3.0) and KERNEL is written in 8031 assembler (in Avocet Systems' AVMAC5 1 syntax).

The programs exchange data and status information through the 2K-byte Debug RAM. That RAM is not dual ported, so either the PC or the 8031, but never both, can have access to it at any one time. INT1 and T1 synchronize the handshaking for this agreement.

All exchanges between the PC

and the 8031 follow this general pattern: DEBUG31 pulls INT1 down to interrupt the 8031, KERNEL accepts the interrupt and raises T1 when it is finished using the Debug RAM, then DEBUG31 examines and changes the Debug RAM and raises INT1. DEBUG31 takes control of the Debug RAM only after T1 goes high and releases it before raising INT1.

During the time that T1 is active, KERNEL is spinning in a tight loop at an instruction located in the Program RAM (near address 0000H) testing INT1 for a high level. Your 8031 program is inactive at this point because INT1 is the highest-priority interrupt and all other interrupts are locked out. You can change INT1 to a low-priority interrupt to allow "background" interrupt processing during single stepping, but if your interrupt routine should lock up, DDT-51 can't get control.

Because the two routines proceed in lock step during the hand-

shaking there are no time limits on how long the 8031 may take to respond to an interrupt, nor how long the PC may wait to release INT1. The latter is fortunate because while both INT1 and T1 are active, DDT-51 is displaying the contents of the 8031 registers on the PC screen and waiting for your input. Imagine all that computer power going to waste!

Minimal Targeting

Figure 1 shows the simplest useful 8031 target system: it has 8K bytes of program storage and one output bit connected to an LED. Admittedly this is an expensive way to get a blinking LED, but that LED will indicate that something is happening inside the 8031.

Notice the jumper connection at the 8031 RESET input (pin 9). Because DDT-51 must control RESET through the DIP clip, pin 9 cannot be connected to the normal reset circuit. Remember to install the jumper shown in Figure 1 when the

8031 system is running without the DDT-51 clip because very strange things happen if the RESET pin is left floating or tied low when the power comes on.

The EPROM that normally holds the 8031 program code must be replaced with a RAM so DDT-51 can load programs directly. Although most of the pins are identical on 2764 EPROMs and 6264 static RAMs, there are a few significant differences. The connections shown in Figure 1 account for these differences, but a more complex system may need additional address decoding or jumpers to select EPROM or RAM pinouts.

The target system must connect the 8031 -WR signal to RAM -WE (pin 27) to allow writes into the Program RAM. You can tie pin 27 high when using an EPROM or leave it connected to -WR, because the signal is normally high. As long as pin 1 isn't connected to the programming voltage your system can't normally write the EPROM.

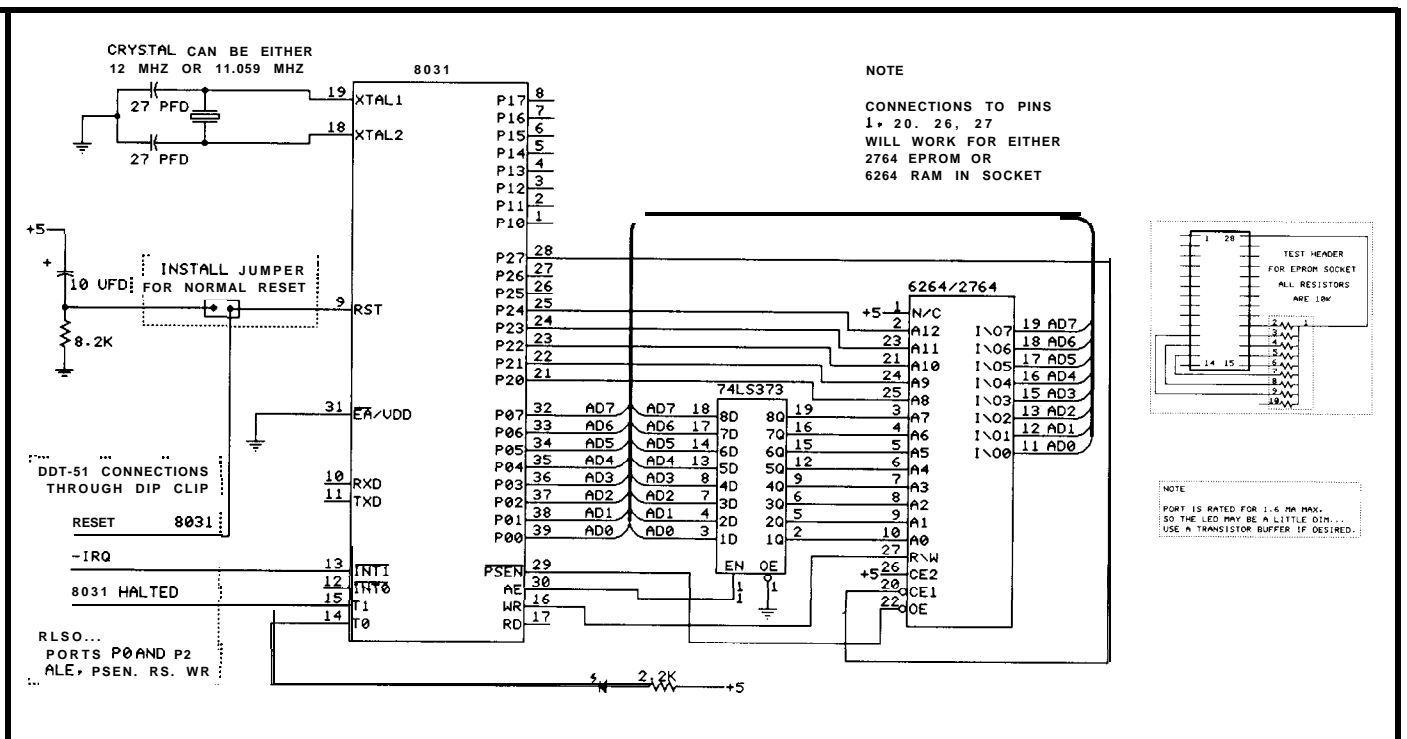


Figure 1 -- This is a very expensive way to get a blinking LED, but it will provide a working circuit for you to practice debugging.

An address decoder on the DDT-51 board locates the Debug RAM at 8000H, so the only decoding needed in the target system is A15 at the RAM's -CE1 input (pin 20). This ensures that the RAM is active only for addresses below 7FFFH, and inactive when the Debug RAM is accessed at 8000H and above. The CE2 input (pin 26) is not connected on EPROMs, but must be tied to +5V to enable RAMs.

The INT1 input and T1 output are connected to the DDT-5 1 hardware directly through the DIP clip on the 803 1 itself. Because the 803 1 provides an internal pull-up on INT1, there is no need for an external resistor in normal operation.

The LED is connected to +5V through a 2.7k resistor because an 803 1 port pin can sink only 1.6 mA. A practical project would use a transistor or gate to increase the drive current, but for our purposes a dim glow is OK.

Getting Started

Contrary to what you might think, the first thing you do with a new 803 1 system isn't plug in all the chips, clamp on the DIP clip, fire it up, and start debugging! Remember that DDT-51 can't diagnose hardware problems because it depends on the 8031 to execute KERNEL: if the 8031 doesn't work, neither will DDT-5 1. So the first few steps are somewhat simpler and require less functional hardware.

After you make sure that the power and ground wiring is correct, plug in all the chips and the crystal (if it's socketed). Remove the RESET jumper to give DDT-5 1 control over the RESET signal and remember to use a RAM chip instead of an EPROM. Connect the DDT-51 clip, turn on the power, and start TESTER. You can skip through the tests until you get to the

ones that exercise the target system hardware. Run these and use your oscilloscope to make sure that the address latch is working and that the RAM is getting good pulses.

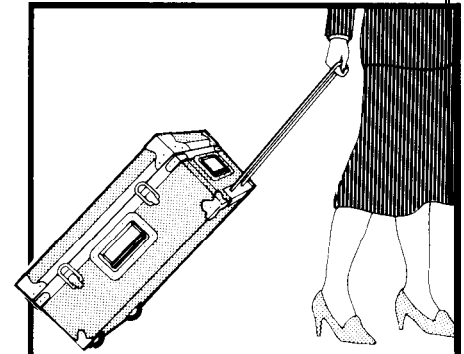
TESTER's last exercise writes data into both the Debug and Program RAMs, and verifies that both contain the right data. When your system passes that test, it's ready for the big time.

Disconnect the DIP clip and install the RESET jumper to provide a proper reset pulse after the power goes on. Build a simple EPROM socket header to pull up the data bus lines with 10k resistors, as shown in Figure 1. Just floating the bus lines won't work because the 8031 forces the address out on the same lines it's reading a fraction of a microsecond later.

When you turn on the power the 8031 will reset and start executing instructions starting at address 0000H. The pullups on the header ensure that it always "reads" FFH, which is the opcode for the MOV R7,A instruction. Because this instruction executes in one machine cycle and uses only internal registers, the external bus actions are easy to figure out with an oscilloscope.

Look for steady 1 -MHz pulses on ALE and -PSEN. Both ALE and -PSEN should go high at about the same time, which marks the start of an instruction fetch. ALE will go low (inactive) first, when the low byte of the address is stable on the data lines, so the LS373 can latch that byte. -PSEN will then go low (active), which normally turns on the EPROM output buffers. In this case, though, the resistors in the EPROM socket header drag the bus lines high to simulate FFH data.

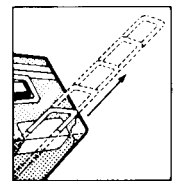
Because the 8031 is executing a million MOV R7,A instructions every second (with a 12-MHz crystal), pin 2 of the LS373 should be toggling every microsecond. Check for a steady binary count at the LS373 outputs; pin 19 should be toggling



TAKE A LOAD OFF...

A rugged CABBAGE CASE? lined with plenty of foam for your equipment can TAKE A **LOAD** OFF YOUR MIND when you've got to travel.

TAKE A **LOAD OFF YOUR BACK** with our exclusive tilt-wheels and extension handle option.



UNLOAD ON US!

Call or write to tell us about your shipping or carrying problems
WE HAVE SOLUTIONS!

- Custom cases □
- Made to order ■
- One at a time ■

CABBAGE CASES

Call or write:

CABBAGE CASES, INC.
1166-C Steelwood Road
Columbus, Ohio 43212-1356
614/486-2495 800/686-2495

every 128 microseconds. Port P2 (8031 pins 28 down through 21) should also be counting up, with pin 28 toggling every 256 microseconds and pin 21 toggling every 32,768 microseconds (about 30 times per second).

Once you're sure the hardware is functional, DDT-51 can start helping you. Reinstall the RAM, remove the reset jumper, reconnect the DIP clip, and fire everything up again. Download the KERNEL code as described in the next section and see if it plays.

If everything works the first time, pat yourself on the back. If not, fall back to TESTER and your oscilloscope to trace the signals through the logic. Make sure that the RAM is getting write pulses and stable addresses, that the chip selects are correct, and so forth. Now you can see why TESTER has loops for the tests: type in a count of 32000 for a test and you can probe around for quite a while with dependable signals on all the pins.

Debugging at Last!

Once the hardware is OK, you can start working on the hard part ... debugging the firmware!

KERNEL includes a short loop where your program would normally start. The loop, shown in Listing 1, increments DPTR, decrements B, and adds 12H to the accumulator. It also complements the output bit connected to the LED so you can actually see when the 8031 executes that instruction. By downloading the unchanged version of KERNEL you can verify that the DDT-51 hardware and software work without worrying about your new code.

Make sure that KERNEL.HEX is in the current subdirectory, then fire up DEBUG31 and tap F3 to download the kernel. DEBUG31 verifies each RAM byte as it's written, so if there is no error

Listing 1

```

; KERNEL Test Loop
BLINKBIT EQU P3.4 ; visible output from dummy loop
;-----
; "Your program starts here"
; First single step is after the next instruction
MOV DPTR,#OFACEH ; identifiable starting value
MOV A,#OBAH ; ... ditto
MOV B,#OBEH ; ... ditto
redoit INC DPTR ; easy to watch
ADD A,#12H ; a cinch to figure
DEC B ; simple to understand
CPL BLINKBIT ; and visible to the naked eye
SJMP redoit ; repeat until forever

; "Your code ends here"
;-----

```

KERNEL is a simple looping program that works with the circuit detailed in Figure 1. It provides a known-out base from which to begin testing your own firmware.

message you can be reasonably sure that the program was successfully loaded. If the download didn't work it's a sure bet that the program will not execute correctly ...so start checking your wiring again.

DEBUG31 holds RESET active after downloading KERNEL, so no code is executed until you press either F9 for normal running or F10 for a single step. In either case, KERNEL will perform a few setup operations immediately after coming out of the hardware reset. Because this setup is part of the "hidden" KERNEL functions, single-step execution will actually take effect

just after the first instruction in the test loop. You cannot single-step through KERNEL's DDT-5 1 code.

Every time you type F10 KERNEL will execute one instruction, copy the 8031's Internal RAM into Debug RAM, and wait for the PC. DEBUG31 will read the Debug RAM and dump the results onto the screen. Figure 2 shows the screen after a single press of F10: notice that the disassembled instruction is MOV A,#BA. This corresponds to the second instruction in Listing 1. DPTR is already loaded with FACEH, showing that the first instruction was executed correctly.

```

Halted at single step
Reading Debug RAM, interrupt count is: 1
A=2C , B=00 DPTR=FACE SP=07 PSW=01 (CY AC FO OV P)
PC=0055 -> MOV A.#BA
Reg Bank 0: EE ED ED ED ED ED ED ED Reg Bank 1: 55 00 01 2C EE 00 CE FA
Reg Bank 2: 3F 84 00 00 ED ED ED ED Reg Bank 3: ED ED ED ED ED ED ED ED
Bit addressable RAM:
20 EDEDEDEDEDEDEDEDEDEDEDEDEDEDE ED ED ED ED
General RAM:
30 EDEDEDEDEDEDEDEDEDEDEDEDEDEDE ED ED ED ED
40 EDEDEDEDEDEDEDEDEDEDEDEDEDEDE ED ED ED ED
50 ED ED ED ED ED ED ED ED ED ED ED ED ED ED ED ED
60 ED ED ED ED ED ED ED ED ED ED ED ED ED ED ED
70 ED ED ED ED ED ED ED ED ED ED ED ED ED ED ED
Use F9 to run, F10 to step

```

Figure 2 -- This sample screen from DEBUG31 shows the results immediately after executing the first instruction in Listing 1.


```

Halted at single step

Reading Debug RAM, interrupt count is: 2
A=BA || B=00 DPTR=FACE SP=07 PSW=01 (CY AC FO OV P)
PC=0057 -> MOV B,#BE
Reg Bank 0: EE ED ED ED ED ED ED ED Reg Bank 1: 57 00 01 BA EE 00 CE FA
Reg Bank 2: 3F 84 00 00 ED ED ED ED Reg Bank 3: ED ED ED ED ED ED ED ED

-----

Reading Debug RAM, interrupt count is: 3
A=BA || B=BE 4 DPTR=FACE SP=07 PSW=01 (CY AC FO OV P)
PC=005A -> INC DPTR
Reg Bank 0: E1? ED ED ED ED ED ED ED Reg Bank 1: 5A 00 01 BA EE BE CE FA
Reg Bank 2: 3F 84 00 00 ED ED ED ED Reg Bank 3: ED ED ED ED ED ED ED ED

-----

Reading Debug RAM, interrupt count is: 4
A=BA || B=BE ↓ DPTR=FACF SP=07 PSW=01 (CY AC FO OV P)
PC=005B -> ADD A,#12
Reg Bank 0: EE ED ED ED ED ED ED ED Reg Bank 1: 5B 00 01 BA EE BE CF FA
Reg Bank 2: 3F 84 00 00 ED ED ED ED Reg Bank 3: ED ED ED ED ED ED ED ED

-----

Reading Debug RAM, interrupt count is: 5
A=CC || B=BE 4 DPTR=FACF SP=07 PSW=00 (CY AC FO OV P)
PC=005D -> DEC B
Reg Bank 0: EE ED ED ED ED ED ED ED Reg Bank 1: 5D 00 00 CC EE BE CF FA
Reg Bank 2: 3F 84 00 00 ED ED ED ED Reg Bank 3: ED ED ED ED ED ED ED ED

-----

Reading Debug RAM, interrupt count is: 6
A=CC || B=BD ↓ DPTR=FACF SP=07 PSW=00 (CY AC FO OV P)
PC=005F -> CPL P3.4
Reg Bank 0: EE ED ED ED ED ED ED ED Reg Bank 1: 5F 00 00 CC EE BD CF FA
Reg Bank 2: 3F 84 00 00 ED ED ED ED Reg Bank 3: ED ED ED ED ED ED ED ED

-----

Reading Debug RAM, interrupt count is: 7
A=CC || B=BD ↓ DPTR=FACF SP=07 PSW=00 (CY AC FO OV P)
PC=0061 -> SJMP F7
Reg Bank 0: EE ED ED ED ED ED ED ED Reg Bank 1: 61 00 00 CC EE BD CF FA
Reg Bank 2: 3F 84 00 00 ED ED ED ED Reg Bank 3: ED ED ED ED ED ED ED ED

-----

Reading Debug RAM, interrupt count is: 8
A=CC || B=BD ↓ DPTR=FACF SP=07 PSW=00 (CY AC FO OV P)
PC=005A -> INC DPTR
Reg Bank 0: EE ED ED ED ED ED ED ED Reg Bank 1: 5A 00 00 CC EE BD CF FA
Reg Bank 2: 3F 84 00 00 ED ED ED ED Reg Bank 3: ED ED ED ED ED ED ED ED

```

Figure 3 -- After executing a few more statements from Listing 1, changes in PSW bits and active register contents become obvious.

Part of the KERNEL setup code fills internal RAM with EDH, a byte that is easy for me to remember. The first byte (Reg 0, Bank 0) is set to EEH and the last byte (at address 7FH) is set to DDH to indicate that the fill operation terminated correctly. Internal RAM is not reset to a specified value after a RESET (unlike the Special Function Registers), so a Program that depends on a particular value will go astray. Filling the RAM with EDH won't affect a program that

assumes nothing about the initial RAM state, but will highlight random stores and fetches.

Figure 3 shows the results of the next few single steps. To conserve space I've removed the unchanging parts of the display, but you'll see the full screen on your PC. The "Interrupt Count" value should increase smoothly because both DEBUG3 1 and KERNEL track the number of INT1 requests and compare notes; if the values differ you will also see an error message. Your screen will show


highlighted characters for the PSW bits and the active register bank; these highlights change to match the PSW.

The KERNEL test loop uses the default stack location starting at OFH so the stack occupies much of Register Banks 1 and 2. KERNEL adjusts the stack pointer to display its value during normal execution, so it remains at 07H during the loop. Your program can move the stack anywhere in internal RAM. Remember to allow about 16 bytes of stack space for KERNEL's use because stack crashes are just as severe in 8031s as they are in PCs.

Pressing F8 allows you to insert a breakpoint at a specified address in the 8031 code. Because the 8031 doesn't include a single-byte breakpoint instruction, KERNEL must insert a three-byte LJMP at the breakpoint address. DEBUG31 checks to make sure that the LJMP won't overwrite the instruction at IP to prevent destruction of the current instruction. The breakpoint is removed when it is executed or when F10 interrupts execution.

On Your Own

The next step, of course, is to replace the test loop in KERNEL with your own code. The key addresses are noted in KERNEL.ASM, and you must ensure that the reset and INT1 vectors and the Debug RAM code all wind up at the right locations. You should name your modified code something other than KERNEL.ASM so you can download the original version for fast hardware checkouts.

So now, with a few chips, a little solder, and some downloaded software, there's no reason why you can't use a single-chip micro in your next project! 



by Steve Garcia

Ctrl

STEVE'S OWN I N K

A Gathering of Eagles

By now most of you know that after the December issue I will no longer be in BYTE. Of course, if you are reading this, you have already solved that problem. I value the last 11 years for its fond memories but most of all for the experience. While it could be said that anything "computer" related could be a success in the late seventies, BYTE did it in style.

In the early days I saw a dedicated editorial and management staff join together in the belief that they had a common goal, a focused direction, and a beneficial purpose in what they were producing. I saw it develop into a technical bible with a cult following and then go through a series of management changes which brought it out of its niche market to be a hopeful contender with a broader focus.

Whatever its direction, I wish BYTE well. I certainly have learned a lot. What I discovered most was the value of team effort and the potential for success an experienced group can bring to a publication. While the late seventies was indeed a boom time, that core team of BYTE had a lot to do with its success.

Circuit Cellar INK is a young publication, barely a year old. But the maturity of the print should not be construed as indicative of the experience of the staff or its potential for success in a competitive marketplace. Readers who grow to depend upon Circuit Cellar INK as a resource have little to worry about.

Circuit Cellar INK has enjoyed a success that seems to be paralleled only by BYTE itself, and that is exactly what we expected. If you take a close look and compare the mastheads of Circuit Cellar INK and an early BYTE (circa '80) you see that Circuit Cellar INK is run by the same key people in the early development of BYTE; some experienced individuals indeed. They include the original copublisher, circulation manager, financial controller, lead columnist, cover artist, and advertising representatives. Only Curt is new on the block, but you already know he's evolved from the same spirit of technical achievement.

With only six issues of Circuit Cellar INK out the door, it is hard for me not to feel awe combined with satisfaction. Achievements such as Circuit Cellar INK's appearance on national newsstands at only its second issue and entering its seventh issue with a circulation of 25,000 means that we must have the combination right. The last time any of us experienced that was at BYTE. But, then again, maybe we had something to do with that too.

Next year Circuit Cellar INK will show more of the outward signs of this growth. Its size will increase and 4-color presentations will be the norm. Rest assured that we will be wary to manage our growth intelligently and keep our editorial focused properly. Fortunately we have both experience and time on our side. We are neither an undercapitalized small publication with publish or perish goals nor a large-company big-budget extravaganza with inflexible P&L targets. Instead, think of us as a gathering of eagles descending from a cloud of vultures whose sole purpose is to once again publish the best computer applications journal.

Steve Garcia